Carolina Rocha Martins de Almeida
Igor Luciano de Lima Moreira

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE ACESSO UTILIZANDO ARDUINO E A TECNOLOGIA RFID

INCONFIDENTES – MG

2017

Carolina Rocha Martins de Almeida
Igor Luciano de Lima Moreira

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE ACESSO UTILIZANDO ARDUINO E A TECNOLOGIA RFID

Trabalho de Conclusão de Curso apresentado como pré-requisito de conclusão do curso de Graduação Tecnológica em Redes de Computadores no Instituto Federal de Educação, Ciência e Tecnologia do Sul de Minas Gerais – Campus Inconfidentes, para obtenção do título de Tecnólogo em Redes de Computadores.

Orientador: Ivan Paulino Pereira

INCONFIDENTES-MG
2017

Carolina Rocha Martins de Almeida
Igor Luciano de Lima Moreira

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE ACESSO UTILIZANDO ARDUINO E A TECNOLOGIA RFID

Data de aprovação:
__/__/___

Orientador: Prof. Ivan Paulino Pereira
IFSULDEMINAS – Campus Inconfidentes

Prof. Kleber Marcelo da Silva Rezende
IFSULDEMINAS – Campus Inconfidentes

Prof. Alessandro de Castro Borges
IFSULDEMINAS – Campus Inconfidentes

Instituto Federal de Educação, Ciência e Tecnologia
Sul de Minas Gerais
Campus Inconfidentes
Praça Tiradentes, 416 - Centro - CEP 37576-000
Telefone: (35) 3464-1200

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE ACESSO UTILIZANDO ARDUINO E A TECNOLOGIA RFID

Carolina R. M. de ALMEIDA[1]; Igor L. de L. MOREIRA[1]; Ivan P. PEREIRA[1]

[1]Setor de Informática e Redes IFSULDEMINAS – Campus Inconfidentes – Inconfidentes, MG – Brasil

carolinarocha32@gmail.com, igrlucianol@gmail.com, ivan.pereira@ifsuldeminas.edu.br

Abstract. *This work shows a web system development for students access control in a scholar restaurant located in IFSULDEMINAS - INCONFIDENTES. For the system implement some utensils have been used, like PHP programming language, the radio frequency identification (RFID) and the open source electronic prototype platform Arduino. The system can manage the restaurant in ticket sales, access control and reports making.*

Resumo. *Este trabalho apresenta o desenvolvimento de um sistema web para controle de acesso dos consumidores do Restaurante Estudantil no IFSULDEMINAS - Campus Inconfidentes. Para a implementação do sistema foram utilizadas a linguagem de programação PHP, a tecnologia de identificação por radiofrequência - RFID e a plataforma aberta de prototipagem eletrônica Arduino. O sistema desenvolvido neste trabalho permite realizar a gerência do Restaurante Estudantil por meio da venda de tíquetes, do controle de acesso e da geração de relatórios gerenciais.*

1. Introdução

Sistemas de informação baseados na tecnologia web (*Web-based Information System - WIS*) têm sido desenvolvidos com o objetivo de reduzir a necessidade de trabalho humano na produção de bens e serviços. Os WISs podem trabalhar integrados com outros sistemas não baseados em web como, por exemplo, os sistemas de controle de acesso (TONIN; CITTOLIN; SOUZA, 2015).

Segundo Galhardo (2011), os sistemas de controle de acesso possibilitam gerenciar o acesso de pessoas, onde seja necessário o controle seletivo de entrada ou o

controle estatístico de movimentação, proporcionando maior segurança em locais de acesso restrito.

De acordo com informações disponibilizadas pela Coordenação Geral de Assistência ao Educando – CGAE, responsável pelo Restaurante Estudantil do IFSULDEMINAS – Campus Inconfidentes, o Restaurante Estudantil serve diariamente cerca de 500 refeições para alunos, funcionários e convidados, sendo que o controle de acesso é realizado manualmente, o que pode causar problemas como acesso de consumidores que perderam o vínculo com o campus e a falta de estimativas reais para preparação do número de refeições.

Diante do exposto, este trabalho apresenta a análise e implementação de um sistema de controle de acesso para o Restaurante Estudantil integrado a web, utilizando a tecnologia de identificação por radiofrequência (*Radio Frequency Identification - RFID*) e a plataforma aberta de prototipagem eletrônica Arduino.

O RFID é uma tecnologia sem fio destinada a coleta de dados, semelhante ao código de barras, contudo esta tecnologia possibilita a leitura dos cartões (*tags*) sem necessidade de linha de visão entre os cartões e o leitor (*transponder*). Por sua vez, o Arduino é um pequeno computador que pode ser programado para tratar entradas e saídas de outros dispositivos como sensores dos mais variados tipos ou qualquer outro dispositivo que emita dados ou possa ser controlado (MCROBERTS, 2011). Com ele é possível desenvolver objetos capazes de interagir com o ambiente por meio de hardware e software. O hardware consiste de uma placa microcontroladora e o software de uma linguagem de programação responsável por controlar as ações do Arduino.

Segundo Mork (2013), as placas Arduino tem sido utilizadas com sucesso em vários projetos baseados em microcontroladores, uma vez que apresentam baixo custo e podem ser integradas com uma série de acessórios como os leitores RFID, os quais dispõem de alguns benefícios como durabilidade, facilidade de leitura e reutilização dos cartões em outros sistemas.

2. Material e Métodos

A metodologia adotada para o desenvolvimento do sistema foi o *Extreme Programming* - XP, pois, de acordo com Libardi e Barbosa (2010), ela viabiliza a implementação rápida do projeto e favorece no cumprimento das estimativas. Além disso, essa metodologia permite que eventuais problemas possam ser resolvidos rapidamente, pois o desenvolvimento dos módulos é feito através de *feedback* do cliente, ou seja, a maleabilidade do desenvolvimento permite implementar apenas o que é necessário para atender a necessidade do cliente. O processo de desenvolvimento iniciou-se com o levantamento de requisitos, nesta etapa foram realizadas reuniões e entrevistas com os interessados no sistema (*stakeholdes*), a fim de levantar as funcionalidades e restrições impostas ao sistema de controle de acesso.

O resultado desta fase foi a criação dos artefatos de software: diagramas de Modelo de Processo de Negócio (*Business Process Model and Notation - BPMN*) e diagrama de Casos de Uso, que permitiram compreender os processos e identificar os requisitos que envolvem o controle de acesso do Restaurante. A figura 01 apresenta o diagrama BPMN, este diagrama permite entender as interações que acontecem entre os módulos do sistema.

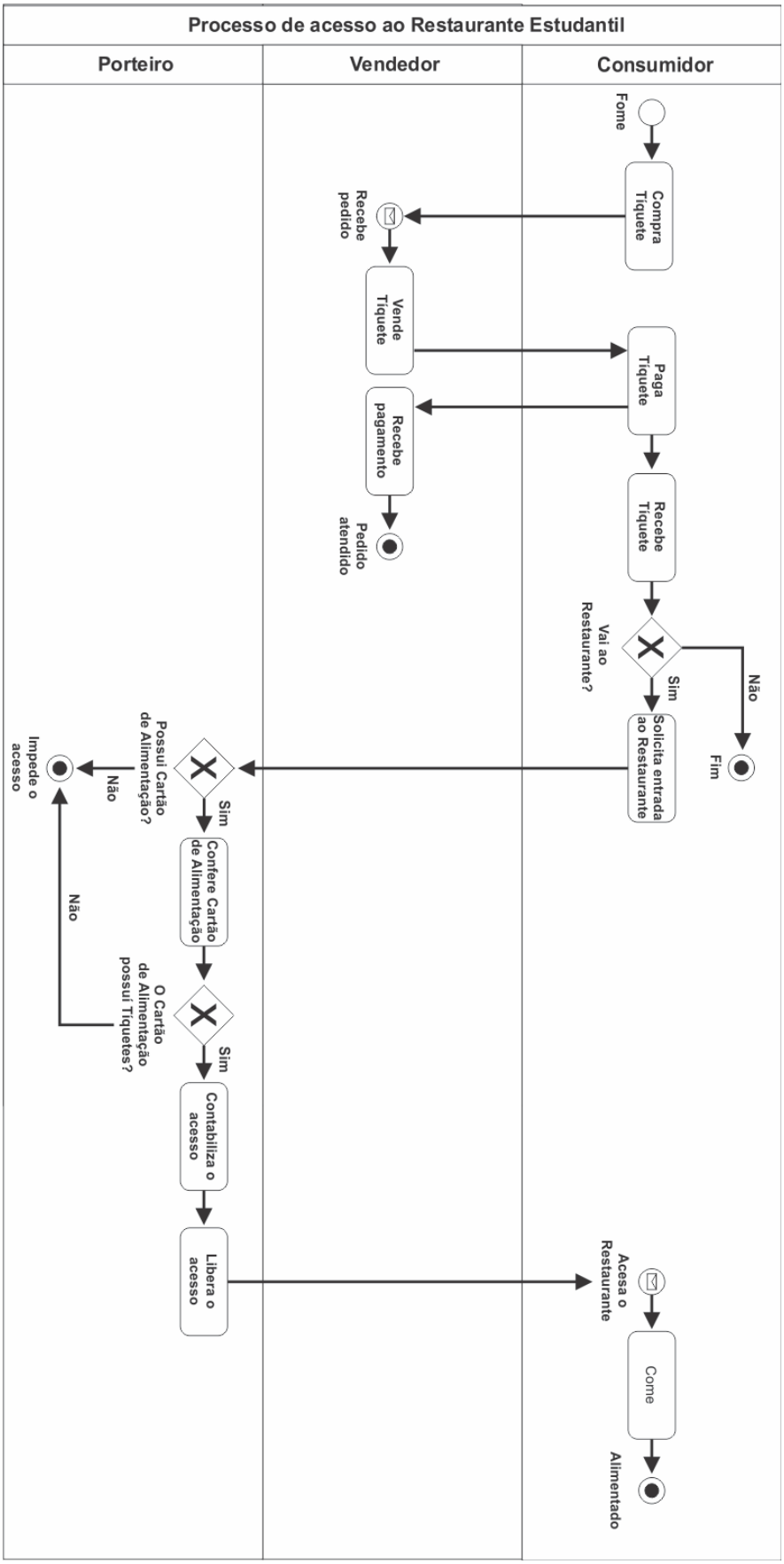

Figura 1: Diagrama BPMN representação do processo de acesso ao Restaurante Estudantil. Fonte: Autoria própria.

A etapa seguinte consistiu na análise e projeto do sistema. Nesta etapa, foram identificados os requisitos de maior valor, os quais foram implementados de forma interativa e incremental. Dessa forma, o sistema foi desenvolvido em pequenos ciclos interativos, onde os requisitos eram refinados, implementados e testados, e ao final um incremento (nova funcionalidade) era agregada ao projeto. Na fase de análise e projeto foram desenvolvidos os diagramas de classes e de entidade e relacionamentos - DER. Esses diagramas são apresentados a seguir nas figuras 2 e 3 respectivamente.

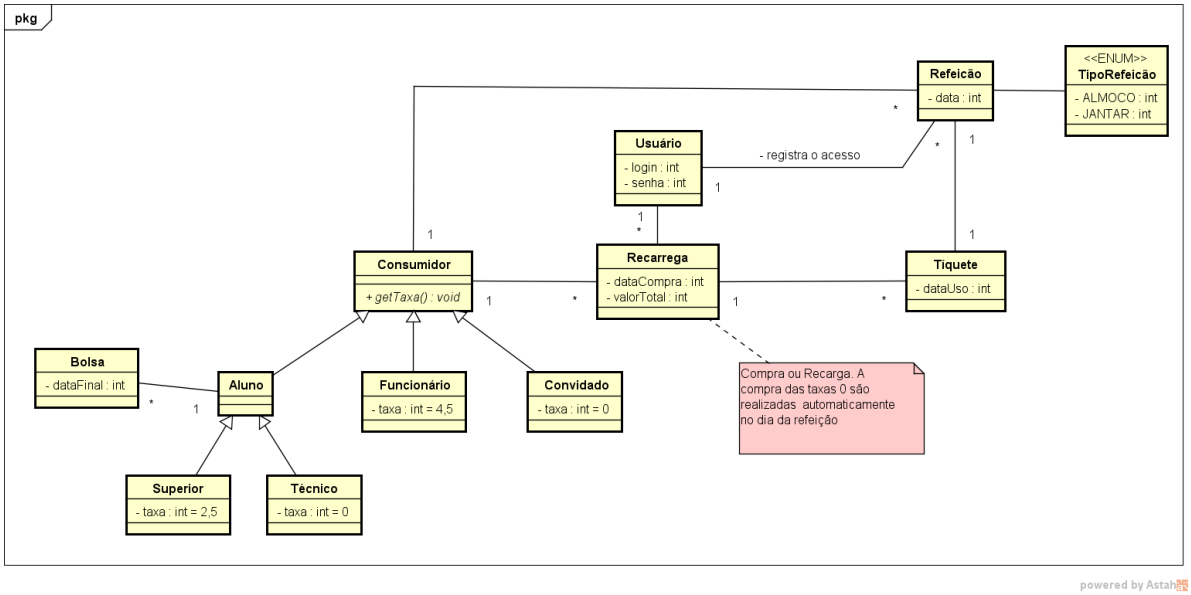

Figura 2: Diagrama de Classes, construído com o Software Astah Community. Fonte: Autoria própria.

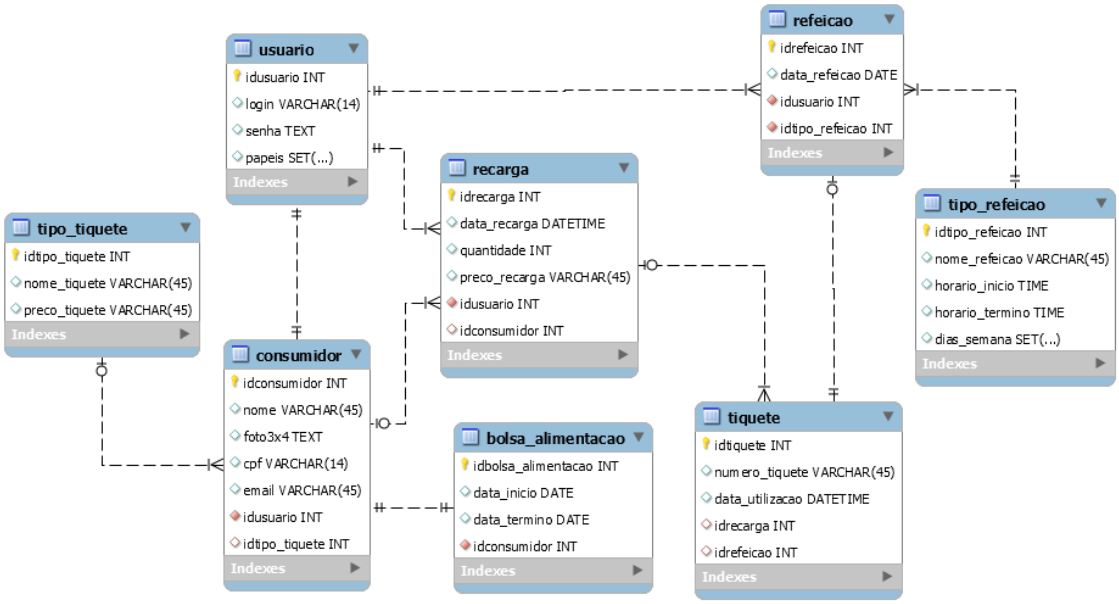

Figura 3: Diagrama de Entidade de Relacionamento, construído pelo Software MySQL Workbench. Fonte: Autoria própria..

Por se tratar de um sistema web foi utilizada a linguagem de marcação de hipertexto (*HyperText Markup Language - HTML*) e a folha de estilo em cascata (*Cascading Style Sheets - CSS*). Para aumentar a produtividade do desenvolvimento foi utilizado o

framework Bootstrap, uma vez que ele simplifica o desenvolvimento web e permite a criação de páginas visualmente bonitas. No *server-side* foi utilizada a linguagem de programação PHP (*Hypertext PreProcessor*) com o *framework* Codeigniter e a biblioteca GroceryCRUD.

O Codeigniter implementa o padrão arquitetural de Modelo, Visão e Controle (*Model-View-Controller - MVC*) que faz a separação de responsabilidades, melhorando a produtividade e reduzindo a complexidade do desenvolvimento. De forma, na camada *Model* são definidas as classes de modelo ou entidades e que fazem as interações com o Banco de Dados, a camada *View* é onde estão contidos arquivos de interface gráfica que interagem com o usuário, por fim a camada *Controller* é responsável por gerenciar e coordenar as camadas *Model* e *View*. Os Anexos 1, 2 e 3 apresentam a implementação da classe Consumidor que implementa o padrão arquitetural MVC do Codeigniter.

A biblioteca GroceryCRUD possibilita criar, de forma segura e com poucas linhas de código, formulários de cadastros comumente presentes em sistemas web. O Sistema Gerenciador de Banco de Dados (SGBD) escolhido foi o MySQL devido á fácil integração com o PHP.

A tecnologia RFID foi utilizada como forma de identificação dos consumidores por meio da leitura de um Cartão Alimentação feita utilizando a plataforma Arduino em conjunto com um leitor RFID. A comunicação do sistema web com hardware responsável por realizar a leitura dos cartões RFID foi implementada por meio da linguagem JavaScript que obtinha os códigos de RFID lidos pelo Arduino e os transmitia para o sistema web.

Para realizar a prototipagem do projeto foi necessário inserir o *Shield Ethernet* sobre o Arduino UNO. As portas lógicas do *Shield Ethernet* e do Arduino de entrada e saída tem a mesma sequência, podendo ser utilizadas da mesma forma.

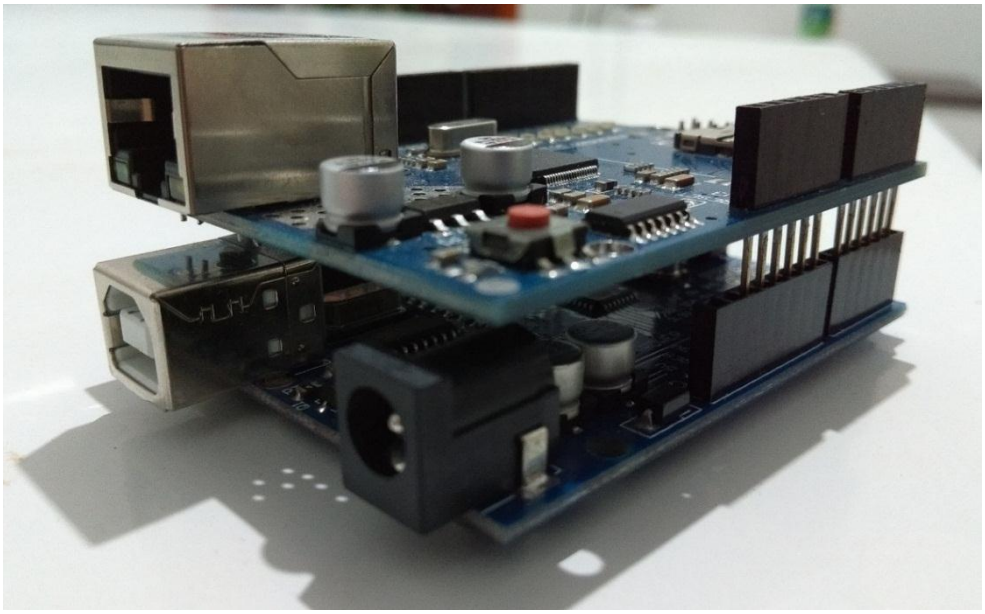

Figura 4: Arduino UNO acoplada com Shield Ethernet. Fonte: Autoria própria.

Após feito o acoplamento do Arduino com o *Shield Ethernet* é necessário fazer a ligação do circuito junto ao leitor RFID RC522, seguindo as ligações da figura 5, para que possa receber as informações dos Cartões de Alimentação.

Pinos do módulo RFID RC522	Arduino
3.3V	Pino 3.3V
RST	Pino 5
GND	Pino GND
IRQ ou NC	Não conectado
MISO	Pino 12
MOSI	Pino 11
SCK	Pino 13
SDA ou SS	Pino 9

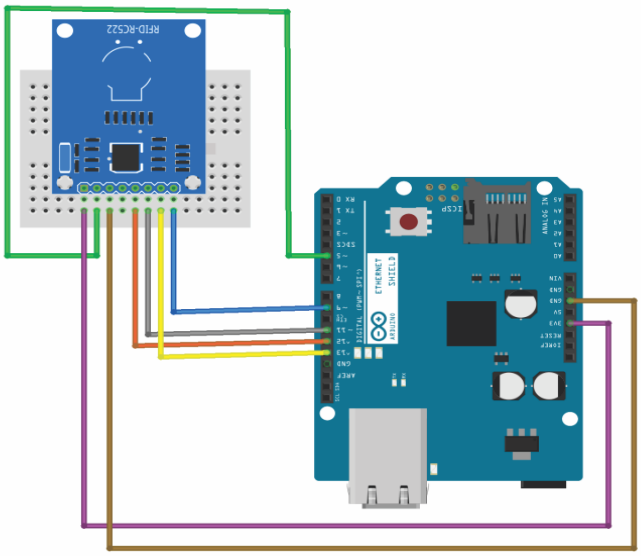

Figura 5: Ligações Arduino/Shield Ethernet com RFID RC522, construído pelo Software Fritzing. Fonte: Autoria própria.

Feitas as ligações do circuito e conectado na alimentação de energia, o protótipo está pronto para realizar a leitura do Cartão Alimentação. A figura 6 apresenta o código de leitura do Arduino, onde é lida a *Tag* UID (Identificação do Usuário) do Cartão Alimentação e exibida na tela em seguida.

```
void Leitura(){
  IDtag = "";
  if (!LeitorRFID.PICC_IsNewCardPresent() || !LeitorRFID.PICC_ReadCardSerial())
  {
    delay(50);
    return;
  }
  for (byte i = 0; i < LeitorRFID.uid.size; i++){
  IDtag.concat(String(LeitorRFID.uid.uidByte[i], HEX));
  }
  Serial.println("Tag Cadastrada: " + IDtag);
}

```

Figura 6: Trecho de código responsável pela leitura das Tags RFID programado com o Arduino IDE.

3. Resultados e Discussões

A metodologia, as linguagens e os *frameworks* utilizados no desenvolvimento permitiram desenvolver o sistema de controle de acesso de forma ágil, além de garantir a qualidade do projeto. O sistema desenvolvido conta com quatro módulos distintos com permissões e responsabilidades específicas que facilitam sua utilização, sendo eles: **Módulo Gerente:** responsável por administrar o funcionamento do sistema e gerar relatórios de gestão; **Módulo Porteiro:** incumbido de conceder ou negar acesso ao restaurante; **Módulo Vendedor:** encarregado de realizar a recarga de tíquetes, e o **Módulo Consumidor:** que possibilita a consulta de saldos.

O sistema apresenta as seguintes funcionalidades:

- Cadastro de Refeições e configuração do horário de abertura e encerramento das mesmas, bem como os dias da semana em que acontecem.
- Cadastro de Tíquetes e definição das taxas a serem cobradas por sua compra.
- Cadastro de Consumidores e definição do seu respectivo Tíquete, bem como suas permissões no sistema as quais podem ser: Consumidor, Gerente, Vendedor ou Porteiro.
- Cadastro de Bolsa Alimentação para os Consumidores por meio da qual é possível conceder acesso livre a todas as refeições durante a data de vigência da Bolsa Alimentação.
- Venda de Tíquetes para consumidores através da realização de Recargas no sistema.
- Emissão de relatório de recargas durante um determinado período.
- Emissão de relatório de acessos por refeição ou durante um determinado período.
- Identificação do acesso dos consumidores ao Restaurante Estudantil por meio de um Cartão-Alimentação, o qual é gerado pelo próprio sistema.

Na figura 7 é apresentado, de forma simplificada, o funcionamento do sistema. Primeiramente, o consumidor deve passar seu Cartão Alimentação no leitor, o qual está conectado ao Arduino que tem a função de transmitir a identificação lida do Cartão para o sistema web. Na sequência, o sistema valida se o consumidor em questão está devidamente cadastrado no Banco de Dados.

Figura 7: Esquema simplificado do funcionamento do sistema, construído utilizando o Software CorelDRAW X4. Fonte: Autoria própria.

Caso o consumidor esteja cadastrado, o sistema verifica o Tíquete, que foi vinculado a este consumidor na hora em que ele foi cadastrado no sistema. De acordo com esse Tíquete o sistema consegue identificar, por exemplo, se o consumidor é um Funcionário, um Aluno do Curso Superior ou até mesmo um Convidado. Assim, se o consumidor tiver realizado uma recarga de Tíquetes e apresentar saldo disponível, o sistema autoriza a sua entrada e, nesse caso, é contabilizado um acesso e debitado uma unidade do saldo atual desse consumidor; se o consumidor possui uma Bolsa Alimentação e o prazo de vigência dela ainda não expirou, o sistema autoriza a entrada deste consumidor e registra seu acesso logo em seguida.

Pode-se observar na figura 8 a tela inicial através da qual o usuário deverá informar suas credenciais corretamente para entrar no sistema. Essa tela apresenta também o campo Tipo de acesso, através do qual o usuário escolhe se deseja acessar o sistema como Consumidor, para poder consultar o seu saldo e histórico de acessos, ou se deseja acessar como Usuário e, nesse caso, sistema identificará automaticamente as permissões do mesmo e concederá acesso aos seus respectivos módulos que podem ser Gerente, Vendedor ou Porteiro.

Figura 8: Tela de Login do sistema.

No caso do usuário ter escolhido acessar o sistema como Consumidor, ele será redirecionado para a página demostrada na figura 9. Nesta tela ele terá acesso a informações como o Número de Tíquetes e seus últimos 10 acessos aos Restaurante Estudantil.

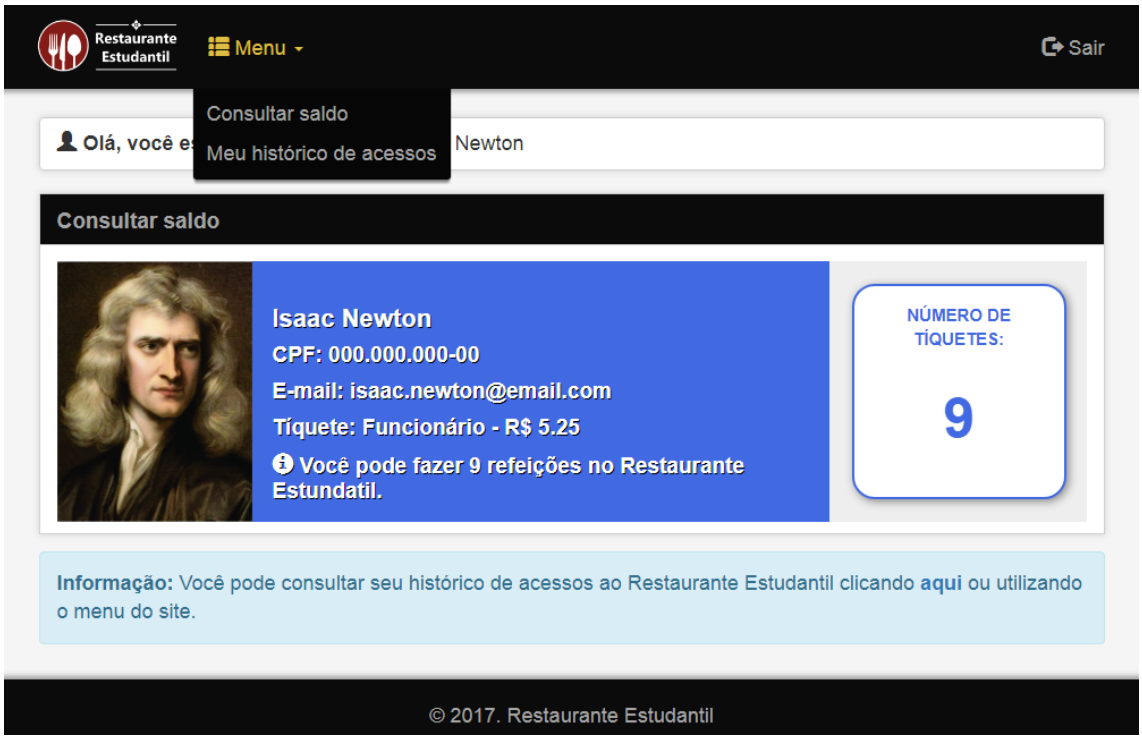

Figura 9: Tela para consulta de saldo do Consumidor – Módulo Consumidor.

Na hipótese da pessoa ter escolhido acessar o sistema como Usuário, ele será redirecionado para a página referente às suas respectivas responsabilidades como demostrado nas figuras 10, 11 e 12 que exemplificam os 3 possíveis perfis de acesso de um Usuário e algumas das ações que o mesmo pode desempenhar no sistema por meio das opções correspondentes presentes no menu do site.

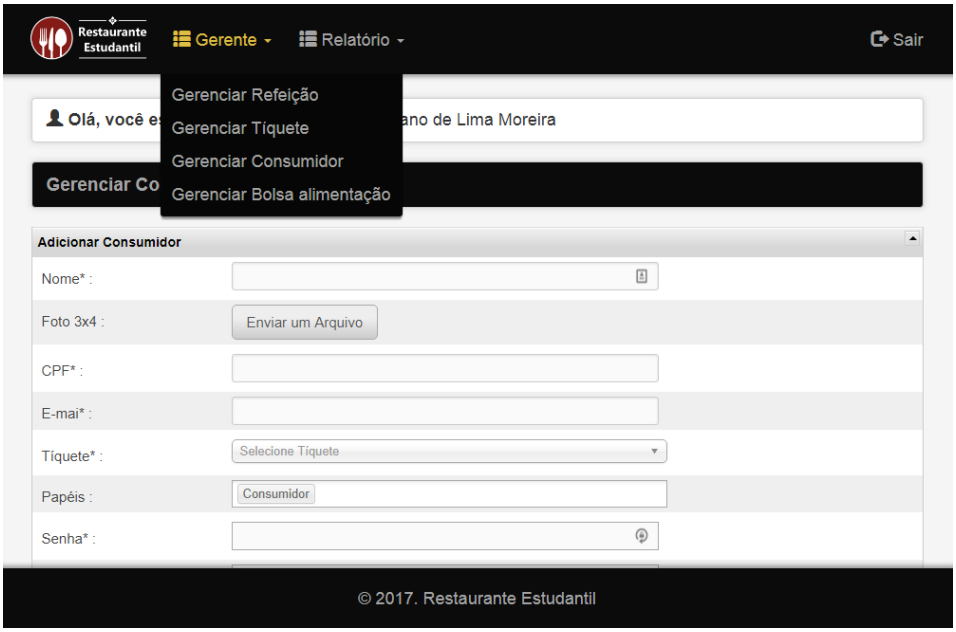

Figura 10: Tela de cadastro de Consumidor – Módulo Gerente.

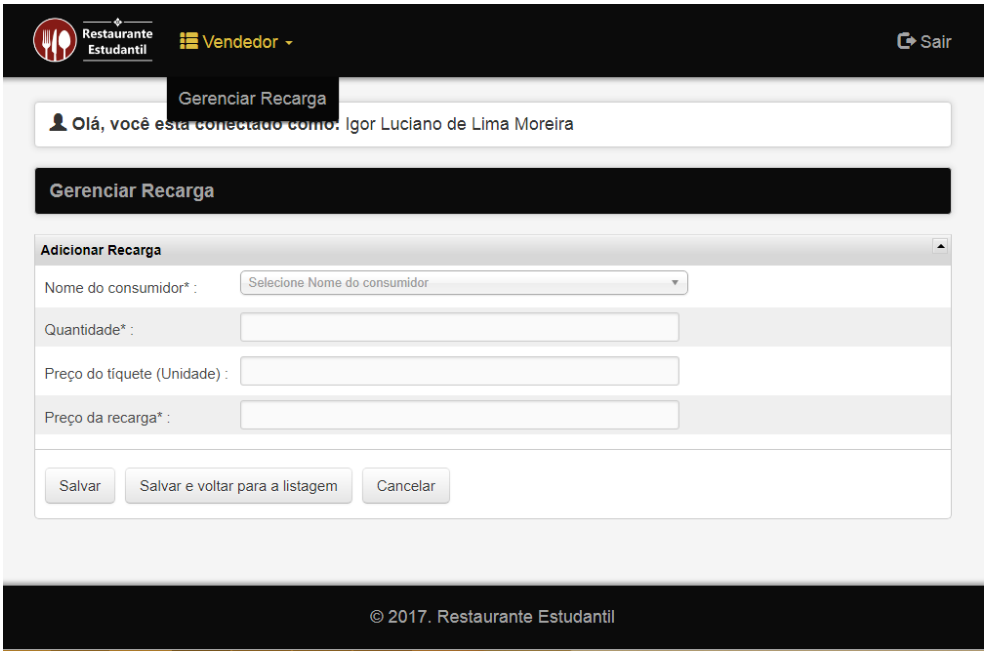

Figura 11: Tela de Realização de Recarga para os consumidores – Módulo Vendedor

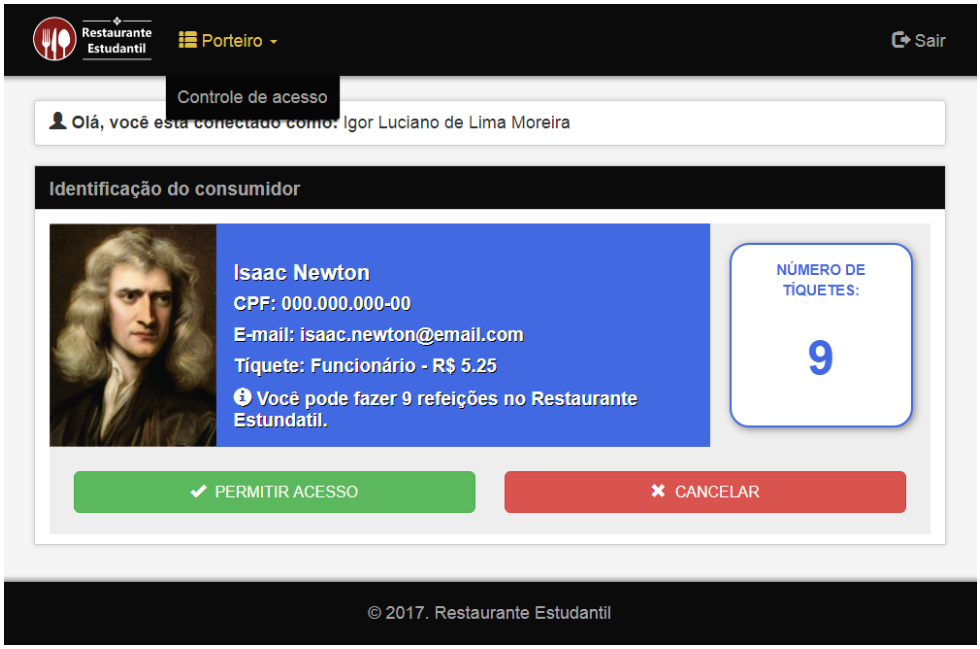

Figura 12: Tela para controle de acesso ao Restaurante Estudantil – Módulo Porteiro.

Concluído o desenvolvimento do sistema, os *stakeholdes* foram convidados a realizar a avaliação de usabilidade do sistema. Para essa avaliação foi utilizado o questionário *System Usability Scale* (SUS) criado por Brooke em 1986. De acordo com Sauro (2011) o SUS é um padrão da indústria e conta com mais de 600 publicações, ele já tendo sido utilizado para testar diversos produtos, como *hardware*, *software* e sites. O questionário é formado por 10 questões, com 5 opções de respostas na escala de *Linkert* variando de 1 – Discordo Totalmente a 5 – Concordo Totalmente. O questionário SUS pode ser encontrado no Anexo 4 desse trabalho.

A pontuação desse questionário é calculada ao somar as respostas de cada questão, sendo que, de acordo com a questão, a resposta que será contabilizada pode variar. Para as questões ímpares (1,3,5,7 e 9), deve ser subtraído da pontuação menos 1. E para as questões de pares (2,4,6,8 e 10), a pontuação na escala deve ser subtraída de 5. Uma vez determinado o valor de cada questão, basta somar todos os valores e multiplicar por 2,5. O resultado obtido deverá ser dividido pelo número de *stakeholdes* para obter o resultado global do SUS. Este resultado global está inserido numa escala de 0 a 100. Participaram da avaliação 13 *stakeholdes* que representaram cada um dos perfis de acesso aos módulos do sistema.

Calculada a média das avaliações realizadas pelos participantes no questionário SUS, o Sistema de Controle de Acesso ao Restaurante Estudantil obteve 93,46 pontos dos 100 possíveis. O valor obtido demonstra que o sistema desenvolvido possui excelente usabilidade e que atende ás necessidades dos stakeholders.

4. Conclusões

O objetivo deste trabalho foi desenvolver um sistema para otimizar o acesso ao Restaurante Estudantil por meio de um conjunto de funcionalidades que permitam gerenciar a utilização do mesmo, de forma que possibilitasse a identificação e contabilização dos acessos dos consumidores, facilitando o trabalho que atualmente é feito manualmente.

A avaliação de usabilidade demonstrou que o sistema possui boa usabilidade e que atende ás necessidades dos *stakeholders*. Atualmente o sistema encontra-se em processo de implantação no IFSULDEMNAS - Campus Inconfidentes e futuramente pode ser implantado nos demais campi.

5. Referências Bibliográficas

BROOKE, John. SUS: a retrospective. **Journal of usability studies**, v. 8, n. 2, p. 29-40, 2013.

GALHARDO, Antonio Tadeu. **Sistemas eletrônicos de controle de acesso**. 2011. 54 f. Monografia (Especialização) - Curso de Engenharia Elétrica, Universidade Sao Francisco, Campinas, 2011.

LIBARDI, Paula L.O.; BARBOSA, Vladimir. **Métodos ágeis**. 2010. 35 f. Monografia (Especialização) - Curso de Tópicos em Informática: Gerenciamento de Projetos e Qualidade, Faculdade de Tecnologia da Unicamp, Limeira - Sp, 2010.

MCROBERTS, Michael. **Arduino Básico**. São Paulo: Novatec, 2011. 456 p. Tradução de: Rafael Zanolli.

MORK, S. **Programação com Arduino**: Começando com Sketches. Porto Alegre. Bookman, 2013.

TONIN, Fabianna Stumpf; CITTOLIN, Guilherme Francescon; SOUZA, Vinicius de. **Desenvolvimento de um sistema web de controle de acesso**. 2015. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná.

SAURO, J. **Measuring Usability With The System Usability Scale (SUS)**. [S.l.], 2011. Disponível em: <http://www.measuringu.com/sus.php>. Acesso em: 11 de Junho de 2017.

Anexos

```
<!-- ANEXO 1: Controller Consumidor -->

<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class Consumidor extends MY_Controller {

    function __construct() {
        parent::__construct();
    }

    public function index() {
        //PREPARA O CRUD
        $crud = new Grocery_CRUD();
        $crud->set_table("consumidor");
        $crud->set_subject('Consumidor');

        //DESABILITA A OPÇÃO DE DELETAR
        $crud->unset_delete();

        //OPÇÃO PARA IMPRIMIR CARTÃO DE ALIMENTAÇÃO DO CONSUMIDOR
        $crud->add_action('Imprimir Cartão de Alimentação', base_url(
        'assets/img/cartao/icone_cartao.png'), 'restrito/relatorio/cartao_alimentacao');

        //CAMPOS DO FORMULÁRIO
        $crud->fields('nome', 'foto3x4', 'cpf', 'email', 'idtipo_tiquete', 'idusuario',
        'papeis', 'senha', 'confirme_senha');

        //CAMPOS OBRIGATÓRIOS
        $crud->required_fields('nome', 'foto3x4', 'cpf', 'email', 'idtipo_tiquete', 'senha',
        'confirme_senha');

        //CAMPOS DA TELA DE DETALHES
        $crud->set_read_fields('nome', 'foto3x4', 'cpf', 'email', 'idtipo_tiquete', 'papeis');

        //COLUNAS EXIBIDAS NA LISTAGEM
        $crud->columns(array('foto3x4', 'nome', 'cpf', 'idtipo_tiquete'));

        //DEFINE RELAÇÃO DE CHAVE ESTRANGEIRA COM A TABELA TIPO_TIQUETE
        $crud->set_relation('idtipo_tiquete', 'tipo_tiquete', '{nome_tiquete} - R$
        {preco_tiquete}');

        //DEFINE O CAMPO IDUSUARIO COMO CAMPO OCULTO
        $crud->change_field_type('idusuario', 'hidden');

        //ALTERA O CAMPO FOTO3X4 PARA O FORMATO DE ENVIO DE ARQUIVOS
        $crud->set_field_upload('foto3x4', 'assets/uploads/files/fotos3x4');

        //CAMPOS DA TELA ADICIONAR REGISTRO
        $crud->callback_add_field('senha', function () {
            return '<input type="password" maxlength="50" value="" name="senha">';
        });

        $crud->callback_add_field('confirme_senha', function () {
            return '<input type="password" maxlength="50" value="" name="confirme_senha">';
        });

        $crud->callback_add_field('papeis', function () {
            return '<select multiple name="papeis[]" id="field-papeis"
            class="chosen-multiple-select" size="8" style="width:459px;"">
                  <!-- marca a opção Consumidor como padrão e impede que ela seja desmarcada
                  -->
                  <option value="Consumidor" selected disabled>Consumidor</option>
                  <option value="Porteiro">Porteiro</option>
                  <option value="Vendedor">Vendedor</option>
                  <option value="Gerente">Gerente</option>
               </select>';
```

```
    });

    //CAMPOS DA TELA EDITAR REGISTRO
    if (($crud->getState() == 'edit') OR ( $crud->getState() == 'update_validation')) {

        //CARREGA O MODEL E BUSCA OS DADOS NO BANCO
        $this->load->model('consumidor_model');
        $dados_consumidor = $this->consumidor_model->Pesquisar_por_idconsumidor($crud->
        getStateInfo()->primary_key);

        //CHECA SE RETORNOU ALGUM DADO
        if ($dados_consumidor) {
            //ARMAZENA OS DADOS EM VARÁVEIS
            $this->idusuario = $dados_consumidor['idusuario'];
            $this->senha = $dados_consumidor['senha'];
            $this->papeis = $dados_consumidor['papeis'];
        }
    }

    $crud->callback_edit_field('senha', function ($value, $primary_key) {
        return '<input type="password" maxlength="50" value="' . $this->senha . '"
        name="senha">';
    });

    $crud->callback_edit_field('confirme_senha', function ($value, $primary_key) {
        return '<input type="password" maxlength="50" value="' . $this->senha . '"
        name="confirme_senha">';
    });

    $crud->callback_edit_field('papeis', function ($value, $primary_key) {

        //PREENCHE  O <SELECT> COM OS PAPÉIS DO USUÁRIO E AS MARCA COMO SELECIONADAS
        $papeis = explode(',', $this->papeis);
        $opcoes = "";

        //ADICIONA AS OPÇÕES AO <SELECT>
        foreach ($papeis as $key => $value) {

            if ($value == "Consumidor") {

                //MARCA A OPÇÃO CONSUMIDOR COMO PADRÃO E IMPEDE QUE ELA SEJA DESMARCADA
                $opcoes .= '<option value="' . $value . '" selected disabled>' . $value .
                 '</option>';
            } else {

                //MARCA AS OPÇÕES DO USUÁRIO QUE ESTÃO GRAVADAS NO BANCO
                $opcoes .= '<option value="' . $value . '" selected>' . $value .
                '</option>';
            }
        }


        //ACRESCENTA AS DEMAIS OPÇÕES QUE NÃO ESTÃO NO BANCO, DEIXANDO-AS COMO DESMARCADAS
        if (!in_array("Gerente", $papeis)) {
            $opcoes .= '<option value="Gerente">Gerente</option>';
        }

        if (!in_array("Vendedor", $papeis)) {
            $opcoes .= '<option value="Vendedor">Vendedor</option>';
        }

        if (!in_array("Porteiro", $papeis)) {
            $opcoes .= '<option value="Porteiro">Porteiro</option>';
        }

        return '<select multiple name="papeis[]" id="field-papeis"
        class="chosen-multiple-select" size="8" style="width:459px;"">' . $opcoes .
        '</select>';
    });
```

```
        //FORMATA O NOME DOS CAMPOS
        $crud->display_as('foto3x4', 'Foto 3x4');
        $crud->display_as('cpf', 'CPF');
        $crud->display_as('email', 'E-mai');
        $crud->display_as('idtipo_tiquete', 'Tíquete');
        $crud->display_as('confirme_senha', 'Confirme a senha');
        $crud->display_as('papeis', 'Papéis');

        //VALIDAÇÃO DO PREENCHIMENTO DO CAMPO SENHA E CONFIRME_SENHA
        $crud->set_rules('senha', 'Senha',
        'trim|min_length[6]|required|matches[confirme_senha]');
        $crud->set_rules('confirme_senha', 'Confirme a Senha', 'required');

        //VALIDAÇÃO DO PREENCHIMENTO DO CAMPO EMAIL
        $crud->set_rules('email', 'Email', 'valid_email|required');

        //VALIDAÇÃO DO PREENCHIMENTO DO CAMPO CPF
        $crud->unique_fields(array('cpf'));
        $crud->set_rules('cpf', 'CPF', 'callback_validar_cpf');

        //COLUNAS DA TELA DE LISTAGEM
        $crud->callback_column('foto3x4', array($this, 'callback_foto_exibir'));

        //CALLBACKS DA TELA DE DETALHES
        $crud->callback_read_field("foto3x4", array($this, 'callback_foto_exibir'));
        $crud->callback_read_field("papeis", array($this, 'callback_papeis_exibir'));

        //CALLBACKS AO INSERIR OU EDITAR UM REGISTRO
        $crud->callback_insert(array($this, 'callback_usuario_inserir'));
        $crud->callback_update(array($this, 'callback_usuario_atualizar'));

        //GERA O CRUD
        $output = $crud->render();

        $this->template->set('title', 'Gerenciar Consumidor');
        $this->template->load('template/restrito', 'restrito/consumidor/index.php', $output);
    }

    //CALLBACK PARA INSERIR UM USUÁRIO NO BANCO DE DADOS
    function callback_usuario_inserir($post_array) {
        //DADOS DO USUÁRIO PARA SEREM GRAVADOS NO BANCO DE DADOS
        $data = array(
            'login' => $post_array['cpf'],
            'senha' => md5($post_array['senha']),
            'papeis' => papeis_formatar($post_array['papeis'])
        );

        //CARREGA O MODEL E INSERE OS DADOS NO BANCO RECUPERANDO O ID DO USUARIO CADASTRADO
        $this->load->model('usuario_model');
        $idusuario = $this->usuario_model->Inserir($data);

        //CHECA SE O IDUSUARIO FOI RETORNADO E A OPERAÇÃO OCORREU CORRETAMENTE
        if ($idusuario) {
            //O CAMPO IDUSUARIO DO CONSUMIDOR RECEBE O IDUSUARIO DO USUARIO QUE FOI INSERIDO
            NO BANCO DE DADOS
            $post_array['idusuario'] = $idusuario;

            //APAGA OS CAMPOS QUE NÃO FAZEM PARTE DA TABELA CONSUMIDOR
            unset($post_array['senha']);
            unset($post_array['confirme_senha']);
            unset($post_array['papeis']);
            unset($post_array['s5563c8f3']);

            //INSERE OS DADOS DO CONSUMIDOR NO BANCO DE DADOS
            return $this->db->insert('consumidor', $post_array);
        } else {
            return false;
        }
```

```
    }

    //CALLBACK PARA ATUALIZAR UM USUÁRIO NO BANCO DE DADOS
    function callback_usuario_atualizar($post_array, $primary_key) {

        //CARREGA O MODEL E BUSCA OS DADOS DO CONSUMIDOR NO BANCO DE DADOS
        $this->load->model('consumidor_model');
        $dados_consumidor = $this->consumidor_model->Pesquisar_por_idconsumidor($primary_key);

        //CHECA SE RETORNOU ALGUM DADO
        if ($dados_consumidor) {
            $senha_nova = $post_array['senha'];
            $senha_antiga = $dados_consumidor['senha'];

            //VERIFICA SE A NOVA SENHA RETORNADA DO FORMULÁRIO É IGUAL A ANTIGA
            if ($senha_nova != $senha_antiga) {
                //CRIPTOGRAFA A NOVA SENHA SE ELA FOR DIFERENTE
                $senha = md5($senha_nova);
            } else {
                //MANTÉM A SENHA DA MESMA FORMA SE ELAS FOREM IGUAIS
                $senha = $senha_antiga;
            }
        }

        //DADOS DO USUÁRIO PARA SEREM ATUALIZADOS NO BANCO
        $data = array(
            'login' => $post_array['cpf'],
            'senha' => $senha,
            'papeis' => papeis_formatar($post_array['papeis'])
        );

        //CARREGA O MODEL E ATUALIZA OS DADOS NO BANCO
        $this->load->model('usuario_model');
        $operacao = $this->usuario_model->Atualizar($post_array['idusuario'], $data);

        //CHECA SE A OPERAÇÃO OCORREU CORRETAMENTE
        if ($operacao) {
            //APAGA OS CAMPOS QUE NÃO FAZEM PARTE DA TABELA CONSUMIDOR
            unset($post_array['senha']);
            unset($post_array['confirme_senha']);
            unset($post_array['papeis']);
            unset($post_array['s5563c8f3']);

            //ATUALIZA OS DADOS DO CONSUMIDOR NO BANCO
            return $this->db->update('consumidor', $post_array, array('idconsumidor' =>
            $primary_key));
        } else {
            return false;
        }
    }

    //CALLBACK PARA FORMATAR EXIBIÇÃO DA FOTO3X4
    function callback_foto_exibir($value, $row) {
        if (!is_null($value) && $value != "") {
            //EXIBE A FOTO DO CONSUMIDOR
            $foto3x4 = base_url('assets/uploads/files/fotos3x4/') . $value;
        } else {
            //EXIBE UMA FOTO PADRÃO SE O CONSUMIDOR NÃO POSSUIR FOTO
            $foto3x4 = base_url('assets/img/cartao/foto_padrao.png');
        }
        return "<img src='" . $foto3x4 . "' height='100px'>";
    }

    //CALLBACK PARA FORMATAR EXIBIÇÃO DOS PAPÉIS DO CONSUMIDOR
    function callback_papeis_exibir($value, $row) {
        //CARREGA O MODEL E BUSCA OS DADOS DO CONSUMIDOR NO BANCO DE DADOS
        $this->load->model('consumidor_model');
        $dados_consumidor = $this->consumidor_model->Pesquisar_por_idconsumidor($row);
```

```
        //CHECA SE RETORNOU ALGUM DADO PARA EXIBIR
        if ($dados_consumidor) {
            return $dados_consumidor['papeis'];
        }
    }

    //REGRA PARA VALIDAR O CPF
    function validar_cpf($cpf) {
        if ($cpf == FALSE) {
            $this->form_validation->set_message('validar_cpf', 'O campo CPF é obrigatório.');
            return FALSE;
        } else if (!validar_cpf($cpf)) {
            $this->form_validation->set_message('validar_cpf', 'O CPF <b>' . $cpf . '</b> é
            inválido. Por favor, informe outro CPF.');
            return false;
        }
        return TRUE;
    }

    public function consultar_saldo() {
        //CARREGA AS INFORMAÇÕES PESSOAIS DO CONSUMIDOR
        $dados_usuario = obter_informacoes_consumidor($this->session->userdata('idconsumidor'
        ));

        $this->template->set('title', 'Consultar saldo');
        $this->template->load('template/restrito', 'restrito/consumidor/consultar_saldo.php',
         $dados_usuario);
    }

    public function historico_acessos() {
        //CARREGA O MODEL E BUSCA OS HISTÓRICOS DE ACESSOS DO CONSUMIDOR NO BANCO DE DADOS
        $this->load->model('consumidor_model');
        $output['historico_acessos'] = $this->consumidor_model->Pesquisar_historico_acessos(
        $this->session->userdata('idconsumidor'));

        $this->template->set('title', 'Histórico de acessos');
        $this->template->load('template/restrito',
        'restrito/consumidor/historico_acessos.php', $output);
    }
}
```

```
<!-- ANEXO 2: Model Consumidor -->

<?php

class Consumidor_model extends CI_Model {

    function __construct() {
        parent::__construct();
        $this->table = 'consumidor';
    }
    //CONSULTA PARA PESQUISAR OS DADOS DE UM CONSUMIDOR
    function Pesquisar_por_idconsumidor($idconsumidor) {
        $this->db->join('usuario', 'usuario.idusuario = consumidor.idusuario');
        $this->db->join('tipo_tiquete', 'tipo_tiquete.idtipo_tiquete =
        consumidor.idtipo_tiquete');
        $this->db->where('idconsumidor', $idconsumidor);
        $query = $this->db->get($this->table);

        if ($query->num_rows() == 1) {
            return $query->row_array();
        } else {
            return null;
        }
    }

    //CONSULTA PARA PESQUISAR UM CONSUMIDOR ATRAVÉS DO CPF
    function Pesquisar_por_cpf($cpf) {
        $this->db->join('usuario', 'usuario.idusuario = consumidor.idusuario');
        $this->db->join('tipo_tiquete', 'tipo_tiquete.idtipo_tiquete =
        consumidor.idtipo_tiquete');
        $this->db->where('cpf', $cpf);
        $query = $this->db->get($this->table);

        if ($query->num_rows() == 1) {
            return $query->row_array();
        } else {
            return null;
        }
    }

    //CONSULTA PARA PESQUISAR O HISTÓRICO DE ACESSO DE UM CONSUMIDOR
    function Pesquisar_historico_acessos($idconsumidor) {
        $query = $this->db->query(
                " SELECT "
                . " t.numero_tiquete, "
                . " date_format(t.data_utilizacao, '%d/%m/%Y às %H:%i:%s ') AS
                data_utilizacao, "
                . " tr.nome_refeicao, "
                . " (SELECT co.nome FROM consumidor co WHERE co.idusuario = rf.idusuario) as
                porteiro "
                . " FROM tiquete t"
                . " INNER JOIN recarga r ON r.idrecarga= t.idrecarga "
                . " INNER JOIN refeicao rf ON rf.idrefeicao = t.idrefeicao "
                . " INNER JOIN tipo_refeicao tr ON tr.idtipo_refeicao = rf.idtipo_refeicao "
                . " INNER JOIN consumidor c ON c.idconsumidor = r.idconsumidor "
                . " WHERE t.idrefeicao is not null "
                . " AND c.idconsumidor = " . $idconsumidor);


        if ($query->num_rows() > 0) {
            return $query->result_array();
        } else {
            return null;
        }
    }

}
```

```
<!-- ANEXO 3: View Consumidor -->

<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="utf-8">
        <!-- CÓDIGO JAVASCRIPT PARA COLOCAR MÁSCARAS NOS CAMPOS -->
        <!-- DEVE SER EXECUTADO ANTES DO GROCERYCRUD PARA EVITAR CONFLITO COM OS ARQUIVO JS
        -->
        <script type="text/javascript">
            jQuery(function ($) {
                //MÁSCARA PARA OS CAMPOS CPF
                $("#field-cpf").mask("999.999.999-99");
            });
        </script>

        <!-- Grocery_CRUD Core CSS -->
        <?php foreach ($css_files as $file): ?>
            <link type="text/css" rel="stylesheet" href="<?php echo $file; ?>" />
        <?php endforeach; ?>

        <!-- Grocery_CRUD Core JavaScript -->
        <?php foreach ($js_files as $file): ?>
            <script src="<?php echo $file; ?>"></script>
        <?php endforeach; ?>
    </head>
    <body>
        <!-- CABEÇALHO -->
        <div class="list-group">
            <div class="list-group-item painel destaque">
                <b>Gerenciar Consumidor</b>
            </div>
        </div>

        <!-- INFORMAÇÃO -->
        <div class="alert alert-info"><a class="close" data-dismiss="alert" href="#">×</a>
            <p><b>Informação: </b></p>
            <p>* Pode-se atribuir 4 (quatro) possíveis papéis para uma pessoa, cada papel
            possui permissões específicas no sistema, sendo elas:</p>
            <div style="margin:10px 0 0 10px">
                <p>- Consummidor: Pode apenas consultar o seu saldo.</p>
                <p>- Gerente: Pode administrar o sistema e gerar relatórios.</p>
                <p>- Porteiro: Pode realizar o controle de acesso.</p>
                <p>- Vendedor: Pode realizar recarga de tíquetes.</p>
            </div>
        </div>

        <!-- CRUD -->
        <p><?php echo $output; ?></p>

    </body>
</html>
```

ANEXO 4: Avaliação de Usabilidade

Sistema para controle de acesso ao Restaurante Estudantil

Nome:

01. Eu acho que gostaria de usar esse sistema com frequência.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

02. Eu acho o sistema desnecessariamente complexo.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

03. Eu achei o sistema fácil de usar.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

04. Eu acho que precisaria de ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

05. Eu acho que as várias funções do sistema estão muito bem integradas.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

06. Eu acho que o sistema apresenta muita inconsistência.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

07. Eu imagino que as pessoas aprenderão como usar esse sistema rapidamente.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

08. Achei o sistema muito difícil de usar

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

09. Eu me senti confiante ao usar o sistema.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

10. Eu precisei aprender várias coisas novas antes de conseguir usar o sistema.

Discordo totalmente ☐ ☐ ☐ ☐ ☐ Concordo totalmente
1 2 3 4 5

Comentários: