**FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS**

**FATEC PROFESSOR Jessen Vidal**

**CESAR AUGUSTO SIQUEIRA SANTOS**

**QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC**

São José dos Campos

2019

**CESAR AUGUSTO SIQUEIRA SANTOS**

**QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados.

**Orientador: Me. Diogo Branquinho**

São José dos Campos

2019

**Dados Internacionais de Catalogação-na-Publicação (CIP)**

**Divisão de Informação e Documentação**

SANTOS, Cesar Augusto Siqueira

QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC.

São José dos Campos, 2019.

999f. (número total de folhas do TG)

Trabalho de Graduação – Curso de Tecnologia em Banco de Dados.

FATEC de São José dos Campos: Professor Jessen Vidal, 2019.

Orientador: Me. Diogo Branquinho.

1. Simulado. 2. *NoSQL*. 3. *Webview*. I. Faculdade de Tecnologia. FATEC de São José dos Campos: Professor Jessen Vidal. Divisão de Informação e Documentação. II. Título

**REFERÊNCIA BIBLIOGRÁFICA**

SANTOS, Cesar Augusto Siqueira **QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC.** 2019. 999f. Trabalho de Graduação - FATEC de São José dos Campos: Professor Jessen Vidal.

**CESSÃO DE DIREITOS**

NOME(S) DO(S) AUTOR(ES): Cesar Augusto Siqueira Santos

TÍTULO DO TRABALHO: QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC

TIPO DO TRABALHO/ANO: Trabalho de Graduação/2019.

É concedida à FATEC de São José dos Campos: Professor Jessen Vidal permissão para reproduzir cópias deste Trabalho e para emprestar ou vender cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste Trabalho pode ser reproduzida sem a autorização do autor.

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Cesar Augusto Siqueira Santos  Rua José Firmino de Moraes, Número 121, Jardim Estoril  12232-020, São José dos Campos – São Paulo |  |

**CESAR AUGUSTO SIQUEIRA SANTOS**

**QuizFATEC: Aplicativo Móvel Simulador dos Vestibulares da FATEC**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Me. Diogo Branquinho – FATEC SJC**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titulação, Nome do Componente da Banca – FATEC SJC**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Titulação, Nome do Componente da Banca – FATEC SJC**

**\_\_\_\_\_/\_\_\_\_\_/\_\_\_\_\_**

**DATA DA APROVAÇÃO**

Dedicatória (opcional)

O autor oferece a obra (elemento sem título e sem indicativo numérico), ou presta homenagem a alguém, de forma clara e breve em folha única.

**AGRADECIMENTOS**

Na página de agradecimentos o autor dirige palavras de reconhecimento àqueles que contribuíram para a elaboração do trabalho. O conteúdo não deve ultrapassar uma página e por isso, é necessário que ele seja sucinto e objetivo.

Agradecer ao orientador, aos amigos, professores que colaboraram.

O texto deve ser escrito em Times New Roman, Tamanho 12, Alinhamento Justificado, Espaçamento entre linhas de 1,5 linhas e com recuo de parágrafo de 1,25 cm.

Epígrafe (opcional)

“É citada uma sentença escolhida pelo autor (elemento sem título e sem indicativo numérico), que deve guardar coerência com o tema abordado na obra.”

Nome do autor

**RESUMO**

Apresentação concisa dos pontos relevantes do documento deve ser exposta no resumo. No presente caso o resumo será informativo, assim deverá ressaltar o objetivo, a metodologia, os resultados e as conclusões do documento. A ordem desses itens depende do tratamento que cada item recebe no documento original. O resumo deve ser composto por uma sequência de frases concisas, afirmativas e não em enumeração de tópicos. Deve ser escrita em parágrafo único e espaçamento de 1,5 linhas. A primeira frase deve ser significativa, explicando o tema principal do documento. Deve-se usar o verbo na voz ativa e na terceira pessoa do singular. Quanto a sua extensão, o resumo deve possuir de 150 a 500 palavras.

**Palavras-Chave**: Com um mínimo de 3 e no máximo 6 palavras, separadas entre si por ponto e vírgula “;” e finalizadas por ponto. As palavras-chave sãopalavras representativas do conteúdo do documento.

**ABSTRACT**

O abstract é o resumo da obra em língua estrangeira, que basicamente segue o mesmo conceito e as mesmas regras que o texto em português. Recomenda-se que para o texto do abstract o autor traduza a versão do resumo em português e faça, se necessário, os ajustes referentes à conversão dos idiomas. É importante observar que o título e texto NÃO DEVEM estar em itálico.

**Keywords**: Recomenda-se que o autor traduza para o inglês as Palavras-Chave em português e faça, se necessário, os ajustes referentes à conversão dos idiomas.

**LISTA DE QUADROS**

[Quadro 1 - Scrapper - Imports e Declarações Globais 32](#_Toc25614377)

[Quadro 2 - Scrapper - Inserção de Dicionários no MongoDB 33](#_Toc25614378)

[Quadro 3 - Scrapper - Raspagem de Gabarito 34](#_Toc25614379)

[Quadro 4 - Scrapper - PDF\_TO\_TEXT 35](#_Toc25614380)

[Quadro 5 - Scrapper - Text to JSON Question 38](#_Toc25614381)

[Quadro 6 - Scrapper Main Function 41](#_Toc25614382)

[Quadro 7 - Backend: Conexão com Banco de Dados 43](#_Toc25614383)

[Quadro 8 - Backend: 3º e 4º Rotas 44](#_Toc25614384)

[Quadro 9 - Backend: 5º e 6º Rota 45](#_Toc25614385)

[Quadro 10 - Classe DataService 48](#_Toc25614386)

**LISTA DE FIGURAS**

[Figura 1 - Estrutura básica JSON 25](#_Toc25614366)

[Figura 2 - Exemplo de JSON estruturado 25](#_Toc25614367)

[Figura 3 - Esquema tático aplicação Cordova 27](#_Toc25614368)

[Figura 4 - Arquitetura Geral do Projeto 29](#_Toc25614369)

[Figura 5 - Composição Geral de um Notebook Jupyter 31](#_Toc25614370)

[Figura 8 - Comparativo de Gabaritos 34](#_Toc25614371)

[Figura 13 - Dashboard QuizFATEC 42](#_Toc25614372)

[Figura 14 - Exemplo de Documento Inserido 42](#_Toc25614373)

[Figura 18 - Arquitetura do Aplicativo QuizFATEC 47](#_Toc25614374)

[Figura 19 - Tela de Login Preenchida 50](#_Toc25614375)

[Figura 20 - Proposta metodológica. 59](#_Toc25614376)

**LISTA** **DE TABELAS**

[Tabela 1 - Bibliotecas Python 28](#_Toc21472262)

[Tabela 2 - População de 15 a 24 anos de idade. 44](#_Toc21472263)

**LISTA DE ABREVIATURAS E SIGLAS**

PDF

API Application Programming Interface

FATEC Faculdade de Tecnologia

HTTP Hypertext Transfer Protocol

JSON *Javascript Object Notation*

SDK *System Development Kit*

TI Tecnologia da Informação

UI *User Interface*

UX *User Experience*

URL *Uniform Resource Locator*

IO *Input Output*

HTML

NoSQL *Not Only Struct Query Language*

IP

WSGI *Web Server Gateway Interface*

CORS *Cross-Origin Resource Sharing*

OCR *Optical Character Recognition*

KNN K Vizinhos Mais Próximos

**SUMÁRIO**

[1. INTRODUÇÃO 17](#_Toc29037162)

[1.1. Motivação 18](#_Toc29037163)

[1.2 Objetivo 18](#_Toc29037164)

[1.3. Escopo 19](#_Toc29037165)

[2. FUNDAMENTAÇÃO TEÓRICA 21](#_Toc29037166)

[2.1. Técnicas de Obtenção de Dados 21](#_Toc29037167)

[2.1.1. Transcrição Manual 21](#_Toc29037168)

[2.1.2. OCR – Reconhecimento Óptico de Caracteres 21](#_Toc29037169)

[2.1.3. Data Scraping e Python 22](#_Toc29037170)

[2.1.3.1 PDFMiner 22](#_Toc29037171)

[2.1.3.2 PyPDF2 22](#_Toc29037172)

[2.2. Tecnologias Semelhantes 23](#_Toc29037173)

[2.2.1. Aplicativo Perguntados 1 e 2 23](#_Toc29037174)

[2.2.2. Simulado Detran-SP 23](#_Toc29037175)

[2.3. Vestibulares e o Fator Psicológico 24](#_Toc29037176)

[2.4. Banco de Dados NoSQL 25](#_Toc29037177)

[2.4.1. JSON 25](#_Toc29037178)

[2.4.2. MongoDB 26](#_Toc29037179)

[2.5. Webview para Dispositivos Móveis – Ionic e Cordova Framework 26](#_Toc29037180)

[3. DESENVOLVIMENTO 29](#_Toc29037181)

[3.1. Arquitetura Global do Sistema 29](#_Toc29037182)

[3.2 Scrapper 30](#_Toc29037183)

[3.2.1 Scrapper – Imports e Declarações Globais 31](#_Toc29037184)

[3.2.2 Scrapper – Inserção de Dicionários no MongoDB 32](#_Toc29037185)

[3.2.3 Scrapper – Raspagem do Gabarito 33](#_Toc29037186)

[3.2.4 Scrapper – Retirada de Texto do PDF de Prova 35](#_Toc29037187)

[3.2.5 Scrapper – Retirada das Questões do Texto da Prova 37](#_Toc29037188)

[3.2.6 Scrapper – Função Principal 40](#_Toc29037189)

[3.3 Persistência dos dados através de MongoDB 41](#_Toc29037190)

[3.4 *Backend* em Flask 43](#_Toc29037191)

[3.5 Aplicativo QuizFATEC 46](#_Toc29037192)

[3.5.1 DataService - Comunicação com API 47](#_Toc29037193)

[3.5.2 Tela de Login e suas funções 49](#_Toc29037194)

[3.5.3 Tela de Cadastro e suas funções 50](#_Toc29037195)

[3.5.4 Tela Home e suas funções 51](#_Toc29037196)

[3.5.4 Tela das Questões e suas funções 52](#_Toc29037197)

[4. RESULTADOS 56](#_Toc29037198)

[4.1. *User Expirence* 56](#_Toc29037199)

[4.2. Tecnologias Semelhantes 56](#_Toc29037200)

[4.3. Requisitos Funcionais 56](#_Toc29037201)

[4.3.1 Vestibulares da FATEC como foco do estudo 56](#_Toc29037202)

[4.3.2 Aplicação Multiplataforma 57](#_Toc29037203)

[4.3.3 Cadastro de Usuário e Login 57](#_Toc29037204)

[4.3.4 Escolher o Tema das Questões que serão foco do estudo 57](#_Toc29037205)

[4.3.5 Resposta Instantânea 57](#_Toc29037206)

[4.3.6 Questões Cronometrados 57](#_Toc29037207)

[5. CONSIDERAÇÕES FINAIS 58](#_Toc29037208)

[5.1. Contribuições 58](#_Toc29037209)

[5.2. Trabalhos Futuros 58](#_Toc29037210)

[REFERÊNCIAS 59](#_Toc29037211)

[APÊNDICE A/ANEXO A – EXEMPLO DE APÊNDICE/ANEXO 62](#_Toc29037212)

[Como deve ser a formatação das Figuras, Tabelas e Equações no trabalho 63](#_Toc29037213)

[Como deve ser mencionada as Siglas no trabalho 65](#_Toc29037214)

[Como deve ser feitas as citações no trabalho 65](#_Toc29037215)

[Como utilizar as referências bibliográficas no texto do trabalho 66](#_Toc29037216)

# 1. INTRODUÇÃO

Em virtude da crise política e econômica vivenciada no Brasil nos anos de 2017 e 2018, o desemprego cresceu, mais precisamente 3,6% considerando o desemprego em janeiro de 2017 e o mesmo período no ano anterior. (g1 2017, p. 1)

O acirramento da competição no mercado de trabalho, leva milhões de pessoas a buscar atualizações, principalmente no âmbito de cursos superiores e pós-graduações. Pensando nisso, a FATEC foi criada pelo Centro Paula Souza, uma instituição pública, mantida pelo governo do Estado de São Paulo.

A FATEC disponibiliza duas vezes por ano mais 14 mil vagas, distribuídos entre 74 cursos focados na área de tecnologia. Levando milhares de jovens, adultos e até idosos a provarem seus conhecimentos em um vestibular, para se aptos ingressar na Faculdade de Tecnologia do Estado de São Paulo.

O Centro Paula Souza, órgão responsável pela administração das FATECs e ETECs espalhadas pelo Estado de São Paulo, foi criado em 1969, atualmente vinculado à Secretaria de Desenvolvimento do Estado de São Paulo, o objetivo do Centro Paula Souza sempre foi implantar um ensino técnico voltado para formação de profissionais direcionados ao mercado de trabalho, transpassando as instituições de ensino superior focados em discussões acadêmicas (Portal do Governo do Estado de São Paulo, 2009).

As primeiras FATECs foram implantadas na década de 1970, começando em Sorocaba, interior do estado de São Paulo, e São Paulo capital. Atualmente já passam das 70 unidades, espalhadas em 66 cidades. (FATEC, 2018).

O vestibular da FATEC tem um diferencial em relação aos outros, ele acontece duas vezes por ano, com provas totalmente distintas. Por se tratar de tecnólogos, são cursos de período relativamente curtos e atraem pessoas das mais distintas idades e classes sociais.

Muitos vestibulandos da FATEC, concluíram o ensino médio a décadas e não voltaram a estudar, tornando o vestibular um desafio muito grande. Ainda mais se o vestibulando não conhece o modelo da prova, áreas de conhecimento abrangidas, moldes da redação etc. Por fim, as chances desse vestibulando ingressar na FATEC, ou qualquer vestibular, são pouco animadoras. Será preciso dedicação e estudo, simulação das provas, dedicação as matérias que têm mais peso para o curso almejado.

Portanto, a falta de conhecimento, também das áreas de conhecimento que mais precisam de atenção, estudo e prática antes de se realizar um vestibular, pode custar a vaga de um vestibulando em busca de melhores condições de vida, ou mesmo de uma quinada na carreira profissional.

Muitos vestibulandos precisam prestar mais de uma vez o vestibular, uma para identificar onde é necessária mais atenção, compreender a prova e suas estratégias, e outra para ter chances reais de ingressar na instituição de ensino superior. Isso é um empecilho na vida de quem está desempregado, buscando se especializar para ter uma chance maior no mercado de trabalho, ou até mesmo de um recém-formado no ensino médio buscando continuar seus estudos com qualidade.

Existe portanto um abismo claro, entre o desejo de ingressar em uma instituição de ensino superior e a obtenção dos conhecimentos necessários para tal. Atualmente isso tem sido suprido com cursinhos e escolas preparatórias para vestibular, entretanto a maior parte dessas instituições é paga, e a maioria das pessoas que desejam aprimorar seus conhecimentos não despende de tempo ou dinheiro sobrando para gastos em cursinhos ou escolas preparatórias. Evidenciando ainda mais, a dificuldade de jovens e adultos com carreira estabelecida ingressarem em instituições de ensino superior.

## 1.1. Motivação

Não existir um banco de questões de vestibulares de acesso publico e facilitado, agilizando o estudo de vestibulandos.

/\*TODO\*/

Como coletar dados de vestibulares de maneira a compor um banco de dados para futuras simulações

Técnicas de raspagem: OCR, digitação, raspagem e afins. Embasamento

Objetivo construir um banco de dados com as questões

Raspar é a solução para o problema

## 1.2 Objetivo

Desenvolver um banco de dados alimentado apenas por questões, repostas e textos provenientes de vestibulares, aliado a um aplicativo para dispositivos moveis que permita que vestibulandos revejam questões de vestibulares passados.

## 1.3. Escopo

De maneira geral, a melhor maneira de se preparar para qualquer vestibular, é entender quais os principais assuntos tratados no vestibular em questão e realizar simulações cronometradas para se habituar com o estilo da prova. Segundo Nestor Tavorá, professor da LFG Concursos, realizar o exame simulado nas condições exatas da realidade, ajudam na aprovação, devido a familiaridade com o modelo de vestibular.

Realizar um vestibular, com conteúdo e estratégia, mais do que só conhecimento técnico, mas já ter vivência no teste, garantem resultados melhores. Os pontos e a nota são baseados no que foi feito e não no que poderia ser feito. (Tasinafo, 2018)

Criação de um aplicativo para smartphones, com a capacidade de simular provas cronometradas no modelo dos Vestibulares da FATEC, ao fim de cada simulação o aplicativo indicará a nota atingida no mesmo modelo do vestibular da FATEC, considerando, portanto a fórmula de cálculo com peso relativo ao curso escolhido.

Quanto ao desenvolvimento do projeto, será feito em 4 etapas:

1 – Raspagem das questões e devidas respostas de vestibulares passados da FATEC.

2 – Alimentação de um Banco de Dados com as questões e repostas extraídas.

3 – Construção de uma API conectando a aplicação Webview e Banco de Dados.

4 – Desenvolvimento de um aplicativo para ambas as plataformas moveis (iOS e Android).

A extração das questões será feita usando scripts de mineração de dados em Python, usando a biblioteca PyPDF2 e PDFminer, criadas propriamente para leitura de arquivos PDF e raspagem. Essa será uma fase desafiadora, já que o processo de raspagem de tantos documentos e possíveis padrões variáveis dificultam a definição de um padrão de raspagem, requerendo um script de raspagem de dados mais adaptativo.

A fonte da informação e questões será o próprio site de vestibular da FATEC, lá se encontram todas os vestibulares ocorridos, assim como as o gabarito de respostas e questões anuladas.

A partir da extração dos dados, um banco de dados será alimentado com as questões e soluções. A definição do modelo, modelagem das tabelas e definição dos campos, será feito de maneira a melhorar a performance do sistema.

O aplicativo será feito em Ionic, pensando na possibilidade de multiplataformas, além disso a capacidade de ágil desenvolvimento. O desenvolvimento com linguagem nativa, não se torna obrigatório, porque a aplicação fará pouco uso dos recursos nativos do smartphone. A maioria dos recursos e ações será executada dentro da própria aplicação, como responder simulados, obter questões randômicas e assim por diante.

Dessa maneira o usuário, poderá comparar o nível de seu conhecimento com a nota geral da FATEC, além de poder compreender áreas de conhecimento que exigem mais atenção, ajudando-o a se preparar para o vestibular propriamente dito.

# 2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo expõe as principais áreas de conhecimento e tecnologias abrangidas durante esse trabalho: técnicas de raspagem de dados, tecnologias semelhantes, vestibulares e fator psicológico, bancos de dados não relacionais e webview para dispositivos móveis.

## 2.1. Técnicas de Obtenção de Dados

Existem inúmeras formas de se extrair dados de textos, arquivos, imagens, sites etc. Este tópico abrangera algumas das alternativas existentes para raspagem de arquivos.

### 2.1.1. Transcrição Manual

A alternativa mais simples seria a transcrição dos vestibulares passados em documentos para serem inseridos dentro do banco de dados, é menos eficiente e extremamente trabalhosa, levaria até mais tempo do que outros métodos, entretanto até a invenção da imprensa a produção de livros e documentos era feito de através de um laborioso processo manual, eram utilizados formas e procedimentos padrão. (BEZERRA, 2011)

### 2.1.2. OCR – Reconhecimento Óptico de Caracteres

*Optical Character Recognition* (OCR), reconhecimento ótico de caracteres em tradução literal, patenteado em 1929 na Alemanha e 1935 nos Estados Unidos, trata-se de algoritmos, ou inteligência artificial para realizar o reconhecimento de caracteres de imagens. As técnicas de OCR tem como essência replicar a capacidade humana de interpretação de símbolos e leitura. Para que essa ação seja possível é necessário aplicar inúmeras tecnologias, porque existem desafios como a variação de fontes, estilos e tamanhos, assim como matérias de impressão, qualidade da imagem, quando o assunto é materiais manuscritos o desafio se torna ainda maior.

As redes neurais são a aplicação de inteligência artificial mais difundida para o OCR, isso se deve a facilidade com o aprendizado, modificando a resposta de acordo com um estímulo de entrada ou aprendizado, essa capacidade se assemelha ao cérebro humano porque 1- o conhecimento é adquirido e interpretado através de um processo de aprendizagem, 2 – o conhecimento adquirido é armazenado através da força entre neurônios, ou pesos sinápticos. (HAYKIN, 2001).

Uma outra maneira técnica utilizado para OCR é o algoritmo de K-vizinho mais próximos, ou KNN, uma técnica de aprendizado baseada em casos, esse algoritmo é fundamentado no princípio de que casos semelhantes deverão ter a mesma classe (PASSOS, 2015). Através de fórmulas matemáticas como a distância euclidiana definida por , ou pela distância de Manhatan, conforme , onde e representam os valores do atributo i nos casos **a** e **b** (PASSOS, 2015). A figura 1 abaixo exemplifica um caso em que KNN foi utilizado para rotular a estrela, assim sendo dado um K = 7 (distância) e rótulos A e B representados por pequenos círculos. Dada a distância a circunferência pontilhada foi desenhada, dentro da circunferência restaram 4 elementos de rotulo A e 3 elementos de rotulo B, assim sendo a estrela foi classificada como rotulo A.

Figura 1 - Exemplo aplicado de KNN

Uma imagem contendo objeto

Descrição gerada automaticamente

[Fonte: PACHECO (2019)](http://fatecsjc-prd.azurewebsites.net/moodle/pluginfile.php/65111/mod_resource/content/0/Exemplo_de_Projeto_Android_de_Tempo_Real_com_Firebase.pdf)

### 2.1.3. Data Scraping e Python

Com o crescimento da internet, na última década, muita informação está à disposição para aqueles que souberem como buscar, extrair e transformar em essa informação em algo realmente útil, a esse processo é dado o nome de Web Scraping, ou Raspagem Web.

O uso mais comum do Web Scraping é a extração de conteúdos web, principalmente de preços de produtos, utilizados em plataformas de comparadores de preço para e-commerces. Não obstante, muitos jornalistas têm feito uso da raspagem de dados para extrair mais informações, de maneira a corroborar com seus textos e reportagens (ADRIOLO 2009).

Data Scraping, ou Raspagem de Dados, é a ciência de extração dos dados de um determinado ambiente, e inserção em outro ambiente, geralmente existe um processo de tratamento dos dados, tornando-os simples e maleáveis (ADRIOLO, 2009). Nesta acepção, torna-se necessário reconhecer o padrão da ambiente fonte dos dados, e através de um script de programação ou software, a raspagem se torna possível. Python é uma das linguagens que mais se destaca na raspagem de dados. Existem inúmeras APIs para raspagem de dados, como BeautifulSoup, PDFQuery, PyPDF2, entre outas.

### 2.1.3.1 PDFMiner

O PDFMiner é uma ferramaneta de extração de informação de documentos PDF. Diferente de outras ferramentas semelhantes, é focada inteiramente na raspagem e análise de dados em texto. PDFMiner permite obter a localização exata do texto em uma página, assim como outras informações como fontes ou linhas. Além disso, é possível converter o PDF em um formato de HTML, por exemplo, facilitando a interpretação em raspagem de dados (PDFMINER, 2017).

### 2.1.3.2 PyPDF2

O PyPDF2 é uma caixa de ferramentas totalmente feita em Python, surgiu a partir do projeto pyPDF em 2005, focado na manipulação de documentos, recorte de páginas, criptografia e descriptografia de documentos. PyPDF2, foi lançado em 2011 com o objetivo de ler todos os tipos de PDF, no entanto o projeto se manteve apenas nos arquivos PDF de texto, podendo criar arquivos PDF novos e raspar dados de arquivos PDF existentes. (PYPDF, 2019)

## 2.2. Tecnologias Semelhantes

O principal objetivo desse projeto de graduação foi permitir que pessoas possam concentrar seus estudos para o vestibular de maneira mais fácil, independentemente do local em que estejam, para isso a alimentação de um banco de dados se fez necessário, permitindo a mobilidade e agilidade nos estudos. Existiram aplicativos principais que foram usados como referências para o desenvolvimento de projeto, ambos fazem uso de bancos de dados de questões e são maneiras de transparecer esses bancos de questões. Durante a pesquisa desse projeto, encontramos alguns exemplos e banco de questões disponibilizados através de uma plataforma com custo mensal.

### 2.2.1. Super Professor – Banco de Questões

Super Professor é uma plataforma Web que disponibiliza mais de 150 mil questões de diferentes assuntos, cerca de 5 mil questões preparatórias para o ENEM (SPRWEB, 2020), a plataforma disponibiliza ainda dois planos de consumo, são eles:

Plano Escola: disponibiliza para todos os professores, número de usuários ilimitados e acessos simultâneos ilimitados, da escola a ferramenta para elaboração de avaliações de maneira mais rápida e com mais qualidade. Além disso, este plano concede acesso por IP específico além de espaço em disco para armazenamento de arquivos (provas) gerados. O custo é preço é obtido sob consulta.

Plano Professor: disponibiliza apenas um acesso, não é permitido acesso simultâneo, e contém apenas 250mb para armazenamento de arquivos. Os custos variam de R$127,00 até R$836,00

Além disso a plataforma contém um acesso exclusivo para alunos que podem acessar simulados lista de exercícios, consultar resultados através de relatórios e gráficos.

Figura 2 - Exemplo de Prova Gerada pelo Super Professor

Uma imagem contendo texto, livro

Descrição gerada automaticamente

[Fonte: SPRWEB (Adaptado, 2019)](http://fatecsjc-prd.azurewebsites.net/moodle/pluginfile.php/65111/mod_resource/content/0/Exemplo_de_Projeto_Android_de_Tempo_Real_com_Firebase.pdf)

Na figura acima é possível ver uma prova gerada pela plataforma Super Professor, são questões extraídas de vestibulares como UFMG, ITA, FUVEST etc. As questões contêm o ano em que estiveram nos vestibulares, além de possuir figuras, textos e fontes. (SPRWEB, 2019)

<https://www.sprweb.com.br/mod_superpro/index.php>

### 2.2.1. Só Exercícios -Banco de Questões

De semelhante modo, Só Exercícios é um banco de questões que contém cerca de 14 mil questões, oferece simulados dinâmicos, com questões aleatórias extraídas de provas anteriores do vestibular em foco; busca de questões, permitindo pesquisa por vestibular, ano, disciplinas e assuntos específicos; estatísticas de desempenho, através de gráficos e relatórios; módulos de estudo, através de algoritmos e frequência analisa quais assuntos foram foco nas ultimas edições de determinado vestibular. (SOEXERCICIOS, 2020).

Até a publicação deste documento, a plataforma Só Exercícios ainda não continha questões do Vestibular da FATEC.

<https://soexercicios.com.br/>

### 2.2.1. Aplicativo Perguntados 1 e 2

/\*TODO: falar de vantagens e desvantagens \*/

Perguntados é um jogo para dispositivos móveis, lançado em outubro de 2013, foi desenvolvido pela Etermax, está disponível para Android, iOS e Aplicação para Facebook (PERGUNTADOS, 2019). O objetivo do jogo é conquistar seis personagens da roleta, esses personagens representam categorias das perguntas, são elas: artes, ciências, esportes, entretenimento, geografia e história. O primeiro que conquistar os 6 personagens ganha a partida, cada partida possui até 25 rodadas (PERGUNTADOS, 2019).

Esse aplicativo serviu como uma referência de interface e descontração, um aplicativo que remete a estudos dificilmente cai no gosto das pessoas, mas esse aplicativo se tornou bem popular ao longo dos anos, Perguntados 1 já passou da casa dos 100.000.000 de downloads, enquanto Perguntados 2 possui mais de 10.000.000 de downloads (PLAY STORE, 2019). Além disso, a interface amigável e descontraída com certeza aperfeiçoou a experiencia do cliente, e serviu de inspiração para o desenvolvimento desse projeto.

### 2.2.2. Simulado Detran-SP

/\*TODO: falar de vantagens e desvantagens \*/

O Simulado do Detran-SP, foi uma inspiração funcional do projeto, foi desenvolvido com um propósito específico, auxiliar estudantes do curso teórico em busca de sua CNH. O curso teórico obrigatório para se obter a CNH é composto por 45 horas de aula, em seguida o estudante deve aguardar duas semanas até realizar a prova teórica oficial, nesse período o aluno é encorajado a continuar estudando em casa e esse aplicativo de auxiliado muitas pessoas, o aplicativo já conta com mais de um milhão de downloads (PLAY STORE, 2019).

A prova teórica do Detran é composta por 30 questões de múltipla escolha que devem ser respondidas em 40 minutos. O banco de dados do Detran, possui cerca de 600 questões de acesso público e transparente a todas as pessoas que se interessarem, semelhante ao caso da FATEC, que também disponibiliza vestibulares anteriores junto com os respectivos gabaritos, porém, apenas através de arquivos PDF, não existe um banco de dados especifico com uma API aberta.

O aplicativo Simulado do Detran-SP serviu principalmente como referência de proposito, cumprindo o papel de auxiliar pessoas interessadas em aprender mais e se acostumar com o modelo do exame teórico. Nesse sentido, o QuizFATEC foi desenvolvido buscando auxiliar vestibulando interessados em aprender mais e identificar matérias que necessitam de uma maior atenção no momento dos estudos.

## 2.3. Vestibulares e o Fator Psicológico

Cada vez mais populares, os vestibulares são a principal entrada para cursos técnicos, superiores e até para bolsas de estudo. Pensando no ponto de vista microssocial, o jovem e sua família, sofrem uma angústia ou ansiedade no período que antecede os vestibulares, entretanto o vestibular é a única maneira de ingressar em inúmeras universidades e faculdades, como a FATEC por exemplo. Torna-se então uma etapa de todo estudante ou jovem que deseja continuar os estudos e muitos deles não se sentem preparados para realizar vestibulares, mais precisamente 60% dos jovens (D’VILA, 2003). Questionados a respeito das categorias que se assemelhavam ao seu estado de espírito durante a execução do vestibular, cerca de 20,1% se sentem emocionalmente frágeis, outros 16,6% se sentem fisicamente frágeis e outro grupo, 20,1% se sentem despreparados em relação aos estudos (D’VILA, 2003). Pensando assim, estar preparado para executar um vestibular é de sua importância e uma das principais maneiras de se preparar é a ambientação com o exame e saber as áreas de conhecimento abrangidas pelo vestibular.

## 2.4. Banco de Dados NoSQL

Os bancos de dados relacionais são predominantes no mercado, mas com o passar dos anos e avanço da tecnologia e a interação humana com os sistemas computadorizados se fez necessário evoluir e agregar outras maneiras de se armazenar os dados, com isso surgiram armazenamento de dados orientados a objetos e XML. O fato é que com o passar dos anos o conceito de um banco de dados genérico de proposito variável multifacetado se torna insustentável (NOSQL, 2019). Através dessa abertura, o movimento NoSQL começou a tomar força, com a intenção inicial de criar um banco de dados moderno e escalável para aplicações Web (NOSQL, 2019). O termo NoSQL vem do inglês “Not Only SQL”, em tradução literal, não somente SQL. Podendo ainda ser descrito como “NoACID” (FORBES, 2010).

### 2.4.1. JSON

JSON é um dado intercambiável de armazenamento leve, é lido naturalmente por humanos e facilmente convertido e interpretado por máquinas (JSON, 2019). Um objeto JSON é comumente formado por um conjunto chave-valor, mas pode possuir um conjunto de chave-valor estruturado numa lista, formando um Array, principal estrutura para armazenar os dados dentro do Firebase.

Figura 3 - Estrutura básica JSON



[Fonte: JSON (2019)](http://fatecsjc-prd.azurewebsites.net/moodle/pluginfile.php/65111/mod_resource/content/0/Exemplo_de_Projeto_Android_de_Tempo_Real_com_Firebase.pdf)

Na imagem abaixo, consta um exemplo de JSON estruturado num conjunto de chave-valor inserido numa lista contendo outros conjuntos de chave-valor, os textos em azul são as chaves, como “Questão01”, “Questão”, “(A)”, “(B)”, “(C)”, “(D)”, “(E)”, em tom cobre estão os valores das respectivas chaves, importante atentar ao fato de “Questão01” equivale a um novo conjunto de chaves-valor, sendo possível então umas estrutura de estruturas de chave-valor.

Figura 4 - Exemplo de JSON estruturado



Fonte: Autor (2019)

A maior vantagem do JSON em relação aos arquivos XML, é a facilidade de leitura, sendo possível uma leitura fácil por humanos e também pela máquina (JSON, 2019)

### 2.4.2. MongoDB

O MongoDB é um banco de dados de propósito geral, baseado em documentos, é um banco de dados distribuído, trata-se de um banco de dados para era moderna de soluções em nuvem (MONGODB, 2019).

## 2.5. Webview para Dispositivos Móveis – Ionic e Cordova Framework

Assim como a tecnologia evolui em torno das redes de computadores e banco de dados, o mesmo ocorreu com o desenvolvimento de páginas para internet e mais recentemente para dispositivos móveis, uma das tecnologias que tem despontado é uso de Webview, consiste em desenvolvimento de páginas para computador e também para dispositivos móveis como tablets e smartphones, isso é possível através de um compilador que converte essa webview em apk, formato padrão de aplicativos android, instaláveis pela própria loja de aplicativos nativa dos smartphones android.

O Cordova é um framework open source, capaz de converter a webview em uma instalável, tanto para dispositivos Android como sistemas iOS, conforme diagrama mostrada abaixo:

Figura 5 - Esquema tático aplicação Cordova

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Adaptado Ionic Forum (2019)

Considerando o diagrama mostrado acima, existem dois grandes grupos Cordova Application e Mobile OS. Considerando o grupo alaranjado definido como Cordova Application, temos três grupos menores definidos como:

Web App: consiste no conjunto Web da aplicação, composto por frameworks, linguagens, configurações e estruturas referentes a uma página de site. Aqui temos um site coeso.

Cordova Plugins: consiste na caixa de ferramentas do Cordova, é a camada intermediária entre a Webview e o conjunto de acessórios do celular, tanto de hardware como acelerômetro, câmera, armazenamento, quanto de software como contatos, outros aplicativos. HTML Rendering Engine: intersecção entre o sistema operacional do dispositivo móvel, APIs HTML e APIs Cordova.

Mobile OS, em destaque pelo grupo azul, representa o sistema operacional do dispositivo móvel, é o gerenciador de hardware e software do celular.

Ionic Framework é um conjunto de ferramentas open source, criado com o objetivo de desenvolver aplicativos Web e para dispositivos móveis usando tecnologias Web, como HTML, JavaScript e CSS (IONIC, 2019).

O Framework Ionic se mostrou o mais vantajoso, principalmente pela possibilidade de acesso nativo para Android e iOS através de uma única Webview, construída pelo conjunto de ferramentas do Ionic, os aplicativos desenvolvidos usando Ionic são suportados pelo Android, a partir da versão 4.4, e o iOS a partir da versão 10.

# 3. DESENVOLVIMENTO

Este capítulo discorrerá a respeito do desenvolvimento do sistema, desde o Banco de Dados NoSQL até o desenvolvimento da aplicação móvel com a leitura das questões raspadas. O projeto foi desenvolvido em etapas, conforme:

1 – Raspagem das Provas e Gabaritos com Python

2 – Estruturação e carga do Banco de Dados NoSQL

3 – Desenvolvimento do Backend com Flask

4 – Desenvolvimento do APP com WebView

## 3.1. Arquitetura Global do Sistema

A figura abaixo demonstra a arquitetura geral da Aplicação escolhida para mitigar as dificuldades na preparação e estudo pré-vestibular da FATEC.

Figura 6 - Arquitetura Geral do Projeto

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

O desenvolvimento do projeto se dará através da extração das questões e respostas disponibilizadas em arquivos PDFs, pelo próprio Centro Paula Souza, isso será feito através de um Script Python usando PDFMiner e PyPDF2, assistidos pelo Jupyter Notebook. Depois de raspar as questões e repostas, os dados serão carregados em um banco de dados NoSQL MongoDB, numa estrutura de acesso em tempo real, através de uma solução cluster em nuvem. Com o banco estruturado em arquitetura de árvore contendo o texto da questão, alternativas e reposta, um aplicativo será desenvolvido usando a estrutura de WebView usando Ionic, através de Angular, CSS e HTML 5. A codificação, lógica e aplicação de serviços e métodos será feito através de TypeScript, dessa maneira o projeto será multiplataforma, com desenvolvimento simultâneo de Web Site e também aplicação de Smartphones(IONIC, 2019). A compilação da aplicação será feita através do Apache Cordova Framework.

## 3.2 Scrapper

Scrapper, ou raspador em tradução literal, é um script python redigido no Jupyter Notebook, uma aplicação web e *open-source*, aberta para padrões e serviços interativos computacionais para uma gama de linguagens de programação (JUPYTER NOTEBOOK, 2019). A plataforma do Jupyter é amplamente utilizada para limpeza e tratamento de dados, assim como manipulação de grandes *dataframes* de dados, possibilitando sua manipulação mais simplificada pelas células individuais, além da facilidade de documentação e comentários. Os scripts criados no Jupyter são facilmente lidos e manuseados através da sua construção modular, a imagem abaixo ilustra essa composição de títulos e blocos de código facilitando a compreensão.

O Scrapper se mostrou uma excelente ferramenta na extração das provas, reduzindo um trabalho manual que tomaria vários dias, para menos de uma hora, isso foi possível devido a substituição da força humana por um script python que carregou cerca de 350 questões de oito provas diferentes num tempo inferior a uma hora. Entanto a raspagem automática mostrou algumas limitações, a impossibilidade de extrair imagens facilmente, grande parte delas contém o link da imagem e isso foi tratado, minimizando o impacto da raspagem das imagens. Outra falha mapeada foi a impossibilidade de separar textos válidos para múltiplas questões, sua raspagem se mostrou complexa por falta de padrão estabelecido e por variar muito entre quantidade de questões apontadas para o texto. Além desses pontos, existiram ainda algumas questões que por motivos incertos não foram raspadas, provavelmente por falta de identificação clara.

A única necessidade prévia para utilização dos Scrapper é que os arquivos referentes a prova e gabarito, precisam estar no mesmo diretório físico que o notebook jupyter Scrapper.

Figura 7 - Composição Geral de um Notebook Jupyter

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

Os arquivos, formato .ipynb utilizados no Jupyter são chamados de Notebooks, bloco de notas em tradução literal (Autor, 2019). Ao longo dessa seção serão destrinchados cada um dos blocos do Scrapper.

### 3.2.1 Scrapper – Imports e Declarações Globais

A primeira célula do Scrapper trata das importações de bibliotecas utilizadas ao longo do script, são elas:

Tabela 1 - Bibliotecas Python

|  |  |  |
| --- | --- | --- |
| Biblioteca | Versão | Função |
| pymongo[srv] | 3.9.0 | Biblioteca responsável pela administração da conexão do código python com o mongoDB atlas cluster. |
| pdfminer | 2014032018 | Dessa biblioteca são importadas várias ferramentas para um conjunto de ações como iterações no PDF, navegação de páginas, pdfpage, responsável pela extração do texto de cada página etc. |
| Io | 3.7.5rc1 | Biblioteca padrão do Python, de acesso a dispositivos de IO, entrada e saída, utilizado para acessar a função StringIO que permite a leitura em fluxo na memória para um texto. |
| PyPDF2 | 1.26.0 | Biblioteca redigida para facilitar a leitura de PDF através do Python, como teve alguns problemas de codificação, foi utilizada apenas para acessar o número de páginas de cada prova e para raspar os gabaritos, onde a codificação e formatação não apresentaram problemas. |

Fonte: Autor (2019)

Todas os *imports* necessários e as versões utilizadas estão disponíveis no arquivo requeriements.txt, são facilmente instaláveis através do comando *pip*, gerenciador de pacotes do python (PIP, 2019).

Quadro 1 - Scrapper - Imports e Declarações Globais

|  |  |
| --- | --- |
| 1 | import pymongo |
| 2 | from pdfminer.pdfinterp import PDFResourceManager, PDFPageInterpreter |
| 3 | from pdfminer.pdfpage import PDFPage |
| 4 | from pdfminer.converter import TextConverter |
| 5 | from pdfminer.layout import LAParams |
| 6 | from io import StringIO |
| 7 | from PyPDF2 import PdfFileReader |
| 8 |  |
| 9 | prova = "PROVA\_2\_2018" |
| 10 | qtdInseridas = 0 |
| 11 | gabarito = prova.replace("PROVA", "GABARITO") |
| 12 | temas = ['MULTIDISCIPLINAR','RACIOCÍNIO LÓGICO','HISTÓRIA','QUÍMICA','INGLÊS','MATEMÁTICA','FÍSICA','GEOGRAFIA', 'BIOLOGIA','PORTUGUÊS','REDAÇÃO','RASCUNHO', 'RAC', 'LÓGICO'] |
| 13 | respgabarito = {} |
|  |  |

Fonte: Autor (2019)

Essa célula também instancia alguns objetos globais como a variável que carrega a edição da prova que será raspada pelo script, assim como a edição do gabarito correspondente. O vetor temas foi criado para limpar os textos referentes aos temas das provas e gabaritos, esses temas são mais facilmente acessados através do número da questão, que segue o mesmo padrão desde a primeira edição da prova da FATEC. O vetor respgabarito armazena as questões globalmente de maneira que possam ser acessadas dentro de diferentes células funções.

### 3.2.2 Scrapper – Inserção de Dicionários no MongoDB

Essa célula é dedicada a função de inserção no banco e dados NoSQL, nesse projeto foi utilizado o MongoDB, desenvolvido em Python, facilitando a implementação conjunta com o Scrapper. A conexão é criada através da biblioteca pymongo, utilizando da função MongoClient, que permite instanciar uma interface de acesso aos recursos do banco de dados alocado no cluster em nuvem do Atlas Cluster, dentro dessa interface é possível acessar o banco QuizFATEC e a *collection* provas, onde todas as questões ficam armazenadas. A figura abaixo demonstra o código python destrinchado.

Quadro 2 - Scrapper - Inserção de Dicionários no MongoDB

|  |  |
| --- | --- |
| 1 | def insert\_question(json): |
| 2 | global qtdInseridas |
| 3 | qtdInseridas += 1 |
| 4 | client = pymongo.MongoClient("mongodb+srv://admin:admin@quizfatec-xl7tb.mongodb.net/test?retryWrites=true&w=majority") |
| 5 | dbProvas = client["QuizFATEC"] |
| 6 | colProvas = dbProvas["provas"] |
| 7 | try: |
| 8 | colProvas.update\_one({"\_id" : json['\_id']}, {"$set": json}, upsert=True) |
| 9 | print("Questão "+json['\_id']+" Inserida com Sucesso") |
| 10 | except: |
| 11 | print("Não foi possivel inserir a questão") |
| 12 | client.close() |
| 13 | def insert\_question(json): |
|  |  |

Fonte: Autor (2019)

A função recebe como parâmetro um dicionário de textos montado pela função *text\_to\_json\_question*, entretanto a variável colProvas que administra a *collection* do MongoDB interpreta esse dicionário de *strings* do python como um JSON estruturado. A exceção no momento da inserção pode ser acionada por motivos como falha na conexão com o cluster na nuvem devido a falta de conexão ou IP bloqueado, um outro motivo relevante é que no MongoDB todos os objetos inseridos nas *collections* possuem um \_id gerado automaticamente, mas que podem ser alterados, como é o caso nesse projeto, onde os \_id de cada objeto foram alterados de maneira a possuírem um padrão referente a cada prova e edição, facilitando que a leitura futura por uma aplicação cliente, como é o caso do aplicativo.

### 3.2.3 Scrapper – Raspagem do Gabarito

Os gabaritos se mostraram muito mais simples de serem raspados devido ao padrão estabelecido, a figura abaixo demonstra o comparativo entre os gabaritos do primeiro semestre de 2010 e segundo semestre de 2019, uma diferença de quase uma década entre edições mas que não refletiu em mudanças significativas de padrão.

Figura 8 - Comparativo de Gabaritos



Fonte: Autor (2019)

Conforme a figura acima, é possível identificar que a estrutura do gabarito se manteve inalterada ao longo dos vestibulares. Atributos como estrutura bi colunar, número de questões padrão de 54, ordem dos temas e gabarito similar facilitaram a raspagem permitindo que a função responsável pela raspagem do gabarito fosse mais simples e 100% efetiva para todos os vestibulares, a figura abaixo demonstra a função *awnser\_extractor* mencionada.

Quadro 3 - Scrapper - Raspagem de Gabarito

|  |  |
| --- | --- |
| 1 | def answer\_extractor(path): |
| 2 | arrumarPadrao = "QUESTÃO ALTERNATIVADISCIPLINA QUESTÃOALTERNATIVA DISCIPLINA" |
| 3 | padrao = "QUESTÃO ALTERNATIVA DISCIPLINA QUESTÃO ALTERNATIVA DISCIPLINA" |
| 4 |  |
| 5 | with open(path, 'rb') as f: |
| 6 | gabarito=PdfFileReader(f) |
| 7 | text = gabarito.getPage(gabarito.numPages-1).extractText().upper().replace('\n','').replace(' ',' ').replace(arrumarPadrao,padrao) |
| 8 |  |
| 9 | #tratamento de execção na prova 2 de 2019 |
| 10 | if prova == 'PROVA\_2\_2019': |
| 11 | text = text.replace('34 MATEMÁTICA', '34 ANULADA MATEMÁTICA') |
| 12 | for tema in temas: |
| 13 | text = text.replace(tema,'') |
| 14 |  |
| 15 | lstSemCabecalho = text[text.find(padrao)+len(padrao):].split(' ') |
| 16 |  |
| 17 | questao = [] |
| 18 | resposta = [] |
| 19 | cont = 0 |
| 20 |  |
| 21 | for el in lstSemCabecalho: |
| 22 | if el!='' and cont%2==0: |
| 23 | questao.append(el) |
| 24 | cont+=1 |
| 25 | elif el!='' and cont%2 ==1: |
| 26 | resposta.append(el) |
| 27 | cont+=1 |
| 28 |  |
| 29 | for i in range(0,54): |
| 30 | respgabarito.update({int(questao[i]):resposta[i]}) |
|  |  |

Fonte: Autor (2019)

A função *answer\_extractor* utiliza do PyPDF2 para transformar o arquivo PDF num texto único excluindo imagens do cabeçalho e transpondo a tabela para um texto simples tabulado, através da função padrão de *strings* do python, *replace* é possível retirar textos desnecessários como os temas das questões, espaços múltiplos e as linhas puladas ao longo do texto. Todas os gabaritos da FATEC possuem um texto padrão composto pela sequência de palavras: QUESTÃO ALTERNATIVA DISCIPLINA QUESTÃO ALTERNATIVA DISCIPLICA, apresentado repetido devido as duas colunas usadas na composição do gabarito, conforme demonstrado na figura 9. Essa sequência de palavras repetidas permitiu a leitura consistente da resposta correta para cada questão excluindo o cabeçalho e as disciplinas mantendo apenas os numerais e respectivas respostas de cada questão. Finalmente, o dicionário de respostas foi composto utilizando o número da questão como chave, permitindo seu acesso para pesquisa ao longo da raspagem das questões.

### 3.2.4 Scrapper – Retirada de Texto do PDF de Prova

A função *pdf\_to\_text* demonstrada abaixo, recebe como parâmetro um endereço de diretório onde se encontra a prova e a edição que será raspada.

Quadro 4 - Scrapper - PDF\_TO\_TEXT

|  |  |
| --- | --- |
| 1 | def pdf\_to\_text(pdfname): |
| 2 | rsrcmgr = PDFResourceManager() |
| 3 | sio = StringIO() |
| 4 | device = TextConverter(rsrcmgr, sio, codec='utf-8', laparams=LAParams()) |
| 5 | interpreter = PDFPageInterpreter(rsrcmgr, device) |
| 6 | with open(pdfname, 'rb') as fp: |
| 7 | for page in PDFPage.get\_pages(fp): |
| 8 | interpreter.process\_page(page) |
| 9 | text = sio.getvalue() |
| 10 | qtdPages = PdfFileReader(fp) |
| 11 | #Remove os temas, são mais facilmente encontrados pelo numero das questões |
| 12 | for tema in temas: |
| 13 | text = text.replace(tema,'') |
| 14 | #Tira o espaçamento de linhas, facilitando o tratamento do texto |
| 15 | text = text.replace("\n","") |
| 16 | #Tratamento de ocorrencias de nomenclatura do vestibular |
| 17 | for i in range(1, qtdPages.numPages): |
| 18 | text = text.replace(str(i)+"VESTIBULAR "+pdfname[13]+"o SEM/"+pdfname[15:19]+" • FATEC ","") |
| 19 | text = text.replace(str(i)+" VESTIBULAR "+pdfname[13]+"o SEM/"+pdfname[15:19]+" • FATEC ", "") |
| 20 | text = text.replace("VESTIBULAR "+pdfname[13]+"o SEM/1"+pdfname[15:19]+" • FATEC "+str(i),"") |
| 21 | text = text.replace("VESTIBULAR "+pdfname[13]+"o SEM/"+pdfname[15:19]+" • FATEC "+str(i),"") |
| 22 | #Cria as chaves de Identificação de Questão |
| 23 | for i in range(0,54): |
| 24 | if i < 10: |
| 25 | n = 'Questão0' + str(i) |
| 26 | ni= '0'+str(i)+'Questão' |
| 27 | else: |
| 28 | n = 'Questão' + str(i) |
| 29 | ni = str(i) + 'Questão' |
| 30 | text = text.replace(n, '[-Chave-]'+n).replace(ni,'[-Chave-]'+n).replace('Leia o texto ', '[-Chave-]Leia o texto ').replace('Leia os textos ', '[-Chave-]Leia os textos ') |
| 31 | #Cria as chaves de Idenfitificação das Alternativas |
| 32 | text = text.replace('(A)', '[-ChaveA-](A)').replace('(B)', '[-ChaveA-](B)').replace('(C)', '[-ChaveA-](C)').replace('(D)', '[-ChaveA-](D)').replace('(E)', '[-ChaveA-](E)') |
| 33 | fp.close() |
| 34 | device.close() |
| 35 | sio.close() |
| 36 | #Comeco de tratamento de textos multiquestões |
| 37 | #text = text.replace("Leia o texto ", "[-ChaveTexto-]Leia o texto ").replace("Leia os textos", "[-ChaveTextos-]Leia os textos") |
| 38 | return text |
|  |  |

Fonte: Autor (2019)

Essa função objetiva interpretar os PDFs e concatená-los em um bloco de texto de maneira a facilitar toda e qualquer manipulação, para isso foi utilizado a biblioteca PDFMiner, através do PDFResourceManager e PDFPageInterpreter foi possível manipular o PDF interagindo ao longo das páginas e permitindo alguns tratamentos que se seguiram. Uma vez que a variável text recebeu o texto concatenado de todas as páginas os seguintes tratamentos ocorreram, em primeiro momento foram retirados os textos dos temas, semelhante ao tratamento aplicado na raspagem de gabaritos, ocorrências como MATEMÁTICA, RACICIONIO LÓGICO, QUÍMICA, REDAÇÃO etc. foram removidos do bloco de texto. Em segundo as quebras de linhas foram removidas, facilitando o armazenamentos dos textos assim como sua manipulação, em terceiro momento os textos padrão de rodapé são removidos, todas as páginas das provas contem uma identificação referente a cada edição do vestibular e número da página, um laço de repetição itera da primeira até a última página buscando no texto e removendo-o.

Como dito anteriormente, todas os vestibulares da FATEC contém 54 questões, dessa maneira um laço de repetição substitui todas as ocorrências de identificação de questão para uma chave que será usada para quebrar o texto em um vetor, facilitando a iteração, na prática todas as ocorrências do texto Questão somando ao numeral identificador da questão são substituídos por uma *tag* que será usada na quebra do texto para vetor. A *tag* utilizada foi criada de maneira a garantir sua não ocorrência em nenhum vestibular, dessa maneira a chave de questões foi definida como [-Chave-].

Finalmente acontece o processo de substituição das alternativas identificadas pelo caractere do alfabeto de A até E, sempre cercadas por parêntesis, permitindo sua identificação, objetivando a facilidade da interpretação também foi inserida uma *tag* única construída seguindo o padrão da *tag* usada nas questões. Ao final do código é retornado o bloco de texto concatenado e tratado.

### 3.2.5 Scrapper – Retirada das Questões do Texto da Prova

Essa função foi a mais desafiadora de todo o projeto, a complexidade de se estabelecer um padrão de raspagem dentro de um vestibular que apresenta em média 24 páginas, 54 questões de 10 temas de conhecimento diferentes, com dezenas de imagens, textos e links para consulta. Embora as provas tenham alguns padrões definidos como a nomenclatura e numerologia padrão, existem outras ocorrências fora de padrão como referências de acesso a links são referenciados como “Acesso em 15.03.2019” ou “Acesso em 17/05/2010” e até mesmo “Acesso em 15.03.09”, outra dificuldade são os textos para múltiplas questões que costumam ser referenciados antes da primeira questão do intervalo mas variam em quantidade, podem ser aplicados para duas ou mais questões, em certas ocorrências existem também imagens.

Considerando todas essas dificuldades, ainda assim foi possível raspar 89% das questões extraindo algum texto para questão, isso significa que apenas 11% das questões tiveram textos ausentes, considerando as oito edições de 2016 até 2019. Apenas 8% das alternativas apresentaram erros na raspagem. Uma curiosidade a ser levantada é que 26% das questões raspadas tiveram a alternativa E como resposta correta, a alternativa A apresentou-se como resposta correta em apenas 12% das questões.

A figura abaixo contém o código de transformação do bloco concatenado em questões numa estrutura de dicionário que pode ser facilmente convertida em um objeto JSON.

Quadro 5 - Scrapper - Text to JSON Question

|  |  |
| --- | --- |
| 1 | def text\_to\_json\_question(textProva): |
| 2 | vetorText = textProva.split('[-Chave-]') |
| 3 |  |
| 4 | for elemento in vetorText: |
| 5 | if elemento[:7] == "Questão" and elemento[7:8] != "": |
| 6 |  |
| 7 | nQuestao = int(elemento[7:9]) |
| 8 |  |
| 9 | textoQst = elemento[9:] |
| 10 |  |
| 11 | arrayQuestoes = elemento[9:].split('[-ChaveA-]') |
| 12 |  |
| 13 | #Busca de Links na Questão, poderão ser usados na webview |
| 14 | preLinks = elemento[9:].split('<') |
| 15 | arrayLinks = [] |
| 16 | for el in preLinks: |
| 17 | if el[:4] == "http" and el.find('>') != -1: |
| 18 | arrayLinks.append(""+ el[:el.find('>')] +"") |
| 19 |  |
| 20 | dicionarioQuestao = {} |
| 21 | dicionarioQuestao.update({'\_id': prova + "\_QUESTAO\_" + str(nQuestao)}) |
| 22 | dicionarioQuestao.update({'prova':prova}) |
| 23 | dicionarioQuestao.update({'numero': nQuestao}) |
| 24 |  |
| 25 | if nQuestao >= 1 and nQuestao <= 9: |
| 26 | dicionarioQuestao.update({'tema': 'MULTIDISCIPLINAR'}) |
| 27 | elif nQuestao >= 10 and nQuestao <= 14: |
| 28 | dicionarioQuestao.update({'tema': 'RACIOCÍNIO LÓGICO'}) |
| 29 | elif nQuestao >= 15 and nQuestao <= 19: |
| 30 | dicionarioQuestao.update({'tema': 'HISTÓRIA'}) |
| 31 | elif nQuestao >= 20 and nQuestao <= 24: |
| 32 | dicionarioQuestao.update({'tema': 'QUÍMICA'}) |
| 33 | elif nQuestao >= 25 and nQuestao <= 29: |
| 34 | dicionarioQuestao.update({'tema': 'INGLÊS'}) |
| 35 | elif nQuestao >= 30 and nQuestao <= 34: |
| 36 | dicionarioQuestao.update({'tema': 'MATEMÁTICA'}) |
| 37 | elif nQuestao >= 35 and nQuestao <= 39: |
| 38 | dicionarioQuestao.update({'tema': 'FÍSICA'}) |
| 39 | elif nQuestao >= 40 and nQuestao <= 44: |
| 40 | dicionarioQuestao.update({'tema': 'GEOGRAFIA'}) |
| 41 | elif nQuestao >= 45 and nQuestao <= 49: |
| 42 | dicionarioQuestao.update({'tema': 'BIOLOGIA'}) |
| 43 | elif nQuestao >= 50 and nQuestao <= 54: |
| 44 | dicionarioQuestao.update({'tema': 'PORTUGUÊS'}) |
| 45 | else: |
| 46 | dicionarioQuestao.update({'tema': 'INVÁLIDO'}) |
| 47 |  |
| 48 | try: |
| 49 | textoQst = arrayQuestoes[0] |
| 50 | except: |
| 51 | textoQst = 'Falha na Leitura da Questão' |
| 52 | dicionarioQuestao.update({'texto': textoQst}) |
| 52 |  |
| 53 | try: |
| 54 | alternativaA = arrayQuestoes[1].replace('(A)','').lstrip().rstrip() |
| 55 | except: |
| 56 | alternativaA = 'Falha na Leitura da Alternativa A' |
| 57 | dicionarioQuestao.update({'a': alternativaA}) |
| 76 | [...] |
| 77 | try: |
| 78 | alternativaE = arrayQuestoes[5].replace('(E)','').lstrip().rstrip() |
| 79 | except: |
| 80 | alternativaE = 'Falha na Leitura da Alternativa E' |
| 81 | dicionarioQuestao.update({'e': alternativaE}) |
| 82 |  |
| 83 | if len(arrayLinks) > 0: |
| 84 | dicionarioQuestao.update({'links': arrayLinks}) |
| 85 |  |
| 86 | dicionarioQuestao.update({'resposta': respgabarito.get(nQuestao)}) |
| 87 | if dicionarioQuestao['resposta'] != 'ANULADA': |
| 88 | insert\_question(dicionarioQuestao) |
| 89 | def text\_to\_json\_question(textProva): |
|  |  |

Fonte: Autor (2019)

Logo na primeira linha da função já ocorre uma quebra em vetor, utilizando como chave para o *split* a *tag* [-Chave-]. Em seguida um laço de repetição itera ao longo dos elementos desse vetor cruzando os primeiros caracteres do texto de maneira a validar a ocorrência do conjunto identificador de Questão e o numeral correspondente, caso o elemento corresponda a uma questão válida, a sequência de tratamentos se inicia. Num primeiro momento é obtido o número da questão, que será utilizado na pesquisa das respostas do dicionário de respostas raspadas do gabarito. Em seguida se inicia o processo de raspagem dos links. Os links foram raspados dessa maneira com o objetivo de tratar mais facilmente na aplicação cliente, será possível disponibilizar o link para consulta do próprio usuário da aplicação. Seguindo pelo código, são instanciados alguns elementos do dicionário como questão, prova e o \_id. O MongoDB como descrito anteriormente, é um banco de dados NoSQL de *schema* flexível, onde o foco da performance é a leitura, dessa maneira inserir esses dados em campos no dicionário facilitam a manipulação e pesquisa tanto no banco de dados em si, como na aplicação cliente.

Os temas de conhecimento são inseridos através do número da questão, respeitando o intervalo estabelecido no padrão das edições que se seguiram após 2016. O texto da questão é obtido após a quebra das *tags* [-ChaveA-], o primeiro elemento do vetor é o próprio texto da questão, na sequência as alternativas como seus textos são obtidos dentro de um tratamento de exceção, que caso acionado invalida o texto da alternativa, dessa maneira será possível tratar essa questão individualmente e manualmente no futuro.

Em seguida a lista de links raspadas é inserida no dicionário, nesse momento o *schema* flexível do MongoDB se mostra vantajoso, permitindo que caso a questão não possua links, não será carregado no banco uma lista vazia, consequentemente acarretando redução de armazenamento do banco de dados e agilizando a interpretação na aplicação cliente.

Finalmente a reposta, que foi raspada e armazenada durante a função *awnser\_extractor* é pesquisada através da função *get*, nativa de dicionários python, completando a estrutura do dicionário, ao final do código uma estrutura condicional impede que questões anuladas sejam inseridas no banco de dados.

### 3.2.6 Scrapper – Função Principal

*Main function*, ou aplicação principal, em tradução livre, é a função que o código python interpreta como início da execução, assim sendo, nessa função é estabelecido o *path* composto pela edição do vestibular que será raspado concatenado com o diretório onde o arquivo está salvo. Em seguida a variável *pathGabarito* é instanciada de semelhante modo. A figura abaixo demonstra a função principal do Scrapper.

Quadro 6 - Scrapper Main Function

|  |  |
| --- | --- |
| 1 | if \_\_name\_\_ == '\_\_main\_\_': |
| 2 | path ="Provas\\"+ prova + ".pdf" |
| 3 | pathGabarito ="Gabaritos\\"+ gabarito + ".pdf" |
| 4 | answer\_extractor(pathGabarito) |
| 5 | print("Inicio de Leitura") |
| 6 | text\_to\_json\_question(pdf\_to\_text(path)) |
| 7 | print("Fim de Leitura") |
|  |  |

Fonte: Autor (2019)

Após instanciar as variáveis de diretório, a função de raspagem de gabaritos é chamada, armazenando na variável global respgabarito. Uma mensagem de início de leitura foi inserida de maneira a facilmente identificar o início e fim da raspagem dos vestibulares. Em seguida são invocadas as funções de *pdf\_to\_text* e *text\_to\_json\_question*, funções responsáveis pela raspagem dos vestibulares e inserção no banco de dados.

## 3.3 Persistência dos dados através de MongoDB

Uma vez raspados os vestibulares e os gabaritos, a inserção no banco de dados aconteceu através da biblioteca python pymongo, utilizada na função *insert\_question*. Caso o banco de dados escolhido para essa aplicação fosse um banco relacional tradicional, seria necessária uma sequência de tabelas e relacionamentos para se obter uma o simples relacionamento de link e questão, questão e alternativa correta, prova e questões etc. Além disso o MongoDB proporciona automaticamente *dashboards*, conforme figura abaixo, com métricas calculadas em cima dos dados cadastrados, facilitando uma análise superficial, para identificação de dados ausentes ou qualquer anomalia na inserção dos dados.

Figura 9 - Dashboard QuizFATEC

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

Através desse *dashboard* é possível ver que as questões que possuem links, possuem em média apenas um link, mas variam de 1 a 3. A raspagem mais efetiva foi da segunda edição do vestibular da FATEC no ano de 2019, 18% das 397 questões raspadas, pertencem a edição do 2º semestre de 2019. Aa raspagem apontou que 28% das respostas eram a letra B e apenas 12% apresentaram a alternativa C como correta.

O *schema*, embora flexível, foi padronizado seguindo os atributos das questões, na figura abaixo é possível ver um documento com todos os dados cadastrados.

Figura 10 - Exemplo de Documento Inserido

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

Os campos \_id, Prova, Questao, Tema, (A), (B), (C), (D), (E), Resposta são obrigatórios e devem estar presentes em todas os documentos, o campo de Links aparece apenas nas questões em que links foram raspados, dessa maneira o espaço de armazenamento do banco de dados é otimizado.

## 3.4 *Backend* em Flask

O *backend* do projeto foi desenvolvido com Python, seguindo na mesma linguagem do script de rasgagem, para isso a biblioteca *Flask* foi escolhida, para criar a comunicação entre o banco de dados e a Webview. *Flask* é um leve WSGI, um framework de aplicações *web* (FLASK, 2019). Foram mapeadas ao todo 6 rotas, algumas requisições *get* e outras *post*, a figura abaixo retrata o código inicial do *backend*:

Quadro 7 - Backend: Conexão com Banco de Dados

|  |  |
| --- | --- |
| 1 | from flask import Flask, jsonify, request |
| 2 | from flask\_pymongo import PyMongo |
| 3 | from flask\_cors import CORS |
| 4 | import pymongo |
| 5 | import random |
| 6 |  |
| 7 | app = Flask(\_\_name\_\_) |
| 8 | CORS(app, resources=r'/QuizFATEC/\*') |
| 9 |  |
| 10 | client = pymongo.MongoClient("mongodb+srv://admin:admin@quizfatec-xl7tb.mongodb.net/test?retryWrites=true&w=majority") |
| 11 | dbProvas = client["QuizFATEC"] |
| 12 |  |
| 13 | @app.route('/QuizFATEC/Provas',methods=['GET']) |
| 14 | def get\_all\_questions(): |
| 15 | colProvas = dbProvas["provas"] |
| 16 | output = [] |
| 17 | for q in colProvas.find(): |
| 18 | output.append({'texto': q['texto'], 'a': q['a'], 'b': q['b'], 'c': q['c'], 'd': q['d'], 'e': q['e'], 'reposta': q['resposta'], 'tema': q['tema']}) |
| 19 |  |
| 20 | return jsonify({'resp' : output}) |
| 21 |  |
| 22 | @app.route('/QuizFATEC/Provas/<id>', methods=['GET']) |
| 23 | def get\_one\_question(id): |
| 24 | colProvas = dbProvas["provas"] |
| 25 | q = colProvas.find\_one({'\_id': id}) |
| 26 | if q is None : |
| 27 | output = 'a busca nao retornou resultados' |
| 28 | else: |
| 29 | output = {'texto': q['texto'], 'a': q['a'], 'b': q['b'], 'c': q['c'], 'd': q['d'], 'e': q['e'], 'reposta': q['resposta'], 'tema': q['tema']} |
| 30 |  |
| 31 | return jsonify(output) |
|  |  |

Fonte: Autor (2019)

No código acima, as primeiras linhas contêm a importação das bibliotecas necessárias para a execução do mesmo, em seguida uma aplicação *Flask* é instanciada, o CORS é configurado de maneira a receber todas as rotas contendo “/QUIZFATEC/” como parte da rota. Assim é possível fazer o acesso de diferentes interpretados, como Google Chrome, Safari, Mozilla Firefox etc. Com o CORS configurado, o *client* do MongoDB é instanciado, e a collection QuizFATEC é selecionada, permitindo a iteração com os registros durante as rotas.

Em seguida se inicia a imposição de rotas através de funções, a primeira rota instanciada é uma requisição *get* que retorna todas as questões cadastradas no banco de dados, essa rota foi usada principalmente para efeito de testes com o Postman, uma plataforma colaborativa para desenvolvimento de APIs (POSTMAN, 2019). A segunda rota mapeada é uma função para retornar apenas uma questão baseada em um ID passado, caso o não exista um documento no banco de dados correspondente a função uma mensagem padrão será retornada.

Quadro 8 - Backend: 3º e 4º Rotas

|  |  |
| --- | --- |
| 1 | @app.route('/QuizFATEC/Provas/Temas/<tema>', methods=['GET']) |
| 2 | def get\_random\_by\_theme(tema): |
| 3 | colProvas = dbProvas["provas"] |
| 4 | lst = [] |
| 5 | for q in colProvas.find({'tema': tema}): |
| 6 | try: |
| 7 | lst.append({'texto': q['texto'], 'a': q['a'], 'b': q['b'], 'c': q['c'], 'd': q['d'], 'e': q['e'], 'resposta': q['resposta'], 'prova': q['prova'], 'numero':q['numero'], 'tema': q['tema'], 'links': q['links']}) |
| 8 | except: |
| 9 | lst.append({'texto': q['texto'], 'a': q['a'], 'b': q['b'], 'c': q['c'], 'd': q['d'], 'e': q['e'], 'resposta': q['resposta'], 'prova': q['prova'], 'numero':q['numero'], 'tema': q['tema']}) |
| 10 |  |
| 11 | i = random.randint(0, len(lst)) |
| 12 | try: |
| 13 | output = lst[i] |
| 14 | except: |
| 15 | output = "A busca nao retornou resultados" |
| 16 |  |
| 17 | return jsonify(output) |
| 18 |  |
| 19 | @app.route('/QuizFATEC/Provas/Random/', methods=['GET']) |
| 20 |  |
| 21 | def get\_random(): |
| 22 | colProvas = dbProvas["provas"] |
| 23 |  |
| 24 | lst = [] |
| 25 | for q in colProvas.find(): |
| 26 | lst.append({'texto': q['texto'], 'a': q['a'], 'b': q['b'], 'c': q['c'], 'd': q['d'], 'e': q['e'], 'reposta': q['resposta'], 'prova': q['prova'], 'numero':q['numero'], 'tema': q['tema']}) |
| 27 |  |
| 28 | i = random.randint(0, len(lst)) |
| 29 | try: |
| 30 | output = lst[i] |
| 31 | except: |
| 32 | output = "A busca nao retornou resultados" |
| 33 | return jsonify({'resp' : output}) |
|  |  |

Fonte: Autor (2019)

A terceira rota é principal rota da aplicação, trata-se de uma rota para requisições *get* que, é utilizada pela Webview, para obter as questões aleatórias baseado no tema selecionado, a estrutura de exceção é utilizada para tratar as questões que não possuem links raspados pelo Scrapper. A quarta rota foi criada para propósitos empírico, é uma requisição *get* sem parâmetros que retorna um documento aleatório raspada pelo Scrapper, nessa rota os links foram desconsiderados.

Quadro 9 - Backend: 5º e 6º Rota

|  |  |
| --- | --- |
| 1 | @app.route('/QuizFATEC/Usuarios/', methods=['POST']) |
| 2 | def post\_user(): |
| 3 | colUsuarios = dbProvas["usuarios"] |
| 4 |  |
| 5 | id = request.json['\_id'] |
| 6 |  |
| 7 | colUsuarios.update\_one({"\_id" : id}, {"$set": request.json}, upsert=True) |
| 8 |  |
| 9 | q = colUsuarios.find\_one({'\_id': id}) |
| 10 | if q is None : |
| 11 | output = '' |
| 12 | else: |
| 13 | output = {'\_id': q['\_id'], 'email': q['email'], 'name':q['name'], 'nickname':q['nickname']} |
| 14 |  |
| 15 | return jsonify(output) |
| 16 |  |
| 17 | @app.route('/QuizFATEC/Usuarios/Login', methods=['POST']) |
| 18 | def get\_authenticated(): |
| 19 | colUsuarios = dbProvas["usuarios"] |
| 20 |  |
| 21 | id = request.json['\_id'] |
| 22 | password = request.json['password'] |
| 23 |  |
| 24 | q = colUsuarios.find\_one({'\_id': id}) |
| 25 | if q is None : |
| 26 | output = "Usuario nao cadastrado" |
| 27 | elif q['password'] != password: |
| 28 | output = "Senha invalida, verifique a ortografia" |
| 29 | else: |
| 30 | output = {'\_id': q['\_id'], 'email': q['email'], 'name':q['name'], 'nickname':q['nickname']} |
| 31 |  |
| 32 | return jsonify(output) |
| 33 |  |
| 34 | if \_\_name\_\_ == '\_\_main\_\_': |
| 35 | app.run(debug = True) |
|  |  |

Fonte: Autor (2019)

A quinta rota é uma requisição *post*, criada para cadastrar novos usuários ou atualizar os existentes, isso é possível através da função de *upsert* do MongoDB . Em seguida é feito uma busca do usuário inserido, de maneira a retornar o documento recém cadastrado ou atualizado. A última rota/função é referente a autenticação na aplicação, trata-se de uma requisição POST que recebe dois parâmetros, login e senha. Uma consulta é realizada no banco buscando pela ocorrência do *username*, que é a chave única identificadora dos documentos definidos no banco de dados. Em seguida, acontece a validação da senha, caso seja inválida uma mensagem é retornada, tratando especificamente da senha incorreta, caso seja válida, o objeto é retornado pela função, para que aplicação web consiga interpretar esses dados, através de uma mensagem customizada na tela inicial, por exemplo.

## 3.5 Aplicativo QuizFATEC

O Aplicativo foi desenvolvido utilizando o framework Ionic, com TypeScript para as logicas de programação das interfaces, e Angular, para desenvolvimento das Webviews, CSS foi utilizado para caracterizar e customizar alguns elementos. A figura abaixo demonstra a arquitetura geral do aplicativo, comunicação entre telas e afins.

Figura 11 - Arquitetura do Aplicativo QuizFATEC

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

Através da figura acima ilustra os comportamentos e ações disponíveis dentro do aplicativo, a primeira tela acessada é a tela de Login, onde o usuário pode realizar login ou se cadastrar, através da tela de cadastro. Uma vez cadastrado, o usuário realiza o login e a tela home é disponibilizado, conforme mostrado na figura acima os blocos verdes são acessíveis apenas com login efetuado. Na tela Home o usuário terá a opção de selecionar o tema de estudos desejados, uma vez selecionado o tema, a aplicação redireciona o usuário a tela respectiva, onde a questão será exibida junto com as respectivas alternativas. O usuário pode selecionar uma das alternativas e validar a reposta, ou reportar uma questão caso ela esteja ilegível segundo critérios do próprio usuário da aplicação, nesse caso um e-mail será gerado automaticamente, no aplicativo de e-mail padrão do smartphone. Caso o usuário reporte a questão ou acerte a reposta correta, ele passa a ter apenas a opção de voltar a tela home, onde pode selecionar novamente um tema e o ciclo se repete.

Pensando na agilidade do desenvolvimento e facilidade na correção de ponto único de falha, foi criado um componente Angular, denominado “*Question*”, que foi utilizado em todas as telas referentes as questões.

### 3.5.1 DataService - Comunicação com API

A classe DataService foi desenvolvida com TypeScript, tem o papel de impor as requisições GET e POST utilizadas pela aplicação. O resultado retornado na API é interpretado pelas telas responsáveis pela chamada. O Código abaixo demonstra a composição e funções/rotas estabelecidas dentro do Aplicativo.

Quadro 10 - Classe DataService

|  |  |
| --- | --- |
| 1 | import { Injectable } from '@angular/core'; |
| 2 | import { HttpClient, HttpHeaders } from '@angular/common/http'; |
| 3 |  |
| 4 | @Injectable({ |
| 5 | providedIn: 'root' |
| 6 | }) |
| 7 | export class DataService { |
| 8 |  |
| 9 | public baseUrl = "http://bc9394bd.ngrok.io/QuizFATEC"; |
| 10 | //public baseUrl = "http://localhost:5000/QuizFATEC"; |
| 11 |  |
| 12 | constructor(private http: HttpClient) { } |
| 13 |  |
| 14 | public getAuthenticated(data: any) { |
| 15 | return this.http.post(`${this.baseUrl}/Usuarios/Login`, data); |
| 16 | } |
| 17 |  |
| 18 | public postUser(data: any) { |
| 19 | return this.http.post(`${this.baseUrl}/Usuarios/`, data); |
| 20 | } |
| 21 |  |
| 22 | public getAllQuestions() { |
| 23 | return this.http.get(`${this.baseUrl}/Provas`); |
| 24 | } |
| 25 |  |
| 26 | public getRandomQuestion() { |
| 27 | return this.http.get(`${this.baseUrl}/Provas/Random/`); |
| 28 | } |
| 30 |  |
| 31 | public getRandomTheme(theme: string) { |
| 32 | return this.http.get(`${this.baseUrl}/Provas/Temas/${theme}`); |
| 33 | } |
| 34 | } |
|  |  |

Fonte: Autor (2019)

Essa classe é utilizada sempre que a comunicação com a API se faz necessária, dessa maneira, foram estabelecidas rotas para:

* getAuthenticated: função post responsável por enviar login e senha e retornar um documento JSON com os dados do usuário logado, ou uma mensagem de erro é retornada informando que a senha está errada ou usuário não cadastrado.
* postUser: função responsável por enviar os dados para cadastro do usuário, esses dados são enviados via JSON intrínseco na rota POST, ou também efetuar alguma atualização de login, devido a função upsert do MongoDB.
* getAllQuestions: esta rota foi utilizada principalmente para testes no inicio do desenvolvimento da aplicação, nenhum parâmetro é passado e todas as questões e repostas armazenadas no banco de dados são retornadas. Trata-se de uma requisição GET.
* getRandomQuestion: assim como rota comentada acima, essa função foi utilizada para testes, nenhum parâmetro era passado e uma questão completamente aleatória era retornada. Trata-se de uma requisição GET.
* getRandomTheme: função mais utilizada pela aplicação, a função recebe como parâmetro o texto da área de conhecimento selecionada, dentro das áreas abrangidas pelo Vestibular da FATEC, em seguida um JSON é retornado com uma questão sortida do tema selecionado.

### 3.5.2 Tela de Login e suas funções

É a primeira tela a ser exibida quando o aplicativo é instalado, nessa tela o usuário do sistema tem a possibilidade de realizar o login, após preencher os campos usuário e senha, que são obrigatórios, ou abrir a conta através do botão “ABRA SUA CONTA AQUI!”. A figura abaixo exibe a interface inicial da aplicação.

Figura 12 - Tela de Login Preenchida



Fonte: Autor (2019)

A tela de login funciona como barreira para o uso da aplicação apenas por aqueles que tiverem os seus dados pessoais persistidos no ambiente da aplicação. Essa tela possui também validadores de campos obrigatórios, limitador de tamanho de texto para o campo senha.

### 3.5.3 Tela de Cadastro e suas funções

A tela de cadastro permite o cadastro de novos usuários da plataforma ou a atualização de usuários já cadastrados. A figura abaixo ilustra a tela de cadastro.

Figura 13 - Tela de Cadastro



Fonte: Autor (2019)

A tela de cadastro possui os seguintes campos: *username*, nome completo, apelido, e-mail, senha. Todos os campos possuem validadores de preenchimento obrigatório, limitador de tamanho de texto para senha e um validador customizado de e-mail.

### 3.5.4 Tela Home e suas funções

A tela home é a interface principal da aplicação e deriva para as telas de quiz, nessa tela o usuário tem a capacidade de selecionar o tema para ênfase de estudos. A figura abaixo ilustra a tela inicial da aplicação.

Figura 14 - Tela Home



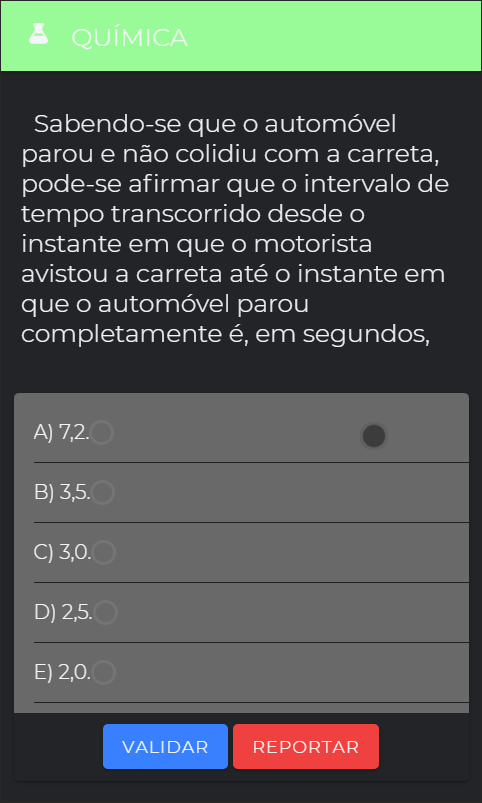
Fonte: Autor (2019)

A figura acima ilustra ainda uma mensagem boas vindas seguida do apelido do usuário, o objetivo é manter a comunicação leve e prática.

### 3.5.4 Tela das Questões e suas funções

Conforme figura 9, a aplicação possui 10 telas para questões, cada uma é customizada segundo a interface da tela Home (figura 12), isso foi feito pensando na melhor experiencia do cliente. A figura abaixo ilustra a composição da tela, o primeiro texto é sempre referente ao texto das questões, em seguida as alternativas são colocadas, e no fim da tela existem dois botões, VALIDAR e REPORTAR.

Figura 15 - Questão de Química



Fonte: Autor (2019)

A figura 14 ilustra os retornos possíveis após o do botão VALIDAR ter sido acionado. O objetivo é atender ao requisito funcional referente a reposta em tempo real durante o momento do quiz. Caso o usuário do sistema tenha acertado a reposta uma mensagem em tom verde é exibida no fim da tela retornando uma mensagem de sucesso e incentivando os estudos, conforme ilustrado na figura abaixo, em seguida o botão de validar é substituído pelo botão VOLTAR AO MENU, que leva o usuário a tela Home, onde ele pode selecionar novamente uma tema de questão a ser estudado.

Figura 16 – Botão Validar

Uma imagem contendo captura de tela

Descrição gerada automaticamente

Fonte: Autor (2019)

Caso o usuário do sistema erre a resposta, uma mensagem em tom vermelho será exibida na borda inferior da tela junto de uma mensagem estudando o usuário a tentar novamente, o botão de VALIDAR continuará disponível ao usuário até que acerte a reposta.

Finalmente, a tela de questões possui ainda a função de reportar questões, que venham a ser incompreensíveis, considerando que todas as questões foram raspadas através de um script padrão, sem iteração humana, existe uma margem considerável de falha, assim sendo a aplicação foi feita de maneira que caso uma questão esteja inconsistente, o usuário da aplicação poderá reportar a mesma através de um e-mail gerado pela própria aplicação, conforme ilustrado na figura abaixo.

Figura 17 - Diagrama Reportar Questão

Uma imagem contendo captura de tela, equipamentos eletrônicos

Descrição gerada automaticamente

Fonte: Autor (2019)

Conforme a figura ilustrada acima, a questão de Raciocínio Lógico trazida acima possui as alternativas em branco, tornando impossível para o usuário responder corretamente a questão, para isso a função reportar foi desenvolvido, através desse botão, o e-mail mostrado na figura é gerado automaticamente no aplicativo de e-mail padrão no smartphone do usuário. O usuário tem a alternativa de enviar o e-mail ou não, a aplicação gera apenas o título, destinatário e texto padrão. O envio do e-mail deve ser feito pelo próprio usuário, ao voltar para o QuizFATEC, tendo enviado o e-mail ou não, os botões de VALIDAR e REPORTAR são ocultados, restando apenas a opção de VOLTAR AO MENU para o usuário.

# 4. RESULTADOS

Este capítulo tem o objetivo de pontuar os resultados colhidos com a aplicação e comparar com os requisitos e tecnologias semelhantes levantados durante o capítulo 2. A análise trazida nos próximos pontos, foi feita considerando a opinião de um grupo de pessoas e comparando a aplicação final com os requisitos levantados e as tecnologias semelhantes.

## 4.1. *User Expirence*

O aplicativo foi desenvolvido pensando principalmente na experiência do usuário, considerando que uma boa experiência retém o usuário por mais tempo. A interface minimalista foi foco, principalmente para não cansar e nem enjoar o usuário que fizer uso constante. Além disso, o tom escuro do *background* do aplicativo foi escolhido para não lesar os olhos dos usuários, principalmente os que estudam a noite.

## 4.2. Tecnologias Semelhantes

Foram citadas duas tecnologias semelhantes no capítulo de fundamentação teórica, o aplicativo/jogo Perguntados e o Simulado do Denatran. Comparando o QuizFATEC com essas aplicações é perceptível uma interface mais amigável que o Simulado do Denatran, porém mais simples que o aplicativo Perguntados. Em questão de proposito, o QuizFATEC se assemelha ao Simulado do Denatran, onde o propósito é claro, auxiliar os estudos dos alunos do curso teórico, antes obrigatório, enquanto o Perguntados é um aplicativo de propósito geral, com o objetivo de incentivar uma competição sadia, por conhecimentos de mundo.

## 4.3. Requisitos Funcionais

A lista a seguir traz a memória os requisitos funcionais levantados no capítulo 2 referente a fundamentação teórica, e um paralelo com resultados colhidos.

### 4.3.1 Vestibulares da FATEC como foco do estudo

O requisito referente ao vestibular da FATEC como objeto principal de estudo foi plenamente satisfeito através da criação de um banco de dados alimentado por questões, repostas e textos retirados unicamente de vestibulares passados da FATEC, através de um código fonte de raspagem.

### 4.3.2 Aplicação Multiplataforma

O requisito referente a criar uma aplicação que fosse multiplataforma foi satisfeito através do uso do framework Ionic, que permite o desenvolvimento multiplataforma com um único código fonte (IONIC, 2019). A compilação da Webview em .apk, formato nativo do Android, e .ipa, formato nativo do iOS, é feita através do Apache Cordova.

### 4.3.3 Cadastro de Usuário e Login

O requisito referente ao cadastro de usuários foi contemplado através do desenvolvimento de uma tela no aplicativo que permite ao usuário se cadastrar, conforme figura 11, seguido de uma tela de login, figura 10. Dessa maneira é possível persistir dados dos usuários da aplicação, esses dados podem ser usados num trabalho futuro para pesquisas de público etc.

### 4.3.4 Escolher o Tema das Questões que serão foco do estudo

O requisito referente a permitir ao vestibulando selecionar o tema a ser estudado, foi contemplado através da interface da tela Home, figura 12, através dessa tela o vestibulando pode selecionar o tema a ser estudado.

### 4.3.5 Resposta Instantânea

O requisito referente e a resposta instantânea foi atendido através da composição da componente Angular de questão, a reposta é trazida junto a questão, no momento em a tela é exibida, dessa maneira a reposta é retornada instantemente no momento em que o botão VALIDAR é clicado.

### 4.3.6 Questões Cronometrados

O requisito referente a cronometrar as questões, permitindo que o vestibulando avaliasse o seu tempo de resposta conforme o tempo máximo por questão do vestibular da FATEC, não foi atendido nesse primeiro momento do projeto, mas ficará como próximos passos de um trabalho futuro.

# 5. CONSIDERAÇÕES FINAIS

O capítulo que se segue tratará sobre as conclusões obtidas durante e após o desenvolvimento desse projeto e sugestões para trabalhos futuros seguindo esse tema.

## 5.1. Contribuições

## 5.2. Trabalhos Futuros

Visando aprimorar a experiencia e aprendizado do vestibulando que utilizarem o QuizFATEC, as seguintes *features* foram recomendadas e mapeadas:

* Aprimorar o Scrapper de maneira a categorizar melhor os temas das questões.
* Inserir a função no aplicativo de selecionar mais questões do mesmo tema.
* Inserir a função no aplicativo de simulado, gerando uma prova automática de 54 questões na mesma estrutura dos Vestibulares da FATEC.
* Cadastrar os acertos e erros dos usuários do aplicativo para gerar métricas, possibilitando um sistema de ranking dos usuários.

# REFERÊNCIAS

AGENDA 21. **Conferência da Nações Unidas sobre Meio Ambiente e Desenvolvimento**. Disponível em http://www.mma.gov.br/sitio/index.php?ido=conteudo.monta&idEstrutura=18 Acesso em: 12/10/2010.

ALVES, J. M. **Proposta de um Modelo Híbrido de Gestão da Produção**: **aplicação na indústria aeronáutica. 2001.** 236 f. Tese (Doutorado em Engenharia Mecânica) - Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas, 2001.

ALVES FILHO, A. G.; CERRA, A. L.; MAIA, J. L. ; SACOMANO NETO, M. e BONADIO, P. V. G. Pressupostos da Gestão da Cadeia de Suprimentos: Evidências de Estudos sobre a Indústria Automobilística. **G&P – Gestão & Produção.** Vol. 11, n. 3, p. 275-288, Set.-Dez. 2004.

ANGERHOFER, B. J. e ANGELIDES, M. C. *A model and a performance measurement system for collaborative supply chains.* **Science Direct - Decision Support Systems**, Vol. 42, p. 283-301, 2006.

BALLOU, R. H. **Gerenciamento da Cadeia de Suprimentos**. São Paulo: Artmed, 2005.

SANTOS, R. F. **Proposta de um sistema híbrido de Contabilidade Gerencial: Estudo de Caso na Empresa Siber do Brasil S.A. 2005.** 168 f. Dissertação (Mestrado em Ciência no Curso de Engenharia Aeronáutica e Mecânica, Área de Produção) - ITA - Instituto Tecnológico de Aeronáutica, São José dos Campos, 2005.

SANTOS, R. S. e ALVES, J. M. Proposta de um Modelo de Gestão da Cadeia de Suprimentos com o Apoio da Teoria das Restrições, VMI e B2B. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 2009, Salvador. **Anais...** Salvador, 2009. 12 f.

ZILIO, S. D. Modeling and verification of parallel processes. In: CASSEZ, Franck et al (Ed.). **Mobile processes:** a commented bibliography. New York: Springer-Verlag, 2001. p. 206-222. (Lectures Notes in Computer Science, v. 2067).

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR** 5462: 1994: confiabilidade e mantenabilidade: terminologia. Rio de Janeiro, 1994.

EMBRAPA. Unidade de Apoio, Pesquisa e Desenvolvimento de Instrumentação Agropecuária (São Carlos, SP). Paulo Estevão Cruvinel. **Medidor digital multissensor de temperatura para solos.** BR n. PI 8903105-9. 26 jun. 1989, 30 maio 1995.

MICROSOFT. **Project for windows 95:** project planning software. Version 4.1: [S.l.]: Microsoft Corporation, 1995. Conjunto de programas. 1 CD-ROM.

ALLISON, D.O.; MINECK, R.E. **Aerodynamic characteristics and pressure distributions for an executive-jet baseline airfoil section**. Washington, DC: NASA, 1993. 25 p. (NASA TM-4529).

MARINHO, P. A pesquisa em ciências humanas. Petrópolis: Vozes, 1980 apud MARCONI, M. A.; LAKATOS, E. M. Técnicas de pesquisa. São Paulo: Atlas, 1982.

As referências acima são das fontes:

Amarelo: Internet

Verde: Trabalho de Graduação FATEC, Dissertação ou Tese de Mestrado e Doutorado

Azul Claro: Artigo publicado em periódico

Magenta: Livro

Azul Escuro: Congresso

**Vermelho:** Capítulo de livro

**Cinza:** Normas técnicas

**Roxo:** Patentes

**Verde Escuro:** Programa de computador

**Marrom:** Relatório técnico

**AZUL Petróleo:** Exemplo de referência com apud

CRUZ, H. R. **Carona Solidária: Um Aplicativo para Promover Sustentabilidade, Colaboração e Economia na FATEC São José dos Campos.** 65 f. Dissertação (Trabalho de Graduação em Tecnologia de Banco de Dados) - FATEC – Faculdade de Tecnologia Prof. Jessen Vidal, São José dos Campos, 2016.

FORBES, D., GETTING REAL ABOUT NOSQL AND THE SQL ISN’T SCALABLE LIE, Publicação Blog, 2010. – encontrar ref

D’AVILA, G. T. **Vestibular: Fatores Geradores de Ansiedade na “Cena da Prova”.** 12 f. Artigo Científico, Universidade Federal de Santa Catarina, Florianópolis, 2003.

O'Reilly Media, Inc. Christopher J. Date. **What is Database Desing, Anyway?.** UK n ISBN: 9781492048428. 1 jan. 2016.

G1 SP. **Inscrições para Vestibular da FATEC Aberta**. Disponível em https://g1.globo.com/sp/sao-paulo/noticia/fatec-abre-inscricoes-para-vestibular-de-meio-de-ano-nesta-terca-feira.ghtml Acesso em: 16/09/2019.

Abril Guia do Estudante. **Técnicas para Estudo para qualquer prova**. Disponível em https://guiadoestudante.abril.com.br/enem/7-otimas-tecnicas-de-estudo-para-qualquer-prova/ Acesso em: 16/09/2019.

Abril Guia do Estudante. **Porque fazer simulados ajuda na hora da prova**. Disponível em https://guiadoestudante.abril.com.br/estudo/por-que-fazer-simulados-ajuda-na-hora-da-prova/ Acesso em: 16/09/2019.

Administradores. **Importância de se preparar par o mercado de trabalho**. Disponível em http://www.administradores.com.br/artigos/carreira/a-importancia-de-preparar-se-para-o-mercado-de-trabalho/109009/ Acesso em 16/09/2019.

Portal do Governo Estado de São Paulo. **História do Centro Paula Souza**. Disponível em http://www.saopaulo.sp.gov.br/spnoticias/ultimas-noticias/especial-40-anos-do-centro-paula-souza-conheca-a-linha-do-tempo-da-instituicao/ Acesso em 16/09/2019.

Vestibular da FATEC. **Unidades e Cursos da FATEC**. Disponível em http://www.vestibularfatec.com.br/unidades-cursos/ Acesso em 20/04/2019.

WIEDERHOLD, G. **The Structural Model for Database Design.** 22 f. Artigo Científico Stanford University, California, 1983.

Infoq. **Google Firebase: Back-end completo para aplicações web e mobile**. Disponível em https://www.infoq.com/br/news/2016/07/google-firebase Acesso em 13/06/2019.

Knight Center for Journalism in the Americas. **Unraveling data scraping: Understanding how to scrape data can facilitate journalists' work**. Disponível em https://knightcenter.utexas.edu/en/blog/00-9676-unraveling-data-scraping-understanding-how-scrape-data-can-facilitate-journalists-work Acesso em 23/09/2019

PDFQuery. **Concise, friendly PDF scraping using JQuery or XPath syntax**. Disponível em https://github.com/jcushman/pdfquery Acesso em 15/05/2019

PyPDF2. **PyPDF2 Documentation**. Disponível em https://pythonhosted.org/PyPDF2/ Acesso em 17/05/2019

PyPDF2’s Origin. **Home page for the PyPDF2 project**. Disponível em http://mstamy2.github.io/PyPDF2/ Acesso em 17/05/2019

Apache Cordova, **Mobile apps with HTML, CSS & JS Target multiple platforms with one code base Free and open source**. Disponível em https://cordova.apache.org/ Acesso em 20/05/2019

Jupyter Notebook, **The Jupyter Notebook**. Disponível em https://jupyter.org/ Acesso em 20/09/2019

MongoDB, **The MongoDB 4.2 Manual**. Disponível em https://docs.mongodb.com/manual/?\_ga=2.118473677.532038213.1558491340-1615551956.1558491340 Acesso em 08/10/2019

TinyURL, **Making over a billion long URLs usable! Serving billions of redirects per month**. Disponível em https://tinyurl.com/ Acesso em 08/10/2019

ORG NOSQL, **Your Ultimate Guide to the Non-Relational Universe!**. Disponível em <http://nosql-database.org/> Acesso em 08/10/2019

JSON, **Introducing JSON**. Disponível em https://www.json.org/ Acesso em 08/10/2019

Referencias faltantes:

<https://buildmedia.readthedocs.org/media/pdf/pdfminer-docs/latest/pdfminer-docs.pdf>

postman

[http://www.preguntados.com/pt#](http://www.preguntados.com/pt)

<https://play.google.com/store/apps/details?id=com.etermax.preguntados.lite&hl=pt_BR>

<https://play.google.com/store/apps/details?id=com.etermax.trivia.preguntados2&hl=pt_BR>

<https://play.google.com/store/apps/details?id=br.gov.sp.detran.simulado&hl=pt_BR>

Haykin, S. (2001) “Redes Neurais: Princípios e Prática”. 2.ed. Porto Alegre: Bookman.

http://computacaointeligente.com.br/algoritmos/k-vizinhos-mais-proximos/

# APÊNDICE A/ANEXO A – EXEMPLO DE APÊNDICE/ANEXO

**A.1 Exemplo de Subseção do Apêndice A**

Apêndice e anexos são opcionais no documento. O documento pode conter quantos apêndices ou anexos forem necessários. Lembrando que **Apêndice** é um documento ou texto elaborado pelo autor a fim de complementar sua argumentação e **Anexo** é um documento ou texto **não** elaborado pelo autor que servem de fundamentação ou comprovação (por exemplo: relatórios, mapas, leis, estatutos dentre outros). Os apêndices devem aparecer após as referências, e os anexos, após os apêndices, e ambos devem constar no sumário.

Caso tenha mais do que um apêndice e ou um anexo, deve-se utilizar a nomenclatura: Apêndice A, Apêndice B, Apêndice C etc.

***INSTRUÇÕES GERAIS PARA FORMATAÇÃO DO TRABALHO DE GRADUAÇÃO***

## Como deve ser a formatação das Figuras, Tabelas e Equações no trabalho

É caracterizado como figura todo desenho, esquema, fluxograma, fotografia, gráfico, mapa, organograma, planta, quadro, retrato, figura, imagem, entre outros.

Para as figuras sua identificação aparece na parte superior, precedida da palavra Figura seguida de seu número de ordem de ocorrência no texto, em algarismos arábicos, ponto (em negrito) e da respectiva legenda. A identificação da figura e a legenda devem ser em texto centralizado, e em espaçamento simples, caso ocupe mais de uma linha do texto. A legenda da figura deve conter as informações necessárias à sua compreensão.

Na parte inferior da figura, deve ser indicado a fonte consultada de acordo com o modelo de referência adotado no trabalho (elemento obrigatório, mesmo que seja produção do próprio autor). A fonte deve ser alinhada à esquerda na figura em Times New Roma tamanho 10. A ilustração deve ser citada no texto como Figura (com a palavra iniciando em maiúsculo) seguida de seu número, o mais próximo possível do trecho a que se refere.

EXEMPLO:

Para atender os objetivos [...] e procedimentos técnicos utilizados na Figura 1. (Observe que a palavra figura inicia com letra maiúscula).

|  |
| --- |
| Figura 18 - Proposta metodológica. |

Fonte: Adaptada de Santos (2010).

Para as tabelas sua identificação aparece na parte superior, precedida da palavra Tabela seguida de seu número de ordem de ocorrência no texto, em algarismos arábicos, ponto (em negrito) e da respectiva legenda. A identificação da tabela e a legenda devem ser em texto centralizado, e em espaçamento simples, caso ocupe mais de uma linha do texto. A legenda da tabela deve conter as informações necessárias à sua compreensão.

Na parte inferior da tabela, deve ser indicado a fonte consultada de acordo com o modelo de referência adotado no (elemento obrigatório, mesmo que seja produção do próprio autor). A fonte deve ser alinhada à esquerda na tabela em Times New Roma tamanho 10. A tabela deve ser citada no texto como Tabela (com a palavra iniciando em maiúsculo) seguida de seu número, o mais próximo possível do trecho a que se refere.

EXEMPLO:

A Tabela 1 apresenta a população entre... (observe que a palavra tabela inicia com letra maiúscula).

Tabela 2 - População de 15 a 24 anos de idade.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ano** | **População de 15 a 24 anos de idade** | | | | |
| **Total Absoluto** | **Variação** | | **Participação em relação à população total** | **Taxa de crescimento (%)** |
| **Absoluta** | **Relativa (%)** |
| 1940 | 8246733 |  |  | 20,1 |  |
| 1950 | 10489368 | 2426352 | 27,2 | 20,3 | 2,4 |
| 1960 | 13413413 | 2924048 | 27,9 | 19,2 | 2,5 |
| 1970 | 18539088 | 5125672 | 38,2 | 19,9 | 3,3 |
| 1980 | 25089191 | 6550103 | 35,3 | 21,1 | 3,1 |
| 1991 | 28582350 | 3493159 | 13,9 | 19,5 | 1,2 |
| 1996 | 31088484 | 2506134 | 8,8 | 19,8 | 1,7 |

Fonte: Oliveira (2015)

No caso das equações, para facilitar a leitura, devem aparecer no texto como Equação seguida de seu número de ordem de ocorrência no texto, em algarismos arábicos. As variáveis da equação devem estar descritas em seguida.

EXEMPLO:

A Equação 1 representa a condição... (observe que a palavra equação inicia com letra maiúscula).

x2 + y2 = z2 (1)

Onde x, y e z são variáveis do processo.

## Como deve ser mencionada as Siglas no trabalho

Caso o(s) autor(es) do trabalho opte em não utilizar a lista de abreviaturas e siglas, quando mencionadas pela primeira vez no texto, deve ser indicada entre parêntesis, precedida do nome completo. EXEMPLO: Segundo a Associação Brasileira de Normas Técnicas (ABNT) ... (observe que as palavras referentes à abreviação iniciam com a letra maiúscula).

## Como deve ser feitas as citações no trabalho

As citações no texto, figuras e tabelas devem seguir o sistema “autor-data”. Este sistema deve ser seguido consistentemente ao longo de todo o trabalho, permitindo sua correlação na lista de referências (item REFERÊNCIAS BIBLIOGRÁFICAS).

**Sistema autor-data**

No texto, deve-se indicar o(s) Autor(es) pelo SOBRENOME sem as iniciais, em maiúsculas, seguido do ano da publicação, separados por vírgula e entre parênteses. Casos especiais de citação devem seguir o modelo (ver item Como utilizar as referências bibliográficas no texto do trabalho). No texto das referências, o sistema data-autor, devem aparecer em ordem alfabética.

EXEMPLOS:

**(a)** Robôs flexíveis apresentam graus de liberdade adicionais (SOUZA, 2013).

**(b)** Citações de mais de um documento de autores diferentes devem ser separados por “;”. Exemplo: (SILVA, 2003; COSTA, 2000; OLIVEIRA, 2014).

**(c)** Quando houver coincidência de sobrenomes de autores, acrescentar as iniciais de seus prenomes: (BARBOSA, C., 1958) e (BARBOSA, O., 1958). Se mesmo assim existir coincidência, colocam-se os prenomes por extenso: (BARBOSA, Cássio, 1965) e (BARBOSA, Celso, 1965).

**(d)** As citações de diversos documentos do mesmo autor, publicados num mesmo ano, são distinguidas pelo acréscimo de letras minúsculas, em ordem alfabética, após a data e sem espacejamento. Acrescentar as letras após a data, tanto a citação, quanto na referência. Exemplo: a pesquisa apresentou um resultado (SILVA, 2010a) e também outro resultado (SILVA, 2010b).

## Como utilizar as referências bibliográficas no texto do trabalho

No texto há várias maneiras de referenciar a literatura utilizada para o desenvolvimento do trabalho. Há várias maneiras de se fazer uma citação como, citação indireta, citação indireta, citação de citação e entre outras.

**(a) Citação indireta:** No caso de citações indiretas onde o texto foi baseado na obra de um autor consultado. No texto, pode ser referenciado como:

exemplo:

Segundo Santos (2010), o apoio ao...

Santos (2010) acredita que...

O sistema deve ser dimensionado (SANTOS, 2010).

**(b) Citação direta:** No caso de citações diretas, onde ocorreu a transcrição textual de parte da obra de um autor consultado, deve-se colocar a citação entre aspas e indicar a página onde se encontra a citação na referência.

exemplo:

Santos (2010, p. 23) afirma que “seu método será aplicado nos trabalhos em série”.

“O trabalho pode ser entendido como um ponto chave” (SANTOS, 2010).

**(c) Citação com 4 ou mais autores:** Em uma citação com 4 ou mais autores coloca-se o nome do primeiro autor seguido de et al..

exemplo:

Segundo Miguel et al. (2010), a diferença [...] e qualitativa é que...

A diferença [...] e qualitativa é que [...] final (MIGUEL et al., 2010).

**(d) Citação de citação:** É uma citação, direta ou indireta, de um texto em que não se teve acesso ao original.

exemplo:

Segundo Pires (2008 apud SANTOS, 2010), o apoio ao...

Segundo Pires (2008) citado por Santos (2010), o apoio ao... (opção ao apud)

O sistema de testes do perfil é subliminar (PIRES, 2009 apud SANTOS, 2010).

**(e) Citação longa:** Citações com mais de 3 linhas devem receber uma formatação especial, onde o tamanho da letra será 10, com espaçamento simples e início do parágrafo com 4 cm. Com espaçamento simples.

exemplo: Para sistema data-autor

Esta relevância também foi constatada por Hansen e Mowen (2001, p. 31) na afirmação de que:

“A grande melhoria no transporte e na comunicação levaram a um mercado global para muitas empresas de manufatura e de serviços. Várias décadas atrás, as empresas não sabiam sobre, e nem se importavam com, o que empresas similares do Japão, França, Alemanha e Cingapura estavam fazendo. Estas empresas estrangeiras não eram concorrentes, já que os mercados eram separados por uma distância geográfica.”