MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - *Campus* Formiga
Curso de Ciência da Computação

ESTUDO E IMPLEMENTAÇÃO DE UM INDEXADOR DE DOCUMENTOS COM *WEBSTORAGE*

Guilherme Cardoso Silva

Orientador: Prof. Dr. Manoel Pereira Júnior

Formiga - MG
2018

GUILHERME CARDOSO SILVA

ESTUDO E IMPLEMENTAÇÃO DE UM INDEXADOR DE DOCUMENTOS COM *WEBSTORAGE*

Monografia do trabalho de conclusão de curso apresentado ao Instituto Federal Minas Gerais - Campus Formiga, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Manoel Pereira Júnior

Formiga - MG

2018

Silva, Guilherme Cardoso.
004 Estudo e implementação de um indexador de documentos com webstorage / Guilherme Cardoso Silva.. -- Formiga : IFMG, 2018.
80p. : il.

Orientador:Prof. Dr. Manoel Pereira Júnior
Trabalho de Conclusão de Curso – Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – *Campus* Formiga.

1. Indexador . 2. OCR. 3. Webstorage. 4. Gerenciador de Documentos. 5. Laravel. I. Título.

CDD 004

Ficha catalográfica elaborada pela Bibliotecária MSc. Naliana Dias Leandro CRB6-1347

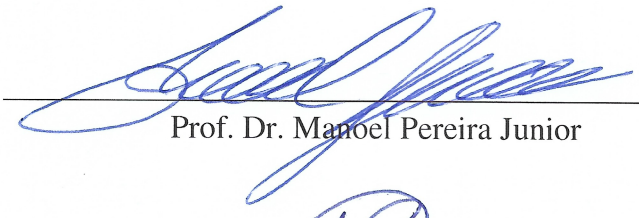
GUILHERME CARDOSO SILVA

ESTUDO E IMPLEMENTAÇÃO DE UM INDEXADOR DE DOCUMENTOS COM WEBSTORAGE

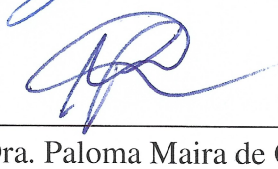
Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Minas Gerais-Campus Formiga, como Requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Aprovado em 21 de Novembro de 2018.

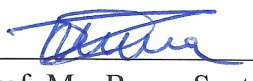
BANCA EXAMINADORA



Prof. Dr. Manoel Pereira Junior



Profa. Dra. Paloma Maira de Oliveira



Prof. Me. Roger Santos Ferreira

Dedico este trabalho a meus pais Dalmo e Patricia, em especial a minha mãe, que nunca duvidou da minha capacidade e sempre me incentivou em sempre continuar estudando e buscando pelos meus sonhos.

Agradecimentos

Primeiramente eu agradeço a Deus que me concebeu o dom da vida, me fortalecendo e me guiando, não apenas nestes últimos 4 anos, mas por toda a minha caminhada cristã, onde se fez presente nos momentos mais felizes até os mais difíceis e tristes.

Aos meus colegas de turma, companheiros de trabalhos e principalmente aos meus verdadeiros amigos que fizeram parte da minha formação de diversas formas. Em especial a Renata Cunha que partilhou de cada momento que vivenciei no meu processo de formação, sempre me apoiando, dividindo as responsabilidades e sempre se empenhando a me ensinar matérias que me pareciam impossíveis, mas a sua dedicação me ajudou a passar por cada uma delas. E também ao Roger Ferreira que dedicou grande parte do seu tempo a me auxiliar com toda a paciência, graças a sua dedicação pude ampliar minha visão e meus conhecimentos para além da instituição de ensino, espero que possamos trabalhar em muitos outros projetos.

Agradeço aos meus professores que sempre se empenharam em transmitir o enorme conhecimento que possuem, sempre ensinado com carácter e respeito, permitindo que eu pudesse crescer profissionalmente e também pessoalmente. Agradeço em especial ao meu professor, orientador e amigo Manoel Pereira Júnior, que nunca se limitou ao ensino em sala de aula, mas sempre me motivou, me instruiu, me desafiou e me proporcionou oportunidades de aprendizado sempre levando em consideração meus limites e necessidades, mas também sempre acreditando que eu era capaz de muito mais.

Agradeço a Meiriane Leal, a mulher com quem amo partilhar a minha vida, que apesar das dificuldades e de todos os sacrifícios que vivemos juntos durante este tempo, sempre esteve ao meu lado me apoiando, me fortalecendo e me edificando. Obrigado por toda a paciência, o carinho, o companheirismo, o amor e principalmente pela sua capacidade em me trazer a paz na transição de cada semestre, sem você eu não teria conseguido.

E por fim agradeço aos meus familiares, meus avós, as minhas irmãs e principalmente aos meus meus pais, Dalmo e Patricia, que dedicaram todos os dias da minha vida a me motivar, me ensinar e principalmente a me educar para me tornar o homem que sou hoje. Foi graças a sacrifício que fizeram por mim todos os dias, que pude chegar a onde estou hoje.

E a todos os demais que contribuíram de alguma forma para que eu pudesse chegar até aqui, o meu muito obrigado.

"Jamais considere seus estudos como uma obrigação, mas como uma oportunidade invejável para aprender a conhecer a influência libertadora da beleza do reino do espírito, para seu próprio prazer pessoal e para proveito da comunidade à qual seu futuro trabalho pertencer."

Albert Einstein

Resumo

Este trabalho tem caráter prático e apresenta a implementação de um protótipo de um indexador de documentos com armazenamento em *webstorage*. O protótipo foi implementado com as principais ferramentas que compõem um Gerenciador Eletrônico de Documentos, sendo elas: reconhecimento ótico de caracteres, indexação, armazenamento e busca. Por meio do sistema é possível que um usuário possa indexar, armazenar e a partir de uma necessidade informacional realizar uma buscar por documentos, levando em consideração o conteúdo textual contido neles. São apresentados conceitos sobre o gerenciamento de documentos eletrônicos, armazenamento em nuvem (*webstorage*), o processo de indexação, técnicas de recuperação de documentos, entre outros. O indexador foi implementado em um ambiente *web* utilizando o *framework* Laravel, permitindo que o sistema seja acessado por qualquer aparelho conectado à internet. Por fim, são apresentadas algumas considerações finais sobre o desenvolvimento do projeto, junto a trabalhos futuros para serem realizados.

Palavras-chave: Indexador, OCR, *Webstorage*, Gerenciador de Documentos, Laravel.

Abstract

This practical work presents the implementation of a prototype of a document indexing tool with webstorage. The prototype implements some parts of an Electronic Document Manager, namely: Optical Character Recognition, Indexing, Storage and Search. Through the system it is possible to index, store and conduct a search for documents, taking into account the textual content contained in them. Concepts on the management of electronic documents, cloud storage (webstorage), the process of indexing, document retrieval techniques, among others are presented. The indexing tool was implemented in a web environment using the Laravel framework, allowing the system to be accessed by any device connected to the internet. Finally, some final considerations about the development of the project are presented as well future ideas to be implemented.

Keywords: *Index*, *Ocr*, *Webstorage*, *Document Manager*, *Laravel*.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AJAX	*Asynchronous Javascript and XML*
API	Interface de Programação de Aplicativos (*Application Program Interface*)
BD	Banco de Dados
BPM	Gerenciamento de Processos de Negócio *Business Process Management*
CRUD	Criar, Ler, Atualizar e Deletar (*Create, Read, Update and Delete*)
COLD	Saída de Computador para Disco Laser (*Computer Output to Laser Disk*)
CSS	Folhas de Estilo em Cascata (*Cascading Style Sheets*)
DER	Diagrama de Entidade-Relacionamento
DM	Gerenciamento de documento (*Document Management*)
DOM	Modelo de Documento por Objetos (*Document Object Model*)
EAP	Estrutura Analítica do Projeto
ECM	Gestão de Conteúdo Empresarial (*Enterprise Content Management*)
EDM	Gerenciamento Eletrônico de Documentos (*Electronic Document Management*)
ERM	Gerenciamento de Relatórios Corporativos (*Enterprise Report Management*)
GED	Gerenciador Eletrônico de Documentos
JS	*JavaScript*
HTML	Linguagem de Marcação de Hipertexto (*HyperText Markup Language*)
ICR	Reconhecimento Inteligente de Caracteres (*Intelligent Character Recognition*)
IDE	Ambiente de Desenvolvimento Integrado (*Integrated Development Environment*)
IFMG	Instituto Federal de Minas Gerais

LSTM	Unidades de Memória de Curto Prazo (*Long short-term memory*)
MVC	Modelo-Visão-Controle (*Model-View-Controller*)
OCR	Reconhecimento Ótico de Caracteres (*Optical Character Recognition*)
ORM	Mapeamento Objeto-Relacional (*Object Relational Mapper*)
PDF	Formato de Documento Portátil (*Portable Document Format*)
PHP	Pré-processador de Hipertexto (*Hypertext Preprocessor*)
POO	*Programação Orientada a Objetos*
POST	Autoteste de Inicialização (*Power On Self Test*)
RAM	Memória de Acesso Aleatório (*Random Access Memory*)
RI	Recuperação da Informação
RIM	Registros e Gerenciamento de Informações (*Records and Information Management*)
SGBD	Sistemas de Gerenciamento de Banco de Dados
SRI	Sistema de Recuperação de Informação
URL	Localizador Uniforme de Recursos *Uniform Resource Locator*
XML	Linguagem de Marcação Extensível (*Extensible Markup Language*)
WWW	Rede Mundial de Computadores (*World Wide Web*)

Sumário

1 INTRODUÇÃO

Este capítulo apresenta detalhes do problema abordado, a motivação, a justificativa, os objetivos propostos para uma possível solução do problema e um breve resumo dos capítulos a seguir.

1.1 Problema Estudado

Os homens primitivos, sob a necessidade de comunicação, recorreram a objetos simbólicos e sinais materiais possibilitando a transmissão de informação entre eles, os quais passaram a ser classificados como os primeiros sistemas de escrita (FREIBERGER, 2010, p. 123).

Freiberger (2010, p. 124) afirma que a medida que a sociedade evoluía em grupos mais organizados, os indivíduos começaram a compreender o valor que os documentos apresentavam, passando então a conservar e armazenar em alguma forma física o resultado de suas atividades políticas, sociais, econômicas, religiosas e de suas vidas particulares.

A informação vem sendo registrada em papel há séculos (FANTINI et al., 2001, p. 1). Ao decorrer dos anos é possível observar um grande acúmulo de papéis, sendo que essa quantidade vem aumentando cada dia mais (SILVA et al., 2003, p. 1).

Fantini et al. (2001, p. 1) cita que de acordo com informações levantadas pela empresa Coopers & Lybrand, um funcionário gasta cerca de 30 dias em um período de 1 ano procurando por documentos, aproximadamente 10% da sua jornada de trabalho, além de gerar cerca de 19 cópias de cada arquivo. Santoyo (2008) diz que uma das consequências de uma má gestão de documentos faz com que uma empresa perca um documento a cada 12 segundos, sendo que o custo médio de recuperação é cerca de US$120. O autor ainda completa dizendo que "(...) no caso de uma corporação, os arquivos respondem por uma parte expressiva das despesas."

Segundo Menezes (2016, p. 27-28), a constante geração de novos documentos nas organizações resulta em um acúmulo de massa documental. Por isso, são necessárias novas formas de armazená-los, que permitam ao mesmo tempo a diminuição do espaço físico de armazenamento e também o aumento da velocidade na busca por um documento.

> O documento tradicional, aposto em papel, não mais se adéqua à necessidade atual de dar agilidade à circulação de informações. São evidentes as suas limitações, tanto em relação à conservação, como à transmissibilidade e segurança (GANDINI; SALOMÃO; JACOB, 2001, p. 2).

Com a evolução dos meios digitais, diversas formas diferentes de informações são geradas por organizações, como por exemplo papéis, áudios, fotos e planilhas. Como já mencionado, essas informações acarretam em uma grande massa documental, sendo que o problema de controle e organização dessas informações se torna cada vez mais complexo (AMAZONAS et al., 2008, p. 3).

Barros et al. (2016) diz que:

> À medida que os documentos são criados, muitas vezes sem nenhuma organização que contemple todo o seu ciclo de vida, massas documentais vão se acumulando e se perdendo em meio digital, além desta problemática uma nova se origina, a preservação desses documentos, à longo prazo, afinal são de extrema relevância para manutenção e difusão da memória e cultura da humanidade.

A partir da dificuldade de acesso e recuperação de informações, surgiu a necessidade do desenvolvimento de métodos que aumentassem a produtividade de funcionários em escritórios, como engenheiros, bancários, secretárias, advogados e gerentes (FANTINI et al., 2001, p. 1). Um dos métodos que possibilita o gerenciamento dessa inúmera quantidade de documentos de forma ágil e eficiente é conhecido como Gerenciador Eletrônico de Documentos (GED) (SILVA et al., 2003, p. 2).

Um GED é um sistema de informação que tem como parte fundamental o processo de indexação (MACEDO et al., 2003, p. 24). De acordo com Rubi e Fujita (2003, p. 67) essa etapa pode ser considerada como a parte mais importante desse tipo de sistema. As autoras ainda afirmam que "esse processo é determinante para a qualidade dos resultados das buscas realizadas pelos usuários".

1.2 Justificativa

Diariamente são produzidos inúmeros documentos, em diversos formatos. A grande quantidade desses arquivos tem um enorme impacto na produtividade do trabalho contemporâneo, que a cada dia acumula mais informações (MENEZES, 2016, p. 27-28).

Muitas vezes funcionários de uma empresa não sabem como acessar ou encontrar determinados arquivos, desperdiçando horas ou até dias de trabalho apenas para encontrá-los. Uma pesquisa da empresa Coopers & Lybrand afirma que são gastos cerca de quatro semanas em um ano buscando por arquivos que foram arquivados de forma equivocada (FANTINI et al., 2001, p. 1).

Segundo Gandini, Salomão e Jacob (2001, p. 5) documentos físicos já se tornaram um meio inviável de armazenamento. Papel é uma representação de informação muito frágil e quando exposto a várias condições de temperatura em que estão arquivados ou a forma como são manuseados, podem facilmente levar ao seu desgaste. Silva et al. (2003,

p. 7) cita que o custo para restauração desses arquivos tem um valor médio de US$ 120 a US$ 250.

O gerenciamento e organização de documentos empresariais pode ser uma tarefa complexa. Grandes e pequenas empresas recebem uma grande quantidade de correspondência de seus clientes e fornecedores, além de toda a documentação gerada pela própria organização, como memorandos, cartas, requisições, documentação fiscal, *etc.* (ANDRADE et al., 2002, p. 3).

Barros et al. (2016, p. 3) afirma que a preservação dos documentos e relatos passados são importantes para a história, desenvolvimento educacional, ciência e até para a cultura de um povo, que permite ainda o rompimento das barreiras de comunicação entre gerações.

O presente trabalho tem sua importância devido à contribuição com a sociedade informacional, que diariamente necessita de novos métodos, cada vez mais eficientes e ágeis, a fim de reduzir o tempo gasto em processos de recuperação gasto por organizações (MUSAFIR, 2001).

1.3 Objetivos

Um GED é composto por diversas funcionalidades e módulos. Ele tem como função o gerenciamento de um documento por todo o seu ciclo de vida, desde sua criação até sua recuperação, como pode ser visto na Figura 1.

No entanto, o escopo deste trabalho foi limitado à quatro funcionalidades (destacadas na Figura 1 em cores) de um GED genérico, que permite a indexação e armazenamento de documentos. Mais detalhes sobre um GED serão discutidos na seção 2.3.

Figura 1 – Protótipo proposto

1.3.1 Objetivo Geral

O objetivo principal deste trabalho é o desenvolvimento de um protótipo que possibilite a indexação e armazenamento de documentos em um ambiente *web* que visa a privacidade, segurança e principalmente a redução do tempo gasto por funcionários na recuperação de um documento específico, permitindo um melhor investimento de tempo que poderá ser aplicado em outros processos da organização.

1.3.2 Objetivos Específicos

Os objetivos específicos são os seguintes:

- Implementar métodos de Reconhecimento Ótico de Caracteres (OCR) a fim de realizar a extração do conteúdo textual de um documento digitalizado para indexação;
- Implementar um indexador de documentos permitindo a busca de documentos armazenados no gerenciador;
- Replicação de um motor de busca para testes sobre os documentos indexados;
- Realizar testes no sistema desenvolvido, a fim de realizar correções de erros.

1.4 Visão Geral do Documento

O restante do trabalho está estruturado conforme descrito a seguir. No Capítulo 2, são descritos os fundamentos teóricos sobre recuperação da informação (RI), sistemas de recuperação da informação (SRI), gerenciador eletrônico de documentos, técnicas de processamento de imagens, indexação e *webstorage*. O Capítulo 3 aborda os materiais e métodos utilizados para o desenvolvimento deste trabalho. O Capítulo 4 descreve detalhes técnicos sobre o protótipo desenvolvido, detalhando sua estrutura hierárquica, a metodologia de implementação, junto as ferramentas acopladas no sistema.O Capítulo 5 relata os resultados obtidos após o desenvolvimento do indexador proposto. E por fim o Capítulo 6 apresenta as considerações sobre o protótipo desenvolvido.

2 FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta uma breve conceituação teórica sobre alguns conceitos necessários para a compreensão do trabalho, como recuperação da informação (RI), sistemas de recuperação da informação (SRI), gerenciador eletrônico de documentos (GED), reconhecimento óptico de caracteres (OCR), indexação webstorage e um conjunto de trabalhos relacionados que serviram como base para referencial teórico desde trabalho.

2.1 Recuperação da Informação

Recuperação da Informação é o nome do processo pelo qual um determinado usuário é capaz de converter sua necessidade de informação em uma lista de documentos contendo informações relevantes (MOOERS, 1951, p. 25).

A necessidade de RI já existe há muitos anos. Já na Segunda Guerra Mundial, Cruz (2011, p. 11-13) diz que Vannevar Bush especulava o quanto a ciência e a tecnologia poderiam trazer para a sociedade em tempos de paz. Seu foco de pesquisa era tentar modificar a forma como se pensava e organizava o conhecimento, utilizando alguma ferramenta que permitisse ao usuário ter acesso à grande massa de informações criada pela humanidade.

Segundo Cruz (2011), por volta de 1945 Vannevar Bush, tinha como ideia a criação de uma ferramenta que pudesse armazenar todos os livros, registros e comunicações e que, a partir do momento que fossem indexados, pudessem ser consultados de forma rápida e automática. Essa necessidade informacional fez com que as pesquisas na área de RI tivessem início.

Shafi'I et al. (2014, p. 1) afirma que a RI é de extrema importância para a documentação e organização do conhecimento armazenado. Ele também afirma que para ter acesso a documentos relevantes é necessário que um sistema seja eficiente para recuperá-los mediante a grande massa de dados criados diariamente.

Ainda de acordo com Shafi'I et al. (2014, p. 1), o requisito para a recuperação de informação é representado em forma de consulta a qual pode ser composta por um ou mais termos de pesquisa, sendo que a decisão para a recuperação é feita a partir de uma comparação entre os termos pesquisados com os termos indexados contidos em documentos.

A decisão para a recuperação de um documento pode ser binária (recuperar ou rejeitar) e pode até estimar o grau de relevância de um documento consultado (SHAFI'I et al., 2014, p. 1). Ferreira (2016, p. 24) reforça que a busca por documentos deve acontecer

por grau de relevância, aceitando documentos que poderiam ser rejeitados em uma pesquisa binária, mas que podem possuir alguma relevância para o usuário.

Shafi'I et al. (2014, p. 1) relata que antes do processo de indexação existe também um processo de pré-processamento dos dados, que consistem em três fases, sendo elas: remoção de caracteres especiais, remoção de palavras e conversão de palavras.

De acordo com Shafi'I et al. (2014, p. 1) e Huang e Zhang (2009, p. 3057) as etapas de pré-processamento são definidas da seguinte forma:

- Na primeira fase são removidos todos caracteres especiais, como acentuação e pontuação e por fim o texto é segmentado em termos.
- Na segunda fase são eliminadas todas *stopwords* existentes na linguagem. Essas palavras podem ser consideradas irrelevantes por sua alta frequência em textos ou por serem consideradas como palavras chaves ou conectores lógicos. Exemplos: as, e, os, de, para, com, sem, foi, *and* , *or*.
- Na terceira fase todo o texto é convertido para caixa baixa (*lowercase*), e em seguida todas as palavras são convertidas para sua raiz (*stemming*), para minimizar o número de *tokens* e consequentemente o tamanho do índice. Essa etapa é normalmente aplicada em técnicas de compactação de índices (SHAFI'I et al., 2014, p. 2).

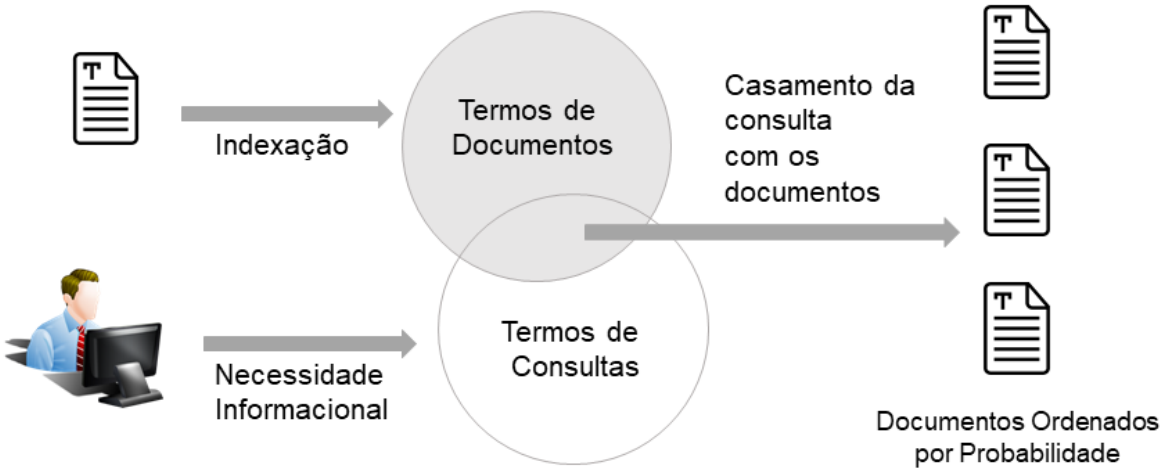
2.2 Sistemas de Recuperação da Informação

Atualmente, Sistemas de Recuperação da Informação são muito utilizados no auxílio a usuários que desejam encontrar informações específicas. O objetivo deste sistema é a recuperação de documentos que possuem conteúdo relevante para uma determinada busca. Os requisitos para SRIs são um conjunto de termos ou expressões que serão utilizados como base para a recuperação de documentos relevantes (SHAFI'I et al., 2014, p. 1).

De acordo com Baeza-Yates, Ribeiro-Neto et al. (1999), o processo de RI é caracterizado por dois componentes: o índice que é responsável por receber como entrada documentos para a indexação e um SRI que é responsável por receber consultas do usuário.

A Figura 2 detalha como é o processo de recuperação, onde o usuário com alguma necessidade informacional realiza uma consulta no sistema, que realiza uma comparação entre os termos utilizados na pesquisa e os termos já cadastrados no processo de indexação, retornando ao usuário um conjunto de documentos que atendam a sua demanda informacional. O sistema realiza um ranqueamento dos resultados encontrados, com base em alguma técnica de Recuperação da Informação, ou seja os documentos encontrados são ordenados em uma lista por relevância através de um calculo probabilístico.

Figura 2 – Processo de Recuperação da Informação em um SRI



Fonte: Adaptado de Ferreira (2016, p. 24)

De acordo com Amazonas et al. (2008) e Ferreira (2016) os modelos de recuperação de informação mais conhecidos são o booleano, o vetorial e o probabilístico.

O modelo booleano traz como resultado para a busca todos os documentos que contém os dados solicitados pelo usuário e permite que sejam utilizados conectores lógicos para interligar os termos como *AND*, *OR* e *NOT*. As pesquisas são formuladas com base nas operações lógicas possibilitando que o usuário formule expressões com termos que deseja encontrar ou termos que não gostaria que fossem encontrados nos documentos.

O modelo vetorial, diferente do modelo booleano, permite que os documentos retornados sejam classificados por um grau de relevância.

> Isto é possível devido à representação de cada documento como vetores no espaço de dimensão n, onde n é o número de palavras contidas em todos os documentos que possuem representatividade no índice. A determinação do grau de relevância pode ser feita com base na distância vetorial entre os documentos e as consultas mapeadas no espaço n-dimensional (AMAZONAS et al., 2008, p. 198).

Por fim, o modelo probabilístico, que também permite retornar uma lista de documentos ordenados por um grau de relevância, pressupõe que um conjunto de documentos pode satisfazer o usuário. Para isto calcula-se o grau de relevância dos termos pesquisados para cada documento, permitindo que seja construído uma lista ordenada de documentos baseada na probabilidade de satisfazer o usuário.

De acordo com Huang e Zhang (2009) o modelo probabilístico representa o estado da arte com relação aos modelos de RI. Esse fato também pode ser constatado a partir de

experimentos realizados por Ferreira (2016) que desenvolveu como projeto um buscador que pudesse comparar os resultados entre os três modelos mencionados anteriormente.

2.3 Gerenciador Eletrônico de Documentos

Fantini et al. (2001, p. 30) diz que um Gerenciador Eletrônico de Documentos tem como objetivo a captura de documentos realizando sua conversão para o meio digital, permitindo que estes sejam capturados, armazenados e indexados, aceitando arquivos de diversos formatos (texto, imagens, páginas HTML, documentos escaneados, formatos multimídia) e também devem assegurar a integridade e reutilização do documento.

Um GED é um conjunto de módulos que tem como propósito final o gerenciamento completo de todos os documentos arquivados por um usuário (Portal GED, 2018). Souza et al. (2013, p. 3) complementa que um GED fornece um meio de gerar, arquivar, consultar, acessar, difundir e recuperar facilmente informações existentes em documentos, além de prometer uma forma ágil e segura de acessar seus arquivos localmente ou em nuvem.

De acordo com o Portal GED (2018) os principais módulos de um GED são:

- ***Capture*/Captura**: acelera processos de negócio através da captação de documentos e formulários, transformando em informações confiáveis e recuperáveis, passíveis de serem integradas a todas as aplicações de negócios;
- ***Document Imaging* (DI)**: é a tecnologia de GED que propicia a conversão de documentos do meio físico para o digital;
- ***Document Management* (DM)**: é a tecnologia que permite gerenciar com mais eficácia a criação, revisão, aprovação e descarte de documentos eletrônicos;
- ***Workflow/BPM***: controle e gerência de processos dentro de uma organização, garantindo que as tarefas sejam executadas pelas pessoas corretas no tempo previamente definido;
- ***COLD/ERM***: tecnologia que trata páginas de relatórios, incluindo a captura, indexação, armazenamento, gerenciamento e recuperação de dados;
- ***Forms Processing*/Processamento de Formulários**: tecnologia que possibilita reconhecer as informações e relacioná-las com campos em bancos de dados, automatizando o processo de digitação (utilização de Reconhecimento Inteligente de Cracteres (ICR) e Reconhecimento Ótico de Caracteres);
- ***Records and Information Management*/Registros e Gerenciamento de Informações (RIM)**: é o gerenciamento do ciclo de vida de um documento, independente da mídia em que se encontre.

2.4 Técnicas para Processamento de Imagens

Para que um documento possa ser armazenado e que suas informações possam ser indexadas possibilitando uma futura busca, é necessário que este arquivo seja convertido para o formato digital. Todo documento físico pode ser convertido para uma versão digital. Segundo o PRODimage Tecnologia (2006) documentos podem ser digitalizados a partir de um scanner, para que possam ser convertidos em imagens.

Por muito tempo existiram limitações para que os documentos além de serem convertidos para forma digital, também se tornassem editáveis, possibilitando a pesquisa e localização de informações dentro deles. Uma pasta cheia de imagens digitalizadas por si só, não apresenta agilidade e eficácia para encontrar um determinado documento, apenas garante a preservação de suas versões físicas.

De acordo com Andrade et al. (2002, p. 5) SRIs utilizam ferramentas de processamento de imagens, com a finalidade da conversão de caracteres gerados de forma mecânica (sendo eles de forma datilográfica ou impressa) em texto digital. Essas ferramentas de OCR realizam o processo de conversão de imagens para caracteres.

Fantini et al. (2001, p. 39) diz que o amadurecimento de tecnologias de reconhecimento como OCR possibilitaram que aplicações realizassem a extração de texto de imagens.

O Portal ABBYY (2018) define OCR como "uma tecnologia que permite converter tipos diferentes de documentos, como papéis digitalizados, arquivos em PDF e imagens capturadas com câmera digital em dados pesquisáveis e editáveis." Diversas ferramentas são oferecidas para que seja realizado OCR de variados tipos de documentos digitais, como: ABBYY[1], Google Cloud Vision[2], Tesseract[3], entre outras.

O processo de conversão ocorre da seguinte forma: (1) o programa analisa a estrutura do documento, dividindo os elementos encontrados na página em blocos de texto, tabelas, imagens, entre outros. (2) As linhas obtidas nestes elementos são quebradas em palavras e em seguida em caracteres. (3) Após a extração, cada caractere é comparado a uma base de dados, onde o programa de OCR verifica por entre dos possíveis símbolos (uma mesma letra pode ser escrita em diferentes fontes por exemplo) qual é o mais semelhante. Após o processamento de todas as possíveis variáveis, o programa finalmente responde com o texto reconhecido (Portal ABBYY, 2018).

[1] Disponível em: https://www.abbyy.com/. Acessado em: 12/10/2018
[2] Disponível em: https://cloud.google.com/vision/?hl=pt-br. Acessado em: 12/10/2018
[3] Disponível em: https://github.com/tesseract-ocr/tesseract. Acessado em: 12/10/2018

2.5 Indexação

Rubi e Fujita (2003, p. 67) afirmam que um bom resultado de busca é totalmente dependente de uma boa indexação, que se trata de um processo anterior. Amazonas et al. (2008) completa dizendo que a indexação de dados incorpora diversos conceitos multidisciplinares de linguística, matemática, psicologia cognitiva, ciência da computação, que se relacionam com as técnicas de recuperação e de interpretação de informações a serem utilizadas.

O processo de indexação se trata de uma etapa de pré-processamento que permite a recuperação de textos (HUANG; ZHANG, 2009). Este processo não se limita apenas a documentos no formato textual, mas também pode ser utilizado para diversos outros formatos de arquivos como vídeos e áudios (ZOBEL; MOFFAT, 2006, p. 3).

Segundo Amazonas et al. (2008, p. 200) para a construção de um índice, onde os dados a serem indexados não são importantes, são definidos os passos a seguir.

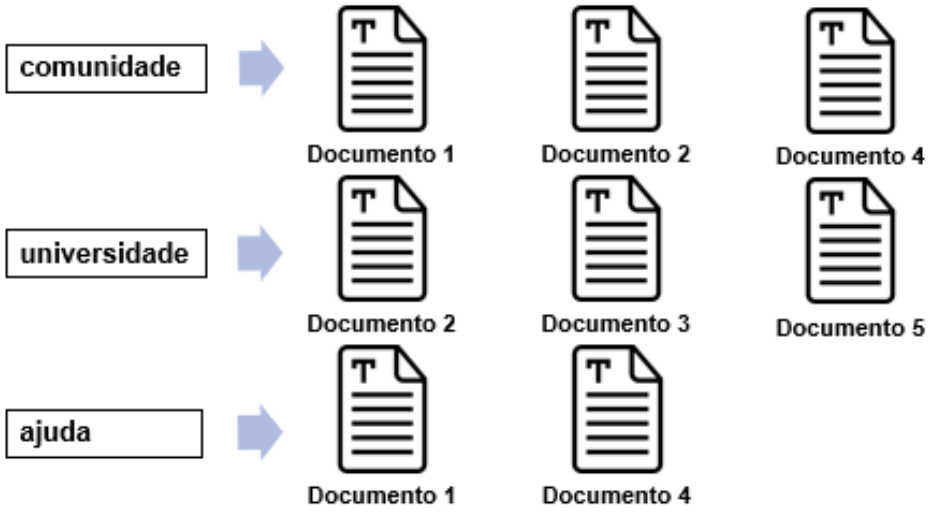
Na primeira etapa nomeada como *Tokenize*, são separados e armazenados todos os *tokens* relevantes, gerando ao final do processo uma lista com todos os *tokens*. Logo a seleção destes é definida pela língua utilizada no documento.

Em seguida na segunda etapa inicia o processo de *Analysis*, onde os *tokens* pouco relevantes são removidos da lista, como artigos, proposições, pontuações, espaços em branco, entre outros (AMAZONAS et al., 2008, p. 200).

Por fim Amazonas et al. (2008, p. 200) conceitua que na terceira etapa chamada de *Stemming*, todos os *tokens* que sobraram da etapa anterior são radicalizados para suas palavras bases. Neste processo são removidos os prefixos, sufixos, plural, bem como flexões verbais e qualquer outra flexão existente na palavra original. Assim se torna possível que seja efetuada a recuperação de palavras derivadas da mesma base ao se realizar uma única consulta.

Existem duas formas de índices que são mais populares, o Índice Invertido e o Índice Sequencial. Ainda de acordo com Amazonas et al. (2008, p. 200) o Índice Invertido consiste no mapeamento de cada token para uma lista de documentos aos quais ele pertence. Um exemplo para este índice pode ser visualizado na Figura 3, onde as palavras a esquerda possuem uma lista de documentos que os possuem detalhadas no lado direito.

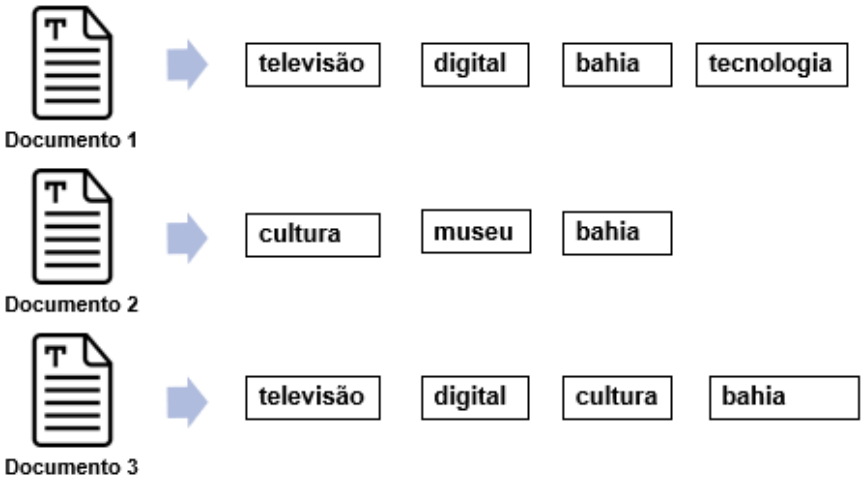
Figura 3 – Exemplo de índice invertido



Fonte: Adaptado de Amazonas et al. (2008, p. 201).

Já o Índice Sequencial se trata de lista de pares (documento, lista de *tokens*), ordenados por documentos. Um exemplo para este tipo de índice pode ser observado na Figura 4 onde no lado esquerdo possui uma lista de documentos, onde cada um aponta para a lista de *tokens* contidas neles.

Figura 4 – Exemplo de índice sequencial



Fonte: Adaptado de Amazonas et al. (2008, p. 201).

2.6 Webstorage

O armazenamento em *webstorage* se trata de um armazenamento realizado em nuvem onde são centralizados todos os dados de um determinado usuário (ASUS, 2016).

Muitas empresas como a Asus, Apple e o Google fornecem serviços de armazenamento em nuvem como o Asus *WebStorage*[4], o iCloud[5] e o Google Drive[6], estes serviços são projetados com o intuito de oferecer diversas vantagens para o usuário, como obter um limite de armazenamento maior que o próprio aparelho físico forneça, oferecer um serviço de segurança para com seus arquivos, compartilhamento eficiente de arquivos, redução de custos, dentre outras (ISBRASIL, 2017).

Algumas dessas plataformas de armazenamento também oferecem uma API para desenvolvedores integrarem suas aplicações com o armazenamento, como é o caso do Google Drive, que sua documentação pode ser consultada na url <https://developers.google.com/drive/>.

2.7 Trabalhos Relacionados

Nessa subseção serão apresentados os principais trabalhos que inspiraram o desenvolvimento deste projeto. Esses trabalhos foram identificados na realização da Revisão Sistemática da Literatura, como será visto na subseção 3.2.1

2.7.1 Implementação e análise experimental de uma máquina de busca a documentos pdf

O trabalho desenvolvido por Ferreira (2016) tem como objetivo apresentar o estado da arte relacionado a Sistemas de Recuperação da Informação com o intuito de implementar um protótipo de um buscador. Nesse processo são abordados conceitos sobre índices de documentos e modelos de Recuperação da Informação.

Nesse trabalho são descritas as atividades realizadas no desenvolvimento do buscador, apresentando como resultados conjunto de testes realizados, onde foi constatado que para o conjunto de documentos utilizado o modelo de recuperação probabilístico apresentou os melhores resultados. Seguindo os resultados de Ferreira (2016), o indexador proposto neste trabalho utiliza o modelo de recuperação probabilístico.

[4] Disponível em: https://www.asuswebstorage.com/navigate/. Acessado em: 15/10/2018
[5] Disponível em: https://www.icloud.com/. Acessado em: 15/10/2018
[6] Disponível em: https://www.google.com.br/drive/apps.html. Acessado em: 15/10/2018

2.7.2 An efficient information retrieval system using query expansion and document ranking

O trabalho desenvolvido por Shafi'I et al. (2014) realiza uma revisão da literatura sobre recuperação da informação e índice invertido com o intuito de aprimorar a técnica de indexação invertida para indexar todos os termos de forma compactada.

Esse trabalho apresenta uma complexidade maior em relação a indexação básica, pois além de compactar os *tokens*, também aborda a similaridade entre palavras e a expansão de consulta avaliando possíveis sinônimos de palavras durante a indexação e a recuperação. É detalhado o processo de pré-processamento de dados que ocorre antes do processo de indexação sendo eles: remoção de caracteres especiais, remoção de palavras e conversão de palavras. O processo de pré-processamento descrito será utilizado como base para o processo de indexação do protótipo, que será modelado sobre as duas primeiras etapas: remoção de caracteres especiais e remoção de palavras.

Por fim é feita uma avaliação de desempenho dos experimentos utilizando as técnicas aplicadas.

2.7.3 Aplicação do gerenciamento eletrônico de documentos: estudo de caso de escolhas de soluções

Nesse trabalho realizado por Fantini et al. (2001) é realizado um estudo sobre as tecnologias com relação ao GED, abordando processos relacionados ao gerenciador, como: digitalização, indexação, análise de aplicações, hardware e até questões legais sobre autenticidade de documentos.

No capítulo de introdução é descrita toda a história que levou até o GED como conhecemos hoje, destacando as dificuldades em lidar com grandes quantidades de papéis e em como o gerenciamento eletrônico pode facilitar e agilizar em processos internos de uma empresa.

No segundo capítulo é feito uma definição teórica sobre diversos conceitos necessários para compreender a grande área que o gerenciamento eletrônico representa e também os assuntos que aborda, como os tipos de documentos, formas de armazenamento, ciclo de vida de um documento, padrões definidos para criação de novos documentos como o ISO 9000 e acessibilidade. Destaca também algumas dificuldades envolvidas no gerenciamento de documentos, que devem ser levadas em consideração no desenvolvimento de uma ferramenta GED completa para empresas.

No terceiro capítulo é feito uma revisão completa sobre o Gerenciamento Eletrônico de Documentos. Iniciando com uma introdução da origem do termo e consequentemente da própria ferramenta. Junto a uma definição de vários autores, descrevendo os formatos

suportados, rotinas, padrões, expectativas, detalhando funcionalidades que são definidas como essenciais para estarem presentes no GED.

O trabalho de Fantini et al. (2001) permitiu um aprofundamento nos conceito envolvidos no funcionamento de um sistema GED completo.

2.7.4 Electronic Records Management - still playing catch-up with paper

Esse artigo aborda uma pesquisa de mercado sobre Gerenciamento Eletrônico de Documentos. A AIIM Market Intelligence (2009), autora desse documento, é uma organização sem fins lucrativos, que fornece educação, pesquisa de mercado e certificação profissional da informação. Foi essa corporação que, em 2000, criou o termo *Enterprise Content Management* (ECM) que se refere a um conceito estratégico, métodos e ferramentas que são utilizadas para gerenciar o ciclo de vida de documentos de organizações.

Nessa pesquisa são avaliados como são gerenciados os documentos empresariais, se os registros eletrônicos são levados a sério e como a utilização do papel vem sendo cada vez mais "precária".

Apesar de um documento antigo, a base dos resultados encontrados sobre registros eletrônicos versus o papel, deixa claro que as empresas se mostraram relutantes na migração, sendo que poucas passaram a utilizar o Gerenciamento Eletrônico e deixaram de utilizar o papel como meio de arquivamento. Sua pesquisa apresenta resultados de como a utilização do papel vem diminuindo ao longo dos anos, motivando a construção de sistemas tecnológicos que possam gerenciar com facilidade esta grande massa de documentos digitais.

2.7.5 Bases para a implantação de um sistema de gerenciamento eletrônico de documento-GED: estudo de caso

Nesse trabalho realizado por Macedo et al. (2003) é descrito como os documentos são utilizados, com o intuito de auxiliar na implementação de um sistema de gerenciamento de documentos. O principal objetivo é mapear como os documentos são utilizados em uma determinada empresa em seus processos, identificando o formato, meio físico, frequência de uso e ciclo de vida de cada documento.

No primeiro capítulo desse trabalho é feita uma introdução sobre a evolução histórica do avanço da tecnologia; No segundo capítulo é apresentado uma conceituação para o GED, detalhando o ciclo de vida de documentos e abordando alguns motivos para empresas investirem no gerenciamento; No terceiro capítulo são listados algumas inovações tecnológicas relacionadas a tecnologia GED; E nos capítulos finais são abordados requisitos tecnológicos para que uma empresa pudesse implantar um gerenciador.

Os requisitos levantamentos para implantação serve como base para avaliar como é o

fluxo de documentos em uma empresa, compreendendo por quais processos um documento passa para aplicar para o protótipo desenvolvido.

2.7.6 Text Indexing and Retrieval

Neste artigo escrito por Huang e Zhang (2009) é feita uma contextualização e caracterização do que é a indexação de documentos, que é a parte central de um sistema de recuperação. Existem muitas técnicas de indexação, como índice invertido, matriz de sufixos e assinatura, sendo que o que mais se destaca atualmente é o de índice invertido, por sua simplicidade e eficiência.

Muitos modelos de recuperação também foram desenvolvidos, como o booleano, vetorial e probabilístico, sendo este último o estado da arte para modelos de recuperação de texto, pela sua alta eficácia em gerar um *rank* de resposta de acordo com a probabilidade dos documentos responderem a uma busca.

Este trabalho apresenta uma boa conceituação para compreender melhor o processo de RI, pois aborda os cálculos realizados para a seleção de documentos relevantes para os métodos de recuperação mencionados anteriormente, que destacam a eficiência da técnica de indexação de índice invertido que será utilizado no desenvolkvimento de protótipo.

2.7.7 Gerenciamento eletrônico da informação: ferramenta para a gerência eficiente dos processos de trabalho

Este artigo escrito por Andrade et al. (2002) aborda o Gerenciador Eletrônico de Documentos como ferramenta para obtenção de RI que fornece resultados de forma rápida e precisa, destacando as características básicas de um Sistema GED e o quanto sua utilização permite uma redução de espaço físico.

Após uma abordagem de técnicas, requisitos, são detalhadas algumas vantagens vistas em um futuro além do papel junto a uma série de características que podem ser utilizadas para a avaliação de produtos GED. Seu trabalho apresenta um referencial teórico para uma melhor compreensão sobre os métodos de RI e GED, que são bases para o desenvolvimento deste projeto.

3 METODOLOGIA

Nesse capítulo são apresentados os materiais e métodos utilizados para o desenvolvimento do trabalho.

3.1 Materiais

Esta seção tem como objetivo fornecer uma visão geral dos materiais, sendo eles de *hardware* e/ou *software*, utilizados no desenvolvimento do indexador de documentos. Serão detalhados equipamentos físicos, tecnologias *web*, linguagens de programação e suas respectivas versões utilizadas, que podem ser devidamente consultadas na Tabela 1 da seguinte forma: primeiramente são descritos os equipamentos físicos, em seguida as configurações de *software* e por fim as tecnologias *web* utilizadas para o desenvolvimento.

Tabela 1 – Materiais e Métodos para desenvolvimento do estudo

CONFIGURAÇÃO DE HARDWARE	
Notebook Asus Modelo x555LF - BRA - xx189T	
Processador x64 Core i5 - 5200 5ª Geração 2.2 GHz	
NVIDIA GeForce GT 930M com 2GB DDR3 VRAM e Intel HD Graphics 5500	
8GB DDR3L1600 MHz SDRAM	
Armazenamento de 1TB 5400R SATA	
CONFIGURAÇÃO DE SOFTWARE	
ITEM	***DESCRIÇÃO***
Sistema Operacional Microsoft Windows 10 Home Single Language x64	Sistema Operacional fornecido pela própria Asus que é a fabricante notebook.
XAMP[1] versão 3.2.2	É um servidor independente de plataforma de código aberto, que fornecesse principalmente serviços MySQL, Apache e interpretadores de linguagem como PHP e Perl.
MySQL Workbench[2] versão 6.3.10	Ferramenta gráfica para modelagem e gerenciamento de banco de dado.
Editor de texto Atom[3] versão 1.30.0	Editor de texto *open source* que permite a instalação de pacotes e interpretadores de linguagem.
Tesseract OCR[4] versão 4.0 beta	Serviço de OCR local *open source*.
Navegador Mozilla Firefox[5] versão 62.0	Browser utilizado para testes da aplicação e responsividade

TECNOLOGIAS WEB DE DESENVOLVIMENTO	
ITEM	**DESCRIÇÃO**
Linguagem de programação PHP[6] versão 7.2.5	Linguagem de programação *server-side* com base em pré-processamento de *scripts*.
Framework de desenvolvimento PHP: Laravel[7] versão 5.6	*Framework* de desenvolvimento PHP.
Composer[8] versão 1.6.5	Gerenciador de dependências para PHP.
jQuery[9] versão 3.2.1	Biblioteca JavaScript utilizada para manipulação do DOM.
Cascading Style Sheets 3 (CSS3)[10]	Mecanismo de estilização simples de páginas *web*.
HyperText Markup Language 5 (HTML5)[11]	Linguagem de marcação padrão de desenvolvimento *client-side* para *web*
Assynchronous JavaScript and XML (AJAX)[12]	Permite que a navegação de páginas *web* seja assíncronas.
Framework de desenvolvimento *front-end* Bootstrap[13] versão 3.3.1	*Framework* HTML, CSS e JavaScript *open source* de desenvolvimento responsivo, *mobile first* para aplicações *web*.
Template Bootstrap *Admin* LTE[14] *Control Panel* versão 2.4	Modelo de interface de painel de controle baseado em Bootstrap 3.
JavaScript Object Notation (JSON)[15]	Notação em formato JavaScript para troca de objetos entre requisições entre máquinas na *web*.

3.1.1 Linguagem PHP

O Pré-processador de Hipertexto conhecido como PHP é uma linguagem interpretada livre que é capaz de gerar conteúdo dinâmico para aplicações *web*. Foi inicialmente projetada em 1994 para o desenvolvimento de aplicações *web* e pelo fato de ser altamente modularizada é indicada para servidores *web*. Atualmente o PHP é usada em 79,1% dos servidores, e é a linguagem de programação mais utilizada em servidores *web* (Portal W3techs, 2018).

[1] Disponível em: https://www.apachefriends.org/pt_br/index.html. Acessado em: 6/4/2018
[2] Disponível em: https://www.mysql.com/products/workbench/. Acessado em: 6/4/2018
[3] Disponível em: https://atom.io/. Acessado em: 6/4/2018
[4] Disponível em: https://github.com/tesseract-ocr/tesseract/wiki/4.0-with-LSTM. Acessado em: 6/4/2018
[5] Disponível em: https://www.mozilla.org/pt-BR/. Acessado em: 6/4/2018
[6] Disponível em: http://www.php.net/. Acessado em: 17/10/2018
[7] Disponível em: https://laravel.com/. Acessado em: 20/10/2018
[8] Disponível em: https://getcomposer.org/. Acessado em: 20/10/2018
[9] Disponível em: https://blog.jquery.com/. Acessado em: 6/10/2018
[10] Disponível em: https://www.w3schools.com/css/. Acessado em: 6/10/2018
[11] Disponível em: https://www.w3schools.com/html/html5_intro.asp. Acessado em: 6/10/2018
[12] Disponível em: http://api.jquery.com/jquery.ajax/. Acessado em: 6/10/2018
[13] Disponível em: https://getbootstrap.com/docs/3.3/. Acessado em: 6/10/2018
[14] Disponível em: https://adminlte.io/. Acessado em: 6/10/2018
[15] Disponível em: https://www.json.org/. Acessado em: 6/10/2018

O código é interpretado no lado servidor pelo módulo PHP, mas também pode ser embutido dentro do HTML para serem visualizados pelo *browser* no lado cliente. O que diferencia o PHP do JavaScript no lado do cliente é que o código é executado no servidor, retornando o HTML que é interpretado pelo o navegador.

Uma grande vantagem em utilizar o PHP é que ele pode ser de fácil aprendizado para um iniciante, porém também oferece recursos avançados para programadores profissionais.

O PHP oferece como paradigma de programação a orientação a objetos, também conhecido como POO, que está presente na linguagem desde sua versão 3. "O termo orientação a objetos significa organizar o mundo real como uma coleção de objetos que incorporam estrutura de dados e um conjunto de operações que manipulam estes dados" (POMARICO, 2018). Esse paradigma permite que o desenvolvedor organize o código em classes. Através da definição de uma classe, descreve-se que propriedades ou atributos o objeto terá, ou seja, os objetos construídos através de classes são uma forma representacional de objetos no mundo real.

Constantemente existem atualizações para solucionar problemas de compatibilidade e segurança, sempre oferecendo suporte e uma extensa documentação.

3.1.2 Dependências e *Plugins*

Esta subseção descreve quais os *plugins* foram utilizados no desenvolvimento do Indexador, abordando o motivo da escolha, sua utilidade, suas dependências, entre outros detalhes relevantes para uma breve conceituação sobre cada um.

3.1.2.1 *Framework* Laravel

O Laravel criado por Taylor Otwell é um *framework* para aplicações *web* com sintaxe expressiva e elegante. Se comparado aos demais *frameworks* de aplicação *web* modernos, o Laravel possui uma das mais extensas documentações. Além disso, o Laracast[16] é um portal voltado para o Laravel e tecnologias *web* que contém cerca de 1100 tutoriais em vídeo abordando assuntos como o Laravel, PHP, JavaScript, entre outros.

Este *framework* é baseado no padrão de projeto *Model-View-Controller*(MVC). O MVC é um padrão de arquitetura que divide uma aplicação em três camadas: a visão (view), o controlador (controller) e o modelo (model). O padrão popularizado atualmente para aplicações *web* foi desenvolvido em 1979 por Trygve Reenskaug sendo que naquela época sua principal finalidade eram as aplicações desktop (MASSARI, 2017).

De acordo com Massari (2017) as principais características da arquitetura MVC são:

[16] http://laracasts.com/. Acessado em: 20/10/2018

- Dividir a aplicação em três camadas: uma para a interface do usuário chamada de *View*, uma controlar o fluxo da aplicação denominada *Controller* e por último uma camada para o controle de dados, regras de negócio, lógica e funções, denominada de *Model*;
- Permitir que as lógicas construídas possam ser reutilizadas por toda a aplicação;
- Ocultar a camada de controle de dados (Modelo) das demais camadas do sistema;
- A camada de Controle tem como função receber as solicitações da camada de visão e convertê-las para ações na camada de Modelo.

3.1.2.2 Sweet Alert

O Sweet Alert é uma substituição responsiva, personalizável e acessível para as caixas de pop-up do JavaScript. Ele se centraliza automaticamente na página em qualquer aparelho navegando pela *web*, seja computador, celular ou tablet.

Para integração deste componente *web* foi utilizado do pacote *uxweb/sweet-alert* configurado para requerer versões iguais ou superiores a 1.4 desta dependência[17].

3.1.2.3 Datatables

O Datatables é necessário para fazer a comunicação entre *server-side* (também conhecido como back-end ou lado servidor, este conceito é utilizado para em aplicações cliente-servidor para definir o processamento no lado servidor) e o plugin JQuery Datatables, que permite a paginação de tabelas e pesquisa dos dados que serão exibidos na tela.

Através de um processamento *server-side*, utilizando requisições AJAX, é possível realizar as buscas e ordenações na tabela de forma bem simples.

Para a integração do componente *web* com o servidor, foi utilizado o pacote *yajra/laravel-datatables-oracle* configurado para requerer versões iguais ou superiores a 8.6 desta dependência[18].

3.1.2.4 Google Drive API

O Drive oferece um conjunto de APIs para ajudar desenvolvedores a realizarem a integração de seus aplicativos privados com o Drive. Sua principal função é a de realizar *download* e *upload* de arquivos para o Google Drive, a plataforma oferece também outras funcionalidades, como pesquisa, compartilhamento, exportação e conversão de arquivos.

[17] Disponível em: https://github.com/uxweb/sweet-alert. Acessado em: 5/10/2018
[18] Disponível em: https://datatables.yajrabox.com/. Acessado em: 5/10/2018

Para a integração do armazenamento em nuvem com o indexador foi utilizado o pacote *google/apiclient* configurado para requerer versões iguais ou superiores a 2.2 desta depêndencia[19].

3.1.2.5 Tesseract OCR

O Tesseract foi desenvolvido na Hewlett-Packard Laboratories Bristol por volta de 1985 a 1994, como mecanismo local *open-source* de OCR. Atualmente está disponível na versão beta 4.0 baseada no LSTM (Long short-term memory ou Unidades de memória de curto prazo longas), que utilizando de redes neurais, pode ser treinada para reconhecer outras novas linguagens.

Para a integração da aplicação com o Indexador foi utilizado o pacote *thiagoalessio/tesseract_ocr* configurado para requerer versões iguais ou superiores a 2.4 desta dependência[20].

3.1.2.6 HP Heaven On Demand

O *HP Heaven On Demand* oferece diversas APIs para processamento textual, indexação, processamento de imagens, entre outros. Uma das ferramentas oferecidas é a de Reconhecimento Óptico de Caracteres, que permite a submissão de diversos tipos de arquivos como pdf e imagens para a extração de texto[21].

Este serviço não possui um pacote PHP referente para ser configurado como dependência, então para isto foi desenvolvido um modulo de comunicação dentro da aplicação. Sempre que um documento precise ser processando utilizando esse método de OCR, seu módulo é instanciado e uma requisição é feita ao servidor da HP, submetendo o documento em uma rota POST e recebendo uma resposta o conteúdo textual presente nele.

3.1.2.7 Google Cloud Vision

O *Google Cloud Vision* permite a integração de ferramentas de visão computacional para o processamento de imagens utilizando modelos de aprendizado de máquina por meio de uma API Rest. A API conta com ferramentas de marcação de imagens, detecção de rostos e pontos de referência, OCR e tags em conteúdo explícito.

Para a integração desta API com o indexador foi utilizado do pacote *thangman22/google-cloud-vision-php* configurado para requerer versões iguais ou superiores a 1.02 desta dependência[22].

[19] Disponível em: https://packagist.org/packages/google/apiclient. Acessado em: 5/10/2018
[20] Disponível em: https://github.com/thiagoalessio/tesseract-ocr-for-php. Acessado em: 5/10/2018
[21] Disponível em: https://dev.havenondemand.com/apis/ocrdocument#overview. Acessado em: 5/10/2018
[22] Disponível em: https://cloud.google.com/vision/docs/ocr. Acessado em: 5/10/2018

3.1.2.8 jQuery

O jQuery se trata de uma eficiente, pequena e rica biblioteca JavaScript. Com ela é possível realizar diversas manipulações no *DOM* em poucas linhas, diferente do JavaScript puro, é possivel também selecionar e manipular elementos no HTML, manipular CSS, aplicar efeitos e animações, navegar pelo *DOM*, utilizar de funções AJAX e aplicar eventos a elementos da página.

Utilizado na versão 3.2.1.

3.1.2.9 Admin LTE

Construído com base em Bootstrap 3, o AdminLTE é um template que oferece um ambiente administrativo simples e com diverços componentes já responsivos.

Utilizado na versão 2.4.

3.1.2.10 Bootstrap 3

O Bootstrap se trata de um *Framework* construído para o desenvolvimento de sites responsivos. Ele é uma biblioteca de código CSS que foi projetado para ser utilizado em qualquer tamanho de tela, exibindo uma interface que se adapta a medida que a tela aumenta/diminui.

Utilizado na versão 3.3.1 não se trata da última versão disponibilizada, porém o template utilizado tem como dependência o Bootstrap 3.

3.1.2.11 DropzoneJS

DropzoneJS é uma biblioteca de código aberto que possibilita realizar upload de arquivos com *drag'n'drop* (arrastar e soltar). Permite controlar como os arquivos serão enviados para o servidor e ainda utilizar de templates para customizar a área de upload.

Utilizado da versão 5.2.0[23].

3.1.2.12 FontAwesome

Para que pudessem ser utilizados ícones e símbolos pela página, independente da plataforma onde se acessa o site, são utilizados de ícones vetoriais. O fontAwesome funciona semelhante a uma fonte, porém em vez de se utilizar caracteres são utilizados ícones. Permitindo a manipulação CSS de seus elementos pela página, alterando cores, tamanhos e posições.

Utilizado na versão 4.7[24].

[23] Disponível em: https://www.dropzonejs.com/. Acessado em: 6/10/2018. Acessado em: 6/10/2018
[24] Disponível em: https://fontawesome.com/v4.7.0/. Acessado em: 6/10/2018

3.2 Métodos

Com o intuito de controlar melhor o desenvolvimento do presente trabalho, foi construído uma Estrutura Analítica do Projeto, que se trata de uma ferramenta utilizada por gerentes de projeto para fracionar todo o trabalho em pequenas partes, separando tarefas em diversas fases (AECWEB; RABECHINI, 2018).

Consultando uma EAP é possível ter uma ampla visão do projeto do início ao fim, permitindo controlar as etapas a serem seguidas no processo de desenvolvimento, evitando que sejam inseridas alterações no escopo das tarefas após a aprovação e definição das etapas (BUCHTIK, 2013).

Embora esse trabalho não tenha sido realizado em equipe, a EAP permite ainda que, em equipes com diversos desenvolvedores, todos podem ter uma visão geral de todo o trabalho que está sendo feito, permitindo uma melhor comunicação entre os colaboradores através de uma compreensão comum do projeto. Além disso, também se trata de uma documentação do produto desenvolvido, destacando o tempo gasto em cada etapa, o que é extremamente importante para levar em conta o custo gasto em seu desenvolvimento (BUCHTIK, 2013).

As etapas previstas no processo de desenvolvimento do indexador podem ser vistas na Tabela 2, onde foram definidas 6 macro entregas com seus respectivos pacotes de trabalhos.

A primeira macro entrega se trata da preparação e do planejamento necessário para que este projeto fosse desenvolvido. Nesta etapa é realizado uma revisão sistemática junto a uma avaliação dos resultados encontrados, com o intuito de conceituar-se sobre o assunto abordado, conhecendo técnicas e trabalhos relacionados.

Na segunda macro entrega são realizadas implementações sobre os algoritmos encontrados em artigos relacionados, que abordam o processo de indexação e busca.

Após um estudo sobre o assunto e testes sobre os algoritmos existentes que apresentam uma solução para o processo de indexação é definido uma macro entrega para a preparação do desenvolvimento, preparando o ambiente, definindo a estrutura hierárquica do projeto, configurando o *template* e a modelagem do banco de dados.

Na quarta macro entrega é feito toda a implementação no sistema, implementado todas as funcionalidades definidas anteriormente. Esta é a etapa que demanda o maior tempo neste projeto, e consequentemente a que possui o maior número de pacotes de trabalhos.

As duas últimas macro entregas são referentes aos testes no sistema. Todas as funcionalidades são testadas para verificar se não existem erros que possam interferir para que um determinado usuário possa interagir. Prevendo que podem existir erros em

determinadas rotinas do sistema, é definido uma etapa para a correção destes.

Tabela 2 – Estrutura Analítica do Projeto (EAP).

Item	Macro Entrega	Subitem	Pacote de Trabalho
1	Preparação e planejamento	1.1	Revisão sistemática
		1.2	Avaliação sistêmica sobre os resultados da revisão sistemática
2	Estudo e replicação de algoritmos de indexação e busca	2.1	Estudo e testes sobre o funcionamento do algoritmo de indexação
		2.2	Estudo e testes sobre o funcionamento do algoritmo de busca
3	Preparação para desenvolvimento	3.1	Preparação do ambiente de desenvolvimento
		3.2	Definição da estrutura do projeto
		3.3	Modelagem do *template*
		3.4	Modelagem do banco de dados
4	Desenvolvimento do projeto	4.1	Criação do *CRUD* de Usuários
		4.2	Criação de views e JavaScript para o indexador de documentos
		4.3	Implementação de métodos de OCR
		4.4	Implementação de métodos de armazenamento
		4.5	Implementação do indexador
		4.6	Implementação de filas de processamento para indexação
		4.7	Implementação/Adaptação do buscador
		4.8	Implementação do gerenciador
5	Testes no sistema	5.1	Testes em *upload* de arquivos
		5.2	Testes em armazenamento de arquivos
		5.3	Testes em seleção de configurações do sistema
		5.4	Testes da indexação a partir do buscador
6	Correções de possíveis erros	6.1	Correção de erros encontrados nos testes no sistema

3.2.1 Revisão Sistemática

Neste pacote de trabalho estava previsto uma pesquisa sobre o estado da arte relacionado aos temas de estudo: OCR; Indexador; *Webstorage*; Gerenciador Eletrônico de Documentos.

3.2.1.1 Introdução

Uma revisão sistemática é uma forma de pesquisar na literatura sobre temas específicos. A partir destas pesquisas é possível acessar informações relacionadas sobre determinado assunto, assim como filtrar os dados encontrados por parâmetros definidos no processo (RF, 2007).

De acordo com RF (2007) o processo de pesquisa utilizando a revisão sistemática "disponibiliza um resumo das evidências relacionadas a uma estratégia de intervenção específica, mediante a aplicação de métodos explícitos e sistematizados de busca, apreciação crítica e síntese da informação selecionada."

O autor ainda acrescenta que a grande vantagem na utilização deste processo de pesquisa é poder assimilar todos os trabalhos relacionados a um determinado tema. Tendo acesso aos resumos para uma avaliação detalhada, é possível observar a evolução da pesquisa ao longo do tempo. Sento este um fator importante para direcionar o pesquisador na direção que pode contribuir de algum modo ao estado da arte do tema abordado.

3.2.1.2 Planejamento

O objetivo da fase de planejamento é identificar e descrever as ferramentas de apoio ao processo de gerenciamento eletrônico de documentos com a utilização de um indexador e *webstorage*. Organizando e analisando tais ferramentas no intuito de especificar um conjunto de requisitos para o desenvolvimento do protótipo proposto.

Para isto foram definidas algumas questões de pesquisa, onde o foco de cada pergunta é de auxiliar a buscar por artigos que abrangem pontos específicos sobre ferramentas de apoio ao gerenciamento eletrônico de documentos realizando comparações, análises e sugerindo requisitos para desenvolvimento de um novo protótipo.

- O que é um GED e quais suas vantagens?
- Quais são as principais tecnologias de um GED?
- Como se deu a evolução até a formação de um GED?
- O que é um indexador de documentos?
- O que é a técnica de índice invertido?
- Quais as etapas de um indexador utilizando índice invertido?
- O que é *Webstorage*?

Apenas documentos em inglês e português foram selecionados, onde estes devem estar disponibilizados na *web*, abordar assuntos da Ciência da Computação ou áreas de tecnologia e descrever ferramentas de suporte ao gerenciamento de documentos eletrônicos.

Essa busca foi realizada por meio do Portal de Periódicos da CAPES[25] através de uma pesquisa nas bases a seguir:

[25] Disponível em: http://www.periodicos.capes.gov.br/. Acessado em: 27/4/2018

- Scopus[26];
- SpringerLink[27];
- Elsevier ScienceDirect[28];
- El Compendex[29];
- Google[30];
- Outros, onde foram encontrados em fontes diversificadas que foram inseridos para avaliar se seu conteúdo apresentava alguma relevância para a pesquisa.

Por fim foram definidos por quais as palavras chaves seriam pesquisadas nas bases de dados, que são descritas na Tabela 3.

Tabela 3 – Lista de palavras chaves da revisão sistemática

Termo 1	**Termo 2**
Document Indexer	
Electronic Documento Management	
Web Storage	
Webstorage	
Cloudstorage	
Inverted Index	Document
Gerenciador Eletrônico de Documentos	GED
Soluções	GED
Gestão de Documentos e Arquivísticas	
Gerenciador Eletronico de Documentos	
EDM	

3.2.1.3 Execução

Após a elaboração e execução do protocolo, são selecionados os artigos com potencial conteúdo relevante para conceituação do assunto. Inicialmente é realizada uma busca nas bases pela combinação das palavras chaves pré-selecionadas que pode ser visualizada na Tabela 3, os resultados são transcritos e serão posteriormente filtrados. Como a quantidade de resultados encontrados pode ser imensa (como visto na Tabela 4), é impossível que todos sejam analisados. Para isto foram contabilizados no máximo 100 registros mais relevantes de cada busca como pode ser visto na Tabela 5, por motivos de limitação de download das bases e para tornar possível a leitura de um grupo mais relevante de artigos.

[26] Disponível em: http://www.scopus.com. Acessado em: 27/4/2018
[27] Disponível em: http://www.springerlink.com/. Acessado em: 27/4/2018
[28] Disponível em: http://www.sciencedirect.com. Acessado em: 27/4/2018
[29] Disponível em: http://www.engineeringvillage2.org. Acessado em: 27/4/2018
[30] Disponível em: http://www.google.com.br. Acessado em: 27/4/2018

Na primeira etapa foi analisado se existiam artigos repetidos entre todos os 2683 títulos, visto que é possível que um termo de busca possa resultar em possíveis resultados iguais em diversas bases ou diferentes termos fazerem referência a um mesmo artigo.

Na segunda etapa foram avaliados os títulos dos 2134 artigos, todos aqueles que os nomes não tinham relações com as perguntas definidas no escopo eram removidos da lista, quando evidentemente seu conteúdo abordava outras tecnologias.

Na terceira etapa foi realizado uma leitura do resumo dos 93 artigos que foram selecionados na etapa anterior. Através desta leitura é possível conhecer com maior riqueza de detalhes qual o foco principal do artigo. Sendo que resumos que a contextualização fugia do escopo a ser respondido pelo protocolo eram removidos e não seguiam para a próxima etapa.

Na quarta e última etapa é realizado uma leitura completa dos 27 artigos finalistas. Os artigos selecionados nesta etapa foram classificados como um referencial para o desenvolvimento desse trabalho de conclusão de curso, contendo técnicas e embasamento teórico para uma contextualização sobre o projeto proposto.

Tabela 4 – Totais de artigos encontrados inicialmente nas buscas.

Filtros: Ciência da Computação, Inglês; Relevância;							
Termo 1	**Termo 2**	**Scopus**	**Springer**	**ScienceDirect**	**Compendex**	**Google**	**Total**
Document Indexer		0	19	1,132	1	54,900	56052
Electronic Document Management		114	139	264	205	17,800	18522
Web Storage		49	194	82	51	2,200,000	2200376
Webstorage		2	31	10	3	7,170	7216
Cloudstorage		3	10	5	5	23,600	23623
Inverted Index	Document	313	543	619	197	70,900	72572
Gerenciador Eletronico de Documentos	GED	0	0	0	0	2,030	2030
Soluções	GED	0	0	10	0	2,940	2950
Gestão de Documentos e Arquivística		0	0	0	0	1	1
Gerenciador Eletronico de Documentos		1	0	0	0	13,600	13601
EDM		273	161	18,441	9,670	146,000	174545
Total		755	1,097	20,563	10,132	2,538,941	2,571,488

Tabela 5 – Totais de artigos avaliados inicialmente nas buscas.

Filtros: Ciência da Computação, Inglês; Relevância;							
Termo 1	**Termo 2**	**Scopus**	**Springer**	**ScienceDirect**	**Compendex**	**Google**	**Total**
Document Indexer		0	19	100	1	100	220
Electronic Document Management		100	100	100	100	100	500
Web Storage		49	100	82	51	100	382
Webstorage		2	31	10	3	100	146
Cloudstorage		3	10	5	5	100	123
Inverted Index	Document	100	100	100	100	100	500
Gerenciador Eletrônico de Documentos	GED	0	0	0	0	100	100
Soluções	GED	0	0	10	0	100	110
Gestão de Documentos e Arquivística		0	0	0	0	1	1
Gerenciador Eletrônico de Documentos		1	0	0	0	100	101
EDM		100	100	100	100	100	500
Total		355	460	507	360	1,001	2,683

A execução do protocolo, filtrando artigos pelas 4 etapas a partir das regras definidas resultaram nos resultados da Tabela 6, sendo 8 o número de artigos selecionados após a RSL.

Tabela 6 – Quantidade de artigos restantes após cortes

ETAPA	**QUANTIDADE**
Todos os Títulos	2683
Etapa 1 - Repetidos	2134
Etapa 2 - Leitura dos Títulos	93
Etapa 3 - Leitura dos Resumos	27
Etapa 4 - Leitura Total	7

Após uma leitura completa dos artigos resultantes da etapa 4, os 7 artigos que podem ser visualizados na Tabela 7 foram selecionados como referencial base para a conceituação teórica deste trabalho e são melhor detalhados na seção 2.7.

Tabela 7 – Artigos resultantes da RSL

Título	**Autores**	**Ano**	**Fonte**
Implementação e análise experimental de uma máquina de busca a documentos pdf	Roger Ferreira	2016	Outros
An efficient information retrieval system using query expansion and document ranking	Janarthanan, Kavitha, Victor, Rajkumar	2014	Scopus
Aplicação do gerenciamento eletrônico de documentos: estudo de caso de escolhas de soluções	Fantini	2001	Google
Electronic Records Management - still playing catch-up with paper	AIIM Market Intelligence	2009	Outros
Bases para a implantação de um sistema de gerenciamento eletrônico de documento-GED: estudo de caso	Macedo	2003	Google
Text Indexing and Retrieval	Haoda Huang, Benyu Zhang	2009	Springer
Gerenciamento eletrônico da informação: ferramenta para a gerência eficiente dos processos de trabalho	MVM Andrade	2002	Google

4 PROTÓTIPO DESENVOLVIDO

O protótipo desenvolvido foi implementado com as principais etapas de uma ferramenta GED, possibilitando o OCR, a indexação, armazenamento e busca dos arquivos submetidos para o sistema. Para que estas funcionalidades sejam utilizadas pelos usuários são necessárias algumas configurações iniciais. O sistema inicialmente precisa de um usuário do tipo Administrador para definir as configurações iniciais de um usuário do tipo Cliente. Isto se deve ao fato que existem varias ferramentas que fornecem o reconhecimento de caracteres e também pelas duas formas de armazenamento.

A partir do momento que um Cliente se regista e um Administrador define suas configurações de OCR e armazenamento, é possível utilizar as funcionalidades do Indexador. O usuário pode realizar o *upload* de documentos, que poderão ser buscados a partir do indexador.

Ao realizar um *upload*, o documento será processado, onde será feita a extração do texto, em seguida indexado e por fim armazenado. O armazenamento poderá ser feito localmente ou no Google Drive.

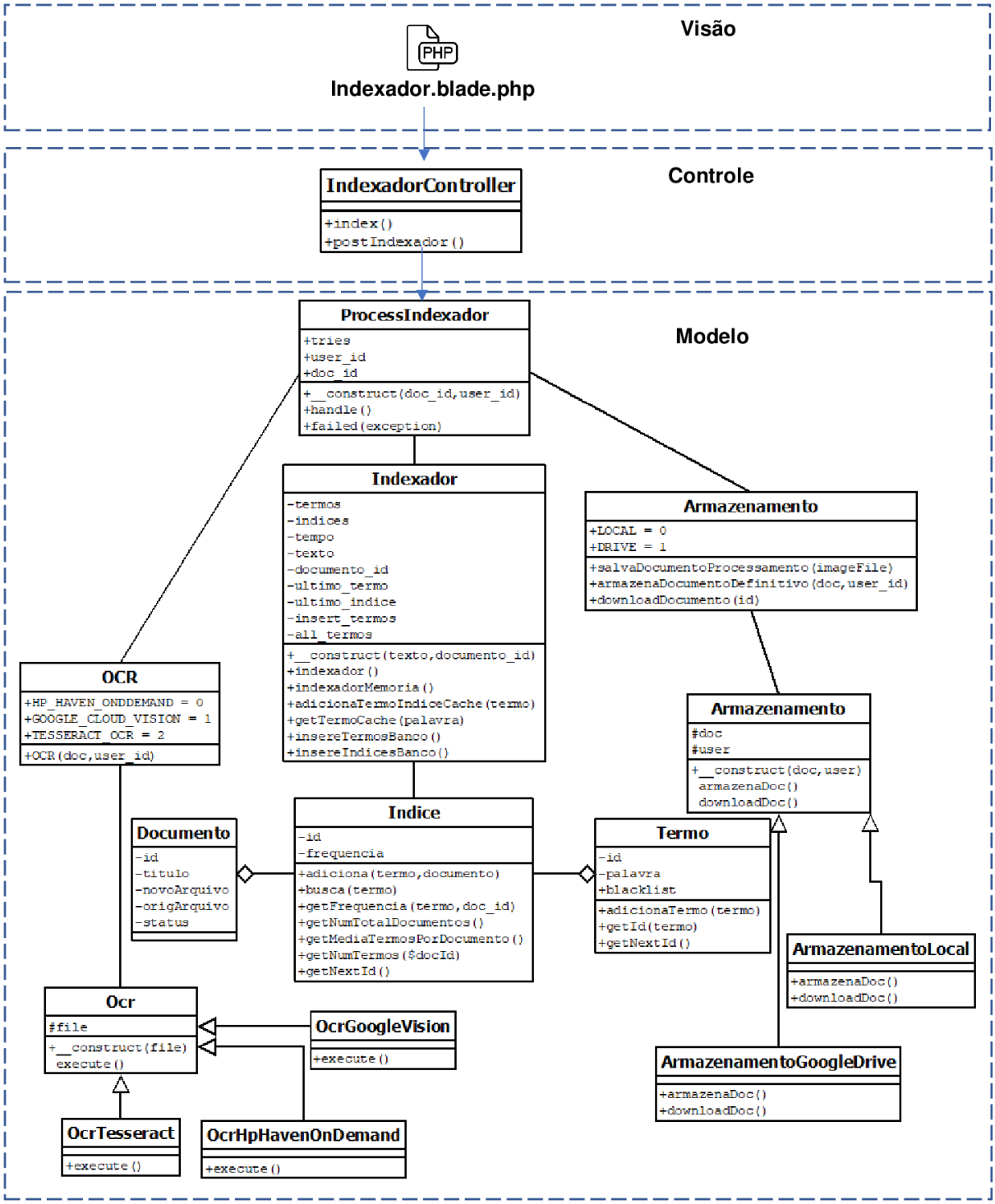
Após o processo de indexação ser concluído o documento pode ser recuperado através do buscador, que foi implementado para validar se a indexação está sendo realizada com sucesso e fornecer uma forma de recuperar os arquivos armazenados pelo usuário.

O projeto completo está disponível em <https://drive.google.com/drive/folders/1Qc5XSZDDBRrAOcq1Inh-lR7_2nEm4xtp?usp=sharing>.

4.1 Diagrama de classes

A Figura 5 apresenta o diagrama de classes UML do Indexador proposto, nele são representadas apenas as classes envolvidas na parte de indexação que é referente ao foco deste trabalho. Nele são contemplados os 2 métodos de armazenamento utilizado: local e Google Drive e os 3 métodos de OCR, onde ambos são definidos nas configurações de cada usuário.

Figura 5 – Diagrama de classes da hierarquia do Indexador



O esquema hierárquico utilizado foi construído sobre o padrão utilizado pelo Laravel, o MVC. Por isto as classes são divididas entre três pacotes: Visão, Controle e Modelo. No pacote de Visão apenas o arquivo *indexador.blade.php* é utilizado como visão, sua função é prover uma interface de comunicação entre o cliente e o servidor da aplicação.

A partir do momento em que o usuário interage com a interface, ele dispara os métodos da classe *IndexadorController* na camada de Controle através de requisições AJAX.

Assim que a camada de Controle tem o método disparado ela encaminha a requisição para a camada de Modelo onde é instanciado uma *Queue* nomeada como *ProcessIndexador* que irá realizar o processo de indexação de forma sequencial e paralela com relação a aplicação, ou seja, o usuário não ficará preso na tela aguardando o termino da indexação, e sim será notificado assim que ela for finalizada.

Através das configurações do sistema selecionadas para o usuário a *Queue ProcessIndexador* ficará encarregada de instanciar as devidas classes de OCR e Armazenamento para que o processo de indexação ocorra.

4.2 Rotas

Todas as rotas da aplicação *web* no Laravel são definidas no arquivo *routes.php* que é carregado automaticamente pelo *framework* e são tratadas internamente pelo mecanismo *Routing*[1]. Toda vez que o usuário realiza a iteração com a interface gráfica uma requisição GET ou POST é capturada pelo Laravel e será tratada no arquivo de rotas. Um exemplo de um arquivo de rotas pode ser visualizado na Figura 6.

Os eventos acionados por cada rota podem ser roteados pelos Controles, conforme exemplificado na Figura 6. Ao criar-se uma nova rota pelo método *Route::get()* é necessário informar 2 parâmetros, sendo o primeiro a rota e a segunda uma *string* composta pelo Controle e a função que irá processar a requisição, estes dois valores são separados pelo carácter @.

[1] Disponível em: https://laravel.com/docs/5.6/routing. Acessado em: 20/10/2018

Figura 6 – Exemplo de Rotas

```php
<?php

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/

Route::get('/', function () {
    return Redirect('/login');
});

Route::group([ 'middleware' => 'admin', 'prefix' => 'admin'], function () {
    Route::get('usuario/', 'UserController@index');
    Route::get('usuario/{id}/edit', 'UserController@edit');
    Route::get('usuario/{id}/delete', 'UserController@destroy');
    Route::post('usuario/{id}/edit/pessoa', 'UserController@updatePessoa');
    Route::post('usuario/{id}/edit/config', 'UserController@updateConfig');
});
```

4.3 Camada de Modelo

O Laravel torna a interação do desenvolvedor com o banco de dados simples, utilizado de seu construtor de consultas em SQL Bruto e o *Eloquent ORM*.

O Laravel suporta nativamente quatro bancos de dados: MySQL, PostgreSQL, SQLite e servidor SQL, onde o próprio desenvolvedor tem total liberdade para escolher qual deseja utilizar na aplicação. Para realizar esta configuração é necessário apenas configurar no arquivo *config/database.php* com o SGBD desejado.

Com as migrações, o Laravel permite também controlar todo o versionamento do banco de dados através do próprio *framework*, permitindo que uma equipe possa trabalhar em trabalho colaborativo sem afetar uns aos outros. Cada *migration* é referente a uma tabela no BD e são versionadas separadamente, permitindo que um desenvolvedor possa atualizar uma tabela sem interferir nas outras. Um exemplo de migration pode ser visualizado na Figura 7.

Figura 7 – Exemplo de *Migration*

```php
<?php

use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class CreateTermosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('termos', function (Blueprint $table) {
            $table->increments('id');
            $table->string('palavra', 100);

            $table->timestamps();
            $table->softDeletes();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('termos');
    }
}
```

Para que não fosse necessário que a cada alteração no BD todos os dados precisassem ser recadastrados, o *framework* Laravel fornece um método para semear o Banco de dados utilizando dos *Seeders* que tem como propósito agilizar o processo de população do banco de dados. Um exemplo de Seed pode ser visualizado na Figura 8.

Figura 8 – Exemplo de *Seed*

```php
<?php

use App\Models\UserConfig;
use Illuminate\Database\Seeder;

class UserConfigTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        factory(UserConfig::class)->create([
            'tipo_ocr'           => 0,
            'tipo_armazenamento' => 1,
        ]);

        factory(UserConfig::class)->create([
            'tipo_ocr'           => 0,
            'tipo_armazenamento' => 0
        ]);

        factory(UserConfig::class)->create([
            'tipo_ocr'           => 1,
            'tipo_armazenamento' => 1
        ]);
    }
}
```

4.4 Camada de Visão

O *framework* Laravel oferece como componente para criação de *views* o motor de *template Blade*[2], que se trata de uma linguagem desenvolvida para criação de blocos de *layout* que possibilitam a criação dinâmica de conteúdo.

Os códigos projetados nos arquivos *Blade* são escritos em HTML de uma forma diferente. Além do código em HTML é possível acrescentar código em PHP que ao ser interpretado pelo *framework* será convertido todo para PHP e então interpretado pelo navegador.

[2] Disponível em: https://laravel.com/docs/5.6/blade. Acessado em: 20/10/2018

O template *Blade* permite que códigos PHP como comandos estruturais e de repetição como *if, else, while, for e foreach* sejam inseridos no escopo do arquivo *.blade.php* apenas utilizando o "@"como prefixo para o comando. Isto é de grande vantagem para páginas construídas dinamicamente, que dependem de valores cadastrados em um banco de dados.

O *Blade* oferece a possibilidade de herança de *templates*, que permite que um conteúdo comum entre várias telas seja escrito apenas uma vez, evitando repetição de código. Essas herança são muito comuns em elementos como barras de laterais, cabeçalhos, rodapés, menus entre outros. Um exemplo de arquivo *Blade* pode ser visto na Figura 9.

Figura 9 – Exemplo de arquivo *Blade*

```
1  <!-- Heran a de layout -->
2  @extends('layouts.layout')
3  <!-- Inser   o de conteudo em se   o -->
4  @section('title', 'GED - ')
5  <!-- Inser   o de conteudo em se   o -->
6  @section('content')
7    <section class="content-header">
8      <h1>GED <small>Gerenciador Eletr nico de Documentos</small></h1>
9      <ol class="breadcrumb">
10       <li class="active"><i class="fa fa-dashboard"></i> Home</li>
11     </ol>
12   </section>
13   <section class="content">
14
15   </section>
16 @endsection
```

4.5 Camada de Controle

Para que a implementação de eventos acionados ao acessar determinadas rotas no arquivo *routes.php* não sejam todos configuradas em um mesmo arquivo, os *Controllers*[3], ou Controles são classes que permitem que o desenvolvedor implemente os métodos públicos que serão ações referentes as rotas da aplicação. Essa camada permite processar as interações do usuário com a aplicação e traduz os comandos para que os Modelos possam processa-los. A Figura 10 representa um exemplo de *Controller* utilizado no sistema.

[3] Disponível em: https://laravel.com/docs/5.6/controllers. Acessado em: 20/10/2018

Figura 10 – Exemplo de *Controller*

```php
<?php

namespace App\Http\Controllers;

use App\Armazenamento;
use Illuminate\Http\Request;
use App\Jobs\ProcessIndexador;
use Illuminate\Support\Facades\Auth;

class IndexadorController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index() {
        return View('cliente.indexador');
    }

    public function postIndexador(Request $request) {
        if($request->hasFile('file')) {
            $files = $request->file('file');
            foreach ($files as $file) {
                $doc = Armazenamento::salvaDocumentoProcessamento($file);
                ProcessIndexador::dispatch($doc->id, Auth::user()->id);
            }
        }

        return 'sucess';
    }
}
```

4.6 Autenticação e Autorização

O Laravel fornece um mecanismo de segurança chamado *middleware*[4] que permite filtrar solicitações HTTP que são feitas na aplicação. Por padrão o Laravel fornece o *middleware auth* que verifica se o usuário está autenticado para acessar determinadas páginas do sistema. Neste caso quando um usuário realiza uma requisição GET e ele ainda não está logado ele é redirecionado para a tela de login, porém se o usuário já foi autenticado na aplicação o *middleware* permitirá que a solicitação continue a ser executada.

[4] Disponível em: https://laravel.com/docs/5.6/middleware. Acessado em: 20/10/2018

Um exemplo de *middleware* aplicado nas rotas da aplicação pode ser visto na linha 18 do arquivo de rotas na Figura 6, onde as 5 rotas nas linhas seguintes só poderão ser acessadas por usuários que possuem permissão de Administrador do sistema.

Para controlar o acesso para os dois tipos de usuários previsto no sistema, sendo eles Cliente e Administrador, foram implementados dois *middleware* que permitem que apenas usuários com os respectivos tipos acessem as suas rotas permitidas. O código fonte do *middleware* de permissão de Administrador pode ser visto na Figura 11 e o de permissão para usuários Cliente pode ser visto na Figura 12.

Figura 11 – *Middleware* de autorização de usuários Administradores

```
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Closure;
6  use Illuminate\Support\Facades\Auth;
7
8  class Admin
9  {
10     /**
11      * Handle an incoming request.
12      *
13      * @param  \Illuminate\Http\Request  $request
14      * @param  \Closure  $next
15      * @return mixed
16      */
17     // Tipo 0 = Administador
18     public function handle($request, Closure $next)
19     {
20         if (Auth::check() && (Auth::user()->tipo_usuario == 0)) {
21             return $next($request);
22         };
23
24         return redirect('home');
25     }
26 }
```

Figura 12 – *Middleware* de autorização de usuários Clientes

```php
<?php

namespace App\Http\Middleware;

use Closure;
use App\Models\Documento;
use App\Models\Notificacao;
use Illuminate\Support\Facades\Auth;

class Cliente
{
    /**
     * Handle an incoming request.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  \Closure  $next
     * @return mixed
     */
    // Tipo 1 = Cliente
    public function handle($request, Closure $next)
    {
        if (Auth::check() && (Auth::user()->tipo_usuario == 1)) {
            $this::atualizaSessao();
            return $next($request);
        };

        return redirect('home');
    }

    public static function atualizaSessao() {
        session(['qtdNotificacoes' => Notificacao::
                    where('user_id', '=', Auth::user()->id)->count(),
                'qtdDocumentos' => Documento::
                    where('user_id', '=', Auth::user()->id)->count()]);
    }
}
```

4.7 Processo de Indexação

O processo de indexação como já mencionado anteriormente se trata de uma das partes mais importantes de um GED. A metodologia utilizada para sua implementação será descrita nas subseções a seguir.

4.7.1 Pré armazenamento

A partir do momento em que um documento é carregado para o sistema, a primeira etapa do processamento é arquivar esta imagem em um diretório temporário. Essa etapa existe para que a ação de *upload* de um arquivo possa ser a mais rápida possível e que a interface do usuário não seja travada até que todo o processamento seja realizado.

A etapa de pré armazenamento realiza uma cópia do arquivo e instancia uma fila que irá realizar todo o processo de indexação em segundo plano. Como pode ser visto na linha 25 da Figura 10 é salvo localmente uma cópia do documento e também um registro de seus respectivos dados, que são armazenado no banco de dados para o futuro processamento. Em seguida na linha 26 é disparado uma *queue* que recebe como entrada o *id* do documento que acaba de ser armazenado e qual o usuário realizou o *upload*.

Após ser disparado uma *queue* para cada documento, a requisição POST é respondida e o usuário poderá interagir com o sistema normalmente e a medida que os arquivos forem sendo indexador seu status será alterado e também será possível consulta-los através do buscador.

4.7.2 Fila de processamento de indexação

Quando existem tarefas que exigem um processamento elevado e que demandam muito tempo para serem concluídas, não é convencional deixar que o usuário fique aguardando seu término para continuar a interagir com o sistema. Para isto o Laravel fornece um mecanismo de filas chamado também de *Queues*[5].

As *queues* permitem que o processamento de tarefas demoradas sejam adiadas e processadas em segundo plano, onde podem ser disparados eventos de notificação para o usuário estar ciente que determinado processo foi finalizado, como por exemplo o envio de um email ou uma notificação na própria aplicação.

Para que as filas sejam utilizadas no Laravel é necessário que o método no qual serão processadas seja previamente configurado no arquivo de configuração *.env* que por padrão é definido que o banco de dados padrão da aplicação irá tratar estas filas de processamento.

Já prevendo que um usuário pode submeter vários arquivos ao mesmo tempo e que o processo de indexação pode ser custoso computacionalmente, a implementação de uma *queue* foi realizada para que todo o processo de indexação fosse sequencial, onde apenas um único documento seria indexado por vez.

O processo de indexação é composto por algumas etapas que serão descritas nas subseções a seguir.

[5] Disponível em: https://laravel.com/docs/5.7/queues. Acessado em: 20/10/2018

4.7.2.1 OCR

A primeira etapa de processamento é o OCR, que pode ser realizado através de três formas neste sistema: pela API do Google Cloud Vision, pela API do Hp Heaven OnDemand ou pela ferramenta TesseractOCR.

A ferramenta que irá processar essa etapa depende das configurações de sistema do usuário logado. O método de processamento pode ser alterado por algum administrador do sistema ao editar as configurações de um usuário, permitindo que cada usuário utilize um método de OCR diferente. Esta etapa é simples e o tempo de processamento depende diretamente da quantidade de folhas, palavras e qualidade das imagens.

O fato de que os métodos de Reconhecimento Óptico possam ser chaveados permite que no momento em que um usuário deseja se registrar na aplicação, sejam oferecidos diversos pacotes, que podem ser tarifados de forma diferente, de acordo com a precisão de OCR.

4.7.2.2 Indexação

Após o OCR do documento a ser processado inicia-se o processo de indexação. Conforme descrito na seção 2.5, o processo de indexação ocorre em três etapas, sendo que a terceira é opcional para compactação de índices no banco de dados.

A primeira etapa, chamada de *tokenize* é feita a análise léxica de todo o texto, nesta etapa todas os caracteres são reescritos com letras minúsculas e todos os caracteres especiais são removidos, como acentuação e cedilha.

Na segunda etapa, chamada de *Analysis*, todas as palavras sem significado na linguagem natural são removidas, estas palavras são conhecidas como palavras de parada ou *stopwords*. As *stopwords* utilizadas podem ser visualizadas na Figura 13 e foram adaptadas do projeto proposto por Ferreira (2016).

Por fim todas as palavras são adicionadas aos índices do documento no banco de dados, estes índices contém a informação de qual é a palavra referente e qual a frequência em que ele aparece no documento.

Figura 13 – Lista de *stop-words* utilizada

```
1 public static $blacklist = [
2       'a', 'e', 'i', 'o', 'u',
3       'an', 'as', 'os', 'na', 'ne', 'ni', 'no', 'nu',
4       'da', 'de', 'do', 'por', 'se', 'das', 'dos',
5       'em', 'porque', 'para', 'como', 'pelo', 'pela', 'com',
6       'um', 'uma', 'que', 'ao', 'and', 'at', 'et', 'al',
7       'an', 'in', 'of', 'to', 'the', 'as', 'be', 'it', 'nao', 'ou',
8 ];
```

4.7.3 Armazenamento definitivo

Após o processo de indexação e a partir da configuração de armazenamento definida nas configurações do usuário que submeteu o arquivo, o documento será armazenado. Existem duas opções de armazenamento implementadas, sendo elas local e no Google Drive.

Para o armazenamento local o Laravel fornece uma abstração do gerenciador de arquivos através do pacote Flysystem[6] desenvolvido por Frank de Jonge. Para utilizar do Laravel Flysystem é necessário apenas configurar qual o drive de armazenamento será utilizado no arquivo de configurações *config/filesystems.php*. Por padrão o *framework* já fornece os drivers para armazenamento local, Amazon S3 e Rackspace Cloud onde o Laravel já realiza conversões para seus respectivos métodos automaticamente.

Para o armazenamento em *Webstorage* foi definido como serviço de armazenamento o Google Drive (ver subseção 3.1.2.4) que fornece uma API para a manipulação dos documentos armazenados. Para isto é necessário realizar uma autorização da conta google de armazenamento para que o Indexador possa gerenciar seus dados.

Ao final da indexação é instanciado a classe de armazenamento referente a configuração do usuário e seus respectivos métodos são processados, armazenando o arquivo localmente ou no Google Drive.

4.8 Buscador

Ferreira (2016) propôs o desenvolvimento e análise experimental de resultados de um buscador utilizando dos três metodos de recuperação mais utilizados atualmente: booleando, vetorial e probabilístico. Em seus testes foi constatado que os resultados obtidos a partir das buscas com o método probabilístico se mostraram mais confiáveis quando

[6] Disponível em: https://laravel.com/docs/5.6/filesystem. Acessado em: 21/10/2018

comparado aos outros dois métodos, pois apresentou resultados satisfatórios para as buscas na maioria das consultas realizadas, que podem ser visualizados na Tabela 8. É importante destacar que o modelo probabilístico apresentou melhores resultados para o conjunto de arquivos indexados, e não generaliza que para outros documentos podem apresentar o mesmo resultado.

Tabela 8 – Resultados dos testes de comparação entre os modelos de RI

Consulta	**Booleano**	**Vetorial**	**Probabilístico**
1	1°	3°	2°
2	3°	2°	1°
3	3°	2°	1°
4	2°	3°	1°
5	2°	3°	1°
6	1°	2°	3°
7	3°	2°	1°
8	2°	3°	1°
9	3°	2°	1°
10	3°	2°	1°
11	3°	2°	1°
12	2°	3°	1°

Fonte: Adaptado de Ferreira (2016).

Levando em consideração os modelos conhecidos, em conjunto aos resultados dos testes realizados por Ferreira (2016), foi definido que para validar o resultado do processo de indexação seria desenvolvido uma adaptação do algoritmo de busca utilizado do modelo probabilístico.

5 RESULTADOS

Como resultado deste trabalho foi desenvolvido um protótipo de um indexador de documentos, que possibilita que um usuário possa realizar o *upload* de um documento para o sistema, onde será feito o OCR, a indexação e em seguida o armazenamento. A partir do processo de indexação os documentos submetidos ao sistema também poderão ser recuperados pelo buscador, onde o usuário pode pesquisar por meio de expressões e o SRI irá verificar quais os documentos tem maior probabilidade de corresponder com os termos pesquisados.

5.1 O algoritmo de indexação

O algoritmo representado na Figura 14 se mostrou eficaz nos documentos que lhe foram submetidos, porém seu processamento não foi eficiente, devido ao longo tempo gasto para a indexação. Por exemplo, documentos com 16 páginas levaram aproximadamente 8 minutos para serem indexados. O tempo considerado inclui a indexação de cada documento, bem como seu *upload*, OCR e armazenamento. O tempo gasto por cada documento pode ser visto na terceira coluna (Tempo 1) da Tabela 9.

Ao se observar as linhas 34 e 35 na Figura 14 é possível observar que a cada novo termo no documento é realizada uma nova consulta para verificar se o termo já existe na base e em seguida ele é contabilizado na lista de índices do documento. Esse processamento é extremamente pesado devido ao gargalo que o Mapeamento Objeto-Relacional(ORM) do Laravel cria ao realizar milhares de requisições ao BD. Afim de solucionar esta ineficácia foi proposta uma alteração na forma como o algoritmo realizava a indexação.

Para resolver o problema de tempo de processamento, foi implementada uma solução de tratamento de consultas em memória, disponibilizada pelo próprio *framework* Laravel, que pode oferecer até 300% de melhoria no desempenho [1]. Assim, ao invés de realizar milhares de consultas ao BD para indexar apenas 1 documento são feitas apenas três, uma para carregar os dados em *cache* com todos os termos e índices, que são utilizados para o processamento em memória de todos os novos índices, e outras duas para atualizar todos os novos termos e índices no banco.

[1] Disponível em: https://Laravel.com/docs/5.6/cache. Acessado em: 21/10/2018

Figura 14 – Algoritmo de indexação.

```php
<?php
namespace App;

use DB;
use App\Models\Termo;
use App\Models\Indice;
use Illuminate\Database\Eloquent\Model;

class Indexador extends Model
{
    private $termos;
    private $texto;
    private $documento_id;

    public function __construct($texto, $documento_id) {
        $this->texto        = $texto;
        $this->documento_id = $documento_id;
    }

    public function indexador($arquivo) {
        $blacklist = Termo::getBlacklist();

        // analise lexica do texto, remove caracteres especiais e letras
        //maiusculas
        $arquivo = strtolower(Util::removeAcentos($this->texto));

        // regex para pegar apenas palavras
        preg_match_all('/[a-zA-Z]+/', $arquivo, $termos);
        // para todos os tokens encontrados adiciona ao  ndice
        foreach ($termos[0] as $termo) {
          if (strlen($termo) > 1) {
            //caso n o seja uma palavra de parada adiciona ao indice
            if (!in_array($termo, $blacklist)) {
              $termo_id = Termo::adicionaTermo($termo);
              Indice::adiciona($termo_id, $this->documento_id);
            }
          }
        }
    }
}
```

Tabela 9 – Tempo gasto para indexação de documentos

Tempo gasto para indexação de documentos			
Documento	**Páginas**	**Tempo 1(seg)**	**Tempo 2(seg)**
420-Lei no 4.320.pdf	28	1068	41
8055.png	1	4	4
A abelinha curiosa.png	1	39	12
A Enfase.jpg	1	28	7
B-Mac.png	1	46	11
Boleto Rafaela.jpg	1	45	25
Cellular Automata Based Simulation for Smoke.pdf	6	393	33
Chromosome.png	1	36	18
danfe.jpg	1	76	49
Forum das estatais pela educação.pdf	5	108	59
Gerenciador.pdf	16	419	51
IFMG 1.png	1	8	8
IFMG 2.png	1	5	5
IFMG Horarios 2018 - 1.pdf	5	156	47
IFMG Horarios 2018 - 2.pdf	5	156	14
maxresdefault.jpg	1	34	21
Mini Boleto Direito.jpg	1	23	20
PDF de Exemplo.pdf	1	6	3
Pq eres la persona.jpg	1	16	10
Roteiro Direito.pdf	26	657	120
Simulation of Pedestrian Evacuation.pdf	7	168	62
Sistema de Recuperação.png	1	19	19
Texto verbal.jpg	1	19	10
The quick brown.png	1	6	6

O tratamento de consultas em memória permitiu que a indexação tivesse uma melhora considerável de desempenho, chegando a apresentar resultados mais rápidos do que a implementação anterior, que realizava milhares de inserções no banco para indexar um único documento. Os resultados do método que utiliza do processamento em memória podem ser vistos na quarta coluna (Tempo 2) da Tabela 9. É importante ressaltar que houve uma melhora considerável para os documentos testados, porém não foram realizados experimentos para averiguar se esta solução se adequaria a todos os tipos de documento em diversos outros contextos, visto que foge do escopo deste projeto de conclusão de curso.

Documentos maiores, como o *420-Lei no 4.320.pdf* que possui 28 páginas demorava cerca de 1068 segundos no processamento de varias inserções, e passou a ser indexado com apenas 41 segundos. Documentos menores, com apenas uma página não sofreram muita variação em seu tempo, algumas mantiveram o mesmo. Essa diferença neste conjunto de documentos, se deve pelo tamanho do conteúdo textual de cada documento, quanto maior o documento, maior a quantidade de palavras e consequentemente maior quantidade de inserções e consultas ao banco. Para documentos pequenos a quantidade de inserções é tão pequena que o processamento em memória não apresenta uma grande melhoria no

desempenho, resultando no mesmo tempo de indexação.

5.2 Modelo de Dados

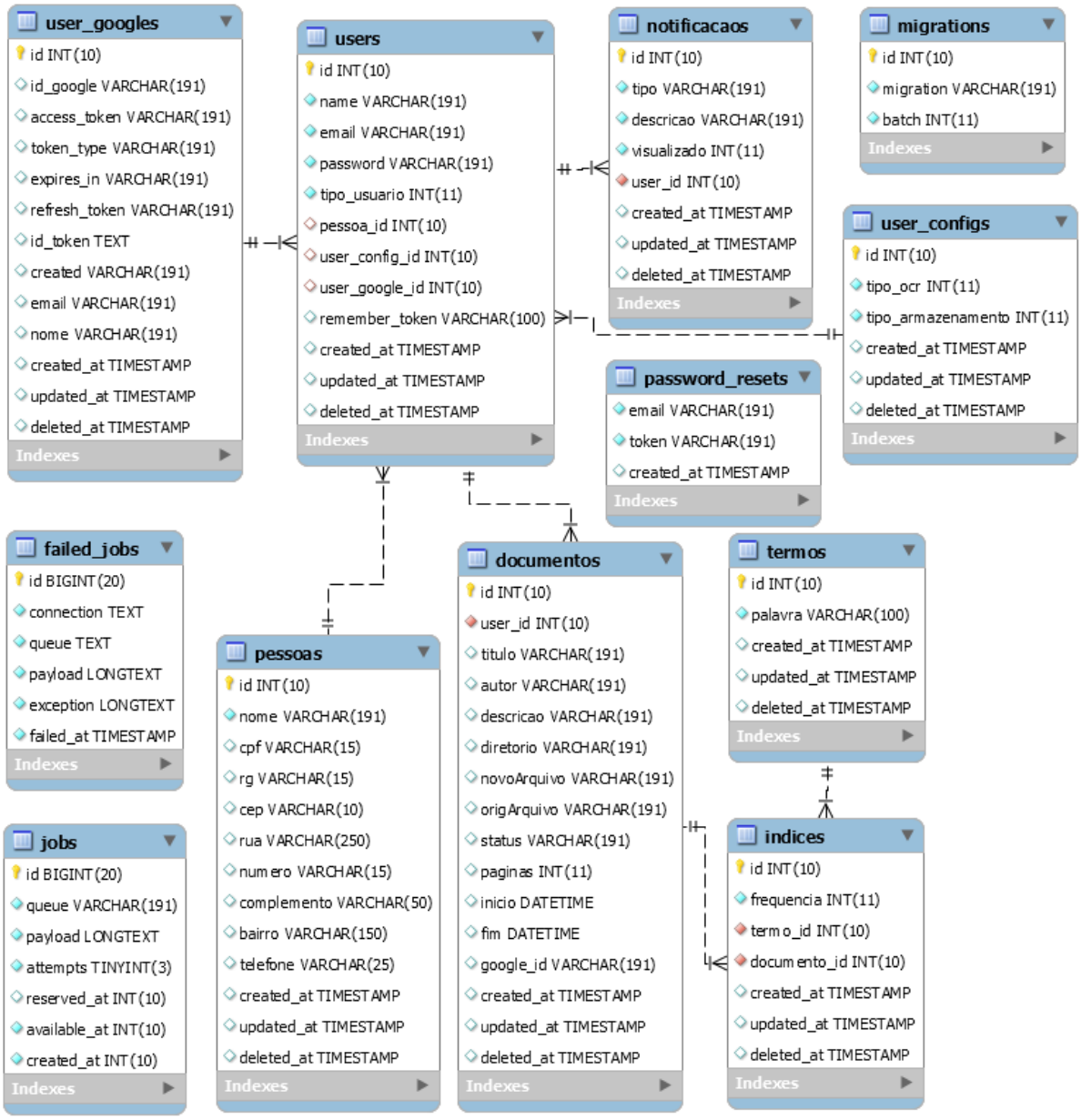
A partir dos requisitos levantados, o Diagrama de Entidade-Relacionamento (DER) detalhado na Figura 15, foi construído utilizando o *software* MySQL Workbench versão 6.3.10. As tabelas e seus respectivos atributos foram definidos com o intuito de satisfazer especificamente os requisitos para o indexador.

A Tabela 10 contém o nome das tabelas e suas respectivas descrições no Indexador.

Tabela 10 – Descrição das tabelas do bando de dados do Indexador.

Tabela	**Descrição**
documentos	Armazena os documentos armazenados no sistema.
failed_jobs	Armazena quais as filas de processamento falharam e quais seus respectivos erros.
indices	Armazena os índices de indexação entre os documentos e termos, para que possam ser buscados quando necessário.
jobs	Armazena as filas de processamento para indexação utilizando *queues.*
migrations	Armazena as migrações, responsável pelo controle de versão do banco de dados do Laravel.
notificacaos	Armazena as notificações dos usuários quando alguma indexação não é realizada com sucesso.
password_resets	Armazena as solicitações de recuperação de senha.
pessoas	Armazena os dados pessoais de cada usuário do sistema.
termos	Armazena os termos já conhecidos pelo sistema, que foram encontrados em algum documento.
users	Armazena os usuários do sistema.
users_configs	Armazena as configurações do sistema de cada usuário, como OCR e armazenamento.
users_googles	Armazena os dados da autorização do armazenamento no Google Drive.

Figura 15 – DER do banco de dados do Indexador



5.3 Interface de Login

A interface de *login* onde o usuário realiza a autenticação na aplicação, foi projetada para ser simples e conter apenas os campos de *login* e senha para que o usuário pudesse logar na aplicação. A interface também contém um *link* chamado "Eu esqueci minha senha"para que sejam redirecionados para uma outra interface que irá solicitar o e-mail para onde será enviado o *link* de recuperação. Esta funcionalidade ainda não foi implementada por fugir do escopo deste trabalho de conclusão de curso e será colocado na lista de tarefas futuras. A interface de login pode ser visualizada na Figura 16.

Figura 16 – Interface de Login.

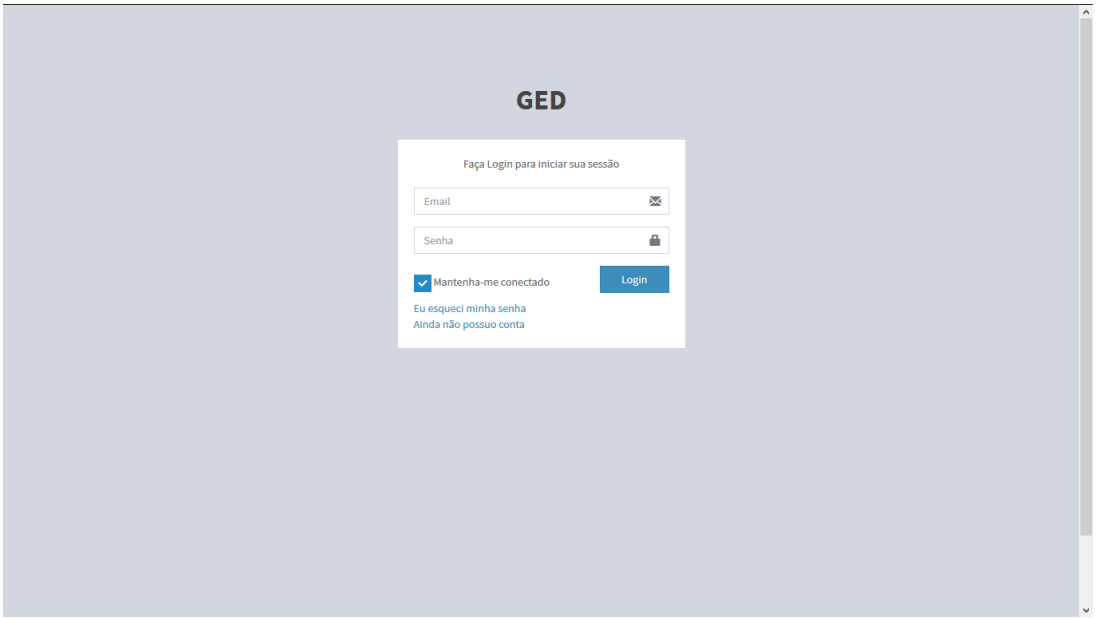
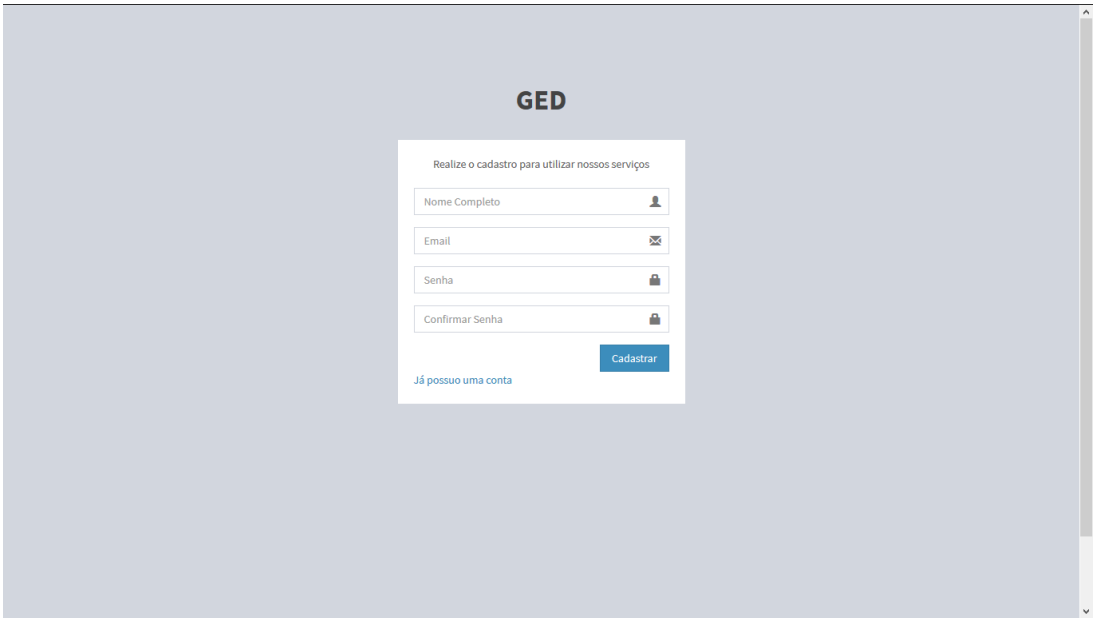


Figura 17 – Interface de Cadastro.

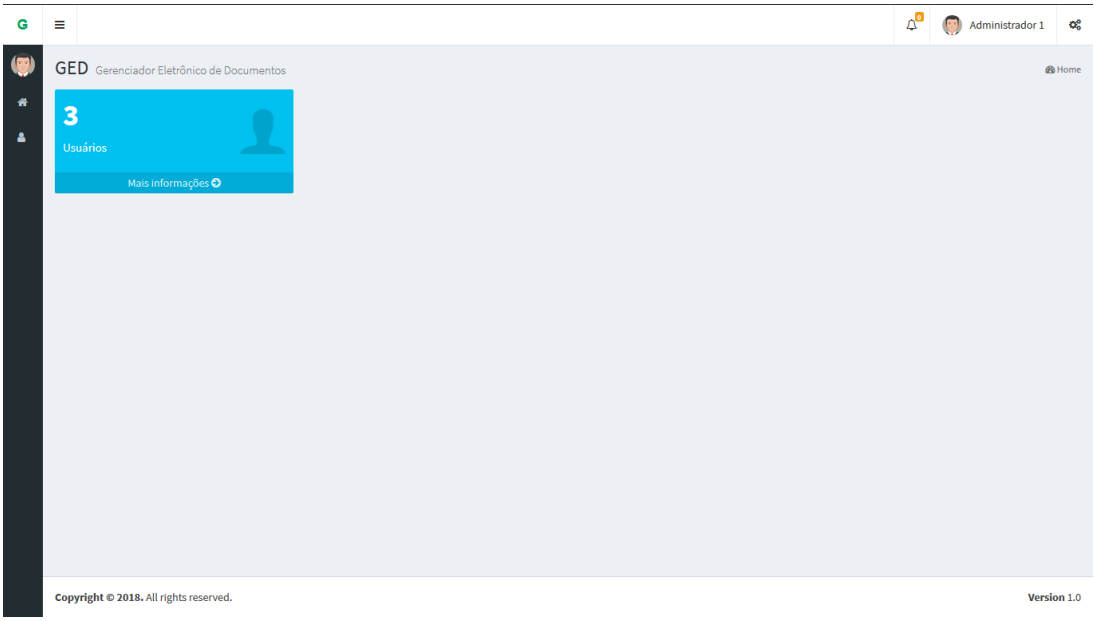


Os usuários que ainda não são cadastrados podem clicar no *link* "Ainda não possuo conta"na interface de *login* onde serão redirecionados para a interface de cadastro, que pode ser visualizado na Figura 17. Nesta página o usuário deve informar seu nome completo, seu e-mail (que deve único no sistema) e sua senha.

5.4 O Usuário Administrador

A partir do momento que um usuário realiza *login* na aplicação e seu tipo de usuário é Administrador ele é redirecionado para o *Dashboard* da visão do Administrador, que pode ser visualizado na Figura 18. Nesta página o usuário possui um menu lateral e um uma caixa azul no centro da interface mostrando quantos usuários estão cadastrado, ambos os elementos são *links* para acessar os usuários cadastrados no sistema.

Figura 18 – *Dashboard* da visão do Administrador.



5.4.1 Gerenciador de Usuários

Ao clicar nos *links* de usuários, o Administrador será redirecionado para a página de gerenciar usuários, que pode ser visto na Figura 19. Nesta página são listados em um Datatable todos os usuários cadastrados, mostrando os campos Id, Nome, E-mail, o Tipo de Usuário, que até o momento pode ser Administrador ou Cliente, um botão para editar e outro para deletar o usuário.

Figura 19 – Interface para gerenciar todos os usuários da aplicação.

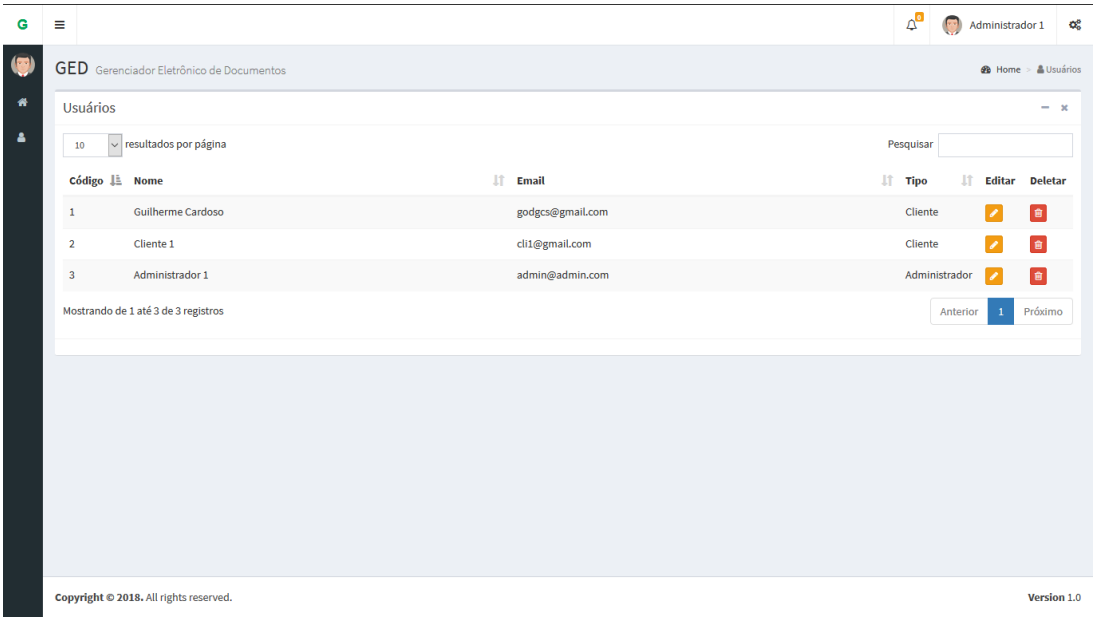
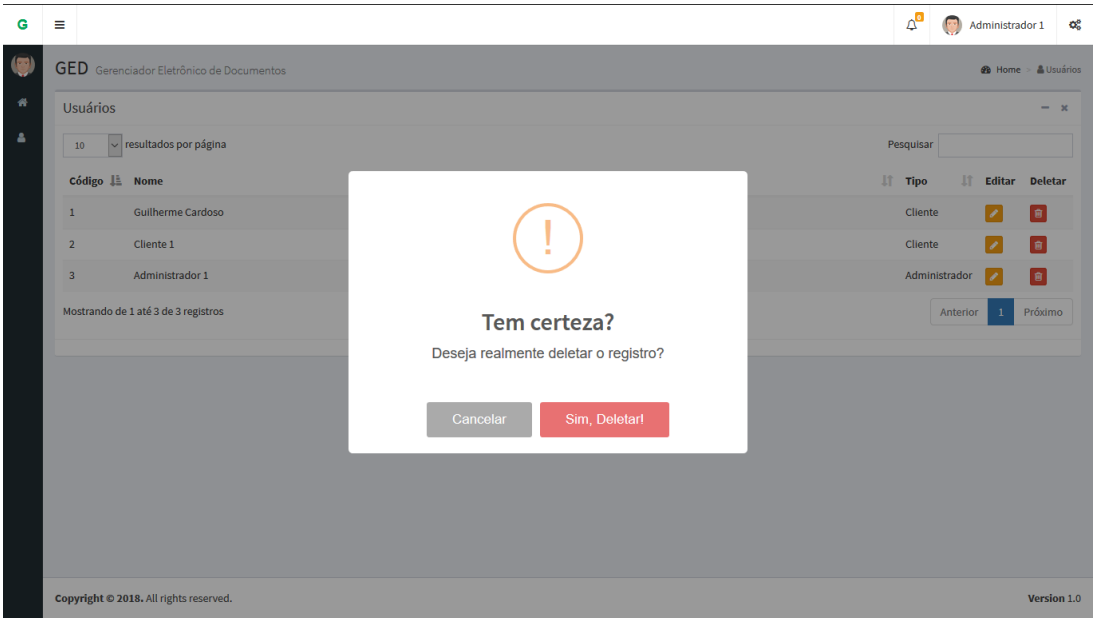


Figura 20 – Confirmação para apagar um usuário.



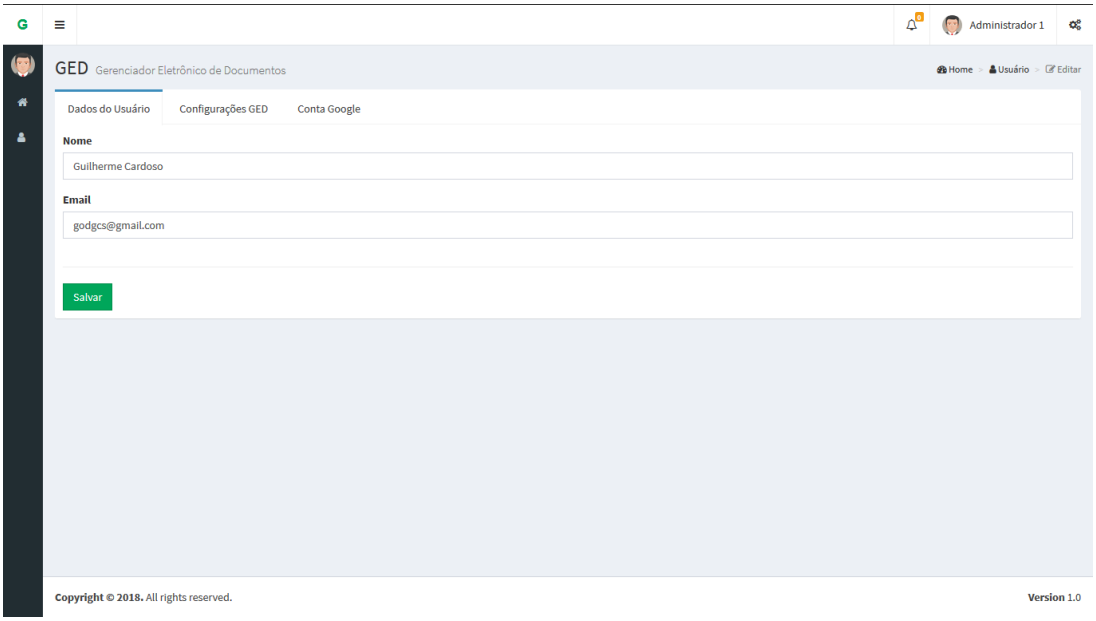
Ao clicar no no botão de excluir o usuário será exibido na interface um *alert*, utilizando do *plugin* Sweet Alert confirmando a exclusão. A exclusão é feita de forma lógica, caso o usuário confirme a exclusão uma solicitação AJAX será feira e então o usuário selecionado será excluído.

Ao se clicar no botão de editar, o usuário será redirecionado para a página de edição de usuário, essa interface é composta por três abas, a primeira contém apenas

os campos de pessoais, a segunda aba contém campos de configurações do sistema para aquele usuário, e a terceira aba contém as configurações de armazenamento em *webstorage* do Google Drive.

A primeira aba contém dados pessoais do usuário, no momento foram colocados apenas dois campos, o nome e o e-mail como pode ser visto na Figura 21. Em versões futuras da aplicação está aba será utilizada para diversos outros valores que agregam a identidade pessoal do usuário.

Figura 21 – Primeira aba para gerenciar um usuário.



A segunda aba contém configurações do sistema do usuário selecionado, estas configurações incluem:

- A seleção do método de OCR a ser utilizado, podendo ser TesseractOCR, Google Cloud Vision, HP Heaven On Demand;
- A seleção do método de armazenamento escolhido pelo usuário, podendo ser local ou através do Google Drive;
- A seleção do tipo do usuário, permitindo que este seja definido como Administrador ou Cliente.

A interface referente a segunda aba pode ser visualizada na Figura 22.

Figura 22 – Segunda aba para gerenciar um usuário.

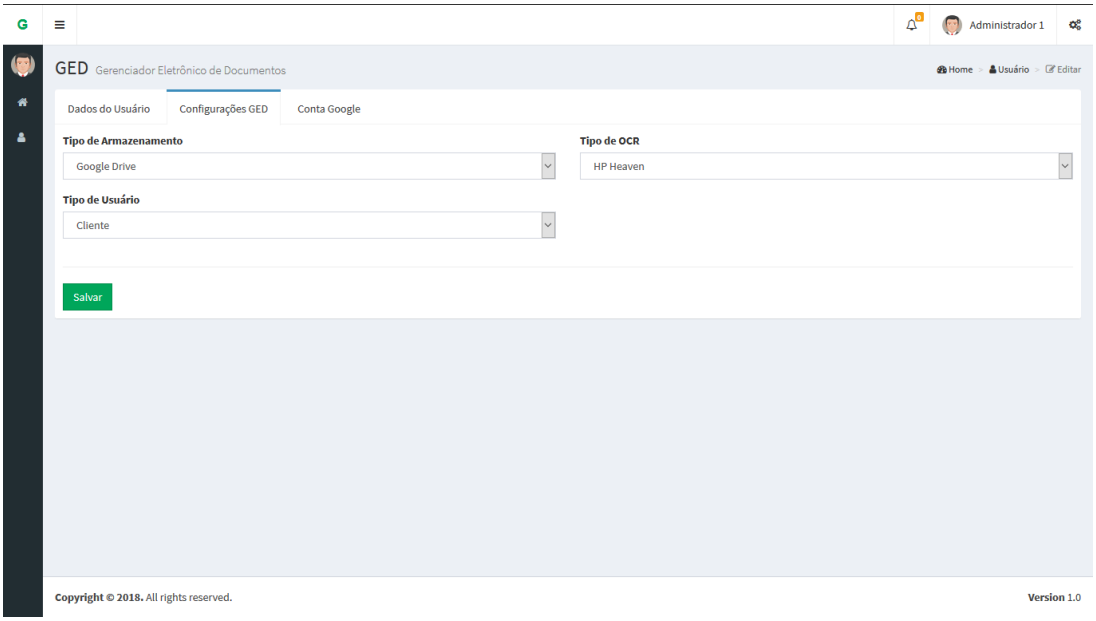
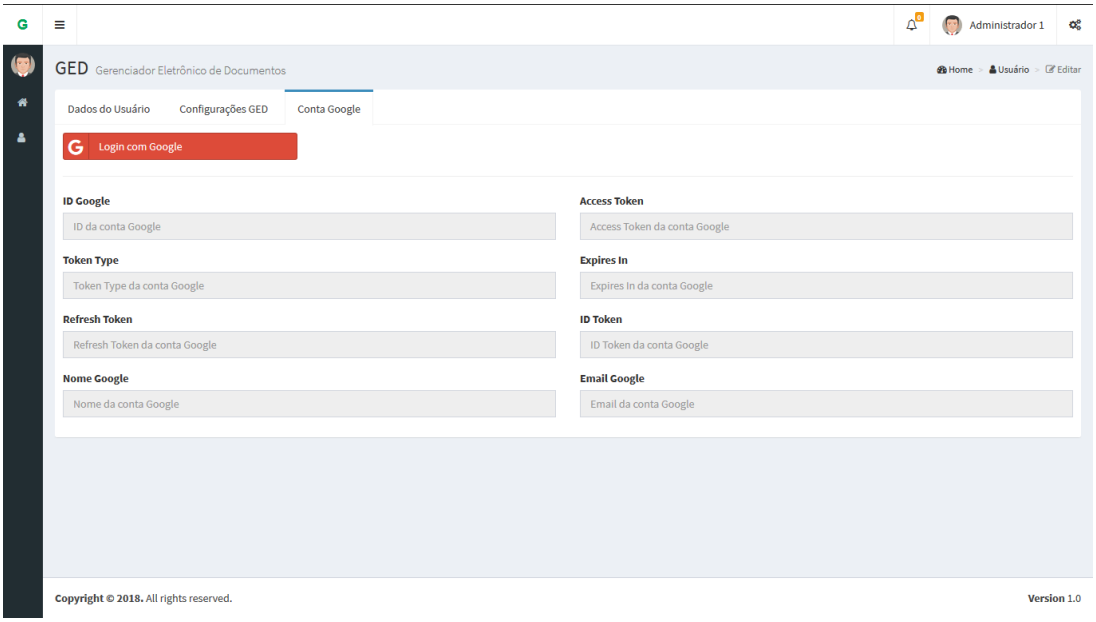


Figura 23 – Terceira aba para gerenciar um usuário.

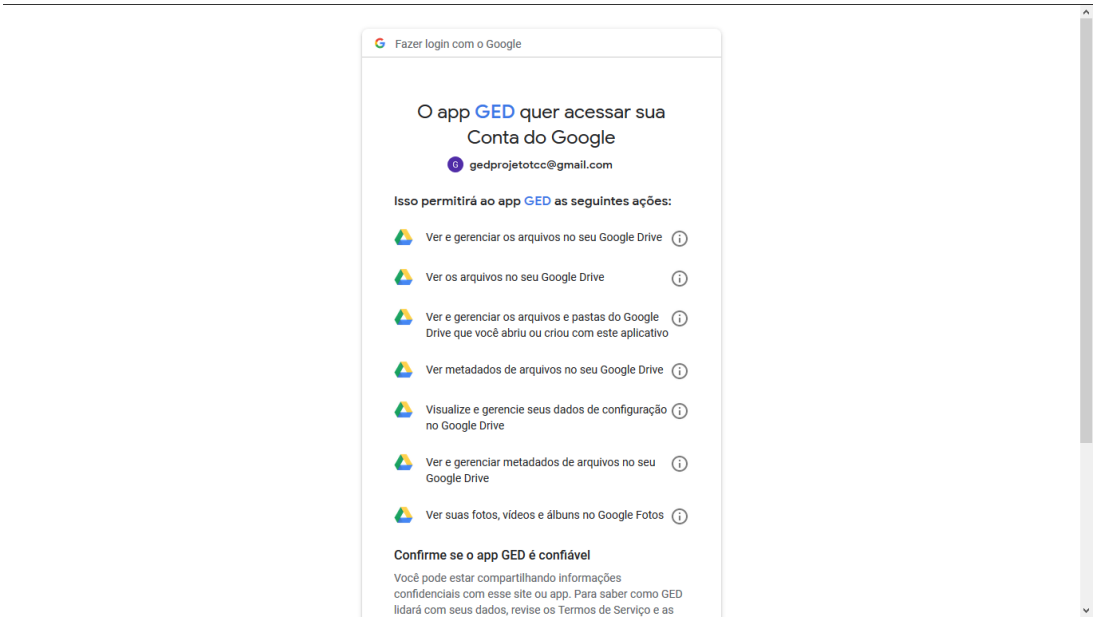


Fonte: Elaborado pelo Autor.

A terceira aba, que pode ser vista na Figura 23, contém o botão de autorização para a aplicação poder utilizar da conta do Google Drive do usuário para utilizar como armazenamento, a página também contém diversos campos onde é mostrado os dados da conta conectada, são apenas conteúdo informacional.

O Botão "Login com Google"redireciona o Administrador para uma interface de autenticação do Google para permitir que a aplicação possa manipular a conta do Google Drive por meio da API. A página de redirecionamento para permissões pode ser visualizada na Figura 24.

Figura 24 – Interface de autorização para API do Google Drive.



5.5 O Usuário Cliente

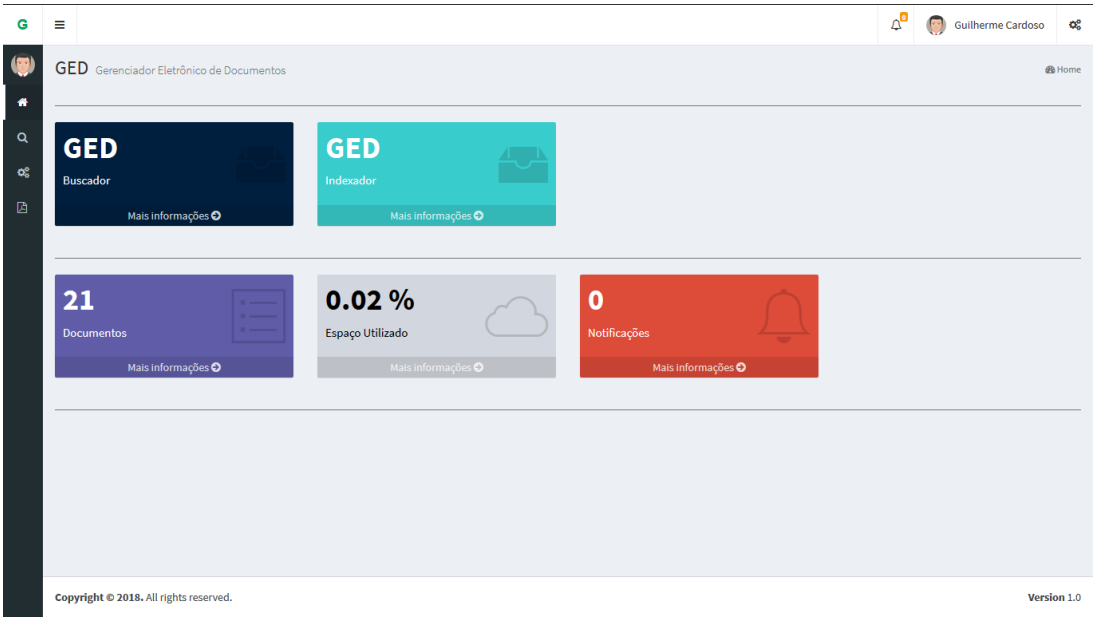
Caso o usuário logado seja do tipo Cliente, após sua autenticação ele será redirecionado para a *Dashboard* da visão do cliente, retratado na Figura 25. A esquerda da página o usuário tem acesso a um menu lateral que da acesso as funções da aplicação: buscador, indexador e gerenciador (página onde são listados todos os documentos armazenados). No centro da página existem caixas contendo *links* e informações sobre a conta.

Na primeira linha existem duas caixas que levam para as interfaces do buscador e de indexação. Na segunda linha existem três caixas:

- A primeira mostra a quantidade de documentos armazenados e também possui um *link* para o gerenciador de documentos, nele estão presentes todos os documentos armazenados pelo usuário no sistema.
- A segunda contém a informação de quanto de espaço livre o usuário possui, está caixa é visível apenas para usuários com o tipo de armazenamento selecionado para o Google Drive. Este valor é consultado utilizando a API do Google.

- A terceira contabiliza a quantidade de notificações não lidas do usuário e também possui um *link* para sua respectiva página.

Figura 25 – *Dashboard* da visão do Cliente.

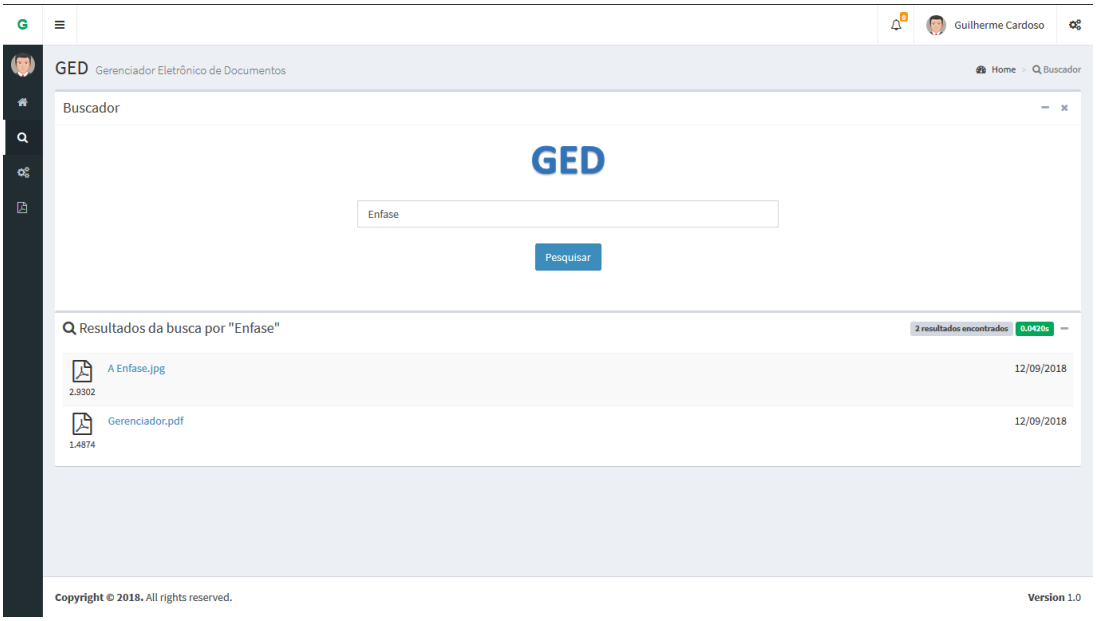


5.5.1 Buscador

A interface do buscador é extremamente simples como vista na Figura 26, ela contém apenas um campo de pesquisa no centro da página onde o usuário pode informar suas expressões de busca. Ao clicar no botão pesquisar a requisição será processada via AJAX e os resultados serão listados logo a baixo, junto ao tempo de processamento gasto. A ordem dos documentos retornados será por probabilidade de atender a necessidade do usuário.

Para acessar os documentos retornados o usuário deve clicar no nome do arquivo na listagem retornada que será realizado o download de acordo com a configuração do navegador utilizado.

Figura 26 – Interface para buscar documentos.



5.5.2 Indexador

A interface de indexação, que pode ser visualizada na Figura 27 basicamente é composta por um dropzone, que permite que vários documentos sejam adicionados e enviados para o servidor de forma paralela.

Com a utilização do *dropzone* o usuário pode arrastar e soltar arquivos que serão enviados para o o processamento no servidor, para que o cliente pudesse ter maior controle de quais arquivos e em qual ordem seriam processados foram adicionados alguns botões de controle geral e específicos .

Os botões de controle geral são melhor visualizados na Figura 28. O primeiro botão permite que o usuário selecione quais arquivos ele deseja armazenar no sistema, onde ao ser clicado ele irá abrir uma janela de seleção. O segundo botão permite que usuário remova todos os documentos que foram selecionados para indexação, ele limpa a lista de *upload*. O terceiro botão faz com que todos os arquivos selecionados sejam processados, onde são enviados para o servidor para serem indexados e armazenados de acordo com as configurações do sistema do usuário.

Figura 27 – Interface para indexar documentos.

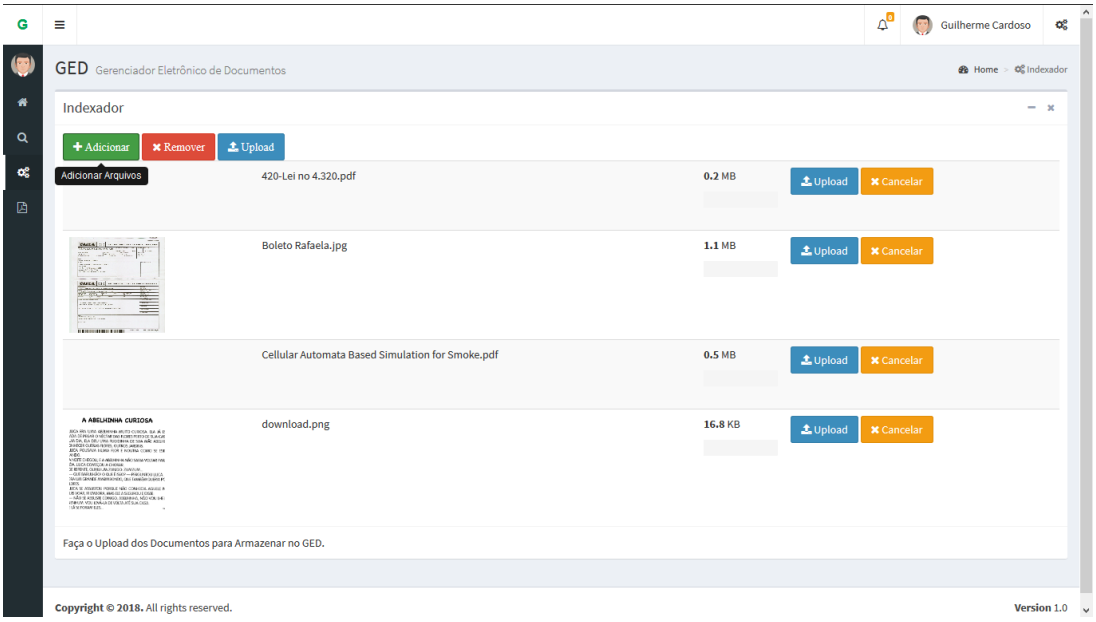
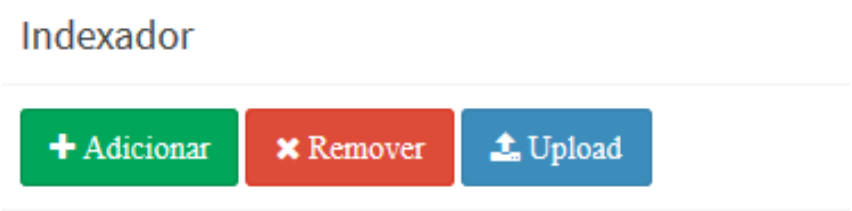
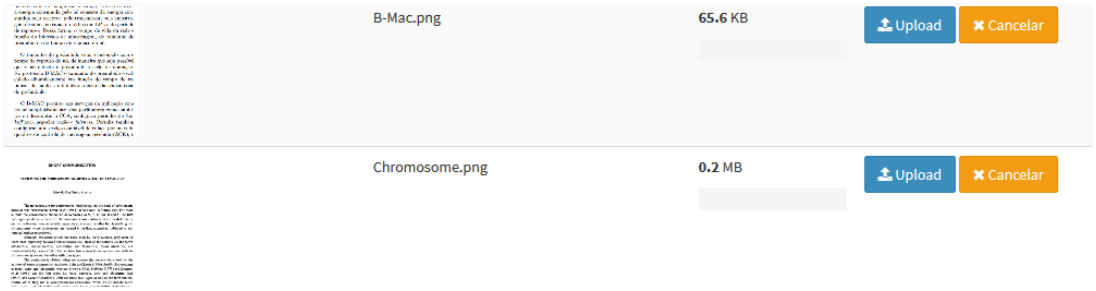


Figura 28 – Botões de controle geral do Indexador.



Os botões de controle especifico de cada documento somente são visíveis a partir do momento que algum arquivo é selecionado para o upload, conforme pode ser visto na Figura 29. A partir deste evento logo abaixo dos botões de controle geral são listados os documentos selecionados, onde na primeira coluna é possível ver uma previa do documento caso o arquivo seja uma imagem, na segunda coluna o nome do arquivo, na terceira coluna o seu tamanho e por fim na quarta coluna são exibidos os dois botões de controle especifico. O primeiro botão permite que o usuário realize a submissão apenas do documento referente ao botão, já o segundo botão remove seu respectivo documento da lista.

Figura 29 – Botões de controle específicos de documentos do Indexador.



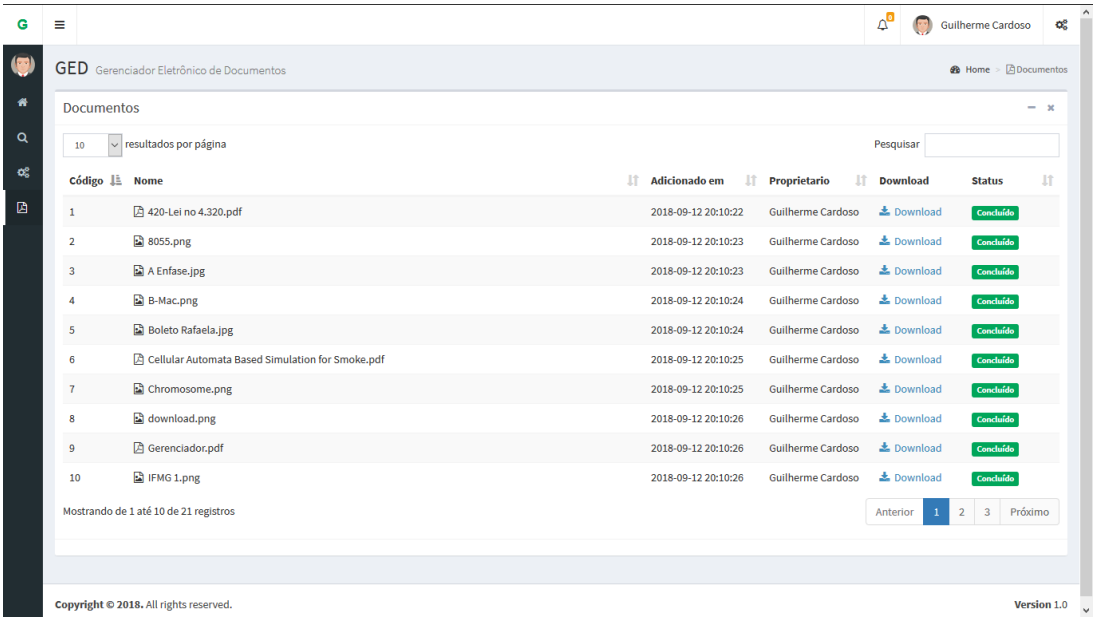
5.5.3 Visualizador de Documentos

A página de visualização de documentos que pode ser vista na Figura 30 tem o intuito de detalhar todos os documentos indexados no sistema, informando ao usuário seus dados principais. Utilizando de um Datatable são listados algumas informações dos documentos, como: o Id, o Nome, a Data de *Upload*, o proprietário, um *link* para *download* e o *status* do documento.

O status de um documento pode assumir 4 valores:

- Pendente: Destacado na cor amarela, representa o momento em que o documento foi enviado para o servidor, porém aguarda na fila para ser indexado;
- Processando: Destacado na cor azul, quando o documento está no processo de indexação;
- Concluído: Destacado na cor verde, este *status* representa o momento no qual o documento já foi indexado e já pode ser consultado através do buscador;
- Falhou: Destacado na cor vermelha, quando ocorre alguma falha no indexador por erro de comunicação com o servidor de OCR ou no armazenamento seu *status* é definido como falha.

Figura 30 – Página para visualizar todos os documentos armazenados.



6 CONSIDERAÇÕES FINAIS

Neste trabalho foi desenvolvido um protótipo de um sistema de gerenciamento de documentos eletrônicos por meio de uma aplicação *web*, que conta com armazenamento local ou *web* (*webstorage* no Google Drive), além de um buscador para os documentos arquivados.

Neste protótipo foram implementadas apenas as funcionalidades principais de um GED. Porém, o protótipo tem o potencial para fomentar o desenvolvimento de uma ferramenta completa, que possa gerenciar todo o fluxo de documentos, desde sua criação até seu descarte.

O principal requisito para o processo de indexação é o reconhecimento óptico dos caracteres de um documento. Nesse protótipo, foram implementadas três ferramentas OCR: Google Cloud, Hp Haven OnDemand e TesseractOCR. É possível selecionar qual dessas aplicações OCR funcionarão junto ao indexados através de uma opção na interface do administrador. O protótipo foi pensado dessa maneira para permitir o (des)acoplamento futuro de ferramentas diferentes, já que esse mercado tem evoluído bastante e a cada dia surge uma nova ferramenta.

Da mesma forma, os meios de armazenamento também podem ser selecionados por um administrador do sistema, permitindo que os documentos sejam armazenados de forma local e em um servidor de *webstorage*. Embora tenha muitas vantagens, como acesso de qualquer lugar e *backup* facilitado, o armazenamento em *webstorage* pode trazer riscos de acesso à dados sensíveis. Assim, a forma de armazenamento local também foi implementada pensando em usuários que desejam abrir mão das vantagens de um *webstorage* em troca de mais segurança para seus documentos. Essa opção é viável para aqueles usuários que possuam estrutura física interna para suportar a aplicação.

Utilizando de um buscador por consulta probabilística, que retorna como resultado uma lista de documentos ordenadas pela probabilidade de atenderem a uma demanda informacional, foi possível constatar a eficácia do algoritmo de indexação. Como relatado anteriormente, o algoritmo não se mostrou eficiente inicialmente, porém com as alterações para compactações de inserções de índices e termos utilizando de um processamento com *Caches* fornecidas pelo *framework* Laravel, foi possível observar o quanto foi grande o impacto no tempo gasto pela indexação, que passou a apresentou uma eficiência de 300% comparado ao algoritmo que realiza milhares de inserções ao banco de dados.

A possibilidade de se trabalhar com um *framework web*, como o Laravel para o desenvolvimento deste projeto, proporcionou grandes vantagens no processo de implemen-

tação, como a otimização de rotinas de comunicação, uma codificação com códigos mais limpos e organizados pelo fato de que existem dependências que se encarregam de gerenciar todas as conexões e também pela facilidade e acessibilidade ao codificar. Utilizar comandos do *php artisan* que permitem gerenciar as versões do banco de dados e gerar esqueletos de códigos, como *models*, *controllers*, *middlewares* entre outros, permite que muito tempo seja economizado na parte de codificação. Assim, o desenvolvedor pode dedicar seu tempo a regras de negócio da aplicação.

O protótipo concebido neste trabalho tem potencial para se tornar uma ferramenta que permita às mais diversas instituições que trabalham com documentos digitais, controlar o seu acervo e acessá-los de forma eficiente.

6.1 Trabalhos Futuros

O escopo do protótipo desenvolvido foi reduzido dado o tempo, se comparado à uma aplicação GED completa. Porém, melhorias podem ser realizadas para que o protótipo se torne uma ferramenta completa:

- **Gerenciador de arquivos e pastas:** Permite que o usuário recupere seus documentos também por um gerenciador de arquivos, onde ele pode criar pastas e salvar seus arquivos de acordo com sua própria organização pessoal.
- **Compartilhamento de documentos:** Permitir que um determinado usuário possa compartilhar seus arquivos com usuários de outra organização.
- **Estudo aprofundado sobre melhores métodos de OCR:** A ferramenta que realiza o OCR impacta diretamente da qualidade da indexação de um documento, então realizar um estudo sobre quais métodos fornecem os melhores resultados, pode proporcionar um melhor desempenho na recuperação destes documentos.
- **Estudo aprofundado sobre melhores métodos de armazenamento:** A forma como os documentos serão armazenados é um ponto importante nas regras de negocio do sistema. O preço do armazenamento, a qualidade do serviço, o limite de armazenamento e vários outros fatores, precisam ser avaliados com o propósito de fornecer um bom serviço aos usuários, garantindo sua segurança e valores acessíveis.
- **Hospedagem da aplicação:** Hospedagem da aplicação em um serviço que forneça a hospedagem de aplicações *web*.
- **Testes da aplicação em diversos dispositivos:** Mesmo utilizando de *templates* que fornecem recursos para que a aplicação possa se redimensionar para qualquer tamanho de tela, é necessário realizar verificação se o comportamento está como o

previsto, pois até o momento os testes foram realizados apenas no computador do desenvolvedor.

Referências

AECWEB; RABECHINI, R. J. *EAP é ferramenta essencial para o gerente de projeto*. 2018. Disponível em: <https://www.aecweb.com.br/cont/m/rev/eap-e-ferramenta-essencial-para-o-gerente-de-projeto_13582_3_0>. Citado na página 38.

AIIM Market Intelligence. Electronic records management - still playing catch-up with paper. 2009. Citado na página 30.

AMAZONAS, A. M. et al. Integrando motores de indexaçao de dados para a construçao de sistemas de recupreaçao de informaçao em ambientes heterogênos. *JISTEM-Journal of Information Systems and Technology Management*, Universidade de São Paulo-USP, v. 5, n. 2, p. 193–222, 2008. Citado 4 vezes nas páginas 17, 23, 26 e 27.

ANDRADE, M. V. M. et al. Gerenciamento eletrônico da informação: ferramenta para a gerência eficiente dos processos de trabalho. *Anais do Seminário Nacional de Bibliotecas Universitárias, 12. Recife: UFPE, 2002.*, UFPE, 2002. Citado 3 vezes nas páginas 18, 25 e 31.

ASUS. *O que é o ASUS WebStorage?* 2016. Disponível em: <https://www.asus.com/pt/support/FAQ/1030008/>. Citado na página 28.

BAEZA-YATES, R.; RIBEIRO-NETO, B. et al. *Modern information retrieval*. [S.l.]: ACM press New York, 1999. v. 463. Citado na página 22.

BARROS, C. M. et al. Preservação do conhecimento e programa estratégico ifla sobre preservação e conservação–pac. *Múltiplos Olhares em Ciência da Informação-ISSN 2237-6658*, v. 5, n. 1, 2016. Citado 2 vezes nas páginas 17 e 18.

BUCHTIK, L. Secrets to mastering the wbs in real world projects. In: PROJECT MANAGEMENT INSTITUTE. [S.l.], 2013. Citado na página 38.

CRUZ, C. H. B. Vannevar bush: an introduction. *Revista Latinoamericana de Psicopatologia Fundamental*, SciELO Brasil, v. 14, n. 1, p. 11–13, 2011. Citado na página 21.

FANTINI, S. R. et al. Aplicação do gerenciamento eletrônico de documentos: estudo de caso de escolhas de soluções. Florianópolis, SC, 2001. Citado 6 vezes nas páginas 16, 17, 24, 25, 29 e 30.

FERREIRA, R. S. Implemenntação e análise experimental de uma máquina de busca a documentos pdf. Formiga, MG, 2016. Citado 7 vezes nas páginas 21, 23, 24, 28, 56, 57 e 58.

FREIBERGER, Z. Gestão de documentos e arquivística. Curitiba: IFPR, 2010. Citado na página 16.

GANDINI, J. A. D.; SALOMÃO, D. P. d. S.; JACOB, C. A segurança dos documentos digitais. *Disponıvel em< https://jus.com.br/artigos/2677/a-seguranca-dos-documentos-digitais>. Acesso em: Agosto*, v. 11, 2001. Citado 2 vezes nas páginas 16 e 17.

HUANG, H.; ZHANG, B. Text indexing and retrieval. In: *Encyclopedia of Database Systems*. [S.l.]: Springer, 2009. p. 3055–3058. Citado 4 vezes nas páginas 22, 23, 26 e 31.

ISBRASIL. *Quais as vantagens de armazenamento em nuvem*. 2017. Disponível em: <https://www.isbrasil.info/blog/quais-as-vantagens-de-armazenamento-em-nuvem.html>. Citado na página 28.

MACEDO, G. M. F. d. et al. Bases para a implantação de um sistema de gerenciamento eletrônico de documento-ged: estudo de caso. Florianópolis, SC, 2003. Citado 2 vezes nas páginas 17 e 30.

MASSARI, J. *O que é Model-View-Controller (MVC)*. 2017. Disponível em: <https://www.portalgsti.com.br/2017/08/padrao-mvc-arquitetura-model-view-controller.html>. Citado na página 34.

MENEZES, L. R. d. Ged-gerenciamento eletrônico de documentos: a preservação da informação e diretrizes para implantação. 2016. Citado 2 vezes nas páginas 16 e 17.

MOOERS, C. N. Zatocoding applied to mechanical organization of knowledge. *American documentation*, Wiley Online Library, v. 2, n. 1, p. 20–32, 1951. Citado na página 21.

MUSAFIR, V. E. N. Ged e workflow–soluções inovadoras para nossos clientes. *Tematec. Brasília, ano VII*, n. 57, 2001. Citado na página 18.

POMARICO, D. *POO – Programação Orientada a Objetos em VB.NET – Parte 1*. 2018. Disponível em: <http://www.linhadecodigo.com.br/artigo/585/poo-programacao-orientada-a-objetos-em-vbnet-parte-1.aspx>. Citado na página 34.

Portal ABBYY. *O que é OCR?* 2018. Disponível em: <https://www.abbyy.com/pt-br/ocr/>. Citado na página 25.

Portal GED. *GESTÃO ELETRÔNICA DE DOCUMENTOS (GED)*. 2018. Disponível em: <https://ged.net.br/>. Citado na página 24.

Portal W3techs. *Usage of server-side programming languages for websites*. 2018. Disponível em: <https://w3techs.com/technologies/overview/programming_language/all>. Citado na página 33.

PRODimage Tecnologia. *GED*. 2006. Disponível em: <http://www.prodimage.com.br/ged.php>. Citado na página 25.

RF, S. Estudos de revisão sistemática: um guia para síntese criteriosa da evidência científica. *Revista brasileira de fisioterapia*, SciELO Brasil, v. 11, n. 1, p. 83–89, 2007. Citado 2 vezes nas páginas 39 e 40.

RUBI, M. P.; FUJITA, M. S. L. Elementos de política de indexação em manuais de indexação de sistemas de informação especializados. *Perspectivas em ciência da informação*, v. 8, n. 1, 2003. Citado 2 vezes nas páginas 17 e 26.

SANTOYO, L. A. *Quem guarda, tem.* 2008. Disponível em: <http://www.administradores.com.br/noticias/negocios/quem-guarda-tem/16898>. Citado na página 16.

SHAFI'I, M. A. et al. An efficient information retrieval system using query expansion and document ranking. *Journal of Theoretical and Applied Information Technology*, v. 63, n. 1, 2014. Citado 3 vezes nas páginas 21, 22 e 29.

SILVA, D. P. da et al. Ged–gerenciamento eletrônico de documentos a tecnologia que está mudando o mundo. *INICIA*, v. 37, p. 38, 2003. Citado 2 vezes nas páginas 16 e 17.

SOUZA, A. F. d. S. et al. Ged – gerenciador eletrônico de documentos. Boa Vista, 2013. Citado na página 24.

ZOBEL, J.; MOFFAT, A. Inverted files for text search engines. *ACM computing surveys (CSUR)*, ACM, v. 38, n. 2, p. 6, 2006. Citado na página 26.