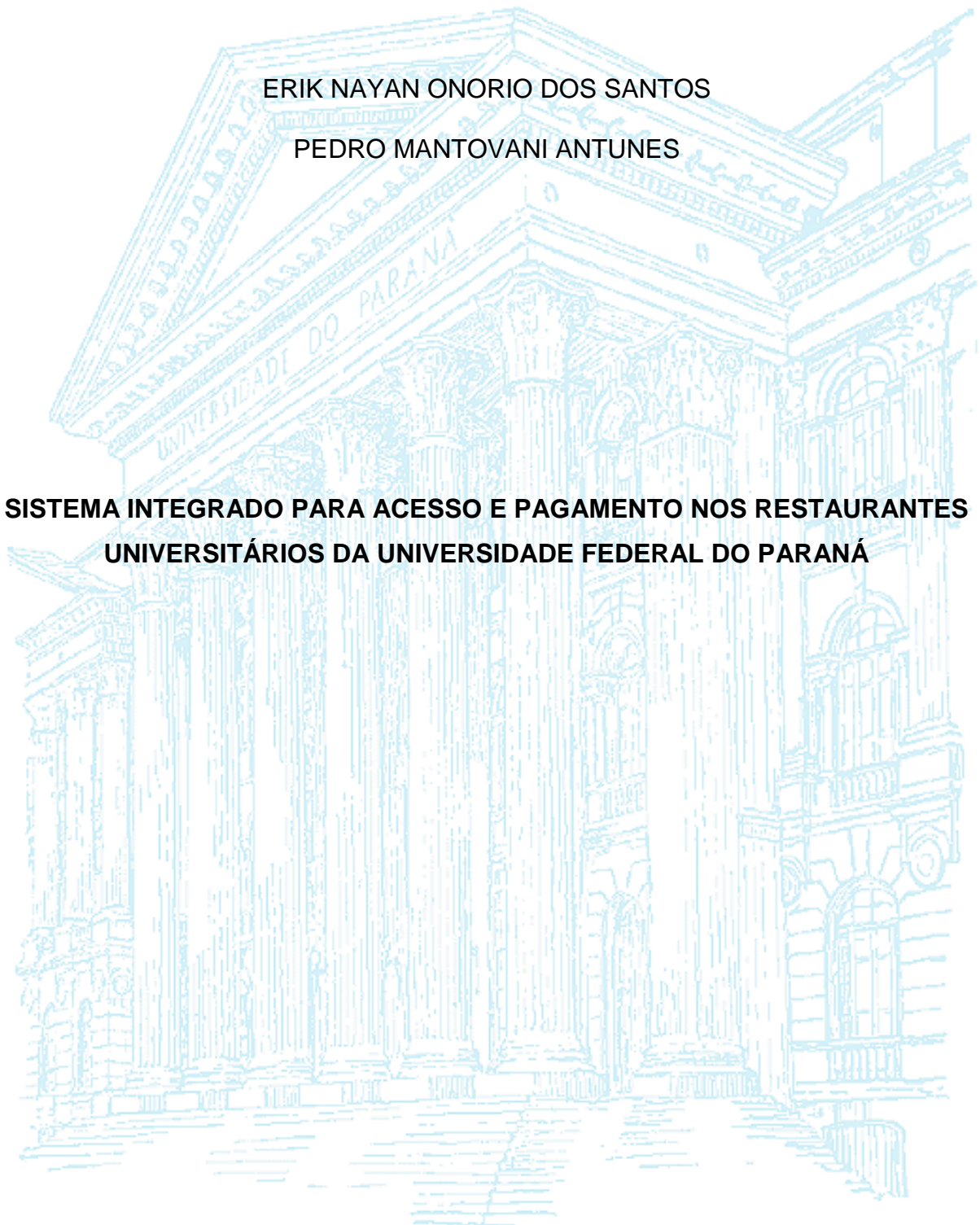


UNIVERSIDADE FEDERAL DO PARANÁ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

ERIK NAYAN ONORIO DOS SANTOS
PEDRO MANTOVANI ANTUNES

**SISTEMA INTEGRADO PARA ACESSO E PAGAMENTO NOS RESTAURANTES
UNIVERSITÁRIOS DA UNIVERSIDADE FEDERAL DO PARANÁ**



CURITIBA
2016

ERIK NAYAN ONORIO DOS SANTOS

PEDRO MANTOVANI ANTUNES

**SISTEMA INTEGRADO PARA ACESSO E PAGAMENTO NOS RESTAURANTES
UNIVERSITÁRIOS DA UNIVERSIDADE FEDERAL DO PARANÁ**

Trabalho apresentado como nota parcial à
disciplina de Projeto Integrado A do curso de
Engenharia Elétrica da Universidade Federal do
Paraná

Professor responsável: Prof. Dr. João da Silva
Dias

Orientador: Prof. Dr. Carlos Marcelo Pedroso

CURITIBA

2016

LISTA DE FIGURAS

FIGURA 1 - “MYCARD” DA UNIVERSIDADE DE LUXEMBURGO.....	10
FIGURA 2 - COMPONENTES E RECURSOS PRINCIPAIS DO SISTEMA.....	12
FIGURA 3 - SCRIPT DE AUTOMAÇÃO DO REPOSITÓRIO	19
FIGURA 4 - <i>USE CASE</i> DO SISTEMA.....	25
FIGURA 5 - AVALIAÇÃO DE DESEMPENHO (<i>MOBILE</i>)	27
FIGURA 6 - AVALIAÇÃO DE DESEMPENHO (<i>DESKTOP</i>).....	28
FIGURA 7 - SEGUNDA AVALIAÇÃO DE DESEMPENHO (COMPREENSÃO DA PERFORMANCE).....	29
FIGURA 8 - SEGUNDA AVALIAÇÃO DE DESEMPENHO (TAMANHO DE ARQUIVOS).....	30
FIGURA 9 - SEGUNDA AVALIAÇÃO DE DESEMPENHO (NOTAS)	31
FIGURA 10 - FOTO DE TELA DO INÍCIO DA PÁGINA PRINCIPAL NO COMPUTADOR	32
FIGURA 11 - FOTO DE TELA DO INÍCIO DA PÁGINA PRINCIPAL NO CELULAR (PARTE 1)	33
FIGURA 12 - FOTO DE TELA DO INÍCIO DA PÁGINA PRINCIPAL NO CELULAR (PARTE 2)	34
FIGURA 13 - CAMPO DE NÚMERO DE MATRÍCULA (<i>EDGE</i>).....	36
FIGURA 14 - CAMPO DE NÚMERO DE MATRÍCULA (<i>FIREFOX</i>)	36
FIGURA 15 - CAMPO DO NÚMERO DE MATRÍCULA (<i>INTERNET EXPLORER</i>)	36
FIGURA 16 - VISUALIZAÇÃO DO CONTEÚDO DA TABELA “USERS”	37
FIGURA 17 - ATUALIZAÇÃO DO CONTEÚDO DA TABELA “USERS”	37
FIGURA 18 - VISUALIZAÇÃO DO CONTEÚDO ATUALIZADO DA TABELA “USERS”	38
FIGURA 19 - SIMULAÇÃO DO FUNCIONAMENTO DO PROGRAMA DAS LEITORAS DE ACESSO	39
FIGURA 20 – PÁGINA DE CADASTRO.....	40
FIGURA 21 – PÁGINA DE SALDO E TRANSAÇÕES PÓS- <i>LOGIN</i>	41
FIGURA 22 – PÁGINA DE DADOS CADASTRAIS PÓS- <i>LOGIN</i>	41
FIGURA 23 – PÁGINA DE RECARGA PÓS- <i>LOGIN</i>	42
FIGURA 24 – VISTA FRONTAL DA CARTEIRINHA DE ESTUDANTE UFPR	43
FIGURA 25 – VISTA TRASEIRA DA CARTEIRINHA DE ESTUDANTE UFPR.....	43

SUMÁRIO

1.	INTRODUÇÃO	6
1.1.	Objetivos	7
1.1.1.	Objetivo Geral	7
1.1.2.	Objetivos Específicos	7
1.2.	Estrutura do trabalho	8
2.	REVISÃO BIBLIOGRÁFICA E METODOLOGIA	9
2.1.	Procedimento de acesso ao RU	9
2.2.	Moeda virtual	10
2.3.	Proposta	11
2.4.	Servidor Central	12
2.4.1.	Servidor Web	13
2.4.2.	Servidor de Base de Dados	13
2.4.3.	Servidor de Email	14
2.5.	Cliente	14
2.6.	Aplicativo de Smartphone	14
2.7.	Validação	15
2.8.	Versionamento de Software	15
3.	DESENVOLVIMENTO	17
3.1.	Divisão do trabalho	17
3.2.	Servidor Linux	18
3.3.	Página Web	19
3.3.1.	Página principal	20
3.3.2.	Página de cadastro	21
3.3.3.	Página de login	21
3.3.4.	Página de pós-login do usuário	22
3.3.5.	Página da área restrita	22
3.4.	MySQL	22

3.4.1.	Criptografia das senhas.....	23
3.5.	Programa das Leitoras de Acesso.....	24
4.	RESULTADOS PARCIAIS E DISCUSSÃO	25
4.1.	Página Web.....	26
4.1.1.	Avaliação de Desempenho	26
4.1.2.	Responsividade.....	31
4.1.3.	Compatibilidade entre navegadores.....	35
4.2.	Banco de Dados	36
4.3.	Programa das Leitoras de Acesso.....	38
4.4.	Visualização de páginas do site.....	40
5.	CONCLUSÃO	44
5.1.	Trabalhos Futuros.....	46
6.	REFERÊNCIAS.....	47

1. INTRODUÇÃO

O Restaurante Universitário da UFPR (Universidade Federal do Paraná), mais conhecido entre a comunidade como “RU”, é um importante serviço para discentes, docentes e técnicos administrativos da universidade, pois proporciona refeições de qualidade, nutricionalmente adequadas e por um custo acessível a todos que o utilizam.

O Restaurante Universitário teve sua origem em 05/08/1966, administrado por membros do Diretório Central dos Estudantes (DCE). A partir de 1980 o Restaurante passou a ter administração da UFPR, quando recebeu, então, a denominação “Restaurante Universitário” (PRÓ-REITORIA DE ADMINISTRAÇÃO DA UFPR, 2016).

Após longo período utilizando sistema convencional de distribuição/servimento “porcionado” – onde os usuários são servidos de porções pré-estabelecidas de alimentos – em 1995, o RU Central adotou o sistema “self-service”, sendo porcionadas apenas o prato proteico (carnes) e a sobremesa, o que proporcionou maior conforto ao usuário em determinar as quantidades desejadas (PRÓ-REITORIA DE ADMINISTRAÇÃO DA UFPR, 2016).

Recentemente, o RU completou 50 anos de existência. De lá para cá, muitas mudanças ocorreram tanto na universidade, quanto no corpo discente/docente que o frequentam. Como é de se esperar, a comunidade acadêmica da UFPR aumentou e a tendência é de que continue a crescer (PROPLAN UFPR, 2014).

Atualmente, o RU conta com 4 unidades em Curitiba e outras 4 nos campi do interior (PRÓ-REITORIA DE ADMINISTRAÇÃO DA UFPR, 2016). Na conjuntura atual, é impossível não deparar-se com grandes filas e verdadeiros “congestionamentos” nos horários de pico, seja na sede do RU Politécnico, Central, Agrárias ou Jardim Botânico. Na verdade, o problema das longas filas não é recente: alunos já realizaram protestos a mais de 10 anos atrás cobrando melhorias (TRIBUNA PARANÁ, 2005). Outro problema grave e decorrente são as fraudes. Alunos que obtêm por mais de uma vez a mesma refeição, ou até mesmo pessoas de fora da comunidade que acessam o Restaurante Universitário como membros da UFPR.

Sendo assim, o desenvolvimento de um sistema de acesso simples e ágil, mas que garanta a segurança e a comodidade dos que o utilizam torna-se pertinente. Além de gerar maior organização no acesso e diminuição no tamanho das filas, fraudes serão evitadas e, por consequência, diminuirá o custo do serviço já subsidiado pela UFPR, convertendo então em benefícios para os próprios estudantes, professores e técnicos.

1.1.OBJETIVOS

Os objetivos desse trabalho são apresentados a seguir, separados de forma a apresentar o objetivo geral e os objetivos específicos.

1.1.1. Objetivo Geral

Desenvolver um sistema para diminuir o tempo de fila de espera nos Restaurantes Universitários e aumentar a segurança contra fraudes, através de um cartão em que o usuário pode inserir créditos via plataforma digital, acessando os restaurantes de forma ágil e eficiente, sem problemas com a manipulação de dinheiro ou identificação.

1.1.2. Objetivos Específicos

Dentre os objetivos específicos destacam-se:

- desenvolver e implementar um *site*, em servidor próprio, para que os usuários possam efetuar o seu cadastro e controlar seus créditos (Projeto Integrado A);
- criar e estruturar uma base de dados para controle e gerenciamento das informações dos usuários e suas transações (Projeto Integrado A);

- garantir a segurança das senhas armazenadas no servidor através de métodos de criptografia avançados e modernos (Projeto Integrado A);
- desenvolver um protótipo de cliente que irá atuar lendo códigos de barras e consultando a base de dados pela rede (Projeto Integrado B);
- utilizar uma plataforma segura e eficiente para inserção de créditos (Projeto Integrado C);
- disponibilizar um aplicativo para *smartphones* capaz de realizar consultas ao cadastro do usuário (Projeto Integrado D);
- enviar *emails* de confirmação automáticos após preenchimento do formulário de cadastro (Projeto Integrado A);
- ampliar a gama de utilizações do sistema para outros departamentos da universidade (empréstimo de equipamentos, biblioteca, etc) (Projeto Integrado D).

1.2. ESTRUTURA DO TRABALHO

Com a devida introdução realizada brevemente, neste capítulo mostra-se a motivação pelo projeto a ser desenvolvido. No capítulo 2 é feita uma revisão bibliográfica dos temas envolvidos bem como o detalhamento de como será desenvolvida a proposta de solução. Já no capítulo 3 é apresentado o desenvolvimento do projeto, partindo dos cronogramas de trabalho até como foi realizada a configuração de componentes do sistema. No capítulo 4 são demonstrados os resultados preliminares obtidos por meio de fluxogramas, simulações, fotos de telas, entre outros testes relevantes. Por fim, o capítulo 5 é a conclusão parcial do projeto, fazendo uma avaliação da aplicabilidade do projeto bem como propondo evoluções e melhorias para desenvolvimentos futuros.

2. REVISÃO BIBLIOGRÁFICA E METODOLOGIA

Neste capítulo, será discutido sobre o estado da arte atual dos sistemas de acesso e pagamento em restaurantes universitários a nível global, bem como de que forma será implementada a proposta deste trabalho, detalhando os principais componentes do projeto e de que forma estes deverão atuar.

2.1. PROCEDIMENTO DE ACESSO AO RU

O procedimento atual de acesso ao RU é um tanto burocrático. Ele está devidamente descrito nos passos a seguir:

- primeiramente, na entrada do restaurante, deve-se apresentar um documento que comprove seu vínculo com a universidade e então recebe-se uma ficha indicando qual é esse vínculo. Ex: “ALUNO”, “SERVIDOR”;
- com a ficha em mãos, você deve dirigir-se ao caixa e então efetuar o pagamento da refeição;
- o indivíduo recebe um ticket fiscal que comprova o pagamento da taxa estipulada para a respectiva refeição (café da manhã, almoço ou jantar);
- o ticket deve ser entregue a um terceiro funcionário, e só então a pessoa tem acesso ao buffet.

Evidentemente, tal processo gera um certo retardo no acesso ao RU, além de ser pouco eficiente em termos de controle e agilidade. Dificuldades com o troco são igualmente comuns e o manuseio de cédulas de dinheiro e moedas antes da refeição pode ser desagradável.

Ao redor do mundo, sistemas mais modernos e versáteis já são utilizados. Na Universidade de Luxemburgo, existe um cartão denominado “*mycard*”, mostrado na Figura 1. O aluno pode utilizá-lo nos diversos campi da universidade, tanto para consumir as refeições do restaurante universitário, quanto para fazer lanches ou utilizar máquinas que vendem comidas e bebidas.

Funciona como uma espécie de cartão de crédito universitário. A recarga pode ser feita diretamente no caixa do restaurante ou através de uma plataforma *online* (UNIVERSITY OF LUXEMBOURG, 2016).

Figura 1 - “myCard” da Universidade de Luxemburgo



Fonte:

http://www.wen.uni.lu/students/useful_information_from_a_to_z/university_restaurant_university_restaurant_card (Acesso em 06/09/2016).

Não há dúvidas de que os restaurantes universitários da UFPR precisam se atualizar e optar por um sistema mais moderno e robusto. É com este intuito que será apresentado um novo modelo de sistema de acesso para os RUs da universidade.

2.2. MOEDA VIRTUAL

Cada vez mais transferências de dinheiro *online* são realizadas pelo mundo, expandindo as possibilidades de comércio globalmente e aumentando a velocidade e compatibilidade com que todas estas transações são feitas. Dentro desse contexto, os serviços de cartões virtuais, como o *Neteller*, proporcionam facilidades para que usuários, através da criação de uma conta gratuita, realizem transações de dinheiro pela Internet com o uso de “carteiras virtuais” que são vinculadas ao seu cadastro. Caso o usuário tenha necessidade, pode requisitar um cartão físico para realização de pagamentos, funcionando como uma espécie de cartão de crédito pré-pago.

Os sistemas de manipulação de moeda virtual requerem transações e processos extremamente seguros e confiáveis, visto que falhas de processamento ou brechas de segurança indevidas podem ocasionar prejuízos

financeiros incalculáveis. Deste modo, empresas como a PaySafe possuem políticas de segurança extremamente robustas, a fim de garantir o sucesso e o sigilo bancário de toda e qualquer operação virtual realizada (NEIL, 2014).

A criação de um sistema de pagamento via cartão pré-pago nos restaurantes universitários baseia-se nesta proposta de utilização de uma conta cujo saldo e transferências encontram-se disponíveis *online*, assim como todas as outras informações relativas a conta, proporcionando maior comodidade ao inserir fundos e efetuar o pagamento das refeições. O método de depósito de dinheiro na conta do usuário dependerá da maneira mais viável de acordo com as possibilidades e dimensões que o sistema atinja.

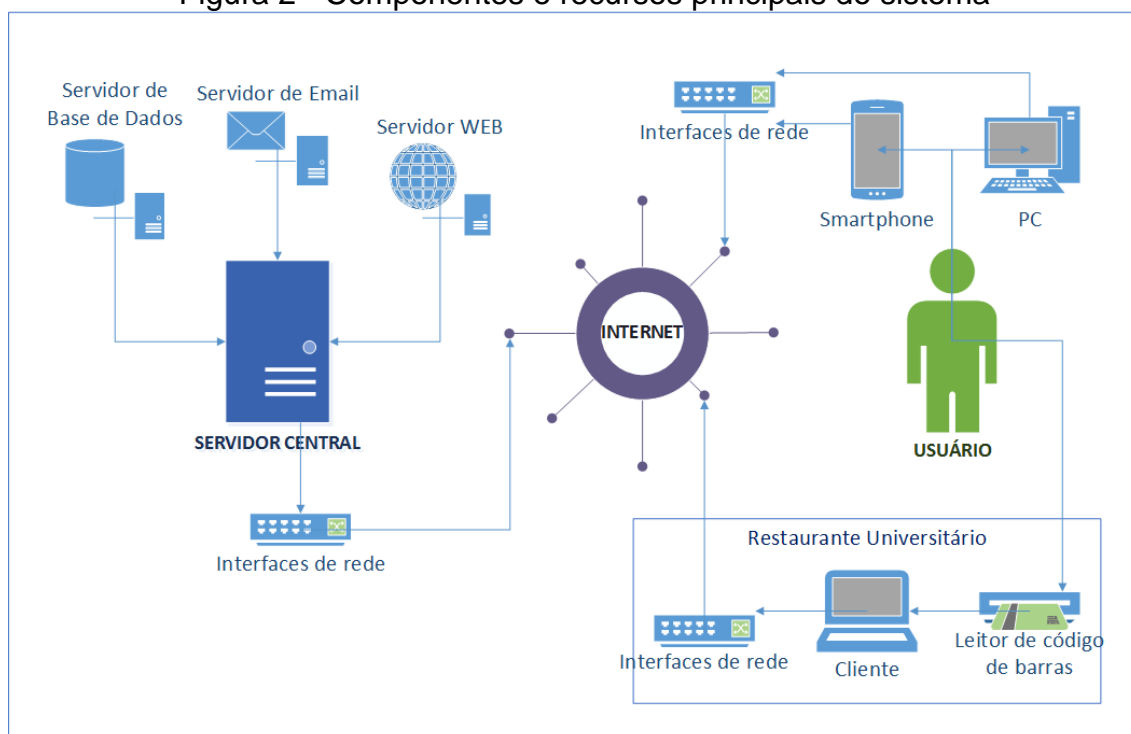
2.3. PROPOSTA

O sistema de acesso rápido ao RU irá atuar através da identificação do usuário pelo código de barras presente no seu documento de vínculo com a universidade, bem como utilizando créditos, que eliminam a necessidade de manuseio de dinheiro e acabam com as dificuldades com o troco. Futuramente, considerando a implementação prática do sistema e a completa adesão da comunidade, será eliminada a necessidade de caixas convencionais.

Inicialmente, o cadastro dos usuários será realizado por um operador do sistema na área reservada para tal no *website*, através de uma confirmação de identificação. Futuramente, este cadastro poderá ser feito de forma automática, no momento da matrícula do estudante/servidor. Contudo, este procedimento dependerá da aceitação e integração dos órgãos responsáveis da UFPR.

O usuário poderá inserir créditos na sua credencial pelo *website* outra forma que for mais conveniente para a universidade. Ao fazê-lo, o usuário poderá acessar os restaurantes universitários simplesmente passando por um leitor de código de barras, que comprovará sua identidade e efetuará o desconto de uma unidade de crédito correspondente a refeição atual – preços diferentes são aplicados para café da manhã e almoço/jantar. Essa plataforma também irá limitar a quantidade de vezes que uma credencial pode ser usada por refeição (uma vez). A Figura 2 exemplifica o funcionamento geral do projeto.

Figura 2 - Componentes e recursos principais do sistema



Fonte: Dos autores.

Uma das vantagens deste modelo é o desenvolvimento do mesmo pela própria comunidade, contando com muitos recursos já existentes e portanto barateando os custos do projeto. Outro aspecto que vale ser citado é a possibilidade de expansão fácil no futuro, podendo atuar como um sistema de controle para as mais diversas aplicações dentro da universidade. Além disso, o sistema conta com um *site* exclusivo para o projeto.

2.4. SERVIDOR CENTRAL

O servidor central deverá ser capaz de responder a todos os outros elementos do sistema pela rede, atendendo solicitações de leitura e escrita na base de dados, bem como fornecer as páginas *Web* adequadas ao usuário. Além disso, ele enviará os *emails* de confirmação assim que um usuário se cadastrar na plataforma. Uma máquina equipada com *Debian Linux*, o conjunto *LAMP server* (*Linux – Apache – MySQL – PHP*) e um servidor de *emails* atenderá a esta aplicação, reunindo todos esses recursos.

2.4.1. Servidor Web

Com a finalidade de hospedar o *site* do projeto, um servidor de páginas Web será empregado. O *site* tem por objetivo permitir a inscrição dos usuários via preenchimento de formulário, bem como explicar as funcionalidades e o objetivo do sistema, algumas outras informações úteis também poderão ser exibidas. Os dados inseridos no *site* serão armazenados no servidor central.

Uma ferramenta para recarga *online* dos créditos dos usuários, bem como registros de transações e histórico de utilização do sistema também deverão ser implementados nesta plataforma, tornando a experiência com o sistema mais dinâmica e facilitada.

O *Apache* é o servidor de páginas mais utilizado no mundo, possui suporte a *PHP*, mensagens de erro e criptografia (ALECRIM, 2006), por este motivo foi escolhido para atender a tal aplicação.

2.4.2. Servidor de Base de Dados

Para armazenar os dados cadastrais dos usuários do sistema, bem como seus créditos e outros dados pertinentes, uma base de dados relacional se faz necessária, pois é desejado manter controle sobre esses dados, realizar alterações de maneira rápida e garantir a integridade de tudo que está armazenado, evitando redundâncias e inconsistências (DA COSTA, 2011).

Com o objetivo de cumprir essa função, o banco de dados *MySQL* foi escolhido, por apresentar extensa documentação, milhares de referências na *internet* e principalmente pela sua fácil instalação e integração com o servidor Web (OFICINA DA NET, 2007).

A fim de proporcionar maior mobilidade e versatilidade, um *script* realizará a tarefa de criar a base de dados, suas tabelas, colunas e usuários. Desta forma, caso seja necessário recriar a base de dados durante a fase de testes, o *script* torna-se uma ferramenta muito útil e simples.

2.4.3. Servidor de Email

Visando o envio de *emails* de confirmação de cadastro, possíveis avisos, e também o recebimento de *emails* vindos da comunidade, uma API (*Application Programming Interface*) foi utilizada para facilitar o envio de *e-mails* através de uma conta no *gmail* criada para este fim. Aqui, foi utilizada a biblioteca PHPMailer, por trabalhar com a linguagem PHP e ser de fácil uso (BOINTON, 2016).

Posteriormente, um servidor de *emails* será instalado no servidor central. Desta forma, será possível utilizar endereços com o domínio do *website*.

2.5. CLIENTE

O cliente – que neste caso refere-se a um computador e não a um usuário do sistema – deverá ser capaz de realizar a leitura das credenciais dos usuários dos restaurantes universitários (via leitor de código de barras), consultar remotamente a base de dados a fim de verificar o status do utilizador (saldo, validade do cadastro, etc.) e então debitar o valor da refeição, seguido da liberação de entrada. Uma máquina rodando uma aplicação escrita em *Python* desempenhará tal papel.

A escolha da linguagem de programação *Python* para desenvolver este aplicativo deve-se, sobre tudo, a sua simplicidade, disponibilidade de documentação em português e ferramentas de integração com a base de dados escolhida (PYTHON HELP, 2012).

2.6. APLICATIVO DE SMARTPHONE

Cada vez mais as pessoas se conectam a Internet através de seus smartphones (FENAINFO, 2016). Assim sendo, um aplicativo que possibilite consultar seu saldo - além de outras informações relevantes - é definitivamente

um complemento muito interessante ao sistema. Portanto, será desenvolvido um aplicativo para *Android* que proporcione tal funcionalidade. A ideia é que o usuário possa exercer um controle de forma mais ágil sobre o seu balanço, consultando-o com poucos toques na tela de seu aparelho pessoal.

2.7. VALIDAÇÃO

A fim de avaliar o correto funcionamento do sistema e a devida atuação em conjunto dos diversos componentes, um procedimento de teste adequado deve ser empregado. Logo, será simulado a utilização do núcleo principal do projeto como se o mesmo já tivesse devidamente instalado e operacional. Para tal, propõem-se a seguinte sequência de procedimentos para validação da primeira etapa do projeto:

- acesso ao *website* do projeto e todas as suas páginas;
- cadastro completo de um usuário pelo *site*;
- inserção de créditos para este usuário;
- verificação da integridade dos dados inseridos na base de dados;
- verificação da segurança da criptografia das senhas armazenadas.

Com relação à inserção de créditos, tal procedimento ainda não será realizado com dinheiro real, nem através de uma transação pelo *site*, pois este recurso será melhor desenvolvido no futuro. Por enquanto, os créditos serão atribuídos diretamente no campo correspondente na base de dados.

Por último, é importante frisar que etapas adicionais e testes específicos podem ainda ser necessários para validar cada um dos procedimentos apresentados.

2.8. VERSIONAMENTO DE SOFTWARE

A partir da revolução dos computadores pessoais, na década de 90, a engenharia envolvida por trás do desenvolvimento de *software* evoluiu em um

ritmo muito acelerado. Logo, os projetos de *software* foram ficando cada vez mais complexos, e a manutenção e desenvolvimento de novos programas precisaram ser feitas muitas vezes por um grupo muito grande de programadores. Desta forma, manter a consistência dos programas sem *bugs* e evoluí-los em um ritmo rápido tornou-se uma tarefa muito complexa.

Neste cenário, a demanda era muito alta por ferramentas que auxiliassem os desenvolvedores nestas questões de consistência e paralelismo no desenvolvimento. Assim, surgiram diversos programas de versionamento de *software*.

Hoje em dia, o estado da arte neste tipo de utilitário é o *Git* com o *GitHub*. Surgido em 2005 por Linus Torvald, o intuito inicial do *git* era auxiliar no desenvolvimento do código-fonte do *kernel* do *Linux*. Contudo, devido ao seu código-fonte aberto, sua popularidade cresceu muito, e hoje é lugar comum entre desenvolvedores do mundo para guardar seus repositórios de código. Seu serviço possui funcionalidades de versionamento dos arquivos, *branching*, *merging*, *cloning*, armazenamento de nuvem, *pull requests*, entre outros conceitos de engenharia de software.

Durante o desenvolvimento inicial deste projeto, foi percebida a necessidade de diversas destas funcionalidades, uma vez que o escopo do projeto é muito amplo, e sua complexidade também. Desta forma, a equipe decidiu criar um repositório no *GitHub* e armazenar todo o código lá, de forma a agilizar o desenvolvimento do projeto e manter o código atualizado em um lugar único.

O repositório do projeto pode ser encontrado em https://github.com/eriknayan/acesso_ufpr. Assim sendo, por questões de organização, não incluiremos o código deste projeto no Apêndice deste trabalho, porém todos os arquivos podem ser encontrados na URL acima.

3. DESENVOLVIMENTO

Neste trabalho visa-se desenvolver um servidor principal, que conte com servidor de *email*, servidor de dados e servidor *Web*. Além disso, um cliente para leitura dos cartões juntamente com um aplicativo para *smartphone* deverá ser desenvolvido, contando ainda com uma página *Web* para maior facilidade de interação com o sistema de créditos e de cadastramento em geral.

3.1. DIVISÃO DO TRABALHO

O sistema proposto por este trabalho é complexo o suficiente para dividi-lo em diversos módulos independentes. Como este é também um trabalho muito longo para a execução íntegra em apenas um semestre por duas pessoas, é proposto aqui a divisão do trabalho destes diversos módulos ao longo de dois anos.

A ordem de prioridade para a programação dos módulos foi definida pelo quão essencial o módulo é para o funcionamento do sistema de forma transparente. Desta forma, a Quadro 1 relaciona a ordem de prioridade de execução dos módulos e as datas previstas de conclusão.

Quadro 1: Módulos para divisão do trabalho

Módulo	Importância	Data Prevista de Conclusão
Modelagem do Banco de Dados	Muito alta	01/12/2016
Página <i>Web</i> simples funcional	Muito alta	01/12/2016
Aplicativos de consulta ao BD	Muito alta	01/12/2016
Plataforma de Pagamento	Alta	01/06/2017
Página <i>Web</i> completa	Média	01/06/2017
Recarga <i>Web</i> de créditos	Média	01/12/2017
Hospedagem do servidor	Baixa	01/12/2017
Aplicativo Mobile	Baixa	01/06/2018

Fonte: Dos autores.

Dado a complexidade dos módulos e para que houvesse uma organização semanal das atividades pendentes, as tarefas desta segunda etapa foram divididas em metas menores compatíveis com a evolução do projeto. O Quadro 2 contém o cronograma atual.

Quadro 2: Cronograma do Semestre

Semana	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Relatório																		
Estudo do método de codificação das carteirinhas																		
Idealização completa do cliente																		
Escolha da leitora de código de barras																		
Desenvolvimento do software embarcado																		
Área do site privada (admin)																		
Adequação do cliente																		
Testes de funcionalidade em rede																		
Integração completa com site e base de dados																		
Legenda		Finalizado						Não finalizado										

Fonte: Dos autores.

3.2. SERVIDOR LINUX

De forma a ter a disposição uma plataforma para trabalhar e verificar rapidamente o progresso e a aparência dos programas e páginas *Web*, foi configurado um servidor Debian provisório, o qual ficará ligado na maior parte do

tempo e conectado à *Internet*. Neste servidor, serviços essenciais para o funcionamento da plataforma foram instalados, como um servidor *Web Apache*, o banco de dados *MySQL*, *PHP*, *Git*, interpretador *Python*, entre outros.

Como este servidor está fisicamente localizado na residência de um dos autores, o modem de saída para a Internet foi conectado em um serviço de DDNS (*Dynamic Domain Name Service*), que fornece um URL (*Uniform Resource Locator*) temporário para o modem. Então, tal modem foi configurado para realizar um encaminhamento das portas 80 e 22 para o endereço IP (*Internet Protocol*) do servidor Debian na rede interna. Isto possibilita então o acesso da página *Web* e a conexão SSH (*Secure Shell*) com o servidor de uma rede externa.

Além desta modificação na rede, também foi criado um *shell script* para que a página *Web* seja automaticamente atualizada com as mudanças realizadas no repositório do *GitHub*. A Figura 3 mostra o breve *script* descrito acima.

Figura 3 - Script de automação do repositório

```
#!/bin/bash  
  
cd /var/acesso_ufpr  
sudo git pull  
sudo cp -r website/* ../www/html  
cd ~
```

Fonte: Dos autores.

3.3. PÁGINA *Web*

Para o desenvolvimento do *front-end* da página *Web*, optou-se por não utilizar nenhum CMS (*Content Management System*). Esta escolha foi feita para que o projeto não ficasse limitado apenas às utilidades que o CMS escolhido traria, além de que é possível otimizar mais o *site* caso este seja programado diretamente, sem o intermédio de mais um programa (SUMMERS, 2014).

Como o *site* foi programado diretamente em HTML/CSS, decidiu-se utilizar o *Bootstrap* como *framework*, para que se tivesse as facilidades de um

sistema de *grid* já consolidado, uma fácil adaptação para deixar o *site* responsivo, uma estrutura para formulários completa e um *layout* para utilizar como base (GIMMER, 2014). Em cima deste *layout* base, foram realizadas customizações próprias através de arquivos CSS.

Decidiu-se aproveitar a utilização do PHP para também modularizar as páginas. Como exemplo, colocando o cabeçalho e o rodapé das páginas em um arquivo separado, e apenas incluindo estes arquivos quando necessário nas páginas. A visualização final para o usuário é idêntica, porém agora o PHP é processado pelo servidor antes de enviar a página em forma de *HTML* puro para o usuário. Assim, obteve-se o mesmo resultado, mas o código com maior sustentabilidade foi deixado para futuras expansões (SANDERSON, 2009).

No momento, três páginas estão em completo funcionamento: A página principal (*homepage*), a página de *login* do usuário e a página de cadastro, que será movida futuramente para a área restrita, onde somente administradores do sistema terão acesso. A página pós-login do usuário, a página pós-cadastro e a página pós-confirmação de *email* encontram-se em desenvolvimento.

3.3.1. Página principal

Dentre as três páginas, a *homepage* é a mais complexa, pois exigiu um trabalho visual mais elaborado. Esta página serve para explicar de maneira interativa e fácil o sistema de acesso e pagamento proposto. Esta página foi dividida em quatro seções principais. A primeira consiste apenas de uma imagem com um descritivo básico e botões com *links* para as páginas de cadastro e *login*. Na segunda seção, é descrito o funcionamento do sistema em 4 passos. A terceira parte apresenta os desenvolvedores do sistema, enquanto a quarta seção serve para mostrar possíveis *feedbacks* de usuários do sistema, atualmente contando com alguns depoimentos meramente ilustrativos.

3.3.2. Página de cadastro

A página de cadastro possui o formulário base para o cadastro de um novo usuário no sistema. A partir deste cadastro, o usuário terá acesso à sua página, onde poderá consultar e inserir mais créditos, além de verificar o histórico de transações. Todos os campos do formulário serão passados para o servidor Web através do método POST, uma vez que ele é recomendado para a transferência de informações sensíveis, pois o faz através do protocolo HTTP (*Hypertext Transfer Protocol*) subjacente (W3C, 2004). A fim de preservar a privacidade e segurança dos dados, um sistema de criptografia de senha através de *hash* foi empregado, juntamente com um *captcha* para confirmar que trata-se de uma pessoa realizando o cadastro. Após realizado o cadastro, o usuário deverá confirmá-lo através de um *link* enviado automaticamente para o endereço de *email* informado. Este procedimento busca garantir que o endereço eletrônico fornecido realmente pertence a pessoa que o cadastrou. Tal *email* é enviado a partir de uma biblioteca disponível para *PHP* (BELÉM, 2009), o que permitiu a utilização de uma conta de envio do serviço *gmail*, facilitando a implementação e evitando maiores complicações com a configuração do servidor interno de *email*, que sofria com dificuldades relacionados a confiabilidade e segurança.

Para garantir a integridade dos dados inseridos bem como o seu real vínculo ao aluno ou servidor em questão, o cadastro de usuários deverá ser realizado mediante acesso a área restrita do site, onde o responsável, de posse dos documentos de identificação necessários, poderá criar um novo perfil de utilizador do sistema.

3.3.3. Página de *login*

A página de *login* serve apenas para a autenticação do utilizador do sistema, fornecendo ou não acesso para a sua página. Nela, o usuário entrará com o seu *email* e senha definidos na etapa de cadastro, e caso validados com sucesso, encaminhará para a página pós-*login*. Uma sessão para o usuário será

criada através de *cookies*, que poderão ser mantidos por até 60 dias, caso o usuário deseje.

3.3.4. Página de pós-*login* do usuário

Após a verificação com sucesso do *email* e senha do usuário na página de *login*, o mesmo terá acesso a página pós-*login*, que em outras palavras é a página pessoal do utilizador do sistema, onde ele encontrará informações sobre suas últimas transações, verificará seus dados cadastrais, poderá consultar seu saldo e realizar recargas em seu cartão. Cada uma destas opções está organizada por abas, permitindo assim uma experiência mais agradável ao usuário quando este desejar realizar operações e/ou consultas.

3.3.5. Página da área restrita

Dedicada a uso exclusivo dos administradores do sistema, esta página exigirá o *login* e senha do administrador, encaminhando-o em seguida para a área onde será possível efetuar e/ou modificar cadastros, verificar logs e fazer quaisquer outras alterações pertinentes no sistema. Tal ambiente busca facilitar a manutenção e gerenciamento do sistema, permitindo que alterações possam ser feitas via *Web* de maneira rápida e prática.

3.4. MYSQL

O banco de dados *MySQL* ainda está sendo estruturado conforme o progresso do projeto. Atualmente, ele possui quatro tabelas, denominadas “*Users*”, “*Tempusers*”, “*Restaurants*” e “*Transactions*”. A tabela “*Users*” manipula os dados de cadastro e saldo do usuário. A tabela “*Tempusers*” também possui um *link* direto com os usuários, porém foi desenhada com o objetivo de armazenar temporariamente os usuários que ainda não passaram pelo processo de confirmação do cadastro, tendo estes um período de três dias após o cadastro para confirmá-lo através do *link* recebido no *email* informado no ato.

Caso não confirmados, o cadastro destes usuário não é efetivado e eles não são transferidos para a tabela “*Users*”. A tabela “*Restaurants*” armazena os dados dos restaurantes cadastrados. E, por último, a tabela “*Transactions*” serve para armazenar um log de todas as transações já realizadas, incluindo as operações de recarga. Alguns usuários também foram criados no *MySQL* com permissões de acesso às tabelas diferentes para aumentar a segurança. O *script* completo da criação do banco de dados e das tabelas com seus respectivos campos pode ser encontrado no Apêndice 7.1. Vale salientar novamente que mudanças na estrutura banco de dados ainda podem ser realizadas de acordo com as necessidades encontradas no decorrer do desenvolvimento do projeto.

3.4.1. Criptografia das senhas

A fim de prover um sistema seguro e confiável para todos os usuários, foi implementado um método de criptografia moderno para o armazenamento das senhas no banco de dados. O método escolhido foi o *BCRYPT*, já consolidado e muito utilizado em grandes sistemas pela Internet (BELÉM, 2012).

Neste método, a senha é criptografada utilizando o método de *hash sha-256*, que possui uma chave de 256 *bits*. Porém, caso a senha fosse armazenada simplesmente utilizando o *hash*, um invasor poderia descobrir a senha do usuário utilizando uma tabela arco-íris (O'DONNELL, 2016). A fim de resolver este problema, o *BCRYPT* implementa a técnica do *salt*, que basicamente mistura a senha original com uma *string* aleatória para só então realizar o *hash*. Desta forma, as chances desta combinação estar em uma tabela arco-íris são ínfimas.

Além disso, o *BCRYPT* é seguro contra-ataques de senha temporizados e possui um mecanismo de gasto de tempo personalizável para processar a senha, buscando diminuir a efetividade de ataques de força bruta.

3.5. PROGRAMA DAS LEITORAS DE ACESSO

Em cada unidade do RU, haverá no mínimo um computador que se conectará com o servidor principal para consultar o banco de dados. O papel da aplicação desenvolvida em *Python* é realizar o intermédio desta operação. O computador rodará o programa em background, que a cada leitura do leitor de código de barras abrirá uma conexão TCP (*Transmission Control Protocol*) e consultará o banco de dados pela porta padrão do *MySQL* 3306. Além disso, caso o usuário seja encontrado e tenha saldo disponível para a transação, o programa em *Python* garantirá que essa transação seja executada em segurança. Conforme o resultado da consulta ao banco de dados, o programa irá retornar diversos códigos de status diferentes, que garantirão ou não o acesso ao usuário.

Este *software* também poderá obter e enviar dados para a base de dados que possibilitem a realização de levantamentos estatísticos sobre o sistema, como por exemplo número de visitantes e horários de pico. Tais dados são importantes para análise de desempenho e também pela possibilidade de proporcionar melhorias ao sistema.

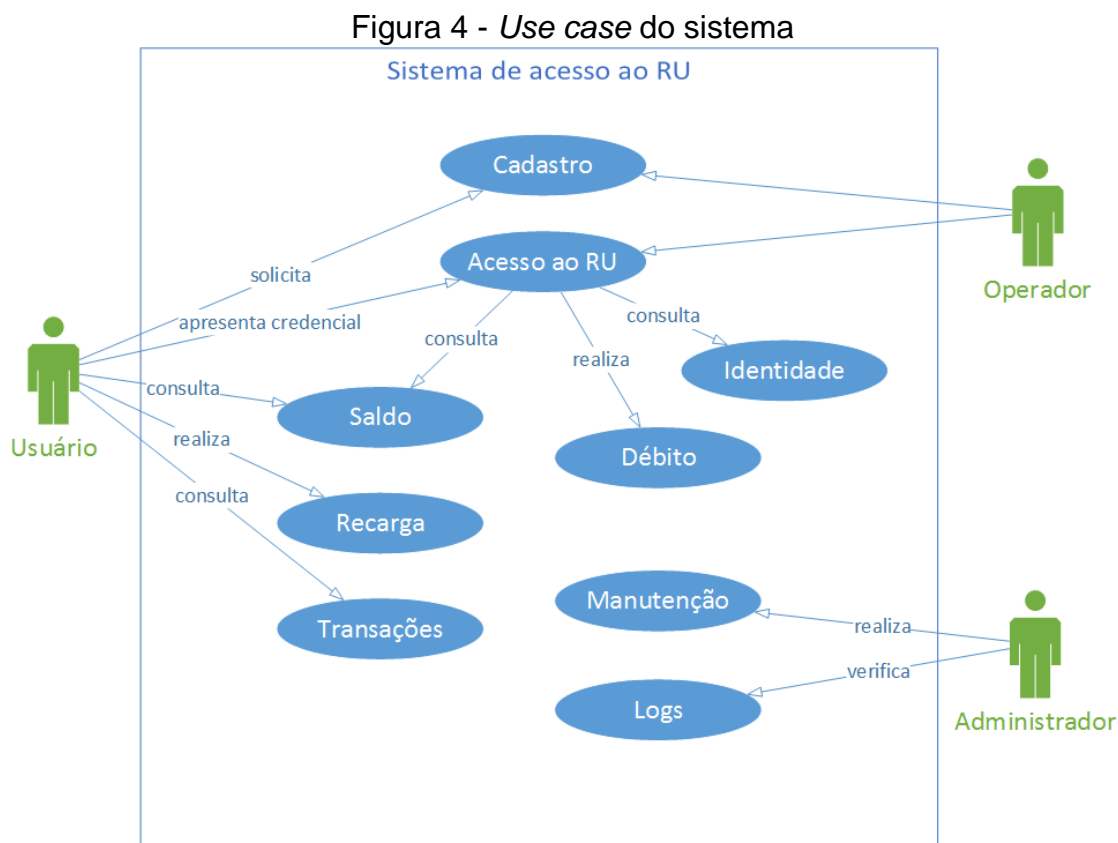
É muito importante assegurar a confiabilidade, segurança e eficiência deste programa cliente, pois será ele o responsável pelo débito de saldo dos usuários, não podendo haver por consequência nenhum equívoco ou falha com a manipulação dos dados pela rede. Tendo isto em vista, faz-se necessário uma atenção especial a este processo de alteração dos dados na base, considerando todos os possíveis tipos de problemas técnicos e invasões indesejadas que possam ocorrer no sistema.

A implementação prática deste cliente dependerá da forma como a gerência dos restaurantes universitários julgar mais adequada, visto que deve-se assegurar que o cartão apresentado a leitora pertence de fato ao usuário que o está portando.

4. RESULTADOS PARCIAIS E DISCUSSÃO

De forma a avaliar o progresso do projeto, bem como debater os resultados até então obtidos, este capítulo exhibe alguns resultados e testes pertinentes a fim de melhor analisá-lo tecnicamente. A página *Web* foi avaliada nos quesitos desempenho, responsividade e compatibilidade entre navegadores. Tais fatores são muito importantes considerando o elevado número de possíveis usuários do sistema que irão acessá-lo de diferentes dispositivos. O banco de dados também foi testado, buscando verificar sua integridade e funcionalidade na inserção, alteração e exclusão de dados. Por fim, o cliente foi posto à prova em uma simulação de funcionamento real, onde este realiza o intermédio de transações entre usuários e a base de dados.

Para um melhor entendimento sobre os “atores” e ações por eles tomadas no sistema, um diagrama do tipo *use case* é apresentado na Figura 4, onde os principais processos são exibidos.



Fonte: Dos autores.

4.1. PÁGINA Web

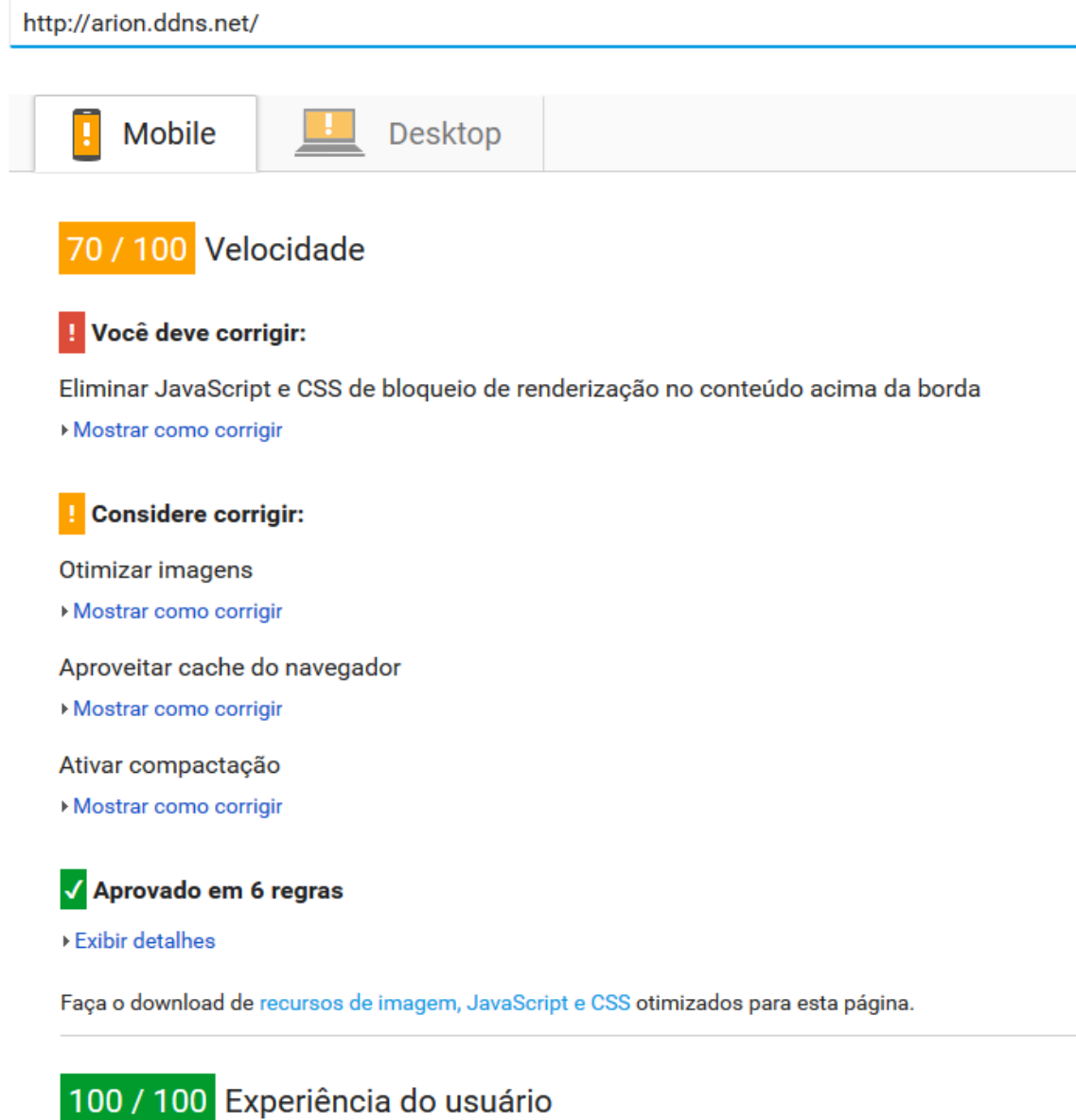
No desenvolvimento das páginas para o *website*, a experiência do usuário foi colocada em primeiro plano. Portanto, definiu-se que o *layout* da página deve ser atrativo o suficiente, e ao mesmo tempo o *site* deve responder rapidamente, ser responsivo e funcionar em diversos navegadores. Outra característica buscada foi um *layout* simples e ao mesmo tempo elegante, evitando uma poluição desnecessária da página.

Como avaliação das páginas desenvolvidas, utilizou-se três métricas diferentes. A primeira consistiu em utilizar *websites* que avaliam desempenho de outros. No segundo método, a responsividade do *site* foi verificada em aparelhos com telas menores. A terceira métrica consistiu em verificar a compatibilidade em vários navegadores diferentes. Tais métricas foram definidas de modo a abordar diferentes parâmetros que juntos tornam um *website* atrativo e funcional.

4.1.1. Avaliação de Desempenho

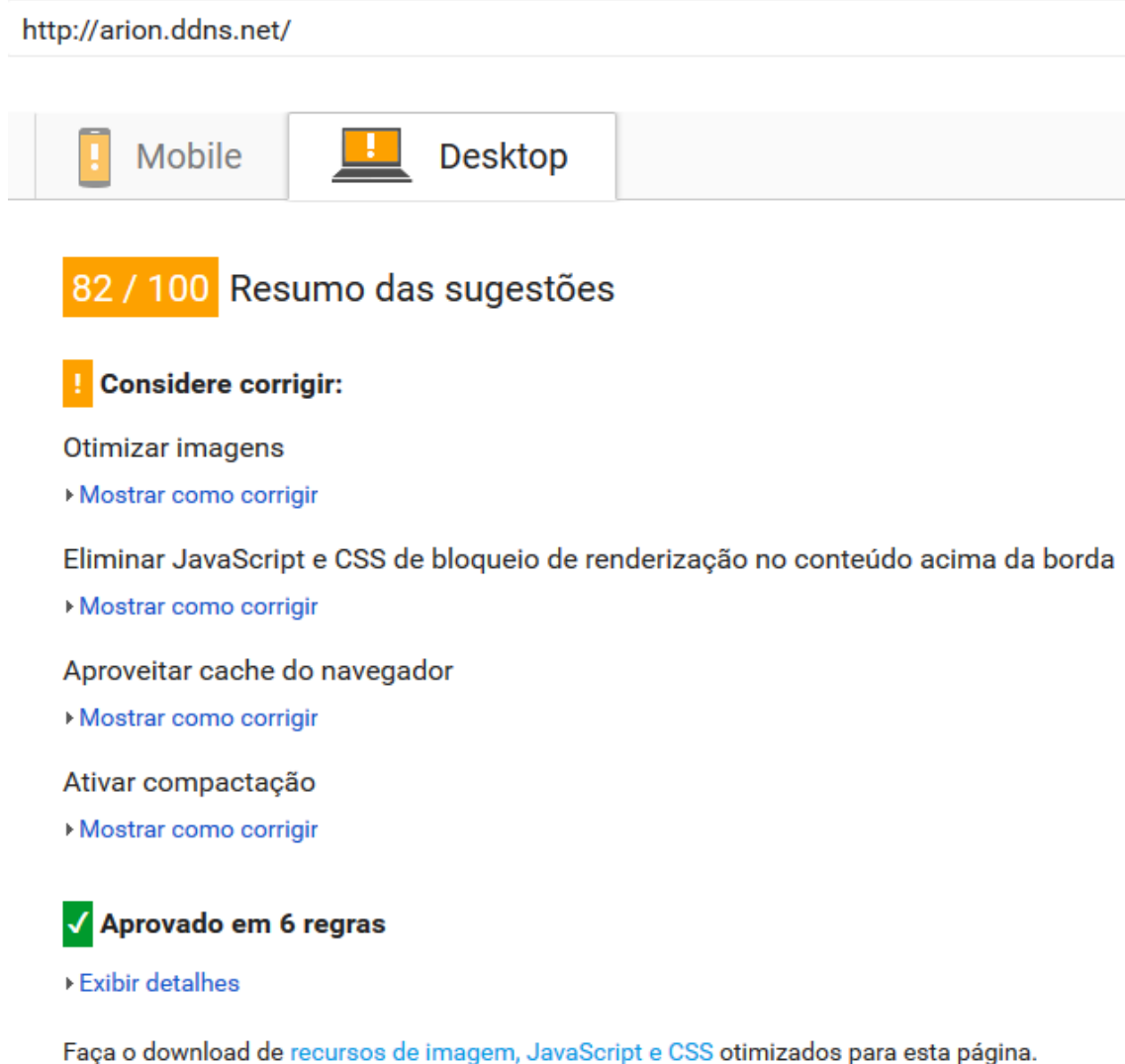
Para este teste, utilizou-se as páginas <https://developers.google.com/speed/pagespeed/insights> e <https://tools.pingdom.com>, que são ferramentas online gratuitas para avaliação de *websites*. As Figuras 5 e 6 mostram o resultado obtido da avaliação da Google, através do primeiro *website*.

Figura 5 - Avaliação de desempenho (*Mobile*)



Fonte: Dos autores.

Figura 6 - Avaliação de desempenho (*Desktop*)

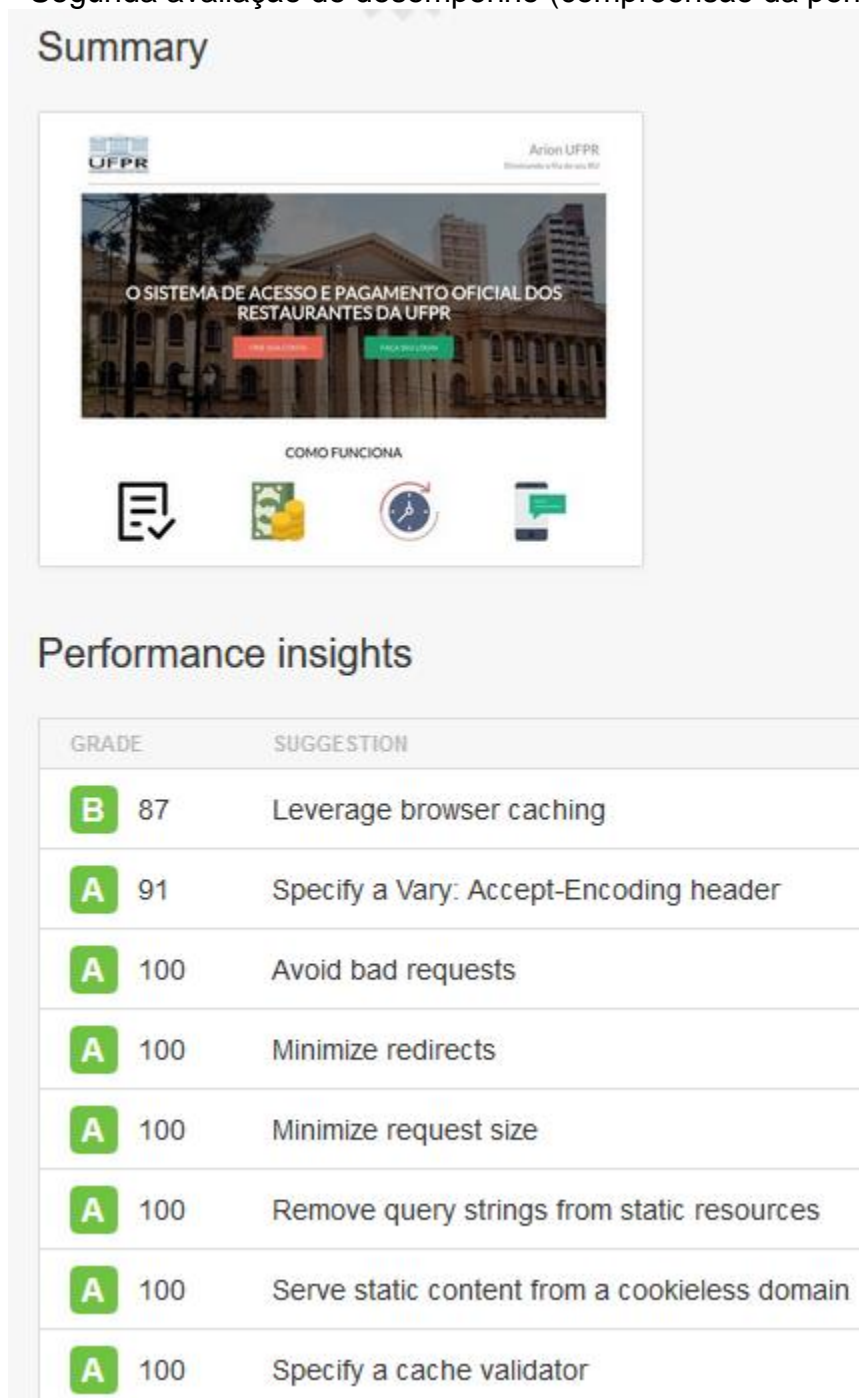


Fonte: Dos autores.

Observando os resultados das primeiras avaliações, conclui-se que ainda há diversos aspectos nos quais há espaço para melhoras futuras. As principais melhorias que podem ser feitas é a remoção dos *JavaScripts* e *CSS* de bloqueios no cabeçalho do *HTML*, a otimização das imagens, a compactação de certos arquivos, e o aproveitamento do cache do navegador para todos os recursos. O primeiro item pode ser resolvido através do carregamento assíncrono destes dados para o navegador. A otimização das imagens pode ser feita através de programas específicos para tal, porém com o *trade-off* de uma imagem com pior qualidade. A compactação de arquivos e o aproveitamento de cache podem ser resolvidos diretamente através de configurações no *Apache*.

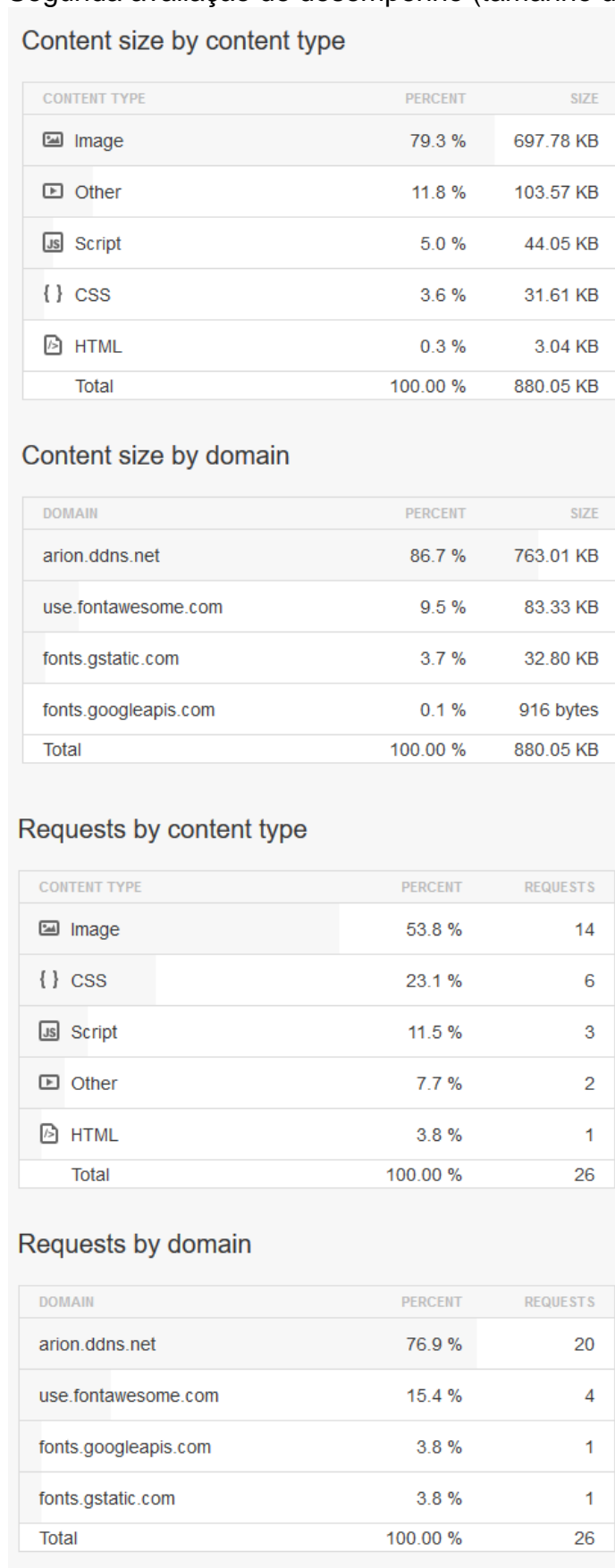
O segundo *website* para avaliação de desempenho fornece mais informações sobre os tempos de carregamento da página, além de atribuir notas e mostrar informações adicionais sobre o tamanho ocupado pelos diversos tipos de conteúdo, o que pode ser extremamente útil em um processo futuro de otimização. As Figuras 7, 8 e 9 apresentam os resultados obtidos.

Figura 7 - Segunda avaliação de desempenho (compreensão da performance)



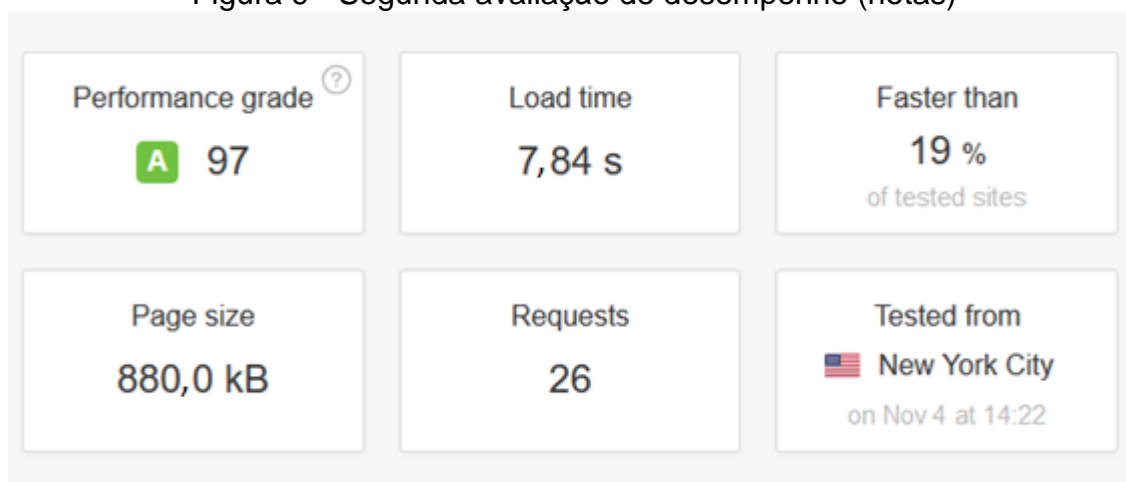
Fonte: Dos autores.

Figura 8 - Segunda avaliação de desempenho (tamanho de arquivos)



Fonte: Dos autores.

Figura 9 - Segunda avaliação de desempenho (notas)



Fonte: Dos autores.

A partir destes resultados, verifica-se que uma grande porcentagem do tamanho total dos arquivos sendo carregados na página é de imagens somente. Este seria mais um motivo para otimizar as imagens e, portanto, deixar o *site* ainda mais rápido. Vale notar, contudo, que a página está sendo atualmente hospedada em um servidor doméstico com acesso à uma rede com velocidade limitada. Portanto, os resultados obtidos nesta seção podem ser enviesados para pior, já que este *site* utilizaria um servidor dedicado em uma rede comercial uma vez que em operação.

4.1.2. Responsividade

Nos últimos anos, o uso de celular como forma primária de acesso à Internet tem subido cada vez mais (GRUBER, 2015). Por isso, é de extrema importância que os *sites* desenvolvidos agora sejam responsivos por natureza. Nos testes realizados, o *website* foi avaliado em um computador de 15 polegadas (resolução de 1920x1080px), em um celular *Nexus 5* de 5 polegadas (resolução de 1080x1920px) e de forma dinâmica variando o tamanho do navegador no mesmo computador.

A visualização em um computador é mostrada na Figura 10, enquanto a visualização em *smartphone* teve que ser dividida em duas imagens, Figuras 11 e 12, já que esta foi composta por imagens agrupadas obtidas com o rolamento

da tela do aparelho, o que a tornou excessivamente vertical para somente uma página deste documento.

Para fins de comparação, o mesmo trecho superior da página inicial foi registrado, de forma a verificar como o *site* adaptaria a exibição do mesmo conteúdo para diferentes aparelhos.

Figura 10 - Foto de tela do início da página principal no computador



Fonte: Dos autores.

Figura 11 - Foto de tela do início da página principal no celular (Parte 1)



COMO FUNCIONA

Fonte: Dos autores.

Figura 12 - Foto de tela do início da página principal no celular (Parte 2)



CADASTRO DA CARTEIRINHA

Crie sua conta e cadastre sua carteirinha da UFPR aqui. O processo leva menos de 5 minutos.



INSIRA SEUS CRÉDITOS

Use seu cartão de crédito para facilitar o pagamento do restaurante universitário. O sistema é 100% seguro e monitorado pela própria UFPR.



SEU RU SEM FILAS

Acesse o RU normalmente utilizando sua carteirinha, sem filas! O resto deixa que a gente cuida.



CONSULTE SEUS CRÉDITOS

Seja na Web ou no seu celular, consulte e insira créditos em menos de 2 minutos. Tudo isto porque sabemos que o tempo de um estudante universitário é muito valioso.

Fonte: Dos autores.

Quanto à responsividade, o *site* comportou-se da maneira esperada, redimensionando-se conforme a largura de tela. Para telas muito pequenas, os itens distribuídos em colunas tornam-se distribuídos em fileiras diferentes, tornando o acesso uma experiência visual melhor para o usuário. Esta característica de responsividade tornou-se mais simples de ser implementado graças ao sistema de *grid* do *Bootstrap*, que já possui as métricas certas para que as colunas sejam distribuídas corretamente.

4.1.3. Compatibilidade entre navegadores

Nesta seção, os navegadores *Google Chrome v53*, *Internet Explorer v11*, *Microsoft Edge 25.1* e *Mozilla Firefox 48.0.2* foram utilizados para verificar a compatibilidade. Este teste incluiu verificar a aparência e a funcionalidade de todos os elementos e *Javascripts*, além de checar a correta visualização das animações.

Durante o desenvolvimento do projeto no Projeto Integrado A, foi detectado o problema do aparecimento dos *spinners* indesejados nos campos numéricos no *Mozilla Firefox*. Este problema foi resolvido no Projeto Integrado B, como parte do esforço de correção de *bugs* da equipe.

Outro problema também foi detectado nestes campos numéricos. Eles deveriam ser limitados para a digitação de somente caracteres numéricos, rejeitando a entrada de qualquer caractere não-numérico. Isto acontece como esperado no *Google Chrome*, porém nos outros três navegadores testados, o campo permite a entrada de texto, como pode-se observar nas Figuras 13, 14 e 15. Apesar de o campo ser invalidado pelo formulário e o usuário não puder submeter o formulário desta maneira, seria mais adequado não permitir a entrada de texto. Para a resolução deste problema, é possível introduzir um pequeno *script* em *Javascript* para deletar qualquer caractere que não seja numérico nestes campos.

Figura 13 - Campo de número de matrícula (*Edge*)



Número de Matrícula (GRR)

testestesteXXXXXXXX

GRR Inválido

Fonte: Dos autores.

Figura 14 - Campo de número de matrícula (*Firefox*)



Número de Matrícula (GRR)

testesteste

GRR Inválido

Fonte: Dos autores.

Figura 15 - Campo do número de matrícula (*Internet Explorer*)



Número de Matrícula (GRR)

testesteste

GRR Inválido

Fonte: Dos autores.

Apesar destes pequenos detalhes, todas as páginas responderam como o programado, todas as animações funcionaram, e o conteúdo foi apresentado como desenhado. Além disso, todos os campos no formulário de cadastro foram validados corretamente, evitando possíveis inserções com erro na base de dados.

4.2. BANCO DE DADOS

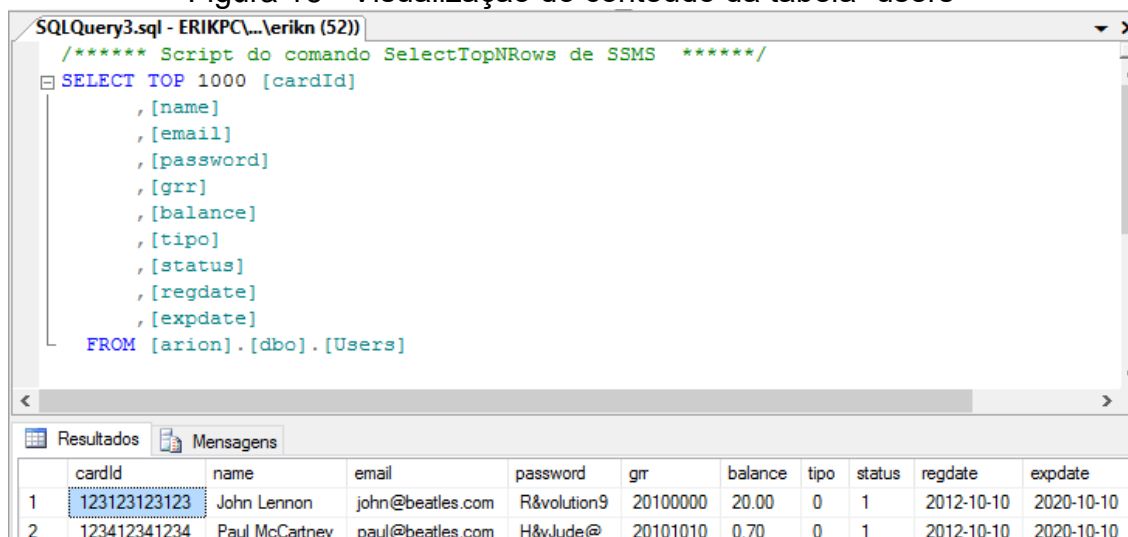
Como citado na seção anterior, a estrutura atual do banco de dados consiste de cinco tabelas. Para o teste, foram inseridos dois usuários teste na tabela “*users*”. Depois disso, foram realizadas algumas consultas ao banco fazendo alterações no saldo de cada um, simulando descontos e créditos.

Deve ser ressaltado que nem todos os campos e tabelas tiveram seus testes demonstrados neste relatório, pois isso o tornaria demasiadamente longo e não acrescentaria novas informações, já que os testes são todos executados

da mesma maneira. Sendo assim, será exemplificado o procedimento somente para o campo “*balance*” (saldo) da tabela “*users*”.

A visualização dos dados da tabela “*users*” encontra-se na Figura 16, os comandos para atualização dos dados de saldo encontram-se na Figura 17 e o estado da tabela após as alterações pode ser visualizado na Figura 18.

Figura 16 - Visualização do conteúdo da tabela “*users*”

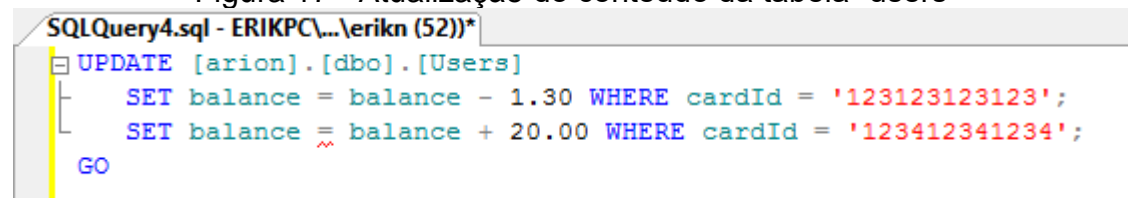


The screenshot shows a SQL query window titled "SQLQuery3.sql - ERIKPC\...erikn (52)". The query is a SELECT statement that retrieves the top 1000 rows from the "Users" table in the "arion.dbo" schema. The columns selected are cardId, name, email, password, grr, balance, tipo, status, regdate, and expdate. Below the query, the "Resultados" (Results) tab is active, displaying a table with 11 columns and 2 rows of data.

	cardId	name	email	password	grr	balance	tipo	status	regdate	expdate
1	123123123123	John Lennon	john@beatles.com	R&volution9	20100000	20.00	0	1	2012-10-10	2020-10-10
2	123412341234	Paul McCartney	paul@beatles.com	H&yJude@	20101010	0.70	0	1	2012-10-10	2020-10-10

Fonte: Dos autores.

Figura 17 - Atualização do conteúdo da tabela “*users*”



The screenshot shows a SQL query window titled "SQLQuery4.sql - ERIKPC\...erikn (52)". The query is an UPDATE statement that modifies the "balance" column in the "Users" table. It uses two SET clauses with WHERE conditions to update specific rows based on their "cardId".

```

UPDATE [arion].[dbo].[Users]
SET balance = balance - 1.30 WHERE cardId = '123123123123';
SET balance = balance + 20.00 WHERE cardId = '123412341234';
GO

```

Fonte: Dos autores.

Figura 18 - Visualização do conteúdo atualizado da tabela “users”

SQLQuery5.sql - ERIKPC\...erikn (52)

```

/***** Script do comando SelectTopNRows de SSMS *****/
SELECT TOP 1000 [cardId]
, [name]
, [email]
, [password]
, [grr]
, [balance]
, [tipo]
, [status]
, [regdate]
, [expdate]
FROM [arion].[dbo].[Users]

```

	cardId	name	email	password	grr	balance	tipo	status	regdate	expdate
1	123123123123	John Lennon	john@beatles.com	R&volution9	20100000	18.70	0	1	2012-10-10	2020-10-10
2	123412341234	Paul McCartney	paul@beatles.com	H&yJude@	20101010	20.70	0	1	2012-10-10	2020-10-10

Fonte: Dos autores.

Como observado, a tabelas “users” está tratando os *updates* e *selects* de forma adequada, mudando o saldo do usuário conforme o indicado pelo *update*.

O procedimento de teste acima também foi realizado para os outros campos de outras tabelas e todos mostraram-se funcionais.

4.3. PROGRAMA DAS LEITORAS DE ACESSO

O programa para as leitoras de acesso foi desenvolvido em *Python*, e no momento funciona apenas na interface de texto. Futuramente, pretende-se evoluir o programa para possuir alguma interface visual, utilizando alguma biblioteca como o *WxPython*, por exemplo. Porém, o programa já está com total funcionalidade, realizando os *queries* no banco de dados através da biblioteca *MySQLdb* em *Python*, e atualizando os campos do banco de dados com transações, alterando o saldo final do usuário. Na Figura 19, algumas simulações do programa em funcionamento são apresentadas. Nelas, o programa requer uma entrada do teclado, que tentará achar uma carteirinha com o número correspondente no banco de dados. Caso o usuário digite algum formato inválido de texto, o programa informará isto de volta ao usuário, que poderá digitar novamente o número. É mostrado também o caso em que o número foi digitado corretamente, porém o usuário não existe no banco de

dados. Desta forma, o programa realizará a *query*, porém não achará nenhum correspondente, e retornará a mensagem “*No such user found*” para o usuário. Similarmente, caso o usuário exista no banco de dados, porém não possua saldo, o programa informará a mensagem “*No balance on account*”. Finalmente, caso o usuário exista no banco de dados e possua saldo, o programa descontará o preço da refeição do saldo final do usuário no banco de dados e informará o sucesso da transação para o usuário.

Este programa será ainda aprimorado, buscando empregar métodos mais seguros e confiáveis para a realização dos débitos de saldos dos usuários.

Figura 19 - Simulação do funcionamento do programa das leitoras de acesso


```
pedro@pedro-VirtualBox ~ $ python client.py
Type your carteirinha number:
Invalid query format
Type your carteirinha number:
Invalid query format
Type your carteirinha number: asdf
Invalid query format
Type your carteirinha number: m456
Invalid query format
Type your carteirinha number: 1234
No such user found
Type your carteirinha number: 4321
No such user found
Type your carteirinha number: 12345678
No such user found
Type your carteirinha number: 12341234
('Pedro', 1000.0, 12341234L)
Transaction Complete
Type your carteirinha number: 12341234
('Pedro', 998.7, 12341234L)
Transaction Complete
Type your carteirinha number: 12341234
('Pedro', 997.4, 12341234L)
Transaction Complete
Type your carteirinha number: 43214321
('Guilherme', 1.1, 43214321L)
No balance on account
Type your carteirinha number: 43214321
('Guilherme', 1.1, 43214321L)
No balance on account
Type your carteirinha number: 12341234
('Pedro', 996.1, 12341234L)
Transaction Complete
Type your carteirinha number:
```

Fonte: Dos autores.

4.4. VISUALIZAÇÃO DE PÁGINAS DO SITE

De modo a demonstrar o funcionamento e o *layout* de algumas páginas do *website*, algumas fotos de tela foram tiradas e adicionadas a este documento. A Figura 20 mostra a página de cadastro, onde os dados inseridos são verificados para assegurar que nenhum dado inválido seja copiado para a base de dados. A Figura 21 exibe a aba inicial pós-*login*, onde é possível verificar o saldo atual e as últimas transações realizadas. Na Figura 22, são exibidos os dados cadastrais do usuário. Por fim, na Figura 23, o usuário encontra-se na aba de recarga, onde informações sobre o método para pagamento são requisitadas.

Figura 20 – Página de cadastro



Arion UFPR
Eliminando a fila do seu RU

Nome

Erik Nayan ✓

Endereço de Email

eriknayan@gmail.com ✓

Confirme seu Email

eriknayan@gmail.com ✓

Vínculo com a UFPR

Estudante ▼

Número de Matrícula (GRR)

20131349 ✓

Número da carteirinha (Código de barras)

121212121212 ✓


Senha

•••••••• ✓

Confirme sua senha

•••••••• ✓

✓ Não sou um robô


reCAPTCHA
Privacidade - Termos

Enviar

Fonte: Dos autores.

Figura 21 – Página de saldo e transações pós-login



Arion UFPR
Eliminando a fila do seu RU

Arion Saldo Recarga Transações Informações pessoais Sair

Bem-vindo Erik Nayan!

Seu saldo é de: R\$97.30
[Faça uma recarga aqui](#)


Suas últimas 5 transações foram:

ID	Data e Hora	Valor	Local
3	2016-11-22 13:20:00	1.30	RU Agrarias
2	2016-11-21 13:20:00	1.30	RU Politecnico
1	2016-11-20 16:20:00	100.00	Recarga Online

[Veja seu histórico completo aqui](#)

Fonte: Dos autores.

Figura 22 – Página de dados cadastrais pós-login



Arion UFPR
Eliminando a fila do seu RU

Arion Saldo Recarga Transações Informações pessoais Sair

Dados cadastrais de Erik Nayan

Nome
Erik Nayan

Endereço de Email
eriknayan@gmail.com

Vínculo com a UFPR
Estudante

Número de matrícula (GRR)
20131349

Número da carteirinha
121212121212

Fonte: Dos autores.

Figura 23 – Página de recarga pós-*login*

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

Arion UFPR
Eliminando a fila do seu RU

Arion Saldo **Recarga** Transações Informações pessoais Sair

Para realizar uma recarga, preencha os dados do seu cartão de crédito abaixo. Todas as informações enviadas são seguras.

Detalhes de pagamento

VISA MasterCard DISCOVER American Express

Cartão a ser recarregado
121212121212

Valor da Recarga
R\$ 10,00

Número do cartão de crédito
Número do cartão

Nome no cartão
Nome no cartão

Data de validade
MM / AA

Cód. Verificação
CVC

Realizar pagamento

Fonte: Dos autores.

4.5. CÓDIGO DE BARRAS DAS CARTEIRINHAS

Até o momento não foi possível contatar o responsável técnico pelas carteirinhas de estudante da universidade, contudo, já foi iniciada a análise do tipo de codificação utilizado nelas. A Figura 24 mostra a parte frontal das carteirinhas, enquanto a Figura 25 mostra a parte traseira, onde fica evidente o código de barras empregado pela UFPR. Algumas leitoras de código de barras já foram testadas para leitura das carteirinhas, porém sem sucesso. Assim que definido qual o tipo exato de código de barras que é utilizado, poderá ser adquirida a leitora mais adequada e então integrá-la ao programa cliente.

Figura 24 – Vista frontal da carteirinha de estudante UFPR



Fonte: Dos autores.

Figura 25 – Vista traseira da carteirinha de estudante UFPR



Fonte: Dos autores.

5. CONCLUSÃO

Ao fim dessa etapa de pesquisa e realização do projeto, alguns objetivos específicos puderam ser alcançados e outros encontram-se ainda em desenvolvimento. O servidor central provisório está operacional e os recursos necessários ao projeto já se encontram instalados, mas podem evidentemente ser alterados de acordo com necessidades futuras. Hospedado neste servidor, o *site* do projeto está no ar e a página inicial, bem como a de preenchimento do formulário de inscrição, já estão implementadas e devidamente elaboradas para oferecer um *layout* simples, moderno e objetivo. Na sequência, a meta é finalizar a estrutura principal do sistema, ou seja, permitir que uma credencial da UFPR seja cadastrada, receba a inserção de créditos e, ao ser apresentada ao cliente equipado com leitor de código de barras, este então consulte sua situação na base de dados e efetue o débito de uma unidade de crédito, atualizando o saldo do usuário.

Um dos primeiros requerimentos para este projeto sair do papel era configurar um servidor e disponibilizá-lo para acesso externo pela Internet. Assim sendo, esta foi a primeira prioridade do projeto, e foi rapidamente realizada através da configuração de um computador rodando Debian na rede local de um dos integrantes do grupo, e disponibilizando seu acesso por uma combinação de *port forwarding* e *dynamic DNS*.

Como o sistema proposto é muito orientado a dados, o próximo passo foi instalar um banco de dados e definir as tabelas a serem utilizadas no sistema. Foi utilizado então o banco de dados MySQL, por apresentar fácil integração com o PHP e Apache. Apesar de a estrutura das tabelas ter sido constantemente alterada ao longo do desenvolvimento do projeto, ela foi sempre sendo aprimorada a fim de representar os dados de uma maneira confiável e atualmente está em um estado satisfatório para os autores.

Durante o desenvolvimento do módulo para armazenamento de senhas no banco de dados, foi tomado um cuidado extra para a segurança destes dados. Neste quesito, foi utilizado o método de BCrypt para *hash* das senhas, que se provou seguro contra vários tipos diferentes de ataque ao banco de dados.

Na implementação final do projeto, um computador cliente estará recebendo leituras dos cartões e precisará de um programa que realizará a comunicação com o banco de dados. O desenvolvimento deste cliente é a prioridade atual do projeto, a fim de que ao final desta etapa esteja disponível uma versão protótipo que efetue as transações necessárias, com a devida confiabilidade e robustez. O programa responsável pela manipulação dos dados, que já foi desenvolvido em sua versão inicial, será expandido buscando prover mais recursos aos operadores e usuários.

Outra meta estipulada no início do projeto seria o envio de *emails* no momento do cadastro de novos usuários. Este objetivo foi atingido através utilização de uma biblioteca desenvolvida para este fim. Para os próximos trabalhos, é visada a instalação de um servidor de *emails* na máquina utilizada.

Alguns outros objetivos já foram projetados para os próximos semestres e, portanto, ainda não foram alcançados. Entre estes, está incluído uma plataforma segura de pagamento online para os usuários. Neste quesito, as partes mais desafiadoras incluem prover um website que utilize um protocolo de criptografia fim a fim, como o SSL (*Secure Socket Layer*) por exemplo, e a utilização de uma API para realizar as transações do cartão de uma forma confiável.

Similarmente, o desenvolvimento de um aplicativo para smartphones foi projetado para os próximos semestres, uma vez que não está no topo da lista de prioridades do projeto e trata-se de uma tecnologia mais complexa.

Quanto da ampliação da gama de utilizações do sistema, este é um tópico que é mais interessante ser tratado futuramente, pois o fundamental no momento é o funcionamento do sistema base por completo, e só então a expansão para outras áreas.

Durante o período que compreendeu o desenvolvimento apresentado neste documento, houveram grandes avanços como também novos problemas a serem solucionados. Um dos desafios a ser superado é o de assegurar, no momento do cadastro, que o documento de vínculo com a universidade pertence a quem o está cadastrando. Outro obstáculo é o obter a métrica exata de codificação das carteirinhas, para que a sua leitura seja realizada corretamente. Dentre os progressos, vale citar que a operação do servidor *Web* vem sendo

extremamente satisfatória para o objetivo proposto, já possibilitando a sua ampliação para novas *features*.

5.1. TRABALHOS FUTUROS

Este projeto tem como algumas de suas características principais ser expansível e utilizar sistema operacional *open source* nos computadores que o sustentam. Entusiastas e pesquisadores interessados têm a oportunidade de desenvolver trabalhos que atinjam outras esferas dentro e fora da universidade, bem como optar por um recurso de validação de acesso e credenciamento mais avançado, substituindo o código de barras por biometria, por exemplo.

No que diz respeito a ampliação do nível de integração do sistema, existe muito a ser explorado, pois o projeto permite realizar um controle eficiente através da carteira de identificação da universidade. Assim sendo, incentiva-se o uso do conceito para solucionar outras questões, como gerenciamento de equipamentos de almoxarifado, utilização de recursos internos que demandem identificação, acesso a plataformas exclusivas, entre outros.

Por último, anseia-se que o projeto inspire o desenvolvimento de soluções criativas, inovadoras e eficazes para o uso dos acadêmicos como um todo.

6. REFERÊNCIAS

NORMAS TÉCNICAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 14724**: Informação e documentação — Trabalhos acadêmicos — Apresentação. 2011.

DOCUMENTOS CONSULTADOS ONLINE

PRÓ-REITORIA DE ADMINISTRAÇÃO DA UFPR. **Breve Histórico** (2016). Disponível em: <<http://www.pra.ufpr.br/portal/ru/historico/>> Acesso em: 08 set. 2016.

PRÓ-REITORIA DE PLANEJAMENTO, FINANÇAS E ORÇAMENTOS (PROPLAN). **Relatório de Atividades UFPR 2014**. Disponível em: <http://acervodigital.ufpr.br/bitstream/handle/1884/40654/relatorio_de_atividades_2014.pdf?sequence=1&isAllowed=y> Acesso em 08 set. 2016.

FURLAN, NÁJIA. **Alunos da UFPR cobram investimentos em infra-estrutura**. Tribuna Paraná, 2005. Disponível em: <<http://www.tribunapr.com.br/noticias/parana/alunos-da-ufpr-cobram-investimentos-em-infra-estrutura/>> Acesso em: 08 set. 2016.

DA COSTA, ELISÂNGELA ROCHA. **Bancos de dados relacionais** (2011). Disponível em: <<http://www.fatecsp.br/dti/tcc/tcc0025.pdf>> Acesso em 09 set. 2016.

OFICINA DA NET. **Conheça um pouco mais sobre o MySQL** (2007). Disponível em: <https://www.oficinadanet.com.br/artigo/390/conheca_um_pouco_sobre_o_mysql> Acesso em 09 set. 2016.

ALECRIM, EMERSON. **Conhecendo o Servidor Apache (HTTP Server Project)** (2006). Disponível em: <<http://www.infowester.com/servapach.php>> Acesso em 02 set. 2016

PYTHON HELP. **Por que Python?** (2012). Disponível em: <<https://pythonhelp.wordpress.com/por-que-python/>> Acesso em 04 set. 2016.

UNIVERSITY OF LUXEMBOURG. **University Restaurant/University Restaurant Card.** Disponível em: <http://www.en.uni.lu/students/useful_information_from_a_to_z/university_restaurant_university_restaurant_card> Acesso em 06 set. 2016.

FENAINFO. **Com aumento do uso de smartphones e tablets, gestão de dispositivos vai decolar.** Disponível em: <http://fenainfo.org.br/info_ler.php?id=43556> Acesso em 04 set. 2016.

SUMMERS, BRENT. **Do you really need a CMS?** (2014). Disponível em: <<http://www.dtelepathy.com/blog/philosophy/do-you-really-need-a-cms>> Acesso em 12 set. 2016.

GIMMER, CHRISTOPHER. **Top 5 Reasons to use Bootstrap** (2014). Disponível em: <<https://bootstrapbay.com/blog/reasons-to-use-bootstrap/>> Acesso em 05 set. 2016.

SANDERSON, GREG. **How to automatically include your header, navigation, and footer on every page** (2009). Disponível em: <<http://www.apaddedcell.com/how-automatically-include-your-header-navigation-and-footer-every-page>> Acesso em 07 set. 2016.

W3C. **URIs, Addressability, and the use of HTTP GET and POST** (2004). Disponível em: <<https://www.w3.org/2001/tag/doc/whenToUseGet.html>> Acesso em 01 set. 2016.

GRUBER, JOHN. **Which is the most important device you use to connect to the Internet?** (2015). Disponível em: <https://daringfireball.net/2015/08/most_important_device> Acesso em 29 ago. 2016.

BELÉM, THIAGO. **Enviar e-mails pelo PHP usando o PHPMailer** (2009). Disponível em: <<http://blog.thiagobelem.net/enviar-e-mails-pelo-php-usando-o-phpmailer>> Acesso em 27 out. 2016.

BELÉM, THIAGO. **Criptografando senhas no PHP usando bcrypt (Blowfish)** (2012). Disponível em: <<http://blog.thiagobelem.net/criptografando-senhas-no-php-usando-bcrypt-blowfish>> Acesso em 22 nov. 2016.

BOINTON, MARCUS. **PHPMailer** (2016). Disponível em: <<https://github.com/PHPMailer/PHPMailer>> Acesso em 09 dez. 2016.

O'DONNELL, ANDY. **Rainbow Tables: Your Password's Worst Nightmare** (2016). Disponível em: <<https://www.lifewire.com/rainbow-tables-your-passwords-worst-nightmare-2487288>> Acesso em 24 nov. 2016.

NEIL, J. RUBENKING. **5 Things You Should Know About Virtual Credit Cards** (2014). Disponível em: <<http://www.pcmag.com/article2/0,2817,2468612,00.asp>> Acesso em 21 mar. 2017