UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTAÇÃO E SISTEMAS

JOSÉ CASSIMIRO TOLEDO JÚNIOR

SISTEMA DE CONTROLE E GESTÃO DE PEDIDOS ONLINE PARA O RAMO ALIMENTÍCIO DE JOÃO MONLEVADE

João Monlevade
2017

JOSÉ CASSIMIRO TOLEDO JÚNIOR

SISTEMA DE CONTROLE E GESTÃO DE PEDIDOS ONLINE PARA O RAMO ALIMENTÍCIO DE JOÃO MONLEVADE

Monografia apresentada ao curso Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina “Trabalho de Conclusão de Curso II”.

Orientador: Mateus Ferreira Satler

João Monlevade

2017

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E APLICADAS
COLEGIADO DO CURSO DE SISTEMAS DE INFORMAÇÃO

ANEXO II - Formulário de Autorização de Lançamento de Nota

FORMULÁRIO DE LANÇAMENTO DE NOTA

Aluno: JOSÉ CASSIMIRO TOLEDO JÚNIOR **Matricula: 12.1.8271**

Disciplina: CSI499 – Trabalho de Conclusão de Curso II

Orientador: Prof. Dr. Mateus Ferreira Satler

Primeiro Convidado: Prof. Dr. Fernando Bernardes de Oliveira

Segundo Convidado: Prof. Dr. Rafael Frederico Alexandre

Nota[1]: 9.5 (NOVE PONTOS E MEIO)

Data: 05/09/2017

Avaliação do Orientador

Assinatura do Orientador

Assinatura do Primeiro Convidado

Assinatura do Segundo Convidado

*Obs.: Deve ser entregue pelo **professor orientador** até **dois dias úteis** antes do fim do prazo para lançamento de notas pelo Calendário Acadêmico na secretaria do COSI.*

1 Para TCC-I, a nota final do aluno será calculada pela média entre a nota do orientador e as notas dadas pela banca examinadora.

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E APLICADAS
COLEGIADO DO CURSO DE SISTEMAS DE INFORMAÇÃO

ANEXO III – Termo de Responsabilidade

TERMO DE RESPONSABILIDADE

Eu, José Casimiro Toledo Júnior, declaro que o texto do trabalho de conclusão de curso intitulado "Sistema de controle e gestão de pedidos online para o ramo alimentício de João Monlevade" é de minha inteira responsabilidade e que não há utilização de texto, material fotográfico, código fonte de programa ou qualquer outro material pertencente a terceiros sem as devidas referências ou consentimento dos respectivos autores.

João Monlevade, 22 de setembro de 2017

José Casimiro Toledo Júnior
Assinatura do aluno

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E APLICADAS
COLEGIADO DO CURSO DE SISTEMAS DE INFORMAÇÃO

ANEXO IV - Ata de Defesa

ATA DE DEFESA

Aos 05 dias do mês de Setembro de 2017, às 17 horas e 30 minutos, na sala H102 do Instituto de Ciências Exatas e Aplicadas, foi realizada a defesa de Monografia pelo aluno **JOSÉ CASSIMIRO TOLEDO JÚNIOR,** sendo a Comissão Examinadora constituída pelos professores: Prof. Dr. Mateus Ferreira Satler, Prof. Dr. Fernando Bernardes de Oliveira e Prof. Dr. Rafael Frederico Alexandre.

O candidato apresentou a monografia intitulada: "*SISTEMA DE CONTROLE E GESTÃO DE PEDIDOS ONLINE PARA O RAMO ALIMENTÍCIO DE JOÃO MONLEVADE*". A comissão examinadora deliberou, por unanimidade, pela aprovação do candidato, com nota 9.5 (NOVE PONTOS E MEIO), concedendo-lhe o prazo de 15 dias para incorporação das alterações sugeridas ao texto final.

Na forma regulamentar, foi lavrada a presente ata que é assinada pelos membros da Comissão Examinadora e pelo graduando.

João Monlevade, 05 de Setembro de 2017.

Prof. Dr. Mateus Ferreira Satler
Professor Orientador/Presidente

Prof. Dr. Fernando Bernardes de Oliveira
Professor Convidado

Prof. Dr. Rafael Frederico Alexandre
Professor Convidado

José Cassimiro Toledo Júnior
Graduando

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E APLICADAS
COLEGIADO DO CURSO DE SISTEMAS DE INFORMAÇÃO

ANEXO V – Folha de Aprovação

Curso de Sistemas de Informação

FOLHA DE APROVAÇÃO DA BANCA EXAMINADORA

SISTEMA DE CONTROLE E GESTÃO DE PEDIDOS ONLINE PARA O RAMO ALIMENTÍCIO DE JOÃO MONLEVADE

José Cassimiro Toledo Júnior

Monografia apresentada ao Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto como requisito parcial da disciplina CSI499 – Trabalho de Conclusão de Curso II do curso de Bacharelado em Sistemas de Informação e aprovada pela Banca Examinadora abaixo assinada:

Prof. Dr. Mateus Ferreira Satler
DECSI – UFOP

Prof. Dr. Fernando Bernardes de Oliveira
DECSI – UFOP

Prof. Dr. Rafael Frederico Alexandre
DECSI – UFOP

João Monlevade, 05 de Setembro de 2017

RESUMO

Este trabalho visa apresentar um aplicativo que apoie as pessoas a realizar pedidos de online no ramo alimentício no seu dia-a-dia e um sistema baseado na web que apoie os gerentes de restaurantes e atendentes a receber esses pedidos, bem como o administrador do sistema e os franqueados a controlar os restaurantes e gerentes relacionados. Foi visto a necessidade de aumentar a oferta de aplicativos nesta área de atuação no mercado em cidades de pequeno porte. Para isso, foram utilizadas tecnologias e ferramentas de sistemas de informação e uma breve análise das aplicações já existentes no mercado. Além disso, foram alcançados os resultados esperados com base nos objetivos propostos, bem como foram listados alguns pontos de melhorias futuras para que o sistema esteja sempre em evolução e acompanhe a dinamicidade do mercado digital em que está inserido.

Palavras-chave: delivery, alimentos, aplicativos.

ABSTRACT

This project aims to present an application that will support people in ordering food online on a day-to-day basis and a web-based system that supports restaurant managers and attendants receiving these requests, as well as the system administrator and Franchisees to control related restaurants and managers. It was seen the need to increase the number of applications in this market area in small cities. To this end, was used information systems technologies, tools, and a brief analysis of the existing applications in the market. In addition, the expected results were achieved based on the proposed objectives, as well as some points of future improvements were listed so that the system is always evolving and accompany the dynamicity of the digital market in which it is inserted.

Keywords: delivery, food, mobile apps.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 Introdução

Por muito tempo os restaurantes que trabalhavam com delivery de comida recebiam os pedidos através de ligações telefônicas. Com o aumento do uso de smartphones e a oferta de aplicativos móveis do ramo alimentício, as pessoas começaram a utilizar esses aplicativos para solicitar a entrega em sua residência, visto que os aplicativos se mostraram uma maneira mais eficiente de realizar pedidos, tanto para o cliente quanto para o restaurante.

Todavia, os aplicativos de delivery são implantados, na maioria das vezes, em grandes centros urbanos, o que não é o caso de João Monlevade, que atualmente é atendido apenas pelo aplicativo Aiqfome (AIQFOME, 2017).

Para promover uma maior gama de opções para os moradores da cidade, este trabalho propõe o desenvolvimento de um aplicativo móvel do ramo alimentício, que apoie os clientes de restaurantes nas solicitações de refeições e auxilie os gerentes de restaurantes a atendê-los.

2 Objetivos

Este projeto tem como objetivo desenvolver um aplicativo móvel para solicitação de refeições. Com esse aplicativo o usuário será capaz de escolher um restaurante próximo e produtos de sua preferência. Para os gerentes, será desenvolvido um sistema de apoio baseado na web, no qual ele e seus atendentes poderão receber pedidos e direcionar as solicitações.

O objetivo inicial é desenvolver uma solução que possa oferecer suporte na cidade de João Monlevade, entretanto, o intuito é de que o sistema possibilite uma expansão fácil para outras regiões. Para isso, seria viável utilizar o modelo de franquias, ou seja, o sistema web que apoiará os gerentes e atendentes e também possuirá uma área dedicada a apoiar os franqueados, para que possam controlar seus respectivos restaurantes, de modo a atender os objetivos específicos deste trabalho, que são gerar um sistema web, além de gerar um aplicativo de celular para que seja possível o acesso dos usuários de aplicativos para pedir comida em cidades menores.

Neste sentido, será realizado um estudo sobre as áreas de conhecimento que permeiam este projeto, como programação para sistemas baseados na web, tecnologias para desenvolvimento de aplicativos móveis, webservices e comércio eletrônico.

3 Estado da Arte

Esta seção visa abordar todo o referencial bibliográfico necessário no desenvolvimento do trabalho. As subseções discutem sobre desenvolvimento web, móvel, webservices e e-commerce.

O rápido crescimento das aplicações Web, tanto em seu escopo quanto na extensão de seu uso, tem afetado todos os aspectos de nossas vidas (GINIGE; MURUGESAN, 2001). Segundo Alfa Treinamentos (2017) as atividades relacionadas ao desenvolvimento web consistem em planejar, construir, testar e dar manutenção em sites e aplicações de internet. Esse trabalho exige uma visão ampla do mercado, dos processos de gestão de empresas, de marketing e outras competências imprescindíveis.

Neste trabalho será necessário a utilização das tecnologias de front-end e back-end, visto que são pré-requisitos para desenvolvimento de aplicações Web e serão melhor explicadas nos itens subsequentes.

3.1 Programação web – Back-end

Como o nome sugere, o back-end trabalha na parte interior da aplicação, isto é, ele é o responsável, em termos gerais, pela implementação da regra de negócio e lógica do sistema. O front-end é responsável pela interface e trabalha com a parte da aplicação que interage diretamente com o usuário. Por isso, é importante que esse desenvolvedor também se preocupe com a experiência do usuário (DANIEL, 2017). Em termos de linguagens, o front-end abrange HTML (linguagem de marcação), CSS (linguagem de estilo) e JavaScript (linguagem de script/programação).

Se tratando de back-end são inúmeras as linguagens, tecnologias e frameworks existentes para se desenvolver um sistema web, seja um site institucional, uma área administrativa ou uma loja virtual. A Figura 1 apresenta um breve resumo das linguagens de desenvolvimento mais utilizadas no mundo e mostra as dez principais:

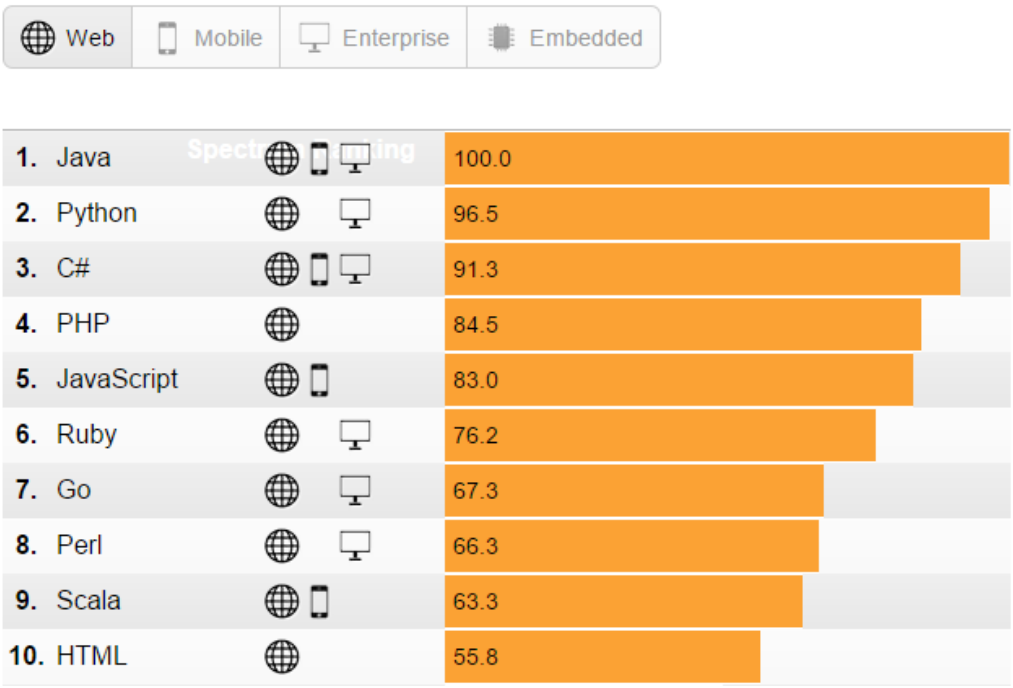

Figura 1: Ranking das principais linguagens web.
(Fonte: IEEE - https://goo.gl/rYM8Tj)

Pode-se notar pela Figura 1 que, dentre as de linguagens mais populares, Java está em primeiro lugar, seguida de Python, C#, PHP e etc. Para os intuitos específicos deste trabalho, optou-se por apresentar nas sessões subsequentes um breve resumo de três dessas linguagens, a saber: Java, Python e PHP.

3.1.1 Java

Em 1996 foi lançada a primeira versão da linguagem Java, desenvolvida por uma equipe da Sun desde 1991. É uma linguagem que permite criar aplicações tanto para desktop quanto para web. Algumas de suas características são: Java é compilada, portável, eficiente, orientada a objetos, segura, permite concorrência e suporta desenvolvimento de sistemas distribuídos.

No final dos anos 90 foi lançada a versão para desenvolvimento web: a Java Enterprise Edition (J2EE), que já sofreu várias alterações em seu código fonte, para se adequar aos requisitos atuais, chegando em 2013 à sua versão 7 (Java EE).

Constituindo o conjunto de APIs disponibilizadas pelo Java, estão:

- JavaServer Pages (JSP), Java Servlets, Java Server Faces (JSF) – sendo utilizadas na web.
- Enterprise Javabeans Components (EJB) e Java Persistence API (JPA) – clusters, acesso remoto a objetos e etc.

- Java API for XML Web Services (JAX-WS), Java API for XML Binding (JAX-B) – usadas para trabalhos com XML e webservices.
- Java Autenthication and Authorization Service (JAAS) – API padrão do Java para segurança.
- Java Transaction API (JTA) – controle de transação no contêiner.
- Java Message Service (JMS) – troca de mensagens assíncronas.
- Java Naming and Directory Interface (JNDI) – espaço de nomes e objetos.
- Java Management Extensions (JMX) – administração da sua aplicação e estatísticas sobre a mesma.

3.1.2 Python

Criada por Guido van Rossum em 1989, Python (VENNERS, 2017) é uma linguagem tanto compilada quanto interpretada, possui tipagem dinâmica e é multiparadigma, ou seja, pode ser procedural, funcional e orientada à objetos. Além disso, é de altíssimo nível (VHLL), o que é ótimo para seu aprendizado e desenvolvimento de sistemas, mas não tão eficiente no que diz respeito a execução, que pode ser lenta se comparada à linguagem C, por exemplo. Atualmente estão sendo mantidas as versões 2.7 e 3, devido ao fato de que os desenvolvedores ainda estão migrando para a última, portando códigos e bibliotecas.

Para implementar aplicações para web utilizando Python é necessário usar algum framework. Um dos mais usados é o Django (FORCIER; BISSEX; CHUN, 2009), porém existem outros como o Flask (GRINBERG, 2014) e o Pyramid (PYRAMID, 2017). O primeiro foi criado em 2003 por Jacob Kaplan-Moss, Adrian Holovaty e Simon Willison, com o intuito de ser usado como um CMS (sistema de gerenciamento de conteúdo) e é, hoje, um framework de alto nível para desenvolvimento web. CMS são sistemas utilizados para publicar, editar, excluir e criar conteúdo em sites, blogs, dentre outros, de maneira que o desenvolvedor não precise escrever linhas de código. São exemplos o Wordpress e Google Sites (CAWLEY, 2017).

As vantagens do uso da linguagem são sua simplicidade, a quantidade de soluções existentes na comunidade e a facilidade de programar uma nova solução usando Python. Algumas de suas características:

- Mapeamento objeto-relacional: os modelos são definidos em Python e possuem uma API que facilita a manipulação dos dados. Apesar disso é possível escrever comandos SQL caso seja necessário.
- Interface de administração: gera facilmente interfaces para inserção e modificação de dados (CRUD, Scaffolding).

- Urls fáceis: usando-se o recurso de reescrita de URLs facilita o acesso às aplicações.
- Sistema de templates: possui uma linguagem de template para separar a lógica de programação do design.
- Sistema de cache: aumento de performance pois as páginas são geradas uma vez e acessadas do cache.
- Internacionalização: possui suporte a desenvolver aplicações de forma a facilitar o processo de internacionalização.

3.1.3 PHP

Seu primeiro release foi lançado em 1995. Desenvolvida especialmente para a implementação de sistemas web, “PHP: Hypertext Preprocessor” (GOLDSTEIN, 2015), mais conhecida como PHP, é uma linguagem interpretada que possui certa semelhança de sintaxe com C e Java.

Uma pesquisa da Netcraft (empresa inglesa que presta serviços no ramo da internet, tais como: serviços de segurança, anti-fraude, anti-phishing e outros) de maio de 1998, indicou que cerca de 60 mil domínios possuíam códigos PHP, o que representava apenas 1% de todos os domínios daquela época. Segundo a documentação da linguagem (PHP, 2017), mesmo sendo um número estimado, pode-se dizer que o PHP está agora instalado em dezenas, ou mesmo talvez centenas de milhões de domínios em todo mundo.

Para se trabalhar com esta linguagem não é necessário, mas sim conveniente, utilizar frameworks, visto que os mesmos tratam questões de segurança, integração com banco de dados, entre outras questões de forma facilitada ou, em alguns casos, de forma transparente ao desenvolvedor. Além disso, deixa o código mais legível e estruturado, melhorando sua manutenção e atualização. Alguns exemplos são: Symfony, CakePHP e Laravel, como descrito por Avoyan (2017).

O Symfony, por exemplo, está em sua segunda versão e foi lançado em 2005. Uma das maiores preocupações dos autores François e Fabien foi com a documentação do framework, que, por sinal ficou bem completa. Seu desenvolvimento foi baseado em arquiteturas MVC (Model-View-Controller) e ORM (Object-Relational Mapping). Algumas de suas características são:

- Fácil de instalar e configurar em várias plataformas e é garantido seu funcionamento em padrões *nix e Windows.
- Simples de usar e flexível o bastante para se adaptar à casos complexos.

- Baseado na premissa de convenção sobre configuração - o desenvolvedor precisa configurar apenas o não convencional.
- Compatível com a maioria das melhores práticas e padrões de web design.
- Adaptável a políticas e arquiteturas de tecnologia de informação existentes e estável o suficiente para projetos de longo prazo.
- Código legível, com comentários do phpDocumentor, para fácil manutenção.
- Fácil de estender, permitindo a integração com outras bibliotecas.

Exemplos de projetos desenvolvidos com Symfony completo ou alguns de seus componentes, segundo sua própria documentação: Laravel – que também é um framework, Joomla, Composer e Magento, de acordo com Symfony (2017)

Outro framework muito utilizado em PHP é o Cakephp (CAKE SOFTWARE FOUNDATION, 2017). Atualmente está na versão 3 e é baseado no padrão MVC. Uma de suas principais características é a utilização de convenções sobre a base de dados, as quais exigem alguns padrões em nomeação e tabelas, relacionamentos e campos, mas permitem que o framework se conecte ao banco de dados de maneira transparente ao desenvolvedor.

Sua documentação é muito extensa, tendo praticamente todo suporte no próprio site da ferramenta, com treinamentos, how-tos, resolução de bugs, etc. A comunidade de desenvolvedores também é ampla, o que permite encontrar soluções no Github, Stack Overflow, proporcionando uma motivação a mais para quem deseja usar o Cakephp em seus projetos.

Grandes companhias fazem uso do Cakephp em suas aplicações, entre elas estão: BMW, Billabong e Hyundai.

3.2 Programação web – Front-end

Para desenvolver a parte front-end do sistema web, é necessário fazer uso de algumas tecnologias, dentre elas o HTML, o CSS e o JavaScript (TABLELESS, 2017). Da mesma forma que existem frameworks para back-end, existem frameworks auxiliadores ao front-end, os quais são: Bootstrap, AngularJS, entre outros (TABLELESS, 2017).

3.2.1 Bootstrap

Foi lançado em 1996 e sofreu várias atualizações, chegando à sua quarta versão no ano de 2017, mas tendo a terceira como estável e a mais usada. O bootstrap traz ferramentas de apoio em HTML, CSS e em JavaScript. Assim como o HTML possui tags que definem a estrutura de uma página web, o bootstrap possui classes que definem o estilo dessas tags, como propriedades de cores, formas, posicionamento, dentre outras. Um exemplo para

JavaScript é o “navbar collapse”, uma classe do bootstrap que tem como objetivo esconder os botões de uma barra de navegação que não caberiam em uma tela de smartphone, como demonstrado nas Figuras 2 e 3 abaixo:

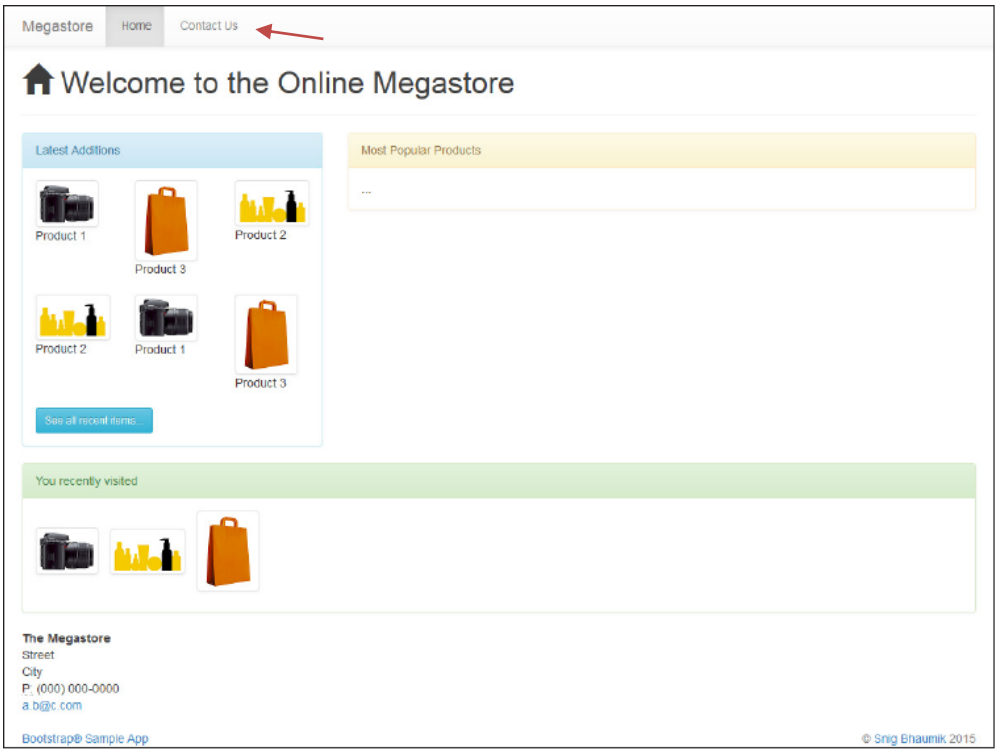

Figura 2: Exemplo de barra de navegação em desktop.

Fonte (BHAUMIK 2015)

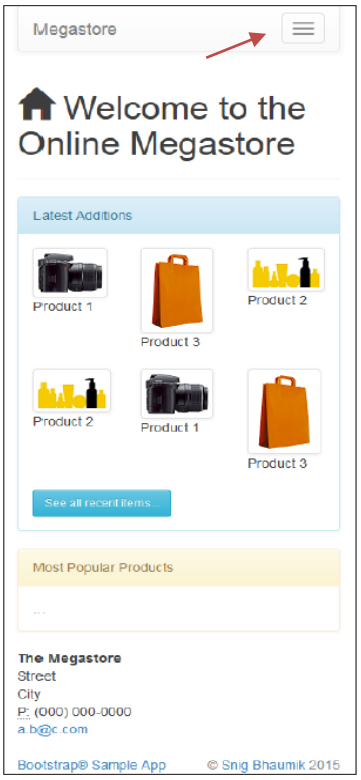

Figura 3: Exemplo de barra de navegação em smartphone.
Fonte (BHAUMIK 2015)

Dentro deste contexto, é importante mencionar umas das questões que levantam muitas discussões entre os web designers nos dias de hoje: a responsividade. Com a quantidade de dispositivos móveis existentes e aumentando, é necessário ter uma preocupação a mais na hora de criar um site, o qual deve se comportar (pensando nisso como o tamanho dos itens e disposição dos mesmos na página) de maneiras diferentes em smartphones, tablets e notebooks, ou seja, deve ser responsível. Isso pode ser resolvido com o bootstrap, pois a responsividade foi embutida no framework, no qual o desenvolvedor precisa apenas saber como utilizar suas classes da forma correta, criando o conceito de Responsive Web Design, no qual a programação é feita de forma que os elementos que compõem a interface se adaptam automaticamente à largura de tela do dispositivo no qual ele está sendo visualizado.

Não obstante, antes que isso fosse possível através da utilização do bootstrap, já existiam outros padrões de desenvolvimento que ainda são utilizados, dois para ser mais preciso, Progressive Enhancement e Graceful Degradation, ambos visando adaptar suas páginas a todos os tamanhos de telas (BHAUMIK 2015). No primeiro, o desenvolvedor cria seu site de maneira a se encaixar nas telas de smartphones e, posteriormente adapta seus componentes aos tablets e pcs, seguindo esta ordem, conhecida como abordagem bottom-up. De forma contrária, o Graceful Degradation inicia pela maior tela e depois as menores,

com a abordagem top-down. Como pôde ser observado, não existe um padrão melhor ou pior, apenas o desenvolvedor deve escolher qual abordagem irá utilizar.

3.2.2 AngularJS

Houve um crescimento nos últimos anos do uso da ferramenta JavaScript para o desenvolvimento front-end. Segundo Valeri (2015), até outubro de 2014 estavam registrados cerca de cem mil pacotes da linguagem no gerenciador de pacotes do NodeJS. Com isso, alguns desenvolvedores, como Misko Hevery, notaram a necessidade de um framework que pudesse tornar o uso da ferramenta mais fácil, além de deixar o código mais limpo, ou seja, menor e mais legível.

Hevery, que era engenheiro de testes na Google, desenvolveu o AngularJS dentro da empresa e o lançou em 2012. O sucesso do framework foi tanto que a Google relacionou uma equipe dedicada a manter e desenvolver novas funcionalidades ao projeto.

AngularJS é baseado em um conceito chamado “Two-way data binding” que permite ao desenvolvedor vincular HTML e CSS à estados de variáveis do JavaScript. Tendo isso, não é mais necessário fazer acesso direto ao DOM (Document Object Model), no qual se utilizaria de um código imperativo para realizar ações. Um exemplo simples e ilustrativo para isso é criar uma variável, com as diretivas do Angular, como atributo de uma tag HTML, no qual à medida que essa tag for atualizada, a variável também será e vice-versa. Isto faz com que, muitas vezes, o código JavaScript possa ficar muito menor, como mostra a Figura 4:

```
<input type="text" ng-model="user" placeholder="Your Name">
<h3>Hello, {{user}}!</h3>
```

Figura 4: Exemplo de código usando conceito de two-way data binding.
Fonte (KARPOV 2015)

As diretivas são poderosas auxiliadoras no desenvolvimento e o Angular possui várias built-in, como a “ng-model” vista na Figura 4. Além de diminuir a quantidade de JavaScript, as diretivas permitem que o código seja reutilizadoem outras partes do projeto.

3.3 Tecnologias para desenvolvimento móvel

Com o mercado móvel em alta, são inúmeras as opções de ferramentas e frameworks que auxiliam na implementação de soluções da área. Esta seção visa mostrar as principais plataformas atuais (Android, Windows Phone e iOS) e as tecnologias usadas na criação de aplicativos nativos. Além disso, serão observados alguns frameworks para desenvolvimento híbrido-nativo como o Ionic (IONIC FRAMEWORK, 2017).

3.3.1 Windows Phone

Chegou a versão 10 no ano de 2015. Versão que foi desenvolvida com base no kernel do Windows 8 que roda em PCs e tablets, segundo Timothy (2014). Algumas tecnologias que apoiam na criação de aplicações são C#, HTML, XAML, .NET, Visual Studio e Xamarin.

Com o lançamento do Windows Phone 7, em 2010, a Microsoft mudou totalmente o conceito de desenvolvimento de aplicativos, no qual o projeto feito para dispositivos móveis também funcionaria em Desktops. O conceito está sendo usado até hoje, porém ainda faltam muitas aplicações para esta plataforma, se comparado a quantidade que existe para Android e iOS. Com isso, a Microsoft vem investindo bastante em capacitação, criando cursos online gratuitos e facilitando a entrada de desenvolvedores no mercado.

Alguns exemplos de aplicações rodando nesta plataforma são Skype, Waze e Whatsapp. A Figura 5 mostra a tela do último aplicativo mencionado:

Figura 5: Ligação realizada através do whatsapp no windows phone 8.
Fonte (Techtudo.com.br)

3.3.2 iOS

Mesmo com a queda no mercado de celulares no Brasil, segundo SERRANO (2017), as vendas de iPhone cresceram 41%, totalizando quase 3 milhões de unidades vendidas. Logo, é possível enxergar as oportunidades que se abrem nesta área, com o desenvolvimento de apps para iOS.

Em meados de 2014 a Apple anunciou uma nova linguagem, projetada especialmente para sua plataforma e de código aberto. Esta foi uma aposta da empresa para tentar se livrar da dependência de tecnologias como Objective-C, que até então, era utilizada na criação de seus aplicativos. A Swift foi bem aceita pela crítica e vem se tornando

fundamental aos desenvolvedores novos e antigos, apesar de não ter sido possível, ainda, a total autonomia em vista à Objective-C. Alguns apps feitos usando Swift estão relacionados na Figura 6:

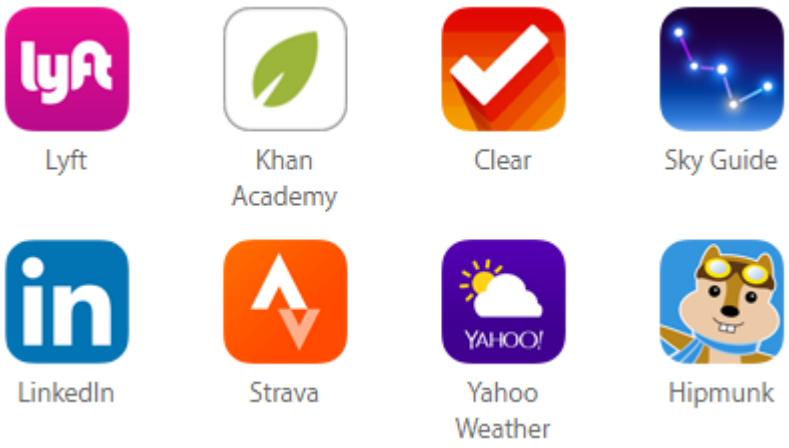

Figura 6: Apps projetados utilizando Swift.
Fonte (Apple.com)

A empresa também ressalta a melhoria no desempenho de tarefas com a nova linguagem. Em seu próprio site exemplificam com um algoritmo de busca em profundidade, o qual, comparado com Objective-C e Python se saiu melhor, como mostra a Figura 7:

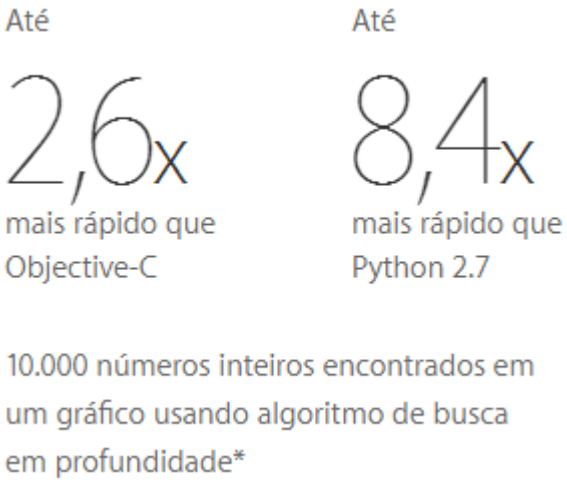

Figura 7: Exemplo de algoritmo de busca em Swift.
Fonte (Apple.com)

De acordo com a Figura 7, é possível notar que realmente a tecnologia Swift veio para melhorar a performance da Apple, fazendo com que os desenvolvedores busquem a independência em relação ao Objective-C, para desta forma tornar o processamento mais rápido ainda.

3.3.3 Android

A plataforma mais popular, com a maior quantidade de aplicativos disponíveis em sua loja e rodando em mais de 1 bilhão de dispositivos foi lançada em 2003 por Andy Rubin, o qual sempre prezou por desenvolver uma tecnologia que fosse open source (código aberto), para que todos pudessem adequá-la a seus interesses. Além disso, o Android foi baseado no kernel do Linux (que também é open source).

A partir disso, foram surgindo novas versões e atualizações para o Android, com a filosofia de sempre usar nomes de doces, chegando no ano de 2017 à versão 8.0 ou Android Oreo. Na Figura 8 pode-se observar as versões mais usadas até a versão 6.0, onde, no topo está a Kitkat 4.4 sendo usada em 36.6% dos aparelhos:

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	3.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.9%
4.1.x	Jelly Bean	16	10.0%
4.2.x		17	13.0%
4.3		18	3.9%
4.4	KitKat	19	36.6%
5.0	Lollipop	21	16.3%
5.1		22	13.2%
6.0	Marshmallow	23	0.5%

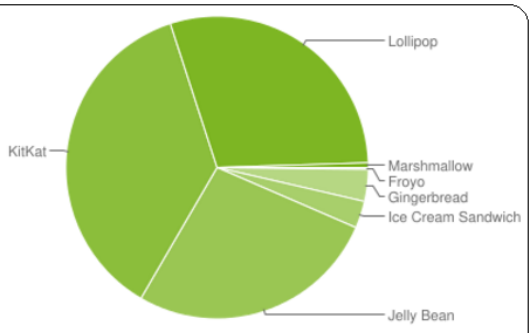

Figura 8: Versões do Android.
Fonte (Android.com)

Para o desenvolvimento de aplicações Android deve-se utilizar alguma IDE para auxílio como o Eclipse, que possui um plugin conhecido como ADT (Android Developer Tools) (THE ECLIPSE FOUNDATION, 2017). Porém, como consta no próprio site da empresa, o suporte para essa ferramenta está terminando. Isso se dá pelo fato de que a Google lançou, ano passado, a IDE Android Studio, implementada especificamente para assistir aplicativos Android. Esta ferramenta traz várias facilidades e funções built-in que facilitam e tornam o desenvolvimento mais rápido e organizado, apesar de que possuir tantos recursos acaba deixando sua interface carregada. Algumas de suas principais características são:

- Editor de traduções: Utilizando um arquivo específico “strings.xml” e com algumas configurações o aplicativo pode ser traduzido em outras línguas.

- Integração com Github: Com esta funcionalidade, o desenvolvedor pode tanto importar códigos exemplo do Github quanto sincronizar seus projetos com a plataforma.
- Suporte multiplataforma: Como o Android roda não só em smartphones e tablets, o Android Studio possui suporte à desenvolvimento para TVs, Wearables e Glass.
- Suporte à impressão digital: Os aplicativos podem validar a impressão digital do usuário com o comando “finger”.

3.3.4 Phonegap e Ionic

Com o aumento da demanda por aplicativos, pessoas ficaram pensando em formas de agilizar o processo de desenvolvimento, mas que mantivesse a qualidade. A partir disso foram surgindo frameworks como o Phonegap (PHONEGAP, 2017) e Ionic (IONIC FRAMEWORK, 2017), ambos baesados em Node.js, que possuem um conceito de apps híbridos, nos quais toda a programação base é feita em Javascript e as interfaces em HTML e CSS, tecnologias bastante conhecidas em web design.

Além de trazer rapidez e manter a qualidade, esses frameworks tiram o peso de ter que aprender uma linguagem específica para cada plataforma, seja Android, iOS ou Windows Phone. Eles permitem criar apps multiplataforma com apenas um código e não deixam a desejar quanto à acesso de componentes nativos do hardware, como câmera e geolocalização.

O principal ponto de atenção em relação ao desenvolvimento, que deve ser levado em consideração é a interface gerada com as ferramentas que foram utilizadas, que, como baseadas em tecnologias de web design, podem não se adequarem à padrões de interfaces mobile. As Figuras 9 e 10 mostram alguns projetos com Ionic e Phonegap, respectivamente:

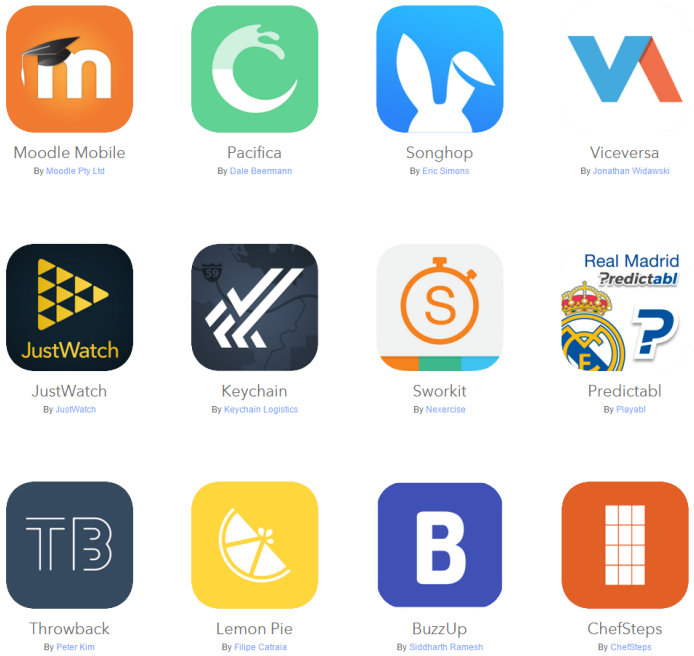

Figura 9: Aplicativos criados usando Ionic.

Fonte (Ionicframework.com)

Figura 10: Projetos desenvolvidos com Phonegap.

Fonte (Phonegap.com)

3.4 Webservices

Com o aumento das transações via Internet, a tecnologia de webservices começou a ganhar mais espaço no mercado (LIMA, 2012). Ao longo desta seção serão abordadas duas maneiras diferentes de se trabalhar com esta tecnologia: SOAP e REST.

3.4.1 SOAP

O SOAP (Simple Object Access Protocol) é um padrão para desenvolvimento de webservices definido pelo consórcio W3C. Geralmente este modelo utiliza o protocolo HTTP para transportar os dados e mensagens através da rede e se baseia na estrutura dos dados em XML.

3.4.2 REST

Seu idealizador, Roy Fielding, buscava nos anos 2000 reunir as melhores práticas de outros modelos em um novo. Baseando-se em modelos como cliente/servidor, sistema em camadas, cache e sem estado, Roy desenvolveu o modelo REST (Representational State Transfer). Suas principais características são o uso adequado dos métodos HTTP, uso apropriado de URL's, uso de códigos de status padrão para representar sucessos e falhas, (SAUDATE, 2013).

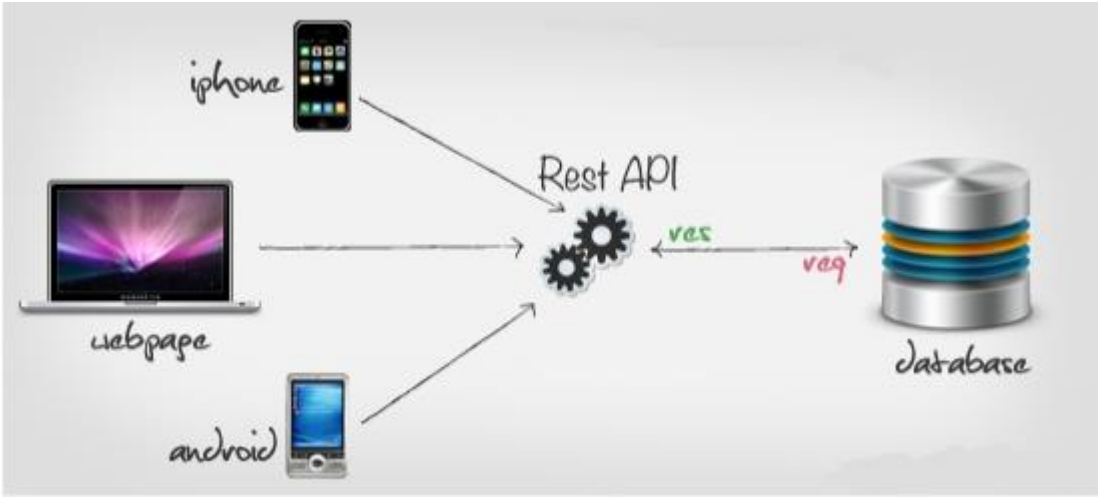

Figura 11: Exemplo de webservice utilizando abordagem REST.
Fonte (http://www.matera.com/br/2012/10/22/como-funciona-um-webservice-rest/)

Na Figura 11 podemos observar com clareza a arquitetura de uma aplicação utilizando o modelo de webservice REST. A aplicação roda em vários dispositivos com plataformas diferentes (MacOS, iOS e Android). Todos esses dispositivos necessitam das informações que estão gravadas no banco de dados, porém cada um possui uma forma de ler, estruturar e tratar os dados. Logo, para não ter que se desenvolver uma ferramenta para cada um, utiliza-se um padrão que atende o conjunto, criando um webservice, o qual envia as requisições ao banco e retorna os dados estruturados com uma linguagem universal, geralmente XML ou JSON.

3.4.3 SOAP x REST

Primeiramente iremos visualizar um exemplo de requisição e resposta realizadas através do SOAP e outra pelo REST, ambos utilizando XML e com o objetivo de listar os clientes cadastrados em um sistema:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <listarClientes xmlns="http://brejaonline.com.br/administracao/1.0/service"
   </soap:Body>
</soap:Envelope>
```

Figura 12: Requisição SOAP

Fonte (SAUDATE, 2013)

```
<soap:Envelope xmlns:domain="http://brejaonline.com.br/administracao/1.0/domain"
   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
       <listarClientesResponse xmlns="http://brejaonline.com.br/administracao/1.0
          <domain:clientes>
             <domain:cliente domain:id="1">
                <domain:nome>Alexandre</domain:nome>
                <domain:dataNascimento>2012-12-01</domain:dataNascimento>
             </domain:cliente>
             <domain:cliente domain:id="2">
                <domain:nome>Paulo</domain:nome>
                <domain:dataNascimento>2012-11-01</domain:dataNascimento>
             </domain:cliente>
          </domain:clientes>
       </listarClientesResponse>
    </soap:Body>
 </soap:Envelope>
```

Figura 13: Resposta SOAP

Fonte (SAUDATE, 2013)

Acima observamos nas Figuras 12 e 13 como são feitas as requisições no padrão SOAP e como o mesmo retorna os dados. Uma das características é a quantidade de tags presentes na resposta, que talvez possa tornar o código um pouco complexo para trabalhar em se tratando de enormes quantidades de dados.

No modelo REST a requisição é montada diretamente na URL. Exemplo: http://localhost:8080/cervejaria/clientes. Neste ponto já se nota uma certa facilidade em relação ao modelo anterior, observa-se a seguir a resposta:

```
<clientes>
   <cliente id="1">
      <nome>Alexandre</nome>
      <dataNascimento>2012-12-01</dataNascimento>
   </cliente>
   <cliente id="2">
      <nome>Paulo</nome>
      <dataNascimento>2012-11-01</dataNascimento>
   </cliente>
</clientes>
```

Figura 14: Resposta REST

Fonte (SAUDATE, 2013)

Pode-se notar como o modelo REST é mais leve, enxuto e organizado apenas com um simples exemplo, possuindo apenas as tags essenciais para estruturar as informações.

Outras vantagens do REST em relação ao SOAP são:

- SOAP utiliza apenas o método POST do HTTP para fazer diversas operações.
- Necessidade de criar interface WSDL, que é trabalhoso.
- Por utilizar apenas o método POST, não pode ser armazenado em cache, o que perde em segurança, pois o método citado pode alterar os dados de um recurso.

Não obstante, o SOAP possui alguns pontos fortes em relação ao REST:

- Como os cabeçalhos estão dentro da mensagem, ela fica independente do protocolo, podendo-se utilizar HTTP, SMTP e etc.
- Conceitos de segurança estão melhor especificados e difundidos.
- Conceitos sobre transação estão melhores especificados.

Portanto, é possível notar que ambas as abordagens possuem seus pontos negativos e positivos e não é cabível definir uma sendo melhor ou pior que a outra. Além do mais, deve-se definir a utilização de REST ou SOAP de acordo com as necessidades da aplicação que será desenvolvida.

3.5 E-commerce

O comércio eletrônico é um setor que vem crescendo bastante, juntamente com o aumento do número de internautas. Segundo pesquisa da Mintel (2014) sobre o e-commerce no Brasil, pôde-se observar que este teve uma ampliação de 250% entre os anos de 2008 e

2013 e prevê-se um desenvolvimento de 130% até 2018, chegando a marca de 115 bilhões de reais, considerando o valor do segmento.

Entretanto, ainda existem algumas barreiras que impedem que o comércio eletrônico se sobreponha às lojas físicas. A questão de segurança, por exemplo, é um dos maiores fatores. De acordo com a Mintel, cerca de 25% dos compradores preferem não fazer compras em sites que pedem o CPF. Outras questões como originalidade do produto e fraudes, como phishing, atrapalham o setor.

Por outro lado, temos uma extensa oportunidade de divulgação e propaganda de empresas que praticam e-commerce através das redes sociais, onde 17% dos consumidores brasileiros visitam os sites de varejistas pelo fato de terem visto seus anúncios em redes sociais, como por exemplo, o Facebook. Além disso, com o aumento no uso da internet e aplicativos, os dispositivos que permitem ao usuário realizar compras online como notebooks, tablets e smartphones vem auxiliando no desenvolvimento do comércio eletrônico.

Trazendo o assunto para o foco principal deste trabalho, nota-se que o setor de pedidos de alimentos, que se concentrava em atendimentos via ligação telefônica, vem se tornando pouco eficiente, pois é demorado, há problemas de a linha estar ocupada, má comunicação e custo da ligação. Com isso, foram surgindo alguns projetos para pedidos online, seja por sites ou aplicativos, como o iFood (IFOOD 2017), PedidosJá (PEDIDOSJÁ, 2017), Devorando (DEVORANDO, 2017) e Aiqfome (AIQFOME, 2017).

O aplicativo Aiqfome, presente em João Monlevade desde 2015, utiliza um modelo de negócio diferente dos outros citados. Em vez de centralizar suas ações na empresa, o Aiqfome possui um modelo de franquia, no qual uma pessoa pode adquirir os direitos e abrir uma franquia do aplicativo em sua cidade. Atualmente possui mais de 150 mil clientes cadastrados, quase 2 mil restaurantes, mais de 60 mil pedidos mensais, sendo 87% destes via mobile (smartphones e tablets) e apenas 13% via computadores. Além disso, o Aiqfome já movimentou cerca de 26 milhões de reais em pedidos por todo o Brasil.

Uma estimativa da Associação Brasileira de Bares e Restaurantes (ABRASEL, 2017) indica que o mercado de pedidos de alimentos deve movimentar R$ 9 bilhões em 2015, totalizando um aumento de R$ 1 bilhão em relação ao ano passado. Com a recente fusão com a concorrente RestauranteWeb no ano passado, a previsão do iFood é fechar o ano com R$ 1 bilhão em vendas brutas. Só em Porto Alegre, são feitos 15 mil pedidos por mês em mais de 50 lojas cadastradas.

No entanto, a maioria desses aplicativos se difundiram apenas em grandes centros urbanos, deixando de lado um grande potencial das cidades de pequeno e médio porte. Com isso, surge uma boa oportunidade de investimento nessas regiões, pois as grandes marcas ainda não estão inseridas.

Na tabela abaixo pode-se observar uma breve comparação entre 3 aplicativos: iFood, PedidosJá e Devorando. Nota-se que é possível ao cliente acompanhar o pedido e realizar pagamentos online nos 3 aplicativos. Já para o restaurante, não possui custos fixos o iFood e o PedidosJá. O Devorando não possui informações sobre esse quesito. A integração com o caixa é possível para os aplicativos iFood e Devorando, enquanto não foi informado para o PedidosJá.

	Para o cliente		**Para o restaurante**	
	Acompanhar o pedido	**Pagamento online**	**Custos fixos**	**Integração com caixa**
iFood	Sim	Sim	Não	Sim
PedidosJá	Sim	Sim	Não	Não informado
Devorando	Sim	Sim	Não informado	Sim

Tabela 1: Comparação entre aplicativos seus serviços.

De acordo com todos os aspectos abordados nesta seção, pôde-se notar uma gama considerável de tecnologias, ferramentas, frameworks e padrões que permeiam o desenvolvimento de sistemas baseados na web e aplicativos móveis. Além disso, as características do comércio eletrônico e como sua evolução em relação ao ramo de delivery de alimentos levou a idealização deste projeto.

Na próxima seção abordar-se-á de forma mais clara o porquê da utilização de cada uma, mas, de modo geral, será utilizado o framework Ionic juntamente com o AngularJS como base back-end e front-end para o aplicativo móvel e o framework CakePHP para a criação do Webservice. No lado do sistema web, o framework CakePHP entrará em cena novamente, compondo o back-end. Para o front-end, serão utilizadas as linguagens HTML, CSS (Bootstrap) e JavaScript.

4 Proposta

Com o aumento da venda dos smartphones, o uso de ligações, seja por telefone fixo ou móvel está cada vez menor. Os aplicativos se mostram mais vantajosos pelo seu melhor custo benefício, além de serem rápidos e práticos. De acordo com o portal Proxxima (2015), o uso de aplicativos deu um salto de 76% em relação a 2013, utilizando dados do relatório anual da Flurry, empresa do Yahoo.

As empresas começaram a notar que deveriam se adequar a essa nova tecnologia para melhor atender seus clientes. Com isso, surgiram diversos aplicativos de venda em vários seguimentos da indústria, inclusive no setor alimentício, que vêm alcançando

êxito considerável, como o “ifood” que apresenta mais de 200 mil pedidos por mês e o “pedidos já” que atende em 11 países, em mais de 400 cidades, com uma rede de mais de 15.000 restaurantes com serviço a domicílio.

Dentro deste contexto e aproveitando a pequena e, talvez, pouco expressiva difusão destes aplicativos nas pequenas cidades, o principal objetivo deste trabalho é desenvolver um aplicativo para dispositivos móveis com o intuito de atender a cidade de João Monlevade, podendo o cliente pesquisar restaurantes e lanchonetes, ver o cardápio e realizar o pedido. Para a administração do app, será desenvolvido um sistema web para controle de vendas, franqueados, restaurantes, gerentes, atendentes, produtos, preços.

Os principais objetivos do aplicativo são:

- Facilitar a pesquisa de restaurantes e cardápios;
- Prover uma gama considerável de variedade de restaurantes e cardápios;
- Diminuir os custos financeiros e de tempo do cliente para realização de encomendas;
- Agilizar o processo de realização de pedidos;
- Aumentar as vendas por delivery;

Os principais objetivos do sistema web que dará suporte ao aplicativo são:

- Permitir maior recebimento de solicitações e em menos tempo;
- Geração de gráficos e relatórios no intuito de gerenciar as vendas realizadas pelo aplicativo;
- Boa usabilidade através de uma interface simples, concisa e responsiva;

Para cumprir tais objetivos foram selecionadas algumas das ferramentas descritas na seção anterior, baseando-se no conhecimento do desenvolvedor sobre as mesmas e na garantia de qualidade que podem oferecer.

Um estudo feito por Hamann (2014) mostrou que o grau de participação no mercado global de aplicativos Android é de 84,7%, seguido de 11,7% de iOS e 2,5% de Windows Phone. Para complementar, ainda existem 1,3 milhão de apps na Play Store, seguido de 1,2 milhão na App Store e 1 bilhão de usuários ativos na plataforma Google contra 800 milhões na plataforma da Apple.

Com isso e com a premissa de que em João Monlevade a média se mantem, foi decidido começar o desenvolvimento do aplicativo deste trabalho utilizando as capacidades do Ionic, no intuito de iniciar a distribuição do app através da plataforma Android, mas futuramente, podendo se estender para outras plataformas com mais facilidade.

No contexto da Plataforma Web, foram analisados algumas linguagens e frameworks mostrados na seção anterior. Como quesito de escolha, foi levada em consideração a

experiência do desenvolvedor em CakePHP, uma vez que este foi trabalhado durante a disciplina relacionada a Sistemas Web e que faz parte da grade do curso de graduação. Como foi mostrado, é uma tecnologia segura, de ótimo desempenho e muito utilizada no mercado.

Outras tecnologias auxiliares que serão utilizadas são: Bootstrap, JavaScript e MySQL, sendo as duas primeiras na parte de Front-End, provendo dinamicidade ao site e área administrativa, além de responsividade e a última na parte de banco de dados.

Para melhor compreensão da finalidade dos sistemas e de suas funcionalidades, foi realizada uma etapa de análise e definição de requisitos e escopo, englobando a construção de artefatos como: requisitos funcionais e não funcionais; o product backlog; diagrama de casos de uso e definições da arquitetura.

4.1 Requisitos funcionais

Segundo Sommerville (2004), o artefato gerado pelos requisitos funcionais irá descrever o que os sistemas devem e/ou não devem fazer e como devem se comportar mediante interação do usuário. De acordo com as essas definições, este projeto visa atender os seguintes requisitos funcionais:

- O usuário do app deve conseguir realizar pedidos;
- O usuário do app deve conseguir cadastrar vários endereços de entrega, podendo troca-los de maneira fácil;
- O usuário do app deve ser capaz de pesquisar restaurantes mais próximos do seu raio de localização;
- O usuário do app deve conseguir acompanhar seu pedido através de tipos de status;
- O administrador do sistema deve ser capaz de inserir novos franqueados através do sistema web;
- O franqueado deve conseguir cadastrar restaurantes, bem como seus respectivos gerentes;
- O gerente deve ser capaz de inserir os atendentes do seu estabelecimento;
- O gerente e atendente devem ser capazes de visualizar pedidos que são realizados pelos usuários do app através do painel administrativo web;

4.2 Requisitos não funcionais

Os requisitos não funcionais, de acordo com Sommerville (2004), definem restrições sobre os serviços dos sistemas e, como o próprio nome sugere, não dizem respeito diretamente a funções específicas e sim a propriedades como confiabilidade, segurança,

usabilidade, desempenho, dentre outras. Os itens de requisitos não funcionais deste projeto dizem respeito às quatro propriedades citadas anteriormente e podem ser vistos abaixo:

- Desempenho – As páginas ou telas devem buscar informações essenciais ao seu funcionamento, de forma que não fiquem carregando por muito tempo, desanimando o usuário;
- Usabilidade – Deve ser fornecida uma interface limpa, responsiva (no caso do sistema web) e nomes amigáveis de menu e opções para facilitar a navegação do usuário;
- Segurança – Os sistemas têm de prover formas de garantir a privacidade dos dados dos usuários, bem como níveis de permissões com intuito de restringir acesso somente às funcionalidades permitidas em cada nível;
- Confiabilidade – Como o app estará sempre buscando dados pela web, caso ocorra alguma instabilidade ou perda de conexão com a internet o seu funcionamento será prejudicado, podendo ocorrer a quebra da aplicação. Portanto, o app deverá conseguir se manter através de tratamentos para esses tipos de exceções.

4.3 Product Backlog

O product backlog é o documento com a listagem de todas as funcionalidades do sistema e aplicativo separadas em seis serviços, os quais são: autenticação, encomenda, atendimento, gerência, franqueado e administração. Este documento é a única fonte de quaisquer mudanças a serem feitas no projeto, segundo Schwaber e Sutherland (2013), desenvolvedores do Scrum. Logo, esta seção foi subdividida de acordo com os seis serviços.

4.3.1 Autenticação

O escopo deste serviço é realizar tarefas que dizem respeito à autenticação dos usuários no sistema, tanto no aplicativo quanto na área administrativa. São essas tarefas:

- Login e logout;
- Recuperação de senha;
- Mostrar informações do perfil do usuário;
- Excluir conta do usuário (somente para clientes);
- Atualizar informações do perfil do usuário;

4.3.2 Encomenda

Este serviço será exclusivo do aplicativo na fase inicial do projeto, ou seja, na primeira versão deste projeto, as encomendas de comida serão realizadas apenas através do app. O escopo deste será realizar todas as tarefas que podem ser demandadas em uma encomenda feita pelo usuário. Logo, temos:

- Realizar pedido (buscar restaurantes e produtos)
- Ver todos os pedidos (status: em preparo, transporte, entregue)
- Visualizar cardápio de restaurantes

4.3.3 Atendimento

Área de acesso do atendente na qual ele poderá realizar suas tarefas. Este serviço pertence à área administrativa e tem como escopo duas tarefas:

- Visualizar todos os pedidos (filtrar por nome, status, data, localização)
- Atender pedido (mudar status: em preparo, transporte, entregue)

4.3.4 Gerência

Nesta parte, os usuários que poderão ter acesso deverão possuir credenciais de gerentes do sistema, para que se possa diferenciá-los dos atendentes e dar maior nível de acesso à área administrativa. Entende-se por gerente do sistema a pessoa responsável pela empresa que possui cadastro neste sistema de encomendas, por exemplo, gerente ou dono da empresa. Logo, suas tarefas são as seguintes:

- Visualizar todos os pedidos (filtrar por nome, status, data, localização)
- Atender pedido (mudar status: em preparo, transporte, entregue)
- Cadastrar, alterar, visualizar e excluir produtos
- Visualizar detalhes do seu restaurante
- Alterar dados do seu restaurante

4.3.5 Franqueado

Este serviço será destinado aos franqueados do sistema. Franqueados serão pessoas vinculadas ao aplicativo e que terão a responsabilidade de desenvolver parcerias com restaurantes interessados, eximindo o desenvolvedor desta tarefa. Este modelo tem como objetivo aumentar a escabilidade do sistema, podendo entrar em expansão em cidades e estados rapidamente, de modo que um franqueado possa conseguir vários parceiros

(restaurantes) e cada cidade tenha vários franqueados. Seguem as tarefas do franqueado dentro do sistema:

- Cadastrar, visualizar, alterar e excluir restaurantes
- Cadastrar, visualizar, alterar e excluir gerentes

4.3.6 Administração

A administração do sistema, a princípio, será realizada pelo desenvolvedor, porém a interface do sistema e funcionalidades que este usuário pode realizar serão desenvolvidas no sistema web com a premissa de que mais usuários administradores poderão ser adicionados ao sistema posteriormente, por isso o acesso via sistema não será totalmente liberado, o que será melhor explicado na seção 4.1 (sobre regras da modelagem relacional). Abaixo segue a lista com as tarefas do administrador:

- Cadastrar, alterar, visualizar e excluir franqueados, restaurantes e gerentes
- Visualizar pedidos

4.4 Casos de uso

Para o melhor entendimento do papel de cada usuário em relação às funcionalidades que pode utilizar, mostra-se na Figura 15 abaixo, o diagrama de casos de uso, porém em uma visão macro e não tão detalhada como no escopo:

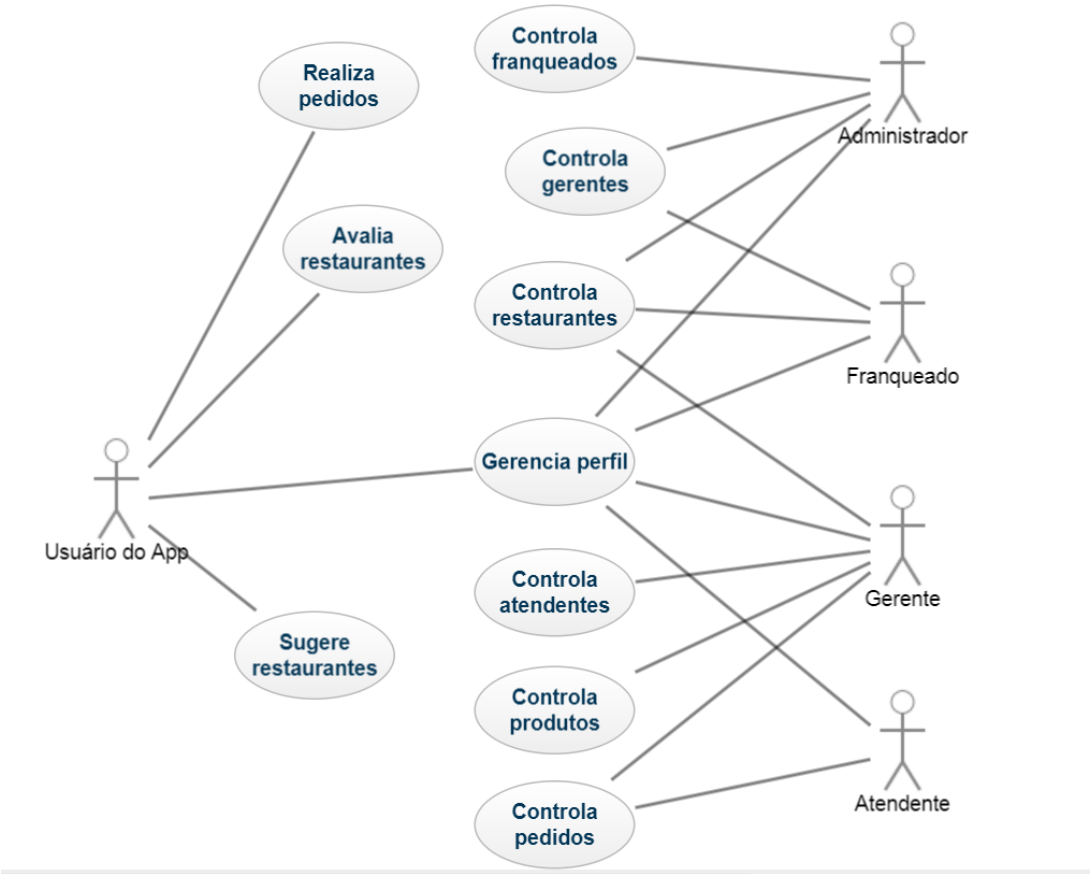

Figura 15: Diagrama de casos de uso do sistema

4.5 Arquitetura

Na seção 2.4 foi mostrada uma arquitetura geral de um webservice REST com intuito de apresentar esta abordagem, suas características e como funciona. Na Figura 16 abaixo pode-se observar a arquitetura utilizada neste projeto, que foi baseada no modelo REST. Logo após pode-se acompanhar a explicação de como as tecnologias se comunicam:

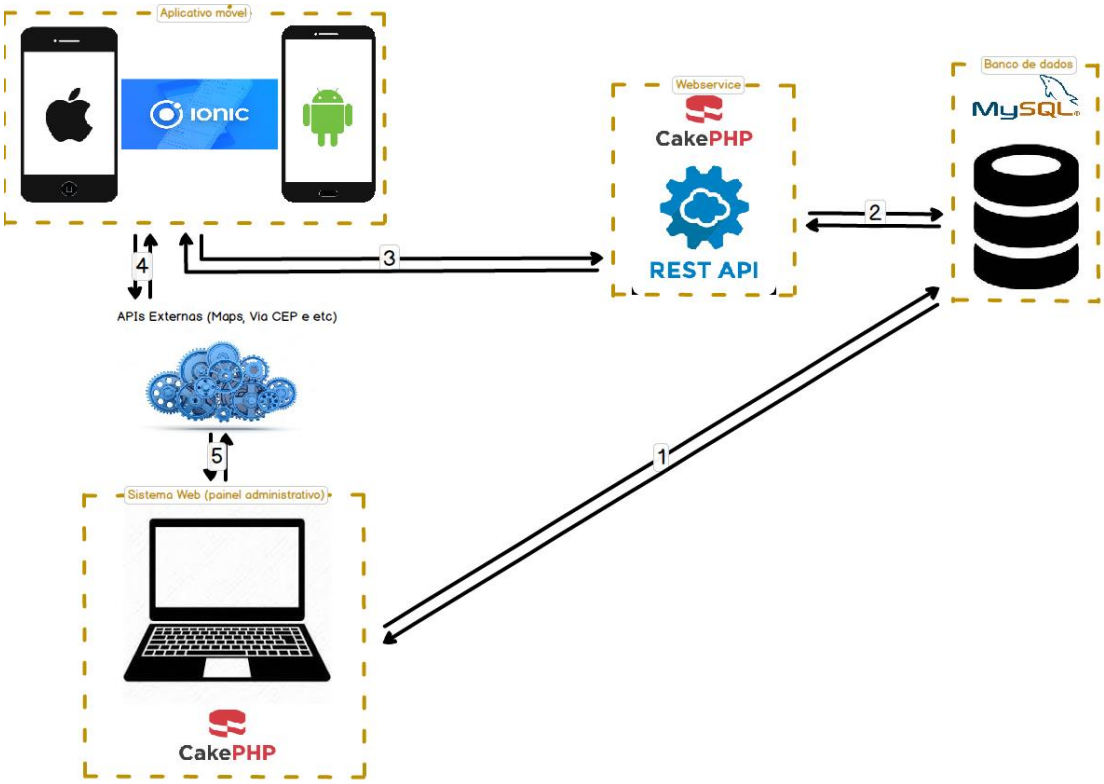

Figura 16: Arquitetura do projeto, comunicação entre as tecnologias

Para cumprir o que foi citado anteriormente no início da seção 3 - onde se propôs um aplicativo para realização de pedidos de comida e um sistema web para controlar as vendas – os quatro sistemas estão destacados pelos retângulos tracejados na Figura 16 (pode-se observar a logomarca de cada tecnologia base para implementação de cada um) e as APIs externas.

Será empregue, a partir deste momento, um padrão de nomenclatura para facilitar a citação dos sistemas com base nos títulos de cada retângulo:

- Aplicativo móvel = app delivery
- Sistema Web (painel administrativo) = painel admin
- Webservice = webservice
- Banco de dados = bd

Para apresentar a comunicação dos sistemas será utilizada a numeração de cada uma das ligações conforme a Figura 16, logo se tem:

- Ligação 1: Permite a troca de dados entre o painel admin e o bd. A comunicação é feita diretamente através das classes “Model” do framework Cakephp. Logo, o painel admin envia uma requisição ao bd, solicitando dados necessários para

realizar algum tipo de operação (inserir, consultar, alterar ou deletar) e o bd retorna estes dados.

- Ligações 2 e 3: Tornam possível a troca de informações entre o app delivery e o bd, através do intermediário: webservice. Logo, quando o app delivery precisa realizar alguma operação no bd (inserir, consultar, alterar ou deletar) é enviada uma requisição http ao webservice com as devidas informações para que o webservice entenda a necessidade do app delivery. O webservice, por sua vez, solicita os dados ao bd (da mesma forma da ligação 1, pois também usa o Cakephp como base) e padroniza a resposta em JSON, retornando-a para o app delivery.
- Ligações 4 e 5: As APIs externas serão auxiliares em vários momentos da execução dos sistemas, mas veremos detalhadamente suas funções na seção 4. Sobre a comunicação, elas irão ocorrer via requisições http e com respostas em JSON, semelhante às ligações 2 e 3, porém com intuito de buscar informações externas aos sistemas deste projeto.

Com isso, é possível notar como os sistemas devem estar interligados e trabalhando juntos, através das várias ligações citadas acima, para uma melhor experiência do usuário que poderá usufruir das funcionalidades. Em outras palavras, pode-se dizer que o aplicativo móvel se comunica com o banco de dados através do webservice e também se comunica com APIs externas para realizar algumas operações. Por outro lado, o sistema web é interligado diretamente ao banco de dados e realiza algumas requisições à APIs externas, de forma similar ao aplicativo.

Portanto, com todos os requisitos funcionais e não funcionais especificados, com todas as funcionalidades dos sistemas listadas no Product Backlog, os casos de uso ilustrando a atuação de cada usuário no sistema que utilizará e a arquitetura definida, pode-se iniciar o capítulo de desenvolvimento do projeto, no qual serão mostradas as características das funcionalidades desenvolvidas, que servirão para cumprir todos os requisitos citados neste capítulo.

5 Desenvolvimento

A seção de desenvolvimento aborda os aspectos que dizem respeito à construção de todos os sistemas envolvidos neste projeto. Para iniciar, será mostrada a modelagem relacional estabelecida para conseguir estruturar os dados. Após isso, será apresentado o painel administrativo web com suas caraterísticas, principais algoritmos e telas. Por fim, as

exposições da implementação do aplicativo móvel, bem como o serviço construído para integra-lo ao banco de dados relacional.

5.1 Modelagem Relacional

O banco de dados relacional foi modelado utilizando a tecnologia MySQL (sistema de gerenciamento de banco de dados). Abaixo é possível observar a Figura 17, correspondente ao modelo relacional completo e a seguir a explicação de sua estrutura e regras que foram adotadas para cumprir os requisitos do projeto:

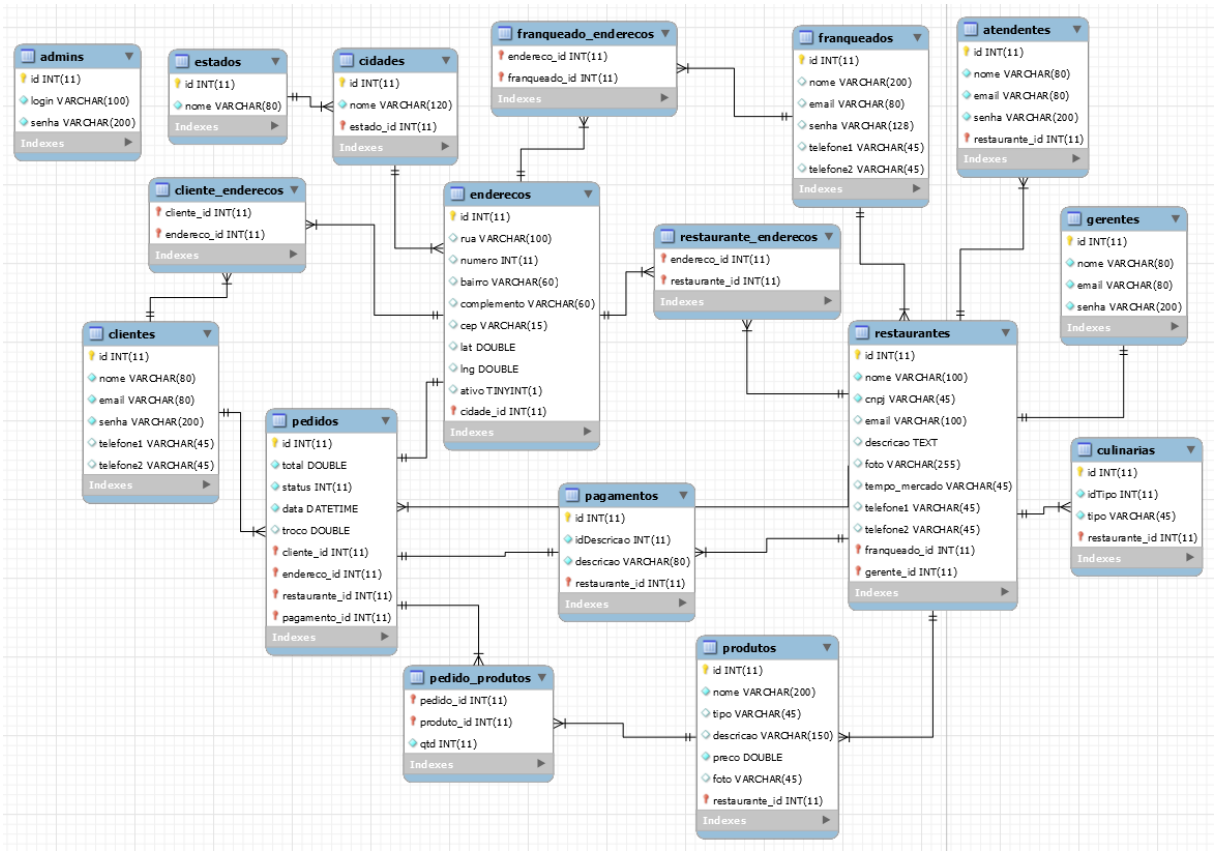

Figura 17: Modelo Entidade Relacionamento

A nomenclatura das entidades e campos de chave primária e estrangeira seguem as convenções do framework Cakephp e assim foi possível utilizar a ferramenta de automação (bake) que foi citada na seção 3.1.3.

Pode-se ver que existem cinco entidades que representam os tipos de usuários, são elas: admins, franqueados, gerentes, atendentes e clientes. Essas tabelas foram criadas para definir os níveis de permissão de cada um e para separar os relacionamentos que podem possuir. É válido lembrar que os clientes só têm acesso ao aplicativo e as outras quatro entidades ao sistema web.

O tipo de usuário admin não tem nenhum relacionamento, pois sua tabela servirá apenas para autenticação e então possui apenas os campos id, login e senha. Com isso, é decidido à nível de sistema quais as páginas que esse poderá acessar, pois apesar de estar no maior nível de permissão, o acesso via interface não conterá todas as páginas do sistema e sim as funcionalidades que foram listadas no product backlog.

Existem os franqueados um nível abaixo na hierarquia, cuja função principal é controlar gerentes e restaurantes. Do ponto de vista do negócio, os franqueados serão intermediadores entre o administrador e os gerentes, pois assim o administrador terceiriza as tarefas de negociação com os restaurantes, podendo perder um pouco de rentabilidade de curto prazo, mas com possibilidade de escalar o aplicativo em um tempo menor, com ganhos em médio e longo prazo. Ou seja, os gerentes passarão uma parte do que foi vendido pelo app ao franqueado e o franqueado repassa parte do que recebeu ao administrador. Para isso, o franqueado será vinculado e responsável por cada restaurante que inserir no sistema, podendo ser vários restaurantes, mas um mesmo restaurante não pode possuir mais que um franqueado. Assim chega-se a uma relação de um para muitos entre franqueados e restaurantes. Nota-se também uma relação de muitos para muitos entre franqueados e endereços, o que será explicado posteriormente quando for abordado o tema da tabela de endereços. Os atributos desta entidade foram definidos para autenticação e contato, sendo id, email e senha para autenticação e nome, telefone1 e telefone2 para contato.

Os gerentes serão responsáveis por gerir as informações de seu restaurante e atender pedidos. Como um gerente só pode estar associado a um restaurante e um restaurante só pode estar associado a um gerente, foi criada uma relação de um para um. Os campos do gerente também foram pensados para autenticação e contato, sendo id, email e senha para autenticação e nome para contato (o telefone estará na entidade restaurante, por não existir a necessidade de inserir o telefone pessoal do gerente dentro do contexto deste projeto).

Atendentes são usuários que terão apenas a função de atender pedidos, como seu nome sugere. O gerente poderá inserir vários atendentes em seu restaurante e um mesmo atendente não poderá estar associado a mais de um restaurante, logo existe uma relação de muitos para um entre o atendente e o restaurante. Seus atributos serão apenas para autenticação: id, nome, email, senha e a chave estrangeira restaurante_id para realizar a relação.

A entidade clientes possui atributos de autenticação e contato: id, nome, email, senha, telefone1 e telefone2. Para realizar várias compras os clientes têm uma relação de um para muitos com a entidade ‘pedidos’. O caso da tabela de endereços e clientes também será explicado a seguir.

A entidade ‘enderecos’ servirá para armazenar dados dos endereços de clientes, franqueados, restaurantes e pedidos. Para relacionar com todas essas entidades de maneira direta teriam de ser criadas quatro chaves estrangeiras e o sistema deveria tratar para que nenhuma fique nula, mas apenas uma contenha um valor relevante, pois um endereço não pode pertencer à mais de uma entidade ao mesmo tempo. Com o intuito de tratar este problema, foi decidido utilizar tabelas intermediárias entre cada entidade e a tabela de endereços. Assim, à primeira vista pode-se pensar que todas essas entidades teriam a possibilidade de portar vários endereços, porém isso não é verdade e o sistema, via interface, permite que seja inserido apenas um endereço às entidades. A tabela de endereços possui os campos id, rua, número, bairro, complemento, cep, latitude, longitude e cidade_id. Os atributos latitude e longitude serão utilizados para cálculos de distâncias. O atributo cidade_id faz referência à tabela de cidades, sendo uma relação de muitos endereços para uma cidade.

A tabela de cidades armazena o nome de uma cidade e o estado_id que indica à qual estado a cidade pertence, indicando uma relação de muitas cidades para um estado e a entidade ‘estados’ registra o nome dos estados.

A entidade restaurantes tem como papel armazenar todos os dados essenciais que serão apresentados no app aos potenciais clientes, além de dados para contato. Para isso temos os campos id, nome, cnpj, email, descrição, foto, tempo_mercado, telefone1, telefone2 e as chaves estrangeiras de franqueado_id e gerente_id. Os campos de descrição e tempo de mercado são informativos e serão mostrados na tela específica de detalhes do restaurante para o cliente que queira saber mais sobre o mesmo. Os atributos nome e foto serão apresentados já na lista de restaurantes para que os clientes possam escolher em qual restaurante realizar o pedido e os demais campos servirão para contato.

Foi criada uma tabela para gerenciar os tipos de culinária que o restaurante oferece. Dessa forma, o gerente conseguirá manipular facilmente, via sistema, as culinárias de seu restaurante, incluindo novas, excluindo ou mantendo outras. Esta entidade possui uma relação de muitos para um com o restaurante e carrega os campos id, idTipo, tipo e restaurante_id. O campo idTipo e tipo são utilizados para a criação de listas do tipo chave e valor no sistema web como forma de cumprir o requisito de fácil manipulação citado anteriormente.

A entidade ‘pagamentos’ segue ideia similar à ‘culinárias’, no que diz respeito à sua relação com restaurante e os campos que porta, apresentando assim uma interface simples com o gerente. Ademais, essa tabela possui uma relação com a tabela ‘pedidos’ de um para um, isto é, para cada pedido realizado pode ser utilizada uma forma de pagamento.

A tabela ‘pedidos’ carrega os campos id, total (valor total do pedido), status (pendente, em preparo, a caminho, entregue), data e hora do pedido e um atributo de troco

(caso o tipo de pagamento escolhido seja dinheiro). Os pedidos possuem um relacionamento de muitos para um com os clientes, isto é, um cliente pode realizar vários pedidos. Existem duas relações de um para um, sendo uma com a tabela ‘pagamentos’ que já foi explicada e a outra com ‘endereços’, para que um determinado pedido aceite apenas um endereço de entrega. Apesar dos pedidos não possuírem uma relação com os restaurantes, foi inserido o campo restaurante_id para facilitar na listagem de pedidos por restaurante. Há uma relação de muitos para muitos com a tabela de produtos, pois vários produtos podem estar em pedidos diferentes e os pedidos podem conter vários produtos. Com isso, foi gerada uma tabela intermediária ‘pedido_produtos’ que possui as chaves estrangeiras pedido_id e produto_id, além do campo de quantidade, que servirá para calcular o valor total da venda.

Por fim, a entidade ‘produtos’ carrega dados como id, nome do produto, tipo do produto (massa, hambúrguer, dentre outras possibilidades), descrição do produto, preço, foto e restaurante_id. Esta tabela possui uma relação de muitos para muitos com pedidos, conforme sinalizado no parágrafo anterior e uma relação de muitos para um com os restaurantes, ou seja, um restaurante pode ter vários produtos, mas um produto não pode estar em mais de um restaurante.

5.2 Implementação - Painel Administrativo Web

Nesta seção será mostrado o processo de implementação do painel administrativo web, bem como suas características, funcionalidades e particularidades, expondo apenas as principais funcionalidades do sistema (o Apêndice A conterá todas as funcionalidades). Para isso, apresentar-se-á o ambiente de desenvolvimento que foi configurado e apoiou todo o procedimento, destacando as ferramentas, frameworks e linguagens utilizadas. Posteriormente, será especificado o que foi desenvolvido em cada funcionalidade do product backlog (seção 3.3).

De modo geral, a ideia deste painel gerencial é auxiliar os usuários (administradores, franqueados, gerentes e atendentes) a realizar suas respectivas tarefas, abrangendo desde o cadastro de restaurantes e produtos até atendimentos de pedidos. Os usuários terão a possibilidade de controlar informações de perfil como: email, senha, telefone, entre outras.

Para o ambiente de desenvolvimento deste software têm-se o framework Cakephp 2.7.8, que utiliza a linguagem PHP sendo, neste caso, a versão 5.6.19. Essas tecnologias servirão de base para toda a lógica do sistema, bem como bloqueio de acessos e validações. Para hospedar o sistema localmente e realizar os testes (que serão abordados na seção 5) foi utilizado o servidor Apache 2.4.18. Já o editor de texto Sublime 3 foi usado para a escrita dos códigos.

A seguir será apresentado o processo que culminou com a construção das funcionalidades.

5.2.1 Autenticação

I. Login:

A tela de login será a primeira a aparecer quando o usuário entrar no link do sistema, conforme a Figura 18:

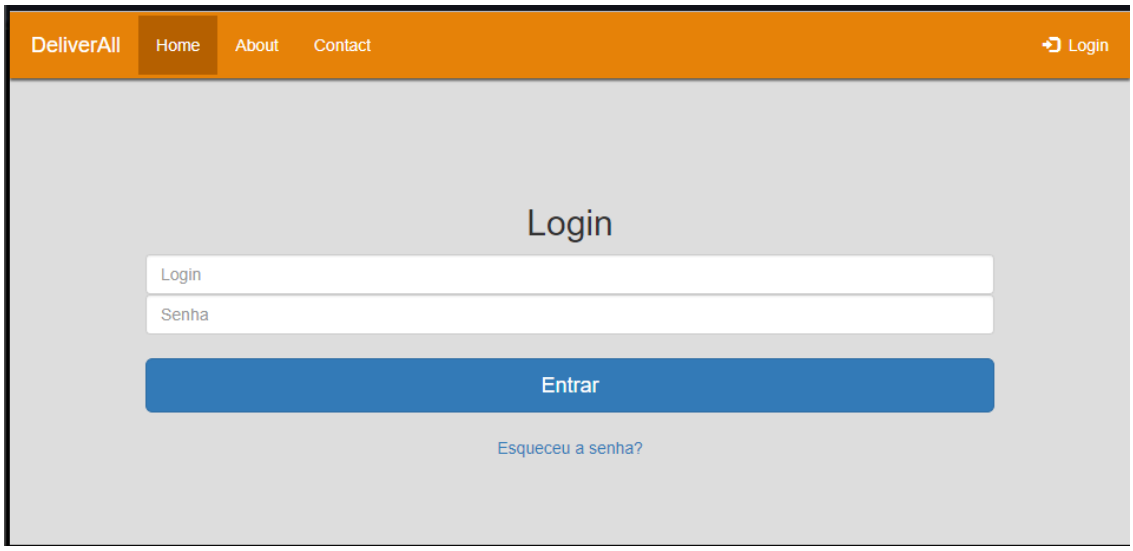

Figura 18: Tela de Login

Essa tela possui campos de email e senha, um botão para fazer login e um botão para recuperar a senha (será apresentado em um tópico separado – 5.2.1 III). Existe uma função de validação que é acionada pelo botão de login que é responsável por procurar o email e senha digitados nas tabelas correspondentes. A função para de procurar quando encontra um registro ou quando pesquisa todos os registros e não encontra nenhum, sendo que, no primeiro caso é aberta uma sessão e o usuário é enviado à sua respectiva tela principal. Já no segundo caso gera uma mensagem de erro acima do formulário.

II. Logout:

Todas as telas irão possuir uma barra de navegação superior (na cor laranja) e, ao passo que o usuário estiver logado será mostrado um botão para logout na extremidade direita desta barra, segundo mostra a Figura 19:

Figura 19: Barra de navegação com botão de Logout

Esse botão apenas encerrará a sessão do usuário e o redirecionará à tela de login.

III. **Recuperação de senha**:

Para recuperar a senha, o usuário deve informar um email cadastrado no sistema, como pode-se notar na Figura 20:

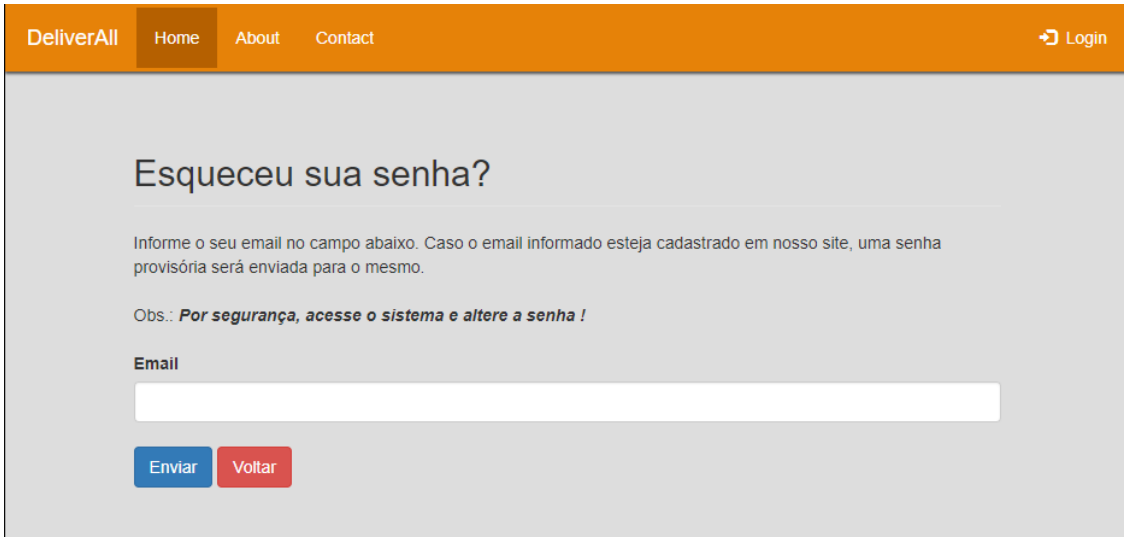

Figura 20: Tela para recuperação de senha

Existe uma função que valida o email e retorna uma mensagem de erro caso ele não exista ou envia uma senha gerada aleatoriamente ao email especificado.

IV. **Mostrar informações do perfil do usuário:**

Todos os usuários (exceto administrador) possuem uma tela para visualização dos dados de perfil, como informações pessoais e endereço, como mostrado na Figura 21:

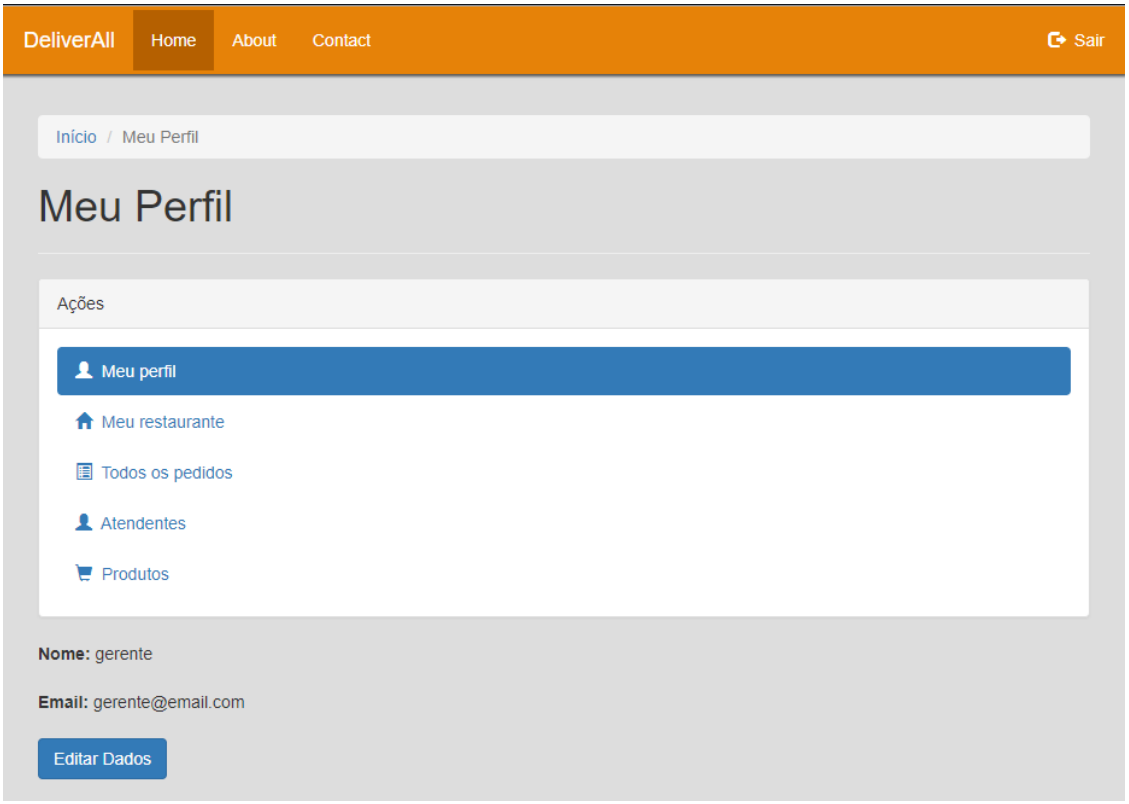

Figura 21: Tela “meu perfil”

A tela de perfil mostra essas informações e possui um botão “Editar dados” que redireciona à uma outra tela na qual o usuário pode editar suas informações.

5.2.2 Atendimento

I. Visualizar todos os pedidos:

Esta funcionalidade permite aos atendentes e gerentes a visualização, em apenas uma tela, do histórico de pedidos já realizados no restaurante em questão.

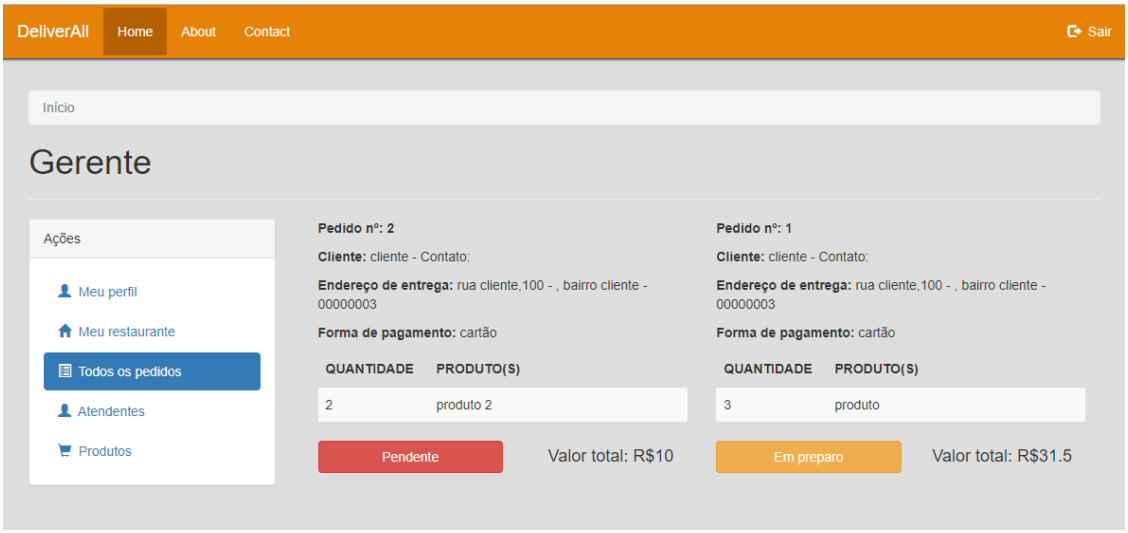

Figura 22: Histórico de pedidos

Como mostrado na Figura 22, os pedidos são carregados por padrão em ordem ascendente de status (“pendente”, “em preparo”, “à caminho” e “entregue”) e do identificador, pois pedidos pendentes e mais antigos têm maior prioridade de atendimento, ou seja, aqueles que possuem status “pendente” e menor identificador.

II. Atender pedido:

Na mesma tela em que é exibido o histórico de pedidos (Figura 22) existe um botão na parte inferior de cada pedido que, quando acionado, atualiza o status do pedido. Logo, haja visto que o último status será “entregue”, quando o mesmo é atingido o usuário não tem mais permissão para alterá-lo.

5.2.3 Gerência

I. Cadastrar produto:

Para que os gerentes possam inserir novos produtos existe uma tela de cadastro com um formulário.

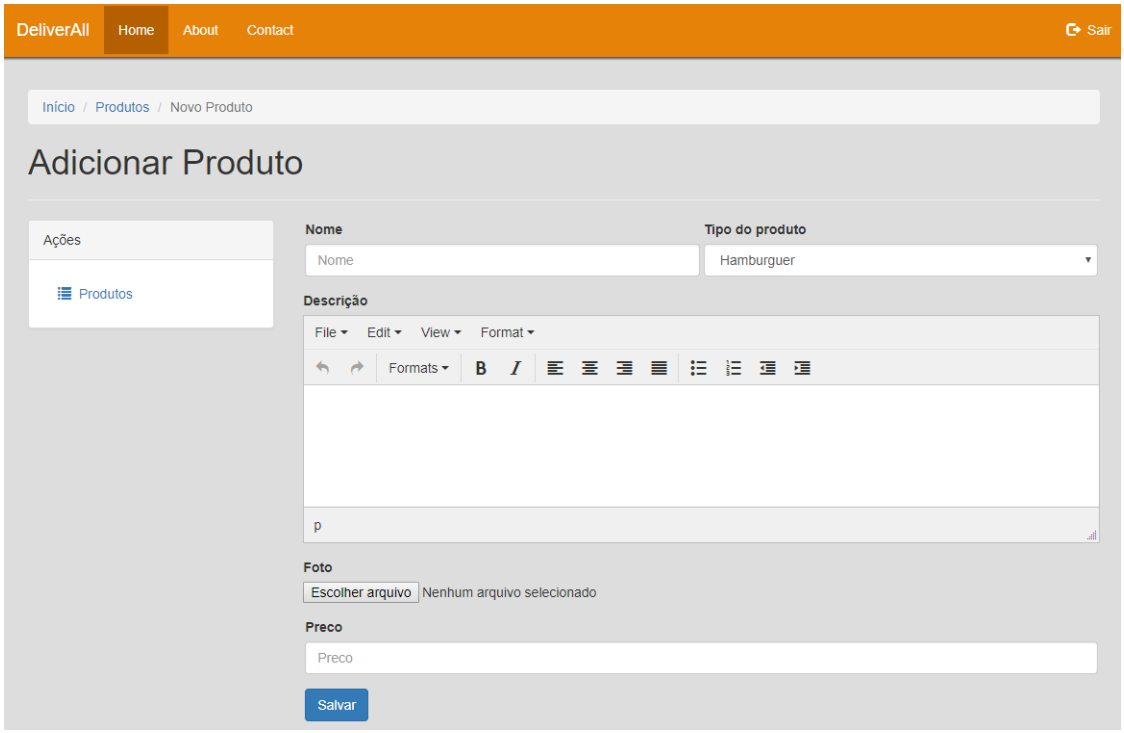

Figura 23: Tela para cadastro de produto

Basta preencher os campos exibidos na Figura 23 e clicar no botão “Salvar”, para que o novo produto seja armazenado no banco de dados da aplicação.

II. Visualizar detalhes do seu restaurante:

Todos os gerentes podem visualizar os detalhes de seu restaurante nesta tela, como informações de endereço, contato, cadastro em geral, além das culinárias e formas de pagamento que a empresa trabalha. A Figura 24 mostra a tela:

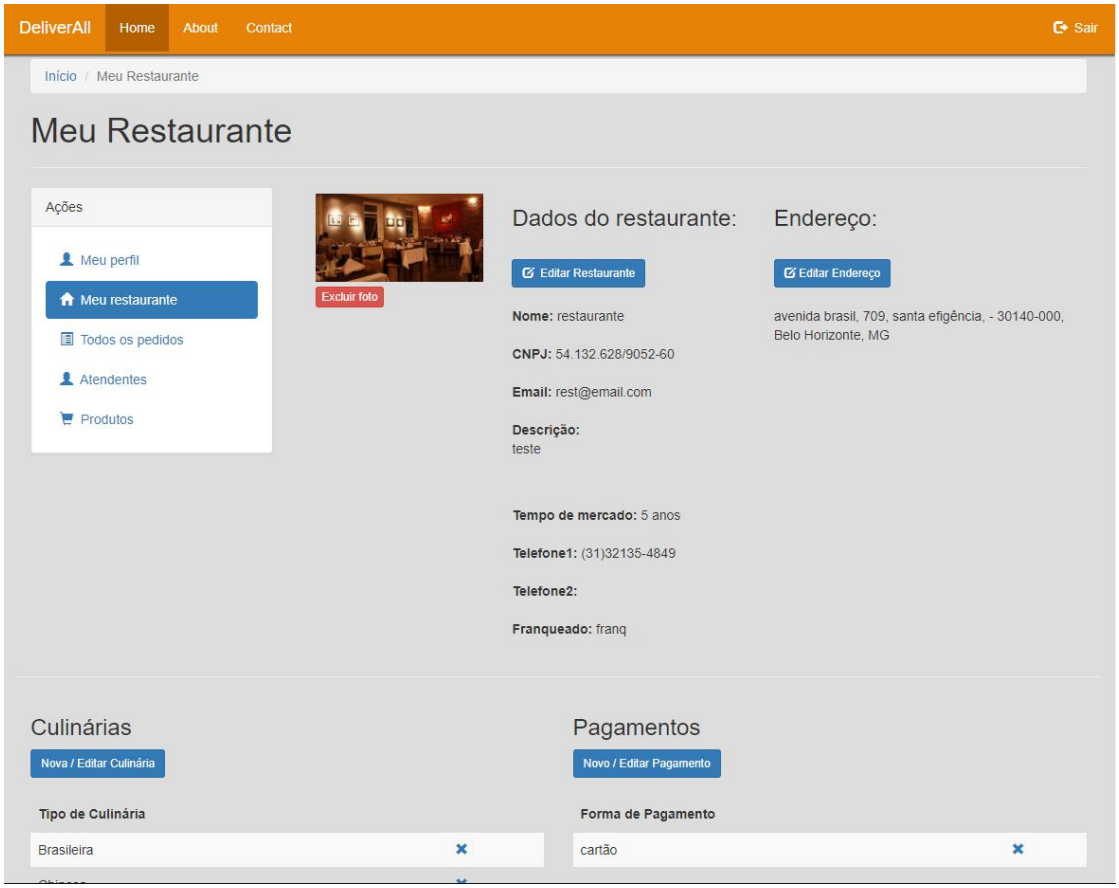

Figura 24: Tela “meu restaurante”

Para que se consiga edita-las existe um botão “Editar” que redireciona para a tela de atualização de informações.

5.2.4 Franqueado

I. Cadastrar restaurante:

Os franqueados deverão inserir novos restaurantes na base de dados através da seguinte página:

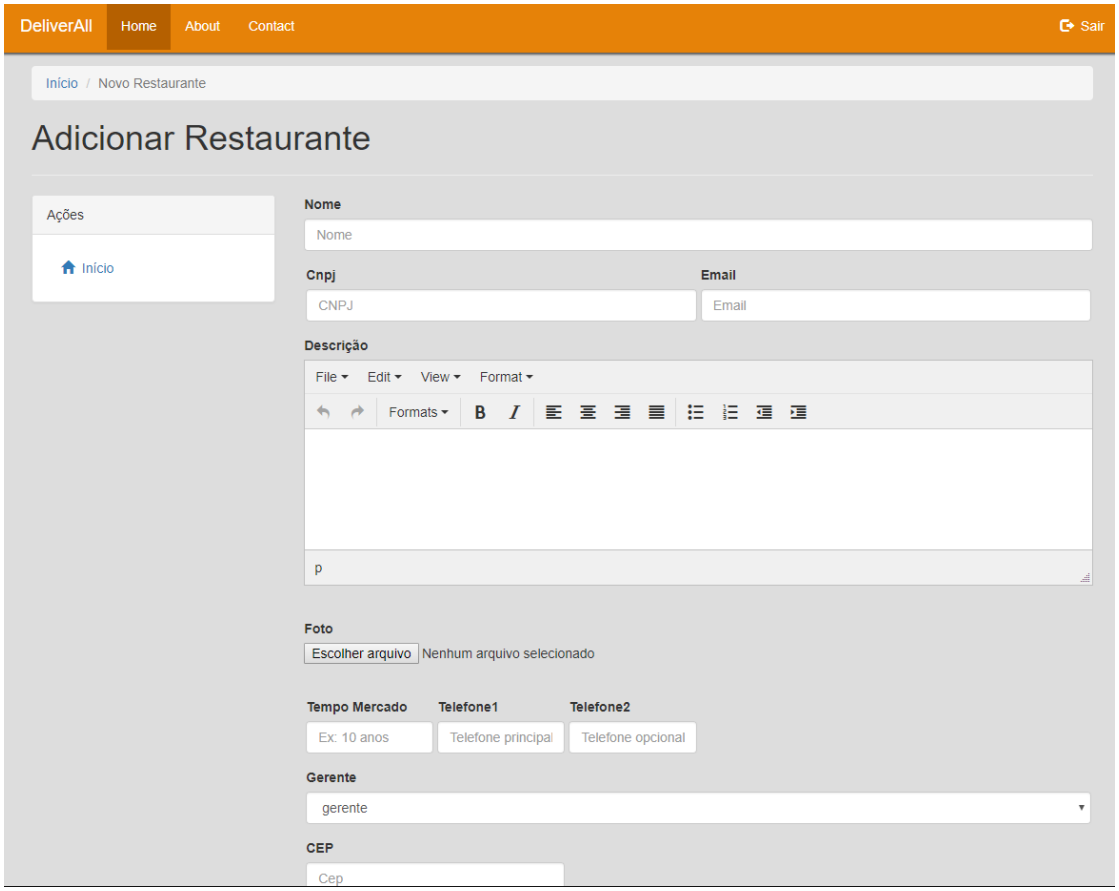

Figura 25: Tela para cadastro de restaurante

Conforme pode-se notar na Figura 25, é necessário preencher o formulário com os devidos dados. Como os restaurantes podem possuir uma logomarca, existe um botão para realizar o upload. Quando o usuário clica no botão abre-se o explorador de arquivos para que ele possa escolher uma imagem e, após isso, a imagem é copiada para uma pasta pública dentro dos arquivos da aplicação e o banco de dados salva apenas o caminho para esta imagem. Outra funcionalidade desta tela está no cadastro do endereço, na qual o usuário precisa informar apenas o CEP para que o sistema recupere os detalhes do endereço através de uma requisição “GET” na API “Via CEP”. A requisição retorna os dados em formato JSON para que a aplicação possa trata-los da maneira correta e preencher os demais campos de endereço.

5.2.5 Administração

I. Cadastrar franqueado:

A tela de inserção de novos franqueados possui um formulário com os respectivos campos que compõem o registro de um franqueado. Um detalhe que foi introduzido para

facilitar o preenchimento pelo usuário é a utilização de máscaras nos campos de CEP e telefones, conforme a Figura 26:

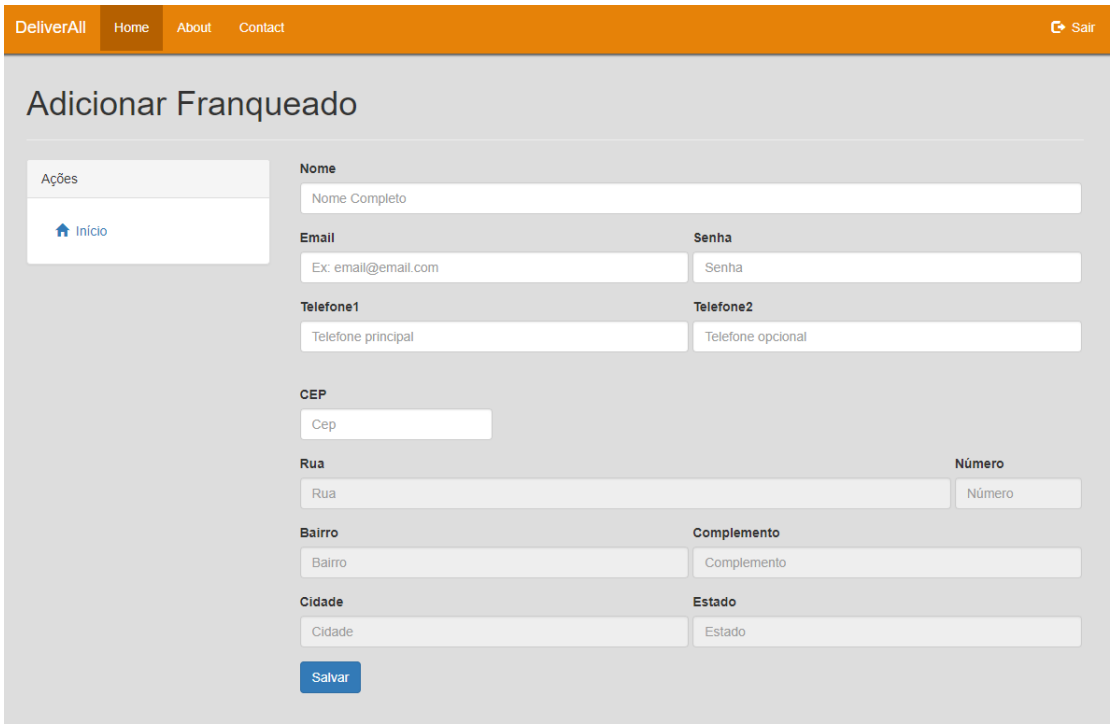

Figura 26: Tela para inserção de franqueados

As máscaras são caracteres pré-definidos inseridos nos campos do formulário e que são substituídos à medida que o usuário digita. O mais importante é que a quantidade de caracteres que podem ser inseridos não pode ultrapassar a quantidade já existente no campo, trazendo mais segurança aos dados e prevenindo possíveis erros de digitação.

5.3 Implementação - Aplicativo Móvel e Webservice

Nesta seção será mostrado o processo de implementação do aplicativo e webservice, bem como suas características, funcionalidades e particularidades expondo apenas as principais funcionalidades do sistema, ficando as demais disponibilizadas em detalhes no ANEXO A. Para isso, apresentar-se-á o ambiente de desenvolvimento que foi configurado e que apoiou todo o procedimento, destacando as ferramentas, frameworks e linguagens utilizadas.

O framework base para a construção deste aplicativo é o Ionic na versão 3, cuja função é resolver toda a lógica do software e a interface. O editor de textos Sublime Text 3 também foi utilizado nesta parte da tarefa. De acordo com o que foi explicado na seção 3.5 sobre a arquitetura, mais uma vez entra em cena o framework Cakephp 2.7.8 que, desta vez, funcionará como um webservice. Um detalhe importante e que será extensamente abordado

a partir deste ponto é a codificação de erros utilizada no webservice. Para facilitar a comunicação entre os sistemas, foi adotado um padrão que funciona da seguinte maneira:

- Retorno -1 – Registro não encontrado.
- Retorno -2 – Registro duplicado.
- Retorno -10 – Erro genérico (ex: falta de internet, campo vazio).

A seguir mostra-se a relação das principais funcionalidades do aplicativo desenvolvido.

5.3.1 Autenticação

I. Login:

Para que o cliente possa usufruir de funcionalidades, o aplicativo de delivery deve possuir um cadastro e autenticação através da tela de login, como pode-se ver na Figura 27:

Figura 27: Tela de login

Após efetuar o login, o usuário será redirecionado à tela de cadastro de endereço (caso ainda não possua nenhum) ou à tela de lista de restaurantes. Para realizar a validação de email e senha e autenticar o cliente, o sistema realiza o seguinte processo:

- Validação dos campos – Existe uma função para verificar se algum dos campos está vazio e, caso verdadeiro solicita que o usuário preencha, caso falso passa ao próximo passo.

- Requisição “POST” ao webservice – Esta requisição envia os dados digitados (email e senha) para o webservice.
- Webservice recebe requisição, valida e responde – O webservice começa validando os campos vazios novamente, para o caso de o aplicativo falhar na validação dos mesmos. Caso verdadeiro retorna -10, caso falso busca um cliente no banco de dados com as credenciais passadas. Caso encontre, retorna o objeto cliente encontrado em formato JSON, caso contrário retorna -1.
- Aplicativo trata resposta – Caso a resposta seja o objeto cliente, o aplicativo salva em uma variável e atualiza uma variável de controle no armazenamento interno do dispositivo para que sempre que o cliente abra o aplicativo, ele permaneça logado. Caso a resposta seja algum erro, o aplicativo exibirá a mensagem de erro no topo da tela.

II. Cadastro de usuário:

A tela de cadastro de usuário possui campos para nome, email e senha, conforme pode-se observar na Figura 28:

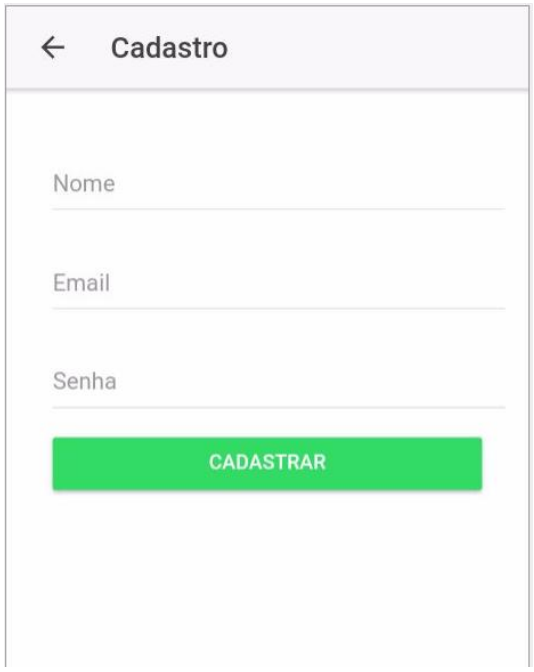

Figura 28: Tela de cadastro

O processo de cadastro é semelhante ao de login, porém os dados são registrados no banco de dados como um novo cliente e a aplicação redireciona o usuário para o cadastro de endereço (já atualizando a variável para mantê-lo logado). Nesta funcionalidade existe uma validação de cliente duplicado, na qual o webservice realiza uma busca no banco pelo email

e caso o email já esteja cadastrado retorna o código de erro -2 ao aplicativo, que trata de exibir uma mensagem amigável ao usuário.

III. Cadastro de endereço:

O cadastro de endereço pode ser realizado de duas formas, conforme pode-se notar na Figura 29:

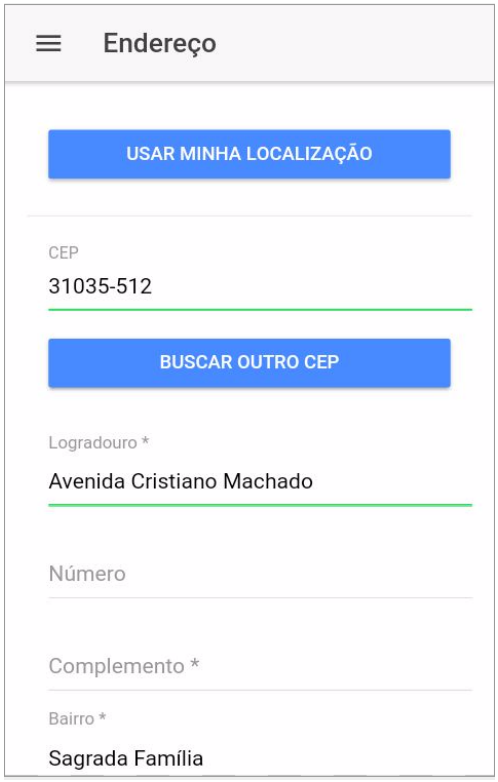

Figura 29: Tela para inserir endereço

A primeira forma é pela geolocalização do dispositivo, ou seja, o próprio smartphone irá informar ao aplicativo suas coordenadas geográficas (latitude e longitude), que serão convertidas em endereço e já preencherão os devidos campos, bastando o usuário apenas conferir e informar o restante. Já a segunda alternativa permite ao cliente informar o CEP e a aplicação se utilizará novamente do serviço “Via CEP” para realizar as conversões e preencher os campos.

IV. Excluir conta de usuário

O cliente pode fechar sua conta a qualquer momento, bastando navegar à tela de “Meu Perfil” e acionar o botão “Excluir Conta”, como mostra a Figura 30:

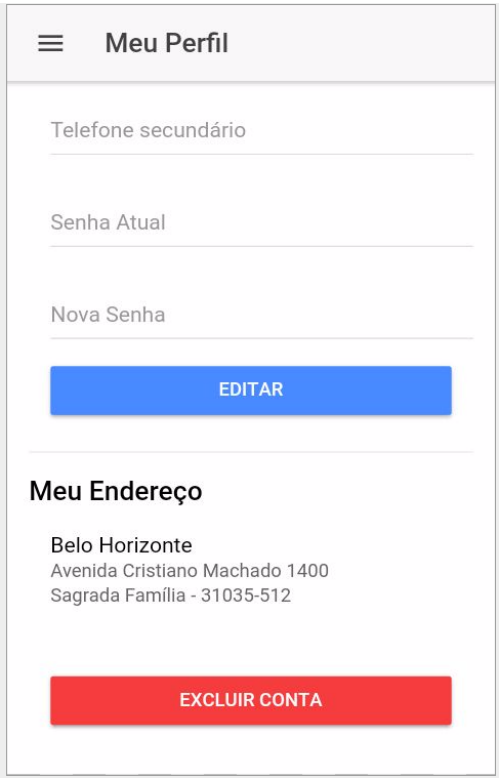

Figura 30: Excluir conta

Fazendo isso, todos os registros deste cliente serão apagados do banco, ou seja, pedidos, endereços e etc.

5.3.2 Encomenda

I. Realizar pedido:

O fluxo para realizar um pedido será o seguinte: primeiro o usuário deverá escolher um restaurante na lista, conforme a Figura 31. Neste caso, os restaurantes serão apresentados conforme a cidade na qual o usuário está cadastrado e em ordem crescente de distância entre o restaurante e o endereço do usuário.

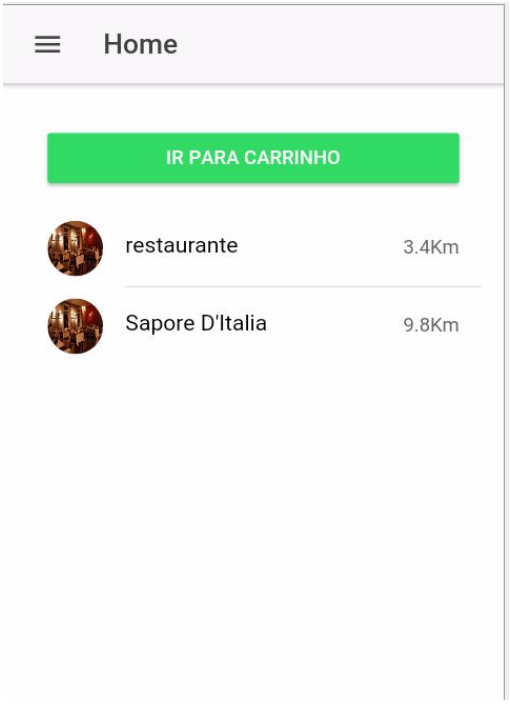

Figura 31: Lista de restaurantes

Feito isso, deverá escolher um produto no cardápio do restaurante em questão (Figura 32) e a quantidade do produto e adicionar ao carrinho (Figura 33):

Figura 32: Produtos

Figura 33: Adicionar produto ao carrinho de compras

Feito isso, poderá escolher outros produtos e adiciona-los ao carrinho conforme sua vontade. Clicando no botão do carrinho de compras (Figura 34) o cliente verá o valor total de seu pedido e poderá retirar produtos ou prosseguir para escolher forma de pagamento e finalizar o pedido, como ilustrado na Figura 35:

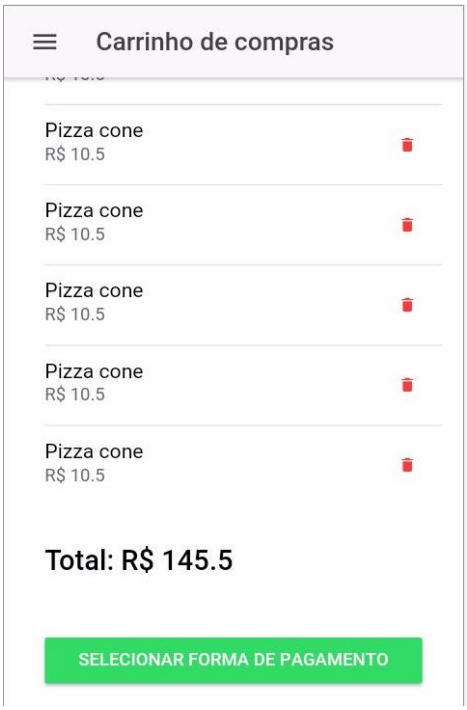

Figura 34: Carrinho de compras

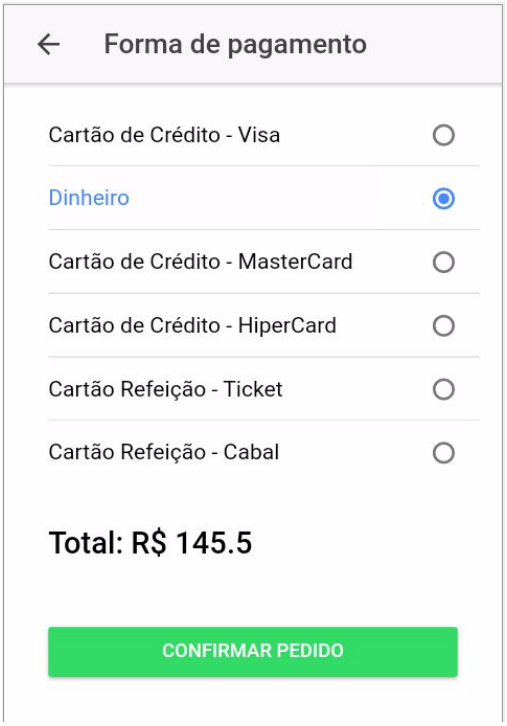

Figura 35: Escolher forma de pagamento

Ao finalizar o pedido, o restaurante solicitado poderá iniciar o atendimento assim que algum atendente ou gerente visualizar o pedido em seu painel administrador web.

Com base em todas as funcionalidades que foram explicadas nesta seção, foi possível entender como os sistemas foram desenvolvidos e como é o seu funcionamento. A seguir, será iniciada a seção de testes, na qual cada teste descrito compõe cenários que permitirão entender como será o comportamento dos sistemas em cada funcionalidade de acordo com possíveis ações do usuário.

6 Testes

Esta seção irá apresentar o roteiro de testes utilizados em cada funcionalidade implementada. É importante mencionar que os testes foram escritos antes do desenvolvimento para que pudessem servir de base e, com isso, ter uma visibilidade maior de como o sistema deverá se comportar em cada um dos casos. Os testes foram escritos em forma de histórias de usuário, com cenários principais e alternativos, isto é, para cada caso foi criado um cenário principal e zero ou mais cenários alternativos. Segundo Helm e Wildt (2014), as histórias de usuário são uma prática ágil de desenvolvimento de sistemas, que auxiliam na construção mais eficiente dos requisitos. A estrutura base de uma história de usuário deve responder à três perguntas: Quem? O quê? Por quê? Sendo assim o padrão utilizado neste trabalho será:

Como um *usuário,*
Eu *quero/devo/preciso de alguma coisa,*
Para *alguma certa finalidade.*

Pode-se notar a utilização de formatação em itálico em partes da história de usuário. O objetivo dessa formatação diferenciada é enfatizar que esses pedaços serão preenchidos em cada serviço do sistema conforme suas especificações.

Cada história de usuário representará um serviço deste projeto e cada serviço possui diversas funcionalidades. Os testes serão apresentados para cada funcionalidade e serão divididos em cenários, isto é, o fluxo de interação entre usuário e sistema e as respostas do sistema em determinada ação do usuário. Serão construídos cenários principais e alternativos, nos quais os principais representarão um fluxo normal em que não ocorrem erros. Já os alternativos serão construídos pensando em possíveis casos de erro ou até mesmo casos em que uma mesma funcionalidade pode ser realizada de diversas formas. A seguir pode-se observar os testes descritos para as principais funcionalidades.

6.1 Autenticação

6.1.1 Autenticação no painel administrativo

Como um usuário,
Eu preciso me autenticar no sistema,
Para ter acesso às funcionalidades do painel administrativo web.

6.1.1.1 Login

Cenário principal:
Dado que o usuário abra o link do site,
Então será redirecionado à tela de Login,
Quando o usuário preencher e-mail e senha,
Então será redirecionado à página principal do sistema de acordo com o tipo de usuário que for.

6.1.1.2 Logout

Cenário principal:
Dado que o usuário esteja em qualquer tela do sistema,
Quando o usuário clicar em “Sair”,
Então abre um alerta: “Deseja realmente sair?”,

Quando o usuário clicar em “Sim”,
Então sai do sistema e redireciona à tela de Login.

6.1.1.3 Recuperação de senha

Cenário principal:
Dado que o usuário abra o site, não esteja logado e tenha esquecido a senha,
Quando o usuário clicar no botão “Esqueci a senha”,
Então redireciona para a tela de “Recuperar senha”,
Quando preencher e-mail e clicar em enviar
Então envia um e-mail com nova senha aleatória e exibe uma mensagem “Nova senha foi enviada para seu e-mail”.

6.1.1.4 Mostrar informações do perfil do usuário

Cenário principal:
Dado que o usuário esteja logado no sistema,
Quando o usuário clicar em “Meu perfil”,
Então entra na tela “Meu perfil” e mostra os dados do usuário como nome, e-mail e etc.

6.1.2 Autenticação no aplicativo

Como um cliente,
Eu preciso me autenticar no sistema,
Para ter acesso às funcionalidades do aplicativo.

6.1.2.1 Login

Cenário principal:
Dado que o usuário abra o aplicativo, tenha cadastro e não esteja logado,
Quando o usuário inserir o login e senha e clicar no botão entrar,
Então abre a tela para cadastro de endereço.

Cenário alternativo (I - Login e/ou senha inválidos):
Dado que o usuário abra o aplicativo, tenha cadastro e não esteja logado,
Quando o usuário inserir login e/ou senha inválidos e clicar no botão entrar,
Então abre um alerta dizendo “login e/ou senha inválidos”.

Cenário alternativo (II - Já possui endereço cadastrado):
Dado que o usuário abra o aplicativo, tenha cadastro e não esteja logado,

Quando o usuário inserir o login e senha e clicar no botão entrar,
Então abre a tela principal com a lista de restaurantes.

6.1.2.2 Cadastro de usuário

Cenário principal:

Dado que o usuário abra o aplicativo e não tenha cadastro no mesmo,
Quando o usuário clicar em “Cadastro”
Então entra na tela de Cadastro
Quando inserir nome, e-mail e senha e clicar em “Cadastrar”
Então mostra mensagem “Cadastro realizado com sucesso” e redireciona para a tela de cadastro de endereço.

Cenário alternativo (I - E-mail já está cadastrado):

Dado que o usuário abra o aplicativo já tenha cadastro no mesmo,
Quando o usuário clicar em “Cadastro”
Então entra na tela de Cadastro
Quando inserir nome, e-mail e senha e clicar em “Cadastrar”
Então mostra mensagem “Usuário já cadastrado, faça login ou recupere a senha” e redireciona para a tela de login.

6.1.2.3 Cadastro de endereço

Cenário principal:

Dado que o usuário abra o aplicativo e esteja logado e não possua endereço cadastrado,
Então entra na tela “Endereço”,
Quando o usuário preencher os dados do meu endereço e clicar em “prosseguir”,
Então insere meu endereço e redireciona à tela principal com a lista de restaurantes.

6.1.2.4 Excluir conta de usuário

Cenário principal:

Dado que o usuário esteja logado,
Quando o usuário clicar no Menu -> Meu Perfil-> Excluir Conta,
Então exibe alerta: “Deseja realmente excluir sua conta?”,
Quando clicar em “Sim”,
Então apaga o usuário e redireciona para tela inicial (Login).

Cenário alternativo (I - Desistiu de excluir ou clicou por engano):

Dado que o usuário esteja logado,
Quando o usuário clicar no Menu -> Meu Perfil-> Excluir Conta,
Então exibe alerta: “Deseja realmente excluir sua conta?”,
Quando clicar em “Não”,
Então apenas fecha o alerta e mantém na tela ”Meu Perfil”.

6.2 Encomenda

Como um cliente,
Eu quero realizar um pedido,
Para receber comida em casa.

6.2.1 Realizar pedido

Cenário principal:
Dado que o usuário abra o aplicativo, esteja logado e possua pelo menos um endereço cadastrado,
Então o aplicativo me redirecionará à tela com a lista de restaurantes,
Quando o usuário selecionar um restaurante, selecionar um produto, a quantidade e a forma de pagamento e clicar em “Fechar pedido”,
Então meu pedido é enviado ao restaurante em questão e serei redirecionado à tela com detalhes do pedido para acompanhar o processo de entrega através dos status.

6.3 Atendimento

Como um atendente,
Eu quero utilizar a área de atendimento,
Para ver pedidos e atendê-los.

6.3.1 Visualizar todos os pedidos

Cenário principal:
Dado que o usuário esteja logado no sistema,
Então já serei redirecionado à tela com a lista de todos os pedidos.

6.3.2 Atender pedido

Cenário principal:
Dado que o usuário esteja logado no sistema,
Quando o usuário clicar em “Visualizar Pedidos”,
Então já serei redirecionado à tela com a lista de todos os pedidos,

Quando o usuário clicar no botão “Pendente”,
Então o status deste pedido troca para “Em preparo” e o botão sofrerá alteração em sua cor e seu texto, de acordo com o status seguinte “Em preparo”,
Quando o usuário clicar no botão “Em preparo”,
Então o status deste pedido troca para “À caminho”,
Quando o usuário clicar em “À caminho”,
Então o status deste pedido troca para “Entregue”.

6.4 Gerência

Como um gerente,
Eu quero acessar o sistema,
Para gerenciar meus produtos e visualizar e atualizar informações do meu restaurante.

6.4.1 Cadastrar produto

Cenário principal:
Dado que o usuário esteja logado no sistema,
Quando o usuário clicar em “Produtos”,
Então abre a página de produtos listando todos os cadastrados,
Quando o usuário clicar em “Novo produto”,
Então abre a página de cadastro de produto,
Quando o usuário informar (nome, tipo, descrição, preço, imagem e clicar em “Salvar”,
Então cadastra novo produto e retorna à tela “Produtos”.

6.4.2 Visualizar detalhes do restaurante

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar em “Meu restaurante”,
Então redireciona para a página contendo os dados do restaurante.

6.5 Franqueado

Como franqueado,
Eu quero acesso minha área restrita,
Para controlar restaurantes e gerentes.

6.5.1 Cadastrar restaurante

Cenário principal:

Dado que o usuário esteja logado,
Quando o usuário clicar em “Novo restaurante”,
Então abre a página de cadastro de restaurante,
Quando o usuário informar (nome, cnpj, email, descrição, foto, tempo de mercado, telefones, gerente e endereço) e clicar em “Salvar”,
Então cadastra novo restaurante e retorna à tela “Restaurantes”.

6.6 Administração

Como administrador,
Eu quero acessar minha área restrita,
Para gerenciar franqueados, restaurantes e controlar o sistema de forma geral.

6.6.1 Cadastrar franqueado

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar em “Novo franqueado”,
Então abre a página de cadastro de franqueado,
Quando o usuário informar (nome, email, senha, telefones e endereço) e clicar em “Salvar”,
Então cadastra novo franqueado e retorna à tela “Franqueados”.

6.7 Resultados obtidos

Como citado anteriormente no início da seção de testes, eles foram escritos antes do desenvolvimento, exclusivamente para guiar etapa e deixar claro como deveria ser implementada cada funcionalidade e como o sistema se comportaria diante das ações do usuário.

Com isso, os testes foram executados em paralelo com o desenvolvimento para garantir o cumprimento de cada cenário. Entretanto, alguns cenários foram alterados ao longo do decorrer do projeto, pois não se adequavam às limitações das ferramentas de desenvolvimento ou simplesmente não foram escritos e pensados da maneira correta e, por isso, não iriam preencher os devidos requisitos.

Então, na versão final dos testes que está neste projeto (no Apêndice A), os sistemas conseguem desempenhar todos os casos, tanto os cenários principais, quanto os alternativos.

7 Conclusões e trabalhos futuros

Este projeto foi muito importante para exercitar todo o conhecimento adquirido durante o curso de sistemas de informação. Nota-se que várias áreas foram permeadas, como a engenharia de software, modelagem relacional, computação móvel e sistemas para a web.

De acordo com as funcionalidades e testes apresentados, pode-se concluir que os objetivos iniciais do projeto foram alcançados. O aplicativo consegue apoiar os clientes nas solicitações de pedidos de delivery e o sistema web permite que o administrador, os franqueados, gerentes e atendentes controlem suas respectivas tarefas.

A utilização dos testes em formato de histórias de usuário foi fundamental para auxiliar os trabalhos de verificação e validação de cada funcionalidade após seu desenvolvimento. Os frameworks utilizados (Cakephp e Ionic) apoiaram muito o desenvolvimento, gerando um aumento na produtividade e reuso de códigos.

É válido mencionar que, como em qualquer projeto, surgiram dificuldades que valem a pena mencionar. A conciliação entre horas de trabalho em regime CLT e algumas poucas horas diárias para dedicação em um projeto não tão simples é um exemplo disso. A pretensão em produzir um aplicativo bonito, com logomarca e telas fluidas e com diversas animações para uma melhor experiência de utilização logo foi se tornando uma tarefa árdua e posteriormente, houve uma priorização das funcionalidades em relação à interface, pois de nada adiantaria um aplicativo vistoso que não conseguiria cumprir seu principal objetivo: realizar pedidos.

Como foi o primeiro projeto utilizando o Ionic, mesmo encontrando ótimas referências e uma extensa documentação, algumas questões exigiram horas e horas de trabalho para construir elementos simples.

Com isso, o aplicativo possui algumas limitações, como não permitir que um cliente possua mais de um endereço cadastrado (endereço residencial e comercial). Outra questão é a funcionalidade de busca, tanto para restaurantes quanto para produtos, que resultaria em uma possível agilidade para a realização de pedidos. Se tratando de agilidade, também é possível citar os favoritos, no qual um usuário poderia acessar os restaurantes em que mais compra de forma rápida. Existem outras funcionalidades que podem ser agregadas ao projeto em trabalhos futuros, assim como as que já foram citadas. Permitir que usuários avaliem os restaurantes em que já realizaram pedidos, melhorias na interface tanto do aplicativo quanto no sistema web, aplicação de notificações no aplicativo, para que os clientes possam acompanhar o pedido mais facilmente e atualização automática da tela de pedidos no sistema web, para alertar a entrada de um novo pedido.

Referências

ABRASEL. **Mercado de delivery de alimentos cresce no Brasil**: Tecnologia no atendimento e diversidade de estabelecimentos impulsionam hábito de consumo. Disponível em: <http://ap.abrasel.com.br/noticias/119-mercado-de-delivery-de-alimentos-cresce-no-brasil>. Acesso em: 21 jun. 2017.

SERRANO, Filipe. **Apesar dos altos preços, vendas de iPhone crescem 41%**. 2015. Disponível em: http://exame.abril.com.br/tecnologia/noticias/apesar-dos-altos-precos-vendas-de-iphone-crescem-41. Acesso em: 16/12/2015.

MERCATO, Mattia. **A doce história do Android**. 2015. Disponível em: http://www.androidpit.com.br/historia-do-android. Acesso em: 17/12/2015.

Schwaber, K., Sutherland, J. (2013). **The Scrum Guide™: The Definitive Guide to Scrum: The Rules of the Game.** Austin: Scrum.Org and ScrumInc.
Disponível em: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf. Acesso em: 29/02/2016.

MINTEL **Setor de comércio eletrônico no Brasil cresceu 250% nos últimos cinco anos**. 2014. Disponível em: http://brasil.mintel.com/imprensa/varejo-imprensa/setor-de-comercio-eletronico-no-brasil-cresceu-250-nos-ultimos-cinco-anos. Acesso em: 20/12/2015.

AGÊNCIA BRASIL **Brasil está acima da média em compras online via smartphones**. 2015. Disponível em: http://exame.abril.com.br/economia/noticias/brasil-esta-acima-da-media-em-compras-online-via-smartphones. Acesso em: 20/12/2015.

FOOD MAGAZINE **Mercado de delivery de alimentos cresce no Brasil**. 2015. Disponível em: http://www.foodmagazine.com.br/food-service-noticia-fique-por-dentro/mercado-de-delivery-de-alimentos-cresce-no-brasil. Acesso em: 20/12/2015.

FELIPINI, Dailton. **EMPREENDEDORISMO NA INTERNET: Como agarrar esta nova oportunidade de negócios.** São Paulo: Blue, 2015.

MENDES, Laura Zimmermann Ramayana. **E-Commerce: origem, desenvolvimento e perspectivas.** 2013. 63 f. Tese - Curso de Ciências Econômicas, Universidade Federal do Rio Grande do Sul, Porto Alegre.

RAVULAVARU, Arvind. **Learning Ionic: Build real-time and hybrid mobile applications with Ionic.** Birmingham: Packt, 2015.

NEUBURG, Matt. **Programming iOS 9: Dive deep into views, view controllers and frameworks.** 6. ed. Estados Unidos: O'reilly Media, 2015.

KARPOV, Valeri; NETTO, Diego. **Professional AngularJS.** Indianapolis: Wrox, 2015.

JUNEAU, Josh. **Introducing to Java EE 7: A look at what's new.** Estados Unidos: Apress, 2015.

BHAUMIK, Snig. **Bootstrap Essentials: Use the powerful features of Bootstrap to create responsive and appealing web pages.** Birmingham: Packt, 2015.

HAMANN, Renan. **IOS, Android e Windows Phone: números dos gigantes comparados.** 2014. Disponível em: <http://www.tecmundo.com.br/sistema-operacional/60596-ios-android-windows-phone-numeros-gigantes-comparados-infografico.htm>. Acesso em: 20 jun. 2016.

LIMA, Jean Carlos Rosário. **WEB SERVICES (SOAP X REST)**. 2012. 41 p. Monografia (Tecnólogo em Processamento de Dados) - Faculdade de Tecnologia de São Paulo, São Paulo, 2012.

SAUDATE, Alexandre. **REST**: Construa API's inteligentes de maneira simples. São Paulo: Casa do Código, 2013. 101 p.

SOMMERVILLE, Ian. **Engenharia de Software**. 6ª. ed. São Paulo: Pearson Education, 2004.

VENNERS, Bill. **The Making of Python**: A Conversation with Guido van Rossum, Part I. Disponível em: <http://www.artima.com/intv/python.html>. Acesso em: 17 maio 2017.

FORCIER, Jeff; BISSEX, Paul; CHUN, Wesley. **Pyhton Web Development with Django**. [S.l.]: Pearson Education, 2009. 380 p.

GRINBERG, Miguel. **Flask Web Development**: Developing Web Applications with Python. California: O Reilly, 2014. 238 p.

PYRAMID Framework. Disponível em: <https://trypyramid.com/>. Acesso em: 23 maio 2017.

PARREIRAS, F. S., BAX, M. P. **Gestão de conteúdo com softwares livres**. KMBrasil, 2003, São Paulo.

CAWLEY, Christian. **10 Most Popular Content Management Systems Online**. Disponível em: <http://www.makeuseof.com/tag/10-popular-content-management-systems-online/>. Acesso em: 31 maio 2017.

GOLDSTEIN, Emily. **PHP: The Ultimate**: Step by Step guide for beginners on how to learn PHP and MYSQL programming in just 6 hours. [S.l.: s.n.], 2015. 182 p.

PHP, Manual. **História do PHP**. Disponível em: <https://secure.php.net/manual/pt_BR/history.php.php>. Acesso em: 16 maio 2017.

AVOYAN, Hovhannes. **As melhores plataformas PHP para 2015**. Disponível em: <https://goo.gl/Pxroxm>. Acesso em: 19 maio 2017.

HELM, Rafael; WILDT, Daniel. **Histórias de Usuário**: Por que e como escrever requisitos de forma ágil. 3. ed. 2014.

AIQFOME. **Leve a terceira maior plataforma de delivery online do Brasil para sua cidade**: Tenha seu próprio negócio. Disponível em: <http://licenciamento.aiqfome.com>. Acesso em: 10 ago. 2017.

GINIGE, Athula; MURUGESAN, San. **Web engineering**: An introduction. IEEE multimedia, v. 8, n. 1, 2001.

SERRANO, Felipe. **Vendas de iPhone crescem 41% no Brasil em 2014, mesmo com mercado em desaceleração**: Apesar dos altos preços, Apple teve forte crescimento no país; mercado de celular teve queda de 11%. EXAME, 02 abr. 2017. Disponível em: <http://exame.abril.com.br/tecnologia/vendas-de-iphone-crescem-41-no-brasil-em-2014/>. Acesso em: 02 ago. 2017.

THE ECLIPSE FOUNDATION. **Eclipse**: About the Eclipse Foundation. Disponível em: <https://eclipse.org/org/>. Acesso em: 21 jun. 2017.

DANIEL, Viana. **O que é front-end e back-end?**: Entenda melhor sobre oque e, e onde aplicar. Disponível em: <http://www.treinaweb.com.br/blog/o-que-e-front-end-e-back-end>. Acesso em: 20 maio 2017.

SYMFONY. **Projects using Symfony**. Disponível em: <http://symfony.com/projects>. Acesso em: 03 ago. 2017.

TABLELESS. **Introdução ao Elm - diga adeus aos runtime errors**: Uma introdução básica à linguagem Elm. Disponível em: <https://tableless.com.br/>. Acesso em: 10 maio 2017.

IONIC FRAMEWORK. **The top open source framework for building amazing mobile apps**. Disponível em: <https://ionicframework.com/>. Acesso em: 25 jun. 2017.

PHONEGAP. **Build amazing mobile apps powered by open web tech**. Disponível em: <https://phonegap.com/>. Acesso em: 20 set. 2017.

CAKE SOFTWARE FOUNDATION. **Cake Software Foundation**. Disponível em: <https://cakefoundation.org/>. Acesso em: 21 jun. 2017.

IFOOD. **Delivery pra qualquer fome:** peça e receba em casa. Disponível em: <https://www.ifood.com.br/>. Acesso em: 10 jul. 2017.

PEDIDOSJÁ. **Delivery de comida online**. Disponível em: <https://www.pedidosja.com.br/>. Acesso em: 10 jul. 2017.

DEVORANDO. **Delivery**. Disponível em: <https://www.devorando.com.br/>. Acesso em: 15 mar. 2017.

Apêndice A

8 Implementação - Painel Administrativo Web

8.1 Autenticação

I. Atualizar informações do perfil do usuário:

A tela para alterar informações de perfil é semelhante à tela de cadastro, porém, ao invés de inserir um novo registro na base, ela apenas atualiza o registro em questão, caso exista alguma alteração. É importante salientar que os campos obrigatórios não podem ficar em branco, por isso, esses campos continuarão a possuir validações HTML.

8.2 Gerência

I. Visualizar todos os pedidos:

Este item será implementado da mesma forma que no serviço de atendimento.

II. Atender pedido:

Este item será desenvolvido da mesma forma que no serviço de atendimento.

III. Alterar produto:

Na tela de alterar dados de produtos existirá um formulário com todos os dados em modo de edição, como mostrado na Figura 36:

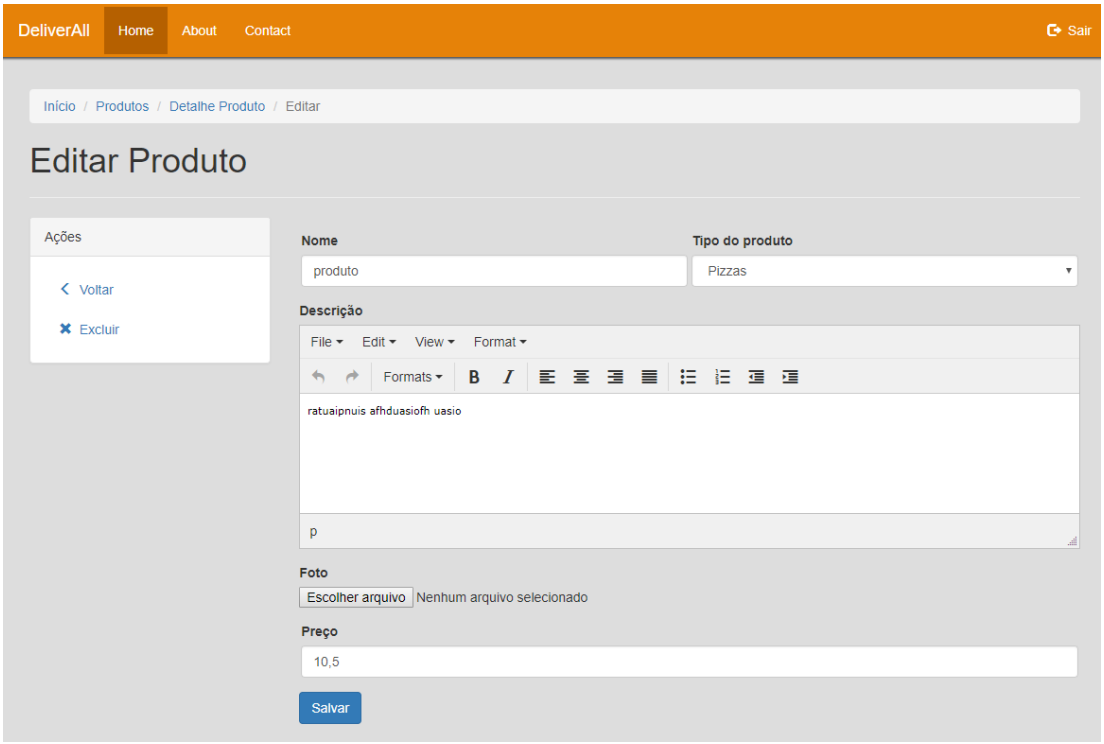

Figura 36: Tela para editar dados de produtos

Basta que o usuário altere os campos desejados e clique no botão salvar, assim o sistema salva essas informações no banco de dados.

IV. Visualizar produtos:

Para que o gerente possa visualizar todos os produtos cadastrados em seu restaurante foi criada uma tela que mostra uma tabela listando-os, como se pode observar na Figura 37:

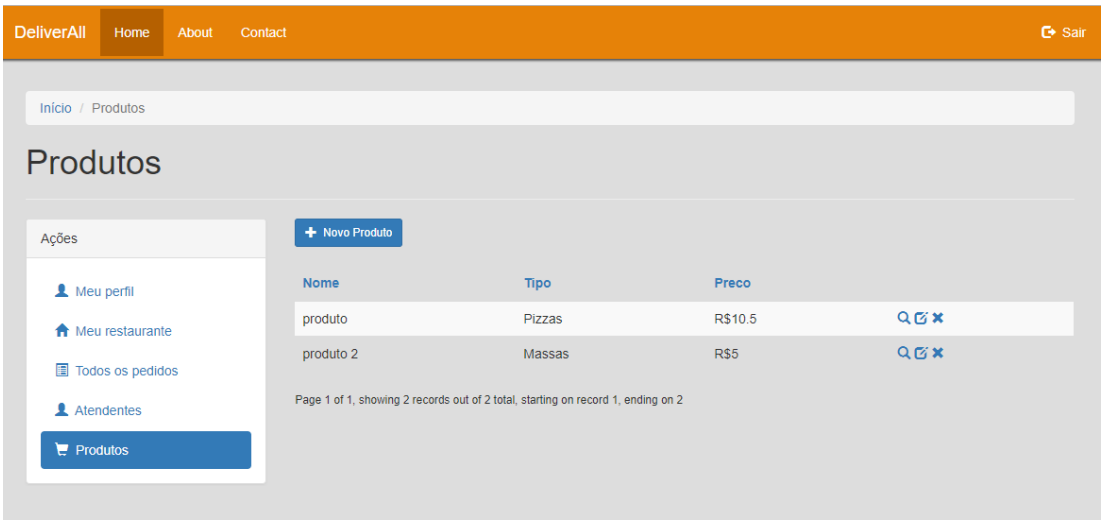

Figura 37: Tela que lista todos os produtos

Pode-se notar que em cada linha da tabela são exibidas informações de um produto, bem como botões que facilitam a navegação do usuário para as páginas de visualização de detalhes do produto, página de editar dados e botão de exclusão do produto.

A página de detalhes do produto apresentará todas as informações que um registro de produto pode ter, como pode-se notar na Figura 38:

Figura 38: Tela de detalhes do produto

V. Excluir produto:

Para excluir um produto, basta que o gerente clique no botão de exclusão mostrado na Figura 39. Com isso, uma tela de confirmação será exibida:

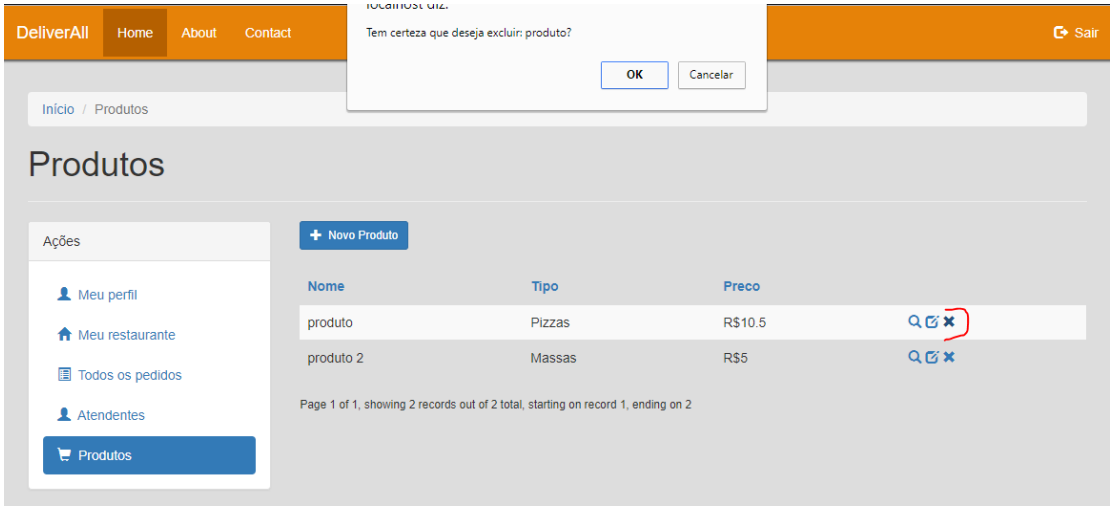

Figura 39: Tela de exclusão de produto

Dada a confirmação o sistema apagará completamente esse registro da base de dados.

VI. Alterar dados do seu restaurante:

Esta tela será similar a tela de cadastro, porém sua funcionalidade é salvar os dados alterados nos campos em vez de criar um novo registro.

VII. Cadastrar atendente:

A tela de cadastro de atendente exibirá um formulário HTML com os campos necessário para inserir um novo registro de atendente no banco, como mostra a Figura 40:

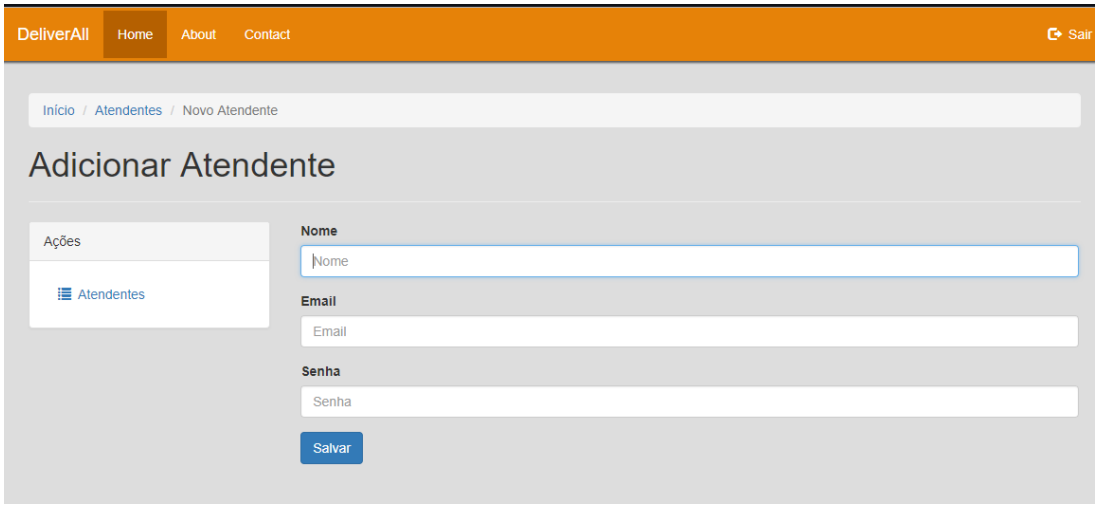

Figura 40: Cadastro de atendente

VIII. Alterar atendente:

Para editar os dados de um atendente será apresentada uma tela similar a tela de cadastro (Figura 40), porém com a função de salvar os dados editados no mesmo registro em questão.

IX. Visualizar atendente:

A tela de visualização de atendentes exibirá uma lista em forma de tabela para que o usuário consiga visualizar todos os atendentes de seu restaurante, como apresenta a Figura 41:

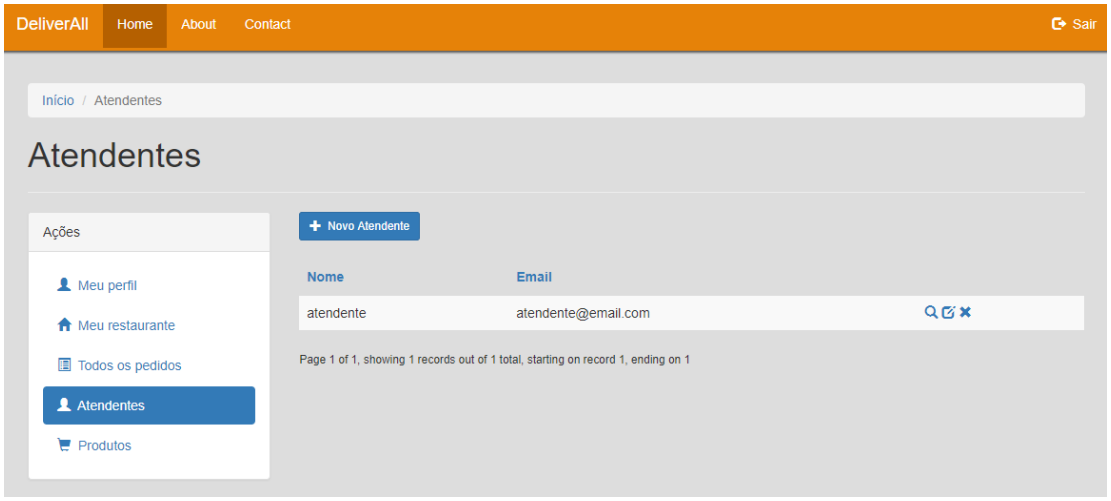

Figura 41: Lista de atendentes

Para visualizar os detalhes de um atendente específico será oferecida uma tela para isso. Essa tela exibirá todos os campos do atendente, como mostra a Figura 42:

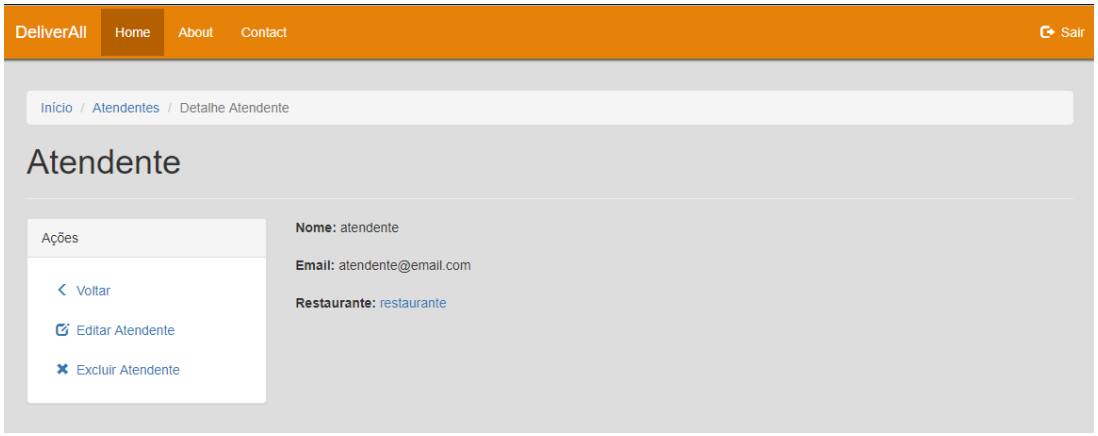

Figura 42: Detalhes do atendente

X. Excluir atendente:

A funcionalidade de excluir atendentes é similar à de excluir produtos. Basta clicar no ícone e confirmar a ação desejada.

8.3 Franqueado

I. Visualizar restaurante:

Os franqueados conseguirão visualizar uma lista em forma de tabela com todos os restaurantes relacionados com ele, como mostrado na Figura 43:

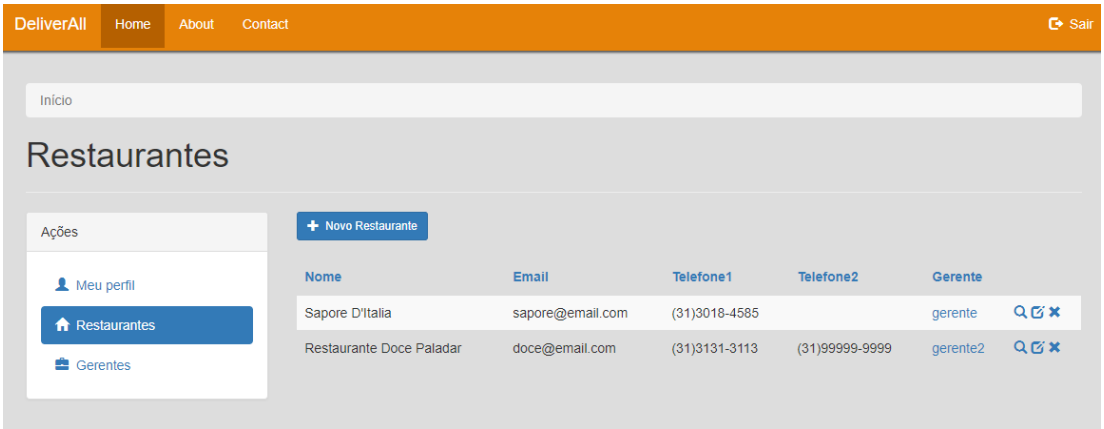

Figura 43: Tela com a lista de restaurantes

Esta lista permitirá, através dos botões no canto direito, que o franqueado navegue nas opções de cada restaurante, podendo visualizar detalhes, editar dados e excluir. A tela com os detalhes do restaurante mostrará todas as informações do restaurante em questão e assim o usuário conseguirá confirmar se os dados estão corretos. A Figura 44 mostra a tela de detalhes:

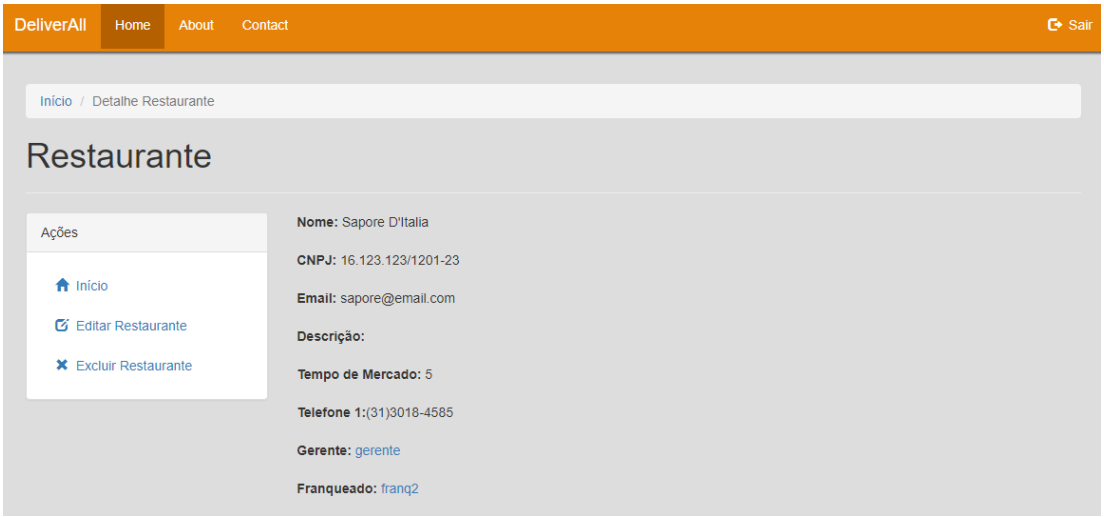

Figura 44: Detalhes do restaurante

II. Alterar restaurante:

A tela para alteração dos dados do restaurante será similar à tela de cadastrado, somente sua funcionalidade mudará, salvando os dados editados em vez de criar um novo registro.

III. Excluir restaurante:

A funcionalidade de excluir restaurantes é similar à de excluir produtos. Basta clicar no ícone e confirmar a ação desejada.

IV. Cadastrar, visualizar, alterar e excluir gerentes:

Todas estas funcionalidades serão desenvolvidas de maneira similar às páginas referentes aos atendentes citadas no tópico de gerência.

8.4 Administração

I. Alterar franqueado:

Tela similar à tela de cadastro.

II. Visualizar e excluir franqueados:

Funcionalidades similares aos itens citados no tópico de gerência, sobre os atendentes.

III. Cadastrar, alterar, visualizar e excluir restaurantes:

Funcionalidades similares aos itens citados no tópico sobre franqueados.

IV. Cadastrar, visualizar, alterar e excluir gerentes:

Funcionalidades similares aos itens citados no tópico sobre franqueados.

V. Visualizar todos os pedidos:

Esta funcionalidade será desenvolvida de maneira similar à funcionalidade citada nos tópicos sobre atendentes e gerentes.

9 Implementação - Aplicativo Móvel e Webservice

9.1 Autenticação

I. Logout:

Nesta funcionalidade o aplicativo utilizará o armazenamento interno do smartphone para apagar a variável que havia sido definida no momento do login (dado que o usuário confirme que deseja realmente fazer logout). Após isso, redirecionará o usuário à tela de login.

II. Recuperação de senha:

Esta funcionalidade é semelhante à recuperação de senha do sistema web, isto é, envia ao email do cliente uma senha aleatória, dado que esse email seja válido e realmente esteja cadastrado no banco de dados. A Figura 45 abaixo ilustra a tela de recuperação de senha:

Figura 45: Recuperar senha

III. Mostrar informações do perfil do usuário:

Ao passo que o cliente realiza login no aplicativo, um objeto do tipo cliente é criado para facilitar os trabalhos futuros da aplicação com os campos do cliente. Com isso, toda vez que o usuário navega entre as telas do sistema, o objeto contendo suas informações é enviado. Portanto, para exibir as informações de perfil do usuário, basta ler os dados deste objeto, tendo como resultado a tela ilustrada na Figura 46:

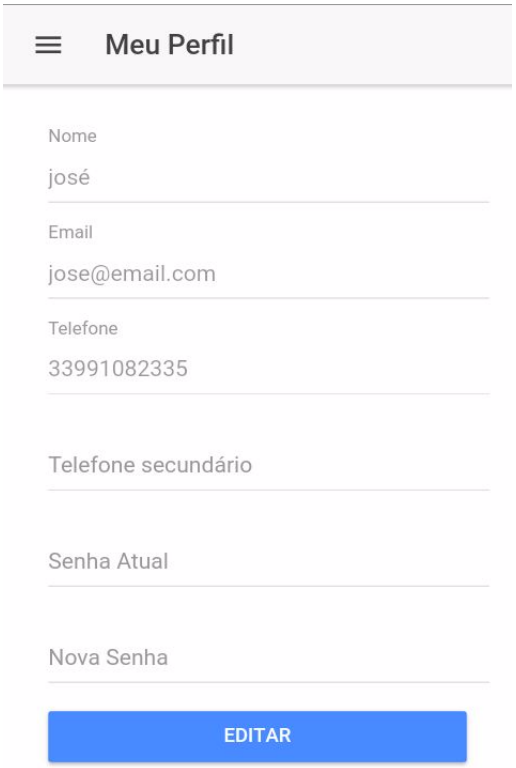

Figura 46: Visualizar dados de perfil

IV. Atualizar informações do perfil do usuário:

Esta funcionalidade é similar à anterior, porém, exibirá campos em modo de edição para que o cliente possa editar suas informações de cadastro. Após confirmação do cliente o sistema enviará uma requisição “POST” ao webservice com as novas informações para que sejam salvas no banco de dados. A Figura 47 mostra a estrutura da tela de edição de informações do perfil do usuário:

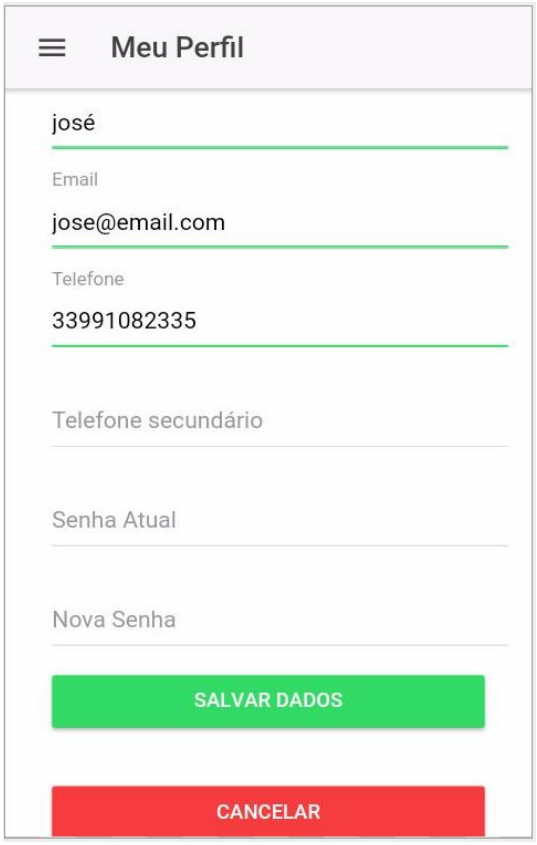

Figura 47: Atualizar dados de perfil

9.2 Encomenda

I. Ver todos os pedidos:

Para conseguir listar todos os pedidos de um cliente, a aplicação envia uma requisição “GET” ao webservice que, através do identificador do cliente, consegue buscar no banco de dados os seus respectivos pedidos e retorna-los ao aplicativo. Assim, o aplicativo poderá listar os pedidos, conforme mostrado na Figura 48:

Figura 48: Listar pedidos

II. Visualizar cardápio de restaurantes:

Esta funcionalidade é semelhante à funcionalidade que lista os pedidos de um determinado cliente, porém, esta listará os produtos de um restaurante específico, justamente com base no identificador do restaurante. A Figura 49 ilustra a tela do cardápio:

Figura 49: Cardápio de restaurantes

10 Testes

10.1 Autenticação no painel administrativo

Como um usuário,

Eu preciso me autenticar no sistema,

Para ter acesso às funcionalidades do painel administrativo web.

10.1.1 Atualizar dados de usuário

Cenário principal:

Dado que o usuário esteja logado no sistema,

Quando o usuário clicar em “Meu perfil”,

Então entra na tela “Meu perfil” e mostra os dados do usuário como nome e e-mail.

Quando o usuário clicar em “Editar dados”,

Então abre a tela de edição de dados do perfil,

Quando o usuário alterar os campos e clicar em salvar,

Então salva as alterações e mostra mensagem de sucesso.

10.2 Autenticação no aplicativo

Como um USUÁRIO,

Eu preciso me autenticar no sistema,

Para ter acesso às funcionalidades do aplicativo.

10.2.1 Logout

Cenário principal:

Dado que o usuário esteja logado no aplicativo,
Quando clicar no menu e clicar em “Sair”,
Então exibe um alerta: “Deseja realmente sair?”,
Quando clicar em “Sim”,
Então faz o logout do usuário e redireciona para tela de login.

Cenário alternativo (I - Desistiu de sair ou clicou por engano):

Dado que o usuário esteja logado no aplicativo,
Quando clicar no menu e clicar em “Sair”,
Então exibe alerta: “Deseja realmente sair?”,
Quando clicar em “Não”,
Então fecha o alerta e volta à tela que estava.

10.2.2 Recuperar senha

Cenário principal:

Dado que o usuário abra o aplicativo, não esteja logado e tenha esquecido a senha,
Quando o usuário clicar no botão “Esqueci a senha”,
Então redireciona para a tela “Recuperar senha”,
Quando preencher meu e-mail e clicar em enviar,
Então envia email com nova senha aleatória e exibe uma mensagem “Nova senha foi enviada para seu e-mail”.

Cenário alternativo (I - E-mail não cadastrado):

Dado que o usuário abra o aplicativo, não esteja logado e tenha esquecido a senha,
Quando o usuário clicar no botão “Esqueci a senha”,
Então redireciona para a tela “Recuperar senha”,
Quando preencher meu e-mail e clicar em enviar,
Então abre um alerta dizendo “e-mail não cadastrado, tente novamente”

10.2.3 Mostrar o perfil do usuário

Cenário principal:

Dado que o usuário abra o aplicativo e esteja logado,

Quando o usuário clicar no menu e clicar em “Meu perfil”,
Então entra na tela “Meu perfil” e mostra os dados do usuário como nome, e-mail e etc.

10.2.4 Atualizar dados de usuário

Cenário principal:

Dado que o usuário abra o aplicativo e esteja logado,
Quando o usuário clicar no menu e clicar em “Meu perfil”,
Então entra na tela “Meu perfil” e mostra os dados do usuário como nome, e-mail e etc,
Quando o usuário clicar em “Editar dados”,
Então abre a tela de edição de dados do perfil,
Quando o usuário alterar os campos e clicar em “Salvar”,
Então exibe mensagem de sucesso e retorna para tela principal.

10.3 Encomenda

Como um cliente,
Eu quero realizar um pedido,
Para receber comida em casa.

10.3.1 Ver todos os pedidos

Cenário principal:

Dado que o usuário esteja logado,
Quando o usuário clicar no menu -> Ver pedidos,
Então redireciona para tela de pedidos, listando todos os pedidos realizados.

10.3.2 Detalhes de pedidos

Cenário principal:

Dado que o usuário esteja logado,
Quando o usuário clicar no menu -> Ver pedidos,
Então redireciona para tela de Ver pedidos, listando todos os pedidos realizados,
Quando o usuário clicar em algum pedido,
Então redireciona para tela de Detalhes, mostrando todos os respectivos campos do pedido.

10.3.3 Visualizar informações de restaurantes

Cenário principal:

Dado que o usuário esteja na tela principal,
Quando o usuário clicar no restaurante desejado,

Então redireciona para tela do restaurante,
Quando o usuário clicar no botão “Sobre”,
Então mostra as informações do restaurante.

Cenário alternativo (I - Buscar restaurantes):
Dado que o usuário esteja na tela principal,
Quando o usuário clicar no ícone de buscar, digitar e clicar em buscar,
Então redireciona para tela da busca listando os restaurantes relacionados,
Quando o usuário selecionar um restaurante,
Então redireciona para tela do restaurante,
Quando o usuário clicar na aba “Sobre”,
Então mostra as informações do restaurante.

10.3.4 Visualizar cardápio de restaurantes

Cenário principal:
Dado que o usuário esteja na tela principal,
Quando o usuário clicar no restaurante desejado,
Então redireciona para tela do restaurante mostrando seu cardápio.

10.4 Gerência

Como um administrador,
Eu quero acessar o sistema,
Para visualizar relatórios e gráficos, classificações e comentários, obtendo controle sobre as vendas.

10.4.1 Alterar dados do restaurante

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar em “Meu restaurante”,
Então redireciona para a página contendo os dados do restaurante,
Quando o usuário editar os campos desejados e clicar em “Salvar”,
Então salvar os dados e redireciona à tela “Meu restaurante”.

10.4.2 Alterar produto

Cenário principal:
Dado que o usuário esteja na Home,

Quando o usuário clicar em “Produtos”,
Então abre a página de produtos listando todos cadastrados,
Quando o usuário clicar no ícone “Editar” no produto desejado,
Então abre uma página contendo os campos do produto em questão,
Quando alterar os campos e clicar em “Salvar”,
Então salva os dados e retorna à página “Produtos”.

10.4.3 Visualizar produtos

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Produtos”,
Então redireciona para a página Produtos listando todos cadastrados.

10.4.4 Excluir produtos

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Produtos”,
Então abre a página de produtos listando todos cadastrados,
Quando o usuário clicar no ícone “Excluir” no produto desejado,
Então abre alerta: “Deseja realmente excluir o produto?”,
Quando o usuário clicar em “Sim”,
Então apaga o produto do banco de dados e retorna à tela “Produtos”.

10.4.5 Cadastrar atendente

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Atendentes”,
Então abre a página de atendentes listando todos cadastrados,
Quando o usuário clicar em “Novo”,
Então abre a página de cadastro de atendente,
Quando o usuário informar (nome, email e senha) e clicar em “Salvar”,
Então cadastra novo atendente e retorna à tela “Atendentes”.

10.4.6 Alterar atendente

Cenário principal:
Dado que o usuário esteja na Home,

Quando o usuário clicar em “Atendentes”,
Então abre a página de atendentes listando todos cadastrados,
Quando o usuário clicar no ícone “Editar” no atendente desejado,
Então abre uma página contendo os campos do atendente em questão,
Quando alterar os campos e clicar em “Salvar”,
Então salva os dados e fica na página de Detalhes.

10.4.7 Visualizar atendente

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Atendentes”,
Então abre a página de atendentes listando todos cadastrados.

10.4.8 Excluir atendente

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Atendentes”,
Então abre a página de atendentes listando todos cadastrados,
Quando o usuário clicar no ícone “Excluir” no atendente desejado,
Então abre alerta: “Deseja realmente excluir o atendente?”,
Quando o usuário clicar em “Sim”,
Então apaga o atendente do banco de dados e retorna à tela “Atendentes”.

10.5 Franqueado

Como franqueado,
Eu quero acesso minha área restrita,
Para controlar restaurantes e gerentes.

10.5.1 Alterar restaurante

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar no ícone “Editar” no restaurante desejado,
Então abre uma página contendo os campos do restaurante em questão,
Quando alterar os campos e clicar em “Salvar”,
Então salva os dados e redireciona à página de detalhes do restaurante.

10.5.2 Visualizar restaurantes

Cenário principal:

Dado que o usuário esteja na logado,

Então já serei redirecionado à tela com a lista de todos os restaurantes.

10.5.3 Excluir restaurantes

Cenário principal:

Dado que o usuário esteja na logado,

Então já serei redirecionado à tela com a lista de todos os restaurantes.

Quando o usuário clicar no ícone “Excluir”,

Então abre alerta: “Deseja realmente excluir restaurante?”,

Quando o usuário clicar em “Sim”,

Então apaga o restaurante do banco de dados e retorna à tela “Restaurantes”.

10.5.4 Cadastrar gerente

Cenário principal:

Dado que o usuário esteja na Home,

Quando o usuário clicar em “Gerentes”,

Então abre a página de gerentes listando todos cadastrados,

Quando o usuário clicar em “Novo”,

Então abre a página de cadastro de Gerente,

Quando o usuário informar todos os campos necessários e clicar em “Salvar”,

Então cadastra novo gerente e retorna à tela “Gerentes”.

10.5.5 Alterar gerente

Cenário principal:

Dado que o usuário esteja na Home,

Quando o usuário clicar em “Gerentes”,

Então abre a página de gerentes listando todos cadastrados,

Quando o usuário clicar no ícone “Editar” no gerente desejado,

Então abre a página com os campos do gerente em modo de edição,

Quando alterar os campos e clicar em “Salvar”,

Então salva os dados e volta à página de Detalhes.

10.5.6 Visualizar gerente

Cenário principal:

Dado que o usuário esteja na Home,
Quando o usuário clicar em “Gerentes”,
Então abre a página de Gerentes listando todos cadastrados,
Quando o usuário clicar no ícone “Detalhes”,
Então abre a tela com todos os dados do gerente.

10.5.7 Excluir gerentes

Cenário principal:
Dado que o usuário esteja na Home,
Quando o usuário clicar em “Gerentes”,
Então abre a página de Gerentes listando todos cadastrados,
Quando o usuário clicar no ícone “Excluir”,
Então abre alerta: “Deseja realmente excluir o gerente?”,
Quando o usuário clicar em “Sim”,
Então apaga o gerente do banco de dados e retorna à tela “Gerentes”.

10.6 Administração

Como administrador,
Eu quero acessar minha área restrita,
Para gerenciar franqueados, restaurantes e controlar o sistema de forma geral.

10.6.1 Alterar franqueado

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar no ícone “Editar”,
Então abre página com os campos do franqueado em modo de edição,
Quando o usuário preencher os campos e clicar em “Salvar”,
Então salva os dados e fica na página de Detalhes.

10.6.2 Visualizar franqueado

Cenário principal:
Dado que o usuário esteja logado,
Quando o usuário clicar no ícone “Detalhes” no franqueado desejado,
Então abre uma página contendo os detalhes do restaurante franqueado em questão.

10.6.3 Excluir franqueado

Cenário principal:

Dado que o usuário esteja logado,

Quando o usuário clicar no ícone “Excluir” do franqueado desejado,

Então abre alerta: “Deseja realmente excluir franqueado?”,

Quando o usuário clicar em “Sim”,

Então apaga o franqueado do banco de dados e retorna à página “Home”.