UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GUSTAVO HENRIQUE SILVA SALOMÃO

DESENVOLVIMENTO DE APLICAÇÃO WEB PARA GERENCIAMENTO DE OBJETOS DE APRENDIZAGEM

BAURU
Outubro/2018

GUSTAVO HENRIQUE SILVA SALOMÃO

DESENVOLVIMENTO DE APLICAÇÃO WEB PARA GERENCIAMENTO DE OBJETOS DE APRENDIZAGEM

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. Kleber Rocha de Oliveira

BAURU
Outubro/2018

Gustavo Henrique Silva Salomão Desenvolvimento de aplicação Web para gerenciamento de Objetos de Aprendizagem/ Gustavo Henrique Silva Salomão. – Bauru, Outubro/2018- 36 p. : il. (algumas color.) ; 30 cm.
Orientador: Prof. Dr. Kleber Rocha de Oliveira
Trabalho de Conclusão de Curso – Universidade Estadual Paulista "Júlio de Mesquita Filho" Faculdade de Ciências
Ciência da Computação, Outubro/2018.
1. Objetos de Aprendizagem 2. Aplicação Web 3. Repositório 4. Educação

Gustavo Henrique Silva Salomão

Desenvolvimento de aplicação Web para gerenciamento de Objetos de Aprendizagem

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Kleber Rocha de Oliveira
(Orientador)
Departamento de Computação
Faculdade de Ciências - Unesp

Profª Drª Simone das Graças Domingues Prado
Departamento de Computação
Faculdade de Ciências - Unesp

Profª Drª Andrea Carla Gonçalves Vianna
Departamento de Computação
Faculdade de Ciências - Unesp

Bauru, _____ de _____ de _____.

Agradecimentos

A esta universidade, aos docentes, diretores, coordenadores e administração por terem oferecido um ambiente criativo e amigável.

"Education is the most powerful weapon which you can use to change the world."
(Nelson Mandela)

Resumo

Os Objetos de Aprendizagem (OA) são ferramentas que trazem grandes vantagens no aprendizado, sua flexibilidade e possibilidade de reutilização facilitam a propagação do conhecimento, assim como sua atualização. Atualmente existe a dificuldade de se encontrar OAs com conteúdo de qualidade, específicos à busca e que contenham definições claras. A falta de organização e de detalhes a respeito dos OAs acarreta uma perda significativa do tempo que poderia ser utilizado para aprendizagem e dificulta a busca aprofundada pelo conhecimento. Além disso, outro inconveniente é a perda de OAs interessantes por falta de uma forma prática de armazená-los. O foco deste trabalho é desenvolver uma aplicação web que possibilite o armazenamento dos Objetos Digitais de Aprendizagem como URLs ou arquivos, junto com suas respectivas tags, metadados e definições. Isso de forma rápida, organizada, intuitiva, com fácil gerenciamento, possibilitando buscas eficientes dos mesmos e o compartilhamento entre os usuários.

Palavras-chave: Objetos de Aprendizagem, Aplicação Web, Repositório, Educação

Abstract

Learning Objects (LO) are tools that bring great advantages in learning. Its flexibility and the possibility of reuse facilitate the propagation of knowledge, as well as its updating. Currently there is the difficulty of finding LO with quality content, specific to the search and containing clear definitions. The lack of organization and detail about LO entails a significant loss of time that could be used for learning and makes difficult the in-depth search for knowledge. In addition, another drawback is the loss of interesting LO found for lack of a practical way of storing them. The focus of this work is to develop a web application that allows the storage of Digital Learning Objects as URLs or files, along with their respective tags, metadata and definitions. In a fast, organized, intuitive, easy to manage way, enabling efficient search of them and sharing among users.

Keywords: Learning Objects, Web Application, Repository, Education.

Lista de Figuras

Figura 1 – Página da bliblioteca pessoal do Diigo 14
Figura 2 – Modelo Cascata na Engenharia de Software 18
Figura 3 – Página inicial da aplicação desenvolvida 30
Figura 4 – Página de login da aplicação desenvolvida 30
Figura 5 – Página de cadastro da aplicação desenvolvida 31
Figura 6 – Página *Home* da aplicação desenvolvida 31
Figura 7 – Página inicial com o foco na janela de adição de URLs 32
Figura 8 – Página de URLs 32

Lista de abreviaturas e siglas

OA	Objeto de Aprendizagem
URL	*Uniform Resource Locator* (Localizador Uniforme de Recursos)
HTTP	*Hypertext Transfer Protocol*
FTP	Protocolo de transferência de arquivos
SMTP	Protocolo de transferência de mensagens simples
DNS	Servidor de Nomes de Domínio
IP	*Internet Protocol*
PHP	*Hypertext Preprocessor*
MVC	*Model-View-Controller*
ORM	*Object Relational Mapper*
HTML	*Hypertext Markup Language*
Diigo	*Digest of Internet Information, Groups and Other stuff*
IIS	*Internet Information Services*
SQL	*Structured Query Language*
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais

Sumário

1 INTRODUÇÃO 11

2 REFERENCIAL TEÓRICO 13

3 METODOLOGIA 16
3.1 Aplicação Web 16
3.2 Modelo em Cascata 17

4 DESENVOLVIMENTO 20
4.0.1 Servidor Web 20
4.0.2 PHP 20
4.0.3 Laravel 21
4.0.3.1 Composer 22
4.0.3.2 Artisan 22
4.0.4 Vagrant 22
4.0.5 Homestead 23
4.0.6 MySQL 24
4.1 ORM 25
4.2 Desenvolvimento 25
4.2.1 Requisitos 25
4.2.2 Viabilidade do Projeto 26
4.2.3 Implementação 26
4.2.3.1 Configurando o ambiente Homestead 26
4.2.3.2 Configurando o provedor 27
4.2.3.3 Configurando Pastas Compartilhadas 27
4.2.3.4 Configurando Sites Nginx 27
4.2.3.5 O arquivo de hosts 28
4.2.3.6 Iniciando a Vagrant Box 28
4.3 Aplicação 28

5 CONSIDERAÇÕES FINAIS 34

REFERÊNCIAS 36

1 Introdução

Um dos reflexos da evolução das tecnologias de informação nos últimos anos é o aumento da quantidade de conteúdos digitais disponíveis para a aprendizagem, por exemplo: *e-learnings*, *e-books*, pesquisas e artigos científicos, videoaulas. Esses exemplos se enquadram no conceito de Objetos de Aprendizagem. Segundo Wiley (2000), um Objeto de Aprendizagem "[. . .] é qualquer recurso digital que pode ser reusado para apoiar a aprendizagem". Para Tarouco, Fabre e Tamusiunas (2003):

> Um Objeto de Aprendizagem é qualquer recurso, suplementar ao processo de aprendizagem, que pode ser reusado para apoiar a aprendizagem, termo geralmente aplicado a materiais educacionais projetados e construídos em pequenos conjuntos visando a potencializar o processo de aprendizagem onde o recurso pode ser utilizado.

Os Objetos de Aprendizagem (OA) são ferramentas que trazem grandes vantagens no aprendizado, sua flexibilidade e possibilidade de reutilização facilitam a propagação do conhecimento, assim como sua atualização. Segundo Mendes, Souza e Caregnato (2004), as características que integram os OA são:

- Reusabilidade: reutilizável diversas vezes em diferentes ambientes de aprendizagem;
- Adaptabilidade: adaptável a qualquer ambiente de ensino;
- Granularidade: conteúdo dividido em partes, para facilitar sua reusabilidade;
- Acessibilidade: acessível facilmente via Internet;
- Durabilidade: possibilidade de continuar a ser usado, independente da mudança de tecnologia;
- Interoperabilidade: habilidade de operar através de diversos hardwares, sistemas operacionais e browsers, intercâmbio efetivo entre diferentes sistemas.

Uma definição mais detalhada é apresentada por AlfredoSanchez, Perez-Lezama e Starostenko (2014), na *4th World conference on educational technology researches*. Um OA representaria o conhecimento adquirido depois de entender, aplicar, sintetizar e avaliar um tópico específico. Assim, um OA deve conter seis elementos: os objetivos de aprendizagem a serem alcançados, as habilidades ou competências que o educando irá adquirir, as competências necessárias para a sua utilização, o conteúdo das atividades que o estudante deve realizar para adquirir conhecimentos ou desenvolver competências e os meios para medir se novos conhecimentos foram adquiridos.

Atualmente existe a dificuldade de se encontrar OAs com conteúdo de qualidade, específicos à busca e que contenham os componentes da definição acima. A falta de organização e de detalhes acarreta uma perda significativa do tempo que poderia ser utilizado para aprendizagem e dificulta a busca aprofundada pelo conhecimento. Além disso, outro inconveniente é a perda de OAs interessantes encontrados por falta de uma forma prática de armazená-los.

Durante a busca dos OAs no meio digital, visita-se diferentes *websites* que possuem seu conteúdo em diversos formatos (documentos de hipertexto, vídeos, fotos, arquivos, entre outros), uma vez que se encontra um OA desejado é de grande valia armazená-lo e especificá-lo através de metadados, tags, para que futuramente seja fácil reavê-los.

Armazenar o conteúdo em si, baixando e salvando cada tipo de dado em seus respectivos repositórios, passa a ser um processo cansativo, podendo infringir os Direitos Digitais dos mesmos. Por se tratar de conteúdos digitais, possuem um endereço URL (*Uniform Resource Locator*) ou Localizador Padrão de Recursos, o que possibilita acessá-los através de um navegador.

Visto que o uso/reuso desses materiais pode ajudar significativamente no processo de aquisição de conhecimento, como no caso da área de tecnologia de informação, onde há uma grande quantidade de material didático relacionado no meio digital, para a melhor utilização dos OAs é necessário que os mesmos estejam organizados e bem estruturados.

O foco deste trabalho foi desenvolver uma aplicação web que possibilitasse o armazenamento dos Objetos Digitais de Aprendizagem como URLs ou arquivos, junto com suas respectivas tags, metadados e definições. De forma rápida, organizada, intuitiva, com fácil gerenciamento, possibilitando buscas eficientes dos mesmos e o compartilhamento entre os usuários. Com o intuito de melhorar a experiência do usuário com os OAs, estimulando sua utilização e de ajudar o processo de aprendizagem tornando-o mais diversificado, amplo e otimizado.

O presente trabalho é composto pelos seguintes tópicos: o referencial teórico onde é discutido trabalhos existentes relacionados com o que será desenvolvido, a metodologia onde se define o método científico e o processo de desenvolvimento de software utilizado, o desenvolvimento onde são apresentadas as ferramentas e como a aplicação foi desenvolvida, e por fim as considerações finais.

2 Referencial Teórico

O crescimento das tecnologias de comunicação tornou-se parte do cotidiano, sendo ainda mais notável na popularização dos computadores e da internet, tanto que a sociedade moderna acaba sendo influenciada a adequar constantemente o seu estilo de vida. Como Teixeira e Brandão (2003) definem:

> O papel de destaque das novas tecnologias de informação na sociedade atual é atribuído à valorização da informação. Assim, tudo aquilo que potencialize o seu manuseio representa um elemento importante nesse processo, no qual a informação emerge como matéria-prima e a tecnologia, como um meio de agir sobre ela. Nesse sentido, podem-se apontar tais tecnologias como as principais propulsoras e mantenedoras da atual sociedade.

Soares, Filho e Machado (2003) afirmam que "o objeto de aprendizagem tem a propriedade de, quando manipulado dentro de um contexto de busca de conhecimento, servir de mediação e facilitação para a formação e consolidação de um saber novo".

A respeito dos metadados Silva, Café e Catapan (2010) levantam pontos importantes:

> A importância da classificação de metadados e do armazenamento dos objetos de aprendizagem em "um repositório integrável a um sistema de gerenciamento de aprendizagem" é ressaltada por Tarouco, Fabre e Tamusiunas (2003), quando defendem que uso de metadados traz benefícios relacionados à acessibilidade, interoperabilidade e durabilidade. Seu uso para descrever o OA é uma excelente ferramenta para documentar tais objetos, fornece uma organização para o repositório, permite que os usuários do OA os pesquisem e localizem. Para Guzmán (2005), os metadados são especialmente úteis quando os recursos não são textuais e não podem ser indexados por sistemas automatizados, como recursos multimídia ou de áudio.
>
> Os metadados permitem a descrição e posterior recuperação para reutilização dos objetos de aprendizagem nos repositórios desenvolvidos para esse fim, ou seja, os metadados tornam os objetos de aprendizagem acessíveis, segundo Dinis e Silva (2006), representam as informações estruturadas que descrevem, explicam e possibilitam a localização e recuperação da OA. O papel dos metadados seria promover a identificação e permitir que o OA compartilhe, integre, use, reutilize, gerencie e recupere com mais eficiência. Metadados agem como organizadores e facilitadores na recuperação da OA.
>
> Os metadados, segundo Marchi e Costa (2004), "podem ser comparados a um sistema de rotulagem que descreve o recurso, seus objetivos e características, mostrando como, quando e por quem o recurso foi armazenado e como é formatado." Metadados, para esses autores "são essenciais para entender o recurso armazenado, eles descrevem informações semânticas sobre o recurso."

No entanto, a maior parte das pesquisas foca nas questões tecnológicas associadas à produção, à busca da reusabilidade e da padronização. Mais recentemente, o foco passa para a constituição de repositórios e referatórios, onde os OAs possam ser localizados, referenciados e reutilizados. (GALAFASSI; GLUZ; GALAFASSI, 2013)

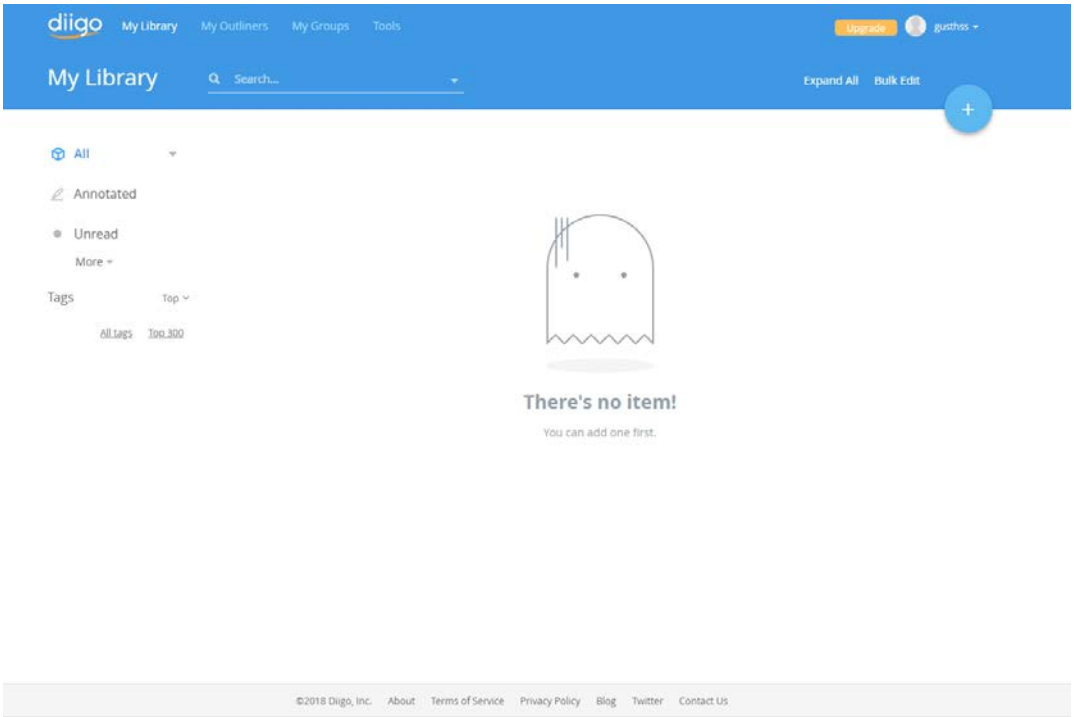
Uma aplicação existente com um propósito parecido do trabalho a ser desenvolvido, é chamada Diigo (*Digest of Internet Information, Groups and Other Stuff*). Trata-se de uma ferramenta multi-gestão de conhecimento pessoal, sendo um site de *bookmarking* social com recursos de anotação. *Bookmarking* social é um método para os usuários da Internet organizarem, armazenarem, gerenciarem e procurarem marcadores de recursos *on-line*. Ao contrário do compartilhamento de arquivos, os recursos em si não são compartilhados, apenas os marcadores que os referenciam.

O Diigo permite aos seus usuários as opções de marcar, destacar, arquivar, pesquisar e acessar informações encontradas na web. É possível também adicionar notas adesivas interativas a qualquer página da Web e escrever comentários pessoais para análise posterior.

O *bookmarking* social fornece aos leitores eletrônicos a capacidade de criar, compartilhar e interagir sobre o conteúdo. Os usuários podem optar por escrever notas particulares ou comentários públicos em páginas da web selecionadas. Combinando bookmarking social, recortes, anotações em sites, marcação, pesquisa de texto completo, fácil compartilhamento e interações.

A Figura 1 ilustra a página principal da plataforma citada.

Figura 1 – Página da bliblioteca pessoal do Diigo



Fonte: https://www.diigo.com/user/gusthss. (2018)

Como ferramenta de *bookmarking* o Diigo oferece as funcionalidades essenciais, porém no âmbito dos objetos educacionais faltam pontos fundamentais; um sistema de avaliação do conteúdo, opções para por exemplo, definição dos componentes de um Objeto de Aprendizagem

e a forma de relacionar os materiais, são fatores importantes que não são abordados por esta plataforma.

3 Metodologia

Com a finalidade de tentar suprir as necessidades existentes discutidas até então, dos OAs, a pesquisa tem como objetivo a aplicação de teorias e conhecimentos técnicos para o desenvolvimento de um produto capaz de prover uma solução prática: uma aplicação web.

Uma classificação teórica para a metodologia proposta é discutida por Garces (2010), quando define que uma pesquisa de desenvolvimento ou tecnológica tem o objetivo de resolver problemas concretos, propondo soluções mais imediatas, visando a aplicação das teorias às necessidades humanas.

Também chamada de tecnológica, produz produtos, processos e patentes, gera novas tecnologias e conhecimentos resultantes do processo de pesquisa. É comumente utilizada pelos Cursos de Engenharia da Produção e Ciência da Computação. Como seu objetivo principal é a produção de novos produtos e processos, os seus resultados nem sempre são divulgados, pois podem gerar novas patentes internacionais, então, neste caso, se diz que a pesquisa é reservada.

Além de gerar novos produtos e processos, produz conhecimentos que são veiculados diretamente pelos pesquisadores em empresas, ou através de congressos, feiras, seminários, consultorias para assistência técnica, publicação de boletins industriais e manuais técnicos. A pesquisa de desenvolvimento pode utilizar métodos de estudos descritivos ou experimentais.

É de grande importância, antes de começar desenvolver, definir de forma clara como será feito, com o que será feito, para evitar erros estruturais, retrabalho e melhor entendimento geral de quanto de esforço cada etapa representará. Considerando que a solução prática em seu estado final se resume em uma aplicação web, para embasar essa escolha e como será desenvolvida, segue sua definição junto com uma análise de suas vantagens e a metodologia de desenvolvimento de software escolhida.

3.1 Aplicação Web

Uma aplicação web, também chamada de aplicativo baseado na Web é qualquer aplicativo que usa um site como interface ou *front-end*. Os usuários podem acessar facilmente o aplicativo de qualquer computador conectado à Internet usando um navegador padrão. Isso contrasta com aplicativos de *desktop* tradicionais, instalados em um computador local.

Com aplicativos baseados na Web, os usuários acessam o sistema por meio de um ambiente uniforme - o navegador da web. Embora a interação do usuário com o aplicativo precise ser testada em diferentes navegadores da Web, o próprio aplicativo precisa ser desenvolvido apenas para um único sistema operacional, sem a necessidade de desenvolvê-lo e testá-lo em

todas as versões e configurações possíveis do sistema operacional. Isso torna o desenvolvimento e a solução de problemas muito mais fáceis.

A interface do usuário de aplicativos baseados na Web é mais fácil de personalizar do que nos aplicativos de *desktop*. Isso facilita a atualização da aparência do aplicativo ou a personalização da apresentação de informações para diferentes grupos de usuários.

É possível obter um nível maior de interoperabilidade entre aplicativos da Web do que com sistemas de desktop isolados, a arquitetura baseada na web possibilita a rápida integração de sistemas corporativos, melhorando o fluxo de trabalho e outros processos de negócios.

Com a abordagem baseada na web, a instalação e a manutenção também se tornam menos complicadas. Quando uma nova versão ou atualização é instalada no servidor *host*, todos os usuários podem acessá-la imediatamente e não há necessidade de atualizar o PC de todo e qualquer usuário em potencial. A implementação de novos softwares pode ser realizada com mais facilidade, exigindo apenas que os usuários tenham navegadores e *plug-ins* atualizados.

O aumento da capacidade do processador também se torna uma operação muito mais simples com aplicativos baseados na web. Se um aplicativo exigir mais energia para executar tarefas, somente o hardware do servidor precisará ser atualizado. A capacidade do software baseado na web pode ser aumentada pelo "clustering" ou executando o software em vários servidores simultaneamente. À medida que a carga de trabalho aumenta, novos servidores podem ser adicionados ao sistema facilmente.

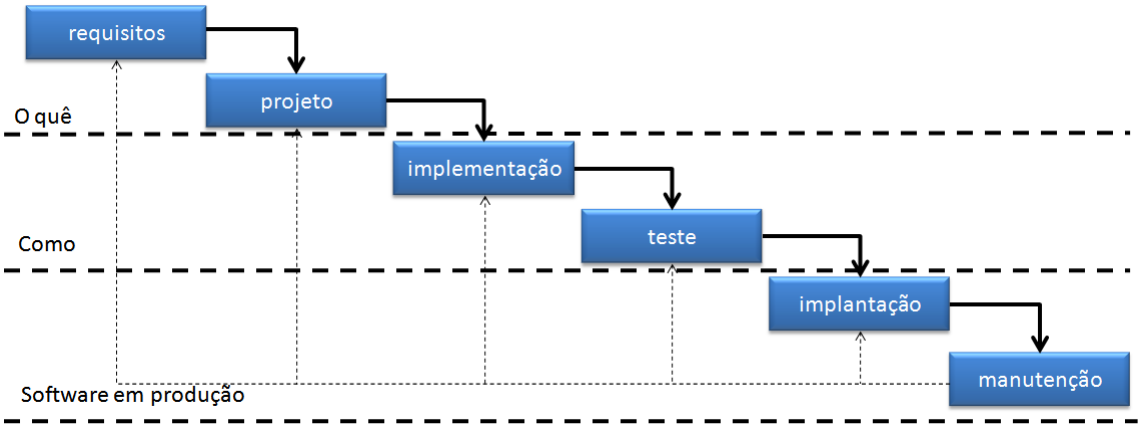
Os aplicativos baseados na Web normalmente são implantados em servidores dedicados, que são monitorados e mantidos por administradores de servidores, sendo mais eficaz do que monitorar centenas ou até milhares de computadores clientes, como é o caso dos aplicativos de *desktop*. Isso significa que a segurança é mais rigorosa e que qualquer violação potencial deve ser notada com mais rapidez.

3.2 Modelo em Cascata

O modelo em cascata é um processo de desenvolvimento de software que enfatiza a progressão lógica das etapas, o que significa que cada fase deve ser concluída antes que a próxima fase possa se sobrepor, pois o resultado de uma fase atua como entrada para a próxima. Caracterizando-se um modelo de ciclo de vida sequencial linear.

A Figura 2 ilustra o modelo:

Figura 2 – Modelo Cascata na Engenharia de Software



Fonte: https://engenhariasoftware.files.wordpress.com/2013/01/cascata.png. (2018)

- Levantamento e análise de requisitos - Todos os requisitos possíveis do sistema a ser desenvolvido são capturados nesta fase e documentados em um documento de especificação de requisitos.
- Projeto do Sistema - As especificações de requisitos da primeira fase são estudadas nesta fase e o projeto do sistema é preparado. Esse design de sistema ajuda a especificar requisitos de hardware e sistema e ajuda a definir a arquitetura geral do sistema.
- Implementação - O sistema é desenvolvido primeiramente em pequenos programas chamados unidades, que são integrados na próxima fase. Cada unidade é desenvolvida e testada por sua funcionalidade, que é chamada de Teste Unitário.
- Integração e Teste - Todas as unidades desenvolvidas na fase de implementação são integradas em um sistema após o teste de cada unidade. Após a integração, todo o sistema é testado quanto a faltas e falhas.
- Uma vez que a implantação do sistema o teste funcional e não funcional é feito; o produto é implantado no ambiente do cliente ou lançado no mercado.
- Manutenção - Existem alguns problemas que surgem no ambiente do cliente, para corrigi-los os *patches* são liberados. Também para melhorar o produto, algumas versões atualizadas são lançadas. A manutenção é feita para entregar essas mudanças no ambiente do cliente.
- Todas essas fases são cascateadas umas às outras, nas quais o progresso é visto como fluindo para baixo (como uma cachoeira) através das fases.

As vantagens do desenvolvimento em cascata são que ele permite a departamentalização e controle. Um cronograma pode ser definido com prazos para cada estágio de desenvolvimento

e um produto pode prosseguir através das fases do modelo de processo de desenvolvimento uma por uma.

O desenvolvimento inicia no conceito, passando pelo design, implementação, teste, instalação, solução de problemas e termina na operação e manutenção. Cada fase de desenvolvimento procede em ordem estrita.

Algumas das principais vantagens do Modelo Cascata são as seguintes:

- Simples e fácil de entender e usar
- Fácil de gerenciar devido à rigidez do modelo. Cada fase tem resultados específicos e um processo de revisão.
- Fases são processadas e completadas uma de cada vez.
- Funciona bem para projetos menores, onde os requisitos são bem compreendidos.
- Estágios claramente definidos.
- Fácil de organizar tarefas.
- O processo e os resultados são bem documentados.

4 Desenvolvimento

A presente pesquisa desenvolvimento seguiu as etapas descritas na metodologia e desenvolveu o programa com as seguintes ferramentas:

4.0.1 Servidor Web

Um servidor da Web é um software que administra os sites, distribuindo páginas da web conforme são requisitadas. O objetivo básico do servidor web é armazenar, processar e entregar páginas da web para os usuários. Essa intercomunicação é feita usando HTTP (*Hypertext Transfer Protocol*). Essas páginas da Web são principalmente conteúdo estático que inclui documentos HTML, imagens, folhas de estilo, teste etc. Além do HTTP, um servidor da Web também suporta protocolo SMTP (Protocolo de transferência de mensagens simples) e FTP (Protocolo de transferência de arquivos) para envio por e-mail e transferência de arquivos e armazenamento.

O principal trabalho de um servidor da Web é exibir o conteúdo do site. Quando alguém solicita um site adicionando o URL ou o endereço da web em uma barra de endereços do navegador (como o Chrome ou o Firefox), o navegador envia uma solicitação à Internet para visualizar a página da Web correspondente para esse site, endereço. Um Servidor de Nomes de Domínio (DNS) converte essa URL em um endereço IP (por exemplo, 192.168.216.345), que por sua vez aponta para um servidor da Web, e por fim servidor da Web é solicitado a apresentar o site de conteúdo ao navegador do usuário.

Todos os sites na Internet têm um identificador único em termos de um endereço IP. Esse endereço de Protocolo da Internet é usado para comunicação entre diferentes servidores na Internet. Atualmente, o servidor Apache é o servidor web mais comum disponível no mercado. O Apache é um software de código aberto que lida com quase 70% de todos os sites disponíveis atualmente. A maioria dos aplicativos baseados na Web usa o Apache como seu ambiente padrão do Servidor da Web. Outro servidor da Web que geralmente está disponível é o *Internet Information Service* (IIS).

4.0.2 PHP

PHP é um acrônimo recursivo para "PHP: *Hypertext Preprocessor*", sendo uma linguagem de script do lado do servidor que é incorporada em HTML. Ele é usado para gerenciar conteúdo dinâmico, bancos de dados, rastreamento de sessão e até mesmo construir sites inteiros de comércio eletrônico.

O PHP suporta um grande número de protocolos principais, como POP3, IMAP e

LDAP. O PHP4 adicionou suporte para arquiteturas de objetos distribuídos e Java (COM e CORBA), tornando o desenvolvimento de n camadas uma possibilidade pela primeira vez. Ele é integrado a diversos bancos de dados populares, incluindo MySQL, PostgreSQL, Oracle, Sybase, Informix e Microsoft SQL Server.

O PHP é muito utilizado para: executar funções do sistema, ou seja, a partir de arquivos em um sistema, ele pode criar, abrir, ler, escrever e fechá-las; manipular formulários, ou seja, coletar dados de arquivos, salvar dados em um arquivo e através de email você pode enviar dados, retornar dados para o usuário; adicionar, excluir, modificar elementos dentro do seu banco de dados; acessar variáveis de cookies e definir cookies; restringir os usuários para acessar algumas páginas do seu site; criptografar dados, entre outras funções.

4.0.3 Laravel

O Laravel é um framework PHP de código aberto, que é robusto e fácil de entender. Segue um padrão de projeto de *Model-View-Controller* (MVC). Reutiliza os componentes existentes de diferentes *frameworks*, o que ajuda na criação de um aplicativo da web, tornando-o mais estruturado e pragmático.

Ainda oferece um rico conjunto de funcionalidades que incorpora os recursos básicos de frameworks PHP como o CodeIgniter, Yii e outras linguagens de programação como o Ruby on Rails, quando se cria um aplicativo da web baseado nele a aplicação web torna-se mais escalável. Um tempo considerável é economizado ao projetar o aplicativo da web, uma vez que o Laravel reutiliza os componentes de outras estruturas no desenvolvimento de aplicativos da web, e inclui namespaces e interfaces, ajudando a organizar e gerenciar recursos.

Algumas de suas funcionalidades são: Bibliotecas e módulos integrados que ajudam no aprimoramento do aplicativo. Cada módulo é integrado ao gerenciador de dependência do Composer, o que facilita as atualizações. Recursos que ajudam nos testes, abordagem flexível ao usuário para definir rotas no aplicativo da web. O roteamento ajuda a dimensionar o aplicativo de uma melhor maneira e aumenta seu desempenho.

O Laravel incorpora um construtor de consultas que ajuda na consulta de bancos de dados usando vários métodos de cadeia simples. Ele fornece ORM (*Object Relational Mapper*) e implementação do *ActiveRecord* chamado *Eloquent*.

Utiliza do mecanismo *Blade Template*, uma linguagem leve usada para projetar blocos e layouts hierárquicos com blocos pré definidos que incluem conteúdo dinâmico. Ajuda a trabalhar com o PHP / HTML de uma melhor maneira, inclui uma classe de correio que ajuda a enviar e-mails com conteúdo e anexos avançados do aplicativo da web.

Facilita a criação de autenticação, pois inclui recursos como registro, senha esquecida e envio de lembretes de senha. Também ajuda a organizar a lógica de autorização e controla o acesso aos recursos. Na mais recente reformulação do Laravel, existe um código de validação

pré-construído. No desenvolvimento, a lógica da aplicação pode ser implementada em qualquer aplicativo usando os controladores ou diretamente nas declarações de rota usando a sintaxe. Possui serviços de fila que ajudam a concluir as tarefas de maneira mais fácil sem esperar que a tarefa anterior seja concluída.

4.0.3.1 Composer

O Composer é um gerenciador de pacotes em nível de aplicativo para a linguagem de programação PHP que fornece um formato padrão para gerenciar dependências de software PHP e bibliotecas necessárias. O Composer executa na linha de comando e instala dependências (por exemplo, bibliotecas) para um aplicativo. Ele também permite que os usuários instalem aplicativos PHP que estão disponíveis no "Packagist", que é seu repositório principal contendo pacotes disponíveis. Ele também fornece recursos de carregamento automático para bibliotecas para facilitar o uso de códigos de terceiros.

4.0.3.2 Artisan

A interface de linha de comando usada no Laravel é chamada de Artisan. Inclui um conjunto de comandos que ajudam na criação de um aplicativo da web. O Artisan permite executar a maioria das tarefas repetitivas e demoradas de programação que os desenvolvedores evitam executar manualmente. Ele é usado para criar um código base e a estrutura do banco de dados para facilitar seu gerenciamento.

4.0.4 Vagrant

O Vagrant é uma ferramenta para criar e gerenciar ambientes de máquinas virtuais em um único fluxo de trabalho. Com um fluxo de trabalho fácil de usar e foco em automação, o Vagrant reduz o tempo de configuração do ambiente de desenvolvimento, aumenta a paridade de produção e faz com que os trabalhos não fiquem restritos à configuração de uma máquina.

O Vagrant fornece ambientes fáceis de configurar, reproduzíveis e portáteis, construídos sobre tecnologia padrão da indústria e controlados por um único fluxo de trabalho consistente para ajudar a maximizar a produtividade e a flexibilidade do projeto. O Vagrant irá isolar as dependências e suas configurações em um único ambiente descartável, sem sacrificar nenhuma das ferramentas com as quais você está acostumado (editores, navegadores, depuradores, etc.).

Uma vez criado um único Vagrantfile, é preciso executar o comando “vagrant up” e tudo será instalado e configurado. Outros membros podem criar seus ambientes de desenvolvimento a partir da mesma configuração, portanto, seja um desenvolvimento no Linux, Mac OS X ou Windows, todos os membros da equipe estarão executando o código no mesmo ambiente, com as mesmas dependências, todos configurados da mesma forma.

4.0.5 Homestead

O Laravel Homestead é uma Vagrant Box pré-empacotada e oficial que fornece um ambiente de desenvolvimento sem precisar instalar o PHP, um servidor web e qualquer outro software de servidor em sua máquina local, sem se preocupar com bagunçar o seu sistema operacional. As Vagrant Box são completamente descartáveis. Se algo der errado, é possível destruir e recriar em minutos.

Softwares inclusos:

- Ubuntu 18.04
- Git
- PHP 7.3
- PHP 7.2
- PHP 7.1
- PHP 7.0
- PHP 5.6
- Nginx
- Apache (Optional)
- MySQL
- MariaDB (Optional)
- Sqlite3
- PostgreSQL
- Composer
- Node (With Yarn, Bower, Grunt, and Gulp)
- Redis
- Memcached
- Beanstalkd
- Mailhog
- Neo4j (Optional)

- MongoDB (Optional)
- Elasticsearch (Optional)
- ngrok
- wp-cli
- Zend Z-Ray
- Go
- Minio

4.0.6 MySQL

O MySQL tem uma grande parte do mercado online, onde o Microsoft SQL Server e o Oracle já foram dominantes, a arquitetura gratuita e a interface aprimorada do MySQL ao longo dos anos fizeram dele um dos três principais bancos de dados usados em todo o mundo. É usado por alguns dos maiores sites, incluindo Facebook e Pinterest. Pequenas *startups* que usam o WordPress usam o MySQL inerentemente, já que o aplicativo é integrado ao banco de dados.

Pequenos bancos de dados com apenas alguns registros normalmente irão funcionar bem, mesmo que sejam mal projetados. No entanto, se as tabelas do banco de dados crescerem para milhões de registros, o desempenho e a estabilidade do aplicativo podem ser afetados, o que afeta seus clientes e funcionários.

O MySQL é seguro, os administradores podem criptografar dados e configurar a autenticação para proteger todos os ativos da empresa, já que a proteção dos registros de seus clientes e funcionários deve ser uma grande preocupação

O MySQL é um banco de dados relacional, oferecendo a integridade de dados e trabalhando com conceito de um relacionamento de chave primária e externa. Também é um banco de dados transacional, o que significa que você pode reverter alterações em seu banco de dados.

A integridade dos dados é o que diferencia o MySQL e outros bancos de dados relacionais dos bancos de dados mais modernos, como o NoSQL. O MySQL requer que seus dados sejam mais estruturados, portanto, é um sistema de banco de dados confiável para pessoas que desejam proteger a estrutura e o relacionamento entre as tabelas.

Em empresas maiores, os bancos de dados MySQL e NoSQL funcionam juntos. O banco de dados MySQL armazena dados estruturados, como pedidos e informações de clientes, e o banco de dados NoSQL armazena dados não estruturados, como marketing e números de

tráfego. Você pode exportar dados de um banco de dados MySQL para um banco de dados NoSQL para trabalhar com eles e obter os melhores recursos.

É importante entender os relacionamentos e os dados relacionais para construir e manter de forma eficiente uma arquitetura de banco de dados MySQL. Se ela for projetada incorretamente, irá apresentar dificuldades com o desempenho e em manter seus dados.

4.1 ORM

Escrever um software não depende apenas de um código eficiente e bem estruturado, mas também de como esse código é capaz de gerenciar e interagir com dados importantes. Construir sistemas complexos requer que ambos sejam organizados e cooperativos.

Objetos em uma linguagem de programação orientada a objetos, contém dados e comportamento lógico, uma idéia baseada em princípios de engenharia de software. Enquanto isso, os dados modelados com essa lógica são melhores armazenados em bancos de dados relacionais (enraizados em princípios matemáticos) e classificados por colunas e linhas que dependem de sua própria linguagem específica de domínio para consultar essas informações. Os dados nessas tabelas fazem referência a outras tabelas com o uso de chaves estrangeiras (colunas designadas de uma tabela que apontam para uma conexão com outra tabela).

Para facilitar o acesso e o uso dessas informações, precisa-se de uma maneira de traduzi-las do formato de banco de dados relacional para objetos, o mapeamento relacional de objeto - ou ORM, é um padrão de design para converter (agrupar) os dados armazenados em um banco de dados relacional em um objeto que pode ser usado em uma linguagem orientada a objetos.

Mapeamento Objeto-Relacional (ORM) é uma técnica que permite consultar e manipular dados de um banco de dados usando um paradigma orientado a objeto. Ao falar sobre o ORM, a maioria das pessoas está se referindo a uma biblioteca que implementa a técnica Mapeamento Objeto-Relacional, daí a frase "um ORM".

Uma biblioteca ORM é uma biblioteca completamente ordinária escrita na sua linguagem de escolha que encapsula o código necessário para manipular os dados, para que você não use mais o SQL, permitindo interagir diretamente com um objeto na mesma linguagem que está usando.

4.2 Desenvolvimento

4.2.1 Requisitos

Os requisitos de software são a descrição de recursos e funcionalidades que o sistema deve ter, eles que transmitem as expectativas dos usuários do produto de software. Existem

dois tipos de classificação de requisitos, os Requisitos Funcionais (RF): qualquer requisito que especifique o que o sistema deve fazer, e os Requisitos Não-Funcionais (RNF): qualquer requisito que especifique como o sistema executa uma determinada função.

Um resumo dos RFs, e em alguns RNFs, o necessário para entender o ciclo de vida da aplicação. Os requisitos definidos para esta aplicação foram:

- É preciso uma tela inicial, onde o usuário pode cadastrar uma nova conta, logar com uma já existente ou ter a opção de recuperar seu acesso. Caso já esteja com uma sessão ativa (logado), será redirecionado para a página principal (*home*).
- A página principal (*home*) deve conter uma forma de acessar as informações do usuário, uma forma de deslogar e assim retornar para página inicial, a opção de adicionar um novo Url e seus respectivos campos (Metadados, Tags), uma forma de buscar pelos Urls, um campo para visualizar a busca, e uma forma de editar ou deletar um URL.
- Na página de informações do usuário é preciso ter uma forma de editar suas informações (p. ex. senha, nome).

Essa seria a versão mais básica e funcional da aplicação web em questão, a partir disso espera-se adicionar outras funções.

4.2.2 Viabilidade do Projeto

Após estudar as ferramentas a serem utilizadas, entendendo suas funcionalidades, verificou-se que seria viável realizar o projeto.

4.2.3 Implementação

4.2.3.1 Configurando o ambiente Homestead

Primeiramente foi necessário instalar o VirtualBox, e o Vagrant, no caso do windows, o SO presente na máquina de desenvolvimento, bastou apenas baixar o executável nos respectivos sites e executar a instalação.

Para melhor desempenho foi preciso ativar a virtualização de hardware (VT-x). Geralmente, ele pode ser ativado por meio do seu BIOS. Uma vez que o VirtualBox / VMware e o Vagrant foram instalados, foi preciso adicionar a box laravel / homestead à instalação do Vagrant usando o seguinte comando no terminal.

vagrant box add laravel/homestead

É possível instalar o Homestead clonando o repositório. Recomenda-se clonar o repositório em uma pasta Homestead dentro do seu diretório "home", pois a Homestead Box servirá como host para todos os projetos do Laravel:

git clone https://github.com/laravel/homestead.git ~/Homestead

Depois de clonar o repositório Homestead, foi preciso executar o comando bash *init.sh* no diretório Homestead para criar o arquivo de configuração Homestead.yaml.

init.bat

4.2.3.2 Configurando o provedor

A chave do provedor no arquivo Homestead.yaml indica qual provedor Vagrant deve ser usado: virtualbox, vmware_fusion, vmware_workstation, parallels ou hyperv. Pode-se definir isso para o provedor:

provedor: virtualbox

4.2.3.3 Configurando Pastas Compartilhadas

A propriedade folders do arquivo Homestead.yaml lista todas as pastas que são compartilhadas com o ambiente Homestead. Como os arquivos dentro dessas pastas são alterados, eles serão mantidos em sincronia entre sua máquina local e o ambiente Homestead. É possível configurar quantas pastas compartilhadas forem necessárias:

pastas:
- mapa: / code
para: / home / vagrant / code

Criando apenas alguns sites, esse mapeamento genérico funcionará bem. No entanto, como o número de sites pode crescer, isso resultará em uma queda de desempenho. Esse problema pode ser aparente em máquinas low-end ou projetos que contêm um grande número de arquivos. Para contornar esse problema, é necessário mapear cada projeto para sua própria pasta Vagrant:

pastas:
- mapa: / code / project1
para: / home / vagrant / code / project1
- mapa: / code / project2
para: / home / vagrant / code / project2

4.2.3.4 Configurando Sites Nginx

A propriedade sites permite mapear facilmente um “domínio“ para uma pasta no ambiente Homestead. Um exemplo de configuração do site está incluído no arquivo Homestead.yaml.

Novamente, tem-se a opção de adicionar tantos sites ao ambiente de Homestead quanto necessário. O Homestead serve como um ambiente conveniente e virtualizado para todos os projetos do Laravel em que se trabalha:

sites:

- mapa: homestead.test

para: / home / vagrant / code / Laravel / public

Caso altere-se a propriedade sites depois de iniciar a Homestead Box, deverá executar novamente o *vagrant reload –provision* para atualizar a configuração do Nginx na máquina virtual.

4.2.3.5 O arquivo de hosts

Foi preciso adicionar os "domínios" para os sites Nginx no arquivo de hosts na máquina. O arquivo hosts redirecionará as solicitações dos sites da Homestead para a máquina Homestead. No Mac e no Linux, esse arquivo está localizado em */ etc / hosts*. No Windows, ele está localizado em *C: Windows System32 drivers etc hosts*. A linhas adicionadas a este arquivo serão semelhantes à seguinte:

192.168.10.10 homestead.test

Depois de adicionar o domínio ao arquivo de hosts e iniciar a Box do Vagrant, foi possível acessar o site através do navegador:

http: //homestead.test

4.2.3.6 Iniciando a Vagrant Box

Com o Homestead.yaml configurado, executou-se o comando *vagrant up* do seu diretório Homestead. O Vagrant inicializará a máquina virtual e configurará automaticamente as pastas compartilhadas e sites Nginx.

Para desligar a máquina, usou-se o comando

vagrant halt

Para destruir a máquina, usou-se o comando

vagrant destroy –force

4.3 Aplicação

O ambiente Homestead por padrão, possui o MySQL instalado e pré-configurado, para criar suas tabelas e relações dois conceitos são importantes, Migrations e Models.

As *migrations* foram usadas para o controle de versão do banco de dados, permitindo modificar e compartilhar facilmente o esquema do banco de dados do aplicativo. As migrations possibilitaram o emparelhamento com o construtor de esquemas do Laravel criando o esquema do banco de dados.

As principais *migrations* usadas foram as dos usuários, de recuperação de senha, das URLs, e das tags. Para cada tabela do banco de dados foi criado um *Model* correspondente que é usado para interagir com essa tabela. Os *Models* permitiram consultar dados em suas tabelas, além de inserir novos registros na tabela.

Para criar as relações entre as tabelas foi utilizado as relações do Eloquent ORM:

- One To One
- One To Many
- Many To Many
- Has Many Through
- Polymorphic Relations
- Many To Many Polymorphic Relations

Relações que possibilitaram o relacionamento entre as URLs, tags e metadados.

Para a autenticação foram usados os controladores de autenticação pré-construídos, que estão localizados no namespace App / Http / Controllers / Auth. O *RegisterController* manipula o novo registro de usuário, o LoginController manipula a autenticação, o *ForgotPasswordController* manipula os links de e-mail para redefinir as senhas e o *ResetPasswordController* contém a lógica para redefinir as senhas. Cada um desses controladores usa uma característica para incluir seus métodos necessários.

Para a visualização das páginas e rotas entre elas, foi utilizado o conceito de *Routing* e *Views*. As informações são enviadas utilizado um Forms com *middlewares* para autenticação das mesmas.

A desenvolvimento teve como foco o *back-end*, concentrando-se na lógica de criação, integração, gerenciamento, segurança dos dados.

As figuras 3 a 8 mostram a interface da aplicação.

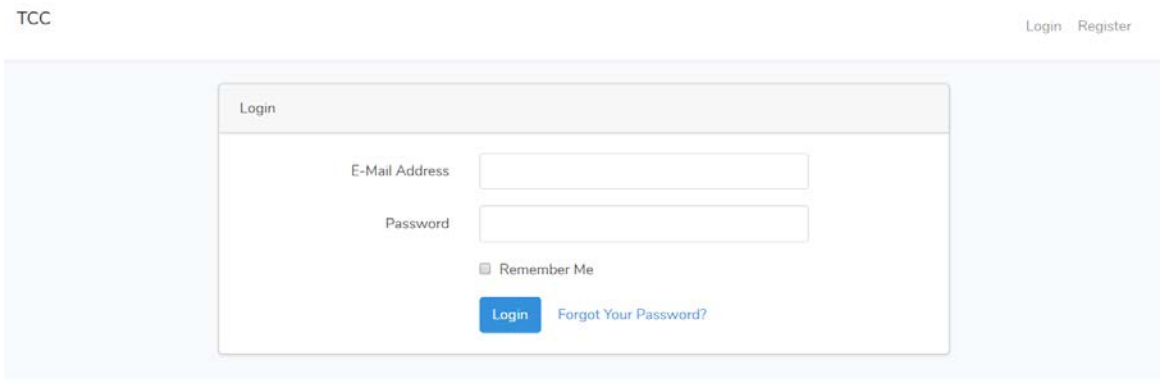
A Figura 3 representa a página inicial da aplicação que foi desenvolvida para organizar OAs encontrados na web, onde é possível ver o logo e as opções de login e cadastro.

Figura 3 – Página inicial da aplicação desenvolvida

Fonte: elaborada pelo autor.

A Figura 4 representa a página inicial de Login, onde é possível logar com uma conta já registrada e caso precise recuperar a senha.

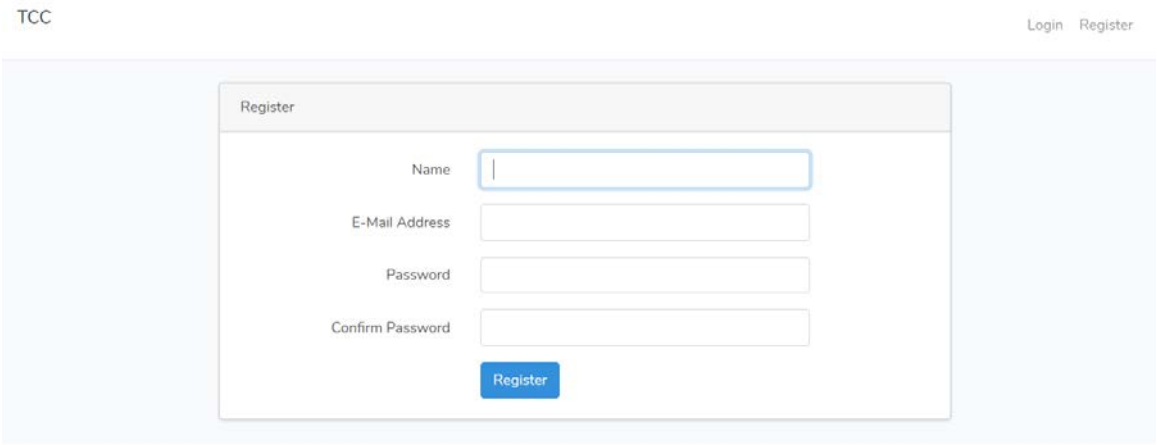
Figura 4 – Página de login da aplicação desenvolvida



Fonte: elaborada pelo autor.

A Figura 5 representa a página inicial de cadastro, que possibilita cadastrar uma nova conta de usuário utilizando um nome, email válido, e senha.

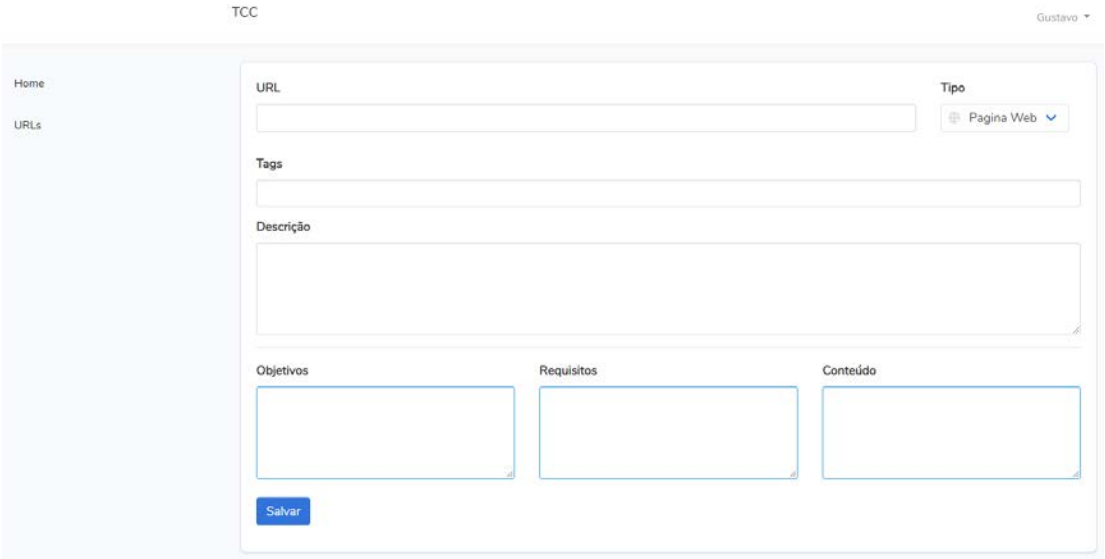
Figura 5 – Página de cadastro da aplicação desenvolvida



Fonte: elaborada pelo autor.

A Figura 6 representa a página Home, onde é preciso estar logado para poder acessá-la e poder navegar pelas opções do aplicativo pela barra lateral.

Figura 6 – Página *Home* da aplicação desenvolvida



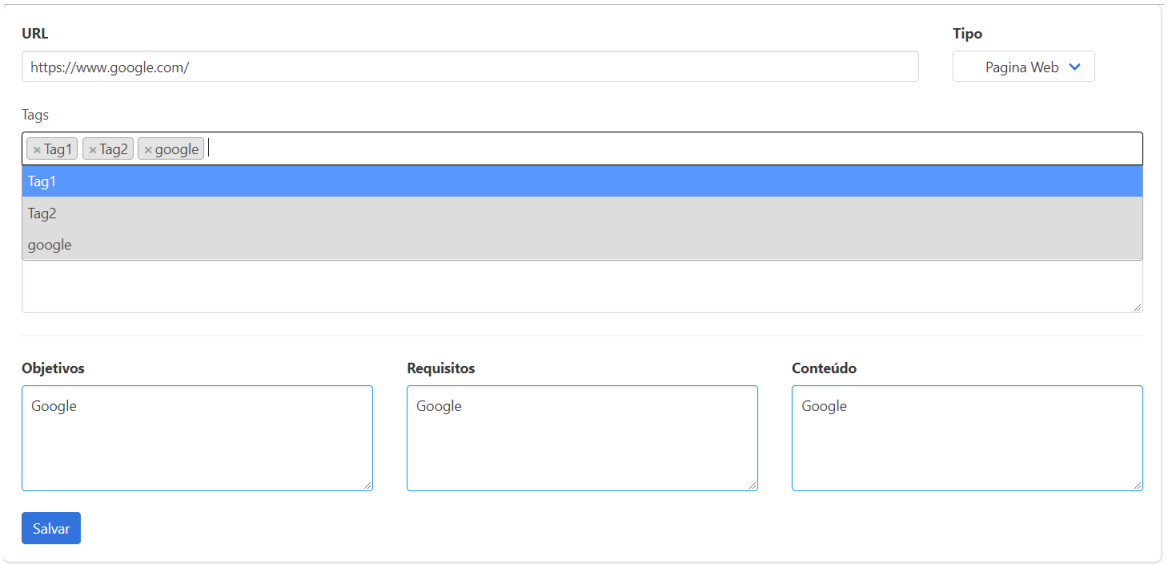
Fonte: elaborada pelo autor.

A Figura 7 representa a página Home com o foco na janela de adição de novos URLs, os campos existentes são os de:

- URL: Campo obrigatório, permitindo apenas entradas de URLs válidos.
- Tipo: Campo com opções fixas, responsável por definir o tipo do URL a ser salvo.
- Tags: Campo designado as tags, permite utilizar de tags já usadas em outros URLs, e criar novas.

- Descrição: Campo onde é possível definir uma descrição para o URLs, essa descrição será mostrada junto com o URL no página de URLs.
- Objetivos: Campo destinado a descrever os objetivos do conteúdo que se encontra no URL.
- Requisitos: Os requisitos para compreender o conteúdo que se encontra no URL.
- Conteúdo: Um resumo do conteúdo em si que se encontra no URL.

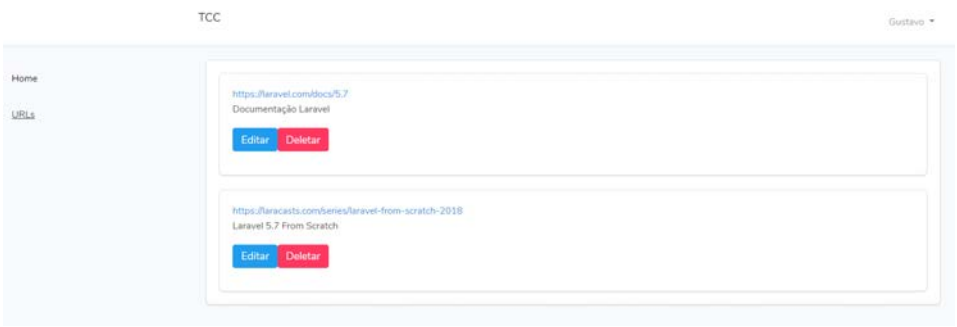
Figura 7 – Página inicial com o foco na janela de adição de URLs



Fonte: elaborada pelo autor.

A Figura 8 representa a página de URLs relacionadas com o usuário logado, nesta página é possível visualizar os URLs salvos e suas respectivas descrições, editar e deletar os mesmos.

Figura 8 – Página de URLs



Fonte: elaborada pelo autor.

A aplicação iniciou a fase de testes para que posteriormente seja realizada as fases de implantação e manutenção, como previsto na metodologia.

5 Considerações Finais

A crescente quantidade de conteúdos digitais educacionais, traz consigo a necessidade de melhor organização desses materiais. Para melhor aproveitamento das vantagens que um OA Digital pode oferecer, é preciso especificá-los coerentemente com a sua finalidade didática.

A definição formal analisada na introdução e utilizada como base dos componentes que um OA deveria possuir, ajudou nessa questão. Porém acompanha um grau de complexidade na hora de definir cada um dos elementos necessários. Uma das dificuldades é que os critérios usados nessas definições a princípio, ficam sob responsabilidade do usuário, o que pode variar em termos de qualidade e precisão. Uma forma de melhorar isso seria implementar uma maneira de avaliação colaborativa, onde seria possível corrigir ou aprimorar outras avaliações.

É recomendado que as *tags* a serem utilizadas nos URLs sejam de preferência palavras existentes, escritas da forma correta e mais comumentes usadas, já que a mesma tag pode ser utilizada em diferentes OAs, e na hora da busca não existam tags com o mesmo significado porém escritas de forma diferente. Por exemplo cachorro e cão.

O trabalho possibilitou uma forma rápida para armazenar os OAs, é preciso apenas estar logado na página, copiar sua URL e definir seus campos para poder salvar o mesmo. A complexidade de uso para quem está acostumado com páginas web, pode ser considerada baixa, uma vez que a aplicação é intuitiva e a interface é simples.

Analisando o uso geral, a aplicação garante um armazenamento e definição consistentes. A procura, avaliação, e compartilhamento de novos AOs fica a critério do usuário, o que causa uma flutuação na qualidade desses serviços oferecidos.

A utilização do Laravel reduziu significantemente a codificação repetitiva, melhorando o desenvolvimento do processo e assegurando uma codificação adequada ao criar aplicativos da web.

As estruturas PHP aceleraram a construção do aplicativo e também forneceu segurança em sua codificação. Existem vários frameworks PHP disponíveis e programadores selecionam frameworks de acordo com suas necessidades.

O ambiente de desenvolvimento Homestead, ajudou significativamente no processo economizando tempo de configuração da máquina local, e permitindo desenvolver de várias computadores diferentes.

O sistema de busca e ordenamento está em desenvolvimento e através dele espera-se filtrar os URLs salvos pelas tags, tipos, descrição e data do armazenamento. Futuramente outra função a ser implementada seria a de compartilhamento dos URLs pelos usuários, e grupos que permitam um repositório colaborativo.

Futuramente uma análise do *feedback* de uso dos usuários poderá contribuir com melhorias a serem feitas. Investimentos e parceria com mais desenvolvedores poderiam ajudar consideravelmente no aprimoramento da aplicação.

Referências

ALFREDOSANCHEZ, J.; PEREZ-LEZAMA, C.; STAROSTENKO, O. A formal specification for the collaborative development of learning objects. In: ELSEVIER (Ed.). *4th World conference on educational technology researches, WCETR-2014*. [S.l.]: Procedia - Social and Behavioral Sciences, 2014.

DINIS, P. C.; SILVA, A. Análise funcional de plataformas de objectos de aprendizagem. 2006.

GALAFASSI, F. P.; GLUZ, J. C.; GALAFASSI, C. Análise crítica das pesquisas recentes sobre as tecnologias de objetos de aprendizagem e ambientes virtuais de aprendizagem. *Revista Brasileira de Informática na Educação - RBIE)*, v. 21, n. 3, 2013.

GARCES, S. B. B. *Classificação e Tipos de Pesquisas*. [S.l.], 2010.

GUZMáN, C. L. *Los Repositorios de Objetos de Aprendizaje como soporte a un entorno e-learning*. Tese (Doutorado) — Universidad de Salamanca, 2005. Disponível em: <https://gredos.usal.es/jspui/bitstream/10366/56649/1/DIA_Repositoriosobjetos.pdf.pdf>. Acesso em: 11 set. 2018.

MARCHI, A. C. B. D.; COSTA, A. C. da R. Uma proposta de padrão de metadados para objetos de aprendizagem de museus de ciências e tecnologia. In: . [S.l.: s.n.], 2004.

MENDES, R. M.; SOUZA, V. I.; CAREGNATO, S. E. A propriedade intelectual na elaboração de objetos de aprendizagem. In: *Encontro Nacional de Ciência da Informação*. Salvador, Bahia: [s.n.], 2004.

SILVA, E. L. da; CAFé, L.; CATAPAN, A. H. Os objetos educacionais, os metadados e os repositórios na sociedade da informação. *Ciência da Informação*, v. 29, n. 3, 2010.

SOARES, C.; FILHO, S.; MACHADO, E. de C. *O computador como agente transformador da educação e o papel do objeto de aprendizagem*. Dissertação (Mestrado) — Universidade Federal do Ceará - UFC, 2003. Disponível em: <http://www.abed.org.br/seminario2003/texto11.htm>. Acesso em: 05 set. 2018.

TAROUCO, L. M. R.; FABRE, M.-C. J. M.; TAMUSIUNAS, F. R. Reusabilidade de objetos educacionais. *Revista Novas Tecnologias na Educação*, p. 1–11, 2003.

TEIXEIRA, A. C.; BRANDãO, E. J. R. Internet e democratização do conhecimento: repensando o processo de exclusão social. In: *WIE 2002 - SBC*. [S.l.: s.n.], 2003.

WILEY, D. *Learning object design and sequencing theory*. Tese (Doutorado) — Brigham Young University, 2000. Disponível em: <https://opencontent.org/docs/dissertation.pdf>. Acesso em: 01 set. 2018.