Universidade Federal de Goiás
Regional Catalão
Unidade Acadêmica Especial de Biotecnologia
Curso de Bacharelado em Ciências da Computação

Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC.

Erikson Dutra de Miranda Silva

Catalão – GO
2019

Erikson Dutra de Miranda Silva

Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC.

Monografia apresentada ao Curso de Bacharelado em Ciências da Computação da Universidade Federal de Goiás – Regional Catalão, como parte dos requisitos para obtenção do grau de Bacharel em Ciências da Computação. *VERSÃO REVISADA*

Orientadora: Prof.ª Dr.ª Luanna Lopes Lobato
Coorientador: Prof. Dr. Thiago Jabur Bittar
Coorientador: Prof. Me. Wanderlei Malaquias Pereira Junior

Catalão – GO
2019

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Silva, Erikson Dutra de Miranda
Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC. [manuscrito] / Erikson Dutra de Miranda Silva. – 2019.
91 p.: il.

Orientadora: Prof.ª Dr.ª Luanna Lopes Lobato
Coorientador: Wanderlei Malaquias Pereira Junior
Monografia (Graduação) – Universidade Federal de Goiás, Unidade Acadêmica Especial de Biotecnologia, Ciências da Computação, 2019.
Bibliografia.

1. Engenharia de Software. 2. Desenvolvimento Web. 3. Engenharia Civil. 4. Gestão de Orçamento. 5. Laravel. 6. MVC. I. Lobato, Luanna Lopes, orient. II. Bittar, Thiago Jabur, coorient. Junior, Wanderlei Malaquias Pereira, coorient. III. Título.

CDU 004

Erikson Dutra de Miranda Silva

Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC.

Monografia apresentada ao curso de Bacharelado em Ciências da Computação da Universidade Federal de Goiás – Regional Catalão.

Trabalho aprovado em 12 de Dezembro de 2019.

Luanna Lopes Lobato
Orientadora

Thiago Jabur Bittar
Coorientador

Veríssimo Guimarães Júnior
UFG - Regional Catalão

Cláudio Lemos de Souza
UFG - Regional Catalão

Catalão – GO
2019

Este trabalho é dedicado a todos os pequenos e grandes desenvolvedores,
que por meio de linguagens sonham em tornar o mundo melhor.
Em especial, aos alunos e professores do curso de Ciências da Computação e Engenharia Civil
da Universidade Federal de Goiás – Regional Catalão.

AGRADECIMENTOS

Os agradecimentos principais são direcionados à Universidade Federal de Goiás por ter proporcionado experiencias únicas e um grande crescimento pessoal e profissional.
Ao professor Wanderlei Malaquias Pereira Junior pelo seu virtuoso empenho, paciência e confiança para a realização deste trabalho.
A professora Luanna Lopes Lobato por ter levantado a moral, compartilhado seu conhecimento e ajudado em todas as disciplinas e trabalhos ministrados.
Agradeço a todos os professores desta instituição por compartilharem suas experiencias e sabedorias, que foram fundamentais para moldar o caminho que tenho seguido. Isto tem proporcionado uma visão critica e metódica para a solução dos problemas.
Aos meus pais, por me apoiarem nos estudos das ciências exatas, sempre me dando carinho e me ajudando nas adversidades.
A minha mãe Keila Maria Dutra de Miranda Silva pelo seu trabalho incansável que serviu de motivação e inspiração para seguir com os estudos.
Ao meu pai Uzelino José da Silva pela sua admiração e orgulho que me guiaram ate o final.
A minha esposa Anna Paula Neri de Sousa Dutra de Miranda que lutou ao meu lado, me levantando com carinho e dedicação em todos os momentos que pensei em desistir.
Aos meus amigos que sempre estiveram ao meu lado, participando da minha formação e dando apoio.

“Um livro é a prova de que os homens
são capazes de fazer magia.”
(Carl Sagan)

RESUMO

SILVA, E. D. M.. **Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC.**. 2019. 91 p. Monografia (Graduação) – Unidade Acadêmica Especial de Biotecnologia, Universidade Federal de Goiás – Regional Catalão, Catalão – GO.

Com a crescente evolução do mercado novas necessidades surgem para otimização e garantia de qualidade na execução de serviços. Para tanto, a busca por ferramentas de software tem crescido, as quais são utilizadas para otimizar a gestão do tempo, os custos das operações e os recursos disponíveis. Não obstante a isso, na área da construção civil tem-se visto a necessidade por sistemas de software cada vez mais robustos e eficientes, no tocante ao auxilio das tarefas e realização de funções que poderiam ser custosas quanto feitas de maneira manual. Visando atender essas necessidades se encontram disponíveis, na literatura e em repositórios, diversas ferramentas para gestão orçamentaria em construção civil. A maioria destes sistemas são de uso comercial e sua licença possui um alto custo, principalmente se tratando da necessidade de sua utilização nas Universidades. Deste modo, neste trabalho é apresentada uma solução gratuita, em forma de ferramenta de software, que tem por objetivo principal realizar orçamentos em construção civil. Esta foi desenvolvida utilizando, como tecnologia Web, o *Framework* Laravel MVC. Assim, possibilitou-se, por meio da ferramenta, que os alunos do curso de Engenharia Civil pudessem utiliza-la como auxílio para o estudo e desenvolvimento de orçamentos, de maneira otimizada e com garantia de qualidade sobre as regras seguidas. Ainda, por meio deste trabalho, espera-se que alunos da Engenharia de Software possam utilizar a pesquisa como base para proposta de novos trabalhos e melhorias deste, na condução dos trabalhos futuros. Como resultado, têm-se o sistema desenvolvido e um estudo foi aplicado para teste do mesmo, com usuários alvos.

Palavras-chave: Engenharia de Software, Desenvolvimento Web, Engenharia Civil, Gestão de Orçamento, Laravel, MVC.

ABSTRACT

SILVA, E. D. M.. **Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o *Framework* Laravel MVC.**. 2019. 91 p. Monografia (Graduação) – Unidade Acadêmica Especial de Biotecnologia, Universidade Federal de Goiás – Regional Catalão, Catalão – GO.

With the growing evolution of the market new needs arise for optimization and quality assurance in the execution of services. To this end, the search for software tools has grown, which are used to optimize time management, operating costs and available resources. Nonetheless, in the area of construction, there has been a need for increasingly robust and efficient software systems for assisting tasks and performing functions that could be costly when done manually. To meet these needs, various tools for budget management in civil construction are available in the literature and repositories. Most of these systems are for commercial use and their license has a high cost, especially when it comes to their use in universities. Thus, this work presents a free solution, in the form of software tool, whose main objective is to make budgets in construction. This was developed using, as web technology, the Laravel MVC Framework. Thus, it was possible, through the tool, that the students of the Civil Engineering course could use it as an aid for the study and development of budgets, in an optimized way and with quality assurance on the rules followed. Also, through this work, it is expected that Software Engineering students can use the research as a basis for proposing new work and improving it, in conducting future work. As a result, the system has been developed and a study has been applied to test it with target users.

Keywords: Software Engineering, Web Development, Civil Engineering, Budget Management, Laravel, MVC.

LISTA DE ILUSTRAÇÕES

LISTA DE CÓDIGOS-FONTE

LISTA DE TABELAS

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

Com a crescente evolução do mercado, a busca por ferramentas tecnológicas tem crescido para otimizar a gestão do tempo, os custos operacionais, os recursos e os processos. Desde a concepção do projeto até a etapa de sua execução é importante a coleta e gerenciamento correto das informações, afim de convertê-las em conhecimento e evolução de um negócio.

A utilização dos sistemas de software tem se mostrado cada dia mais essencial para a realização das atividades nos mais diferentes setores, uma vez que a informação cresce de forma acelerada, em um curto período de tempo, propiciando limitações em seu uso devido à necessidade de gerenciamento de maneira automatizada, (FILHO, 2016).

De acordo com RALL, CAMPGNA E OLIVEIRA JÚNIOR (2014) é cada vez mais precisa a necessidade pelo uso de ferramentas para otimização de processos e gerenciamento de informações na área de Engenharia Civil. MARTINS (2000) mostra que a utilização da computação em orçamentos é uma tarefa complexa, pois cada projeto é uma circunstância especial, com variáveis próprias e com imprevisibilidade. Além de haver uma dificuldade adicional a respeito do Banco de Dados referentes aos custos, pois eles raramente são realimentados pelo sistema, necessitando de permanentes atualizações.

O software a ser usado para esta finalidade precisa realizar a medição correta dos insumos necessários, da mão de obra e dos equipamentos utilizados para a realização de todos os serviços que serão empregados. Segundo MUTTI (2006), deve-se estimar a duração da obra e cotar os preços de cada item, sendo necessária a quantificação correta dos custos diretos e indiretos da obra. Os custos diretos estão diretamente ligados ao projeto e produtos de uma construção. Os custos indiretos são aqueles que não são perfeitamente quantificáveis, vinculados geralmente à administração do canteiro de obras e às despesas da administração da própria empresa.

Em se tratando de softwares, para tal atividades, são diversos os sistemas como o Compor 90 [1], Arquimedes [2] e Sienge [3] que são softwares privados e com um custo elevado. Tal situação dificulta a utilização desse tipo de sistema em instituições de ensino de engenharia.

Além disto, o ensino desta etapa nos cursos de Engenharia Civil pode não ter o máximo de proveito uma vez que não haja a apresentação de alguma ferramenta que gerencie os orçamentos de maneira eficiente. Além disso, alguns desses softwares usam tecnologias que estão entrando em desuso, sendo que a maioria tem por característica funcionar em modo desktop, utilizando de uma estação fixa e Sistema Operacional específico, não apresentando assim aspectos relacionados a portabilidade.

Visando atender esta necessidade neste trabalho têm-se como meta o desenvolvimento e o refinamento de uma ferramenta para realizar orçamentos em construção civil utilizando tecnologias voltadas para o desenvolvimento Web e arquitetura de projeto, como o Model.®view.®controller (MVC). Para LEFF e RAYFIELD (2001) o padrão MVC apresenta um conceito de particionamento flexível para aplicações da Web, que permite aos desenvolvedores aplicar soluções de maneira simplificada, focando no reúso de código.

Como tecnologia principal para o desenvolvimento desta ferramenta utilizou-se o *Framework* Laravel MVC. Para SUAVÉ (2002) um *Framework* é uma abstração que une códigos comuns entre vários projetos de software provendo uma funcionalidade genérica. O *Framework* Laravel MVC será abordado com detalhes no Capítulo 2.

A principal razão do uso de tecnologias Web foi devido a portabilidade e praticidade que essas tecnologias proveem. Deste modo, é possível acessar o software em qualquer dispositivo e localidade, bastando para isto ter acesso a Internet por meio de um *browser*.

1.1 Motivações e Objetivo

O objetivo desse trabalho é a elaboração de uma ferramenta orçamentária em ambiente Web que permita o ensino e prática de orçamentação para estudantes de Engenharia Civil. A ferramenta desenvolvida é de uso gratuito, sendo que a tecnologia pode ser executada em vários tipos de aparelhos como smartphones, computadores, notebooks e tablets devido a portabilidade empregada na elaboração do sistema, a qual possibilita ao futuro profissional uma maior flexibilidade no acesso e estudo da ferramenta.

Para tanto, surgiu o grupo Coretec na Universidade Federal de Goiás (UFG). O grupo é formado por estudantes e professores de graduação e pós-graduação das áreas de Ciência da Computação e Engenharia Civil. Esta parceria tem por objetivo criar uma ferramenta com vários módulos para o ensino de Engenharia Civil. Este trabalho se encontra no módulo de orçamentação e teve como base os requisitos levantados pela primeira versão do sistema denominado EngPack Budget. Este sistema foi desenvolvido por SILVA (2018) e tem por objetivo a realização e controle de orçamentos para construção civil.

[1] Compor 90. https://noventa.com.br/.
[2] Arquimedes. https://multiplus.com/software/arquimedes/
[3] Sienge. https://www.sienge.com.br/o-sienge/

1.1.1 Objetivos Gerais

Como objetivos gerais desta pesquisa cita-se o desenvolvimento da ferramenta denominada Engpack Budget 2.0, software desenvolvido para o ensino da realização e controle de orçamentos para construção civil.

Outro objetivo foi a realização de estudos e emprego das metologias de desenvolvimento ágeis, *Extreme Programming* (XP) e Scrum. A metologia XP enfatiza o desenvolvimento rápido do projeto e visa garantir a satisfação do cliente, além de favorecer o cumprimento das estimativas, sendo *feedbacks* constante, o uso de abordagem incremental e encorajamento na comunicação entre as pessoas envolvidas já o Scrum apresenta uma abordagem empírica que aplica algumas ideias da teoria de controle de processos industriais para o desenvolvimento de softwares, reintroduzindo os conceitos de flexibilidade, adaptabilidade e produtividade.

Como este projeto se deu pela parceria entre os cursos de Ciência da Computação e Engenharia Civil, foi preciso estudar e conhecer aspectos relacionados a Engenharia de Software, bem como o domínio ao qual o projeto aplicado para controle de orçamentação na construção civil. Assim, na Ciência da Computação técnicas de Engenharia de Softwares foram estudadas e aplicadas no desenvolvimento e refinamento do software. Em contrapartida, foi necessário o envolvimento de alunos e professores da Engenharia Civil, uma vez que os *stakeholders* envolvidos nesse projeto, da área de computação, não detiam conhecimento sobre o domínio da pesquisa. Com o uso da Engenharia de Software foi possível observar as soluções existentes e identificar qual seria mais adequada para o desenvolvimento desta ferramenta. Para tanto, foram necessárias diversas contribuições de alunos e professores da Engenharia Civil para o entendimento de todo o processo necessário para a realização de um orçamento de construção civil.

1.1.2 Objetivos Específicos

O objeto de estudo deste trabalho é a primeira versão do sistema EngPack Budget, em que o sistema foi refatorado afim de melhorar a qualidade do software e inserir funcionalidades antes não implementadas, as quais estão descritas nas seções sobre a ferramenta. Deste modo, a estrutura interna do software foi modificada de modo que seu comportamento externo continuasse preservado. TRAVASSOS, GUROV E AMARAL (2002) diz que isto é uma abordagem orientada à melhoria evolutiva, em que assume-se a existência de algum modelo do processo ou produto de software e modifica este modelo com propósito de melhorar os objetos do estudo. Assim, os objetivos específicos deste projeto estão enumerados e descritos abaixo:

1. Análise e teste da primeira versão do EngPack Budget;
2. Estudo e uso de novas tecnologias para o desenvolvimento Web;
3. Emprego de normalização no Bancos de Dados;

4. Estudo dos softwares existentes no mercado da Engenharia Civil;
5. Adaptação do sistema para o uso dos dados orçamentários do sistema estadual da AGETOP e base de dados personalizada;
6. Implementação de novos relatórios;
7. Desenvolvimento das calculadores de BDI e Encargos;
8. Aplicação de testes unitários, de desenvolvimento e de usuário;
9. Aplicação de estudo de caso para verificação do sistema desenvolvido;
10. Elaborações de relatórios e artigos científicos e escrita da monografia.

1.1.3 Justificativas

A ferramenta apresentada por SILVA (2018) não atendia aos novos requisitos descritos no tópico 1.1.2 e dependia de emuladores para ser executada. Um emulador é um software que reproduz as funções de um determinado ambiente, a fim de permitir a execução de outros softwares sobre ele. Assim, seu funcionamento se restringia a uma estação fixa, não sendo possível ser acessado por vários dispositivos.

O sistema apresentava lentidão a medida que os dados aumentavam e não possuía documentação das funções que foram desenvolvidas ao qual contribuí negativamente com a manutenção das funcionalidades existentes e, em alguns casos, inviabiliza a criação de novos recursos. Além disto, o sistema não possuía estrutura para ser executado em um smartphone ou tablet, visto que, o mesmo não era responsivo. Deste modo, esta pesquisa visa desenvolver uma nova versão da ferramenta utilizando os recursos descritos nos tópicos anteriores para tratar estes problemas

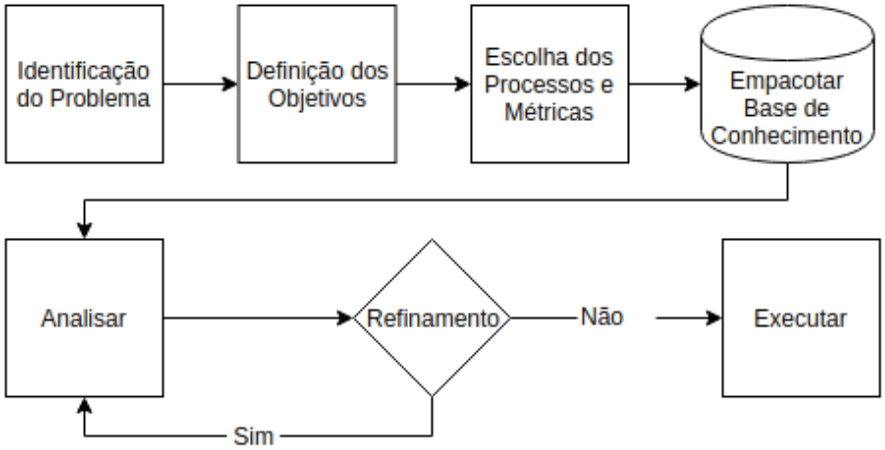
1.1.4 Metodologia de Pesquisa

Este projeto está inserido na Ciência da Computação, mais precisamente na área de Engenharia de Software, no que tange ao desenvolvimento de sistemas. Portanto, para a condução deste, no que se refere à metodologia de pesquisa, foi utilizada uma metodologia apresentado por TRAVASSOS, GUROV E AMARAL (2002).

Para TRAVASSOS, GUROV E AMARAL (2002) um experimento deve ser tratado como um processo da formulação ou verificação de uma teoria. A fim de que o processo ofereça os resultados válidos, ele deve ser propriamente organizado e controlado ou, pelo menos, acompanhado. Um bom exemplo da metodologia da experimentação avançada é o Paradigma da Melhoria da Qualidade, que tem por essência a melhoria contínua do desenvolvimento de um software.

Nesta metologia o processo do desenvolvimento oferece, além do próprio software, o *feedback* do projeto que representa as informações coletadas. As informações servem como base para a análise, a avaliação das práticas atuais, a determinação dos problemas, à proposição da melhoria decorrente e, por fim, toda a informação relevante está empacotada para utilização futura. Essas informações foram utilizadas para o desenvolvimento de artigos e o refinamento do sistema Engepack Budget. Na Figura 1 são ilustrados os passos seguidos no desenvolvimento do sistema.

Figura 1 – Passos seguidos utilizando o Paradigma da Melhoria do Software



Fonte: Elaborada pelo autor.

Como apresentado na Figura 1, foram identificados os problemas que a primeira versão apresentava. Isto possibilitou a definição dos objetivos da segunda versão, com o uso de métricas analisou-se os resultados a fim de validar se os objetivos foram alcançados. Todos os erros e acertos foram utilizados para o refinamento dos processos e métricas definidas inicialmente.

O Paradigma da Melhoria da Qualidade usa a abordagem *Goal/Question/Metric* (GQM), definida por SOLIGEN (1999). Nesta abordagem é definido o processo da melhoria do software baseado na compreensão da meta que se deseja alcançar, nas perguntas levantadas para o controle do desenvolvimento e nas métricas estabelecidas para atingir a melhoria do software. Para TRAVASSOS, GUROV E AMARAL (2002), esses objetivos são focados no custo para o desenvolvimento, o risco, o tempo que será gasto e a qualidade do produto final. Assim, para a aplicação desta abordagem é necessário a realização de quatro fases, sendo elas o planejamento, a definição, a coleta de dados e a interpretação.

O planejamento é a fase em que há a definição das técnicas que serão utilizadas, além de se definir as estimativas de prazos e as tecnologias que resultaram o plano do projeto. Na fase da definição são estabelecidos os objetivos que se deseja alcançar, as hipóteses e, por fim, as métricas necessárias para o desenvolvimento do software.

Seguindo para a fase de coleta de dados é necessário realizar a interpretação dos dados que se deseja melhorar no sistema, levantando todos os pontos que funcionavam e os que não operavam. Por fim, na fase de interpretação é feita o processamento das informações reunidas com base nas métricas, nas questões e nos objetivos definidos.

TRAVASSOS, GUROV E AMARAL (2002) diz que para fazer uso do Paradigma da Melhoria da Qualidade é necessário a análise do seu custo-benefício, ou seja, se o benefício irá superar o custo da produção. Em alguns casos não é possível realizar o aproveitamento de códigos, estruturas do Banco de Dados e *layouts*. Assim, tudo o que se aproveitou foi o funcionamento externo, isto é, a tarefa que o software realiza. Nesses casos, o custo do desenvolvimento é alto, sendo necessário o planejamento de um novo sistema com outras tecnologias que atenderam as novas necessidades.

A primeira versão do sistema não apresentou uso de padrões de projetos ou documentações e comentários nas funções implementadas. Isto dificultou o desenvolvimento de novos requisitos e correções dos erros. Além disto, não houve emprego de normalização no Banco de Dados. Deste modo, optou-se pelo uso de uma nova tecnologia para o desenvolvimento do sistema que faz uso do MVC. Esta escolha possibilitou a implementação de recursos com um menor número de linhas e, consequentemente, o emprego de mais recursos em menos tempo. Por tanto, o ganho superou o custo do desenvolvimento.

Para BASILI, CALDIERA E ROMBACH (1994), esta abordagem combina em si a maioria das abordagens atuais para avaliação e medição dos recursos de um produto. Isso a torna adaptável a diferentes ambientes, como confirmado pelo fato da mesma ter sido aplicada em várias organizações, por exemplo, Nasa, Hwelett Packard, Motorola e Coopers Lybrand. Outra etapa da metologia utilizada neste trabalho é apresentada por SPÍNOLA (2008) e adaptada por LOBATO (2012), sendo dividida em duas fases, Concepção e Avaliação.

Na fase de Concepção é realizado o estudo afim de obter uma maior compreensão sobre a área da pesquisa. Deste modo, foram identificados os benefícios e dificuldades, possibilitando a definição do escopo de estudo e o levantamento de dados para construção do conhecimento e proposta de trabalho.

A importância desta fase se dá pela realização de um levantamento prévio da revisão da literatura para que se possa selecionar alguns trabalhos relacionados ao tema da pesquisa. Isto foi necessário para verificar a validade da mesma, afim de trazer resultados e contribuições para à área científica. Assim, nesta fase foi identificado que são poucos os estudos publicados voltados para o desenvolvimento de um software Web gratuito para auxiliar o ensino da realização de orçamentos para construção civil nos cursos de Engenharia Civil.

Como mencionado, além do uso do Paradigma da Melhoria da Qualidade este software faz uso das metologias ágeis SCRUM e XP. Na etapa de Concepção foram realizados estudos do funcionamento destas técnicas, uma vez que os requisitos do sistema proposto neste trabalho são

grandes e complexos. Além disto, o emprego destas técnicas prioriza a comunicação afim de manter o melhor relacionamento possível entre clientes e desenvolvedores, preferindo conversas pessoais a outros meios de comunicação. Para SOARES (2004) estes métodos visam a criação de códigos simples que não devem possuir funções desnecessárias. Com isso, foi possível implementar o software com o menor número possível de classes e métodos em um curto período de tempo.

Ainda na fase da Concepção foi realizado um estudo sobre as atuais tecnologias do desenvolvimento Web afim de selecionar as que mais se adequariam para a realização desta pesquisa. Este projeto faz parte do grupo de pesquisa Coretec, cujo o objetivo é a criação de ferramentas para auxiliar o ensino no curso de Engenharia Civil. Deste modo, houve a necessidade de padronizar as tecnologias utilizadas para a programação destas ferramentas. Esta padronização se tornou necessária pois o objetivo é que vários alunos de Ciência da Computação, Marketing e Engenharia Civil continuem a participar dando continuidade a esta e outras pesquisas que já foram citadas. Após a realização das reuniões necessárias, com o grupo de pesquisa na fase da Concepção foram definidas as linguagens em que essas ferramentas seriam desenvolvidas. As linguagens são dividas em linguagens de programação, linguagem de marcação de texto e linguagem de estilização.

Visando a aplicação das técnicas de desenvolvimento ágil, também foram definidos a utilização de alguns *Frameworks*, isto é, um conjunto de tecnologias e ferramentas de código-fonte aberto para o desenvolvimento de componentes de interfaces para sistemas Web. Assim, para linguagens de programação foi optado a utilização do *Framework* Laravel baseado na linguagem de programação PHP. Este *Framework* utiliza o padrão de desenvolvimento MVC, um padrão de projeto de software focado no reuso de código que é abordado no Capítulo 3. Outra linguagem de programação utilizada foi o JavaScript, responsável por manipular as entradas de dados para posteriormente serem processadas pelo Laravel.

Para obter uma melhor apresentação do sistema proposto foi realizado o estudo da "folha de estilo em cascata" (*Cascading Style Sheets – CSS*), a qual é uma linguagem de estilização cujo o objetivo é tratar estilos de cores, tamanho dos textos e posições dos objetos do sistema.

Para estruturar e executar corretamente as dimensões e *layouts* dos formulários contidos no sistema nos mais variados tamanhos de telas e monitores existentes foi realizado o estudo do framework Bootstrap. Este framework utiliza a linguagem de marcação HTML (*HyperText Markup Language – HTML*), a linguagem de estilização CSS e a linguagem de programação JavaScript. A junção destas tecnologias possibilita a criação de modelos de design para a tipografia, responsável por dar ordem estrutural e forma à comunicação escrita, melhorando a experiência do usuário em um site amigável e responsivo.

O sistema proposto necessitava armazenar dados para posteriormente serem manipulados. Desta forma, foi analisado um sistema gerenciador de Banco de Dados (SGBD) de código aberto denominado MySQL. Esta tecnologia tem uma grande portabilidade, funciona em mais de 20

plataformas, sendo elas as principais, Linux, Windows, Mac e Solaris e foi escrito na linguagem de programação C (MYSQL, 2010).

Ainda na fase da Concepção foi realizado o estudo de ferramentas para auxiliar o desenvolvimento com base nas tecnologias citadas. Foi selecionado um Ambiente de Desenvolvimento Integrado, do inglês *Integrated Development Environment* (IDE), PHPStorm desenvolvido pela empresa JetBrains para a manipulação e criação de códigos fontes. Esta ferramenta é de uso comercial porém possui uma licença gratuita para estudantes de graduação. O uso desta ferramenta agiliza o processo de desenvolvimento por reunir características e ferramentas de apoio no desenvolvimento de software.

Outra ferramenta utilizada foi o Mysql WorkBench. Esta ferramenta permite visualizar a estrutura do Banco de Dados, criar modelos e gráficos da mesma para auxiliar o desenvolvimento do Banco de Dados além de realizar consultas SQL. Deste modo, é possível criar estruturas, deletar as mesmas e inserir ou remover dados sem grandes complexidades.

Para compreender e identificar os benefícios e dificuldades da área do trabalho e alinhar as expectativas dos resultados propostos foram realizadas diversas reuniões com professores, alunos e profissionais de Engenharia Civil e Ciência da Computação. Nestas reuniões foram compartilhados conhecimentos e experiencias referente a área de orçamentos para construção civil e processos para auxiliar o desenvolvimento do sistema.

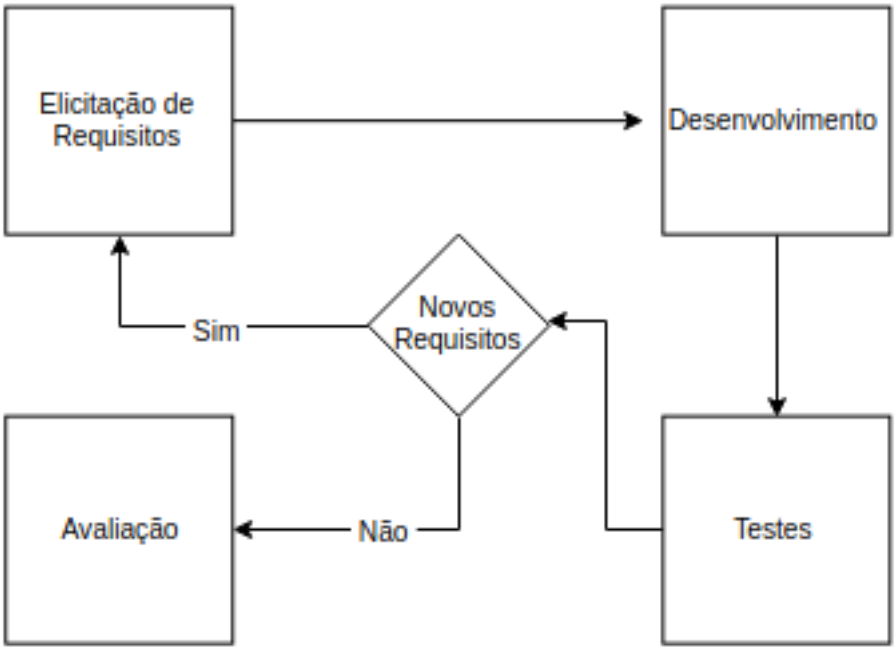
As reuniões foram realizadas em um período de 8 meses, com intervalos preestabelecidos de 15 dias. Nas reuniões decisões eram tomadas à medida que o sistema evoluía. Assim, foi possível avaliar o que estava sendo desenvolvido e sanar as duvídas que foram surgindo. Adicionalmente, foram identificados e desenvolvidos novos requisitos para o sistema. Algumas entrevistas foram realizadas afim de compreender as falhas que o sistema poderia apresentar na usabilidade de seus recursos, o que possibilitou o refinamento dos requisitos e incrementou a base de conhecimento. O resultado destas experiências facilitou a condução deste trabalho, uma vez que foi possível identificar novas necessidades, tomar decisões de modo que os objetivos fossem alcançados de forma válida com base nas regras definidas.

Assim, ao fim desta etapa afirmou-se a necessidade pelo desenvolvimento de um novo sistema, de modo a apresentar características de usabilidade, contemplar as funcionalidades requeridas, de maneira concisa ao requisitado, performance eficiente, bem como garantia de sua eficácia. Para tanto, foram identificadas as funcionalidades que deveriam ser adicionadas e desenvolvidas no sistema, sendo que toda necessidade foi tratada como um requisito. Deste modo, o projeto pode ser compreendido em detalhes, considerando a definição de como será o sistema de orçamento para construção civil.

Após analisar os estudos necessários para o desenvolvimento desta pesquisa na fase da Concepção é apresentado os passos realizados na fase da Avaliação. Nesta fase houve a implementação do sistema seguindo as estratégias e metodologias definidas. Além disto, é

mostrada uma forma de validar o software afim de coletar resultados e medir às expectativas depositadas nesta pesquisa. Para tando, esta fase da metodologia é dividida em Elicitação de requisitos, Desenvolvimento do Software de Orçamento de obra na Construção Civil, Testes e Avaliação do Sistema. O esquema representado na Figura 2 ilustra os passos da fase de Avaliação.

Figura 2 – Processo de Avaliação do Desenvolvimento



Fonte: Elaborada pelo autor.

Como observa-se na Figura 2 a fase de Elicitação de requisitos é dada pela efetividade dos requisitos que foram levantados na fase da Concepção. Isto é, os requisitos são tratados como funcionalidades que o sistema irá disponibilizar. Nesta etapa protótipos foram desenvolvidos e avaliados nas reuniões que eram realizadas. Ainda, nesta fase, novas necessidades foram levantadas e tratadas como novos requisitos a serem desenvolvidos.

A etapa de Desenvolvimento do Software foi realizada em conjunto com a metologia GQM, definida por SOLIGEN (1999). Para BASILI, CALDIERA E ROMBACH (1994) esta etapa é dada pela definição de objetivos em uma melhoria evolutiva da qualidade do software. Nesta etapa questões foram levantadas sobre a primeira versão do sistema Engpack Budget afim de compreender a meta que se desejava alcançar com o desenvolvimento deste trabalho. Posteriormente algumas métricas foram estabelecidas afim de validar e verificar a melhoria do sistema. As questões e métricas são abordas no Capítulo 4.

Um dos desafios desta etapa foi o Banco de Dados, uma vez que para realizar um orçamento em construção civil é necessário quantificar insumos, composições, mão de obra e equipamentos necessários, AVILA, LIBRELOTTO E LOPES (2006). Para tanto, foi necessário realizar diversos contatos com a diretoria da Agência Goiana de Transporte e Obras (AGETOP)

afim de conseguir uma Base de Dados contento estes insumos e composições fundamentais. Além destes insumos nesta etapa foi desenvolvido a possibilidades do usuário criar seu próprio Banco de Dados contendo os insumos e composições personalizadas. A segunda versão do sistema foi desenvolvida com base nestes dados visando atender as necessidades que a primeira versão não conseguiu alcançar.

Alguns protótipos foram desenvolvidos afim de alcançar o melhor resultado possível. Uma vez aprovados os mesmos eram integrados como funcionalidades do sistema. Para chegar neste ponto foi necessário a realização da etapa de Testes. Estes testes foram feitos com base no estudo de SOMMERVILLER (2011) que visava eliminar todos os possíveis problemas no software desenvolvido. Para alcançar esta meta foram realizados relatórios com os erros identificados, os quais eram armazenados na base de conhecimento, seguindo a metologia do Paradigma da Melhoria da Qualidade. Após uma revisão cuidadosa destes relatórios as correções e possíveis melhorias eram implementadas. É importante ressaltar que este etapa teve um apoio crucial de alunos, professores e profissionais dos cursos de Engenharia de Software e Ciência da Computação.

Apos realizadas as correções e melhorias necessárias seguiu-se para a etapa da Avaliação do Sistema. Neste ponto foram desenvolvidos e aplicados estudos sobre o produto final. Estes estudos são estudos de casos, com base nas recomendações apresentadas na literatura (MILLS, DUREPOS e WIEBE, 2010)(KITCHENHAM e PFLEEGER, 2008) (HOST e RUNESON, 2007), e posteriormente experimentos, seguindo as diretivas mostradas por Wohlin et al.(2000) afim de avaliar e coletar dados empíricos, assim como apresentado por CRUZES e DYBÅ (2010), da segunda versão do software EngPack Budget.

Nos estudos de casos foram executados os testes indicados na etapa de teste e em seguida foram realizadas as análises. Isto impactou em alterações com base nas melhorias que foram identificadas nesta etapa, o que resultou no desenvolvimento de novas funcionalidades. Estes estudos possibilitaram a evolução do sistema, tratando os problemas apresentandos na primeira versão e implementando novos recursos que a mesma não disponibilizou. Por fim, o resultado foi um sistema com maior eficácia conforme mostrado no Capítulo 6.

CAPÍTULO

2

REVISÃO SOBRE O DESENVOLVIMENTO WEB

Este Capítulo aborda os recursos que a Internet provê e as tecnologias utilizadas para dar suporte aos mesmos. Além disto, são mostradas técnicas e linguagens para o desenvolvimento Web. Exemplos simples destas linguagens são apresentados para exemplificar as estruturas que existem na segunda versão do Engpack Budget e, por fim, alguns trabalhos relacionados a esta pesquisa serão apresentados e discutidos.

2.1 Breve Histórico do Surgimento da Internet

É notório que com o surgimento da Internet, vários benefícios vieram juntos, seja pela disponibilização de serviços automatizados, de modo a viabilizar as tarefas que antes só eram feitas manuais, seja pela segurança e qualidade de transações ou, até mesmo, pelo benefício em se ter mais entretenimento, por meio das redes sociais, jogos, dentre outros softwares. Assim, são vários os recursos que a Internet provê, dentre os quais pode-se citar a facilidade na comunicação, por meio de um correio eletrônico, isto é, um *e-mail*; facilidade na transferência de arquivos utilizando um protocolo de transferência de arquivos (FTP - *File Transfer Protocol*); capacidade de encontrar grupos de discussão, denominado *News* e compartilhar informações por meio da Web (WWW - *World Wide Web*).

Concomitante a isto, pode-se afirmar que a Internet também trouxe alguns problemas, uma vez que, se não usada de maneira adequada, pode trazer problemas irreversíveis. No entanto, tais aspectos não são abordados nesta pesquisa por não ser o foco da mesma.

COSTA (2001) apresenta uma definição da Web, retratando-a como um sistema de informação cliente-servidor em hipertexto no formato HTML distribuído na Internet, sendo que, o programa cliente é o *browser* que envia informações para serem processadas por um servidor.

Um servidor é responsável por responder requisições do *browser* utilizando o protocolo HTTP, do inglês *Hypertext Transfer Protocol*. Deste modo, o servidor Web é um servidor de ficheiros. Contudo, na medida em que executa aplicações fornecendo resultados para as mesma, este sistema funciona como um servidor de aplicações. Existem vários servidores para Web, dentre eles podem ser citados o Apache [1] e Microsoft *Internet Information Services* [2].

O HTTP é um dos protocolos responsáveis por realizar a transferência de informação para obter recursos. WONG (2000) diz que o HTTP é o protocolo por trás da rede mundial de computadores, auxiliando no processamento de todas as solicitações de documentos ou mídias da Web. Estas solicitações são chamadas de requisições ou *requests*. Assim, um usuário ou cliente solicita algum conteúdo a aplicação servidor, que por sua vez envia o conteúdo por meio de respostas, ou *responses*. As *responses* são interpretadas e exibidas ao usuário por meio do *browser* através de páginas HTML, como ilustrado na Figura 3.

Figura 3 – *Hypertext Transfer Protocol*

Fonte: Elaborada pelo autor.

É possível observar uma divisão na Figura 3 entre o Cliente e o Servidor, sendo que as aplicações, de fato, possuem essa divisão denominadas em client-side e server-side. O *client-side* é responsável por realizar determinadas operações solicitadas pelo cliente, como, por exemplo, receber o login de usuário. O *server-side* é responsável por validar essas informações como, por exemplo, verificar se a senha está correta ou errada.

Existem tecnologias e técnicas específicas para tratar problemas de cada lado da aplicação. Os dois lados da aplicação ainda podem ser separados em *front-end* e *back-end*, tendo o *front-end* atuando no *client-side* e o *back-end* operando no *server-side*. As tecnologias utilizadas neste trabalho foram previamente citadas no tópico 1.2 e são descritas no tópico seguinte.

[1] Apache. https://www.apache.org/.
[2] Serviços de Informações da Internet (IIS). https://www.microsoft.com/pt-BR/download/details.aspx?id=48264.

2.2 Surgimento dos Padrões e Linguagens de Projetos

As tecnologias foram se consolidando contribuindo com um crescimento exponencial de dispositivos conectados a Internet. De acordo com o instituto *Internet Systems Consortium* (ISC), em janeiro de 1993 haviam cerca de 1,3 milhões de dispositivos conectados a Internet. O número cresceu de ano em ano, sendo 2,2 milhões em 1993; 4,9 milhões em 1996; 16,1 milhões em 1997 e assim por diante, ISSN (2008). O senso mais recente é dado pela Statista [3], portal alemão de estatísticas, mostrando que a previsão para 2025 será de 75 bilhões de dispositivos conectados a Internet, sendo que neste ano de 2019 é de 26,66 bilhões.

Essa quantidade de dispositivos conectados a Internet contribuiu para o aprimoramento de novas tecnologias, os equipamentos foram melhorando, a quantidade de *bytes* transferidos na rede foi aumentando, vários usuários acessando ao um mesmo conteúdo, serviços de *streaming* e jogos online. Deste modo, diversas tecnologias e técnicas de Engenharia de Software surgiram para desenvolver soluções para esses dispositivos.

Com o objetivo de tornar mais acessível e usual os sistemas para internet as tecnologias necessitavam de padrões. Assim, em 1994 foi fundado o W3C, por Berners-Lee, um dos idealizadores da Web. O objetivo deste órgão é padronizar as tecnologias envolvidas a Web, definindo protocolos relativos à mesma, como por exemplo o HTTP, as linguagens HTML, CSS e XML.

Para LALLI, BUENO E ZACHARIAS (2008) os padrões Web funcionam hoje como ferramentas de trabalho para os desenvolvedores e passaram a ser um pré-requisito na criação de *websites* compatíveis com a rede, reduzindo o custo de muitas empresas com a reestruturação de suas páginas a cada navegador ou nova versão do mesmo.

2.2.1 Linguagens para o Desenvolvimento Web

Diversas linguagens existem para o desenvolvimento Web, sendo assim, ao escolher qual tecnologia deve ser empregada em um determinado projeto é necessário definir o objetivo do mesmo. É importante ressaltar que deve-se utilizar uma linguagem que mais se adeque às necessidade do projeto, uma vez que cada uma apresenta característica peculiares, tendo vantagens e desvantagens. As linguagens de programação Web são usadas especificamente para desenvolver sites, portais e aplicações Web em geral.

2.2.2 PHP

Visando tornar os sistemas mais acessíveis, em 1994, surgiu a linguagem de programação PHP. Seu propósito era facilitar a criação de páginas dinâmicas sem a necessidade do desenvolvedor utilizar CGI (*Common Gateway Interface*), uma tecnologia que recebe parâmetros dos navegadores para serem processadas no servidor em que a mesma é executada.

[3] Statista. https://www.statista.com/.

Deste modo, no PHP, o código de programação é escrito junto com o código HTML, simplificando e, consequentemente, agilizando o desenvolvimento de um sistema. Assim, O *browser* executa o código que se encontra em uma determinada tag, as quais são etiquetas que estruturam os conteúdos exibidos pelo *browser*.

Para que o servidor processe o arquivo PHP contendo as instruções, o mesmo necessitará de um *Middleware*, uma tecnologia que fornece serviços para a comunicação, entrada e saída de dados. Isto contribui para agilizar os processos do sistema, decodificando, interpretando e determinando quais arquivos estão sendo requisitados. Assim, a informação é processada e o conteúdo é devolvido em um formato que o *client-side* consiga interpretar.

2.2.3 HTML

Em uma página exibida por um *browser* é possível observar que existem conteúdos de textos e mídias: nos textos existem títulos, parágrafos, tabelas, entre outros; nas mídias há fotos, imagens animadas e vídeos. Visando estruturar e organizar esses conteúdos surge em 1989 a linguagem HTML.

HTML é uma linguagem de marcação que consiste basicamente na formatação do texto e mídias através de tags. Essas marcações também permitem a inclusão de links para outras páginas. O *browser* interpreta o HTML contido nos sites e sistemas Web, exibindo o conteúdo com suas respectivas formatações.

Existem tags específicas para cada formatação, por exemplo, a tag <*title*> define o título da página que será exibida; a tag <*img*> define a exibição de uma determinada imagem. As tags começam sempre com o simbolo de menor e são finalizadas com o simbolo de maior, < ... >.

2.2.4 CSS

A linguagem HTML apenas organiza e indexa o conteúdo de uma página Web. É possível informar a cor, tamanho e posição do texto através da tag <*font*>. Contudo, para recursos mais avançados como animações e estilos de textos mais elaborados é necessário a utilização de outras tecnologias, como o CSS.

Em paralelo a grande quantidade de dispositivos conectados a Internet existe uma grande quantidade de sistemas para realizar funções similares. Então um sistema com *layout* e *designer* mais usual e escalável irá se sobressair a outro que não empregou essas técnicas.

O CSS surgiu em 1994, e tem a função de definir estilos que serviram para incluir aspectos de apresentação de uma página HTML. Com ele é possível configurar tamanho e cor de fontes, espaçamento entre linhas, margens e adaptar o *layout* em diferentes tamanhos de monitores e telas, COSTA (2001).

Com o CSS é possível definir um estilo para ser aplicado em varias páginas, isto agiliza

o desenvolvimento de um sistema Web. Tendo em vista que, com o HTML, se o desenvolvedor desejar colocar todo parágrafo com uma determinada coloração em todas as páginas, o mesmo deverá fazer uso da tag <*font*> página por página. Com o CSS bastaria definir um estilo em um arquivo CSS e aplicá-lo nas páginas desejadas. O exemplo abaixo ilustra esta situação:

Código-fonte 1 – Estilizar Parágrafos de uma Página

```
1 P {font-size: 12pt;
2     color: blue;
3 }
```

A sintaxe no exemplo acima diz que para toda tag em HTML <*p*> em que este arquivo CSS for aplicado terá a fonte de tamanho 12 e a cor azul. Este arquivo poderá ser utilizado em quantas páginas o desenvolvedor desejar. Caso a cor requerida no futuro seja vermelha, bastará que o programador mude o arquivo CSS a ter que mudar todas as páginas que utilizaram esta configuração.

Para COSTA (2001), a folha de estilo permite fazer uma declaração global com a configuração de *design* desejada para as paginas definidas pelo desenvolvedor. Para informar que a página A necessita do arquivo CSS *estilos.css* é necessário incorporá-lo a mesma através da tag <*link*>, conforme observado no exemplo abaixo:

Código-fonte 2 – Inserir Estilo nas Páginas

```
1 <"link href="estilos.css" rel="sytlesheet" type="text/css">
```

Deste modo, o browser saberá que para a página A a mesma utilizará a folha de estilo *estilos.css* e irá requisitar este arquivo para o servidor por meio do HTTP. Este arquivo ficará em memoria na estação cliente e será utilizado em todas as páginas que necessitam do mesmo.

2.2.5 JavaScript

As tecnologias HTML e CSS tem a função de formatar e estilizar uma página. Elas não conseguem, por exemplo, somar dois valores que são informados por um usuário. Isto ocorre porque elas não são linguagens de programação. Para executar esta operação simples o browser receberia os valores por meio do HTML, enviaria ao servidor e o servidor realizaria a operação de adição. Isto demanda um custo, determinadas ações poderiam ser realizadas no *client-side*. Para atender esta necessidade surge em 1996 o linguagem de programação JavaScript, COSTA (2001).

O JavaScript é uma tecnologia flexível de tipagem dinâmica, isto é, a definição de uma variável do tipo string ou interior será dinamicamente. Ela é uma linguagem procedural, funcional e orientada a objetivos. A linguagem é interpretada pelo *browser*, assim executada no *client-side*.

Como no CSS, o código JavaScript pode ser embutido no código HTML por meio da tag <*script*> conforme o exemplo a seguir, COSTA (2001).

Código-fonte 3 – "Olá Mundo"em JavaScript

```
1 <html>
2     <head>
3     </head>
4     <body>
5         <script type="text/javascript">
6             alert("Olá Mundo!")
7         </script>
8     </body>
9 </html>
```

No exemplo acima será exibido uma caixa de diálogo na página informando o texto "Olá Mundo!". Outro aspecto do JavaScript é o suporte a eventos de interação do usuário. Como por exemplo: exibir um botão ao clicar em uma região da página; mudar uma imagem ao mover o mouse sobre um item definido; mudar o texto ao selecionar determinada opção. Os principais eventos que o JavaScript suporta estão listados abaixo:

***onClick*:** executado quando o usuário clica em uma região da página, como por exemplo um botão;

***onChange*:** executado quando o usuário muda o valor de um elemento, como por exemplo um combo de seleção;

***onBlur*:** é executando quando o usuário retira o foco de um elemento delimitado na página;

***onFocus*:** é executado quando o usuário dá o foco em algum elemento;

***onLoad*:** é executado quando a página é carregada no browser;

***onSubmit*:** é executado quando o usuário submete alguma informação ao servidor por um formulário;

Estes eventos ocorreram como resultado de ações realizadas pelo usuário. Além disto, por se tratar de uma linguagem de programação é possível fazer uso de Estruturas de Controle, como as de Decisão e as de Repetição, tornando esta tecnologia muito importante e significativa no desenvolvimento de um sistema Web.

2.2.6 FrameWorks

Mesmo estas tecnologias tendo recursos que agilizem o desenvolvimento Web as mesmas tem suas limitações. Como por exemplo, sempre que houver a necessidade de criar interfaces e layouts mais elaborados será necessário codificar cada tecnologia de maneira separa para somente depois unir os arquivos de estilos *".css"*, os códigos e eventos do JavaScript *".js"* junto as páginas *".html"*. Seria de grande valia o desenvolvimento de recursos que unissem essas tecnologias para facilitar e agilizar ainda mais o desenvolvimento. Visando atender esta necessidade surgiu a ideia do *Framework*, que é a junção de tecnologias, códigos e recursos para executar funcionalidades específicas de maneira mais simplificada e ágil, Costa (2001). Existem Frameworks para os dois lados da aplicação, *client-side* e *server-side*.

A Internet surgiu com o objetivo de compartilhar informação. Essas informações eram, no início, em sua maior parte, estática. Contudo, com a crescente quantidade de dispositivos conectados a rede as informações se tornaram dinâmicas, onde o usuário pode consumir, criar e transmitir conteúdo em qualquer momento. Para DAS E SAIKIA (2016), o mercado necessitava que os desenvolvedores produzissem aplicativos com melhores recursos em menos tempo. Assim, novas tecnologias para o desenvolvimento Web foram criadas, contribuindo para a economia do tempo e aprimoração dos recursos da aplicação. Essas tecnologias provaram sua capacidade para lidar com as demandas do mercado.

O PHP é uma tecnologia bem conhecida e frequentemente associada ao desenvolvimento Web para o *server-side*, porém, ela pode ser aplicada para resolver outras necessidades, como por exemplo, um sistema *off-line*. De acordo com DAS E SAIKIA (2016), o PHP é uma linguagem bastante popular nos sistemas Web, com 82% de cobertura.

Um *Framework* bastante conhecido baseado nesta linguagem de programação é o Laravel. O Laravel tem o objetivo de desenvolver códigos curtos, com melhor desempenho e programação ágil. Para DAS E SAIKIA (2016), sua estrutura é a mais rápida, poderosa e afirma que o Laravel é um Framework PHP de alto desempenho para o desenvolvimento Web.

2.2.7 MVC

Estas vantagens em partes se justificam pelo fato do Laravel utilizar uma arquitetura denominada MVC. Este é um padrão que visa ser a ponte entre o modelo mental do usuário e o modelo computacional. Para LUCIANO E ALVES (2011), em uma empresa cujo o negócio principal não é Tecnologia da Informação (TI), o investimento em TI fica entre 70% e 80% na manutenção de um software, e somente o restante para a criação de novas soluções. Isso significa que a maior parte do investimento vai para a correção de problemas deste sistema, que em muitos casos é difícil, por não haver documentação das funções implementadas, e em outros casos impossíveis de serem resolvidas.

A arquitetura de projetos MVC é baseada em três camadas: as requisições da aplicação

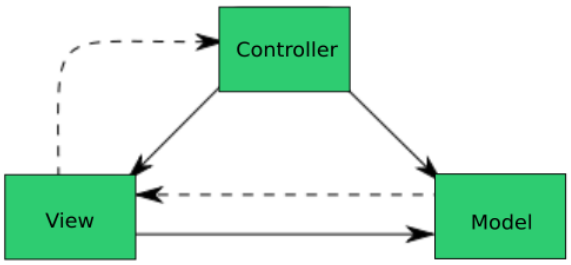
são direcionadas para a camada *Controller*, que por sua vez acessa a camada *Model* para processar esta requisição, por fim, a informação processada é exibida na camada *View*. LUCIANO E ALVES (2011) diz que, o padrão MVC separa as camadas de apresentação, de lógica de negócio e de gerenciamento do fluxo da aplicação, aumentando as capacidades de reutilização e manutenção do projeto.

Os requisitos da aplicação podem ser modificados com o passar do tempo. Assim, novos requisitos podem surgir e outros podem não ser mais necessários, ao passo que vários desenvolvedores podem criar e modificar o código do projeto. Um projeto separado nessas camadas é mais simples, possui menos códigos e consequentemente a manutenção possui um menor custo, uma vez que cada função e necessidade estarão em sua respectiva camada.

Esta organização em camadas é a chave para a independência entre os componentes e esta independência é que vai atingir os objetivos de eficiência, escalabilidade, reutilização e facilidade de manutenção, LUCIANO E ALVES (2011). Se no futuro houver a necessidade de alterar todo o layout de um software desenvolvido com arquitetura MVC será necessário apenas a manutenção na camada *View*, uma vez que a logica de negocio e o gerenciamento do fluxo dos dados estarão nas camadas *Controller* e *Model*.

Um sistema sem padrões de projetos, sem documentação das funções e do Banco de Dados teria um alto custo para realizar esta mesma mudança. Em alguns casos tais mudanças seriam inviáveis, havendo a necessidade de desenvolver um sistema com outras tecnologias. A Figura 4 ilustra o fluxo das camadas MVC:

Figura 4 – Fluxo das camadas MVC



Fonte: Elaborada pelo autor.

Neste diagrama simplificado da Figura 4 é exemplificado a relação entre as camadas *Model, View* e *Controller*. As linhas sólidas indicam uma associação direta enquanto que as linhas tracejadas indicam uma associação indireta. Para LUCIANO E ALVES (2011), estes três elementos têm baixo acoplamento e alta coesão. Sendo que, o *Model* mantém o estado da aplicação, é uma classe que controla o fluxo e armazenamento dos dados, nesta camada devem estar implementadas todas as regras de negócios envolvendo conexões com o Banco de Dados. A camada *View* especificará exatamente como o modelo deve ser apresentado. Nesta camada será tradada a apresentação do sistema, o *layout* e fluxos de interfaces. Por fim, a camada *Controller*

irá traduzir as iterações do usuário com a camada *View*, mapeando estas ações para tarefas que o *Model* irá realizar.

A utilização desta arquitetura permite que o Laravel possua um alto desempenho além de uma melhor segurança se comparado com um sistema desenvolvido em PHP simples. A pesquisa desenvolvida por DAS E SAIKIA (2016) realizou um comparativo entre um sistema desenvolvido em PHP simples e outro utilizando o *Framework* Laravel. No trabalho foi implementado um sistema que realiza gravação, leitura, edição e deleção de dados nas duas tecnologias. O tempo e recursos gasto para cada tecnologia realizar as tarefas foi comparado. Assim, foi possível observar uma vantagem no sistema que utilizou o Laravel.

Além das vantagens citadas, a utilização do *Framework* Laravel facilita a realização de tarefas básicas como as mencionadas acimas. Para ler um conteúdo do Banco de Dados basta fazer uso da função *get()*, conforme demostrado no exemplo abaixo:

Código-fonte 4 – Obtendo dados com o Laravel

```
1 public function retorna_clientes(){
2  $clientes = DB::table('tabela-clientes')->get();
3
4  return view('pagina_clientes.index',['clientes' => $clientes])
     ;
5 }
```

Neste exemplo foi criada uma função que retorna todos os clientes cadastros na tabela *tabela-clientes*. Uma vez recebidos os dados eles são armazenados na variável do tipo *array* denominadas *$clientes* e, por fim, estes dados são transferidos para a página responsável por exibir os mesmos.

Da mesma maneira é possível realizar a edição, exclusão e gravação destes dados utilizando as funções: *update()*, *delete()* e *create()*. Para realizar estas mesmas operações com PHP simples seria necessário a criação de um SQL, o que demandaria um maior tempo. Caso houver necessidade de trocar o SGBD em algum momento o trabalho será ainda maior, uma vez que a linguagem SQL poderá ser diferente.

2.2.8 SQL

Estruturas SGBD são responsáveis por gerenciar os dados que o sistema produz. Para tanto, é utilizado a linguagem SQL. COSTA (2001) diz que com a linguagem SQL é possível criar, alterar e remover os componentes de uma Base de Dados, como exemplo as tabelas. Também é possível inserir, alterar e apagar dados específicos. Ainda é possível controlar o acesso aos dados dos utilizadores da mesma, e o controle das operações que cada usuário irá realizar.

Isto garante uma consistência e integridade dos dados produzidos pelo software, as principais funções da linguagem SQL são listadas abaixo:

***Create Database*:** cria uma base dados que irá armazenar os dados em tabelas;

***Create Table*:** cria uma tabela, que são estruturas baseadas em colunas e linhas. Onde a coluna identifica a informação e a linha é uma dada informação. Exemplo: Coluna Nome; Linha 1: Erikson Dutra de Miranda Silva; Linha 2: João Souza;

***Drop Table*:** apaga a tabela criada;

***Alter Table*:** altera a estrutura da tabela designada;

***Select*:** seleciona os dados especificados;

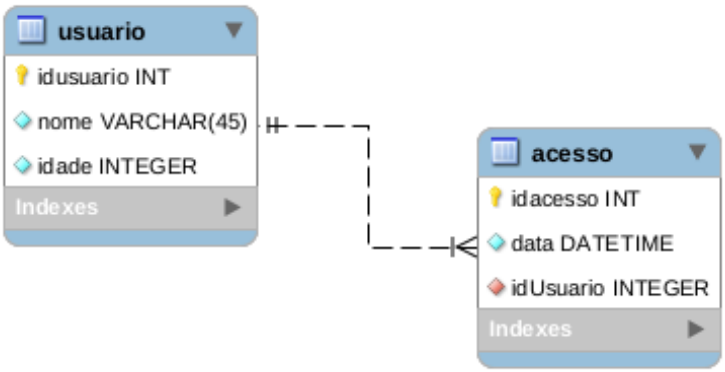
***Insert Into*:** insere dados em alguma tabela;

***Delete From*:** deleta os dados da tabela informada;

***Update*:** atualiza algum dado existente;

As tabelas possuem um identificador, este identificador é único para cada linha inserida em uma tabela. Assim, é possível relacionar dados que possuem um identificador em comum. Como demonstrado no exemplo da Figura 5:

Figura 5 – Relacionamento de Tabelas



Fonte: Elaborada pelo autor.

Neste exemplo é possível visualizar uma relação entre as tabelas *usuario* e *acesso*. Quando um usuário acessa um determinado sistema o mesmo irá gravar qual usuário acessou e em qual data e horário esse acesso foi realizado. É possível identificar o usuário que acessou verificando os dados contidos na tabela *acesso* por meio da coluna *idUsuario*. Essa coluna é comum para as duas tabelas citadas. A coluna *idUsuario* da tabela *usuario* é chamada de chave. Uma coluna chave identifica o dado, esta chave é única em todo o sistema. Na tabela *acesso* a

coluna *idUsuario* é denominada chave estrangeira, uma vez que a coluna "idAcesso" é a coluna chave desta tabela. A chave estrangeira pode ter vários valores repetidos, porem a coluna chave é único na tabela e no sistema.

2.2.9 Bootstrap

Um *Framework* bastante utilizado é o Bootstrap. SUPRLOCK (2013) diz que o Bootstrap é um produto de código aberto de Mark Otto e Jacob Thornton que, quando foi lançado inicialmente, eram ambos funcionários do Twitter. Havia uma necessidade de padronizar o conjunto de ferramentas de *front-end* de engenheiros em toda a empresa. Para tanto, foi necessário combinar as tecnologias CSS, JavaScript e o HTML. O resultado é uma interface mais rápida de se desenvolver e que se ajusta em qualquer tamanho de tela ou monitor. Isto agiliza o desenvolvimento, uma vez que com uma única linguagem é possível criar páginas totalmente responsivas com um *layout* fluido.

É possível observar que as tecnologias utilizadas para o desenvolvimento Web evoluíram muito no decorrer do tempo. A necessidade da geração de conteúdos dinâmicos e cada vez mais rápidos impulsionaram a criação de vários recursos para o desenvolvido de sistemas, tanto para o *client-side* como para o server-side. Cada sistema necessitará de uma determinada tecnologia. Assim caberá ao desenvolvedor realizar o levantamento dos requisitos e analisar quais serão as tecnologias indicadas para cada cenário. Com o tempo, novas necessidades iram surgir e isto irá contribuir para a evolução das tecnologias existentes para o desenvolvimento Web. Isto irá otimizar o custo e trabalho que se tem ao desenvolver uma solução para Web.

2.3 Trabalhos Relacionados

Neste tópico são discutidos alguns trabalhos relacionados à este estudo, visando analisar outras pesquisas científicas que fizeram uso de alguma tecnologia, voltadas ou não a Web, para o desenvolvimento de sistemas com foco na construção civil. Assim, foi possível colher dados relevantes para auxiliar nas dificuldades enfrentadas no desenvolvimento desta pesquisa.

2.3.1 EngPack Budget: Estudo e implementação de uma ferramenta de orçamento para construção civil.

Este trabalho foi desenvolvido por SILVA (2018), como Trabalho de Conclusão de Curso (TCC) para o curso de Ciência da Computação da Universidade Federal de Goiás (UFG). O objetivo foi implementar uma ferramenta para realizar orçamentos em construção civil utilizando a linguagem de programação PHP.

Pontos Positivos: Este trabalho possui uma boa estrutura para realizar orçamentação em construção civil. Sendo possível criar cadastros de insumos, composições e clientes. Também, é

possível consultar os dados cadastrados através de relatórios. O sistema realizou testes com estudantes do curso de Engenharia Civil na UFG e demonstrou estar apto a ser utilizado em instituições que tenham esta necessidade.

Pontos Negativos: Apesar de ter sido desenvolvido com tecnologias Web, o sistema não foi publicado na Internet, dependendo de emuladores para ser executado nas estações. Deste modo, seu funcionamento ficou *off-line*. Alem disto, os relatórios demoravam muito tempo para serem gerados, contribuindo negativamente com o uso da aplicação. Outro aspecto negativo é que o sistema não se adapta a diferentes tamanhos de telas, dificultando o uso em *smartphones* e *tablets*.

Diferenças para esta pesquisa: Embora esta pesquisa tenha apresentado um sistema com a mesma proposto deste trabalho as pesquisas se diferenciam. O trabalho de SILVA (2018) utilizou PHP simples. Entretanto, este trabalho fez uso de *frameworks* e padrões de projetos, o que possibilitou o desenvolvimento de mais recursos em menos tempo. Além de que, o foco desta pesquisa foi totalmente voltado para a experiencia do usuário e dos profissionais que darão manutenção neste sistema, enquanto que, a pesquisa de SILVA (2018) focou-se na apresentação da ferramenta.

2.3.2 Desenvolvimento de um software para o controle de obras na Construção Civil

Este trabalho desenvolvido por RALL, CAMPAGNA E JR (2014), apresenta um sistema para controle de obras no setor da construção civil com foco em pequenos negócios desenvolvido na tecnologia Delphi 7 Enterprise.

Pontos Positivos: Esta pesquisa apresentou uso de padrões de projetos com uso de recursos da Engenharia de Software além de ter contribuído com aspectos técnicos do controle de obra no setor da construção civil. Isto colaborou com o desenvolvimento deste trabalho, ajudando na tomada de decisões.

Pontos Negativos: O trabalho não empregou testes com usuários, o que dificulta a avaliação da aceitação do software. Além de tudo, o sistema utilizou uma tecnologia voltada para desktop. Que enfrenta os problemas descritos no Capítulo 1.

Diferenças para esta pesquisa: A pesquisa de RALL, CAMPAGNA E JR (2014) se difere desta no emprego da tecnologia utilizada, sendo que este trabalho utiliza recursos da Web enquanto o outro utiliza Delphi, mantido por uma organização privada, a Embarcadero [4]. Outra divergência se da pelos objetivos, sendo que o estudo apresentou o desenvolvimento de um software para controle de obras no setor da construção civil enquanto esta pesquisa apresentou o desenvolvimento de um sistema Web para orçamentação em obras.

2.3.3 Considerações Finais

Com a análise destes trabalhos foi possível observar os pontos positivos e negativos dos mesmos. Estes pontos serviram de base para o desenvolvimento desta pesquisa, contribuindo com soluções que as mesmas não apresentaram.

Outro aspecto observado é o fato de nenhum destes sistemas descritos terem sido disponibilizadas na Internet, inviabilizando o seu uso para o ensino e estudo de orçamentação e gestão em construção civil. Esses resultados apontam a relevância do desenvolvimento deste trabalho como uma forma alternativa para utilização e estudo de sistemas orçamentários.

[4] Embarcadero. https://www.embarcadero.com/br/products/delphi

CAPÍTULO

3

ORÇAMENTO NA CONSTRUÇÃO CIVIL

Este Capítulo abordará os conceitos e as estruturas necessárias para realizar um orçamento em construção civil. Serão apresentandos a Estrutura Analítica de Projeto utilizada na segunda versão do Engpack Budget e alguns exemplos de orçamentação.

3.1 Estrutura Analítica de Projeto

Do ponto de vista da gestão de edificações após a fase de elaboração de projetos executivos é primordial que se construa uma estrutura base para os ritos de orçamentação e planejamento da construção. Essa estrutura base pode ser chamada de EAP – Estrutura Analítica de Projeto. MATTOS (2010) e OBERLENDER (2014) define que a EAP consiste em dividir o projeto em pacotes de trabalhos que podem ser melhor controlados e administrados. A Figura 6 apresenta um modelo completo e genérico de EAP enquanto a Figura 7 apresenta a discretização dos elementos componentes de uma EAP da estrutura de concreto armado de um edifício.

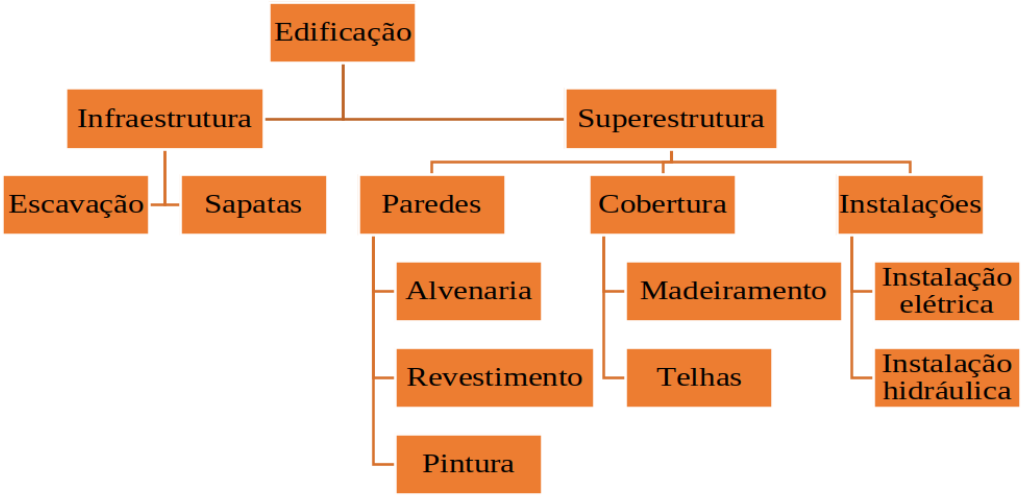
Portanto com base nessa EAP tanto orçamento quanto planejamento de obras podem ser construídos. A seguir, nas seções subsequentes serão abordados alguns aspectos relativos ao processo de orçamentação e planejamento de edificações.

3.2 A orçamentação em uma edificação civil

Para uma eficiente estimativa dos custos é preciso levar em considerações alguns aspectos, como o custo unitário dos produtos, a variabilidade do preço desses produtos, facilidade de acesso aos materiais para construção, e por último, o momento do orçamentário.

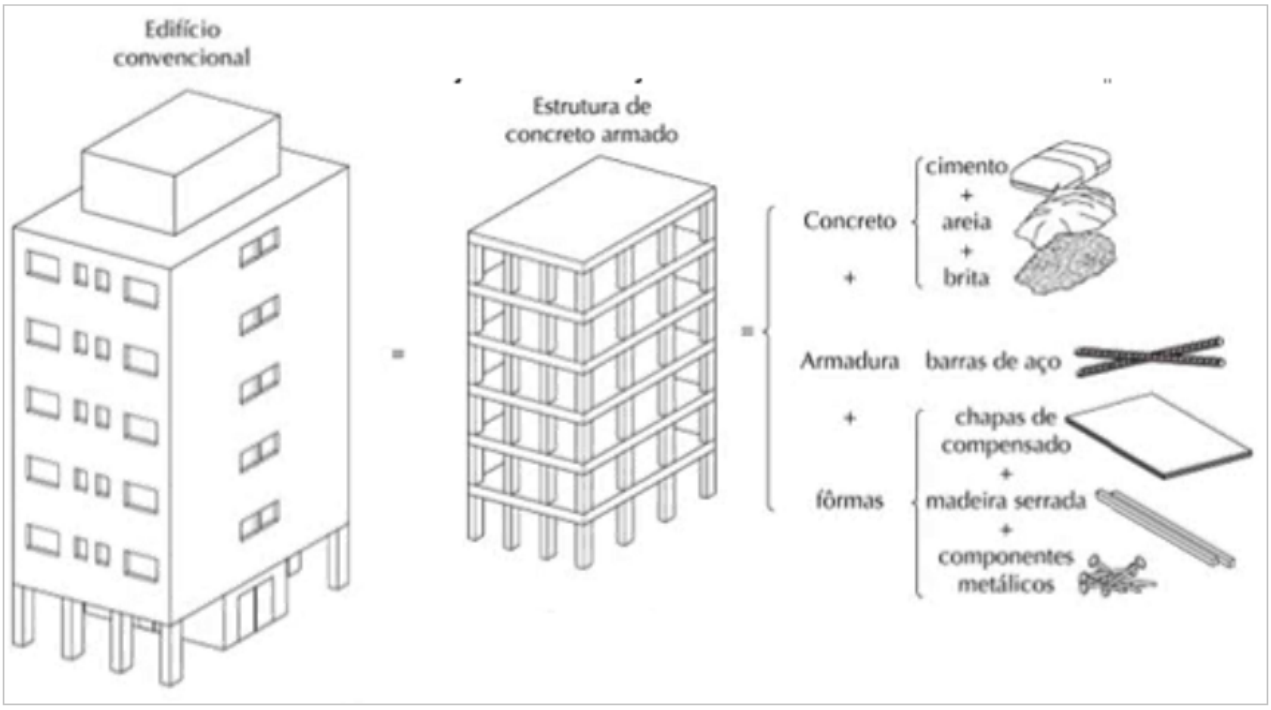
De acordo com ROWINGS JR. (2003), o orçamento é, em geral, utilizado para se ter uma ideia do valor do projeto que será executado. Geralmente, os envolvidos no processo de concepção de um projeto tendem a fazer a seguinte pergunta: “quanto custará a execução do

Figura 6 – Modelo básico de uma EAP para uma edificação residencial



Fonte: Elaborada pelo autor.

Figura 7 – EAP da estrutura de concreto armado para um edifício de múltiplos pavimentos



Fonte: Elaborada pelo autor.

projeto?". Para responder esta pergunta, vários tipos de orçamentos e estimativas devem ser desenvolvidos. A meta do orçamentista é se aproximar do real custo de execução.

De uma maneira rudimentar, pode-se dizer que na engenharia o orçamento utilizado é referente ao levantamento de custo, para executar uma obra ou um empreendimento, no qual quanto mais detalhado for o orçamento mais ele vai se aproximar do custo real SAMPAIO (1991).

Para JUNGLES E AVILA (2006) o orçamento pode ser visto de três formas distintas, como um levantamento de custo, como um produto e como um processo. No caso desse trabalho o orçamento será tratado sempre como um levantamento de custo onde o usuário fará o levantamento real de serviços na tentativa de estimar o custo real da edificação adicionalmente estimando também um tempo para realização desses serviços. Para esse levantamento de custo o software desenvolvido toma como base composições unitárias de custo.

Esse processo de orçamentação com enfoque no levantamento de custos pode-se dividir basicamente em duas vertentes, são elas: (a) A estimativa de custo; e (b) Orçamento detalhado ou analítico.

BAETA (2012) define a estimativa de custo como uma avaliação expedita realizada com base em "custos históricos, índices, gráficos, estudos de ordem de grandeza, correlação ou comparação com projetos similares".

Uma das ferramentas mais utilizadas para estimativa de custo é o CUB ou Custo Unitário Básico que consiste em um índice que leva em consideração a área da construção de forma a se obter o valor total da execução da uma construção em específico. Diversos órgãos disponibilizam esses valores em função das regras estabelecidas pela NBR 12721 (2006) . A Figura 8, por exemplo, apresenta o custo de total da construção de edificações residenciais para o estado de Goiás.

Figura 8 – Custo Unitário Básico (CUB) para projetos padrão residenciais com valores em Reais / Metro Quandrado

Padrão baixo		Padrão normal		Padrão alto	
R-1	1.252,05	R-1	1.524,30	R-1	1.807,07
PP-4	1.095,00	PP-4	1.417,31	R-8	1.440,61
R-8	1.038,39	R-8	1.228,51	R-16	1.548,00
PIS	815,97	R-16	1.183,74		

Nota legenda:
R-1: Residência Unifamiliar/1 pavimento
PP-4: Prédio Popular/4 pavimentos
R-8: Residência Multifamiliar/8 pavimentos
PIS: Projeto de Interesse Social
R-16: Residência Multifamiliar/16 pavimentos

Fonte: (SINDUSCON-GO, 2019)

LOSSO (1995) afirma que deve ficar claro que a estimativa de custo não tem a pretensão nem mesmo o objetivo de precisar o valor de uma determinada obra, e sim apresentar um intervalo no qual, dependendo das considerações tomadas como parâmetros, o custo do empreendimento esteja compreendido.

CABRAL (1988) define o orçamento analítico como sendo resultado da discriminação da obra nos seus diversos serviços que por sua vez, tem quantidades determinadas e associadas ao

custo unitário de execução. Segundo ele, ainda, nesta modalidade de orçamento são orçados os serviços independentemente, onde atuam três variáveis: o quantitativo dos serviços, a composição unitária e o preço dos insumos.

Em linhas gerais um orçamento analítico fornece uma grande quantidade de detalhes a fim de se realizar a localização e análise dos possíveis impactos de custos, como também permitir a equipe de engenharia total gerencia sobre a gestão dos custos da construção. MATTOS (2014) afirma que devido a esse alto grau de detalhamento o mesmo é mais próximo do custo "real" da construção.

FREIRE (2018), SILVA E CAMPOS (2015) afirmam que um orçamento analítico pode ser divido nas seguintes etapas:

1. Análise das características: Através da análise dos projetos executivos observa-se os itens e as divergências do empreendimento, como local de acesso a entrada de materiais, armazenamento, disponibilidade de água e energia até a completa definição da Estrutura Analítica de Projeto (EAP);

2. Levantamento quantitativo: Trata-se de uma das etapas mais importantes, pois quantifica-se todos os materiais e atividades necessárias para a execução do empreendimento. Deve ser feita de maneira minuciosa e abranger todos os serviços e quantidade de trabalho;

3. Composição de custos unitários: Onde acrescenta-se a cada serviço um custo unitário básico, levando em consideração a produtividade de mão de obra e os insumos para a produção do serviço;

4. Memorial do orçamento: Por fim gera-se o memorial de orçamento, contendo todas as especificações de materiais e serviços inerentes a obra, assim como o orçamento final SOUZA (2005).

Com base nessa EAP os levantamentos de quantidade podem ser efetuados de maneira detalha e com observância ao projeto executivo em questão de forma que o mesmo contemple todos os serviços requeridos em uma obra tornando o mesmo o mais preciso quanto se possa.

Ferramentas computacionais, como a plataforma BIM (*Building Information Model*), têm sido bastante utilizada no ambiente de criação de projetos de engenharia RODRIGUES (2017) como também no processo de quantificação de serviço. Exemplos de uso dessa ferramenta podem ser visualizados em: BIOTTO, FORMOSO E ISATTO (2015); CARMONA E CARVALHO (2017); FENATO (2018); FREIRE (2018); SANTOS (2009) e RUNDELL (2006) afirma que que este processo está sujeito a erro humano e tende a propagar imprecisões. A quantificação também é demorada e pode exigir 50% a 80% de uma estimativa de custo de tempo em um projeto.

SANTOS (2009) ainda afirma que na lógica do orçamento executivo, a forma de levantamento de quantitativos vai ao encontro à estratégia de execução do empreendimento, ou seja, o mesmo seguirá a sequência estabelecida ne EAP do projeto.

Com toda a EAP definida e também os levantamentos é possível estabelecer conexões com as composições de custo do projeto. MATTOS (2014) afirma que a composição é o estabelecimento dos custos incorridos para a execução de um serviço ou atividade, individualizado por insumo e de acordo com certos requisitos pré-estabelecidos. A composição lista todos os insumos que entram na execução do serviço, com suas respectivas quantidades, e seus custos unitários e totais.

MATTOS, 2014) afirma que uma composição de custo contempla os seguintes componentes:

1. Insumo: Cada um dos itens de material, mão de obra e equipamento que entram na execução direta do serviço;

2. Unidade: Unidade de medida/cotação do insumo (kg, metro cubico, metro quadrado, metro, hora, homem-hora);

3. Índice: Incidência de cada insumo na execução de uma unidade do serviço;

4. Custo unitário: Custo de aquisição ou emprego de uma unidade do insumo;

5. Custo total: Custo total do insumo na composição de custos unitários. É obtido pela multiplicação do índice pelo custo unitário.

Neste contexto, a Tabela 1 apresenta um exemplo de composição unitária de custo para a produção de 1 metro quadrado de alvenaria de vedação com blocos vazados de concreto.

As composições de custo devem ser estudadas com cuidado, pois estas podem interferência regional tanto na produtividade como também no preço dos insumos. Em linhas gerais essas composições podem ser criadas dentro da própria empresa através de indicadores de produtividade da série histórica dessa empresa, como também através da utilização de bancos de composições, sendo que esse último é a alternativa mais utilizada nas instituições e empresas de engenharia. Segundo MARCHIORI (2009) destacam-se bancos nacionais como: (1) o SINAPI (Sistema Nacional de Preços e Índices para a Construção Civil) que tem sua origem ligada a Caixa Econômica Federal (CEF) e é amplamente utilizado em licitações a nível federal em instituições públicas; e (2) a TCPO (Tabela de Composição de Preços para Orçamentos) que é um manual editado pela Editora Pini e é amplamente utilizado no mercado da construção civil. Já na esfera estadual e municipal normalmente agências de obras apresentam anualmente suas versões de bancos de informações como por exemplo no estado de Goiás a Agência Goiana de Infraestrutura e Transportes – GOINFRA.

Tabela 1 – Composição unitária de custo para 1 metro quadrado de alvenaria

Insumo	**Unidade**	**Índice**	**Custo unitário (R$/ Metro Quadrado)**	**Custo total dos (R$/ Metro Quadrado)**
Pedreiro	h	0,7200	17,50	12,60
Servente	h	0,3600	12,23	4,40
Bloco vedação concreto 9x19x39 cm	un	13,3500	1,80	24,03
Argamassa (cimento, cal e areia média)	m3	0,0088	352,49	3,10
Tela de aço	m	0,7850	1,35	1,03
Pino de aço	cento	0,0094	43,29	0,41
Total da composição de custo unitário do serviço				45,60

3.2.1 Considerações Finais

Portanto o custo final de uma obra será delimitado pelos somatórios dos custos diretos e indiretos envolvidos na construção de uma edificação. KERN E FORMOSO (2006) afirma que os custos diretos são estimados por composições de custos relativas às atividades de transformação da obra, através de coeficientes de consumo para cada insumo da atividade orçada, enquanto os custos indiretos geralmente são estimados para cada construtora ou construção JESUS E BARROS (2011), podendo esses conter por exemplo: (1) mobilização e desmobilização de equipamento e itens do canteiro de obras; (2) administração local e central; (3) Itens de segurança; e (4) Taxas de financiamento.

CAPÍTULO

4

DESENVOLVIMENTO DO SISTEMA

Neste Capítulo serão descritas as técnicas e ferramentas utilizadas para o desenvolvimento do sistema e as questões que foram levantadas para realizar a comparação dos dois sistemas. Por fim, os requisitos levantados serão descritos para exemplificar os recursos que o sistema possui.

4.1 Descrição das técnicas e tecnologias utilizadas

Para o desenvolvimento do sistema foi necessário realizar o estudo de diversas técnicas e tecnologias citadas no Capítulo 2 deste trabalho. O principal objetivo era melhorar a versão 1.0 do Engpack Budget desenvolvida por SILVA (2018), tratando as principais falhas que a mesma apresentava e adequando a ferramenta para versão Web.

De acordo com Sommerville (2011), os processos de desenvolvimento ágil de softwares são utilizados para produzir, de forma rápida, softwares úteis. O software não é desenvolvido como uma única unidade e sim como uma série de módulos, incrementos, sendo que cada módulo inclui uma nova funcionalidade ao sistema final.

Conforme discutido no tópico 1.2, foram utilizados estudos e aplicação das metologias agíeis XP e SCRUM. Como apresentado por Sommerville (2011) a metodologia XP, talvez seja a mais conhecida. Ela tem como característica o fato de que os requisitos são expressos como cenários, conhecidos como estórias do usuário, sendo implementados como uma cadeia de tarefas. Outra característica e o fato dos programadores trabalharem em pares, além de desenvolverem testes para cada tarefa antes de escrever o código propriamente dito.

De acordo com Sommerville (2011) o método de desenvolvimento ágil SCRUM é composto de três fases. Iniciando pela fase de planejamento geral, onde e estabelecido os objetivos gerais do projeto e a arquitetura do sistema, seguindo pela fase de ciclos de sprints, onde em cada ciclo e desenvolvido um incremento do sistema. Por último, finaliza-se o projeto, completando nessa fase a documentação necessária, que contem quadros de ajuda do sistema

juntamente com manuais do usuário, além de avaliar as lições aprendidas com o projeto.

Além destas metologias outra metologia serviu de base para atingir os objetivos que foram propostos neste trabalho, a utilizam da abordagem GQM. Nesta metologia foram definidos os pontos que deveriam ser melhorados da primeira versão do sistema, as questões que serviram de base para realizar as melhorias propostas e por fim as métricas com objetivo de comparar os resultados alcançando.

Sendo assim, o trabalho seguiu-se para o levantamento dos Objetivos na Fase *Goal*. O primeiro objetivo definido era a necessidade de haver padrões. Visto que, o trabalho tem por meta ser continuado por outros alunos do grupo Coretec.

Visando solucionar os problemas discutidos no Capítulo 2 foram definidas tecnologias especificas para todas as ferramentas que seriam desenvolvidas pelo grupo Coretec a partir do ano de 2019, além disto, foi definido que todos os sistemas seriam voltados para Web. Essas tecnologias são separadas em *Front-End* e *Back-End*, conforme observa-se nos itens listados abaixo:

Tecnologias para o *Front-End*: Foram definidas a utilização do *Framework* Bootstrap, discutido no Capítulo 2, para criação das páginas. A escolha deste *Framework* foi necessária, visto que os sistemas deveriam ter a capacidade ser executado em qualquer dispositivo. Podendo assim ser acessado de qualquer localidade, havendo acesso a Internet.

Além deste *Framework*, foi definido o uso da linguagem de programação JavaScript para controlar interações do usuário com os sistemas desenvolvidos.

Tecnologias para o *Back–End*: Foram definidas a utilização do *Framework* Laravel, baseado no PHP. As vantagens deste *Framework* foi discutida no Capítulo 2. Sua utilização foi de grande importância a fim de dimuniur a dificuldade na solução dos problemas na manutenção e criação de recursos nos sistemas que existem e os que serão desenvolvidos. Uma vez que o objetivo final é que todas as ferramentas se unem. Cada ferramenta desenvolvida será um modulo de uma ferramenta global.

Como SGBD foi definido a utilização do MySQL. Uma vez que este SGBD é bastante popular e de uso gratuito. Além disto, o mesmo possui uma excelente documentação com exemplos práticos, COSTA (2001).

4.1.1 Levantamento das Questões

Após a definição das tecnologias com o grupo este trabalho partiu para análise da primeira versão do sistema, o Engpack Budget 1.0. Para tanto foi necessário a realização de diversas reuniões com os *stakeholders* e testes na versão 1.0 do sistema. Os testes foram divididos em testes de Interface, Banco de Dados e Desempenho. A Tabela 2 abaixo ilustra os principais problemas identificados nestas Fases:

Tabela 2 – Principais Problemas da Versão 1.0

Interface	Utilização de Varias Tecnologias, faltando padrões no projeto; *Layout* não responsivo; Interface sem fluidez.
Banco de Dados	Tabelas não são normalizadas; Banco de Dados não possui funcionários.
Desempenho	Lentidão para gerar relatórios, fazer leitura e gravação no Banco de Dados e carregar formulários.

Foi possível identificar que os formulários não seguiam um padrão de desenvolvimento. Alguns lançamentos eram realizados utilizando a tecnologia Bootstrap. Enquanto que em outros formulários os lançamentos eram realizados utilizando somente a linguagem de marcação HTML. Isto contribuiu para uma interface sem fluidez. Visto que em alguns pontos o *layout* seguia um tema e outro não. Além disto, o sistema não era capaz de ser executado em dispositivos que possuem uma tela com poucas polegadas, como smartphones.

Outro problema identificado foi a estrutura do Banco de Dados, não havia normalização. Ou seja, não foram implementadas regras a todas as tabelas do Banco de Dados com o objetivo de evitar falhas no projeto, como redundância de dados e mistura de diferentes assuntos numa mesma tabela. Outro aspecto que dificultou a análise foi a falta de documentação do Banco de Dados e funções. Deste modo, não era possível saber com clareza o que cada tabela armazenava.

Como não havia normalização no Banco de Dados o resultado com consultas era muito lento. Gastando vários minutos para gerar um relatório com pouco mais que 1000 registros. Isto dificultava o carregamento das páginas e o que poderia dificultar a navegação.

Como a primeira versão do sistema não foi desenvolvida com as tecnologias definidas pelo grupo e apresentava falhas difíceis de identificar e tratar foi optado por desenvolver um outro sistema utilizando os recursos definidos. Além disto, foi definida a documentação da segunda versão do sistema, manual do usuário e a utilização de algumas ferramentas para que o grupo pudesse acompanhar os requisitos que estavam sendo levantados e posteriormente desenvolvidos.

Algumas regras de negócio e fluxos foram aproveitados da primeira versão. As ferramentas mencionadas são citadas abaixo:

GitHub [1]: Plataforma de hospedagem de código-fonte com controle de versão. Assim é possível que os desenvolvedores e usuários cadastrados na plataforma consultem os códigos e documentos que eram produzidos.

Trello [2]: É uma aplicação de gerenciamento de projeto baseado na Web. Deste modo era possível informar os requisitos que seriam desenvolvidos, controlar o prazo de entrega e controlar o que estava sendo desenvolvido.

Com isto, o desenvolvimento do sistema seguiu-se para a Fase *Question*. Esta parte da metologia é responsável pela medição, *feedback* e avaliação, BASILI, CALDIERA E ROMBACH (2003). É nesta Fase que, também, foram levantados os dados e realizada as reuniões necessárias para o desenvolvimento da segunda versão do Engpack Budget.

Para elaborar estas perguntas reuniões foram realizadas reuniões com profissionais de programação que tinham experiências nas tecnologias que a primeira versão utilizou. A finalidade foi entender a dificuldade e o tempo que seria gasto em uma manutenção do sistema no futuro. Assim, foram definidas as seguintes perguntas:

1. O Software atual demora quanto tempo para exibir um relatório?

2. Qual seria a dificuldade para realizar uma alteração no Banco de Dados?

3. Qual seria a dificuldade para um novo desenvolvedor entender a função responsável por gravar o orçamento no Banco de Dados?

4. Caso fosse necessário usar outro SGBD seria possível realizar a troca?

5. Quantas linguagens foram utilizadas para desenvolver o *Front-End*?

6. Quantas linguagens foram utilizadas para desenvolver o *Back-End*?

Estas perguntas foram necessárias para poder traçar as métricas obtidas com o desenvolvimento da segunda versão. Deste modo, reuniões foram realizadas periodicamente a fim de responder estas perguntas. Na fase das reuniões foram definidos prazos e testes para a entrega do sistema. Apresentações foram realizadas em quinzenas afim de entregar partes do software funcionais, como, por exemplo, o cadastro de Insumos, Composições e Clientes. Nesta etapa eram definidas correções e melhorias para o item apresentado.

4.1.2 Levantamento dos Requisitos

As revisões eram feitas após a fase de reunião. Nesta etapa, falhas ou novas necessidades eram identificadas a fim de obter melhorias no software como um todo. Caso houvesse alguma necessidade outra reunião era realizada afim de definir com detalhes os requisitos.

O desenvolvimento foi a última etapa desta pesquisa. Sempre visando o desenvolvimento ágil os requisitos eram bem definidos para minimizar ou anular o retrabalho. O sistema ficou online na internet e podê ser acessado pelos participantes do projeto com o intuito de realizar testes e identificar possíveis falhas e melhorias. Abaixo, nas Tabela 3 e 4, segue os requisitos levantados para o desenvolvimento do sistema:

[1] GitHub. https://github.com/.
[2] Trello. https://trello.com/.

Tabela 3 – Requisitos Funcionais do sistema

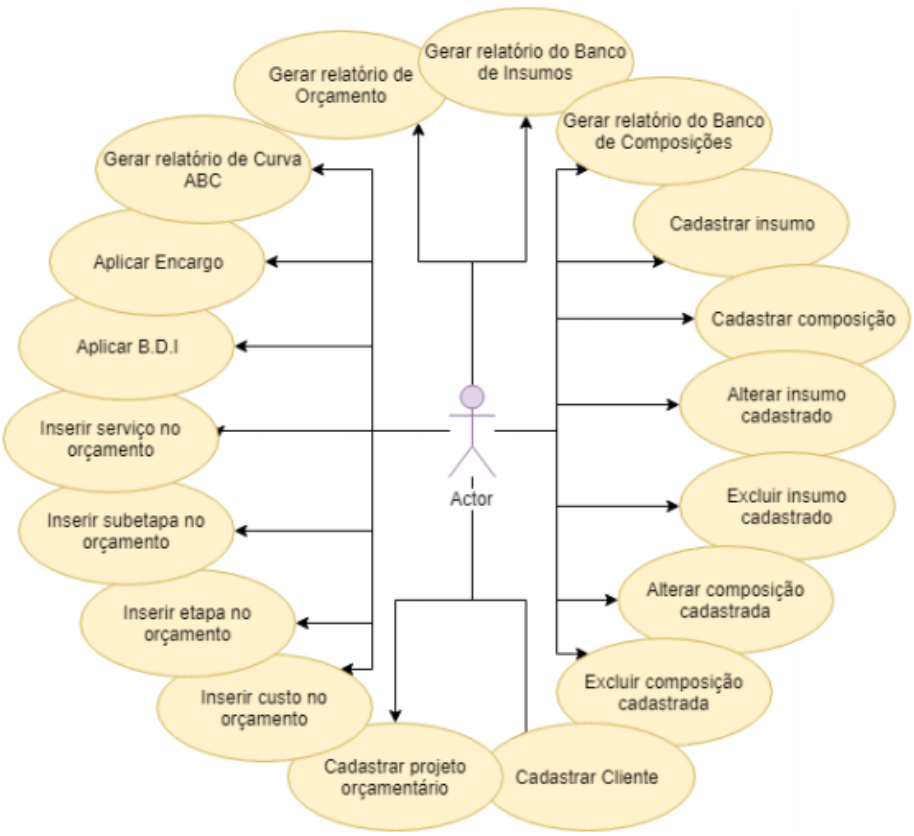
Nº	**Requisitos Funcionais do sistema**
R01	Criação de um método para cadastramento de Insumos no Banco de Dados.
R02	Criação de um método para cadastramento de Composições no Banco de Dados.
R03	Criação de um método para manipular os Insumos cadastrados no Banco de Dados. Manipulação: Alterar e Excluir.
R04	Criação de um método para manipular as Composições cadastradas no Banco de Dados. Manipulação: Alterar e Excluir.
R05	Criação de um método para cadastramento de clientes.
R06	Criação de um método para criação de um novo projeto orçamentário.
R07	Criação de um método para manipular os projetos orçamentários criados. Manipulação: Alterar e Excluir.
R08	Criação de um método para transacionar para a página inicial do sistema.
R09	Implementação de um sistema de aviso, para quando for excluir algum Insumo, Composição e/ou Projeto.
R10	Criação de uma tabela com característica hierárquica para inserção do orçamento.
R11	Criação de um método para inserção de itens a nível de Custo, no orçamento.
R12	Criação de um método para inserção de itens a nível de Etapa, no orçamento.
R13	Criação de um método para inserção de itens a nível de Subetapa, no orçamento.
R14	Criação de um método para inserção de itens a nível de Serviço, no orçamento.
R15	Criação de um método para análise detalhada dos itens do nível de Serviço.
R16	Criação de um método para manipulação dos itens de todos os níveis, no orçamento. Manipulação: Alterar e Excluir.
R17	Criação de um método para inserção do B.D.I, no orçamento.
R18	Criação de um método para inserção dos Encargos, no orçamento.
R19	Implementação de um método para coloração única para cada nível do orçamento.
R20	Criação de um método para gerar relatório de Curva ABC.
R21	Criação de um método para gerar relatório do Orçamento.
R22	Criação de um banco de Insumos personalizado
R23	Criação de um banco de Composições personalizado
R24	Criação de informações na Página Inicial do sistema

Tabela 4 – Requisitos Não Funcionais do sistema

Nº	**Requisitos Não Funcionais do sistema**
RN01	O sistema deve ser intuitivo e de fácil utilização.
RN02	O usuário não precisa ter um treinamento especializado para utilizar o sistema.
RN03	O sistema deverá estar disponível para a plataforma Web.
RN04	Versões do sistema devem ser entregues sempre na próxima reunião, com as funcionalidades elicitadas na reunião passada.
RN05	O sistema deve ser responsivo.
RN06	O sistema deve ser gratuito.
RN07	O sistema deve ser desenvolvido com Laravel MVC, Bootstrap e JavaScript
RN08	O sistema deve possuir manual de usuário
RN09	O sistema deve possuir documentação

Para uma melhor compreensão destes requisitos apresentados na Tabela 3 alguns diagramas foram criados. De acordo com SOMMERVILLE (2011), diagramas de casos de uso apresentam interações entre os usuários e o sistema. Ou seja, cada caso de uso representa uma tarefa que engloba a interação externa com um determinado sistema. Assim, é ilustrado na Figura 9 o diagrama de caso de uso da segunda versão do Engpack Budget.

Figura 9 – Diagrama de Caso de Uso da Segunda Versão



Fonte: Elaborada pelo autor.

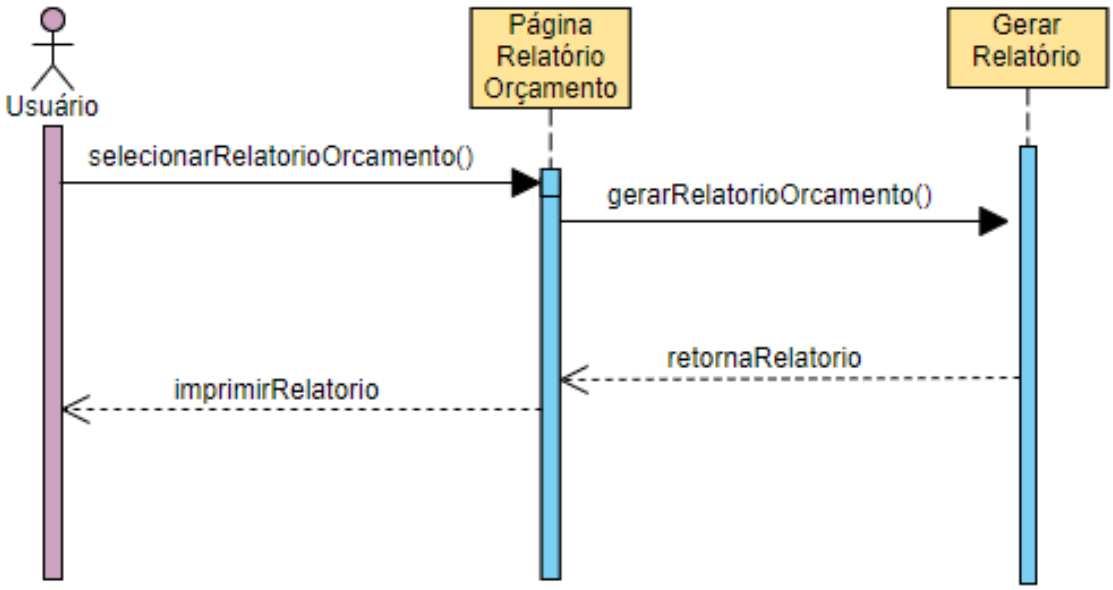
Na Figura 9 são representadas todas as possíveis iterações com o sistema. Essas funci-

onalidades foram levantadas nas reuniões realizadas com todos envolvidos no projeto. Outro diagrama desenvolvido para melhor compreender as funcionalidades do sistema é o de Sequência.

Conforme SOMMERVILLE (2011), diagramas de sequência são utilizados, em sua maioria, para modelar as interações entre os atores e objetos do sistema, e/ou atores com atores. Segundo o autor, um diagrama de sequência é feito sobre um único caso de uso, como uma forma complementar, no qual, irá apresentar a sequência de interações que ocorrem.

Em um diagrama de sequência, tantos os objetos quanto os atores, são listados na parte superior do diagrama, com uma linha tracejada verticalmente a partir deles; Interações são indicadas com setas, que carregam consigo funções do sistema. Por último, os retângulos sobre as linhas tracejadas, representam o tempo de vida do objeto em questão, ou seja, o tempo em que determinado objeto está envolvido no processamento, SOMMERVILLE (2011). A Figura 10 ilustra um diagrama de sequência do recurso de gerar relatório do orçamento.

Figura 10 – Diagrama de Sequência para gerar relatório do orçamento



Fonte: Elaborada pelo autor.

De modo resumido, o diagrama de sequência da Figura 10 informa que o usuário irá solicitar o relatório de orçamento por meio da função *selecionarRelatorioOrcamento()*, a função irá executar uma consulta no Banco de Dados retornando o relatório para o usuário.

4.2 Considerações Finais

As ferramentas descritas neste Capítulo foram indispensáveis para o desenvolvimento deste sistema, uma vez que, sem as mesmas, a organização do projeto seria custosa. Deste modo, foi possível focar nos requisitos necessários para o funcionamento do software, descritos

na Tabela 3. Os requisitos descritos na Tabela 4 são definidos como não funcionais, visto que os mesmos são relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenção e tecnologias envolvidas.

CAPÍTULO

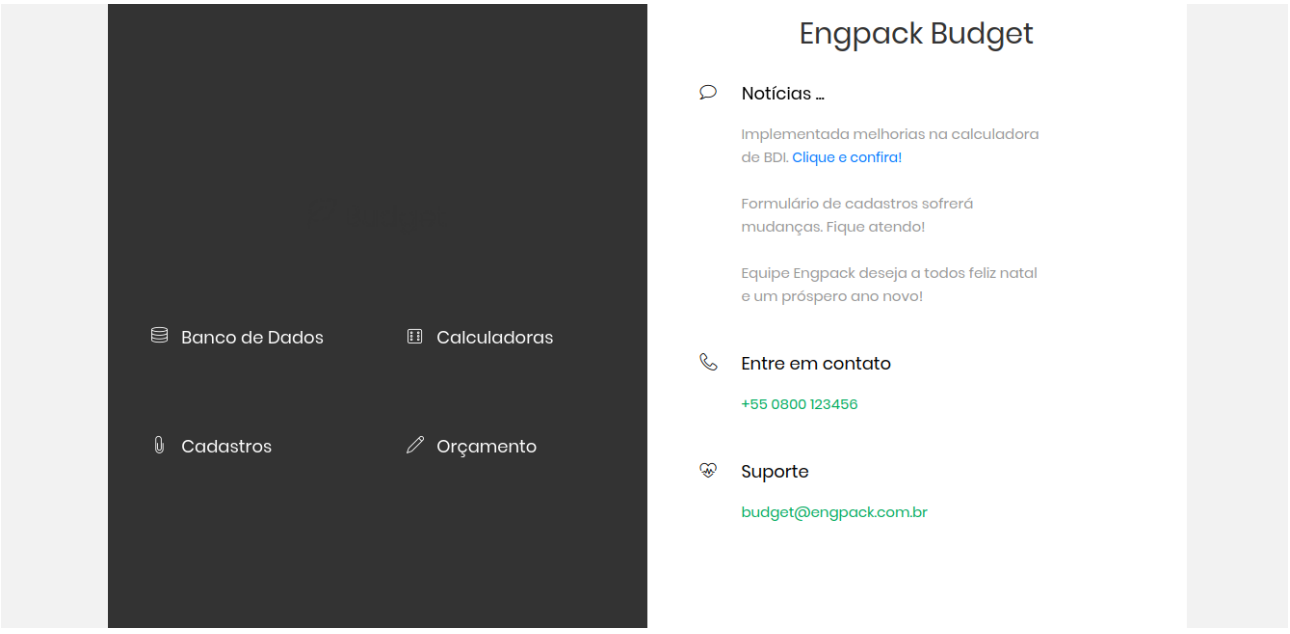
5

APRESENTAÇÃO DO SISTEMA

Este Capítulo apresenta a segunda versão do EngPack Budget. São exibidas todas as páginas juntamente com suas devidas funcionalidades. Adicionalmente, é apresentado o diagrama de relacionamento do Banco de Dados que, devido aos motivos citados no Capítulo anterior, teve que ser refeita.

No último Capítulo deste trabalho serão apresentadas as métricas e os resultados obtidos com o desenvolvimento desta pesquisa. Assim, na Figura 11 é apresentada a página inicial da segunda versão do sistema.

Figura 11 – Página Inicial do sistema Engpack Budget



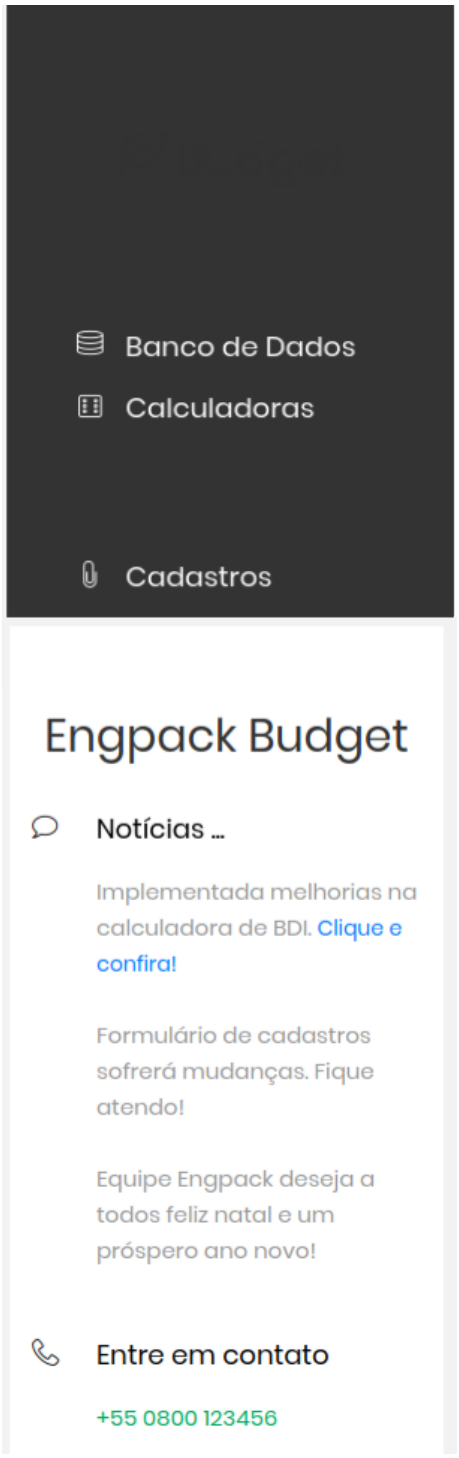
Fonte: Elaborada pelo autor.

A Página Inicial do Sistema foi pensada da maneira mais simplificada possível. Como é possível observar, do lado esquerdo ficam os menus, onde é possível acessar os recursos do

sistema. Do lado direito, ficam informações úteis para o usuário, onde é exibido um pequeno *feed* de notícias que será atualizado de acordo com a necessidade e informações para contato.

Como o sistema foi desenvolvido inteiramente com o Bootstrap, esta página será redimensionada para se adaptar a qualquer monitor que a exiba conforme é possível observar na Figura 12.

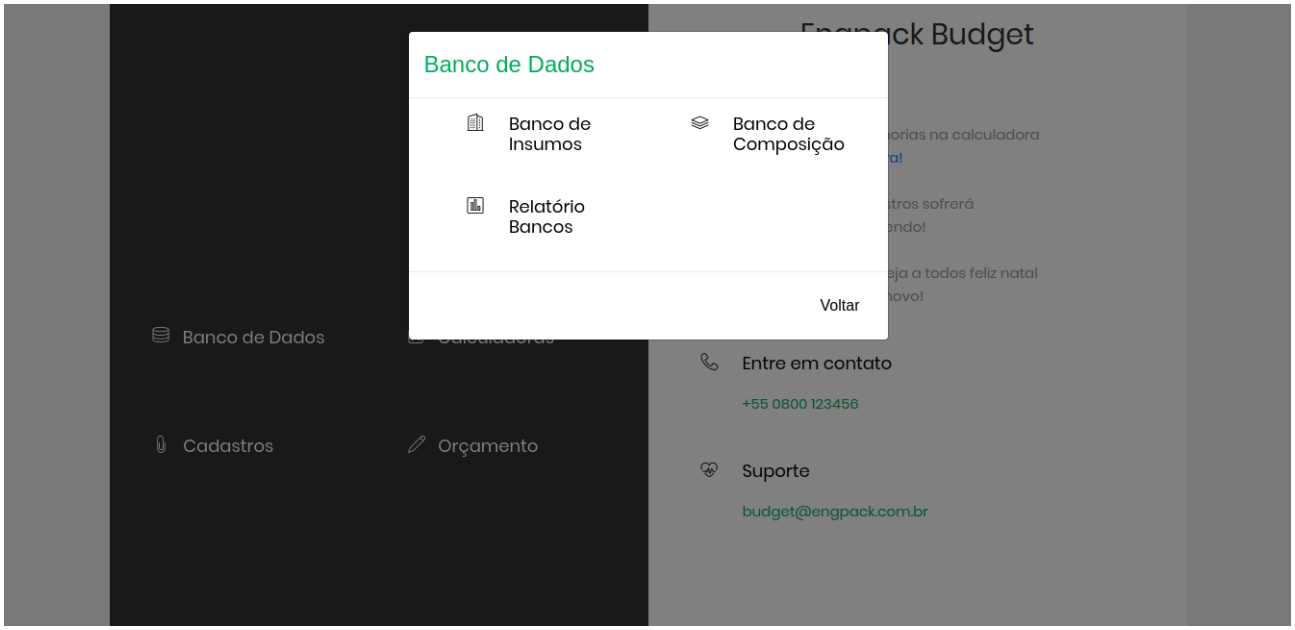
Figura 12 – Página Inicial Aberta a Partir de um SmartPhone



Fonte: Elaborada pelo autor.

As cores foram retiradas da paleta de cores disponibilizada pela Goolge Inc, citada no Anexo A, foi optado por usar cores neutras a fim de agradar a maior quantidade de usuários possível. Os ícones foram obtidos pelo repositório Font Awesome, informado no Anexo A, que foi desenvolvido com elementos específicos para serem utilizados junto ao HTML. Seguindo com a apresentação da segunda versão do sistema, é apresentado na Figura 13 o Menu de Banco de Dados.

Figura 13 – Menu Banco de Dados



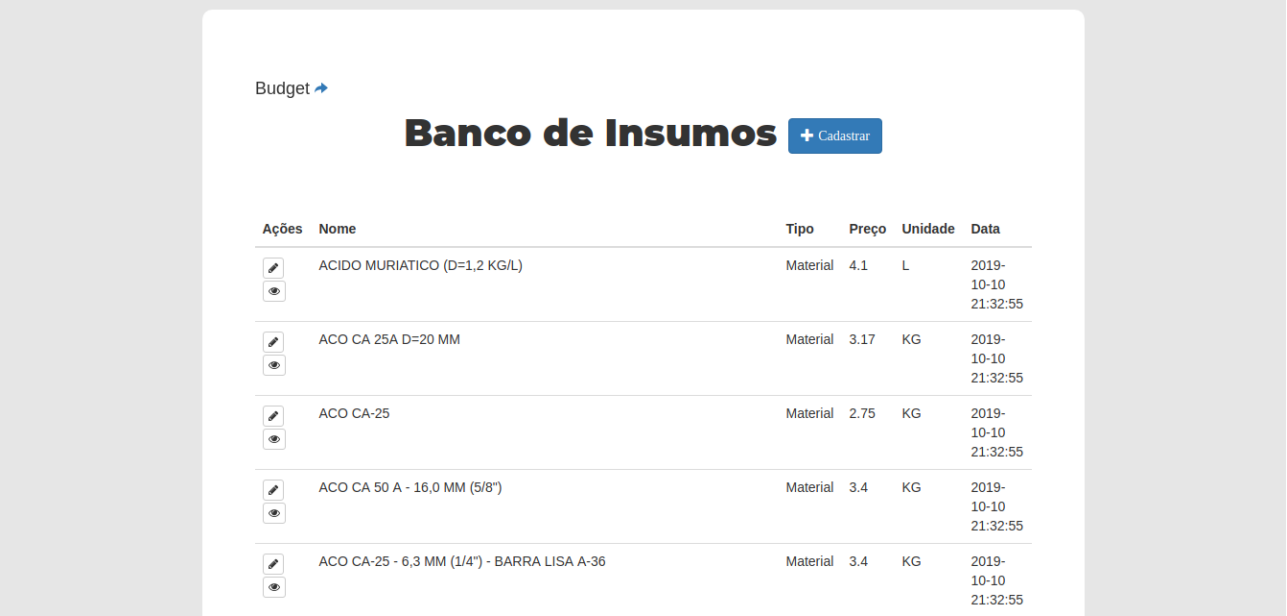
Fonte: Elaborada pelo autor.

A Figura 13 exibe os itens contidos no Menu Banco de Dados. Uma particularidade neste menu é o uso de um recurso que o Bootstrap disponibiliza denominado Modal. O uso de modal tem o objetivo de economizar espaço e recursos. Levando em conta que o sistema será executado em tablets e smartphones que possuem poucos recursos e espaço a segunda versão explorou bastante esta funcionalidade do Bootstrap. Deste modo, quando o usuário clicar no botão *Banco de Dados* a Página Inicial do sistema ficará por trás de um *pop-up*.

Quando o Modal é exibido a página que esta por trás do mesmo fica com um tom escuto porém transparente, isto possibilita um foco maior no Modal. Ao Passar o mouse em algum item do Menu exibido uma função do JavaScript será disparada, *onMouseOver*, colocando uma sombra abaixo para destacar o item em questão. Estas pequenas animações contribuem com a fluidez do sistema, o tornando mais usual e convidativo para o usuário. A próxima página apresentada é a de exibição do Banco de Insumos, representada na Figura 14.

A Figura 14 ilustra a Página de gerenciado do Banco de Insumos. Nesta Página é possível consultar os Insumos cadastrados, editá-los e se necessário apagá-los. Os itens são exibidos utilizando o recurso de tabelas do Bootstrap. Diversos eventos existem ao usar estes recursos,

Figura 14 – Banco de Insumos



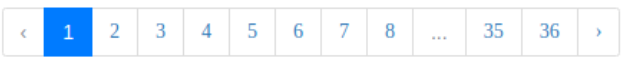
Fonte: Elaborada pelo autor.

como, por exemplo o evento, *onMouseOver* e disparado ao posicionar o mouse por cima de um item, assim é possível destacá-lo dos demais.

Os itens também se ajustam aos diferentes tamanhos de telas. Somente as principais informações são exibidas: o nome do Insumo, o Tipo do Insumo, o Preço, a Unidade e a Data de realização do cadastro. Na coluna “Ações” é possível realizar a edição do Insumo, clicando no ícone representado com um lápis e a visualização completa dos dados do Insumo, sendo necessário clicar no ícone representado por um desenho de um olho. Ao passar o mouse por cima do ícone uma pequena mensagem será exibida para orientar o usuário sobre o que será exibido caso o mesmo clique no botão.

Outro recurso implementando com a utilização do Laravel foi o de paginação. Neste caso, o sistema exibe os itens incrementando de 50 em 50. Isto otimiza a tempo gasto para carregar a página, uma vez que podem haver muitos registros no Banco de Dados. A Figura 15 demonstra o recurso de paginação que o Laravel disponibiliza.

Figura 15 – Paginação com o Laravel



Fonte: Elaborada pelo autor.

Neste caso, existem mais de 2000 registros de Insumos cadastrados. As páginas são exibidas no rodapé da mesma, os insumos são ordenados por ordem alfabéticas afim de facilitar a consulta. Seguindo com a apresentação a Figura 16 ilustra a edição do Insumo.

Figura 16 – Edição de Insumos

Fonte: Elaborada pelo autor.

A edição do Insumo é simplificada. Nela é possível alterar os itens mostrados na Figura 16. Quando o usuário clicar em “Salvar” os dados inseridos no formulário serão submetidos para a tabela de Insumos. A data e hora de alteração será preenchida para futuras auditorias.

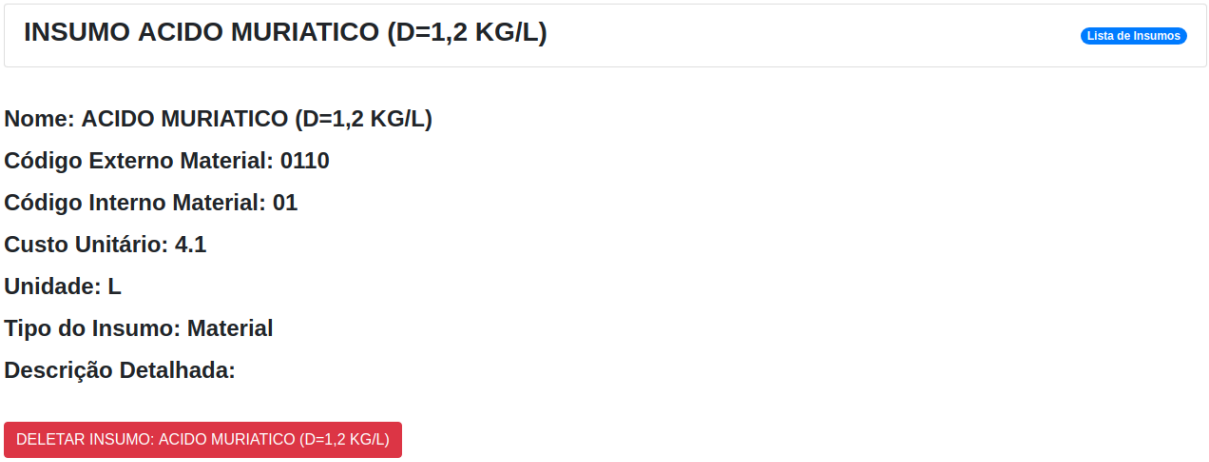
O Fluxo fica assim: o Laravel pede para um Controller a Pagina de edição com os dados do Insumo “X”; o Controller solicita os dados do Insumo “X” para o *Model*; o Model encaminha os dados para o *Controller* e por fim o Controller encaminha a página de edição com o dados para a *View* que será exibida para o usuário. A próxima página envolvendo os Insumos é representada na Figura 17.

Esta Figura ilustra a Página que possui duas funções: a função de visualizar o Insumo de maneira detalhada e a de deletar o mesmo. Ao clicar em “Deletar Insumo” uma caixa de diálogo Modal será exibida perguntando se o usuário realmente deseja apagar aquele Insumo. Caso afirmativo o mesmo será excluído do Banco de Dados. É importante ressaltar que somente os Insumos personalizados poderão ser excluídos.

Deste modo entende-se que todo Insumo que for cadastro no sistema é personalizado. Os Insumos da Base de Dados da Agetop não podem ser editados ou excluídos. Para esta finalidade foi implementada uma lógica de negócio na tabela Insumos. Nesta Tabela existe uma coluna denominada “TipoInsumo” que representa o tipo do Insumo, que pode ser personalizado e não personalizado. Ao Cadastrar um novo Insumo o mesmo receberá o “TipoInsumo” com o valor “1”. Assim, somente os Insumos que possuem este valor é exibido o ícone de edição demonstrado na Figura 14.

Páginas bastante similares são exibidas para o Banco de Composições. As Composições

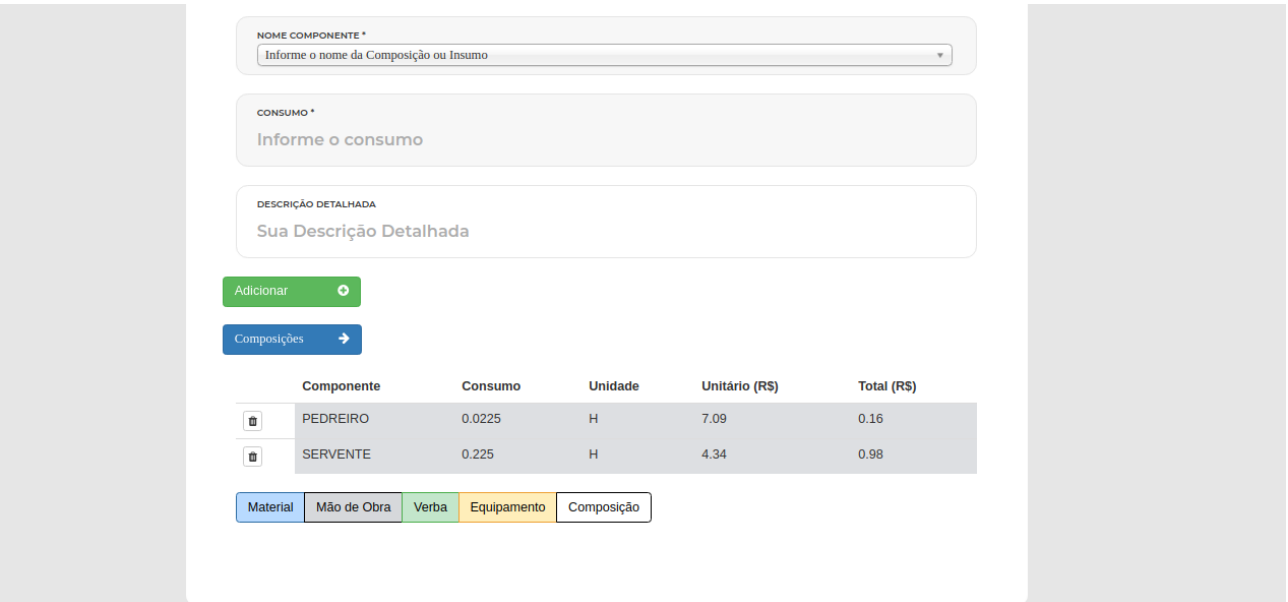
Figura 17 – Excluir Insumos



Fonte: Elaborada pelo autor.

seguem as mesmas lógicas e regras que os Insumos. É possível visualizar uma Composição, editar e excluir a mesma. Na Tabela de Composições também existe a coluna “TipoComposicao” que segue a regra descrita acima. Uma particularidade das Composições é que ela possui uma relação com os Insumos. Ou seja, uma Composição contem Insumos ou mesmos outras Composições. Assim, é possível excluir partes dos componentes que uma compõe uma dada Composição. A Figura 18 ilustra a edição/exclusão destes componentes da Composição.

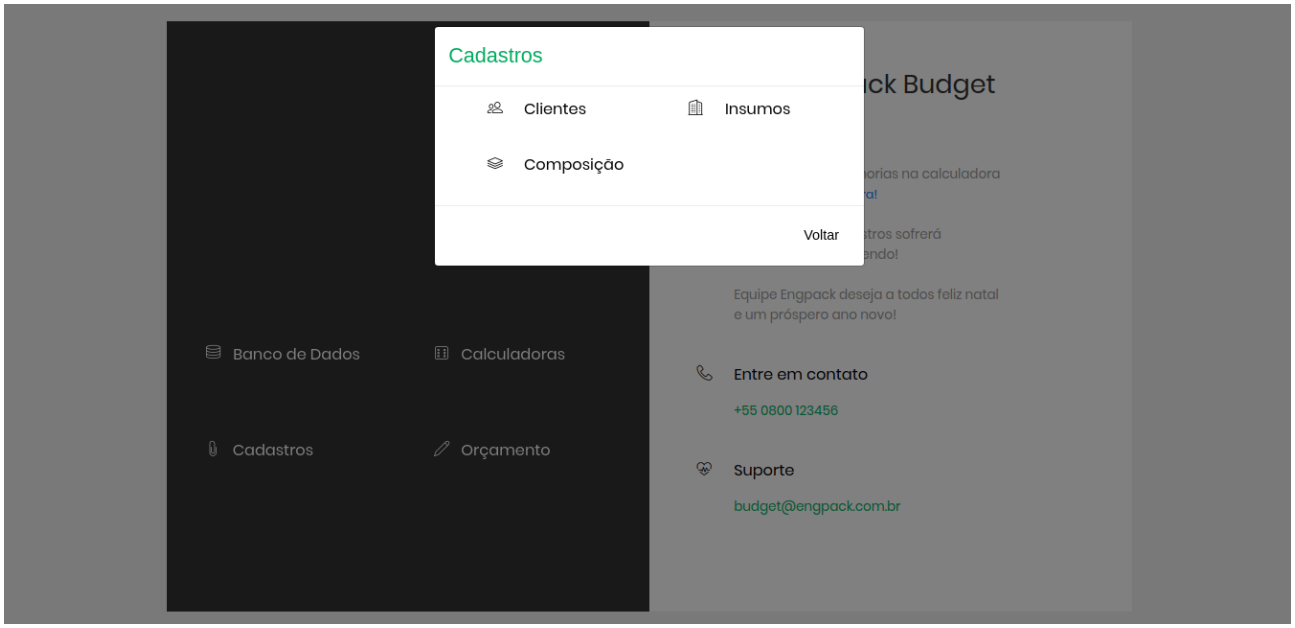
Figura 18 – Edição/Exclusão dos Componentes da Composição



Fonte: Elaborada pelo autor.

É possível observar uma legenda abaixo da lista de componentes da Composição. Cada tipo de componente recebe uma determinada cor afim de orientar o usuário sobre quais Insumos ou Composições compõe uma determinada Composição. Seguindo para o menu de Cadastros, a Figura 19 exibe quais os cadastros que são possíveis realizar na segunda versão do EngPack Budget.

Figura 19 – Menu de Cadastros



Fonte: Elaborada pelo autor.

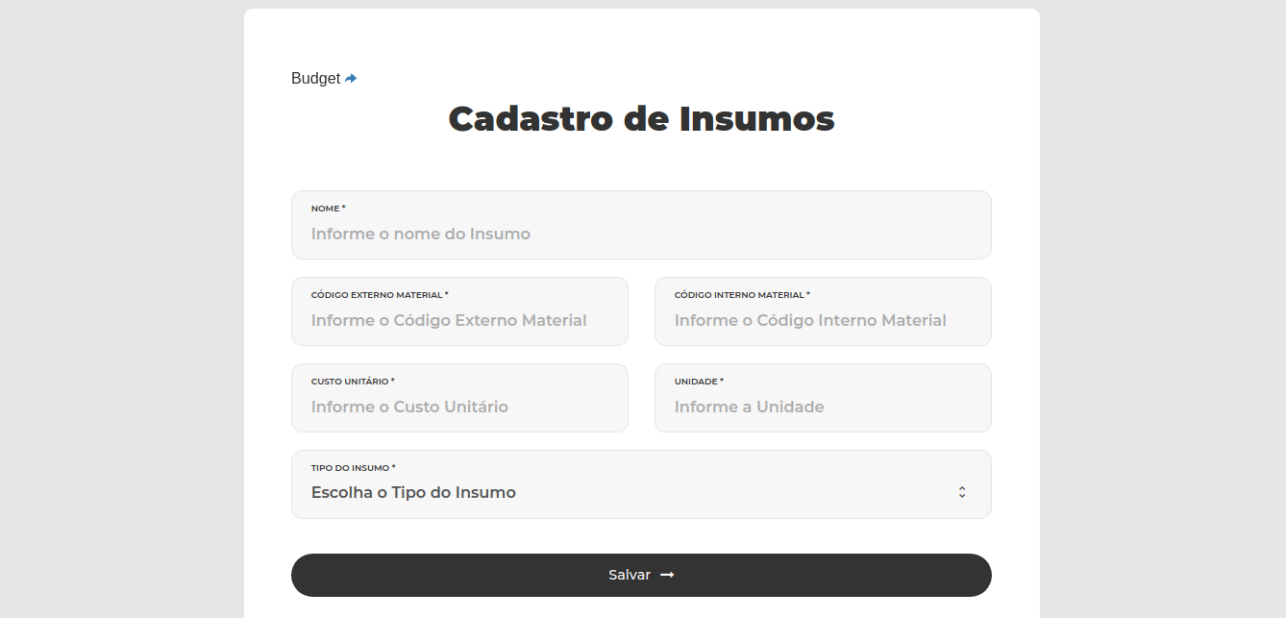
Seguindo a ideia de Modais ao clicar em Cadastros o Modal da Figura 19 é exibido. O comportamento dos eventos mencionados na Figura 15 são mantidos, isto é importante para o sistema ser coeso. É possível observar que existem três cadastros bases: o de Clientes, necessário para realizar um orçamento, o de Insumos e o de Composições. A Figura 20 Ilustra o Cadastro de Clientes:

Figura 20 – Cadastro de Clientes

Fonte: Elaborada pelo autor.

A Figura 20 representa o cadastro de Clientes. O cliente é fundamental para a realização de um orçamento. Pois o orçamento será centrado no mesmo. O Cadastro é relativamente simples, bastando o preenchimento dos dados e posteriormente clicar no botão “Salvar”. As fontes são um pouco maiores afim de possibilitar uma melhor acessibilidade na utilização do sistema. Os campos obrigatórios são grafados em vermelho caso os mesmos não sejam informados. Além disto é possível observar o “*”, indicando que este campo é de preenchimento obrigatório. Outro cadastro simplificado é o de Insumos representado na Figura 21.

Figura 21 – Cadastro de Insumos



Fonte: Elaborada pelo autor.

Segue a mesma logica descrita na Figura 20. Após informar os campos obrigatórios o Cadastro será realizado. Vale ressaltar que os insumos são devidamente tipados (rotulados) entre Material, Mão de Obra, Verba ou Equipamento, para quando for inserido em alguma composição ter o preço estritamente separado por tipo, tendo assim um maior controle do orçamento. Outro aspecto é que os Insumos são exclusivamente compostos em uma Composição. Deste modo, não é possível realizar um orçamento com Insumos. O último item do Menu de Cadastro é o Cadastro de Composições que será apresentado na Figura 22.

Figura 22 – Cadastro de Composições

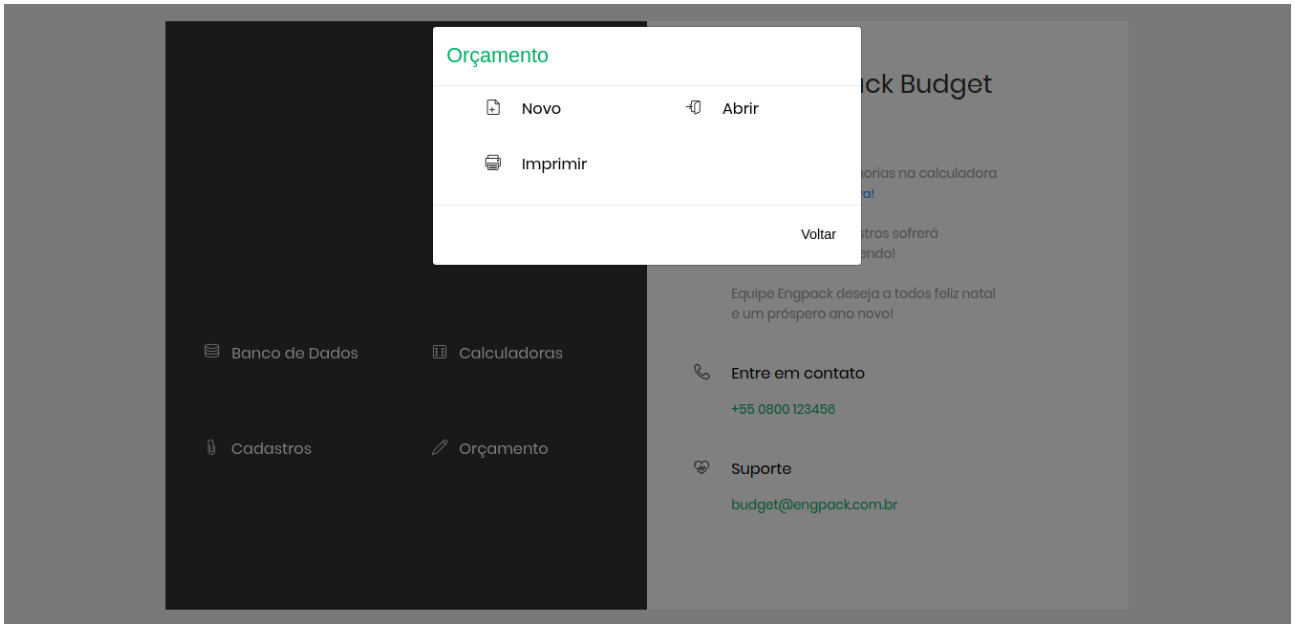
Fonte: Elaborada pelo autor.

Uma composição é um conglomerado de insumos e/ou outras composições. Logo é necessário um cadastro prévio dos insumos e das composições que farão parte da composição que está sendo cadastrada. Uma vez cadastros será necessário informar os campos da Figura 22. A maior parte destes campos são de preenchimento obrigatório, assim o sistema irá validar se estas informações foram preenchidas para concluir o cadastro.

O item “Nome Componente” será exibido a lista com os Insumos e Composições Cadastradas. O usuário irá informar a descrição do componente desejado e o sistema irá filtrando aos poucos, de modo que se o usuário informar a palavra “aço” será exibido todos os componentes que possuem esta palavra.

Depois será necessário informar o consumo deste componente. Por fim o usuário irá clicar em adicionar, conforme demonstrado na Figura 18, para incluir o componente selecionado. O usuário poderá adicionar quantos componentes forem necessários. Ao Finalizar o usuário devera fechar a Página ou Clicar no botão “Composições” para ser direcionado para a Página de Banco de Composições. Seguindo com o fluxo, o próximo menu a ser apresentado é o de Orçamento demonstrado na Figura 23.

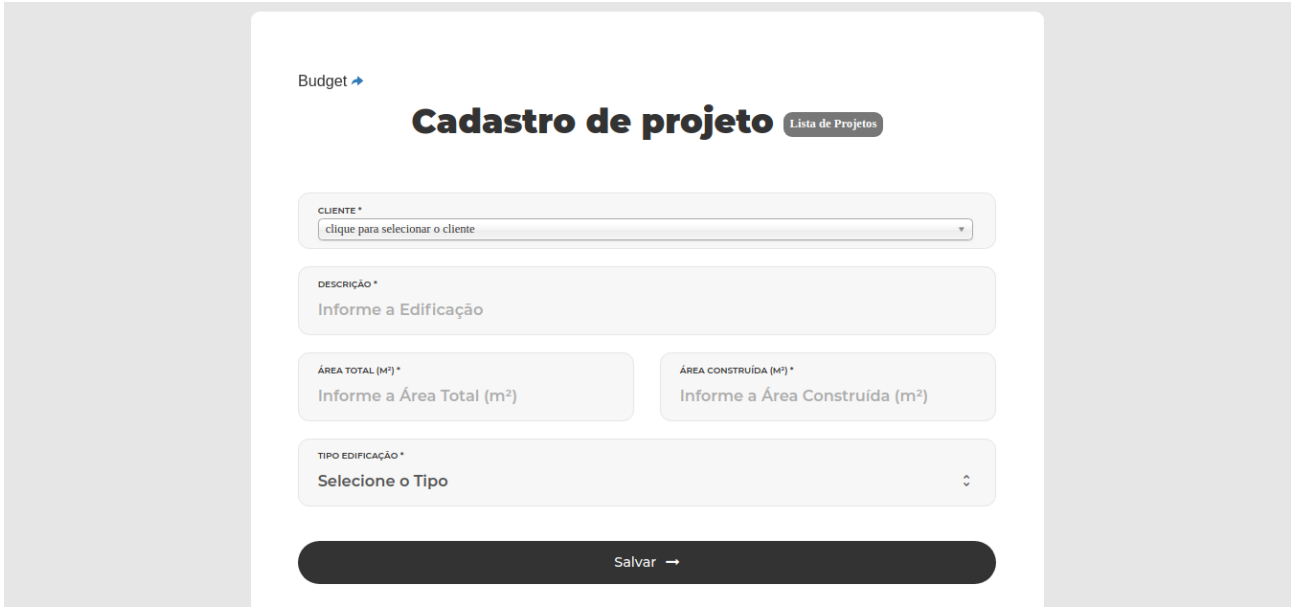
Figura 23 – Menu de Orçamento



Fonte: Elaborada pelo autor.

Neste menu são demonstradas os itens necessário para realizar um Orçamento. O item “Novo” irá cadastrar um novo projeto, o item “Abrir” irá listas todos os projetos que estão em desenvolvimentos e os que foram finalizados, por fim o item “imprimir” gera um relatório do orçamento desejado. Começando com o item “Novo”, representando na Figura 24.

Figura 24 – Cadastro de Projeto

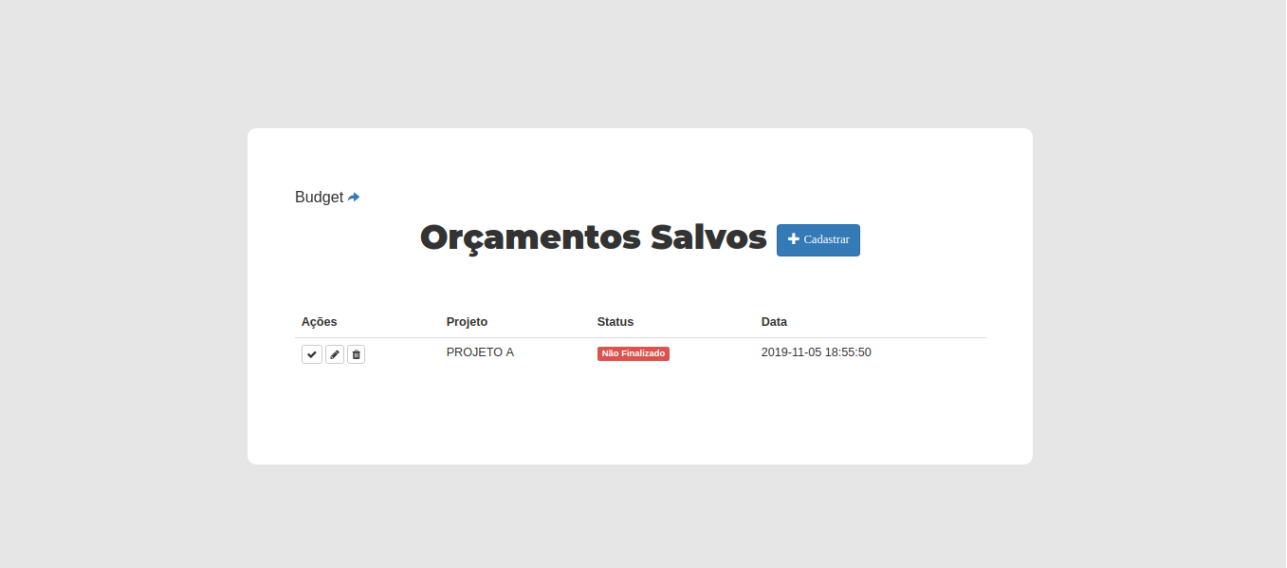


Fonte: Elaborada pelo autor.

O Cadastro do Projeto é necessário para um melhor controle. Assim deve ser informado

o cliente do projeto, a descrição, como, por exemplo “Projeto do José”, as áreas, o tipo da edificação que pode ser comercial, residencial ou ambos e por fim o endereço. Os campos obrigatórios são validados pelo sistema. É um cadastro relativamente simples. Uma vez cadastro o projeto o sistema irá se direcionar automaticamente para o item “Abrir” representando pela Figura 25.

Figura 25 – Item Abrir Projeto



Fonte: Elaborada pelo autor.

O Item Abrir Projeto lista todos os orçamentos em desenvolvimento, sejam eles concluídos ou não. É uma lista que segue as regras descritas na Figura 14. A coluna Projeto informa a descrição inserida no cadastro do projeto; A coluna Status informa a situação do projeto que pode ser “Finalizado” ou “Não Finalizado”; A coluna Data informa a data e hora em que o projeto foi cadastrado. A coluna ações as seguintes opções:

Finalizar Projeto: muda o status do projeto para Finalizado afim de obter um controle sobre quais orçamentos foram concluídos;

Editar Projeto: acessa a página de lançamento do orçamento. Nesta página será realizado o orçamento, ou seja, a inserção das composições, encargos e B.D.I;

Excluir Projeto: exclui o orçamento realizado. Uma caixa de dialogo será exibida solicitando a confirmação da exclusão.

Ao clicar em “Editar Projeto” o usuário será encaminhando para a página de lançamento do orçamento. Esta página é responsável por realizar os orçamentos. Aqui são agrupadas as Composições com os Insumos, o Cliente ao Projeto Cadastro, conforme é possível observar na Figura 26.

Figura 26 – Lançamento do Orçamento

Fonte: Elaborada pelo autor.

É possível observar na Figura 26 que a estrutura EAP do orçamento é organizada em níveis e subníveis. Assim é possível ter quantos forem necessários. Esses níveis são identificados por cores para facilitar a visualização, uma vez que um orçamento terá vários níveis. Esses níveis são descritos abaixo:

1. Custo: primeiro nível da hierarquia, tendo sua coloração em tons de azul e seu preço total correspondente à somatória dos níveis inferiores (etapa, subetapa e serviço);
2. Etapa: segundo nível da hierarquia, tendo sua coloração em tons de cinza e seu preço total correspondente à somatória dos níveis inferiores (subetapa e serviço);
3. Subetapa: terceiro nível da hierarquia, tendo sua coloração em tons de verde e seu preço total correspondente à somatória do nível inferior (serviço);
4. Composição ou Serviço: quarto nível da hierarquia, tendo sua coloração em tons de amarelo, sendo que somente neste nível que há inserção de valores. Os valores estão contidos nas composições inseridas. Assim, o valor total que é exibido na coluna “Total Item” é a somatória de todas as composições inseridas no orçamento.

É possível observar que a interface sofre uma mudança. Isto ocorreu pois o grupo decidiu optar por um padrão nas páginas especificas para realizar os lançamentos conforme identidade visual adotada nas outras ferramentas do produto.

Assim existe uma barra na lateral esquerda da página com algumas opções pertinente ao orçamento. Como, por exemplo, acessar o banco de Insumos e Composições, bastando para

isto clicar no ícone que representa os mesmos; acessar a lista de Projetos; acessar a página de Relatórios; Acessar as calcularas para auxiliar o orçamento.

No canto superior da Página existem algumas opções que são descritas abaixo:

Excluir: Exclui o projeto. Similar ao excluir projeto da Figura 25;

Incluir: Inclui um novo nível no lançamento do projeto;

Somar: Exibe um pequeno relatório com o somatório do projeto, tendo como base os valores lançados ate o momento;

B.D.I: Abre uma página pop-up para realizar o do B.D.I;

Encargos: Abre uma página *pop-up* para realizar o cálculo dos Encargos;

Algumas funcionalidades foram desenvolvidas para auxiliar o usuário na hora do lançamento do orçamento. Como, por exemplo, para incluir um novo níveis é possível fazê-lo de duas maneiras, clicando na opção "Incluir" localizado no canto superior da Página ou simplesmente clicando no ícone representado por uma seta direcionando para baixo, conforme é possível observar na Figura 27.

Figura 27 – Exemplo de inclusão de um nível

Fonte: Elaborada pelo autor.

Ao clicar no ícone descrito um modal como o da Figura 27 será exibido com o nível superior já informado. Assim basta o usuário informar a descrição do próximo nível. É importante ressaltar que esses níveis possuem uma hierarquia, deste modo o Nível 1, Custo, é o de maior hierarquia.

O Nível 1 recebe uma numeração, como, por exemplo, "1". O Nível 2 recebe duas numerações, citando caso parecido, "1.1". Caso o Nível 2 possua três itens a numeração destes ficaram respectivamente, "1.1", "1.2" e "1.3". O Nível 3 possuirá a numeração "1.1.1" e por fim

o Nível 4 a numeração “1.1.1.1”. Caso algum item for excluído a numeração seguirá a mesma regra.

É possível observar a possibilidade de editar e excluir os itens dos níveis. Deste modo é possível mudar qualquer nível de dependência. Como, por exemplo, supondo que em algum lançamento há dois Custos, de Nível 1, o Custo A e Custo B. O Custo A possui duas Etapas. Caso o usuário deseje colocar estas Etapas para o Custo B, basta o mesmo clicar no ícone de edição e um dialogo Modal como o da Figura 26 será exibido, assim o mesmo irá clicar em “Custo” e informar qual o novo custo destas Etapas.

Ao clicar no ícone representando por uma lixeira uma caixa de dialogo Modal será exibida perguntando se realmente o usuário deseja excluir este Nível. Ao excluir um nível de hierarquia maior todos os níveis abaixo dele serão excluídos. Então deve-se analisar de maneira cuidadosa. Na edição de um nível também é possível alterar a sua descrição.

5.1 *Migration*

Para armazenar os dados das estruturas apresentadas foi necessário realizar um estudo sobre o Banco de Dados da primeira versão. Nesta análise foi identificado alguns pontos negativos que foram discutidos no capítulo 4. Deste modo, optou-se por desenvolver uma estrutura completamente nova para tratar as falhas identificadas.

A segunda versão foi inteiramente desenvolvida com o *Framework* Laravel. Um recurso bastante útil do mesmo é a utilização de *migrations*. A documentação do Laravel (2019), no anexo A, diz que as *migrations* são como um controle de versão do Banco de Dados, permitindo que a equipe modifique e compartilhe de maneira simplificada a estrutura do mesmo. As migrations geralmente são emparelhadas como um construtor de esquema. Deste modo, é possível alterar facilmente a estrutura de um bando de dados independente do SGBD escolhido. A Figura 28 ilustra o exemplo de uma migration do Laravel.

Na figura é possível identificar a estrutura que uma *migration* possui. Basicamente a função necessita no nome da tabela que será criada, neste caso a de Insumos e posteriormente as colunas que a mesma possuirá. Ainda é possível observar que é necessário informar o tipo de dado que a coluna irá armazenar, seja uma string ou uma dado numérico e a quantidade máxima de valores que poderá ser guardado. Além disto, é possível informar um valor default, isto é, caso não seja informado nenhum valor o Laravel irá gravar no Banco de Dados o valor configurado como *default*. Também é possível realizar relacionamentos de tabelas.

Caso seja necessário acrescentar ou excluir alguma coluna em uma versão futura basta o desenvolvedor acrescentar ou remover os itens necessários na função exemplificada pela Figura 30. Para criar ou alterar a estrutura de uma tabela do sistema é necessário executar os comandos listado abaixo:

Figura 28 – Exemplo de uma Migration

```php
public function up()
{
    Schema::create('insumos', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->char('nomeInsumo',200);
        $table->char('codigoExterno',15);
        $table->char('codigoInterno',15);
        $table->double('custoUnitario',8,2);
        $table->char('unidade',10);
        $table->char('tipoInsumo',1);
        $table->longText('descricaoDetalhada')->nullable();
        $table->char('tipoBancoInsumo',1)->default("1"); //1 -> personalizado | 2 -> não
        personalizado
        $table->timestamps();
    });
}
```

Fonte: Elaborada pelo autor.

1. *migration::create*: cria as tabelas na ordem que o desenvolvedor estabeleceu.

2. *migration::refresh*: compara a estrutura do Banco de Dados existente com as definidas nas migrations. Caso haja divergência a estrutura é atualizada.

5.2 Diagrama de Entidade de Relacionamento

Toda a estrutura da segunda versão do sistema seguiu este conceito. Isto possibilita uma manutenção mais simplificada e torna o sistema flexível além de escalável. A Figura 29 ilustra a estrutura completa do Banco de Dados normalizado desenvolvida na segunda versão do sistema Engpack Budget.

A estrutura orçamentaria se inicia na tabela Clientes, conforme observa-se na Figura 29. Assim ela é a Tabela de maior hierarquia. Todo orçamento esta vinculado a um determinado cliente. Posteriormente, há a tabela de Projetos, que possui uma chave estrangeira da tabela clientes.

É possível identificar a chave estrangeira observando o losangolo avermelhado. Os itens com losango azulado são de preenchimento obrigatórios e os representado por uma pequena chave representam a coluna chave da tabela, conforme discutido no Capítulo 4.

Seguindo com a analise da Figura 29 é possível observar que existe uma chave estrangeira na Tabela Custo. Esta chave estrangeira refere-se a tabela Projetos. O uso de chave estrangeira é indispensável, dada a complexidade do sistema. Ao criar uma chave estrangeira em um SGBD relacional o mesmo cria índices para otimizar a consultas, inserção e deleção.

Isto contribui para o sistema possuir um alto desempenho, tendo em vista que estas tarefas são executadas diversas vezes. Assim, caso o usuário solicite para o sistema excluir um

projeto o SGBD saberá previamente quais itens pertencem a este projeto em função da existência das chaves estrangeiras. Evitando a realização de consultas desnecessárias. O mesmo vale no caso de inserção e edição dos dados.

Na tabela Composições é armazenada o cabeçalho de uma composição, contendo a descrição da mesma e o somatório dos itens que pertencem a esta composição. Os itens, como discutido, são insumos ou outras composições. Este tipo de relacionamento também é chamado de mestre-detalhe. As demais tabelas não possuem relacionamento pois são apenas cadastros para controle do sistema.

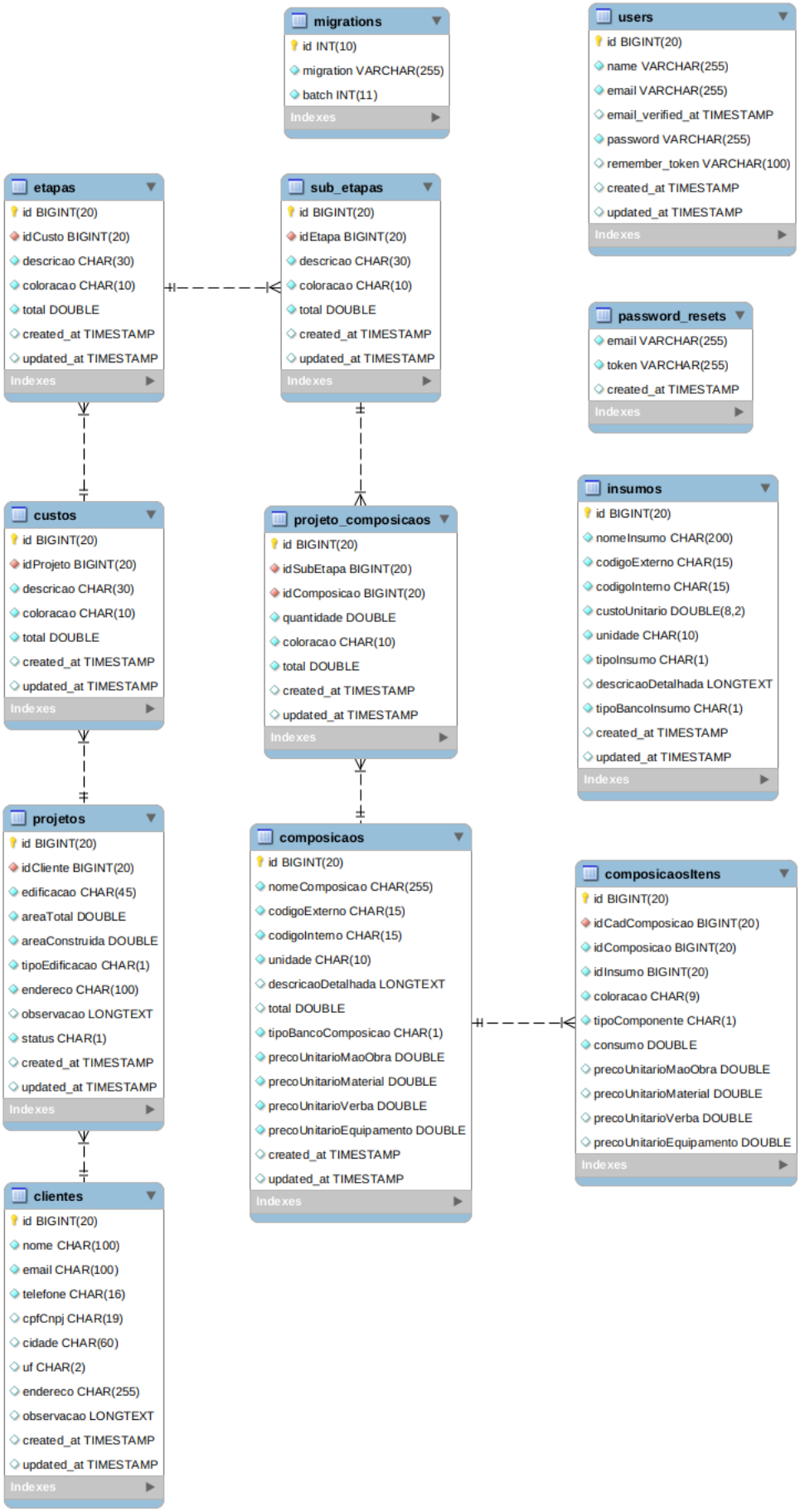
A tabela *Migration* armazena as *migration* definidas, conforme ilustrado na Figura 30. Assim o Laravel saberá quais as migrations devem ser executadas e em qual ordem. A ordem é importante pois em uma relação mestre-detalhe o mestre deverá ser criado antes do detalhe e nunca o inverso.

A estrutura apresentada na Figura 29 pode ser considerado como o "motor" do sistema. Sem esta estrutura nenhuma ação será realizada. É muito importante o planejamento correto desta estrutura e analisar com cuidado quais tabelas serão mestre e quais serão detalhes. Uma boa parte do tempo do desenvolvimento da segunda versão do sistema foi gasto no planejamento desta estrutura. Como mencionado, outro ponto positivo é que o Laravel é capaz de criar esta estrutura em qualquer SGBD. Deste modo, em uma atualização futura o aluno que for dar continuidade a esta aplicação não enfrentará grandes desafios para realizar esta troca.

5.2.1 Considerações Finais

Isto conclui a apresentação da segunda versão do Engpack Budget. Neste tópico foi discutido as páginas principais que o sistema possui e explicado as funcionalidades de cada uma. Foi exemplificado alguns lançamentos e discutido algumas técnicas de designer que foram aplicadas. O sistema seguiu com as tecnologias definidas pelo grupo. Isto foi importante tendo em vista a dimensão deste sistema, no futuro novos requisitos serão desenvolvidos e a mudança de versão não será tão custosa. Algumas funcionalidades serão desenvolvidas em uma versão futura. Estes itens serão discutidos no próximo Capítulo.

Figura 29 – Banco de Dados Engpack Budget 2.0



Fonte: Elaborada pelo autor.

CAPÍTULO

6

AVALIAÇÕES E RESULTADOS

Neste trabalho diversas técnicas foram estudas e aplicadas. O objetivo de desenvolvimento de uma ferramenta livre para ensino de orçamentação foi atingido. Em especial este software que promove uma sistematização do processo orçamentário, no âmbito da engenharia financeira que é um dos campos da Engenharia Civil. O sistema tem o intuito de promover o conhecimento através de um sistema gratuito de fácil acesso por ser desenvolvido para web, com isso aumentando ainda mais a acessibilidade.

6.1 Resultados e Metricas

Como ultima etapa do projeto foi necessário a analise das métricas, *Metric*, que foram levantadas na fase das perguntas. A Tabela 5 ilustra os resultados obtidos.

Tabela 5 – Analise das Métricas com Profissionais

Question	***Metric do Engpack Budget 1.0***	***Metric do Engpack Budget 2.0***
O Software atual demora quanto tempo para exibir um relatório?	300 segundos	6 segundos
Qual seria a dificuldade para realizar uma alteração no banco de dados?	Alta	Baixa
Qual seria a dificuldade para um novo desenvolvedor entender a função responsável por gravar o orçamento no banco de dados?	Média	Baixa
Caso fosse necessário usar outro SGBD seria possível realizar a troca?	Não	Sim
Quantas linguagens foram utilizadas para desenvolver o Front-End?	4	2
Quantas linguagens foram utilizadas para desenvolver o Back-End?	2	1

A Tabela 5 ilustra as principais Métricas obtidas com a segunda versão do sistema. Para responder essas questões foi necessário realizar diversos testes com as duas versões do sistema e posteriormente entrevistas com 5 profissionais das áreas relacionadas a Ciência da Computação que tinham domínio e experiencia com a linguagem de programação PHP e o *framework* Laravel. Com as entrevistas realizadas foi possível identificar a necessidade do emprego de padrões de projeto. Assim, o uso do *framework* Laravel MVC foi indispensável, tendo em vista que outros desenvolvedores iram dar suporte e manutenção neste sistema.

Para uma melhor compreensão da aceitação do sistema Engepack Budget foi levantado outras questões com 23 alunos do curso de Engenharia Civil da Universidade Federal de Goiás, regional de Catalão. As questão são referentes a funcionalidade do sistema. As *question* definidas foram enumeradas abaixo:

1. Qual a quantidade de estudantes que acharam Fácil?
2. Qual a quantidade de estudantes que acharam Médio?
3. Qual a quantidade de estudantes que acharam Difícl?

O teste foi realizado em um laboratório do curso de Engenharia Civil da Universidade Federal de Goiás e consistiu em solicitar aos participantes que realizassem funções de cadastros, edição e exclusão de dados. Por fim a realização de uma orçamentação para o levantamento de um muro. A Tabela 6 demostra os resultados obtidos com a primeira versão do sistema e a Tabela 7 os resultados com a segunda versão.

Tabela 6 – Analise das Métricas com Alunos Versão 1.0

Question	***Metric Fácil***	***Metric Médio***	***Metric Difícil***
Cadastrar Insumo	18	5	0
Alterar Insumo	19	4	0
Cadastrar Composição	16	5	2
Realizar Orçamento	21	1	1

Tabela 7 – Analise das Métricas com Alunos Versão 2.0

Question	***Metric Fácil***	***Metric Médio***	***Metric Difícil***
Cadastrar Insumo	22	1	0
Alterar Insumo	23	0	0
Cadastrar Composição	17	6	0
Realizar Orçamento	14	9	0

Outra questão avaliada foi com relação ao layout do sistema. Para tanto as *question* definidas são enumeradas abaixo:

1. O layout do sistema é péssimo?
2. O layout do sistema é muito ruim?
3. O layout do sistema é regular?
4. O layout do sistema é bom?
5. O layout do sistema é excelente?

Os resultados são demostrados nas Tabelas 8, resultados da primeira versão, e Tabela 9, resultados da segunda versão.

Tabela 8 – Analise das Métricas com Alunos Versão Layout 1.0

Question	***Metric Péssimo***	***Metric Muito Ruim***	***Metric Regular***	**Metric Bom**	**Metric Excelente**
Layout do sistema	0	0	1	9	16

Tabela 9 – Analise das Métricas com Alunos Versão Layout 2.0

Question	***Metric Péssimo***	***Metric Muito Ruim***	***Metric Regular***	**Metric Bom**	**Metric Excelente**
Layout do sistema	0	0	3	17	3

A ultima questão levantada nos testes é se o usário recomendaria o sistema. Assim a *question* para esta metrica foi:

1. Você recomendaria o sistema?

As Tabelas 10 e 11 apresentam os resultados obtidos na primeira e segunda versão do sistema:

Tabela 10 – Analise das Métricas Recomendação do sistema Versão 1.0

Question	***Metric Sim***	***Metric Não***
Você recomendaria o sistema?	23	0

Tabela 11 – Analise das Métricas Recomendação do sistema Versão 2.0

Question	***Metric Sim***	***Metric Não***
Você recomendaria o sistema?	23	0

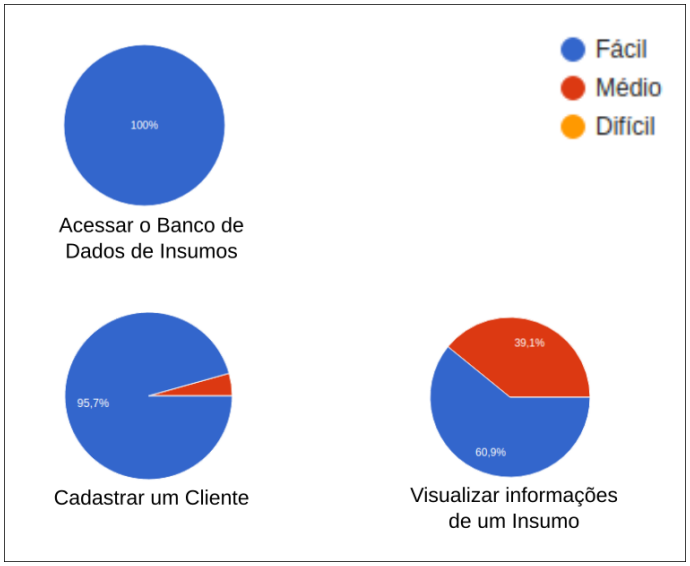
6.2 Considerações Finais

Algumas observações sobre melhorias no sistemas foram levantadas com os testes empregados e serão implementadas na próxima versão do sistema. Algumas sugestões feitas pelos futuros profissionais da Engenharia Civil são demonstradas abaixo:

1. "Fazer um login do cliente antes de qualquer atividade";
2. "Na tela inicial, para fazer a escolha dos itens: Banco de dados, Calculadoras, Cadastros e orçamentos, ao posicionar o mouse sobre os respectivos itens, seria interessante se aparecesse o símbolo de uma mão";
3. "Adicionar máscara de entrada nos textboxes";
4. "Adicionar ajuda em forma de balão ao passar mouse em cada textbox";
5. "Implementação de uma barra de pesquisa para facilitar a procura dos insumos cadastrados";
6. "A impressão do relatório poderia já ser gerada, ter a opção de ser gerada, já na tela de lançamento/criação do orçamento. Inserir a opção "Imprimir"nessa tela";
7. "Melhoria nas cores, visto que em algumas barras fica dificil a visualização dos links".

Essas melhorias serão tratadas como requisitos do sistema e deveram passar pelo processo descrito no Capítulo 2 a fim de serem desenvolvidas. A Figura 30 ilustra em forma de gráfico os resultados de outras métricas que foram definidas somente na versão 2.0:

Figura 30 – Métricas Exclusivas da Versão 2.0



Fonte: Elaborada pelo autor.

Apesar das melhorias necessárias ao observar as Tabelas e Gráficos é visto que, em concordância aos resultados, em sua maioria, os alunos que testaram o sistema não tiveram dificuldades. Muito pelo contrário, rotularam as atividades como "Fácil", dando com isso um respaldo positivo ao sistema.

Assim, foi possível concluir que este trabalho tornou melhor a segunda versão do sistema. Obtendo melhores resultados nos teste que foram aplicados além de ter um desempenho melhor que a primeira versão, levando em conta as tecnologias que foram utilizadas no seu desenvolvimento.

Deste modo, o sistema é um sucesso perante aos requisitos coletados. Então, é possível sua utilização na universidade, facilitando o estudo e qualquer tipo de contato com problemas orçamentários, sendo satisfatório para as premissas do mesmo. Vale a pena ressaltar, que o sistema aqui proposto não se restringe apenas para alunos, podendo ser utilizado por qualquer profissional do ramo, que necessite fazer algum tipo de orçamento onde o mesmo tem que ficar ciente das limitações do sistema.

Essas limitações serão tratadas em trabalhos futuros. A adição de novos recursos tornará a ferramenta ainda mais completa. Podendo ser implantada em órgãos com maiores estruturas. Para tanto, o desenvolvimento de um sistema de Login será indispensável. Contudo, como o sistema foi desenvolvido utilizando o *framework* Laravel MVC este requisito será de fácil implementação, visto que, a estrutura do projeto esta preparada para tratar acessos ao sistema, conforme é possível observar na Figura 31. Para tanto, a primeira versão do sistema foi necessária para ter ideias dos desafios que este tipo de sistema impõe ao desenvolvedor, levando em conta que é um sistema de complexidade elevada. Deste modo, a realização deste trabalho só foi possível graças aos requisitos e conhecimentos adquiridos com o desenvolvimento da primeira versão.

CAPÍTULO

7

CONCLUSÃO

Este projeto está inserido na Ciência da Computação, mais precisamente na área da Engenharia de Software, em parceria entre os cursos de Ciência da Computação e Engenharia Civil, bem como o domínio ao qual o projeto aplicado para controle de orçamentação na construção civil.

Através da análise e testes na primeira versão do EngPack Budget, foi possível detectar que a ferramenta não atendia aos novos requisitos, dependia de emuladores, apresentava lentidão a medida que os dados aumentavam, tinha pouca documentação e o sistema não era responsivo. Não houve o emprego de normalização no Banco de Dados ao qual contribuiu com a redundância das informações, diminuindo a integridade dos dados e consequentemente o desempenho do sistema.

A ferramenta orçamentária em ambiente Web denominada EngPack Budget 2.0, cumpriu sua função de facilitar a prática e ensino de orçamentação para alunos do curso de Engenharia Civil, com uso gratuito e disponível para acessar em vários dispositivos através da Internet. Deste modo, esta pesquisa possui um recurso inédito, uma vez que os softwares que existem para esta finalidade não foram desenvolvidos com tecnologias Web e possuem licenças com um alto custo. Sendo, em muitos casos, inviável a compra para o ensino de orçamentação para estudantes de Engenharia Civil.

Porém, precisa de melhorias futuras como um sistema de login e implementação das calculadoras de BDI e Encargos. O sistema de login consiste em um processo cujo o acesso ao sistema seja restrito, sendo necessário uma autenticação ou identificação do usuário. Assim possibilitará a agrupação dos orçamentos por usuários a fim de evitar que um terceiro tenha acesso aos dados que não os pertencem. Os insumos e composições inseridos no orçamento apresentando no Capítulo anterior não possui encargos fiscais. Para tanto, é necessário implementar as calculadoras de BDI e Encargos.

Contudo, após os testes realizados pelos alunos, obteve-se boas avaliações e concluísse

que a ferramenta está apta para implementação com as funções descritas no Capítulo anterior compostas por: página inicial, contendo informações sobre contatos e funcionalidade; menu de cadastros, possibilitando o cadastros de insumos, composições e clientes; menu de banco de dados, responsável por exibir e alterar as informações cadastradas; menu de orçamentos, fornecendo e agrupando os recursos necessários para a elaboração de orçamentação para construção civil.

CAPÍTULO

8

REFERÊNCIAS

AMARAL, G. A. E.; GUROV, D.; TRAVOSSOS, H. G. **Introdução à Engenharia de Software Experimental**. Universidade Federal do Rio de Janeiro, 2002. Citado ns paginas 30 e 32.

ÁVILA, A. and LIBRELOTTO, L. (2003). Ilha; lopes, **oc orçamentos de obras**. Florianópolis: Universidade do Sul de Santa Catarina-UNISUL. Citados na página 52, 53 e 58.

BASILI, R. V.;CALDIERA, G.; ROMBACH, D. H. **THE GOAL QUESTION METRIC APPROACH**, Institute for Advanced Computer Studies 1994. Citado na página 35.

BIOTTO, Clarissa Notariano; FORMOSO, Carlos Torres; ISATTO, Eduardo Luis. **Uso de modelagem 4D e Building Information Modeling na gestão de sistemas de produção em empreendimentos de construção**. Ambiente construído: revista da Associação Nacional de Tecnologia do Ambiente Construído. Porto Alegre. vol. 15, n. 2 (abr./jun. 2015), p. 79-96, 2015. Citados na página 53.

COSTA, C. G. A. d. et al. **Desenvolvimento e avaliação tecnológica de um sistema de prontuário eletrônico do paciente, baseado nos paradigmas da world wide web e da engenharia de software**. [sn], 2001. Citado nas páginas 40, 45, e 49.

CRUZES, D. S.; DYBÅ, T. Synthesizing evidence in software engineering research. In: ACM. **Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement**. [S.l.], 2010. p. 1. Citado na página 37.

FILHO, G. A. F. O. **Desenvolvimento de sistema gestor de apoio para geração e armazenamento de documentos: Gagad**. Universidade Federal de Goiás, Ciência da Computação, p. 69, 2016. Citado na página 29.

FREIRE, Juarez Francisco Junior, COSTA, Ricardo Vieira, PEREIRA, Wanderlei Malaquias Junior, MORAES, Matheus Henrique Morato de, FREITAS, João Eduardo Sousa de. (2018). **APLICAÇÃO DE UM SOFTWARE BIM NO PROCESSO DE LEVANTAMENTO**

DE QUANTITATIVOS. 4º CONPEEX: Universidade Federal de Goiás - Regional Catalão. Citados na página 54.

KERN, A. P., FORMIGA, A. d. S., and FORMOSO, C. T. (2004). **Considerações sobre o fluxo de informações entre os setores de orçamento e produção em empresas construtoras**. ENCONTRO NACIONAL DE TECNOLOGIA DO AMBIENTE CONSTRUÍDO, 10:591–601. Citado na página 55.

KERN, A.P.; FORMIGAS, A.S.; FORMOSO, C.T. **Considerações sobre o fluxo de informações entre os setores de orçamento e produção em empresas construtoras**. In: ENCONTRO NACIONAL DO AMBIENTE CONSTRUÍDO, 10., 2004, São Paulo. Anais. São Paulo: ANTAC. 2004. Citado na página 55.

Journal ISSN Publisher Date Volume Number Page URL URL(DOI) **Internet Systems Consortium Inc**. ISC Internet Domain Survey , 2008. Citado na página 35.

MARTINS, A. **Orçamentos de obras de edificações. Programa de Aperfeiçoamento Profissional**, CREA-PR. Goiânia, 2000. Citado na página 29.

MATTOS, A. D. (2010). **Planejamento e controle de obras**. Pini. Citado na páinga 55

MATTOS, Aldo Dórea. **Como preparar orçamentos de obras**. São Paulo: Pini, 2006. Citado na páinga 55

MUTTI, C. d. N. **Apostila da Disciplina Administração da Construção: ECV 5307-UFSC**. [S.l.]: Florianópolis, 2006. Citado na páginas 29.

MILLS, A. J.; DUREPOS, G.; WIEBE, E. **Encyclopedia of case study research: L-Z**; index. [S.l.]: Sage, 2010. v. 1. Citado na página 37.

Sampaio, F. M. (1991). **Orçamento e custo da construção. Hemus**. Citado na página 52.

SILVA, P. H. **EngPack Budget: Estudo e implementação de uma ferramenta de orçamento para construção civil**, Universidade Federal de Goiás, 2018. Citado nas páginas 30 e 32.

SOMMERVILLE, I. **Software Engineering, Boston, Massachusetts: Pearson Education**. [S.l.]: Inc, 2011. Citado na página 58.

SOMMERVILLE, I.; ARAKAKI, R.; MELNIKOFF, S. S. S. **Engenharia de software**. [S.l.]: Pearson Prentice Hall, 2008. Citado na página 37.

Wong, C.; DUREPOS, G.; WIEBE, E. **HTTP Pocket Reference**. O'Reilly, 2000. Citado na página 40.

GLOSSÁRIO

Framework: é uma abstração que une códigos comuns entre vários projetos de *software* provendo uma funcionalidade genérica. *Frameworks* são projetados com a intenção de facilitar o desenvolvimento de *software*, habilitando designers e programadores a gastarem mais tempo determinando as exigências do *software* do que com detalhes de baixo nível do sistema.

Padrões de projeto: ou *Design Pattern*, descreve uma solução geral reutilizável para um problema recorrente no desenvolvimento de sistemas de *software* orientados a objetos. Não é um código final, é uma descrição ou modelo de como resolver o problema do qual trata, que pode ser usada em muitas situações diferentes.

Template: é um documento sem conteúdo, com apenas a apresentação visual (apenas cabeçalhos por exemplo) e instruções sobre onde e qual tipo de conteúdo deve entrar a cada parcela da apresentação.

Web: Sinônimo mais conhecido de *World Wide Web* (WWW). É a interface gráfica da Internet que torna os serviços disponíveis totalmente transparentes para o usuário e ainda possibilita a manipulação multimídia da informação.

ANEXO

A

PÁGINAS INTERESSANTES NA INTERNET

<https://laravel.com/docs/6.x> Página em inglês contendo a documentação tecnica referente ao Framework Laravel MVC;

<https://fontawesome.com> Página em inglês contendo os icones utilizados no sistema apresentado;

<https://flatuicolors.com> Página em inglês contendo uma paleta de cores com foco no desenvolvimento voltados para *User Interface* e *User Experience*;

<https://getbootstrap.com> Página em inglês contendo a documentação tecnica referente ao Framework Bootstrap.