RELATÓRIO DO PROJETO INTEGRADO A

Sistema integrado para acesso e pagamento nos Restaurantes Universitários da Universidade Federal do Paraná

|  |  |
| --- | --- |
| **Disciplina:** | **Projeto Integrado A** |
| **Aluno(s):** | **ERIK N. O. DOS SANTOS**  **PEDRO M. ANTUNES** |

CURITIBA  
2016

**Sumário**

[1. Introdução 4](#_Toc462322733)

[1.1. Objetivos 5](#_Toc462322734)

[1.1.1. Objetivo Geral 5](#_Toc462322735)

[1.1.2. Objetivos Específicos 5](#_Toc462322736)

[2. REVISÃO BIBLIOGRÁFICA E METODOLOGIA 6](#_Toc462322737)

[2.1. Visão Geral 6](#_Toc462322738)

[2.2. Proposta 7](#_Toc462322739)

[2.3. Servidor Central 9](#_Toc462322740)

[2.3.1. Servidor *Web* 9](#_Toc462322741)

[2.3.2. Servidor de Base de Dados 9](#_Toc462322742)

[2.3.3. Servidor de *Email* 10](#_Toc462322743)

[2.4. Cliente 10](#_Toc462322744)

[2.5. Aplicativo de *Smartphone* 10](#_Toc462322745)

[2.6. Validação 11](#_Toc462322746)

[3. DESENVOLVIMENTO 11](#_Toc462322747)

[3.1. Divisão do trabalho 12](#_Toc462322748)

[3.2. Servidor Linux 13](#_Toc462322749)

[3.3. Página *W*eb 14](#_Toc462322750)

[3.3.1. Página principal 15](#_Toc462322751)

[3.3.2. Página de cadastro 15](#_Toc462322752)

[3.3.3. Página de login 15](#_Toc462322753)

[3.4. *MySQL* 16](#_Toc462322754)

[3.5. Programa das Leitoras de Acesso 16](#_Toc462322755)

[4. Resultados parciais 17](#_Toc462322756)

[4.1. Página *W*eb 17](#_Toc462322757)

[4.1.1. Avaliação de Desempenho 17](#_Toc462322758)

[4.1.2. Responsividade 21](#_Toc462322759)

[4.1.3. Compatabilidade entre navegadores 24](#_Toc462322760)

[4.2. Banco de Dados 26](#_Toc462322761)

[4.3. Programa das Leitoras de Acesso 27](#_Toc462322762)

[5. Conclusão 30](#_Toc462322763)

[5.1. Trabalhos Futuros 31](#_Toc462322764)

[6. Referências 32](#_Toc462322765)

[7. Anexos 35](#_Toc462322766)

[7.1. Script.sql 35](#_Toc462322767)

[7.2. Header.php 36](#_Toc462322768)

[7.3. Index.php 36](#_Toc462322769)

[7.4. Footer.php 40](#_Toc462322770)

[7.5. Login.php 41](#_Toc462322771)

[7.6. Register.php 42](#_Toc462322772)

[7.7. Style.css 44](#_Toc462322773)

[7.8. Client.py 47](#_Toc462322774)

# 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 univerdade, 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 protéico (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 obtem 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.

## Objetivos

### Objetivo Geral

Desenvolver um sistema para diminuir o tempo de fila de espera nos Restaurantes Universitários e aumentar a segurança contra fraudes.

### 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;
* Criar e estruturar uma base de dados para controle e gerenciamento das informações dos usuários e suas transações;
* Desenvolver um protótipo de cliente que irá atuar lendo códigos de barras e consultando a base de dados pela rede;
* Utilizar uma plataforma segura e eficiente para inserção de créditos;
* Disponibilizar um aplicativo para *smartphones* capaz de realizar consultas ao cadastro do usuário;
* Enviar *emails* de confirmação automáticos após preenchimento do formulário de cadastro;
* Ampliar a gama de utilizações do sistema para outros departamentos da universidade (empréstimo de equipamentos, biblioteca, etc).

# REVISÃO BIBLIOGRÁFICA E METODOLOGIA

## Visão Geral

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://wwwen.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 RU’s da universidade.

## 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.

Ao inserir créditos na sua credencial, 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.

## 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.

### 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 explanar 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.

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.

### 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.

### Servidor de *Email*

Visando o envio de *emails* de confirmação de cadastro, possíveis avisos, e também o recebimento de *emails* vindos da comunidade, um servidor de *emails*  deve ser instalado no servidor central.

Atendendo a tal proposta, o servidor *Postfix* foi escolhido, devido a sua baixa complexidade e documentação bastante acessível. (VIVA O LINUX, 2003).

## 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).

## 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.

## 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;
* Apresentação da credencial do usuário ao leitor de código de barras do cliente;
* Verificação do desconto no saldo do usuário e do registro da transação na base de dados.

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.

# DESENVOLVIMENTO

## 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, propomos aqui a divisão do trabalho destes diversos módulos ao longo de dois anos.

Nossa 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 Tabela 1 relaciona a ordem de prioridade de execução dos módulos e as datas previstas de conclusão.

Tabela 1: Módulos para divisão do trabalho

|  |  |  |
| --- | --- | --- |
| Módulo | Importância | Data Prevista de Conclusão |
| Modelagem do Banco de Dados | Muito alta | 01/12/2016 |
| Página WEB 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 WEB completa | Média | 01/06/2017 |
| Recarga WEB de créditos | Média | 01/12/2017 |
| Hospedagem do servidor | Baixa | 01/12/2017 |
| Aplicativo Mobile | Baixa | 01/06/2018 |

Fonte: Dos autores.

Ainda que divididos os trabalhos para semestres diferentes, achamos interessante agrupar as atividades correspondentes a este semestre e subdividi-las ainda mais, para que pudéssemos nos organizar semanalmente com as atividades pendentes ou a serem realizadas. A Tabela 2 contém o cronograma atual.

Tabela 2: Cronograma do Semestre

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dia | 1/Ago | | | 22/Ago | | 5/Set | | | | 3/Out | | | | | 7/Nov | | | |
| Semana | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Relatório 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Relatório 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Relatório 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Definição do Projeto |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Setup Apache/Linux Server |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Funcionamento num. carteirinha |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Aprender HTML/CSS/PHP |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Criação db MySql |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Página Web Funcional |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Integração db com página Web |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Aplicação Python |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testes e avaliação de perspectiva |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Legenda |  | Cumprido | | | |  | Pedro e Erik | | | | |  | Pedro | |  | Erik | | |

Fonte: Dos autores.

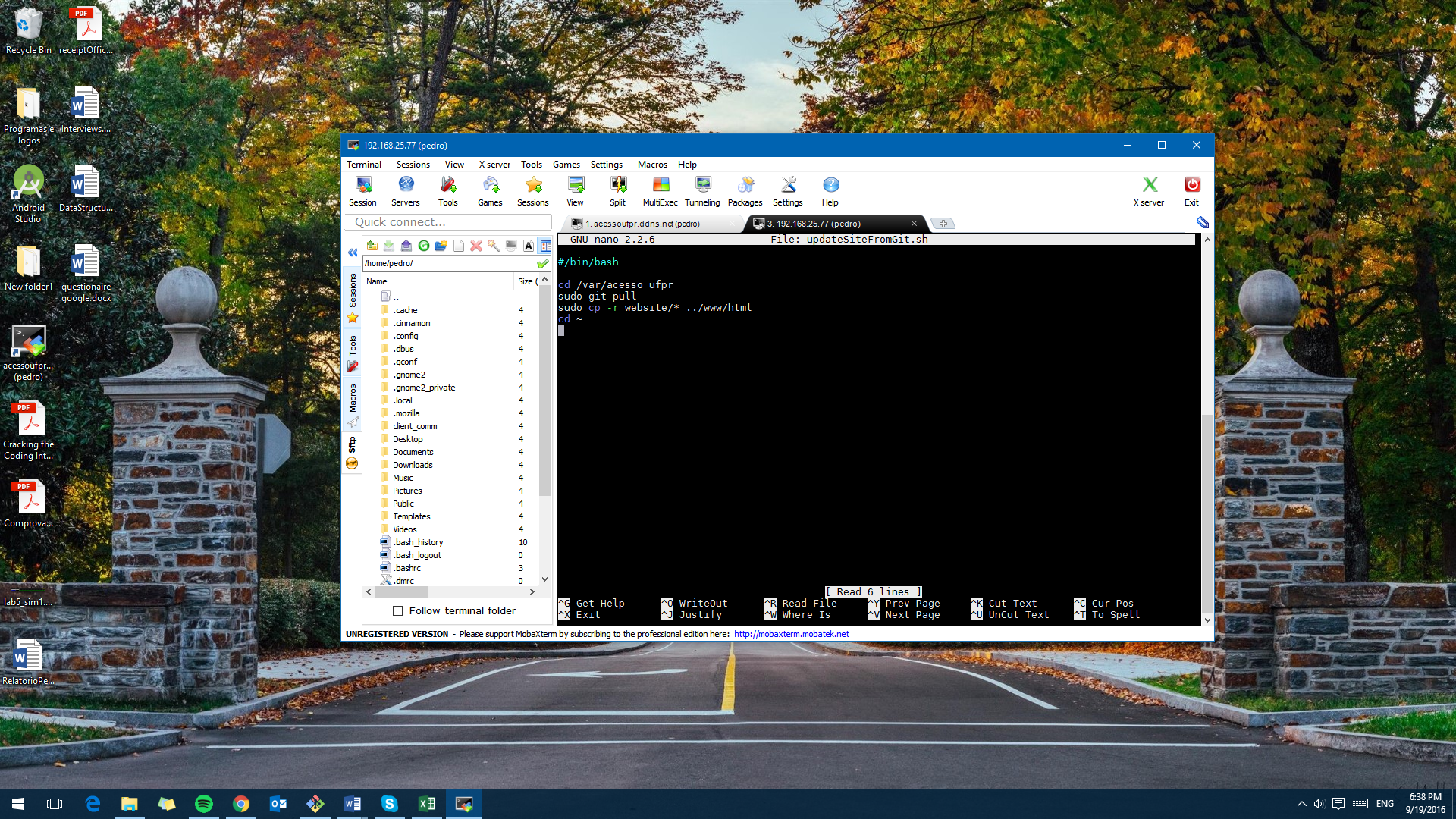
## Servidor Linux

De forma a ter uma plataforma para trabalharmos e verificarmos rapidamente o progresso e a aparência dos nossos programas e páginas *Web*, configuramos um servidor Debian provisório, o qual ficará ligado na maior parte do tempo e conectado à *Internet*. Neste servidor, instalamos serviços essenciais para o funcionamento da nossa plataforma, 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, tivemos de conectar o modem de saída para a Internet em um serviço de DDNS (*Dynamic Domain Name Service*), que fornece um URL (*Uniform Resource Locator*) temporário para o modem. Então, configuramos o modem 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 criamos um *shell script* para automaticamente atualizarmos a nossa página Web com as mudanças que realizarmos no nosso repositório no *GitHub*. A Figura 3 mostra o breve *script* descrito acima.

Figura 3: Script de automação do repositório



Fonte: Dos autores.

## Página *W*eb

Para o desenvolvimento do *front-end* da nossa página Web, optamos por não utilizar nenhum CMS (*Content Management System*). Esta escolha foi feita para que não ficássemos limitados 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 programamos o site diretamente em HTML/CSS, decidimos utilizar o *Bootstrap* como *framework*, para que tivéssemos 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, fizemos nossas próprias customizações através de arquivos CSS.

Decidimos aproveitar a utilização do PHP para também modularizar as nossas páginas. Como exemplo, colocamos o cabeçalho e o rodapé das páginas em um arquivo separado, e apenas incluímos 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, conseguimos o mesmo resultado, mas deixamos o código com maior sustentabilidade para futuras expansões (SANDERSON, 2009).

No momento, possuímos três páginas em completo funcionamento: A página principal (*homepage*), a página de cadastro, e a página de *login*.

### 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, descrevemos o funcionamento do sistema em 4 passos. A terceira parte apresenta os desenvolvedores do sistema, enquanto a quarta seção serve para mostrar alguns testemunhos de usuários do sistema para o usuário. Evidentemente, não possuímos testemunhos reais no momento, já que o sistema ainda não foi implementado.

### 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).

### Página de login

A página de *login* serve apenas para a autenticação do usuário para acesso 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 do usuário.

## *MySQL*

O banco de dados *MySQL* ainda está sendo estruturado conforme a necessidade do projeto. Como o sistema ainda está em uma fase inicial, criamos apenas três tabelas, com o intuito de testarmos os módulos enquanto desenvolvemos. A tabela “*users*” manipula os dados de cadastro do usuário. A tabela “*clients*” também possui um link direto com os usuários, porém foi desenhada apenas para ter acesso às informações de saldo e acesso do usuário, evitando assim o acesso a informações sensíveis do usuário. Por fim, a tabela “*transactions*” serve para manter um log de todas as transações já realizadas. Alguns usuários também foram criados no *MySQL* com permissões de acesso às tabelas diferentes para isolar o uso. O *script* completo da criação do banco de dados e das tabelas pode ser encontrado no Anexo 7.1.

## 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.

# Resultados parciais

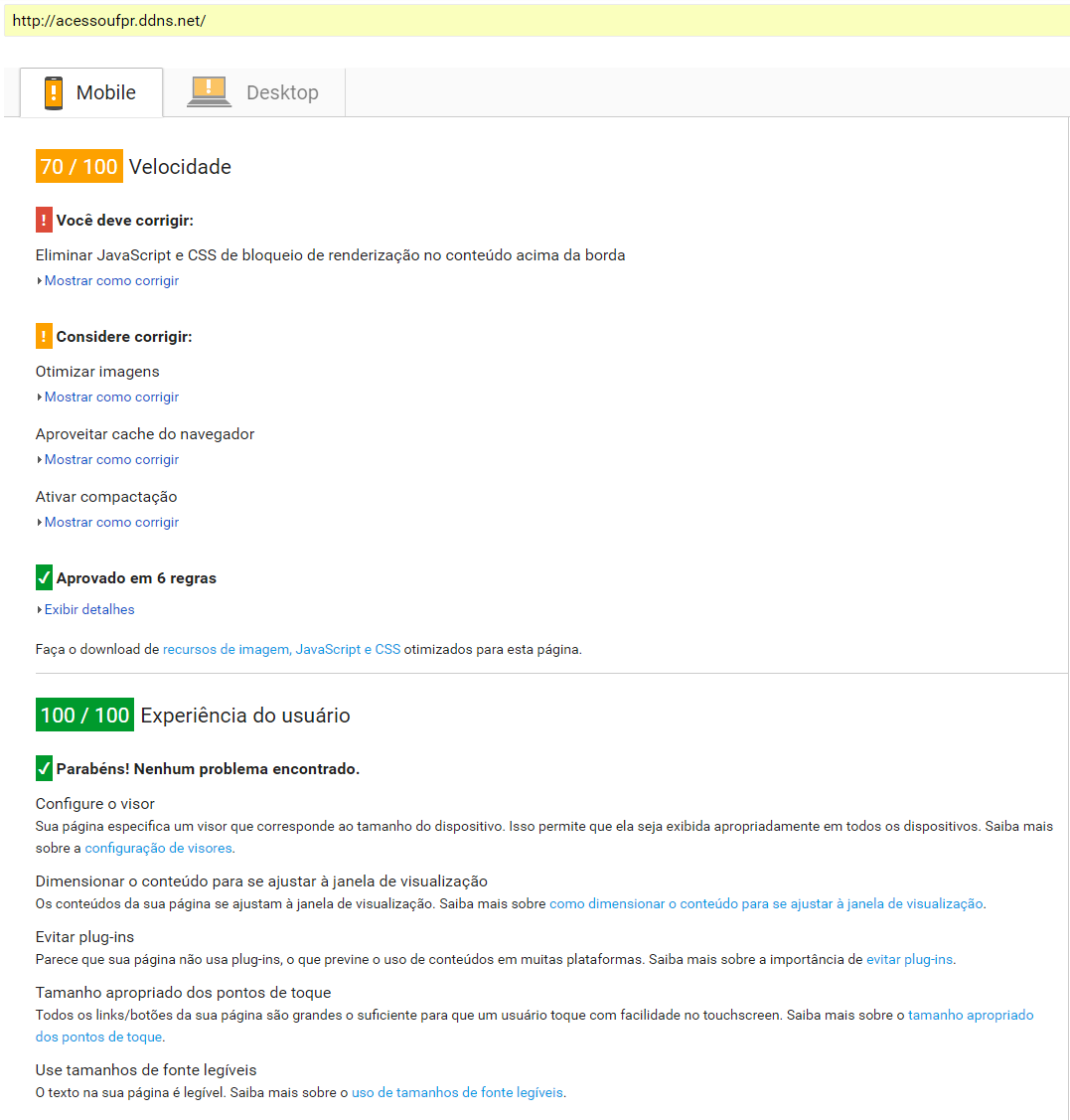
## Página *W*eb

Ao desenvolver as páginas para o *website*, pensamos primariamente na experiência do usuário. Portanto, definimos que o *layout* da página deveria ser atrativo o suficiente, e ao mesmo tempo o *site* deve responder rapidamente, ser responsivo e funcionar em diversos navegadores.

Como avaliação das páginas desenvolvidas, utilizamos três métricas diferentes. A primeira consistiu em utilizar *websites* que avaliam desempenho de outros. No segundo método, checamos a responsividade do *site* em aparelhos com telas menores. A terceira métrica consistiu em verificar a compatibilidade em vários navegadores diferentes.

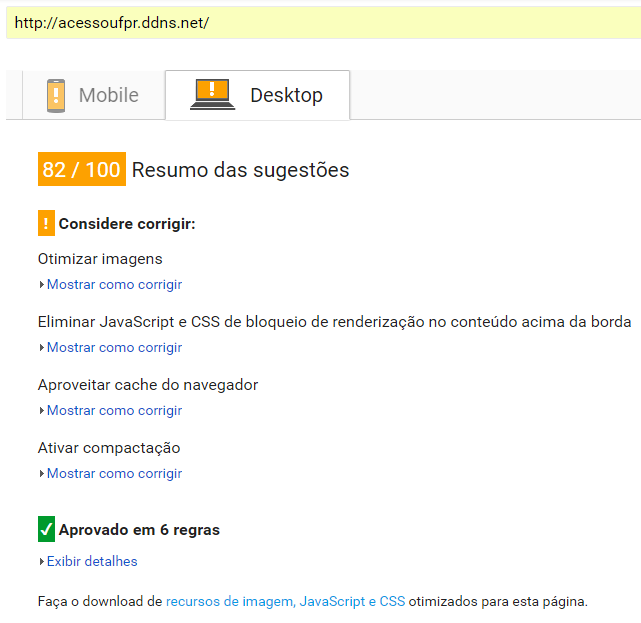
### Avaliação de Desempenho

Para este teste, utilizamos 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 4 e 5mostram o resultado obtido da avaliação da Google, através do primeiro *website*.

Figura 4: Avaliação de desempenho (*Mobile*)

Fonte: Dos autores.

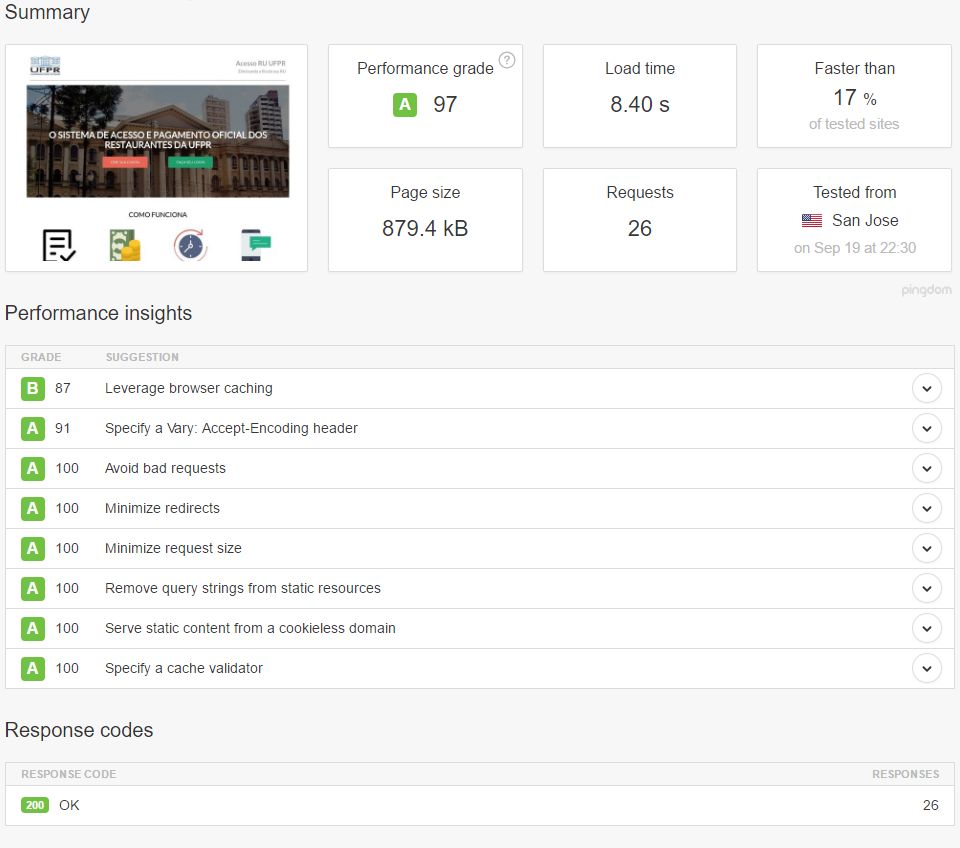
Figura 5: Avaliação de desempenho (*Desktop*)



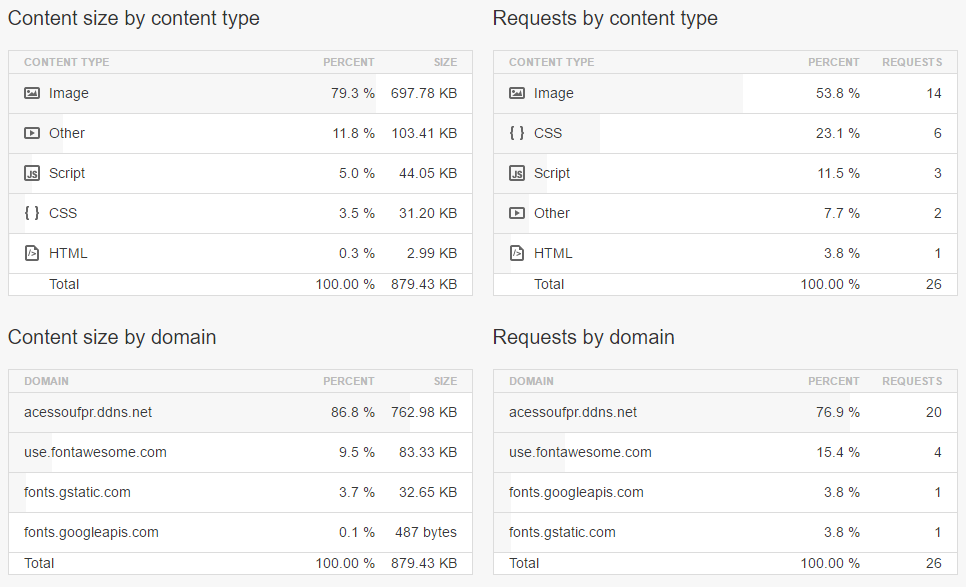
Fonte: Dos autores.

Como observado pelo resultado da avaliação, 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 nos dá mais informações sobre os tempos de carregamento da página, o que pode ser extremamente útil em um processo futuro de otimização. As Figuras 6 e 7 apresentam os resultados obtidos.

Figura 6: Segunda avaliação de desempenho (notas)

Fonte: Dos autores.

Figura 7: Segunda avaliação de desempenho (tamanho de arquivos)

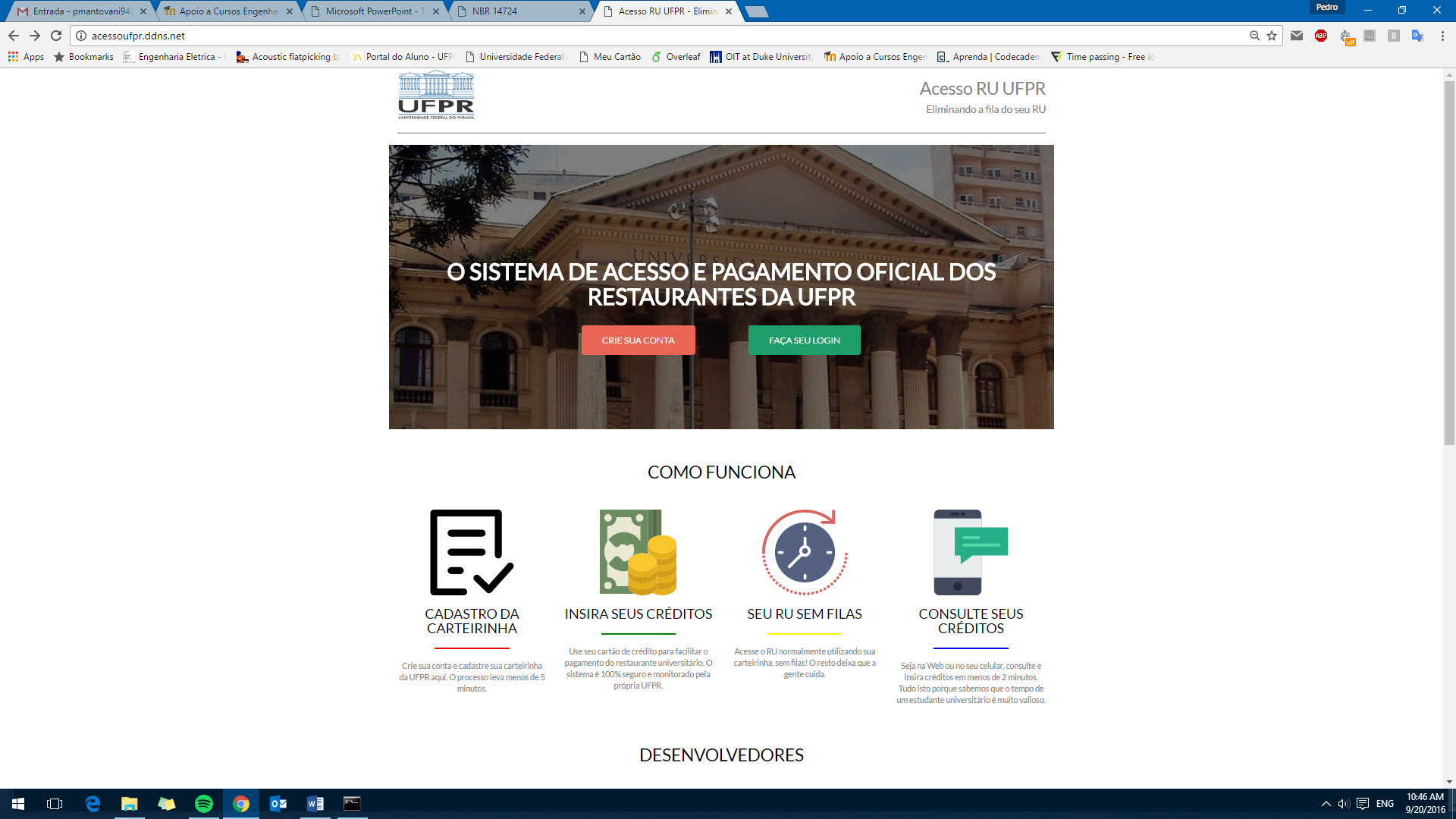
Fonte: Dos autores.

A partir destes resultados, podemos verificar 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.

### 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. Em nossos testes, avaliamos o nosso *website* 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. As fotos de tela nas Figuras 8 e 9 mostram a aparência diferenciada do *website* em dispositivos diferentes.

Figura 8: Foto de tela da página principal no computador



Fonte: Dos autores.

Figura 9: Foto de tela da página principal no celular



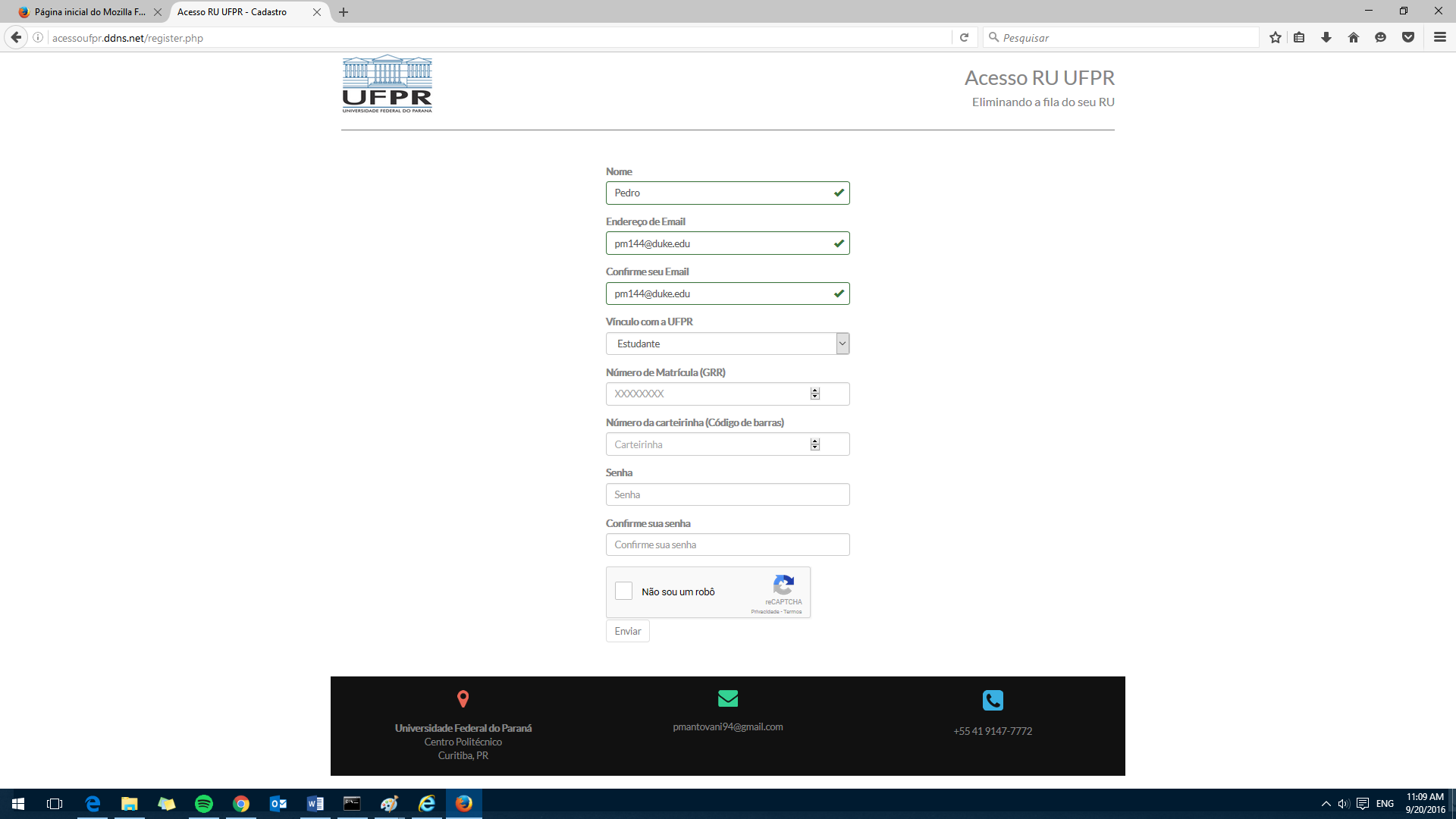
Fonte: Dos autores.

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.

### Compatabilidade entre navegadores

Nesta seção, utilizamos os navegadores *Google Chrome* *v53*, *Internet Explorer* *v11*, *Microsoft Edge* *25.1* e *Mozilla Firefox* *48.0.2* 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.

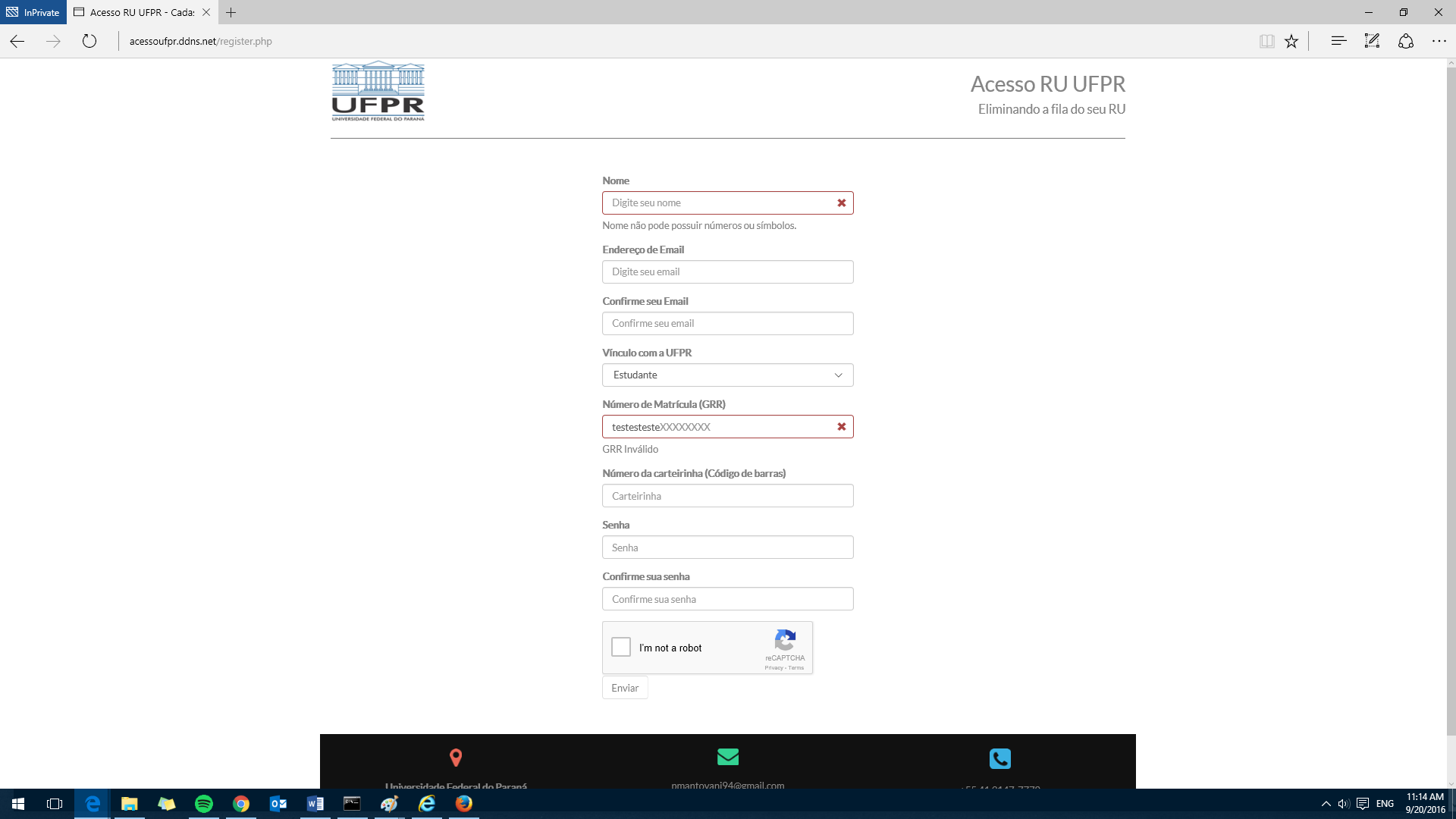
Durante este procedimento, encontramos apenas duas irregularidades, na página de cadastro. A primeira encontra-se nos campos numéricos do formulário no *Mozilla Firefox*. Nestes campos, removemos os *spinners* ao lado do campo para melhor visualização do usuário no nosso arquivo *CSS*. Contudo, este campo ainda aparece no *Mozilla*, conforme é apresentado na Figura 10.

Figura 10: *Spinners* ao lado dos campos numéricos

Fonte: Dos autores.

Além deste pequeno problema visual, também obtivemos outro problema 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 Figuras11, 12 e 13. 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, podemos introduzir um pequeno *script* em *Javascript* para deletar qualquer caractere que não seja numérico nestes campos.

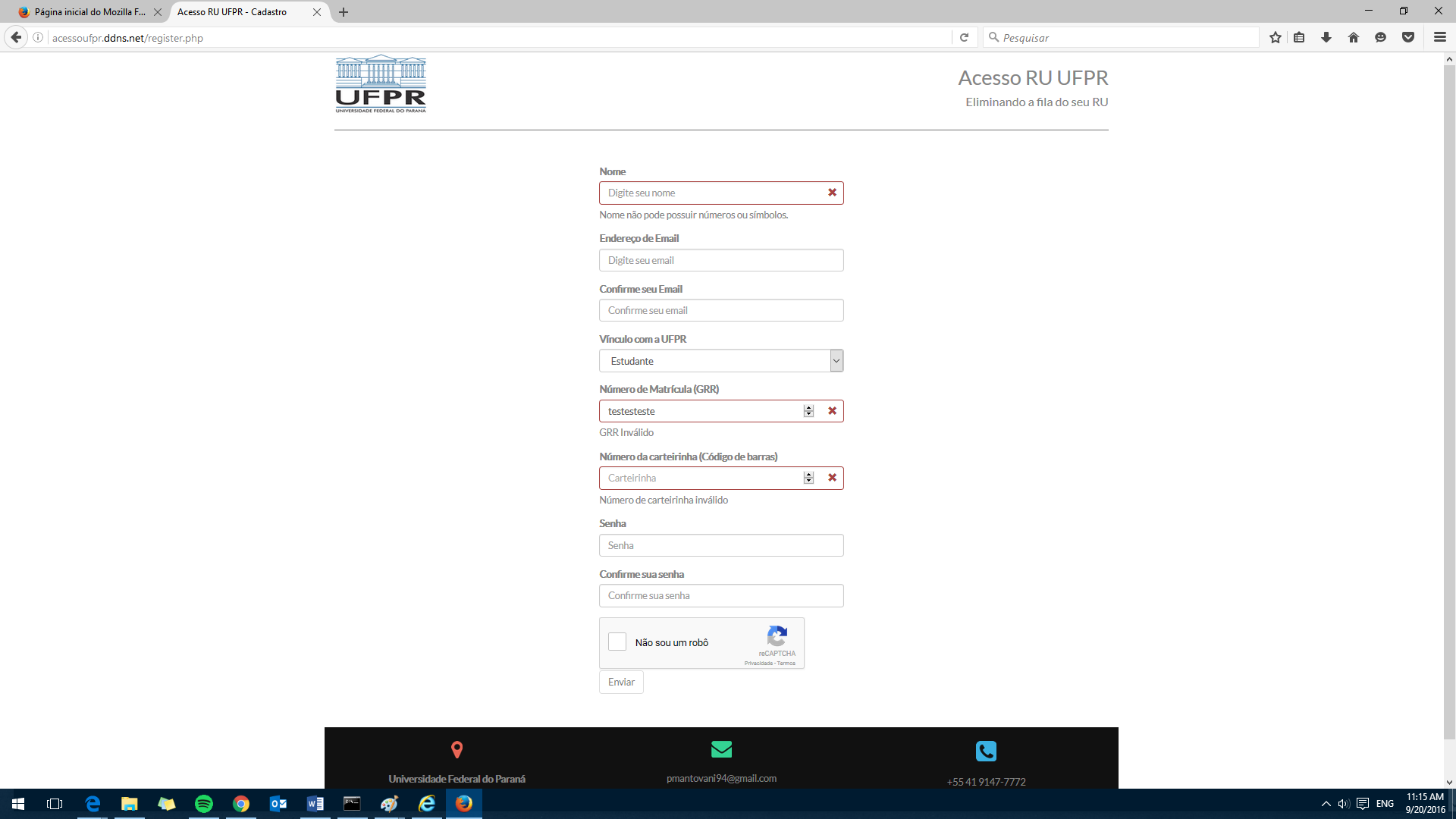
Figura 11: Campo de número de matrícula (*Edge*)



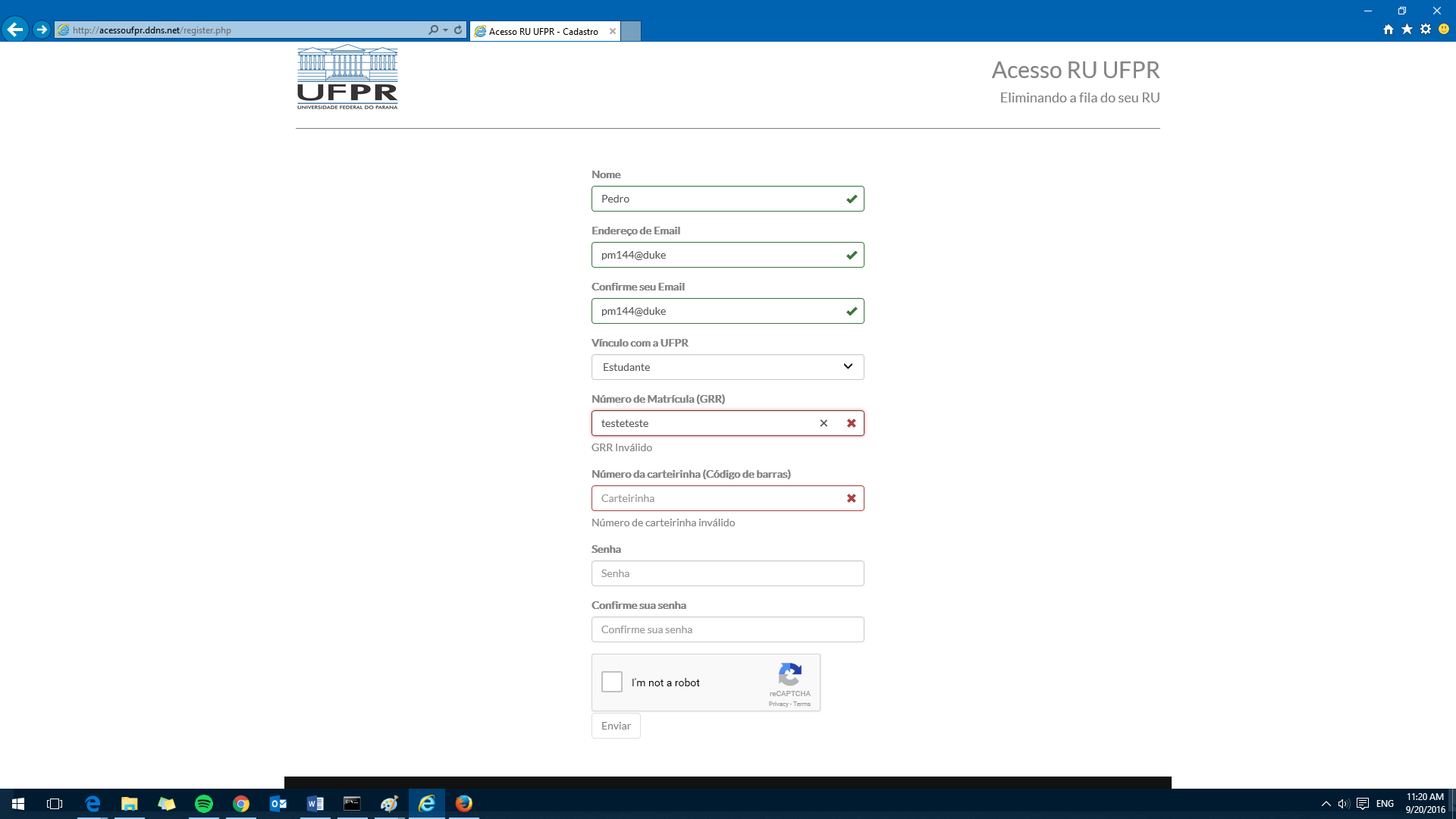
Fonte: Dos autores.

Figura 2: Campo de número de matrícula (*Firefox*)

Fonte: Dos autores.

Figura 13: Campo do número de matrícula (*Internet Explorer*)

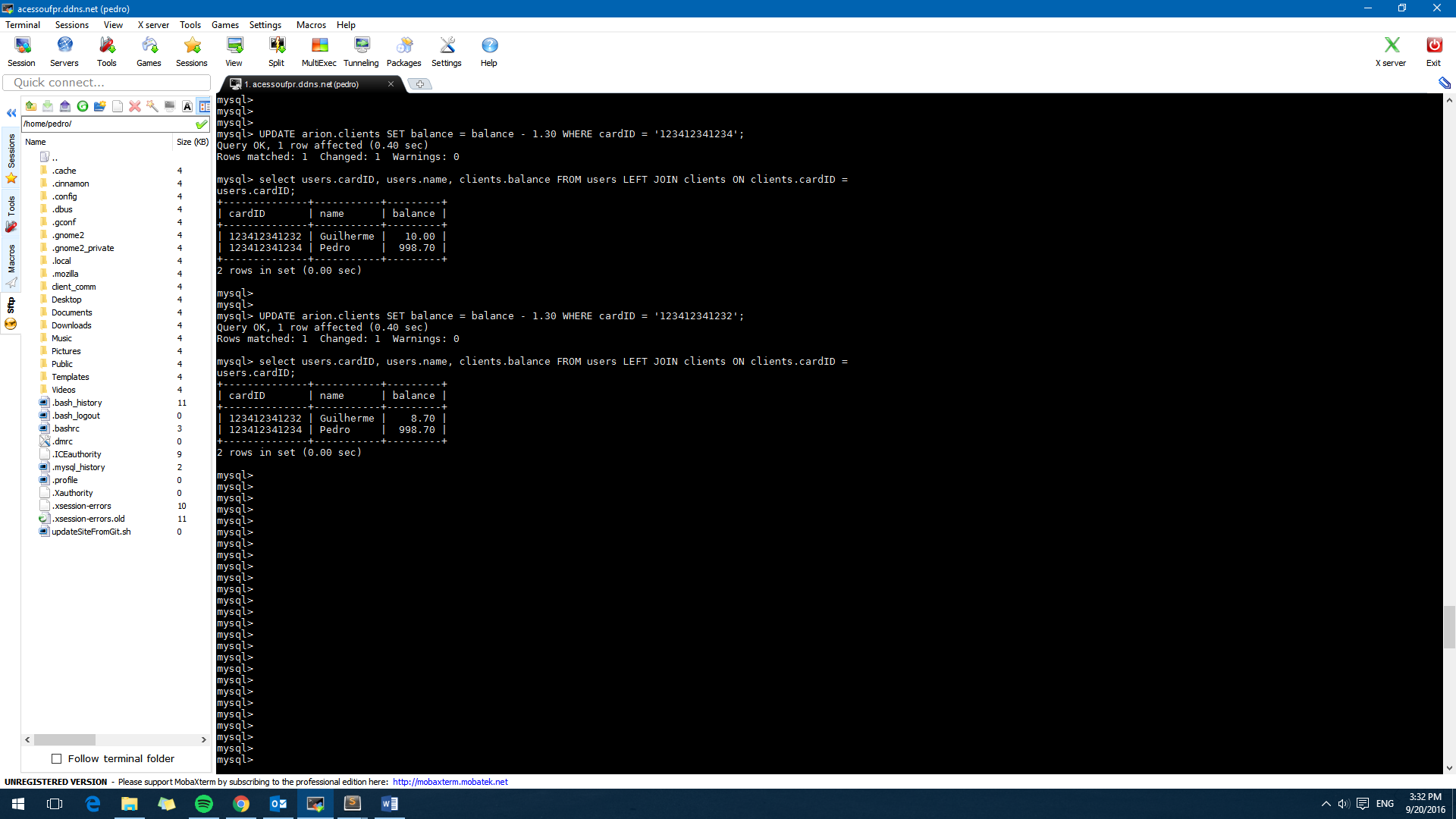
Fonte: Dos autores.

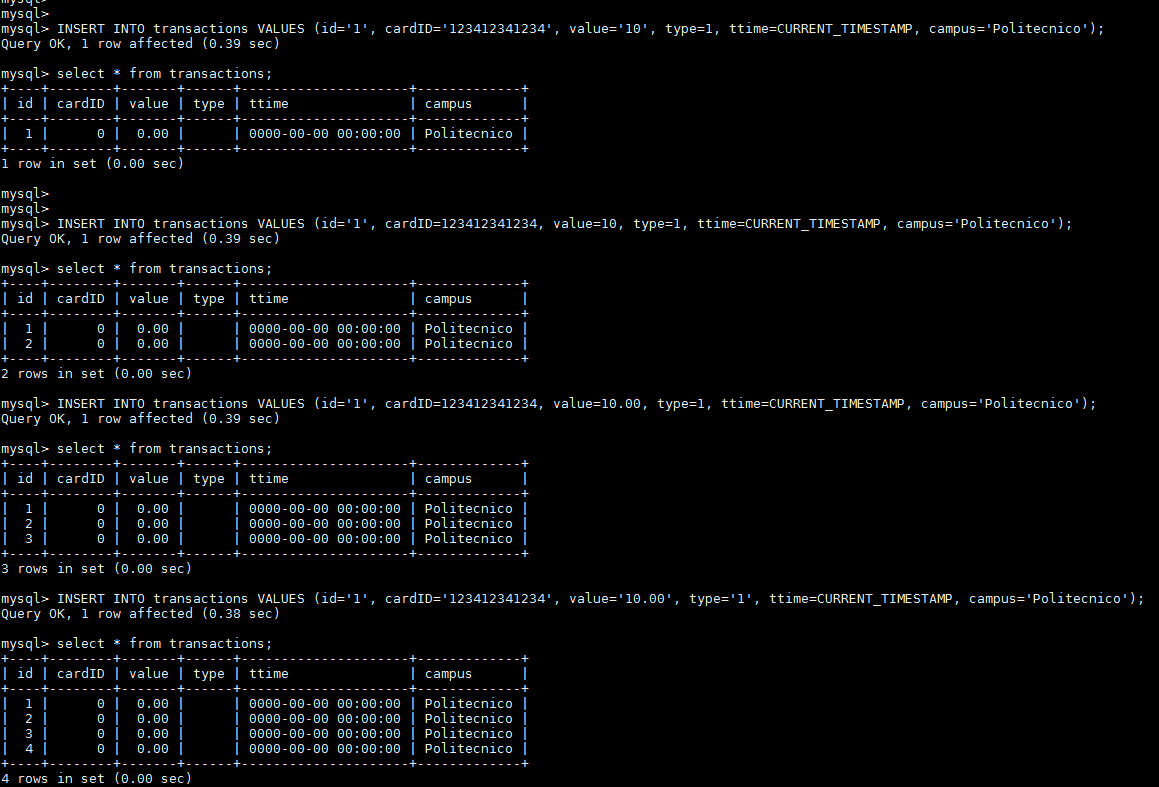
****Apesar destes pequenos detalhes, todas as páginas responderam como o programado, todas as animações funcionaram, e o conteúdo foi apresentado como desenhado. Além disso, todos os campos no formulário de cadastro foram validados corretamente, evitando possíveis inserções com erro na base de dados.

## Banco de Dados

Como dito na seção anterior, a estrutura atual do nosso banco de dados consiste de três tabelas. Para nosso teste, inserimos dois usuários teste na tabela “*users*”. Depois disso realizamos algumas consultas ao banco fazendo alguns *joins* entre as tabelas “*users*” e “*clients*”, de forma a mostrar também o saldo de cada usuário. A representação destas consultas e *updates* estão na Figura 14. Finalmente, inserimos algumas transações na tabela “*transactions*”, a fim de mostrar um *log* de todas transações ocorridas, mostradas na Figura 15, juntamente com o resultado de outras *queries*.

Figura 14: Resultado de *queries* nas tabelas “*users*” e “*clients*”

 Fonte: Dos autores.

Figura 15: Resultado das *queries* na tabela “transactions”

Fonte: Dos autores.

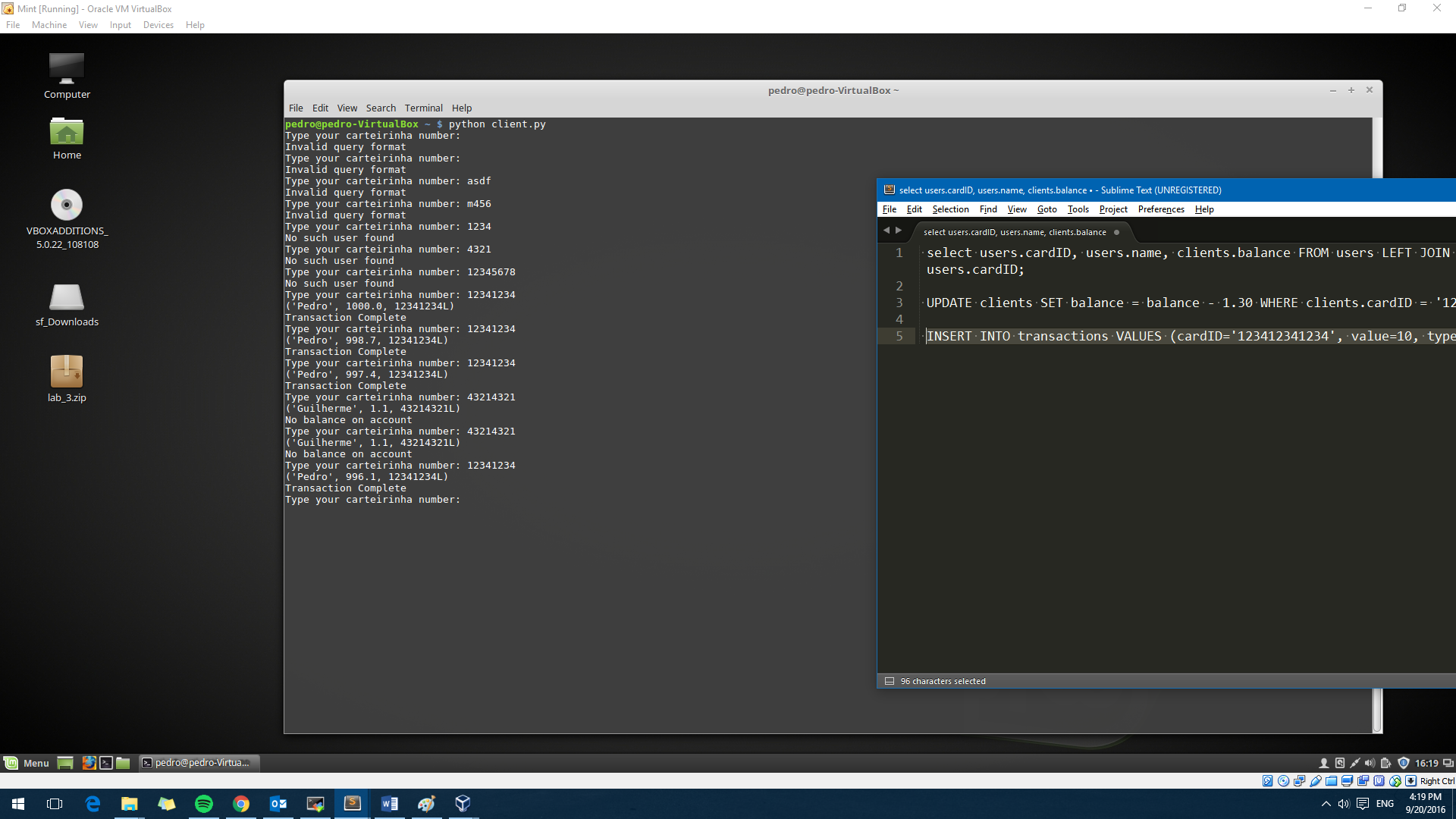
Como observado, as tabelas “*users*” e “*clients*” estão tratando os *updates* e *selects* de forma adequada, mudando o saldo do usuário conforme o indicado pelo *update*. Contudo, não conseguimos fazer com que a tabela “*transactions*” respondesse da maneira desejada. Como observado na figura 15, os resultados das consultas após as inserções mostraram-se erradas, com vários campos em branco. Este é um ponto que devemos retomar no futuro próximo buscando consertar este problema.

## 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, pretendemos 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 16, mostramos algumas simulações do programa em funcionamento. Nelas, o programa requere 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. Mostramos 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.

Figura 16: Simulação do funcionamento do programa das leitoras de acesso

Fonte: Dos autores.



# Conclusão

Ao fim dessa primeira 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 encontram-se 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.

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. Dentre os progressos, vale citar que a operação do servidor *Web* vem sendo extremamente satisfatória para o objetivo proposto.

No decorrer do desenvolvimento do projeto busca-se aprimorar todos os módulos envolvidos a fim de alcançar uma solução robusta, inovadora e útil para toda comunidade da UFPR.

## 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.

# 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.

MÁXIMO, MARCO A. S. **Servidor de email com SMTP, POP, IMAP, quota e MySQL.** VIVA O LINUX, 2003. Disponível em: <<https://www.vivaolinux.com.br/artigo/Servidor-de-email-com-SMTP-POP-IMAP-quota-e-MySQL/>> Acesso em 04 set. 2016.

UNIVERSITY OF LUXEMBOURG. **University Restaurant/University Restaurant Card.** Disponível em: <<http://wwwen.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.

# Anexos

## Script.sql

/\*

Script de criação de Base de Dados - Acesso UFPR

Erik Nayan & Pedro Mantovani

Versão 1.0 - 10 set 2016

\*/

CREATE DATABASE IF NOT EXISTS arion;

USE arion;

\r arion;

CREATE TABLE IF NOT EXISTS users (

cardID BIGINT(12) UNSIGNED NOT NULL,

name VARCHAR(50) NOT NULL,

email VARCHAR(50) NOT NULL,

password VARCHAR(20) NOT NULL,

grr INT(8) UNSIGNED NOT NULL,

type ENUM('Estudante','Professor','Servidor') NOT NULL,

regdate DATE DEFAULT '2017-01-01' NOT NULL,

status BIT DEFAULT 1 NOT NULL, -- 1=Ativo, 0=Inativo;

expiration DATE DEFAULT '2100-01-01' NOT NULL,

PRIMARY KEY (cardID)

);

CREATE TABLE IF NOT EXISTS clients (

cardID BIGINT(12) UNSIGNED NOT NULL,

balance DECIMAL(6,2) NOT NULL,

status BIT DEFAULT 1 NOT NULL, -- 1=Ativo, 0=Inativo;

PRIMARY KEY (cardID)

);

CREATE TABLE IF NOT EXISTS transactions (

id BIGINT UNSIGNED NOT NULL AUTO\_INCREMENT,

cardID BIGINT(12) UNSIGNED NOT NULL,

value DECIMAL(2,2) NOT NULL,

type BIT NOT NULL, -- 1=Crédito, 0=Desconto;

ttime TIMESTAMP DEFAULT CURRENT\_TIMESTAMP NOT NULL,

campus ENUM('Politecnico','Botanico','Agrarias','Reitoria') NOT NULL,

PRIMARY KEY (id)

);

CREATE USER IF NOT EXISTS 'scan'@'%' IDENTIFIED BY 'scan@acesso\_ufpr16!'; -- o scan pode acessar de qualquer máquina a DB

GRANT SELECT, INSERT, UPDATE ON arion.transactions to 'scan'@'%'; -- acesso somente a tabela apropriada

GRANT SELECT, INSERT, UPDATE ON arion.clients to 'scan'@'%'; -- acesso somente a tabela apropriada

CREATE USER IF NOT EXISTS 'form'@'%' IDENTIFIED BY 'form@acesso\_ufpr16!'; -- o form pode acessar de qualquer máquina a DB

GRANT SELECT, INSERT, UPDATE ON arion.users to 'form'@'%'; -- acesso somente a tabela apropriada

FLUSH PRIVILEGES;

## Header.php

<div class="row header">

<div class="col-xs-6">

<a href="/index.php"><img src="images/ufprlogo.png" class="img-responsive" alt="Logo UFPR"></a>

</div>

<div class="col-xs-6 text-right">

<h2>Acesso RU UFPR</h2>

<h4>Eliminando a fila do seu RU</h4>

</div>

</div>

<hr style="width: 100%; height: 1px; background-color:#868686;">

## Index.php

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="css/bootstrap.min.css">

<script src="js/jquery-3.1.0.min.js"></script>

<script src="js/bootstrap.min.js"></script>

<script src="https://use.fontawesome.com/f56e36ff4f.js"></script>

<link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet">

<link rel="stylesheet" href="css/style.css">

<link rel="stylesheet" href="css/hover-style.css">

<title>Acesso RU UFPR - Eliminando a fila do seu RU</title>

<meta name="description" content="O sistema de acesso oficial da UFPR">

</head>

<body>

<div class="container">

<!-- HEADER SECTION -->

<?php include("header.php"); ?>

<!-- IMAGE AND SHORT DESCRIPTION SECTION -->

<div class="row title-section text-center">

<div class="col-md-12 text-uppercase">

<h1 class="title">O sistema de acesso e pagamento oficial dos restaurantes da UFPR</h1>

</div>

<!-- BUTTONS SECTION -->

<div class="text-center buttons">

<div class="col-md-offset-3 col-md-3">

<a href="register.php" role="button" class="btn btn-primary button-margin button-register text-uppercase">Crie sua conta</a>

</div>

<div class="col-md-3">

<a href="login.php" role="button" class="btn btn-primary button-margin button-login text-uppercase">Faça seu login</a>

</div>

</div>

</div>

<!-- FEATURES TITLE SECTION -->

<div class="row text-center title-margin">

<div class="col-md-12 text-uppercase sub-title"><h2>Como funciona</h2></div>

</div>

<!-- FEATURES SECTION -->

<div class="row features text-center">

<div class="col-sm-3 feature">

<img src="images/form.svg" class="img-responsive center-block img-feature" alt="Cadastro">

<h3 class="text-uppercase sub-title">Cadastro da Carteirinha</h3><hr style="width: 50%; height: 3px; background-color:red;">

<p>Crie sua conta e cadastre sua carteirinha da UFPR aqui. O processo leva menos de 5 minutos.</p></div>

<div class="col-sm-3 feature">

<img src="images/money.svg" class="img-responsive center-block img-feature" alt="Créditos">

<h3 class="text-uppercase sub-title">Insira seus créditos</h3><hr style="width: 50%; height: 3px; background-color:green;">

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

<div class="col-sm-3 feature">

<img src="images/time.svg" class="img-responsive center-block img-feature" alt="Agilidade">

<h3 class="text-uppercase sub-title">Seu RU sem filas</h3><hr style="width: 50%; height: 3px; background-color:yellow;">

<p>Acesse o RU normalmente utilizando sua carteirinha, sem filas! O resto deixa que a gente cuida.</p></div>

<div class="col-sm-3 feature">

<img src="images/smartphone.svg" class="img-responsive center-block img-feature" alt="Mobile">

<h3 class="text-uppercase sub-title">Consulte seus créditos</h3><hr style="width: 50%; height: 3px; background-color:blue;">

<p>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.</p></div>

</div>

<!-- DEVELOPERS TITLE SECTION -->

<div class="row text-center title-margin">

<div class="col-md-12 text-uppercase sub-title"><h2>Desenvolvedores</h2></div>

</div>

<div class="row text-center title-margin">

<div class="col-md-12"><p>Como estudantes, entendemos a importância da praticidade e segurança no nosso cotidiano. Por isso, trabalhamos incansavelmente por uma plataforma fácil e segura para todos os estudantes da UFPR.</p></div>

</div>

<!-- DEVELOPERS SECTION -->

<div class="row developers text-center">

<div class="col-sm-6 developer">

<div class="hovereffect">

<img class="img-responsive center-block img-circle" src="images/pedro.png" alt="Imagem Pedro">

<div class="overlay">

<p class="info">Estudante de Engenharia Elétrica na UFPR desde 2013. Após longas horas de tempo perdido na fila do RU, Pedro começou a desenvolver o sistema de acesso para que outros estudantes não percam este mesmo tempo.</p>

</div>

<h3>Pedro Mantovani Antunes</h3><hr style="width: 50%; height: 3px; background-color:red;">

<h4 class="dev-desc-margin">Engenharia Elétrica - UFPR<br />Back-end e front-end developer</h4>

</div>

<div class="row developer-margin">

<div class="col-xs-3 col-xs-offset-3">

<a href="https://www.facebook.com/pedro.mantovani.12" target="\_blank"><i class="fa fa-facebook fa-2x social-net-btn" aria-hidden="true"></i></a>

</div>

<div class="col-xs-3">

<a href="https://www.linkedin.com/in/pedro-mantovani" target="\_blank"><i class="fa fa-linkedin fa-2x social-net-btn" aria-hidden="true"></i></a>

</div>

</div>

</div>

<div class="col-sm-6 developer">

<div class="hovereffect">

<img class="img-responsive center-block img-circle" src="images/erik.png" alt="Imagem Erik">

<div class="overlay">

<p class="info">Estudante de Engenharia Elétrica na UFPR desde 2013. Cansado de carregar moedas e de problemas com o troco, Erik começou a desenvolver o sistema de acesso para que tudo seja mais simples e ágil, utilizando o sistema de créditos.</p>

</div>

<h3>Erik Nayan</h3><hr style="width: 50%; height: 3px; background-color:green;">

<h4 class="dev-desc-margin">Engenharia Elétrica - UFPR<br />Back-end e front-end developer</h4>

</div>

<div class="row developer-margin">

<div class="col-xs-3 col-xs-offset-3">

<a href="https://www.facebook.com/eriknayan" target="\_blank"><i class="fa fa-facebook fa-2x social-net-btn" aria-hidden="true"></i></a>

</div>

<div class="col-xs-3">

<a href="https://br.linkedin.com/in/eriknayan/en" target="\_blank"><i class="fa fa-linkedin fa-2x social-net-btn" aria-hidden="true"></i></a>

</div>

</div>

</div>

</div>

<!-- TESTIMONY TITLE SECTION -->

<div class="row testimonial-section">

<div class="text-center title-margin">

<div class="col-md-12 text-uppercase sub-title testimonial-title"><h2>Testemunhos</h2></div>

</div>

<div class="text-center">

<div class="col-md-12 testimonial-subtitle">Veja o que os outros estudantes estão dizendo da plataforma.</div>

</div>

<!-- TESTIMONY SECTION -->

<div class="text-center margin-bottom-30">

<div class="col-md-4 testimonial">

<div class="col-xs-offset-1 col-xs-10 margin-20">

<div class="row white-bg img-rounded padding-25">

<div class="col-xs-12">

<p>Com o novo sistema de acesso, economizo pelo menos 30 minutos todos os dias. Mais tempo para estudar cálculo e reprovar da mesma forma!</p>

</div>

<div class="col-xs-6">

<p class="margin-top-20"> - Engenheiro</p>

</div>

<div class="col-xs-6">

<img src="images/rambo.jpg" alt="Engenheiro" class="img-responsive img-circle pull-right img-testimonial">

</div>

</div>

</div>

</div>

<div class="col-md-4 testimonial">

<div class="col-xs-offset-1 col-xs-10 margin-20">

<div class="row white-bg img-rounded padding-25">

<div class="col-xs-12">

<p>Acredito que o sistema de cotas para alunos que usem a plataforma de acesso ao RU seja uma proposta muito interessante. Viva a igualdade!</p>

</div>

<div class="col-xs-6">

<p class="margin-top-20"> - Historiadora</p>

</div>

<div class="col-xs-6">

<img src="images/micanga.jpg" alt="Historiadora" class="img-responsive img-circle pull-right img-testimonial">

</div>

</div>

</div>

</div>

<div class="col-md-4 testimonial">

<div class="col-xs-offset-1 col-xs-10 margin-20">

<div class="row white-bg img-rounded padding-25">

<div class="col-xs-12">

<p>O sistema me ajuda a tornar meu almoço mais ágil para que eu possa voltar a programar meu hack do GTA.</p>

</div>

<div class="col-xs-6">

<p class="margin-top-20"> - Programador</p>

</div>

<div class="col-xs-6">

<img src="images/nerdao.jpg" alt="Programador" class="img-responsive img-circle pull-right img-testimonial">

</div>

</div>

</div>

</div>

</div>

</div>

<!-- FOOTER SECTION -->

<?php include("footer.php") ?>

</div>

</body>

</html>

## Footer.php

<!-- FOOTER SECTION -->

<div class="row footer text-center">

<div class="col-md-4">

<a href="http://maps.google.com/?q=-25.449753,-49.232942" target="\_blank">

<img src="images/pin.png" class="img-responsive center-block margin-20" alt="Pino no mapa">

<address>

<strong>Universidade Federal do Paraná</strong><br>

Centro Politécnico<br>

Curitiba, PR

</address>

</a>

</div>

<div class="col-md-4">

<a href="mailto:pmantovani94@gmail.com">

<img src="images/envelope.png" class="img-responsive center-block margin-20" alt="Contato email">

<p>pmantovani94@gmail.com</p>

</a>

</div>

<div class="col-md-4">

<a href="tel:+554191477772">

<img src="images/telephone.png" class="img-responsive center-block margin-20" alt="Telefone">

<p>+55 41 9147-7772</p>

</a>

</div>

</div>

## Login.php

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="css/bootstrap.min.css">

<script src="js/jquery-3.1.0.min.js"></script>

<script src="js/bootstrap.min.js"></script>

<link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet">

<link rel="stylesheet" href="css/style.css">

<title>Acesso RU UFPR - Cadastro</title>

<meta name="description" content="O sistema de acesso oficial da UFPR">

</head>

<body>

<div class="container">

<!-- HEADER SECTION -->

<?php include("header.php"); ?>

<!-- LOGIN FIELDS -->

<div class="row form-margin">

<div class="col-sm-4 col-sm-offset-4">

<form role="form" method="post">

<div class="form-group">

<label for="emailInput">Endereço de Email</label>

<input type="email" name="name" class="form-control" id="emailInput" placeholder="Digite seu email" autofocus>

</div>

<div class="form-group">

<label for="passwordInput">Senha</label>

<input type="password" name="passwd" class="form-control" id="passwordInput" placeholder="Senha">

</div>

<div class="checkbox">

<label><input type="checkbox">Lembre-se do usuário</label>

</div>

<button type="submit" class="btn btn-default">Login</button>

</form>

</div>

</div>

<!-- FOOTER SECTION -->

<?php include("footer.php") ?>

</body>

## Register.php

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="css/bootstrap.min.css">

<script src="js/jquery-3.1.0.min.js"></script>

<script src="js/bootstrap.min.js"></script>

<link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet">

<link rel="stylesheet" href="css/style.css">

<script src="js/validator.min.js"></script>

<script src="js/jquery-3.1.0.min.js"></script>

<script src='https://www.google.com/recaptcha/api.js' async></script>

<title>Acesso RU UFPR - Cadastro</title>

<meta name="description" content="O sistema de acesso oficial da UFPR">

</head>

<body>

<div class="container">

<!-- HEADER SECTION -->

<?php include("header.php"); ?>

<!-- REGISTRATION FIELDS -->

<div class="row form-margin">

<div class="col-sm-4 col-sm-offset-4">

<form action="result\_registration.php" method="post" data-toggle="validator" role="form">

<!-- NOME -->

<div class="form-group has-feedback">

<label for="nameInput">Nome</label>

<input type="text" name="name" class="form-control" pattern="[A-Za-z\x20áàãâéèêóòõô]{1,}" id="nameInput" data-error="Nome não pode possuir números ou símbolos." placeholder="Digite seu nome" required autofocus>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- EMAIL -->

<div class="form-group has-feedback">

<label for="emailInput">Endereço de Email</label>

<input type="email" name="email" class="form-control" id="emailInput" data-error="Email inválido" placeholder="Digite seu email" required>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- CONFIRME EMAIL -->

<div class="form-group has-feedback">

<label for="emailInput">Confirme seu Email</label>

<input type="email" class="form-control" id="emailInput" data-match="#emailInput" data-match-error="Emails não conferem" placeholder="Confirme seu email" required>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- VINCULO -->

<div class="form-group">

<label>Vínculo com a UFPR</label>

<select class="form-control" name="role">

<option>Estudante</option>

<option>Professor</option>

<option>Servidor</option>

</select>

</div>

<!-- GRR -->

<div class="form-group has-feedback">

<label for="grrInput">Número de Matrícula (GRR)</label>

<input type="number" name="grr" class="form-control" id="grrInput" placeholder="XXXXXXXX" min="10000000" max="99999999" maxlength="8" oninput="maxLengthCheck(this)" data-error="GRR Inválido" required/>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- NUM. CARTEIRINHA -->

<div class="form-group has-feedback">

<label for="grrInput">Número da carteirinha (Código de barras)</label>

<input type="number" name="barcode" class="form-control" placeholder="Carteirinha" min="100000000000" max="999999999999" maxlength="12" oninput="maxLengthCheck(this)" data-error="Número de carteirinha inválido" required/>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- SENHA -->

<div class="form-group has-feedback">

<label for="passwordInput">Senha</label>

<input type="password" name="passwd" class="form-control" pattern="(?=.\*\d)(?=.\*[a-z])(?=.\*[A-Z]).{6,}" data-error="Senha deve conter no mínimo um número, uma letra maiúscula e uma minúscula, e 6 ou mais caracteres." id="passwordInput" placeholder="Senha" required>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- CONFIRMAÇÃO DE SENHA -->

<div class="form-group has-feedback">

<label for="passwdConfirmInput">Confirme sua senha</label>

<input type="password" class="form-control" data-match="#passwordInput" data-match-error="Senhas não conferem" placeholder="Confirme sua senha" required>

<span class="glyphicon form-control-feedback" aria-hidden="true"></span>

<div class="help-block with-errors"></div>

</div>

<!-- BOTÃO ENVIAR -->

<div class="g-recaptcha" data-sitekey="6Lc\_bikTAAAAAP2BKGgqMG1jKeYRwsJi-SLWT2yL"></div>

<button type="submit" class="btn btn-default" id="submitButton">Enviar</button>

</form>

</div>

</div>

<!-- FOOTER SECTION -->

<?php include("footer.php") ?>

</body>

<script type="text/javascript">

function maxLengthCheck(object) {

if(object.value.length > object.maxLength)

object.value = object.value.slice(0, object.maxLength)

}

</script>

<script type="text/javascript">

$(document).ready(function() {

$("#submitButton").click(function() {

if (grecaptcha.getResponse() == ""){

alert("Por favor, valide o captcha para continuar.");

return false;

}

})

})

</script>

</html>

## Style.css

\* {

font-family: 'Lato', sans-serif;

color: #848484;

}

.title-section {

background: linear-gradient( rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5) ), url(../images/ufpr2.jpg) no-repeat center center fixed;

-webkit-background-size: cover;

-moz-background-size: cover;

-o-background-size: cover;

background-size: cover;

min-height: 500px;

}

.title {

margin-top:200px;

color: #FFFFFF;

font-size: 40px;

font-weight: 900;

}

.buttons {

margin-top:80px;

margin-bottom: 40px;

}

.button-margin {

margin-top: 20px;

padding-right: 35px;

padding-left: 35px;

padding-top: 13px;

padding-bottom: 13px;

font-size: 16px;

}

.title-margin {

margin-top:40px;

margin-bottom: 40px;

}

.sub-title {

color: #000000;

}

.form-margin {

margin-top:50px;

margin-bottom:50px;

}

.img-feature {

max-height: 150px;

}

.footer {

margin-top: 40px;

background-color: #111111;

}

a {

color: #848484;

}

.margin-20 {

margin-top:20px;

margin-bottom: 20px;

}

.padding-25 {

padding: 25px;

}

.margin-top-20 {

margin-top: 20px;

}

.margin-bottom-30 {

margin-bottom: 30px;

}

.button-register {

background-color: #e96656;

border-color: #e96656;

-webkit-transition: all 0.3s ease-in-out;

transition: all 0.3s ease-in-out;

transition-property: all;

transition-duration: 0.3s;

transition-timing-function: ease-in-out;

transition-delay: initial;

font-weight: lighter;

}

/\* Changes background color when focused \*/

.button-register:hover, .button-register:focus {

background-color: #9B4439;

border-color: #9B4439;

}

.button-login {

background-color: #1e9e6b;

border-color: #1e9e6b;

-webkit-transition: all 0.3s ease-in-out;

transition: all 0.3s ease-in-out;

transition-property: all;

transition-duration: 0.3s;

transition-timing-function: ease-in-out;

transition-delay: initial;

font-weight: lighter;

}

.button-login:hover, .button-login:focus {

background-color: #146947;

border-color: #146947;

}

.testimonial-section {

background: #8c7620;

}

.testimonial-title {

padding-top: 40px;

color: white;

}

.testimonial-subtitle {

color: white;

margin-bottom: 40px;

}

.img-testimonial {

max-width: 70px;

}

.developer-margin {

margin-bottom: 40px;

}

.dev-desc-margin {

margin-bottom: 20px;

}

.social-net-btn {

color: #848484;

border-color: #848484;

-webkit-transition: all 0.3s ease-in-out;

transition: all 0.3s ease-in-out;

transition-property: all;

transition-duration: 0.3s;

transition-timing-function: ease-in-out;

transition-delay: initial;

}

.social-net-btn:hover, .social-net-btn:focus {

color: #337ab7;

border-color: #337ab7;

}

.white-bg {

background-color: white;

}

/\* Removes spinner for number fields on registration \*/

input::-webkit-outer-spin-button,

input::-webkit-inner-spin-button {

/\* display: none; <- Crashes Chrome on hover \*/

-webkit-appearance: none;

margin: 0; /\* <-- Apparently some margin are still there even though it's hidden \*/

}

## Client.py

import SocketServer

import MySQLdb

PRICE = 1.30

HOST\_DB = "acessoufpr.ddns.net"

USER\_DB = "XXX"

PASSWD\_DB = "XXX"

NAME\_DB = "acessoufpr"

if \_\_name\_\_ == "\_\_main\_\_":

while True:

try:

num = raw\_input('Type your carteirinha number: ')

except SyntaxError: # Case input was left blank

continue

try:

str\_num = long(num) # Tries and converts input to long

except ValueError: # User probably typed a string, reiterate

print 'Invalid query format'

continue

db = MySQLdb.connect( host=HOST\_DB,

user=USER\_DB,

passwd=PASSWD\_DB,

db=NAME\_DB)

cur = db.cursor()

# Query the db

cur.execute("SELECT \* FROM Users WHERE id=%s", (str\_num,))

rows = cur.fetchall()

# Check if found any match in the query

if (len(rows) != 0):

for row in rows:

print row

# Extract fields from db

my\_id = row[0]

name = row[1]

email = row[2]

balance = row[3]

# Check if there is enough balance

if (balance > PRICE):

# Transaction successful

msg = "Hi "+ name + ". Your new balance is : " + str(balance-PRICE)

balance -= PRICE # Update balance

cur.execute("UPDATE Users SET balance=%s WHERE name=%s",(balance,name))

db.commit() # Commit changes to db

print "Transaction Complete"

else:

# User has no balance available

msg = "NO BALANCE"

print "No balance on account"

pass

else:

# No such user found on db

msg = "NO USER"

print "No user found"