UNIVERSIDADE ESTADUAL DE MONTES CLAROS

Centro de Ciências Exatas e Tecnológicas

Curso de Bacharelado Sistemas de Informação

Ryan Lucas Silva Lemos

APLICAÇÃO DO EXTREME PROGRAMMING NO DESENVOLVIMENTO DE UM AMBIENTE WEB DE ENSINO E APRENDIZAGEM DA LÍNGUA INGLESA

Montes Claros/MG

Novembro/2018

**Ryan Lucas Silva Lemos**

**APLICAÇÃO DO EXTREME PROGRAMMING NO DESENVOLVIMENTO DE UM AMBIENTE WEB DE ENSINO E APRENDIZAGEM DA LÍNGUA INGLESA**

Projeto de Pesquisa apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Estadual de Montes Claros como exigência para obtenção de nota na disciplina de Projeto de Trabalho de Conclusão de Curso.

Orientadora: Prof.ª. CHRISTINE MARTINS DE MATOS, MESTRA.

**Montes Claros/MG**

**Novembro/2018**

LISTA DE ABREVIATURAS E SIGLAS

BPMN *Business Process Model and Notation*

CALL *Computer Assisted Language Learning*

CCAA Centro de Cultura Anglo Americana

CSS *Cascading Style Sheets*

EAD Educação a distância

HTML *Hyper Text Markup Language*

IDE *Integrated Development Environment*

IEEE *Institute of Eletrical and Eletronics Engineers*

IHC Interação Humano-Computador

ILC *International Language Center*

JS JavaScript

MVC *Model*, *View* e *Controller*

PHP *Hypertext PreProcessor*

SGBD Sistema de Gerenciamento de Banco de Dados

SPA *Single Page Application*

SQL *Structured Query Language*

TS TypeScript

Unimontes Universidade Estadual de Montes Claros

URL *Uniform Resource Locator*

XP *eXtreme Programming*

Sumário

1 INTRODUÇÃO 6

2 Tema e problema 8

2.1 Tema 8

2.2 Problema 8

3 Justificativa 9

4 Objetivos 11

4.1 Objetivo Geral 11

4.2 Objetivos Específicos 11

5 Referencial teórico 12

5.1 Educação a distância – ambiente virtual 12

5.1.1 Sistemas de apoio ao ensino/aprendizagem – Classroom 12

5.1.2 Metodologias/sistemas de apoio de ensino de idiomas 13

5.2 Desenvolvimento e tecnologias de sistemas Web 16

5.2.1 Criptografia e controle de acessos 17

5.2.2 Interação humano computador (IHC) 18

5.2.3 Engenharia de Software 19

5.2.3.1 Modelagem de processos com o *Business Process Model and Notation* (BPMN) 21

5.2.3.2 Metodologias de desenvolvimento 25

5.2.3.3 Modelo Cascata 26

5.2.3.4 Metodologia Ágil 27

5.2.3.5 *Extreme Programming* (XP) 28

5.2.4 Tecnologias para desenvolvimento WEB 30

5.2.4.1 *Hyper Text Markup Language* (HTML) 31

5.2.4.2 *Cascading Style Sheets* (CSS) 32

5.2.4.3 JavaScript (JS) 34

5.2.4.4 *Hypertext PreProcessor* (PHP) 35

5.2.4.5 *Framework* Laravel 36

5.2.5 Sistema de Gerenciamento de Banco de Dados (MySQL) 37

6 PROCEDIMENTOs METODOLÓGICOs 39

6.1 CLASSIFICAÇÃO DA PESQUISA/MÉTODO 39

6.1.1 Pesquisa Exploratória 39

6.2 TÉCNICAS DE PESQUISA 40

6.2.1 Pesquisa Bibliográfica 40

6.2.2 Observação 41

6.2.3 Entrevista 42

6.3 MÉTODOS 43

7 Cronograma 44

BIBLIOGRAFIA 46

Apendice A - carta de pedido de permissão para uso de informações da escola International language center 49

# INTRODUÇÃO

A tecnologia tem influenciado e modificado a maneira em que se ensina. Com o advento de uma geração conectada, crianças e jovens tem acesso cada vez mais cedo a tecnologias. Isso faz com que se estabeleça novas possibilidades de se adquirir conhecimento, por um outro meio diferente ao convencional. “A integração das tecnologias digitais na educação precisa ser feita de modo criativo e crítico, buscando desenvolver a autonomia e a reflexão dos seus envolvidos, para que não sejam apenas receptores de informações”. (BACICH *et al.*, 2015, p. 29). Portanto a tecnologia pode ser utilizada em prol da educação, mas para que trabalhem juntos deve-se ter um cuidado para que os utilizadores da tecnologia passem a interagir de maneira a não somente receber informações, mas também colaborar e aprender através do uso (BACICH *et al.*, 2015).

O estudo de línguas estrangeiras é disseminado globalmente, sendo requisitado em vários aspectos profissionais, educacionais, de pesquisa e de interação das pessoas nos diferentes países. No estudo de idiomas o atrelamento a tecnologias não foi diferente. Surgiram sistemas (de escolas físicas ou não) com o intuito de auxiliar nos processos de aprendizagem de uma língua. Buscou-se contemplar todas as etapas de aprendizado de um idioma, desde a escrita, leitura, até o entendimento e fala de uma língua. Diversas escolas se prontificam a ensinar os idiomas com diferentes métodos de ensino, material e apoio informatizados, porém, muitas escolas não contam com apoio computacional para auxiliar no processo de ensino e aprendizagem.

Pensando nisso foi identificada uma escola que não possuía esse apoio informatizado. A escola identificada é a *International Language Center* (ILC) que é situada em Montes Claros, Minas Gerais. A ILC foi escolhida pelo fato de não dispor de sistema de auxílio no ensino/aprendizagem, detinha somente de um *site* desenvolvido pela Empresa Júnior do curso de Sistemas de Informação da Universidade Estadual de Montes Claros (Unimontes), a INFOBITS, com a *Uniform Resource Locator* (URL) <http://www.joininit.com.br/>, contendo informações básicas da escola, sendo uma página estática, não tendo recursos de auxílio ao ensino de idiomas. E, portanto, através deste trabalho, buscou-se desenvolver um ambiente *web* para sanar a falta desse apoio informatizado e com esse ambiente, apoiar professores e alunos. Nesse ambiente é possível ao professor auxiliar seus alunos, alunos esses que serão divididos por suas turmas como no Google ClassRoom (subseção 5.1.2). Seja no compartilhamento de materiais ou buscando sanar dúvidas sobre conteúdos. Para os alunos espera-se que haja envolvimento e aprendizado quanto aos conteúdos disponibilizados no ambiente virtual. Acredita-se que com o ambiente, a interação aluno-professor e aluno-turma transpassará da sala de aula, possibilitando interação de aprendizagem de maneira informatizada.

Ainda para o desenvolvimento deste ambiente usou-se de uma metodologia de desenvolvimento ágil, a metodologia *eXtremme Programming* (XP) para apoiar e agilizar o processo de desenvolvimento do sistema.

Portanto o tema deste trabalho pode ser descrito como o desenvolvimento web com técnicas ágeis, por aliar o desenvolvimento de um ambiente web utilizando-se da metodologia ágil XP. Então busca-se resolver o seguinte problema, de entender como o XP pode apoiar no processo de desenvolvimento de um ambiente web para ensino e aprendizagem de inglês. O objetivo geral deste trabalho é então desenvolver ambiente *web* com técnicas ágeis para auxiliar no processo de ensino e aprendizagem de língua inglesa para a escola *International Language Center* (ILC). E como objetivos específicos disponibilizar materiais e exercícios para os alunos no ambiente desenvolvido; apresentar um calendário com datas de exercícios, provas e eventos para os alunos e identificar conteúdos de maior deficiência a partir de gráficos de desempenho.

Este trabalho está dividido de maneira a contemplar a introdução (capítulo 1), o tema e o problema que este trabalho busca solucionar (capítulo 2); o porque este trabalho se justifica, apontando o que se espera colaborar com este trabalho (capítulo 3); os objetivos a serem alcançados no momento de conclusão deste trabalho (capítulo 4); a base teórica para o desenvolvimento deste trabalho (capítulo 5); os métodos utilizados para o alcance dos objetivos deste trabalho, bem como a listagem das tarefas a serem feitas para concepção deste trabalho (capítulo 6); o cronograma proposto (capítulo 7) para execução das tarefas descritas no capítulo 6; e por último a bibliografia em que este trabalho se baseia.

# Referencial teórico

“O referencial teórico é o alicerce para o desenvolvimento de uma monografia.” (DIAS; SILVA, 2010, p. 31). Assim pode-se entender o quão importante é descrever aquilo que servirá de base para a concepção de um trabalho científico. Severino (2002) define essa etapa como sendo a busca por documentos que apresentem conteúdo relevante e que fazem sentido ao tema discutido no trabalho científico. Essa busca e escrita de conteúdo serve para que o leitor tenha uma visão inicial do que será abordado e tenha base para compreender os conceitos que são abordados no decorrer do trabalho científico.

## Educação a distância – ambiente virtual

A maneira de se ensinar mudou adequando-se as necessidades e mudanças (sociais, tecnológicas, etc.). Um exemplo disso é o da Educação a Distância (EAD) que surgiu no Brasil por volta de 1900, por meio de cursos oferecidos por correspondência, a fim de possibilitar mais conveniência no estudo (ALVES *et al.*, 2009).

Com o alcance da internet e a maior acessibilidade a computadores e *smartphones* a EAD se propagou pelo território nacional (ALVES *et al.*, 2009). E tem sido de grande importância tendo em vista que pessoas em localidades sem acesso direto ao ensino presencial, podem estudar e alcançar uma formação (MELO NETO *et al.*, 2012).

Nas seções subsequentes são descritos alguns exemplos de sistemas que utilizam da EAD e de processos informatizados para possibilitar e automatizar o processo de ensino.

### Metodologias/sistemas de apoio de ensino de idiomas

A tecnologia influencia e molda a maneira em que o ensino é feito (ALVES *et al.*, 2009). Isso não é diferente para o ensino de idiomas, uma vez que as escolas têm buscado se adequar e inserir tecnologias nos seus processos de ensino, e soluções independentes tem surgido para auxiliar no aprendizado de línguas. Essas escolas e tecnologias são descritas nos próximos parágrafos, apresentando suas principais características.

Como exemplo de escola que tem se moldado a se adequar a novas tecnologias toma-se a *Wizard by Pearson*, que é uma escola de idiomas internacional (possui sedes em outros países além do Brasil) (WIZARD, 2017b). A instituição conta com uma metodologia de ensino que alia a clássica abordagem de sala de aula (chamada pela *Wizard* de *Connections*) ao uso de tecnologias de apoio informatizado (chamada de *Interactive*). Na *connections* os alunos vão a sala de aula e interagem com outros alunos e professores, e são guiados pelo professor. Já no *interactive* o aluno recebe apoio informatizado por meio de um *tablet* denominado Wiz.tab. Para auxiliar a pronúncia dos alunos dispõe-se de uma caneta (denominada Wiz.pen), que faz a leitura de palavras, frases e expressões contidas no material do aluno (WIZARD, 2017a). A Figura 1 demonstra o modelo de aprendizagem da Wizard com as características anteriormente descritas.

Figura - Modelo de aprendizagem da Wizard



Fonte: WIZARD, 2017a, p.3.

A escola ainda conta com uma aplicação *mobile* e um ambiente *Web* denominados Wiz.me, que servem de apoio aos alunos quando estão fora da sala, auxiliando-os na pronúncia e escrita (WIZARD, 2017a). A Figura 2 descreve algumas funcionalidades da versão móvel dessa aplicação.

Figura - Funcionalidades do Wiz.me



Fonte: WIZARD, 2017a, p.4.

Outra escola que interliga tecnologia e ensino de idiomas é a escola Centro de Cultura Anglo Americana (CCAA). A CCAA detém um espaço virtual denominado espaço CCAA aluno (CCAA, sd.). Nele o aluno da escola terá acesso a conteúdo para *tablet*, como textos, áudios e vídeos (CCAA, sd.). Outro recurso disponível é o *Computer Assisted Language Learning* (CALL) que se dá por um “Software educacional que permite que o aluno realize seus exercícios escritos utilizando o computador. É uma forma rápida, fácil, interativa e agradável de fixar o conteúdo aprendido em sala de aula” (CCAA, sd., p. 1). O espaço virtual ainda conta com atividades complementares para auxílio de escrita, pronúncia e leitura.

Já em questão de aplicações independentes para auxílio de idiomas tem-se o exemplo do Babbel, que é uma aplicação disponível para dispositivos móveis e *Web*. Seu objetivo é o auxílio da compreensão, escrita e fala de diversos idiomas como inglês, português, espanhol, alemão, holandês, entre outros. Contempla diversos níveis de conhecimento, indo do básico ao avançado. Apesar da aplicação ser gratuita, somente alguns níveis estão disponíveis de forma gratuita, sendo necessário assim o pagamento de planos para a liberação de todo o conteúdo disponível (BABBEL, 2018) conforme descrito na Figura 3.

Figura - Preços do Babel



Fonte: BABBEL, 2018, p.1.

Outra aplicação semelhante ao Babbel é o Duolingo que possibilita o aprendizado de idiomas de maneira gratuita. É possível o estudo de línguas como inglês, espanhol, francês, alemão, dentre outras. Apresenta-se alguns conceitos de gamificação, como o avanço de níveis e as vidas (corações). Como nos *games*, ao fazer algo de errado ocorre penalização ou eliminação de uma vida, no Duolingo ao errar uma resposta de uma questão um coração é perdido (DUOLINGO, sd.). A Figura 4 ilustra algumas características da aplicação Duolingo.

Figura - Características do Duolingo



Fonte: DUOLINGO, p.1.

O ambiente proposto deseja unir algumas características dos sistemas citados, como a possibilidade de encontrar conteúdos de auxílio (como no espaço do CCAA) e acesso a testes de escrita e escuta como no Duolingo. Acredita-se que ao aliar essas funcionalidades já utilizadas em outros sistemas à novas funcionalidades, possa-se criar um ambiente mais adequado para apoiar o ensino de língua inglesa na ILC.

## Desenvolvimento e tecnologias de sistemas Web

Para entender-se o que é desenvolvimento pode-se tomar o contexto geral da palavra, que tem por significado ser uma “Série de etapas, acontecimentos, ações, etc. que levam ao surgimento de algo, ou à manifestação em todos os seus aspectos[...]” (FERREIRA, 2001, p. 240). Tomando isso para o meio computacional diz-se então que o desenvolvimento de um sistema se dá pela passagem por todas as etapas da sua concepção até o momento em que o sistema apresente tudo aquilo que lhe foi previsto (PRESSMAN, 2011).

Hirama (2011, p. 3) lista os tipos de atividades presentes no processo de desenvolvimento de sistemas, que são as “[...] atividades técnicas de Engenharia de Sistemas, análise, projeto, codificação e testes”. Essas atividades devem estar presentes e são responsáveis por contemplar o processo de desenvolvimento.

Já para o quesito *Web*, os sistemas desenvolvidos são conhecidos como *WebApps* (PRESSMAN, 2011). Esses sistemas têm por característica ser acessados e utilizados por meio da rede. Pressman (2011) cita alguns atributos que estão presentes na grande maioria dos *WebApps* que são: o uso intensivo de redes, simultaneidade, carga não previsível, desempenho, disponibilidade, orientadas a dados, sensibilidade no conteúdo, evolução contínua, imediatismo, segurança e estética. Portanto, para se desenvolver soluções *web* deve-se levar em conta os atributos apresentados por Pressman (2011).

### Criptografia e controle de acessos

Criptografia pode ser entendido como o conjunto de métodos e técnicas capazes de modificar um texto legível transformando-o em um texto não legível. Isso é possível por meio de um algoritmo, que codifica a mensagem baseado em algum padrão específico. A recuperação das informações originais se dá pelo processo inverso ao da criptografia (MORENO; PEREIRA; CHIARAMONTE, 2005). A Figura 5demonstra esse processo de transformação de uma mensagem e o seu retorno a mensagem original.

Figura - Esquema geral para criptografia de um texto



Fonte: MORENO; PEREIRA; CHIARAMONTE, 2005, p.21.

Moreno, Pereira e Chiaramonte (2005) afirmam haver dois meios de se criptografar uma mensagem, por meio de códigos ou por meio de cifras.

Os códigos protegem as informações trocando partes destas por códigos predefinidos. Todas as pessoas autorizadas a ter acesso a uma determinada informação devem conhecer os códigos utilizados. As cifras são técnicas nas quais a informação é cifrada por meio da transposição e/ou substituição das letras da mensagem original. Assim, as pessoas autorizadas podem ter acesso às informações originais conhecendo o processo de cifragem (MORENO; PEREIRA; CHIARAMONTE, 2005, p. 21).

Existem algoritmos que implementam processos de criptografia, gerando um *hash* contendo uma quantidade predeterminada de caracteres a partir de um determinado conteúdo. Cada algoritmo vem aliado a uma chave, que é um valor alterável dentro do algoritmo e que pode ser modificado para se adequar as necessidades do utilizador. Isso implica que mesmo que se conheça o funcionamento do algoritmo de criptografia uma pessoa só seria capaz de reproduzir o mesmo resultado de outra se ambas usassem a mesma chave no processo de criptografia (HINZ, 2000).

O processo de criptografia é utilizado em auxílio a segurança de senhas de usuário. Os dados sensíveis advindos do usuário, como no caso da senha, somente são salvos na base de dados após serem submetidos a um processo de criptografia, utilizando-se de algum algoritmo em específico (PHP, 2018a).

Já o controle de acessos pode ser entendido como a maneira a qual se controla o que cada usuário tem acesso em um sistema, privando o usuário o que não lhe for autorizado, e permitindo somente o que deve se ter acesso de fato (ZAPATER; SUZUKI, 2005). Comumente se utiliza o modelo de controle de acessos baseado em papeis (*Role-Based Access Control*) (RBAC). Esse modelo utiliza a premissa que cada usuário cumpre um ou mais papeis, como por exemplo, um professor, ou um aluno. E a esses papeis são atribuídas permissões. Essas permissões indicarão as funcionalidades que o usuário pode ter acesso no sistema. Por exemplo, um determinado usuário apresenta o papel de professor, e como professor ele pode lançar notas e faltas aos alunos. O sistema deve ser capaz de identificar o papel do usuário e permitir somente o que aquele papel contiver como permissão (SANDHU, 1998).

O *Framework* Laravel, que é discutido na subseção 5.2.5.5, contém avançados algoritmos de criptografia (OTWELL, 2018). Assim as senhas dos usuários serão submetidas a um processo de segurança. Além disso o Laravel apoia o controle de acessos por meio de papeis como descrito anteriormente, para garantir que cada usuário só acesse o que lhe for permitido. Pensa-se que com a utilização de tais mecanismos de segurança possa se conseguir uma melhora na segurança dos dados sensíveis dos usuários e dos acessos no ambiente.

### Interação humano computador (IHC)

*Softwares* são necessários em praticamente todas as áreas de atuação presentes no mercado. Porém não há uma solução computacional que resolva todos os problemas de todas as áreas. Então para cada problema surge uma maneira de implementar uma solução que seja utilizada por um nicho de usuários (BARANAUSKAS; ROCHA, 2003). Portanto a maneira que os sistemas interagem com os usuários tendem a ser diferentes em cada solução.

Para buscar compreender e melhorar a maneira que usuários se interagem com sistemas surgiu o campo de estudo denominado Interação Humano-Computador (IHC) (BARANAUSKAS; ROCHA, 2003). Baranauskas e Rocha (2003) afirmam que a IHC contempla todo o processo de interação do usuário com o sistema e não somente o processo de *design* das interfaces. Então entende-se como Interação Humano-Computador a: “[...] disciplina preocupada com o design, avaliação e implementação de sistemas computacionais interativos para uso humano e com o estudo dos principais fenômenos ao redor deles.” (BARANAUSKAS; ROCHA, 2003, p. 14), e a Figura 6 apresenta algumas informações sobre IHC.

Figura - Interação humano-computador adaptada da descrição do comitê SIGCHI 1992



Fonte: BARANAUSKAS; ROCHA, 2003, p.15.

É importante que se conheça e se aprimore os processos de interação do usuário para que as pessoas consigam efetuar suas funções corretamente em um sistema, uma vez que se compreende o que cada interface significa e o que deve ser feito em cada uma delas.

### Engenharia de Software

O processo de desenvolvimento de *software* pode ser feito de maneiras diversas. Portanto buscou-se padronizar o processo de desenvolvimento para que se minimizasse os custos e problemas de produção e com uma melhor qualidade final do *software*, surgindo assim a Engenharia de *Software* (PRESSMAN, 2011). Hirama (2011, p. 7) apresenta o contexto histórico em que a definição do conceito de Engenharia de *Software* surgiu:

O conceito “Engenharia de Software” foi cunhado em 1969 por Fritz Bauer em uma conferência patrocinada por um Comitê de Ciência da Organização do Tratado do Atlântico Norte (Otan), no momento em que a chamada crise do software precisava de uma solução para a demanda crescente por software dentro de custo e prazo adequados.

Para entender-se o conceito do que é engenharia de *software* toma-se os significados de engenharia e *software*. A definição de engenharia se dá pela: “Aplicação de conceitos científicos e empíricos, e certas habilitações especificas, à criação de estruturas, dispositivos e processos para converter recursos naturais em formas adequadas ao atendimento das necessidades humanas.” (FERREIRA, 2001, p. 289). Já a palavra *software* tem as seguintes definições segundo Ferreira (2001, p. 682):

1. Em um sistema computacional, o conjunto de componentes informacionais, que não faz parte do equipamento físico e inclui os programas e os dados a eles associados.
2. Qualquer programa ou conjunto de programas de computador.

A partir dos significados pode-se deduzir que engenharia de *software* se trata então da aplicação de conceitos e métodos para apoio ao processo de desenvolvimento de um sistema computacional. Sommerville(2011, p. 5) reforça essa afirmação dizendo que: “Engenharia de software é uma disciplina de engenharia cujo foco está em todos os aspectos da produção de software, desde os estágios iniciais da especificação do sistema até sua manutenção, quando o sistema já está sendo usado”. O *Institute of Eletrical and Eletronics Engineers* (IEEE)(1990, p. 67) porém define como Engenharia de software:

1. A aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, a operação e manutenção do software;
2. O estudo de abordagens assim como descrito na etapa 1.

Pressman (2011) caracteriza a engenharia de *software* como sendo em camadas, conforme descrito na Figura 7, sendo as camadas: Ferramentas, métodos, processo e foco na qualidade.

Figura - Camadas da Engenharia de Software



Fonte: PRESSMAN, 2011, p.39.

Quanto ao foco na qualidade, Pressman (2011), afirma que todas as disciplinas ligadas a engenharias devem ter foco na qualidade. Para isso conta-se com modelos de qualidade que apoiam a qualidade tanto no processo quanto no produto final.

Já em relação ao processo, Pressman (2011) ressalta que existem alguns tipos. Um deles é o de engenharia de *software*, que é a “[...] liga que mantém as camadas de tecnologia coesas e possibilita o desenvolvimento de software de forma racional e dentro do prazo.” (PRESSMAN, 2011, p. 39). O outro é o processo de *software* que:

[...] constitui a base para o controle do gerenciamento de projetos de software e estabelece o contexto no qual são aplicados métodos técnicos, são produzidos produtos derivados (modelos, documentos, dados, relatórios, formulários etc), são estabelecidos marcos, a qualidade é garantida e mudanças são geridas de forma apropriada. (PRESSMAN, 2011, p. 40)

Os métodos por sua vez apresentam informações de caráter técnico para auxiliar no processo de desenvolvimento do *software* (PRESSMAN, 2011). “Os métodos envolvem uma ampla gama de tarefas, que incluem: comunicação, análise de requisitos, modelagem de projeto, construção de programa, testes e suporte.” (PRESSMAN, 2011, p. 40).

Já as ferramentas compreendem como o apoio automatizado aos métodos e ao processo (PRESSMAN, 2011). São por exemplos: *softwares* para modelagem, *Integrated Development Environments* (IDES), etc.

#### Modelagem de processos com o *Business Process Model and Notation* (BPMN)

Um processo pode ser entendido como uma sequência ordenada de atividades, praticadas por um ou mais atores, com propósito de se alcançar um objetivo em específico. Então, todo processo oferece algo como resultado da completitude de todas as atividades (CAMPOS, 2014). Um exemplo de processo pode ser visto na Figura 8 que representa um processo cujo objetivo seja a conclusão de um determinado curso (CAMPOS, 2014).

Figura - Exemplo de processo



Fonte: CAMPOS, 2014, p.18

No mundo dos negócios as empresas são compostas por uma série de processos, sejam eles internos ou externos. Então se faz necessário a compreensão quanto ao funcionamento desses processos (CAMPOS, 2014). Porém um processo de negócio poderia ser representado de várias maneiras, e uma delas, por exemplo, poderia ser a maneira descrita na Figura 8.

Buscando a padronização da notação de modelagem dos processos de negócios criou-se o *Business Process Model and Notation* (BPMN) que se trata de um padrão de notação para modelagem de processos. O BPMN auxilia nessa padronização de notação pois fornece elementos gráficos para a modelagem de fluxo de um determinado processo. O BPMN indica por meio desses elementos gráficos quais são as atividades que compõem um processo e quem são os atores e executores de cada atividade (SILVER, 2017).

Campos (2014, p. 51) define a notação BPMN como “[...] rica em elementos de representação”. Dentre esses elementos de representação, os considerados básicos a toda modelagem são os conectores, *gateways*, eventos, *data objects*, *pools* e *lanes* (CAMPOS, 2014)*.* Cada um deles será descrito nos parágrafos subsequentes, identificando os seus significados e representações visuais.

Os conectores representam a conexão de atividades dentro de um processo. Representam a ligação e por conseguinte sequência de atividades do processo. A representação visual dos conectores se dá por linhas indicativas (setas) conforme visto na Figura 9, que relata um processo de compra de um determinado item (CAMPOS, 2014).

Figura – Exemplo de conectores em um processo de compra



Fonte: CAMPOS, 2014, p. 53.

Já o *gateway* se trata de uma validação necessária que irá determinar o curso do fluxo do processo, alterando ou não o seu fluxo natural. A notação utilizada para representar o *gateway* é a figura de um losango, conforme visto na Figura 10. A Figura 10 relata um processo de compra, porém na atividade de receber, ocorre uma validação de constatação de conformidade com o que foi solicitado, havendo conformidade, paga-se, não havendo retorna-se a atividade anterior (CAMPOS, 2014).

Figura - Exemplo de *gateway* em um processo de compra



Fonte: CAMPOS, 2014, p. 54.

Os eventos para o BPMN servem de indicação de ações externas ao processo como por exemplo indicar o início e o término de um processo (CAMPOS, 2014). A representação gráfica dos eventos é feita por meio de círculos. Os círculos com a borda espessa indicam o fim de um processo. Enquanto os círculos que tem bordas duplas significam eventos intermediários de um processo. Já os círculos com borda menos espessa indicam o início do processo. Um exemplo de utilização dos eventos descritos é o visto na Figura 11.

Figura - Exemplo de utilização de eventos em um processo de compra



Fonte: CAMPOS, 2014, p. 56.

*Data objects* por sua vez, representam dados e informações que são usadas e criadas ao longo de um processo (CAMPOS, 2014). Esses dados e informações servem para utilização na atividade subsequente a atividade que criou a informação. É representado graficamente por uma folha de papel com a ponta superior dobrada, ligada por linhas seccionadas em formato de seta, que ligam a atividade que gerou a informação a atividade que irá consumir a informação (CAMPOS, 2014). Exemplos de *data objects* são vistos na Figura 12, como o pedido de compra que é gerado ao comprar um produto.

Figura - Exemplo da utilização de *data objects* em um processo de compra



Fonte: CAMPOS, 2014, p. 58.

As *Pools*, ou piscinas em português, representam um processo como um todo, englobando todas as atividades de um processo (CAMPOS, 2014). São representados graficamente por um retângulo contendo o nome do processo.

*Lanes*, ou raias em português, representam os atores participantes de um processo, sendo esses atores pessoas, departamentos, setores, cargos, etc. (CAMPOS, 2014). As raias são representadas por um retângulo contendo o nome do ator. As raias e as demais atividades do processo devem estar englobadas dentro de uma piscina, conforme visto na Figura 13.

Figura - Exemplo de utilização de piscinas e raias em um processo de compra



Fonte: CAMPOS, 2014, p. 59.

Acredita-se que a utilização de modelagem de processos neste trabalho servirá de documentação ao desenvolvedor e ao cliente, ajudando-os a compreender um determinado processo.

#### Metodologia Ágil

Metodologia ágil se trata de um movimento que buscou mudar a maneira como os projetos de *software* eram executados. Apresentou-se como premissa o aumento na velocidade de desenvolvimento de sistemas sem que houvesse uma eventual perda de qualidade. E buscou-se diferenciar das metodologias tradicionais como o modelo em cascata. Uma das diferenças se dá pela filosofia iterativa (ou espiral) e não mais linear como no modelo em cascata (HIRAMA, 2011). Esse modo iterativo prega o desenvolvimento de *software* como algo constante. A cada iteração o cliente recebe uma parte do sistema podendo opinar sobre melhorias ou eventuais erros de modelagem (TELES, 2014). A cada iteração então recebe-se do cliente o seu *feedback* acerca do que foi implementado, além do cliente ter a possibilidade de utilizar o sistema sem que esse esteja completamente terminado. Ao final de todas as iterações tem-se o sistema completo e operante, uma vez que o sistema foi validado ao fim de cada iteração (HIRAMA, 2011). A Figura 15 demonstra um exemplo de representação do modelo iterativo ou espiral.

Figura - Modelo em espiral



Fonte: HIRAMA, 2011, p39.

O XP, que será descrito suas características e peculiaridades na subseção 5.2.3.5, e utilizado no decorrer deste trabalho, é um exemplo de uma metodologia ágil.

#### *Extreme Programming* (XP)

O *Extreme Programming* (XP), se trata de uma metodologia de desenvolvimento ágil, que busca aliar agilidade no desenvolvimento, com qualidade no produto final. O XP pode então, ser descrito como “[...] um processo de desenvolvimento que busca garantir que o cliente receba o máximo de valor de cada dia de trabalho da equipe de desenvolvimento” (TELES, 2014, p. 24). Mas ao contrário do modelo em cascata, o XP é uma metodologia que apoia as mudanças. O XP entende que o cliente pode não ter certeza sobre o que quer logo de início, mas que vai descobrindo assim que tem contato com um protótipo do *software*. Assim, ao ter contato direto com o *software*, o cliente é capaz de verificar o que deve ser melhorado ou mudado. Além disso os desenvolvedores podem identificar ainda o que o cliente considera de mais importante. Portanto é de suma importância ao XP o *feedback* do cliente acerca do que foi implementado (TELES, 2014).

O *feedback* é um dos quatro valores fundamentais do XP, junto com a comunicação, a simplicidade e a coragem. Portanto o *feedback* diz respeito a capacidade de se receber informação de volta do cliente. E através dessas informações ser capaz de entender as especificações e requisitos necessários para o desenvolvimento, conhecendo assim as necessidades e anseios do cliente, em vista que o cliente estará em contato direto com o *software* (TELES, 2014).

A comunicação diz respeito a equipe constantemente estar em contato direto com o cliente, pois estabelecendo contato frente a frente, a equipe pode retirar potenciais dúvidas ou conhecer certos comportamentos requeridos no *software* (TELES, 2014). A diferença de se ter um contato frente a frente, para uma documentação escrita, é que a documentação não exprime emoções ou sentimentos, já na comunicação direta a linguagem corporal toma forma, e o sentimento é expressado de maneira mais eficaz como afirma Teles (2014).

A simplicidade consiste em desenvolver código o mais simples possível, somente resolvendo o necessário para o cliente (TELES, 2014). Teles (2014) explica que isso se dá, pois, um código mais simples tende a ser mais fácil de ser alterado em caso de necessidade de alteração.

A coragem diz respeito a equipe ter coragem de enfrentar algo novo. Para desenvolver usando o XP alguns padrões devem ser deixados de lado (TELES, 2014). Por exemplo, o XP prega que o desenvolvimento deve ser feito em par, ou seja, dois desenvolvedores numa mesma máquina, desenvolvendo o mesmo código. Porém pode-se acreditar que isso seja algo custoso ou dispendioso, mas Teles (2014) afirma que se deve ter coragem para esquecer-se de premissas tradicionais para acreditar e assumir o uso dessa metodologia.

Quanto a programação em par citada anteriormente, ela é uma das práticas do XP, junto com o jogo do planejamento, o cliente presente, o *stand up meeting*, desenvolvimento guiado a testes, *refactoring*, código coletivo, código padronizado, *design* simples, metáfora, ritmo sustentável, integração contínua e *releases* curtos. A programação em par é característica que levanta mais dúvida quanto ao uso do XP. Essa dúvida ocorre pelo fato de que se pensa que com a utilização de dois programadores fazendo um único trabalho, está se desperdiçando um programador que poderia estar desenvolvendo em outra máquina. Porém ao se utilizar desse desenvolvimento em par espera-se que a qualidade final do código seja melhor, pois enquanto um programa o outro verifica e valida o código. Além disso pode-se aliar duas soluções distintas para um problema, cada um com uma experiência e maneira de solucionar algo, com o intuito de encontrar uma solução considerada melhor (TELES, 2014).

Além da programação em par, destaca-se o desenvolvimento orientado a testes, que se trata de se testar primeiro antes de codificar de fato (HIRAMA, 2011). Pensa-se que ao se produzir o teste primeiro o desenvolvedor consiga compreender melhor o que será desenvolvido (TELES, 2014). Teles (2014) defende que o ato de testar deve ser uma prática constante no processo de desenvolvimento, pois assim assume-se que os erros encontrados através dos testes serão corrigidos imediatamente ou que pelos testes possa também haver a validação do que foi requisitado.

Outro fator que se destaca no XP são as estórias de usuários que são pequenos trechos escritos em cartões ao qual o cliente descreve como quer que seja uma determinada funcionalidade do sistema (TELES, 2014). Toda estória de usuário se transforma em um requisito e, por conseguinte uma funcionalidade da aplicação desenvolvida. Um exemplo de estória de usuário é o presente na Figura 16.

Figura - Exemplo de uma estória de usuário



Fonte: TELES, 2014, p.78.

Como o XP se trata de uma metodologia iterativa, em certos períodos de tempo, tem-se entregas de porções do *software* para que que o cliente possa utilizar o sistema sem que o mesmo esteja totalmente pronto. Essas entregas são conhecidas como *releases* (TELES, 2014). Segundo Teles (2014) os *releases* no XP duram cerca de 2 meses, e por isso são considerados *releases* curtos, já que o intervalo entre as entregas é pequeno. Após o término de um *release* faz-se uma nova entrevista com o cliente e coleta as estórias que serão desenvolvidas no novo *release* (TELES, 2014). Cada *release* é composto por iterações, uma iteração pode ser entendida como “[...] um pequeno espaço de tempo dedicado para a implementação de um conjunto de estórias” (TELES, 2014, p. 87). Segundo Teles (2014) o tempo de uma iteração pode variar cerca de uma a três semanas, mas que geralmente o tempo de duas semanas é o mais utilizado em empresas com o XP.

Pretende-se neste trabalho fazer utilização da metodologia de desenvolvimento denominada XP. Hirama (2011) afirma que o XP pode não ser recomendado a grandes projetos de *software*. Porém pretende-se fazer utilização desta metodologia a ponto de se chegar a uma avaliação pessoal acerca da experiência de uso do XP e assim verificar como a metodologia conseguiu apoiar o desenvolvimento do ambiente. Deve-se ressaltar que não será possível a aplicação da programação em par, que é uma das práticas do XP, pois a execução do projeto será feita por uma pessoa.

### Tecnologias para desenvolvimento WEB

O desenvolvimento de soluções *web* envolve uma série de tecnologias. Cada uma com uma determinada função e utilidade. As tecnologias Web podem ser divididas em *Front-End* e *Back-End*. Tecnologias *Front-End* são aquelas que estão em contato direto com o usuário, como por exemplo, o *Hypertext Markup Language* (HTML) (ROBBINS, 2013). Já as tecnologias *Back-End* são as que estão em contato direto com o servidor e não mantém vínculo direto com o usuário final, um exemplo é a linguagem de *scripting* PHP: *Hypertext Preprocessor* (PHP) (PHP, 2018b). Todas essas tecnologias trabalham em conjunto para oferecer uma melhor experiência de uso para o usuário (ROBBINS, 2013). Algumas dessas tecnologias são abordadas nas seções seguintes deste trabalho.

#### *Hyper Text Markup Language* (HTML)

*Hyper Text Markup Language* (HTML) se trata de uma linguagem de marcação capaz de transformar documentos de texto em páginas *web*. Seu objetivo é prover uma descrição semântica do conteúdo e formar uma estrutura para o documento. A delimitação de conteúdo no HTML é feita por meio de *tags* que vão indicar onde um determinado conteúdo começa e termina (ROBBINS, 2013). A Figura 19 consiste na estrutura de *tags* básica de um documento HTML.

A *tag* ‘html’ indica onde se inicia e onde se termina o documento HTML. A *tag* ‘*head*’ representa o cabeçalho do documento, dentro dela há uma outra *tag* chamada ‘*title*’ que indica o título da página. Já a ‘*body’* representa o corpo do documento HTML (ROBBINS, 2013).

Figura - Estrutura básica do HTML



Fonte: PRÓPRIA, utilizando o SublimeText 4.

O HTML está na sua versão 5, e essa versão trouxe uma série de mudanças acerca das *tags* a fim de deixar o documento HTML mais semântico. Com essas mudanças deixou-se a leitura de documentos e páginas mais fácil por caracterizar melhor os elementos da página (ROBBINS, 2013).

Por acreditar que o HTML seja uma maneira simples e concisa de marcação, além do Laravel reconhecer HTML, optou-se então pelo seu uso no desenvolvimento deste trabalho.

#### *Cascading Style Sheets* (CSS)

*Cascading Style Sheets,* com acrônimo (CSS), é uma linguagem criada com o objetivo de cuidar do *layout* e do estilo de uma página (CAELUM, 2018). Além de estilização dos elementos, com o CSS também é possível acrescentar efeitos nos elementos como desaparecer da tela por exemplo (MCFARLAND, 2013). E apesar de o CSS colaborar com o HTML, ele não se trata de HTML. Enquanto o HTML provê uma organização estrutural de um documento, o CSS trabalha em colaboração com o *browser* com a finalidade de fazer com que uma página HTML contenha estilo próprio (MCFARLAND, 2013). Um exemplo da sintaxe do CSS é o descrito na Figura 20.

Figura - Sintaxe CSS



Fonte: CAELUM, 2018, p.21.

Quanto a inclusão do CSS em elementos HTML, ela pode ser feita de três maneiras. A primeira delas é inserindo o código CSS diretamente na *tag* HTML do elemento, utilizando-se do atributo ‘*style*’ da *tag* em questão conforme descrito na Figura 21.

Figura - CSS inserido diretamente na tag HTML



Fonte: CAELUM, 2018, p.21.

Outra maneira de se inserir o CSS é por meio de uma *tag* HTML denominada ‘*style*’, onde o seletor do elemento deve ser referenciado como na Figura 22. O seletor pode ser entendido como uma referência aos elementos de uma página. As principais maneiras de se referenciar elementos são pelo atributo id (identificado com o símbolo #), pelo atributo *class* (identificado com o ponto final), ou pela *tag* de um elemento. Na sintaxe CSS dentro da *tag* ‘*style’* deve-se indicar o seletor, e no caso da Figura 22 o seletor é uma *tag* de parágrafo (p), e o que vier dentro das chaves são seus estilos (CAELUM, 2018).

Figura - CSS inserido através da *tag style*



Fonte: CAELUM, 2018, p.22.

A terceira maneira de se estilizar os elementos HTML se dá por meio de um arquivo separado como visto na Figura 23, de maneira que o código CSS fique separado da codificação feita em HTML (CAELUM, 2018).

Figura - CSS contido no arquivo estilos.css



Fonte: CAELUM, 2018, p.23.

A sintaxe CSS segue o modelo descrito pela Figura 22. A única diferença é que para utilizar a estilização contida no arquivo CSS no documento HTML deve-se utilizar no código HTML uma *tag* ‘*link*’. A *tag* ‘*link’* é responsável por carregar e possibilitar o uso dos estilos a partir de um arquivo externo. Para indicar a localização do arquivo CSS, deve-se utilizar um atributo da *tag* ‘*link’* chamado ‘href’, e nesse atributo indicar o caminho até o arquivo de estilos, conforme demonstrado na Figura 24. O estilo criado na Figura 23 é utilizado na *tag* <p> do documento HTML da Figura 24 (CAELUM, 2018).

Figura - CSS inserido através de um arquivo externo



Fonte: CAELUM, 2018, p.22.

Com o uso do CSS no ambiente proposto espera-se contemplar os usuários com páginas mais bonitas e agradáveis para que a experiência de uso seja a melhor possível.

#### JavaScript (JS)

O JavaScript (JS) se trata de uma linguagem de *scripting*,que tem por característica ser uma linguagem interpretada, ou seja, não depende de compilação para sua execução (CAELUM, 2018). Além disso se trata de uma linguagem de tipagem dinâmica, ou seja, as variáveis não estão diretamente ligadas a somente um tipo, como por exemplo int, *double*, mas variam de acordo com o dado inserido na variável (CROCKFORD, 2008).

O JavaScript é comumente utilizado em *web* browsers, e os *web* browsers que têm o papel de interpretar o código em JavaScript e gerar uma saída. O intuito da linguagem é prover dinamicidade aos conteúdos, uma vez que o HTML e o CSS provêm conteúdo estático a um documento *web* (CAELUM, 2018). Para se utilizar o JavaScript em páginas HTML é necessário que se faça uso de uma *tag* específica chamada *script* conforme descrito na Figura 25 e na Figura 26. É possível ainda escrever os *scripts* de duas maneiras. A primeira delas é inserindo o *script* diretamente na página HTML (Figura 25).

Figura - Exemplo de uso do JavaScript diretamente no HTML



Fonte: PRÓPRIA, utilizando o SublimeText 4.

A outra maneira de se utilizar o JavaScript em páginas HTML é escrevendo o *script* em um arquivo separado e anexá-lo ao documento HTML como visto na Figura 26. Acredita-se que assim possa separar melhor HTML de JavaScript e deixando mais coesa a leitura de um documento HTML (CAELUM, 2018). Porém em algumas ocasiões é necessário inserir o *script* diretamente no documento HTML.

Figura - Exemplo de uso do JavaScript por meio de um arquivo externo



Fonte: PRÓPRIA, utilizando o SublimeText 4.

Portanto, visando alcançar dinamicidade nos documentos HTML, optou-se pela utilização do JavaScript. Pensa-se que assim a experiência final do usuário com o sistema possa ser mais dinâmica.

#### TypeScript

O TypeScript, ou TS, é um “superconjunto” ou “*superset*” em inglês da linguagem JavaScript. Isso significa que o TS serve para agregar funcionalidades e melhorias a linguagem JavaScript como o exemplo a criação de classes, objetos e atributos das classes como herança e polimorfismo. Porém o que se destaca mais no TypeScript em diferença ao JavaScript é a tipagem dos dados, onde em um Script TypeScript os dados devem ser tipados para facilitar a leitura e compreensão do código, além de evitar que uma variável receba um tipo de dado não esperado (ABREU, 2017).

O trecho de código da figura 27 se trata de um exemplo de Script TS. Nota-se a tipagem da variável modelo definindo seu tipo como *string*. Além disso como dito anteriormente há também o acréscimo de funcionalidades, na figura demonstra-se o exemplo das classes, algo que não existe no JS comum.

Figura - Classe em TypeScript



Fonte: PRÓPRIA, utilizando o Visual Studio Code.

Para que o TS seja reconhecido nos navegadores é necessário um processo de compilação que transforma o código TypeScript para JavaScript que é entendido pelos navegadores. Além disso esse processo de compilação, também conhecido como “transpilação”, converte o EcmaScript 6 que é a versão mais atual do JavaScript em uma versão ao qual a maioria dos navegadores interpreta que é a EcmaScript 5 (ABREU, 2017).

#### Angular

O angular é uma tecnologia criada pela Google afim de prover dinamicidade no processo de utilização de sistemas web. Então podes se dizer que o angular se trata de um framework *frontend* que gera como resultado aplicações chamadas de *Single Page Application* (SPA), ou em português, aplicações de página única. Ou seja, não há o carregamento da página web a cada interação com o usuário. Entendendo melhor o angular age em um contexto assíncrono, diferente do PHP por exemplo que é um contexto síncrono. No angular o usuário realiza uma ação que gera uma requisição no servidor, porém a página não é recarregada, o utilizador pode fazer outras interações com a página ou até mesmo outras requisições ao servidor, sem que a primeira requisição tenha terminado. Isto gera um ambiente dinâmico de acesso e utilização. Uma aplicação angular é toda particionada em componentes, módulos, serviços, modelos, dentre outros. Um componente se trata de um pedação da aplicação que sofre interação com o usuário, como por exemplo um botão, ou um campo de pesquisa. Esse componente é formado por um *template* que é a parte HTML do componente. Por um Script TS que é responsável pela lógica que será empregada naquele componente. E por último, um arquivo de estilos CSS que é responsável pela estilização do componente. Os módulos são responsáveis por agrupar componentes, serviços etc. (GUEDES 2017). Tendo em vista que este seja um a ambiente que tem por característica a interação contínua com o usuário, optou-se pela utilização do angular para auxiliar nesse processo e deixar a utilização mais fluída e dinâmica.

#### *Hypertext PreProcessor* (PHP)

O *PHP Hypertext* *PreProcessor* (PHP) é uma linguagem de *script* usada principalmente no contexto *web* para fazer a manipulação de páginas HTML (PHP, 2018b). Para seu uso é necessário a abertura e fechamento de uma *tag* PHP conforme descrito na Figura 27.

Figura - Exemplo de código PHP em página HTML



Fonte: PHP, 2018a, p.1.

O PHP tem por característica ser uma linguagem interpretada *Server-Side*, ou seja,executada no lado do servidor (SKLAR, 2016). Assim, uma aplicação utilizando o PHP pode ser disponibilizada para uma série de pessoas através de uma *Uniform Resource Locator* (URL) (SKLAR, 2016). Sklar (2016) explica que se detendo da URL, o usuário faz uma solicitação ao servidor que interpreta o documento PHP e retorna o resultado HTML, que é entendível pelo *Browser*. Os servidores *web* em que geralmente o PHP é utilizado são os servidores Apache ou Ngix (LOCKHART, 2015).

A linguagem apresenta uma série de recursos e aditivos, como por exemplo manipulação de imagens, recursos de abstração de banco de dados, além de pacotes dos mais variados fins criados pela comunidade e que auxiliam na popularização da linguagem e seu uso para diversos fins (SKLAR, 2016).

A utilização do PHP neste trabalho se dará por meio do *Framework* Laravel que será discutido na subseção 5.2.5.5.

#### *Framework* Laravel

O Laravel é um *Framework* baseado na estratégia de desenvolvimento *Model*, *View* e *Controller* (MVC) e feito sobre linguagem PHP, que tem como objetivo facilitar e melhorar o processo de desenvolvimento (STAUFFER, 2017). Stauffer (2017, p. 22) apresenta o porquê da utilização de *frameworks* para desenvolvimento dizendo:

Frameworks como o Laravel – e o Symfony, Sliex, Lumen e Slim – pré-empacotam um conjunto de componentes de terceiros com características personalizadas do framework, como arquivos de configuração, provedores de serviço, estruturas de diretório prescritas e carregadores de aplicativos. Logo, o benefício de usar um framework em geral é o de que alguém tomou decisões por você não só sobre componentes individuas, mas também sobre como eles devem ser integrados.

Portanto pode-se afirmar que um *framework* como o Laravel simplifica alguns processos comuns a sistemas (como por exemplo o processo de autenticação) por meio de pacotes prontos. Isso faz com que o desenvolvedor se concentre em processos específicos da lógica de negócio ao qual seu sistema pertence, agilizando o desenvolvimento (STAUFFER, 2017).

O Laravel demonstra como valores o aumento da velocidade de desenvolvimento e a satisfação do desenvolvedor. Para isso dispõe de uma série de ferramentas que auxiliam em diversos processos no desenvolvimento, como por exemplo o processo de autenticação, o envio de *e-mails*, processos de banco de dados (criação de tabelas, transações e consultas), entre outros (STAUFFER, 2017).

Tendo em vista a quantidade de recursos disponíveis pelo Laravel, uma comunidade que auxilia em momentos de dúvida, além do conhecimento prévio, decidiu-se pela sua utilização neste trabalho de conclusão de curso.

### Sistema de Gerenciamento de Banco de Dados (MySQL)

Bancos de dados estão presentes em todos os lugares em que se tem uso de computadores, desde a gestão de dados de uma multinacional, até numa farmácia de esquina. Mas o que de fato são bancos de dados? Elmasri e Navathe (2011, p. 3) definem como banco de dados “[...] uma coleção de dados relacionados” que possuem as seguintes características (ELMASRI; NAVATHE, 2011):

* Apresentar algum aspecto do mundo real;
* Ser um conjunto de dados que apresentem algum sentindo inerente;
* Projetado, construído e populado com dados com um fim em específico.

Conhecendo sobre o que se trata um banco de dados toma-se o conceito de Sistemas de Gerenciamento de Banco de Dados (SGBD) como “um conjunto de dados associados a um conjunto de programas para acesso a esses dados” (SILBERCHATZ; KORTH; SUDARSHAN, 1999). Portanto, para se gerir os dados que estão presentes em um banco de dados é necessário a atuação de um SGBD. O SGBD que será responsável então pela “[...] definição das estruturas de armazenamento das informações e a definição dos mecanismos para manipulação dessas informações.” (SILBERCHATZ; KORTH; SUDARSHAN, 1999, p. 1).

Dentre os SGBS, um que se destaca é o MySQL (CARVALHO, 2015). O MySQL é um SGBD gratuito e com base de licença *open source*, com ferramentas de desenvolvimento robustas e de qualidade (CARVALHO, 2015). A Figura 28 representa um mapa mental que contém as principais características do MySQL.

Figura - Características do MySQL



Fonte: CARVALHO, 2015, p.3.

Com o MySQL é possível criar, editar e excluir dados através de sentenças na linguagem *Structured Query Language* (SQL). Além disso, como visto na Figura 28, apresenta características como possibilidade de uso de mais de um usuário, estabilidade, etc. Diversas linguagens de programação oferecem integração com o MySQL, e uma delas é o PHP e, por conseguinte o Laravel (STAUFFER, 2017). Um dos motivos da escolha do MySQL como SGBD deste trabalho foi o fato de ser gratuito e a fácil integração com o Laravel, sem se esquecer das características que auxiliam em qualidade e que foram descritas na Figura 28.

# Cronograma

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ATIVIDADES | | 2º SEMESTRE 2018 | | | | | | 1º SEMESTRE 2019 | | |
| JAN/  2019 | FEV/  2019 | MAR/  2019 | ABR/  2019 | MAIO  /2019 | JUN/  2019 | JUL/  2019 | AGO/  2019 | SET/  2019 |
| Release de Cadastros | Coleta de Dados para o primeiro *release* (Pesquisa, Observação e Entrevista) |  |  |  |  |  |  |  |  |  |
| Análise dos Requisitos do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Modelagem dos processos e do banco de dados do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Desenvolvimento (codificação) das funcionalidades do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Testes e correções das funcionalidades do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Reuniões com o cliente para retiradas de dúvidas acerca das funcionalidades e confecção dos testes de aceitação de estórias do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Release do banco de questões | Coleta de Dados para o segundo *release* (Pesquisa, Observação e Entrevista) e entrega do primeiro *release* |  |  |  |  |  |  |  |  |  |
| Análise dos Requisitos do segundo *release* |  |  |  |  |  |  |  |  |  |
| Modelagem dos processos do segundo *release* |  |  |  |  |  |  |  |  |  |
| Desenvolvimento (codificação) das funcionalidades do segundo *release* |  |  |  |  |  |  |  |  |  |
| Testes e correções das funcionalidades do segundo *release* |  |  |  |  |  |  |  |  |  |
| Reuniões com o cliente para retiradas de dúvidas acerca das funcionalidades e confecção dos testes de aceitação de estórias do segundo *release* |  |  |  |  |  |  |  |  |  |

# BIBLIOGRAFIA

ABREU, L. **TypeScript:** O JavasCript Moderno para Criação de Aplicações. Lisboa: FCA – Editora de Informática. 2017.

ALVES, J. R. M. *et al.* **Educação a Distância:** o estado da arte. São Paulo: Pearson Education do Brasil, v. 1, 2009.

BABBEL. **Preços**. 2018. Disponível em: <https://home.babbel.com/prices>. Acesso em: 23 ago. 2018.

BACICH, L. *et al*. **Ensino Híbrido:** Personalização e tecnologia na educação. Porto Alegre: Penso, 2015.

BARANAUSKAS, M. C. C.; ROCHA, H. V. D. **Design e Avaliação de Interfaces Humano-Computador**. Campinas: UNIVERSIDADE ESTADUAL DE CAMPINAS, 2003. Disponível em: <https://www.nied.unicamp.br/biblioteca/design-e-avaliacao-de-interfaces-humano-computador/ > Acesso em: 22 set. 2018.

BENTO, E. J. **Desenvolvimento Web com PHP e MySQL**. São Paulo: Casa do Código, 2013.

CAELUM. **Desenvolvimento Web com HTML, CSS e JavaScript**. São Paulo: Caelum ensino e inovação, 2018. Disponível em: <https://www.caelum.com.br/download/caelum-html-css-javascript.pdf>. Acesso em: 07 out. 2018.

CAMPOS, A. L. N. **Modelagem de Processos com BPMN**. 2. ed. Rio de Janeiro: Brasport, 2014.

CARVALHO, V. **MySQL:** Comece com o principal banco de dados open source do mercado. São Paulo: Casa do Código, 2015.

CCAA. **Espaço CCAA Aluno**. sd. Disponível em: <https://www.ccaa.com.br/espacoccaa/conteudos/>. Acesso em: 23 ago. 2018.

CROCKFORD, D. **JavaScript:** The Good Parts. Sebastopol: O'Reilly, 2008.

DIAS, D. D. S.; SILVA, M. F. D. **Como escrever uma monografia:** Manual de elaboração com exemplos e exercícios. Rio de Janeiro: Atlas, 2010.

DUOLINGO. **Aprenda idiomas de graça. Para sempre**, sd. Disponível em: <https://pt.duolingo.com/>. Acesso em: 23 ago. 2018.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 6. ed. São Paulo: Pearson Education, 2011.

FERREIRA, A. B. D. H. **Mini Aurélio Século XXI:** O minidicionário da língua portuguesa. 5. ed. Rio de Janeiro: Nova Fronteira S.A, 2001.

GIL, A. C. **Como Elaborar Projeos de Pesquisa**. 4. ed. São Paulo: Atlas, 2002.

GOOGLE. **Sobre o Google Sala de aula**, 2018. Disponível em: <https://support.google.com/edu/classroom/answer/6020279?hl=pt-BR#>. Acesso em: 23 ago. 2018.

GUEDES, T. **Crie aplicações com angular**: o novo Framework do Google. São Paulo: Casa do Código, 2017.

HINZ, M. A. M. **Um estudo descritivo de novos algoritmos de criptografia.** 2000. 58f. Monografia (Bacharel em Informática) - Universidade Federal de Pelotas, Pelotas, 2000. Disponível em: < http://www.jabour.com.br/ufjf/apa/Mono-MarcoAntonio.pdf >. Acesso em: 5 out. 2018.

HIRAMA, K. **Engenharia de Software:** Qualidade e Produtividade com Tecnologia. Rio de Janeiro: Elsevier, 2011.

INSTITUTE OF ELETRICAL AND ELETRONICS ENGINEERS. **IEEE Std 610.12-1990:** IEEE Standard Glossary of Software Engineering Terminology. New York: [s.n.], 1990. 84 p. Disponível em: < http://www.mit.jyu.fi/ope/kurssit/TIES462/Materiaalit/IEEE\_SoftwareEngGlossary.pdf>. Acesso em: 9 set. 2018.

LOCKHART, J. **PHP Moderno**. São Paulo: Novatec, 2015.

MARCONI, M. D. A.; LAKATOS, E. M. **Fundamentos de metodologia científica**. 5. ed. São Paulo: Atlas, 2003.

MCFARLAND, D. S. **CSS3:** the missing manual. 3. ed. Sebastopol: O'Reilly, 2013.

MELO NETO, J. A. D. *et al.* **Educação a distância:** o estado da arte. São Paulo: Pearson Education do Brasil, v. 2, 2012.

MORENO, E. D.; PEREIRA, F. D.; CHIARAMONTE, R. B. **Criptografia em Hardware e Software**. São Paulo: Novatec, 2005.

OTWELL, T. **Encryption.** 2018. Disponível em: <https://laravel.com/docs/5.7/encryption>. Acesso em: 05 out. 2018.

PHP. **Modelo de Armazenamento Criptografado.** 2018a. Disponível em: <https://secure.php.net/manual/pt\_BR/security.database.storage.php>. Acesso em: 05 out. 2018.

PHP. **O que é o PHP?**, 2018b. Disponível em: <https://secure.php.net/manual/pt\_BR/intro-whatis.php>. Acesso em: 30 set. 2018.

PRESSMAN, R. S. **Engenharia de Software:** Uma abordagem Profissional. 7. ed. Porto Alegre: Bookman, 2011.

ROBBINS, J. N. **HTML5:** Pocket Reference. 5. ed. Sebastopol: O'Reilly, 2013.

SANDHU, R. S. Role-based Access Control. In: **Advances in Computers.** Fairfax: Academic Press, v. 46, 1998. p. 237-286. Disponível em: <http://www.profsandhu.com/articles/advcom/adv\_comp\_rbac.pdf>. Acesso em: 5 out. 2018.

SEVERINO, A. J. **Metodologia de trabalho científico**. 22. ed. São Paulo: Cortez, 2002.

SILBERCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de Banco de Dados**. 3. ed. São Paulo: Pearson Education, 1999.

SILVER, B. **BPMN Method and Style:** with Bpmn Implementer's Guide. 2. ed. Altadena: Cody-Cassidy Press, 2017.

SKLAR, D. **Aprendendo PHP:** Introdução amigável à linguagem mais popular da WEB. São Paulo: Novatec, 2016.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

STAUFFER, M. **Desenvolvendo com Laravel:** Um Framework para construção de aplicativos PHP modernos. São Paulo: Novatec, 2017.

TELES, V. M. **Extreme Programming:** Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade. 2. ed. São Paulo: Novatec, 2014.

WAZLAWICK, R. S. **Metodologia de Pesquisa para ciência da Computação**. Rio de Janeiro: Elsevier, 2009.

WIZARD. **Experiências Wizard**, 2017a. Disponível em: <http://www.wizard.com.br/experiencias-wizard/>. Acesso em: 23 ago. 2018.

WIZARD. **Sobre a Wizard**, 2017b. Disponível em: <http://www.wizard.com.br/sobre-wizard/>. Acesso em: 23 ago. 2018.

ZAPATER, M.; SUZUKI, R. **Segurança da Informação:** Um diferencial determinante na competitividade das corporações. Promon Business & Tecnology Review. Rio de Janeiro, p. 28. 2005. Disponível em: <http://www.teleco.com.br/promon/pbtr/Seguranca\_4WEB.pdf>. Acesso em: 12 out. 2018.

# Apendice A - carta de pedido de permissão para uso de informações da escola International language center





