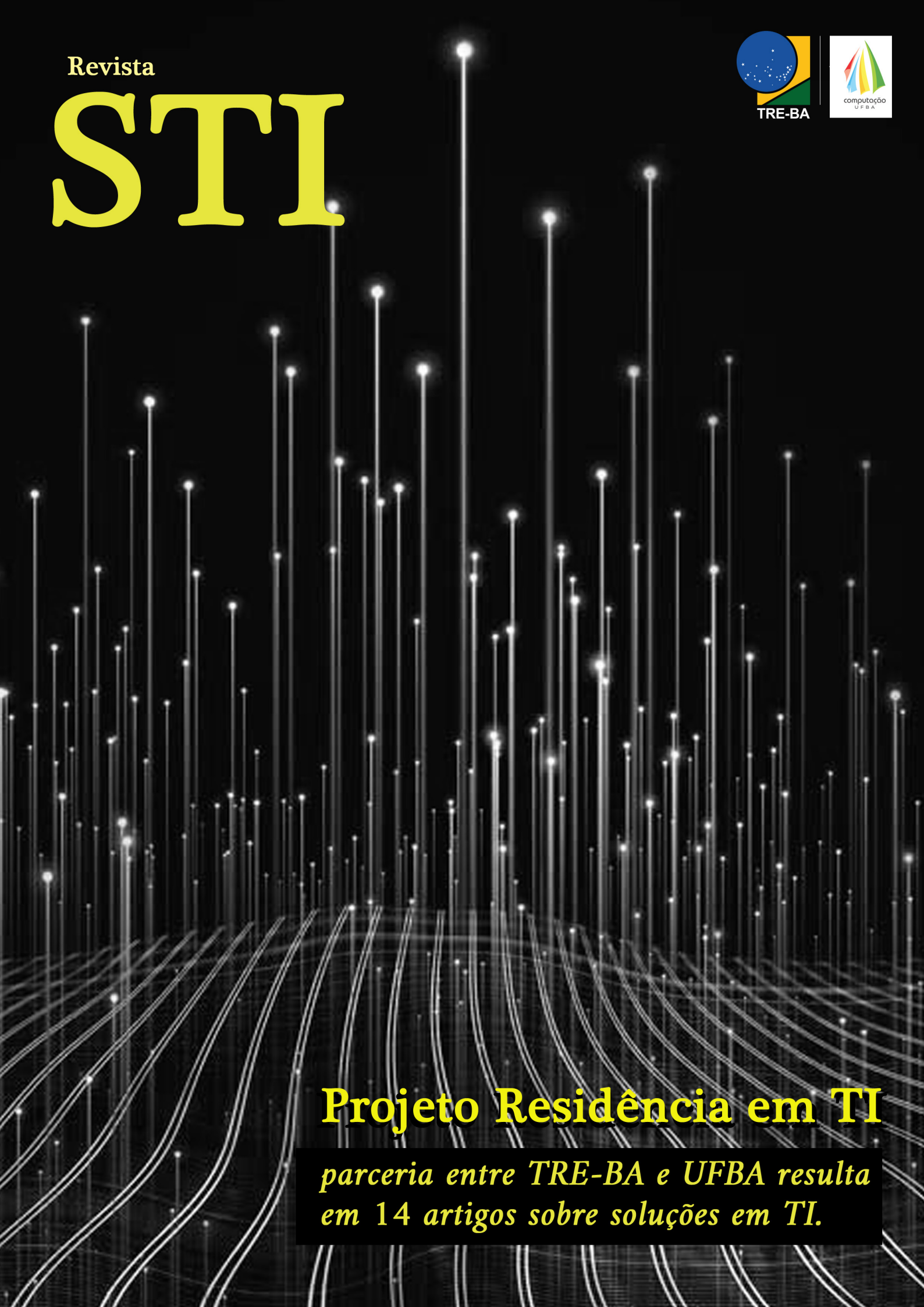
Revista

STI

TRE-BA

computação UFBA



Projeto Residência em TI

parceria entre TRE-BA e UFBA resulta em 14 artigos sobre soluções em TI.

Sumário

Prefácio

Prezado leitor, prezada leitora,

Seja bem-vindo(a) à Revista STI, cujo tema é o Projeto Residência em TI. Esta edição reúne os 14 artigos escritos pelos participantes desse projeto, e sintetiza os resultados alcançados por esta iniciativa. Os textos aqui apresentados detalham os projetos de software desenvolvidos na Residência em TI e discutem os desafios da aplicação prática de alguns conceitos teóricos adquiridos durante o projeto. Esses conceitos estão no estado-da-arte da área de Ciência da Computação, e têm sido amplamente utilizados por importantes organizações no Brasil e no mundo.

Iniciado no segundo semestre de 2020, o projeto Residência em TI foi fruto de um convênio celebrado entre o TRE-BA e o Instituto de Computação da Universidade Federal da Bahia (IC/UFBA) para capacitar servidores da área de Tecnologia da Informação do Tribunal em tecnologias emergentes, bem como promover a integração de profissionais recém-graduados em cursos da área de Tecnologia da Informação, os *residentes*, às atividades do Tribunal.

O modelo do Projeto Residência em TI foi inspirado no programa de residência médica, em que os estudantes, recém-graduados, têm a oportunidade de uma formação teórica na Universidade e desempenham as atividades práticas (como forma de consolidar os conceitos aprendidos) no hospital. Analogamente, neste projeto, todos os estudantes tiveram a formação teórica com professores do curso de Pós-Graduação Lato Sensu (Especialização) em Tecnologia da Informação, ofertado pelo IC/UFBA e desempenharam atividades práticas em ambiente real, no TRE-BA, sob a supervisão de servidores do quadro técnico deste Tribunal e orientação acadêmica de professores do IC/UFBA.

Nesta primeira turma do Projeto Residência em TI, oito residentes atuaram em equipes responsáveis por desenvolver quatro soluções tecnológicas: o portal de *business intelligence*, o projeto de construção da API (*Application Programming Interface*) do TRE-BA, o assistente virtual "Otto" - chatbot, e o projeto de reconhecimento de imagens. Cada dupla de residentes compôs um time, que ficou responsável por desenvolver uma dessas soluções. O desenvolvimento contou com o forte apoio de servidores de diversos setores do Tribunal, que não mediram esforços para auxilia-los no entendimento dos problemas, e na validação frequente das entregas.

O Projeto Residência em TI foi realizado em meio às incertezas trazidas pela pandemia do SARS-COV-2. A pandemia, que afetou a nossa vida sob diversas perspectivas, trouxe um enorme desafio no que diz respeito às formas de trabalho. Outrora majoritariamente presencial, foi necessária uma rápida adaptação para que o modelo virtual de trabalho pudesse ocupar a totalidade das ações. Perdurando por um longo período, afetou sobremaneira as atividades do projeto Residência em TI. Entretanto, é digno de nota a capacidade

dos(as) envolvidos(as) em agir rapidamente no sentido de readaptar-se, e prosseguir com as atividades propostas, com pouco prejuízo à sua realização.

É importante ressaltar que os resultados do Projeto Residência em TI, sintetizados nesta edição da Revista STI, são frutos de um enorme esforço coletivo, desde a concepção do projeto até a sua entrega. Esse esforço envolveu a participação de diversos servidores do TRE-BA, a quem tive o enorme prazer de atuar conjuntamente neste projeto. Quero aqui agradecer a esses servidores, que trabalham incansavelmente por uma Justiça Eleitoral cada vez mais acessível, transparente, ágil e eficaz. Em nome do nosso Instituto de Computação e da Universidade Federal da Bahia, os nossos mais sinceros agradecimentos.

Salvador-BA, Julho de 2022

Prof. Dr. Ivan do Carmo Machado
Professor Adjunto do IC/UFBA
Coordenador do Projeto Residência em TI

Anne Cristie Pinheiro Moreira Gaudencio
Departamento de Ciência da Computação – Universidade Federal da Bahia (UFBA)
Salvador – BA – Brazil
anne.cristie@ufba.br

O uso aplicado de Natural Language Toolkit (NLTK) no desenvolvimento de um chatbot humanizado

Abstract. *This article discusses how the use of the Natural Language Toolkit (NLTK) can contribute to the development of a humanized chatbot. Using Natural Language Processing principles and techniques, a chatbot was developedthat provides online information and services offered by the Tribunal RegionalEleitoral da Bahia, with three customer service options: clickable menus, chat typing and voice commands.*

Resumo. *O presente artigo aborda como o uso do Natural Language Toolkit (NLTK) pode contribuir para o desenvolvimento de um chatbot humanizado. Utilizando princí*pios *e técnicas de Processamento de Linguagem Natural, foi desenvolvido um chatbot que disponibiliza informações e serviços online ofertados pelo Tribunal Regional Eleitoral da Bahia, com três opções de atendimento ao cliente: menus clicáveis, digitação no chat e comandos de voz.*

1. Introdução

O Processamento de Linguagem Natural (PLN) é uma subárea da Inteligência Artificial [OTHERO 2002] que está cada vez mais presente na era digital: seja nas ferramentas de buscas, nos aplicativos de tradução, na análise de sentimento, nos assistentes virtuais, dentre outros. Este artigo terá como enfoque o *Natural Language Toolkit* (NLTK), ferramenta de PLN do *Python* e sua aplicação no *chatbot* humanizado em desenvolvimento para o Tribunal Regional Eleitoral da Bahia (TRE-BA).

O TRE-BA oferece diversos serviços aos cidadãos (como Alistamento Eleitoral, Emissão de Segunda Via de Título de Eleitor, Emissão de Certificado de Quitação Eleitoral, Consulta de Débitos Eleitorais, dentre outros), sendo tanto a oferta quanto a demanda por tais serviços majoritariamente de forma presencial. Porém, com os avanços tecnológicos dos últimos anos e principalmente durante a pandemia da Covid-19 em que manter o distanciamento social é estritamente necessário, passou a crescer rapidamente o número de serviços ofertados *online*.

Um desafio que surge, então, é tornar o acesso a esses diversos serviços, espalhados em vários sistemas e aplicativos, fáceis de serem encontrados e utilizados pelos eleitores. Para ter uma disponibilidade ampla e evitar a sobrecarga dos servidores que trabalham com o atendimento ao eleitor, nasce a necessidade da automatização das informações do TRE-BA no que se refere aos serviços mais requeridos pelo eleitor/usuário por meio de um *chatbot* humanizado, com sintetizador de voz.

Tendo em vista que um *chatbot* tem por base a comunicação e interação entre um humano (que utiliza a linguagem natural) e um robô (que interpreta linguagem de máquina) de forma fluida

e eficaz, torna-se essencial a utilização de Processamento de Linguagem Natural (PLN). Dentre as diversas ferramentas de PLN disponíveis, a *NaturalLanguage Toolkit* (NLTK) foi a escolhida para auxiliar na implementação do *chatbot*. O motivo da escolha do NLTK e suas principais vantagens serão considerados na próxima Subseção.

1.1. Motivação

O NLTK é uma plataforma que visa a construção de programas *Python* que trabalhem com dados de linguagem humana [BIRD et al. 2021]. De acordo com a sua documentação, o NLTK foi considerado como "uma ferramenta maravilhosa para ensinar e trabalhar em linguística computacional usando *Python*"e "uma biblioteca incrível para brincar com a linguagem natural"[BIRD et al. 2021].

A primeira grande vantagem da NLTK que motivou sua utilização no projeto é o fato de ela também ser desenvolvida em *Python*, assim como todo o sistema do *chatbot* . O *Python* além de ser uma linguagem de alto nível, tem uma excelente curva de aprendizado e é atualmente considerada como a linguagem de programação referência na área de Inteligência Artifical, e por tais motivos foi a linguagem de programação escolhida para o desenvolvimento do *chatbot*, que por sua vez influenciou na escolha da ferramenta de PLN.

Uma segunda motivação para a escolha do NLTK foi por sua praticidade e facilidade de utilização: seja na separação de sentenças, na tokenização ou no seu corpora. O NLTK possui mais de cem *corpus* de dados que facilitam a aplicação do PLN, como o *Stopwords Corpus*, que é composto por um *corpus* com os principais *Stopwords* do idioma selecionado, o *Unicode Samples*, que retira a acentuação das palavras, dentre outros.

1.2. Objetivo

O objetivo principal deste artigo é Utilizar Processamento de Linguagem Natural, por meio do *Natural Language Toolkit* para auxiliar no desenvolvimento do *chatbot*. Os objetivos específicos são: Tokenizar as entradas do usuário, Utilizar o *Stopword Corpus* para retirar os *stopwords* da mensagem recebida, Realizar a Etiquetagem Morfossintática de cada *token* e Treinar classificadores para categorizar cada entrada por possível serviço requerido quando o *chatbot* não encontrar nenhuma compatibilidade na consulta ao banco de dados.

2. Processamento de Linguagem Natural

A linguagem natural, seja ela falada ou escrita, é um aspecto importante na vida coletiva e na interação dos humanos entre si. Atualmente, porém, com os avanços da tecnologia e a cada vez mais crescente dependência dela no cotidiano, torna-se imprescindível interagir também com as máquinas e ser entendidos por elas. Considerando que as máquinas entendem linguagem de programação e não a linguagem natural, faz-se necessário utilizar um mecanismo que as auxilie no entendimento dos humanos, tornando o Processamento de Linguagem Natural (PLN) essencial.

PLN é o desenvolvimento de modelos computacionais que realizem tarefas baseadas em informações expressas em linguagem natural [PEREIRA 2011]. O Processamento de Linguagem Natural visa promover o diálogo entre os humanos e as máquinas, aprimorando a comunicação homem-computador por meio da extração de textos em linguagem natural [JURAFSKY and MARTIN 2019]. Sidhu define PLN como um campo da Inteligência Artificial (IA) que se dedica a pesquisar como os computadores podem ser utilizados para manipular e entender a linguagem natural (seja ela falada ou escrita) a fim de desenvolver coisas úteis [SIDHU 2013], citado por [FARIA and BARBOSA 2020]. Dentre essas, destacam-se a construção de: tradutores automáticos, *parsers*, *chatbots* e geradores automáticos de resumos [OTHERO 2002].

O Processamento de Linguagem Natural é composto por diversas técnicas (como a Tokenização, a Etiquetagem Morfossintática, a Lematização, a Sumarização, dentre outras) e a utilização de cada uma depende muito da aplicação em que será utilizada. A seguir, serão consideradas duas das principais técnicas de PLN.

2.1. Tokenização

A tokenização é um dos primeiros passos para a utilização de PLN e um dos mais simples. É considerada como uma etapa de pré-processamento de texto e consiste na quebra da sequência de caracteres em um texto, a partir da localização do limite de cada palavra [BARBOSA et al. 2017].

A tokenização pode ser útil na normalização de um texto através, por exemplo, do mapeamento das palavras para versões apenas com letra minúscula, da expansão de contrações ou da extração do radical de cada palavra [BIRD et al. 2006]. Existe a possibilidade do algoritmo tokenizar expressões com várias palavras (New York, por exemplo) como um único *token*, o que requer um dicionário de expressões com várias palavras de algum tipo. Isso demonstra como a tokenização está intimamente relacionada à tarefa de identificar Entidades Nomeadas, como organizações, nomes, datas, dentre outras [JURAFSKY and MARTIN 2019].

A tokenização facilita a tarefa de análise das palavras mais relevantes para um texto por meio de uma distribuição de frequência, que apresenta a quantidade de vezes que cada *token* aparece em um documento [BIRD et al. 2006]. Distinguir as palavras mais significativas para um texto pode ser muito útil para a Categorização de Textos, Análise de Sentimentos e muitas outras áreas de aplicação do Processamento de Linguagem Natural.

2.2. Etiquetagem Morfossintática

A Etiquetagem Morfossintática é a atribuição de rótulos de classes de palavras a cada um dos *tokens* de uma sentença, considerando o contexto sentencial, [ALENCAR 2011] que pode classificar palavras únicas, expressões multi-palavras e sinais de pontuação de uma sentença de acordo com sua classificação gramatical (substantivos, verbos, adjetivos, etc). Quanto à utilização, a etiquetagem é de grande importância na análise gramatical, na tradução automática e na síntese de fala [DOMINGUES 2011].

Dentre os desafios da etiquetagem morfossintática, destaca-se o fato de que é preciso lidar com a ambiguidade das palavras conforme o contexto da palavra na sentença. Entende-se como palavra ambígua aquela que possui mais de uma categoria gramatical possível [DOMINGUES 2011]. Por exemplo: na frase “Segundo o que me disseram, no segundo dia de trabalho ela gastou menos de um segundo para resolver um impasse que até então perdurava na empresa.”, a palavra Segundo se classifica, respectivamente: no primeiro caso como uma conjunção subordinativa conformativa, no seguinte como numeral ordinal e no último caso como um substantivo [DUARTE].

3. Natural Language Toolkit

Natural Language Toolkit (NLTK) é uma plataforma que visa a construção de programas *Python* que trabalhem com dados de linguagem humana. O NLTK É um *software* livre, de código aberto e que fornece interfaces fáceis de usar para dezenas de corpora e diversos recursos lexicais, além de um pacote de bibliotecas de processamento de texto [BIRD et al. 2021].

Para o auxílio aos usuários do NLTK, seus criadores ainda publicaram um livro, o *Natural Language Processing with Python* (ou Processamento de Linguagem Natural com *Python*, em tradução livre), que se propõe a fornecer uma introdução prática e fácil para quem deseja iniciar na programação para processamento de linguagem [BIRD et al. 2021]. Nesta Seção, serão

abordados os principais recursos do NLTK: tokenização, *POS-tagger*, identificação e remoção de *Stopwords* e *Stemming*.

No NLTK, um *token* é considerado como uma sequência de caracteres que queremos tratar como um grupo [BIRD et al. 2006]. Uma forma simples de tokenizar uma *string* com NLTK é através do método *nltk.word_tokenize (string)*, conforme exemplificado na Figura 1.

```
frase = "Tenho uma consulta com o dentista amanhã, às 9:30."
tokens = nltk.word_tokenize(frase)
print (tokens)
```

```
['Tenho', 'uma', 'consulta', 'com', 'o', 'dentista', 'amanhã', ',', 'às', '9:30', '.']
```

Figura 1. Tokenização de uma *string* com NLTK.

O *POS-tagger* do NLTK corresponde à etapa de etiquetagem das palavras, considerada na Seção 2.2 deste artigo. Uma *POS-tagger* processa uma sequência de *tokens* e anexa uma *tag* gramatical a cada um deles [BIRD et al. 2019]. No NLTK, cada *tag* aparece logo após o *token* de forma abreviada, conforme exemplifica a Figura 2. Para identificar de qual classe gramatical a abreviação corresponde, utilize o comando *nltk.help.upenn_tagset( )* e acrescente entre os parênteses, com aspas simples, o nome da *tag* que deseja identificar.

```
frase = "Tenho uma consulta com o dentista amanhã, às 9:30."
tokens = nltk.word_tokenize(frase)
nltk.pos_tag(tokens)
```

```
[('Tenho', 'NNP'),
 ('uma', 'JJ'),
 ('consulta', 'NN'),
 ('com', 'NN'),
 ('o', 'NN'),
 ('dentista', 'NN'),
 ('amanhã', 'NN'),
 (',', ','),
 ('às', '$'),
 ('9:30', 'CD'),
 ('.', '.')]
```

Figura 2. Exemplo de *tagger POS* no NLTK.

Outro recurso interessante que o NLTK inclui é o *Stemming*, que é o processo de reduzir a palavra à sua raiz, ou parte central. O *Stemming* se concentra no significado básico de uma palavra, em vez de focar nos detalhes de como a palavra está sendo utilizada [JABLONSKI 2021]. Por exemplo: busca, busco, buscando e buscar têm a mesma raiz: busc. Apesar de o NLTK, de forma geral, possuir vários lematizadores prontos para uso e que lidam, inclusive, com casos irregulares [BIRD et al. 2009], os lematizadores para a língua portuguesa são limitados e escassos.

Os *stopwords* podem ser definidos como palavras irrelevantes para o entedimento do sentido de um texto [BARBOSA et al. 2017]. De forma geral, têm pouco conteúdo lexical e sua presença no texto não auxilia na distinção de outros textos [BIRD et al. 2021]. O NLTK possui um *corpus* de *stopwords* para a língua portuguesa, que permite a identificação e remoção das palavras irrelevantes com poucas linhas de comando. No exemplo da Figura 3, as palavras “uma”, “com”, “o” e “às” foram consideradas como *stopwords* e removidas, mas a frase continua fazendo sentido.

```
frase = "Tenho uma consulta com o dentista amanhã às 9:30"
palavras = nltk.word_tokenize(frase)
stopword = set(stopwords.words('portuguese'))
palavras = [palavra for palavra in palavras if palavra not in stopword]
print (palavras)
```

```
['Tenho', 'consulta', 'dentista', 'amanhã', '9:30']
```

Figura 3. Removendo palavras irrelevantes a partir do *Stopwords Corpus*.

4. Proposta

Em decorrência dos avanços tecnológicos dos últimos anos e impulsionado pelo surgimento da pandemia da Covid-19, em que manter o distanciamento social é estritamente necessário, tornou-se essencial disponibilizar o maior número possível de serviços de forma *online*. Um desafio que surge, então, é tornar o acesso a esses diversos serviços, espalhados em diversos sistemas e aplicativos, fáceis de serem encontrados e utilizados pelos eleitores.

O projeto propõe-se a criar uma aplicação para automatização dos serviços e informações do TRE-BA mais requeridos pelo usuário por meio de um *chatbot* humanizado com sintetizador de voz. É necessário que a solução aumente a disponibilidade de horário do atendimento aos eleitores (24 horas por dia, 7 dias por semana) e diminua o fluxo de atendimento presencial.

Outros potenciais benefícios em que espera-se alcançar com o projeto do *chatbot* Otto são: sanar as dúvidas dos usuários com mais agilidade, delegar e automatizar processos repetitivos, evitando assim a sobrecarga dos servidores e tornar mais fácil o acesso dos usuários às informações e/ou aos serviços que requeiram. Otto será utilizado pelos servidores do TRE-BA do Setor de Atendimento ao Cliente (SEACLI) e pelos eleitores de toda a Bahia. O sistema possui 13 Requisitos Funcionais e 9 Requisitos Não Funcionais que estão descritos, respectivamente, no Quadro 1 e no Quadro 2.

Código	Descrição
RF01	O sistema deve cadastrar as perguntas para o treinamento do chatbot.
RF02	O sistema deve cadastrar as respostas para o treinamento do chatbot.
RF03	O sistema deve responder conforme solicitação do usuário-eleitor.
RF04	O sistema deve iniciar com uma saudação.
RF05	O sistema deve responder as perguntas digitadas pelo usuário-eleitor em Linguagem Natural.
RF06	O sistema deve, ao enviar uma informação para o usuário-eleitor, aguardar a resposta.
RF07	O sistema deve interagir com usuário-eleitor por meio de áudio, quando solicitado.
RF08	O sistema deve interagir com o usuário-eleitor por meio de texto.
RF09	O sistema deve responder por meio de links, quando solicitado o serviço.
RF10	O sistema deve disponibilizar um formulário de pesquisa com 5 campos
RF11	O sistema deve terminar uma conversa com o usuário-eleitor através de despedidas.
RF12	O sistema deve durante a conversa com o usuário-eleitor compartilhar informações sobre a confiabilidade das urnas eletrônicas.
RF13	O Sistema deve permitir que o usuário-eleitor realize o agendamento presencial do serviço.

Quadro 1. Requisitos Funcionais do *chatbot*

Código	Classificação	Descrição
RNF01	Confiabilidade	Quando um link sair do ar, o sistema deve exibir uma mensagem com informações de contato.
RNF02	Confiabilidade	Caso o serviço solicitado não possa ser realizado online, o sistema deve disponibilizar o agendamento presencial.
RNF03	Usabilidade	O sistema deve ter uma interface intuitiva.
RNF04	Disponibilidade	O computador do usuário-eleitor deve estar conectado a web para acessar o sistema chatbot.
RNF05	Interoperabilidade	O chatbot será integrado no website do TRE.
RNF06	Usabilidade	Ao final do atendimento, o sistema deve enviar ao usuário-eleitor uma pesquisa de satisfação;
RNF07	Disponibilidade	Se durante a conversa o usuário-eleitor demorar mais de 15 minutos para responder, o sistema deve encerrar a conversa.
RNF08	Segurança	Conversas entre o chatbot e o usuário-eleitor não deverão ser armazenadas no banco de dados.
RNF09	Confiabilidade	O sistema deve identificar uma palavra ambígua numa frase digitada pelo usuário-eleitor e realizar sua desambiguação.

Quadro 2. Requisitos Não Funcionais do *chatbot*

5. Aplicação Prática

A solução compreende um sistema composto pelos seguintes módulos: (1) o módulo de Treinamento, onde são definidas as tabelas e os campos que compõem o banco de dados onde são cadastradas as perguntas e respostas utilizadas para treinamento do *chatbot*; (2) o módulo do *Chat*, que abrage a camada Cliente do *chatbot* e (3) o módulo do *Bot*, que compreende a lógica e configuração do funcionamento do *chatbot* através das 3 opções de navegação: por *Menu*, por Voz ou por digitação no *chat*, conforme apresentado na Figura 4. A qualquer momento, o usuário pode alternar entre as três formas de navegação, de acordo às suas necessidades e preferências.

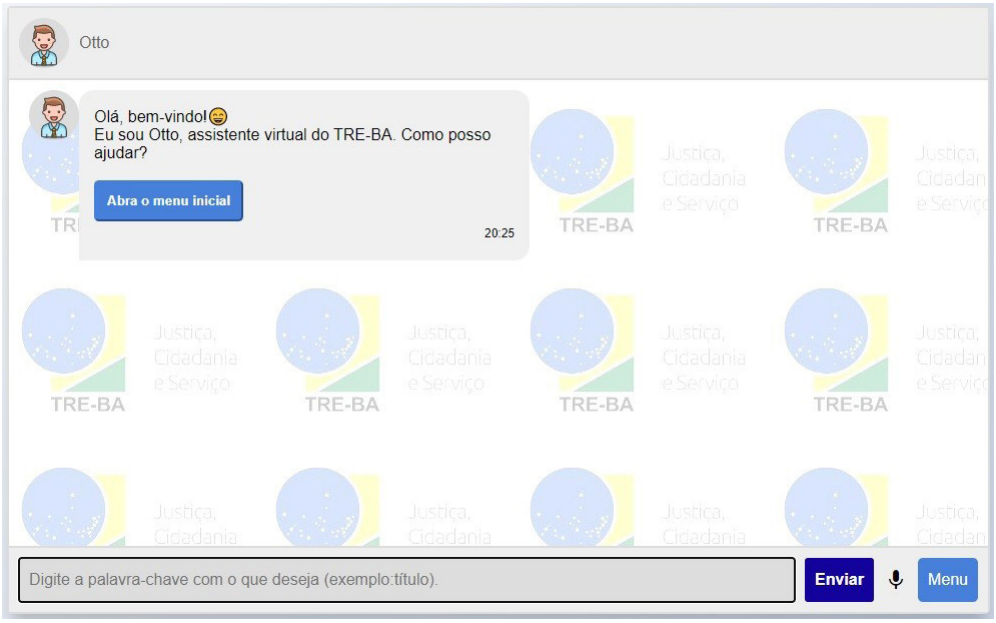


Figura 4. Captura de Tela do *Chatbot*.

O atendimento a partir dos *Menus* é o mais simples: consiste num *Menu* Inicial padrão com os serviços/informações mais buscados pelos usuários, que é composto de um título e várias opções de respostas em que o usuário pode clicar na que mais se aproximar do que ele deseja, conforme demonstrado na Figura 5. Ao passo que o usuário vai clicando na opção desejada, o *bot* vai caminhando na árvore de decisão e disponibilizando novas opções ao usuário, até que ele encontre o que deseja.

O atendimento por voz é realizado da seguinte forma: quando o usuário clicar, a qualquer momento, no ícone do microfone, o *bot* Otto passa a escutar o que o usuário vai falar. Essa mensagem é convertida para texto e enviada para o fluxo de atendimento por *chat*. Se não encontrar um microfone disponível, Otto informa ao usuário que não conseguiu encontrar um microfone e pede para que ele escolha uma das outras opções de navegação para dar continuidade ao seu atendimento. Caso o bot não entenda o que foi dito, ele pede ao usuário que repita o que disse.

Figura 5. Captura de Tela do atendimento por *Menu*.

Quando o usuário digita uma mensagem e clica no botão " Enviar" ou pressiona a tecla *Enter* em seu teclado, o atendimento via digitação no *chat* é ativado. Esse é o fluxo de atendimento que possui mais etapas. Primeiro, a mensagem precisa ser tratada: utilizando o NLTK, a mensagem é tokenizada e os *tokens* identificados como *Stopwords* ou como sinais de pontuação são removidos. Os *tokens* restantes são convertidos em letras minúsculas e sem acento.

A segunda etapa do atendimento por *chat* compreende a consulta dos *tokens* ao banco de dados, que possui um dicionário com os termos que representam cada serviço. Quando um *token* dá *match* com um dos termos cadastrados no banco, então se inicia a terceira etapa: a desambiguação de termos. Neste caso, a desambiguação de termos corresponde a identificar a intenção do usuário com o que foi dito e tal desambiguação é realizada por meio de *menus* de navegação.

Uma melhoria que pode ser aplicada futuramente no sistema é a utilização de um classificador que gere uma categorização de termos. Para isso, foi desenvolvido um *corpus*, em que cada arquivo em formato de texto simula uma frase utilizada pelo usuário para solicitar determinado serviço ou informação. O arquivo foi nomeado de acordo à categoria a qual pertence e cada categoria possui vários arquivos com formas diferentes de fazer a mesma pergunta.

A partir do corpus produzido, utilizando a NLTK, pode-se obter a Distribuição de Frequência de cada palavra por categoria e a relevância daquela palavra para a categoria à qual pertence. Após

isso, basta treinar um classificador que categorize a mensagem digitada pelo usuário. No exemplo a seguir, o classificador é treinado utilizando a métrica *Term Frequence - Inverse Document Frequency* (TF-IDF) e é realizado um teste que compara a categoria real e a categoria prevista pelo classificador (Figura 6).

```
Categoria Real: CertidaoQuitacao_
Categoria Prevista: CertidaoQuitacao_

Categoria Real: SegundaVia_
Categoria Prevista: SegundaVia_

Categoria Real: RevisaoTitulo_
Categoria Prevista: RevisaoTitulo_

Categoria Real: CertidaoQuitacao_
Categoria Prevista: CertidaoQuitacao_

Categoria Real: TransferenciaTitulo_
Categoria Prevista: TransferenciaTitulo_

Categoria Real: RegularizacaoTitulo_
Categoria Prevista: RegularizacaoTitulo_

Categoria Real: TransferenciaTitulo_
Categoria Prevista: TransferenciaTitulo_

Categoria Real: Alistamento_
Categoria Prevista: Alistamento_

Categoria Real: Alistamento_
Categoria Prevista: Alistamento_

Categoria Real: Alistamento_
Categoria Prevista: Alistamento_

Categoria Real: Multa_
Categoria Prevista: Multa_
```

Figura 6. Captura de Tela do teste de Categorização de Termos.

6. Considerações Finais

Este artigo abordou como a aplicação de técnicas de Processamento de Linguagem Natural (PLN) utilizando *Natural Language Toolkit* (NLTK) pode contribuir para odesenvolvimento de um *chatbot* humanizado.

O PLN foi a base para o funcionamento de uma das três opções de atendimento do *chatbot* desenvolvido: o atendimento por digitação no *chat*. Utilizando o NLTK, a mensagem digitada pelo usuário é tokenizada e os *tokens* passam por um pré-processamento. Na sequência, é realizada uma consulta ao banco de dados para identificar a correspondência do *token* com um dos termos cadastrados no banco. Além disso, foi dada uma sugestão de melhoria do sistema que pode ser desenvolvida posteriormente utilizando categorização de termos.

O sistema desenvolvido poderá, após sua implantação, contribuir positivamente com o TRE-BA por possibilitar a automatização de processos repetitivos e a diminuição do fluxo de atendimento presencial. Além disso, Otto poderá ser benéfico a todos os eleitores baianos, por trazer-lhes as informações e serviços necessários de uma forma prática e rápida.

Referências

ALENCAR, L. F. (2011). *Utilização De Informações Lexicais Extraídas Automa- ticamente De Corpora Na Análise Sintática Computacional Do Português.* Dis- ponível em: http://www.periodicos.letras.ufmg.br/index.php/relin/article/viewFile/2553/2505. Acesso em: 19 de nov. de 2021.

BARBOSA, J. L. N., VIEIRA, J. P. A., SANTOS, R. L. S., JUNIOR, G. V. M., MUNIZ, M. S., and MOURA, R. S. (2017). *Introdução ao Processamento de Lingua- gem Natural usando Python*. Disponível em: http://www.facom.ufu.br/~wendelmelo/terceiros/tutorial_nltk.pdf. Acesso em: 15 de nov. de2021.

BIRD, S., LOPER, E., and KLEIN, E. (2006). *Fundamentos De Processamento Linguístico: Tokenização De Textos E Classificação De Palavras*. Disponível em: http://nltk.sourceforge.net/doc/pt-br/tokenize.html. Acessoem: 30 de out. de 2021.

BIRD, S., LOPER, E., and KLEIN, E. (2009). *Natural Language Processing With Python*. Disponível em: http://www.datascienceassn.org/sites/default/ files/Natural%20Language%20Processing%20with%20Python. pdf. Acesso em: 16 de nov. de 2021.

BIRD, S., LOPER, E., and KLEIN, E. (2019). *Natural Language Processing With Python*. Disponível em: https://www.nltk.org/book/. Acesso em: 23 de nov. de 2021.

BIRD, S., LOPER, E., and KLEIN, E. (2021). *Natural Language Toolkit*. Disponível em: https://www.nltk.org/. Acesso em: 23 de out. de 2021.

DOMINGUES, M. L. C. S. (2011). *Abordagem Para O Desenvolvimento De Um Eti-quetador De Alta Acurácia Para O Português Do Brasil*. Disponível em: http: //repositorio.ufpa.br/jspui/handle/2011/2828. Acesso em: 14 de nov. de 2021.

DUARTE, V. M. N. *As Distintas Classes Adquiridas Por Uma Mesma Palavra*. Disponível em: https://brasilescola.uol.com.br/gramatica/as-distintas-classes-adquiridas-por-uma-mesma-palavra. htm. Acesso em: 07 de dez. de 2021.

FARIA, C. R. and BARBOSA, S. C. (2020). *Técnicas De Processamento De Lin- guagem Natural Para Auxiliar O Estudante Na Identificação Das Pragas Da Soja*. Disponível em: https://sol.sbc.org.br/index.php/sbie/article/download/12893/12747/. Acesso em: 07 de nov. de 2021.

JABLONSKI, J. (2021). *Natural Language Processing With Python's NLTK Package*. Disponível em: https://realpython.com/nltk-nlp-python/. Acesso em: 27 de nov. de 2021.

JURAFSKY, D. and MARTIN, J. H. (2019). *Speech and Language Proces- sing*. Disponível em: https://web.stanford.edu/~jurafsky/slp3/old_oct19/ed3book.pdf. Acesso em: 28 de nov. de 2021.

OTHERO, G. d. A. (2002). *Lingüística Computacional: Uma Nova Área De Pesquisa Para Os Estudantes De Letras*. Entrelinhas, 5 edition.

PEREIRA, S. L. (2011). *Processamento de Linguagem Natural*. Disponível em: https: //www.ime.usp.br/~slago/IA-pln.pdf. Acesso em: 01 de dez. de 2021.

SIDHU, B. K. (2013). *Natural Language Processing*. International Journal of ComputerTechnology and Applications.

Cheli dos Santos Mendes
Instituto de Computaçao – Universidade Federal da Bahia (UFBA)
CEP: 40.170-110 - Avenida Adhemar de Barros, s/n - Campus de Ondina –
Salvador – Bahia – Brasil
chelimendes30@gmail.com

Um Sistema de Assistência Virtual para o TRE: Engenharia e Documentação Arquitetural

Abstract. *This study proposes the development of a virtual assistant or chatbot that uses natural language processing to serve the citizen at the Regional Electors Court of Bahia - TRE-BA. The theoretical framework includes definition of chatbot and natural language processing (PLN), Ain the proposal, we chose to develop the requirements specification with functional and non-functional requirements with quality attributes, the UML diagram. System architecture, with records of architectural decisions, description of the product with its requirements and restrictions to adapt to the environment and finally the final conclusions.*

Resumo. Este estudo propõe o desenvolvimento de um assistente virtual ou chatbot que utiliza processamento de linguagem natural para o atendimento ao cidadão no Tribunal Regional Eleitoras da Bahia - TRE-BA. O referencial teórico inclui definição sobre chatbot e processamento de linguagem natural(PLN).Na proposta, optou-se por desenvolver a especificação de requisitos funcionais e não funcionais com os atributos de qualidade, o diagrama da UML, casos de uso do sistema, a arquitetura do sistema com os registros de decisões arquitetural, descrição do produto com seus requisitos e restrições de adaptação ao ambiente e, por fim, as conclusões finais.

1. Introdução

Com as mudanças nos meios de comunicação digital, as ferramentas de automação estão tornando mais eficientes os canais de comunicação. O Chatbot - ou Chatterbot que vem do inglês, onde *chatter* significa conversa fiada e *bot* uma abreviação de robot, rôbo em português - consiste em uma aplicação que proporciona a interação entre uma máquina e um ser humano, usando técnicas de Inteligencia Artificial(IA), com o propósito de simular a habilidade de conversação de um agente computacional como um ser humano [Araújo 2020].A IA foi cunhada na década de 50 por um cientista da computação chamado John McCarthy, como a ciência e engenharia de máquinas inteligentes, especialmente programas de computadores. [Neuhauser, Kreps 2014].

Os chatbots funcionam da seguinte forma: recebem uma mensagem em linguagem natural - enviada pelo o usuário - o programa consulta uma base de conhecimento e em seguida fornece uma resposta que tenta imitar o comportamento humano [Teixeira 2003].Tendo em vista o aumento do número de atendimento de serviços oferecido no Tribunal Regional Eleitoral da Bahia (TRE), principalmente por causa da pandemia, pensou-se no chatbot baseado em inteligência artificial, utilizando também mecanismos de regras. Este chatbot utiliza técnicas de IA para tentar entender a pergunta dos usuários durante uma conversação, com o objetivo de respondê-la corretamente.

Essencialmente, à medida que o usuário escreve sua mensagem, o chatbot vai identificando e entendendo o que esse usuário quer dizer com base em palavras-chave pré-definidas.

A solução chatbot tornou-se uma peça fundamental no atendimento ao cliente, automatizando tarefas repetitivas, disponibilizando atendimento 24 horas, rápido, direcionada, inclusivo, e sua conversação/atendimento humanizado melhora a capacidade do relacionamento com o usuário-eleitor.

O objetivo deste trabalho é apresentar um sistema desenvolvido para possibilitar e auxiliar eleitores na execução de diversos serviços oferecidos pelo TRE, como: consulta de seção, emissão de certificados, busca por zonas eleitorais e mais 189 serviços. Além disso, evitar a sobrecarga dos colaboradores servidores do TRE em momentos de grande demanda.

Atualmente o TRE-BA faz uso de um assistente virtual que atende via WhatsApp, funcionando de forma limitada por ser baseado em regras (não contém inteligência artificial) e está disponível apenas em texto. Ele será substituído pelo chatbot com Inteligência Artificial integrado ao website do TRE.

O projeto compreende dois sistemas: um para armazenamento de perguntas e respostas para o treinamento do chatbot, com uma interface amigável que permite ao usuário servidor cadastrar essas informações de forma fácil e intuitiva; outro que realiza a interação com conversão de texto para áudio (e vice-versa) com o usuário eleitor, utilizando Aprendizagem de Máquina e Processamento de Linguagem Natural.

1.1. Motivação

O presente projeto tem por motivação contribuir com o TRE-BA e com o cidadão elei- tor, na melhoria do atendimento e garantindo disponibilidade dos serviços a qualquer momento do dia.

Levando em conta as tecnologias utilizadas na facilitação de atendimento ao público e utilização em grande escala de smartphones, a pesquisa enfatiza o desenvolvi- mento de um chatbot.

1.2. Objetivos

O presente trabalho tem por objetivo desenvolver uma ferramenta de conversação, um assistente virtual com processamento de linguagem natural para agilizar o atendimento do cidadão eleitor.

1.2.1 Objetivos Específicos

Visando alcançar o objetivo principal, alguns objetivos específicos são requeridos:

- Analisar os conceitos, ferramentas, métodos para melhoria do atendimento ao usuário-eleitor.
- Analisar as aplicações já utilizadas no mercado.
- Identificar os serviços oferecidos pelo TRE.
- Desenvolver a documentação técnica da aplicação.
- Construir a aplicação.

2. Referencial Teórico

Antes de apresentarmos a Engenharia e Documentação Arquitetural do Chatbot, mostramos as principais características do Chatbot e o Processamento de Linguagem Natural(PNL).

2.1. Chatbot - assistente virtual para atendimento ao cliente

A teoria da computação tem auxiliado bastante no desenvolvimento de modelos matemáticos para concepção de novas ferramentas e, até mesmo, novas tecnologias como os bots ou chatbots, que é um programa de computador feito para interação com pessoas, por meio de linguagem natural, simulando um humano[Júnior e Carvalho 2018]. E conciliado com os avanços da inteligência

artificial, processamento de linguagem natural, banco de dados e redes de comunicação de dados foram essenciais para que os assistentes virtuais tivessem os resultados e avanços de hoje [Rebecchi 2020], suportando interações baseadas em voz ou texto.

Assistentes virtuais como o Google Now e a Siri são amplamente utilizados e recebem grandes investimentos de seus desenvolvedores. Google, Facebook, Whatsapp, Telegram, IBM e outras empresas já possuem ou estão desenvolvendo ferramentas para integração com chatbots [Cunha, Silva, Moura 2020].

Cada vez mais os assistentes virtuais têm tornado a vida das pessoas mais fácil, executando tarefas a pedido de seus usuários. Com a inteligência artificial, fazem tarefas mais avançadas, como prever atividades que precisam ser feitas, localizar sites ou realizar atividades na internet [Rebecchi 2020].

2.2. Processamento de Linguagem Natural(PLN)

O processamento da linguagem natural (PLN) trata computacionalmente os diversos aspectos da comunicação humana, como sons, palavras, sentenças e discursos, considerando formatos e referências, estruturas e significados, contextos e usos. Em sentido bem amplo, podemos dizer que o PLN visa fazer o computador se comunicar em linguagem humana, nem sempre necessariamente em todos os níveis de entendimento e/ou geração de sons, palavras, sentenças e discursos [Gonzalez, Lima 2011].

Por conseguinte, tal área tem por objetivo sistematizar e estabelecer uma metodologia para analisar textos e gerar frases do idioma humano, de forma que possamos nos comunicar com as máquinas de maneira intuitiva e descomplicada como se estivéssemos nos comunicando com outra pessoa [Crivelli 2011].

O PLN geralmente faz uso de conceitos linguísticos como classes de palavras (substantivo, verbo, adjetivo,etc.), aqui chamadas de Part-Of-Speech, além de estruturas gramaticais [Barbosa, Vieira, Santos, Junior, Muniz, Moura 2017].Entre muitos exemplos de sistemas que operam com o conceito de PLN, podem-se citar os chatbots.

A diferença entre o Conversational Hypertext Access Technology (CHAT) e os convencionais, e´ que nestes há necessidade de haver pessoas interagindo com pessoas por meio do computador, já os chatbots são programados para tentar imitar um interlocutor em conversação. Em relação a um interlocutor humano, eles têm o diferencial de trabalhar com um banco de dados bem mais preciso em se tratando de um assunto específico, ou seja, não corre o risco de esquecer ou confundir alguma informação [Manfio 2014].

3. Um Sistema de Assistência Virtual para o TRE: Engenharia e Documentação Arquitetural

O propósito do projeto é desenvolver um Sistema de Assistência Virtual para o TRE contendo a Engenharia e Documentação Arquitetural que vai oferecer mais conforto ao cidadão na realização de consultas de informações ou serviços do TRE-BA. Além disso, fornecer uma maior disponibilidade de atendimento (24 horas / 7 dias semanais), sobretudo evitar a sobrecarga dos colaboradores servidores em momentos de grande demanda.

3.1. Especificação de Requisitos

Após levantamento e análise de informações colhidas dos servidores do TRE, apresento os Requisitos Funcionais e Não Funcionais do sistema.

Requisitos Funcionais do Sistema

- RF01 - O sistema (módulo de cadastro) deve permitir cadastrar as perguntas para o treinamento do chatbot.

- RF02 - O sistema (módulo de cadastro) deve permitir cadastrar as respostas para o treinamento do chatbot.
- RF03 - O sistema deve responder conforme solicitação do usuário-eleitor.
- RF04 - O sistema deve iniciar com uma saudação.
- RF05 - O sistema deve responder as perguntas digitadas pelo usuário-eleitor em Linguagem Natural.
- RF06 - O sistema deve, ao enviar uma informação para o usuário-eleitor, aguardar a resposta do usuário-eleitor.
- RF07 - O sistema deve interagir com usuário-eleitor por meio de áudio, quando solicitado.
- RF08 - O sistema deve interagir com o usuário-eleitor por meio de texto.
- RF09 - O sistema deve responder por meio de links, quando solicitado determinado serviço.
- RF10 - O sistema deve disponibilizar formulário de pesquisa com "X" campos (campos).
- RF11 - O sistema deve terminar uma conversa com o usuário-eleitor através de despedidas.
- RF12 - O sistema deve durante a conversa com o usuário-eleitor compartilhar informações sobre a confiabilidade das urnas eletrônicas.
- RF13 - O sistema deve permitir que o usuário-eleitor realize o agendamento presencial do serviço.
- RF14 O sistema deve exibir um menu, quando a(s) palavras chaves ou frases forem digitadas pelo usuário-eleitor.
- RF15 - O sistema deve exibir uma mensagem de informação/explicativa no campo de digitação para o usuário-eleitor.
- RF16 O sistema deve exibir uma mensagem caso não reconheça o que foi digitado pelo usuário -eleitor.

Requisitos Não Funcionais do Sistema

- RNF01 - Confiabilidade: Quando o site de um link sair do ar, o sistema deve exibir uma mensagem com informações (Exemplo: telefone para contato ou e-mail).
- RNF02 - Confiabilidade: Caso o serviço solicitado não possa ser realizado online, o sistema deve disponibilizar o agendamento presencial. .
- RNF03 - Usabilidade: O sistema deve ter uma interface intuitiva.
- RF04 - O sistema deve iniciar com uma saudac¸a˜o.
- RNF04 - Disponibilidade: O computador do usuário-eleitor deve estar conectado à web para acessar o sistema chatbot.
- RFNF05 - Disponibilidade: O sistema deve disponibilizar as opções de digitar a solicitação e menu para o usuário-eleitor.
- RNF06 - Interoperabilidade: O chatbot será integrado ao website do TRE.
- RNF07 - Usabilidade: Ao final do atendimento, o sistema deve enviar ao usuário- eleitor uma pesquisa de satisfação.
- RNF08 - Disponibilidade: Se durante a conversa o usuário-eleitor demorar mais de 15 minutos para responder, o sistema deve encerrar a conversa.
- RNF09 - Segurança: Conversas entre o chatbot e o usuário-eleitor deverão ser armazenadas no banco de dados.
- RNF10 - Confiabilidade: Utilizando Processamento de linguagem natural o sistema deve identificar se na frase digitada pelo usuário-eleitor há algum do substantivo cadastrado no banco de dados.
- RNF11 - Confiabilidade : O sistema deve identificar uma palavra ambígua numa frase digitada pelo usuário-eleitor e realizar sua desambiguação.

  - Desambiguação de Palavras: Para realizar a desambiguação léxica de sentido de uma palavra na frase digitada pelo usuário, deve-se recorrer a frases anteriores da conversa, extraindo as palavras-chave de cada frase para encontrar o contexto da conversa e, através do contexto, descobrir o sentido da palavra.

3.2 Atributos de Qualidade

Visando atender alguns requisitos não-funcionais do sistema como:

- O computador do usuário-eleitor deve estar conectado à web para acessar o sistema chatbot (Se durante a conversa o usuário-eleitor demorar mais de 15 minutos para responder, o sistema deve encerrar a conversa);
- Quando o site de um link sair do ar, o sistema deve exibir uma mensagem com informações (Exemplo: telefone para contato ou e-mail);
- O sistema deve, ao enviar uma informação para o usuário-eleitor, aguardar a resposta;
- Conversas entre o chatbot e o usuário-eleitor não deverão ser armazenadas no banco de dados;
- Apenas os usuários-servidores terão acesso ao módulo de treinamento do chatbot, onde são definidos o que o *bot* deve responder quando questionado sobre cada assunto e o controle de acesso de usuário ao módulo de treinamento será realizado por meio de login e senha;

- Optou-se pelos atributos da Disponibilidade, Modificabilidade e Segurança, propriedades desejadas para solução:

- Disponibilidade é a possibilidade de que os sistemas sempre estejam ligados e funcionando,para que os usuários possam utilizar os serviços sempre que desejarem.
- Modificabilidade visa diminuir tempo e custo para implementar, testar e implantar mudanças.

- O atributo de Segurança foi avaliado de acordo às propriedades descritas a seguir:

- Integridade: a propriedade de garantir que os dados são entregues como combinado pelas partes;
- Confidencialidade: a propriedade dos dados ou serviços serem protegidos de acesso não autorizado;
- Autenticidade: a propriedade de armazenar as atividades realizadas no sistema em níveis suficientes para que as mesmas possam ser desfeitas ou refeitas;
- Garantia: a propriedade de garantir que as partes envolvidas em uma transação são quem elas dizem ser.

- Visando atender os atributos de qualidade, que devem ser considerados em todo o processo de software, será utilizada a arquitetura de três (03) camadas. A Figura 1 apresenta a Estrutura em Camadas que atende à estrutura de qualidade.

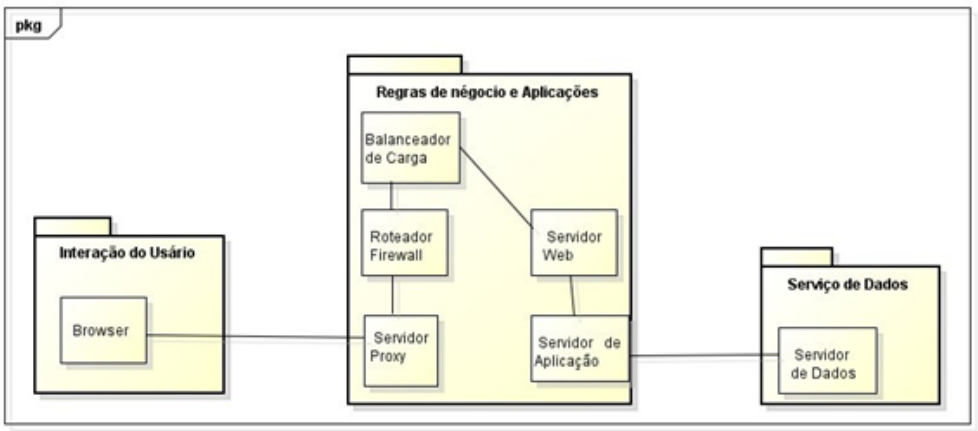


Figura 1. Estrutura da qualidade.

Interfaces que suportam browser são implementadas em HTML. Documentos HTML são facilmente modificados.

Servidores Proxy mantêm no seu cache arquivos disponíveis em servidores remotos, diminuindo o tempo de acesso e aumentando o desempenho.

Firewall - Requisições de um browser (ou servidor proxy) chegam ao roteador que provê comunicação entre os computadores, roteadores incluindo Firewall para prevenir fluxos de informações não autorizadas.

Balanceador de Carga para Desempenho, Escalabilidade e Disponibilidade: distribui a carga entre os vários computadores rodando servidores WEB.

Servidor Web - A requisição HTTPS chega então ao servidor WEB. Servidores multi-thread permitem executar várias tarefas ao mesmo tempo.

Servidor de Aplicação para modificabilidade, desempenho e escalabilidade, requisição é então direcionada do servidor WEB para o servidor de aplicação. Objetivo é disponibilizar uma plataforma, que abstrai do desenvolvedor de software complexidades do sistema computacional.

Servidor de Banco de Dados para modificabilidade, desempenho e escalabilidade. Modernos BD usam replicação interna para conseguir performance, escalabilidade e disponibilidade. Eles também usam cache para garantir bom desempenho.

3.3. Diagrama de Casos de Uso

A Figura 2 apresenta o diagrama de Casos de Uso, que demonstra as diferentes maneiras que os usuários podem interagir com um sistema.

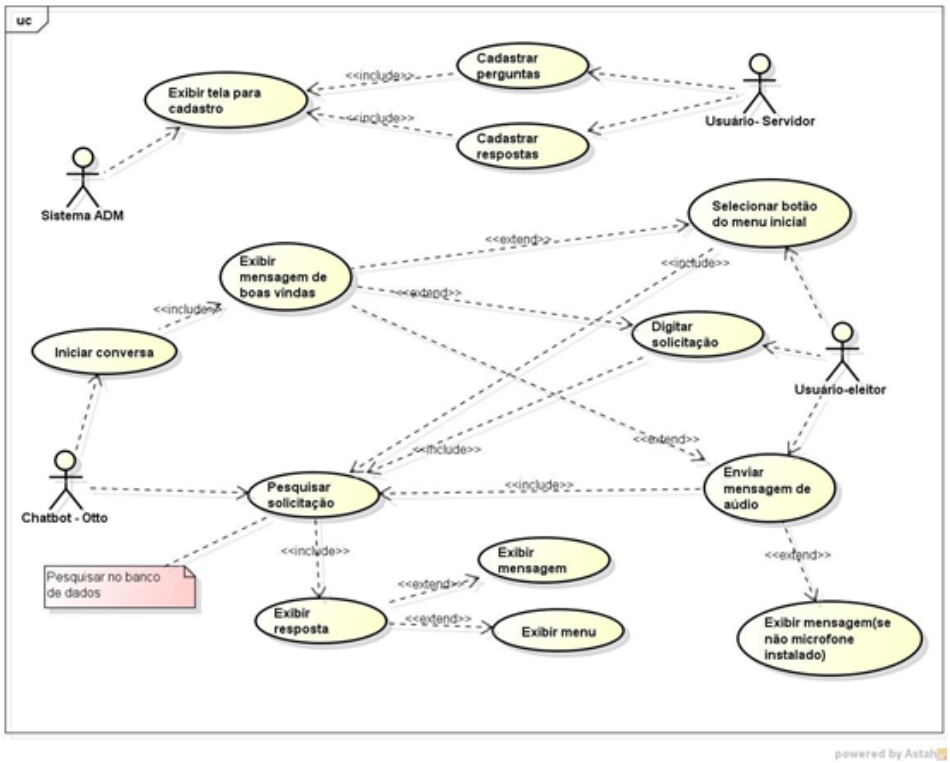


Figura 2. Casos de Uso.

3.4. Arquitetura do Sistema

O estilo arquitetural adotado é o Model-Template-View (MTV), também utilizado no framework Django. A Figura 3 apresenta a visão geral do modelo.

Conforme apresentado na Figura 3, a estrutura do Django é composta de três camadas bem definidas:

- O Model vai servir como base da aplicação, onde vai ser extraído e persistida informação da aplicação, fazendo conexão (mas não necessariamente) com a camada View. O banco de dados padrão do Django é o SQL e é nesta camada que ocorre a manipulação dos dados deste banco.
- Na camada de Template, os dados vão ser apresentados no browser. Esta camada consiste basicamente de arquivos.html com algumas funcionalidades que o Django fornece para apresentar os dados vindos da camada View. Esta é a parte que diz respeito a tudo que o usuário final é capaz de visualizar em seu dispositivo.
- A View é a camada onde as informações dos templates são tratadas. Existem dois tipos de views, as baseadas em função e as baseadas em classes. São funções Python que recebem uma requisição e envia uma resposta em retorno. Ela funciona como um intermediário entre as camadas Model e Template, acessando os dados da primeira camada e transmitindo para a segunda.

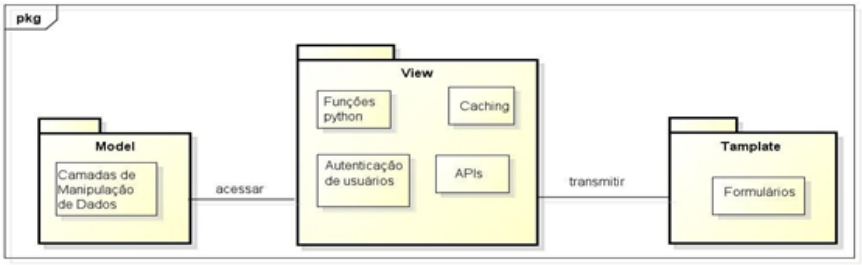


Figura 3. Estrutura em camadas do Django.

3.5. Princípios de Design

Modularidade: para gerenciar a complexidade do sistema e tornar mais prático seu desen- volvimento, a solução será dividida nos seguintes módulos:

- Módulo do Chatbot: é nessa parte do sistema em que há a automatização das respostas e o desenvolvimento da inteligência artificial por meio de Aprendizagem de Máquina e Processamento de Linguagem Natural.
- Módulo de Treinamento: neste módulo com interface amigável, o usuário-servidor irá alimentar a base de treinamento do chatbot, por meio do preenchimento das possíveis perguntas que podem ser feitas pelo usuário-eleitor, e qual deve ser a reação do chat- bot diante de tal estímulo (no caso de uma saudação, responder textualmente; para uma requisição de serviço, enviar o link para o serviço solicitado ou uma possibilidade de agendamento presencial, caso o serviço requerido não possa ser realizado virtualmente).
- Módulo cliente: etapa em que o chat e a interface do usuário-eleitor são desenvolvidos. A Figura 4 demonstra essa modularização.

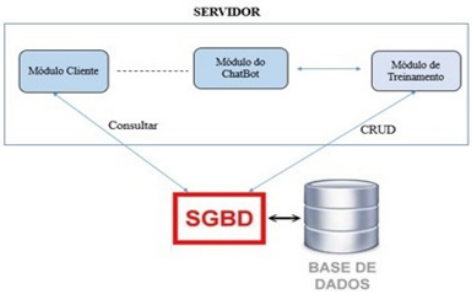


Figura 4. Estrutura dos módulos.

Facilidade de manutenção: todo o sistema está sendo desenvolvido considerando que os mantenedores do código não serão os seus criadores. Por isso, serão adotadas algumas medidas para desenvolver um sistema de fácil manutenção, que são às descritas abaixo:

- Linguagem de programação única para todos os módulos;
- Produção de documentação geral e arquitetural;
- Escolha de um estilo arquitetural que facilite a manutenção (MTV/MVC, conforme abordado anteriormente).

3.6. Design Patterns

O Singleton irá contribuir fortemente para o desenvolvimento deste projeto, pois além de ser um dos padrões de projetos mais utilizados, ele servirá para evitar inconsistências e duplicações de informações no banco de dados. Afinal, os módulos do chatbot e do treinamento são integrados por meio do banco de dados e ambos deverão ser capazes de manipulá-lo de forma organizada e única, como apresentado na Figura 4.

3.7. Registros de Decisões Arquiteturais (ADR)

Nas Tabelas abaixo, são apresentados os registros de decisões arquitetural(ADRs).

ADR 1 - Estilo Arquitetural

Contexto: A escolha de um estilo arquitetural adequado ao projeto é de suma importância para sua viabilidade.

Decisão: A arquitetura selecionada para este projeto foi a arquitetura em 3 camadas no padrão MVC/MTV.

Justificativa: O estilo arquitetural MVC vem sendo amplamente usado para aplicações web, que apresenta como principais vantagens sua modularidade e facilidade de entendimento do projeto por parte de novos programadores que não participaram da sua criação. Este último aspecto é crucial, visto que após o término do Projeto da Residência, os servidores do Tribunal Regional Eleitoral da Bahia que não fizeram parte do projeto passarão a ser os responsáveis por sua manutenção e melhorias.

Situação: Aceito.

Consequência: Estudar de forma mais profunda como funciona o MVC e analisar como ele pode ser integrado ao projeto.

ADR 2 - Linguagem de Programação Python

Contexto: Chatbots baseados em inteligência artificial de alta qualidade exigem uma linguagem de programação que dê um bom suporte à Aprendizagem de Máquina e ao Processamento de Linguagem Natural.

Decisão: Todos os módulos da solução serão desenvolvidos na Linguagem de Programação Python.

Justificativa: Além de ser uma linguagem de alto nível, com boa curva de aprendizagem e ser de código aberto, Python atualmente é a Linguagem de Programação referência no ramo da inteligência artificial.

Situação: Aceito.

Consequência: Aprofundar os conhecimentos sobre o Python e buscar bibliotecas que auxiliem o desenvolvimento da inteligência artificial do Python.

ADR 3 - Escolha dos Atributos de Qualidade

Contexto: Um software bem-feito deve cumprir os atributos de qualidade que atendam aos requisitos definidos para o projeto.

Decisão: Adotar como atributos de qualidade a disponibilidade, a modificabilidade e a segurança.

Justificativa: Pelo fato de ambos serem imprescindíveis para os requisitos funcionais e não-funcionais estabelecidos neste projeto. Afinal, uma das maiores vantagens em ter um chatbot atendendo aos clientes em vez de um humano é que o bot estará ao dispor dos usuários 24 horas por dia, 7 dias por semana, até mesmo durante os feriados. Como para encaminhar o usuário-eleitor ao serviço online que ele deseja é necessário ter acesso a seus dados pessoais, a segurança torna-se igualmente essencial e inegociável para o sucesso do projeto.

Situação: Aceito.

Consequência: Aplicar uma Tática Arquitetural compatível com os atributos de qualidade selecionados.

ADR 4 - Framework Web Django

Contexto: Os frameworks web são importantes para promover uma base estável para o desenvolvimento de projetos web. Escolher o framework ideal para cada tipo de projeto tem impacto direto em ganho ou perda de produtividade.

Decisão: Adotar o Framework Django para o desenvolvimento da solução de chat-

bot.

Justificativa: O Django é uma ferramenta completa, contendo soluções para problemas que vão desde os mais simples aos mais complexos, porque serve tanto a profissi- onais novatos quanto aos mais experientes.

Situação: Aceito.

Consequência: Aprender como funciona o Framework e entender como utilizar sua arquitetura MTV para adaptar a solução à essa arquitetura.

ADR 5 - Princípios de design

Contexto: Com intuito de construir módulos no framework Django, entendendo que estes módulos (Treinamento e Chatbot) vão interagir com o SGBD, sendo que o módulo de treinamento receberá as perguntas e respostas para o treinamento do chatbot e o módulo do chatbot realizará as consultas das informações armazenadas no banco de dados.

Decisão: Utilizar os princípios de design, modularidade e facilidade de manutenção.

Justificativa: Utilizar os princípios de modularidade e facilidade na manutenção facilitará o gerenciamento da complexidade, manutenção e melhorias dos módulos.

Situação: Aceito.

Consequência: Cada módulo tem sua finalidade, com isso a manutenção e atualização ocorrerá de forma fácil e precisa.

ADR 6 - Escolha de Padrões de Projeto

Contexto: Padrões de projeto podem facilitar a vida dos desenvolvedores e otimi- zar o tempo de produção, por evitar que os desenvolvedores precisem ficar "reinventando a roda" o tempo todo. Por outro lado, utilizar um padrão que não se encaixe com o projeto pode ter o efeito contrário.

Decisão: Utilizar o padrão de projeto Singleton.

Justificativa: Visto que dois módulos do chatbot irão manipular o banco de dados, utilizar Singleton fará um isolamento, para que apenas um dos módulos possa manipular as informações do banco por vez. Este padrão também reduz o consumo de memória desnecessário.

Situação: Aceito.

Consequência: Aprofundar os conhecimentos sobre como o Singleton funciona e como implementá-lo.

4. Descrição geral do produto

4.1. Modelo de Dados - DER

Na Figura 5, é apresentado o Diagrama de Entidade e Relacionamento (DER) da Bibliotecas Python (chatterbot) que é utilizada para o funcionamento do sistema chatbot.

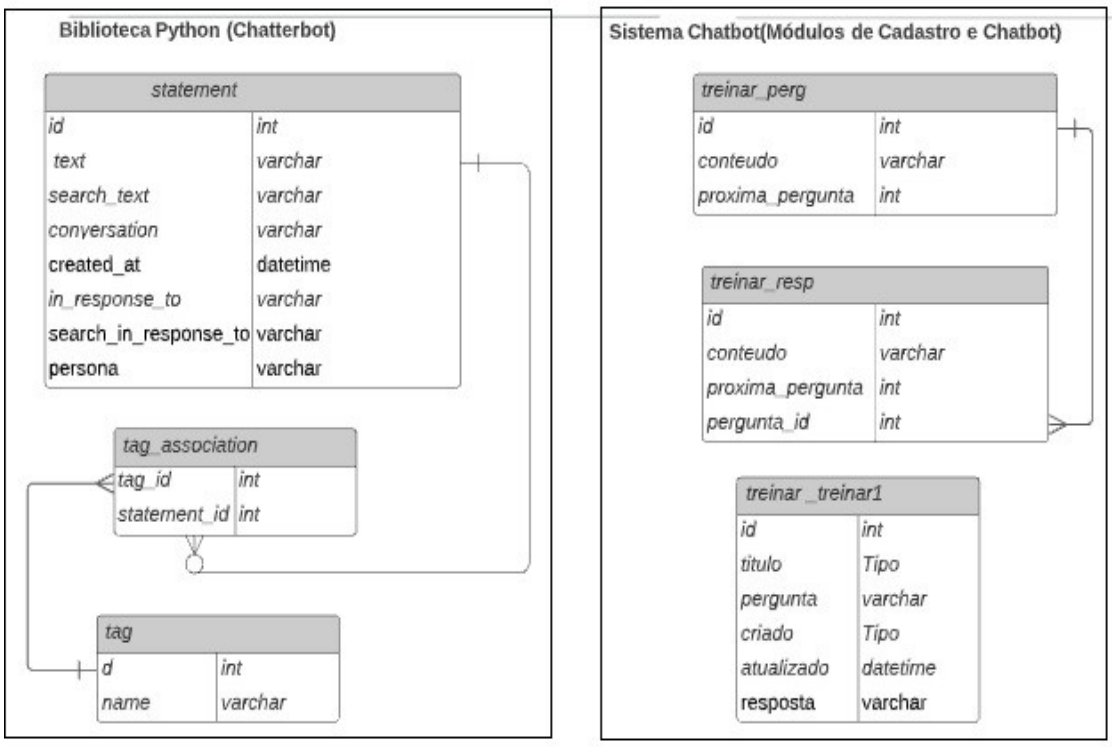


Figura 5. Diagrama de Entidade Relacionamento.

4.2. Interfaces de usuário

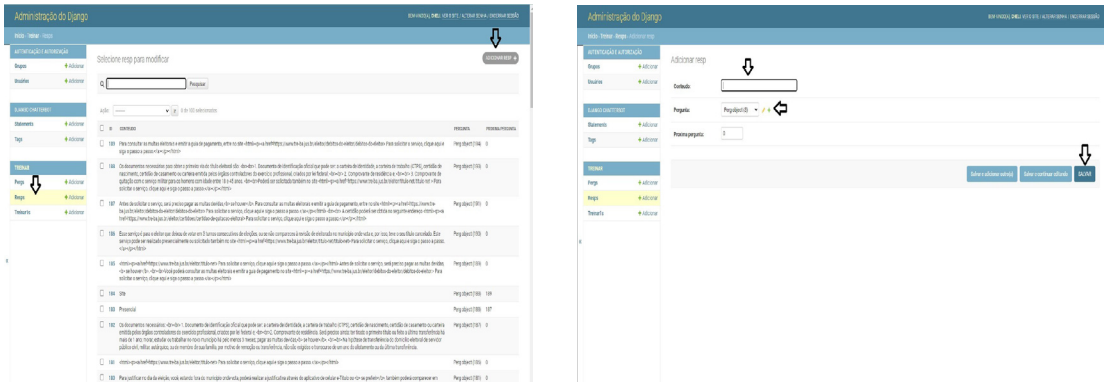
Neste item são apresentadas as interfaces do produto.

(a) Tela consultar pergunta

(b) Tela cadastrar pergunta

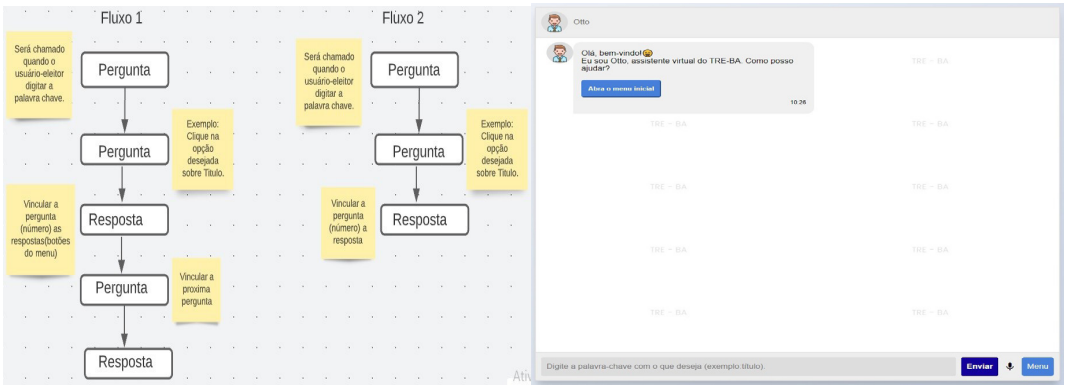
Figura 6. Telas de consulta e cadastro de perguntas.

A Figura 7 apresenta a tela de cadastro das respostas. Salientando que na resposta deve-se vincular a pergunta, também vincular uma outra pergunta caso haja.



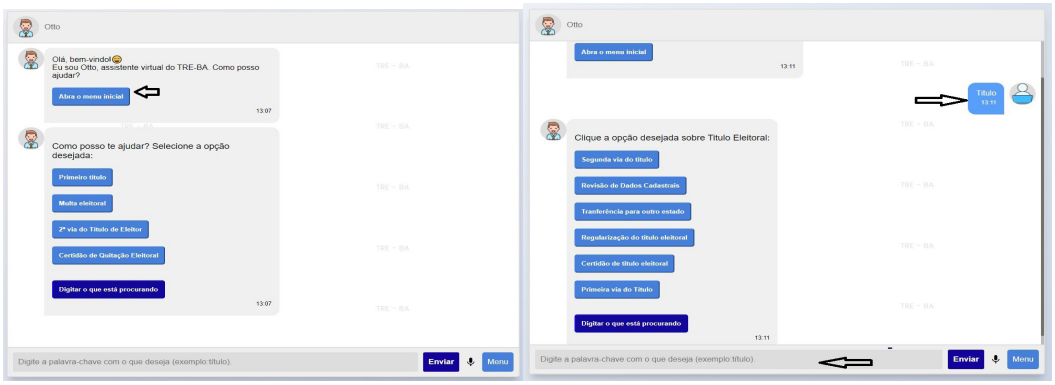
(a) Tela consultar resposta **(b) Tela cadastrar resposta**

Figura 7. Telas de consulta e cadastro de perguntas.



(a)**Fluxo de cadastro das perguntas e respostas** **(b) Tela inicial do chatbot**

Figura 8. Fluxo de como cadastrar as perguntas e respostas no modulo de cadastro e a Tela inicial do chatbot Otto.



(a)**Tela do menu do chatbot Otto** **(b) Tela inicial do chatbot**

Figura 9. Fluxo de como cadastrar as perguntas e respostas no modulo de cadastro e a Tela inicial do chatbot Otto.

Na figura 9, Quando o usuário-eleitor digita uma ou mais palavra chave, Otto exibe um menu inicial relacionando a informação solicitada. (b) Quando o usuário-eleitor clicar no botão"Abra o menu inicial", Otto exibe um menu inicial com os serviços mais solicitados do TRE. (a)

4.3. Requisitos de adaptação ao ambiente

Tabela 1. Requisitos de adaptação ao ambiente

Número de Ordem	Requisitos	Detalhes
1	Instalar o python no computador	Verificar na configuração de variáveis de ambiente do sistema Windows.
2	Configuração do Framework Django	Configuração de administrador, manipulação do bando de dados e ativação para ao funcionamento.
3	Configuração do Docker	Configuração também do pacote de atualização do kernel Linux.
4	Executar o arquivo requirements.txt	Neste arquivo tem todos os comandos para o funcionamento dos módulos
5	Configuração do microfone	Permitir que os aplicativos acessem seu microfone e os app podem acessar seu microfone.
6	Instalar a biblioteca chatterbot do python	Essencial para o funcionamento do chatbot.

4.4. Restrições

Os aspectos técnicos e gerenciais que podem limitar o projeto:

a) Versões antigas do sistema de gerenciamento de banco de dados não funcionam com o framework Django;

b) Deve-se realizar periodicamente a atualização do Docker (contêiner de virtualização);

c) Executar e atualizar sempre o documento requirements.txt.

d) Observar as atualizações recorrentes no framework Django.

4.5. Requisitos adiados

Os requisitos identificados na elaboração deste projeto, mas que se decidiu deixar para versões futuras:

Requisitos Funcionais

RF10 - O sistema deve disponibilizar formulário de pesquisa com “X” campos.

RF12- O sistema deve durante a conversa com o usuário-eleitor compartilhar informações sobre a confiabilidade das urnas eletrônicas.

RF13- O Sistema deve permitir que o usuário-eleitor realize o agendamento presencial do serviço.

Requisitos Não Funcionais

RNF01 - Confiabilidade: Quando o site de um link sair do ar, o sistema deve exibir umamensagem com informações (Exemplo: telefone para contato ou e-mail).

RNF06 - Interoperabilidade: O chatbot será integrado ao website do TRE.

RNF07 - Usabilidade: Ao final do atendimento, o sistema deve enviar ao usuário-eleitor uma pesquisa de satisfação.

RNF09 - Segurança: Conversas entre o chatbot e o usuário-eleitor deverão ser armazenadas no banco de dados.

Arquitetura do sistema

Táticas de Arquitetura: As táticas para garantir o atributo Disponibilidade são baseadas em:

Redundância de componentes (processos, arquivos, equipamentos, monitoramento para prevenir falhas, mecanismos de detecção de falhas a ocorrer ou ocorridas e mecanismos de recuperação de falhas ocorridas.

As táticas devem considerar os seguintes fatores: frequência das falhas, consequências das falhas e tempo para recuperação.

No atributo modificabilidade as táticas tem os seguintes objetivos: manter modificações localizadas, evitar propagação (efeito ripple) e prorrogar tempo de ligação (ping).

Quanto a garantir o atributo da Segurança, está responsável por proteger os dados e controlar o nível de acesso de cada usuário.

5. Aplicação Prática

Em um dos seus arquivos do sistema temos o otto, que tem a classe otto, com a função que executa o treinamento do chat, utilizando a biblioteca chatterbot corpus para o treinamento em português. Utilizando também o NLTK que é uma plataforma para a construção de programas Python para trabalhar com dados de linguagem humana, nesta aplicação utilizamos o pacote nltk.tokenize para dividir as strings em listas de substrings, divide o texto em espaços em branco e pontuação, por exemplo, os tokenizadores podem ser usados para encontrar as palavras e pontuação em uma string.

```
bot = ChatBot(**CHATTERBOT)
primeiro = 1
#classe Otto

class Otto(object):
    global bot
    read_only=True,
    fluxo = FLUXOS['BOAS VINDAS']
    nome_usuario = None
   #Função para o treinamento do chat
    def treinar(self, lista_treinamento):
        trainer = ListTrainer(bot)
        for c in lista_treinamento:
            print(c)
            trainer.train(c)
        trainer.train('chatterbot.corpus.Portuguese') # O corpus utiliza a biblioteca em portugues
     #Função para o treinamento do chat
    def zerar_e_treinar(self, lista_treinamento):
        trainer = ListTrainer(bot)
        for c in lista_treinamento:
            print(c)
            trainer.train(c)
        trainer.train('chatterbot.corpus.Portuguese')

    def perguntar(self, pergunta, tipo):
        global primeiro

        print ("O tipo é:")
        print (tipo)
```

(a)

```
    #Se for uma string, separar em palavras, transformar todos os caracteres em minúsculos e retirar
    # caracteres especiais, acentos e espaço em branco
    if type(pergunta) is str:
        frase = pergunta
        characters = "@#$%&*'><()"
        for x in range(len(characters)):
            frase = frase.replace(characters[x],"")
        #DIVIDINDO A FRASE EM PALAVRAS
        palavras = nltk.word_tokenize(frase)

    if pergunta == 'menu':
            self.fluxo = FLUXOS['MENU']
            primeiro = 1

    if tipo == 'menu':
        self.fluxo = FLUXOS['MENU']


    if tipo == 'inicio' or tipo == 'chat':
        self.fluxo = FLUXOS['AUTO']

    if tipo == 'audio':
        microfone = sr.Recognizer()
        pa = pyaudio.PyAudio()
        n = "Você não tem microfone disponível. Por favor, digite o que procura ou navegue pelo menu inicial."
        if  pa.get_device_count() ==9 or pa.get_device_count()==0:
            return (n)
```

(b) Utilizando o pacote nltk.tokenize

Figura 10. Classe Otto utilizando Processamento de Linguagem natural(PNL).

6. Conclusão

Neste trabalho foi realizada a elicitação de requisitos junto ao cliente, levantamento para identificar quais requisitos são necessários para um chatbot funcionar corretamente e quais as limitações. Foi pensanda e desenvolvida toda a documentação arquitetural, chegando à conclusão de que uma das maiores dificuldades foi decidir quais algoritmos de PNL utilizar para o melhor funcionamento do chatbot, salientando que a língua portuguesa possui muitas variantes, com palavras específicas, erros de digitação entre outros.

Partindo destes resultados, possíveis trabalhos futuros precisam focar na melhoria dochatbot, desenvolvendo os requisitos que não puderam ser implementados, por exemplo a pesquisa de satisfação e o armazenamento das informações digitadas pelo usuário-eleitor, que será de grande valia para o treinamento de Otto.

Referências

Araujo E.(2020). Solução chatbot no ambiente academico da UFTJ.Universidade Federal do Rio de Janeiro.

Barbosa,J. and Vieira, J. and Santos, R. (2017). Introdução ao Processamento de Linguagem Natural usando Python. III Escola Regional de Informática do Piauí. Livro Anais - Artigos e Minicursos, v. 1, n. 1, p. 336-360, www.eripi.com.br/2017 - ISBN: 978-85-7669-395-6.

Crivelli, R.(2011). Recuperação da informação por meio de processamento de linguagem natural. Universidade Estadual do Norte do Paraná.

Cunha, D. and Silva, L. and Moura, R. (2010). Um chatbot para atendimento a clientesde farmácias. Departamento de Computação – Universidade Federal do Piauí (UFPI).
Teresina – Piau´ı – Brasil.

Gonzalez, M. and Lima, V. (2011). Recuperação de Informação e Processamento da Linguagem Natural.Pontifícia Universidade Católica do Rio Grande do Sul- PUCRS - Faculdade de Informática.

Júnior C. and Carvalho K. (2018). Chatbot: uma visão geral sobre aplicações inteligentes.Revista S´ıtio Novo – vol. 2, n. 2: jul./dez. 2018 - ISSN 2594-7036

Manfio, E. (2014). Processamento de Linguagem Natural, Robôs de Conversação e Linguística. Faculdade de Tecnologia de Garça/UEL.

Neuhauser, L. Kreps, G.(2014). Integração da Teoria da Ciência do Projeto e Métodos para Melhorar o Desenvolvimento e Avaliação de Programas de Comunicação em Saúde.v. 19 Journal of health communication

Rebecchi, A. (2020). Robô humano: Estudo sobre humanização no atendimento com chatbot.Dissertação, Mestrado Profissional em Comportamento do Consumidor.São Paulo.

Silva, T.(2021).Desenvolvimento de um sistema do tipo Chatbot para o curso deSistemas de Informação. Universidade Federal de Santa Catarina.

David Silva Santos
Universidade Federal da Bahia - Instituto de Computação
Departamento de Ciência da Computação
Salvador, BA – Brasil
santos.david@ufba.br

Visão Arquitetural e Software: A utilização do Spring Framework como ferramenta de interoperabilidade para o SGRH no desenvolvimento da API-REST do TRE-BA.

Abstract. *This paper reports the planning and development of the TRE-BA API REST, which aims to provide interoperability between the existing systems and the SGRH database in the Court. Will be exposed here from the previous situation, until the final solution through Spring Framework technology.*

Resumo. *Este artigo relata o planejamento e desenvolvimento da API-REST do TRE-BA que teve como objetivo fornecer a interoperabilidade entre os sistemas existentes e a base de dados do SGRH no Tribunal. Serão expostos aqui desde a situação anterior, até a solução final através da tecnologia Spring Framework.*

1. Introdução

A problemática surgiu a partir da necessidade de uma solução que viesse a tornar a consulta ao SGRH[Appio et al. 2008] mais dinâmica e de fácil mantenabilidade. O SGRH trata-se de uma base de dados robusta com informações sobre os servidores e o TRE-BA. Hoje no tribunal, vários sistemas e módulos acessam diretamente as tabelas do SGRH, além de congestionar a base de dados por múltiplos acessos simultâneos, como também deixa brechas a dados sensíveis do tribunal.

Para cada novo sistema, as consultas eram recriadas ou recicladas. Assim, as manutenções realizadas exigiam um caráter de alta complexidade. Qualquer alteração na estrutura de dados do sistema fonte, gerava a necessidade de manutenção em todos os sistemas que todavia acessariam aquelas informações.

Após algumas reuniões com o cliente, foi estabelecida a decisão para criação de uma API-REST, uma interface de programação para aplicações que estão em conformidade com as restrições do padrão arquitetural REST, acrônimo para *Representational State Transfer*. Consiste em uma camada de abstração que fornece a comunicação necessária para que os os sistemas do tribunal interajam com a base de dados do SGRH via JSON. A API também otimizaria a criação de novas soluções de software através dos métodos previamente criados reduzindo assim seu tempo de desenvolvimento e esforço demandado.

Ao longo dos encontros, foram discutidas as tecnologias aplicáveis à solução proposta. Entre as opções de mercado, o Spring Framework [Hunter 2020] foi o mais cotado devido ao fato de ser uma tecnologia já utilizada pelos desenvolvedores do TRE tendo assim o domınio necessário para futuramente os servidores assumirem o projeto, criar outras soluções ou mesmo realizar a manutenção da API.

O Spring framework tem o objetivo de facilitar o desenvolvimento de aplicações utilizando o isolamento das camadas através da injeção de dependência do IoC Container (inversão de controle). Dessa forma, ao adotá-lo no contexto do projeto a ser desenvolvido no tribunal, estará a disposição uma tecnologia que fornece recursos necessários à grande parte das aplicações, como módulos para persistência de dados, integração, segurança, testes, desenvolvimento web assim como um conceito a seguir que permite criar soluções menos acopladas e consequentemente mais fáceis de compreender e manter.

2. Referencial Teórico

Primeiramente nesta seção será abordada toda carga bibliográfica utilizada para embasar este artigo. A partir da solução escolhida, foi realizada uma pesquisa exploratória com a finalidade de levantar informações necessárias para aumentar o campo de conhecimento através de artigos e materiais que abordassem os seguintes descritores: *api*, *arquitetura rest*, *json*, *spring framework*, *sgrh*.

Para o desenvolvimento da solução de software foi utilizado a linguagem de programação Java e o framework Spring em conjunto ao ambiente de desenvolvimento integrado(IDE) [Hat 2019], Eclipse, que permitiu a construção, compilação e publicação da aplicação. Fora utilizado também a base de dados do SGRH através do software SQL Server, que permite a consulta por parte dos métodos da API. Abaixo serão descritos os conceitos de forma aprofundada.

2.1. API

Uma API [Hat 2017] é determinada através de um conjunto de definições e protocolos usados no desenvolvimento e na integração de software. Acrônimo que define *Application Programming Interface*, ela permite que uma solução ou produto se comunique com outros sistemas independente de suas tecnologias, dessa forma criando uma camada de abstração que visa a flexibilidade na criação ou manutenção de produtos, assim como um maior seguranca e controle que podem ser definidos através de atribuições de responsabilidades.

2.2. Arquitetura REST

Com o aumento da popularidade das API's alguns protocolos foram criados com o passar do tempo. Um deles foi o estilo arquitetural REST [Richards 2006], acrônimo de *Representational State Transfer*, se consolidou no mercado como melhor alternativa, o padrão obedece as constraints que determinam como padrões HTTP e URI deveriam ser modelados para que se possa aproveitar ao máximo os recursos oferecidos. Essas constraints são:

• **Cliente-Servidor:** Diz respeito a separar responsabilidade de cada parte do sistema, ou seja, camada de interação com o usuário separada do *backend*, garantindo a escalabilidade do sistema.

• **Stateless**: Para qualquer que seja a requisição, todas as informações necessárias já devem estar contidas na requisição para que ela possa ser processada pelo servidor sem que este se aproveite de informações adicionais sobre o contexto da comunicação.

• **Cache:** A aplicação precisa ser passável de cache, permitindo que o servidor processe apenas tarefas realmente necessárias.

• **Interface:** Uniforme onde deve ser mantido um padrão em termos de recursos, mensagens e corretude nos códigos de erros retornados.

• **Sistema em camadas:** Permite a escalabilidade do software para que seja possível acrescentar elementos intermediários de forma transparente para os clientes. Código sob demanda prega a possibilidade de adaptação do cliente de acordo com novos requisitos e funcionalidades.

2.3. JSON

Abreviação de *JavaScript Object Notation*, é uma forma rápida e compacta de comunicação entre sistemas. Sua maior vantagem é permitir a interoperabilidade de forma independente das linguagens do servidor ou cliente.

Dada a natureza das informações presentes no SGRH, pretende-se utilizar o processamento das informações no servidor e enviar a resposta a aplicação-cliente em formato JSON. O modelo de processamento permite que toda informação sobre a natureza dos dados seja de estrito conhecimento do servidor, simplificando a implementação no lado do cliente. A comunicação inter-aplicações consumidoras da API será realizada através do protocolo HTTP que é o principal protocolo de comunicação para sistemas Web, através de requisições GET efetuadas pelo cliente e processadas na API através dos módulos do Spring Framework.

2.4. Spring Framework

É um framework desenvolvido para a plataforma Java baseado nos padrões de projetos, inversão de controle e injeção de dependência. É constituído por diversos e completos módulos capazes de dar aumentar a performance na aplicação Java.

2.4.1. Características

As principais características podem ser detalhadas logo abaixo:

Design Patterns: São definições de alto nível de como um problema comum pode ser solucionado. Por alguns problemas serem recorrentes, soluções generalistas acabam sendo criadas com o objetivo de representar um bom ganho de produtividade para os desenvolvedores. Contribuindo também para a organização e manutenção de projetos, já que esses padrões se baseiam em baixo acoplamento entre as classes e padronização do código.

Inversão de controle: As instâncias das classes são fracamente acopladas, ou seja, a interdependência entre os objetos é mínima. Assim a interrupção do fluxo de execução de um código pode ser realizada, a fim de transferir o controle para uma dependência ou container.

Injeção de dependências: Com o propósito de evitar o acoplamento de código em uma aplicação a injeção de dependência gera instâncias de classes que um objeto precisa sem que este instancie por si mesmo.

2.4.2. Ecossistema

Todo o ecossistema Spring favorece o projeto e se enquadra nos requisitos para criação da API, abaixo alguns de seus produtos:

Spring Boot: É uma ferramenta que visa facilitar o processo de aplicações que utilizem o ecossistema Spring. Através de um template pré-configurado para desenvolvimento e execução de aplicações baseadas no Spring é possível selecionar módulos que se adequem ao tipo de produto que você deseja criar.

Spring Data JPA: Framework que visa facilitar a criação dos repositórios. O Spring Data JPA, torna desnecessário a implementação de interfaces pois através do JPA Repository é disponibilizado uma interface com vários métodos já predefinidos e prontos para uso.

Spring Security: Estrutura de autenticação e autorização, o Spring Security conta com uma gama altamente personalizável para a proteção de aplicações baseada no Spring Framework.

Swagger: Ferramenta de especificação aberta para API's, o Swagger consegue ler a estrutura do seu código fonte e gerar automaticamente uma documentação que descreve informações como:

• Lista de todas as operações que a API suporta.

• Quais são os parâmetros dos métodos e o que eles retornam.

• Se a API precisa de alguma autorização.

2.4.3. Módulos

Com o objetivo de manter uma arquitetura limpa e de fácil mantenabilidade, a divisão de responsabilidades da API-REST foi definida em módulos. Realizando assim a decomposição de um sistema complexo em camadas como: *Controller*, *Service*, *Model*, *Repository*. Abaixo serão detalhados cada módulo:

• **Controller:** Responsável por gerenciar as requisições e fazer a comunicação com a regra de negócios, e por fim retornar a resposta da requisição a aplicação-cliente em formato JSON.

• **Service:** São chamadas as regras de negócios, aqui fica estabelecido o armazenamento de toda a lógica referente ao sistema.

• **Model:** Neste módulo, ficará armazenado todas as entidades, que são as representações das tabelas da base dados do SGRH.

• **Repository:** Onde se mantêm as consultas relativas a base de dados, através do JPA Repository é possível obter uma gama de métodos predefinidos através da injeção de dependências.

2.5. Insomnia

O insomnia é um software de testes para API's, que permite a validação dos métodos através das requisições HTTP, tendo seu resultado um código de status HTTP [G. 2021] e o retorno da requisição em formato JSON. No contexto do TRE foi amplamente utilizado para testar os métodos da API, forçar e validar seus erros, assim como testar a segurança através das autenticação via token.

3. Trabalhos Relacionados

Em contato com o TSE, pode-se identificar o mesmo problema referente aos acessos diretos a base de dados SGRH, e como solução, uma API-REST também está sendo desenvolvida para otimização dos acessos à base por parte das aplicações existentes no tribunal.

4. Proposta do projeto

A API REST do SGRH/TRE-BA se propõe a ser um sistema capaz de integrar as diversas aplicações que atualmente realizam consultas a base de dados do SGRH e servir de padrão para as consultas realizadas por aplicações futuras, de forma a funcionar como uma camada de abstração para o acesso da lógica de negócio das aplicações e aos dados sensíveis presentes no banco de dados. Abaixo será elencado as visões arquiteturais do projeto.

4.1. Visão macro do fluxo de funcionamento da API

O fluxo de funcionamento da API deve começar primeiramente através da requisição da aplicação-cliente, onde será interpretada e processada pela API que dará seguimento a consulta a base dados, só assim a API retorna a ao cliente um estado representacional dos dados da consulta em formato JSON. [1]

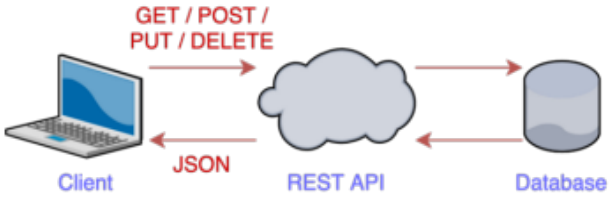


Figura 1. Etapas do processo de funcionamento da API.

4.2. Visão do fluxo de funcionamento interno da API

Já o fluxo de funcionamento interno da API é detalhado após a requisição do cliente. A primeira etapa do processamento da requisição transita no *Controller* onde é utilizado para se comunicar com a regra de negócio, enquanto isso o *JPA Repository* fornece alguns metodos CRUD (Create, Read, Update e Delete) predefinidos através da injeção dependências para a regra de negócios, e enfim se comunicar com o Model, que atua como entidade representacional das tabelas da base de dados do SGRH que ao retornar para o controller, devolverá a requisição em formato JSON a aplicação-cliente. [2]

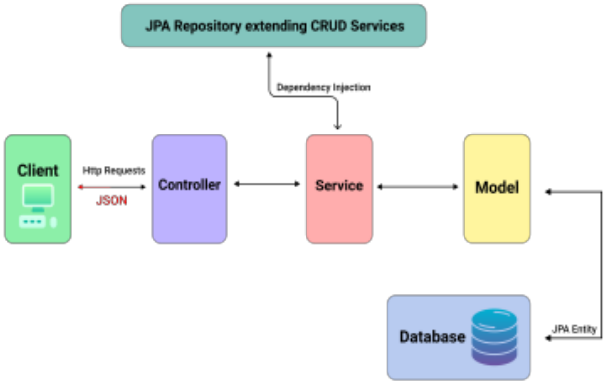


Figura 2. Fluxo arquitetural do Spring Boot

5. Avaliação

Para poder validar os métodos desenvolvidos para a API, foi utilizado o *Insomnia*, um software de testes de requisição que pode ser utilizado localmente.

A primeira etapa da validação consiste na autenticação. Ao passar o título e a senha de usuário no corpo da requisição, a API irá processar a requisição POST, retornando um status 200 caso tenha sucesso, ou 400 em caso de dados inválidos. Em caso de sucesso, o usuário recebe um token que mantém a conexão aberta para que sejam efetuadas as futuras requisições. [3]

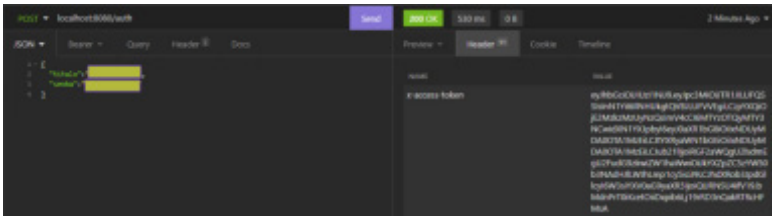


Figura 3. Etapa de autenticação da API-REST, retornando status 200 confirmando a autenticação.

Depois de autenticado, foi possível testar alguns dos métodos desenvolvidos até então, abaixo na demonstração foi testado o método "servidor", onde o número do título eleitoral do usuário é passado como parâmetro na requisição. O status 200 aparecerá se bem sucedido e será retornado os dados do servidor em formato JSON. [4]

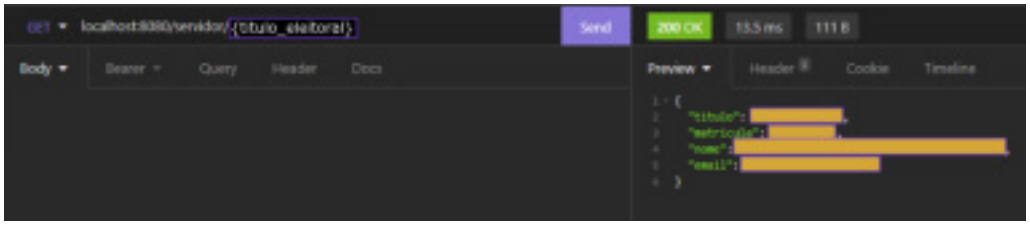


Figura 4. Método GET com objetivo de retornar os dados do servidor. ´

Outro método testado foi “servidorAfastamento”, método que através do número do CPF do usuário na requisição, é possível consultar todos os afastamentos referente àquele servidor. [5]

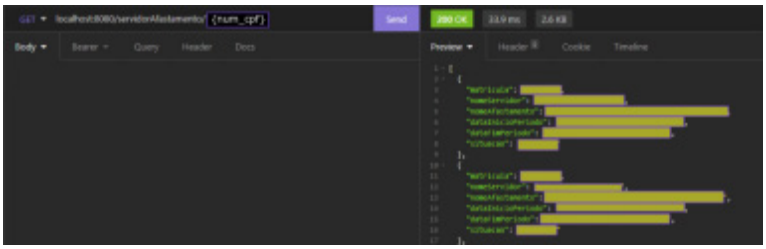


Figura 5. Método GET com objetivo de retornar os afastamento do servidor.

5.1. Resultados e Discussão

Após a validação da autenticação e teste dos métodos através do Insomnia, é possível concluir que a criação da API-REST para o TRE-BA teve êxito. Com isso, no futuro os sistemas irão deixar de acessar diretamente a base de dados e passarão a consumir a API, tornando o desempenho dos sistemas otimizado, seguro e de fácil mantenabilidade. Todavia, é importante ressaltar que a migração dos sistemas existentes no tribunal para consumir a API, deverá ocorrer de forma paulatina e gradual de forma que não comprometa o funcionamento ou integridade do sistema.

6. Conclusão (Considerações Finais)

Portanto, foi possível observar que a implementação de novas soluções de software trarão benefícios as esferas do judiciário, e que as etapas de planejamento e desenvolvimento devem ser pensadas com seriedade e de forma estratégica para que no futuro sejam colhidos frutos de uma boa relação organizacional no que diz respeito a evolução dos sistemas existentes, para que os esforços empenhados hoje, não sejam comprometidos no futuro.

6.1. Ameaças

Com o crescimento da API alguns pontos exigem atenção. A má divisão de responsabilidades entre os módulos assim como o crescimento mal planejado pode tornar o código verboso e confuso, prejudicando o entendimento e a manutenção do sistema com o passar do tempo, comprometendo assim os ganhos de outrora.

6.2. Trabalhos Futuros

Será realizada a migração gradual dos sistemas que consomem de forma direta a base de dados do SGRH para que passem a consumir a API-REST, assim como a elaboração e˜ implementação de novos métodos de consulta.

Referências

Appio, J., de Souza Domingues, M. J. C., and Klaesener, D. (2008). Tributos relevantes na compra do software sgrh–sistema de gestão de recursos humanos: Um estudo exploratório. *Synergism usscyentifica UTFPR*.

G., A. (2021). O que é http, http error e os outros códigos de erro.

Hat, R. (2017). O que é api?

Hat, R. (2019). Ide - ambiente de desenvolvimento integrado.

Hunter, G. (2020). Spring framework: o que é, seus módulos e exemplos.

Richards, R. (2006). Representational state transfer (rest). In *Pro PHP XML and web services*, pages 633–672. Springer.

João Eudes Santos Guedes
Tribunal Regional Eleitoral da Bahia
1ª Av. do Centro Administrativo da Bahia, 150 – CAB
Salvador-BA - CEP: 41.745-901 - Brasil
jguedes@tre-ba.jus.br

Avaliação do uso da linguagem GraphQL através do protótipo de uma aplicação no TRE-BA.

Resumo. *Este artigo apresenta um modelo de API GraphQL para utilização em sistema a ser desenvolvido no TRE-BA como parte do trabalho de conclusão da especialização em tecnologia da informação, residência TRE x UFBA.*

1. Introdução

1.1. Contexto

A Justiça Eleitoral, ramo especializado do Poder Judiciário, tem atuação em três esferas: Administrativa: organiza e realiza eleições, referendos e plebiscitos, além de ser responsável por todo o cadastro eleitoral; Regulamentar: regula e normatiza o processo eleitoral e Jurisdicional: julga questões eleitorais. É composta pelo Tribunal Superior Eleitoral – TSE com sede em Brasília; por um Tribunal Regional Eleitoral nas capitais estaduais; Pelos Juízes Eleitorais e pela Juntas Eleitorais [Justiça Eleitoral 2021].

Na esfera administrativa, o Tribunal Regional Eleitoral da Bahia - TRE-BA possui uma Seção de Desenvolvimento de Sistemas Corporativos - SEDESC subordinada à Coordenação de Soluções Corporativas e Infraestrutura - COSINF da Secretaria de Tecnologia da Informação - STI. A principal atribuição dessa unidade é desenvolver sistemas que atendam às necessidades do tribunal. Além disso, adapta e mantém sistemas de outros tribunais que sirvam à gestão do órgão. O TRE-BA comunica-se através de rede de dados com outras instâncias da justiça eleitoral, a exemplo do TSE, e das Zonas Eleitorais - ZEs da capital e do interior do estado. Todos esses órgãos utilizam aplicações que funcionam de forma distribuída através de Intranet corporativa, Extranet e em alguns casos na Internet. Estes sistemas são cada vez mais demandados e para atender às solicitações de novos sistemas, a SEDESC nesse artigo será avaliada a viabilidade de desenvolver um protótipo de um componente para uma aplicação solicitada pela Secretaria de Gestão de Pessoas – SGP do TRE-BA. Com isso poderemos avaliar o uso da tecnologia GraphQL para a implementação de interfaces de comunicação entre sistemas no âmbito desse tribunal.

1.2. Motivação

O GraphQL é uma tecnologia que vem ganhando participação entre aquelas mais utilizadas para desenvolver interfaces de comunicação entre sistemas. E, embora não seja a única, e nem mesmo, a mais utilizada para tal finalidade, apresenta características que justificam a sua crescente adoção

para o desenvolvimento de soluções dessa natureza. Entre esses atributos estão as estruturas que favorecem a padronização e a facilidade de manutenção dos sistemas baseados nessa tecnologia, e ainda, a capacidade de criar interfaces integradas a partir de diferentes fontes de dados. Ao permitir a integração com outras interfaces baseadas em tecnologias como a arquitetura REST, predominante na maioria das APIs, e ainda conectar-se a sistemas de bancos mais utilizados no mercado, ela oferece flexibilidade e maior interoperabilidade entre os sistemas legados com os sis temas baseados no ambiente GraphQL. Destaque-se ainda a documentação intrínseca e obrigatória ,obtida durante a construção das APIs, Desenvolver um projeto piloto é oportunidade para aplicar na prática a base teórica aqui abordada. A aplicação escolhida foi a do Servidor e Equipe Nota 10, um sistema já priorizado pela administração do TRE-BA e que tem o objetivo de facilitar o registro e divulgação de boas práticas implementadas nesse tribunal, além de permitir aos servidores e colaboradores do órgão escolher as melhores práticas através de voto e ao final da apuração dos conhecer as práticas vencedoras e premiar os servidores, colaboradores e equipes responsáveis pela formulação das mesmas.

Os valores presentes no Planejamento de Tecnologia da Informação – PETI do TRE-BA formulado para o período de 2016-2021 nortearam o desenvolvimento de uma aplicação que almeja apoiar um importante processo organizacional do órgão: a valorização da sua força de trabalho. Entre os atributos de valor aqui referenciados estão a "Inovação: estímulo à criatividade e à busca de soluções inovadoras" e o "Alinhamento: estar em consonância com as necessidades do negócio, antecipando cenários, provendo soluções e participando da tomada de decisões no que se refere à provisão de serviços à sociedade.".[BRASIL, Tribunal Regional Eleitoral da Bahia 2018]

1.3. Objetivo

O objetivo geral desse artigo é conhecer e avaliar o GraphQL afim de criar um protótipo de um modelo de API para o Sistema para realização da Campanha Servidor e Equipe Nota 10, conhecer as tecnologias alternativas de uso corrente no TRE-BA e observar como tais ferramentas podem interagir para a criação dessas interfaces de comunicação e se elas podem ser integradas para benefício comum delas

1.4. Objetivo específicos

Compreender alguns conceitos básicos de API e API Webs. Entender o conceito central do REST. Conhecer GraphQL e avaliar as características que interessam ao desenvolvimento da API abordada neste artigo. Identificar algumas entidades que serão modeladas quando da elaboração do protótipo. Posteriormente, planejar a instalação dos serviços necessários ao funcionamento do sistema.

2. Referencial Teórico

2.1. As APIs Web em geral

Uma interface de programação de aplicações acrônimo em inglês para "Application Program Interface - API" é um conjunto de conjuntos de definições e protocolos usados no desenvolvimento e na integração de aplicações de software. As APIs Web são API remotas projetadas para integração de através e também são identificadas como *Web Services* ou em português: Serviços Web. O HTTP é o protocolo mais solicitado para o intercâmbio de mensagens preferencialmente utilizando a linguagem de marcação extensível, em inglês eXtensive Markup Language, XML ou ainda através da Notação de Objeto Javascript ou como é mais conhecido em inglês Javascript Object Notation, JSON. Os esforços para simplificar as API são contínuos e objetivam incrementar a flexibilidade necessária para simplificar o design, a administração e o uso destas, além de fornecer oportunidades de inovação. (Red Hat,2021).

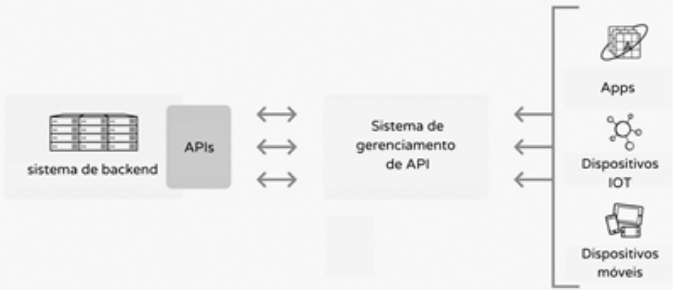


Figura 1. Uma API Web. [RED HAT, 2017]

As APIs representam os meios pelos quais os interessados podem se conectar a uma determinada organização. No caso do TRE-BA, as APIs desenvolvidas devem fornecer integração entre as aplicações que de alguma são utilizadas por interessados como os juízes, advogados, eleitores, candidatos, partidos, servidores do próprio tribunal, órgãos da justiça eleitoral, servidores requisitados de outros órgãos, colaboradores de empresas contratadas e outros. Mesmo quando bem especificadas e planejadas para atender da melhor forma as aplicações clientes, as API estarão suscetíveis a mudanças corretivas e evolutivas devendo ser, portanto flexíveis o suficiente. [Lauret,2019] pondera que o projeto de uma API não deve considerar apenas a interação entre sistemas que ela irá prover, mas, além disso deve oferecer funcionalidades uma experiência de desenvolvimento satisfatória aos desenvolvedores dessas interfaces.

Existem diversas abordagens que podem aplicadas ao projeto de APIs mas uma certamente considera o papel dos interessados na construção da API, ela é conhecida como design-first. Nessa abordagem procura-se identificar, inicialmente, quais requisitos demandados pelo cliente da API deverão ser atendidos, e através de um processo interativo que inclui a participação dos interessados, realizar uma verificação periódica da evolução do desenvolvimento dessa interface. Todo esse processo antecede a sua entrada em produção. Pois para [HIGGINBOTHAM,2021] uma vez em produção torna-se difícil migrar as aplicações consumidoras de uma API para uma nova versão. Caso exista alguma aplicação cliente funcionando em produção, isso já será suficiente para comprometer a integração da API com outros componentes de sistema. Uma abordagem design-first é composta de cinco fases que são executadas rapidamente e de forma iterativa conforme pode ser visto na figura 2.

1. Descoberta: Estabelece os requisitos que a API precisa entregar e verifica se tais requisitos já estão presentes em outras APIs disponíveis na organização.

2. Projeto: Produz um projeto inicial, ou aperfeiçoa um já existente, com o intuito de prover os requisitos necessários, mas ainda não disponíveis.

3. Prototipagem : Produz um protótipo da API afim de obter parecer dos interessados acerca do estágio atual do projeto, realizando ajuste com base nessa informação.

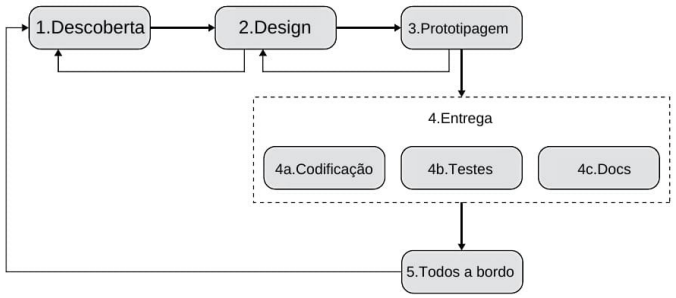


Figura 2. Modelo de desenvolvimento de APIs design-first. [HIGGIBOTHAM, 2021]

4. Entrega: Entrega o projeto através de um esforço conjunto de desenvolvedores, pessoal de certificação de qualidade de software e time de infraestrutura de tecnologia da informação. As entregas são parciais baseadas em acordos estabelecidos na fase projeto.

5. Todos a bordo: Fase que promove a participação de desenvolvedores e outros interessados para validar a integração entre a API e as aplicações clientes.

2.2. As APIs Web - REST

O "Representational State Transfer, REST" cujo significado é Transferência Representacional de Estado é um modelo de arquitetura, e não uma linguagem ou tecnologia de programação. Ele fornece diretrizes para que os sistemas distribuídos se comuniquem diretamente usando os princípios e protocolos existentes da Web sem a necessidade de outro protocolo mais sofisticado [HIGGIBOTHAM 2021]. Atualmente o REST é o recurso mais utilizado para a criação de API Web. Esse tipo de API Web define um conjunto de restrições utilizados para a criação de web services baseados em *endpoints* associados a recursos dos sistema e responsáveis pela a comunicação entre o provedor e consumidor do serviço web.

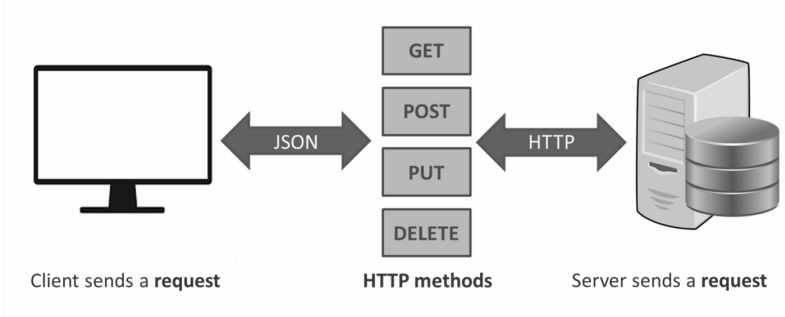


Figura 3. Uma Api REST.[POUDEL 2019]

Uma questão muito relevante quando utilizamos APIs REST é que as aplicações clientes, consumidoras desse tipo de API, necessitam agrupar informações isoladas e com origem em diferentes recursos por meio de chamadas recorrentes. Essas chamadas múltiplas amplificam o consumo de memória, rede e processamento e isso é particularmente crítico quando a aplicação funciona em dispositivos móveis com maior limitação desses recursos. Esse tipo de API também não dispõe de uma linguagem de consulta que lhe permita comunicar suas necessidades com exatidão. Outro problema importante que ocorre com o REST é o versionamento da API quando há necessidade de manter disponíveis diferentes versões de um mesmo recurso, o que pode ocasionar duplicidade de código.[BUNA,2021] Um ponto positivo do REST é o armazenamento de cache automático implementado nativamente pelo protocolo HTTP e favorecido pelo conceito de usar um recurso para cada endpoint. O REST utiliza os métodos do HTTP conforme a tabela 1:

Tabela 1. Métodos HTTP utilizados na arquitetura REST

Método HTTP	Descrição
OPTIONS	Retorna os verbos http de um resource e outras opções. GET Busca um recurso
HEAD	Busca um cabeçalho de um recurso PUT Atualiza um recurso
POST	Cria um recurso
DELETE	Exclui um recurso
PATCH	Atualiza parcialmente um recurso

2.3. As APIs Web - GraphQL

O GraphQL é uma especificação para um ambiente de execução e de uma linguagem declarativa, a "Schema Definition Language - SDL" cujo significado em português é linguagem de definição de esquemas. Ela é capaz de retornar os dados no mesmo formato demandado pela camada de visão, é independente de frameworks e apta a conectar-se a qualquer Sistema Gerenciador de Banco de Dados SGBD. Essa linguagem não deve ser confundida com a linguagem de consulta estruturada presentes nestes sistemas conhecidas em inglês como "Structured Query Language – SQL". Os serviços que implementam a SDL podem ser usados para a criação de APIs Web de forma a realizar consultas e atualizar informações através de diversos tipos de fontes de dados como SGBD, microsserviços e sistemas legados. [CHECINSKY et al, 2020].

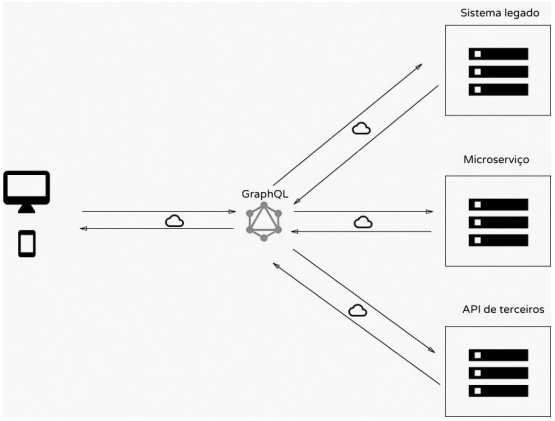


Figura 4. Integração de sistemas e APIs existentes via GraphQL[BUNA 2021]

A linguagem GraphQL , criada pelo Facebook em 2012 e disponibilizada em código aberto a partir de 2015 e foi projetada para ser declarativa, flexível, eficiente e ofertar dados em um formato similar aos esperados pelos sistemas consumidores de uma API nela baseada. Um backend GraphQL requer um ambiente de execução que provê meios para descrever os dados que serão expostos em suas APIs. Nessa estrutura conhecida por esquema (schema) tais APIs utilizam a linguagem por ela suportada para construir a representação exata dos dados requisitados[BUNA 2021]. Ter um ambiente de execução para a camada de modelo de um sistema baseado em uma linguagem como a GraphQL permite fluidez a mudanças necessárias à criação e alteração desse modelo ao longo do ciclo de vida de uma aplicação.

Ao projetarmos APIs REST para atender aplicações *frontend* específicas esse projeto deve estar sempre compatível com as necessidades da interface de usuário, mantendo se a camada de modelo compatível com a camada de visão, para não afetar o trabalho do desenvolvedor, tornando mais oneroso e, ou, também prejudicar o desempenho da aplicação. Nesse caso, criar um *endpoint* personalizado para cada cliente pode melhorar a performance por meio de uma ajuste fino entre o *frontend* e *backend*. Por outro lado, pode aumentar o número de *endpoints* a manter e criar diversas versões desses serviços.

Outros dois desafios que temos ao lidar com a tecnologia REST são: Underfetching: Ocorre quando os dados requisitados só são obtidos através de várias requisições a diferentes *endpoints*. e Overfetching: Quando uma requisição retorna dados que não serão utilizados integralmente pelo *frontend*. Um exemplo de overfetching: suponhamos que uma loja hipotética queira gerar uma lista contendo apenas o nome de seus clientes. Uma API REST consultaria um *endpoint* de clientes que retornaria além do nome, outros dados, como data de aniversário e endereço. Nesse caso as demais informações seriam inúteis, uma vez que a lista requer apenas a informação do nome do cliente.

Em relação ao underfetching considere-se o seguinte: O gerente da loja hipotética acima quer uma lista contendo seu clientes e as compras realizadas nos últimos três meses. Assim, conside-

rando dois *endpoints* teríamos: clientes e compras de cada cliente por mês e que tais informações estariam armazenadas em um esquema devidamente normalizado em um Sistema Gerenciador de Bancos de Relacional – SGBDR, nesta configuração, mais um *endpoint*, o de produtos seria necessário para que a camada de Visão pudesse acessar todos os dados requisitados. Nos casos acima, o GrapQL consegue lidar bem com esses desafios pois não precisa de vários *endpoints* específicos para as funcionalidades da camada Visão e requer através de sua linguagem de consulta exatamente o solicitado pelo *frontend*.

2.4. Arquitetura GraphQL - processo de envio e resposta

Conforme pode ser visto na figura 4, em uma arquitetura típica do GraphQL o cliente envia solicitações HTTP com queries, mutations ou subscriptions, e o GraphQL Server processa as solicitações usando Resolvers. O resolver obtem os dados de um banco de dados ou de uma API Rest. O servidor GraphQL retorna o resultado ao Client GraphQL,normalmente em um formato JSON.

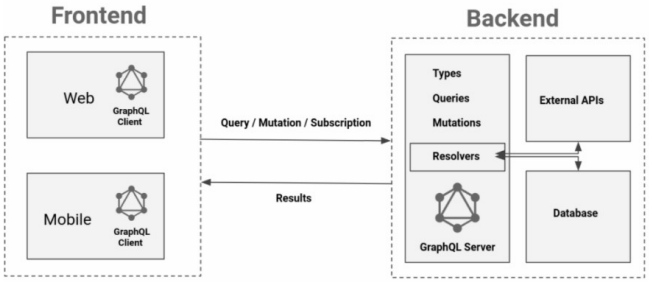


Figura 5. Arquitetura GraphQL.[PRAGMATICREVIEWS 2020]

2.5. Documento GraphQL

Uma solicitação de dados ao servidor GraphQL é realizada através de uma requisição na qual estão inseridas, em um documento de texto, operações da linguagem GraphQL. Essa requisição é o elemento fundamental para a comunicação da API. Um dos tipos presentes nessa linguagem é o *Query* ou consulta, que descreve uma operação de consulta cujo resultado é um texto em formato JSON contendo os dados solicitados, ou uma informação de erro, se for o caso. Outro tipo é o *Mutation*, similar ao tipo Query, ele até retorna dados, mas com uma diferença: através dele podem ser realizadas operações que alteram, inserem,excluem ou atualizam dados. Há ainda o tipo Subscription, utilizado quando aaplicação cliente requer, à API, informação, em tempo real, sobre a atualização de dados do seu interesse. Os tipos citados são fundamentais à linguagem. Um documento pode conter ainda Fragments que representam conjuntos de informação reutilizáveis em uma Query, por exemplo. [PORCELLO and BANKS 2018]

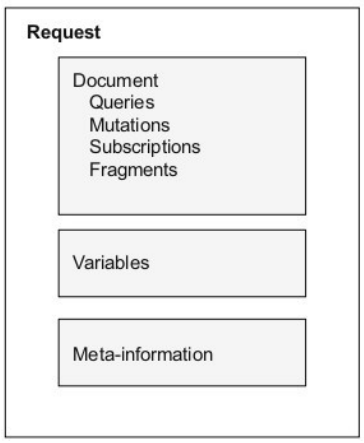


Figura 6. Estrutura de uma requisição GraphQL[BUNA 2021]

2.6. Proposta de criação de API GraphQL para o sistema Servidor Nota 10

O protótipo da API baseada em GraphQL atenderá ao Sistema para realização da Campanha Servidor e Equipe Nota 10 cuja demanda foi feita pela Seção de Desenvolvimento Organizacional - SEDES e autorizada pelo Comitê Gestor de Tecnologia da Informação

CGTI do TRE-BA. Esse sistema será referenciado a partir daqui como Sistema Servidor Nota 10. Trata-se de uma aplicação de pequeno porte, relativamente simples e que não possui muitas funcionalidades, sendo estas algumas das razões para a sua escolha. Note-se, porém, que dadas as limitações desse trabalho, apenas algumas de suas funcionalidades podem ser modeladas utilizando a linguagem GraphQL e nessa fase não teremos informações gravadas em sistemas de banco de dados.

O Sistema Servidor Nota 10 foi solicitado através de Documento de Descrição de Demanda de Solução de TI – DDD. Seu objetivo é cadastrar boas práticas implantadas por servidores ou equipes dentro dos critérios estabelecidos para participar de votação da melhor prática do ano. A campanha anual prescrita por este, está dividida em quatro ciclos:

- Inscrição da boa prática;
- Votação de boas práticas ou indicação de boa prática não inscrita;
- Validação dos votos
- Apuração dos resultados.

A abrangência envolve o TRE-BA e suas zonas da capital e do interior. Ele substituirá o processo atual de realização dessas campanhas que utiliza ferramentas gratuitas para elaboração de questionário como o Google Forms, já para as tarefas de validação/apuração do resultado o Excel da Microsoft é a solução alternativa utilizada. Estes instrumentos mostram-se, porém, precários para controlar, validar e publicar a apuração dos votos.

A API desenvolvida servirá como backend para clientes baseados em computadores de mesa ou dispositivo móveis. Dados relacionados à consulta de servidores e requisitados serão acessados através da API REST do SGRH, enquanto o registro e con- sulta acerca de outras informações do sistema serão realizados desta API GraphQL. O ambiente de execução do NodeJS foi o escolhido para a implementação da solução. Para os testes de execução da API será utilizada a ferramenta graphql-playground

3. Levantamento preliminar de requisitos

3.1. Requisitos funcionais

RF01 - O sistema deve permitir a criação de um novo ciclo de campanha;

RF02 - O sistema deve permitir a atualização das informações de um ciclo de campanha cujas fases ainda não tenham sido iniciadas ou encerradas;

RF03 – O sistema deve poder consultar as informações de servidores, terceirizados e requisitados;

RF04 – O sistema deve permitir o cadastramento de equipes compostas apenas por integrantes pertencentes a uma mesma coordenadoria ou correlatos (CJ1);

RF05 - O sistema deve permitir o cadastramento da comissão julgadora vinculada a um ciclo de campanha em aberto e com as fases de votação e apuração ainda não encerradas;

RF06 - O sistema deve permitir ao servidor ou equipe inscrever mais de uma boa prática por campanha;

RF07 - O sistema deve exibir no formulário de votação a frase criada na inscrição da boa pratica ;

RF08 - O sistema apenas permitir uma única votação por pessoa e o(s) voto(s deve(m) para cada categoria devem ser confidencial(ais);

RF09 - O sistema a permitirá a cada pessoa (eleitor da campanha) indicar uma boa prática por campanha;

RF10 - O sistema deve permitir ao usuário realizar sua inscrição pela Internet;

RF11 - O sistema deve permitir ao usuário votar na campanha através da Internet;

RF12 - O sistema deve permitir ao gestor do sistema habilitar a fase de apuração somente quando as demais fases anteriores forem encerradas;

RF13 - O Sistema deve permitir que os votos sejam contabilizados separadamentepor servidor, por equipe, por terceirizado e por servidor requisitado;

RF14 - O sistema deve permitir que o gestor informe o peso de cada critério(pergunta) constantes nos formulários de inscrição e votação.

4. Conclusão

Através da revisão da bibliografia foi possível adquirir conhecimentos teóricos sobre o GraphQL que serão importantes para aplicar durante a criação do protótipo da API do Sistema Servidor Nota10. Durante a verificação preliminar do Documento de Descrição de Demanda observou-se que maiores informações serão necessárias para a execução desse propósito e nesse momento não foi possível contar com a participação dos interessados no sistema. A adoção de novas tecnologias em qualquer organização requer tempo e maturação suficientes para que elas sejam devidamente aproveitadas. O ideal é que seja feito de forma gradual e considere como a inclusão de uma tecnologia irá impactar o ambiente organizacional. O uso de API Webs traz benefícios importantes para as organizações que as utilizam. O alinhamento das expectativas e possíveis problemas devem ser frequentemente revisados e a criação do protótipo deve caminhar se possível com o desenvolvimento convencional adotado no órgão. Um aspecto bastante positivo TRE-BA reconhece no seu planejamento estratégico de Tecnologia da Informação a importância de valores como a inovação e implementação de importantes a inovação e a implementação de novas ideias conforme pode ser verificado no seu planejamento estratégico de TI e no âmbito do Conselho Nacional de Justiça que considera inovação a implementação de ideias que geram valor para o Poder Judiciário através de novos produtos, serviços, processos de trabalho, ou uma maneira diferente e eficaz de solucionar problemas complexos encontrados no desenvolvimento das atividades que lhe são importantes.

Referências

BRASIL, Tribunal Regional Eleitoral da Bahia (2018).Institui a política de gestão da inovação no âmbito do poder judiciário. https: //www.tre-ba.jus.br/o-tre/planejamento-estrategico/planejamento-estrategico-de-ti. Acessado em 12-12-2021.

BUNA, S. (2021). *GraphQL in action.* Manning Publications Co, Shelter Island, NY.

HIGGIBOTHAM, J. (2021).*Principles of web API design: applying domain-drivendesign to apis and microservices.* Addison-Wesley, Boston, 1st edition.

Justiça Eleitoral (2021). A justiça eleitoral. https://www.justicaeleitoral. jus.br/a-justica-eleitoral.html. Acessado em 04-12-2021.

PORCELLO, E. and BANKS, A. (2018). *Learning GraphQL.* Sebastopol, CA, O'ReillyMedia, Inc.

POUDEL, S. (2019). Consuming rest apis with python. https://faun.pub/ consuming-rest-apis-with-python-eb86c6b724c5. Acessado em 12- 12-2021.

PRAGMATICREVIEWS (2020). What is graphql?learnit in 5 minutes. https://pragmaticreviews.com/what-is-graphql-learn-it-in-5-minutes/. Acessado em 12-12- 2021.

RED HAT (2017). O que e´ api? https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces. Acessado em 06-12-2021.

Gabriel C. Brito
Instituto de Computação – Universidade Federal da Bahia (UFBA) Salvador – BA – Brazil
gabriel.chaves90@gmail.com

TREVis: Uma Ferramenta de Visualização de Dados com Storytelling para o Contexto do Tribunal Regional Eleitoral da Bahia

Abstract. *Business Intelligence is increasingly using features such as visual dashboards (Dashboards) and data narratives (Storytelling) to improve the ex- traction of information from data. It is important that the data to be used is studied in order to offer a clean and intuitive panel for the user. In this work we present the development of the TREVis tool, which helps the user to obtain clear and fast information. At the end, the tool is presented in detail, together with the justification for the choices made for each visual element.*

Resumo. *Cada vez mais a Inteligência de negócios vem utilizando recursos como painéis visuais (do inglês, Dashboards) e narrativas dos dados (do inglês, Storytelling) para melhorar a extração de informação nos dados. É importante que os dados a serem utilizados sejam estudados de forma a oferecer um painel limpo e intuitivo para o usuário. Neste trabalho apresentamos o desenvolvimento da ferramenta TREVis, que auxilia o usuário a obter informações claras e rápidas. Ao final, a ferramenta é apresentada em detalhes, em conjunto com a justificativa das escolhas de cada elemento visual.*

1. Motivação

A utilizaç ao da Inteligência de Negócios, do inglês *Business Intelligence* (BI) torna possível tomar decisões estratégicas baseadas em dados estruturados e coletados através de diversas fontes diferentes. A maioria das soluções em BI oferecem uma boa análise e visualização de dados que, com a tecnologia de captura correta, devem ser capazes, inclusive, de tratar os dados em tempo real. Esta capacidade de conectar dados e construir análises em tempo real pode oferecer um bom suporte na apresentação de informações ao usuário final. Entretanto, para que isto seja possível, algumas áreas de conhecimento precisam ser consistentes e fortes, como a aptidão para estruturar e relacionar adequadamente os visuais e artefatos a serem exibidos em seus relatórios dinâmicos. Atualmente a literatura voltou-se aos conceitos de Visualização de Dados, *Storytelling* e Painéis Visuais. O primeiro trata de apresentar os dados por meio de representações visuais, enquanto o segundo abarca a melhor maneira para apresentar os dados por meio destes recursos visuais. *Storytelling* também pode ser compreendida como o conjunto *"Storytelling with Data"*. Este conceito refere-se ao conjunto de processos e mecanismos que auxiliam as organizações a preparar as informações com a finalidade de comunicar uma história, incluindo um arranjo de elementos: dados, narrativas e visualizações.

Já os painéis visuais, são ferramentas utilizadas para transmitir toda esta análise dos dados. Através deles, os dados que foram representados de forma visual e organizados em uma narrativa própria são estruturados e apresentados ao usuário final. Recentemente o Tribunal Regional Eleitoral da Bahia (TRE-BA) tem implementado sistemas de BI em seus setores de forma gradativa. Entretanto, alguns profissionais não levam em consideração a importância de uma base de dados, visualizações narrativas bem estruturadas, com isso obtém-se painéis confusos ou com dados que não refletem de forma adequada o seu propósito.

2. Objetivo

O principal objetivo é desenvolver painéis visuais inspirados em versões anteriores, de modo que, os conceitos de *Storytelling* possam ser usados através dos dados para que sejam exibidos de maneira clara, consistente e intuitiva. Ainda, faz-se necessário um comparativo entre as versões para verificar a importância da utilização bem estruturada do *Storytelling* na construção de painéis visuais utilizando as premissas de visualização de dados.

3. Referencial Teórico

O termo Business Intelligence (BI), desde a década de 1980. O Dr. Dan Power, fundador da *Association for Information Systems Special Interest Group on Decision Support, Knowledge and Data Management Systems (SIG DSS)* define sistemas de apoio à decisão (DSS), como sistemas ou subsistema baseado em um computador interativo. O propósito é ajudar na tomada de decisão e usar tecnologias de comunicação, dados, documentos, conhecimento e/ou modelos, para identificar e resolver problemas [Hedgebeth 2007]. O mais antigo estudo relevante de BI define Sistemas de BI de uma forma simples, como sendo a recuperação ao automática de dados e sistemas de processamento que podem ajudar a tomar decisões inteligentes com base em várias fontes de dados [Choi et al. 2016]. Esses sistemas de BI estão altamente relacionados com o desenvolvimento posterior no DSS, criado na década de 1970. Os sistemas de suporte à decisão mais atuais são baseados em tecnologias e técnicas de informações avançadas e tem uma capacidade sistemática sofisticada para coletar, analisar os dados e convertê-los em informação ou conhecimento sobre oportunidades e ameaças para fornecer soluções inteligentes para operações corporativas [Chen and Lin 2021]. O BI tornou-se indispensável para a tomada de decisões estratégicas em empresas e governos em todo o mundo. Ele desempenha um papel importante na manutenção do negócio, melhor relacionamento com os setores internos e no cumprimento de metas e objetivos de curto e longo prazo. Alguns estudos também confirmam que nos benefícios da implementação do BI estão incluídos um melhor desempenho, eficiência, produtividade, planejamento de recursos e redução de custos, o que pode levar a uma grande vantagem competitiva [Tavera Romero et al. 2021].

3.1. Storytelling

Uma história pode ser definida como, “uma narração dos eventos na vida de uma pessoa, a existência de uma coisa ou tais eventos como um assunto para narração”. Também pode ser definida como “uma série de eventos que são ou podem ser narrados”. *Storytelling* é um conceito popular usado em várias áreas como, educação, entretenimento e mídia. Contar histórias é uma técnica usada para apresentar relações dinâmicas entre os indivíduos por meio de interações [Tong et al. 2018]. Recentemente essa narrativa de modo visual está recebendo atenção da comunidade, onde ferramentas são desenvolvidas de forma que seja possível criar histórias e fornecer suporte visual. O processo de análise e modificação de dados em histórias compartilhadas visualmente, inclui a exploração desses dados para que seja possível transmiti-los de forma a contar uma narrativa e, em seguida, comunicar- se com o público. Histórias oferecem uma forma eficaz de armazenar informações e conhecimento, tornando mais fácil para as pessoas percebê-las [Salvadorinho et al. 2020]. Então, o que torna a narração de histórias tão especial? As histórias sempre foram uma das ferramentas mais

importante de compartilhamento de informações entre indivíduos. Ao ouvir, ou ver uma história baseada em dados, o corpo libera dopamina, o que coloca emoções por trás destes dados, tornando-os mais fáceis de compreender e lembrar [Pearson et al. 2020].

3.2. Visualização de Dados

O maior objetivo da visualização de dados é obter uma visão por meio de gráficos interativos ou de características relacionadas a algum processo de interesse, que podem ser algum processo no mundo real ou uma simulação científica. Segundo Alexandru Telea, a visualização "é um processo cognitivo realizado por humanos na formação de imagem mental de um espaço de domínio. Na ciência da computação da informação, é, mais especificamente, a representação de um visual em um espaço de domínio com a utilização de gráficos, imagens, aumento de sons e sequências animadas para apresentar os dados. A estrutura e o comportamento dinâmico de grandes conjuntos de dados complexos representam sistemas, eventos, processos, conceitos e objetos" [Telea 2014]. A visualização de dados é estruturada em três campos principais: A visualização científica, a visualização de informação e a visualização analítica. A visualização científica surgiu como resposta à quantidade de dados gerados por simulações numéricas em computadores. O atributo "científico" neste caso refletia o foco em soluções de visualização no fornecimento de *insights* sobre simulações científicas. Processos físicos, como fluxo de fluido, convecção de calor, deformações de materiais, estudos arquitetônicos, meteorológicos, médicos, biológicos entre outros, são estudos que são representados por visualizações científicas. A visualização de informação abrange uma região onde os tipos de dados não têm posições espaciais prescritas. Conjunto de dados relacionados, assim como qualquer outro conjunto de dados, exige uma representação espacial. Esta informação espacial é atribuída aos elementos de dados durante a construção da visualização, em vez de ser fornecida pelos próprios elementos, como ocorre na visualização científica. O principal catalisador para o crescimento no interesse pela visualização de informação é o aumento no número de dados e artefatos digitais, *"big data"*. Já a visualização analítica (do inglês *Visual Analytics*), surgiu com a necessidade de combinar soluções de visualização com análise de dados e *front-ends* de mineração de dados. O objetivo central desta análise é fornecer as técnicas e ferramentas que auxiliem os usuários em seu raciocínio analítico, por meio de interfaces visuais interativas [Wong and Thomas 2004]. Alguns aspectos diferenciam a análise visual de seus predecessores. A análise visual se concentra no processo de criação, que começa com a aquisição dos dados e continua por meio de uma série de cenários de visualização repetidos e refinados (onde é envolvida toda a interação que permite ao usuário explorar diferentes pontos de vista, testar e refinar diferentes hipóteses), e termina apresentando uma visão final do usuário sobre os fenômenos de interesse. Logo, é possível dizer que essa área é caracterizada por uma combinação de análise de dados, mineração de dados e tecnologias e ferramentas de visualização.

3.3. Painéis Visuais *(Dashboards)*

A definição de painéis de visualização *(dashboards)*, se encontram em constante evolução. [Few 2017] descreve de uma forma mais sucinta como uma exibição de informações visuais que os usuários utilizam para monitorar de forma rápida, as condições atuais que exigem uma resposta clara e oportuna, a fim de cumprir determinada função específica.[Wexler et al. 2017] apresentam uma definição mais ampla: "uma exibição visual de dados usados para monitorar as condições e/ou facilitar a compreensão", que podem incluir elementos gráficos ou visualizações narrativas (Figura 1). Consequentemente, o conceito de *dashboards* evoluiu de telas de relatório de visualização única, para incluir interfaces interativas com múltiplas visualizações e propósitos, incluindo comunicação, aprendizado e motivação, além das noções clássicas de monitoramento e suporte a decisão [Sarikaya et al. 2018].

Figura 1. O painel do Gerenciador de Mídia Social do Klipfolio, localizado à esquerda, é um painel tradicional, com grandes números representando as principais métricas e gráficos de dados em tempo real. O painel de Resposta de Emergência de Refugiados/ Imigrantes do UN-CHR, localizado à direita, é uma justaposição de métricas-chave e visualizações simples, porém inclui anotações e elementos narrativos guiados [Sarikaya et al. 2018].

4. Proposta

Como forma de otimizar a consulta por informações, obter relatórios de forma prática e atual com apenas poucos cliques, a Coordenadoria de Auditoria Interna (COAUD) no TRE-BA criou o seu próprio *dashboard* de BI, em 2021. Ao analisar o painel criado, é possível verificar algumas falhas na exibição dos visuais: tabelas extensas e/ou mal formatadas, escolha inadequada de gráficos (pizza e barras horizontais), falta de legenda descritiva em alguns atributos e um posicionamento contra-intuitivo dos menus são alguns pontos de melhorias presentes no painel. Como descrito anteriormente, para que um *dashboard* atenda de forma clara e eficaz, é necessário que se construa uma história por traz destes dados, logo a aplicação do conceito *Storytelling* se torna necessário. Uma mudança de visual, posição dos itens e análise profunda dos dados são alguns passos importantes a serem considerados para a criação de um novo painel para a COAUD.

4.1. Tecnologias Utilizadas

Figura 2. Gráfico do Gartner 2021 exibindo as melhores plataformas de BI obedecendo os eixos de 'Habilidade de Execução' e 'Melhor Visualização'.Tableau e Power BI destacam na facilidade e eficieˆncia na criação de visualizações e painéis visuais.

O *software* utilizado para desenvolver as metáforas visuais em conjunto com os painéis visuais será o Microsoft Power BI. O Power BI é uma ferramenta de BI que torna a visualização bastante rica e interativa, o que permite uma criação diversificada de *dashboards*. Embora exista a opção de executar scripts em R e Python, existem também outros meios de realizar análises e criação de visuais. O programa é capaz de se conectar com várias fontes de dados, extraindo e transformando-os,

facilitando as análises exploratórias de visuais dos dados. Atualmente, de acordo com o Quadrante Mágico do Gartner, o software Power BI está na primeira posição no *ranking*, desde fevereiro de 2019, à frente do Tableau, que até então era a ferramenta de maior reconhecimento na área de BI (Figura 2). Através de ferramentas que permitem uma boa manipulação do *Storytelling*, o Power BI permite também a utilização de *"drill-through"* nos visuais localizados no *dashboard*. A fim de oferecer informações a um nível mais detalhado, sobre algum atributo específico, o usuário pode com um simples clique obter tais informações (Figura 3). Este atributo auxilia na exibição de informação mais detalhada de alguns componentes visuais, facilitando ao usuário a obter informações mais detalhadas sem modificar a estrutura visual e melhorando o *Storytelling*.

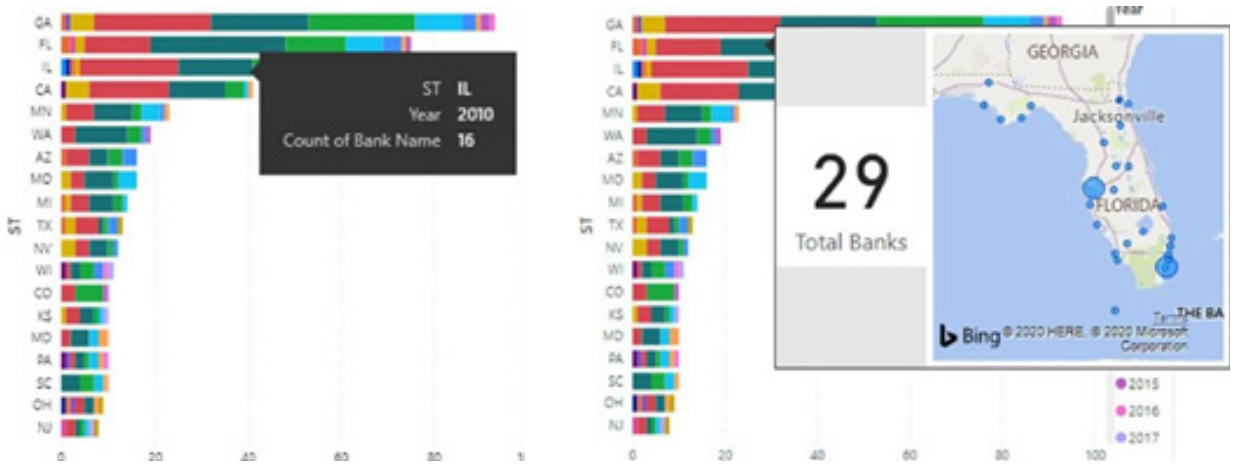


Figura 3. : Função *drill-through* nos visuais. À esquerda é exibida algumas informações básicas sobre o visual. À direita é incorporado à informação um outro visual de georreferenciamento e um cartão exibindo as mesmas informações da imagem à esquerda [Pearson et al. 2020].

4.2. Estruturação e Análise dos Dados

Os dados cedidos pela COAUD exibem informações de auditoria em todo o tribunal. A análise destes dados está voltada a cumprir alguns requisitos que foram solicitados pela coordenadoria. Foram considerados como prioridade o cuidado na exibição de alguns dados por força da Lei de Acesso À Informação, art. 7º, inciso VII, alínea "b"que os tornam públicos e mantendo restrição a informações relacionadas aos dados protegidos pela Lei Geral de Proteção de Dados (13.709/2018).

Como os dados estavam organizados em planilhas, todo o processo de Extração, Transformação e Carregamento (ETL), foi realizado pelo próprio Power BI. Inicialmente foi realizada uma análise do painel desenvolvido pela COAUD. Foram observados os visuais utilizados, a disposição e posicionamento dos elementos visuais, os mapas de cores utilizados para mapear as informações, os tipos de visualizações e os tipos de dados que estavam sendo mapeados e de que forma os requisitos iniciais foram cumpridos. O painel da COAUD era dividido em duas partes, a primeira exibia dados estatísticos, na qual as visualizações adotadas eram gráficos de barras verticais, horizontais, pizza e um cartão. (Figura 4). Através deste painel inicial podemos verificar que o esquema de cores e os gráficos adotados não auxiliam na visualização, pois não obedecem nenhum padrão lógico. Começando pelo visual de barras horizontais, no qual é bastante extenso, notamos que algumas unidades se repetem com valores diferentes, o que pode confundir a acurácia dessas informações. Ao lado temos um gráfico de pizza, no qual é recomendado a utilização de até 3 (três) atributos diferentes para este tipo de visual, pois acima disso não é possível obter informações de forma adequada, O que não ocorre no painel, que neste caso abriga 6 (seis) atributos diferentes e com cores que divergem totalmente do esquema de cores adequado, pois percebe-se o uso de tons de azul que pode confundir o usuário cognitivamente na leitura e interpretação do gráfico. Abaixo temos outro gráfico de barras verticais que possuem a mesma cor que um dos atributos do gráfico de pizza, o que pode confundir a informação que está sendo mapeada em ambos gráficos. Não é informado pelo

painel se os atributos têm alguma ligação. Notamos também que não é informado a legenda de um atributos no eixo do visual de barras e nem o que ele representa, deixando aberto ao usuário a sua interpretação. Acima do gráfico de pizza, é exibido um cartão informativo.

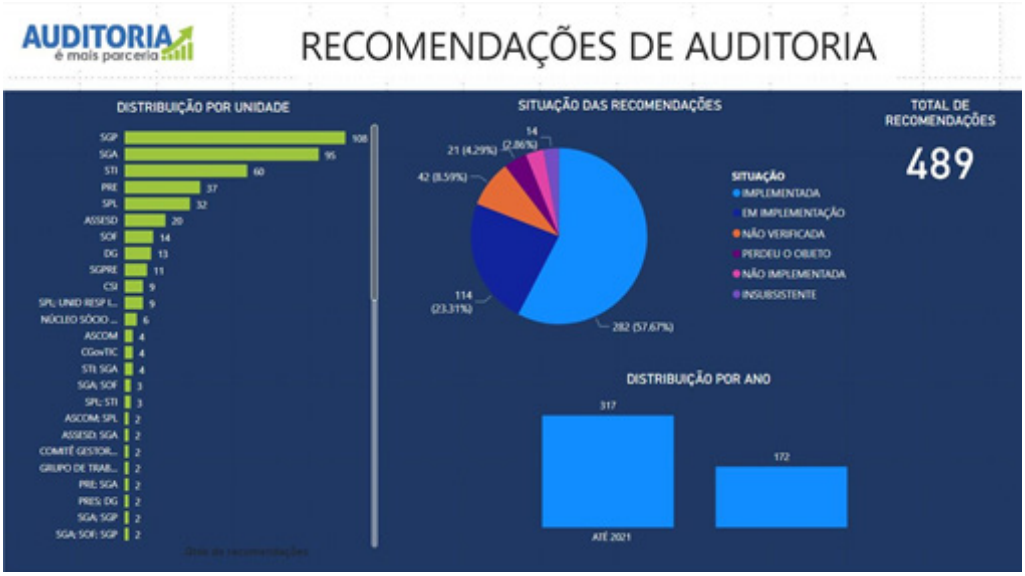


Figura 4. Página inicial do painel da COAUD, onde eram exibidos visualizações para consulta de informações como: distribuição por unidade e a situação das recomendações da unidade

É importante citar que todos os gráficos visuais possuem interação e todas essas interações refletem nos gráficos e no cartão informativo.

O segundo painel da COAUD exibe somente dados em tabelas. São observadas 3 (três) tabelas, duas contendo dados referentes às unidades e uma tabela mais descritiva abaixo. Também são exibidos dois filtros de segmentação de dados, esses filtros são utilizados para filtragem nos dados a serem exibidos. Ao lado dos filtros é exibido novamente um cartão informativo (Figura 5). Novamente verificamos que em duas tabelas são exibidos dados repetidos e/ou agrupados, o que dificulta na obtenção de resultados.

Figura 5. Segunda página do painel da COAUD, onde são exibidos os visuais de consulta. Os dados contidos nas tabelas foram censurados para garantir a segurança das informações.

As tabelas não possuem nenhuma informação adicional, portanto é possível concluir que servem apenas para consulta direta de informações. Não é informado se os dados exibidos estão relacionados aos visuais anteriores e o visual de cartão traz exatamente a mesma informação que no painel anterior. É importante ressaltar também que o título de ambos os painéis, "Recomendações de Auditoria" parece não estabelecer nenhuma relação com os dados exibidos, pois são dados informativos e não de recomendações.

5. Aplicação Prática

Figura 6. Página inicial do painel da COAUD, onde são exibidos os visuais de consulta rápida e de forma interativa.

A partir das informações coletadas no painel e seguindo os mesmos requisitos impostos pela COAUD, foi desenvolvido um novo painel composto de três partes. O objetivo deste painel é estruturar as informações de forma mais adequada, para que os dados exibidos possam contar a sua narrativa própria usando *Storytelling*.

O painel inicial, chamado de Gestão de Projetos, (Figura 6), apresenta um panorama geral dos projetos de auditoria. No canto superior direito temos três botões de navegação entre as páginas, para auxiliar nas consultas, estes botões estão presentes em todos os painéis. Na parte superior central, encontramos um visual de segmentação temporal de dados. Os botões apresentados neste visual auxiliam em uma filtragem por ano, ao selecionar o ano atualiza as informações e gráficos exibidos abaixo, com exceção do gráfico de linhas que não são alterados por esta interação. Em seguida, observamos mais 4 visuais, todos são nomeados de acordo com a sua respectiva função e as cores escolhidas fazem referência às cores do logo da COAUD. "Recomendação por Ano" é um gráfico de linha, o eixo "x" é uma série temporal, no qual mostra a quantidade de recomendações anualmente. Ao lado temos um gráfico de funil, "Situação das Recomendações". Este visual exibe a quantidade de recomendações pela sua situação. É importante notar que uma das barras é apresentada em uma cor diferente (azul), esta barra diz respeito a`s recomendações implementadas e faz um *link* ao visual abaixo, na direita, que exibe o total de recomendações implementadas (azul) e pendentes (verde) por sua respectiva unidade, em "Recomendações por Unidade". No final do painel temos dois visuais de barras verticais, estes visuais exibem as 10 unidades com mais recomendações em ordem decrescente. Esta escolha foi feita pois muitas unidades apresentam poucas ou nenhuma

recomendação. Apresentá-las nos visuais deixaria o painel poluído e não traria nenhum benefício de informação, pois todos os visuais são dinâmicos, então a depender do filtro escolhido é possível obter informação de todas as unidades.

Figura 7. Segunda página do painel da COAUD, "Estatística", no qual exibe alguns visuais de consulta rápida. Os dados contidos na tabela foram censurados para garantir a segurança das informações.

Na página "Estatística" (Figura 7), é possível obter informações rápidas de alguns dados estatísticos de suas respectivas unidades. O painel também apresenta um filtro de segmentação de dados anual, importante para as consultas. Ao lado superior direito temos dois cartões informativos, sobre a quantidade de exercícios realizados e a sua taxa de implementação. Podemos notar no centro esquerdo do painel novamente um filtro de segmentação, neste caso, destinado a filtragem das unidades. Abaixo estão duas visualizações, um gráfico de barras, "Atividade Fiscalizadora por Unidade" e uma tabela "Detalhamento de Atividade Fiscalizadora", no qual exibe informações filtradas pelos demais visuais. O objetivo deste painel é apresentar de forma clara e rápida a atual situação das atividades por unidade.

Por fim temos o terceiro e último painel, "Banco - SEAGO" (Figura 8), apresenta consultas por tabela. São apresentados dois componentes visuais de segmentação, ao qual filtramos as unidades e a situação das informações. Dois cartões que exibem o total das recomendações implementadas e a taxa de recomendações que foram cumpridas. Abaixo uma tabela clara e limpa, contendo as informações necessárias e requisitadas pelo usuário. Apesar de parecer um painel simples, a tabela localizada ocupa aproximadamente metade do painel, o que torna a obtenção de informação bastante densa, por este motivo foram escolhidos somente os filtros necessários para obtenção das informações.

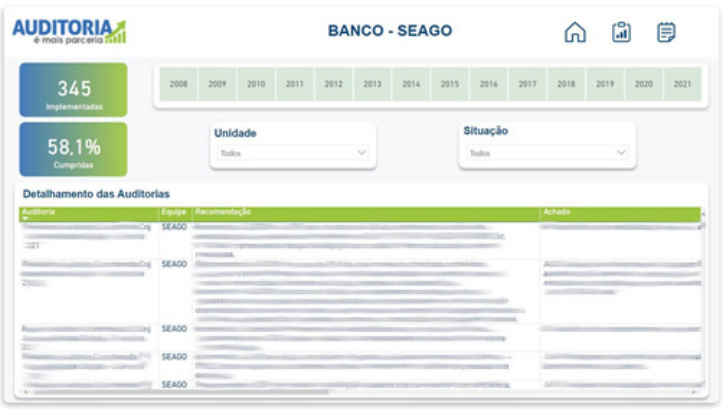


Figura 8. Última página do painel da COAUD, "Banco - SEAGO", exibe poucos visuais de consulta, pois a extensa tabela torna o painel bastante denso. Os dados contidos na tabela foram censurados para garantir a segurança das informações.

6. Considerações Finais

O *dashboard* criado permitiu mostrar uma visão geral e informativa ao usuário, facilitando sua interpretação dos dados obtidos. O Power BI se mostrou uma excelente ferramenta, capaz de "contar uma história" sobre as informações de auditoria no tribunal, permitindo ao usuário criar *insights* sobre o período observado. É importante frisar que todos os dados utilizados para a construção do painel, foram os mesmos utilizados no painel anterior. Enquanto no painel antigo a prioridade era somente exibir as informações, o painel atual teve como prioridade o *Storytelling*. Organizar as informações através de botões, cartões, cores e visuais interativos, facilita no entendimento das informações e auxilia na tomada de decisão. Como trabalho futuro, é importante pensar na manutenção dos dados, já que foram obtidos de planilhas e não diretamente de um banco de dados. Também é importante testar o painel com outros tipos de usuários, para que possíveis informações possam ser acrescentadas, a depender das necessidades.

Referências

Chen, Y. and Lin, Z. (2021). Business intelligence capabilities and firm performance: A study in china. *International Journal of Information Management*, 57:102232.

Choi, T.-M., Chan, H. K., and Yue, X. (2016). Recent development in big data analy- tics for business operations and risk management. *IEEE transactions on cybernetics*, 47(1):81–92.

Few, S. (2017). There's nothing mere about semantics. *Perceptual Edge*.

Hedgebeth, D. (2007). Data-driven decision making for the enterprise: an overview of business intelligence applications. *Vine*.

Pearson, M., Knight, B., Knight, D., and Quintana, M. (2020). Data storytelling with power bi. In *Pro Microsoft Power Platform*, pages 291–305. Springer.

Salvadorinho, J., Teixeira, L., and Santos, B. S. (2020). Storytelling with data in the context of industry 4.0: A power bi-based case study on the shop floor. In *International Conference on Human-Computer Interaction*, pages 641–651. Springer.

Sarikaya, A., Correll, M., Bartram, L., Tory, M., and Fisher, D. (2018). What do we talk about when we talk about dashboards? *IEEE transactions on visualization and computer graphics*, 25(1):682–692.

Tavera Romero, C. A., Ortiz, J. H., Khalaf, O. I., and R´ıos Prado, A. (2021). Business intelligence: business evolution after industry 4.0. *Sustainability*, 13(18):10026.

Telea, A. C. (2014). *Data visualization: principles and practice*. CRC Press.

Tong, C., Roberts, R., Borgo, R., Walton, S., Laramee, R. S., Wegba, K., Lu, A., Wang, Y., Qu, H., Luo, Q., et al. (2018). Storytelling and visualization: An extended survey. *Information*, 9(3):65.

Wexler, S., Shaffer, J., and Cotgreave, A. (2017). *The big book of dashboards: visualizing your data using real-world business scenarios*. John Wiley & Sons.

Wong, P. C. and Thomas, J. (2004). Guest editors' introduction–visual analytics. *IEEE Computer Graphics and Applications, 24 (5): 20-21*, 24(PNNL-SA-41935).

Isabela Plessim[1]
Tribunal Regional Eleitoral da Bahia
1ª Av. do Centro Administrativo da Bahia, 150 - CAB
Salvador-BA - CEP: 41.745-901 - Brasil
isplessim@tre-ba.jus.br

1 Daniela Barreiro Claro
Universidade Federal da Bahia - Instituto de Computação Departamento de Ciência da Computação. Salvador, BA – Brasil. dclaro@ufba.br

Análise da Convocação de Mesários no TRE-BA

Abstract. *In order for the Brazilian Electoral Court to carry out its mission, voters are called to work as election officials on election days. The call-up work requires a prior analysis of voter characteristics, in order to identify those that allow a better definition of a potential poll worker, minimizing the risk of absenteeism. It is important to note which voters would be most likely to work in the polling stations. The main objective of the present work is to classify potential voters to attend the call to work as a polling station and act as expected by the Electoral Zones.*

Resumo. *Para que a Justiça Eleitoral brasileira concretize a sua missão, eleitores são convocados para trabalhar como mesários nos dias dos pleitos. O trabalho de convocação requer uma análise prévia de características do eleitor, a fim de identificar aquelas que permitam melhor definir um potencial mesário, minimizando o risco de faltantes. É importante observar quais eleitores estariam mais propensos a trabalhar nas seções de votação. O presente trabalho tem por principal objetivo classificar potenciais eleitores a atender à convocação para trabalhar como mesário nas seções de votação e atuar da forma esperada pelas Zonas Eleitorais.*

1. Introdução

A participação dos eleitores como mesários nas eleições brasileiras é fundamental para a organização dos trabalhos nas seções eleitorais no dia da votação.

> *O mesário é o representante da Justiça Eleitoral na seção de votação. Cabe a ele receber e identificar os eleitores – seja pela verificação de documentos e coleta de assinaturas, seja pela verificação biométrica –, compor as mesas de votos e justificativas, fiscalizar e desempenhar tarefas logísticas e de organização da seção para a qual foi designado.* Fonte: [Tribunal Superior Eleitoral 2018]

Os eleitores podem se inscrever para trabalhar como mesários voluntários, mas são as Zonas Eleitorais que, conhecendo o quantitativo de suas seções eleitorais e o cadastro dos inscritos, realizam a convocação. Embora o atendimento à convocação seja obrigatório, há registros de faltosos em diversas seções e, entre os que comparecem, alguns não apresentam um desempenho satisfatório. Há ainda aqueles que solicitam dispensa, apresentando justificativas diversas, desde problemas de saúde, até impedimentos relacionados às suas atividades laborais. Em 2020, de acordo com estatísticas do Tribunal Superior Eleitoral (TSE) [Tribunal Superior Eleitoral 2020], houve um percentual geral de 3,74% de mesários faltosos, mas na cidade de Salvador, este percentual foi de 15,06%.

1.1. Contexto

Dados obtidos em [Tribunal Superior Eleitoral 2020] mostram que, em 2020, o eleitorado do Brasil compreendia 147,9 milhões de eleitores aptos a votar. Entre os eleitores aptos, foram convocados para trabalhar como mesários 1.589.896 e, destes, 59.468 não compareceram ou abandonaram a função, ou seja, 3,74% do total de convocados. A Bahia, que é o quarto colégio eleitoral do país, contava com 10,8 milhões de eleitores aptos, dentre os quais 107.187 foram convocados e 4.407 não compareceram, perfazendo um total de 4,2% mesários faltosos. Mas esta distribuição não é homogênea, e o percentual pode variar bastante, se compararmos as zonas eleitorais e os municípios separadamente.

Na Bahia, entre os mesários convocados, 23,6% foram voluntários e em Salvador, 53,2% foram voluntários. Salvador está dividida em dezenove Zonas Eleitorais, cujos números variam de 1 a 19. Ao observar o comparecimento de mesários por Zona Eleitoral na Bahia, percebeu-se que, entre as dez zonas com maior percentual de faltosos no estado, seis são de Salvador: zonas 12, 19, 10, 5, 1 e 15. A Tabela 1 apresenta as zonas com maior percentual de mesários faltosos em 2020.

Tabela 1. Zonas da Bahia com maior percentual de mesários faltosos em 2020

UF	Zona	Comparecimento	% comparecimento	Ausencia	% ausencia	Convocados
BA	12	598	61,59	373	38,41	971
BA	19	698	67,7	333	32,3	1031
BA	166	210	74,2	73	25,8	283
BA	25	360	75,47	117	24,53	477
BA	10	872	76,02	275	23,98	1147
BA	5	595	79,55	153	20,45	748
BA	1	454	79,65	116	20,35	570
BA	15	606	80,91	143	19,09	749
BA	171	672	81,55	152	18,45	824
BA	11	851	83,6	167	16,4	1018
BA	157	797	84,25	149	15,75	946
BA	16	727	84,34	135	15,66	862
BA	14	900	86,71	138	13,29	1038
BA	4	935	86,9	141	13,1	1076
BA	13	754	87,88	104	12,12	858
BA	59	612	88,18	82	11,82	694
BA	200	325	89,29	39	10,71	364
BA	2	628	89,33	75	10,67	703
BA	22	629	89,73	72	10,27	701
BA	8	884	89,84	100	10,16	984

Na Tabela 2, observa-se que o maior percentual de mesários ausentes na Bahia foi verificado nos municípios de Buerarema, Arataca e Salvador.

Tabela 2. Municípios baianos com maior percentual de mesários faltosos em 2020

UF	Município	Convocados	Comparecimento	% comparecimento	Ausencia	% ausencia
BA	BUERAREMA	153	105	68,63	48	31,37
BA	ARATACA	83	61	73,49	22	26,51
BA	SALVADOR	16578	14081	84,94	2497	15,06
BA	MANOEL VITORINO	129	111	86,05	18	13,95
BA	POÇÕES	362	313	86,46	49	13,54
BA	BOA NOVA	159	138	86,79	21	13,21
BA	ILHÉUS	1087	959	88,22	128	11,78
BA	SALINAS DA MARGARIDA	128	113	88,28	15	11,72
BA	POJUCA	270	239	88,52	31	11,48
BA	NOVA CANAÃ	132	117	88,64	15	11,36

Salvador está, então, entre os três municípios com maior percentual de mesários faltosos e é, em termos absolutos, quem mais convoca na Bahia.

1.2. Motivação

O Cadastro Eleitoral está armazenado em um banco de dados relacional e contém, ou podem ser gerados a partir dele, dados demográficos relativos a gênero, idade, ocupação, estado civil, escolaridade, assim como o histórico da relação do eleitor com a Justiça Eleitoral. Por exemplo, se faltou a pleitos anteriores, se justificou, se foi convocado e atendeu à convocação, quando e quais foram os requerimentos realizados de alistamento, revisão, transferência ou emissão de segunda via etc.

O absenteísmo de mesários convocados, além da convocação em si, também fica registrado no Cadastro Eleitoral, assim como os motivos de dispensa e atuação imprópria. Sendo assim, uma técnica que permita indicar se um eleitor que nunca foi convocado tem mais chances de atender à convocação com um desempenho adequado poderia ajudar a Zona Eleitoral na escolha de novos mesários.

1.3. Objetivos (geral e específicos)

Com a aplicação do processo de KDD sobre o Cadastro Eleitoral busca-se verificar se é possível, a partir dos dados armazenados relativos a convocações anteriores, sugerir convocação de eleitores sem histórico de convocação para atuação como mesários. Além do resultado alcançado neste objetivo específico, o estudo e aplicação prática no ambiente do TRE-BA das etapas preconizadas pelo processo de descoberta de conhecimento, assim como do recurso de mineração de dados integrado ao banco de dados institucional, ampliam as possibilidades de novas experiências e utilizações futuras no âmbito deste trabalho para o referido órgão.

Assim, o principal objetivo deste trabalho é minerar dados do Cadastro Eleitoral a fim de gerar uma lista de sugestões de convocações. Especificamente, através deste trabalho pretende-se:

• Obter dados sobre convocações, ausências e dispensas;

• Selecionar atributos possivelmente significativos para a convocação de mesários;

• Preparar os dados, a fim de submetê-los aos algoritmos de mineração;

• Explicar resumidamente o modelo de mineração e algoritmos usados;

• Extrair informação potencialmente útil.

O presente trabalho está organizado em seções. A seção 2 descreve a fundamentação teórica necessária para compreensão do método proposto. A seção 3 apresenta o método proposto desenvolvido neste trabalho. A seção 4 descreve os experimentos e os resultados. E, por fim, a seção 5 apresenta as considerações finais do trabalho e alguns direcionamentos futuros.

2. Fundamentação Teórica

A fundamentação teórica deste trabalho engloba a Descoberta de Conhecimento em Banco de Dados (KDD) e o recurso Oracle Data Mining do banco de dados *Oracle Enterprise Database*, assim como a ferramenta utilizada no experimento denominada *Oracle Data Miner*.

2.1. Descoberta de Conhecimento (KDD)

Knowledge Discovery in Databases (KDD) é o processo de descoberta de conhecimento em bases de dados [Fayyad et al. 1996] e foi formalizado em 1989 como uma área multidisciplinar, que utiliza conhecimentos de Estatística, Banco de Dados, Aprendizado de Máquina e Mineração de Dados. Suas etapas incluem a coleta, seleção, limpeza e preparação dos dados, a análise exploratória, visualização de dados, utilização de técnicas de mineração de dados e a interpretação dos resultados obtidos, com o objetivo de descoberta de conhecimento novo potencialmente útil para os usuários, conforme demonstrado na Figura 1.

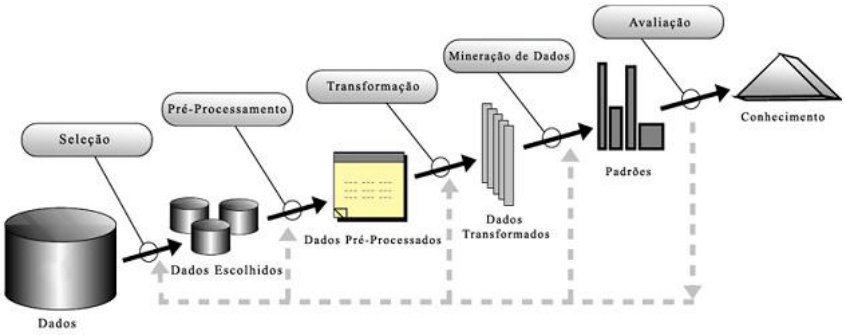


Figura 1. Processo KDD

De acordo com os autores em [Fayyad et al. 1996], o processo KDD é interativo e iterativo e é composto pelas seguintes etapas:

1 - Compreensão do domínio da aplicação, incluindo obtenção de conhecimento prévio relevante e determinação dos objetivos da aplicação de KDD no contexto selecionado.

2 - Criação de um conjunto de dados de destino: inclui a seleção de um conjunto de dados ou um subconjunto de variáveis ou amostras de dados nos quais a descoberta deve ser realizada.

3 - Limpeza e pré-processamento de dados: remoção de *outliers*, se apropriado, tratamento para valores ausentes, estratégias para informações temporais, definição dos tipos de dados a serem usados, entre outros.

4 - Redução e projeção de dados: como representar os dados usando redução de dimensionalidade ou métodos de transformação para reduzir o número efetivo de variáveis em consideração ou para encontrar representações invariantes para os dados.

5 - Escolha do tipo de função de mineração dos dados adequada aos objetivos propostos inicialmente: resumo, classificação, regressão, agrupamento etc.

6 - Escolha do(s) algoritmo(s) de mineração de dados, ou seja, do tipo de modelo a ser usado.

7 - Mineração de dados: buscar padrões utilizando os algoritmos selecionados.

8 - Interpretação: inclui a interpretação dos padrões descobertos e possivelmente retorno a qualquer uma das etapas anteriores para remover padrões redundantes ou irrelevantes.

9 - Usar o conhecimento descoberto: tomar ações com base no conhecimento, ou simplesmente documentá-lo e relatá-lo às partes interessadas, bem como verificar e resolver possíveis conflitos com conhecimento previamente acreditado (ou extraído).

2.1.1. Limpeza e pré-processamento de dados

De modo geral, os dados precisam passar por uma sequência de processamento a fim de estarem prontos a serem utilizados para o processo de descoberta de conhecimento em banco de dados. Estes processamentos incluem procedimentos para resolver dados incompletos, inconsistentes, ausentes ou redundantes, ou atributos cujos valores contrastam sobremaneira em relação aos demais valores, chamados de *outliers*. Em todos esses casos, há necessidade de decidir entre a correção do valor, eliminação do atributo ou do registro em questão do conjunto de dados de entrada ou, ainda, na ocorrência de *outliers*, esclarecer se deve ser mantido, por se tratar de um valor legítimo, ou se realmente ocorreu um erro na alimentação original do dado.

Alteração do tipo de atributos também pode ocorrer nessa fase, como por exemplo, transformação de atributos alfanuméricos em numéricos.

2.1.2. Algoritmos da Mineração de Dados

O aprendizado de máquina pode ser dividido em dois tipos: o **aprendizado supervisionado** e o **aprendizado não supervisionado**. No primeiro tipo, o processo de aprendizagem é dirigido por um atributo previamente conhecido. A partir de um subconjunto dos dados, denominado conjunto de treinamento, os algoritmos aprendem um modelo ou hipótese capaz de relacionar os valores dos atributos de entrada de um objeto do conjunto de treinamento ao valor de seu atributo de saída, ou valor alvo [Carvalho et al. 2011]. Neste caso, a mineração de dados tenta explicar o comportamento do alvo como uma função de um conjunto de atributos independentes, em um contexto em que, tanto os valores dos atributos independentes quanto os valores do atributo alvo são conhecidos. O modelo construído sobre os dados do conjunto de treino é aplicado, então, sobre um novo conjunto de dados, de mesma estrutura do conjunto de treino, porém com novos valores, chamado de conjunto de teste. O objetivo aqui é verificar se o modelo é generalizável, isto é, se aplicado a outro conjunto de dados, consegue predizer o alvo e o grau de acerto.

É importante mencionar que a hipótese formulada pode apresentar uma baixa capacidade de generalização, ou porque está ajustada em demasia aos dados de treinamento, ou porque os exemplos de treinamento não são muito representativos, ou ainda porque o modelo não conseguiu capturar os padrões existentes nos dados. [Carvalho et al. 2011]

As duas técnicas de **aprendizado supervisionado** são classificação e regressão, ambas utilizadas para predição do atributo alvo e utilizadas quando o atributo alvo é discreto ou contínuo, respectivamente. Além disso, também é possível utilizar a *feature importance*, para identificar quais atributos são mais relevantes para a predição.

Como mencionado, a classificação, no contexto de mineração de dados, significa atribuir um rótulo a cada instância do conjunto de dados de entrada. Entre os principais algoritmos de classificação, pode-se citar:

- Árvores de Decisão (*Decision Trees*)
- Vizinhos mais próximos (*k-Nearest Neighbors ou KNN*)
- Classificação Bayesiana (*Bayesian Classification*)
- Máquina de Vetores de Suporte (*Support Vector Machines ou SVM*)
- Modelos Lineares Generalizados (*Generalized Linear Models*)

No **aprendizado não supervisionado** o objetivo é a descoberta de padrões nos dados, sem rótulos específicos previamente conhecidos, através das técnicas de agrupamento, associação e sumarização.

O presente trabalho trata do aprendizado de máquina supervisionado para predição de um atributo alvo discreto, por isso será abordada neste trabalho a técnica de classificação.

2.2. Oracle Data Mining

Principal sistema gerenciador de banco de dados usado no Tribunal Regional Eleitoral da Bahia (TRE-BA), o *Oracle Database Enterprise Edition* possui um recurso chamado *Oracle Data Mining*, que implementa a etapa de mineração de dados no *kernel* do banco de dados e dispõe de algoritmos de mineração e análise de dados para classificação, predição, regressão, associação, seleção de recursos, detecção de anomalias e extração de recursos, para construção de modelos de diversos tipos [Oracle 2017b].

Uma das formas de acessar este recurso é através da ferramenta gráfica *Oracle Data Miner*, que é uma extensão do *Oracle SQL Developer*, amplamente usado no contexto tradicional de manipu-

lação de dados e definição de objetos em banco de dados no TRE-BA, e que permite a realização de várias etapas do processo de KDD além da mineração de dados, como, por exemplo, a limpeza, pré-processamento e redução de dimensionalidade, em um único ambiente, integrando e facilitando a execução e visualização dos resultados obtidos. O *Oracle Data Miner* é baseado em fluxo de processamento de componentes, os quais são escolhidos a partir de um menu gráfico. Assim, toda a construção da descoberta de conhecimento feita na ferramenta fica documentada de modo automático. [Oracle 2017a]

No *Oracle Database 11g*, versão utilizada para este trabalho, o processo de preparação de dados pode ser automatizado, sendo possível também complementar ou substituir as transformações automáticas. As transformações automáticas aplicadas podem ser consultadas nos detalhes do modelo.

Os algoritmos disponíveis na ferramenta para a tarefa de classificação são: Árvore de Decisão (Decision Tree), Naive Bayes, Máquina de Vetores de Suporte (Support Vector Machine) e Modelo Linear Generalizado (GLM - Generalized Linear Model).

Alguns algoritmos usam apenas atributos numéricos, como o SVM e o GLM, e a ferramenta se encarrega de transformar os atributos não numéricos, permitindo que os modelos sejam treinados. Isto é feito de forma transparente, assim, são mostrados os atributos originais nos detalhes dos modelos. Os atributos que passam por transformações, tanto automáticas quanto as especificadas pelo usuário, aparecem nos detalhes do modelo em seu estado pré-transformado, o mais próximo possível dos valores da coluna original. Embora os atributos sejam transformados quando usados pelo modelo, eles são visíveis nos detalhes do modelo de uma forma que pode ser interpretada por um usuário [Oracle 2017a].

É importante mencionar que os testes podem ser realizados também de forma automática, sendo o padrão usar 40% dos dados de entrada para teste. Mas esta configuração pode ser modificada para usar um percentual diferente, ou até todos os dados do treino nos testes, ou ainda para usar dados distintos nos testes, separados do conjunto de entrada [Oracle 2017a].

3. Conjunto de Dados

Os dados foram extraídos do Cadastro Eleitoral, de acordo com o esquema simplificado ilustrado na Figura 2.

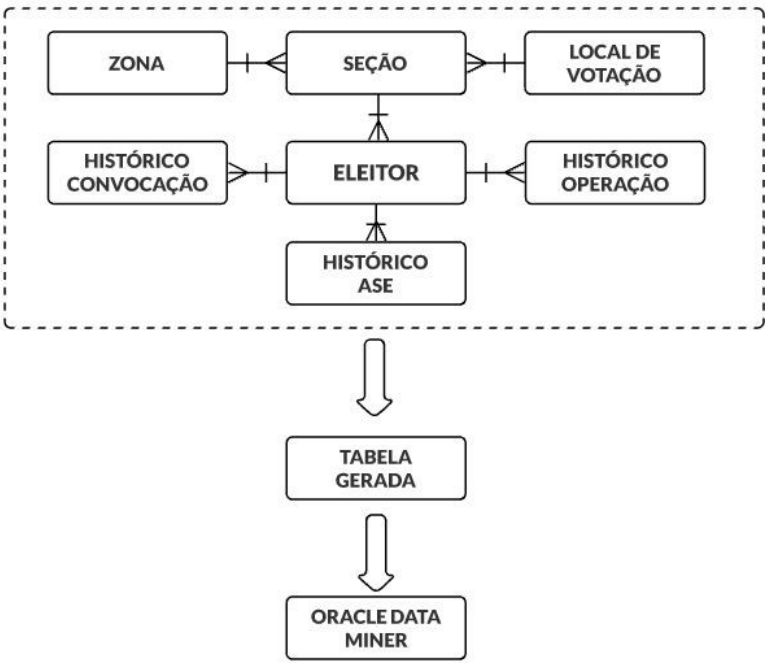


Figura 2. Extração dos dados para a mineração

Do **Histórico Operação** foram obtidas as últimas datas em que o eleitor fez o alistamento, revisão, transferência e segunda via do título. Do **Histórico ASE** foram lidos alguns tipos de eventos ocorridos em 2018 e 2020, tais como:

• ASE 183 2018 e ASE 183 2020 - Convocação para os trabalhos eleitorais em 2018 e em 2020;

• ASE 175 2018 e ASE 175 2020 - Regularização de ausência aos trabalhos eleitorais em 2018 e em 2020;

• ASE 205 2018 e ASE 205 2020 - Habilitação para os trabalhos eleitorais em 2018 e em 2020;

• ASE 442 2018 e ASE 442 2020 - Ausência aos trabalhos eleitorais ou abandono da função em 2018 e em 2020;

• ASE 264 2018 e ASE 264 2020 - Multa eleitoral em 2018 e em 2020;

• ASE 612 2018 e ASE 612 2020 - Registro individual de pagamento de multa eleitoral em 2018 e em 2020;

• ASE 78 2018 e ASE 78 2020 - Quitação de multa em 2018 e em 2020;

• ASE 94 2018 e ASE 94 2020 - Ausência às urnas em 2018 e em 2020;

• ASE 167 2018 e ASE 167 2020 - Justificativa de ausência às urnas em 2018 e em 2020.

A partir do **Histórico Convocação** foram lidos os motivos de dispensa a seguir, se ocorridos em 2018 ou 2020, os quais foram considerados de longo termo e, usados para compor as classes de predição para o treinamento e teste dos modelos:

• Conduta inadequada;

• Decisão judicial;

• Deficiência física;

• Deficiência mental;

• Impedimentos profissionais;

• Inaptidão.

Da tabela **ELEITOR** foram obtidos dados demográficos, como sexo, grau de instrução, estado civil, ocupação e idade, além das datas de domicílio do eleitor na UF e no município e o bairro. Alguns eleitores estavam com o bairro nulo, por isso foi também obtido o bairro do seu local de votação.

Como o aprendizado em estudo é do tipo preditivo, foi criado um atributo que será o rótulo ou alvo da predição. Este atributo pode assumir o valor 1, se o eleitor foi convocado e compareceu, desempenhando de forma satisfatória a função de mesário para a qual foi convocado; ou 0, se foi convocado e não compareceu, abandonou a seção eleitoral antes do encerramento, solicitou dispensa da função de mesário ou se não desempenhou a função de forma satisfatória. Os dados de pesquisa se limitaram aos anos de 2018 e 2020.

O conjunto de dados de entrada ficou então com 34 atributos e 32.096 linhas sobre eleitores do município de Salvador que, nas eleições de 2020 ou de 2018, foram convocados, dispensados por um dos motivos listados acima, ou que estão marcados como faltosos no histórico de convocação, mas sem necessariamente terem um ASE de ausência aos trabalhos eleitorais.

Durante a etapa de compreensão do domínio da aplicação, foram obtidos os quantitativos e percentuais de eleitores convocados em 2020 e em 2018, assim como para os eleitores convocados que não compareceram ou abandonaram a função, inclusive para os dispensados, por zona, por município e totais.

4. Avaliação

Para construção do modelo, foram utilizados os componentes de consulta SQL (*SQL Query*), de exploração dos dados (*Explore Data*), de Amostra (*Sample*) e de construção do modelo de classificação (*Class Build*), conforme Figura 3.

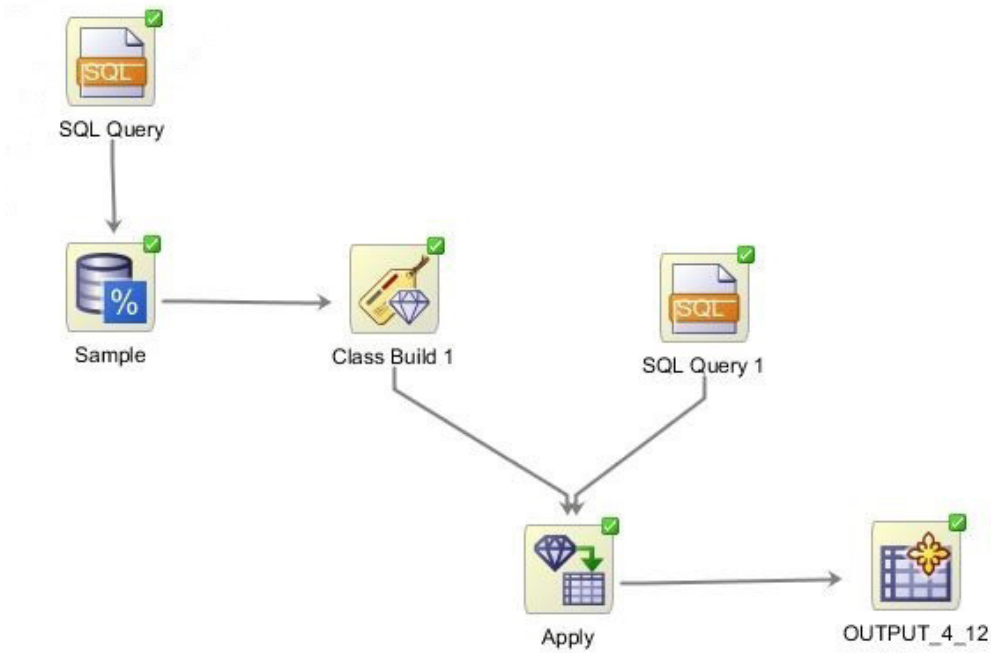


Figura 3. Experimento

Por padrão, o Oracle Data Miner faz transformações nos dados para cada modelo de mineração de modo automático. É possível desmarcar esta opção, mas ela foi mantida, por isso, não definida explicitamente normalização ou qualquer outro procedimento. Apenas ocorreram transformações durante a extração dos dados para a criação da tabela de entrada, conforme descrito na Figura 4.

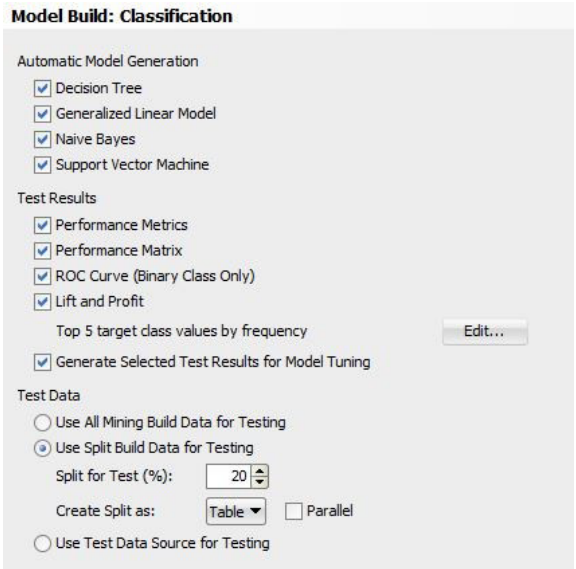


Figura 4. Parâmetros

O componente de exploração dos dados mostra informação estatística, quantidade e percentual de nulos e cardinalidade, conforme figura 5.

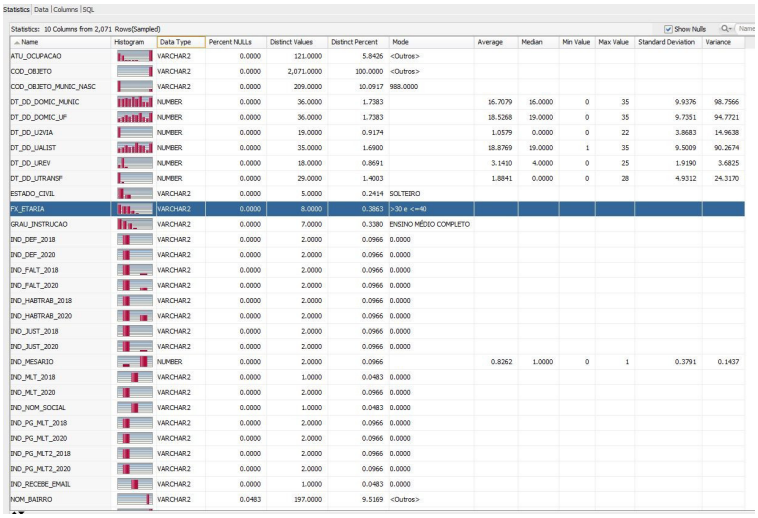


Figura 5. Estatísticas

Veremos a seguir que a ferramenta usa essas informações para decidir se considera ou não um atributo na predição. Com a configuração automática definida, alguns atributos foram excluídos do modelo, o que pode ser verificado na figura 6.

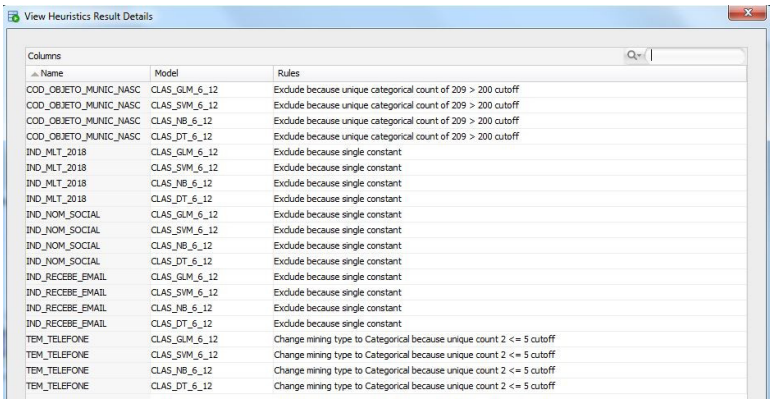


Figura 6. Atributos Excluídos

Como a quantidade de mesários que comparecem é maior que a de faltosos, foi gerada uma amostra com os dados estratificados, de modo que fossem usados 50% de mesários faltantes e 50% de mesários que compareceram. O componente *Sample* foi usado para isso, pois permite selecionar apenas um percentual dos dados de entrada ou, ainda, extrair dele um subconjunto estratificado, mantendo percentuais para os valores do atributo alvo iguais aos existentes no conjunto de entrada, ou definindo que o subconjunto deve ser balanceado. Esta última opção foi a utilizada no experimento, conforme figura 7.

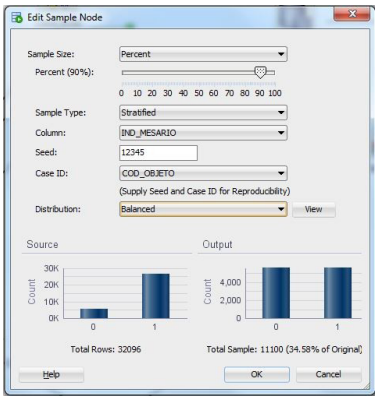


Figura 7. Amostras

Os testes foram realizados de forma automática, selecionando 20% do conjunto de treino para testes. Após os testes, o modelo foi aplicado sobre dados novos de eleitores que não haviam sido convocados nem em 2018 nem em 2020, usando o componente *Apply*, sendo a saída gerada em uma tabela, representada no componente "*Output* 4 12". Esta tabela contém o identificador do eleitor e a predição associada a ele, com o percentual de confiança preditiva encontrado.

Os resultados preliminares são apresentados na próxima seção.

4.1. Resultados Preliminares

A avaliação dos resultados obtidos foi realizada por meio da análise de desempenho de cada preditor na rotulação de teste, usando as medidas disponíveis na ferramenta.

4.1.1. Taxa de confiança preditiva (*Predictive Confidence*)

A confiança preditiva (PC), ou grau de certeza, fornece uma estimativa de quão preciso é o modelo e indica o quanto as previsões feitas pelo modelo testado são melhores do que as previsões feitas por um modelo ingênuo (naive), conforme Equação 1, que sempre prevê a média para alvos numéricos e a moda para alvos categóricos.

$$PC = MAX[(1 - ErroOfmodel/ErrorOfNaiveModel), 0] * 100]$$

Equação 1

Observa-se que o modelo de Árvore de Decisão apresentou uma confiança preditiva de 31,0981%, o que indica que ele reduziu o erro de um modelo ingênuo em 31,0981%.

A acurácia total dos modelos ficou na faixa de 60 e 66% aproximadamente, conforme Tabela 3.

Tabela 3. Tabela comparativa entre os modelos

Algoritmo	Confiança preditiva (%)	Acurácia total
Naive Bayes	21,3321	60,6661
Árvore de Decisão	31,0981	65,5491
SVM	31,1431	65,5716
GLM	32,4932	66,2466

4.1.2. Regras encontradas

Serão apresentadas algumas regras geradas a partir dos dados de entrada pelo algoritmo de Árvore de Decisão. 0 (zero) indica que o eleitor não será indicado para mesário, e 1, que será indicado.

Foi identificado que, se um eleitor votou nas Eleições de 2020, se sua zona não é a 12ª nem a 19ª, se ele também votou em 2018, se ele faz parte do cadastro de eleitores da Bahia há 21 anos ou menos, se foi habilitado aos trabalhos eleitorais em 2020 e se pagou multa individual em 2020, então ele não deve ser indicado como mesário. A taxa de confiança do preditor é de 100%, significando que todos os casos encontrados que satisfazem a estas condições são de eleitores que não devem ser indicados como mesário, seja porque faltaram à convocação, seja porque foram dispensados por um dos motivos considerados neste trabalho como de longo termo. Mas esta previsão tem um suporte de 9 casos, isto é, esta combinação de regras ocorreu em apenas 9 casos.

Outra regra encontrada, desta vez com um suporte de 397 casos, representando 5,96% do total, mostra que, se o eleitor faltou às eleições em 2020 e justificou, se está no Cadastro eleitoral da Bahia há 30 anos ou menos, se sua Zona Eleitoral não é a 3ª, 4ª, 6ª, 7ª, 9ª, 13ª, 14ª, 17ª nem a 18ª, então ele não deve ser indicado como mesário. O preditor tem uma confiança de 73% nesta regra.

Em outra regra, foi apontado que, se o eleitor faltou às Eleições 2020 e não justificou, ele não deve ser indicado como mesário. Esta regra tem uma confiança associada de 77,21% e um suporte de 1.018 casos ou 15,29% do total.

Para a classe positiva, foi revelada a regra de que, se um eleitor votou em 2020, se o número de sua zona é diferente de 12 e 19, se ele também votou em 2018 e se é eleitor na Bahia há mais de 21 anos e meio, então ele deverá ser indicado como mesário. A confiança nesta predição é de 70,39% e o suporte é de 21,92%, ou seja de 1.459 casos. Se, por outro lado, ele pertencer ao cadastro da Bahia há 21 anos e meio ou menos, e se ele não houver sido habilitado aos trabalhos eleitorais em 2020, então ele deve ser indicado como mesário. Esta última regra apresenta uma confiança preditiva de 54,35% e um suporte de 23,47% dos casos.

4.1.3. Matriz de Confusão

A matriz de confusão permite avaliar o desempenho do modelo testado a partir da comparação entre o número de predições corretas e incorretas para cada classe e das relações existentes entre estes números. Nas tabelas 4, 5, 6 e 7 são mostradas as matrizes de confusão para cada classificador usado. Os números 0 e 1 representam as classes negativa e positiva, as quais, neste experimento, são a não indicação do eleitor para mesário ou a indicação para mesário, respectivamente. A coluna real mostra os valores reais, e as colunas 0 e 1 são as predições feitas pelos classificadores usados. Assim, a interseção entre a linha 0 e a coluna 0 mostra a quantidade de verdadeiros negativos ou *true negatives* (TN), ou seja, quantos eleitores não deveriam ser indicados para mesários e não foram indicados; a interseção entre a linha 0 e a coluna 1 mostra os falsos positivos (FP), ou seja, quantos eleitores não deveriam ser indicados para mesários, mas foram indicados; a interseção entre a linha 1 e a coluna 0 mostra os falsos negativos (FN), ou seja, quantos eleitores deveriam ter sido indicados para mesários e não foram; e a interseção da linha 1 com a coluna 1 são os verdadeiros positivos ou *true positives* (TP), isto é, os eleitores que deveriam ser indicados e foram indicados. As tabelas também trazem uma linha e uma coluna chamada ”Acertos(%)”. A coluna de acertos mostra, na primeira linha, a quantidade de verdadeiros negativos em relação à quantidade total de casos negativos, ou seja, o percentual de Especificidade, que é a capacidade do modelo de identificar quem não deve ser convocado para mesário; e na segunda linha, a quantidade de verdadeiros positivos em relação à quantidade total de casos positivos, também chamada de *Recall*, Revocação ou Sensibilidade, que revela a capacidade do modelo de identificar os eleitores que devem ser indicados como mesários. Ao examinar a linha de acertos, constatamos que ela mostra a Precisão Negativa na coluna 0 e a Precisão na coluna 1. A Precisão indica a relação entre todas as predições positivas corretas e todas as predições positivas, inclusive as falsas positivas. E a Precisão Negativa é a medida semelhante à Precisão, mas em relação às predições negativas.

A tabela 4 mostra um total de 4.444 casos testados, divididos meio a meio entre as classes negativa e positiva. Então, 2.222 eleitores dos casos de teste não deveriam ser indicados como mesários e a matriz mostra que 1.259 são verdadeiros negativos, isto é, que realmente eram negativos e foram preditos como negativos, o que significa que houve um acerto na classe negativa de 56,66%. A matriz também mostra 1.654 valores verdadeiros positivos, ou seja, que eram realmente positivos e foram preditos como positivos. Embora o acerto tenha sido de 74,4% para a classe positiva (indicação de mesário), nos casos em que o eleitor não deve ser indicado como mesário, ocorreram 963 predições falso positivas.

Tabela 4. Matriz de confusão da Árvore de Decisão

	predições			
real	0	1	Total	Acertos(%)
0	1.259	963	2.222	56,6607
1	568	1.654	2.222	74,4
Total	1.827	2.617	4.444	
Acertos(%)	68.9108	63.2021		

Tabela 5. Matriz de confusão do SVM

	predições			
real	0	1	Total	Acertos(%)
0	1.272	950	2.222	57,2457
1	580	1.642	2.222	73,8974
Total	1.852	2.592	4.444	
Acertos(%)	68.6825	63.3488		

A tabela 5 mostra a matriz de confusão obtida nos testes do classificador Vetor de Máquina de Suporte (SVM). Podemos observar que a Especificidade e a Precisão foram levemente superiores em comparação com a Árvore de Decisão, mas a Sensibilidade e Precisão Negativa são inferiores.

A matriz de confusão do Modelo Linear Generalizado na tabela 6 mostra que a Especificidade e a Precisão foram superiores aos dois modelos anteriores.

E a matriz de confusão mostrada na tabela 7, do Naive Bayes, é a que apresenta o maior percentual de Especificidade e o menor de Sensibilidade, Precisão e Precisão Negativa.

Tabela 6. Matriz de confusão do GLM

	predições			
real	0	1	Total	Acertos(%)
0	1.371	851	2.222	61,7012
1	649	1.573	2.222	70,7921
Total	2.020	2.424	4.444	
Acertos(%)	67.8713	64.8927		

Tabela 7. Matriz de confusão do Naive Bayes

	predições			
real	0	1	Total	Acertos(%)
0	1.550	672	2.222	69,7570
1	1.076	1.146	2.222	51,5752
Total	2.626	1.818	4.444	
Acertos(%)	59.0251	63.0363		

4.1.4. Curvas ROC

Outra forma de análise dos modelos são as Curvas ROC. A área sob a curva representa a capacidade do modelo de predizer a classe positiva (Figura 9) e a classe negativa (Figura 8). As curvas dos 4 modelos são apresentadas em cada uma das figuras para comparação, e podemos observar que elas ficam bem próximas umas das outras, evidenciando que não houve um modelo cuja capacidade de predizer as classes fosse muito superior à dos demais.

Tanto para a classe negativa quanto para a classe positiva, podemos visualizar que a área sob a Curva ROC associada ao modelo GLM é levemente maior que a dos outros modelos. De fato, a figura 8 mostra um valor de 0,72 para esta área e, para a classe positiva, a curva mostrada na figura 9 indica um valor muito próximo ao da Curva do alvo negativo, também na casa de 0,72, em número aproximado.

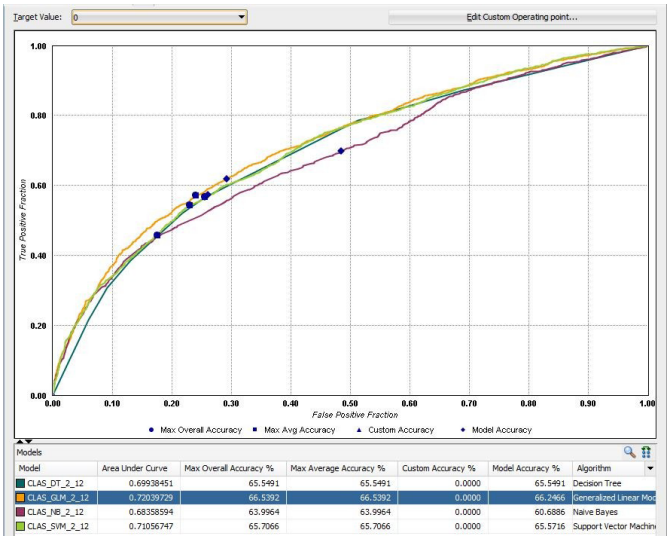


Figura 8. Curva ROC alvo 0

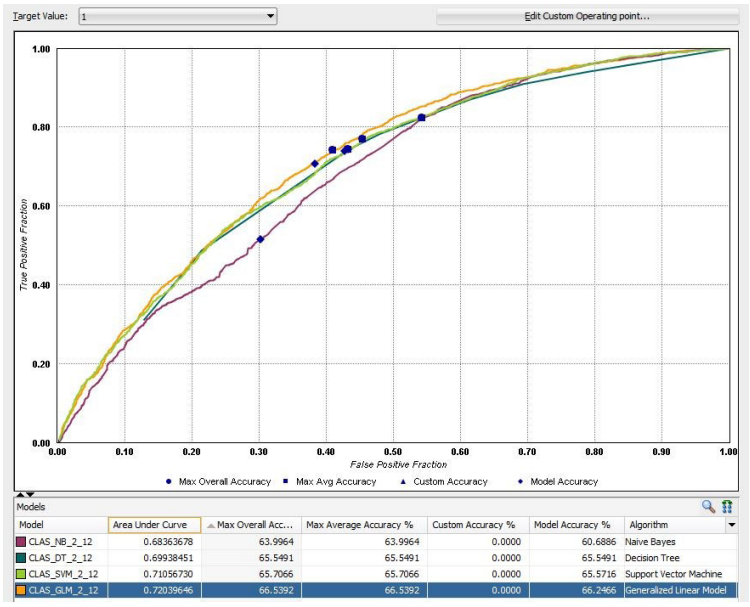


Figura 9. Curva ROC alvo 1

4.2. Discussão

Em 2020, foram encontrados pelo menos 1.530 eleitores de Salvador que foram convocados como mesários, não atenderam à convocação, mas votaram nas suas seções, ou seja, eles possuem um lançamento de mesário faltoso ou que abandonou os trabalhos e não possuem um de ausência às urnas naquele ano. Isto contraria a ideia de que, se um eleitor é convocado como mesário e falta, ele também falta à votação.

Os resultados obtidos não são conclusivos e carecem de uma avaliação mais detalhada. Para validar estes resultados, há necessidade de ouvir especialistas experientes no processo de convocação. Novos experimentos podem ser realizados, usando dados que podem não ter sido considerados neste momento, e possivelmente ajustando os parâmetros usados.

Indicar um eleitor para mesário que não deve ser indicado parece ser pior do que deixar de indicar um mesário que atenderia à convocação de forma satisfatória por causa de uma predição negativa incorreta, desde que exista outro a ser indicado em seu lugar. Neste sentido, o algoritmo Naive Bayes foi o que apresentou o maior acerto na classe negativa e, consequentemente, o que menos errou ao indicar um eleitor para mesário na situação em que ele não deveria ser indicado, com um total de 672 falsos positivos, contra 963 da árvore de decisão, 950 do SVM e 851 do GLM.

Em relação ao uso da ferramenta Oracle Data Miner, a facilidade e rapidez na construção do modelo a tornam bastante interessante, especialmente se considerar que já está disponível no âmbito da Justiça Eleitoral, não havendo um custo adicional associado à sua utilização.

A carga inicial para integração dos dados de origem poderia ter sido realizada inteiramente no Oracle Data Miner, mas, para melhor aproveitamento do tempo, foi decidido realizar as cargas fora da ferramenta.

5. Considerações Finais

O trabalho de mineração sobre os dados de convocação de mesários no Cadastro Eleitoral revelou alguns padrões cujos principais indícios foram apresentados. Contudo, faz-se necessário a presença de um especialista para analisar a relevância dos padrões encontrados e possíveis ajustes no modelo.

Uma possível aplicação deste trabalho seria o rankeamento dos possíveis bons mesários por seção, onde os primeiros da lista seriam aqueles eleitores com maior probabilidade de desempenharem o papel de mesário de modo satisfatório em cada seção eleitoral. A geração de uma lista com indicação de prováveis bons mesários por seção eleitoral, em vez de uma lista única como foi apresentado aqui, potencialmente promoveria uma melhor convocação, uma vez que os eleitores são convocados para trabalhar em uma seção específica, geralmente a sua seção de votação.

5.1. Ameaças

O conhecimento incipiente e o ineditismo da proposta inicial, aliados à restrição de prazo para o encerramento do experimento, podem ter levado a um não aproveitamento de toda a potencialidade da ferramenta usada, especialmente no que se refere aos ajustes possíveis e ao número limitado de simulações realizadas. Com efeito, outros modelos estão disponíveis e podem ser usados nesta e em outras aplicações.

A ausência de uma validação por um especialista da área de negócios eleva o risco de uma interpretação inadequada.

Existe a possibilidade de que os atributos usados sejam pouco representativos para indicação de mesários, ou de que os modelos construídos não tenham capturado os padrões nos dados.

5.2. Trabalhos Futuros

Este trabalho considerou dados apenas de eleitores do município de Salvador, mas pode ser adaptado e ampliado para minerar e analisar dados de outros municípios.

Em relação à ferramenta, outro componente do *Oracle Machine Learning* presente no *Oracle Database Enterprise Edition* é o *Oracle R Enterprise*, que estende as funções estatísticas do banco de dados ao integrar as funções do R[1], permitindo aos usuários a realização de análises estatísticas mais refinadas sobre os dados armazenados no banco. Isto permitiria adicionar novos modelos, como por exemplo florestas aleatórias (Random Forests) ou séries temporais (Time Series), ao fluxo de componentes do Oracle Data Miner. Assim, um próximo trabalho poderia explorar as possibilidades de uso desta integração.

Além disso, a versão do banco de dados usada neste trabalho foi a 11g (11.2.0.3.0). A partir da versão 12c do banco, estão disponíveis outros recursos, como a Análise de Componente Principal (PCA) e a Decomposição em Valores Singulares (SVD) no componente de Extração de Característica, por exemplo.

E por fim, ao confrontar os resultados do experimento com as convocações futuras, passado o pleito, novos dados estarão disponíveis para novas análises, quando poderão ser avaliadas as predições realizadas e comparadas com a distribuição da quantidade de mesários faltosos, iniciando mais um ciclo de mineração, aperfeiçoando os modelos iniciados neste trabalho.

Referências

Carvalho, A., Faceli, K., Lorena, A. C., and Gama, J. (2011). Inteligência Artificial– uma abordagem de aprendizado de máquina. *Rio de Janeiro: LTC*, pages 4–6.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). The KDD Process for Extracting Useful Knowledge from Volumes of Data. *journals/cacm/LangleyS95*, pages 27–34.

Oracle (2017a). *Data Miner User's Guide*. Oracle Corporation. [online]

Disponível em https://docs.oracle.com/database/sql-developer-17.2/DMRUG/data miner-workflows.htm.

Oracle (2017b). *Introduction to Oracle Data Mining*. Oracle Corporation. [online] Disponível em https://docs.oracle.com/database/121/DMCON/.

Tribunal Superior Eleitoral (2018). Conheça a importância da contribuição dos mesários à democracia. Disponível em https://www.tse.jus.br/imprensa/noticias tse/2018/Outubro/conheca-a-importancia-da-contribuicao-dos-mesarios-a democracia-1.

Tribunal Superior Eleitoral (2020). Estatísticas Eleitorais. Disponível em https://www.tse.jus.br/eleicoes/estatisticas/estatisticas-eleitorais.

1 R é uma linguagem e um ambiente para computação estatística e geração de gráficos. Fornece uma ampla variedade de técnicas estatísticas (modelagem linear e não linear, testes estatísticos clássicos, análise de série temporal, classificação, agrupamento etc) e técnicas gráficas. (https://www.r-project.org/about.html)

Laila P. M. Santos[1], **Rodrigo Souza**[1]
[1]Instituto *de Computação – Universidade Federal da Bahia (UFBA)*
Salvador – BA – Brazil
{laila.pereira, rodrigorgs@ufba.br

Desenvolvimento de uma API REST para interoperabilidade de sistemas integrados ao SGRH do TRE-BA com Spring Framework e documentação com Swagger

Abstract. *This article presents the development process and architecture of a REST (Representational State Transfer) API (Application Programming Interface) for the Sistema de Gestão de Recursos Humanos (SGRH) of the Tribunal Regional Eleitoral da Bahia (TRE-BA) in 2021, as a result of a partnership between the Universidade Federal da Bahia (UFBA) and TRE-BA, as well as the results obtained from the implementation using the Swagger documentation framework with Spring framework and its use in the Spring Security module in order to authorize the use of implemented API methods.*

Resumo. *Este artigo apresenta o processo de desenvolvimento e arquitetura de uma API (Application Programming Interface) REST (Representational State Transfer) para o Sistema de Gestão de Recursos Humanos (SGRH) do Tribunal Regional Eleitoral da Bahia (TRE-BA) no ano de 2021, em decorrência de uma parceria entre a Universidade Federal da Bahia (UFBA) e o TRE-BA, bem como os resultados obtidos a partir da implementação usando o framework de documentação Swagger em conjunto com o Spring framework e sua utilização no módulo do Spring Security para a autorização de uso dos métodos da API implementados.*

1. Introdução

1.1. Contexto

Cada vez mais empresas e organizações têm se adaptado às tendências de mercado de forma a otimizar seus processos e sistemas. Uma alternativa utilizada para a interoperabilidade de sistemas que tem ganhado destaque e crescimento nos últimos anos são as APIs (*Application Programming Interface*), que muitas vezes podem servir como uma camada de abstração entre aplicações, protegendo a lógica de acesso direto às bases de dados.

O Tribunal Regional Eleitoral da Bahia (TRE-BA) possui em seu catálogo de sistemas 112 aplicações, das quais 48 atualmente fazem consultas diretas ao Sistema de Gestão de Recursos Humanos (SGRH), uma base de dados robusta com todas as informações sobre a vida funcional e de folha de pagamento dos servidores.

As APIs constituem ferramentas capazes de facilitar a comunicação entre softwares, proporcionando uma padronização na troca de mensagens, ao tempo que não exige de seu usuário profundo conhecimento da lógica de negócio ou da criação de consultas complexas a bancos de dados na obtenção de informações que deseja.

1.2. Motivação

Cada vez mais a utilização das APIs tem ganhado espaço nas organizações por seus benefícios; entretanto, a manutenção desses sistemas pode se tornar uma atividade complexa se não for bem documentado. Por outro lado, documentar um software pode ser uma tarefa cansativa por necessitar de revisão constante em diversos documentos para cada alteração realizada no sistema. Pensando nisso, a utilização de ferramentas que geram documentação de forma dinâmica a partir do código-fonte são extremamente úteis, não apenas para facilitar a documentação em si, como para favorecer que a aplicação e documentação evoluam em sincronia.

A partir da parceria entre a Universidade Federal da Bahia (UFBA) e o TRE-BA, através do projeto de residência em tecnologia da informação, que proporcionou uma vivência prática das etapas de desenvolvimento completo de um software baseado nas demandas do Tribunal, foi proposto o desenvolvimento de uma API REST para a centralização das diversas consultas realizadas ao SGRH por meio das aplicações do TRE-BA, trazendo padronização na comunicação entre sistemas e favorecendo a manutenção das consultas realizadas a base de dados.

1.3. Objetivo

O presente artigo tem por objetivo apresentar o percurso de desenvolvimento, a arquitetura e ferramentas utilizadas na criação da API do SGRH, bem como a importância e utilização do *framework* Swagger no desenvolvimento de uma documentação dinâmica e automatizada, além da avaliação do sistema utilizando a ferramenta SonarQube de análise estática de código-fonte.

2. Referencial Teórico

2.1. API

API é a sigla para *Application Programming Interface* (em português, Interface de Programação de Aplicação). Segundo Brajesh De (2017), de forma geral, uma API é uma interface *software*-para-*software* que define regras e padrões de comunicação entre as aplicações para que estas possam trocar informações sem a necessidade da interação do usuário. Uma API pode ser entendida também como uma forma de comunicação entre aplicações que pode se dar através de uma rede interna (*intranet*) ou uma rede externa (*internet*) e que usa um padrão de linguagem que ambas as aplicações possam entender (Jacobson, Brail, & Woods, 2012).

A comunicação entre a aplicação cliente e o servidor se dá através de endereços, também chamados de *endpoints* (em português, pontos de extremidade), que representam a localização dos participantes de uma troca de informações. No caso da API, um endereço para um método representa o *endpoint* daquele método (Figura 1 - Representação de endpoints de uma comunicação entre cliente e API.).

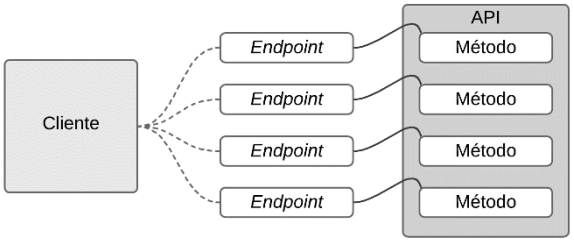


Figura 1 - Representação de endpoints de uma comunicação entre cliente e API.

É esperado de uma API que esteja disponível ininterruptamente e que novas funcionalidades e mudanças na implementação não tragam problemas às aplicações que a consomem. As companhias e organizações que fornecem uma API são chamadas de provedores e, embora várias pessoas de diferentes níveis de conhecimento técnico utilizem APIs atualmente, os desenvolvedores são os principais usuários, são estes que eventualmente construirão aplicações para o consumo das funcionalidades fornecidos por um microsserviço.

As APIs podem ser classificadas em públicas e privadas. Enquanto APIs públicas estão disponíveis para quase todas as pessoas, independentemente de ser uma pessoa desenvolvedora ou não, as APIs privadas são utilizadas para dar suporte interno a uma organização ou a parceiros \cite{jacobson2012apis.

Seja pública ou privada, nos últimos anos houve um crescimento exponencial no número de APIs disponíveis para os mais variados usos. Segundo o site ProgrammableWeb, uma fonte de informações e notícias sobre APIs, o repositório do site atingiu a marca de 22 mil sistemas em junho de 2019 e vem fornecendo alto valor para organizações e desenvolvedores, o que se reflete em seu crescimento contínuo Figura 2 - Crescimento de APIs na internet. (Fonte: ProgrammableWeb, 2019).

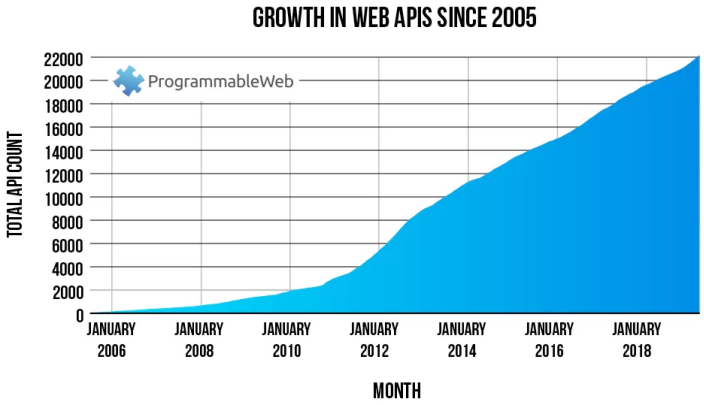


Figura 2 - Crescimento de APIs na internet. (Fonte: ProgrammableWeb, 2019)

2.2. REST

Embora o termo API possa ser utilizado para designar coisas distintas, atualmente é comum associar o termo aos serviços construídos utilizando os princípios REST (*Representational State Transfer*, em português, Transferência de Estado Representativa). O termo foi idealizado e definido por Roy Fielding no ano 2000 em sua dissertação de doutorado intitulada Architectural Styles and the Design of Network-based Software Architectures (em português, *Estilos Arquiteturais e Projeto de Arquiteturas de Software Baseadas em Rede*).

O modelo REST diz respeito ao conjunto de regras e restrições arquiteturais que promove esca-

labilidade das interações de componentes, generalização de interfaces, implantação independente de componentes e componentes intermediários para redução da latência de interação, reforço da segurança e encapsulamento dos sistemas legados (Fielding, 2000)

Essas regras propostas por Fielding também são chamadas de *constraints*, são fruto da preocupação com o futuro e a escalabilidade da internet e determinam como padrões HTTP e URI (*Uniform Resource Identifier*, em português Identificador Uniforme de Recurso, cadeias de caracteres que identificam um determinado recurso na internet), devem ser modelados para que se possa aproveitar ao máximo os recursos fornecidos.

A regra Cliente-Servidor, diz respeito a separação de responsabilidades do sistema, ou seja, a camada de interação com o usuário deve ser separada do *backend* da aplicação, garantindo sua escalabilidade. Essas camadas devem ser capazes de evoluir de forma independente necessitando que o cliente tenha conhecimento apenas das URIs da API, sem a exigência de conhecimento da lógica por trás do sistema.

Outra característica é a *Stateless*, esta regra reflete a inspiração de Fielding (2000) no protocolo HTTP, trata-se de tornar todas as interações entre cliente e servidor sem estado, isto quer dizer, para qualquer que seja a requisição, todas as informações necessariamente já devem estar contidas na requisição para que ela possa ser processada pelo servidor e tratada como totalmente nova, sem registro de sessão ou histórico.

A *constraint* "Cacheável" diz que a aplicação deve ser passível de cache, que deve ser aplicado aos recursos quando necessário, permitindo que o servidor processe apenas tarefas essenciais, sem desperdiçar recursos no processamento de requisições. Esse recurso pode reduzir interações cliente-servidor, melhorando a escalabilidade e a performance do sistema.

Quanto ao quesito Interface Uniforme a regra diz respeito a uma lógica de organização de recursos, as URIs devem seguir um padrão, além disso, os endereços das requisições devem representar bem suas funcionalidades, para que não haja ambiguidade quanto aos recursos disponíveis em cada um dos endereços.

O Sistema em Camadas, favorece a escalabilidade, para que seja possível acrescentar elementos intermediários sem que o cliente precise se preocupar com essas outras instâncias de serviço. Em uma API REST, esse princípio ajuda a criar aplicações mais escaláveis e modulares. Além dos requisitos supracitados, o Código Sob Demanda, é a única opcional, e diz respeito à possibilidade de adaptação do cliente de acordo com novos requisitos e funcionalidades.

Um dos protocolos que podem ser utilizados para a interoperabilidade entre os serviços é o HTTP, essa interação se dá por meio dos métodos de requisição do HTTP, especialmente os verbos GET, POST, DELETE e PUT.

O método GET solicita uma representação de um recurso; sendo assim, esta requisição apenas recupera os dados do servidor. O POST realiza o envio de dados ao servidor e o tipo do corpo da requisição é indicado através do cabeçalho Content-Type. Com o DELETE o objeto ou recurso é removido (Mozilla, 2021).

A principal diferença entre os comandos POST e PUT é que o comando PUT é idempotente, ou seja, efetuar várias requisições do tipo PUT terá o mesmo efeito, enquanto o POST pode ter efeitos adicionais se utilizado repetidamente. Em linhas gerais, o PUT adiciona um recurso em uma URI, de forma que, caso esse recurso já exista nessa URI, será substituído pela nova versão; já o método POST envia o recurso a um URI, mas cabe ao servidor web determinar o que fazer a depender do contexto (Mozilla, 2021).

2.3. JSON

Dentre as formas possíveis de representação dos objetos intercambiados entre serviços, estão o *Extensible Markup Language* (XML, em português Linguagem de Marcação Extensível) e o *Javascript Object Notation* (JSON, em português Notação de Objeto Javascript), mas sua representação pode ser dar por outros formatos de serialização de dados (Seabra, Nazário, & Pinto, 2019).

O JSON é um formato de texto para serialização de dados estruturados, derivado dos objetos literais da linguagem Javascript (Bray, 2017) que facilita a transferência de dados estruturados entre todas as linguagens de programação (ECMA, 2017). Este modelo pode representar quatro tipos primitivos (*strings*, booleanos, números e nulos) e dois tipos estruturados (objetos e *arrays*). As respostas em JSON geralmente apresentam uma estrutura "chave: valor", organizados entre colchetes ([ ]) e chaves ({ }). Cada par de chaves é utilizado para delimitar objetos e os pares de colchetes indicam um *array*, os objetos chave-valor são separados por dois pontos (:) e cada elemento é separado por vírgula, conforme podemos ver na Figura 3 - Exemplo de resposta JSON à uma requisição da API..


```
[
    {
        "lotCod": "1007",
        "lotSigla": "SGP",
        "lotDesc": "SECRETARIA DE GESTÃO DE PESSOAS",
        "lotTipo": "S"
    },
    {
        "lotCod": "86",
        "lotSigla": "",
        "lotDesc": "ASSESSORIA DA PRESIDÊNCIA 1",
        "lotTipo": "S"
    },
```


Figura 3 - Exemplo de resposta JSON à uma requisição da API.

Devido a sua facilidade de leitura e compreensão, em decorrência de sua estrutura simples, o formato JSON atualmente é amplamente utilizado em APIS. Para além da simplicidade, outro fator que corrobora seu uso é a sua performance, uma vez que a estrutura XML pode ser até 3 vezes maior que a estrutura do JSON, isto porque "a análise de XML é lenta e exige muito da CPU" (Davelaar, 2015), pois, além do XML todos os elementos auxiliares também são analisados.

2.4. Documentação

A documentação de uma API é crucial para sua utilização. É a partir da documentação que os usuários obtêm informações a respeito das requisições, parâmetros, autenticação, tipos de informações a serem enviadas e quais esperar de retorno, além dos códigos de erro e sucesso esperados a partir das chamadas dos métodos. Segundo Ambler (2021), os desenvolvedores ágeis reconhecem a documentação como parte intrínseca de qualquer sistema, cuja criação e manutenção é um "mal necessário".

Documentar é a chave para que os desenvolvedores possam compreender um sistema e deve servir para facilitar sua utilização. De acordo com Medina (1984), ao contrário do que muitos podem pensar, a documentação não deve ser adicionada ao final do projeto, mas deve ser construída e atualizada ao longo de todo o processo de desenvolvimento. Além disso, a utilização de comentários, indentação, nomes significativos para as variáveis, espaçamento para enfatizar estrutura lógica, são todos recursos que devem ser utilizados para uma boa organização e facilidade de entendimento dos programas (Medina, 1984).

Uma boa documentação de API deve conter todas as informações necessárias para sua utilização de forma descomplicada para o entendimento humano e sua interface deve ser acessível de forma que não imponha obstáculos no completo entendimento de suas funcionalidades.

Para De (2017), uma documentação eficaz deve conter os seguintes aspectos a respeito de uma API: título, *endpoint*, métodos, parâmetros da URL, carga útil da mensagem (a parte dos dados transmitidos que é a mensagem real pretendida), parâmetros de cabeçalho da requisição, códigos de resposta, códigos de erro, um exemplo de requisição e resposta, tutoriais e passo a passo, acordo de nível de serviço e política de licença de uso.

Enquanto o desenvolvedor de uma API aprende gradualmente todos os aspectos relacionados ao serviço através do processo de desenvolvimento e documentação, esse não é o caso do consumidor da API. O cliente deve ter acesso às respostas para suas questões sobre a funcionalidade, usabilidade, mensagens de erro, formas de autenticação e acesso, dentre muitas outras que possam impactar diretamente sua utilização do sistema. Sendo assim uma documentação de API deve introduzir os usuários rapidamente ao seu uso, incluir informações úteis e relevantes, fornecer um código de exemplo, uma lista de *endpoints* REST e descrever os códigos de status de resposta e mensagens de erro (De, 2017).

Para além do uso por parte do cliente, uma boa documentação serve principalmente para a evolução do software, uma vez que não é incomum que sistemas não possuam a mesma equipe de desenvolvimento trabalhando em sua manutenção de forma permanente. As equipes de desenvolvimento mudam tal qual os sistemas necessitam de adaptações; sendo assim, documentar garante que os próximos a realizar alterações nos códigos o possam fazer sem que seja necessário dispor de muito tempo aprendendo sobre o código e tentando entender suas funcionalidades, ou realizando incansáveis buscas por erros e bugs sem pistas de por onde começar.

O maior desafio quanto à documentação de sistemas em geral é a manutenção desses artefatos em conformidade com o que está de fato implementado (De, 2017), especialmente se a documentação é feita através de documentos externos ao código do projeto ou repositório onde está armazenado. Para auxiliar nessa tarefa pode ser muito útil o uso de ferramentas que realizem a geração automática da documentação.

3. Proposta

3.1. Processo de desenvolvimento

O projeto de uma API REST para o Sistema de Gestão de Recursos Humanos (SGRH), que se trata de uma base de dados cujo objetivo é controlar a vida funcional do servidor e folha de pagamento, se propõe a atuar como uma interface privada capaz de prover a interoperabilidade entre as diversas aplicações e módulos, servindo como padrão para consultas criadas e integralizadas em aplicações futuras.

Utilizando a metodologia de desenvolvimento ágil Scrum e em conformidade com a Portaria nº 245, de 20 de setembro de 2019, que institui os processos de desenvolvimento, sustentação, gerenciamento de escopo e requisitos, gerenciamento de arquitetura e gerenciamento de ciclo de vida de software, bem como o Catálogo e o Gestor Técnico de Soluções de Software, foram adotadas as seguintes etapas de desenvolvimento de software: estabelecimento da visão do produto, planejar entregas e sprints, executar sprints, finalizar sprints, testar entregáveis, homologar e, por fim, implantar o sistema (Tribunal Regional Eleitoral da Bahia, 2019).

Na fase de estabelecimento de visão do produto foram realizadas reuniões com servidores do Tribunal Regional Eleitoral da Bahia (TRE-BA) de vários setores que traziam informações a respeito dos seus processos, em seguida eram discutidas alternativas para automatização dessas atividades a partir da identificação de problemas, prospecção de soluções e os benefícios advindos das possíveis

soluções, gerando portanto, a partir dessas discussões potenciais projetos que foram alocados em equipes de desenvolvimento, responsáveis por realizar um levantamento a respeito das tecnologias mais adequadas à criação das respectivas soluções de modo a permitir a continuidade do projeto após o encerramento da parceria UFBA/TRE-BA.

Inicialmente foram consideradas as linguagens de programação JavaScript em conjunto com o framework React e a linguagem de programação Java juntamente com o *framework* Spring Boot, para a implementação dos métodos e da interface de autenticação de aplicações ao considerar as tecnologias conhecidas pelos residentes. Entretanto, ao analisar as tecnologias empregadas no TRE-BA, optou-se pelo uso da linguagem de programação Java e o *framework* Spring Boot.

Uma vez definidas as tecnologias a serem utilizadas na aplicação, foi iniciado o planejamento das entregas e *sprints*, que são janelas de tempo em que a equipe trabalha em conjunto para concluir um incremento ou nova funcionalidade. Foram estabelecidas *sprints* mensais, porém a partir da terceira *sprint* essa janela foi ajustada para intervalos quinzenais.

Durante a etapa de concepção, foram discutidas as ideias principais, definidas as tecnologias a serem utilizadas e a delimitação do escopo do sistema. Após a concepção realizou-se o levantamento a análise de requisitos, prospectando as necessidades do cliente quanto aos métodos a serem implementados e seus atributos. Os métodos solicitados foram então codificados segundo as definições prévias e para cada novo método implementado foram realizados testes manuais pela equipe de desenvolvedores e, então, validados ou alterados, a fase de conclusão constituiu a reunião de entrega do *backlog*.

Os testes foram realizados de forma local através da aplicação Postman, que é bastante conhecida no mercado e que permite realizar requisições HTTP como cliente de um serviço REST, enviar e receber objetos JSON e verificar a integridade das respostas enviadas pela API, para cada método implementado, os testes deveriam avaliar se os dados retornados eram compatíveis ao esperado.

É importante ressaltar que a documentação foi parte integrante de todas as etapas do desenvolvimento, uma vez que para todas as fases do projeto foram elaborados artefatos que dissertam a respeito das decisões de negócio e suas motivações. Além disso, foi utilizada a plataforma Git para controle e versionamento da solução e o gerenciador GitLab através de um servidor local do TRE-BA para o registro do andamento de todas as atividades relativas ao desenvolvimento, bem como as atas das reuniões de *sprint*.

3.2. Arquitetura

Na construção da API do TRE-BA, foi utilizado o modelo arquitetural REST através do *framework* Spring e a linguagem de programação Java. Também foi utilizado o protocolo HTTP para a comunicação entre cliente/aplicações e servidor/banco de dados.

Para o gerenciamento do projeto foi utilizado o Apache Maven, uma ferramenta de gerência e compreensão de projetos Java, que utiliza um arquivo XML, chamado `pom.xml`, baseado no conceito POM (*Project Object Model*, em português Modelo de Objeto de Projeto) onde ficam armazenadas as informações a respeito do projeto, como sua versão, suas dependências, módulos externos e *plugins*.

Através do Maven várias ferramentas foram adicionadas ao projeto de forma dinâmica para otimizar o desenvolvimento do sistema. Dentre essas ferramentas está o *Actuator*, que é responsável pelo monitoramento da API e de seus *endpoints*, fornecendo informações importantes a respeito da disponibilidade dos recursos.

O projeto foi modularizado em nível de classes em pacotes, designando a cada pacote do projeto as classes de mesma responsabilidade. Isto favorece a modificabilidade e escalabilidade, facilitando a identificação dos recursos por parte dos indivíduos que futuramente trabalharão na ampliação da ferramenta.

Dada a natureza das informações contidas no SGRH, o processamento deveria ser realizado de forma que toda a informação sobre a natureza dos dados fosse de responsabilidade exclusiva do servidor, o que simplifica a implementação do lado cliente. Para garantir a proteção da lógica de negócio, adicionando uma abstração ao acesso dos dados do SGRH, foi utilizado como padrão de projeto o *Data Transfer Object* (DTO, em português, Objeto de Transferência de Dados), garantindo que o cliente tenha acesso estrito às informações necessárias solicitadas através dos métodos.

O padrão DTO é utilizado principalmente quando se quer realizar transferência entre camadas sem que haja exposição da camada de persistência. O objetivo é permitir que os dados estejam disponíveis mesmo após o fim da conexão, evitando chamadas múltiplas ao servidor. O objeto DTO se encarrega de transportar todos os dados necessários serializados entre as camadas em uma mesma chamada. É possível observar através da Figura 4 - Diagrama de funcionamento da API-SGRH. de que forma os objetos da API se relacionam ao receber requisição.

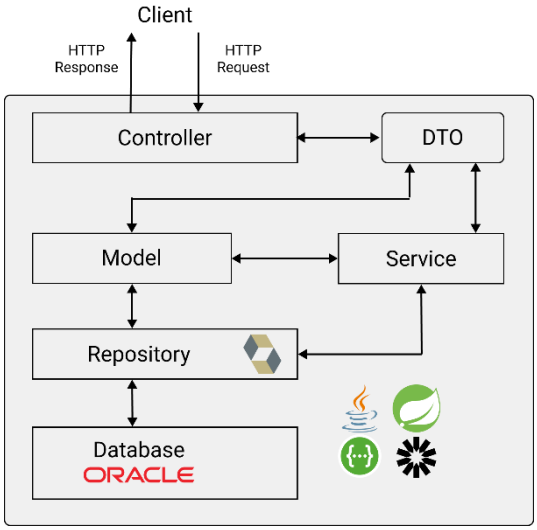


Figura 4 - Diagrama de funcionamento da API-SGRH.

Em um modelo de arquitetura em camadas, as classes organizadas em pacotes representam os níveis hierárquicos das camadas de um sistema. Após o *Client* enviar uma requisição, o *Controller* recebe a solicitação e então, através de um objeto DTO, passa as informações ao *Model* ou *Service* que possuem a lógica de negócio e são entidades responsáveis por solicitar a busca da informação no banco de dados. Essa busca é feita através de uma interface *Repository* que encapsula a lógica de acesso aos dados e é responsável pela persistência da API.

Os métodos implementados na API são majoritariamente métodos de consulta de informações como podemos ver através da Tabela 1 - Lista de métodos implementados na API., dado que o SGRH é uma base de dados sensível e de acesso controlado e, uma vez que o objetivo da API é fornecer um recurso que permita o acesso aos dados por múltiplos módulos sem o acesso direto ao banco, não foram implementados métodos de alteração ou inserção de dados diretamente na base do SGRH.

Todas as escolhas relacionadas à arquitetura do sistema foram tomadas em conformidade com o TRE-BA, levando em consideração quais metodologias e tecnologias que mais se adequassem ao projeto e ao que já era utilizado pela instituição para que a manutenção e evolução do software pudesse ser realizada por colaboradores futuramente.

Tabela 1 - Lista de métodos implementados na API.

URI	Método HTTP	Descrição
/servidorAfastamento/{matricula}	GET	Retorna as informações de afastamento
/servidorAfastamento/férias/{matricula}	GET	Retorna as informações de férias
/servidorAfastamento/listarPeriodosCapacitacao/{matricula}	GET	Lista os períodos de licença capacitação de um servidor
/servidorAfastamento/listarServidoresEmCapacitacao/{matricula}	GET	Lista todos os servidores em licença capacitação
/atualizaDadosPessoais	POST	Solicita atualização de dados pessoais e bancários
/comissões/{matricula}	GET	Retorna os dados de comissões que servidor tenha participado e período
/servidorContraCheque/{nr_cpf}	GET	Retorna os dados de contracheque. Pode receber "mês_ano_folha" como parâmetro
/consultaDIRF/{nr_cpf}	GET	Retorna dados de Declaração de Imposto de Renda
/servidor/{titulo}	GET	Retorna título, matrícula, nome completo e email
/servidor/aniversario/{matricula}	GET	Retorna data de aniversário
/servidor/bancoHoras/{matricula}	GET	Retorna banco de horas
/servidor/contato/{matricula}	GET	Retorna dados de contato
/servidor/dadosPessoais/{matricula}	GET	Retorna dados pessoais e bancários
/servidor/dependentes/{matricula}	GET	Retorna dependentes de um servidor
/servidor/foto/{matricula}	GET	Retorna matrícula e foto
/servidor/gestorComissionamento/{matricula}	GET	Retorna comissionamentos
/unidades/gestor/{cod_unidade}	GET	Retorna dados de gestores de uma unidade
/unidades/listar	GET	Lista todas as unidades

3.3. Tecnologias utilizadas

3.3.1. Spring Framework

Um *framework* é um conjunto de classes e códigos genéricos já implementados a fim de tornar o processo de desenvolvimento mais ágil (Monago Ruiz, 2019). Atualmente existe uma infinidade de *frameworks* para as mais diversas linguagens de programação no mercado. Dentro deste universo o Spring é um dos mais conhecidos para a linguagem de programação Java, que fornece um modelo simples e abrangente de configuração para o desenvolvimento de sistemas independentemente da plataforma de implantação utilizada (Spring, Spring Framework Overview, 2021).

Sendo um projeto de código aberto, o Spring possui uma comunidade bastante ativa e colaborativa que fornece *feedback* contínuo para situações reais, auxiliando na evolução do *framework*. Além disso o framework é dividido em módulos; dessa forma, os desenvolvedores podem escolher integrar a suas aplicações as funcionalidades que mais estão de acordo com seu projeto, usando módulos como Spring Boot, Spring Security, Spring Data, Spring Cloud, Spring Batch, dentre outros disponíveis.

As anotações do Spring tornam o desenvolvimento de aplicações mais dinâmico ao tempo que mantêm o código limpo. Foram introduzidas a partir da versão 2.5, que até então utilizava as configurações dos componentes via XML. Com as anotações, a configuração da conexão dos componentes passa a ser feita diretamente na classe, método ou declaração de campo.

3.3.2. Spring Boot

Spring Boot é uma camada de personalização construída sobre o *framework* Spring que ajuda na criação de aplicações autônomas e de nível de produção com as configurações básicas necessárias para que se possa executar o projeto, necessitando que o desenvolvedor apenas selecione quais módulos pretende usar.

Os princípios fundamentais nos quais o Spring Boot se baseia são fornecer uma experiência inicial radicalmente mais rápida e amplamente acessível para todo o desenvolvimento do Spring; ser opinativo quanto à configuração dos projetos, mas flexível e acessível para mudanças quando os requisitos divergirem dos padrões; fornecer uma gama de recursos não funcionais comuns a grandes classes de projetos e, por fim, gerar absolutamente nenhum código e nenhum requisito para configuração XML (Spring, Introducing Spring Boot, 2021)

3.3.3. Spring Security

O Spring Security é um módulo do ecossistema Spring criado para gerenciar autenticação, autorização e controle de acesso em aplicações Java além de fornecer mecanismos de proteção contra alguns dos ataques mais comuns a sistemas web.

No projeto da API, foi utilizado para gerenciar a autenticação de clientes e autorização para o uso dos métodos, garantindo controle de acesso, impedindo, portanto, que as aplicações possam acessar mais que as informações necessárias aos seus requisitos.

A autenticação foi implementada via LDAP e via *token* de acesso. LDAP (*Lightweight Directory Access Protocol*, em português, Protocolo de Acesso a Diretório Leve) é uma forma de autenticação comum em organizações como um repositório central para informações de usuário. O Spring Security utiliza as informações de usuário e senha fornecidas para validar o acesso de usuários à interface de cadastro e autorização de aplicações que farão uso dos métodos da API, a qual foi criada pelo setor de soluções corporativas do TRE-BA para simplificar o cadastro de novas aplicações, gerando um BearerToken único da aplicação para que possa utilizar a API.

Bearer Authentication ou Autenticação do Portador é um esquema de autenticação HTTP que envolve *tokens* de segurança. Esses *tokens* são *strings* criptografadas, geradas pelo servidor em retorno de uma solicitação de login e que deve ser enviado no cabeçalho de cada requisição para conceder acesso aos recursos.

A *string* enviada através do esquema de autenticação do portador trata-se de um JWT (*JSON Web Token*), um objeto JSON que carrega informações de acesso codificadas e que podem ser validadas pois são assinadas. Um JWT possui um formato "*header.payload.signature*", sendo o *header* um JSON com informações sobre o tipo de *token*, o *payload* um JSON com as informações do usuário ou aplicação autenticada e a *signature* é a concatenação dos *hashes* em `base64UrlEncode` gerados a partir dos objetos anteriores, com uma chave secreta ou certificado RSA. Essa *signature* é utilizada para impedir alterações garantindo a integridade do *token* (Jones, Bradley, & Sakimura, 2015).

Cada método da API possui um identificador único gerado a partir do Swagger, o *framework* de documentação de API. Esse identificador é passado através da anotação @PreAuthorize, que pro-

cessa a solicitação de acesso ao método, verifica se a aplicação tem autorização para a utilização do método, respondendo à requisição com os dados solicitados (Figura 5 - Diagrama de autorização de uso dos métodos da API-SGRH.).

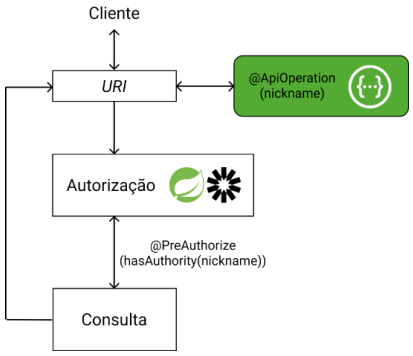


Figura 5 - Diagrama de autorização de uso dos métodos da API-SGRH.

3.4. Documentação e uso do Swagger

A documentação é essencial para que as equipes possam realizar manutenções nos sistemas, o processo de documentar pode ser muito demorado e manter a documentação sincronizada com código é um desafio que se busca superar no desenvolvimento de sistemas. Uma alternativa para a solução deste desafio é a utilização de ferramentas que possam documentar de forma dinâmica e automatizada a partir do código já implementado, facilitando a criação de elementos que servirão de referência às futuras equipes de desenvolvimento da API, para tanto foi utilizado o *framework* Swagger.

O Swagger é um *framework* de documentação de API, que documenta a partir dos elementos das classes *controller* da API criadas a partir do Spring *framework*. A ferramenta utiliza as anotações adicionadas aos *endpoints* dos métodos para coletar as principais informações a respeito do sistema. A partir dessas informações coletadas são criados dois recursos automaticamente: o `swagger-ui.html`, que é a parte visual do Swagger e `/v2/api-docs`, um arquivo JSON (Figura 6 - Arquivo JSON de documentação da API gerado com Swagger.) com o qual a parte visual é gerada e que contém os elementos citados por De (2017) como essenciais para a eficácia de uma documentação de API (De, 2017).

Na Figura 6 - Arquivo JSON de documentação da API gerado com Swagger., podemos observar uma parte do arquivo `/v2/api-docs` gerado pelo Swagger com alguns elementos de documentação gerada como o título, *endpoint* da API e dos métodos, métodos HTTP, parâmetros da URL, carga útil da mensagem, parâmetros de cabeçalho da requisição, códigos de resposta e de erro, necessários para uma documentação eficaz (De, 2017).

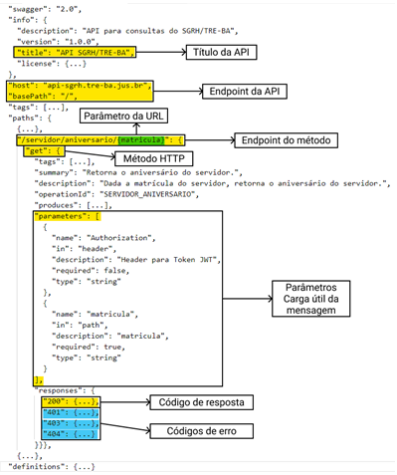


Figura 6 - Arquivo JSON de documentação da API gerado com Swagger.

A partir do arquivo JSON, uma página web, que permite a interação do usuário, é criada. Nessa página constam todas as informações a respeito da API, suas especificações, métodos, descrição e observações úteis, também é possível testar requisições a todos os métodos no mesmo local clicando no botão "*Try it out*". A interface de usuário da API também conta com exemplos de requisição JSON em um corpo com as informações "chave:tipo do valor", como é possível observar através da Figura 7 - Interface visual da documentação da API gerada com Swagger..

A configuração do Swagger é feita através de uma classe no projeto que informa quais pacotes devem ser analisados para a geração da documentação além de conter todas as informações gerais do projeto como título, versão, descrição e licença. Os *endpoints* são definidos através da anotação @Api que é utilizada para declarar um recurso Swagger na API, ou seja, essa anotação informa ao Swagger quais classes devem verificadas no momento da criação da documentação. Para cada método da classe é utilizada a anotação @ApiOperation que é utilizada para declarar uma combinação entre URI e um método HTTP e apenas os métodos que contenham essa anotação serão adicionados à documentação gerada.

Alguns parâmetros podem ser adicionados à anotação @ApiOperation. No projeto da API, foram utilizados os parâmetros *value*, *notes* e *nickname*. O *value* deve ser uma breve descrição do método (preferencialmente até 120 caracteres), enquanto o *notes* permite uma quantidade maior de informações sobre o método. O parâmetro *nickname* serve como um identificador único para o método e foi utilizado pelo Spring Security para validar a autorização de uso do método por parte da aplicação que estiver consumindo a API.

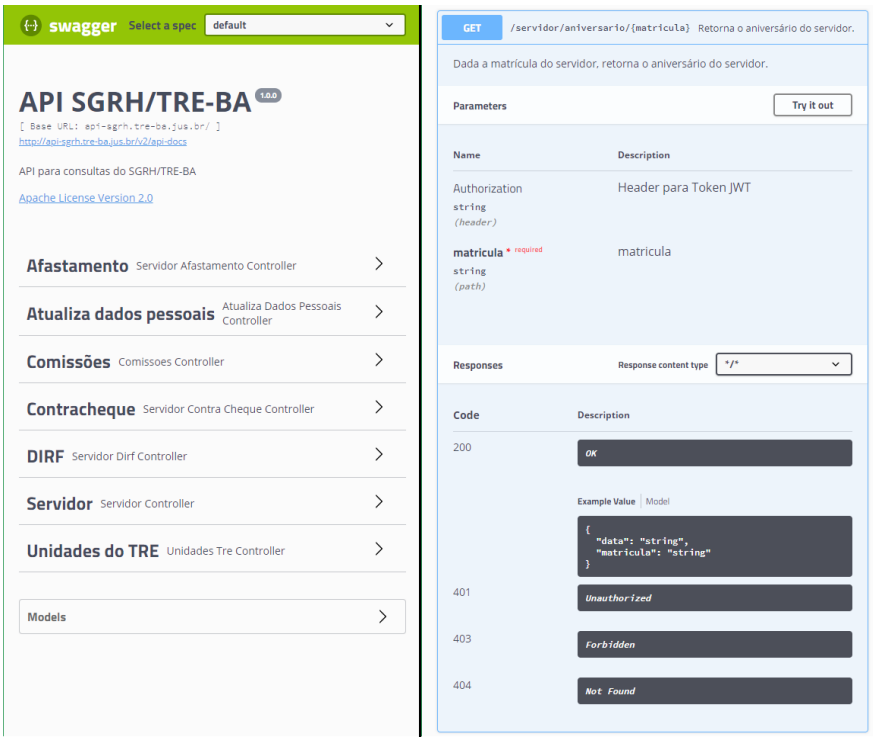


Figura 7 - Interface visual da documentação da API gerada com Swagger.

O uso do Swagger trouxe uma nova abordagem de documentação, sendo ela uma ferramenta dinâmica, tem potencial de promover a evolução da documentação em conjunto com a evolução da API, além de oferecer uma interface interativa e didática com todas as informações essenciais para a utilização do sistema.

4. Avaliação

Para realizar a avaliação da qualidade da API do SGRH enquanto sistema computacional foram utilizadas métricas inspiradas pelo *framework* AgiTSE e o SonarQube como ferramentas de investigação de código-fonte, vulnerabilidades de segurança, redundância de código e cobertura de testes.

O AgiTSE é um framework de desenvolvimento de software, disponibilizado e mantido pelo Tribunal Superior Eleitoral (TSE), que propõe a implantação da cultura ágil nos processos de construção de software. O AgiTSE estabelece métricas para a avaliação da qualidade dos sistemas computacionais desenvolvidos, dentre elas, qualidade do código-fonte, cobertura de testes, funcionalidade e manutenibilidade.

A cobertura de testes é uma medida para indicar a proporção testada do código-fonte de um *software*. Para cada método desenvolvido da API, testes manuais foram realizados pela equipe de desenvolvimento e posteriormente validado pela equipe do TRE-BA. A manutenibilidade diz respeito a facilidade de um software ser modificado, adaptado e melhorado. Em conformidade com este requisito, a API possui todas as suas etapas e processos documentados no GitLab, além disso, a documentação dinâmica implementada com o Swagger auxilia no entendimento do sistema. A funcionalidade descreve a capacidade do software atender as necessidades explícitas e implícitas do sistema, tendo seu desenvolvimento realizado sob demanda e validação a cada sprint, a avaliação da funcionalidade foi realizada em paralelo ao desenvolvimento da API.

A avaliação da qualidade do código-fonte foi realizada com o auxílio da ferramenta SonarQube, uma plataforma de código aberto para inspeção da qualidade do código, revisões automáticas e análise estática do código. Após a execução, o SonarQube retorna algumas avaliações de métricas quanto a qualidade do software.

A API obteve nota “A” nos quesitos Bugs, que avalia a ausência de erros no código, e em *Code Smells*, que avalia a falta de problemas relacionados à manutenção do software. O valor de 0% em *Coverage* está relacionado a ausência de testes automatizados na API. Observou-se, após análise, que o valor de 29% em *Duplications* era composto majoritariamente por trechos *getters* e *setters* da linguagem Java. As métricas Vulnerabilities e *Hotspots Reviewed* estão relacionadas à segurança, e obtiveram uma baixa avaliação (“D” e “E”, respectivamente), pois o SonarQube identificou que o *Cross Site Request Forgery* (CSRF) está desabilitado no Spring Security, isto ocorre pois, a API propõe-se a ser utilizada apenas por sistemas internos e, segundo a documentação do Spring Security, se o serviço pretende ser utilizado apenas por clientes que não são navegadores, recomenda-se sua desativação (Spring, When to use CSRF protection, 2021).

O aspecto *Quality Gates* impõe uma política de qualidade respondendo a uma pergunta: o projeto da API está pronto para ser lançado? Nesse sentido o SonarQube nos traz uma avaliação positiva através do status “*Passed*”, indicando que o projeto está pronto para implantação.

Atualmente o TRE-BA possui 112 sistemas em seu catálogo, sendo 98 sistemas classificados como “em uso”, “em fila”, “em desenvolvimento” ou “em aquisição”. Dos 98 sistemas em funcionamento ou desenvolvimento, 48 sistemas usam atualmente o SGRH e irão ser adaptados para utilização da API no futuro.

5. Conclusão

A documentação dinâmica da API, bem como a interface *web* interativa gerada a partir do Swagger, ofereceu uma alternativa ao modelo convencional de documentos externos ao código que correm o risco de não evoluir com o *software*, resultando em documentações desatualizadas, que não representam o sistema em seu estado atual, dificultando o processo de manutenção e correção de *bugs*.

A partir da análise realizada com a ferramenta SonarQube foi possível observar os pontos fortes da API, a serem preservados, e os elementos que podem ser reavaliados e melhorados nas próximas

versões do sistema. Uma vez que o software possui documentação e que a equipe que dará continuidade também esteve presente em todas as etapas de desenvolvimento, espera-se que as manutenções possam ser realizadas sem grandes obstáculos.

Como perspectiva para o futuro, propõe-se a revisão do sistema de forma a obter uma melhor avaliação tendo como norteador as métricas do SonarQube. Além disso, a criação de um documento ilustrado com um passo-a-passo da utilização da interface interativa do Swagger que, apesar de intuitiva, deve ser considerada acessível para todos os indivíduos.

Referências

Ambler, S. (2021). Agile/Lean documentation: strategies for agile software development. Agile/Lean documentation: strategies for agile software development.

Bray, T. (2017). The JavaScript Object Notation (JSON) Data Interchange Format. The JavaScript Object Notation (JSON) Data Interchange Format. RFC Editor. doi:10.17487/RFC8259

Davelaar, S. (2015). Performance Study–REST vs SOAP for Mobile Applications. Oracle: A-Team Chronicles.

De, B. (2017). Api Documentation. Em B. De, Api Management: An Architect's Guide to Developing and Managing APIs for Your Organization (pp. 59–80). Springer.

ECMA. (2017). Standard ECMA-404 The JSON Data Interchange Syntax. Standard ECMA-404 The JSON Data Interchange Syntax.

Fielding, R. T. (2000). Architectural styles and the design of network-based software architectures. Doutorado em Filosofia, University of California, Irvine.

Jacobson, D., Brail, G., & Woods, D. (2012). APIs: A strategy guide. "O'Reilly Media, Inc.".

Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). JSON Web Token (JWT). RFC Editor. doi:10.17487/RFC7519

Medina, E. A. (1984). Some aspects of software documentation. Proceedings of the 3rd annual international conference on Systems documentation, (pp. 57–59).

Monago Ruiz, A. (2019). Servicio Web API REST sobre el Framework Spring, Hibernate, JSON Web Token y BBDD Oracle. Servicio Web API REST sobre el Framework Spring, Hibernate, JSON Web Token y BBDD Oracle.

Mozilla. (2021). Métodos de requisição HTTP - HTTP: MDN. Métodos de requisição HTTP - HTTP: MDN.

Seabra, M., Nazário, M. F., & Pinto, G. (2019). REST or GraphQL? A performance comparative study. Proceedings of the XIII Brazilian Symposium on Software Components, Architectures, and Reuse, (pp. 123–132).

Spring. (2021). Introducing Spring Boot. Introducing Spring Boot.

Spring. (2021). Spring Framework Overview. Spring Framework Overview.

Spring. (2021). When to use CSRF protection. When to use CSRF protection.

Tribunal Regional Eleitoral da Bahia. (2019). PORTARIA Nº 245, DE 20 DE SETEMBRO DE 2019. PORTARIA Nº 245, DE 20 DE SETEMBRO DE 2019.

Livio de Assis Ara
Tribunal Regional Eleitoral da Bahia
1ª Av. do Centro Administrativo da Bahia, 150 – CAB
Salvador-BA - CEP: 41.745-901 - Brasil
livio.ara@ufba.br

Especificação de um sistema para totalização de Eleições Comunitárias

Resumo. *Desde 1997, a Justiça Eleitoral disponibiliza a urna eletrônica, para que diversas entidades possam realizar as suas eleições. Conhecida como eleição comunitária, a iniciativa tem, dentre as diversas finalidades, a divulgação da urna eletrônica e de seus sistemas associados, bem como, colocar à disposição das instituições um sistema de eleição comprovadamente rápido, seguro e imune a fraudes. Todo o software utilizado nestas eleições é disponibilizado pela Justiça Eleitoral, exceto o sistema de totalização. Assim, o objetivo deste trabalho é desenvolver um sistema de totalização, disponível nas plataformas Web e Android, com a capacidade de ler os resultados das seções através dos QR Codes e disponibilizá-los em site específico.*

Abstract. *Since 1997, the electoral justice provides its voting machines for several organizations to hold their elections. Known as community elections, the initiative has, among others, the objective of promoting the voting machines and its softwares, as well as, make an undoubtedly fast, safe and tamperproof election system available to organizations. All software used in community elections are provided by The electoral justice, except the totalization software. Thus, this work's aim is to develop a totalization system, available on web and Android platforms - able to read the QR Codes on electoral sections results reports and publish them on a particular site.*

1. Introdução

O voto eletrônico tem sido utilizado no Brasil desde 1996, através de dispositivos especiais criados e desenvolvidos pela Justiça Eleitoral, chamados de urnas eletrônicas. Seu projeto teve início em 1995, quando o Tribunal Superior Eleitoral (TSE) formou uma comissão técnica liderada por pesquisadores do Instituto Nacional de Pesquisas Espaciais (INPE) e do Centro Técnico Aeroespacial (CTA) para desenvolver o projeto da "máquina de votar". Nestas mais de duas décadas de atividade, a urna eletrônica coletou e apurou os votos de milhões de eleitores em 25 eleições gerais e municipais (contando os dois turnos), com segurança e total transparência [TSE 2021].

> *"Desde 1997, a Justiça Eleitoral disponibiliza a mesma urna eletrônica utilizada nas eleições oficiais, bem como apoio e suporte, para que entidades públicas e privadas, além das instituições de ensino, possam realizar as suas eleições de forma segura e transparente. Conhecida como eleição comunitária, a iniciativa tem a finalidade de treinar mesários, eleitores e o corpo técnico da Justiça Eleitoral fora do período eleitoral, bem como de fazer a divulgação da urna eletrônica e de seus sistemas associados. Além disso, busca colocar à disposição das instituições um sistema de eleição comprovadamente rápido, seguro e imune a fraudes"* [TSE 2021].

O empréstimo da urna eletrônica é regulamentado pela Resolução TSE nº 22.685/2007[1]. De acordo com a norma, as eleições comunitárias podem ser realizadas até 120 dias antes de um pleito oficial ou 30 dias após as eleições municipais ou gerais. O pedido de empréstimo deve ser encaminhado ao juiz eleitoral da cidade onde ocorrerá a eleição. Quando a votação abranger mais de uma zona eleitoral da mesma Unidade da Federação, o pedido deve ser encaminhado ao Tribunal Regional Eleitoral do respectivo estado. No caso de a eleição envolver mais de um estado, o pedido tem de ser encaminhado ao TSE. Vencida esta etapa, sendo o pedido de empréstimo autorizado pelo Juízo competente, a entidade solicitante deverá encaminhar todos os dados solicitados pela Justiça Eleitoral, que serão parametrizados através do sistema Parametrizador, e posteriormente enviados ao Sistema Gerenciador de Dados, Aplicativos e Interface (GEDAI)[2], sendo este o responsável pela geração das mídias necessárias à preparação das urnas eletrônicas. A figura 1 apresenta as etapas da preparação de uma urna eletrônica para eleição.

Figura 1. Processo de preparação das urnas eletrônicas.

O processo eleitoral, basicamente, compreende 03 etapas: preparação, votação e totalização, e utiliza softwares específicos e privativos da Justiça Eleitoral. Todos os programas são desenvolvidos pela Justiça Eleitoral e cedidos para realização das eleições comunitárias, com exceção do sistema de totalização, conforme prevê a Resolução 22.685/2007, para eleições parametrizadas, em seu artigo 11º, *"Art. 11. O sistema de totalização poderá ser elaborado pela requerente ou pela Justiça Eleitoral, mediante sua disponibilidade, sendo necessário, neste caso, estabelecer os critérios e as condições para a sua cessão"*, ou seja, a totalização dos resultados obtidos após o encerramento da votação é de responsabilidade da entidade cessionária das urnas.

Embora exista um sistema de totalização dos resultados desenvolvido pelo TSE para eleições oficiais, o mesmo não é disponibilizado para as eleições comunitárias, tendo em vista que cada eleição possui um formato específico.

2. Objetivo

Embora o voto seja eletrônico, a urna é composta por diversos equipamentos, dentre eles uma impressora térmica, responsável pela emissão de alguns documentos impressos, sendo eles os comprovantes de carga e de testes de componentes, as zerésimas e os boletins de urna. Estes últimos são emitidos com os resultados oficiais da seção eleitoral. A partir das Eleições 2016, o boletim de urna passou a contar com o Quick Response Code (QR Code – código de resposta rápida), que permite a rápida digitalização do resultado apurado em uma seção [TSE 2021].

1 https://bit.ly/3FL5SW7
2 O sistema Parametrizador e o sistema GEDAI são de uso exclusivo da Justiça Eleitoral.

A Figura 2 apresenta o detalhamento de um boletim de urna e seus dados.

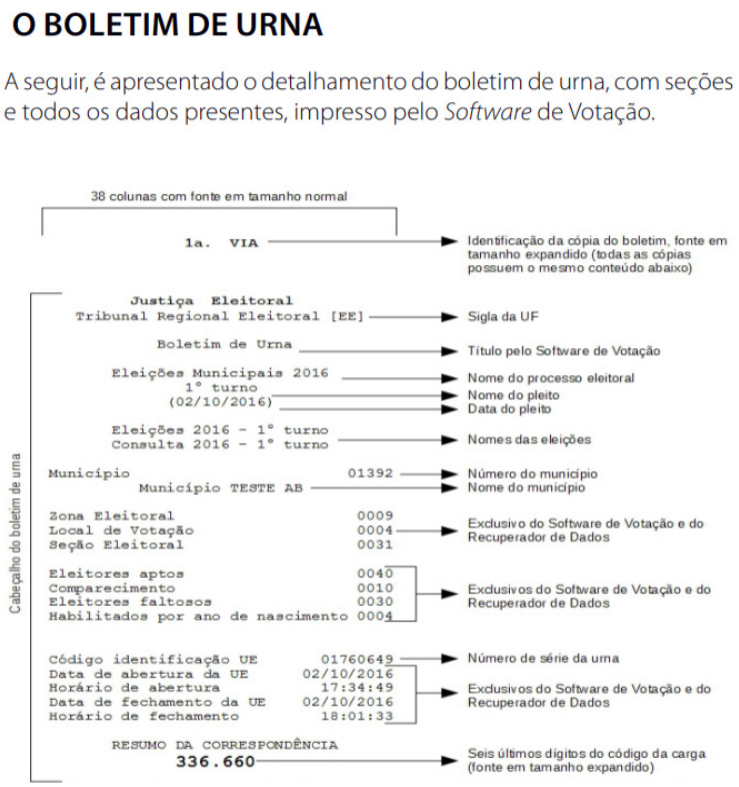


Figura 2. Cabeçalho boletim de urna

Neste contexto, este trabalho tem como objetivo principal especificar os requisitos de um sistema para totalização de eleições comunitárias, que seja capaz de ler QR Codes e armazenar as informações das seções eleitorais em uma base de dados e disponibilizar e resultados em um site específico. A figura 3 apresenta o conteúdo extraído de um boletim de urna eletrônica, através de seu QR Code.

Figura 3. Exemplo de QR Code em um boletim de urna.

3. Metodologia

Este trabalho está fundamentado na tecnologia desenvolvida pela Justiça Eleitoral, através da utilização de QR Codes em seus boletins de urnas, criados para aprimorar o processo de fiscalização das eleições oficiais.

Uma das formas mais antigas de fiscalização é a impressão e publicação do boletim de urna. Encerrada a votação, a urna apura os votos e emite relatório com o resultado oficial da seção eleitoral. Esse relatório é documento público, cuja cópia é afixada no local de votação para que qualquer cidadão possa conferir.

Nos boletins de urnas estão contidos todos os candidatos participantes do pleito que obtiveram votos na seção eleitoral, este boletim é a fonte de informações para a totalização e apuração dos votos de uma eleição comunitária.

Desde as eleições de 2016, o boletim de urna passou a contar com QR Code, tecnologia que permite a rápida digitalização do resultado apurado em uma seção, assim, surgiu a ideia de especificar um sistema, o qual fosse capaz de extrair as informações de forma rápida e confiável desses boletins e transferi-las para um sistema totalizador de votos.

Em um primeiro momento, foi realizado um levantamento, dentro da Justiça Eleitoral, sobre a existência de tal solução. Com a resposta negativa, foram levantadas todas as informações disponíveis sobre a tecnologia do QR Code e também as peculiaridades de totalização de votos de uma eleição comunitária.

Foi desenvolvido um processo com todas as etapas da eleição para a definição das atividades do sistema, além da modelagem de uma base de dados que pudesse dar sustentação às informações da solução.

Por fim, utilizando as especificações da solução contidas neste trabalho, foi desenvolvido um sistema, em colaboração com a OAB, através da empresa Mobtex Softwares, para a realização da totalização dos votos da eleição em 2021.

4. Proposta

A apuração do resultado de uma eleição comunitária é realizada através da soma de todos os dados de votação das seções eleitorais, registrados em seus respectivos boletins de urna, de forma manual. Essas informações são passadas para alguma planilha, onde são totalizados todos os votos.

Uma maneira de auxiliar a captura desses dados dos boletins é especificar um sistema com a capacidade de executar a leitura das informações contidas em um QR Code do boletim [Justiça Eleitoral 2021], através de um dispositivo, e repassá-las de forma automática e digital para um sistema que some e realize a totalização dos votos das seções.

Este sistema deve possuir as funcionalidades de ler os QR Codes dos boletins, separar as informações neles contidas e enviá-las para uma base de dados, onde, outro módulo, com base em informações já cadastradas da eleição, fará a soma e divulgação do resultado.

4.1 Modelagem do sistema

A modelagem de sistema com o auxílio dos diagramas facilita a compreensão das atividades do sistema. São apresentados o Modelo de Banco de Dados (figura 4) e Atividades (figura 5).

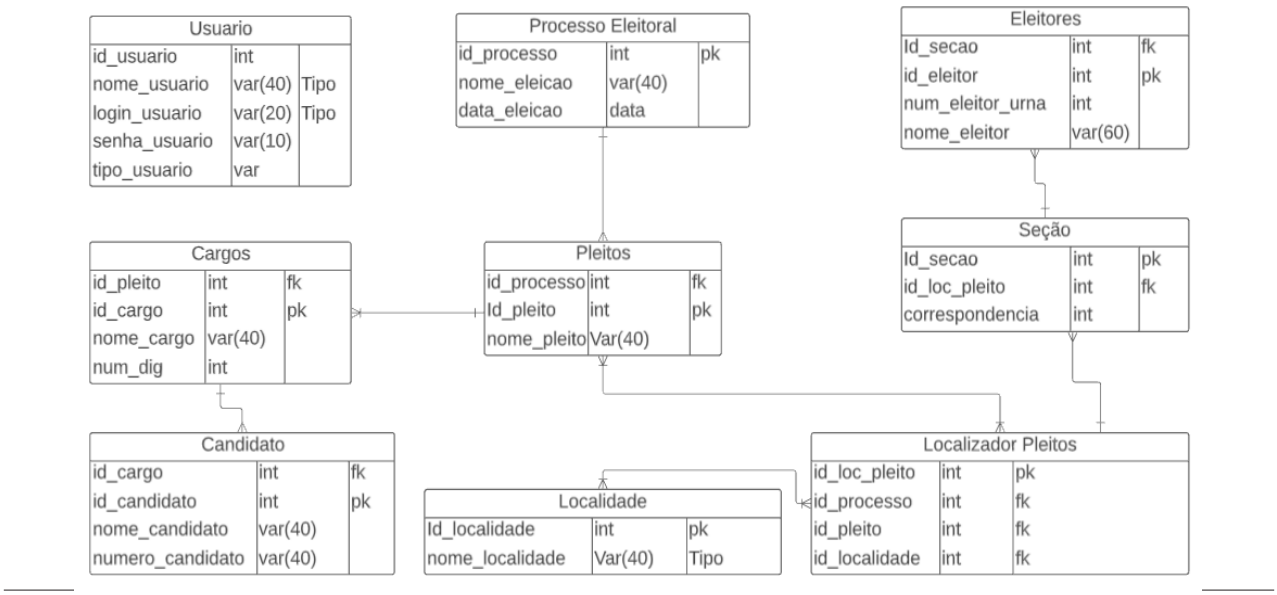


Figura 4 Modelo de banco de dados do sistema para totalização das Eleições Comunitárias .

A figura 5 representa o diagrama de atividades e demonstra o comportamento dinâmico, através do fluxo de controle entre ações executadas pelo sistema. É semelhante a um fluxograma, porém pode exibir fluxos correntes (PRESSMAN, 2016). De acordo com Lima (2011), o diagrama de atividades viabiliza modelar o comportamento do sistema, expondo os caminhos lógicos que um processo pode seguir.

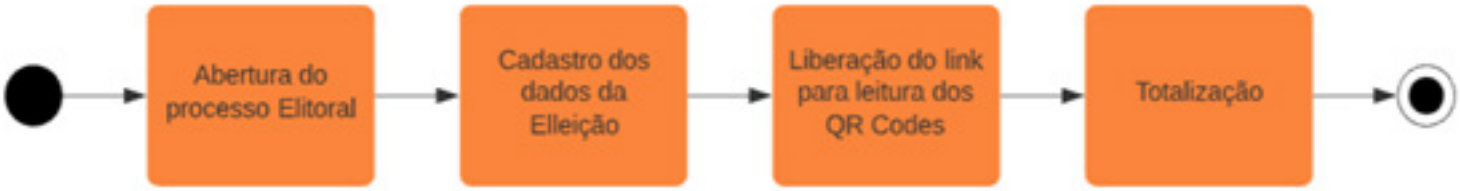


Figura 5. Diagrama de atividades do sistema.

Aprovados todos os dados da eleição por parte da cessionária das urnas, eles deverão ser cadastrados no sistema de totalização. Terminada a etapa do cadastro de todas as informações, deverá ser disponibilizado link para acesso à página ou App, capaz de ler os dados dos QR Codes das seções que farão parte da eleição. Após o encerramento das seções eleitorais, e a disponibilização dos boletins de urnas, o usuário deverá realizar a leitura de todos os boletins, através de seus Qr Codes, enviando os dados de forma online para a aplicação, a qual será capaz de interpretar os dados enviados, salvando em uma base de dados e computando os votos por candidato. O sistema fará a leitura do QR Code do boletim de urna, realizando parsing dos dados, conforme estrutura da Tabela 1, apresentada na seção 5.2.3.

4.2 Módulos do sistema

O sistema de Totalização para Eleições Comunitárias contempla 4 módulos:

- **Cadastro de usuários**: responsável pelo gerenciamento de todos os usuários do sistema, sendo atribuídos os seguintes papéis para os usuários: administrador do sistema e usuário de eleição;
- **Cadastro de Eleições**: responsável por gerenciar todas as informações necessárias para o cadastro das informações necessárias das eleições a serem apuradas;
- **Leitura de QR Code**: deverá ser capaz de ler e interpretar os dados colhidos através de smartphones dos boletins de urnas e salvá-los em base de dados;
- **Totalização:** responsável por realizar a varredura na base de dados correspondente à eleição, realizando o somatório de votos de cada candidato e listado os mesmo por ordem decrescente de votação, ou seja, o candidato mais votado sempre aparecerá em primeiro lugar na divulgação dos resultados.

4.2.1 Módulo de Cadastro de usuários

Este módulo será responsável pelo gerenciamento dos usuários do sistema, sendo os mesmos divididos em dois grupos: administradores e usuários de eleição.

Os administradores poderão:

- Criar usuário
- Deletar usuário
- Modificar a senha de usuário
- Alterar o perfil do usuário
- Criar eleição
- Alterar eleição
- Criar candidato
- Alterar Candidato
- Deletar Candidato
- Criar, alterar e excluir urnas

Os usuários de eleição poderão:

- Inserir os resultados de seções.

4.2.2 Módulo de Cadastro de Eleições

Este módulo será responsável pelo gerenciamento dos dados das Eleições e deverá:

- Criar, alterar e excluir, todas as informações relacionadas a uma eleição, sendo elas:
  - Nome da Eleição
  - Data da Eleição
  - Dados de seções
  - Correspondências
  - Pleitos
  - Nome Candidatos
  - Número Candidato

4.2.3 Módulo Leitura de Resultados

O módulo realizará a leitura das informações, e enviará de forma online, após parsing do QR Code, as seguintes informações:

- PROC:nnnnn
- PLEI:nnnnn
- MUNI:nnnnn
- ZONA:nnnn
- SECA:nnnn
- IDEL:nnnnn
- CARG:nn
- ccccc:nnnn
- HASH:xxxxxx
- ASSI:xxxxxx

Antes de salvar os dados, o módulo fará a conferência se os códigos do HASH e da ASSI, lidos no boletim, são os mesmos cadastrados através do módulo de Cadastro de Eleições, o que confere a segurança necessária a não inclusão de dados não pertencentes àquela eleição. Caso o boletim seja lido e enviado mais de uma vez, o módulo deverá ser capaz de identificar a informação duplicada e avisar que a seção já foi lida.

Tabela 1 Estrutura dados da seção eleitoral contidas no QR Code

Cabeçalho	
QRBU:n:x	Marca de início dos dados. n = índice do QR Code em uma sequência de QR Codes. x = quantidade total de QR Codes
VRQR:n.y	Número da versão do formato da representação do boletim de urna. n = número de ciclos eleitorais desde sua implementação. y = número de revisões do formato dentro de um ciclo.
VRCH:nnnn	Número da versão da chave utilizada para assinar o conteúdo do QR Code.

Conteúdo	
ORIG:xxxx	Origem do boletim de urna (Vota, RED ou SA).
ORLC:xxx	Origem da configuração do processo eleitoral (LEG – eleição legal oficial; COM – eleição comunitária).
PROC:nnnnn	Número do processo eleitoral.
DTPL:aaaammdd	Data do pleito.
PLEI:nnnnn	Número do pleito.
TURN:n	Número do turno (1 – primeiro turno; 2 – segundo turno).
FASE:x	Fase dos dados (O – oficial; S – simulado; T – treinamento).
UNFE:xx	Sigla da UF. No caso de eleição no exterior, a sigla será ZZ.
MUNI:nnnnn	Número do município.
ZONA:nnnn	Número da zona eleitoral.
SECA:nnnn	Número da seção eleitoral.
AGRE:nnnn.nnnn...	Número das seções agregadas separadas por ponto.
IDUE:nnnn...	Número de série da urna.
IDCA:nnnn...	Código de identificação da carga (24 dígitos).
VERS:xxxx...	Texto de tamanho variável com a versão do software da urna (somente números e pontos)
LOCA:nnnn	Número do local de votação.
APTO:nnnn	Quantidade de eleitores aptos.
COMP:nnnn	Quantidade de eleitores que compareceram para votar.
FALT:nnnn	Quantidade de eleitores faltosos.
DTAB:aaaammdd	Data da abertura da urna.
HRAB:hhmmss	Hora da abertura da urna.
DTFC:aaaammdd	Data do fechamento da urna.

HRFC:hhmmss	Hora do fechamento da urna
IDEL:nnnnn	Código da eleição.
CARG:nn	Código do cargo.
TIPO:n	Tipo: 0 – Majoritário; 1 – Proporcional; 2 – Consulta.
VERC:n	Versão do pacote de dados de candidatos/consulta.
ccccc:nnnn	Número do candidato ou resposta, seguido da quantidade de votos que recebeu.
APTA:nnnn	Quantidade de eleitores aptos para votar no cargo.
NOMI:nnnn	Quantidade de votos nominais para o cargo.
BRAN:nnnn	Quantidade de votos em branco para o cargo.
NULO:nnnn	Quantidade de votos nulos para o cargo.
TOTC:nnnn	Total de votos apurados para o cargo.
HASH:xxxxxx...	Hash da seção de conteúdo do boletim. Ao final de cada QR Code, virá um hash cumulativo aos dados de todos os anteriores, o que permite a verificação da leitura em sequência. O cálculo é feito com SHA-512, codificado em hexadecimal.
ASSI:xxxxxx...	Assinatura digital Ed25519 a partir do último hash (incluído somente no último QR Code). Codificado em hexadecimal e também impresso no boletim em papel.

4.2.4 Módulo de Totalização

Este módulo será capaz de ler todas as informações em uma base de dados, listar os candidatos pertencentes à eleição e apresentar os mesmos em ordem decrescente por número de votos.

5.3 Requisitos

A tabela 2 lista os Requisitos Funcionais (RF).

Tabela 2. Requisitos Funcionais do sistema.

Id	Descrição
RF001	Possuir tela de login, com usuário e senha
RF002	Permitir o acesso ao menu respectivo a cada perfil de usuário
RF003	Permitir ao administrador incluir/alterar/excluir usuários do sistema.
RF004	Permitir ao administrador incluir/alterar/excluir data da eleição
RF005	Permitir ao administrador incluir/alterar/excluir nome da Eleição
RF006	Permitir ao administrador incluir/alterar/excluir pleitos
RF007	Permitir ao administrador incluir/alterar/excluir cargos
RF008	Permitir ao administrador incluir/alterar/excluir nome de candidatos
RF009	Permitir ao administrador incluir/alterar/excluir número dos candidatos
RF010	Permitir ao administrador incluir/alterar/excluir correspondência

RF011	Permitir ao administrador incluir/alterar/excluir
RF012	Permitir ao administrador incluir/alterar/excluir
RF013	Permitir ao administrador incluir/alterar/excluir
RF014	Possuir um módulo capaz de realizar a leitura de QR Code através de smartphone
RF015	Realizar parsing dessas informações e inserir os dados dos votos dos candidatos na base de dados.
RF016	O sistema deverá ser capaz de totalizar os votos enviados para a base de dados, através do módulo de QR Code
RF017	O sistema deverá possuir um módulo capaz de somar a quantidade de votos por candidato e ordenando os candidatos em ordem decrescente por número de votos.

A tabela 3 lista os Requisitos Não-Funcionais (RNF).

Tabela 3. Requisitos não funcionais do sistema.

Id	Descrição
RNF001	Somente será acessado por login e senha de usuários cadastrados.
RNF002	Deverá possuir diferentes perfis de usuários, cada um com níveis de privilégio específicos.
RNF003	O sistema deverá ser compatível com diferentes sistemas operacionais e navegadores.
RNF004	Funcionar em ambiente WEB.
RNF005	Funcionar em smartphone
RNF006	Ler QR Code

5. Aplicação Prática

A OAB, entidade representativa da categoria dos advogados em todo território nacional, tem realizado sua eleição para os cargos de direção, em parceria com a Justiça Eleitoral, com a utilização das urnas eletrônicas desde os anos 2000.

Utilizando as especificações contidas neste trabalho, a entidade desenvolveu um sistema para a apuração do resultado da eleição do Presidente da instituição, bem como os diretores de seccionais, eleitos no pleito eleitoral de 2021, para representação da entidade até o ano de 2023.

O sistema foi desenvolvido baseado nas definições propostas neste artigo pela empresa Mobtex Softwares (https://votus.app). A solução poderá não só ser utilizado em todas as eleições comunitárias, como também, através de parametrizações do sistema, ser utilizada como ferramenta para auditoria em eleições oficiais. A modularização do sistema, foi o que permitiu a capacidade da expansão dos recursos do mesmo, possibilitando a multifuncionalidade da aplicação, através de pequenas parametrizações.

O sistema foi desenvolvido com as tecnologias react-native para o front-end e para a leitura do QR Code, optamos por utilizar C++ (android/IOS) e Objective-c (IOS), utilizando componentes de baixo nível. Os boletins de urna são impressos a partir de uma impressora térmica, a qual faz parte da composição do hardware da urna, o que, por vezes, pode gerar uma imagem do QR Code com uma qualidade inferior ao desejado, além de que, muitas vezes após ser manuseado por mesários e integrantes da seção eleitoral, não estão em sua melhor apresentação. Assim, resolvemos optar pela

aplicação nativa nas plataformas dos smartphones, ao invés de um módulo WEB com código padrão do react-native rodando no smartphone, o que resultou no aumento de performance e acurácia na leitura.

Esta eleição foi composta pelos cargos de Presidente de Seccional e Diretor de Subseção, em 37 municípios do Estado da Bahia, com 54 candidatos e 54423 eleitores, distribuídos por 104 seções eleitorais e utilizadas 164 urnas entre seções e reservas.

No município de Salvador os eleitores votam apenas para o cargo de Presidente, enquanto nos demais a eleição é composta de ambos os cargos. Dessa forma, na prática, existem 38 eleições, sendo 37 para Diretores de Subseções e uma para Presidente de Seccional. Assim, o sistema foi ajustado para totalizar as informações por Candidato/Subseção e também somar todos os votos de todos os municípios para o cargo de Presidente de Seccional.

A figura 6 apresenta as telas de login e de funcionalidades do sistema no leitor de QR Code.

Figura 6. Capturas de telas do sistema: (a) tela de login e (b) menu inicial

6. Considerações Finais

Este trabalho apresentou as especificações de um sistema capaz de realizar a apuração de resultados de eleições comunitárias, através da leitura dos QR Codes disponibilizados com os boletins de urnas. Adicionalmente foi desenvolvido pela OAB, utilizando as especificações deste trabalho, um aplicativo piloto para apuração dos votos e divulgação do resultado das eleições desta organização em novembro de 2021.

Como trabalho futuro, pretende-se ampliar o aplicativo desenvolvido para a OAB, criando uma plataforma que tenha a capacidade de ser parametrizada para qualquer outra eleição onde a Justiça Eleitoral seja solicitada a participar com o empréstimo de urnas eletrônicas, além de ferramenta para auxílio na auditoria das eleições oficiais.

Referências

[Lima 2011] LIMA, Adilson da Silva. UML 2.3 do requisito à solução. 1° edição. São Paulo: Érica, 2011.

[Sommervile, I. 2011] Ian Sommerville. Engenharia de Software. 9ª ed. São Paulo, SP: Pearson, 2011

[TSE 2021] Portal do Tribunal Superior Eleitoral, disponível em http://tse.jus.br. Último acesso em Novembro de 2021.

[Justiça Eleitoral 2021] Justiça Eleitoral, QR Code no Boletim de Urna Boletim de Urna Manual para criação de aplicativos de leitura. Disponível em https://www.tse.jus.br/eleicoes/eleicoes-2018/qr-code-no-boletim-de-urna-manual-para-criacao-de-aplicativos-de-leitura. Último acesso em Novembro de 2021.

Lucas de Andrade Lima
Universidade Federal da Bahia - Instituto de Computação Departamento de Ciência da Computação
Salvador, BA – Brasil
lucas@tre-ba.jus.br

Estudo sobre a implementação de BI no TRE-BA

Abstract. *Due to a high amount of service orders for the assembly of data re- ports on TRE-BA, the opportunity presented by the residency was used to applyBusiness Intelligence through personalized dashboards, these being measured posteriorly.*

Resumo. *Devido ao elevado número de pedidos similares para a montagem de relatórios no TRE-BA, utilizou-se da oportunidade apresentada através da residência para a aplicação de Business Intelligence através de painéis perso- nalizados para os problemas de cada setor, com os mesmos sendo avaliados posteriormente.*

1. Introdução

No TRE-Ba, há uma grande quantidade de ordens de serviço para criação de relatórios de dados, vindo de múltiplos setores que resulta em uma sobrecarga no Setor de Banco de dados (SEBDA). Considerando que esses pedidos eram similares entre si quando vindo de um mesmo setor do TRE e que existia uma periodicidade na abertura desses pedidos,julgou-se que a aplicação de Business Intelligence seria uma solução válida para este problema.

Como parte da Residência em TI, foi decidido que cada dupla de alunos iria focar em um ramo de tecnologia e então implementá-la de forma a melhorar a realidade do TRE, estes ramos sendo decididos através de reuniões com funcionários relatando suas dificuldades do dia-a-dia em reuniões com os responsáveis pelo projeto.

Assim, o projeto teve por objetivo desenvolver uma aplicação de Business Intelli- gence para minimizar o problema da criação de relatórios, diminuindo a necessidade da geração de ordens de serviço, dando autonomia aos usuários para montar seus próprios relatórios em painéis personalizados para suas necessidades de rotina.

O presente trabalho está organizado em seções.A seção 2 descreve a fundamentação teórica necessária para compreensão do método proposto. A seção 3 apre- senta os Trabalhos Relacionados. A seção 4 apresenta o método proposto desenvolvido neste trabalho. A seção 5 descreve os experimentos e o resultado. E por fim, a seção 7 conclui o trabalho e apresenta alguns direcionamentos de pesquisa.

2. Referencial Teórico

Com o crescimento massivo de informações geradas e coletadas na última década, o estudo de ferramentas para utilização dessas informações foi intensificado em diversas linhas de estudo. Devido a conectividade e alta-velocidade que são demandandas em empresas e organizações, decisões não podem mais esperar por um grande período de tempo enquanto dados são coletados e analizados, sendo necessários sistemas de auxílio a tomada de decisão.

Sistemas de suporte à decisão estão no mercado desde a década de 60, quando notou-se que a capacidade computacional era capaz de realizar pequenas tarefas automa-tizadas. Com essas modelagens de dados, a definição deste tipo de sistema pôde ser cunhada como todo sistema que: "ajuda tomadores de decisão utilizando tecnologias de comunicação, dados, documentos e modelos para identificar e resolver problemas" [Hedgebeth 2007]. Atualmente, o maior expoente desse tipo de sistema são aplicações de Business Intelligence.

2.1. Business Intelligence (BI)

Inteligência de negócio, ou como é conhecida comumente, Business Intelligence (BI), são sistemas que através da junção de dados de diferentes formatos e fontes em depósitos massivos de informação, utilizando de ferramentas análiticas para através de processa- mento, gerar relatórios que permitem o acompanhamento do mercado ou instituição em operação, muitas vezes através de interfaces gráficas [Djerdjouri 2020].

As partes que formam um projeto de BI podem expressos através dos seguintes passos, tecnicamente: extração, transformação, gerenciamento e análise de dados, com o intuito de dar suporte ao processo de tomada de decisão. [Negash and Gray 2008]. A extração compreende a formação e utilização de uma Data Warehouse, a transformação é feita através de um processo conhecido como ETL e a visualização, que comporta tanto o gerenciamento quanto a análise, é feita por *dashboards* de monitoramento.

Data Warehouses são sistemas que buscam através da sua arquitetura unir informações oriundas de um ou mais bancos de dados, além de outras fontes diversas, como planilhas e afins[Ferreira et al. 2010], devido a` alta heterogenia entre os dados, di- ficultando uma normalização geral. A estrutura de um Data Warehouse é dependente do projeto que é alimentado pelo mesmo, podendo ser subdividido em partições menores que são focadas em dados que possuam uma correlação, como sua origem ou finalidade.

O processo de ETL (Extract, Transform, Load) pode ser feito já dentro da arquitetura do Data Warehouse, e trata das manipulações de dados brutos para a obtenção de informações. Depois de tratados, a etapa de visualização é feita através de softwa- res na maioria das vezes, se utilizando de ferramentas gráficas para representação das informações adquiridas, de forma que através dos relacionamentos entre os dados, permi- tindo que organizações possuam novas visões claras sobre o funcionamento interno das mesmas e assim gerando decisões mais embasadas em fatos.

3. Trabalhos Relacionados

A aplicação de BI tem se expandido para múltiplos ramos, desde a otimização de como recursos são aplicados, aumentando a capacidade de produção em múltiplos níveis em indústrias até a análise de como campanhas de *marketing* estão funcionando. Como exem- plos, serão utilizadas a campanha da presidência de Barack Obama em 2012 e o uso de BI para melhora de decisão em uma seguradora.

Durante campanhas, o uso de análise de dados para a montagem de perfis de eleitorado é algo amplamente utilizado na atualidade, porém, nessa campanha de 2012, utilizou-se de dados analíti-

cos para determinação de como se aproximar de eleitores para passar a mensagem mais efetivamente, ao invés de um modelo para determinar quais elei- tores ainda estão indecisos[Scherer 2012]. Através desse novo modelo, pôde-se definir quais m´ıdias seriam mais efetivas em levar os grupos indecisos a votar e assim gerenciaros recursos, destinando-os aos locais corretos, diminuindo um custo considerável. Ainda no âmbito de gerenciamento de recursos, a seguradora EM utilizou da montagem de um sistema BI para facilitar a análise de alegações, permitindo a manutenção correta de um fundo monetário de reserva através de um modelo preditivo capaz de inferir quais apólices resultariam em algo negativo [Microsoft 2019].

4. BI para o TRE

Devido ao escopo apresentado, uma decisão foi tomada para trabalhar com cada setor in- dividualmente, criando um modelo de solução personalizado e relevante para problemas encontrados na rotina, através de *dashboards* interativos. As soluções deveriam sempre dar autonomia para as sessões do TRE conseguirem visualizar os dados relevantes sem a necessidade da abertura de requisições à outros setores. Antes da implementação do Busi- ness Intelligence, os setores faziam consultas diretas ao banco de dados através do Atena, sistema online para realização de acessos simples ao conjunto, porém sem interatividade de tempo real e sendo necessário um conhecimento básico de SQL por parte dos usuários. Como exemplo, a gestão do estoque de processos e a coleta de dados estatísticos para avaliações era uma tarefa árdua, devido a necessidade de abertura de múltiplos chamados para a Seção de Bancos de Dados. E, embora existiam alguns projetos menores de BI já implementados, os mesmos não tinham o escopo do projeto realizado pelos residentes, sendo limitados apenas a um setor.

Devido a uma pesquisa de mercado feita sobre ferramentas de visualização disponíveis, decidiu-se pelo uso da Power BI da Microsoft. [Microsoft]. A escolhase deu ao fato da versão gratuita apresentar um grande número de funcionalidades (publicação online, melhor conexão com tabelas Excel, suporte a ferramentas de co- munidade) que estavam indisponíveis em outras opções e pelo grande número de ma- terial de aprendizado encontrado para a mesma, sendo a ferramenta em liderança do setor [Richardson et al. 2021]. Como ferramenta, o Power BI permite a importação de conexões com fontes de dados de diferentes origens realizando um robusto pré- processamento das informações oriundas dessas fontes, transformando-as em tabelas mediante necessidade. Feito o pré-processamento, a parte visual é feita através de componentes selecionáveis que são arrastados para uma tela em branco, utilizando os dados como parâmetros. Através de conexões definidas pelos analistas, diferentes tabelas se conectam entre si, permitindo filtragem e interconectividade dos dados.

Assim sendo, começando pela 1ª Jurisdição, montou-se um esquema padrão de como o projeto seria seguido em cada um dos setores, como pode ser visto na figura 1.

Inicialmente, tem-se uma apresentação do setor pelos seus funcionários sobre como é o funcionamento diário do mesmo e das dificuldades apresentadas. Em seguida, uma reunião é feita entre os analistas de BI, a SEBDA e os usuários para o estabeleci- mento dos requisitos que devem ser atendidos pelos painéis criados pela equipe. Com os mesmos definidos, a SEBDA busca na base de dados do TRE informações que possam ser relevantes para a resolução dos problemas apresentados, deixando-as a disposição para uma montagem de sub-divisões desses dados em tabelas menores e especializadas. Essas tabelas são pré-processadas para facilitar a modelagem envolvendo as mesmas e com esta etapa feita, os visuais são montados de acordo com a identidade visual do setor e dos re- quisitos, sofrendo de testes para garantia de veracidade dos dados e prosseguindo para uso dos usuários. Na Etapa de Entrega Parcial, objetiva-se buscar erros que tenham passado pela equipe e receber opiniões sobre o layout e uso, gerando um ciclo de correções até uma entrega final.

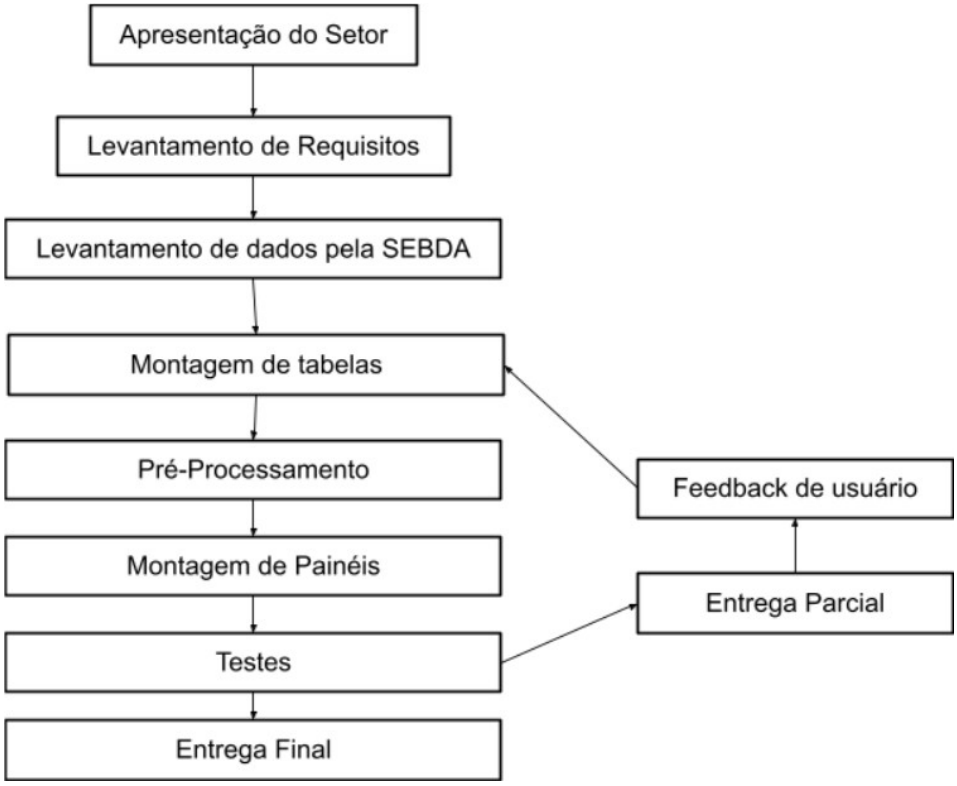


Figura 1. Etapas de implementação BI no TRE-BA. Fonte: Autor.

4.1. 1ª Jurisdição

Observou-se uma certa demanda por menos demonstrações visuais dos dados pedidos nos requerimentos. Devido a opiniões oriundas do setor, foi decidido o uso majoritário de tabelas contendo os dados como pode ser visto na Figura 2, representando os processos individualmente.

Por outro lado, os visuais foram utilizados apenas para representação do número de processos ao longo do tempo, além de um mapeamento do estado da Bahia mostrando as regiões com maior número de processos.

Figura 2. Painel Principal da 1ª Jurisdição. Fonte: TRE-BA.

4.2. Painel Fala Cidadão

O projeto do painel Fala Cidadão foi focado em mostrar visualmente os dados de aten- dimento com o público do TRE, mostrando de forma simples informações para monitoramento interno e possível controle de qualidade da população, como pode ser visto na Figura 3.

Figura 3. Painel Principal da Ouvidoria. Fonte: [TRE-BA].

5. Avaliação Qualitativa

A avaliação da efetividade do projeto de BI nos setores foi realizada através do uso de dois fatores como parâmetros: (1) o número de chamados requisitados à SEBDA vindo dos setores, antes da implementação e depois do começo do uso dos painéis e (2) as opiniões dos usuários que agora utilizam dos serviços disponibilizados pela equipe.

Infelizmente, devido a circunstâncias do próprio TRE, o primeiro fator não se torna viável para análise pois os chamados não são categorizados, tornando custosa a opção da montagem de uma linha do tempo com o número de chamados especificados. Já o segundo fator foi realizado através de um formulário disponibilizado aos usuários.

5.1 Formulario por Usuário

Um formulário online foi disponibilizado para a avaliação da experiência dos usuários. Esse for- mulário foi repassado aos setores com o intuito da coleta de suas opiniões sobre os painéis criados através das seguintes perguntas:

1. Qual seu nível de uso dos painéis?
2. Seus requisitos foram atendidos?
3. Nível de esforço em encontrar informações no painel
4. As informações estão de acordo com o que é visto na vida real?
5. Informações diversas

Os tópicos 2, 3 e 4 são uma seleção escalar de 1 a 5, onde 5 indica um resultado positivo pra pergunta em questão. O tópico 1 é uma seleção de múltipla escolha para indicar a periodicidade do uso dos painéis e o tópico 5 são algumas frases envolvendo o projeto para que os usuários concordem ou discordem com a veracidade da mesma.

6. Resultados e Discussão

O formulário ficou disponível por três semanas. Os resultados foram coletados e são apresentados nas Figuras 4, 5 e 6:

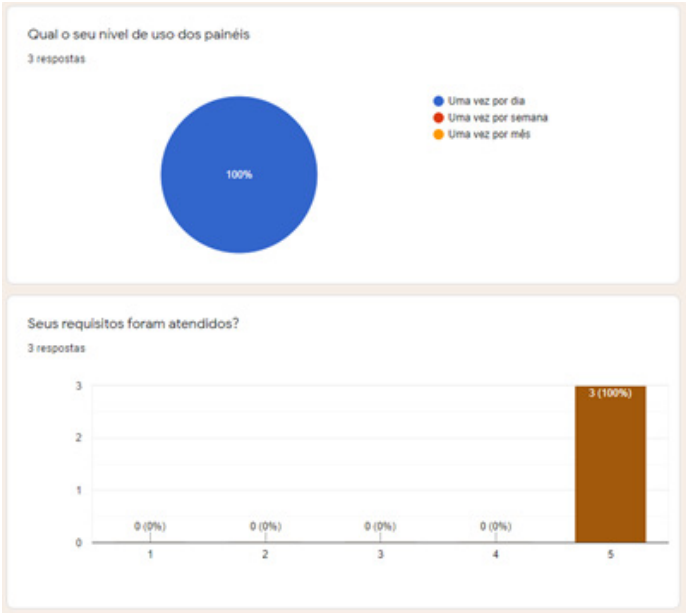


Figura 4. Respostas dos Tópicos 1 e 2. Fonte: Autor.

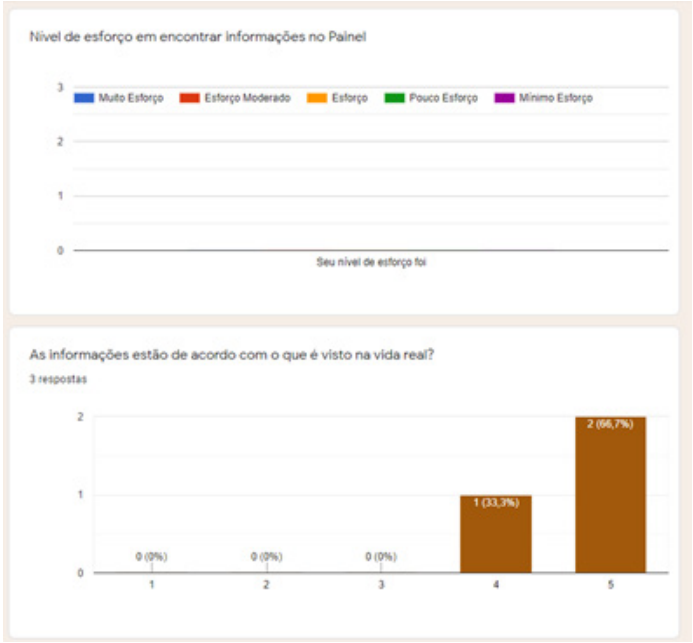


Figura 5. Tópicos 3 e 4. Fonte: Autor.

O tópico 1 observou o uso do *dashboard* pelos funcionários,obtendo uma frequen- cia de pelo menos 1 vez por semana. Enquanto que o tópico 2 demonstrou que 100% dos funcionários tiveram os seus requisitos atendidos pelo dashboard desenvolvido.

O tópico 3 objetivou conhecer o nível de esforço empregado pelosfuncionários para o uso do dashboard.

Devido a um erro de digitação no enunciado do Tópico 3 no formulário, os dados de resposta do tópico foram perdidos durante a correção do erro. Porém, ainda assim foi possível observar na co- leta que o nivel de esforço foi mínimo para uso do *Dashboard*.

Em relação às respostas do Tópico 4, observou-se que todos os funcionários con- sideraram que a informação prestada pelo *dashboard* está de acordo com o exposto no dia-a-dia do TRE.

E por fim, o Tópico 6 discutiu sobre algumas características das informações vi- suais apresenta- das. A Figura 6 denota que os visuais não dificultam a visualização, os dados estão organizados e a navegação e interatividade estão bem apresentadas.

Embora dificuldades tenham sido apresentadas durante a criação dos painéis, de- vido a erros de comunicação entre as partes envolvidas ou detalhes no banco de dados que passaram despercebi- dos inicialmente, a opinião sobre melhorias e correções se apre- sentou geralmente como a adição de funcionalidades a projetos que estão num estágio concluído, com o intuito de facilitar ainda mais a usabilidade dos painéis no dia-a-dia dos setores.

Figura 6. Tópico 5. Fonte: Autor.

7. Considerações Finais

O projeto de residência buscou resolver problemas do TRE de forma inovadora enquanto preparava novos profissionais para as dificuldades de desenvolvimento de software, e pelodemonstrado através de interações com usuários, pode-se então afirmar que a proposta foi exitosa.

Business Intelligence traz consigo a possibilidade de melhorar a visualização de informações já existentes, dando aos tomadores de decisão uma ferramenta que os ajuda a se basear ainda mais em fatos e notar tendências ao longo do tempo, melhorando assim o monitoramento nos setores nos quais o projeto ja´ foi implementado.

Embora ainda não completo, devido a necessidade de mais tempo para a transição de mais setores para este novo modelo, o que ja´ foi feito serve de grande exemplo sobreos benef´ıcios que BI pode trazer ao TRE, com o Portal de BI sendo julgado como um dos fatores que facilitou a conquista do Prêmio de Qualidade, instituído pelo Conselho Nacional de Justiça (CNJ), no dia 03/12/2021.

Um ponto fraco deste estudo é a baixa participação de funcionários na resposta do formulário, quando comparado com o número de funcionários que utilizam os painéis, não gerando uma gran- de população que daria ainda mais veracidade aos resultados obti- dos. Outro problema que pode

ocorrer é a descontinuação do suporte devido a mudanças no TRE, embora ao final deste projeto alguns funcionários já estejam sendo capacitados para a manutenção e criação de novos sistemas BI para os setores.

Assim sendo, o projeto e´ finalizado dando o molde para que o TRE continue a implementação em outros setores, através de colegas que acompanharam a residência e os processos de decisão que levaram a criação dos painéis da 1ª e 2ª Jurisdição, da Ouvidoria e dos muitos outros que serão realizados futuramente. O BI traz consigo uma infinidade de possibilidades de aplicações e através das mesmas, o serviço público disponilizado pode ficar ainda mais rápido e simples de acompanhar, até mesmo ao cidadão comum.

7.1. Trabalhos Futuros

É esperado que o projeto possa ser expandido aos outros setores do TRE, cobrindo os problemas apresentados e reduzindo a dependência de outros setores para a criação de relatórios, sendo possível a realização de outra análise similar a esta para um compara- tivo da satisfação de usuários ao término da implementação ou apenas para um controle de qualidade. Também é possível a abertura de um estudo para checagem da hipótese da queda no número de chamados abertos à SEBDA, porém, talvez seria necessária uma refatoração de como os chamados são contabilizados, pois os mesmos não são categoriza- dos pelo setor de origem. Uma análise por este ângulo demonstraria valores diretamente quantitativos sobre o impacto que a implementação de BI gerou no TRE.

Referências

Djerdjouri, M. (2020). Data and business intelligence systems for competitive advantage:prospects, challenges, and real-world applications. *Mercados y Negocios*, (41):5–18.

Ferreira, J., Miranda, M., Abelha, A., and Machado, J. (2010). O processo etl em sistemasdata warehouse. pages 757–765.

Hedgebeth, D. (2007). Data-driven decision making for the enterprise: an overview ofbusiness intelligence applications. *Vine*.

Microsoft. Microsoft power bi. Microsoft (2019). Customer stories.

Negash, S. and Gray, P. (2008). Business intelligence. In *Handbook on decision supportsystems 2*, pages 175–193. Springer.

Richardson, J., Schlegel, K., Sallam, R., and Sun, J. (2021). Magic quadrant for analyticsand business intelligence platforms. *Gartner*.

Scherer, M. (2012). Inside the secret world of the data crunchers who helped obama win.TRE-BA. Portal bi.

Marcelo Moura Caires
Tribunal Regional Eleitoral da Bahia
1ª Av. do Centro Administrativo da Bahia, 150 – CAB
Salvador-BA - CEP: 41.745-901 - Brasil
caires.marcelo@ufba.br

Controle de Acesso em APIs Rest com Spring Security, API Keys e JWT - Uma Abordagem Prática

Resumo. *Este artigo pretende apresentar uma solução de segurança para a API do Sistema de Gerenciamento de Recursos Humanos (SGRH-API), desenvolvida no âmbito da especialização em tecnologia da informação, residência TRE x UFBA.*

1. Introdução

1.1. Contexto

A segurança é um dos aspectos primordiais de qualquer sistema computacional. Conforme Spilca (2020) "segurança é uma das qualidades não funcionais essenciais de um sistema de software". A melhor maneira de compreender a importância da segurança para um sistema de software, ainda segundo Spilca (2020), é tentar observá-la do ponto de vista do usuário. A utilização de qualquer sistema implica no envio e recebimento de dados, dados esses que podem ser sensíveis ou não. Para um usuário, o vazamento de certos tipos de dados podem ser irrelevantes. Já outros, como informações bancárias ou de cunho íntimo, podem trazer enorme prejuízo financeiro e/ou pessoal.

A análise desses fatores/requisitos de segurança, deve fazer parte do projeto de qualquer tipo de software, não podendo ser diferente no projeto de APIs. Madden (2020) descreve 3 fatores que todo desenvolvedor deve se ater no desenvolvimento de uma API:

1. Uma mesma API pode ser acessível a usuários de níveis distintos de autoridade. Por exemplo: algumas operações podem ser permitidas apenas para administradores ou outros usuários com uma função especial. A API também pode ser exposta a usuários na internet que não deveria ter nenhum acesso. Sem um controle de acesso apropriado, qualquer usuário pode executar qualquer ação.
2. Embora cada operação em uma API possa ser segura individualmente, as combinações de operações podem não ser. A segurança de uma API deve ser considerada como um todo, e não como operações individuais.
3. Podem haver vulnerabilidades de segurança devido à própria implementação da API. Validações de dados incompletas ou mal feitas, podem permitir o envio de grandes quantidades de informação ou informações incorretas, gerando falhas de execução ou ataques de negação de serviço (DoS).

Além desses fatores, deve-se considerar o que se espera proteger em um sistema, quais recursos, dados, arquivos, etc. Quais objetivos de segurança da informação são importantes, como confiden-

cialidade, integridade, disponibilidade e autenticidade. Quais os mecanismos disponíveis para atingir esses objetivos, o ambiente em que a API deve operar e as ameaças que existem nesse ambiente.

Espera-se, assim, apresentar neste artigo uma solução de segurança para a SGRH-API, desenvolvida no âmbito da especialização em tecnologia da informação, residência TRE x UFBA, considerando os diversos fatores, ferramentas e técnicas de segurança em APIs.

1.2. Motivação

Embora seja comum o desenvolvimento de projetos de APIs privadas no TRE-BA, com o uso do framework Spring Boot. O projeto da SGRH-API , por ser uma API pública, apresentou um desafio novo em termo de segurança de dados, sendo necessário pensar em uma solução para atribuir de forma dinâmica os acessos às endpoints da mesma, permitindo a manutenção desses acessos conforme a necessidade e característica de cada usuário(cliente) que irá consumi-la.

No desenvolvimento das APIs privadas do TRE-BA, os requisitos de segurança se restringem a autenticar usuários via LDAP e permitir que eles utilizem a aplicação cliente, sem restringir o acesso à métodos ou endpoints específicos, salvo em casos especiais de endpoints de configuração e administração do sistema. Nesses casos, uma configuração estática do spring security, seja de forma programática ou através de arquivos de configuração, resolve a contento o problema.

Porém, no caso de uma API pública, como a SGRH-API, faz-se necessária a autorização de acesso para cada endpoint requisitado e a manutenção dessas autorizações, podendo ser criadas, restringidas ou alteradas a qualquer momento, conforme a necessidade do serviço. Para suprir esse requisito, seria necessário, além de buscar uma solução para a implementação da gestão de autorizações na API, implementar uma interface cliente para gerenciar essas autorizações, que seria, inclusive, um usuário(cliente) da SGRH-API.

Diante desse desafio, propõe-se a contribuir com o projeto da SGRH-API, buscando e implementando uma solução de segurança para a mesma.

1.3. Objetivos

Implementar uma solução de segurança para controle de acesso à cada método(endpoint) da SGRH-API.

1.4. Objetivos específicos

- Estudar o framework Spring Security e técnicas de segurança em APIs para a implementação dos requisitos de segurança da API.
- Implementar a solução de segurança na API.

2. Referencial teórico

O mundo atual é um mundo conectado. A conectividade é um aspecto essencial da vida em sociedade e permeia todas as relações humanas, sejam elas pessoais ou profissionais. Essa realidade sempre existiu e continuará existindo, pois qualquer relacionamento exige conexão, seja ela física(presencial) ou virtual(remota).

O ambiente virtual da internet provê um meio riquíssimo de conectividade, onde pessoas podem se conectam umas às outras, ou à empresas, utilizando softwares para os mais variados propósitos, como: transações financeiras, negócios, comunicação, entretenimento, etc. Por trás de todo software disponível na internet, quase sempre haverá uma API.

> *"As interfaces de programação de aplicações (APIs) estão em toda parte. Abra seu smartphone ou tablet e veja os aplicativos que você instalou. Quase sem exceção, aqueles aplicativos estarão conversando com uma ou mais APIs remotas, para baixar novos conteúdos e mensagens, pesquisar notificações, fazer upload de conteúdo novo e executar ações em seu nome. Carregue sua página da web favorita, com as ferramentas de desenvolvedor abertas em seu navegador, e você provavelmente verá dezenas de chamadas de API, acontecendo em segundo plano, para renderizar a página." (MADDEN, 2020)*

API, acrônimo em inglês que significa Interface de Programação de Aplicações, é um conjunto de padrões e protocolos que possibilita a comunicação entre plataformas independentes, de diferentes implementações. Conforme Madden(2020) "em termos gerais, uma API é um limite entre uma parte de um sistema de software e outro. Ele define um conjunto de operações que um componente fornece para outras partes do sistema (ou outros sistemas)". Podemos entender API, dessa forma, como uma interface padronizada de comunicação entre sistemas, um acordo estabelecido entre duas partes para compartilhamento de informações e recursos.

Atualmente, o estilo REST (REpresentational State Transfer), tem se apresentado como umas principais e mais amplamente utilizadas especificações para implementação de APIs.

> "O estilo REST (REpresentational State Transfer) foi desenvolvido por Roy Fielding para descrever os princípios que levaram ao sucesso do HTTP e da web, E foi, mais tarde, adaptado como um conjunto de princípios para design de API. Em contraste com outros estilos, enfatiza formatos de mensagem padrão e um pequeno número de operações genéricas para reduzir o acoplamento entre um cliente e uma API específica e o uso de hiperlinks para navegar na API, reduz o risco de quebra de clientes à medida que a API evolui com o tempo." (MADDEN, 2020)

REST utiliza, dessa forma, a própria arquitetura Web para definição das operações e protocolos disponíveis em seu estilo arquitetônico. Conforme descrito pelo próprio Fielding:

> "O estilo REpresentational State Transfer (REST) é uma abstração dos elementos arquitetônicos em um sistema hipermídia distribuído. O REST ignora os detalhes da implementação [...] e abrange as restrições fundamentais sobre componentes, conectores e dados que definem a base da arquitetura da Web e, portanto, a essência de seu comportamento como um aplicativo baseado em rede." (FIELDING, 2000)

Para Madden (2020), no contexto da segurança, uma API encontra-se na interseção de três importantes disciplinas, conforme a figura 1:

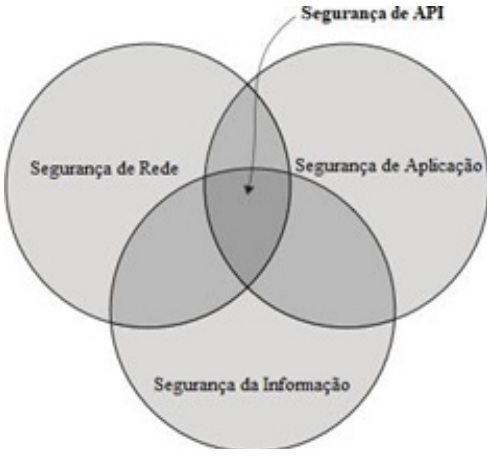


Figura 1. Intersecção de áreas na segurança de API (MADDEN,2020)

1. Segurança da informação: preocupa-se com a proteção da informação ao longo de todo o seu ciclo de vida

2. A segurança de rede: trata da proteção dos dados que fluem em uma rede e prevenção de acesso não autorizado à própria rede.

3. A segurança do aplicativo: garante que os sistemas de software sejam projetados e construídos para resistirem a ataques e uso indevido.

Um desenvolvedor não precisa, necessariamente, ser especialista em cada uma dessas disciplinas de segurança, para projetar APIs seguras. Porém o conhecimento de cada uma delas permite que o mesmo conjugue o trabalho de diferentes profissionais no objetivo de tornar sua aplicação segura. A segurança de aplicação é sua área de atuação principal e nela são aplicadas as principais técnicas para proteção da API, contra uso indevido. Conforme veremos no decorrer deste trabalho

2.1. A SGRH-API

O Tribunal Regional Eleitoral da Bahia - TRE-BA, conforme Resolução 102 do CNJ, em quadro de quantitativo de cargos e funções, divulgado em 13/09/2021, possui um total de 902 cargos ativo e ocupados (TRE-BA, 2009). Para gerenciamento de um quadro funcional dessa dimensão, é imprescindível a utilização de um sistema informatizado, o Sistema de Gerenciamento de Recursos Humanos - SGRH. Nesse sistema são controlados aspectos funcionais como frequência, benefícios, remuneração, afastamentos, férias, comissionamentos e substituições, dentre outros.

O SGRH é um sistema utilizado no TRE-BA, mas desenvolvido e mantido pelo Tribunal Superior Eleitoral - TSE. Embora seja um sistema desenvolvido para atender às regionais da Justiça Eleitoral, em muitos casos as especificidades de cada uma delas limita o seu uso e demanda o desenvolvimento de sistemas administrativos auxiliares, para a completa automatização dos processos.

No desenvolvimento dos sistemas administrativos do TRE-BA, utiliza-se, muito frequentemente, a base de dados do SGRH. Embora haja um criterioso trabalho da Seção de Banco de Dados - SEBDA, na administração dessa base de dados, há um constante retrabalho da seção à cada solicitação de acesso e de consultas à mesma.

Uma forma de evitar o retrabalho mencionado acima, bem como de padronizar e centralizar o acesso à consultas na base dados do SGRH, é o desenvolvimento de uma API com esse objetivo. Essa API deve possui os seguintes requisitos, funcionais e não funcionais:

1. Utilização de arquitetura Rest
2. Disponibilização de métodos para diferentes consultas do SGRH
3. Documentação extensa e acessível
4. Autenticação via Tokens
5. Autorização de acesso individual para cada método acessado
6. Manutenção de usuários e perfis de acesso

Para o desenvolvimento da SGRH-API, pela equipe da Residência TRE-BA x UFBA, deveria ser empregada uma tecnologia já em uso pela Seção de Soluções Corporativas - SEDESC, do TRE-BA. De forma que a solução pudesse ser mantida posteriormente, ao término do contrato, pelos servidores da seção. Para tanto foi adotado o Framework Spring com Spring Boot como ferramenta de desenvolvimento da API.

Conforme Sharma (2021), Spring é um Framework escrito em linguagem Java que possui módulos para variados própósitos, como Spring Data(para persistência e suporte a bancos de dados) e Spring Security(para implementação de segurança). Spring suporta injeção de dependência (inversão de controle) e orientação a aspectos e conta com o Spring Boot, uma ferramenta que facilita a construção de aplicativos baseados em Spring. Com Spring Boot pode-se produzir o esqueleto da estrutura de um projeto Spring com qualquer funcionalidade, bastando adicionar os módulos desejados. De forma rápida e automatizada.

3. Proposta: Controle de Acesso em duas Etapas

A solução para qualquer projeto de segurança de software, assim como de uma API, pressupõe o conhecimento das técnicas, mecanismos e ferramentas disponíveis para sua implementação. Conforme MADDEN(2020), ameaças ao sistema podem ser combatidas com mecanismos, que garantem que os requisitos de segurança da API sejam atendidos. A Figura 7 apresenta os principais mecanismos de segurança adotados na maioria dos sistemas:

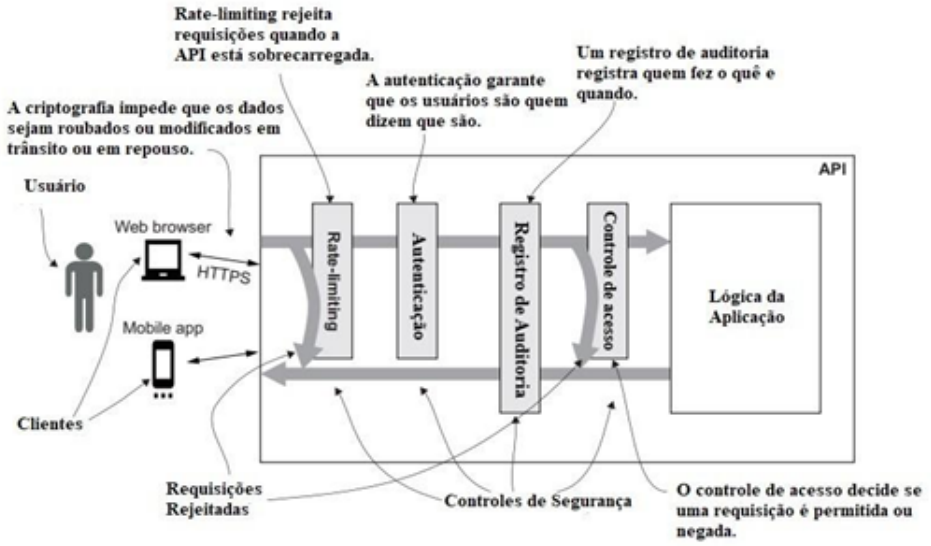


Figura 2. Mecanismos de segurança aplicados ao fluxo de requisições de uma API (MADDEN,2020)

- **Criptografia:** Garante que os dados não possam ser lidos por terceiros.
- **Rate-limiting:** Evita que usuários sobrecarreguem os recursos da API.
- **Autenticação:** Garante que seus usuários e clientes sejam quem eles dizem que são.
- **Registro de auditoria:** Garante que todas as operações sejam registradas para permitir a prestação de contas e o monitoramento adequado da API.
- **Controle de acesso(autorização):** Garante que cada solicitação feita à API esteja devidamente autorizada.

Desses mecanismos, dois serão fundamentais para a implementação dos requisitos de segurança da SGRH-API: autenticação e autorização.

3.1. Autenticação

"Autenticação é o processo de verificar se um usuário é quem diz ser. [...] A maneira mais fácil de fazer isso é fazer com que o cliente nos diga quem ele é para, então, verificarmos se ele está falando a verdade" (MADDEN, 2020). Na SGRH-API, a autenticação será utilizada para verificar a identidade do usuário e, uma vez autenticado, fornecer as permissões necessárias para a sua utilização, através de um Token JWT, como veremos mais adiante.

Segundo SCARIONI(2013), no processo de autenticação, um usuário apresenta à API informações confidenciais(credenciais) que o identificam (normalmente, um nome de usuário e uma senha). Em seguida, compara essas informações com as credenciais salvas previamente. Se as credenciais frnecidas pelo usuário corresponderem com as credenciais salvas, o usuário é autenticado com sucesso.

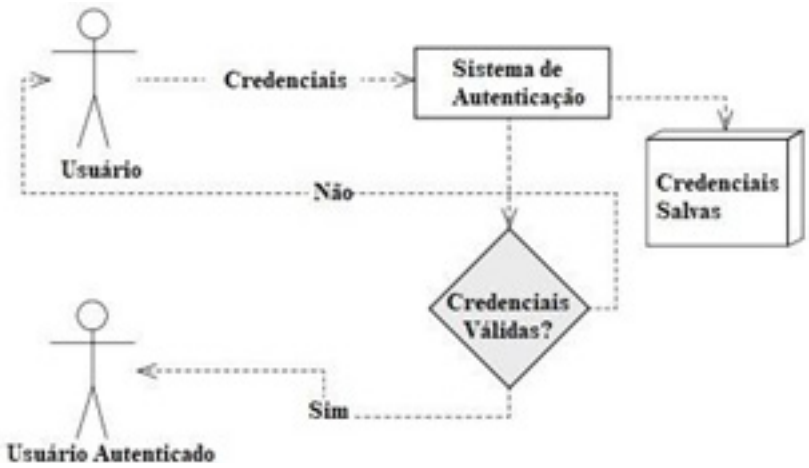


Figura 3. Mecanismo de autenticação simples (SCARIONI, 2013)

No contexto da SGRH-API, um usuário será representado pela entidade *API\USUARIO*, responsável por armazenar as credenciais de cada usuário, através dos seguintes campos:

Figura 4. Entidade API USUARIO

Além dos campos autoexplicativos, essa entidade possui um campo token, que armazena uma string única codificada em SHA-256[1] através da combinação dos campos id, nome e da data e hora em que o token foi criado. Esse token será utilizado futuramente, no header da requisição de autenticação, para fornecer as credenciais do usuário e autenticá-lo, conforme veremos adiante.

3.2. Autorização

Em sistemas complexos, onde há uma separação bem definida de papéis e atribuições, o processo de autenticação, em geral, não garante quais operações, módulos ou métodos serão permitidas ao usuário autenticado, apenas que o usuário foi reconhecido pelo sistema.

Uma vez autenticado, conforme SCARIONI(2013), a próxima etapa do processo é determinar quais recursos e ações o usuário tem permissão para acessar ou executar. Nesse processo, o sistema compara o conjunto de permissões do usuário com as permissões necessárias para executar uma ação específica, permitindo ou negando o acesso conforme o resultado.

1 SHA-256 é uma função hash criptográfica patenteada que produz um valor de 256 bits.

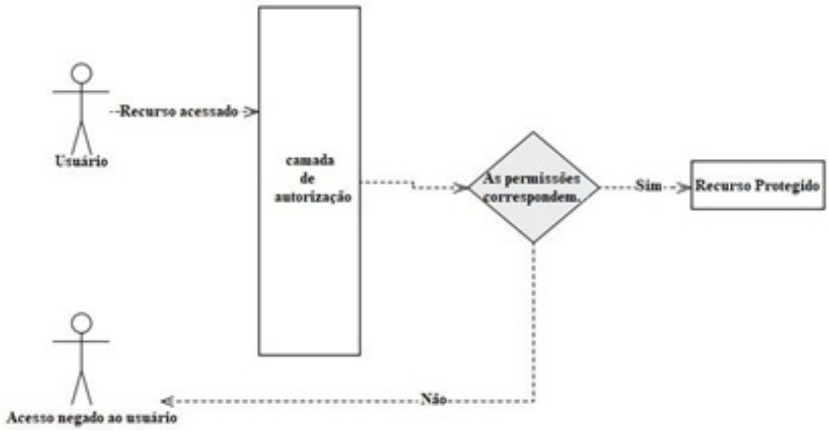


Figura 5. Processo de autorização simples. O usuário autenticado tenta acessar um recurso seguro (SCARIONI, 2013)

Para Madden(2020), existem duas abordagens principais para o controle de acesso em APIs:

- ***Controle de acesso baseado em identidade***: Nesse tipo de abordagem, primeiro o sistema identifica o usuário e, em seguida, determina o que ele pode fazer com base em suas credenciais. As credenciais são passadas em cada requisição, para serem autenticadas e verificadas.
- ***Controle de acesso baseado em capacidade:*** Essa abordagem utiliza tokens especiais ou chaves conhecidas com capacidades(autoridades) para acessar uma API. A própria capacidade diz quais operações ou recursos podem ser acessados, ao invés de credenciais de usuário. As permissões de acesso são definidas para cada recurso, de forma que o usuário não seja capaz de acessá-lo indevidamente.

A abordagem escolhida para o controle de acesso da SGRH-API será uma combinação das duas abordagens vistas anteriormente. Nela, serão utilizadas duas etapas para controle do usuário às funcionalidades da API, conforme a figura 5:

Figura 6. Controle de acesso em duas etapas da SGRH-API

1. O usuário acessa a api para obter a autenticação de suas credenciais, através do token de acesso cadastrado previamente. Uma vez autenticado, a API gera um token JWT para o usuário, com duração definida, no qual estarão encapsuladas suas permissões(capacidades) de acesso.

2. O usuário envia o token JWT gerado no cabeçalho de cada futura requisição. Para cada requisição recebida com token JWT válido, a API irá extrair as permissões(capacidades) de acesso, encapsuladas nele, para comparar com a permissão cadastrada para o recurso acessado, permitindo ou negando o acesso conforme o resultado.

4. Aplicação Prática

A implementação da proposta acima consiste da utilização do Spring Boot e de alguns dos frameworks do seu ecossistema. Conforme as etapas a seguir:

1. Definição e criação do banco de dados: nessa etapa serão criadas as tabelas no banco de dados (*Spring Data*) que armazenarão os dados dos usuários e suas respectivas autorizações de acesso.

2. Configuração do módulo de segurança: nessa etapa será realizada a configuração da API (*Spring Security*) para atender aos requisitos de segurança definidos previamente.

3. Configuração dos métodos: nessa etapa serão configurados cada método (*Spring Framework*) com suas respectivas permissões.

4.1 Projeto do Banco de Dados

Para o armazenamento dos dados necessários ao controle de acesso da API, será criada uma estrutura simples de banco de dados, consistindo de apenas três tabelas e suas relações, conforme a figura 6:

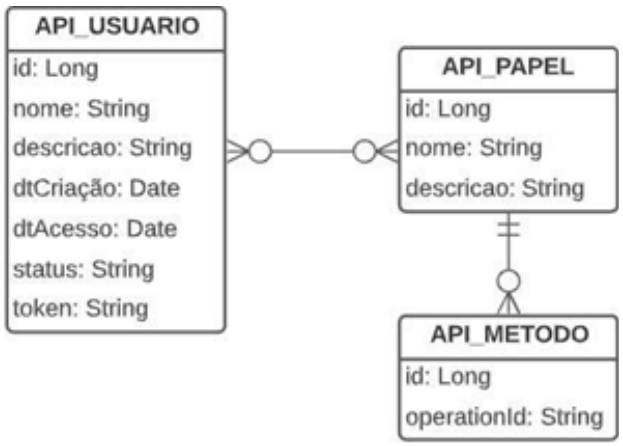


Figura 7. Banco de Dados para o controle de acesso da SGRH-API

- **API USUARIO**: Como explicado anteriormente, essa entidade possui um campo token, utilizado no header da requisição de autenticação, para fornecer as credenciais do usuário e autenticá-lo. Pode possuir numero indeterminado de papéis, inclusive nenhum, que definirão suas autorizações de acesso á API.
- **API PAPEL**: Essa entidade tem como objetivo armazenar grupos de autorização, representados por diferentes métodos da API, que poderão ser atribuídos aos usuários para acesso aos respectivos métodos. Possui os campos: id, nome, descrição e pode possuir número indeterminado de métodos.
- **API METODO**: O Objetivo dessa entidade é armazenar uma lista de métodos vinculados á cada papel. Possui os campos id e operationId (que armazena uma id de autorização do respectivo método da API).

4.2. Módulo de Segurança

Como abordado anteriormente, o módulo de segurança da *SGRH-API* será implementado conforme a estratégia de controle em duas etapas, sendo necessário configurá-lo nas etapas de autenticação e autorização, atendendo os requisitos de segurança predefinidos para a mesma. O *Spring Security* será ferramenta adotada para a execução desta estratégia, por ser um framework do próprio ecossistema do *String Boot*, altamente integrado com o mesmo, e uma solução de mercado consolidada, amplamente utilizada e testada.

> *Spring Security* é descrito como uma estrutura poderosa e altamente personalizável para autenticação e controle de acesso. [...] é uma estrutura que simplifica enormemente a aplicacação de segurança em aplicacaçõesSpring e é a escolha primária para a implementação de segurança, em nível de aplicativo, para essas aplicações. Geralmente, seu objetivo é oferecer a você uma maneira altamente personalizável de implementaçãode autenticação, autorização e proteção contra ataques.(SPILCA,2020)

4.3. Módulo de Segurança: Dependências

Para utilização do *Spring Security*, em uma aplicacação *Spring Boot*, será necessária a inclusão das dependências abaixo:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
    <groupId>io.jsonwebtoken</groupId>
    <artifactId>jjwt</artifactId>
    <version>0.9.1</version>
</dependency>
```

Listing 1. Dependências do *Spring Security* e JWT

4.4. Módulo de Segurança: Configuração padrão

Um vez instaladas as dependências do *Spring Security*, em um projeto *Spring*, o mesmo habilita por ”default” um padrão básico de segurança (*HTTP Basic*), sem necessidade de qualquer configuração adicional. Esse padrão gera uma senha diferente, exibida a cada da API, na saída do console, que deve ser enviada a cada requisição feita à mesma, no campo *Authorization* do *header*.

No entanto, esse padrão, obviamente, não atende aos requisitos de segurança da SGRH-API, sendo necessário sobrescrever e ampliar as configurações do *Spring Security*.

> A autenticação *HTTP Basic* não se encaixa na maioria das arquiteturas dos aplicativos. Ás vezes, gostaríamos de alterá-lo para corresponder ao nosso aplicativo. Da mesma forma, nem todos os *endpoints* de um aplicativo precisam ser protegidos, e para isso, talvez seja necessário escolher regras de autorização diferentes. Para fazer tais mudanças, começamos estendendo a classe *WebSecurityConfigurerAdapter*. Estender esta classe nos permite substituir o método configure (HttpSecurity http). (SPILCA,2020)

```
1 ...imports
2
3 @Configuration
4 @EnableWebSecurity
5 @EnableGlobalMethodSecurity(prePostEnabled=true)
6 public class SecurityConfigurations extends WebSecurityConfigurerAdapter{
7
8     @Autowired private TokenService tokenService;
9
10    @Override
11    protected void configure(HttpSecurity http) throws Exception {
12        http.authorizeRequests()
13            .antMatchers("/v2/api-docs", "/configuration/**", "/swagger*/**", "/webjars/**").permitAll()
14            .antMatchers(HttpMethod.POST,"/auth/**").permitAll()
15            .anyRequest().authenticated();
16        http.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
17        http.addFilterBefore(new AutenticacaoViaTokenFilter(tokenService), UsernamePasswordAuthenticationFilter.class);
18        http.csrf().disable();
19        http.cors();
20    }
21 }
```

Listing 2. Classe WebSecurityConfigurerAdapter

Segundo Madden (2020), ”um dos aspectos confusos de criar configurações com Spring Security é ter várias maneiras de configurar a mesma coisa”. No método configure é definida a estratégia de autenticação da API, que consiste na utilização de um filtro, linha 24, que é aplicado á cada requisição feita á API. Esse filtro recebe como argumento uma classe responsável por fazer a interceptação das requisições, chamada AutenticacaoViaTokenFilter. Toda requisição feita á API é interceptada pelo método doFilterInternal(). Esse método verifica no header da requisição a existência do token JWT, validando-o e atenticando-o no sistema.

```
1 ...imports
2
3 public class AutenticacaoViaTokenFilter extends OncePerRequestFilter {
4
5     private TokenService tokenService;
6     public AutenticacaoViaTokenFilter(TokenService tService) {
7         tokenService=tService;
8     }
9
10    @Override
11    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain filterChain)
12            throws ServletException, IOException {
13        String appToken= request.getHeader("appToken");
14        String token= recuperarToken(request);
15        boolean isTokenValido= tokenService.validarToken(token);
16        if(isTokenValido) {
17            autenticarCliente(token,appToken);
18        }
19        filterChain.doFilter(request, response);
20    }
21
22    private void autenticarCliente(String token,String appToken) throews IOException{
23        String subject= tokenService.getSubjectToken(token);
24        List<GrantedAuthority> authorities = tokenService.getAuthorities(token);
25        UsernamePasswordAuthenticationToken auth=new UsernamePasswordAuthenticationToken(subject,appToken,authorities);
26        SecurityContextHolder.getContext().setAuthentication(auth);
27    }
28
29    private String recuperarToken(HttpServletRequest request) {
30        String token= request.getHeader("Authorization");
31        if(token == null || token.isEmpty() || !token.startsWith("Bearer ")) {
32            return null;
33        }
34        return token.substring(7, token.length());
35    }
36 }
```

Listing 3. Classe AutenticacaoViaTokenFilter

4.5. Módulo de Segurança: Autenticação

A primeira etapa se inicia quando o usuário solicita que a api faça a sua autenticação e devolva o token jwt com suas permissões de acesso. Nesse processo, o mesmo acessa a endpoint ”auth”, informando no header da requisição o appToken fornecido pelo administrador da api. Para isso a api fará uso de 2 classes, e seus respetivos métodos:

4.5.1. Classe AutenticacaoController

Essa classe fornece a endpoint ”auth”, que deve ser acessada pelo usuário quando deseja acessar a api. Deve ser, sempre, o primeiro endpoint acessado, pois, através dele, o usuário irá obter o token necessário para acessar o restante da API.

```
1 ...imports
2
3 @RestController
4 @RequestMapping("/auth")
5 public class AutenticacaoController {
6     @Autowired private ApiUsuarioRepository usuarioRepo;
7     @Autowired private TokenService tokenService;
8     @PostMapping
9     public ResponseEntity<?> autenticar(
10            @RequestHeader(value="appToken",required=true) String appToken){
11        try {
12            Optional<ApiUsuario> opUsuario=usuarioRepo.findByToken(appToken);
13            if(opUsuario.isPresent()) {
14                ApiUsuario usuario=opUsuario.get();
15                usuario.setDtAcesso(new Date());
16                usuario.setStatus(UsuarioAppStatus.tokenAtivo.getDescricao());
17                List<GrantedAuthority> authorities = new ArrayList<GrantedAuthority>();
18                usuario.getPapeis().stream().forEach(
19                    papel ->{
20                        papel.getMetodos().stream().
21                        forEach(
22                            metodo->{
23                                authorities.add(new SimpleGrantedAuthority(metodo.getOperationId()));
24                            }
25                        );
26                    }
27                );
28                String token= tokenService.gerarToken(appToken,usuario,authorities);
29                usuarioRepo.save(usuario);
30                HttpHeaders responseHeaders = new HttpHeaders();
31                responseHeaders.set("Access-Control-Allow-Headers", "Authorization, X-PINGOTHER, Origin, X-Requested-With, Content-Type,
  Accept, X-Custom-header");
32                responseHeaders.set("Access-Control-Expose-Headers", "x-access-token");
33                responseHeaders.set("x-access-token",token);
34                return ResponseEntity.ok()
35                        .headers(responseHeaders)
36                        .body(null);
37            }else {
38                return ResponseEntity.status(403).build();
39            }
40        }catch(AuthenticationException e) {
41            return ResponseEntity.status(403).build();
42        }
43    }
44 }
```

Listing 4. Classe AutenticacaoController

4.5.2 Classe TokenService

A classe TokenService fornece os métodos necessários para manipulação, validação e geração dos tokens JWT.

```
1 ...imports
2
3 @Service
4 public class TokenService {
5
6     @Value("${api.jwt.expiration}") private String tokenExpiration;
7     @Value("${api.jwt.secret}") private String apiSecretKey;
8     @Value("${spring.application.name}") private String apiName;
9
10    public String gerarToken(String appToken, ApiUsuario apiUsuario,List<GrantedAuthority> authorities) {
11        String token= null;
12        String tokenTipo="Bearer";
13        Date dtCriacao= new Date();
14        Date dtExpiracao= new Date(dtCriacao.getTime()+Long.parseLong(tokenExpiration));
15        token = Jwts.builder()
16                .setIssuer(apiName)
17                .setSubject(apiUsuario.getNome())
18                .setIssuedAt(dtCriacao)
19                .setExpiration(dtExpiracao)
20                .signWith(SignatureAlgorithm.HS256,apiSecretKey)
21                .claim("usuario",usuarioDTO)
22                .claim("authorities",authorities)
23                .compact();
24        return token;
25    }
26
27    public boolean validarToken(String token) {
28        try {
29            Jwts.parser().setSigningKey(apiSecretKey).parseClaimsJws(token);
30            return true;
31        }catch(Exception e) {
32            return false;
33        }
34    }
35
36    @SuppressWarnings("unchecked")
37    public List<GrantedAuthority> getAuthorities(String token){
38        List<GrantedAuthority> authorities=new ArrayList<GrantedAuthority>();
39        Claims claims= Jwts.parser().setSigningKey(apiSecretKey).parseClaimsJws(token).getBody();
40        List<LinkedHashMap<String,String>> authoritiesList = (List<LinkedHashMap<String, String>>) claims.get("authorities");
41        authoritiesList.forEach(a->{
42            Set<String> keys = a.keySet();
43            for (String key : keys) {
44                authorities.add(new SimpleGrantedAuthority(a.get(key)));
45            }
46        });
47        return authorities;
48    }
49 }
```

Listing 5. Classe TokenService

4.6. Autorização

Assim como a autenticação, há várias maneiras de realizar autorização em Spring Secu- rity. Os requisitos de segurança definidos para a nossa API especificam que, para todo método(endpoint) da mesma, deve haver uma autorização específica, que deve ser verificada antes do seu acesso.

Um usuário só poderá acessar determinado método se apresentar um token con- tendo a autorização expressa para esse acesso. Logo, uma configuração de autorização genérica não atende esse requisito. É necessário definir uma regra de autorização indivi- dualmente para cada método da API.

Para esse requisito, a melhor estratégia de segurança é configurar a autorização em nível de método. Em Spring Security, configurações em nível de métodos são desabilitadas por padrão, conforme Madden (2020). Sendo necessário habilitar expressamente essa Configuração no arquivo de Configuração, através da seguinte anotação: @EnableGlobal- MethodSecurity(prePostEnabled=true), conforme o Listing 6 (linha 5):

```
1 ...imports
2
3 @Configuration
4 @EnableWebSecurity
5 @EnableGlobalMethodSecurity(prePostEnabled=true)
6 public class SecurityConfigurations extends WebSecurityConfigurerAdapter{
7
8 ...
```

Listing 6. Classe TokenService

Uma vez habilitado, o nível de segurança de método pode ser utilizado, utilizando anotações em cada método, seguindo alguma dessas estratégias, conforme Madden (2020):

- **Pré-autorização** (*@PreAuthorize*): A estrutura verifica as regras de autorização antes da chamada do método.
- **Pós-autorização**(*@PostAuthorize*): A estrutura verifica as regras de autorização após a execução do método.

Como não é interessante que a API execute um método, caso a autorização do usuário seja negada. Usaremos a estratégia *PreAuthorize* nos métodos da SGRH-API. Conforme exemplo a seguir:

```
1 @GetMapping("/contato/{matricula}")
2 @PreAuthorize("hasAuthority('SERVIDOR_CONTATOS")
3 public ResponseEntity<ServidorContatoDTO> contato(@PathVariable String matricula){
4     Optional<ServidorContato> servidorContato = servidorContatoRepository.findByMatricula(matricula);
5     if(servidorContato.isPresent()) {
6         return ResponseEntity.ok(new ServidorContatoDTO(servidorContato.get()));
7     } else {
8         return ResponseEntity.notFound().build();
9     }
10 }
```

Listing 7. Exemplo de método anotado com *@PreAuthorize*

Esse processo deve ser repetido em todos os métodos a serem protegidos. Ao anotar o método do exemplo com *@PreAuthorize("hasAuthority('SERVIDOR_CONTATOS"),* o *Spring Security* é informado que esse método só pode ser acessado caso o usuário possua uma autoridade de nome *SERVIDOR_CONTATOS*. Que deve ser informada no *header* da requisição, como visto anteriormente

5. Conclusão

Foi visto, no presente artigo, que segurança é um requisito não funcional essencial para qualquer projeto de software, assim como no desenvolvimento da SGRH-API, que necessitava de uma solução para atender aos seus requisitos de segurança: autorização em nível de método e manutenção de usuários e perfis de acesso.

No estudo foi possível conhecer as diferentes disciplinas envolvidas na segurança de uma API e entender as técnicas de segurança em nível de aplicação, como as fases de autenticação e autorização, bem como, as ferramentas necessárias à sua implementação em uma API, com o Framework Spring Security.

Como resultado deste estudo, foi proposta e implementada uma solução de segurança em duas etapas. Nesta solução o usuário necessita, para acessar qualquer método, primeiro se autenticar na API

e obter um token contendo suas autorizações de acesso, o qual deverá ser submetido no cabeçalho de cada requisição, para verificação.

A próxima etapa para a finalização desta solução é a implementação de uma interface para a manutenção dos usuários e perfis de acesso. Permitindo, assim, o controle total das permissões de acesso da SGRH-API.

6. Referências

FIELDING, Roy Thomas. **Architectural Styles and the Design of Network-based Software Architectures**. 2000. 180 f. Tese (Doutorado) - Curso de Informa- tion And Computer Science, University Of California, Irvine, 2000. Dispon´ıvel em: https://www.ics.uci.edu/ fielding/pubs/dissertation/fielding dissertation.pdf. Acesso em: 05 dez. 2021.

MADDEN, Neil. **API security in action**. Shelter Island: Manning Publications, 2020. SCARIONI, Carlo. **Pro spring security**. New York, NY: Apress, 2013.

SHARMA, Sourabh. **Modern API Development with Spring and Spring Boot**. Bir- mingham, Uk: Packt Publishing, 2021.

SPILCA, Laurentiu. **Spring Security in action**. Shelter Island, NY: Manning Publicati- ons Co, 2020.

TRE-BA. **Resolução 102 Cnj - Anexo IV- Quantitativo de Cargos e Funções**. Disponível em: https://www.tre-ba.jus.br/transparencia-e-prestacao-de-contas/relatorios- cnj/recursos-humanos-e-remuneracao/arquivos-membros-2/tre-ba-anexo-iv-a-cargos- efetivos-agosto2021/at download/file. Acesso em: 07 dez. 2021.

Mirela Gico Casado
Tribunal Regional Eleitoral da Bahia[1]
1ª Av. do Centro Administrativo da Bahia, 150 – CAB
Salvador-BA - CEP: 41.745-901 - Brasil
mgcasado@tre-ba.jus.br

1 Daniela Barreiro Claro
Universidade Federal da Bahia - Instituto de Computação Departamento de Ciência da Computação. Salvador, BA – Brasil. dclaro@ufba.br

Identificação de perfil de eleitores faltantes em pleitos eleitorais – uma análise no TRE da Bahia

Abstract. *This article describes the data mining process that was applied to thedatabase of the Cadastro de Eleitores da Bahia. Elections are mandatory in Brazil, and the TRE is responsible for preparing all the necessary structure for elections to take place in a transparent, safe and efficient manner. The abstention of voters to the polls in 2020 was a record. In addition to the fact that we areexperiencing a pandemic and an economic crisis, there is great political disinterest on the part of voters. Thus, this work aims to discover patterns of behavioror try to establish profiles of voters who have been lacking in electoral electionsin Brazil. With this analysis, the TRE will be able to carry out clarifying and motivating campaigns that aim to reduce the absence of voters to the polls.*

Resumo. *Este artigo descreve o processo de mineração de dados que foi aplicado ao banco de dados do Cadastro de Eleitores da Bahia. As eleições são obrigatórias no Brasil, e o TRE é responsável pela preparação de toda a estrutura necessária para que as eleições aconteçam de maneira transparente, segura e eficiente. A abstenção dos eleitores às urnas em 2020 foi recorde. Além do fato de estarmos vivendo uma pandemia e uma crise econômica, observa-se um grande desinteresse político do eleitor. Desta forma, este trabalho visa descobrir padrões de comportamento ou tentar estabelecer perfis de eleitores que têm faltado aos pleitos eleitorais no Brasil. Com esta análise, o TRE poderá realizar campanhas esclarecedoras e motivadoras que objetivam a diminuição da ausência dos eleitores às urnas.*

1. Introdução

Em todas as fases da história do Brasil, quando foi colônia de Portugal, depois Império e por último, República, o voto sempre foi utilizado, direta ou indiretamente, como meio de escolha dos seus governantes. O direito de votar e ser votado foi garantido e vetado, ampliado e restringido, já foi sinônimo de poder, dinheiro e superioridade. O processo eleitoral evoluiu junto com a história política e com o surgimento de tecnologias capazes de tornar cada vez mais transparente e seguro este processo [TSE 2014].

O voto é o grande símbolo da democracia no Brasil e foi um direito conquistado através dos anos e muitas lutas [TSE 2014]. O processo democrático das eleições é importante para o país e a EJE (Escola Judiciária da Bahia) [EJE 2021] promove uma série

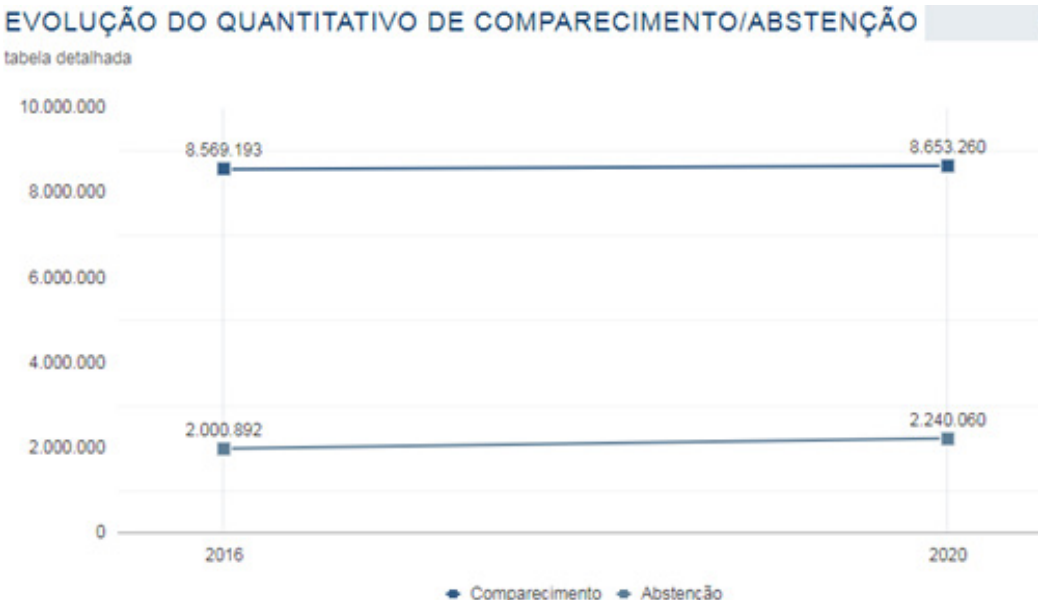


Figura 1. Eleitorado e Abstenção através dos anos - (Fonte - TSE)

de programas que fomentam a educação para cidadania e a formação política do cidadão. A abstenção às eleições tem aumentado através dos anos, Figura 1. A abstenção em 2020 foi a maior das últimas décadas. No segundo turno, 29,5% dos eleitores não compareceu às urnas. Este número é bem superior ao das últimas eleições (2018, 2016 e 2014) que ficou em torno de 21%. Número também é muito superior ao verificado nos demais pleitos para prefeitos e vereadores em 2012 (19,12%), 2008 (18,09%), 2004 (17,3%), 2000 (16,2%) e 1996 (19,99%). Em algumas cidades o índice de abstenção foi recorde: Rio de Janeiro (35,4%), Porto Alegre (32,8%) e São Paulo (30,8%), Goiânia (36,7%), Petrópolis (35,6%), Ribeirão Preto (35,6%), Blumenau (31%), Joinville (28%) e Aracaju (27,8%). Na maioria desses municípios, a abstenção, somada aos votos nulos e brancos, supera a votação obtida pelo vencedor do pleito.

Então, a principal motivação deste trabalho é encontrar padrões de comportamento dos eleitores que faltam às eleições .

1.1. Objetivos

O principal objetivo deste trabalho é analisar os perfis dos eleitores que faltam eleições municipais e gerais. Especificamente, através deste trabalho pretende-se:

- Definir as bases de dados mais apropriadas;
- Identificar padrões e comportamentos dos eleitores que faltam as eleições municipais e gerais;
- Identificar as ocorrências registradas e relevantes ao processo eleitoral;
- Avaliar os resultados no TRE-BA;

A próxima seção apresenta a metodologia do trabalho.

1.2. Metodologia

A metodologia empregada neste trabalho foi dividida utilizando as fases da técnica de KDD(*Knowledge Discovery Databases*) [Fayyad et al. 1996].

Os dados dos Eleitores do Cadastro de eleitoral da Bahia foram utilizados como fonte de dados. Os dados sensíveis, que pudessem de alguma forma identificar unicamente os eleitores, foram re-

tirados ou anonimizados na base, estando em conformidade com a Lei Geral de Proteção a Dados (LGPD) [Senado 2020] em vigor desde 16 de agosto de 2020.

O Oracle Data Mining [Center 1991] foi utilizada para a implementação da análise proposta neste trabalho, visto que o TRE-BA e o TSE utilizam o Oracle Enterprise Edition como Sistema Gerenciador de Banco de Dados [Elmasri and Navathe 2019]. A Versão atual do Oracle no TRE-BA é o Oracle Enterprise Edition 11.2.0.4 e o Oracle Developer 21.2.1.204 que implementa o Oracle Data Miner.

O presente trabalho está organizado em seções. A seção 2 descreve a fundamentação teórica sobre Descoberta do Conhecimento. A seção 3 apresenta o método desenvolvido neste trabalho. Na seção 4 são detalhados os experimentos realizados para a eleição de 2018. A seção 5 descreve os experimentos realizados para a eleição de 2020. A seção 6 são realizadas análises e discussões sobre os resultados preliminares obtidos. Por fim, a seção 7 descreve as considerações finais do trabalho, apresenta as ameaças encontradas e apresenta alguns direcionamentos de pesquisas futuras.

2. Referencial Teórico

O termo "Data Mining" é usado, frequentemente, como sinônimo para o processo de extração de informação útil de bases de dados [Fayyad et al. 1998]. É um processo de descoberta de padrões em grandes conjuntos de dados envolvendo métodos de aprendizado de máquina, estatísticas e sistemas de banco de dados. A mineração de dados é um subcampo interdisciplinar da Ciência da Computação e Estatística com o objetivo geral de extrair informações de um conjunto de dados e transformá-las em uma estrutura compreensível. O processo de analisar dados e tentar descobrir conexões entre eles vêm de longa data. Às vezes referenciado como 'descoberta de conhecimento' em bancos de dados, o termo mineração de dados não foi mencionado até a década de 1990. Na última década, os avanços na capacidade e velocidade de processamento evoluíram de práticas manuais, tediosas e demoradas, para uma análise de dados rápida, fácil e automatizada. Quanto mais complexos forem os conjuntos de dados coletados, maior será o potencial de descobrir informações relevantes [Koul 2020].

2.1. Descoberta de Conhecimento (KDD)

O processo de descoberta de conhecimento, também denominado do inglês *Knowledge Database* Discovery [Fayyad et al. 1996] consiste em um conjunto de fases que visam extrair conhecimento de grandes bases de dados, conforme a Figura 2. Pode-se dizer que é um processo ao mesmo tempo, iterativo e interativo. Iterativo, pois, poderá ser repetido quantas vezes for necessário na busca de melhores resultados, e, interativo, porque requer a participação o dos usuários. São etapas que podem ser tratadas independentes, entretanto, há uma forte dependência entre estas fases. Desta forma, para que os dados sejam preparados, deve haver antes uma seleção dos dados, e assim sucessivamente.

O KDD pode ser dividido em fases conforme a Figura 2.

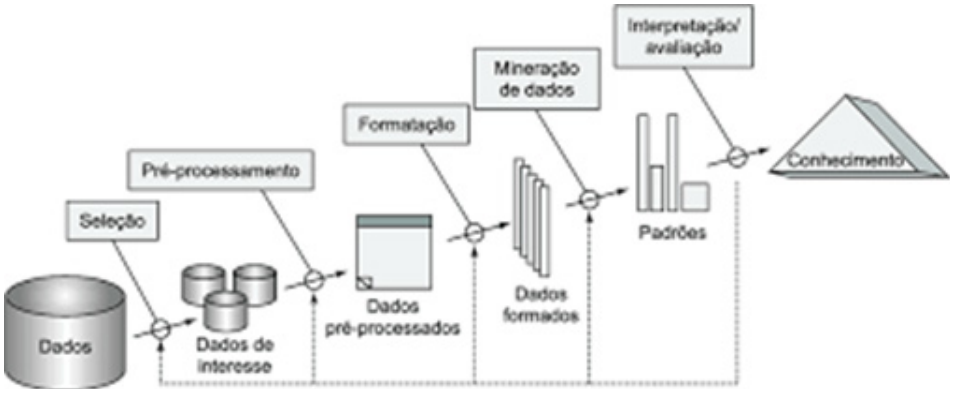


Figura 2. Etapas do Processo KDD

A **fase 1** é responsável pelo entendimento do domínio da aplicação e o conhecimento relevante para os objetivos da aplicação;

A **fase 2** cria um conjunto de dados de destino: inclui a seleção de um conjunto de dados ou subconjunto de variáveis, ou amostras de dados nas quais descobertas deverão ser realizadas;

A **fase 3** é responsável pela Limpeza e pré-processamento de dados: inclui operações básicas, como remoção de ruído ou outliers, se for apropriado, coletando as informações necessárias para modelar ou levar em conta o ruído, decidir sobre estratégias para lidar com a falta de campos de dados e contabilidade para informações de sequência de tempo e mudanças conhecidas, bem como decidir sobre problemas relacionados a tipo de dados, mapeamento de valores ausentes e desconhecidos, padronização de valores, normalização, entre outros;

A **fase 4** redução e projeção dos dados: inclui encontrar recursos úteis para representar os dados, dependendo do objetivo da tarefa, e usando redução de dimensionalidade ou método de transformação para reduzir o número efetivo de variáveis a serem levadas em consideração;

A **fase 5** é responsável pela escolha da função dos dados: inclui decidir o propósito do modelo derivado pelo algoritmo de mineração de dados (por exemplo, resumo, classificação, regressão , clusterização, entre outras);

A **fase 6** define o(s) algoritmo(s): inclui selecionar o método a ser usado para pesquisar padrões nos dados, como decidir quais modelos e os parâmetros podem ser apropriados (por exemplo, os modelos de dados categóricos são diferentes dos modelos de vetores) e combinar um método particular de mineração com os critérios gerais do processo KDD (por exemplo, o usuário pode estar mais interessado em compreender o modelo do que duas capacidades preditivas);

A **fase 7** minera os dados na busca de padrões de interesse com representações, que podem incluir regras de classificação ou árvores, regressão, agrupamento, modelagem de sequência, dependência e análise de linha.

A **fase 8** interpretação dos padrões minerados: inclui a interpretação dos padrões descobertos e possivelmente retornando a qualquer uma das etapas anteriores, bem como visualização dos padrões extraídos , removendo padrões redundantes ou irrelevantes e tra- duzindo os úteis para que fiquem compreensíveis pelos usuários;

A **fase 9** Utilização dos dados e conhecimento gerado para a agregação de valor e tomada de decisão. Essa fase inclui incorporar esse conhecimento ao sistema de desempenho, tomada de decisões com base neste conhecimento, ou simplesmente documentá-lo e relatá-lo às partes interessadas, bem como verificar e resolver possíveis conflitos.

A jutilização do KDD neste trabalho ocorreu através da ferramenta Oracle Data Mining e Oracle Data Miner, conforme descrita na seção seguinte.

2.2. A ferramenta Oracle Data Mining e Oracle Data Miner

O Oracle Data Mining (ODM) [Corporation 2012] é um componente do Oracle Enterprise Edition (*Oracle Advanced Analytics Option*), uma plataforma única, segura e escalável de mineração de dados que reside nativamente no núcleo do banco de dados e que pode ser aplicada nas bases de dados armazenadas também no próprio SGBD Oracle. O que contrasta com a maioria de ferramentas de Data Mining que requerem a extração dos dados do ambiente do banco, muitas vezes inviabilizando a execução do processo, por causa de alto custo e segurança. Todo o processo, desde a seleção dos dados, pré-processamento até a análise dos resultados, ocorre no ambiente do SGBD Oracle. As funções de mineração do ODM podem ser aplicadas em dados estruturados ou não. Cada função especifica uma classe de problemas de mineração que podem ser resolvidos usando algoritmos de mineração.

Oracle Data Mining suporta técnicas de aprendizado supervisionado e não supervisionado. Inclui algoritmos de Classificaçãoe Regressão, Regras de Associação, Modelos de Clustering, Seleção e Importância de Atributos [Umadevi and Marseline 2017].

Nos algoritmos supervisionados o processo de aprendizagem é dirigido por um atributo alvo previamente conhecido, ao contrário dos algoritmos não supervisionados, nos quais não há nenhum resultado conhecido anteriormente para guiar o algoritmo na construção do modelo.

A Justiça Eleitoral utiliza como Sistema Gerenciador de Banco de Dados [Elmasri and Navathe 2019] o *Oracle Enterprise Edition*. Uma das principais vantagens em utilizar o Data Mining da Oracle na mineração dos dados é que os dados não precisam ser exportados do ambiente da rede do TRE-BA, que no caso da Justiça Eleitoral, foi condição fundamental, pois, os dados dos eleitores são sigilosos e apenas a Justiça Eleitoral deve ter acesso.

3. Descoberta do conhecimento no TRE-Bahia - Método Proposto

O método desenvolvido neste trabalho envolveu as etapas do processo do KDD, conforme descrito na Figura 2.

3.1. Primeira Fase - Escolha dos dados

Este trabalho utilizou como fonte principal de dados o Cadastro de eleitores que está hospedado no TSE. Apesar do TRE-BA possuir acesso de consulta aos dados dos eleitores da Bahia, esses dados foram copiados para um SGBD local no qual poderiam ser manipulados, para que pudessem ser ajustados à necessidade deste experimento. Assim, é possível criar ou transformar os dados originais sem que haja qualquer prejuízo para as estruturas originais. Os dados originais como estão na base de dados estão apresentados na Tabela 1.

Além da tabela de eleitores, foi necessário criar cópias de outras tabelas auxiliares para que fossem utilizadas na fase de pré-processamento, tais como:

TABELA	DESCRIÇÃO
LOCALIDADE	Municípios
FASE	Eventos possíveis aos eleitores
HISTORICO FASE	eventos que ocorreram com os eleitores
OCUPACAO	Tabela com as ocupações
BAIRRO	Bairros
HISTORICO CONVOCACAO	Histórico das Convocações a trabalhos eleitorais dos eleitores
HISTORICO OPERACAO	Histórico de eventos especficos da vida do eleitor: ALISTAMENTO, TRANSFERÊNCIA, 2a VIA;

3.2. Segunda Fase - Seleção dos dados

O Cadastro de eleitores da Bahia possui 12.285.942 eleitores aptos cadastrados[1]. Destes,

2.249.129 são eleitores da capital do estado da Bahia: Salvador. Foi decidido utilizar apenas os dados de Salvador neste experimento, por causa do grande volume de dados envolvido no processamento. Além do grande volume de dados, os perfis dos eleitores da capital são diferentes dos elei-

1Dados de 28/10/2021

tores das cidades do interior. Posteriormente com o resultado obtido da análise da capital do estado, é possível estender o experimento para o interior do estado, ajustando os dados para considerar as características dos eleitores do interior.A análise também irá considerar separadamente os eleitores que faltam eleições municipais e eleições gerais [TSE 2020b]. A Bahia possui 418 municípios e 199 zonas eleitorais [TRE-BA 2021]. Os eleitores das zonas de 1 a 19 foram selecionados visto que correpondem às zonas eleitorais de Salvador. Foi então gerada uma tabela ELEITOR SSA com os dados apenas dos eleitores de Salvador.

3.3. Terceira Fase - Pré-processamento

A Tabela de ELEITOR SSA continha inicialmente todos os campos da tabela ELEITOR.Porém, foi necessário, para otimizar os experimentos, acrescentar alguns atributos que indicam a presença ou não dos eventos para cada eleitor de Salvador. Foram utilizadas as tabelas auxiliares para preencher colunas de código pelas suas respectivas descrições, tais como *ocupação*, *grau de instrução* ao *município de nascimento*, *município que realizou biometria*, *entre outros.*

O Oracle Data Miner possui suporte à transformação automática dos dados. No modo de preparação automática de dados (ADP), o próprio modelo transforma ou constroi os dados de acordo com os requisitos do algoritmo usado. As instruções de transformação são incorporadas ao modelo e utilizadas sempre que o modelo é aplicado.

Tabela 1. Atributos da tabela de ELEITOR

ATRIBUTO	DESCRIÇÃO	TIPO DE DADO	RETIRADO
COD_OBJETO	Chave primária da tabela	VARCHAR2(18)	
NUM_REGISTRO_CIVIL	Número de Registro Civil	NUMBER(12)	LGPD
NOM_ELEITOR	Nome do Eleitor	VARCHAR2(70)	LGPD
NOM_MAE	Nome da Mãe	VARCHAR2(70)	LGPD
DAT_NASC	Data de Nascimento	NUMBER(8)	
COD_SEXO	Código do Sexo (2- Masculino, 4 – feminino, 0 – outros)	NUMBER(1)	
NUM_INSCRICAO	Número do Título de eleitor	VARCHAR2(12)	LGPD
DAT_DOMIC_UF	Data de inscrição ou transferência do eleitor na UF	DATE	
DAT_DOMIC_MUNIC	Data de inscrição ou transferência do eleitor na UF	DATE	
NUM_TEL_ELEITOR	Número do telefone do eleitor	VARCHAR2(15)	LGPD
COD_GRAU_INSTR	Código do Grau de Instrução	NUMBER(1)	
COD_ESTADO_CIVIL	Código do Estado civil	NUMBER(1)	
DES_ENDERECO	Endereço do Eleitor	VARCHAR2(120)	LGPD
NUM_CEP	Número do CEP do endereço do eleitor	VARCHAR2(15)	ALTA CARDINALIDADE
COD_SIT_ELEITOR	Situação do Eleitor	NUMBER(2)	
COD_OBJETO_OCUPACAO	Código da Ocupação do eleitor	VARCHAR2(18)	
COD_OBJETO_MUNIC_NASC	Código do município de nascimento do eleitor	VARCHAR2(18)	
COD_OBJETO_SECAO	Código da Seção do eleitor	VARCHAR2(18)	
DES_COMPLEMENTO	Complemento do complemento	VARCHAR2(50)	
COD_OBJETO_BAIRRO	Código do Bairro onde mora o eleitor	VARCHAR2(18)	
COD_OBJETO_LOGRADOURO	Código do Logradouro onde mora o eleitor	VARCHAR2(18)	ALTA GRANULARIDADE
NUM_ENDERECO	Número do endereço onde mora o eleitor	NUMBER(10)	ALTA CARDINALIDADE
NOM_ELEITOR_LIMPO	Nome do eleitor sem acentos ou caracteres especiais	VARCHAR2(70)	LGPD
NOM_PAI_LIMPO	Nome do pai do eleitor sem acentos ou caracteres especiais	VARCHAR2(70)	LGPD
NOM_MAE_LIMPO	Nome da mãe do eleitor sem acentos ou caracteres especiais	VARCHAR2(70)	LGPD
NUM_TEL_ELEITOR_2	2º numero de telefone do eleitor	VARCHAR2(15)	LGPD
TIP_DOCUMENTO	Tipo do documento do eleitor	NUMBER(2)	LGPD
NUM_DOCUMENTO	Número do documento do eleitor	VARCHAR2(32)	LGPD
ORG_EXPEDIDOR	Órgão expedidor do documento	VARCHAR2(30)	IRRELEVANTE
NUM_CPF	Número do CPF do eleitor	VARCHAR2(11)	LGPD
COD_FON_NOME_ELEITOR	Código fonético do nome do eleitor	VARCHAR2(12)	LGPD
COD_FON_MAE	Código fonético do nome do pai do eleitor	VARCHAR2(12)	LGPD
COD_FON_PAI	Código fonético do nome da mãe do eleitor	VARCHAR2(12)	LGPD
COD_OBJETO_MUNICIPIO_COLETA	Código do município de coleta biométrica do eleitor	VARCHAR2(18)	IRRELEVANTE
DES_EMAIL	Email do eleitor	VARCHAR2(100)	LGPD
IND_RECEBE_EMAIL	Indica se o eleitor quer receber email	NUMBER(1)	
DES_CIDADE_RESIDE_EXTERIOR	Cidade de residência no exterior	VARCHAR2(70)	
DES_BAIRRO_RESIDE_EXTERIOR	Bairro de residência no exterior	VARCHAR2(70)	
SGL_UF	Sigla da UF do eleitor	VARCHAR2(2)	
NOM_SOCIAL_ELEITOR	Nome Social do eleitor	VARCHAR2(70)	LGPD

Entretanto, foi realizado um pré-processamento nos dados utilizando rotinas PL/SQL e DML para preencher os dados e os objetos SQL Query (que pode fazer consultas sobre os dados selecionados), Sample (que pode fazer Estratificação dos dados), TRANSFORMATION e FILTER do próprio Oracle Data Miner, que realizam normalização e transformação de tipos e mudança de dados categóricos para numéricos e vice-versa. As colunas que de alguma forma pudessem identificar os eleitores ou que fossem irrelevantes à análise foram retiradas, tais como atributos com alta cardinalidade, que não conseguem capturar informações relevantes dos eleitores. Existe um histórico de eventos para cada eleitor, chamados ASÉs, que foram consideradas no experimento. Para cada tipo de evento foram adicionadas na tabela de ELEITOR SSA duas novas colunas, uma para o evento ocorrido na eleição de 2018(eleição municipal) e a outra para o mesmo evento ocorrido na eleição de 2020 (eleição geral). Estas colunas foram preenchidos com os valores 1 ou 0, se o eleitor possui este evento ou não, respectivamente.

Os ASEs considerados foram:

ASE 167	JUSTIFICATIVA DE AUSÊNCIA ÀS URNAS
ASE 396	ELEITOR COM DEFICIÊNCIA
ASE 94	AUSÊNCIA ÀS URNAS
ASE 183	CONVOCAÇÃO PARA OS TRABALHOS ELEITORAIS
ASE 205	HABILITAÇÃO PARA OS TRABALHOS ELEITORAIS
ASE 442	AUSÊNCIA AOS TRABALHOS ELEITORAIS OU ABANDONO DA FUNÇÃO
ASE 612	REGISTRO INDIVIDUAL DE PAGAMENTO DE MULTA ELEITORAL
ASE 280	DESATIVAÇÃO DA HABILITAÇÃO PARA OS TRABALHOS ELEITORAIS

Além destes eventos (ASES´s), foram consideradas operações básicas que podem ocorrer com os eleitores tais como:

- **Data do primeiro alistamento**: É quando o cidadão passa a ser chamada de ELEITOR para a JustiçaEleitoral;
- **Data da última transferência**: O Eleitor pode solicitar alteração da zona, município e até unidade federativa onde vota;
- **Data da última revisão eleitoral**: Quando o Eleitor altera dados cadastrais, tais como, endereço, escolaridade, estado civil, colhe digitais biométricas, entre outras.
- **Data de solicitação de Segunda via**: Quando o eleitor solicita a impressão de novo título em papel. Cada vez menos solicitada com a implantação do aplicativo e-Título [TSE 2020a].

Entretanto, para que estes dados pudessem ser manipulados pela ferramenta Oracle Data Miner, que não consegue utilizar os campos do tipo DATA, foram todos convertidos para TEMPO.

Então:

DAT NASC - Adicionado o campo IDADE

DAT DOMIC MUNIC - Adicionado o campo TEMPO DOMIC MUNIC

DAT DOMIC UF - Adicionado o campo TEMPO DOMIC UF

DAT ULT REVISAO - Adicionado o campo TEMPO ULT REVISAO

DAT ALISTAMENTO - Adicionado o campo TEMPO ALISTAMENTO

DAT ULT 2VIA - Adicionado o campo TEMPO ULT 2VIA

DAT ULT TRANSF - Adicionado o campo TEMPO ULT TRANSF

3.4. Quarta Fase - Transformação

Não foi aplicada diretamente nenhuma técnica de redução de dimensionalidade nos dados. A própria ferramenta, Oracle Data Miner, seleciona os dados relevantes à análise no momento da aplicação dos algoritmos, retirando os atributos que tenham alta cardinalidade. Estas transformações são informadas pela ferramenta no momento da aplicação do algoritmo (Figura 3).

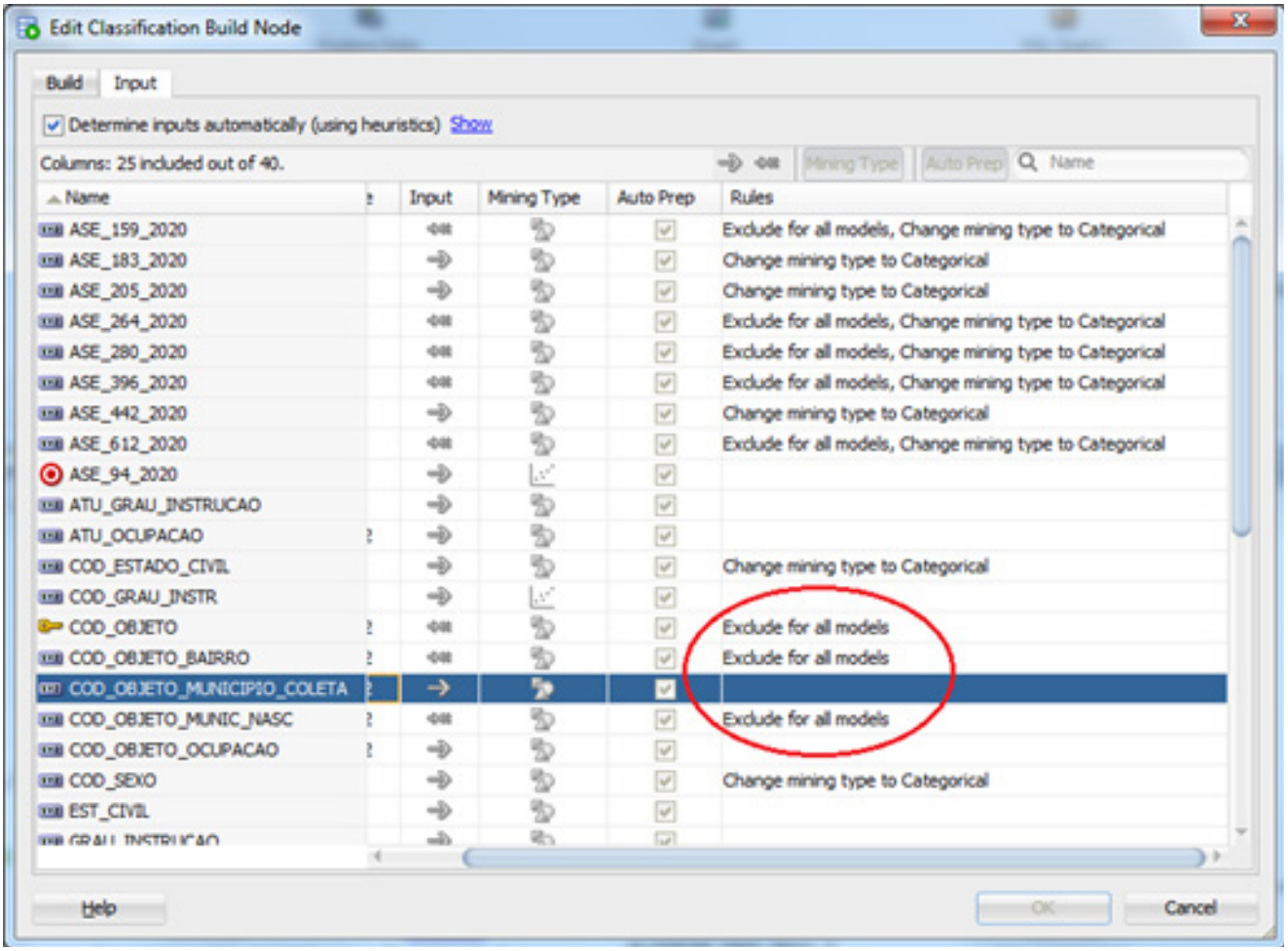


Figura 3. Eliminação de colunas pelo Oracle Data Mining

3.5. Quinta Fase - Escolha da Tarefa

A tarefa de **Classificação** foi escolhida visto que os dados eram rotulados através da identificação dos eleitores que já faltaram a alguma eleição passada. Foram selecionadas duas eleições para rotular os dados : A eleição de 2018 e a eleição de 2020. Então as colunas *ASE 94 2018* e *ASE 94 2020* foram preenchidas com 0 se o eleitor não faltou a estas eleições e com 1 se o eleitor faltou.

3.6. Sexta Fase - Escolha dos Algoritmos

O Oracle Data Miner implementa quatro algoritmos de classificação: Árvore de decisão (DT), Naive Bayes (NB), Suport Vector Machine(SVM) e Modelo Linear Generalizado (GLM). Com o intuito de melhor analisar o desempenho dos algoritmos para a tarefa definida, os quatro algoritmos foram utilizados nos experimentos realizados neste trabalho.

3.7. Sétima Fase - Mineração dos dados

As bases de dados foram utilizadas separadamente para os experimentos realizados. O primeiro experimento referente aos eleitores que faltam as eleições municipais (2018) e outro experimento para as eleições gerais (2020).

3.7.1. Critérios e Métricas de Avaliação do Oracle Data Miner

Em todos os experimentos realizados, independente do ano eleitoral, foram utilizados os seguintes critérios:

ATRIBUTO ALVO - ASE 94 2018, para 2018, e ASE 94 2020, para 2020

ATRIBUTO CHAVE - COD OBJETO (chave primária da tabela de ELEITOR)

ÁRVORE DE DECISÃO - Métrica GINI e profundidade máxima entre 5 e 7

SVM - Kernel - LINEAR

O Oracle Data Miner calcula as várias métricas para modelos de classificação[ORACLE 2020]. Foram utilizadas as seguintes métricas nos experimentos realizados:

Confiança Preditiva (*Predictive Confidence*): fornece uma estimativa da precisão do modelo. A confiança preditiva é um número entre 0 e 1. O Oracle Data Miner exibe a confiança preditiva como uma porcentagem. Por exemplo, a confiança preditiva de 59 significa que a confiança preditiva é 59 por cento (0,59). A confiança preditiva indica o quanto as previsões feitas pelo modelo testado são melhores que as previsões feitas por um modelo *Naive*. O modelo Naive sempre prevê a média para alvos numéricos e a moda para alvos categóricos;

Acurácia Média (*Average Accuracy*): A precisão média refere-se à porcentagem de previsões corretas feitas pelo modelo quando comparada com as classificações reais nos dados de teste;

Precisão Geral(*Overall Accuracy*) : A precisão geral se refere à porcentagem de previsões corretas feitas pelo modelo quando comparadas com as classificações reais nos dados de teste;

Matriz de confusão (*Performance Matrix*) : Exibe o número de previsões corretas e incorretas feitas pelo modelo em comparação com as classificações reais nos dados de teste;

Curva ROC : A área sob a curva ROC mede a capacidade discriminativa de um modelo de classificação binário. Quanto maior a área abaixo da curva melhor é a predição do modelo;

4. Experimentos e Resultados sobre eleições de 2018

Vários experimentos foram realizados para o ano eleitoral de 2018, entretanto, apenas o melhor desempenho é apresentado neste trabalho.

4.1. Conjunto de Dados

Os atributos que se referenciem aos eventos ocorridos nas eleições de 2020 (ASE 94 2020, ASE 264 2020, ...) foram removidos. O Oracle Data Miner possui um objeto chamado

Sample que faz a estratificação automática dos dados com opção de gerar esta estratificação de forma balanceada. Entretanto, como era desejável utilizar no experimento, quantidade de dados iguais para cada zona eleitoral, pois, cada zona mapeia bairros e regiões geográficas da cidade de Salvador, o balanceamento foi realizado de forma manual, ao invés da utilização do objeto Sample do Oracle Data Miner. Este balanceamento foi realizado através de consultas SQL, selecionando 20.000 linhas de cada zona eleitoral, onde 10.000 eram de eleitores faltosos e 10.000 de eleitores não faltosos. Em um dos experimentos, foram utilizados os dados não balanceados. O que ocorre é que como a quantidade de eleitores que não falta às eleições (85%) é maior que a que falta (15%), o algoritmo acaba enviesando os resultados para os eleitores que não faltam às eleições. Após o balanceamento, foram extraídas da tabela de eleitores de Salvador 228.810 linhas.

Na Figuras 4, é apresentada a estatísticas do ASE 94 antes e depois da estratificação dos dados.

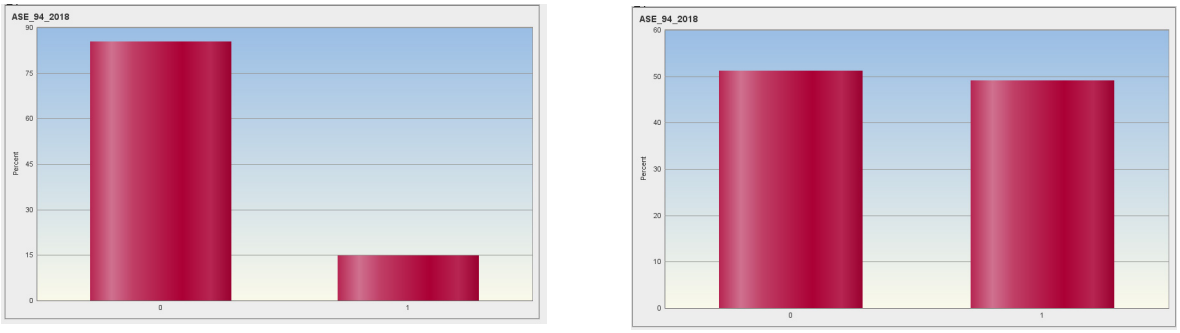


Figura 4. Histograma de ASE 94 em 2018 antes e depois do balanceamento

4.2. Resultados preliminares de 2018

Os algoritmos que apresentaram melhores resultados foram Naive Bayes e a Árvore de Decisão, como apresentado nas Figuras 5 e 6.

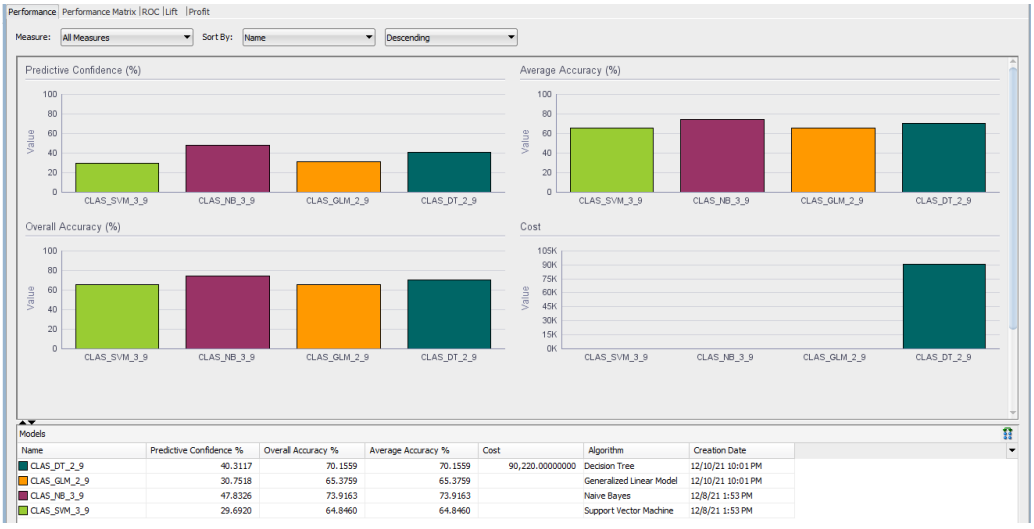


Figura 5. Comparação dos Algoritmos - Experimento - 2018

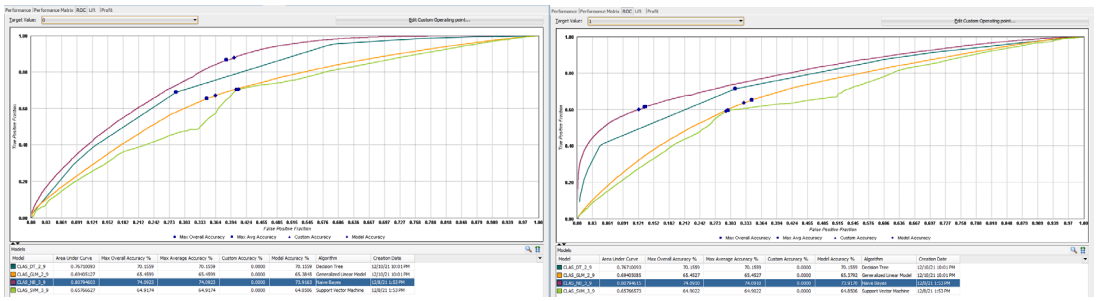


Figura 6. Curva ROC para 2018 para o valor 0 (esquerda) e para valor 1 (direita)

A Matriz de confusão dá Árvore de decisão está apresentada na Figura 8. Ela é resultado da análise da aplicação do algoritmo no conjunto de dados que foi reservado para testes. Pode-se observar que 52.030 eleitores foram classificados corretamente como não faltantes (TN - True Negatives), 23.536 eleitores foram classificados equivocadamente como faltantes (FP - False Positives), 21.546 foram classificados equivocadamente comonão faltantes (FN - False Negatives) e 54.012 foram classificados corretamente como eleitores faltantes. O Percentual de acertos para os True Positives(TP) é de 70,69% e o percentual de acertos para os True Negatives(TN) é de 69,64%.

A profundidade da árvore foi limitada a 5 níveis . Em alguns experimentos os níveis chegaram a 10 níveis , mas tornou a árvore de difícil interpretação. Foram considerados apenas as folhas ou nós que tiveram mais de 90% de precisão .

Performance Matrix: Rows=Actual, Columns=Predicted:

	0	1
0	52,030	23,546
1	21,564	54,012
Total	73,594	77,558
Correct %	70.6987	69.6408
Cost	43,128.0000	47,092.0000

Figura 7. Matriz de confusão da árvore de decisão de 2018

Em relação à Árvore de Decisão (Figura 9) observa-se que :

- 100% dos Eleitores com até; 2,5 anos de tempo de alistamento Não faltam as eleições
- 93,7% dos Eleitores com tempo de alistamento acima de 22,5 e que estejam comoeleitores da Bahia entre 23,5 e 27,5 anos faltam as eleições ;
- 95,83% dos Eleitores com tempo de dimicílio na Bahia entre 32,5 a 34,5 anos ecom mais de 50,5 anos faltam as eleições ;
- 97,1% dos Eleitores que estejam em Salvador entre 15,5 e 18,5 anos faltam as eleições;

5. Experimentos e Resultados sobre eleições de 2020

Neste experimento foi considerado como atributo alvo ASE 94 2020. Foram realizados vários experimentos e, como em 2018, será detalhado apenas o de melhor desempenho.

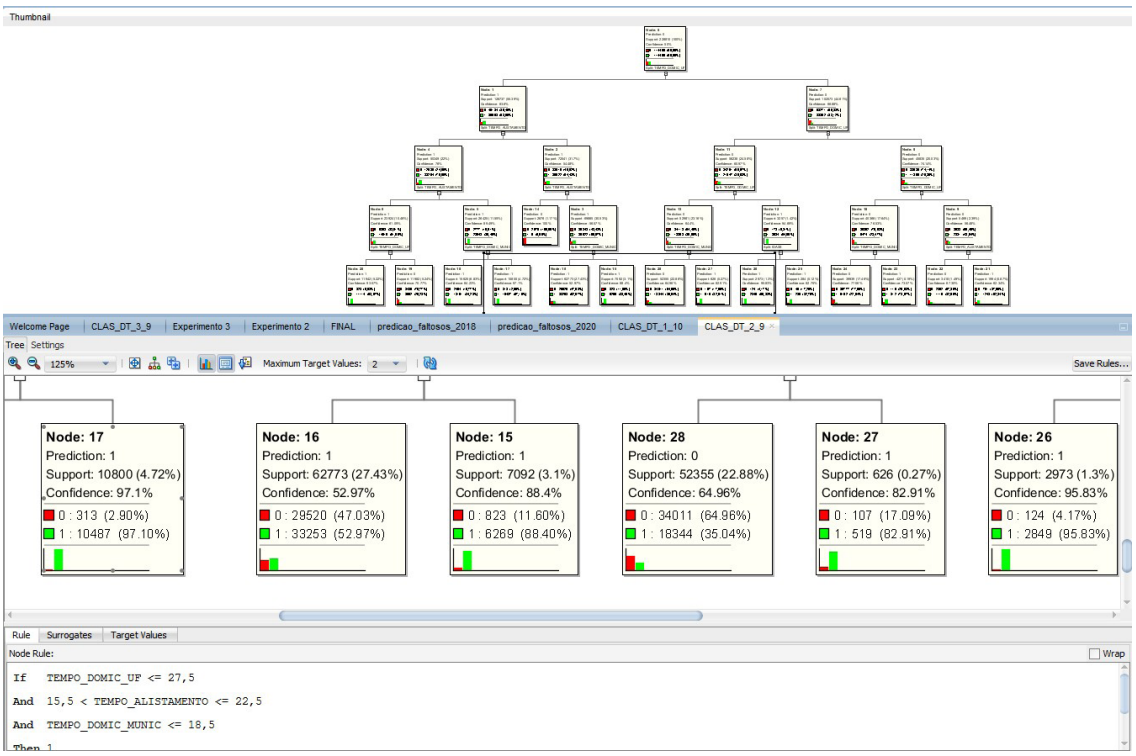


Figura 8. Resultado da Árvore de Decisão para 2018

5.1. Conjunto de Dados

Os atributos que se referenciavam a eventos ocorridos nas eleições de 2018 (ASE 94 2018, ASE 264 2018, ...) foram suprimidos da base. Foram aplicados os algoritmos e os resultados estão apresentados na Figura 10.

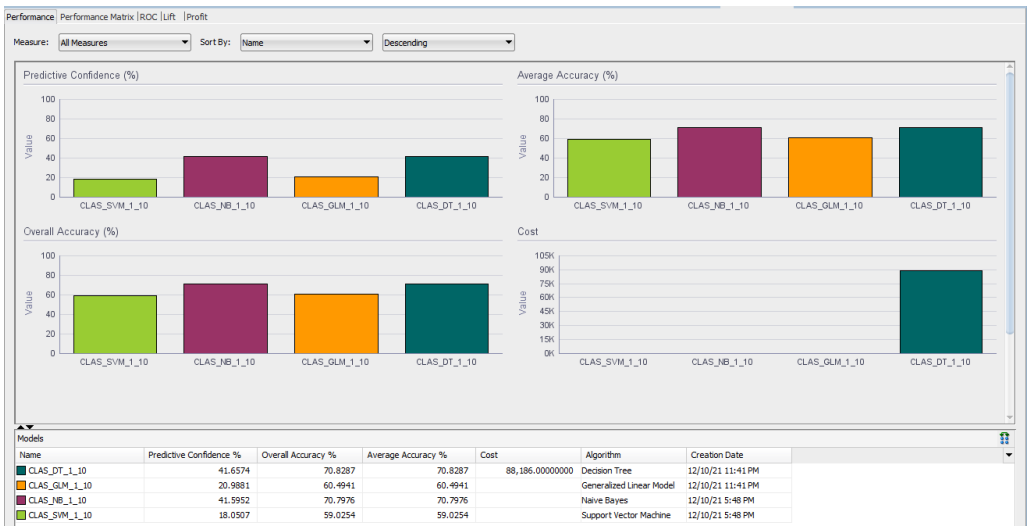


Figura 9. Comparação dos Algoritmos - Experimento - 2020

Semelhante ao que foi realizado com o experimento em 2018, foi realizado o balanceamento manual, utilizando consultas SQL. Da mesma forma que em 2018, o balanceamento foi realizado, selecionando 20.000 linhas de cada zona eleitoral, onde 10.000 eram de eleitores faltosos e 10.000 de eleitores não faltosos.

O percentual de faltantes em 2020 foi de quase 21%, maior que o de 2018 que foi de 15%. A estatística dos eleitores faltantes em 2020 antes e depois da estratificação dos dados está representada na Figura 11.

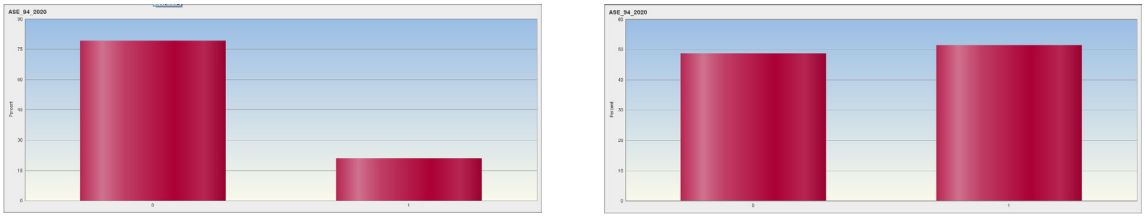


Figura 10. Histograma de ASE 94 em 2020 antes e depois do balanceamento

5.2. Resultados Preliminares para 2020

O Algoritmo com melhor desempenho foi a Árvore de Decisão como mostram a matriz de confusão (Figura 12) e a curva ROC (Figura 13).

Performance Matrix: Rows=Actual, Columns=Predicted:

	0	1
0	64,158	11,418
1	32,675	42,901
Total	96,833	54,319
Correct %	66.2563	78.9797
Cost	65,350.0000	22,836.0000

Figura 11. Matriz de confusão da Árvore de decisão - 2020

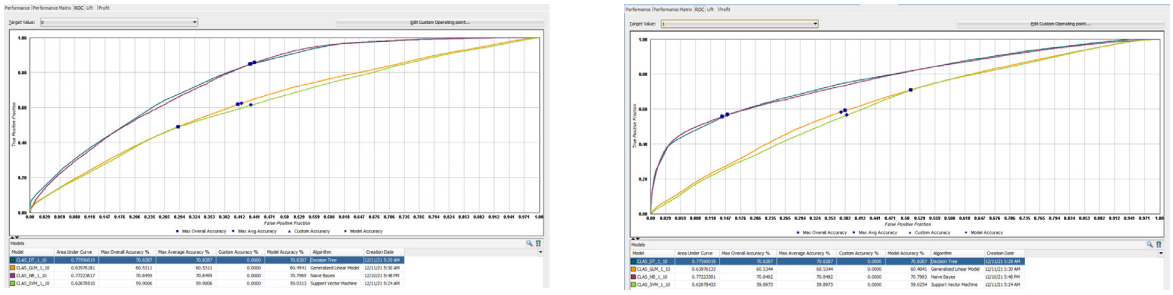


Figura 12. Curva ROC para 2020 para o valor 0 (esquerda) e para valor 1 (direita)

A Matriz de confusão mostra que o percentual de acertos do algoritmo aplicado no conjunto de dados que foi reservado para testes. Pode-se observar que foram classificados corretamente como eleitores não faltantes 64.128 eleitores (TN- True Negatives), classificados equivocadamente como eleitores faltantes 11.418 (FP - False Positives), 32.675 foram classificados equivocadamente como eleitores não faltantes (FN - False Negatives) e 54.319 foram classificados corretamente como eleitores faltantes (TP - True Positives).

Isto corresponde a um percentual para os True Positives (TP) de 66,25% e para os True Negatives(TN) de 78,97%.

A profundidade da a´rvore foi limitada a 7 níveis, pois, a utilizac¸a˜o de uma profundidade maior, inviabilizou sua análise. Foram considerados apenas as folhas ou no´s que tiveram mais de 90% de precisão.

Analisando a Árvore de Decisão, observa-se que:

- 100% dos eleitores que estão na Bahia a mais de 32,5 e menos que 34,5 anos faltam a eleições ;
- 99,63% dos Eleitores que estão na Bahia a menos de 32,5 anos, estão em Salvador a mais de 27,5 anos, mas fizeram revisão eleitoral a menos de um ano não faltam eleições ;
- 92,13% dos eleitores que tem mais de 32,5 anos na Bahia e têm menos que 55,5 anos faltam eleições ;
- 97% dos Eleitores que estão em Salvador entre 10,5 e 15,5 anos que fizeram a última revisão eleitoral há menos de um ano não faltam eleições ;
- 98,34% dos Eleitores que estão na Bahia entre 20,5 e 21,5 anos faltam eleições ;
- 97,86% dos Eleitores que estejam em Salvador entre 10,5 e 15,5 anos e fizeram revisão eleitoral no último ano não faltam eleições .
- 96,67% dos Eleitores que estejam na Bahia ha´ mais de 32,5 anos e fizeram revisão eleitoral no último ano não faltam eleições .
- 97,1% dos Eleitores que estejam em Salvador entre 15,5 e 18,5 anos faltam eleições .

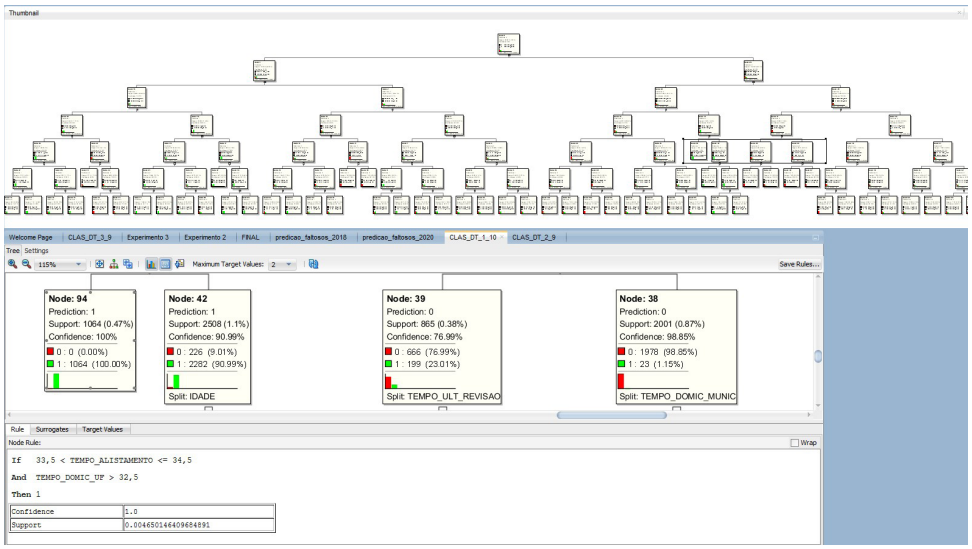


Figura 13. Resultado da Árvore de Decisão para 2020

6. Análise e Discussões

Em 2018 observou-se que os eleitores mais novos e com menos tempo de dimicílio no estado ou em Salvador são inclinados a não faltar as eleições. Enquanto os que são mais velhos ou têm mais tempo de dimicílio no estado, ou município de Salvador estão inclinados a faltar a eleições .

Em 2020 os eleitores que estão na Bahia ou em Salvador a mais tempo também estão inclinados a faltar a eleições . Entretanto, se tiverem realizado revisão eleitoral a menos de um ano, estão inclinados a não faltar.

7. Considerações Finais

Durante a pesquisa para realização deste trabalho, não foram encontrados trabalhos que utilizassem o cadastro de eleitores para esta mesma finalidade, o que dificultou um comparativo com

trabalhos relacionados. Esses resultados preliminares demandam de novas análises com o intuito de proferir um resultado conclusivo.

Diante do exposto, observaram-se indícios de que em ambos os anos analisados os atributos determinantes para a falta ou não do eleitor estão associados ao tempo, seja o tempo em que o eleitor se alistou, o tempo em está como eleitor do estado da Bahia ou como eleitor do município de Salvador. Em ambas as eleições foi realizado um experimento retirando estes atributos de tempo do conjunto de dados, na tentativa de melhorar o desempenho, mas o desempenho caiu em média 50% em relação em comparação aos experimentos em que estes atributos estavam presentes. Desta forma, há uma tendência em concluir para estas eleições e este perfil de eleitorado o tempo realmente foi um fator determinante para a identificação do eleitor que pode vir a faltar uma eleição.

7.1. Ameaças

O TSE instituiu uma norma para a implantação da Política de Segurança da Informação [TSE]. Desta forma, se todo o processamento envolvido na mineração dos dados não fosse realizado completamente em ambiente da JustiçaEleitoral, talvez este projeto não pudesse ter sido realizado. Então a utilização da ferramenta Oracle Data Miner foi fundamental para este projeto.

Foi escolhido para a execução dos experimentos um banco de teste, entretanto, ele está hospedado na mesma máquina que nosso banco de produção. Se este processamento sobrecarregasse a máquina a ponto de atrapalhar as aplicações dos usuários ou uso do banco de produção, isto também poderia inviabilizar a execução deste trabalho.

O Banco de Dados Oracle do TRE-BA não está instalado em sua última versão. Por este motivo, alguns recursos na ferramenta não estavam disponíveis na versão utilizada. Desta forma, se não tivesse executado os algoritmos adequadamente o projeto também não teria sido executado.

A pouca experiência com a ferramenta utilizada, Oracle Data Miner, poderiam ter sido impeditivas para obtenção de quaisquer resultados. Se houvesse mais tempo para o amadurecimento e desenvolvimento de outros experimentos, poderíamos ter obtido melhores resultados.

A ferramenta não foi configurada em ambiente distinto do ambiente do banco de produção. Não se pode instalar ou mesmo modificar muitos parâmetros, pois, poderia afetar de alguma forma o ambiente de produção. Se a ferramenta não tivesse conseguido ser executada sem alteração do ambiente o projeto poderia não ter sido implementado.

7.2. Trabalhos Futuros

Foi realizada análise apenas com os eleitores de dados de Salvador. Um próximo trabalho seria preparar os dados para eleitores do interior do estado, separando os municípios porquantidade de eleitores ou mesmo região geográfica e aplicando os algoritmos pra estes novos dados.

Como um trabalho futuro poderíamos aplicar o modelo gerado para o experimento de 2018 nos dados de 2020 e verificar o seu resultado.

Além disso, foram utilizadas nesta análise apenas as faltas nas duas últimas eleições , 2018 e 2020. Este trabalho poderia ser expandido para utilizar um número maior de eleições (2016, 2014, ...) ou mesmo analisar as faltas independentemente do tipo da eleição.

O modelo gerado por este trabalho foi aplicado a um conjunto de eleitores de Salvador. Estes dados estão armazenados em uma tabela do banco. Uma análise poderá ser realizada após a eleição de 2022 para verificar a precisão do que foi predito pelo algoritmo deste trabalho.

Poderia ser utilizada outra ferramenta de mineração de dados, utilizando os mesmos dados utilizados neste trabalho, com o objetivo de comparar os resultados obtidos entre as duas ferramentas. Os dados deveriam ser anonimizados para que possam ser utilizados em ambiente externo ao banco da Justiça Eleitoral.

Além dos dados de eleitores, existe uma quantidade enorme de dados dentro do TRE-BA que poderia ser beneficiada com a mineração de dados. Como um exemplo, a Seção de compras poderia utilizar a mineração de dados para identificar quais itens do estoque de suprimentos acabam ao mesmo tempo, podendo com isso iniciar processos de compras conjuntamente. Pode-se fazer um levantamento para identificar que áreas do TRE-BA poderiam ser beneficiadas de alguma forma com a utilização da mineração de dados.

Referências

Center, O. H. (1991). Home / database / oracle database online documen- tation 12c, release 1 (12.1) / data warehousing and business intelligence. site Oracle https://docs.oracle.com/database/121/DMCON/GUID-2116E665-721E- 4EBA-AFE1-A30D6E8078C6.htmDMCON620.

Corporation, O. (2012). Oracle data mining 11g release 2 competing on in-database analy- tics. site Oracle https://www.oracle.com/technetwork/database/options%20/advanced- analytics%20/odm/twp-data-mining-11gr2-160025.pdf.

EJE (2021). Eje - escola judici eleitoral no tre-ba. site TRE-BA http://eje.tre-ba.jus.br/. Elmasri, R. and Navathe, S. B. (2019). Sistemas de Banco de Dados. Pearson Universidades, 7 th edition.

Fayyad, Shapiro, and Smith (1996). The kdd process for extracting useful knowledge from volumes of data. COMMUNICATIONS OF THE ACM site, 39.

Fayyad, Shapiro, and Smith (1998). Mining databases: Towards algorithms for kno- wledge discovery. Bulletin Of The IEEE Computer Society Technical Committee On Data Engineering, 21.

Koul, R. (2020). Overview of data mining. International Journal of Trend in Scientific Research and Development, 4(4):1333–1336.

ORACLE (2020). Testing classification models. site ORACLE https://docs.oracle.com/cd/E5574701/doc.41/e58114/test.htm#DMRUG818.

Senado, A. (2020). Lei geral de proteção de dados entra em vigor. Senado notícias.

TRE-BA (2021). Zonas eleitorais. site TRE-BA https://www.tre-ba.jus.br/o-tre/zonas- eleitorais/zonas-eleitorais.

TSE. Tse institui norma para a implantação da política de segurança da informação. site TSE https://www.tse.jus.br/imprensa/noticias-tse/2021/Julho/tse-institui-norma-para- a-implantacao-da-politica-de-seguranca-da-informacao.

TSE (2014). Eleições no brasil , uma história de 500 anos. site TSE https://www.tse.jus.br/hotsites/catalogo-publicacoes/pdf/tse-eleicoes-no-brasil-

uma-historia-de-500-anos-2014.pdf.

TSE (2020a). Aplicativo e-título. site TSE https://www.tse.jus.br/eleitor/servicos/aplicativo-e-titulo.

TSE (2020b). Entenda a diferenc¸a entre eleições gerais e municipais. site TSE https://www.tse.jus.br/videos/entenda-a-diferenca-entre-eleicoes-gerais-e-municipais.

Umadevi, S. and Marseline, K. S. J. (2017). A survey on data mining classification algo- rithms. In 2017 International Conference on Signal Processing and Communication *(ICSPC)*, pages 264–268.

Neima Prado dos Santos
Tribunal Regional Eleitoral da Bahia
1ª Av. do Centro Administrativo da Bahia, 150 – CAB
Salvador-BA - CEP: 41.745-901 - Brasil
npsantos@tre-ba.jus.br

Principais fatores de sucesso em projetos de desenvolvimento de software

Abstract. *The central role of software today makes it increasingly necessary to know the factors that contribute to its successful construction. Recent research has shown that despite the efforts used to study practices and techniques, there are still many projects that do not reach the expected success. Through the survey of works associated with the theme, we identified organizational factors that may increase the chance of success in software development projects.*

Resumo. *O papel de destaque do software na atualidade torna cada vez mais necessário conhecer os fatores que contribuem para sua construção bem sucedida. Pesquisas recentes têm apontado que a despeito dos esforços empregados no estudo de práticas e técnicas, muitos ainda são os projetos que não alcançam o sucesso esperado. Por meio do levantamento de trabalhos associados ao tema, foram identificados fatores gerenciais que podem aumentar a chance de obter sucesso no desenvolvimento dos projetos de desenvolvimento de software.*

1. Introdução

Nos primórdios da humanidade, o homem precisou construir um abrigo para as intempéries da natureza ou um arco para melhorar suas chances de conseguir alimento. Avançando um pouco mais, os egípcios tiveram a necessidade de construir um local onde pudessem esperar pela outra vida. Por sua vez, o povo chinês idealizou sua grande muralha para se defender das ameaças de invasão por outros povos [Berkun 2008, Watt 2014].

A partir desses exemplos, é possível perceber então que a humanidade executa projetos desde os primórdios da sua existência, sendo que o sucesso de alguns desses projetos pode ter tido papel crucial na sobrevivência humana. A capacidade de caçar animais de grande porte e dispor de uma alimentação diferenciada, trazida pela utilização do arco e flecha, foi uma das vantagens que permitiram a prevalência do homem moderno sobre outras espécies. Com a construção de abrigos, nossos ancestrais se protegeram de ameaças como clima hostil e animais selvagens [ScienceFocus 2019, History 2021].

À medida que a caça foi sendo substituída pela agricultura como forma de subsistência, criando terreno para o surgimento das cidades e a chegada da civilização, os projetos foram ficando cada vez maiores e mais complexos, a exemplo da construção das pirâmides de Gizé. Os números relacionados à sua construção, assim como sua grandiosidade, são substanciais. Estima-se que foram necessários 20 anos de trabalho, entre 2580 e 2560 AC, e cerca de 20.000 trabalhadores, em um projeto que exigiu planejamento e execução cuidadosos para o seu êxito [Hozac-Holland 2011, Richman 2011].

Grande parte dos projetos da atualidade é realizada com o apoio de ferramentas de software, ou tendo o software como a totalidade ou parte do produto sendo desenvolvido. Áreas como medicina, comércio, transporte, energia, justiça e governo, não teriam alcançado tantos avanços sem o auxílio do software. Dentre os diversos exemplos de emprego do software, podemos citar desde veículos espaciais, automóveis não-tripulados e robôs cirurgiões, a aplicativos de comércio eletrônico, assistentes digitais e chips de smartphones.

Justamente pela sua abrangência, e sua participação em aplicações de importância crítica, seja pela natureza da área de emprego, quantidade de usuários e impacto na reputação da organização, é cada vez mais necessária a adoção de práticas que permitam construir software que atenda as necessidades para as quais foi idealizado, em tempo hábil ao atendimento da sua finalidade, observando atributos que favoreçam seu uso, evolução, manutenção e segurança. Junte-se a isso o fato de o desenvolvimento de software ser considerado uma atividade de grande complexidade [Brooks 1987, Reel 1999, Valente 2020], uma vez que cada software é único e, portanto, possui requisitos específicos, o hardware em que opera é diverso e possui grande número de estados, e sua complexidade aumenta com o tamanho.

O relatório CHAOS (*Comprehensive Human Appraisal for Originating Software*) mais recente do *Standish Group*, publicado em 2020, com base em dados de projetos dos últimos cinco anos, reporta que somente 31% dos projetos foram considerados bem sucedidos, sendo que 50% foram considerados em risco de não obterem sucesso, e 19% simplesmente fracassaram. Por sua vez, o relatório CISQ (*Consortium for Information & Software Quality*) de 2020 estima um impacto de $260 bilhões causado pelos projetos de software mal sucedidos nos Estados Unidos.

No âmbito do Tribunal Regional Eleitoral da Bahia, sistemas são implantados, desenvolvidos e mantidos para apoiar suas várias atividades. As características de alta densidade populacional, extensão territorial, infraestrutura heterogênea e níveis diversos de familiaridade com ferramentas computacionais pelos usuários, são alguns dos desafios encontrados na construção de um produto que entregue as funcionalidades necessárias, no momento esperado e com desempenho adequado. Fica então evidenciada a pertinência de estudar e aplicar as melhores práticas da área que promovam o sucesso dos sistemas desenvolvidos.

Ainda que alguns defendam que o desenvolvimento de sistemas não deve ser realizado no contexto de projetos [Johnson and Mulder 2020a], consideramos que conhecer os fatores de sucesso no desenvolvimento de software pode ajudar na construção de software que atenda ao propósito para o qual foi idealizado, por meio das melhores práticas disponíveis, independentemente de essa construção ser parte de um projeto formal ou não. O verdadeiro propósito do software é a solução de problemas por meio da entrega de um produto de valor, no tempo necessário, que contribua com o sucesso das atividades para as quais se propõe a auxiliar. Dessa forma, o objetivo deste trabalho é contribuir no levantamento dos principais fatores existentes na literatura que atuam no sucesso do desenvolvimento de software, cuja aplicação pode tornar mais provável que o software construído atenda ao propósito para o qual tenha sido idealizado.

Nesse intuito, foram estudados trabalhos relacionados à área de gerenciamento de projetos e desenvolvimento de software, disponíveis nos principais veículos da área, procurando identificar os fatores mais comumente associados ao sucesso no desenvolvimento de software sob a perspectiva do seu gerenciamento. O resultado desse estudo foi então apresentado, com o propósito de contribuir na avaliação das práticas existentes que melhor possam contribuir com essa atividade.

As demais seções deste artigo estão estruturadas como se segue: a Seção 2 apresenta um breve histórico das disciplinas de Gerenciamento de Projetos e Engenharia de Software que ajuda a compreender o papel de destaque do gerenciamento de projetos de software na atualidade. A Seção 3 discorre sobre os fatores de sucesso encontrados na literatura da área, de modo a contribuir na decisão de quais práticas têm o maior potencial de promover o sucesso no desenvolvimento de software. E finalmente a Seção 4 versa sobre a conclusão e possíveis trabalhos futuros.

2. Breve Histórico

2.1 Gerenciamento de projetos

Embora projetos de grande porte tenham sido executados por anos e anos, somente no final do século XIX o termo Gerenciamento de Projeto começou a ganhar popularidade. A construção da Rodovia Transcontinental nos Estados Unidos foi um projeto de larga escala com decisões que forneceram a base a uma metodologia de gerenciamento de projetos [Watt 2014]. A NASA e o Departamento de Defesa dos Estados Unidos estabeleceram padrões de gerenciamento que seus contratados deveriam seguir para projetos críticos tais como aqueles das naves espaciais Polaris e Apollo [Shenhar and Dvir 2004, Larry 2011].

No ano de 1911, Frederick Taylor detalhou seus estudos sobre o trabalho no livro intitulado "Os Princípios do Gerenciamento Científico". Seu assistente e discípulo Henry Gantt estudou em grande detalhe a ordem das operações no trabalho e ficou conhecido por ter criado em 1910 o gráfico de Gantt, utilizado até os dias atuais no gerenciamento de projetos [Shenhar and Dvir 2004, Watt 2014].

Os anos 50 marcaram o início do gerenciamento de projetos moderno com o desenvolvimento de dois modelos: a técnica de avaliação e revisão de programa (PERT) e o método do caminho crítico (CPM). O PERT foi desenvolvido no âmbito do programa da Marinha Americana de construção dos mísseis submarinos Polaris, e trata principalmente da análise das tarefas envolvidas na execução de um projeto, tais como o tempo demandado por cada tarefa, as dependências entre elas e tempo mínimo necessário para completar todo o projeto. O CPM foi elaborado com foco em projetos de manutenção de plantas industriais, sendo que o caminho crítico que dá nome à técnica consiste no conjunto mínimo de tarefas que precisam ser completadas no projeto [Jenkins 2005, Barron 2019].

Do ponto de vista profissional, a institucionalização do processo de gerenciamento de projetos teve início em 1965 com a criação da primeira associação mundial de gerenciamento de projetos, conhecida atualmente como a IPMA (Associação Internacional de Gerenciamento de Projetos). Quatro anos mais tarde, o Instituto de Gerenciamento de Projetos (PMI) foi fundado por cinco voluntários, com o objetivo inicial de estabelecer uma organização em que membros pudessem compartilhar experiências. O primeiro exame de certificação PMP foi conduzido em 1984 e, em 1987, o PMI publicou sua primeira versão do PMBOK (*PMI Management Body of Knowledge*) [Seymour 2014, Barron 2019].

O Brasil foi o primeiro país a constituir uma filial (*Chapter*) do PMI fora dos Estados Unidos, no início da década de 80, que foi destituída em 1984, e novamente estabelecida em 1990 (PMI-SC).

2.2 Surgimento da Engenharia de Software

Similarmente ao gerenciamento de projetos, a história da computação evoluiu com o esforço individual e coletivo, que culminou na engenharia de software como conhecemos atualmente.

Embora a primeira teoria sobre o software seja atribuída a Alan Turing, considerado o pai da computação moderna por ter criado o modelo matemático dos computadores utilizados atualmente [ScientificAmerican 2012, Filho 2007], o primeiro a empregar de forma impressa o termo software foi John Turkey em 1958, para denotar rotinas, compiladores e outros aspectos da programação, em oposição ao termo hardware, representativo de tubos, transistores, fios, fitas e correlatos [The Economist 2000].

O termo Engenharia de Software surgiu no final de 1960, no âmbito da conferência da OTAN (Organização do Tratado do Atlântico Norte) de 1968, cujo objetivo era chamar atenção para a "crise do software", abordando os problemas identificados no desenvolvimento de software, já que com a popularização dos computadores surgiram demandas por sistemas maiores e mais complexos, que desafiavam o modo de trabalho da época [Filho 2007, Niklaus 2008, Valente 2020].

A primeira sociedade de computação no mundo foi criada em 1947 com o nome de *Eastern Association for Computing Machinery*. Em 1948, passou a se chamar apenas *Association for Computing Machinery* (ACM) [Wazlawick e Silva Junior 2021].

Em 1963, as sociedades *American Institute of Electrical Engineers* (AIEE) e *Institute of Radio Engineers* (IRE) se juntaram formando o IEEE (I*nstitute of Electrical and Electronics Engineers*) [IEEE 2020]. Por sua vez, a IEEE *Computer Society* tem sua origem no Subcomitê em Dispositivos de Computação em Larga Escala, formada no âmbito do AIEE em 1946, e no Grupo Profissional em Computadores Eletrônicos, criado cinco anos mais tarde pela IRE [IEEE 2021].

No Brasil, com a evolução do contexto industrial da computação, foi fundada a Sociedade dos Usuários de Computadores Eletrônicos (SUCESU), em 1965. A Sociedade Brasileira de Computação (SBC) foi fundada em 1978, com o objetivo de reunir a comunidade científica e acadêmica de computação do país. [Wazlawick e Silva Junior 2021].

Na atualidade, o software continua sendo a tecnologia singular mais importante do cenário mundial, já que o mundo moderno como conhecemos não poderia existir sem as vantagens do software [Somerville 2011, Pressman e Maxin 2015].

As definições de software são diversas na literatura. Na sua definição do termo software, Somerville (2011) considera que *"Softwares são programas de computador e documentação associada. Produtos de software podem ser desenvolvidos para um cliente específico ou para o mercado em geral."* Pressman e Maxin (2015) fornecem a seguinte definição *"Software se trata de: (1) instruções (programas de computação) que quando executadas provêm requisitos desejados, funcionalidades e desempenho; (2) estruturas de dados que permitem aos programas manipular informação adequadamente e (3) informação descritiva em ambas as formas, física e virtual, que descreve a operação e uso dos programas."* Grady Booch (2018), no fechamento do seu ensaio intitulado *The History of Software Engineering*, baseado na palestra que ministrou na conferência ICSE 2015 em Florença, oferece uma definição mais emocional: *"Software é a escrita invisível que sussurra as histórias de possibilidade ao nosso hardware. E vocês são os contadores de história."*

Por sua vez, o termo Engenharia de Software, tem as seguintes definições no *Software and Systems Engineering Vocabulary* (SEVOCAB), um projeto conjunto da ISO (*International Organization for Standardization*), IEC (*International Electrotechnical Commission*) e IEEE, que fornece definições de padrões internacionais para os termos de software e engenharia de sistemas: (1) aplicação sistemática de conhecimento científico e tecnológico, métodos e experiência ao projeto, implementação, teste e documentação de software. (2) aplicação de uma abordagem sistemática, disciplinada e quantificável ao desenvolvimento, operação e manutenção de software, ou seja, a aplicação de engenharia de software.

3. Referencial Teórico

3.1 Principais dificuldades em projetos de desenvolvimento de software

Várias características do software desafiam o seu gerenciamento, sendo sua complexidade uma delas [Brooks 1987, Reels 1999, Wirth 2008, Valente 2020]. Frederick Brooks, um dos pioneiros da área de Engenharia de Software, chama atenção para o fato de que o software é diferente de qualquer outro produto da engenharia, e identifica dificuldades particulares à área de Engenharia de Software, entre elas, a complexidade inerente ao software, que dificilmente será superada por qualquer nova tecnologia ou método que se invente [Brooks 1987]. O relatório The Cost of Poor Software Quality (CPSQ) 2020 avalia que o desenvolvimento de software pode ser a profissão mais intelectualmente desafiadora de todos os tempos, apontando fatores desafiadores como complexidade, aumento de

tamanho, mudança na natureza dos problemas, ataques cibernéticos, avanços tecnológicos rápidos, evolução das necessidades dos usuários e pressão para entregar valor com rapidez [CISQ 2020]. Reels (1999) endossa esse entendimento quando afirma que "*o problema básico da computação é o domínio da complexidade"*, e que o fato de os desenvolvedores tentarem atingir um alvo móvel, ou seja, os requisitos do sistema, torna a atividade ainda mais complexa. Para Brooks (1987), *"não existe um único desenvolvimento, em tecnologia ou técnica de gerenciamento, que por si mesma prometa um aprimoramento de sequer uma ordem de magnitude, dentro de uma década, em produtividade, em confiabilidade, em simplicidade."* Segundo o autor, muitos dos problemas clássicos no desenvolvimento de software derivam da sua complexidade, e aumentam com o seu tamanho. O autor também chama atenção para a dificuldade de obter uma especificação completa e consistente no desenvolvimento de software.

Em outro texto amplamente conhecido, *The Mythical Man Month* [Brooks 1978], Brooks atribui à dificuldade de atender ao cronograma os problemas enfrentados pelos projetos de software. Preocupado com a crença de que o atraso em um projeto pode ser superado com o acréscimo de pessoal, Brooks postulou a conhecida Lei de Brooks: "*Adicionar mão de obra a um projeto atrasado o torna ainda mais atrasado.*"

Versando sobre fatos frequentemente esquecidos sobre a Engenharia de Software, Glass (2001) avalia que para cada 10% de acréscimo na complexidade do problema, existe 100% de acréscimo na complexidade da solução. Ele observa ainda que as duas maiores causas de projetos fora de curso são requisitos instáveis e estimação otimista. Sobre estimativas, ele acrescenta que elas geralmente são realizadas no início do projeto, antes do levantamento de requisitos e entendimento apropriado do problema, e não são ajustadas no curso do projeto.

Em seu artigo sobre fatores críticos de sucesso em projetos de software, Reels (1999) apresenta sinais de que o projeto de software irá falhar, tais como falta de entendimento das necessidades dos usuários, escopo mal definido, gerenciamento de mudanças ineficiente, mudança nas necessidades do negócio, prazos não realistas, resistência de usuários, perda de patrocinador e desconsideração de melhores práticas e lições aprendidas.

Charette (2005) lista, entre os fatores mais comuns de fracasso de projetos, metas confusas ou não realistas, estimativas inexatas, requisitos de sistema mal definidos, acompanhamento insuficiente, riscos não gerenciados, comunicação pobre, inabilidade de lidar com a complexidade do projeto e políticas do patrocinador, e avalia que decisões equivocadas de gerenciamento são mais prejudiciais ao projeto do que questões técnicas.

3.2 Critérios e fatores de sucesso em projetos de software

Várias percepções sobre o que pode ser caracterizado como um projeto de software bem sucedido podem ser encontradas na literatura. A definição tradicional de sucesso de um projeto costuma ser aquela que leva em consideração apenas os aspectos do triângulo de restrições: escopo, custo e tempo [Van Wyngaard et al. 2012]. Desse modo, um projeto bem sucedido seria aquele realizado no tempo previsto, em que custo não ultrapassou o esperado, e que entregou o que havia sido combinado. Entretanto, ao longo dos anos, outras variáveis foram sendo introduzidas, tais como qualidade, satisfação do cliente, satisfação da equipe e benefícios alcançados.

A definição de sucesso de projeto também tem evoluído nos relatórios CHAOS, do Standish Group. Inicialmente, a definição se limitava ao triângulo de restrições. Em 2015, passaram a ser considerados os atributos *OnTime* (NoTempo), *OnBudget* (NoOrçamento), *OnTarget* (NaMeta), *OnGoal* (NoObjetivo), *Value* (Valor) e *Satisfaction* (Satisfação). Na versão de 2020, sucesso é definido como conformidade ao prazo, orçamento, escopo, mas também são considerados objetivo e valor

[Standish Group 2015, Standish Group 2020]. Dentre os achados reportados na sua versão mais recente, está a percepção de que projetos menores têm maior probabilidade de obterem sucesso, o mesmo ocorrendo com projetos que seguem a metodologia ágil em comparação ao modelo cascata. De acordo com o relatório, são três os fatores que mais contribuem para o sucesso de projetos de software: O Bom Lugar (*The Good Place*), O Bom Time (*The Good Team*) e O Bom Patrocinador (*The Good Sponsor*) [Johnson and Mulder 2020b, Portman 2021].

Como atributos de sucesso, Chow e Cao (2008) consideraram qualidade, escopo, prazo e custo no seu estudo sobre os principais fatores de sucesso em projetos de software com utilização de métodos ágeis. Como fatores de sucesso, os autores apontaram dentre os mais relevantes o comprometimento da alta gestão, o ambiente organizacional, o ambiente e a capacidade da equipe, e o envolvimento do cliente.

Entretanto, Glass (1999) sustenta que a percepção de sucesso pode variar a depender do público considerado. Ele cita o estudo realizado em um projeto avaliado como mal sucedido, que foi considerado pelos participantes o de maior sucesso em que haviam participado. O projeto foi considerado ter fracassado devido a ultrapassar o prazo, orçamento e tamanho previstos. Entretanto, fatores como satisfação com o trabalho em equipe, liberdade para realizar um bom design e menor pressão devido aos prazos influenciaram a percepção da equipe de que o projeto havia sido um sucesso.

Reels (1999) apresenta fatores essenciais para o gerenciamento bem sucedido de projetos de software, entre eles: adotar objetivos e expectativas realistas, alocar a equipe adequada e prover os recursos de que necessita, prezar pela qualidade, acompanhar o progresso, tomar decisões inteligentes e institucionalizar retrospectivas.

Em seu estudo comparativo, Nasir e Sahibuddin (2011) listaram requisitos e especificações claras, objetivos e metas claros, cronograma realista, habilidades efetivas de gerenciamento de projetos, suporte da alta administração, envolvimento do usuário, comunicação efetiva e feedback, recursos realísticos, equipe habilidosa e suficiente, como os dez fatores de sucesso mais frequentes na literatura levantada por eles.

Ahimbisibwe et al. (2015) contabilizaram os fatores de sucesso mais comumente encontrados na literatura, e os categorizaram em organizacionais, equipe e cliente. Foram identificados 37 fatores mais comumente citados nas publicações, observando que os mais frequentes tendem a ser aqueles relacionados ao alto gerenciamento, estratégia de tomada de decisões e cultura organizacional. Características da equipe tais como comprometimento, comunicação, composição e empoderamento também tiveram destaque nas publicações levantadas. Na categoria organizacional, os três fatores mais frequentemente citados foram apoio da alta gerência, cultura organizacional e nível de planejamento do projeto. Comprometimento, comunicação e empoderamento foram os fatores mais frequentes na categoria equipe. Para a categoria cliente, participação e suporte do usuário, treinamento e educação do cliente foram os mais encontrados na literatura pesquisada.

Em um trabalho mais recente, Garousi et al. (2018) relatam que estenderam os fatores de sucesso presentes em Ahimbisibwe et al. (2015), e conduziram uma investigação sobre os fatores mais importantes relacionados ao sucesso dos projetos de software. Foram considerados mais importantes aqueles fatores mais correlacionados com os critérios de sucesso tais como orçamento, cronograma, escopo, construção e dinâmica do time, satisfação do cliente, satisfação do time e satisfação da alta administração. De acordo com os resultados obtidos, os fatores mais relacionados a critérios de sucesso foram monitoramento e controle, nível de planejamento, experiência da equipe com a metodologia, capacidade de gerenciamento de mudanças e familiaridade com a tarefa. Os autores consideraram ainda que, de modo geral, fatores relacionados à organização e à equipe tiveram maior correlação com as variáveis que descreviam o sucesso do projeto do que aqueles nas catego-

rias cliente e projeto. Uma vez que os fatores monitoramento, controle e planejamento do projeto foram os mais importantes, os autores reforçaram o entendimento de outros trabalhos de que, apesar de importantes, as habilidades técnicas da equipe são menos determinantes ao sucesso do projeto do que seu gerenciamento apropriado [Glass 1999, Reels 1999, Charette 2005].

O relatório CPQS 2020 menciona estudo que aponta diferença substancial nas práticas de empresas de software de alto desempenho e de baixo desempenho, revelando uma diferença de desempenho de cinco a dez vezes entre os 10% de empresas do topo e os 10% de empresas da base, de organizações da amostra considerada. Com base nos critérios de custo, cronograma e qualidade, foram identificados entre os fatores chave para o bom desempenho dos projetos: processo de software bem definido e adaptável, excelentes métodos de estimação, disciplina de gerenciamento de projetos, excelência da equipe, visão de qualidade, foco na satisfação do cliente e cultura de qualidade.

Verner e Cerpa (2005) apresentam o resultado de uma pesquisa com participantes de projetos de software em que, por meio de regressão logística, os fatores observados para o sucesso do projeto foram qualidade dos requisitos, reconhecimento do esforço por parte do gerente de projetos, levantamento inicial adequado de requisitos com evolução durante o projeto, levando a entregas de software bem definido, além de habilidade de comunicação, relacionamento e conhecimento do problema por parte do gerente de projetos, bem como realização de retrospectivas. A inclusão tardia de pessoal para atender a cronogramas agressivos foi avaliada como tendo impacto negativo ao sucesso do projeto.

Em um levantamento extensivo da literatura, McLeod e MacDonell (2011) organizaram uma lista de critérios e fatores que afetam o resultado dos projetos de desenvolvimento de software. Alguns dos critérios observados na literatura pesquisada foram desistência, alcance dos objetivos, satisfação com o sistema, impacto organizacional, qualidade, adequação ao cronograma e orçamento e especificações obedecidas. Dentre os fatores de sucesso levantados, organizados em categorias, constam suporte da alta gerência, expectativas realistas, relacionamento da equipe, estando o suporte da alta gerência entre os mais frequentes na categoria Pessoas e Ação. Na categoria Processo de Desenvolvimento, entre os critérios encontrados estão planejamento do projeto, participação do usuário, requisitos bem definidos, treinamento do usuário, gerenciamento de mudanças, liderança, sendo os fatores relacionados a requisitos adequados, participação do usuário e planejamento do projeto os mais frequentes. Na dimensão institucional, cultura organizacional e cooperação foram fatores de sucesso apontados.

Em sua revisão de literatura, Aldahmash et al. (2017) contabilizaram os fatores de sucesso no desenvolvimento ágil de software, e constataram que entre os mais frequentes, relacionados aos aspectos gerenciais, estavam envolvimento do cliente, processo de gerenciamento do projeto, cultura organizacional, comunicação e suporte da alta gerência. Em relação ao gerenciamento do projeto, os autores mencionam a necessidade de um planejamento inicial, que possa orientar as estimativas e contribuir na seleção do processo adequado de gerenciamento.

A Tabela 1 organiza os fatores de sucesso mais encontrados, em ordem decrescente da frequência em que ocorreram na literatura pesquisada.

Tabela 1. Fatores de sucesso mais encontrados na literatura pesquisada

Fator	Trabalhos	Frequência
Equipe (comunicação, composição, empoderamento, capacidade, satisfação, comprometimento, experiência)	Standish Group (2020), Chow e Cao (2008), Glass (1999), Reels (1999), Nasir e Sahibuddin (2011), Ahimbisibwe et al. (2015), Garousi et al (2018), CISQ (2020), McLeod e MacDonel (2011), Aldahmash et al (2017)	10
Gerência do projeto (planejamento, cronograma, objetivos, recursos, acompanhamento/monitoramento, decisões, mudanças)	Glass (1999), Reels (1999), Nasir e Sahibuddin (2011), Ahimbisibwe et al. (2015), Garousi et al (2018), CISQ (2020), Verner e Cerpa (2005), McLeod e MacDonel (2011), Aldahmash et al (2017)	9
Apoio da alta gestão	Standish Group (2020), Chow e Cao (2008), Nasir e Sahibuddin (2011), Ahimbisibwe et al. (2015), McLeod e MacDonel (2011), Aldahmash et al (2017)	6
Envolvimento do cliente/ usuário	Chow e Cao (2008), Nasir e Sahibuddin (2011), Ahimbisibwe et al. (2015), CISQ (2020), McLeod e MacDonel (2011), Aldahmash et al (2017)	6
Ambiente/ cultura organizacional	Standish Group (2020), Chow e Cao (2008), Ahimbisibwe et al. (2015), McLeod e MacDonel (2011), Aldahmash et al (2017)	5
Requisitos bem definidos e de qualidade	Glass (1999), Nasir e Sahibuddin (2011), Verner e Cerpa (2005), McLeod e MacDonel (2011)	4
Retrospectivas	Reels (1999), Verner e Cerpa (2005)	2
Foco na Qualidade	Reels (1999), CISQ (2020)	2

4. Considerações finais

Devido à importância do tema, muitos são os estudos que tentam identificar os fatores determinantes para o sucesso dos projetos de software. Apesar de tantos esforços, Mohagheghi e Jørgensen (2017) ponderam, a partir da comparação de dois trabalhos com 45 anos de distância entre sua publicação, que projetos de software parecem falhar pelos mesmos motivos de antes.

Os critérios para avaliar essa questão evoluíram de tempo, escopo e custo, para incluir atributos como valor para o usuário, qualidade, satisfação da equipe e benefícios alcançados. Conhecer os fatores de sucesso para os projetos de software tem o potencial de aprimorar a qualidade dos softwares desenvolvidos, diminuir o percentual de projetos considerados mal sucedidos, com consequente redução do desperdício de recursos, e aumentar a satisfação da equipe e do cliente com os resultados obtidos. Na literatura pesquisada, alguns fatores se sobressaem para contribuir com esse objetivo, sendo que dentre os principais estão suporte da alta gerência, participação efetiva do cliente, requisitos bem definidos, realização de retrospectivas, bem como planejamento e acompanhamento adequados.

A partir dos trabalhos consultados, é possível constatar a atualidade do paradoxo de Cobb: "*Nós sabemos por que projetos fracassam, nós sabemos como prevenir seu fracasso – então por que eles ainda fracassam?*".

Em seu artigo, Verner e Cerpa (2015) comentaram, com base nos projetos analisados, que a gerência de negócios parecia não apreciar as práticas necessárias à execução bem sucedida dos projetos, uma vez que as estimativas de cronograma e custos dos gerentes de projeto não eram consideradas. Devido a isso, eles recomendaram que a gerência sênior fosse melhor esclarecida sobre a importância de "*requisitos adequados, estimativas de esforço e cronograma sistemáticas e importância de consultar os gerentes de projeto nas decisões sobre o projeto.*"

Embora o levantamento apresentado neste trabalho possa colaborar com o conhecimento de fatores importantes ao sucesso dos projetos de software, se faz necessária uma revisão mais sistemática dos trabalhos relevantes à área. Em trabalhos futuros, também seria de grande utilidade investigar cada fator de sucesso identificado, bem como formas de encorajar a sua adoção nos projetos executados. Desse modo, o conhecimento adquirido poderia ser empregado no aumento da frequência de projetos bem sucedidos, o que, diante do papel crítico desempenhado pelo software na atualidade, não é somente desejável, e sim mandatório.

Referências

Aldahmash, A.et al. (2017). A review on the critical success factors of agile software development. European conference on software process improvement (pp. 504-512). Springer International Publishing. doi: 10.1007/978-3-319-64218-5_41

Ahimbisibwe, A., Cavana, R. Y and Daellenbach, U. (2015). A contingency fit model of critical success factorsfor software development projects. Journal of Enterprise Information Management, 28(1), 7–33.

Berkun, Scott (2008) Making Things Happen Mastering Project Management. O'Reilly Media Inc.

Booch, Grady. The History of Software Engineering.

Brooks, Frederick P. (1987) No Silver Bullet Essence and Accidents of Software Engineering. Computer, vol. 20, no. 4, pp. 10-19, April 1987, doi: 10.1109/MC.1987.1663532.

Brooks, Frederick P. (1978) The Mythical Man-Month: Essays on Software Engineering. Addison -Wesley Longman Publishing Co., Inc.

Bourque, P and Fairley, R.E. *Guide to the Software Engineering Body of Knowledge, Version 3.0*, IEEE Computer Society, 2014.

Charette, R.N. (2005) Why Software Fails [Software Failure]. IEEE Spectrum, 42, 42-49. http://dx.doi.org/10.1109/MSPEC.2005.1502528

CISQ 2021. The Cost of Poor Software Quality in the US: A 2020 Report.

Chow, T and Cao, D. B. (2008) A survey study of critical success factors in agile software projects. The Journal of Systems and Software 81 (2008) 961–971

DOD-STD-2167 (1985). Defense System Software Development. Disponível em https://www.product-lifecycle-management.com/download/DOD-STD-2167A.pdf. Acesso em 20/11/2021.

Filho, Cléuzio Fonseca (2007). História da Computação. EDIPUCRS. Disponível em https://engsoftmoderna.info/. Acesso em 18/11/2021.

Garousi, Vahid et al. (2018) Correlation of critical success factors with success of software projects: an empirical investigation. Software Quality Journal. https://doi.org/10.1007/s11219-018-9419-5.

Glass, Robert L. (1999) Evolving a new theory of project success. Commun. ACM 42, 11 (Nov. 1999), 17–19. https://doi.org/10.1145/319382.319385

Glass, Robert L. (2001) Frequently forgotten fundamental facts about software engineering. *IEEE Software*, vol. 18, no. 3, pp. 112-111, May-June 2001, doi: 10.1109/MS.2001.922739.

History (2021). How Early Humans Survived the Ice Age. Disponível em https://www.history.com/news/ice-age-human-survival. Acesso em 12/12/2021

IEEE. (2021) Computer Society – CS. Disponível em https://site.ieee.org/sb-cefetrj/societies-chapters/capitulos/computer-society-cs/. Acesso em 27/11/2021.

IEEE. (2020) História. Disponível em https://r9.ieee.org/northeast-br/ieee/historia-do-ieee/. Acesso em 27/11/2021.

Johnson, Jim and Mulder, Hans. (2020a). Go with the Flow: Envisioning a Successful Pipeline of Software Projects. Conference paper.

Johnson, Jim and Mulder, Hans (2020b) Endless Modernization: How Infinite Flow Keeps Software Fresh. The Standish Group.

Hozac-Holland Mark (2011). The History of Project Management. Multi-Media Publications Inc. Disponível em https://historyofprojectmanagement.com/. Acesso em 29/10/2021.

Kakar, Ashish and Kakar, Akshay (2020) A Brief History of Software Development and Manufacturing. Proceedings of the Southern Association for Information Systems Conference (SAIS 2020)

McLeod, L. and MacDonell, S. G. (2011). Factors that affect software systems development project outcomes: A survey of research. ACM Computing Surveys (CSUR), 43(4), 1-56.

Mohagheghi, P and Jørgensen, M. (2017). What Contributes to the Success of IT Projects? An Empirical Study of IT Projects in the Norwegian Public Sector. J. Softw., 12(9), 751-758.

Nasir, M.H.N. and Sahibuddin, S. (2011) Critical success factors for software projects: A comparative study. Scientific Research and Essays. 2011, vol 6, pp 2174–2186

Reel, J. S (1999). Critical success factors in software projects. IEEE Software, vol. 16, no. 3, pp. 18-23, May-June 1999.

Richman, Larry (2011). Successful Project Management. Amacom.

PMI-SC. PMI no Brasil. Disponível em https://pmisc.org.br/sobre/o-que-e-pmi/pmi-no-brasil/. Acesso em 10/11/2021.

Portman, Henry. (2021) Review Standish Group – CHAOS 2020: Beyond Infinity. Disponível em https://hennyportman.wordpress.com/2021/01/06/review-standish-group-chaos-2020-beyond-infinity/. Acesso em 20/11/2021.

Pressman, Roger S and Maxin, Roger S. (2015) Software engineering: a practitioner's approach. 8ª edição.

Rico, David F. Short History of Software Methods. Disponível em http://ww.davidfrico.com/rico04e.pdf. Acesso em 28/11/2021

Royce, Winston W. (1970) Managing the Development of Large Software Systems: Concepts and Techniques. Technical Papers of Western Electronic Show and Convention (WesCon) August 25-28, 1970, Los Angeles, USA

ScienceFocus (2019) The invention of spears and bows and arrows may have helped early humans drive Neanderthals to extinction. Disponível em https://www.sciencefocus.com/news/the-invention-of-spears-and-bows-and-arrows-may-have-helped-early-humans-drive-neanderthals-to-extinction/. Acesso em 12/12/2021

ScientificAmerican (2012). How Alan Turing Invented the Computer Age. Disponível em https://blogs.scientificamerican.com/guest-blog/how-alan-turing-invented-the-computer-age/. Acesso em 28/10/2021.

Shenhar, A. and Dvir, D. (2004). Project management evolution: past history and future research directions. Paper presented at PMI® Research Conference: Innovations, London, England. Newtown Square, PA: Project Management Institute.

Sommerville, Ian (2011) Engenharia de Software. 9ª edição. Pearson Prentice Hall.

Spalek, S. (2005). Critical success factors in project management. To fail or not to fail, that is the question! Paper presented at PMI® Global Congress 2005—EMEA, Edinburgh, Scotland. Newtown Square, PA: Project Management Institute.

Standish Group. (2015). CHAOS Report 2015.

Standish Group. (2020). CHAOS Report 2020.

Verner, J. M. and Cerpa, N. Australian Software Development: What Software Project Management Practices Lead to Success? *2005 Australian Software Engineering Conference*, 2005, pp. 70-77, doi: 10.1109/ASWEC.2005.14

Taherdoost, Hamed and Keshavarzsalehc, Abolfazl (2016) Critical Factors that Lead to Projects' Success/Failure in Global Marketplace. Procedia Technology, Volume 22, 2016, https://doi.org/10.1016/j.protcy.2016.01.151. Disponível em (https://www.sciencedirect.com/science/article/pii/S2212017316001523). Acesso em 29/10/2021.

The Economist (2000). How software got its name. The Economist Group Limited,

Valente, Marco Tulio (2020). Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade, 2020. Disponível em https://engsoftmoderna.info/. Acesso em 22/11/2021.

Van Wyngaard, C. J., Pretorius, J. H. C. and Pretorius, L. (2012) Theory of the triple constraint - A conceptual review. *2012 IEEE International Conference on Industrial Engineering and Engineering Management, 2012, pp. 1991-1997, doi: 10.1109/IEEM.2012.6838095.*

Watts, Adrienne (2014). Project Management. BCcampus. Disponível em https://opentextbc.ca/projectmanagement/. Acesso em 28/10/2021.

Wazlawick, Raul Sidnei and Silva Junior, Deógenes Pereira. (2021). Histórico de Eventos da SBC no Brasil. SBC Horizontes, março. 2021. ISSN 2175-9235. Disponível em: http://horizontes.sbc.org.br/?p=5493. Acesso em 27/11/2021.

Wirth, Niklaus (2008). A Brief History of Software Engineering. IEEE Annals of the History of Computing. IEEE Computer Society.

Pedro Augusto Vitor de Santana
Instituto de Computação – Universidade Federal da Bahia (UFBA)
Salvador – BA – Brasil
pedroavs@ufba.br

Ferramentas e abordagens que auxiliam na extração informações de documentos

Abstract. *This article describes tools and approaches that help extract information from documents, with the aim of increasing accuracy, reducing human labor in repetitive processes, and increasing execution speed. For this, computer vision techniques such as image processing and optical character recognition are used*

Resumo. *Este artigo descreve ferramentas e abordagens que auxiliam na extração de informações de documentos, com o objetivo de aumentar a precisão, reduzir o trabalho humano em processos repetitivos e aumentar a velocidade de execução. Para isso são utilizadas técnicas de visão computacional como tratamento de imagens e reconhecimento ótico de caracteres (OCR).*

1. Introdução

A análise de conteúdo de documento está presente em grande parte dos processos burocráticos, analisar se determinada imagem anexada se adequa ao que foi solicitado, se o conteúdo respeita as regras previamente definidas, se a validade do documento encontra-se em um período aceitável, além de inúmeras validações que se fazem necessários na análise do documento.

A otimização do processo de análise de documentos é fundamental para que qualquer instituição possa completar processos de forma segura e rápida. Com um processo bem estruturado, o negócio consegue reduzir riscos, evitar erros e se posicionar de forma mais eficaz no mercado. Além disso, a instituição criará uma melhor experiência de uso dos seus serviços tanto pelo cidadão que utiliza-se do mesmo, quanto para o colaborador que pode evitar atividades repetitivas e desestimulantes, tornando-se muito mais eficiente e aumentando a qualidade de vida de todos os envolvidos no processo.

Em grande parte esse processo é executado por um ser humano, sabendo disso propomos a intervenção deste processo por meio de computação visual com o objetivo de automatizar a verificação desses documentos extraindo as informações e convertendo-os em texto de forma a permitir a validação do conteúdo, analisando o contexto e permitindo que outras aplicações utilizem uma interface de comunicação fazendo o envio do conteúdo na forma de imagem e recebendo as informações de forma estruturadas, reduzindo ao máximo a intervenção humana nesse processo, aumentando a eficiência do processo, reduzindo o tempo necessário para a conclusão e impedindo a ocorrência de gargalos em períodos em que há uma maior demanda deste tipo de serviço.

Utilizando várias técnicas de visão computacional foi possível desenvolver uma interface de programação de aplicações(API) que possibilita outras aplicações envie imagens e receba as infor-

mações ali contidas, de forma estruturada, sendo essa comunicação feita utilizando protocolos de comunicação como Protocolo de Transferência de Hipertexto(HTTP) e seus parâmetros de entrada e saídas previamente bem definidos e disponibilizados por meio de documentação.

2. Referencial Teórico

Sabendo que a maioria dos documentos seguem uma estrutura fixa, inicialmente foi buscada uma forma de conversão das imagens em texto, chegando assim à decisão de utilização de OCR. OCR é uma tecnologia que permite o reconhecimento de material escrito a partir de diversos formatos de imagens[1]. Dentro desse contexto existem muitas variáveis que podem influenciar na precisão da conversão, a cor do fundo da imagem, a disposição das informações e a qualidade da imagem. O processo de conversão de imagem texto descrito acima consiste nos seguintes passos:

Obtenção da imagem fonte externa, como scanner ou câmera.

O pré-processamento consiste em remoção de ruído, limiar e linha de base de extração de imagem e alguns outros tratamentos que podem melhorar a precisão da ferramenta.

Segmentação de caracteres, entre as técnicas mais simples estão a análise de componentes conectados e podem ser usados perfis de projeção. No entanto, em situações complexas, onde os personagens estão sobrepostos, quebrados ou algum ruído presente na imagem. Entre as técnicas mais avançadas de segmentação de caracteres são usadas a extração de recursos que tem como objetivo minimizar as variações dentro das classes e maximizar as variações entre as classes.

Classificação de caracteres mapeia os recursos da imagem segmentada para diferentes categorias ou classes. As técnicas de classificação estrutural são baseadas em características extraídas da estrutura da imagem e usam diferentes regras de decisão para classificar as imagens. Os métodos de classificação de padrões estatísticos são baseados em modelos probabilísticos e outros métodos estatísticos para classificar os caracteres.

Pós-processamento pode ser realizado para melhorar a precisão dos sistemas OCR. Essas técnicas utilizam processamento de linguagem natural, contexto geométrico e linguístico para corrigir erros nos resultados de OCR.

Dentro desse contexto foi escolhido o Tesseract OCR como ferramenta de conversão das imagens. Tesseract é um mecanismo de OCR de código aberto que foi desenvolvido na Hewlett-Packard entre 1984 e 1994 [2], se destacou com seus resultados e logo em seguida teve sua pesquisa levada em sigilo. O Tesseract foi iniciado como um projeto de pesquisa de doutorado no HP Labs, Bristol, e ganhou notoriedade como um possível ferramenta que seria acoplada para a linha de scanners de mesa da HP. O objetivo inicial era aperfeiçoar bem mais a rejeição do que na aceitação dos caracteres. Em 1995 a aplicação foi enviada a um evento anual que acontecia na Universidade de Nevada (UNLV), em Las Vegas, esse evento acontecia todos os anos e o Tesseract mostrou-se bem mais eficiente que outras aplicações que tinham o mesmo objetivo. No final de 2005, a HP lançou o Tesseract para código aberto e isso permite sua utilização no projeto.

No campo da recuperação de informações, a busca e substituição de palavras-chave é um problema padrão. Frequentemente, queremos encontrar palavras-chave específicas no texto ou substituir palavras-chave por nomes padronizados. Para resolver esses problemas, as expressões regulares (Regex) são as mais comumente usadas[7]. O Regex como ferramenta é muito versátil e útil para correspondência de padrões. Podemos pesquisar padrões como '\d{2}[.]\d{2}' (que corresponderá a qualquer número de 2 dígitos seguido de um ponto e mais 2 dígitos) ou palavras-chave como 'Data de nascimento' em texto. A expressão regular é bastante útil na hora de formatar texto, eliminando caracteres indesejados ou buscando padrões dentro do resultado obtido pelo OCR.

Quando falamos em imagens obtidas por um dispositivo como uma câmera fotográfica, podemos afirmar que é uma representação de objetos tridimensionais em um plano, variando o ângulo do

centro da câmera é possível obter variações dos mesmos objetos. Homografia é a transformação que relaciona duas imagens que passam por uma rotação em torno do centro da câmera[3].

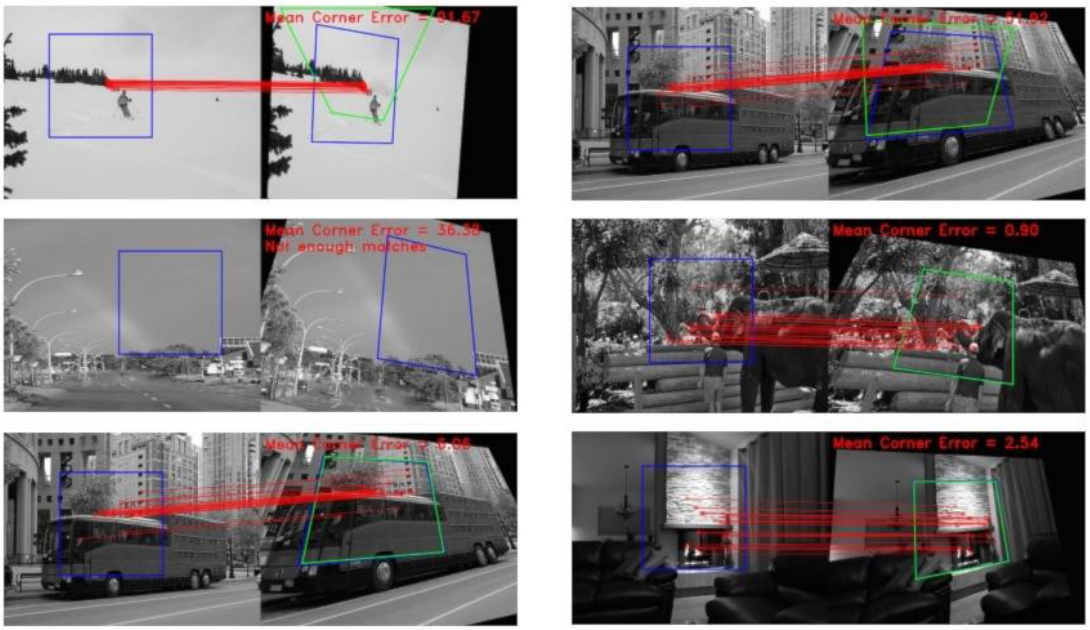


Figura 1. Traditional Homography Estimation, em cada um dos 6 exemplos, o azul representa a região da verdade fundamental, mostrando a saída da Estimativa de Homografia baseada em ORB, os recursos correspondentes em vermelho e o mapeamento resultante em verde.

Não é surpreendente que homografias sejam essenciais para lidar com documentos que serão capturados com uma variedade imensa de angulações, faz-se necessário a conversão dessas imagens de forma que seja possível a sua tradução para um modelo comum permitindo a demarcação de áreas de interesses. Na visão computacional, a matriz de transformação usada para descrever a relação de mapeamento de posição entre o sistema de coordenadas mundial e o sistema de coordenadas de pixel é chamada de matriz de homografia. Ela desempenha um papel importante no campo de estimativa de posição de câmera. Usando uma abordagem mais direta, considera-se pelo menos 4 pares de pontos correspondentes para calcular a matriz de homografia, sendo esse um cenário extremamente simples, em um cenário real para tornar o cálculo mais preciso, geralmente é usado muito mais do que quatro pares de pontos no cálculo da matriz de homografia. A precisão e estabilidade do programa de estimativa serão melhoradas quando a cena for assumida como composta por um plano. Um método de obtenção do movimento da câmera por decomposição de deformação simples comum pode ser encontrado em [4]. Em [5], Fauger as e Lustman têm uma descrição mais específica da decomposição de deformação única quando a câmera se move no plano, que é o cenário que ocorrerá nos casos que iremos abordar na extração das informações dos documentos.

A correspondência de modelos é a classificação de amostras desconhecidas comparando-as a protótipos ou modelos conhecidos[8]. Consideramos as aplicações em processamento de imagem em que o modelo e as amostras são imagens digitalizadas ou fotografias. Existem dois cenários comuns:

Uma pesquisa de ocorrências de um único modelo em uma imagem que chamaremos de Pesquisa

Classificação de uma amostra extraída de uma imagem como um de vários protótipos de modelo que chamaremos de Classificação.

Em nosso problema abordamos apenas o primeiro caso, onde definimos o modelo como sendo um cabeçalho e um segundo modelo como rodapé, a partir da correspondência desses modelos na imagem, identificamos as coordenadas cartesianas de cada um e com isso é possível definir a região da área de interesse em documentos que possuem seu conteúdo em disposição variável.

Após definir as técnicas de processamento de imagem e texto, foi necessário a decisão de como disponibilizar a aplicação na forma de serviço. REST (Representational State Transfer) é o estilo arquitetônico que tem impulsionado o desenvolvimento da web moderno e escalonável recentemente[9]. Com isso partimos para a implementação do projeto utilizando o framework Django.

3. Proposta

A proposta inicialmente era a extração de informações de documentos utilizados no processo de primeiro título de eleitor(comprovantes de residência, documentos de identificação pessoal, documento de quitação eleitoral, documento de quitação militar e etc), durante o processo de estudo foi identificado que esse processo necessitava de intervenções que fugiam do escopo do projeto, sendo assim foi tomada a decisão de alterar o foco para outro objetivo. Com isso, foi necessário buscar informações sobre técnicas que auxiliassem na extração de informações de documentos diferentes do escopo inicial. Antes da alteração de escopo já havia sido concluída a extração de informações de um comprovante de residência, para isso utilizamos algumas técnicas de tratamento de imagens que melhoraram a precisão da conversão.

4. Aplicação Prática

O escopo do projeto foi alterado para possibilitar a utilização do projeto e documentos que faziam parte do processo de auditoria de suprimento de fundos, segue algum dos documentos levantados:

- Extrato Bancário
- Nota Fiscal
- Cupom Fiscal

Cada documento da lista apresentou alguma peculiaridade, o extrato bancário dispõe de quantidade de informações variadas, alterando seu desenho, a nota fiscal dispõe de uma chave de acesso que possibilita sua análise no portal do ministério da fazenda, porém existe validação anti-robô, o cupom fiscal dispõe de um Qrcode que permite a consulta, mas sem nenhuma verificação anti-robô, possibilitando a coleta dessas informações não por OCR, mas por requisição à um serviço terceiro, sendo assim foi necessário a implementação de várias técnicas, não só de visão computacional, que permitisse uma acurácia e eficiência na extração das imagens.

4.1. OCR

A utilização do OCR no projeto é fundamental, em apenas um dos documentos não foi necessário a utilização da ferramenta, sendo de extrema necessidade. Existiram algumas dificuldades na utilização do OCR direto no documento, como a demora no processamento, a conversão de informações que não era interessante naquele contexto, a falta de informações por vários fatores e também a falta de estruturação das informações. Com isso foi necessário a utilização de outras técnicas com o objetivo de reduzir o tempo de processamento e eliminação de informações desnecessárias, além do aumento na precisão do OCR.

4.2. Expressões Regulares

A aparição de palavras descritivas mostrou-se frequente ao utilizar-se OCR em documentos, como por exemplo: *"Nome completo: "* seguido do nome completo do portador do documento, apesar disso ajudar bastante na estruturação das informações, é interessante que essas palavras sejam removidas nos últimos passos do processamento.

As expressões regulares são representações que facilitam a busca por padrões no texto e isso torna a manipulação dos textos obtidos pela OCR mais ágil e robusta, facilitando a identificação e remoção de caracteres que não agregam utilidade extremamente facilitadas dentro dos textos convertidos.

4.3. OpenCV

OpenCV (Open Source Computer Vision Library) é uma biblioteca de software de visão computacional multiplataforma e de código aberto. O OpenCV foi construído para fornecer uma infraestrutura comum para aplicativos de visão computacional com o objetivo da sua utilização em diversos contextos, por isso além de ter ser um produto licenciado por BSD, o OpenCV consegue ser utilizado em diversos ambientes por ser uma biblioteca mantém suporte a programadores que utilizam Java, Python e Visual Basic e desejam incorporar a biblioteca a seus aplicativos. A biblioteca foi desenvolvida nas linguagens de programação C/C++ sua versão 1.0 foi lançada no final de 2006. A utilização do OpenCV dentro do projeto tem como objetivo a implementação de técnicas de segmentação de imagens para definição de áreas de interesse, a redução de ruídos e aplicação de filtros com o objetivo do aumento da acurácia do OCR, a utilização do método de homografia que tem utilização no redimensionamento e ajuste das imagens, trazendo para padrões já conhecidos e também o reconhecimento de padrão implementado pelo método de correspondência de modelo, que tem como objetivo a extração de informações em documentos que não há um arranjo físico dos elementos fixo, havendo variação na posição dos dados.

4.4. Homografia

No campo da visão computacional, quaisquer duas imagens da mesma superfície plana no espaço são relacionadas por uma homografia (assumindo um modelo de câmera pinhole). Isso tem muitas aplicações práticas, como retificação de imagem, registro de imagem ou cálculo do movimento da câmera entre duas imagens. Uma vez que a rotação e translação da câmera tenham sido extraídas de uma matriz de homografia estimada, essas informações podem ser usadas para navegação ou para inserir modelos de objetos 3D em uma imagem ou vídeo, de modo que sejam renderizados com a perspectiva correta e pareçam ter feito parte da cena original. No projeto, essa técnica tem o objetivo de paralelizar as linhas de conteúdo textual em determinados documentos, possibilitando a utilização do OCR em algumas imagens e melhorando o desempenho do mesmo em outros. Tendo com principal abordagem na maioria dos documentos a definição de áreas de interesse com o objetivo de reduzir o processamento computacional, além de facilitar a estruturação dos dados depois da extração, fez-se necessário o mapeamento desses documentos de forma a conseguir a localização de cada segmento a partir de coordenadas cartesianas, considerando que as imagens têm origem em sua maioria de câmeras e scanners é obrigatório que essas imagens sejam rotacionadas, transladadas e redimensionadas para um tamanho padrão previamente conhecido.

Tudo isso é possível com a homografia, desde que haja um modelo padrão para ser usado como referência, nesse projeto foi obtida uma imagem de boa qualidade dos documentos que utilizam homografia, utilizado editores de imagens que possibilitaram a remoção de dados que variáveis, mantendo apenas o desenho padrão do documento

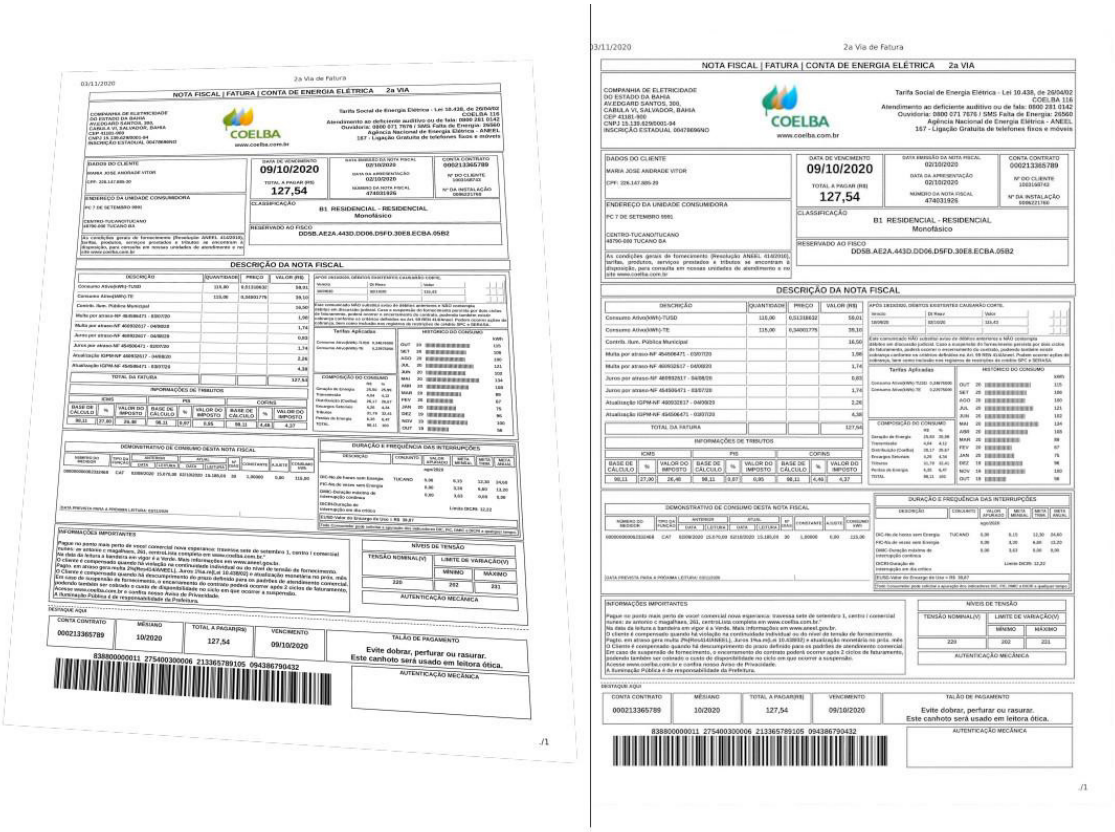


Figura 2. A imagem ao lado esquerdo é um exemplo de documento deformado, que não está adequada a imagem processada, à sua esquerda têm-se a mesma imagem após a aplicação de homografia utilizando um desenho padrão previamente configurado.

4.5. Correspondência de modelo

A correspondência de modelo é um método para pesquisar e encontrar a localização de uma imagem de modelo em uma imagem maior. Ele simplesmente desliza a imagem do modelo sobre a imagem de entrada (como na convolução 2D) e compara o modelo e o patch da imagem de entrada sob a imagem do modelo. Vários métodos de comparação são implementados no OpenCV. Ele retorna uma imagem em tons de cinza, onde cada pixel denota o quanto a vizinhança desse pixel corresponde ao modelo.

Se a imagem de entrada for do tamanho (LxA) e a imagem do modelo for do tamanho (largura x altura), a imagem de saída terá um tamanho de (L-w + 1, H-h + 1). Considere-o como o canto superior esquerdo do retângulo e considere (w, h) a largura e a altura do retângulo. Esse retângulo é a sua região de modelo.

Houve a necessidade de implementação desse método em documentos em que a área que contém informações tem tamanho variado. Um exemplo é o extrato bancário que é composto por movimentações financeiras, onde há a necessidade de extração dessas informações, como a quantidade de movimentações não é preestabelecida, a definição da área de interesse respectiva as movimentações necessita de uma abordagem que identifique de alguma forma onde essa área se inicia, e onde finaliza.

Ao estudar a composição do documento, foi identificado o que pode ser chamado de cabeçalho do documento, logo abaixo do cabeçalho seguem linhas paralelas de tamanho fixo, cada linha contém uma movimentação financeira, após um número desconhecido de linhas segue o que podemos chamar de rodapé, com informações fixas. Utilizando a técnica de correspondência de modelo utilizando o cabeçalho como padrão a ser buscado na imagem obtém-se as coordenadas cartesianas do início das movimentações e seguindo a mesma lógica obtém-se o final das movimentações utilizando o rodapé como padrão.

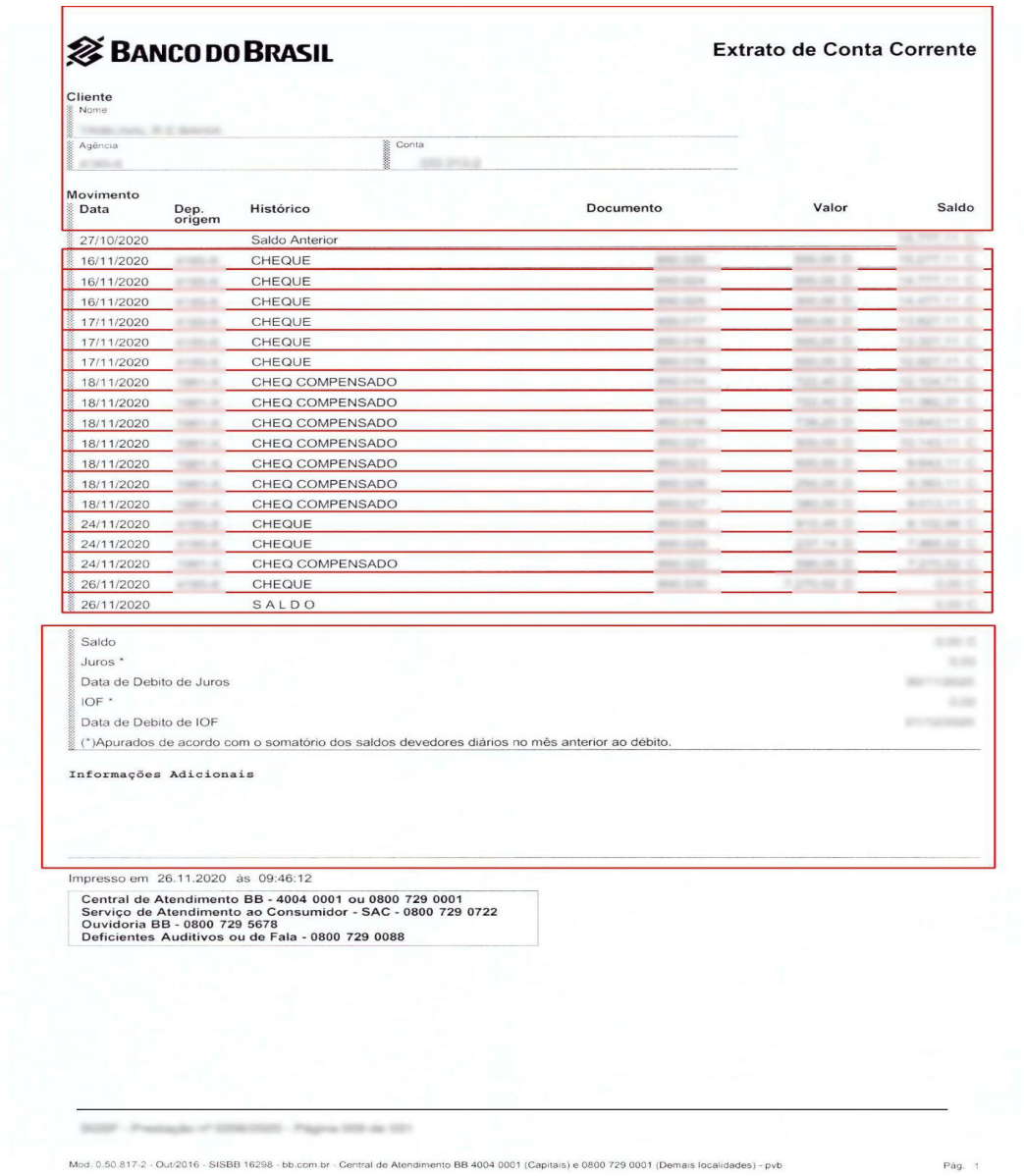


Figura 3. Exemplo da utilização da correspondência de modelo, sendo o primeiro retângulo o cabeçalho, o meio o conteúdo das movimentações e o último retângulo o rodapé.

Calculando o início da área, o tamanho de cada linha captura-se cada movimentação de forma isolada.

4.6. Django

Havendo a necessidade de se colocar todas técnicas abordadas para que possa ser utilizada por outras aplicações que venham a consumir essa ferramenta, uma opção viável é transformar toda a aplicação na forma de API restful, possibilitando assim a comunicação entre as aplicações através de um conjunto de definições e protocolos usado no desenvolvimento e na integração de aplicações. Descrevendo alguns padrões, como contrato entre o serviço usuário de informações, ou seja a aplicação que enviará os documentos na forma de imagem, estabelecendo a chamada e a resposta como conteúdo exigido pelo produtor, faz-se a comunicação. Para isso acontecer utiliza-se um framework que nos fornece a estrutura necessária para fazer a comunicação atrás do protocolo HTTP, considerando as restrições, foi escolhido o framework Django, que é um framework para aplicações web gratuito e de código aberto, escrito em Python. A utilização da linguagem Python foi feita considerando o amplo ecossistema em volta da linguagem quando se tem a necessidade de

utilizar visão computacional. Frameworks tem um intuito de evitar que haja retrabalho e para ajudar a aliviar parte do trabalho extra quando você está construindo algo.

Quando uma requisição chega a um servidor web, ela é passada para o Django, que tenta entender o que está sendo requisitado de fato. Primeiro, ele pega um endereço de página web e tenta descobrir o que fazer. Isso parte é feito pelo urlresolver do Django que é o componente do framework que lida com as chamadas a partir da URL. A partir de uma lista de padrões e tenta-se achar qual deles a URL se encaixa. O Django checa os padrões de cima para baixo, e se algum se encaixa, ele passa a requisição para a função associada com isso temos a divisão de cada documento abordado por URL e consequentemente cada URL acionará o método responsável por extrair as informações do arquivo desejado.

5. Conclusão

Embora tenha havido uma mudança de escopo no meio do projeto, foi-se mantido o objetivo principal e conseguimos utilizar várias técnicas de extração de informações das imagens. Tomamos a decisão de utilizar uma gama de técnicas que permitisse à qualquer um que venha a manter o projeto estender sua utilização de forma fácil, reutilizando essas mesmas técnicas e reaproveitando o conhecimento que nos foi adquirido durante esse processo, garantindo a continuidade do projeto por quem venha a mantê-lo, após o fim desse percurso. Como inicialmente propomos um serviço que extraísse informações de imagens e que pudesse ser consumido por outras aplicações, consideramos esse trabalho feito.

Referências

[1] ARCHANA A. SHINDE, D. 2012.Text Pre-processing and Text Segmentation for OCR. International Journal of Computer Science Engineering and Technology, pp. 810- 812.

[2] SMITH, R. 2007. An Overview of the Tesseract OCR Engine. In proceedings of Document analysis and Recognition.. ICDAR 2007. IEEE Ninth International Conference.

[3] DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich. “Deep image homography estimation.” arXiv preprint arXiv:1606.03798 (2016).

[4] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. Int. Journal of Pattern Recognition and Artificial Intelligence, 2(3):485–508, 1988.

[5] Y. Ma, S. Soatto, J. Kosecka, and S.S. Sastry. An Invitation to 3-D Vision: From Images to Geometric Models. SpringerVerlag, 2003.

[6] Patel, Chirag, Atul Patel, and Dharmendra Patel. “Optical character recognition by open source OCR tool tesseract: A case study.” International Journal of Computer Applications 55.10 (2012): 50-56.

[7] Singh, Vikash. “Replace or retrieve keywords in documents at scale.” arXiv preprint arXiv:1711.00046 (2017).

[8] Cox, Greg S. “Template matching and measures of match in image processing.” University of Cape Town, South Africa (1995).

[9] Hillar, Gaston C. Django RESTful Web Services: The Easiest Way to Build Python RESTful APIs and Web Services with Django. Packt Publishing Ltd, 2018.

[10] Dauzon, Samuel, Aidas Bendoraitis, and Arun Ravindran. Django: Web Development with Python. Packt Publishing Ltd, 2016.

Rita de Cassia Ulm da Silva Guedes[1]
Departamento de Ciência da Computação - Instituto de Matemática e Estatística – Universidade Federal da Bahia (UFBA)
Salvador – BA – Brazil
rita.ulm@ufba.br

1Residência em Tecnologia da Informação (TRE-BA)

Sistema Web/Mobile para Manutenção de Urnas Eletrônicas

Resumo. *Este artigo tem como objetivo apresentar uma solução informatizada web/mobile utilizando a plataforma Ionic e o banco de dados Oracle, para o gerenciamento e controle das manutenções preventivas e corretivas das urnas eletrônicas do Tribunal Regional Eleitoral da Bahia (TRE-BA). O sistema deverá possuir funcionalidades que sejam capazes de agilizar de forma segura e eficiente, o gerenciamento e controle das Urnas, Depósitos, Técnicos, Ordens de Serviço, Atividades de Manutenção e Peças.*

1. Introdução

O Tribunal Regional Eleitoral da Bahia (TRE-BA) possui atualmente um parque com mais de 40.000 urnas eletrônicas, localizadas em 18 locais em todo o Estado da Bahia, tais quais a Central de Apoio Técnico (CAT) em Salvador e os Depósitos de Urnas (Polos de Informática) que estão distribuídos pelos municípios de Cruz das Almas, Alagoinhas, Jacobina, Camaçari, Feira de Santana, Ribeira do Pombal, Ipirá, Irecê, Seabra, Barreiras, Brumado, Guanambi, Eunápolis, Ilhéus, Jequié, Vitória da Conquista e Juazeiro e mais cinco depósitos temporários em Bom Jesus da Lapa, Conceição do Coité, Paulo Afonso, Itapetinga e Teixeira de Freitas, além de 200 zonas eleitorais espalhadas pelo estado.

A atividade de conservação das urnas eletrônicas é de fundamental importância e tem por finalidade garantir o funcionamento das mesmas. Realizada de forma periódica, consiste em verificar as condições de desempenho, executar reparos e, caso necessário, repor peças. É realizada em média por dia, a manutenção de 400 urnas em Salvador e 30 no interior do estado. É imprescindível que as urnas estejam disponíveis, a qualquer tempo, e em perfeitas condições de uso para a realização de eleições, que é a atividade fim do Tribunal Regional Eleitoral da Bahia. Para a execução dessa atividade é necessário fazer o gerenciamento e controle das manutenções preventivas e corretivas dessas urnas eletrônicas. Atualmente esse procedimento é feito de forma manual, o que demanda muito trabalho, tempo, além de estar sujeito a erros.

As eleições ordinárias podem ser, por exemplo, eleições de vários tipos: eleições gerais (presidente, governador, deputado federal, deputado estadual e senador), eleições municipais (prefeito e vereador), eleições suplementares (eleições que ocorrem em razão de indeferimento do registro, cassação do diploma ou perda do mandato de candidato eleito em pleito majoritário por decisão da Justiça Eleitoral) e consultas populares (plebiscitos e referendos). As eleições extraordinárias

podem ser: entidades de classes (ex: OAB) ou comunitárias (ex: eleição do conselho tutelar). Além das eleições, as urnas eletrônicas podem ser utilizadas para treinamentos realizados com mesários, técnicos de urnas, além de eleitores, em locais de fácil acesso e de grande circulação de cidadãos [TSE, 2021].

Os procedimentos para a conservação das urnas buscam minimizar o processo de degradação e manter sua vida útil de no mínimo 10 (dez) anos, no caso das urnas eletrônicas, e de aproximadamente 5 (cinco) anos, no caso das baterias. As atividades de conservação das urnas e dos demais componentes internos ou externos são realizados com periodicidade quadrimestral e encontram amparo nas recomendações contidas no Relatório de Estudos de Metodologia de Conservação das Urnas Eletrônicas, elaborado a partir de ensaios, pesquisas e testes realizados nas urnas pela Fundação de Apoio à Capacitação em Tecnologia da Informação (FACTI), com anuência do Centro de Tecnologia da Informação Renato Archer, por força do Contrato TSE nº 126/2008, no qual foram indicados os procedimentos essenciais para a manutenção preventiva e o prazo em que serão realizados, de modo a assegurar a funcionalidade e disponibilidade das urnas e preservar seu tempo de vida útil.

1.1. Contexto

Atualmente o gerenciamento e controle das manutenções preventivas e corretivas das urnas eletrônicas é feito de forma manual pelos gerentes de polos e técnicos de urnas na Central de Apoio Técnico (CAT) e nos polos. Esse processo além de ser lento, está sujeito a erros e retrabalhos. Nos períodos eleitorais e "runin" (ciclo de manutenção preventiva das urnas eletrônicas), principalmente, a quantidade de manutenções aumenta consideravelmente impactando ainda mais o volume de trabalho [TRE-BA, 2021].

O processo manual é feito da seguinte forma, conforme ilustrado na Figura 1:

1. Ordem de serviço (O.S.) é aberta pelo gerente de polo para manutenção preventiva ou corretiva das urnas eletrônicas (uma ordem de serviço pode se referir a uma ou mais urnas). Na O.S. constam: os depósitos, as urnas, os técnicos, as atividades de manutenção e as peças (caso haja);

2. As urnas eletrônicas são manutenidas pelos técnicos de urnas;

3. Ordem de serviço é fechada pelo gerente de polo após a manutenção das urnas eletrônicas;

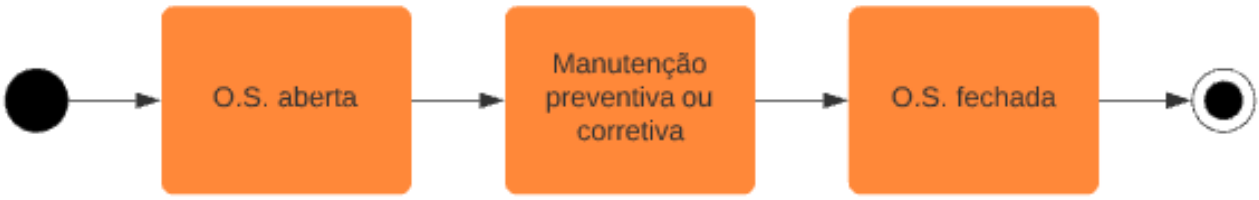


Figura 1: Passos da manutenção manual das urnas eletrônicas

1.2. Motivação

A autora deste trabalho é lotada na Seção de Desenvolvimento e Soluções Corporativas (SEDESC) e também é membro da Comissão Permanente de Totalização do TRE-BA (TOT), estando com isso envolvida em todas as eleições geridas pelo TRE-BA, sejam gerais, municipais ou suplementares.

As manutenções das urnas eletrônicas são de extrema importância para o desenvolvimento dos trabalhos eleitorais, pois minimizam a ocorrência de problemas técnicos nos equipamentos nas eleições, o que impacta diretamente no andamento dos procedimentos necessários para a totalização dos resultados das eleições.

O gerenciamento e controle manual de manutenção de urnas causam problemas para todos os envolvidos, por exemplo, quando ocorrem erros ou atrasos na abertura de ordens de serviço, consequentemente atrasa a manutenção da urna, além disso, geram limitações para os gestores, pois não existem, por exemplo, relatórios com o volume e o custo dessas manutenções em tempo real .

Além disso, quanto mais automatizamos os processos manuais efetuados pelo Tribunal Regional Eleitoral da Bahia, mais facilmente atingimos os objetivos elencados no Planejamento Estratégico Institucional (PEI) 2021-2026 que segue as diretrizes estabelecidas no Conselho Nacional de Justiça (CNJ) na Resolução nº 325/2020 [TRE-BA, 2021].

1.3. Objetivo

O objetivo desse artigo é ser um estudo para viabilizar o desenvolvimento de um aplicativo web/mobile para o gerenciamento e controle das manutenções preventivas e corretivas das urnas eletrônicas do Tribunal Regional Eleitoral da Bahia (TRE-BA) para a Seção de Urna Eletrônica (SEUEL), visando automatizar esse procedimento de forma rápida, segura e eficiente.

A aplicação ainda não tem um prazo previsto para ser desenvolvida, já que antes disso tem que haver uma solicitação formal da Seção de Urna Eletrônica (SEUEL) através do OTRS (Sistema de Gerenciamento de Serviços do TRE-BA), em seguida ser aprovada pelo Comitê Gestor da Informática e por fim entrar na fila de sistemas a serem desenvolvidos de acordo com a prioridade definida por esse comitê.

1.4. Organização do Restante do Texto

O restante do texto está organizado da seguinte forma. A Seção 2 descreve a metodologia usada para levantar e documentar os requisitos do sistema proposto. A Seção 3 apresenta o referencial teórico sobre os artefatos usados na documentação dos requisitos e modelagem do sistema, bem como sobre a manutenção de urnas eletrônicas. A Seção 4 descreve os requisitos e modelagem do sistema proposto. A Seção 5 apresenta a conclusão do artigo.

2. Metodologia

A metodologia utilizada para o projeto de desenvolvimento da aplicação para manutenções preventivas e corretivas de urnas eletrônicas foram entrevistas feitas com o chefe da Seção de Urnas Eletrônicas (SEUEL), para entendimento das regras do negócio.

A partir dessas entrevistas os requisitos do sistema proposto foram levantados e documentados através de um documento de requisitos de software e os protótipos das telas principais foram feitos, e em seguida foram enviados ao chefe da SEUEL, que após algumas alterações, deu aceite na versão final.

Após a validação da versão final, foi feita a modelagem de dados do sistema proposto através de diagramas, sendo eles o Modelo de Entidade Relacionamento (MER) (Figura 2) e a Modelo de Banco de Dados (Figura 3), que foram validados pela Seção de Banco de Dados (SEBDA).

3. Referencial Teórico

3.1. Manutenção de Urna Eletrônica

O site do Tribunal Regional Eleitoral de São Paulo define urna eletrônica como *"um microcomputador projetado pelo Tribunal Superior Eleitoral (TSE) para uso exclusivo nas eleições, que há 25 anos, tem propiciado o voto rápido, acessível e seguro. A máquina é totalmente isolada, não existindo nenhum componente, seja placa de rede ou bluetooth, que permita entrada e saída de dados via internet nem acesso remoto"* [TRE-SP, 2021].

A manutenção preventiva de urnas eletrônicas ou "run in" é um procedimento realizado quadrimestralmente em Salvador e nos 17 polos de urnas espalhados pelo estado pelos técnicos de urnas, cuja intenção é prevenir 80% de possíveis falhas com os componentes das urnas, evitando gastos com manutenção corretiva durante as eleições partidárias, no qual as máquinas são ligadas às tomadas elétricas para que as baterias sejam carregadas, um software (STE) é acionado com a finalidade de exercitar os componentes eletrônicos da urna e também é feita a limpeza e realizados testes de hardware e software nos equipamentos [TRE-BA, 2021].

A manutenção corretiva de urna eletrônica ocorre quando o equipamento apresenta algum tipo de problema e é realizado na Central de Apoio Técnico (CAT) em Salvador e nos polos de informática no interior pelos técnicos de urnas.

3.2. Artefatos de definição do sistema

Como dito na Seção 2, uma das atividades realizadas neste trabalho foi o levantamento de requisitos do sistema de manutenção de urnas eletrônicas. Segundo o blog de tecnologia Cedro Tecnologics, levantamento de requisitos é *"o processo de compreensão e identificação das necessidades que o cliente espera ser solucionado pelo sistema que será desenvolvido, definindo o que o software vai fazer. É a primeira etapa no ciclo de desenvolvimento de software, onde são definidas as funcionalidades e o escopo do projeto"* [Site CEDRO, 2021].

Outra atividade realizada neste trabalho foi a prototipagem das telas do sistema proposto. De acordo com o blog de Tecnologia Medium, prototipagem de software é *"uma estratégia concreta, simples e familiar para apresentar ao usuário antecipadamente as funcionalidades requeridas para o software"* [Site Medium, 2021].

Além das atividades acima foram feitos os diagramas Modelo de Entidade Relacionamento (MER) e Modelo de Banco de Dados do sistema proposto. Segundo o site de tecnologia Devmedia, Modelo de Entidade Relacionamento (MER) é *"um modelo conceitual utilizado na Engenharia de Software para descrever os objetos (entidades) envolvidos em um domínio de negócios, com suas características (atributos) e como elas se relacionam entre si (relacionamentos)"* [Site Devmedia, 2021]. Segundo o site Lucidchart, o Modelo de Banco de Dados mostra *"a estrutura lógica de um banco de dados, incluindo as relações e restrições que determinam como os dados podem ser armazenados e acessados"* [Site Lucidchart, 2021].

4. Requisitos e modelagem do sistema proposto

O sistema proposto tem como fundamentação a necessidade de informatizar o controle e gerenciamento das manutenções preventivas e corretivas das urnas eletrônicas que hoje são feitas na Central de Apoio Técnico (CAT) e nos polos pelos gerentes e técnicos de urna.

A partir dessa necessidade surgiu uma demanda da Seção de Urna Eletrônica (SEUEL), através do chefe da seção, por um aplicativo web/mobile que fizesse o gerenciamento e controle dessas manutenções.

A aplicação deverá gerenciar as seguintes informações: Urnas Eletrônicas, Depósitos de Urnas Eletrônicas, Técnicos de Urnas Eletrônicas, Ordens de Serviço, Atividades de Manutenção e Peças, através de cadastramentos e relatórios.

Módulos do Sistema proposto:

- Cadastramento de Urnas Eletrônicas

Opções: Inclusão, Alteração e Exclusão das Urnas Eletrônicas

- Cadastramento de Depósitos de Urnas Eletrônicas

Opções: Inclusão, Alteração e Exclusão de Depósitos de Urnas Eletrônicas

- Cadastramento de Técnicos de Urnas Eletrônicas

Opções: Inclusão, Alteração e Exclusão de Técnicos de Urnas Eletrônicas

- Cadastramento de Ordens de Serviço

Opções: Inclusão, Alteração e Exclusão de Ordens de Serviço de manutenção preventiva e corretiva de Urnas Eletrônicas

- Cadastramento de Atividades de Manutenção

Opções: Inclusão, Alteração e Exclusão dos tipos de Atividades de Manutenção preventiva e corretiva de Urnas Eletrônicas

- Cadastramento de Peças

Opções: Inclusão, Alteração e Exclusão de Peças

- Relatório de Urnas Eletrônicas
- Relatório de Depósitos de Urnas Eletrônicas
- Relatório de Técnicos de Urnas Eletrônicas
- Relatório de Ordens de Serviço
- Relatório de Atividades de Manutenção
- Relatório de Peças

4.1 Recursos tecnológicos previstos

Será utilizada a ferramenta Ionic, que é um framework open source para desenvolvimento de aplicativos móveis multiplataforma [Devmedia, 2021] já utilizado pela SEDESC e banco de dados Oracle, que é o banco de dados relacional utilizado pelo TRE-BA .

4.2 Requisitos funcionais e não funcionais

Os requisitos funcionais são os problemas e necessidades que devem ser atendidos e resolvidos pelo sistema por meio de funções ou serviços, ou seja, descrevem o que será feito. Os requisitos não funcionais são aqueles relacionados à forma como o sistema tornará realidade o que está sendo planejado, ou seja, descrevem como será feito. [Mestres da Web, 2021]

4.2.1 Requisitos funcionais:

- RF01 - Efetuar login

  O sistema deverá possuir uma função que permita efetuar o login com usuário e senha, e redirecionar o usuário para a tela relacionada a seu perfil que pode ser Administrador, Gerente de Pólo ou Técnico de Urna.

- RF02 - Gerenciar perfil

  O sistema deverá possuir uma função que permita ao Administrador, incluir, alterar ou excluir perfil de Gerente de Pólo e Técnico de Urna.

- RF03 - Gerenciar Ordem de Serviço

  O sistema deverá assegurar que a Ordem de Serviço terá um número único

  O sistema deverá assegurar que a Ordem de Serviço estará associada a um Depósito de Urnas

O sistema deverá assegurar que a Ordem de Serviço terá uma descrição

O sistema deverá assegurar que a Ordem de Serviço terá uma data de início e de fim de execução

O sistema deverá assegurar que a Ordem de Serviço terá uma quantidade de urnas a serem manutenidas

O sistema deverá assegurar que a Ordem de Serviço poderá ou não ter baterias a serem manutenidas

O sistema deverá assegurar que a Ordem de Serviço terá as atividades realizadas em cada urna registrada

O sistema deverá assegurar que a Ordem de Serviço terá as peças utilizadas na ordem de serviços registradas

- RF04 - Gerenciar Atividade de Manutenção

  O sistema deverá assegurar que a Atividade de Manutenção terá um código único

  O sistema deverá assegurar que a Atividade de Manutenção terá uma descrição

- RF05 - Gerenciar Peça

  O sistema deverá assegurar que a Peça terá um código único

  O sistema deverá assegurar que a Peça terá uma descrição

- RF06 - Gerenciar Urna

  O sistema deverá assegurar que toda Urna deverá possuir:

  Ano do modelo

  Número do patrimônio

  O sistema deverá assegurar que toda Urna estará em um Depósito de Urnas

- RF07 - Gerenciar Depósito de Urnas

  O sistema deverá assegurar que todo Depósito de Urnas deverá possuir:

  Código único

  Município

  O sistema deverá assegurar que todo Depósito de Urnas terá um gerente de pólo

- RF08 - Gerenciar Técnico de Urna

  O sistema deverá assegurar que todo Técnico de Urna deverá possuir:

  Número do RG

  Nome

  O sistema deverá assegurar que o Técnico de Urna estará associado a um Depósito de Urnas

4.2.2 Requisitos não funcionais:

- RNF01 - Implementação

  O sistema será desenvolvido utilizando a ferramenta Ionic e o Banco de Dados Oracle.

- RNF02 - Portabilidade

  O sistema deverá funcionar na última versão dos navegadores Google Chrome e Firefox, e mobile no IOS e Android.

- RNF03 - Disponibilidade

  O sistema deverá estar disponível para ser utilizado a qualquer tempo, exceto nos horários de backup programados pela Seção de Infraestrutura (SEINFRA).

- RNF04 - Segurança

  O sistema deverá estar disponível somente para pessoas autorizadas, através de login e senha da rede do TRE-BA.

4.3 Diagramas do sistema proposto

A Figura 2 descreve o Modelo de Entidade Relacionamento (MER) do sistema proposto. As entidades são objetos do mundo real e são representadas por retângulos, os relacionamentos são as interações entre as entidades e são representados por losangos e os atributos são as propriedades dessas entidades e são representados por elipses. A partir desse modelo pode-se observar, por exemplo, que um Depósito pode armazenar zero ou várias Urnas, mas uma Urna só pode estar armazenada em um Depósito, uma ou mais Urnas podem ser manutenidas por zero ou mais O.S., uma ou mais O.S. podem realizar uma ou mais Atividades de manutenção .

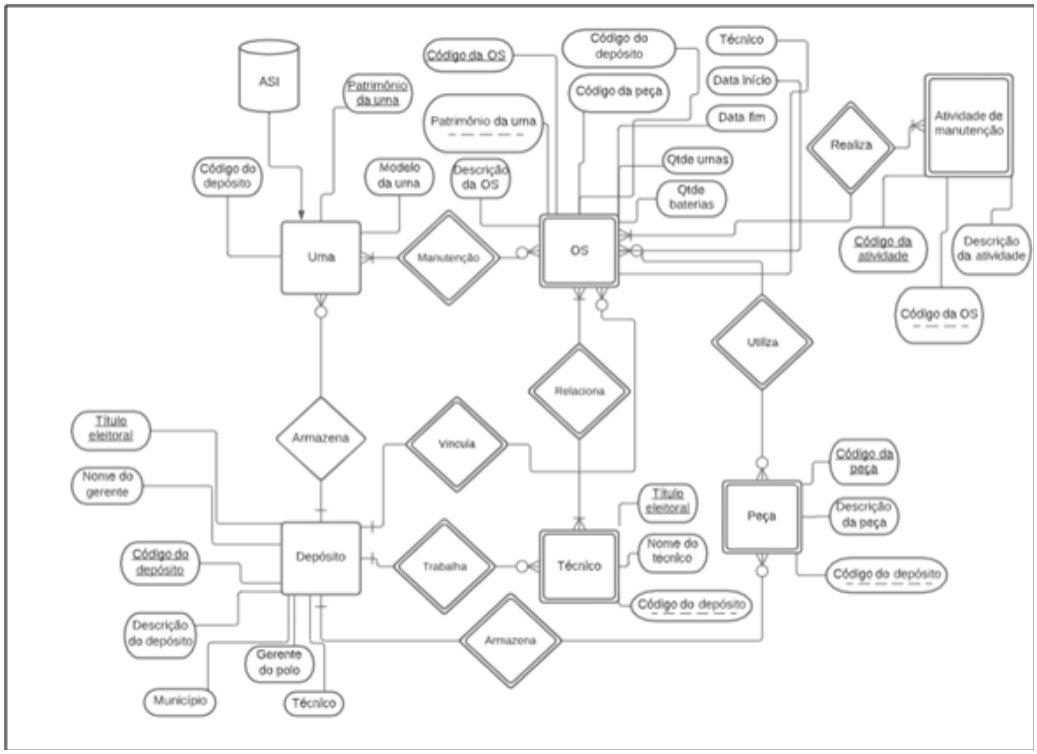


Figura 2: Modelo de Entidade Relacionamento (MER)

A Figura 3 descreve o Modelo de Banco de Dados do sistema proposto. A partir desse modelo observa-se que as tabelas entidade-relacionamento estão representadas na figura e são: urna, deposito, os, tecnico, atividade_manutencao, peca, urna_os_ativ, urna_os_peca. A tabela urna, por exemplo, é composta pelos campos: patrimonio_urna (varchar com 8 posições e pk (chave primária)), modelo (inteiro), cod_deposito (inteiro e fk (chave estrangeira)).

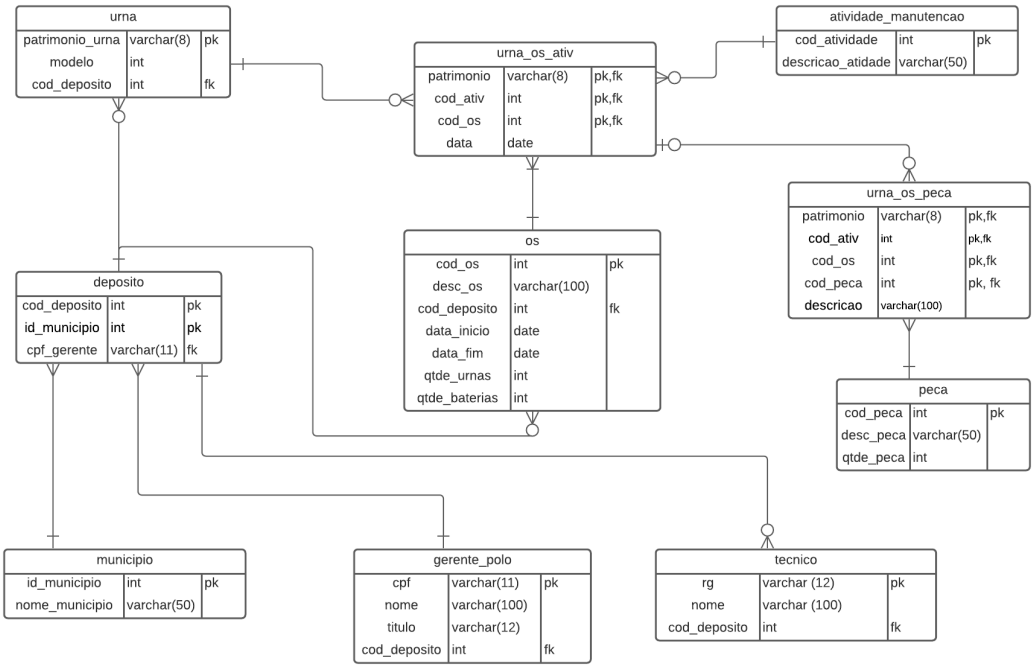


Figura 3: Modelo de Banco de Dados

4.4 Protótipos das telas principais do sistema proposto

Os protótipos de algumas telas do sistema já estão prontos, sendo eles: a Tela de Login do Sistema (Figura 4), onde o usuário irá acessar o sistema com o seu login e senha da rede; Tela Inicial do Sistema (Figura 5), com as opções Cadastros e Relatórios; Tela Cadastros do Sistema (Figura 6), com todos os cadastramentos do sistema, sendo eles: Urnas Eletrônicas, Depósitos de Urnas, Técnicos de Urnas, Ordens de Serviço, Atividades de Manutenção e Peças; e Tela Relatórios do Sistema (Figura 7), com todos os relatórios do sistema, sendo eles: Urnas Eletrônicas, Depósitos de Urnas, Técnicos de Urnas, Ordens de Serviço, Atividades de Manutenção e Peças.

Figura 4: Tela Login do Sistema

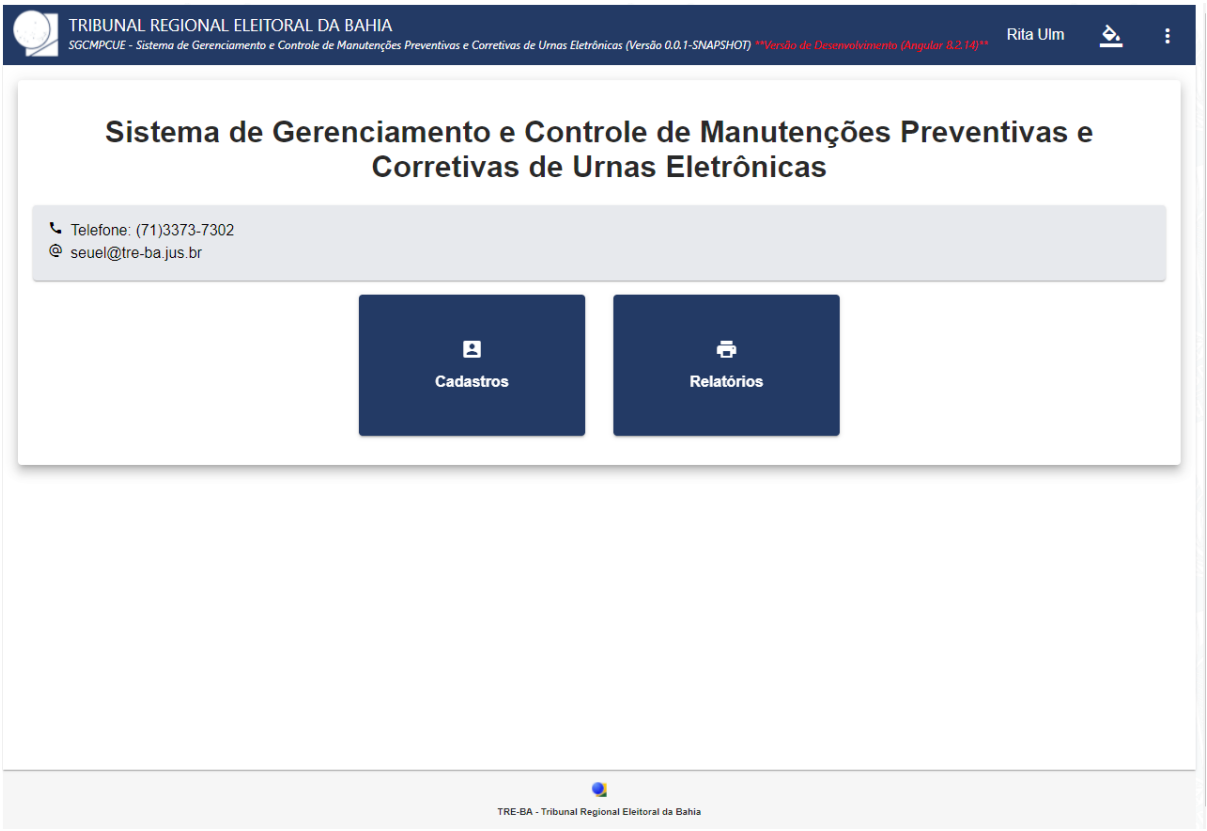


Figura 5: Tela Inicial do Sistema

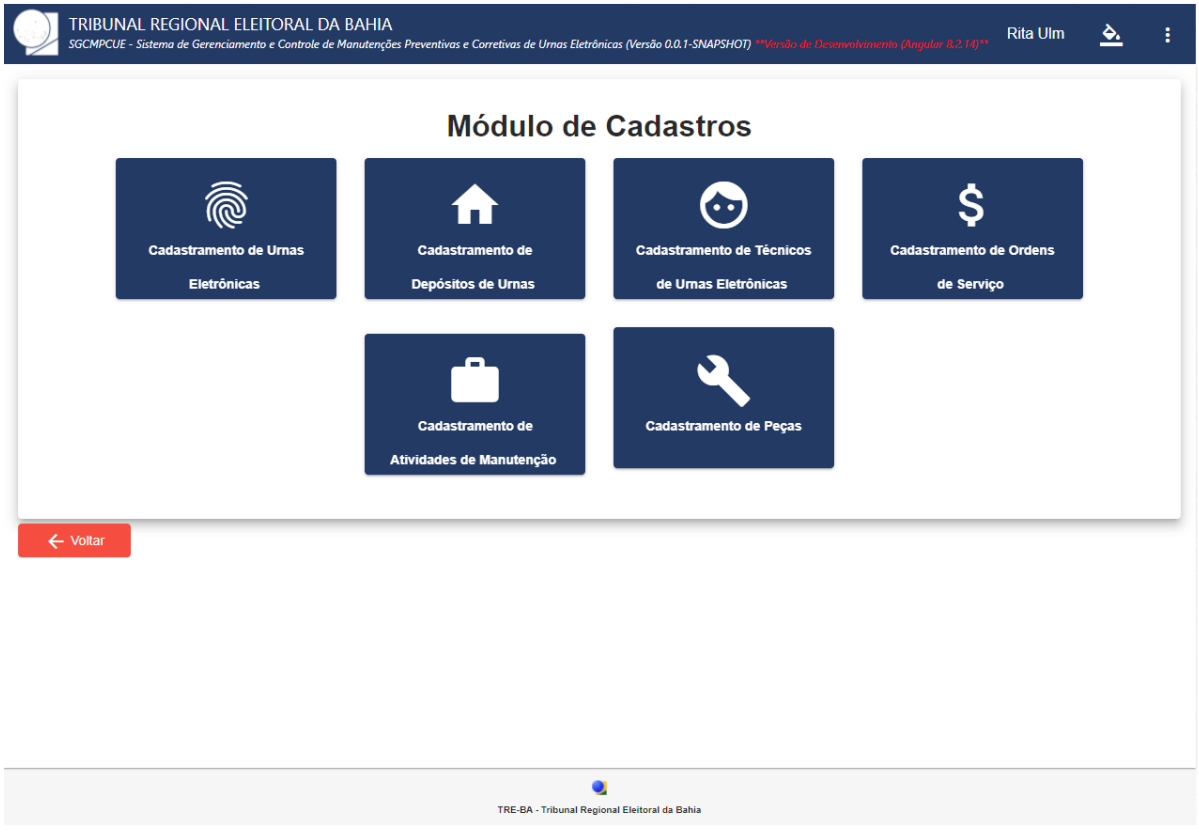


Figura 6: Tela Cadastros do Sistema

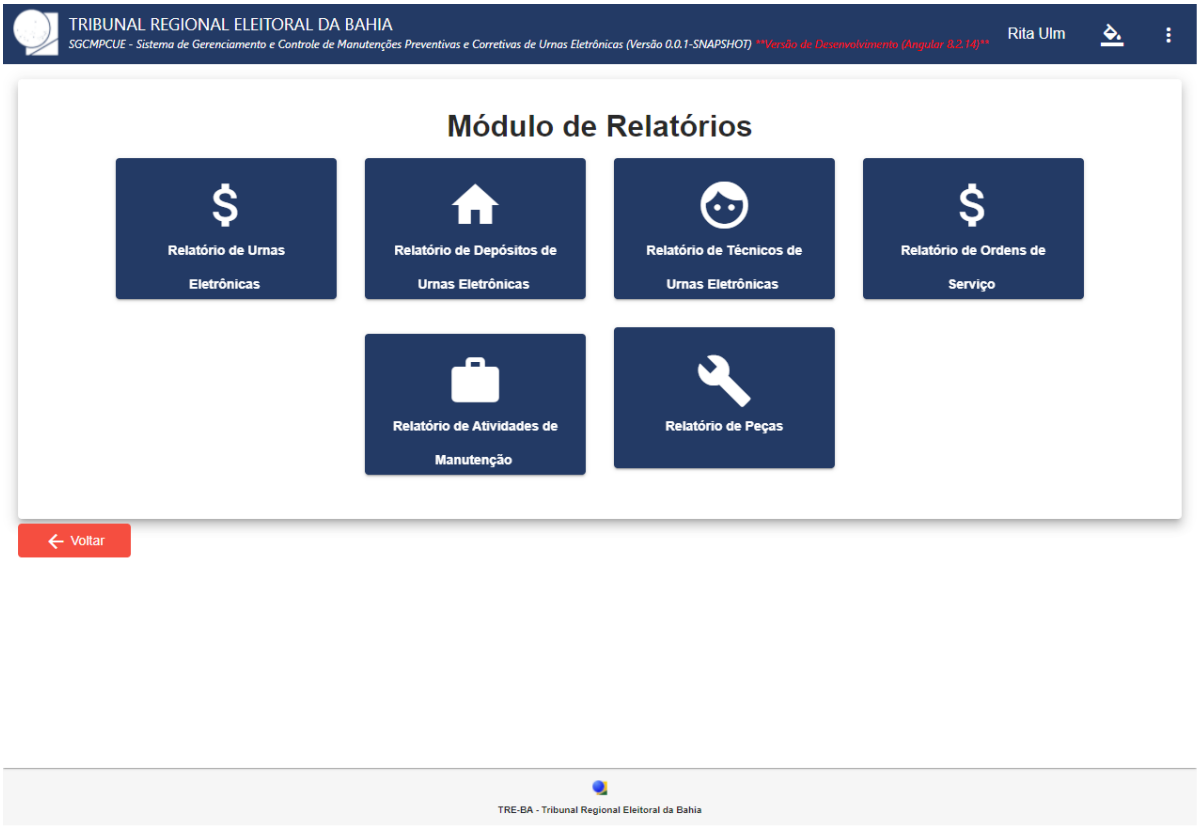


Figura 7: Tela Relatórios do Sistema

5. Conclusão

Diante do exposto no artigo, conclui-se que a implementação do aplicativo de gerenciamento e controle de manutenções preventivas e corretivas de urnas eletrônicas pela Seção de Desenvolvimento de Soluções Corporativas (SEDESC) será de grande relevância, não somente para a Seção de Urna Eletrônica (SEUEL), mas também para todo o Tribunal Regional Eleitoral da Bahia, tendo em vista ser a missão do TRE-BA "Garantir a legitimidade do processo eleitoral" e a visão "Fortalecer a credibilidade da Justiça Eleitoral, prestando serviço de qualidade e promovendo a cidadania" [TRE-BA, 2021].

Baseado nesses direcionadores estratégicos definidos pelo TRE-BA para 2021-2026, que integra o Planejamento Estratégico Institucional (PEI) para os próximos cinco anos e norteará as ações a serem implantadas durante o período, é de fundamental importância a realização de eleições seguras e rápidas, e para isso é necessário que as urnas estejam em perfeitas condições de uso para as eleições [TRE-BA, 2021].

Ainda não foi feita uma aplicação prática do sistema, porque ele ainda não foi desenvolvido, mas a ideia é que isso seja feito em um futuro próximo. Caso isso ocorra, o sistema irá automatizar o gerenciamento e controle de manutenções preventivas e corretivas de urnas eletrônicas, que hoje é feito de forma manual, gerando com isso um enorme ganho de produtividade, além de aumentar substancialmente a segurança do processo.

Referências

[TRE-BA] Portal do Tribunal Regional Eleitoral da Bahia, disponível em https://www.tre-ba.jus.br/. Último acesso em Novembro de 2021.

[TSE] Portal do Tribunal Superior Eleitoral, disponível https://www.tse.jus.br/. Último acesso em Novembro de 2021.

[TRE-SP] Portal do Tribunal Regional Eleitoral de São Paulo, disponível em https://www.tre-sp.jus.br/. Último acesso em Novembro de 2021.

[Site Devmedia] Devmedia, disponível em https://www.devmedia.com.br/. Último acesso em Novembro de 2021.

[Site Lucidchart] Lucidchart, disponível em https://lucidchart.com. Último acesso em Novembro de 2021.

[Site CEDRO] Cedro Tecnologics, disponível em https://blog.cedrotech.com. Último acesso em Novembro de 2021.

[Site Medium] Medium, disponível em https://medium.com/. Último acesso em Novembro de 2021.

[Site Mestres da Web] Mestres da Web, disponível em https://mestresdaweb.com.br/. Último acesso em Novembro de 2021.