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

# Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário

**Marco Túlio Macedo Rodrigues**

Catalão – GO
2017

**Marco Túlio Macedo Rodrigues**

# Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário

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 Jabur
Coorientador: Prof. Dr. Thiago Jabur Bittar

Catalão – GO
2017

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

Rodrigues, Marco Túlio Macedo
Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário [manuscrito] / Marco Túlio Macedo Rodrigues. – 2017.
127 p.: il.

Orientadora: Prof.ª Dr.ª Luanna Lopes Lobato Jabur
Coorientador: Thiago Jabur Bittar
Monografia (Graduação) – Universidade Federal de Goiás, Unidade Acadêmica Especial de Biotecnologia, Ciências da Computação, 2017.
Bibliografia.

1. Engenharia de Software. 2. Larável. 3. UFG. 4. Reengenharia de Software. I. Jabur, Luanna Lopes Lobato, orient. II. Bittar, Thiago Jabur, coorient. III. Título.

CDU 004

**Marco Túlio Macedo Rodrigues**

# Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário

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

Trabalho aprovado em 01 de Setembro de 2017.

---

**Luanna Lopes Lobato Jabur**
Orientadora

---

**Thiago Jabur Bittar**
Universidade Federal de Goiás

---

**Liliane do Nascimento Vale**
Universidade Federal de Goiás

---

**Márcio Antônio Duarte**
Universidade Federal de Goiás

Catalão – GO
2017

*Esse trabalho é dedicado ao senhor Deus, criador de tudo e de todos, que permite e nos dá forças para superar desafios e dificuldades. Dedico esse trabalho a Universidade Federal de Goiás – Campus Catalão que recebeu um menino, aluno de ensino médio, e lhe transformou em um cientista. Espero que esse trabalho contribua para o seu funcionamento e lhe ajude em suas rotinas administrativas.*

# AGRADECIMENTOS

Muitos tentam, porém, poucos conseguem chegar em seus objetivos. Mas o mérito não é de quem consegue alcança-los e sim de Deus que dá força, sabedoria, inteligência e perseverança para podermos lutar por eles. Tendo sempre esse pensamento em mente, deixo claro, meus primeiros agradecimentos ao senhor Deus, que sempre cuidou de mim por toda minha jornada durante a graduação e que continua cuidando. Mesmo não sendo merecedor ele me presenteia com sua graça e sempre me dá forças para alcançar meus objetivos.

A dona Celina Macedo, minha mãe guerreira e meu pai, Abadio Carlos. Deixo aqui registrado meus mais sinceros agradecimentos também. Pessoas guerreiras que nunca mediram esforços físicos ou financeiro para me manter em uma terra distante. São pessoas que senão fosse por elas, esse trabalho não teria acontecido.

O apoio da família sempre é importante, minha avó Dorcelina Macedo e meu avô Benedito Macedo, gostaria de agradece-los pelo apoio que me deram nessa caminhada. Em especial ao meu avô, que me acompanhou na fase inicial da graduação e me ajudou com a mudança numa cidade, onde não tinha nenhum parente ou conhecidos.

Deixo registrado os mais sinceros agradecimentos ao melhor casal de doutores que conheço, Dr.ª Luanna Lopes Lobato e Dr. Thiago Jabur Bittar. Esses que foram peças fundamentais nessa pesquisa. Atuando como minha orientadora e com grande conhecimento na área de engenharia de software (ES) ajudou muito com a parte metodológica e teórica da pesquisa, muito obrigado Luanna. O Dr. Thiago Jabur ajudou muito trabalhando como cliente do sistema, co-orientador e com seu grande conhecimento na área de ES como foco na usabilidade contribuiu muito para a pesquisa. Gostaria de agradece-lo por ter me recebido tantas vezes em sua sala, repetidamente, mesmo com muitas outras atividades para fazer. Muito obrigado também, Thiago.

Ao pessoal da Coordenação de Assuntos da Comunidade (CCOM) que sempre foram presentes para sanar dúvidas, reuniões e dispuseram de tempo para estar contribuindo com a pesquisa. Devido toda a sua dedicação, obrigado CCOM.

*“Buscai, assim, em primeiro lugar,*
*o Reino de Deus e a sua justiça,*
*e todas essas coisas vos serão acrescentadas.”*
*(Mateus 6:33)*

# RESUMO

RODRIGUES, M. T. M.. **Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário**. 2017. 127 p. Monografia (Graduação) – Unidade Acadêmica Especial de Biotecnologia, Universidade Federal de Goiás – Regional Catalão, Catalão – GO.

Este trabalho é uma reformulação de software web, com foco em melhorar a usabilidade do programa antigo e criar novas funcionalidades em sua versão final reformulada. O programa tem como objetivo fazer a fiscalização e controle dos alunos ao restaurante universitário dentro da Universidade Federal de Goiás – Regional Catalão. Em sua versão original o programa apresentava poucas funcionalidades e baixa usabilidade, problemas que este trabalho visou resolver.

**Palavras-chave:** Engenharia de Software, Larável, UFG, Reengenharia de Software.

# ABSTRACT

RODRIGUES, M. T. M.. **Reformulação de software Web como Ferramenta de fiscalização a Restaurante Universitário**. 2017. 127 p. Monografia (Graduação) – Unidade Acadêmica Especial de Biotecnologia, Universidade Federal de Goiás – Regional Catalão, Catalão – GO.

This work is a redesign of web software, focusing on improving the usability of the old program and creating new features in its final redesigned version. The program aims to supervise and control university students within the Federal University of Goiás - Regional Catalão. In its original version of the program presenting few features and low usability, problems that this work aimed to solve.

**Keywords:** Software Engineering, Larável, UFG, Software Reengineering.

# LISTA DE ILUSTRAÇÕES


Figura 1 – Fase de Concepção . . . . . 31
Figura 2 – Fase de Avaliacao . . . . . 33
Figura 3 – Fase de Avaliacao . . . . . 38
Figura 4 – Tela do aplicativo (imagem retirada da Play Store) . . . . . 39
Figura 5 – Modelo Clássico . . . . . 44
Figura 6 – Interatividade entre funcionalidades e técnicas da mineração . . . . . 47
Figura 7 – Funcionalidades em mineração de dados . . . . . 49
Figura 8 – Base de Dado, Fiscaliza-RU.v1 . . . . . 71
Figura 9 – Outros VCS . . . . . 73
Figura 10 – Git como sistema de versionamento . . . . . 74
Figura 11 – Relacionamento entre as bases . . . . . 75
Figura 12 – Página Inicial do Fiscaliza-RU . . . . . 78
Figura 13 – Pagina Inicial do Fiscaliza-RU (continuação) . . . . . 79
Figura 14 – Tela inicial, acesso ao Fiscaliza-RU . . . . . 80
Figura 15 – Tela de Cadastro/Edição de usuário . . . . . 80
Figura 16 – Tela de Cadastro/Edição de usuário . . . . . 81
Figura 17 – Tela de detalhes do aluno . . . . . 82
Figura 18 – Tela de detalhes do aluno (continuação) . . . . . 82
Figura 19 – Tela de detalhes do aluno (continuação) . . . . . 83
Figura 20 – Tela de Cursos . . . . . 84
Figura 21 – Tela de adicionar Cursos . . . . . 84
Figura 22 – Tela de Bolsas . . . . . 85
Figura 23 – Tela para vincular refeições à uma bolsa . . . . . 86
Figura 24 – Tela de tipos de refeições . . . . . 86
Figura 25 – Tela de refeições realizadas . . . . . 88
Figura 26 – Tela para importar refeições . . . . . 89
Figura 27 – Tela de refeições importadas . . . . . 90
Figura 28 – Modal para upload de planilha de refeições . . . . . 90
Figura 29 – Tela de ajuda para importação de refeições . . . . . 91
Figura 30 – Estatísticas de Importações . . . . . 92
Figura 31 – Relatório de Importação de alunos . . . . . 92
Figura 32 – Modal para upload de planilhas de alunos . . . . . 93
Figura 33 – Tela de Ajuda para Importação de alunos . . . . . 94

Figura 34 – Frequência dos alunos . . . . . 95
Figura 35 – Frequência dos alunos em gráfico . . . . . 96
Figura 36 – Relatório de alunos que nunca frequentaram . . . . . 97
Figura 37 – Alerta de um usuário que nunca apresentou consumo . . . . . 97
Figura 38 – Produtos . . . . . 98
Figura 39 – Novo Produto . . . . . 99
Figura 40 – Tela de Cardápios . . . . . 100
Figura 41 – Tela para novo cardápio . . . . . 101
Figura 42 – Tela para visualizar cardápio cadastrado . . . . . 101
Figura 43 – Página Pública de visualização do cardápio . . . . . 102
Figura 44 – Página Pública sem Cardápio . . . . . 102
Figura 45 – Menu Sobre . . . . . 103
Figura 46 – Relacionamentos na Model UsuariosRest . . . . . 104
Figura 47 – Tela Home do Fiscaliza-RU.v1 . . . . . 106
Figura 48 – Tela Home do Fiscaliza-RU, versão atual . . . . . 106
Figura 49 – Adicionar novo nsuário (Fiscaliza-RU.v1) . . . . . 107
Figura 50 – Adicionar novo nsuário (Fiscaliza-RU) . . . . . 107
Figura 51 – Relatório de importações (Fiscaliza-RU.v1)) . . . . . 108
Figura 52 – Relatório de importações (Fiscaliza-RU)) . . . . . 109
Figura 53 – Erros de importações (Fiscaliza-RU.v1) . . . . . 109
Figura 54 – Erros de importações (Fiscaliza-RU) . . . . . 110
Figura 55 – Código Fonte 1. Consulta sem Eloquent no Fiscaliza-RU.v1 . . . . . 111
Figura 56 – Código Fonte 2. Consulta utilizando Eloquent no Fiscaliza-RU . . . . . 111
Figura 57 – Modularização das restrições . . . . . 113
Figura 58 – Modulo de Importação, Fiscaliza-RU.v1 . . . . . 114
Figura 59 – Modulo de Importação, Fiscaliza-RU . . . . . 114
Figura 60 – Resultados da Pesquisa . . . . . 121

# LISTA DE TABELAS

Tabela 1 – Questões de Pesquisa . . . . . 55
Tabela 2 – Características identificadas (possíveis requisitos) - RI . . . . . 56
Tabela 3 – Características identificadas (possíveis requisitos) - RII . . . . . 57
Tabela 4 – Características identificadas – RIII * Todos os sub-menus. . . . . 58
Tabela 5 – Características identificadas – RVI * Todos os sub-menus . . . . . 59
Tabela 6 – Características identificadas – RV * Todos os sub-menus . . . . . 60
Tabela 7 – Características identificadas - RVI . . . . . 60
Tabela 8 – Características identificadas - RVII . . . . . 61
Tabela 9 – Características identificadas - RVIII . . . . . 62
Tabela 10 – Características identificadas - RIX . . . . . 63
Tabela 11 – Requisitos Funcionais . . . . . 66
Tabela 12 – Requisitos Não Funcionais . . . . . 67
Tabela 13 – Questões aplicadas ao usuário (primeira seção) . . . . . 118
Tabela 14 – Perguntas ao usuário (seção 2) . . . . . 119
Tabela 15 – Resultados Obtidos . . . . . 120

# SUMÁRIO


1 INTRODUÇÃO . . . . . 23
1.1 Justificativa . . . . . 24
1.2 Motivação . . . . . 25
1.3 Objetivos . . . . . 26
1.4 Declaração do Problema . . . . . 27
1.5 Contribuições do Trabalho . . . . . 28
1.6 Fora do Escopo . . . . . 28
1.7 Organização da Monografia . . . . . 29

2 METODOLOGIA DE PESQUISA . . . . . 31
2.1 Fase de Concepção . . . . . 31
*2.1.1 Fase de Avaliação* . . . . . *33*

3 TRABALHOS RELACIONADOS . . . . . 37
3.1 Trabalho 1: Utilização de software Web como ferramenta de apoio à fiscalização de Restaurante Universitário . . . . . 37
3.2 Trabalho 2: Cardápio RU – UFT . . . . . 38

4 ESTADO DA ARTE . . . . . 41
4.1 Desenvolvimento de Software . . . . . 41
4.2 Metodologia de Desenvolvimento de Software . . . . . 43
*4.2.1 Metodologia Tradicional* . . . . . *44*
*4.2.2 Agil* . . . . . *45*
4.3 Data Mining . . . . . 46
*4.3.1 Funcionalidades do Data Mining* . . . . . *46*
4.4 Reengenharia e Remodelagem . . . . . 49

5 LEVANTAMENTO DE DADOS . . . . . 53
5.1 Entrevista . . . . . 53
5.2 Restrições do Sistema . . . . . 63
5.3 Identificação de Requisitos . . . . . 64
*5.3.1 Requisitos Funcionais* . . . . . *65*
*5.3.2 Requisitos Não Funcionais* . . . . . *67*

6 INFORMAÇÕES TÉCNICAS . . . . . 69
6.1 Características técnicas do sistema . . . . . 69
*6.1.1 Base de desenvolvimento* . . . . . *69*
*6.1.2 Base de Produção* . . . . . *70*
6.2 Base de Dados . . . . . 70
6.3 Ambiente de Desenvolvimento . . . . . 72
*6.3.1 GIT* . . . . . *72*
*6.3.2 Utilização do GIT neste Trabalho* . . . . . *74*
*6.3.3 GITLab no Projeto* . . . . . *74*
6.4 Framework Laravel . . . . . 75

7 FISCALIZA-RU . . . . . 77
7.1 Tela Inicial (Home) . . . . . 78
7.2 Usuários . . . . . 79
*7.2.1 Acesso ao Sistema* . . . . . *79*
*7.2.2 Restaurante* . . . . . *80*
*7.2.3 Detalhes do Aluno* . . . . . *81*
7.3 Cursos . . . . . 83
7.4 Bolsas . . . . . 85
*7.4.1 Refeições* . . . . . *86*
7.5 Restaurante . . . . . 87
7.6 Importações . . . . . 87
*7.6.1 Importar Refeições* . . . . . *88*
*7.6.2 Estatísticas* . . . . . *91*
*7.6.3 Importar Alunos* . . . . . *91*
7.7 Relatórios . . . . . 94
*7.7.1 Frequência dos Alunos* . . . . . *94*
*7.7.2 Alunos com Frequência Mediana* . . . . . *95*
*7.7.3 Alunos que nunca frequentaram* . . . . . *96*
7.8 Alimentação . . . . . 98
*7.8.1 Produtos* . . . . . *98*
*7.8.2 Cardápio* . . . . . *99*
7.9 Sobre . . . . . 102
7.10 Remodelagem e Reengenharia . . . . . 103
*7.10.1 7.10.1 Estrutura, Linguagem e FrameWork* . . . . . *103*
*7.10.2 SoftDeletes* . . . . . *104*
*7.10.3 Home* . . . . . *105*
*7.10.4 Novos Usuários Reformulados* . . . . . *107*
*7.10.5 Importação Reformulada* . . . . . *108*
*7.10.6 Erros de Importação* . . . . . *108*

*7.10.7 Querys com Eloquent . . . . . 110*
*7.10.8 Optimização das importações . . . . . 112*
*7.10.8.1 Redução de código . . . . . 112*
*7.10.8.2 Redução de consultas ao BD . . . . . 113*

**8 AVALIAÇÃO DO SISTEMA . . . . . 117**
**8.1 Questionário . . . . . 117**
**8.2 Resultados Obtidos . . . . . 119**

**9 CONCLUSÃO . . . . . 123**
**9.1 Trabalho Futuro . . . . . 125**

**REFERÊNCIAS . . . . . 127**

CAPÍTULO

1

# INTRODUÇÃO

A Tecnologia da Informação (TI) tem evoluído consideravelmente nos últimos anos, levando a computação a ser utilizada em quase todas as áreas do conhecimento e setores, se mostrado essencial para otimização de diferentes atividades. Isso pode se dar devido ao crescimento da quantidade de informações a serem gerenciadas ou mesmo pela necessidade de automatização de processos, os quais antes eram feitos de maneira manual.

Assim, o uso de sistemas de informação tem crescido e a imersão das pessoas ao mundo digital tem sido feita de maneira natural, uma vez que a informática está cada dia mais presente no cotidiano das pessoas. Neste sentido, percebe-se que uma parcela considerável da população mundial tem acesso a dispositivos digitais e a sistemas de software para realização de suas atividades, o que impulsiona o mercado a inovar com sistemas relevantes, seguros e que se tornam essenciais. De acordo com Laudon e Laudon (1999), pode-se definir sistemas de informação como um conjunto de componentes inter-relacionados, trabalhando juntos para coletar, recuperar, processar, armazenar e distribuir informações com a finalidade de facilitar o planejamento, o controle, a coordenação, a análise e o processo decisório em organizações. Pressman (1995), a mais de décadas, afirmou que softwares podem ser aplicados a qualquer situação para especificar procedimentos. Dentre os diferentes usos da computação, pode-se citar a informatização nas Universidades, de modo a otimizar processos burocráticos e agilizar o gerenciamento das informações, de maneira segura e satisfatória.

Seguindo essa linha de pensamento, percebe-se que nas Universidades não é diferente, uma vez que os sistemas têm por objetivo facilitar a execução das atividades e gerenciamento das informações. Como apresentado por Laudon e Laudon, sistemas desenvolvidos para gerenciamento de informação e funcionalidades das Universidades já estavam, no ano de 1999, sendo cada vez mais implantados, possibilitando um meio de controle para gerar, armazenar e transferir a informação entre departamentos da Instituição (Laudon e Laudon,1999). Para tanto, este trabalho teve como objetivo uma pesquisa científica sobre Engenharia de Software, mais es-

pecificamente sobre a criação de sistemas de software, tendo como resultado o desenvolvimento de um sistema para o gerenciamento automatizado das informações do Restaurante Universitário (RU) da Universidade Federal de Goiás (UFG) – Regional Catalão (RC), chamado Fiscaliza-RU. O sistema é relevante visto que os relatórios do RU eram feitos de maneira manual e, ao fim do processo, alguns apresentavam inconsistências em relação à dados e valores duplicados.

## 1.1 Justificativa

A UFG – RC tem crescido estruturalmente no decorrer dos anos, tendo vários objetivos sendo alcançados, como a criação de novos cursos, contratação de mais docentes e técnicos, construção e ampliação da estrutura física, bem como melhorias por todo o campus. Especificamente, este trabalho está intrinsicamente relacionado à abertura do RU, voltado para a comunidade estudantil e externa.

O RU foi inaugurado na UFG – RC no dia 26 de outubro de 2015, sendo as refeições servidas aos alunos e a comunidade, com valores diferentes dependendo da classe em que o usuário do restaurante ocupa. Para o funcionamento do RU, desde o fornecimento da comida ao gerenciamento de equipamentos e do espaço local, foi aberta uma licitação para que uma empresa terceirizada assumisse as responsabilidades devidas.

De acordo com a Coordenação de Assuntos da Comunidade Universitária (CCOM), em 2015 um público de 3.204 alunos frequentou o restaurante, sendo servidas, diariamente, 1.200 refeições.

A empresa gestora do RU utilizou um software particular que se comunica com um sistema de catraca, que através de biometria dos alunos faz o controle de acesso dos usuários. Como a UFG arca com uma parcela dos custos das refeições dos alunos, o valor recebido pela empresa no final do mês é diretamente proporcional a quantidade de alunos que frequentou o restaurante em determinado período.

A UFG possui alunos que são divididos em três tipos: subsidiado, bolsista integral e bolsista parcial. Cada tipo de aluno apresenta custos diferentes à Universidade e, assim, precisam ser diferenciados no sistema visto que o auxílio dado a cada tipo tem valor diferente. O gerenciamento correto do número de alunos e de quais tipos é extremamente necessária pois é o que determina o valor pago pela UFG à empresa no final do mês.

Ao final de cada dia, quando a empresa terceira já possui o balancete de todas as refeições diárias servidas, é enviado um relatório para a CCOM com a relação de todos alunos que frequentaram o restaurante, sendo apresentados dados referentes a data e hora da refeição, nome e matrícula do aluno, tipo de bolsa do aluno (subsidiado, integral ou parcial). O problema deste processo é que os relatórios são fornecidos em planilhas de Excel, com possibilidade de serem editáveis por qualquer pessoa que tenha acesso as mesmas, podendo ocasionar enganos.

Assim, é importante ressaltar sobre a possibilidade de fraudes nos relatórios de frequência, a passível edição dos dados e os custos gerados à Universidade, uma vez que servidores da CCOM tinham que disporem de um tempo considerável para analisarem todos os relatórios recebidos, verificando os alunos manualmente.

A análise manual em dados volumosos pode acarretar em erros, passando despercebidos aos olhos dos servidores uma eminente inconsistência e o mesmo a não percepção de um erro. Durante verificações rotineiras, por parte da CCOM, observaram-se algumas inconsistências e valores duplicados nos relatórios, sendo que tais conflitos não podem ocorrer uma vez que a Universidade ou a empresa podem ser prejudicadas. Neste sentido, percebeu-se que o processo é, demasiadamente, demorado e propenso a erros, sendo inviável fazer todo esse processo diariamente de forma manual.

Para resolver esse problema foi desenvolvido originalmente em outra pesquisa o Fiscaliza-RU.v1, porém, esse sistema não atendia as demandas da CCOM e apresentava baixa usabilidade. Tendo em vista esses problemas o software nunca chegou a ser utilizado oficialmente pela coordenação e de forma a sanar esses problemas o Fiscaliza-RU.v1 foi reformulado gerando o Fiscaliza-RU.

## 1.2 Motivação

Uma vez que o gerenciamento do RU é feito de maneira manual, referente ao número de refeições e tipo de apoio que o aluno recebe da Universidade, o que impacta no valor final a ser pago pela UFG para a empresa que controla o RU, este trabalho traz como fator motivacional o desenvolvimento de um sistema para gerenciamento automático das refeições e dos bolsistas do RU.

Como descrito anteriormente, a fiscalização realizada manualmente pelos membros da CCOM pode ser falha, devido ao grande volume de refeições realizadas e à quantidade de regras a serem verificadas, tais como refeições fora de horário, refeições duplicatas, alunos em modalidades erradas.

Adicionalmente, o trabalho manual tende a ser mais complicado com a chegada de novos cursos e, consequente, aumento do número de alunos, uma vez que o número de refeições poderá aumentar e as regras de validações devem ser completamente respeitadas. Assim, presume-se que o número de refeições servidas diariamente será maior do que atualmente e o trabalho de controle possa seja prejudicado. Tendo em vistas os problemas identificados e, partindo do pressuposto de um pensamento visionário, viu-se a necessidade pela criação do software Fiscaliza-RU. Este, por sua vez, é software capaz de receber os relatórios fornecidos pela empresa e validá-los, automaticamente, de maneira rápida, eficiente, segura e consistente, gerindo as regras de frequência ao RU automaticamente.

## 1.3 Objetivos

Neste trabalho foi feita a reformulação de um software que atenda as demandas da CCOM em fiscalizar os relatórios de frequência dos alunos ao RU fornecidos pela empresa terceira contratada. O software Fiscaliza-RU foi, originalmente, desenvolvido por pesquisadores da UFG – RC, do curso de Ciência da Computação (CC), porém a pesquisa foi finalizada e o software ainda não atendia as demandas da CCOM, não sendo dessa forma instalado para sua efetiva utilização. O principal objetivo desta monografia foi desenvolver um sistema de software, chamado Fiscaliza-RU, que ofereça suporte gerencial e otimização na forma como as informações são gerenciadas no RU da UFG – RC. Assim, espera-se trazer mais agilidade e facilidade nos processos de gerenciamento, sejam estes referentes ao número de refeições a serem pagas à empresa contratada, de acordo com o auxílio oferecido pela UFG ao aluno, frequência dos bolsistas e subsidiados, bem como informações sobre os tipos de usuários que frequentam o RU, se estudantes, servidores ou comunidade externa. Além disso, o sistema oferece mais segurança e confiabilidade no processamento de informação.

Outro objetivo foi o de utilizar os dados coletados, por meio do sistema, sobre a frequência dos alunos ao RU para validar a assiduidade de bolsistas ao restaurante e verificar quais de fato estão utilizando o recurso destinado, auxiliando a CCOM a acompanhar os alunos bolsistas.

Adicionalmente, como objetivo, teve-se a construção de uma pesquisa mais detalhada sobre aspectos básicos da Engenharia de Software (ES), com o intuito de se ter como resultado não só o sistema de software desenvolvido, mas também uma documentação clara, definição de requisitos coerentes às necessidades impostas, o que leva, ao final, a um sistema relevante, que contemple os requisitos e licitados. Também foi considerado a reformulação do código e, consequente, melhorias neste, seja em relação a desempenho e clareza. Ainda, estudos foram feitos e incorporados ao sistema aspectos referentes a Banco de Dados (BD), como otimização de consultas, com o objetivo de tornar o sistema eficiente e seguro. O software foi re-desenvolvido de maneira a possibilitar construir uma base de dados rica em informações sobre os alunos que frequentam o restaurante. Neste sentido, pode-se trabalhar com a mineração desses dados, o que possibilitou gerar relatórios e gráficos sobre assiduidade dos alunos ao restaurante, gastos dos alunos e gastos da Universidade. Por meio deste trabalho, acredita-se em uma fiscalização mais ágil e precisa dos relatórios de frequência dos alunos e também numa forma adicional de acompanhamento mais detalhado dos bolsistas por parte da CCOM. Assim, este trabalho além de trazer como objetivo principal o desenvolvimento de um sistema para gerenciamento do RU, permitiu que mais funcionalidades fossem otimizadas, facilitando o trabalho das pessoas envolvidas. O projeto em sua proposta inicial tinha como objetivo apenas automatizar a fiscalização dos relatórios de refeições que eram disponibilizados diariamente. Contudo, observou-se que o sistema apresentava potencial para muito mais funcionalidades do que fazer apenas a fiscalização das planilhas. A base de dados montada ao realizar a importação das refeições continha informações sobre todos alunos que frequentavam o restaurante. Essa base

de informação abriu chance para implementação de vários relatórios, que por sua vez permitiu fazer o controle de frequência dos alunos ao restaurante. Portanto, ficou situado como objetivo de a pesquisa, durante a reformulação, possibilitar a fiscalização e realizar controle de assiduidade dos alunos ao RU, através de relatórios e gráficos.

## 1.4 Declaração do Problema

O desenvolvimento de software tem trazido diversos benefícios a diferentes setores e para as Universidades não seria diferente, visto que permite que tarefas antes executadas de maneira manual possam ser automatizadas.

Uma questão principal de pesquisa foi definida, como declaração do problema, para guiar o desenvolvimento deste, sendo ela: **Quais os principais desafios para o desenvolvimento de software voltado para o uso em uma Universidade?**.

De modo a responder esta pergunta algumas atividades e estudos foram definidos como passos a serem executados durante o desenvolvimento deste trabalho, os quais são:

- Revisão da literatura: para analisar a literatura existente sobre desenvolvimento de software e aplicação em sistemas institucionais. Ainda, estudos foram feitos sobre o desenvolvimento ágil, uma vez que esta abordagem foi utilizada nesta pesquisa e estudos sobre BD, de modo a identificar técnicas viáveis a serem aplicadas durante o desenvolvimento do sistema
- Reuniões dirigidas: foram feitas reuniões com os stakeholders a fim de identificar os requisitos necessários ao sistema, bem como as restrições que devem ser seguidas para desenvolvimento do mesmo,
- Reformulação: análise de um sistema já existente e estado do código, com vistas a identificar problemas e possíveis soluções para o novo sistema a ser desenvolvido;
- Desenvolvimento de sistema: desenvolvimento do Fiscaliza-RU;
- Pesquisa dirigida: aplicada aos usuários direto do sistema desenvolvido, com vistas a coletar melhorias para o software.

Uma segunda questão foi definida, como declaração do problema, para guiar o desenvolvimento desse: **Quais as principais funcionalidades que a CCOM necessita, para poder realizar o acompanhamento dos alunos?** .

De forma a resolve a pergunta imposta, foram realizadas reuniões com os stackholders, com o objetivo de descobrir a resposta para a mesma. Essas reuniões são detalhadas precisamente na seção 5 (Levantamento de Dados). Onde que, algumas ocorreram com próprios membros

da CCOM e outras ocorreram com pessoas com experiência na área de ES e que apresentavam facilidade em identificar possíveis novas funcionalidades e problemas de usabilidade

## 1.5 Contribuições do Trabalho

Este trabalho traz como contribuição a disponibilização de uma pesquisa sobre a área de ES e correlatas, de modo que sirva como lições aprendidas a outras pesquisas no mesmo domínio. Ainda, apresenta o desenvolvimento de um sistema de software, com uso de técnicas da computação e aspectos de BD.

De modo especial, uma contribuição está relacionada à sociedade, uma vez que o Fiscaliza-RU auxilia no controle e fiscalização do dinheiro público destinado a assistência estudantil no quesito de alimentação no RU. O valor que a UFG aplica em alimentação é alto devido à grande quantidade de alunos que realizam suas refeições nesse ambiente, portanto se as relações de frequência dos alunos apresentar inconsistências o dinheiro pago pela Universidade pode ser superior, implicando no gasto do dinheiro público de maneira equivocada.

Outra contribuição importante é o auxílio à CCOM, por meio do uso do Fiscaliza-RU, de modo a facilitar o acompanhamento dos alunos e identificar quais de fato utilizam do recurso para a alimentação e, assim, devem estar no programa de assistência estudantil.

## 1.6 Fora do Escopo

O Fiscaliza-RU disponibiliza ao administrador do sistema a possibilidade de cadastrar um cardápio semanalmente, através do painel de controle, e o mesmo é exibido em uma página externa de fácil acesso a todos os alunos da internet. No entanto, estão fora do escopo deste trabalho quaisquer outras atividades em relação ao cardápio, tais como avaliação da alimentação ou nível de satisfação dos alunos.

Ainda, esta pesquisa não tem como foco nenhum controle de caixa ou especificações financeiras, dessa forma, está fora do escopo a gestão de contas a receber ou de contas a pagar. Apesar do sistema realizar relatórios de gastos que os alunos tiveram e gastos que a UFG subsidiou para cada aluno, esses relatórios visam ser de cunho informativo para a CCOM realizar o acompanhamento estudantil dos alunos bolsistas. Atualmente os alunos frequentam o RU através de uma catraca que realiza autenticação via biometria. O Fiscaliza-RU não faz o controle de biometria e a comunicação com o sistema de catraca. O sistema de catraca é de responsabilidade da empresa contratada e a única comunicação entre a Universidade e a empresa é via relatórios diários com a relação de alunos que fizeram refeições.

Todos os relatórios gerados no Fiscaliza-RU são destinados apenas aos administradores do sistema previamente cadastrado. Alunos não tem acesso ao sistema, portanto é fora do escopo

desse trabalho fornecer qualquer tipo de acesso aos alunos à relatórios de frequência ou gastos que o sistema disponibiliza.

## 1.7 Organização da Monografia

Este trabalho está organizado como segue: No capítulo 1 foi apresentada a introdução, seguido da justificativa, motivação e objetivos deste trabalho. Ainda são apresentados a declaração do problema, as contribuições e as atividades que estão fora do escopo desta pesquisa.

No capítulo 2 é apresentada a metodologia de pesquisa utilizada para o desenvolvimento dessa monografia, quais métodos foram utilizados para fazer toda a pesquisa que levaram ao desenvolvimento do software. Durante o capítulo 3 são mostrados os trabalhos que apresentam relação com esta pesquisa. Nesse capítulo são destacados quais aspectos são relacionados, as diferenças e os problemas identificados. O capítulo 4 apresenta o estado da arte da pesquisa, sendo mostrados conceitos acerca da ES, mais especificamente sobre desenvolvimento de software, as principais metodologias de desenvolvimento utilizadas e a justificativa de quais foram utilizadas para desenvolvimento do Fiscaliza-RU. Adicionalmente, é mostrada uma breve descrição sobre Data Mining e aspectos de BD.

No capítulo 5 é explicado como foram efetuados os levantamentos de dados para o desenvolvimento do software Fiscaliza-RU. Para tanto, etapas da ES foram consideradas, desde reuniões, com os nomeados clientes, para identificação das necessidades e características do sistema até a implementação deste.

No capítulo 6 é apresentado como foi a preparação do ambiente de desenvolvimento, antes do início do desenvolvimento do código fonte do software. Uma considerável preparação do ambiente teve que ser feita para garantir modularidade no sistema, definição de documentação e suporte para que futuros desenvolvedores continuassem a desenvolver com qualidade o software. Muitas tecnologias modernas foram utilizadas e nesse capítulo é apresentado cada uma delas e como elas foram inseridas neste trabalho. Algumas dessas tecnologias foi o sistema de versionamento git utilizado para fazer o versionamento local e no servidor, outra tecnologia utilizada foi o paradigma de desenvolvimento Model View Controller (MVC).

Durante o capítulo 7 é apresentado o Fiscaliza-RU, com uma breve descrição de suas principais funcionalidades e como o mesmo se comporta. Após é apresentado o software, detalhando todas as funcionalidades que cada menu apresenta. Ainda, é apresentado onde são utilizadas as tecnologias e ferramentas, como Data Mining, Banco de Dados, Reengenharia e Remodelagem.

No capítulo 8 é mostrado como foi realizada a avaliação do sistema, para verificar a real qualidade do mesmo. Todos os testes e participações de usuários precisaram de um consentimento formal, por meio de um comitê de ética.

Por fim, no capítulo 9 são mostradas as conclusões, seguido de trabalhos futuros e lições aprendidas.

CAPÍTULO

2

# METODOLOGIA DE PESQUISA

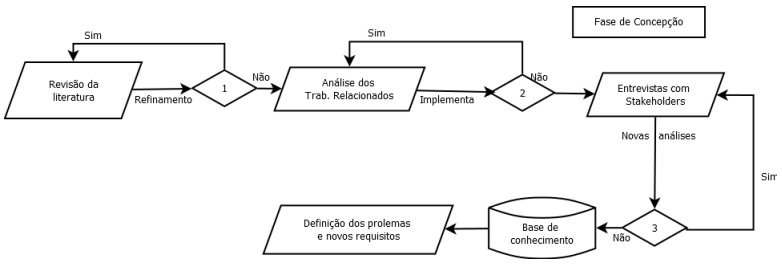
Esta pesquisa é inserida na área de Engenharia de Software, no contexto de projetos de desenvolvimento de sistemas. Para a formação deste trabalho foi utilizada uma metodologia dividida em duas fases, a fase de concepção e de avaliação, de acordo com uma pesquisa apresentada por Lobato (2002) e Spínola et al. (2008), as quais são descritas a seguir.

## 2.1 Fase de Concepção

Esta fase envolve a definição de alguns passos que foram realizados para desenvolver o trabalho, seguidos de análises sobre esses passos para conhecer os problemas a serem resolvidos por esta pesquisa.

Na figura 1 é apresentada a fase de concepção, que é composta por estudos empíricos realizados para fornecer conhecimento sobre os problemas no Fiscaliza-RU.v1. Estes estudos buscaram compreender, desde erros de implementação, de estruturação da arquitetura até a falta de funcionalidades extremamente necessárias ao correto funcionamento e gerenciamento, sendo estes últimos os novos requisitos que foram implementados no Fiscaliza-RU (sistema após a reformulação)

Figura 1 – Fase de Concepção




Fonte: Elaborada pelo autor.

A fase de concepção, apresentada na Figura 1, representa cinco principais passos, realizados para desenvolver esta pesquisa, sendo dividida em:

- Passo 1 – Revisão da Literatura: Neste passo foi feito uma revisão detalhada da literatura a fim de compreender melhor sobre alguns aspectos da ES, focando na criação de sistemas que utilizam a metodologia de desenvolvimento incremental adicionada à metodologia ágil. Uma breve descrição dos achados dessas áreas é apresentada na seção de "Levantamento de Dados";

- Passo 2 – Análise dos Trabalhos Relacionados: Tendo feito a revisão da literatura, foram selecionados alguns trabalhos relacionados a esta monografia, de modo que pudesse ser verificada a validade desta pesquisa, bem como as principais características que esta deveria acomodar. Neste caso, uma análise mais criteriosa foi feita sobre os requisitos que os sistemas relacionados apresentavam, de modo a identificar quais vantagens estes apresentavam e que seriam relevantes ao Fiscaliza-RU, bem como quais problemas deveriam ser evitados na reformulação do novo sistema;

- Passo 3 – Entrevistas com stakeholders: Após análise dos trabalhos da literatura, foram planejadas várias reuniões com os stakeholders envolvidos nesta pesquisa, sendo estes os possíveis usuários do sistema Fiscaliza-RU. Esta etapa foi feita de maneira incremental, ou seja, a cada necessidade verificada novas reuniões eram feitas para resolução de conflitos, validação do que estava sendo desenvolvido, bem como identificação de novos requisitos. O passo 1 foi extremamente necessário para que fossem aplicados, de maneira correta, os aspectos inerentes a metodologia de desenvolvimento incremental e ágil. Entrevistas foram aplicadas aos stakeholders para compreensão das falhas do sistema em sua primeira versão. Os dados gerados pelas entrevistas foram coletados para gerar a base de conhecimento, apresentada no próximo passo;

- Passo 4 – Base de Conhecimento: Este passo é resultado dos demais apresentados, uma vez que sua execução é feita baseada nos resultados obtidos anteriormente. Neste, uma base de conhecimento é formada, verificada e validada, de modo a ser utilizado o conhecimento válido e relevante para reformulação do Fiscaliza-RU.v1. Assim, este passo é importante para que o novo sistema, Fiscaliza-RU, apresente características de usabilidade, funcionalidades coerentes ao que é necessitado, performance eficiente, não causando aborrecimento nos usuários finais durante sua utilização, bem como a melhoria de sua eficácia, considerando aspectos de qualidade de software e garantindo atividades atômicas para o correto funcionamento do sistema;

- Passo 5 – Definição dos Problemas e Novos Requisitos: O último passo da fase de concepção é um resultado direto dos passos realizados anteriormente. Neste, são identificados os problemas a serem resolvidos, os quais são resultantes do Fiscaliza-RU.v1, e elicitados os

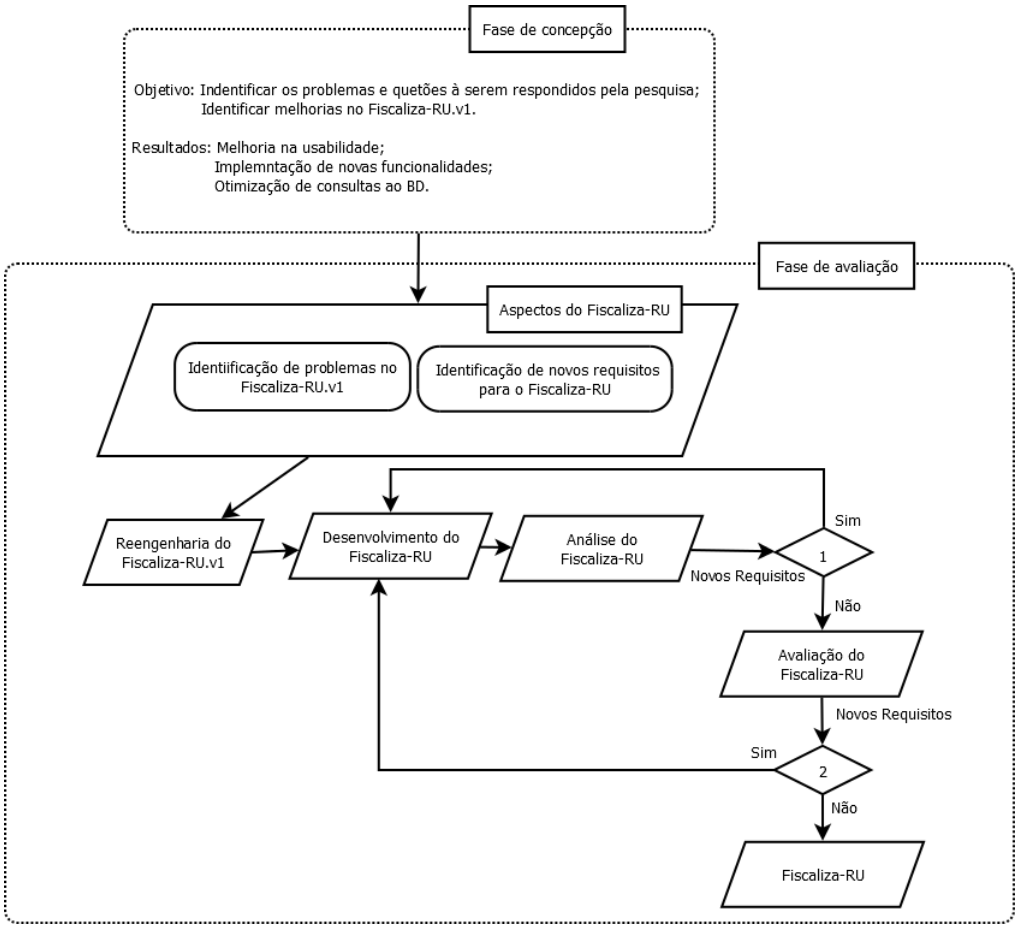
novos requisitos a serem implementados. Este passo se resume na definição de como será o Fiscaliza-RU, seus objetivos, características e funcionalidades finais.

Na próxima subseção é apresentado o detalhamento do desenvolvimento do sistema Ficasliza-RU, composto da reengenharia do código, remodelação da arquitetura, reestruturação do BD, bem como a criação de novos módulos compostos pelos requisitos e características identificados na fase 1.

### 2.1.1 Fase de Avaliação

Depois de concluir a fase de concepção, esta fase está relacionada ao desenvolvimento do sistema Fiscaliza-RU e avaliação do mesmo, conforme apresentado na Figura 2.

Figura 2 – Fase de Avaliacao




Fonte: Elaborada pelo autor.

Conforme apresentado na Figura 2, a ferramenta proposta foi desenvolvida com base nos resultados coletados na fase de concepção, onde foram observados dois pontos principais, a identificação de problemas no Fiscaliza-RU.v1 e a identificação de novas funcionalidades para o Fiscaliza-Ru.

A fase de avaliação é dividida em 6 passos principais, sendo eles:

- Passo 1 – Aspectos do Fiscaliza-RU: Este passo tem como entrada a saída da fase de concepção, sendo dividido em "Identificação de problemas no Fiscaliza-RU.v1" e "Identificação de novos requisitos para o Fiscaliza-RU". Assim, são verificados os problemas recorrentes da versão inicial e elicitados os novos requisitos para o sistema. É importante ressaltar que necessidade adicionais são identificadas neste passo, as quais também são adicionadas ao sistema como características deste;

- Passo 2 – Reengenharia do Fiscaliza-RU.v1: A reengenharia do sistema, em sua versão inicial, é feita com bastante detalhamento para que erros não sejam passados despercebidos e outros não sejam inseridos. Neste passo, o código fonte é analisado e depurado, a arquitetura é remodelada, o BD é reestruturado para dar início ao desenvolvimento do novo sistema;

- Passo 3 – Desenvolvimento do Fiscaliza-RU: Este passo é focado em atender o objetivo central desta pesquisa, que é o desenvolvimento de um sistema de software usável, eficiente e eficaz, considerando aspectos da qualidade de software para um produto final que seja satisfatório às necessidades dos usuários finais. As experiências coletadas ao longo do curso de Ciência da Computação, nos estágios realizados e no decorrer desta pesquisa foram essenciais para o desenvolvimento do Fiscaliza-RU e implementação de estratégias para melhor desempenho do sistema;

- Passo 4 – Análise do Fiscaliza-RU: Ao realizar a reengenharia do Fiscalia-RU.v1, aplicar as alterações e desenvolver o novo sistema, este foi submetido a análise dos usuários. Esta análise consistiu em verificações no Fiscaliza-RU a medida que ele era desenvolvido. Neste sentido, o sistema passava por intervenções incrementais dos stakeholders. Ao identificar novas demandas, o processo era voltado para a fase de desenvolvimento e o ciclo se repetiu até não existir novas achados;

- Passo 5 – Avaliação do Fiscaliza-RU: Após o sistema ser finalizado, todas as alterações sido feitas e as novas funcionalidades desenvolvidas, este passo consiste na avaliação do mesmo para verificação de aspectos relacionados a usabilidade, utilidade e relevância. Neste sentido, o Fiscaliza-RU foi submetido à uma avaliação pelos membros da CCOM, para verificação das funcionalidades, e análise por alguns alunos do curso de Ciência da Computação, para averiguação de aspectos de desempenho e implementação. O objetivo final deste passo se deve a quantificar a usabilidade e encontrar possíveis problemas ainda não descobertos. Ao encontrar novas demandas no sistema, o processo voltava para o passo anterior, passo de desenvolvimento, e o Fiscaliza-RU era novamente analisado. Quando novas demandas, relevantes e justificáveis, não foram mais encontradas nessa etapa o processo de desenvolvimento é encaminhado ao passo final;

- Passo 6 –Fiscaliza-RU: Este passo consiste na entrega do produto final, o sistema Fiscaliza-RU. Assim, concluiu-se que objetivo foi alcançado e que a reformulação do sistema foi concluída, estando o Fiscaliza-RU pronto para sua efetiva utilização pelos usuários da CCOM.

CAPÍTULO

3

# TRABALHOS RELACIONADOS

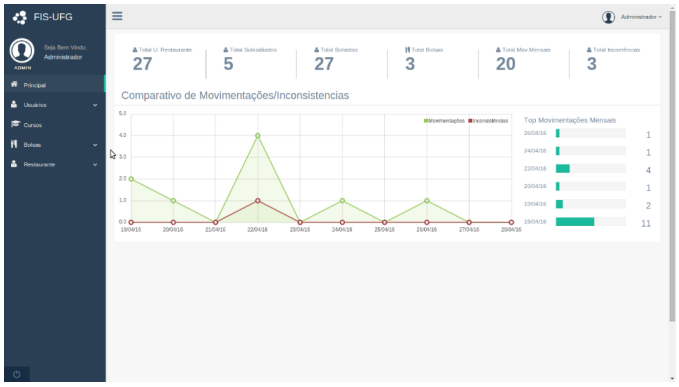
Neste capítulo são apresentados os trabalhos relacionados a esta pesquisa. Para tanto, outros softwares foram pesquisados com o intuito de identificar as qualidades destes e quais objetivos não foram atingidos. Dentre os trabalhos verificados, pôde-se observar que a maioria trata a relação ao desenvolvimento de sistemas com o objetivo de facilitar o gerenciamento de informações e, consequentemente, facilitar a manipulação destas.

## 3.1 Trabalho 1: Utilização de software Web como ferramenta de apoio à fiscalização de Restaurante Universitário

Esse trabalho é fruto de uma pesquisa desenvolvida na UFG – RC (Souza, 2016), o qual resultou na primeira versão do Fiscaliza-RU, aqui chamado como Fiscaliza-RU.v1 (versão 1). O trabalho teve por objetivo o desenvolvimento de um software capaz de realizar importações de refeições do RU, importar alunos, cadastrar curso, bolsas e administradores do sistema, sendo apresentada na Figura 3 a interface do Fiscaliza-RU.v1.

- Pontos Positivos: Apesar de apresentar poucas funcionalidades o sistema entregava o que havia sido planejado inicialmente, verificar inconsistências em relatórios. Toda base para o Fiscaliza-RU foi baseada na necessidade de verificar possíveis fraudulências nos relatórios do RU. O trabalho utilizou como base de desenvolvimento o Framework Laravel, em sua versão 5.1, banco de dados MySQL, em sua versão 5.5.49, hospedado em servidor Apache/2.4.10 (Debian), padrão MVC de desenvolvimento.
- Pontos Negativos: É possível notar que foram desenvolvidas poucas funcionalidades, sendo exibidas nos menus. O Fiscaliza-RU.v1 apresenta apenas cinco módulos, não sendo abordadas todas as necessidades atuais da CCOM e nem consideradas todas as restrições.

Figura 3 – Fase de Avaliacao




Fonte: Sousa (2016).

- Diferenças para esta pesquisa: Para o desenvolvimento dessa pesquisa o foco principal foi a usabilidade, a principal missão na reformulação foi tornar o programa útil ao usuário, de maneira que ele viesse auxiliar o usuário em suas atividades cotidianas. Outro diferencial é o desempenho das importações e a facilidade com que elas são realizadas. Após a reformulação do Fiscaliza-RU a forma de realizar as importações foi totalmente alterada, sendo disponibilizado ao usuário um tutorial de como realizar as importações com todas as informações necessárias para realizar um upload sem dificuldades.

## 3.2 Trabalho 2: Cardápio RU – UFT

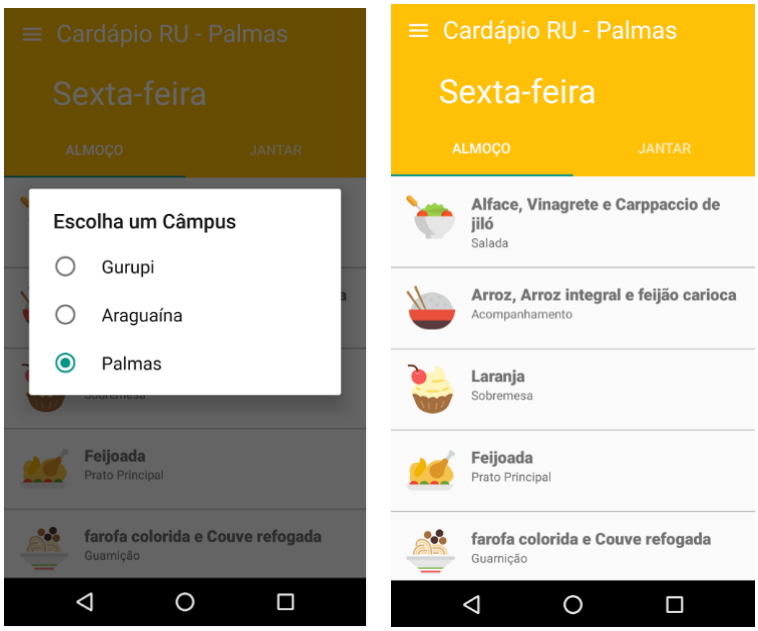
Esse trabalho foi desenvolvido pela Diretoria de Tecnologia da Informação (DTI) da Universidade Federal de Tocantins (UFT) tendo como objetivo apenas disponibilizar um cardápio online aos alunos do restaurante.

O cardápio online foi desenvolvido para o mercado mobile, mais especificamente para dispositivos Android estando disponível para download na loja da Play Store como Cardápio RU – UFT. Na Figura 4 são apresentadas duas telas do aplicativo.

Na figura é possível ver o momento em que o aplicativo abre um menu de seleção ao usuário, para que ele possa alterar entre câmpus da Universidade e visualizar o cardápio desejado. Adicionalmente, ao lado direito é apresentado o menu do aplicativo que mostra as refeições de um dia da semana. Cada categoria é representada por um ícone para facilitar o uso do aplicativo, por exemplo: a categoria sobremesa é representada por um ícone de doce, a categoria salada possui um ícone que é uma tigela com alface e tomate.

- Pontos Positivos: A UFT é dividida em três câmpus, sendo Palmas, Gurupi e Araguaína, tendo o cardápio sido desenvolvido para abranger todos eles. Ao entrar no aplicativo o usuário escolhe o câmpus cujo o qual deseja ver o cardápio do RU.

Figura 4 – Tela do aplicativo (imagem retirada da Play Store)




(a) Selecione um Campus (b) Home

- Pontos Negativos: O aplicativo é apenas de cunho informativo ao cardápio e não disponibiliza nenhuma outra opção ao usuário. Como aplicativo é voltado apenas para cardápio, uma possível avalição de satisfação dos alunos agregaria muito valor ao aplicativo, porém infelizmente isso não é disponibilizado no mesmo.

- Diferenças para esta pesquisa: Atualmente o Fiscaliza-RU não tem por objetivo ser um aplicativo e nem abranger todas as regionais da UFG, porém, ao realizar a pesquisa por softwares relacionados, este serviu como inspiração para um trabalho futuro, assim, espera-se que o Fiscaliza-RU possa evoluir de maneira atender toda a regional.

CAPÍTULO

# 4

# ESTADO DA ARTE

Como previamente mencionado, o mercado de desenvolvimento de software vem crescendo em decorrência da facilidade pelo uso de sistemas da computação, sendo disponibilizados sistemas informatizados que permitem um maior controle do negócio e ajudam o mesmo a se desenvolver, melhorando significativamente a execução de suas atividades e o encontro dos objetivos definidos.

A décadas atrás, o desenvolvimento de software já é visto como um fator importante para o gerenciamento e manipulação de dados importantes, sendo classificado como o ato de transformação de uma necessidade em um produto de software (Birrel e Ould, 1985).

Neste capítulo são apresentados alguns temas diretamente relacionados a área de ES, tendo sido estes conceitos relevantes para este trabalho e podendo ser úteis para pesquisas correlatas. Os conceitos aqui apresentados serão demasiados no livro de Sommerville (2013).

## 4.1 Desenvolvimento de Software

De acordo com Sommerville (2013) o mundo moderno não teria como existir sem os softwares. Os serviços e infraestruturas são gerenciados por sistemas computacionais e muito dos produtos eletroeletrônicos são constituídos de um computador e um software que os controlam.

Existem diferentes tipos de software, desde software simples para pequenos comércios locais até softwares robustos de range global, portanto não faz sentido buscar notações, métodos ou técnicas universais para a ES, uma vez que diferentes tipos de software exigem abordagens diferentes (Sommerville, 2013).

Infelizmente, ainda existem softwares sendo lançados ao mercado com erros e bugs que deveriam ser tratados e resolvidos. De acordo com Sommervile (2013), muitas dessas falhas são consequências de dois fatores:

- Aumento de demanda: Conforme novas técnicas auxiliam criar sistemas cada vez maiores e mais robustos, as demandas também aumentam. Os sistemas precisam ser construídos e entregues mais rápidos. Novos requisitos surgem e funcionalidades, que antes eram consideradas impossíveis, devem ser implementadas. Com a dificuldade em seguir e implantar os métodos tradicionais de desenvolvimento de software para diversificados cenários que surgiram, novas técnicas de ES precisaram ser desenvolvidas para atender a essas novas demandas.

- Expectativas baixas: Muitas empresas são forçadas a desenvolver software à medida que seus produtos e serviços aumentam, sem seguir nenhum método de desenvolvimento de software proposto em softwares, isso ocasiona em programas mais caros e menos confiáveis

Estes e tantos outros casos poderiam ser minimizados se um processo básico de ES fosse seguido. É evidente que diferentes pessoas podem escrever desenvolver software, sejam eles para hobby pessoal, simplificação de atividades rotineiras ou mesmo relacionado a aspectos profissionais. Neste último, espera-se que o sistema seja usado por usuários, sendo esses programas, geralmente, desenvolvidos por equipes profissionais treinadas e não por indivíduos. Neste sentido, é nítido que ainda há muito que ser desvendado na área de desenvolvimento de sistemas de software, mas já é evidente a necessidade da ES para construção de sistemas relevantes.

Segundo Somerville (2013) o objetivo da ES está mais focado em apoiar o desenvolvimento profissional de software por equipes treinadas do que a programação individual, já que é incentivado o uso por técnicas de apoio a especificação e evolução de programas, o que normalmente não é relevante em uma abordagem de desenvolvimento individual.

Muitas pessoas definem software como sinônimo para programa de computador. Porém, dentro da ES há um sentido mais profundo e importante, pois um software não envolve apenas um programa em si, mas toda a documentação e arquivos de configuração relacionados para que funcione corretamente.

O processo de software é um conjunto de atividades de resultados associados que levam à produção de um produto de software. Em contrapartida, modelo é uma representação abstrata de um processo de software. Cada modelo de processo representa um processo a partir de uma perspectiva particular, de uma maneira que proporciona apenas informações parciais sobre o processo (Sommerville, 2011).

Neste trabalho, todos foram considerados os aspectos trazidos pela ES para desenvolvimento de sistema de software, de modo que foi feito desde o planejamento inicial, encontro com stakeholders para identificação de requisitos, prototipação para a reengenharia do sistema, definição da arquitetura e BD, bem como construção de interface amigável que considere questões de usabilidade, codificação do sistema e sua documentação.

## 4.2 Metodologia de Desenvolvimento de Software

Segundo (KOSCIANSKI, 1985), diversas metodologias de desenvolvimento foram criadas para sistematizar a criação de software. Essas metodologias podem ser divididas em tradicionais ou ágeis. A metodologia tradicional enfatiza a documentação de cada passo do desenvolvimento do software, enquanto a metodologia ágil considera um novo paradigma de desenvolvimento de software, prometendo melhorias na produção e qualidade do software.

Metodologia de desenvolvimento de software pode ser definida como um conjunto de atividades que auxiliam a produção de software (Koscianski, 2007). Embora várias metodologias tenham sido criadas, existem algumas atividades fundamentais comum a todas elas (SOMMERVILLE, 2013)).

Como previamente apresentado, o Fiscaliza-RU seguiu processos definidos nos fundamentos da ES, onde todas as etapas seguidas foram planejadas de maneira sistematizada, baseadas na realização de atividade que levem ao atendimento dos requisitos identificados e nas restrições importas, para, assim, permitir o desenvolvimento de um sistema relevante ao propósito definido, as etapas seguidas foram:

- **Especificação:** Definição das funcionalidades (requisitos) e demais características do produto. Essa fase é bastante explorada durante conversas da equipe desenvolvedora, ou seu representante, com o cliente para definir as características do novo software.
- **Projeto e Implementação:** O software é produzido de acordo com as especificações. Nesta fase são propostos modelos por meio de diagramas que serão implementados em alguma linguagem de programação.
- **Validação:** Atividades de revisão e testes visando assegurar que os requisitos sejam cumpridos.
- **Evolução:** Atividades de manutenção no sistema desenvolvido, como por exemplo, para adaptação do software a novas necessidades do cliente.

Antes de iniciar a reengenharia do sistema, foi necessário estudar e escolher o modelo de desenvolvimento de software a ser seguido. Para tanto, foi feita uma pesquisa detalhada de modo a identificar quais são os modelos mais utilizados, verificar as vantagens e desvantagens destes e verificar o mais vantajoso em relação ao escopo da pesquisa.

Existem vários modelos de desenvolvimento, os quais apresentam vantagens e desvantagens e, dessa forma, devem ser analisados e utilizados de acordo com as peculiaridades de cada (Sommerville, 2011). Neste trabalho, são apresentados alguns dos modelos, baseados em Sommerville (2011), sendo ressaltados os selecionados para o desenvolvimento do projeto em questão, apresentados na próxima subseção

### 4.2.1 Metodologia Tradicional

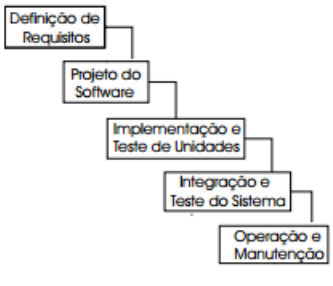
As metodologias tradicionais são também conhecidas como metodologias pesadas ou orientadas a documentação. É importante ressaltar que essas metodologias surgiram em um contexto muito diferente do que vivemos atualmente, pois eram voltadas para mainframes e terminais burros (ROYCE, 1970). Na época, o custo para fazer alterações e correções era muito alto, uma vez que o acesso a computadores era limitado e não existiam ferramentas de apoio ao desenvolvimento de software, como depuradores e analisadores de código (Koscianski, 2007).

Tendo como base todo esse engessamento inicial ao desenvolver softwares, surgiu a principal metodologia tradicional, muito utilizada até hoje, que é o modelo clássico. Este, ficou muito conhecido também como modelo cascata ou sequencial (PRESSMAN, 2002), onde cada etapa tem associada ao seu término uma documentação que deve ser aprovada para que a etapa posterior possa ter início.

Nos modelos clássicos o projeto é dividido em fases de uma maneira inflexível (Reis, 2001). O modelo cascata, por exemplo, após finalizar a fase de desenvolvimento não se propõem realizar mudança das especificações. Já, o modelo espiral, proposto por (BOEHM, 1998), diferentemente do cascata, até permite um retorno às fases anteriores para algumas modificações nas especificações, porém não implementa a ideia de execução paralela de tarefas.

De uma forma geral, o modelo clássico deve ser usado como metodologia de desenvolvimento de software somente quando no projeto os requisitos forem concisos e estáveis, podendo assim chegar em um resultado de sucesso (Pressman, 2002). Na Figura 5 são apresentadas as fases de evolução do modelo clássico.

Figura 5 – Modelo Clássico




Fonte: Soares (2014).

Como apresentado na Figura 5, as fases desse modelo são engessadas e devem ser finalizadas antes do início da seguinte. Esta metodologia não foi utilizada neste trabalho, uma vez que o sistema já havia sido desenvolvido e necessitou-se fazer uma reengenharia de sistema e, consequente, remodelagem do BD.

A metodologia tradicional apresenta muita documentação e suas fases são todas muito engessadas, que por sua vez, torna o prazo de desenvolvimento muito grande. Os requisitos são todos levantados em uma primeira etapa, apenas depois de todos os requisitos implementados que eles são mostrados ao cliente. Nessa pesquisa, como os requisitos eram descobertos a medida que o cliente tinha contato com o programa, ficou claro que a metodologia tradicional não atenderia a necessidade.

### 4.2.2 Agil

Atualmente as empresas trabalham em cenário global de operações onde todas as mudanças ocorrem muito rápido. Dessa forma, é preciso responder rapidamente a novas oportunidades e novos seguimentos de mercado, as constantes mudanças na economia e o surgimento de produtos e serviços concorrentes. Nos dias atuais, os softwares fazem parte de quase todas essas operações, assim, novos sistemas devem ser desenvolvidos rapidamente para obterem proveito das oportunidades e responder as pressões competitivas do mercado.

Neste sentido, o desenvolvimento com qualidade e a entrega rápida passam a serem requisitos críticos para o desenvolvimento de sistemas de software (Sommerville, 2013), assim os processos são elaborados com o objetivo de produzir, rapidamente, softwares úteis, conhecido como metodologia de desenvolvimento ágil.

Para tanto, os softwares tendem a não serem desenvolvidos como uma entidade única, mas como uma série de incrementos onde cada um inclui uma nova funcionalidade ao sistema e as versões (releases) são liberadas. A cada nova release o software vai ganhando mais maturidade, visto que a anterior foi exaustivamente testada por usuários e o feedback pode ser utilizado como lição aprendida para as novas releases que serão lançadas.

Existem muitas abordagens para o desenvolvimento ágil de software, porém elas compartilham de algumas caraterísticas em comum (Sommerville, 2013):

- As fases de especificação, projeto e implementação são intercaladas. Não existe uma especificação detalhada do sistema e a documentação é a mínima possível. Os requisitos do usuário definem apenas as características mais importantes do sistema.

- O sistema é desenvolvido baseado em uma série de versões. Cada versão é avaliada pelos usuários finais e stakeholders. Esses podem propor alterações ao software e gerar novos requisitos para serem implementados em uma futura versão. O sistema está apto a evolução.

- Geralmente as interfaces do usuário serão desenvolvidas por meio de um sistema interativo de desenvolvimento que permite a criação rápida de interfaces e posicionamento de ícones na interface

Os métodos ágeis são considerados métodos de desenvolvimento incremental, onde os incrementos são pequenos e novas versões são geradas e disponibilizadas ao cliente em curtos prazos. O envolvimento do cliente é necessário para obter um feedback rápido sobre a evolução do software. Dessa forma, a documentação é minimizada, pois irá se utilizar mais a comunicação informal do que reuniões formais com documentos escritos.

A filosofia refletida por trás dos métodos ágeis pode ser encontrada no manifesto ágil, que foi acordado por muito dos principais desenvolvedores desses métodos ágeis (Sommerville, 2013). O manifesto afirma o seguinte:

> " *Estamos descobrindo melhores maneiras de desenvolver softwares, fazendo-o e ajudando outros a fazê-lo. Através desse trabalho, valorizamos mais os indivíduos e interações do que os processos e ferramentas, software em funcionamento do que documentação abrangente, colaboração do cliente do que negociação de contrato, respostas à mudanças do que seguir um plano. Ou seja, embora itens à direita sejam importantes, valorizamos mais os que estão à esquerda*"(SOMMERVILLE, 2013)

Assim, é importante ressaltar que o Manifesto Ágil não rejeita os processos e ferramentas, a documentação, os contratos ou o planejamento do desenvolvimento, mas que esses têm uma importância de grau menor ou secundária quando comparada com as interações com indivíduos, o software estar utilizável, a colaboração dos usuários finais ou stakeholders e o rápido feedback.

Neste sentido, este trabalho teve o desenvolvimento ágil como metodologia de desenvolvimento, devido a característica do projeto em se fazer a evolução de um sistema já iniciado.

## 4.3 Data Mining

A expressão Data Mining e está relacionada com mercado de informática, cuja a tradução é mineração de dados. Consiste em uma funcionalidade que agrupa e organiza dados, encontrando neles padrões, associações, mudanças e anomalias relevantes (Cortês, 2002).

Porém, mineração de dados é muito mais do que isso, pois ela é apenas parte de um processo ainda maior de pesquisa denominado Busca de Conhecimento em Banco de Dados (Knowledge Discovery in Database – KDD), esse que por sua vez possui uma metodologia própria para preparação e exploração de dados, interpretação de resultados e assimilação de conhecimentos minerados (Cortês, 2002).

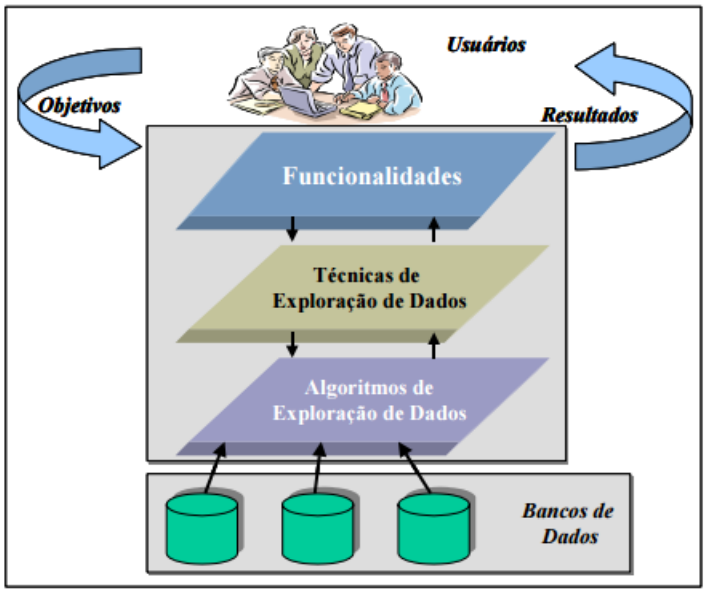
### 4.3.1 Funcionalidades do Data Mining

Para realizar a aplicação de mineração de dados são necessários vários estágios de desenvolvimento, mas o principal estágio que deve ser definido antes de iniciar a busca do conhecimento é definir claramente quais resultados deseja-se alcançar.

Existem variados tipos de armazenamento de dados e de BDs, podendo ser aplicar o processo de mineração de dados em todos eles. Tendo definido os tipos dos dados armazenados, pode-se decidir pelo tipo de padrões e quais relacionamentos deseja-se alcançar com a mineração. Essa funcionalidade é tratada por vários autores como resultados (outcomes) ou tarefas (tasks). (Cortês, 2002), (Pieter, 1996), (Michael, 1997).

Ao pesquisar sobre Data Mining é possível que, em muitos casos, as diferenças entre funcionalidades e técnicas não fiquem claras. Por exemplo, uma coleção de técnicas podem ser utilizas para realizar a análise de produtos, porém a tarefa de análise é também uma aplicação que busca produtos (Cortês, 2002). Observe a figura 4, que ilustra um exemplo de como esse processo ocorre.

Figura 6 – Interatividade entre funcionalidades e técnicas da mineração




Fonte: Cortês (2002).

Como pode ser observado, na Figura 6 foi exemplificado, através de camadas, as interações entre funcionalidades, técnicas e algoritmos, visando esclarecer como essas se dão a fim de realizar a mineração dos dados como as técnicas a serem empregadas.

É importante ressaltar que diferentes autores tratam as funcionalidades da mineração de dados de forma diferente. A seguir são apresentadas algumas das formas de tratamento defendidas por esses autores, as quais foram estudas para uso neste trabalho:

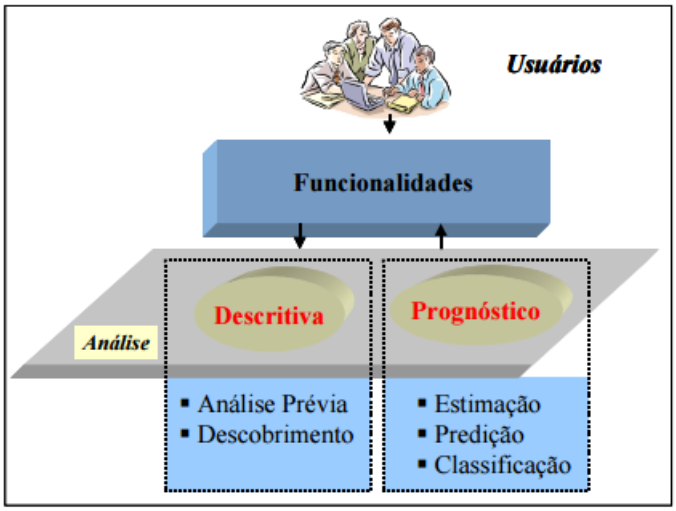
- Descoberta de conhecimento e Predição (Pieter, 1996).
- Classificação, Estimação, Predição, Afinidade em grupos, Agrupamentos (Clustering) e Descrição (Michael, 1997).
- Previsão, Identificação, Classificação e Otimização (Bhavani, 1999).

- Descrição e Predição (Ramez, 1999).
- Predição, Classificação, Agrupamentos (Clustering), Segmentação, Associação, Visualização e Otimização (Jiawei, 2001).
- Classificação, Estimação, Segmentação e Descrição (Mena, 1999).
- Predição, Detecção de desvio, Segmentação, Agupamentos (Clustering), analise de ligações e Regras de associação, Sumarização e Visualização e Garimpagem em Textos (Westphal, 1998)
- Descoberta de conhecimento e Predição (Pieter, 1996).
- Classificação, Estimação, Predição, Afinidade em grupos, Agrupamentos (Clustering) e Descrição (Michael, 1997).
- Previsão, Identificação, Classificação e Otimização (Bhavani, 1999).
- Descrição e Predição (Ramez, 1999).
- Predição, Classificação, Agrupamentos (Clustering), Segmentação, Associação, Visualização e Otimização (Jiawei, 2001).
- Classificação, Estimação, Segmentação e Descrição (Mena, 1999).
- Predição, Detecção de desvio, Segmentação, Agupamentos (Clustering), analise de ligações e Regras de associação, Sumarização e Visualização e Garimpagem em Textos (Westphal, 1998)

Como pode ser observado, a funcionalidade em mineração de dados está longe de ser tratada com um senso comum entre os autores e eles tendem a tratar o termo mais pela sua área de atuação do que pelo formalismo necessário. No entanto, podemos considerar que definir bem os conceitos de funcionalidade da mineração de dados, quais resultados deseja-se obter, é fundamental para o processo como um todo, uma vez que se bem definidos fica mais fácil escolher quais técnicas serão mais relevantes no trabalho para se atingir os resultados esperados. Na Figura 7 é representada a abordagem das funcionalidades em mineração de dados proposta por Cortês (2002).

Como apresentado na Figura 7, as funcionalidades em mineração de dados podem ser divididas em análise descritiva (análise prévia e descobrimento) e análise de prognóstico (estimação, predição, classificação). A principal vantagem em focar essa abordagem das funcionalidades está relacionada à facilidade com que elas podem ser aplicadas quando surge a necessidade de uma nova análise de dados. Por exemplo, no caso de uma nova análise basta identificar quais resultados deseja-se obter e imediatamente partir para identificação de qual técnica aplicar.

Figura 7 – Funcionalidades em mineração de dados




Fonte: Cortês (2002).

## 4.4 Reengenharia e Remodelagem

Em sua grande maioria, sistemas de software apresentam um estado continuo de mudanças, podendo a manutenção destes sistemas torna-se difícil a medida que essa prática se torna excessivamente redundante, sendo necessário reconstruir o sistema partindo do sistema existente, recorrendo algumas técnicas de Reengenharia de Software (RS), para que a manutenção seja mais flexível (Dinis, 2009).

A RS tem como objetivo melhorar um sistema de alguma maneira, por meio de alterações significativas que proporcionem uma melhor experiência de uso ao usuário, porém, sem alterar suas funcionalidades base. De acordo com Dinis (2009), o processo de reengenharia pode ocorrer em dois principais níveis de abstração distintos: ao nível de negócio e ao nível do software.

Ao nível de negócio a reengenharia é focada nos processos de negócio com a intenção de alterá-los, aumentando a competitividade da organização em determinada área. Enquanto que a nível de software atua analisando o sistema, procurando reconstruí-lo ou restaurá-lo com a maior qualidade possível.

Como o foco nesta monografia foi voltado para a reformulação apenas do software, a pesquisa e desenvolvimento teve como foco a reengenharia de software.

Quando um software começa a demonstrar sinais de que já não atende as demandas do usuário ou da organização, ou seja, quando os requisitos mudam muito, se considerado o que foi planejado e as atuais necessidades emergentes, surge a necessidade de se reconstruir um sistema similar ou adaptar o sistema existente, adicionando novas funcionalidades, melhorando a performance e sua consequente manutenção (Dinis, 2009).

Os sistemas de softwares são variáveis, podendo sofrer constantes atualizações, tais

como para a correção de erros, aplicação de ferramentas para melhoria do desempenho, adição de novas funcionalidades ou até mesmo adaptações para novas plataformas, podendo ser estas de hardware e de software.

A atividade de RS, dependendo do nível de modificações, pode ser algo com elevado custo às organizações, no sentido financeiro e de tempo empregado, por isso a organização necessita de uma estratégia prática (Pressman, 2006).

Antes de apresentar as efetivas tarefas de reengenharia, empregadas neste trabalho, apresenta-se, a seguir, alguns aspectos considerados como relevantes para o sucesso desta monografia, apresentados por Dinis (2009):

- Analisar o software criando uma lista de critérios, de modo que o processo de RS seja sistemático.
- Avaliar o nível de reengenharia, demorando menos tempo, assim o custo é reduzido.
- Analisar o sistema antigo, pois serve de base para aplicação do processo de reengenharia

Ranson e Waren (2002) ressaltam que a reengenharia é uma solução que deve ser considerada pelas empresas, pois os riscos de desenvolvimento e implementação de um novo sistema podem ser muito altos, inviabilizando a implementação do mesmo. Além disso, estudos têm demonstrado que a reengenharia, quando aplicada apropriadamente, geralmente promove custo efetivo e menos riscos que desenvolver um novo sistema (Ranson e Waren, 2002).

Segundo Pressman (2006), o modelo para reengenharia de software pode ser dividido em 6 atividades, sendo elas: Análise de inventário; Reestruturação de Documentos; Engenharia Reversa; Reestruturação de Código; Reestruturação de Dados e Engenharia Progressiva. Essas atividades podem ocorrer em sequência linear, mas nem sempre é isso que o ocorre, em vezes elas são aplicadas seguindo um modelo cíclico, isso demostra que cada uma das atividades apresentadas pode ser revisitada (Pressman, 2006).

A seguir é apresentada uma breve introdução sobre cada uma das atividades, propostas por Pressman (2006), para o modelo de reengenharia de software.

- Análise do Inventário: Em geral as organizações mantêm um inventário de todas as aplicações de software, que é uma lista com alguns detalhes dos softwares de uma determinada organização (Dinis, 2009).
- Reestruturação de documentos: A atividade de criar documentos é demorada. Sendo assim, em alguns casos a restruturação de documentos é correta e em outros não é necessário refazer toda documentação. Nessa perspectiva, Pressman nos apresenta 3 opções:
  - Se o sistema é estável e não existem alterações para ser realizada sobre ele, logo não existe a necessidade de dedicar tempo na documentação desses.

  - Se o sistema não foi completamente alterado, apenas parte dele está a ser analisado, é necessário documentar apenas a parte do mesmo que foi modificada e a documentação refletir a alteração.
  - Se o sistema é crítico para o negócio como um todo, então é altamente recomendado que se produza documentação para todo o sistema.

- Engenharia Reversa: A Engenharia Reversa teve sua origem na indústria de hardware, onde empresas concorrentes desmanchavam produtos de suas concorrentes a fim de descobrirem os seus segredos de fabricação. Em geral, a Engenharia Reversa parte de um sistema existente, analisando-o de forma a identificar os seus componentes e as inter-relações.

- Restruturação de Código: Alguns sistemas antigos apresentam uma arquitetura de desenvolvimento bem sólida, porém alguns módulos podem ter sido codificados de uma forma não elegante, o que torna o código difícil de compreender, testar ou realizar manutenção. A estrutura é registada e o código é então reestruturado, revisto e testado para garantir que nenhuma anomalia foi inserida, sendo a documentação interna do código atualizada.

- Reestruturação de Dados: A arquitetura de dados atual do sistema deve ser analisada e os modelos de dados necessários definidos, deste modo, quando a estrutura atual é julgada ineficiente os dados devem passar pela reengenharia. Como a arquitetura dos dados está fortemente interligada a arquitetura de programa e na forma como foi desenvolvido os algoritmos, esses também serão afetados pela restruturação do código.

- Engenharia Progressiva: A Engenharia Progressiva, também chamada de Engenharia de Renovação ou Recomposição, recupera as informações do software existente e usa esses dados para alterar ou reconstruir o sistema num objetivo de melhorar a sua qualidade global.

Os assuntos discorridos neste capítulo foram relevantes neste trabalho uma vez que, para reengenharia do sistema e desenvolvimento de novas funcionalidades, foram utilizados conceitos, técnicas e aspectos de cada tópico apresentado.

No próximo capítulo é apresentado o levantamento de dados, necessário para remodelação do sistema, identificação e licitação de novos requisitos, seguido do capítulo que apresenta o desenvolvimento do trabalho.

CAPÍTULO

# 5

# LEVANTAMENTO DE DADOS

Neste capítulo é apresentado como foi realizado o levantamento de dados e requisitos para a reformulação do sistema. Partindo das formalidades defendida pela Engenharia de Requisitos (ER), é, inicialmente, apresentado como a ES foi aplicada durante o desenvolvimento da reformulação.

## 5.1 Entrevista

De acordo com Sommerville (2009) as entrevistas formais ou informais, com os stackholders envolvidos no processo, são parte da maioria dos processos de ER. Neste trabalho, várias entrevistas foram realizadas com as pessoas envolvidas com o sistema, de modo que sejam identificadas todas as características que o sistema deve ter, sendo estes os requisitos já existentes e novos requisitos, para que o sistema possa ser reformulado, fazendo a reengenharia, remodelação de arquitetura e BD de maneira adequada.

Seguindo os conceitos e princípios levantados por Sommerville (2009), uma entrevista deve conter em único assunto principal, para que o usuário não fique confuso. Portanto, as perguntas dessa entrevista focaram em um único alvo, sendo: **"Compreensão do motivo pelo qual o sistema Fiscaliza-RU versão 1 (Fiscaliza-RU.v1) ainda não é utilizado"**.

Ainda, segundo o mesmo autor, para os conceitos referente a entrevista, o entrevistador deve preparar as perguntas relacionadas apenas ao assunto da reunião, para evitar que o encontro seja vazio e pouco produtivo. As questões estão a seguir, nomeadas como Questão de Pesquisa (QP), seguidas de demais dados pertinentes ao encontro. É importante ressaltar que são apresentados os dados da entrevista e uma representação (S1...Sn) para indicar o stakeholder envolvido na entrevista, os nomes foram omitidos por questões de sigilo, sendo utilizados como segue:

- S1: Nutricionista, responsável pela fiscalização do cardápio oferecido aos alunos. Possível usuário do sistema para uso efetivo;

- S2: Funcionário da CCOM, responsável por receber a planilha de movimentações do restaurante. Potente usuário do sistema, para uso efeito.
- S3: Prof. Dr., envolvido na pesquisa, que conhece da área de ES, BD e das regras de negócio que devem ser seguidas para correta implementação do sistema. Possível usuário do sistema, para verificação de implementação

É importante ressaltar que, por meio das entrevistas, foram identificadas alterações, no Fiscaliza-RU.v1, que devem ser implementadas, no Fiscaliza-RU. No entanto, nem todas elas foram referenciadas como novos requisitos, por serem apenas mudanças recorrentes do não correto desenvolvimento ou não adequada aplicação no sistema, em sua versão inicial, e que, em vezes, não se encaixavam como novos requisitos, porém precisaram serem corrigidas. Assim, neste capítulo são apresentadas as características, bem como as novas funcionalidade e requisitos que o Fiscaliza-RU deve conter.

1. **PrimeiraPrimeira Reunião (RI)**

   **Data:** 25/01/2017 - 1º encontro com os usuários do sistema.

   **Horário de Início:** 14:30 – **Horário de Término:** 15:00;

   **Participantes:** S1, S2 e S3.

   **Ponto analisado:** O sistema Fiscaliza-RU.v1 já existe há algum tempo, porém, este nunca foi utilizado. Neste primeiro encontro foram levantas os pontos positivos e negativos do sistema em sua versão inicial, para que fosse possível entender o motivo pelo qual ele ainda não é utilizado e, assim, não errar novamente na reformulação do Fiscaliza-RU, versão atual desenvolvida neste trabalho.

   Durante a entrevista as QP foram feitas algumas questões, conforme representadas na Tabela 1.

Tabela 1 – Questões de Pesquisa

| Questão de Pesquisa | Reposta |
|---|---|
| QP1: Quais serão as pessoas que irão utilizar o sistema? | P2, funcionário da CCOM. |
| QP2: Como o sistema foi disponibilizado? | Foi disponibilizado apenas através de um link, porém não tinham os dados de acesso. |
| QP3: Foi realizado algum treinamento de como utilizar o sistema? | Apenas um treinamento informal, com apresentação das funcionalidades do mesmo. |
| QP4: Já haviam tentado utilizar o sistema antes? Se sim, quais foram as dificuldades? | Não tentaram pois não tinham os dados de acesso. |
| QP5: No seu primeiro contato com o sistema ou quando ele lhes foi apresentado, sentiram falta de alguma funcionalidade? | Não, pois sua experiência com o sistema era quase nula, para chegar alguma conclusão. |
| QP6: Para verificação de usabilidade, em uma nota de 0 a 10, onde 0 representa pouco usável e 10 representa muito usável, qual nota você atribuiria ao sistema? | Não foi possível opinar, devido ao pouco contato com o sistema. |

Observa-se, com os resultados conseguidos pela entrevista, apresentados na Tabela 1, que os usuários do sistema tiveram muita dificuldade em relação ao acesso ao Fiscaliza-RU.v1 e aos dados que este deveria apresentar, não sendo possível o uso do sistema em sua versão inicial.

O contato com os usuários finais do sistema foi limitado pois grande parte das entrevistas foram realizadas com um representante dos usuários, sendo este o S3. Como anteriormente mencionado, este conhece o processo que deve ser seguido para que o sistema seja relevante à UFG-RC, apresentando aspectos de usabilidade, além dos aspectos de regras de negócios e necessidades específicas para implementação relacionados às áreas da ES e BD.

Em cada encontro foi realizado o levantamento de características essenciais ao sistema, sendo esses os novos requisitos funcionais e não funcionais apresentados neste trabalho. Na Tabela 2 são apresentadas algumas das características identificadas na primeira reunião.

Tabela 2 – Características identificadas (possíveis requisitos) - RI

| **Descrição da atividade e Novos Requisito** | **MENU** |
|---|---|
| Busca por nome de aluno na primeira página | PRINCIPAL |
| Na página Home, adicionar link para redirecionar as páginas | PRINCIPAL |
| Implementar a busca por nome de aluno | USUARIOS - RESTAURANTE |
| Detalhe do aluno, mostrando todos os seus dados e informações de uso de restaurante | USUARIOS - RESTAURANTE |
| Filtrar lista de movimentações por datas (período), por curso e por modalidade de bolsa | USUARIOS - RESTAURANTE |
| Filtrar lista de alunos por curso | USUARIOS - RESTAURANTE |
| Filtrar lista de alunos por modalidade de bolsa | USUARIOS - RESTAURANTE |
| Relatórios de alunos que usam menos o restaurante (ou que nunca usaram) | RELATORIOS - GRÁFICOS |
| Relatórios de alunos que usam mais o restaurante | RELATORIOS - GRÁFICOS |
| Criar o menu cardápio | CARDÁPIO - CARDÁPIO |

As reuniões destacadas a seguir, RII até RIX, foram realizadas com desenvolvedor com alto conhecimento na área de engenharia de software e aspectos voltado para qualidade na usabilidade de sistemas. Esse conhecimento do entrevistado permitiu que as reuniões acontecessem de forma mais rápida e breve do que normalmente ocorreria com um usuário normal, sem conhecimentos sobre a área de ES.

2. **Segunda Reunião (RII)**

   **Data:** 05-05-2017.

   **Horário de Início:** 15:00 – Horário de Término: 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 3).

Tabela 3 – Características identificadas (possíveis requisitos) - RII

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Alterar a expressão nos filtros "selecione" pela palavra "todos" | USUARIOS - RESTAURANTE |
| Alterar a expressão nos filtros "selecione" pela palavra "todos" | USUARIOS - RELATORIO - TABELA |
| Corrigir a palavra "movimentação", pois apresenta erro de português | USUARIOS – RESTAURANTE -ALUNO |
| Corrigir o formato da data para: DD-MM-YYYY | USUARIOS – RESTAURANTE -ALUNO |
| Retirar as colunas "matricula", "nome" da tabela de movimentações | USUARIOS - RESTAURANTE - ALUNO |
| Retornar o valor total de movimentações na tabela | USUARIOS - RESTAURANTE - ALUNO |
| Trazer como default ao filtro de busca das movimentações, a última movimentação do usuário | USUARIOS - RESTAURANTE -ALUNO |
| Exibir o alerta que o aluno nunca teve movimentação | USUARIOS - RESTAURANTE -ALUNO |
| Criar um novo sub-menu para o seguinte relatório: alunos que nunca utilizaram o RU | RELATORIOS |
| Criar novos filtros para a busca: almoço, jantar | RELATORIOS - GRAFICOS |
| Criar novos filtros para a busca: almoço, jantar | RELATORIOS - TABELA |
| Nos filtros retirar a expressão: “filtrar”, pois, está redundante | RELATORIOS - GRAFICOS e TABELA |
| Corrigir as palavras da paginação, que contêm termos em inglês | BOLSAS - TIPOS DE BOLSA - REFEIÇÃO |
| Adicionar campo título | BOLSAS - TIPOS DE BOLSA - REFEIÇÃO |
| Fazer gráfico de gastos: Alunos x Universidade | USUARIOS - RESTAURANTE-ALUNO |

3. **Terceira Reunião (RIII)**

   **Data:** 11-05-2017.

   **Horário de Início:** 10:40 – **Horário de Término:** 11:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 4).

Tabela 4 – Características identificadas – RIII * Todos os sub-menus.

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Colocar o filtro em form-line | USUARIOS - RESTAURANTE - ALUNO |
| Renomear "Cardápio"por "Alimentação" | ALIMENTAÇÃO |
| Inserir Produtos | ALIMENTAÇÃO |
| Ao clicar na logo "FIS-UFG"é redirecionado para uma página inexistente, alterar isso | HOME |
| Centralizar os filtros em um arquivo para diminuir código repetido | RELATORIOS - * |
| Indexar os valores nos gráficos | RELATORIOS - MAIS/MENOS FREQUENTAM |
| Implementar menu que permita usuário inserir/editar/excluir categorias manualmente | ALIMENTAÇÃO |
| Implementar uma tela de visualização externa para que alunos possam acessar sem autenticar | ALIMENTAÇÃO |
| Permitir que o usuário possa gerar novo cardápio a partir de um modelo já existente | ALIMENTAÇÃO |

4. **Quarta Reunião (RIV)**

   **Data:**30-05-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:40.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 5).

Tabela 5 – Características identificadas – RVI * Todos os sub-menus

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Nos relatórios retirar os segundos, manter o formato HH:MM | RELATORIOS - * |
| Formatar datas no padrão brasileiro | CARDÁPIO / RELATORIOS |
| Renomear 'Importacao' para 'Importação' | IMPORTAÇÕES |
| Inserir algo no relatório que remeta a ideia de comida/restaurante | HOME |
| Ao gerar erro de importação avisar ao usuário que nenhuma importação foi realizada | IMPORTAÇÕES |
| Ao clicar sobre o nome de um aluno, ser redirecionado aos detalhes do aluno | RELATORIOS - * |
| Ao clicar sobre uma modalidade de bolsa, ser redirecionado à um relatório de alunos apenas da modalidade selecionada | BOLSAS |
| Ao clicar sobre um curso ser redirecionado à um relatório de alunos apenas de alunos do curso selecionado | CURSOS |
| Na página inicial retorna um gráfico das refeições realizadas | HOME |
| Na página inicial retorna um Top10 de alunos que mais/menos frequentam o restaurante | HOME |
| Na tela de adicionar usuários, gera erro ao tentar adicionar um novo usuário | USUÁRIOS - ACESSO AO SISTEMA |
| Na tela de adicionar usuário, permitir excluir usuário | USUÁRIOS - ACESSO AO SISTEMA |

5. **Quinta Reunião (RV)**

   **Data:**05-06-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 6).

Tabela 6 – Características identificadas – RV * Todos os sub-menus

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Retirar gráficos de movimentações realizadas da página HOME | HOME |
| Retirar gráficos de refeições importadas da página HOME | HOME |
| Ao clicar sobre o nome de um bolsa, o usuário é redirecionado para um relatório de alunos dessa modalidade, porém o filtro não vem pré-selecionado | BOLSAS |
| Ao clicar sobre o nome de um curso, o usuário é redirecionado para um relatório de alunos desse curso, porém o filtro não vem pré-selecionado | CURSOS |
| Corrigir paginação que está em inglês | RELATORIOS - * |
| Disponibilizar uma planilha de importação modelo para download | IMPORTAÇÕES |
| Ao clicar sobre uma modalidade de bolsa ser redirecionado à um relatório de alunos apenas da modalidade selecionada | BOLSAS |
| Ao clicar sobre um curso ser redirecionado à um relatório de alunos apenas de alunos do curso selecionado | CURSOS |
| Na página inicial retorna um gráfico das refeições realizadas | HOME |
| Na página inicial retorna um Top10 de alunos que mais/menos frequentam o restaurante | HOME |
| No menu de detalhes do aluno, exibir uma mensagem ao usuário que as movimentações exibidas são referentes aos últimos dias que o usuário teve consumo | USUÁRIOS - RESTAURANTE - DETALHES |
| Implementar gráfico das estáticas das importações | IMPORTAÇÕES |

Tabela 7 – Características identificadas - RVI

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Editar Usuário, não está funcionando | Usuários - Restaurante |
| Usuários detalhes apresentar dados do aluno quando o mesmo não apresentar consumo | Usuários - Restaurante-Detalhes |
| Ao buscar um usuário manter o nome no input | Usuários - Restaurante |
| Verificar e validar importação de alunos | Usuários - Restaurante |
| Paginação de detalhes do aluno está em inglês | Usuários - Restaurante-Detalhes |
| Cardápio Online apresenta erro quando não existe cardápio programado para a semana. | Cardápio Online |

6. **Sexta Reunião (RVI)**

   **Data:**19-06-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 7).

Tabela 8 – Características identificadas - RVII

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Remover a expressão “Dados do Aluno" | Usuários - Restaurante-Detalhes |
| Curso está referenciando bolsa e vice-versa | Usuários - Restaurante-Detalhes |
| Corrigir Modalidade do Curso pois não está efetivando a alteração | Cursos |
| Inserir botão para editar alunos | Usuários - Restaurante-Detalhes |
| A view de detalhes dos alunos deve trazer todos atributos do estudante | Usuários - Restaurante-Detalhes |
| Corrigir "Guarnicao"para "Guarnição" | Alimentação - Cardápio |

7. **Sétima Reunião (RVII)**

   **Data:**25-07-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 8).

Tabela 9 – Características identificadas - RVIII

| **Descrição da atividade e Novos Requisito** | **Menu** |
|---|---|
| Nos relatórios de despesas retirar ":"após o símbolo de dólar ($). | Usuários - Restaurante-Detalhes |
| Ao editar aluno deve-se permitir alterar seu status (ativo ou inativo) | Usuários - Restaurante-Editar |
| Ao visualizar alunos inativos, permitir reativa-los | Usuários - Restaurante |
| Ao cadastrar novo aluno manualmente, inserir obrigatoriedade para os campos bolsas e curso | Usuários - Restaurante-Adicionar |
| Corrigir a paginação do laravel para que não perder os filtros do relatório | Usuários - Restaurante |
| Excluir do sistema usuários que foram inseridos de forma errada | Usuários - Restaurante |
| Destacas alunos que foram inativados | Usuários - Restaurante |

8. **Oitava Reunião (RVII)**

   **Data:**27-07-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 9).

Tabela 10 – Características identificadas - RIX

| Descrição da atividade e Novos Requisito | Menu |
|---|---|
| Ao clicar sobre cursos, os filtros não estão sendo preenchidos. | Cursos |
| Quando editar um curso, campos obrigatórios estão ficando em branco. | Cursos - Editar Curso |
| Trazer o nome da bolsa que está sendo editada, no título da página. | Bolsas - Tipos de Bolsas - Editar Bolsas |
| Não permitir excluir um curso enquanto houver alunos vinculados | Cursos |

9. **Nona Reunião (RIX)**

   **Data:**01-08-2017.

   **Horário de Início:** 15:00 – **Horário de Término:** 15:30.

   **Participantes:** S3.

   **Ponto analisado:** Novas características relevantes ao sistema (apresentados na Tabela 10).

## 5.2 Restrições do Sistema

Para desenvolvimento do sistema, algumas restrições foram consideradas de acordo com regras de validação para o funcionamento do RU, identificadas durantes as reuniões com stakeholders, as quais são apresentadas a seguir:

- Restaurante Universitário:
  - Refeições divididas em dois turnos: Almoço: 10:30 e 14:00 – Jantar: 18:30 e 21:00.
  - Quaisquer refeições que estejam fora desse horário são consideradas inválidas e o software, Fiscaliza-RU, é capaz de identificá-las e são consideradas como inconsistentes pelo software ao analisar os relatórios submetidos pela empresa contratada.
  - O software foi desenvolvido originalmente com esse horário pré-definido como padrão, porém é possível o administrador do sistema, através do painel de controle, alterar esse escopo de horário.
- Tipos de alunos:
  - Segundo a CCOM, em 2016 a Universidade realiza o acompanhamento de 3 tipos de alunos diferentes.

  - O valor de cada refeição cobrada pela empresa contratada, em 2015, era de R$ 6.96 (seis reais e noventa e seis centavos), porem parte desse valor é custeado pela UFG e outra parte pelo próprio aluno. Assim, o valor final que a Universidade custeia por aluno, é relativo a qual modalidade esse aluno pertence.
  - Esses alunos são divididos em:
    * Subsidiado: Esse tipo de aluno não apresenta nenhum tipo de bolsa, portanto a Universidade custeia R$ 3.96 e o aluno deve pagar o restante. Por exemplo, se o valor é R$ 6.96 ele deve pagar R$ 3.00.
    * Parcial: Esse tipo de aluno recebe uma refeição totalmente custeada pela Universidade, caso o aluno faça outra refeição no mesmo dia, ele se comportará da mesma forma que um aluno subsidiado. Por exemplo: O aluno poderá almoçar sem custos a ele, porem seu jantar será igual de um aluno subsidiado.
    * Integral: Esse aluno recebe duas refeições no mesmo dia, sendo as duas refeições custeadas pela Universidade. Por exemplo: esse aluno poderá almoçar e jantar sem custos.

- Cada aluno recebe ao dia apensa 1 almoço e 1 jantar. O software de fiscalização verifica essa regra ao analisar os relatórios e também verifica se um aluno é de fato da modalidade integral.

## 5.3 Identificação de Requisitos

O Levantamento de requisitos é fase que identifica o que o cliente deseja e o que espera em relação ao sistema. Um requisito é uma funcionalidade que o sistema deverá executar (Bezerra, 2002). Os requisitos podem ser levantados através de questionários, entrevista, observação do funcionamento do sistema atual, dentre outros.

Como apresentado por McConnell (2011), um trabalho satisfatório deve ser feito para preparar a construção do software, uma vez que muito do sucesso ou falha dos projetos já tem sido determinada antes do desenvolvimento do sistema começar. O autor ressalta que se a base, sendo esse o planejamento do projeto, não tiver sido bem definido e for inadequado, o melhor a fazer durante a construção é minimizar os danos. Esses apontamentos são importantes, uma vez que este trabalho é continuação de uma pesquisa que falhou, mas que, no entanto, tinha potencial para ser uma pesquisa de sucesso devida à necessidade do sistema e aos benefícios decorrentes de sua utilização pelo setor interessado.

Neste trabalho, diferentes funcionalidades foram identificadas durante as entrevistas com os stakeholders, podendo estas serem requisitos a serem inseridos no Fiscaliza-RU. Nesta seção é discorrido sobre alguns dos requisitos com maior profundidade e importância, sendo eles classificados como requisitos funcionais e não funcionais.

Segundo Souto-Maior et al. (2006), a análise de requisitos possibilita que a função e o desempenho do software sejam especificados, que as interfaces com outros softwares sejam indicadas e que as restrições de projeto sejam estabelecidas, proporcionando critérios para avaliar a qualidade, logo que o software for construído. Assim sendo, a análise de requisitos consiste na coleta de informações e no planejamento do software a fim de que este se adeque as especificidades do caso concreto.

Os requisitos de um software são divididos em funcionais ou não funcionais, que, de acordo com Silva Filho (2010), os que requisitos funcionais se referem a funções que o software deve ser capaz de realizar, sob o ponto de vista do usuário, ou seja, o comportamento do software. Em contrapartida, os requisitos não funcionais definem as características do software, basicamente como o software fará para que sejam executadas as funções e comportamentos dos requisitos funcionais.

Assim, nesta seção foram levantados alguns requisitos importantes para desenvolvimento de um sistema relevante e de qualidade. Tais requisitos foram identificados e licitados após as reuniões com stakeholders envolvidos no processo. Nas próximas subseções os requisitos identificados são apresentados.

### 5.3.1 Requisitos Funcionais

A Tabela 11 representa os requisitos funcionais que foram levantados durante todo o processo, em relação à prioridade foi usada a legenda 0 para Baixa, 1 para Média e 2 para Alta prioridade.

Tabela 11 – Requisitos Funcionais

| ID | Descrição da atividade | MENU | Prioridade |
|---|---|---|---|
| RF01 | Busca por nome de aluno logo na primeira página | PRINCIPAL | 2 |
| RF02 | Na pagina home, adicionar link para redirecionar as páginas. | PRINCIPAL | 0 |
| RF03 | Implementar a busca por nome de aluno | USUARIOS - RESTAURANTE | 2 |
| RF04 | Detalhe do aluno, mostrando todos os seus dados e informações de uso de restaurante | USUARIOS - RESTAURANTE | 2 |
| RF05 | Filtrar lista de movimentações por datas (período), por curso e por modalidade de bolsa | USUARIOS - RESTAURANTE | 1 |
| RF06 | Filtrar lista de alunos por curso | USUARIOS - RESTAURANTE | 1 |
| RF07 | Filtrar lista de alunos por modalidade de bolsa | USUARIOS - RESTAURANTE | 1 |
| RF08 | Relatórios de alunos que usam menos o restaurante (ou que nunca usaram). | RELATORIOS - GRÁFICOS | 2 |
| RF09 | Relatórios de alunos que usam mais o restaurante | RELATORIOS - GRÁFICOS | 2 |
| RF10 | Criar o menu cardápio. | CARDÁPIO - CARDÁPIO | 0 |
| RF11 | Alterar a expressão nos filtros "selecione" pela palavra "todos" | USUARIOS - RESTAURANTE | 1 |
| RF12 | Alterar a expressão nos filtros "selecione" pela palavra "todos" | USUARIOS - RELÁTORIO - TABELA | 1 |
| RF13 | Corrigir a palavra "movimentação", pois apresenta erro de português | USUARIOS - RESTAURANTE - ALUNO | 0 |
| RF14 | Corrigir o formato da data para: dd-mm-yyyy | USUARIOS - RESTAURANTE - ALUNO | 1 |
| RF15 | Retirar as colunas "matricula", "nome"da tabela de movimentações | USUARIOS - RESTAURANTE - ALUNO | 0 |
| RF16 | Retornar o valor total de movimentações na tabela. | USUARIOS - RESTAURANTE-ALUNO | 1 |
| RF17 | Trazer como default ao filtro de busca das movimentações, a última movimentação do usuário. | USUARIOS - RESTAURANTE - ALUNO | 2 |
| RF18 | Criar um novo sub-menu para o seguinte relatório: alunos que nunca utilizaram o ru. | RELATORIOS | 1 |
| RF19 | Criar novos filtros para a busca: almoço, jantar | RELATORIOS - GRAFICOS | 0 |
| RF20 | Criar novos filtros para a busca: almoço, jantar | RELATORIOS - TABELA | 0 |
| RF21 | Nos filtros retirar a expressão: "filtrar" pois está redundante | RELATORIOS - GRAFICOS E TABELA | 0 |
| RF22 | Corrigir as palavras da paginação, que contem termos em inglês. | BOLSAS -TIPOS DE BOLSA - REFEIÇÃO | 1 |
| RF23 | Adicionar Titulo As Paginas de Navegações | BOLSAS -TIPOS DE BOLSA - REFEIÇÃO | 1 |
| RF24 | Gráfico de gastos: alunos x universidade | USUARIOS - RESTAURANTE-ALUNO | 2 |
| RF25 | Colocar o filtro em form-line | USUARIOS - RESTAURANTE – ALUNO | 1 |
| RF26 | Implementar possibilidade de Inserir Produtos no Cardápio | ALIMENTAÇÃO | 2 |
| RF27 | Ao clicar na logo "fis-ufg"é redirecionado para uma página inexistente. | HOME | 2 |
| RF28 | Implementar menu que permita usuário inserir/editar/excluir categorias manualmente | ALIMENTAÇÃO | 2 |
| RF29 | Ao clicar sobre o nome de um aluno, ser redirecionado aos detalhes do aluno. | RELATORIO S - * | 2 |
| RF30 | Ao clicar sobre uma modalidade de bolsa, ser redirecionado à um relatório de alunos apenas da modalidade selecionada | BOLSAS | 1 |
| RF31 | Ao clicar sobre um curso ser redirecionado à um relatório de alunos apenas de alunos do curso selecionado | CURSOS | 1 |
| RF32 | Na pagina inicial retorna um gráfico das refeições realizadas | HOME | 1 |
| RF33 | Na pagina inicial retorna um top10 de alunos que mais/menos frequentam o restaurante | HOME | 1 |

Tabela 12 – Requisitos Não Funcionais

| ID | Descrição da atividade | MENU | Prioridade |
|---|---|---|---|
| NF01 | Centralizar Os Filtros Em Um Arquivo Para Diminuir Código Repetido. | RELATORIOS - * | 2 |
| NF02 | Indexar Os Valores Nos Gráficos | RELATORIOS - MAIS/MENOS FREQUENTAM | 1 |
| NF03 | Implementar Uma Tela De Visualização Externa Para Que Alunos Possa Acessar Sem Autenticar | ALIMENTAÇÃO | 2 |
| NF04 | Permitir Que Usuário Possa Gerar Novo Cardápio A partir De Um Modelo Já Existente | ALIMENTAÇÃO | 2 |
| NF05 | Nos Relatórios Retirar Os Segundos, Manter O Formato HH:MM | RELATORIOS - * | 0 |
| NF06 | Formatar Datas No Padrão Brasileiro | CARDÁPIO / RELATORIOS | 0 |
| NF07 | Renomear 'Importacao' Para 'Importação' | IMPORTAÇÕES | 0 |
| NF08 | Inserir Algo No Relatório Que Remeta A Ideia De Comida/Restaurante | HOME | 1 |
| NF09 | Exibir O Alerta Que O Aluno Nunca Teve Movimentação | USUARIOS - RESTAURANTE -ALUNO | 2 |
| NF10 | Ao Gerar Erro De Importação Avisar Ao Usuário Que Nenhuma Importação Foi Realizada. | IMPORTAÇÕES | 2 |
| NF11 | Renomear "Cardápio"Por "Alimentação" | ALIMENTAÇÃO | 2 |
| NF12 | Corrigir Paginação Que Está Em Inglês | RELATORIOS - * | 2 |
| NF13 | No Menu De Detalhes Do Aluno, Exibir Uma Mensagem Ao Usuário Que As Movimentações Exibidas São Referentes Aos Últimos Dias Que O Usuário Teve Consumo | USUÁRIOS - RESTAURANTE - DETALHES | 2 |
| NF14 | Paginação De Detalhes Do Aluno Está Em Inglês | USUARIOS - RESTAURANTE-DETALHES | 2 |
| NF15 | A View De Detalhes Do Alunos Deve Trazer Todos Atributos Do Estudante | USUARIOS - RESTAURANTE -DETALHES | 2 |
| NF16 | Corrigir "Guarnicao"Para "Guarnição" | ALIMENTEÇÃO - CARDÁPIO | 2 |
| NF17 | Nos Relatórios De Despesas Retirar ":"Após O Símbolo De Dólar ($). | USUARIOS - RESTAURANTE -DETALHES | 2 |
| NF18 | Excluir Do Sistema Usuários Que Foram Inseridos De Forma Errada | USUARIOS - RESTAURANTE | 2 |

### 5.3.2 Requisitos Não Funcionais

Na Tabela 12 são apresentados os requisitos não funcionais que foram levantados durante todo o processo, em relação à prioridade foi usada a legenda 0 para Baixa, 1 para Média e 2 para Alta prioridade.

Neste capítulo foram apresentados os métodos utilizados para coleta de dados, por meio dos stakeholders, para tanto, reuniões foram feitas e foram identificadas novas características necessárias ao sistema Ficaliza-RU, das quais muitas se tornaram requisitos funcionais e não funcionais.

CAPÍTULO

# 6

# INFORMAÇÕES TÉCNICAS

O desenvolvimento do trabalho é voltado para a reformulação do Fiscaliza-RU.v1 com foco em melhorias de usabilidade e desempenho. A usabilidade era sua principal desvantagem, o sistema só fazia o que lhe era proposto, realizar importações de refeições, nada além dessa funcionalidade. Ainda, para realizar essa tarefa o usuário era obrigado a realizar uma série de passos difíceis, o que lhe exigia muita cautela para chegar ao seu objetivo. O desempenho do sistema era muito baixo ao realizar a importação de planilhas com muitas linhas, devido a um número alto de consultas feitas ao BD.

Vale ainda ressaltar que todos os usuários criados no sistema possuíam o mesmo perfil, sendo identificado que o Fiscaliza-RU.v1 seria limitado aos funcionários da CCOM e que o número de usuários com acesso ao sistema seria pequeno. Por esses motivos ficou definido que os usuários cadastrados teriam apenas um único perfil, sendo esse de administrador, com acesso a todas as funcionalidades.

## 6.1 Características técnicas do sistema

O sistema apresenta duas bases diferentes (desenvolvimento e produção), portanto, segue algumas características técnicas que cada base apresenta.

### *6.1.1 Base de desenvolvimento*

Para desenvolvimento do sistema foi utilizado os seguintes programas:

- **Sublime Text 3:** Utilizado como editor de texto, com suporte a sintaxe do PHP.
- **XAMPP Control Panel V3.2.2:** Utilizado para instanciar o servidor Web local (Apache) e o banco de dados (MariaDB).

- **Servidor Web:** Apache/2.4.23.
- **Database:** MariaDB 10.1.19-MariaDB
- **phpMyAdmin:** Utilizado como plataforma de gerenciamento ao banco de dados em sua versão 4.5.1.
- **Git:** Versão do Git utilizada na base de desenvolvimento foi 2.11.0.windows.1
- **GitBash:** Terminal git utilizado para controle e versionamento de arquivos em sua versão 8.6.6, com suporte a comandos UNIX.

### 6.1.2 Base de Produção

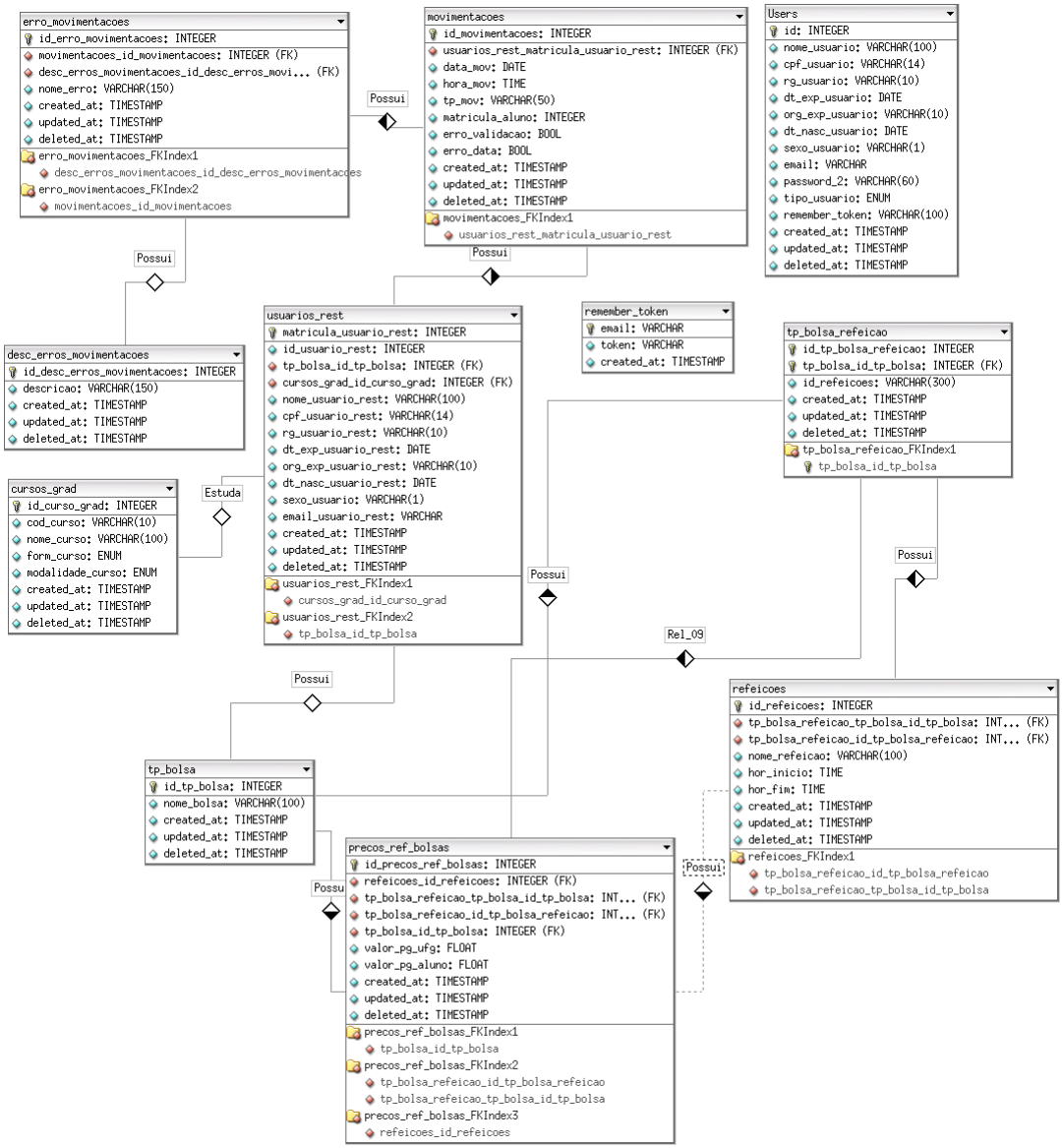
A base de produção foi utilizada apenas para hospedagem do sistema. A hospedagem é ofertada pela UFG e só pode ser acessada via SSH através de redes dentro do campus da Universidade. O acesso pode ser realizado através do domínio: *http://www.fiscaliza-ru.catalao.ufg.br/* ou *IP: 200.137.238.2*. Algumas características que valem ser apresentadas sobre a ela são:

- **Git:** Versão do git utilizada no servidor foi a 2.1.4
- **Servidor Web:** Apache/2.4.10 (Debian)
- **Database:** Mysql 5.5.49-0+deb8u1 - (Debian)
- **Putty:** Cliente SSH utilizado para acessar o servidor em sua versão 0.67.

## 6.2 Base de Dados

A base de dados do Fiscaliza-RU.v1 apresenta grande semelhança com a base de dados após a reformulação. Apenas poucas tabelas foram necessárias adicionar ao sistema. Conforme apresentado na Figura 8, são apresentadas as tabelas do sistema em sua versão original.

Figura 8 – Base de Dado, Fiscaliza-RU.v1




Fonte: Sousa (2016).

Algumas tabelas foram adicionadas com a reformulação do sistema, sendo elas:

- **Importações:** Histórico das importações de refeições
- **Importações_de_aluno:** Histórico das importações de aluno
- **Cardapio:** Histórico dos cardápios cadastrados
- **Semana:** Cada cardápio pertence uma semana
- **Cardapio_dia:** Cada semana, possui o cardápio do dia
- **Cardapio_composicoes:** Cada cardápio do dia, possui uma composição de produtos

- **Produtos:** Produtos que irão compor o cardápio;
- **Categoria_de_produtos:** Cada produto possui uma categoria.

Algumas tabelas foram modificadas:

- **Usuarios_rest:**
  - Foi adicionado o campo “importação_id” para saber qual a importação de origem do aluno.
  - Foi adicionado o campo “imagem_perfil” contem a URL para a imagem do aluno no servidor.
- **Movimentação:**
  - Foi adicionado o campo “importação_id” para saber qual a importação de origem a movimentação (refeição) no sistema.
  - Foi adicionado os campos “valor_pg_aluno” e “valor_pg_ufg” para ter o controle de quanto custava a refeição na época da importação.

## 6.3 Ambiente de Desenvolvimento

Durante ao discorrer deste capítulo é mostrado, em detalhes, como o ambiente de desenvolvimento foi utilizado para melhor a experiência de desenvolvimento e agilidade na migração dos códigos entre base de desenvolvimento e produção.

Ao desenvolver o código localmente era comum desenvolvedores levarem muito tempo para realizar a migração de códigos e o risco do trabalho se perder com um eventual acidente sem backup era alto (Git-scm, 2017). Com o advento da tecnologia muitas várias ferramentas foram desenvolvidas, facilitando o trabalho dos desenvolvedores que tentam minimizar os impactos em eventuais acidentes e diminuir a trabalhosa tarefa de migração de base de dados.

O GIT apresenta grandes benefícios e se adapta muito bem a metodologia ágil adotada nesse projeto. Uma vez que, com seu controle de versionamento é possível monitorar as evoluções que houve de um incremento para outro e também manter salvo o estado cada arquivo durante determinado incremento. Portanto, o desenvolvedor consegue manter um histórico de todos os incrementos e voltar o software para aquele determinado estado conforme sua necessidade.

### *6.3.1 GIT*

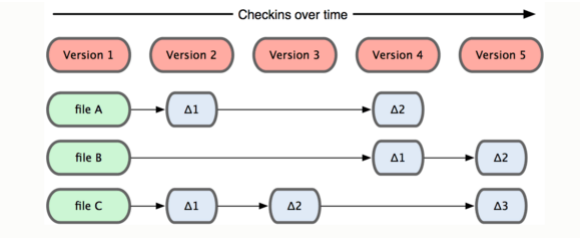
O desenvolvimento de softwares não é uma tarefa simples de se realizar, em geral, envolve equipes e vários programadores trabalhando em códigos compartilhados. Um dos problemas

encontrados com o compartilhamento de código e divisão de trabalho estava relacionado em como manter o código com a versão mais atual. O GIT veio para solucionar esses problemas dentre tantos outros envolvendo o desenvolvimento de software, utilizando controle de versionamento do código.

Assim, GIT é um sistema de controle de versão de arquivos, permitindo desenvolver projetos na qual diversas pessoas podem contribuir simultaneamente, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas (SCHMITZ, 2015). Existem outros sistemas de controle de versionamentos, tais como CVS, Subversion, Perforce, Bazaar, dentre outros. No entanto, existem certas diferenças sutis entre o GIT e esses, o que o torna mais atraente que os outros VCS (Subversion e similares inclusos).

A maioria dos VCS armazenam a informação como uma lista de mudanças por arquivo, tratando a informação que mantém como um conjunto de arquivos e as mudanças feitas a cada arquivo ao longo do tempo (Git-scm, 2017). Na Figura 9 é exibido como outros VCS tratam as mudanças que ocorrem em um arquivo, em geral estes costumam armazenar dados como mudanças em uma versão inicial de cada arquivo.

Figura 9 – Outros VCS

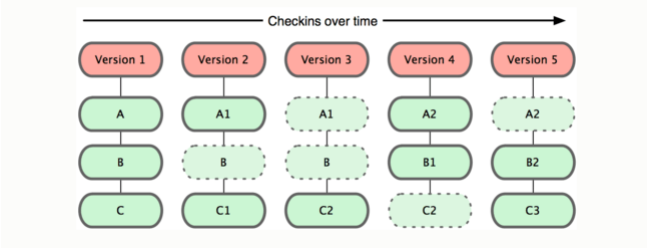



Fonte: GIT-Scm (2017).

Em contrapartida, GIT trata as modificações que ocorrem em um arquivo de forma totalmente diferente, considerando que os dados são como um conjunto de snapshots (captura de algo em um determinado instante, como em uma foto) de um mini-sistema de arquivos. Cada vez que é salvo ou consolidado (commit) o estado do projeto é como se ele tirasse uma foto de todos os arquivos naquele momento e armazenasse uma referência para essa captura (Git-scm, 2017). Na Figura 10 é mostrada a forma como que o Git realiza o controle de versionamento dos arquivos.

Como apresentado na Figura 10, para ser eficiente, se nenhum arquivo foi alterado, a informação não é armazenada novamente, apenas um link para o arquivo idêntico anterior que já foi armazenado.

Figura 10 – Git como sistema de versionamento




Fonte: GIT-Scm (2017).

### 6.3.2 Utilização do GIT neste Trabalho

Como apresentado anteriormente, este trabalho trata da reformulação de um sistema Web armazenado em uma base de produção em um servidor da UFG, com acesso restrito apenas a redes dentro da Universidade. Essa restrição dificultava muito o acesso aos códigos fontes da base de produção e à manutenção dos mesmos, para correção e evolução do código.

Neste sentido, para maior facilidade de acesso ao código fonte, boa qualidade de desenvolvimento, documentação das modificações, backup e integridade dos códigos foi realizado o instanciamento de um repositório GIT na base de produção dentro do servidor da UFG que hospeda o software. Para tanto, foi feita a cópia do software para um servidor local, na máquina do aluno responsável por esta monografia, sendo esta tratada como base de desenvolvimento. Assim, foi possível realizar as modificações necessárias, realizar commit para consolidar a modificação e depois, com um simples comando, git push enviar os arquivos ao servidor, se não houver conflitos os arquivos são copiado para a nova base de desenvolvimento, no servidor local. Outra vantagem é que a segurança do código foi aumentada, visto que existe um controle de versão para a nova base de desenvolvimento e outro controle de versão para a base de produção já existente.

Ainda, o GIT guarda um histórico de todos os seus commits e comentários sobre o que foi realizado em cada um deles, permitindo o programador voltar às diferentes versões do código.

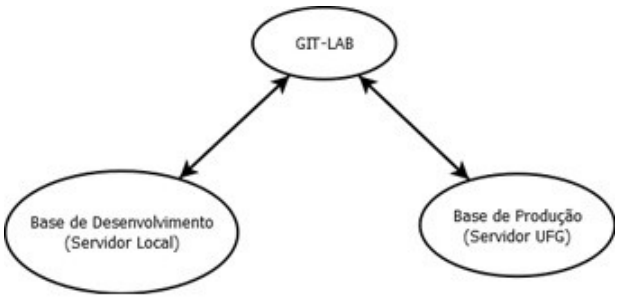
### 6.3.3 GITLab no Projeto

Como foi mencionado na sessão anterior, o acesso a base de produção era limitado às redes dentro da Universidade e para manter a integridade do código foi utilizada uma ferramenta Web para soluções GIT, que é o GitLab.

O GitLab é apenas um gerenciador de repositórios GIT para Web, além de dispor de outas ferramentas para times de desenvolvimento, tais como Wiki, Rastreamento de Issues. Nesse projeto serviu apenas de ponte entre a base de desenvolvimento (servidor local) e a base de produção (servidor UFG), possibilitando trabalhar no código na base de desenvolvimento e enviá-lo para o GitLab de qualquer rede, mantendo o código sempre atualizado e seguro na

nuvem. Apenas depois, quando julgasse necessário, e a rede da UFG estivesse disponível, seria preciso acessar o servidor da base de produção e realizar um git pull, para baixar os arquivos, para atualizar o software, na Figura 11 é apresentado como se deu essa comunicação

Figura 11 – Relacionamento entre as bases




Fonte: Elaborada pelo autor.

A Figura 11 representa as relações entre a base de desenvolvimento, GitLab e a base de produção, em que as modificações são centralizadas no GitLab e podem ser enviadas a todas as bases conectadas a ele. Essa prática é eficiente caso haja mais bases de outros desenvolvedores trabalhando no projeto também.

Outro fator que influenciou na escolha do GitLab é que ele é voltado para desenvolvimento entre times, apresentando várias soluções para que os times se interagem com maior dinâmica durante o processo de desenvolvimento. Visando que esse projeto tem potencial para se tornar maior e mais desenvolvedores venham atuar nele, o GitLab seria uma excelente ferramenta para integração do time de desenvolvimento, mantendo tudo documentado.

## 6.4 Framework Laravel

Para desenvolvimento mais seguro e rápido no projeto foi utilizado o framework Laravel que é uma potente e elegante ferramenta de desenvolvimento. Dando continuidade ao projeto desenvolvido originalmente, toda a reformulação do software foi feita utilizando o Laravel e as boas práticas de programação.

Laravel é um framework PHP utilizado no desenvolvimento Web, que utiliza arquitetura MVC (Model-View-Controller) e tem como principal objetivo o desenvolvimento de aplicações seguras e de boa performance de maneira rápida (ADRIEL, 2015).

A arquitetura MVC é oferecida como padrão pelo Larável e faz parte de como o framework trabalha e realiza suas rotinas. Contudo, a arquitetura oferece grandes benefícios ao desenvolvedor, como a possibilidade de separar códigos do back-end de códigos do front-end.

Para poder gerar a criação da interface gráfica, o framework utiliza uma engine de template denominada blade, que traz consigo uma gama de ferramentas que ajudam na criação rápida de interfaces elegantes, aproveitando ao máximo a reutilização de código.

No Laravel, a comunicação com o BD é facilitada ao máximo, pois ele utiliza uma ActiveRecord chamada Eloquent ORM (Object Relational Mapper) (Adriel, 2017). O Eloquent traz muitas funcionalidades que ajudam o programador automatizar seus códigos, facilita a reutilização de códigos e diminui muito a redigibilidade, pois permite ao programador fazer muitas coisas escrevendo pouco, apenas utilizando suas funcionalidades corretamente.

Atualmente, o framework está em sua versão 5.4, última versão em agosto de 2017 (Laravel, 2017). Porém, quando este projeto foi desenvolvido foi a utilizada a versão 5.1 para o desenvolvimento. Devido ao tempo restante para finalização deste trabalho, decidiu-se continuar com a versão 5.1, uma vez que é considerada ainda versão estável.

CAPÍTULO

7

# FISCALIZA-RU

McConnell (2011) menciona que a construção de software é a parte central no desenvolvimento de sistemas e que as principais atividades na construção estão relacionadas a um design detalhado, codificação, debugging, integração e teste de desenvolvimento (envolvendo testes de unidade e de integração).

Para tanto, neste trabalho essas atividades foram consideradas já que o sistema foi baseado em uma proposta de pesquisa já iniciada. A qualidade na reformulação do Fiscaliza-RU foi tida como meta para desenvolvimento deste trabalho, uma vez que, de acordo com McConnell (2011), a qualidade da construção afeta substancialmente a qualidade do software final.

A possibilidade de gerar diversos relatórios foi implementada na nova versão do sistema, como para realizar o acompanhamento de frequência dos alunos ao restaurante. Os relatórios podem ser verificados de forma geral ou específica. Os relatórios gerais são voltados para analisar os alunos de uma maneira genérica, permitindo filtros para curso, bolsa, horário (almoço ou jantar). Os relatórios específicos foram implementados para acompanhar um aluno em particular, verificando sua assiduidade ao restaurante, quantidade de almoços ou jantares por período de tempo.

Adicionalmente, foi desenvolvido um histórico de importações onde todas as importações são relacionadas, possibilitando maior controle das importações realizadas pelos usuários do sistema. Nesse histórico, cada importação recebe um número identificador (ID) e o usuário pode realizar diversos filtros conforme sua necessidade. Outra significante modificação, implementada no Fiscaliza-RU, foi possibilitar à CCOM realizar um acompanhamento detalhado dos alunos bolsistas e subsidiados, pois com o controle de gastos e frequência implementados a coordenação é capaz de identificar quais são os alunos que apresentam maior assiduidade ao restaurante.

Ao concluir a reformulação da proposta deste trabalho, para evolução do Fiscaliza-RU, teve-se ao final um número significativo de novas funcionalidades, se comparado ao Fiscaliza-RU.v1. Isto resultou em um número grande de novas telas, tendo estas sido disponibilizadas

em menus. Neste capítulo é apresentado o sistema Fiscaliza-RU reformulado, sendo exibidas algumas das telas criadas para adicionar as novas funcionalidades identificadas. Adicionalmente, os resultados adquiridos também são apresentados ao final.

Neste capítulo é apresentado o sistema Fiscaliza-RU reformulado, sendo exibidas algumas das telas criadas para adicionar as novas funcionalidades identificadas. Adicionalmente, os resultados adquiridos também são apresentados ao final.

## 7.1 Tela Inicial (Home)

A tela inicial do Ficaliza-RU reúne o máximo de informação necessária ao usuário. O objetivo foi manter mais visível, com acesso rápido e facilitado, as informações mais relevantes ao cotidiano do usuário do sistema. Na Figura 12 é possível observar algumas dessas informações de fácil acesso.

Figura 12 – Página Inicial do Fiscaliza-RU

Fonte: Elaborada pelo autor.

A Figura 12 representa a tela principal (home) do Fiscaliza-RU. Ao realizar login no sistema, o usuário é redirecionado para essa página. Um gráfico de refeições, realizadas em um período, é gerado logo no início para maior facilidade, o período pode variar de acordo com a necessidade do usuário, alterando os filtros de busca. Outra funcionalidade importante nessa tela é a opção de buscar aluno logo ao iniciar o sistema, permitindo acesso rápido aos alunos cadastrados no BD, uma vez que está é uma atividade recorrente do usuário. Na Figura 13 é apresentada a continuação da página principal.

A tela inicial, como foco na usabilidade do usuário, foi desenvolvida a fim de mostrar a

maior quantidade de informações possíveis. Dessa forma o usuário poderia conseguir o maior número de telas com a menor quantidade de cliques possíveis.

Figura 13 – Pagina Inicial do Fiscaliza-RU (continuação)

Fonte: Elaborada pelo autor.

Na Figura 13 é possível observar que, além das opções de busca e gráfico de refeições realizadas, outra informação importante é disponibilizada ao usuário. Um segundo gráfico relatando os alunos que mais e que menos frequentam o restaurante é importante para que a CCOM possa realizar o acompanhamento de quais alunos de fato estão frequentando o restaurante.

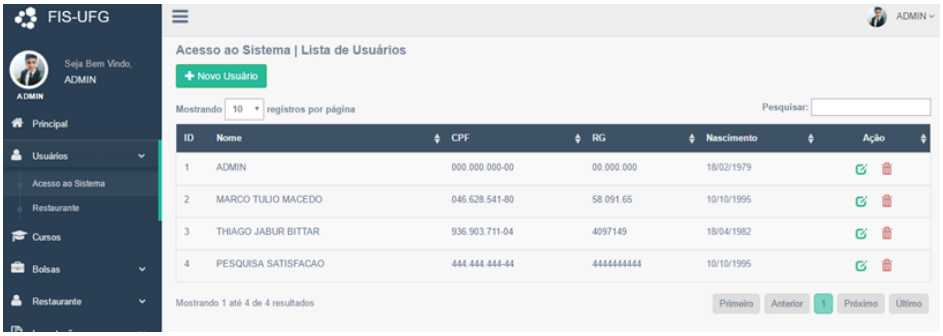
## 7.2 Usuários

O menu “Usuários” é composto por dois sub-menus, “Acesso ao sistema” e “Restaurante”. O “Acesso ao Sistema” representa os usuários com acesso ao programa e suas rotinas. Já o “Restaurante” representa os usuários que frequentam o restaurante, que nesse contexto, são os alunos.

### *7.2.1 Acesso ao Sistema*

A tela de acesso ao sistema é disponível apenas ao usuário com perfil de administrador, que é quem poderá cadastrar/alterar usuários no Fiscaliza-RU, como mostrado na Figura 14.

Figura 14 – Tela inicial, acesso ao Fiscaliza-RU

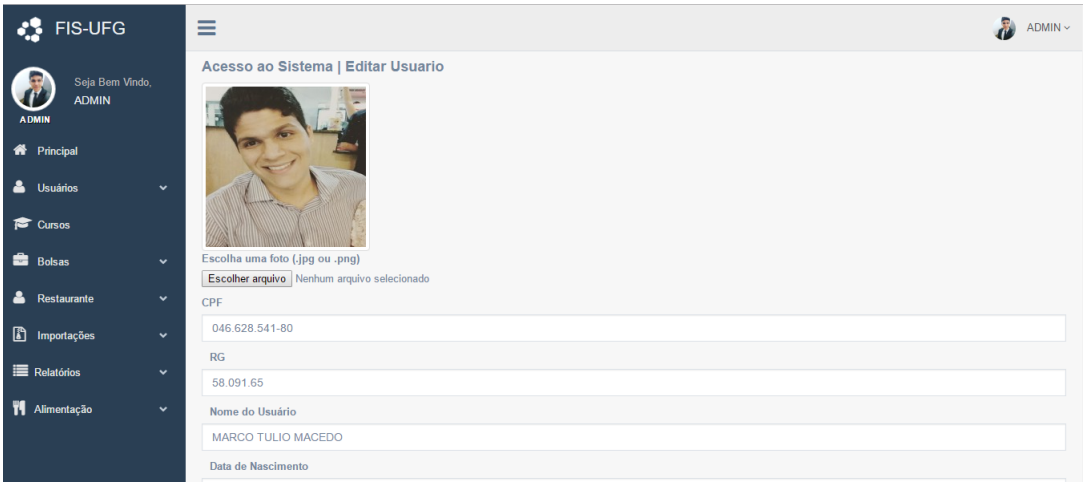



Fonte: Elaborada pelo autor.

Ao observar a Figura 14 é possível identificar que existem diversas funcionalidades nessa tela, sendo elas: “Adicionar novo usuário”; “Editar usuário”; “Remover Usuário” e “Filtrar Usuários”.

O cadastro/edição de usuários é feito de maneira similar, sendo utilizada a mesma tela para realização desses procedimentos, mostrado na Figura 15.

Figura 15 – Tela de Cadastro/Edição de usuário




Fonte: Elaborada pelo autor.

Assim, na Figura 15 é mostrada a tela para onde o usuário é redirecionado ao tentar realizar um cadastro/edição de outro funcionário ao sistema. O usuário deve preencher os dados solicitados e inserir uma foto de perfil.

### 7.2.2 Restaurante

O sub-menu restaurante representa os usuários com acesso ao RU, sendo estes os alunos da Universidade. Os alunos cadastrados nesse móodulo são verificados e acompanhados nos

relatórios e nas importações de refeições realizadas no RU. Na Figura 16 é apresentada a tela que relaciona todos os alunos cadastrados no Fiscaliza-RU.

Figura 16 – Tela de Cadastro/Edição de usuário

Fonte: Elaborada pelo autor.

Ao observar a Figura 16 é possível identificar várias funcionalidades que a tela permite o usuário realizar, dentre as quais pode-se citar:

- **Adicionar novo aluno:** Permite adicionar um novo aluno manualmente;
- **Editar aluno:** Permite editar informações manualmente sobre um aluno já cadastrado ao sistema;
- **Ver aluno:** Permite visualizar detalhes em específico sobre a frequência do aluno ao restaurante, quantidade de refeições, etc;
- **Inativar:** Não existe a opção excluir no Fiscaliza-RU. Os usuários do sistema conseguem apenas inativar um aluno, para que ele não seja considerado como aluno válido nas rotinas de importações do sistema. Caso o usuário deseja voltar atrás é possível editar o status do aluno para ativo novamente;
- **Importar alunos:** É possível adicionar alunos em lote, através da opção de importar alunos. Mais detalhes sobre essa funcionalidade são fornecidos no menu “Importações”.
- **Filtros:** O usuário possui vários filtros no relatório, podendo realizar buscas conforme sua necessidade.

### 7.2.3 Detalhes do Aluno

A tela detalhes do aluno é chamada quando o usuário seleciona a opção “Ver aluno”, conforme foi mostrado no menu anterior. Ao entrar nessa tela todas as informações do usuário são exibidas, desde o básico como nome, matrícula, até informações mais específicas como frequência ao restaurante, valores gastos, dentre outras. Na Figura 17 são apresentadas as informações básicas do usuário, enquanto a Figura 18 representa as informações mais detalhadas.

Figura 17 – Tela de detalhes do aluno

Fonte: Elaborada pelo autor.

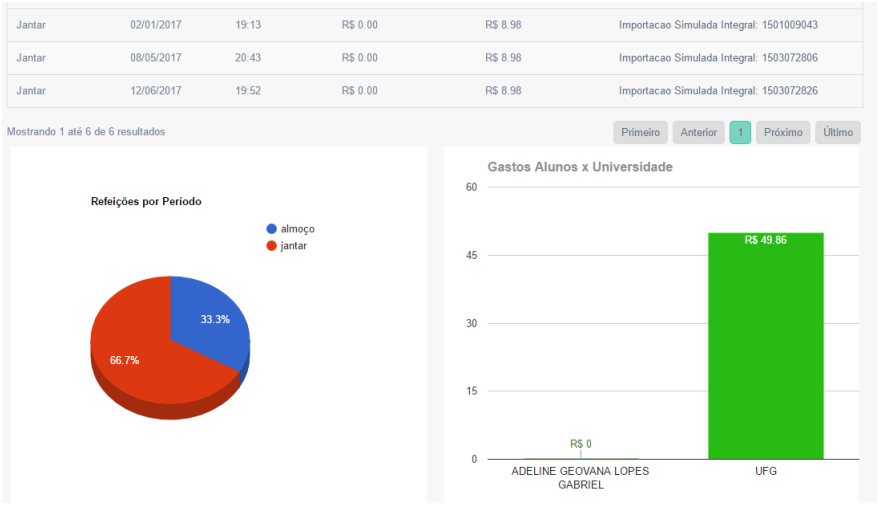
A figura 14 representa as informações básicas do aluno contempladas pelo usuário ao entrar na tela, por exemplo, se o aluno nunca apresentou consumo no restaurante, um aviso é gerado nessa tela e as informações avançadas, representadas na Figura 18, não são geradas.

Figura 18 – Tela de detalhes do aluno (continuação)

Fonte: Elaborada pelo autor.

A figura 15 representa uma continuação da tela de detalhes do aluno, onde o usuário pode contemplar algumas informações mais detalhadas sobre o aluno, que é o extrato de refeições em algum período de tempo. O período de tempo que pode ser filtrado conforme a necessidade do usuário utilizando o botão “buscar”. Ao entrar nessa tela, o sistema preenche, por padrão, o período de tempo automaticamente da seguinte forma: “De” (30 dias antes da última refeição) “Até” (última refeição). Dessa forma o usuário não precisa ficar procurando qual foi a última vez que um aluno frequentou o restaurante. A Figura 19 representa o final da tela de detalhes do aluno, onde algumas informações são resumidas em gráficos para melhor compreensão.

Figura 19 – Tela de detalhes do aluno (continuação)




Fonte: Elaborada pelo autor.

Na Figura 19 é exibida a continuidade da tela de detalhes do aluno. Um extrato das movimentações do aluno é gerado e este pode ser filtrado em um período de tempo conforme a necessidade do usuário. Gráficos também são gerados para uma melhor representação das informações. O gráfico em pizza representa, em porcentagem (%), quantas vezes o aluno almoçou ou jantou. O gráfico em barras representa os gastos do aluno e da Universidade. Alunos pertencentes a bolsa integral podem apresentar gastos nulos, pois suas refeições são integralmente pagas pela UFG.

## 7.3 Cursos

O menu “Cursos”, Figura 20, foi desenvolvido para permitir uma manutenção dinâmica dos cursos que a Universidade oferece na Regional. Cada aluno possui apenas um curso, porém, um curso possui vários alunos vinculados. Nesse menu o usuário poderá realizar as seguintes tarefas:

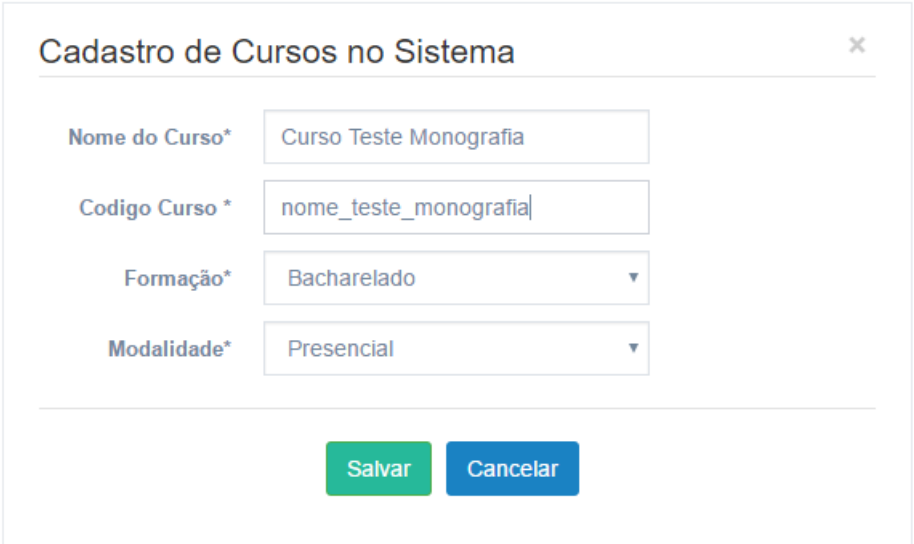
- **Adicionar novo curso:** Onde novos cursos são criados;
- **Editar curso:** Para alterar informações de curso já cadastrado;
- **Remover curso:** A remoção só poderá ocorrer caso não tenha alunos vinculados ao curso.

Figura 20 – Tela de Cursos

Fonte: Elaborada pelo autor.

Na Figura 20 são apresentados os cursos cadastrados no sistema. Cada curso possui alguns atributos básicos, como código (definido pelo usuário), nome do curso, formação (bacharelado ou licenciatura), modalidade (presencial ou EAD). O botão “Adicionar” permite adicionar um novo curso ao sistema. Ao selecionar esse botão, um modal, Figura 21, é exibido para realização do cadastro.

Figura 21 – Tela de adicionar Cursos




Fonte: Elaborada pelo autor.

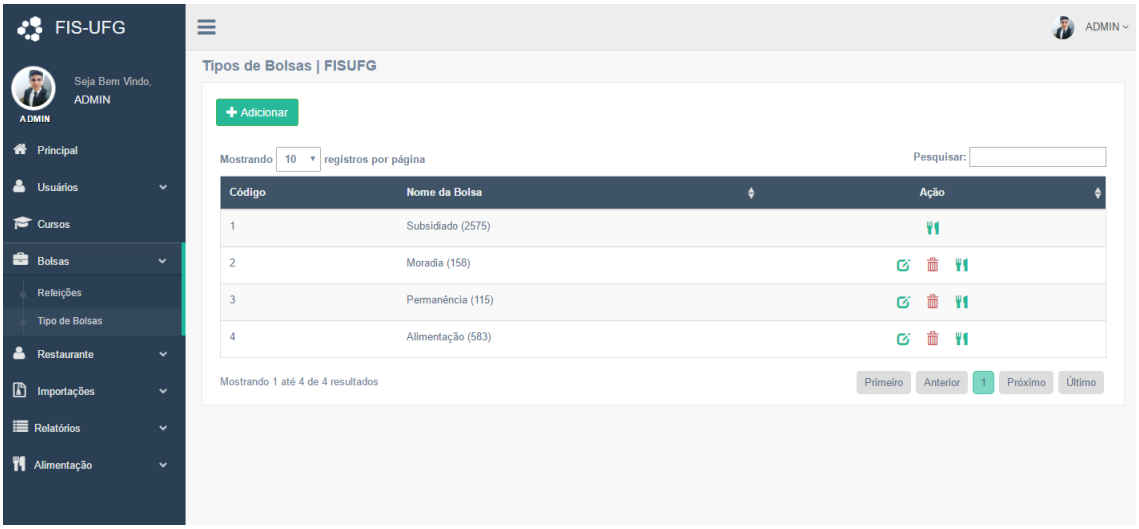
O modal apresentado, Figura 21, é utilizado para adição de novos cursos ao Fiscaliza-RU, o usuário deve preencher obrigatoriamente todos os campos do modal para adição do novo curso.

## 7.4 Bolsas

O menu “Bolsas”, Figura 22, foi desenvolvido para permitir uma manutenção dinâmica das bolsas que a UFG oferece na RC. Cada aluno possui apenas uma bolsa, porém uma bolsa possui vários alunos vinculados. Nesse menu o usuário poderá realizar as seguintes tarefas:

- **Adicionar nova bolsa:** Cadastra nova modalidade de bolsa;
- **Editar bolsa:** Faz alterações no tipo de bolsa já cadastrado;
- **Remover bolsa:** A remoção só poderá ocorrer caso não tenha alunos vinculados a bolsa

Figura 22 – Tela de Bolsas



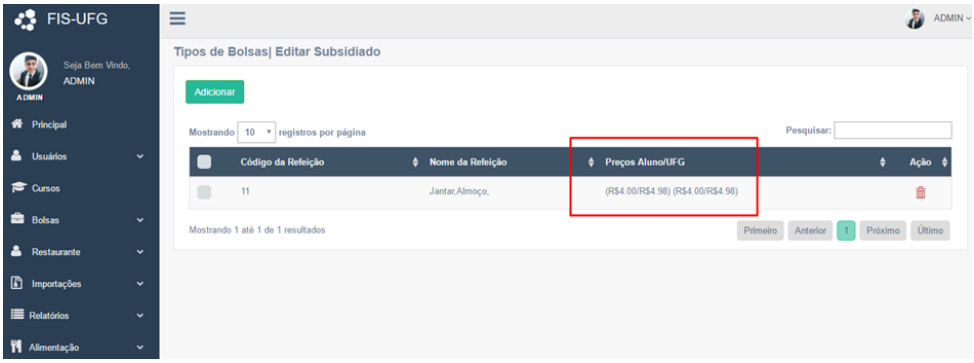

Fonte: Elaborada pelo autor.

Na Figura 22 são apresentadas as modalidades de bolsas cadastradas no sistema. Cada bolsa possui refeições vinculadas, onde é parametrizado o valor a ser pago pelo aluno e pela UFG. O botão “Adicionar” deverá ser utilizado caso seja necessário adicionar uma nova bolsa ao sistema, ao selecionar esse botão um modal é exibido para realização do cadastro.

Cada tipo de bolsa permite diferentes tipos de acesso aos alunos. Como já foi tratado na seção de “Restrições do Sistema”, o aluno subsidiado deverá pagar uma parte do valor pela refeição e a Universidade deverá oferecer um subsidio pelo restante do valor, enquanto que para o aluno bolsista integral a UFG deverá pagar o valor integralmente. O usuário deverá cadastrar no sistema qual o valor pago pelo aluno e pela Universidade, em cada tipo de bolsa, através da tela apresentada na Figura 23.

A Figura 23 representa a tela onde o usuário realiza as parametrizações de valores, total pago pelo aluno e pago pela UFG, por tipo de bolsa. Nesse caso é exibido um exemplo para alunos do tipo subsidiado, onde está parametrizado para que os alunos paguem R$4.00 e a Universidade R$4.98.

Figura 23 – Tela para vincular refeições à uma bolsa

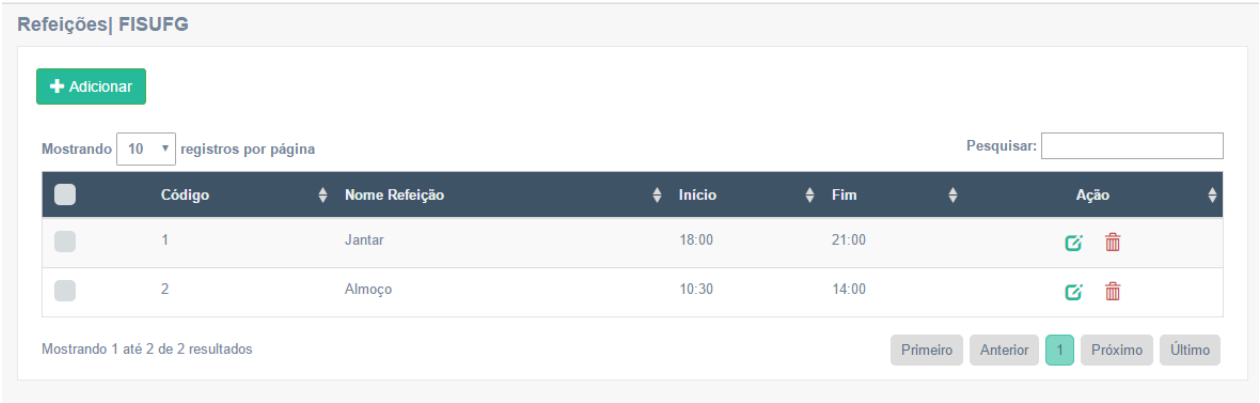



Fonte: Elaborada pelo autor.

### 7.4.1 Refeições

As refeições, como tratado na seção de “Restrições do Sistema”, ocorrem apenas em uma faixa de horário parametrizadas pelo usuário. Refeições fora do horário parametrizado devem ser consideradas como uma refeição inconsistente durante as importações. Na Figura 24 são apresentadas as refeições cadastradas no Ficaliza-RU e seus respectivos horários.

Figura 24 – Tela de tipos de refeições




Fonte: Elaborada pelo autor.

Atualmente existem apenas dois tipos de refeições servidas no restaurante, Figura 24, sendo almoço e jantar, onde almoço está definido para ocorrer entre 10:30 e 14:00 horas e o jantar entre 18:00 e 21:00 horas.

Essa tela é importante, pois torna o sistema flexível a novas mudanças que podem ocorrer no restaurante. Por exemplo, se no futuro o restaurante começar a servir café da manhã ou lanche da tarde, seria facilmente adicionado essas novas refeições utilizando o botão “Adicionar” e parametrizando o horário que elas iriam ocorrer.

## 7.5 Restaurante

O menu “Restaurante” foi desenvolvido para permitir visualizar as refeições realizadas no RU. Todas as refeições importadas no sistema são exibidas nesse menu, portanto esse é o menu onde o usuário pode contemplar todas as refeições realizadas pelos alunos.

O relatório foi desenvolvido de maneira a apresentar todas as informações sobre uma refeição, portanto os atributos que o relatório apresenta são:

- **ID:** Cada refeição possui um número identificador único;
- **Nome do aluno:** O nome do aluno que realizou a refeição;
- **Matrícula do aluno:** O número de matrícula do aluno que realizou a refeição;
- **Bolsa do aluno:** O tipo de bolsa do aluno na época que realizou a refeição, pois um aluno está sujeito a mudar de modalidade de bolsa no decorrer do curso;
- **Curso do aluno:** O curso do aluno, na época que realizou a refeição, pois um aluno está sujeito a mudar de curso ao decorrer da graduação;
- **Data da refeição:** O dia em que o aluno realizou a refeição;
- **Hora da refeição:** A hora em que o aluno realizou a refeição.

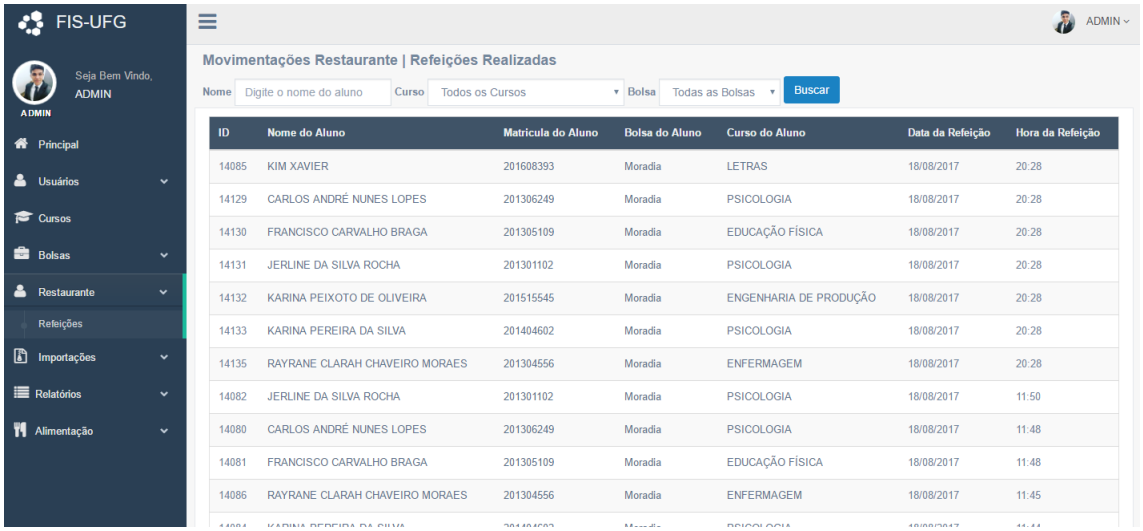
Com o objetivo de tornar o relatório de refeições mais funcional e permitir ao usuário realizar buscas conforme sua necessidade, 3 (três) filtros foram adicionados ao relatório. Portanto, o usuário pode filtrar as refeições por “nome do aluno”, “curso” ou “bolsa”. Na Figura 25 é apresentado o menu Restaurante e o relatório de refeições é descrito.

Ao observar a Figura 25 é possível identificar o relatório de refeições dos alunos, podendo o usuário realizar uma filtragem no relatório de acordo com a sua necessidade. Ao selecionar um aluno, nesse relatório, o usuário é redirecionado para a tela de detalhes do aluno, apresentado na Figura 17.

## 7.6 Importações

O menu “Importações” foi desenvolvido para centralizar todas as funcionalidades que englobasse a importação. No sistema é possível realizar dois tipos de importações, sendo que o usuário pode importar refeições em lote ao sistema através das planilhas de Excel e também pode importar alunos em lote para um cadastro automatizado.

Figura 25 – Tela de refeições realizadas




Fonte: Elaborada pelo autor.

### 7.6.1 Importar Refeições

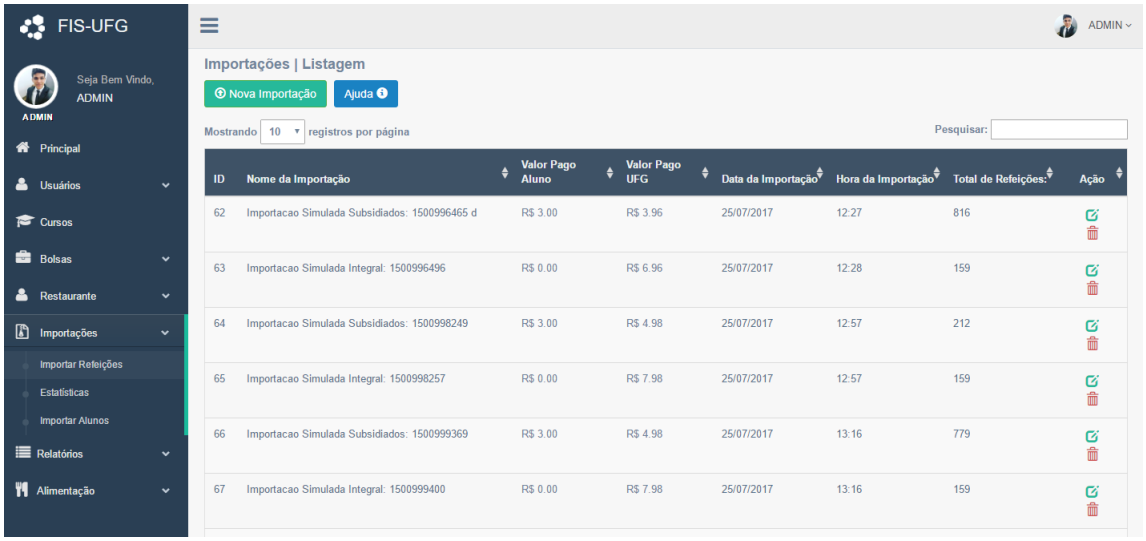
O modulo de importar refeições era uma das poucas funcionalidades que o Fiscaliza-RU.v1 apresentava originalmente. Porém, ele foi implementado de forma pouco elegante e não muito funcional, pois não permitia gerar relatórios ou gerar um histórico das refeições que foram importadas.

Para implementar a possibilidade de relatórios ou históricos, toda a codificação do módulo foi refeita de maneira mais elegante. É dito de maneira elegante uma vez que foi feito o vínculo de cada refeição importada ao ID da importação, permitindo controle de origem de cada refeição importada ao sistema. O código fonte da importação foi divido em funções, para que cada uma realizasse a verificação de uma restrição do sistema, melhorando a legibilidade do código e facilitando a manutenção para os programadores. Os resultados, da implementação desse módulo, podem ser observados na Figura 26.

Na Figura 26 é apresentada a tela de importação de refeições ao sistema. A importação é realizada clicando no botão “Nova Importação” e um modal é exibido, Figura 27, onde o usuário deve preencher alguns dados, selecionar uma planilha do Excel, seguindo um template fornecido com os dados da refeição, e as verificações são realizadas pelo Fiscaliza-RU, de acordo com as regras já apresentadas na seção de “Requisitos”. Ao selecionar o botão “Ajuda” o usuário é redirecionado para uma tela, Figura 29, com todas as informações necessárias para uma importação de sucesso, nessa tela o usuário pode realizar o download do template da planilha do Excel.

O relatório de importações, apresentado na Figura 26, é o histórico de todas as importações realizadas no sistema. O relatório pode ser filtrado de acordo com a necessidade do usuário. O campo “Pesquisar” realiza uma pesquisa em todos os atributos da tabela de forma automática.

Figura 26 – Tela para importar refeições




Fonte: Elaborada pelo autor.

O relatório apresenta alguns atributos básicos, sendo eles:

- **ID:** Cada importação possui um número identificador único;
- **Nome da Importação:** O nome atribuído pelo usuário à importação, para fácil identificação da mesma;
- **Valor Pago Aluno:** O valor pago pelo aluno na época em que a importação foi realizada, pois os valores das refeições estão sujeitos a mudanças;
- **Valor Pago UFG:** O valor pago pela UFG na época em que a importação foi realizada, pois os valores das refeições estão sujeitos a mudanças;
- **Data da Importação:** O dia em que a importação foi realizada;
- **Hora da Importação:** A hora em que a importação foi realizada;
- **Total de Refeições:** A quantidade total de refeições importadas ao sistema.
- **Ação:** Apresenta dois botões, sendo eles:
  - **Editar:** Permite alterar o campo “Nome da Importação”;
  - **Excluir:** Realiza uma exclusão em cascata da importação, ou seja, exclui a importação do sistema e todas as refeições que foram importadas também.

Ao selecionar uma importação do relatório, apresentado na Figura 26, o usuário é redirecionado para uma tela que contém todas as refeições importadas ao sistema naquela importação em específico. A Figura 27 representa o relatório de refeições importadas.

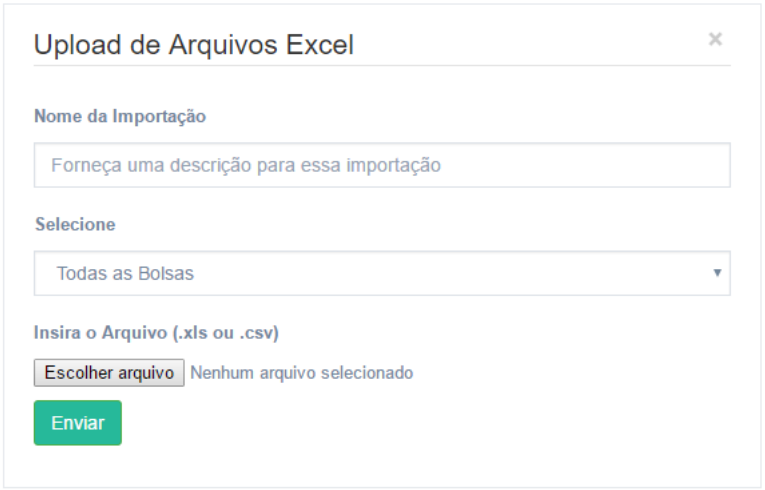
Figura 27 – Tela de refeições importadas

Fonte: Elaborada pelo autor.

Na Figura 27 é mostrado o relatório de refeições importadas em lote durante uma importação em específico. Por exemplo, neste caso é possível ver as refeições que foram importadas pela importação “Importação Integral 28/03”. É trivial imaginar que ao realizar uma importação ao sistema, deseja-se saber quais foram as refeições de fato importadas, qual horário que elas ocorreram e quais foram os alunos. Após a reformulação do Fiscaliza-RU, cada refeição passou a ser vinculada ao ID de uma importação, portanto, dessa forma passou a ser possível saber qual a origem de cada refeição cadastrada no sistema.

Ao clicar no botão “Nova Importação” o modal, apresentado na Figura 28, é exibido ao usuário, que deve preencher todos os campos obrigatoriamente. O campo “Nome da Importação” deve ser de preferência do usuário, com objetivo de identificar a importação com algum nome de fácil recordação. O segundo campo solicita que o usuário identifique de qual tipo são as refeições que ele está importando (subsidiado, integral ou parcial). Por último, o usuário deve inserir a planilha que contém as refeições a serem importadas ao Ficaliza-RU.

Figura 28 – Modal para upload de planilha de refeições




Fonte: Elaborada pelo autor.

Contudo, concluiu-se que a atividade de realizar importações de refeições ao sistema seja trabalhosa e o usuário poderia apresentar dificuldades para realizar a tarefa e, como o foco da reformulação do Fiscaliza-RU foi a usabilidade, foi desenvolvido uma tela de ajuda para orientar o usuário em como realizar as importações de refeições de maneira simples e rápida. A Figura 29 representa a tela de ajuda desenvolvida para este fim.

Figura 29 – Tela de ajuda para importação de refeições

Fonte: Elaborada pelo autor.

Na Figura 29 pode-se verificar que o usuário possui instruções de como deve ser a planilha usada para importar as refeições. Caso o mesmo ainda apresente dúvidas de como deve ser o arquivo, ele pode realizar o download de uma planilha com o modelo a ser seguido, através do botão “Download”.

### 7.6.2 Estatísticas

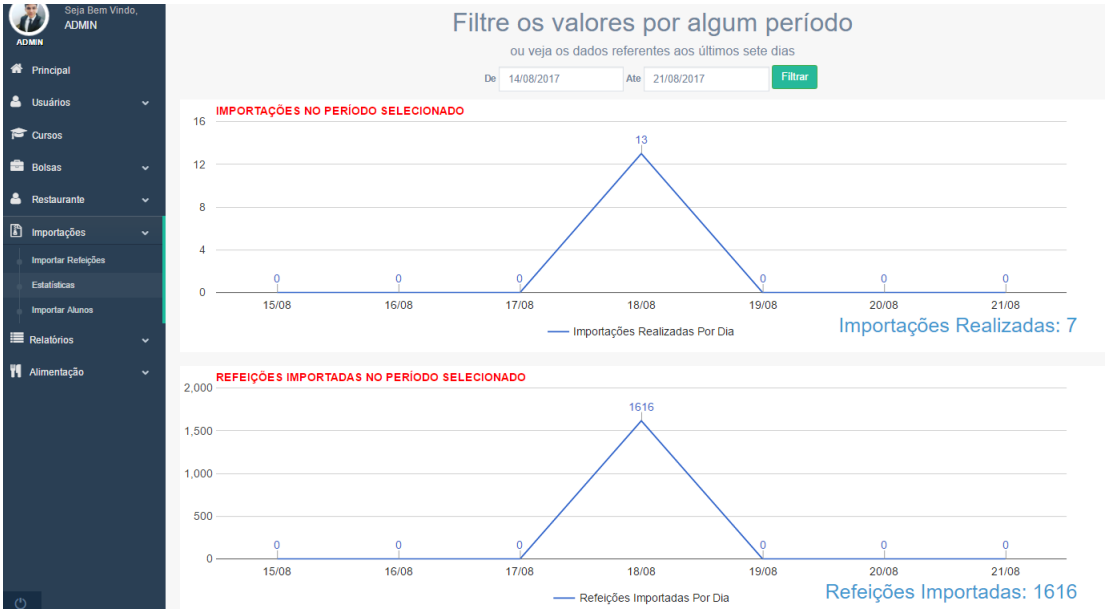
O sub-menu “Estáticas”, dentro do menu “Importação”, permite o usuário ter um relatório de quantas importações e quantas refeições foram importadas ao sistema em determinado período de tempo. A Figura 30 representa os gráficos com as estatísticas de importações.

Na Figura 30 são apresentados dois gráficos de estatísticas de importações e refeições importadas, pois cada importação contém várias refeições sendo importadas, para melhor visualização desses dados o sub-menu de estatísticas foi desenvolvido. O período de tempo, intervalo de data, pode ser filtrado conforme a necessidade do usuário.

### 7.6.3 Importar Alunos

A UFG – RC apresenta um número grande de alunos, sendo inviável o cadastro manual de todos eles para gerenciamento no RU. Atualmente, já estão cadastrados, no Fiscaliza-RU, 3.431 alunos.

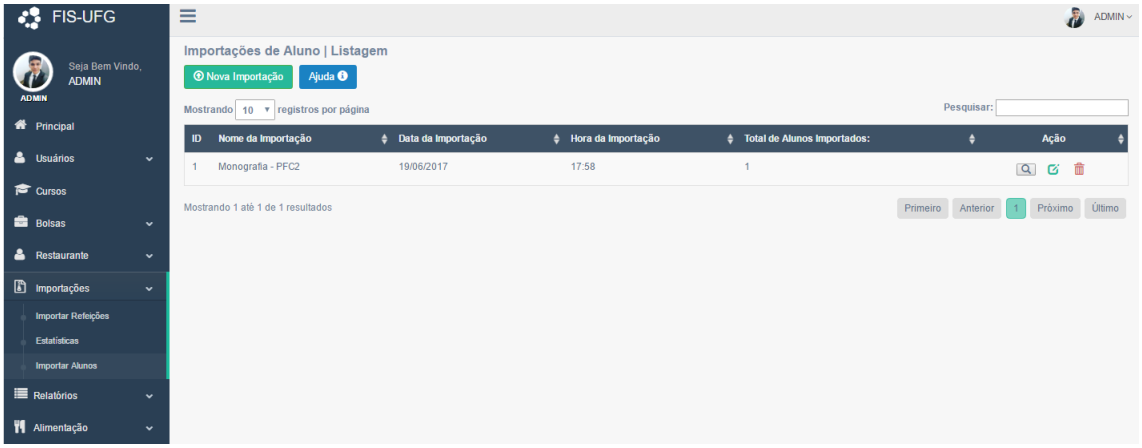
Figura 30 – Estatísticas de Importações




Fonte: Elaborada pelo autor.

O Fiscaliza-RU.v1 também apresentava um módulo para realizar a importação de alunos, porém foi desenvolvido igual ao modulo de importar refeições, de maneira pouco elegante, não provendo suporte para geração de relatórios e históricos. Assim, para padronização das funcionalidades do sistema, uma vez que a reengenharia estava sendo aplicada à versão original do sistema, optou-se pela reformulação desse módulo também. Portanto, toda a codificação do módulo de importação dos alunos foi refeita, com o objetivo de permitir geração de relatórios e históricos de importações. Os resultados podem ser observados na Figura 31.

Figura 31 – Relatório de Importação de alunos




Fonte: Elaborada pelo autor.

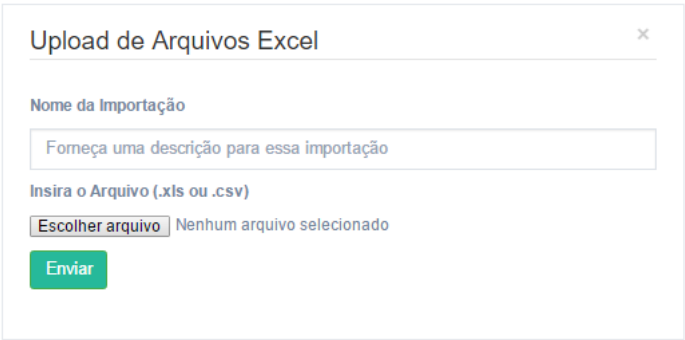
O relatório de importações de alunos, apresentado na Figura 31, é o histórico de todas as importações realizadas no sistema. O relatório pode ser filtrado de acordo com a necessi-

dade do usuário. O campo “Pesquisar” realiza uma pesquisa em todos os atributos da tabela automaticamente. O relatório apresenta alguns atributos básicos, sendo eles:

- **ID:** Cada importação de alunos possui um número identificador único;
- **Nome da Importação:** Nome atribuído pelo usuário a importação, para uma fácil identificação da mesma;
- **Data da Importação:** O dia que a importação foi realizada;
- **Hora da Importação:** A hora que a importação foi realizada;
- **Total de alunos Importados:** O número total de alunos que foram importados;
- **Ação:** Possui 3 botões, sendo eles:
  - **Lupa:** Ver detalhes da importação, possibilita visualizar quais foram os alunos importados e seus dados;
  - **Editar:** Permitir alterar o campo “Nome da Importação”;
  - **Excluir:** Realiza uma exclusão em cascata da importação, ou seja, exclui a importação do sistema e todos os alunos que foram importados também.

Para realizar uma nova importação o usuário deve selecionar o botão “Nova Importação”. Um modal, Figura 32, será exibido ao clicar no botão e o usuário deverá preencher todos os campos obrigatoriamente.

Figura 32 – Modal para upload de planilhas de alunos




Fonte: Elaborada pelo autor.

Ao clicar no botão “Nova Importação” o modal, Figura 32, é exibido ao usuário. Ele deve preencher todos os campos obrigatoriamente. O campo “Nome da Importação” deve ser de preferência do usuário com objetivo de identificar a importação com algum nome de fácil recordação. Por último, o usuário deve inserir a planilha que contém as refeições a serem

importadas ao sistema. O modelo da planilha para importação de alunos pode ser encontrado clicando no botão “Ajuda”, que fica ao lado do botão “Nova Importação”.

Ao selecionar o botão “Ajuda”, no módulo de importação de alunos, o usuário é redirecionado para uma tela de ajuda, Figura 33, onde encontrará todas as informações necessárias para realizar a importação de alunos com sucesso.

Figura 33 – Tela de Ajuda para Importação de alunos

Fonte: Elaborada pelo autor.

A Figura 33 mostra a tela de ajuda do módulo de importar alunos. Nessa tela o usuário possui todas as instruções de como deve ser a planilha usada para importar os alunos. Caso o usuário ainda apresente dúvidas de como deve ser o arquivo, ele pode realizar o download de uma planilha com o modelo a ser seguido, através do botão “Download”.

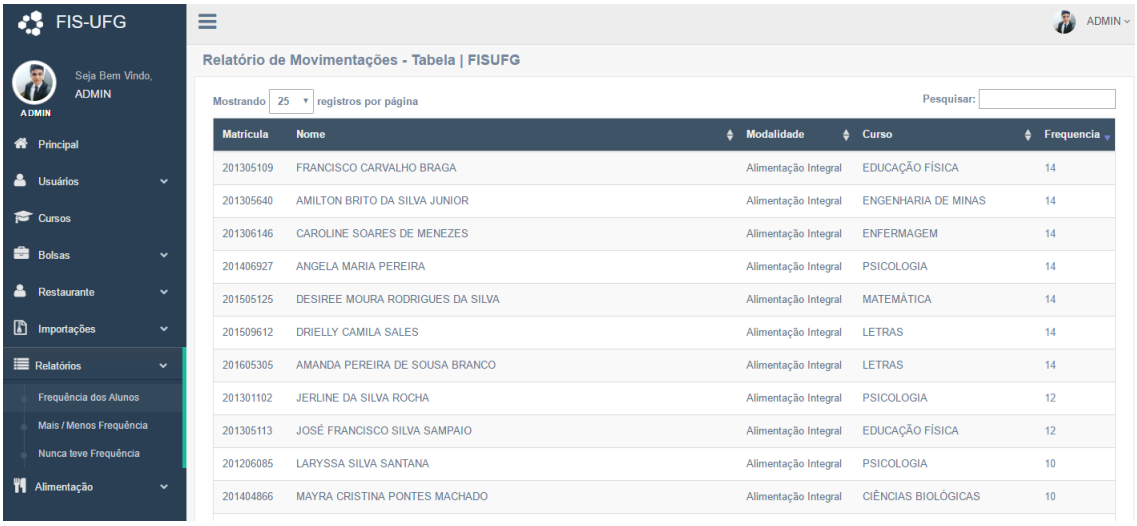
## 7.7 Relatórios

O menu relatórios foi desenvolvido com objetivo de centralizar todos os relatórios em apenas um único menu, sendo apresentados 3 tipos de relatórios: “Frequência dos alunos”, “Mais/Menos Frequências”, “Nunca Teve Frequência”. Esses relatórios têm como objetivo mostrar os alunos com maiores assiduidades e os que nunca apresentaram nenhum consumo no restaurante

### *7.7.1 Frequência dos Alunos*

O relatório de frequência dos alunos possui todos alunos que já apresentam consumo no RU, mensurando quantas vezes ele já frequentou o restaurante, mostrado na Figura 34.

Figura 34 – Frequência dos alunos




Fonte: Elaborada pelo autor.

O relatório é composto por vários atributos, sendo eles:

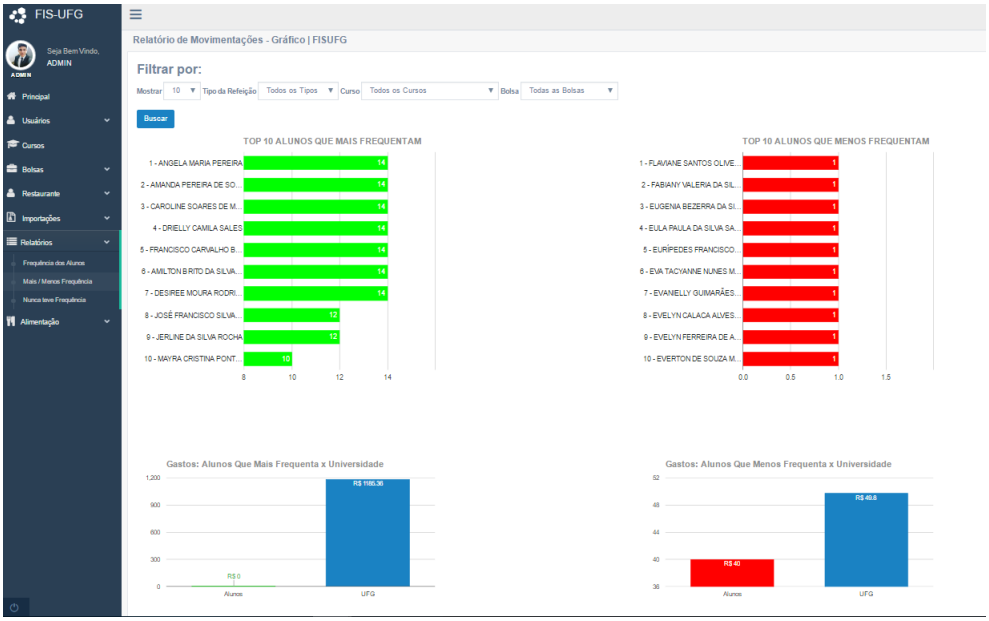
- **Matrícula:** Número da matrícula do aluno;
- **Nome:** Nome do aluno;
- **Modalidade:** Bolsa que o aluno está vinculado;
- **Curso:** Curso que o aluno está vinculado;
- **Frequência:** Quantas vezes o aluno já frequentou o restaurante.

O relatório de frequência dos alunos pode ser filtrado conforme a necessidade do usuário. O campo “Pesquisar” realiza uma pesquisa em todos os campos da tabela. Ao clicar sobre a coluna “Nome” o sistema organiza-os, automaticamente, em ordem alfabética. Se o usuário selecionar a coluna “Modalidade” o sistema, automaticamente, agrupa os alunos por bolsa ou se ele selecionar a coluna “Curso” serão agrupados os alunos por curso. Por último, se o usuário selecionar a coluna “Frequência” o sistema agrupa os alunos com maior/menor frequência, em ordem crescente ou decrescente, respectivamente.

### 7.7.2 Alunos com Frequência Mediana

O relatório de frequência mediana, Figura 35, trabalha da mesma forma que o relatório frequência dos alunos, Figura 34, seu principal objetivo é mensurar a frequência dos alunos ao RU. O diferencial desse relatório é que os dados são mensurados em gráficos, onde o usuário possui os dados mensurados de uma forma diferente. Na Figura 35 são apresentados exemplos de gráficos que podem ser gerados nesse relatório.

Figura 35 – Frequência dos alunos em gráfico




Fonte: Elaborada pelo autor.

A Figura 35 mostra os dados do relatório, sendo composto por 4 gráficos: 10 alunos que mais frequentam o RU e os gastos que esses alunos e a UFG tiveram, 10 alunos que menos frequentam e os gastos que esses alunos e a UFG tiveram. Os gráficos podem exibir até 30 resultados, de acordo com a opção selecionada também podem ser filtrados como "Tipo da Refeição" (almoço ou jantar), "Curso" ou "Bolsa".

### 7.7.3 Alunos que nunca frequentaram

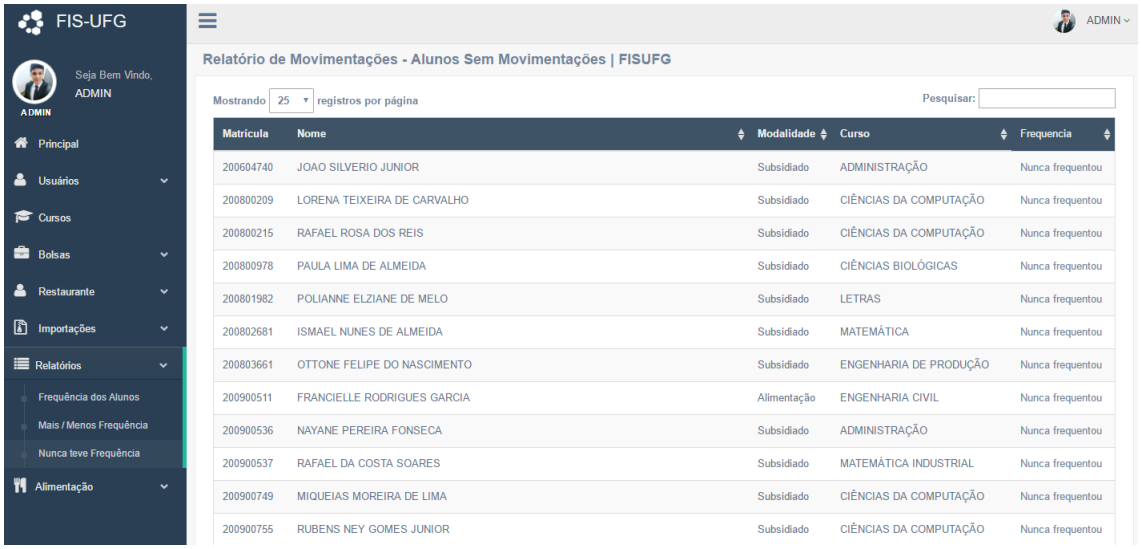
O relatório dos alunos que nunca frequentaram possui todos alunos que nunca apresentam consumo no RU. Na Figura 36 é representado o relatório de alunos que nunca frequentaram o restaurante.

O relatório é composto por vários atributos sendo eles:

- **Matrícula:** Número da matrícula do aluno;
- **Nome:** Nome do aluno;
- **Modalidade:** Bolsa que o aluno está vinculado;
- **Curso:** Curso que o aluno está vinculado;
- **Frequência:** Quantas vezes o aluno foi ao RU, que neste caso é nula.

O relatório de "Nunca teve frequência" dos alunos pode ser filtrado conforme a necessidade do usuário. O campo "Pesquisar" realiza uma pesquisa em todos os campos da tabela. Ao

Figura 36 – Relatório de alunos que nunca frequentaram

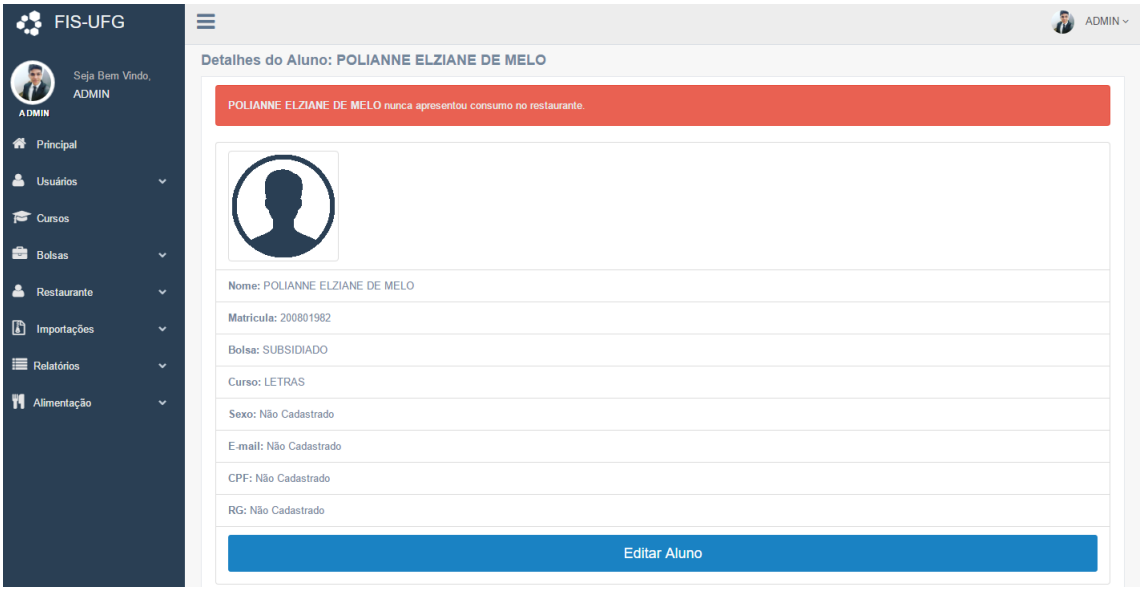



Fonte: Elaborada pelo autor.

clicar sobre a coluna “Nome” o sistema organiza em ordem alfabética. Se o usuário selecionar a coluna “Modalidade” o sistema agrupa os alunos por bolsa ou se ele selecionar a coluna “Curso” serão agrupados os alunos por curso, todos são feitos automaticamente.

Ao selecionar um aluno, o Fiscaliza-RU, por padrão, leva o usuário à tela de detalhes do aluno, porém, como o usuário nunca apresentou consumo no restaurante não existirá dados para serem detalhados. Assim um alerta será exibido, mostrado na Figura 37, informando ao usuário de que aquele aluno nunca apresentou consumo ao restaurante.

Figura 37 – Alerta de um usuário que nunca apresentou consumo




Fonte: Elaborada pelo autor.

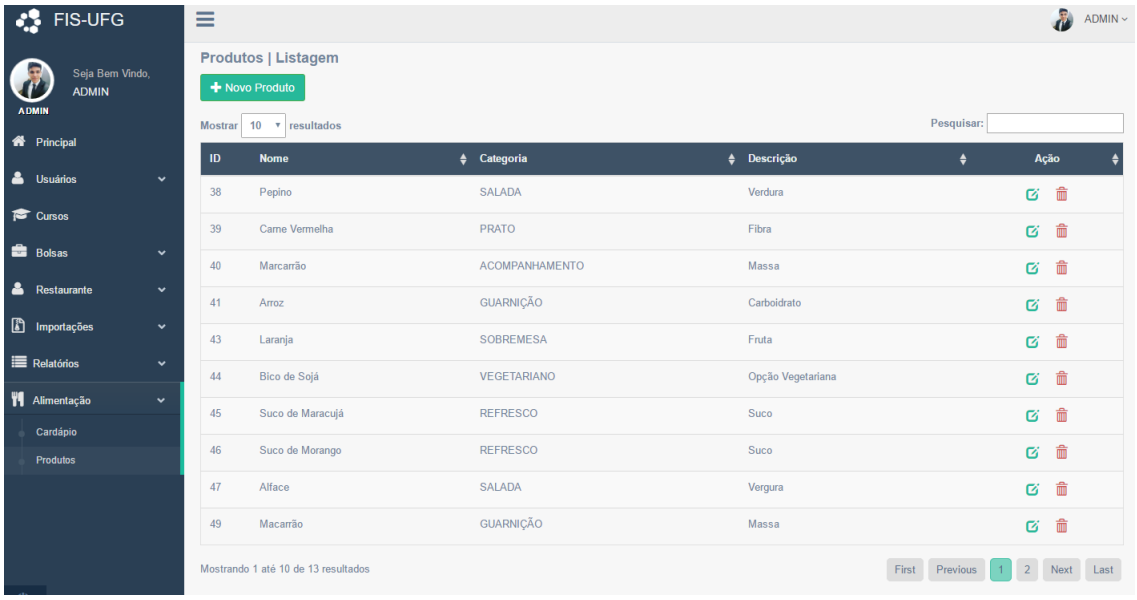
## 7.8 Alimentação

O menu alimentação foi desenvolvido para centralizar as funcionalidades relacionadas a alimentação dos alunos e cardápio. Como já abordado na seção “Fora de Escopo”, no Fiscaliza-RU não foi implementada nenhuma funcionalidade para controle de qualidade ou avaliação do cardápio, porém o módulo está preparado para essa implementação no futuro. O módulo alimentação implementa apenas a possibilidade do cadastro de cardápio no sistema e dos produtos que irão compor o cardápio.

### 7.8.1 Produtos

O cardápio é composto por vários alimentos, que foram tratados como produtos no desenvolvimento dessa pesquisa, podendo pertencer a uma categoria diferente. Todos os produtos estão disponibilizados ao cadastrar um novo cardápio de acordo com sua categoria, o usuário pode selecionar algum produto para realizar uma edição ou exclusão. Na Figura 38é exibido um relatório de produtos cadastrados no sistema.

Figura 38 – Produtos




Fonte: Elaborada pelo autor.

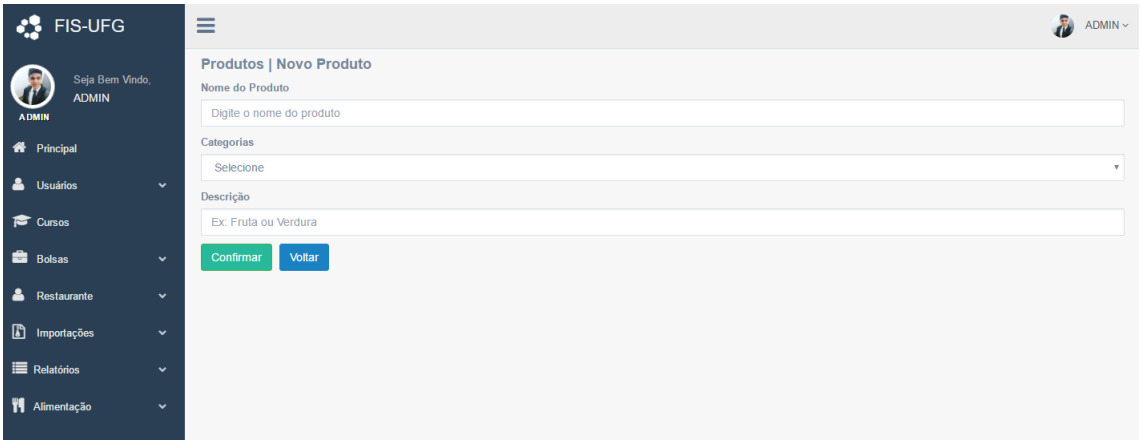
O relatório de produtos cadastrados é o histórico de todas os produtos cadastrados no sistema, podendo ser filtrado de acordo com a necessidade do usuário. O campo “Pesquisar” realiza uma pesquisa em todos os atributos da tabela. O relatório apresenta alguns atributos básicos, sendo eles:

- ID: Cada produto é representado por um número único;
- Nome: Nome atribuído ao produto;

- Categoria: Categoria que o produto está vinculado;
- Descrição: Campo, não obrigatório, aberto ao usuário para escrever alguma observação;
- Ação: Composto por dois botões:
  - Editar: Permitir alterar os atributos do produto;
  - Excluir: Realiza a exclusão do produto do sistema.

A adição de novos produtos ao sistema pode ser realizada selecionando o botão “Novo Produto” que irá redirecionar o usuário a uma nova tela Figura 39.

Figura 39 – Novo Produto




Fonte: Elaborada pelo autor.

A figura 36 mostra a tela de adicionar um novo produto ao sistema, onde o usuário deve, obrigatoriamente, preencher os campos “Nome do Produto” e “Categorias”. O campo “Descrição” é opcional e ele deve preencher conforme sua necessidade.

### 7.8.2 Cardápio

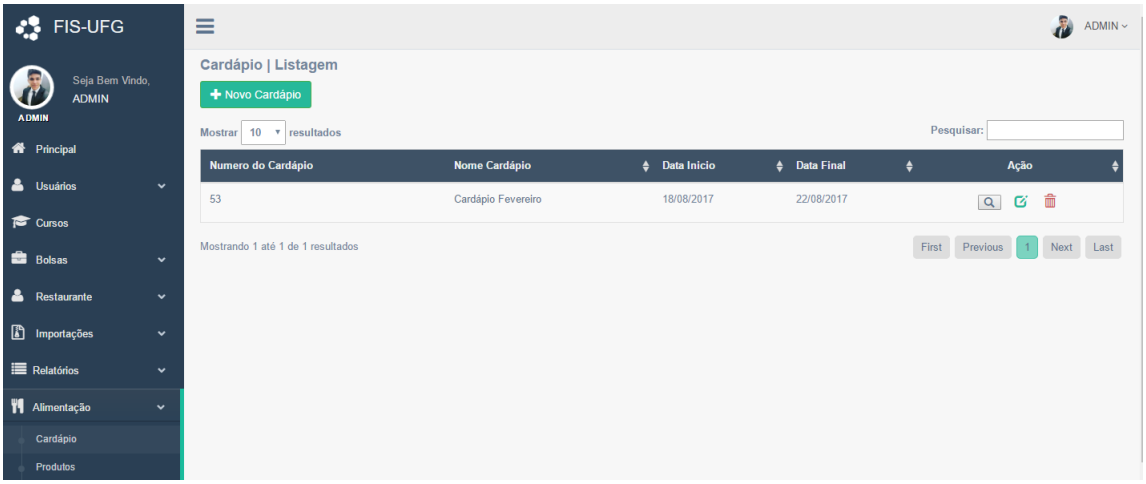
O módulo cardápio foi desenvolvido para funcionar de maneira inteligente e o usuário necessitar realizar o cadastro apenas uma vez, evitando retrabalho e otimizando as atividades do usuário. A empresa terceira planeja o cardápio do mês e repassada à Universidade, assim o usuário do sistema pode realizar o cadastro do cardápio com data de validade, dessa forma cadastrando o mês inteiro.

O relatório de cardápios, Figura 40 é o histórico de todos os cardápios cadastrados no sistema, podendo ser filtrado de acordo com a necessidade do usuário. O campo “Pesquisar” realiza uma pesquisa em todos os atributos da tabela automaticamente. O relatório apresenta alguns atributos básicos, sendo eles:

- **Número do Cardápio:** Cada cardápio é representando por um número único;

- **Nome do Cardápio:** Nome atribuído ao cardápio, para ser referenciado facilmente;
- **Data Início:** Data de início da validade do cardápio;
- **Data Final:** Data de fim da validade do cardápio;
- **Ação:** Composta por 3 botões, sendo eles:
  - **Lupa:** Visualiza os produtos vinculados ao cardápio;
  - **Editar:** Altera os produtos vinculados ao cardápio;
  - **Excluir:** Remove o cardápio do sistema.

Figura 40 – Tela de Cardápios




Fonte: Elaborada pelo autor.

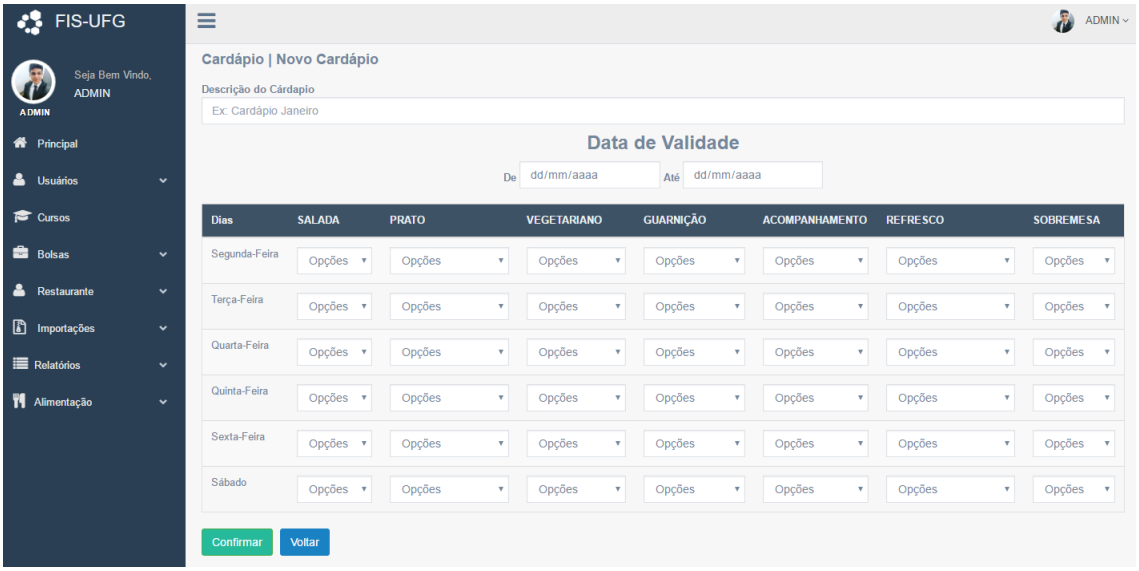
Na Figura 40 é mostrada uma relação dos cardápios cadastrados no sistema. O usuário pode selecionar algum cardápio para realizar uma edição ou exclusão. A adição de novos cardápios ao sistema pode ser realizada selecionando o botão “Novo Cardápio” que irá redirecionar o usuário a uma nova tela Figura 41.

Pode-se visualizar, na figura para adicionar um novo cardápio ao sistema, que o usuário, obrigatoriamente, deve preencher os campos “Descrição do Cardápio” e “Data de Validade”. Os campos de “Segunda-feira” até “Sexta-feira” e as categorias de “Salada” até “Sobremesa” podem ser preenchidos conforme a necessidade do usuário.

Ao cadastrar um cardápio o sistema permite que o usuário visualize-o em uma nova tela, Figura 42. Esta tela possui dois botões, “Novo Cardápio” e “Editar Cardápio”.

Ao selecionar o botão de edição, o usuário é redirecionado para uma tela que lhe permite alterar os dados cadastrados no cardápio. Ao selecionar o botão de novo cardápio o comportamento é um pouco diferente do que o apresentado na Figura 40, pois irá redirecionar a tela de novo cardápio, porém com todos os produtos pré-selecionados, de acordo com o cardápio

Figura 41 – Tela para novo cardápio




Fonte: Elaborada pelo autor.

Figura 42 – Tela para visualizar cardápio cadastrado

Fonte: Elaborada pelo autor.

que estava sendo visualizado, fica a cargo do usuário apenas alterar os poucos produtos que se diferem de um cardápio para outro. Esse comportamento é para ajudar na usabilidade do programa e evitar retrabalho do usuário ao criar um novo cardápio, uma vez que muitos produtos se repetem diariamente no cardápio. Por exemplo, arroz e feijão, são produtos que quase sempre estão presentes e, assim, o usuário poderá gerar um novo cardápio a partir de outro.

Os cardápios cadastrados no sistema, forma implementados com principal objetivo de fornecer aos alunos uma forma de visualização fácil do cardápio da semana. Para que esse objetivo fosse completo, uma página web pública do sistema foi disponibilizada, para que os alunos pudessem visualizar o cardápio sem a necessidade de autenticação. O sistema exibe na página pública, Figura 43, o cardápio correspondente ao que estiver dentro do período de

validade, de acordo com a data que aluno estiver acessando.

Figura 43 – Página Pública de visualização do cardápio

**Cardápio Fevereiro**

**Válidade:**18/08/2017 até 22/08/2017

| Dias | SALADA | PRATO | VEGETARIANO | GUARNIÇÃO | ACOMPANHAMENTO | REFRESCO | SOBREMESA |
|---|---|---|---|---|---|---|---|
| Segunda | Pepino | Frango | Bico de Soja | Arroz | Marcarrão | Suco de Maracujá | Laranja |
| Terça | Alface | Carne Vermelha | Bico de Soja | Arroz | Marcarrão | Suco de Morango | Laranja |
| Quarta | Alface | Carne Vermelha | Bico de Soja | Arroz | Marcarrão | Suco de Morango | Laranja |
| Quinta | Alface | Carne Vermelha | Bico de Soja | Macarrão | Marcarrão | Suco de Maracujá | Laranja |
| Sexta | Alface | Carne Vermelha | Bico de Soja | Macarrão | Marcarrão | Suco de Morango | Laranja |
| Sábado | Vazio | Vazio | Bico de Soja | Arroz | Marcarrão | Suco de Maracujá | Laranja |

Fonte: Elaborada pelo autor.

Caso nenhum cardápio tenha sido cadastrado ou não exista nenhum cardápio para a data que o usuário estiver acessando, a tela pública irá alertar que nenhum cardápio foi cadastrado ainda para aquela data, Figura 44.

Figura 44 – Página Pública sem Cardápio

Fonte: Elaborada pelo autor.

Na Figura 44 é apresentado como a página irá se comportar caso um usuário acesse a página para visualizar o cardápio e não houver cadastrado ou o período solicitado estiver fora da data de validade.

Na próxima subseção são apresentados alguns aspectos considerados para a remodelagem e reengenharia do Fiscaliza-RU.v1 para o Fiscaliza-RU.

## 7.9 Sobre

O menu “Sobre” foi desenvolvido para identificar os responsáveis pelo desenvolvimento da pesquisa e o consequente produto, conforme representado na Figura 45.

Na Figura 45 é apresentado uma breve descrição sobre os responsáveis pelo projeto, sendo eles o aluno Marco Túlio Macedo Rodrigues, como responsável pela pesquisa atuando

Figura 45 – Menu Sobre

Fonte: Elaborada pelo autor.

como desenvolvedor do sistema Fiscaliza-RU, sob orientação da Prof.ª Dr.ª Luanna Lopes Lobato e co-orientação do Prof. Dr. Thiago Jabur Bittar.

## 7.10 Remodelagem e Reengenharia

A reformulação do Fiscaliza-RU.v1 foi pensada de maneira a melhorar sua usabilidade, legibilidade e funcionalidade de código, desempenho das consultas ao BD, bem como criação de novas funcionalidades ainda não existentes na versão inicial. Assim, muitas alterações foram realizadas nos códigos e algumas otimizações foram implementadas, sendo apresentado neste capítulo exemplos de modificações na estrutura do sistema e códigos. É importante ressaltar que não são mostradas todas as mudanças em código e nem todas as decisões tomadas referentes à arquitetura do sistema, uma vez que o número de essas alterações e decisões é relativamente alto.

### *7.10.1 7.10.1 Estrutura, Linguagem e FrameWork*

O fiscaliza-RU.v1 foi desenvolvido originalmente já utilizando o framework Laravel e este conceito foi continuado durante a reformulação do mesmo. A arquitetura MVC já é oferecia pela própria ferramenta e faz parte de como ela trabalha e executa suas rotinas, portanto, nenhuma outra arquitetura foi adotada.

A linguagem de desenvolvimento, assim como o próprio framework citado anteriormente, é a mesma: PHP. Não foram realizadas modificações nesses aspectos estruturais do software, adotando-se o que já tinha sido desenvolvido originalmente.

A arquitetura MVC era oferecida ao desenvolvedor do Fiscaliza-RU-v1 desde o princípio, porém, foi identificado que a estrutura não era seguida. O desenvolvimento das classes era realizado de maneira manual violando os conceitos da arquitetura MVC. Portanto, durante a reformulação foi necessário remodelar todas as models do sistema de maneira que elas seguissem o conceito da arquitetura imposta pelo framework.

Durante a remodelagem das models foi informado ao sistema com quais classes a model se relacionava, dessa forma os objetos instanciados de cada classe implementavam métodos mais robustos. Conforme apresentado na Figura 46, alguns relacionamentos são adicionados a model: UsuariosRest (alunos).

Figura 46 – Relacionamentos na Model UsuariosRest

```
public function movimentacao(){
    return $this->hasMany('App\Models\Movimentacoes', 'matricula_aluno', 'matricula_usuario_rest');
}
public function bolsa(){
    return $this->belongsTo('App\Models\TpBolsa', 'id_tp_bolsa');
}

public function curso(){
    return $this->belongsTo('App\Models\Cursos', 'id_curso_grad');
}

public function refeicao(){
    return $this->hasMany('App\Models\PrecosRefBolsas', 'id_tp_bolsa' ,'id_tp_bolsa');
}
```

Fonte: Elaborada pelo autor.

A model UsuariosRest é responsável por fazer a modelagem dos alunos no sistema. Conforme apresentado na Figura 46, foram adicionados os seguintes relacionamentos ao modelo:

- **Movimentação:** Adiciona o relacionamento, hasMany (possui várias), com a model Movimentação, fazendo amarração entre os atributos ‘matricula_aluno’ da model UsuariosRest e o atributo ‘matricula_usuario_rest’ da model Movimentação.
- **Bolsa:** Adiciona o relacionamento, belongsTo (possui apenas um), com a model Bolsa, fazendo amarração entre os atributos ‘’id_tp_bolsa’ que se repete nas duas models.
- **Curso:** Adiciona o relacionamento, belongsTo (possui apenas um), com a model Bolsa, fazendo amarração entre os atributos ‘’id_tp_bolsa’ que se repete nas duas models.
- **Refeição:** Adiciona o relacionamento, hasMany (possui várias), com a model Refeição, fazendo amarração entre os atributos ‘’id_tp_bolsa’ que se repete nas duas models. Um aluno pode possuir várias refeições, porem as restrições foram implementadas manualmente, para que o aluno não possa ter refeições duplicadas ou fora do horário.

### 7.10.2 SoftDeletes

A model UsuariosRest passou implementar uma funcionalidade chamada pelo framework como softDeltetes (Deletes Leves). Isso quer dizer que todos os objetos instanciados de Usuari-

osRest ao utilizar o método, ‘delete , herdado do Eloquent ORM não serão deletados do sistema, mas inativados.

A tabela de referente a model é a ‘usuarios_rest’, nela existe um atributo chamado ‘delete_at’ que indica quando um dado foi inativado. Quando a model está configurada para usar softDeletes em vez de deletar o registro da tabela o campo é preenchido.

A grande vantagem de se usar o Eloquent ORM junto ao SoftDeletes é devido ao fato de que em todos os lugares onde as consultas são realizadas utilizando os métodos herdados do Eloquent, eles automaticamente irão retornar apenas os dados ativos.

Pois em uma consulta do Eloquent ele realiza uma verificação, onde: ‘deleted_at’ é diferente de ‘null’. Dessa forma os métodos sabem diferenciar quais são os dados ativos ou inativos do sistema.

Para trabalhar com os dados inativos e ativos, alguns métodos são oferecidos, onde o desenvolvedor pode manipular essas informações facilmente, segue eles:

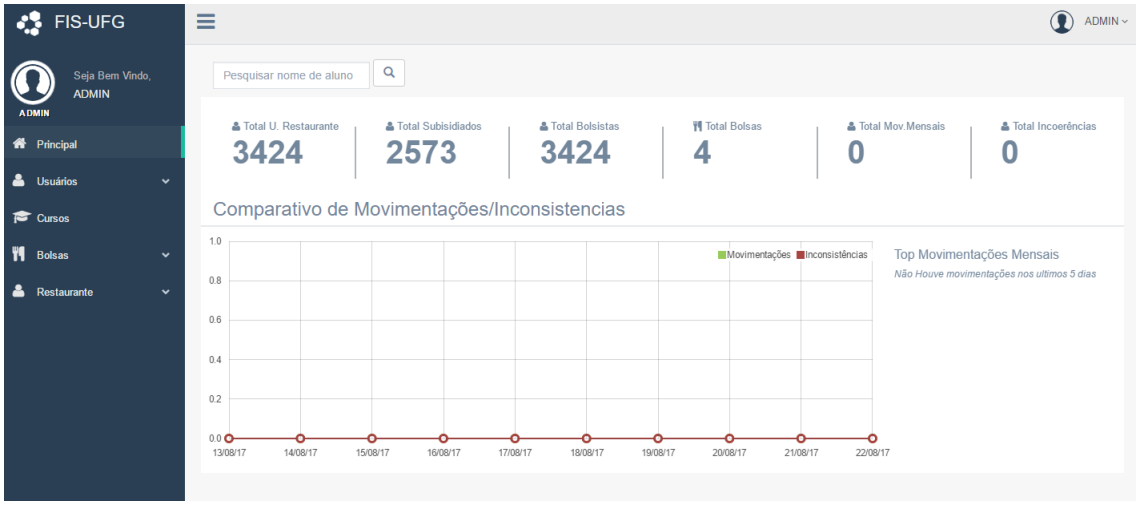
- **All:** retorna apenas os dados ativos do sistema;
- **OnlyTrashed:** Retorna apenas os dados inativos do sistema;
- **WithTrashed:** Retorna todos os dados do sistema, ativos e inativos;

A vantagem de usar Eloquent é que ele oferece todos esses métodos prontos ao desenvolvedor, sem que ele precise desenvolver nada, apenas usar. Todos esses recursos eram ofertados pelo framework em sua versão original e não eram utilizados.

### *7.10.3 Home*

Uma modificação inicial, no Fiscaliza-RU.v1 para o Fiscaliza-RU, versão deste trabalho, pode ser observada ao compararmos a tela home do software antes e depois da reformulação, conforme apesentado na Figura 47 e Figura 48.

Figura 47 – Tela Home do Fiscaliza-RU.v1




Fonte: Elaborada pelo autor.

Figura 48 – Tela Home do Fiscaliza-RU, versão atual

Fonte: Elaborada pelo autor.

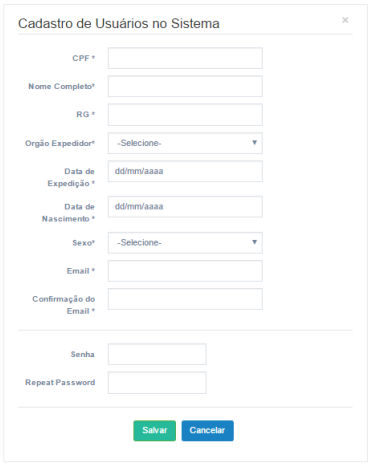
A principal diferença visual, entre as duas telas, é referente a grande quantidade de menus novos que foram adicionados ao novo sistema. Isso significa que novas funcionalidades foram desenvolvidas e o usuário possui novas opções de utilização do software, já que antes essas não estavam disponíveis na versão inicial do sistema.

Ainda, a forma como os gráficos são implementados foi totalmente modificada, para que novos estilos de gráficos fossem possíveis. Os gráficos passaram a ser implementados por uma aberta, chamada Google Charts. A documentação da API pode ser encontrada em *https://developers.google.com/chart*.

### 7.10.4 Novos Usuários Reformulados

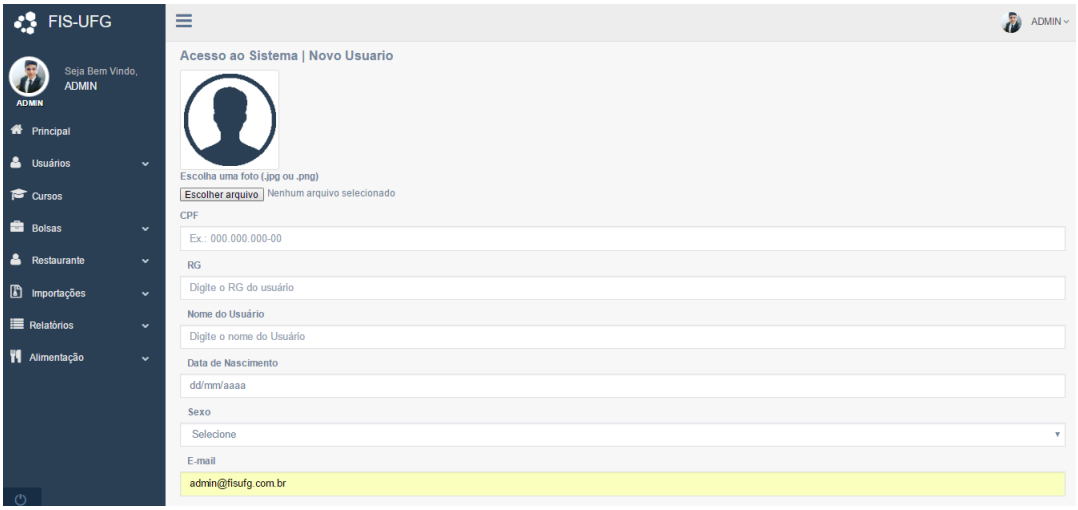
A opção de adicionar novos usuários ao sistema era totalmente ineficiente, pois eram exigidos muitos campos que não apresentavam nenhuma utilidade, sendo que o formulário era apresentando em um modal, Figura 49. Após a reformulação, o formulário passou a ser apresentado em uma página, Figura 50, apresentando apenas campos relevantes ao funcionamento do sistema.

Figura 49 – Adicionar novo nsuário (Fiscaliza-RU.v1)




Fonte: Elaborada pelo autor.

Figura 50 – Adicionar novo nsuário (Fiscaliza-RU)



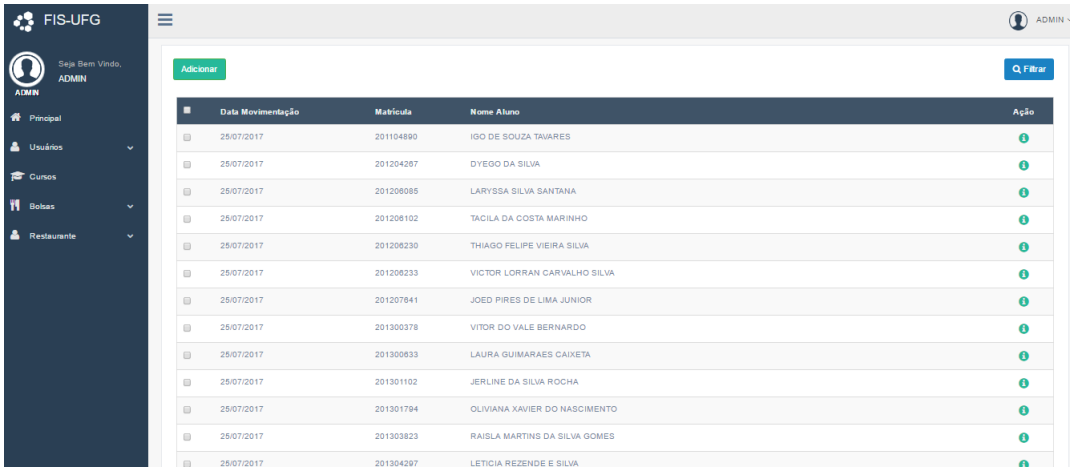

Fonte: Elaborada pelo autor.

É possível notar, que no Fiscaliza-RU.v1, versão antiga, o sistema não implementava a opção de foto de perfil, essa opção começou a ser possível apenas após a reformulação, estando disponível no Fiscaliza-RU, considerando o desenvolvimento do RF36.

### 7.10.5 Importação Reformulada

O Fiscaliza-RU.v1, originalmente, não possuía um modulo de importação dedicado apesar para apresentar a opção de realizar importações. As importações eram feitas na mesma tela do relatório de refeições, não sendo indexadas, portanto não era possível identificar quantas ou quais importações ocorreriam no sistema. As refeições também não eram vinculadas às importações, portanto não era possível relacionar as refeições importadas e qual era a sua importação de origem. Na Figura 51 é representado o relatório de importações do sistema antigo, enquanto que na Figura 52 é mostrada a evolução do sistema após sua reformulação.

Figura 51 – Relatório de importações (Fiscaliza-RU.v1))




Fonte: Elaborada pelo autor.

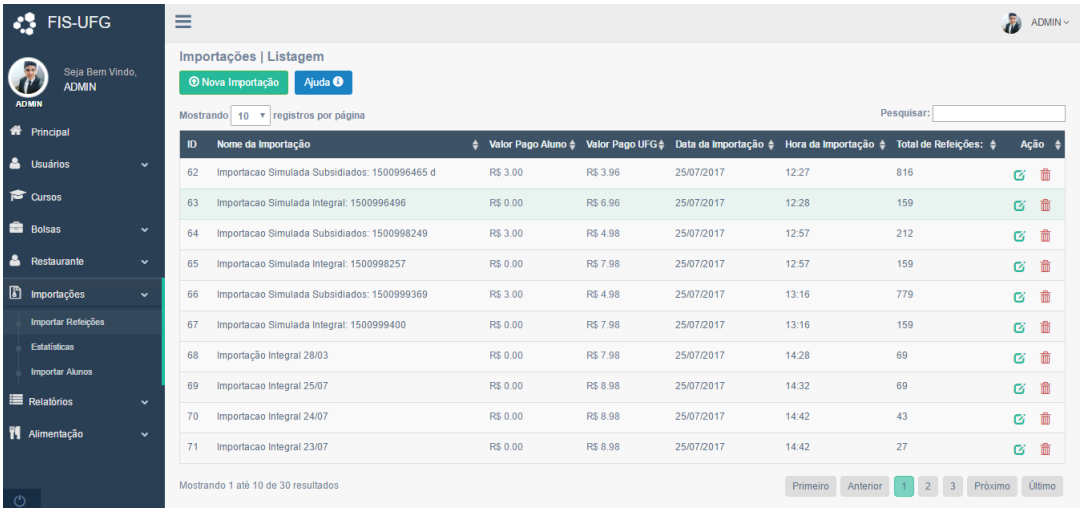
Na Figura 51 é apresentado o relatório de importações do Fiscaliza-RU.v1. É possível observar que no relatório eram identificadas apenas as refeições importadas, não era implementado nenhum tipo de controle sobre quais importações haviam ocorrido e qual era a origem das refeições apresentadas no relatório.

Em contrapartida, na Figura 52, é mostrado o relatório de importações no Fiscaliza-RU, sendo possível verificar que o relatório guarda um histórico de todas as importações que ocorreram no sistema. Com objetivo de melhorar a usabilidade, foram adicionados ícones sugestivos ao botão de nova importação e ao botão de ajuda. Ao selecionar alguma importação nesse relatório o usuário é redirecionado a outra tela detalhando quais foram as refeições importadas.

### 7.10.6 Erros de Importação

Uma importante mudança que houve no comportamento do sistema ao realizar as importações após sua reformulação é que elas passaram a ser absolutas. No Fiscaliza-RU.v1, ao realizar uma importação, analisando uma planilha de refeições, as refeições que não apresentavam inconsistências eram adicionadas ao sistema e apenas aquelas inconsistentes eram alertadas ao usuário

Figura 52 – Relatório de importações (Fiscaliza-RU))

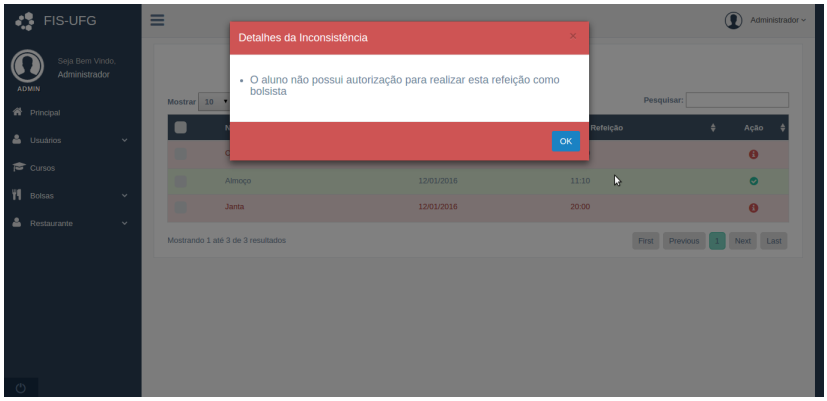



Fonte: Elaborada pelo autor.

e descartadas. Após, a reformulação, no Fiscaliza-RU, as importações começaram a ocorrer de forma absoluta, sendo que o sistema analisa todas as refeições da planilha e adiciona todas caso nenhuma inconsistência seja encontrada. Se algum erro for encontrado o sistema alerta o usuário e não adiciona nenhuma refeição ao BD enquanto houver alguma refeição inconsistente na planilha, nem as corretas.

As inconsistências que podem ocorrer em uma planilha com várias refeições são muitas, conforme já discutido na seção Restrições do Sistema. O software em sua versão original mostrava os erros em um pequeno modal, Figura 53, o que não seria problema ao tentar realizar uma importação onde sejam encontrados poucos erros. Porém, o fluxo de alunos no restaurante é muito grande, aproximadamente 1.200 refeições diárias, segundo a CCOM, e observou-se que quando ocorriam muitos erros, estes não ficavam claros ao usuário utilizando o modal. Na Figura 53 é mostrado como os erros de importação eram apresentados no Fiscaliza-RU.v1.

Figura 53 – Erros de importações (Fiscaliza-RU.v1)




Fonte: Elaborada pelo autor.

Na Figura 53, os erros eram apresentados em um pequeno modal, que era limitado e insuficiente quando uma planilha apresentava muitas linhas com erro, dificultando a visualização.

Com base nesse problema, na reformulação do sistema, foi identificada a necessidade de uma nova forma de apresentar os erros. Assim, no Fiscaliza-RU todos os erros são exibidos em uma página, devidamente separados e identificados por tipo (por exemplo, refeição duplicada ou fora de horário, aluno cadastrado com a bolsa errada, dentre outros). Na Figura 54 é apresentado como os erros passaram a ser representados na versão reformulada do sistema.

Figura 54 – Erros de importações (Fiscaliza-RU)

Fonte: Elaborada pelo autor.

O Fiscaliza-RU agora trata os erros de importações de maneira a deixar as inconsistências mais claras e fáceis de serem identificadas para serem solucionadas, Figura 54. Para tanto, utilizou-se um alerta vermelho, para informar ao usuário que nenhuma importação da planilha foi realizada, comportamento de importações absolutas. Cada problema identificado na planilha é destacado em amarelo e os alunos que estão apresentando o respectivo problema são listados logo abaixo. Como uma página possui a barra de rolagem, o número de inconsistências em uma importação não irá dificultar a visualização pelo usuário de quais são os problemas que o sistema está alertando, ou seja, com muitos ou poucos o usuário poderá identificar para tratar cada inconsistência observada.

### 7.10.7 Querys com Eloquent

O Larável é um framework robusto, sendo desenvolvido para que programadores construir aplicações de forma rápida e segura, sendo suas características apresentadas na seção “Ambiente de Desenvolvimento”, bem como conceitos sobre o Eloquent. O Eloquent implementa muitos métodos e todas as suas subclasses herdam seus métodos, permitindo o programador trabalhar de forma homogênea em seus códigos.

Neste capítulo são exibidas algumas modificações relevantes que foram feitas no código fonte utilizando o Eloquent, de modo a tornar algumas querys mais legíveis. O código fonte 1

exemplifica como é feita uma consulta ao BD em Larável sem o uso do Eloquent, no sistema em sua versão inicial, Figura 55.

Figura 55 – Código Fonte 1. Consulta sem Eloquent no Fiscaliza-RU.v1

```
$data =  DB::table('usuarios_rest')
        ->leftjoin('cursos_grad','usuarios_rest.id_curso_grad', '=','cursos_grad.id_curso_grad')
        ->leftjoin('tp_bolsa','usuarios_rest.id_tp_bolsa', '=','tp_bolsa.id_tp_bolsa')
        ->select('usuarios_rest.id_usuario_rest', 'usuarios_rest.nome_usuario_rest', '
          usuarios_rest.cpf_usuario_rest','tp_bolsa.nome_bolsa',
                  'usuarios_rest.rg_usuario_rest','usuarios_rest.dt_exp_usuario_rest','
                    usuarios_rest.org_exp_usuario_rest',
                  'usuarios_rest.dt_nasc_usuario_rest','usuarios_rest.email_usuario_rest','
                    usuarios_rest.matricula_usuario_rest',
                  'usuarios_rest.id_tp_bolsa','usuarios_rest.id_curso_grad','cursos_grad.nome_curso')
->where('usuarios_rest.nome_usuario_rest', 'like', '%'.$nome_usuario_rest.'%')
->where('tp_bolsa.id_tp_bolsa', 'like', '%'.$bolsa_selecionada.'%' )
->where('cursos_grad.cod_curso', 'like', '%'.$curso_selecionado.'%')
->orderBy('usuarios_rest.nome_usuario_rest', 'asc')
->whereNull('usuarios_rest.deleted_at')
->paginate(50);
```

Fonte: Elaborada pelo autor.

No código fonte 1, do Fiscaliza-RU.v1, é mostrada uma consulta SQL ao BD MySql, sem utilizar nenhum método do Eloquent. O que torna a query muito grande e difícil de ler, ferindo a legibilidade do código.

No Fiscaliza-RU, versão deste trabalho, essa query foi totalmente substituída pelo código fonte 2.

Figura 56 – Código Fonte 2. Consulta utilizando Eloquent no Fiscaliza-RU

```
$data = UsuariosRest::where('nome_usuario_rest', 'like', '%'.$nome_usuario_rest.'%')
        ->where('id_tp_bolsa', 'like', '%'.$bolsa_selecionada.'%' )
        ->where('id_curso_grad', 'like', '%'.$curso_selecionado.'%')
        ->paginate(50);
```

Fonte: Elaborada pelo autor.

No código fonte 2 é apresentada uma solução mais elegante, em termos de redigibilidade e legibilidade do código. O retorno das duas querys são os mesmos, porém apresentam certas sutilezas ao serem destacadas.

A consulta realizada no código fonte 1, Fiscaliza-RU.v1, é extensa e grande, pois realiza uma consulta ao BD, buscando as informações diretamente na tabela 'usuarios$_rest'$, $que por sua vez no possui to$

Enquanto que a consulta no código fonte 2, Fiscaliza-RU, é reduzida, pois está realizando a busca utilizando a classe 'UsuariosRest', através dos métodos do Eloquent que em vez de retornar um array de dados, como no primeiro caso, retorna um array de objetos da classe 'UsuariosRest'. Ao possuir um array de objetos a vantagem é muito maior do que um array de dados. Uma vez que a quantidade de informações contida no array será menor, pois os objetos para retorna suas informações utilizam métodos, enquanto que, o array de dados todas as informações possíveis são retornadas até mesmo aquelas não desejadas ou utilizadas.

Várias outras consultas foram otimizadas com o uso do Eloquent, melhorando o desempenho nas buscas a dados feitas pelo sistema, uma vez que o Fiscaliza-RU.v1 utilizava o SQL em sua forma mais básica em todas as consultas. Após a reformulação do sistema grande parte das consultas passaram a serem realizadas com o Eloquent, tornando os códigos mais rendíveis e legíveis ao programador.

Com isso, pode-se concluir que a reformulação do sistema não trouxe apenas benefícios de usabilidade ao usuário final, mas trouxe também vantagens e melhorias para o programador e futuros pesquisadores que darão continuidade a esta pesquisa.

### 7.10.8 Optimização das importações

As importações realizadas no Fiscaliza-RU.v1 eram feitas de uma forma que várias consultas eram realizadas ao banco para consultar as informações referente aos alunos. O código era muito extenso e não modularizado em funções, o que diminuía muito a legibilidade do mesmo e dificultava na manutenção. O código originado modulo de importação foi totalmente excluído e um novo foi feito.

#### 7.10.8.1 Redução de código

O código do modulo responsável pelas importações em sua versão original, antes da reformulação, apresentava 594 linhas. Sendo que 300 linhas eram responsáveis apenas por realizar a importação e verificar as restrições. As demais linhas eram a respeito de outras funcionalidades do modulo.

O código do modulo após ser refeito utilizando todas as query's com Eloquent ORM e models redesenhadas, começou apresentar 392 linhas. Sendo que apenas 50 linhas eram responsáveis por realizar a importação e verificar as restrições. As demais linhas são a respeito de outras funcionalidades novas do modulo implementado, tais como: estatísticas e gráficos das importações, detalhes e histórico das importações.

Em sua versão original, o modulo de importação foi implementado de forma imperativa e nada era modularizado. Também é importante destacar que o reuso de código não era incorporado. Todas essas características diminuíam a legibilidade do código fonte e deixava as funcionalidades mais suscetíveis a erros e bugs.

Para poder melhorar esse aspecto e permitir reuso de código, cada restrição foi implementada separadamente em um sub-programas que realiza a devida validação da restrição e retorna os erros encontrados ou uma mensagem de sucesso. Na Figura 57 é apresentado um exemplo de como essas restrições foram implementadas separadamente.

Conforme apresentado na Figura 57, para cada restrição do sistema existe uma função que realiza a validação dos dados contidos no relatório sendo importado. Dessa forma, passou a se ter vários códigos menores realizando tarefas especificas. O reuso fica por conta de não ser

Figura 57 – Modularização das restrições

```
$nao_existem = $this::verficarUsuariosExistem($movimentacoes);
$mov_existentes  = $this::VerficaMovimentacoesExistentes($movimentacoes, $tp_mov);
$duplicatas  = $this::VerficaRefeicoesDuplicadas($movimentacoes);
$mov_fora_de_horario  = $this::VerficaMovimentacoesForaDeHorario($movimentacoes);
```

Fonte: Elaborada pelo autor.

necessário ler os dados na planilha ou consultar o banco várias vezes, isso ocorre apenas uma vez e os valores são armazenados na variável "$movimentações", que depois é utilizada apenas como parâmetro para cada uma das funções.

#### 7.10.8.2 Redução de consultas ao BD

A forma como que a planilha passou a ser lida pelo sistema após a reformulação passou a funcionar de maneira completamente diferente. Esse novo comportamento na forma de ler os dados na planilha permitiu diminui consideravelmente o número de consultas ao banco de dados.

Primeiramente é necessário apontar como a leitura era realiza no software em sua versão original e como passou a ocorrer após a reformulação, para poder apontar como foi possível diminuir drasticamente o número de consultas ao BD.

Na versão original, Fiscaliza-RU.v1, a planilha que continha o relatório era lida linha a linha. Cada linha representava um aluno e a refeição realizada por ele, para cada linha lida uma nova consulta era realizada ao banco de dados para verificar as restrições, se o aluno existia na base, se a refeição não era duplicada, etc. Por exemplo: se uma planilha contivesse 1200 refeições, seriam realizadas no mínimo 1200 consultas ao banco de dados apenas para validar se aquele aluno existia na base, porem muitas outras consultas ainda eram realizadas para verificar as outras restrições. Conforme apresentado na Figura 58, um exemplo do comportamento em loop da leitura linha a linha da planilha.

Figura 58 – Modulo de Importação, Fiscaliza-RU.v1

```
if($arquivo){
    while(!feof($arquivo)){
        $linha    = fgets($arquivo);
        $str=explode(',', trim($linha));
        //dd($str);

        if($linha!=''){

            //Verifica se o item a ser inserido já foi cadastrado
            $mov_exists = Movimentacoes::where('data_mov','=',substr($str[0],6,4).'-'.substr($str[0],3,
                2).'-'.substr($str[0],0,2))
                ->where('hora_mov','=',substr($str[1],0,5))
                ->where('tp_mov','=',$valForms['tp-bolsa'])
                ->where('matricula_aluno','=',$str[4])
                ->exists();

            //dd($mov_exists);
            if(!$mov_exists && $str[4]!=""){
                //Verifica se a matricula do usuario existe na base de dados
                $exite_mat = UsuariosRest::where('matricula_usuario_rest','=',$str[4])
                    ->exists();

                if($exite_mat){

                    //Realiza a inserção da movimentação no banco de dados;
                    $movimentacao = new Movimentacoes();
                    @$movimentacao->data_mov = substr($str[0],6,4).'-'.substr($str[0],3,2).'-'.substr($
                        str[0],0,2);
                    @$movimentacao->hora_mov = substr($str[1],0,5);
                    @$movimentacao->tp_mov = $valForms['tp-bolsa'];
                    @$movimentacao->matricula_aluno = $str[4];

                    $str[1] = substr($str[1],0,5);

                    $insert=$movimentacao->save();
```

Fonte: Elaborada pelo autor.

Na Figura 58, a planilha é lida linha por linha em um loop enquanto não chegar ao final do arquivo. Onde que cada linha representa uma movimentação (refeição), realizada por um aluno. Dessa forma imperativa todas as verificações de restrições são realizadas. Apenas esse while possui 300 linhas de código, a legibilidade do código fica comprometida devido tamanha extensão.

Na versão reformulada do sistema, Fiscaliza-RU, a planilha é lida em sua totalidade de uma vez só. A model Excel faz parte de uma biblioteca chamada: Laravel Excel. Essa model implementa métodos voltados para leitura de planilhas de Excel e retorna um objeto da planilha importada. Utilizando o objeto retornado, alguns métodos podem ser utilizados a fim de melhorar a forma de trabalhar com planilhas. Nesse caso, foi utilizado o método “toArray” que transforma todos os dados da planilha em um grande array indexados pelo nome das colunas, na primeira linha do arquivo. Conforme apresentado na Figura 59, a variável “$reader” recebe o retorno da leitura da planilha.

Figura 59 – Modulo de Importação, Fiscaliza-RU

```
$reader = Excel::load($ruta, function($reader){})->formatDates(true, 'Y-m-d')->get();

$movimentacoes_array = $reader->toArray();
$myJSON = json_encode($movimentacoes_array);
$movimentacoes = json_decode($myJSON);
```

Fonte: Elaborada pelo autor.

Como pode ser observado na Figura 59, a variável "$reader" recebe o retorno da leitura da planilha. Depois, é utilizado o método "toArray" para transformar todos os dados em um grande array de dados. Esses dados são indexados pelo nome atribuído em cada coluna na primeira linha da planilha sendo importada. Para poder manipular os dados de forma mais fácil, é utilizado uma função do próprio PHP, json_enconde e jscon_decode respectivamente, para transformar o array em formato json.

Dessa forma todos os dados necessários sobre a planilha estão na variável "$movimentações". Logo se uma planilha contiver 1200 refeições de alunos, em vez de fazer 1200 consultas ao banco de dados, aluno por aluno, igual na forma antiga. O sistema faz apenas uma consulta retornando todos os alunos e guarda eles em uma segunda variável denominada "$alunos". Para saber se os alunos da planilha que estão armazenados na variável "$movimentações" existem de fato no sistema, basta um loop, comparando com a variável "$alunos". Portanto, apenas 1 consulta foi realizada ao banco de dados, para comparar se os 1200 alunos de fato existem no sistema.

CAPÍTULO

# 8

# AVALIAÇÃO DO SISTEMA

Sendo a usabilidade e a satisfação do usuário duas das características mais importantes que devem ser apresentadas para a qualidade de software, foram aplicados, neste trabalho, formulários com atividades de modo a medir o nível de satisfação do usuário ao utilizar o Fiscaliza-RU.

Para tanto, neste capítulo são mostrados os procedimentos seguidos para a realização de um estudo de usabilidade e avaliação do sistema, envolvendo profissionais da CCOM e alunos do curso de Ciência da Computação, para avalição da eficiência e da relevância da mesma. Os participantes foram essenciais nesta pesquisa uma vez que utilizaram sistema para fornecer feedbacks para melhoria do mesmo.

## 8.1 Questionário

A aplicação do questionário ocorreu online, por meio do Google Formulários que é uma ferramenta robusta para essa atividade e livre podendo ser encontrado em https://www.google.com/forms.

Durante a fase de uso do Fiscaliza-RU o participante do estudo para análise da mesma foi instruído à realizar várias atividades no sistema, tais como procurar alunos, adicionar e remover aluno, realizar importações, dentre outras. Após cada atividade, o usuário deveria responder qual foi o nível de facilidade para a realização das tarefas e, também, se algum problema ou dificuldade fosse observada, esses deveriam ser informados.

O questionário foi divido em duas seções, para melhor organização das perguntas. Na primeira seção foram apresentadas as atividades e os entrevistados responderam qual o nível de facilidade com que conseguiam realizar as tarefas no sistema. Na segunda seção os usuários responderam perguntas que não envolveram as funcionalidades do Fiscaliza-RU, mas que revelassem como foi a experiência na seção anterior, em relação à observação de benefícios, desvantagens e problemas apresentados pelo sistema.

Tabela 13 – Questões aplicadas ao usuário (primeira seção)

| **Questões de Pesquisa para avaliação do Fiscaliza-RU** |
|---|
| AT1: Você deve pesquisar um aluno e identificar quantas refeições ele realizou.<br>Q1: Qual foi a<br>facilidade de realizar a tarefa? |
| AT2: Você deve adicionar um aluno manualmente.<br>Q2: Qual foi a facilidade de realizar a tarefa? |
| AT3: Você deve inativar um aluno.<br>Q3: Qual foi a facilidade de realizar a tarefa? |
| AT4: Você deve restaurar o aluno inativado.<br>Q4:Qual foi a facilidade de realizar a tarefa? |
| AT5: Você deve adicionar um novo Curso.<br>Q5: Qual foi a facilidade de realizar a tarefa? |
| AT6: Você deve tentar remover um curso com<br>alunos vinculados e remover um curso sem alunos vinculados.<br>Q6. Qual foi a facilidade de realizar a tarefa? |
| AT7. Você deve adicionar uma nova Bolsa.<br>Q7. Qual foi a facilidade de realizar a tarefa? |
| AT8. Você deve tentar remover uma bolsa com<br>alunos vinculados e remover uma bolsa sem alunos vinculados.<br>Q8. Qual foi a facilidade de realizar a<br>tarefa? |
| AT9. Você deve tentar realizar uma importação de refeições utilizando<br>a planilha fornecida nesse link: https://goo.gl/jpY4gm. A planilha contém refeições fora do horário válido e erros, lembrando que os horários válidos são de 10:30 às<br>14:00 ou 18:30 às 21:00. Você deve tentar corrigi-la de maneira a conseguir<br>realizar a importação. Caso seja apresentado o erro de que a movimentação já<br>existe, tente alterar a data da refeição.<br>Q9. Qual foi a facilidade de realizar a tarefa? |
| AT10. Você deve tentar localizar o histórico de importações<br>e encontrar a importação realizada por você na pergunta anterior.<br>Q10. Qual foi a facilidade de realizar a<br>tarefa? |
| Dúvidas, Sugestões ou melhorias? Deixe sua opinião aqui |

As perguntas realizadas na primeira seção são mostradas na Tabela 13, onde são mostradas as atividades a serem realizadas, representadas por AT, e as questões que o participante deve responder, representadas por Q.

As respostas ao questionário, na primeira seção, restringiram-se a uma nota de 1 a 5 para facilitar o preenchimento e simplificar o entendimento dos conceitos, sendo 1 pouco fácil até 5 que é muito fácil. Não foi obrigatório ressaltar a justificativa pela escolha da reposta, já que as perguntas são dispostas de forma clara. No entanto, caso o participante considerasse relevante e quisesse justificá-las foi reservado um espaço no formulário para as observações discursivas.

As perguntas realizadas na segunda seção são dispostas na Tabela 14, onde nas perguntas

Tabela 14 – Perguntas ao usuário (seção 2)

| Pergunta sobre o Fiscaliza-RU | Sim | Não | Parcialmente | Não sei |
|---|---|---|---|---|
| Q1. O sistema atual atende as necessidades do setor? | | | | |
| Q2. O sistema atual é fácil de usar (usabilidade)? | | | | |
| Q3. O sistema é fácil de instalar (permitindo uso em diferentes computadores ao mesmo tempo)? | | | | |
| Q4. Foi feito um treinamento para uso do sistema? | | | | |

é verificado como foi a experiência dos participantes ao realizar as atividades na primeira seção.

As perguntas realizadas na segunda seção tem por objetivo compreender como foi a experiência no uso do Fiscaliza-RU para as atividades definidas na primeira seção.. As respostas ao questionário, na Tabela 14, restringiram-se a “Sim”, “Não”, “Parcialmente” e “Não Sei”, para facilitar o preenchimento e simplificar o entendimento dos conceitos.

Algumas outras perguntas foram feitas para os participantes responderem de forma aberta, sendo elas:

- Existem funcionalidades que deveriam ser incorporadas ao sistema? Se sim, quais são?
- Você já presenciou alguma(s) falha(s) do sistema? Se sim, quais foram?
- Foi feito um treinamento para uso do sistema?
- Quais são as principais qualidades do sistema?
- Quais são os principais problemas do sistema?

Essas perguntas foram realizadas de forma aberta possibilitando o participante expressar sua opinião de forma mais abrangente, não sendo estas obrigatórias.

## 8.2 Resultados Obtidos

Nesta seção são apresentados os resultados coletados por meio da aplicação das perguntas aos participantes. É importante ressaltar a importância dessa seção visto que este trabalho não teve como objetivo apenas a reengenharia do sistema, mas também uma avaliação acerca das opiniões dos participantes que serão os usuários efetivos do Fiscaliza-RU.

Assim, a aplicação dos formulários objetivou-se para descobrir se a meta desta pesquisa foi alcançada, avaliando se usuários iriam conseguir utilizar o sistema facilmente, sem a necessidade de uma explicação prévia. Na Tabela 16 os resultados são tabulados.

Na Tabela 15 pode-se visualizar os resultados obtidos com as respostas dos participantes da pesquisa de acordo com a nota 1 para “Pouco Fácil” até nota 5 para “Muito Fácil”. Para

Tabela 15 – Resultados Obtidos

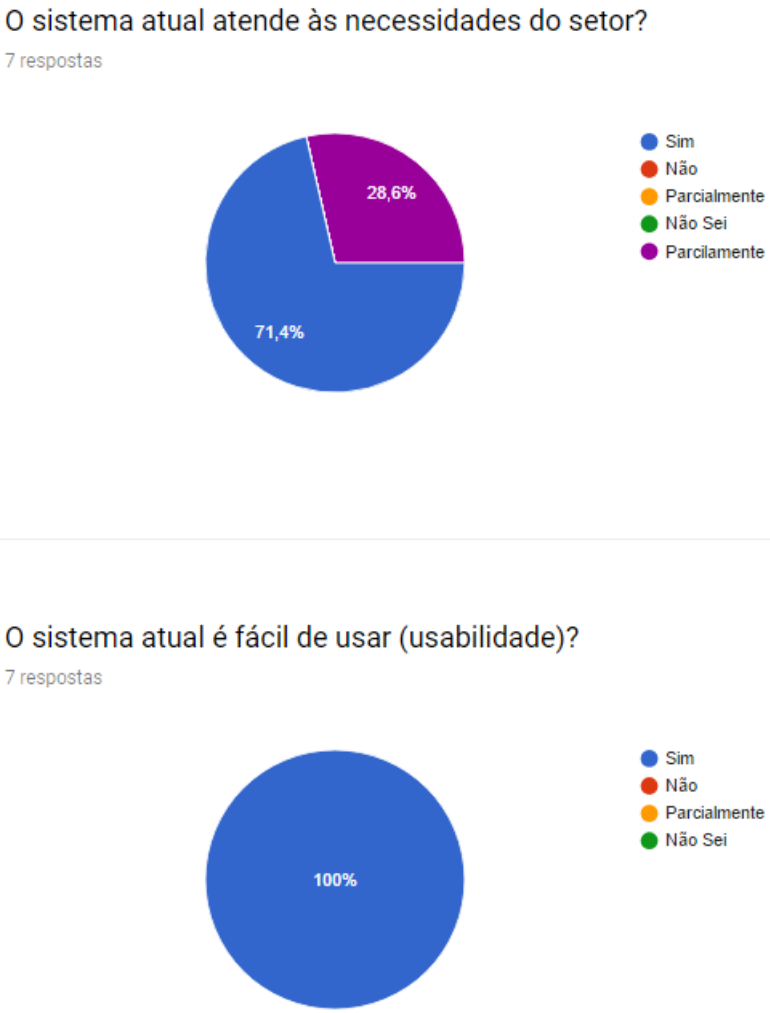
| **Atividade** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| AT1 | - | - | 14,3% | 14,3% | 71,4% |
| AT2 | - | - | 14,3% | 42,9% | 42,9% |
| AT3 | - | - | - | 28,6% | 71,4% |
| AT4 | - | - | - | 42,9% | 57,1% |
| AT5 | - | - | - | 14,3% | 85,7% |
| AT6 | - | - | 14,3% | - | 85,7% |
| AT7 | - | - | - | 14,3% | 85,7% |
| AT8 | - | - | - | 14,3% | 85,7% |
| AT9 | 14,3% | - | 42,9% | 14,3% | 85,7% |
| AT10 | - | - | - | 28,6% | 71,4% |

melhor exibição dos dados, as perguntas foram omitidas, já que as mesmas foram apresentadas na Tabela 13. Ao analisar a Tabela 15, observa-se que praticamente todas as atividades receberam uma avaliação de que é "Muito Fácil" realizar a atividade proposta, ocorrendo isso em mais de 50% dos casos. É relevante destacar uma exceção para a pergunta referente a importação, questão 9, que teve 28,6%.

Na atividade de importação (AT9), observou-se que os usuários apresentaram dificuldade em realizá-la, pois recebeu uma pontuação inferior as demais atividades. Porém, concluiu-se que a forma com que atividade foi atividade proposta dificultou a compreensão dos usuários mais leigos em realizá-la. Entretanto, ao realizar essa pesquisa com membros da CCOM, que já tinha experiência com a rotina de relatório de refeições diariamente, estes não apresentaram dificuldades.

Algumas perguntas chave foram realizadas na segunda seção, essas perguntas apresentaram resultados muito bons, sendo avaliadas positivamente em mais de 70% dos casos, sendo os valores graficamente apresentados na Figura 60.

Figura 60 – Resultados da Pesquisa




Fonte: Elaborada pelo autor.

Ao observar os resultados, na segunda seção de perguntas, representados na Figura 60, compreende-se que os usuários avaliaram o programa positivamente em mais de 70% dos casos. Portanto, mesmo aqueles que apresentaram dificuldade para realizar as atividades na primeira seção, representadas na Tabela 13, avaliaram o programa positivamente pois apresentaram uma boa experiência de uso, é importante ressaltar que uma das dificuldades que os participantes tiveram foi devido a não terem tido contato prévio com o sistema.

CAPÍTULO

9

# CONCLUSÃO

O sistema que desenvolvido ajudará o departamento da Coordenação de Assuntos da Comunidade (CCOM) na fiscalização dos alunos bolsistas sobre sua assiduidade ao RU. Muitos alunos que recebem bolsa, não vão ao restaurante com frequência, enquanto existem outros alunos não bolsistas que apresentam uma grande frequência. O objetivo da CCOM é identificar quais são os alunos assíduos, para ajudá-los com o benefício da bolsa, e quais alunos apresentam baixa frequência ao restaurante, para saber se são de fato necessitados da bolsa que usufruem.

Para desenvolver um sistema de software é necessário identificar e levantar as necessidades, conhecer o problema e as funcionalidades que este deve ter. Neste trabalho, um estudo detalhado foi feito no Fiscaliza-RU.v1, de modo a entender o sistema, desde o código fonte até os relacionamentos entre as tabelas do BD. Ainda foi necessário estudar a tecnologia empregada no sistema inicial e os aspectos e regras inerentes ao sistema.

Tal estudo detalhado foi necessário para que as decisões tomadas em relação a reengenharia do sistema, remodelação da arquitetura, reestruturação da base de dados fossem pautadas em um conhecimento seguro e análise coerente do que deveria ser continuado no sistema, do que era interessante ser evoluído e do que deveria ser criado como nova funcionalidade.

Assim, para análise e especificação do sistema várias reuniões foram feitas, de modo a levantar as funcionalidades importantes. Estas reuniões aconteceram no início da pesquisa e durante o desenvolvimento do software. Isso foi possível devido a escolha pela metodologia de desenvolvimento ser focada em desenvolvimento incremental e ágil, onde as etapas da construção do sistema não foram engessadas e não se gastou muito tempo com extensa documentação.

Uma questão principal de pesquisa havia sido definida, como declaração do problema, para guiar o desenvolvimento deste trabalho, sendo ela: **Quais os principais desafios para o desenvolvimento de software voltado para o uso em uma Universidade?**. Após o desenvolvimento e finalização dessa pesquisa, pode-se concluir que alguns desafios em se fazer software dentro da Universidade, os quais estão relacionados aos trâmites burocráticos, desde ao acesso a

dados de uso restrito, acesso ao local de hospedagem do sistema e aos stakeholders envolvidos no processo, sendo cada um dos cenários descritos a seguir.

O contato com os stakeholders foi fundamental no levantamento de requisitos e compreensão do problema. No entanto, estes, muitas vezes, não tinham muito tempo disponível por terem suas atividades diárias e por não estarem tão envolvidos com a pesquisa, apesar de necessitar do produto final dela, neste caso o sistema Fiscaliza-RU.

Outro desafio encontrado é a disponibilização de servidores para hospedagem do sistema e futura manutenção do mesmo, uma vez que um grande problema, reportado pelos gerentes de TI da Universidade, é que pesquisadores desenvolvem softwares e depois não dão a manutenção devida, o que pode ocasionar em brechas para ataques por falta de segurança nesses.

Ainda, o acesso aos servidores da UFG é restrito, sendo este o local onde a base de produção está hospedada, só podendo ser acessados por redes oferecidas dentro da Regional.

Tendo sido verificados os principais desafios desta pesquisa, foi-se analisado o objetivo central deste trabalho, por meio de uma questão de pesquisa, sendo ela: **"Compreensão do motivo pelo qual o sistema Fiscaliza-RU versão 1 (Fiscaliza-RU.v1) ainda não é utilizado?"**.

A resposta para essa pergunta ficou clara nas primeiras reuniões com os stakeholders, onde foram identificadas as reais necessidades destes e quais funcionalidades o sistema deveria englobar. Sendo assim, pode-se verificar que o Fiscaliza-RU.v1 não contemplava todos os requisitos necessários para a demanda.

Ainda, após a reengenharia do sistema antigo e finalização do novo sistema, foi possível ter mais clareza sobre as diferenças entre o Fiscaliza-RU.v1 e Fiscaliza-RU, basedo na quantidade de funcionalidades oferecidas pelo novo sistema.

Assim, conclui-se que os usuários ainda não utilizavam o sistema pois este não apresentava nenhuma funcionalidade que lhe gerasse benefício ou agilidade nas atividades cotidianas. Sua principal funcionalidade base era permitir as importações de relatórios a fim de se encontrar inconsistências, porém, esta era uma atividade altamente complexa e nenhum suporte foi oferecido, por meio de tutorial ou treinamento, para que os usuários pudessem entender o processo, o que desmotivou sua utilização.

Um benefício adicional aos alunos, provido pelo Fiscaliza-RU, é relacionado a disponibilização de cardápios online com periodicidade, facilitando o acesso a esse tipo de informação. Existem outras soluções para disponibilização dos cardápios na UFG, porém apresentam a desvantagem pela demora do cadastro do cardápio ou a sua não atualização. Neste caso, Fiscaliza-RU estará sempre atualizado devido a obrigatoriedade da empresa terceira, acordo firmado em licitação, enviar o cardápio do mês a CCOM sempre que disponível. Assim, a coordenação poderá realizar o cadastro do cardápio uma única vez ao sistema, ajustando a periodicidade de cada cardápio, com a sua respectiva data de validade, permitindo assim que isso seja propagado para a página pública que os alunos têm acesso.

A UFG – RC será beneficiada pelo uso do sistema para otimização de atividades da CCOM e maior garantia na integridade dos dados. Um ponto relevante está na melhor gerência da verba pública destinada a alimentação de alunos, que poderia estar sendo gasta de maneira equivocada devido a relatórios inconsistentes de refeições. Com o uso do Fiscaliza-RU, haverá maior controle e verificação dos dados, uma vez que as atividades são automatizadas para a fiscalização dos relatórios.

Assim, poderá se ter mais segurança de que a verba está sendo utilizada apenas para relatórios de refeições devidamente consistentes, podendo-se concluir que o dinheiro público está sendo gasto seguindo os propósitos previstos.

Um dado relevante a esta pesquisa foi referente a necessidade em se reformular o Fiscaliza-RU.v1 para garantir a entrega de um sistema funcional, apresentando altos índices de usabilidade. Analisando os dados obtidos na seção de avaliação percebe-se que os participantes acharam o Fiscaliza-RU de fácil utilização, concluindo-se que o objetivo deste trabalho foi alcançado.

## 9.1 Trabalho Futuro

Apesar de muitas melhorias terem sido identificadas e implementadas no Fiscaliza-RU, existem alguns pontos a serem melhoradas, que estão aqui como trabalhos a serem feitos em pesquisas futuras, os quais são descritos nessa subseção.

Atualmente o sistema não apresenta nenhuma forma de controle dos alunos ao restaurante. Porem a base de dados do sistema está preparada para realizar esse tipo de controle, necessitando apenas do desenvolvimento de algum modulo para autenticação dos alunos.

O sistema atualmente está aberto a várias novas funcionalidades, o módulo de autenticação do usuário poderia ser implementado para que os alunos autenticassem por biometria e o sistema automaticamente liberasse o acesso do aluno ao RU. Essa implementação seria interessante pois iria descartar a necessidade de realizar a importação das refeições dos alunos ao restaurante, tornando o sistema mais funcional.

O Fiscaliza-RU não apresenta nenhuma funcionalidade para controle de qualidade ou avaliação do cardápio, porém o módulo está preparado para essa implementação no futuro.

Os alunos não possuem nenhum tipo de controle sobre suas frequências ao restaurante, futuramente o Fiscaliza-RU poderia ser modificado para que alunos autenticassem no sistema com seu número de matrícula e senha, para verificarem suas frequências ao restaurante.

As refeições e o cardápio são repassados a CCOM através de e-mail e os membros da coordenação são responsáveis pelo cadastro das informações ao sistema. Uma futura implementação poderia possibilitar a disponibilidade de um usuário com acesso restrito apenas para as funcionalidades de importação, sem a possibilidade de edição das importações, e adição de

cardápio, para que a própria empresa contratada realizasse o cadastro das informações.

Como conclusão final, pode-se afirmar que muito foi aprendido em relação às práticas de ES, desde o uso de metodologias de desenvolvimento de sistemas até sua efetiva aplicação no contexto de uma fábrica de software. Ainda foi preciso um estudo detalhado em código fonte desconhecido, visto que a reengenharia, re-modulação, reestrutura e reformulação do código eram tarefas essenciais para construção deste trabalho. Adicionalmente, foi necessário um estudo aprofundado na área e sobre as tecnologias, uma vez que o Fiscaliza-RU.v1 foi desenvolvido usando tecnologias diferentes das empregadas para reengenharia do Fiscaliza-RU.

É importante ressaltar que as questões científicas, relacionadas às pesquisas feitas, estudos empíricos realizados e investigações analisadas, foram extremamente enriquecedoras para este trabalho e para a formação do profissional responsável por esta monografia.

Acredita-se que esta monografia será de grande valia a área de Engenharia de Software, mais especificamente para o desenvolvimento de sistemas e qualidade de software, servindo como lição aprendida para futuras pesquisas.

# REFERÊNCIAS

ADRIEL, W. 2015. Disponível em: <http://www.devmedia.com.br/introducao-ao-laravel-framework-php/33173/.> Acesso em: 01/02/2012. Citado na página 75.

BOEHM, B. **Virtual Systems Acquisition: Approach and Transitions, Acquisition Review Quarterly**. [S.l.: s.n.], 1998. Citado na página 44.

CORTêS, S. C. **Mineração de dados – Funcionalidade, Técnicas e Abordagens**. [S.l.: s.n.], 2002. Citado nas páginas 47 e 49.

GIT-SCM. 2017. Disponível em: <https://git-scm.com/book/pt-br/v1/Primeiros-passos-No%C3%A7%C3%B5es-B%C3%A1sicas-de-Git>. Acesso em: 01/02/2012. Citado nas páginas 73 e 74.

KOSCIANSKI, A. **Qualidade de Software. 2 ED.** [S.l.: s.n.], 1985. Citado na página 43.

PRESSMAN, R. S. **Engenharia de Softwares. 5. ed.** [S.l.: s.n.], 2002. Citado na página 44.

ROYCE, W. **Managing the development of large software systems: concepts and techniques**. [S.l.: s.n.], 1970. Citado na página 44.

SCHMITZ, D. 2015. Disponível em: <https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/.Acessoem: Agostode2017>. Acesso em: 01/02/2012. Citado na página 73.

SOARES, M. S. **Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software**. [S.l.: s.n.], 2014. Citado na página 44.

SOMMERVILLE, I. **Técnicas para levantamento de requisitos. Engenharia de Software. 9. ed.** [S.l.: s.n.], 2013. Citado nas páginas 43 e 46.

SOUSA, T. A. de. **UTILIZAÇÃO DE SOFTWARE WEB COMO FERRAMENTA DE APOIO À FISCALIZAÇÃO DE RESTAURANTE UNIVERSITÁRIO**. Monografia (Graduação) — Universidade Federal de Goiás - Regional Catalão, Catalão, 2016. Citado nas páginas 38 e 71.