

SOCIEDADE EDUCACIONAL DE SANTA CATARINA – SOCIESC
CENTRO UNIVERSITÁRIO SOCIESC - UNISOCIESC

MARCELO BEPLER JUNIOR
YURI LONGARAY CALDEIRA

SOFTWARE PARA COMANDA ELETRÔNICA COM FOCO EM SISTEMAS DE
RECOMENDAÇÃO DE PRODUTOS

JOINVILLE
2017/2

**MARCELO BEPLER JUNIOR
YURI LONGARAY CALDEIRA**

**SOFTWARE PARA COMANDA ELETRÔNICA COM FOCO EM SISTEMAS DE
RECOMENDAÇÃO DE PRODUTOS**

**Trabalho de Conclusão de Curso apresentado
ao Centro Universitário SOCIESC como
requisito parcial para a obtenção do título de
Bacharel em Sistema de informação.**

Guilherme Defreitas Juraszek

**JOINVILLE
2017/2**

MARCELO BEPLER JUNIOR

YURI LONGARAY CALDEIRA

**DESENVOLVIMENTO DE UM SISTEMA DE COMANDA ELETRÔNICA COM O
FOCO EM RECOMENDAÇÃO DE PRODUTOS**

Este trabalho foi conferido e aprovado pela
Banca Examinadora do Centro Universitário
SOCIESC, dando o título de Bacharel em
Sistema de informação.

Joinville, ____ de _____ de _____

Prof. (orientador)

Prof. (membro da banca)

(membro da banca)

RESUMO

Por muitos anos, estabelecimentos comerciais possuem uma forma arcaica de atendimento ao cliente onde todos são tratados da mesma maneira, sem possuir conhecimento do perfil de cada cliente.

O objetivo deste trabalho é o desenvolvimento de uma aplicação com suporte tanto para plataforma web quanto para mobile que, com o auxílio de um sistema de recomendação, efetue sugestão de produtos conforme os perfis dos usuários. Por meio da utilização deste projeto, o estabelecimento passará a atender seus clientes de maneira centralizada de forma a evitar problemas que são vivenciados com a utilização de comandas impressas.

Os conceitos utilizados abordam temas como análise de perfis e sistemas de recomendação. Foi realizado uma pesquisa por meio de um questionário para 37 (trinta e sete) usuários a fim de verificar seus gostos e preferências. Com isso, foi possível a sugestão de produtos conforme o perfil do usuário. O trabalho propõe uma saída inovadora para melhor atendimento ao cliente e aumento da lucratividade.

Palavras-chave: Inteligência Artificial. Qualidade de Atendimento. Mobilidade. Sistemas de Recomendação.

ABSTRACT

For many years, commercial establishments have an archaic method for customer service, in which they communicate with all the clients the same way, not considering that each personality has a product that fits better with it.

The objective of this work is to develop an application for support on a web platform and mobile, focusing on a system that is capable to making suggestions of products based on the customer's profile. Through this project, the establishment is able to treat their clients in a smarter and more centralized way, by creating opportunities that a paper card can not offer.

The concepts used approach themes like profile analysis, recommendation systems and mobile and web applications. To collect data, It was requested to customers of some city popular establishments to fill a form with their opinion about the food of two choosable places and three questions about themselves. After that, it was possible to suggest products based on what their profile had indicated. The main object of this work was to improve customer service in an innovative way, raising profits at the same time.

Keywords: Artificial intelligence. Quality of Service. Mobility. Recommendation Systems.

LISTA DE ILUSTRAÇÕES

FIGURA 1: ESTRUTURA ANDROID	16
FIGURA 2: RELAÇÃO DE NAVEGADORES X SISTEMAS OPERACIONAIS SUPORTADOS.....	21
FIGURA 3: DIVISÃO DE COLUNAS NO BOOTSTRAP	21
FIGURA 4: ESTRUTURA DO CÓDIGO EM REACT	25
FIGURA 5: COMPONENTES REACT X REACT NATIVE	26
FIGURA 6: ÁRVORE DE OBJETOS REDUX.....	27
FIGURA 7: ARQUITETURA REST CLIENTE-SERVIDOR	30
FIGURA 8: SISTEMA DE RECOMENDAÇÃO DA AMAZON	34
FIGURA 9: ARQUITETURA MOTOR RECOMMENDER.....	41
FIGURA 10: ALGORITMO DE SEMELHANÇA COLABORATIVA	44
FIGURA 11: FORMULÁRIO DE LOGIN DO ATENDENTE.....	48
FIGURA 12: FORMULÁRIO DE REGISTRO.....	48
FIGURA 13: LISTAGEM DE PEDIDOS	49
FIGURA 14: ALTERAÇÃO DO STATUS DO PEDIDO.....	49
FIGURA 15: ITENS DO MENU	50
FIGURA 16: CADASTRO DO ITEM DE MENU.....	50
FIGURA 17: CADASTRO DE CLIENTE.....	51
FIGURA 18: LISTAGEM DE ESTABELECIMENTOS	52
FIGURA 19: AVALIAÇÃO DO ITEM	53
FIGURA 20: PERGUNTAS AO CLIENTE	54
FIGURA 21: MENU DO ESTABELECIMENTO	54
FIGURA 22: DIAGRAMA DE CASOS DE USO	55
FIGURA 23: DIAGRAMA DE CLASSES.....	56
FIGURA 24: MODELO ENTIDADE RELACIONAMENTO	57
FIGURA 25: PERGUNTAS PRELIMINARES	58
FIGURA 26: SEXO DOS USUÁRIOS	59
FIGURA 27: FAIXA ETÁRIA.....	59
FIGURA 28: ESTABELECIMENTOS AVALIADOS	60
FIGURA 29: RELAÇÃO DE ALGORITMOS COM/SEM PERGUNTAS PRELIMINARES.....	62
FIGURA 30: MÉDIA E DESVIO PADRÃO COM PERGUNTAS PRELIMINARES E AVALIATIVAS ..	62

LISTA DE TABELAS

TABELA 1: NOTAS DE USUÁRIOS	35
TABELA 2: SIMILARIDADES ENTRE USUÁRIOS	36
TABELA 3: SIMILARIDADE COM DO COSSENO	38
TABELA 4: MÉDIA E DESVIO PADRÃO SEM PERGUNTAS PRELIMINARES	61
TABELA 5: MÉDIA E DESVIO PADRÃO COM PERGUNTAS PRELIMINARES	61
TABELA 6: RELAÇÃO DE SOFTWARES X FUNCIONALIDADES	64

LISTA DE EQUAÇÕES

EQUAÇÃO 1: DISTÂNCIA DE EUCLIDEAN.....	35
EQUAÇÃO 2: PREVISÃO DE NOTA.....	36
EQUAÇÃO 3: SIMILARIDADE DO COSSENO.....	38
EQUAÇÃO 4: PREVISÃO DE SIMILARIDADE	38
EQUAÇÃO 5: ERRO MÉDIO ABSOLUTO	43
EQUAÇÃO 6: SEMELHANÇA COLABORATIVA	45

LISTA DE ABREVIATURA E SIGLAS

API – Application Programming Interface ou Interface de Aplicação Programada;

APK – Pacote de Arquivos Android;

CSS – Cascading Style Sheets;

DOM – Document Object Model;

HTML – HyperText Markup Language;

HTTP – HyperText Transfer Protocol;

IA – Inteligence Artificial ou Inteligencia Artificial;

IAAS – Infrastructure as a Service ou Infraestrutura como Serviço;

IPC – Comunicação Inter Processos;

JSON – Javascript Object Notation.

MAE – Mean Absolute Error;

OSI – Iniciativa de código aberto;

PAAS – Plataform as a Service ou Plataforma como Serviço;

PCS – Pearson Collaborative Similarity;

REST – Representational State Transfer;

REST – Representational State Transfer;

SAAS – Software as a Service ou Software como Serviço;

SOAP – Simple Object Access Protocol;

T.I – Tecnologia da Informação;

URI – Uniform resource identifier;

URL – Uniform Resource Locator;

XML – Extensible Markup Language.

SUMÁRIO

1 INTRODUÇÃO	12
2 TECNOLOGIAS	14
2.1 COMPUTAÇÃO EM NUVEM	14
2.2 PLATAFORMA ANDROID	16
2.2.1 Camada de aplicação	17
2.2.2 Camada de framework	17
2.2.3 Camada de bibliotecas	18
2.2.4 Camada de runtime	19
2.2.5 Camada de kernel	19
2.2.6 Recursos	19
2.3 LINGUAGEM HTML	20
2.4 FRAMEWORK BOOTSTRAP	21
2.5 LINGUAGEM PHP E MYSQL	22
2.6 LINGUAGEM JAVASCRIPT	23
2.7 LINGUAGEM REACT	25
2.7.1 Versão React Native	26
2.7.2 Arquitetura Redux	27
2.8 LINGUAGEM JAVA	28
2.9 ARQUITETURA REST	29
2.10 CÓDIGO QR CODE	31
2.11 SISTEMAS DE RECOMENDAÇÃO	32
2.11.1 Sistema de recomendação baseado em perfil mais próximo	33
2.11.2 Sistema de recomendação baseada em filtragem colaborativa por item	37
2.11.3 Estimação Bayesiana	39
2.12 INTELIGÊNCIA ARTIFICIAL	40
2.13 APACHE MAHOUT	41
2.13.1 Fórmula de erro médio absoluto (MAE)	42

2.13.2 Iniciando com o Apache Mahout	43
2.13.3 Semelhança colaborativa de usuário com Apache Mahout	44
2.13.4 Limite de vizinhança de usuário	45
3 DESENVOLVIMENTO	47
3.1 METODOLOGIA	47
3.2 DESENVOLVIMENTO DA APLICAÇÃO WEB	47
3.3 DESENVOLVIMENTO DA APLICAÇÃO MOBILE	51
3.2 DESENVOLVIMENTO DO SISTEMA DE RECOMENDAÇÃO	52
3.3 MODELAGEM DO SISTEMA	55
3.4 RESULTADOS ALCANÇADOS	57
4 CONCLUSÃO	65
5 REFERÊNCIAS	67
6 APÊNDICE	72

1 INTRODUÇÃO

A utilização de comandas impressas e, sem o auxílio da tecnologia da informação, permanece dominante em muitos estabelecimentos. Esta forma de atendimento, ineficiente, resulta em diversas falhas de processos e, muitas vezes, em problemas no faturamento. Geralmente, a quantidade monetária para investimento em uma solução se apresenta menor que a quantidade utilizada para agir perante a ocorrência destes problemas.

Para que um estabelecimento mantenha, ou até mesmo expanda, sua produtividade, o mesmo deve ser competitivo com relação aos demais estabelecimentos presentes no mercado. Buscar tanto a inovação quanto a agilidade de processos, deve ser uma tarefa presente no dia-a-dia de um estabelecimento que resultará em um crescimento constante.

O objetivo principal do trabalho, é a sugestão de produtos que são apresentados em formato destacado de acordo com as análises efetuadas no perfil do cliente. Outros objetivos que serão atendidos com o desenvolvimento do projeto são:

- a) evitar perdas de comandas: as comandas serão totalmente virtuais de forma que o cliente não pode perder pois sempre existirá registros;
- b) receber feedbacks através de pesquisas de satisfação: o cliente pode responder a uma pesquisa de satisfação onde é possível selecionar uma nota de um a cinco;
- c) agilizar o atendimento com o uso de um menu na aplicação mobile: o sistema é ágil e disponibiliza o menu virtualmente, desta forma, o cliente solicita pelo smartphone e dispensa filas de atendimento;
- d) evitar fraudes e adulterações nas comandas: o sistema possui validação de campos entre outras verificações para evitar fraudes e adulterações de registros;
- e) reunir valores para o estabelecimento: a imagem do estabelecimento passa a ser mais amigável, uma vez que o mesmo demonstra preocupação com relação ao gosto do cliente e busca inovação para agilizar seus processos.

Para cumprimento dos objetivos foram desenvolvidas uma aplicação mobile e uma aplicação servidora, onde ambas utilizam um sistema de recomendação baseado

em perfil mais próximo com os algoritmos de distância de Euclidean e semelhança colaborativa de usuário (PCS).

Como metodologia, foi considerado a pesquisa exploratória e aplicada onde, a pesquisa exploratória proporcionou a identificação das necessidades existentes de um processo ágil para estabelecimentos noturnos.

Este trabalho está dividido em 6 capítulos onde, o primeiro capítulo é a introdução, o segundo são as tecnologias estudadas, o terceiro é o desenvolvimento de trabalho (com a utilização das tecnologias estudadas), o quarto é a conclusão do trabalho, o quinto são as referências bibliográficas e o sexto são os apêndices.

2 TECNOLOGIAS

A pesquisa proposta aqui propõe o desenvolvimento de uma aplicação a fim de demonstrar o uso e a resolução do problema encontrado. Para o bom desenvolvimento desta aplicação fez-se necessário o estudo de diferentes tecnologias e suas aplicabilidades. O referido estudo está descrito neste capítulo.

2.1 COMPUTAÇÃO EM NUVEM

Cloud Computing, ou computação em nuvem, é um mecanismo que permite ao usuário final o acesso, independentemente da plataforma, a uma grande quantidade de aplicações e serviços de qualquer lugar por meio de um terminal com acesso à internet (PEDROSA; NOGUEIRA, 2011).

O modelo de computação em nuvem ganhou espaço na área empresarial e, denomina-se nuvem, a camada conceitual que abstrai toda infraestrutura computacional, torna os serviços transparentes aos usuários e reduz consideravelmente determinados custos e preocupações em comparação com um serviço não categorizado como *Cloud* (OLIVEIRA; PEREIRA, 2015).

Segundo (OLIVEIRA; PEREIRA, 2015), a computação em nuvem consiste em armazenar dados, softwares e hardwares sem a necessidade de instalação de qualquer tipo de programa. Com os grandes avanços tecnológicos e a grande disponibilidade da internet, muitas empresas no mercado começaram a adotar este mecanismo. Existem plataformas que são disponibilizadas como serviços, são elas:

- a) software como serviço (SaaS) possui a responsabilidade de disponibilizar aplicações ao usuário final e é considerada a camada mais alta da arquitetura da computação em nuvem, abaixo dela estão: Infraestrutura como Serviço (IaaS) e Plataforma como Serviço (PaaS). A camada de SaaS fornece acesso através de portais web, sendo completamente transparente ao usuário e para oferecer esta transparência, a camada de software como serviço utiliza-se das outras duas camadas inferiores.
- b) infraestrutura como serviço (IaaS), onde são oferecidos serviços referentes a hardware, ou seja, recursos de virtualizados como armazenamento,

comunicação e computação. Esta classe de serviços prove servidores para execução de aplicações customizadas e, dispões de uma interface única para a administração destes servidores e hardwares comunicados.

- c) plataforma como serviço (PaaS), que diz respeito a camada intermediária onde o serviço oferecido é um ambiente utilizado por desenvolvedores. Com este serviço é possível implementar aplicações sem preocupações com processadores e memória a ser utilizada. São fornecidos hardwares com alto nível de integração e compatibilidade com diversas linguagens de programação, sistemas operacionais e ambientes de desenvolvimento.

Na computação em nuvem, o conceito de elasticidade, proporciona a ilusão de recursos computacionais infinitos disponíveis para utilização, uma vez que é efetuado, de forma automática, o aumento e consolidação de recursos de acordo com quantidade demandada (PEDROSA; NOGUEIRA, 2011).

Segundo (OLIVEIRA; PEREIRA, 2015), devido ao fato de que toda parte de hardware e software está por conta da empresa que disponibiliza o serviço, as empresas não precisam investir tanto em hardware e software e, com isso, a parte de T.I da empresa pode focar em outros serviços uma vez que o controle é efetuado pelo T.I da empresa contratada.

Segundo (PEDROSA; NOGUEIRA, 2011), quando o assunto é preço, o serviço de computação em nuvem é pago somente pelo uso, desta forma, a organização ou empresa, evita desperdícios de recursos e amplia os recursos conforme sua necessidade.

A partir do conhecimento das características da computação em nuvem os riscos relacionados à infraestrutura são minimizados e a organização não se prenderá aos recursos físicos (PEDROSA; NOGUEIRA, 2011).

Com o serviço de computação em nuvem, as informações não são mais armazenadas. Privacidade e integridade das informações são itens extremamente importantes e que podem ser expostos a ataques com maiores chances se a nuvem for pública. Para evitar riscos e ataques, as empresas adotam mecanismos de segurança como de criptografia de dados, controles rigorosos de acessos e sistemas de cópias de segurança (PEDROSA; NOGUEIRA, 2011).

Segundo (PEDROSA; NOGUEIRA, 2011), escalabilidade que é a capacidade de aumentar o tamanho do software ou do seu uso e é fundamental para um serviço de nuvem. Os recursos utilizados podem ser alterados conforme a demanda exigida

pela aplicação para então ocorrer a estabilização do serviço e evitar possíveis latências.

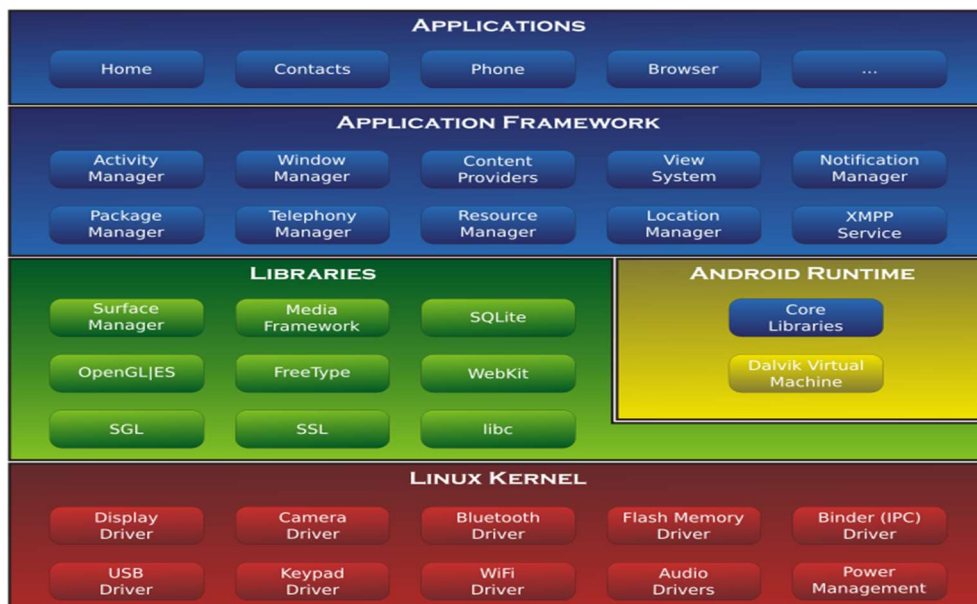
Atualmente existe a necessidade de implementação de padrões para possibilitar a portabilidade entre os usuários e a nuvem, ou seja, os usuários podem acessar seus dados em nuvens diferentes (PEDROSA; NOGUEIRA, 2011).

Segundo (PEDROSA; NOGUEIRA, 2011), um serviço de nuvem deve ser confiável e possuir uma arquitetura tolerante a falhas. A confiabilidade é um termo essencial para que os dados permaneçam intactos mesmo após a ocorrência de falhas e erros.

2.2 PLATAFORMA ANDROID

Baseada no sistema operacional Linux, a plataforma Android além de possuir diversos componentes também dispõe de uma variada disponibilidade de bibliotecas, interfaces gráficas e ferramentas para seus aplicativos (SIMÕES; PEREIRA, 2014).

Figura 1: Estrutura Android



Fonte: Pereira (2009)

Segundo (SIMÕES; PEREIRA, 2014), a iniciativa do Google com o projeto Android era de disponibilizar uma plataforma aberta aos fabricantes, tornando o

sistema flexível e atualizável. Para isso, conforme apresentado na Figura 1, foi desenvolvido uma arquitetura dividida em cinco camadas, sendo elas:

- a) *Applications*: camada de aplicação;
- b) *Application Framework*: camada de framework;
- c) *Android Runtime*: camada de máquinas virtuais;
- d) *Libraries*: camada de bibliotecas;
- e) *Kernel*: núcleo do sistema operacional.

Com o conhecimento da arquitetura do sistema operacional Android, é possível concluir que o mesmo se trata de uma pilha de software com base em Linux e, criado com o propósito de operar em diversos dispositivos (SIMÕES; PEREIRA, 2014).

2.2.1 Camada de aplicação

A camada de aplicação (*Applications*), que está na primeira camada da Figura 1, agrupa todos os aplicativos fundamentais (programados em Java) do Android, exemplos: mapas, navegadores, programas de SMS, calendários, clientes de e-mails entre outros aplicativos que serão desenvolvidos pela comunidade (PEREIRA; SILVA, 2009).

2.2.2 Camada de framework

Na camada de Framework (*Application Framework*), estão localizadas as APIs (*Application Programming Interface*) e demais recursos utilizados pelos aplicativos. Nesta camada atuam os elementos:

- a) *Active Manager*, mecanismo que gerencia o ciclo de vida de todas as atividades, mais precisamente quando iniciam e quando terminam;
- b) *Package Manager* mecanismo que efetua a leitura dos Pacotes de arquivos Android (APKs);
- c) *Window Manager*, mecanismo que gerencia as apresentações de janelas;
- d) *Content Providers* mecanismos que possibilitam o compartilhamento de dados entre aparelhos;

- e) *View System* mecanismo que disponibiliza todo o tratamento gráfico para a aplicação.

Não menos importantes, e ainda na camada de Framework, se encontram outros elementos como o *Location Service*, *Bluetooth Service*, *Wi-Fi Service*, *USB Service* e *Sensor Service* (PEREIRA; SILVA, 2009).

2.2.3 Camada de bibliotecas

Segundo (PEREIRA; SILVA, 2009) o Android além de possuir um conjunto de bibliotecas C/C++, também possui bibliotecas de multimídia, visualização de camadas 2D e 3D, funções para gráficos, funções para navegadores Web, funções para aceleração de hardware, renderização 3D, fontes *bitmap* e vetorizadas e funções de acesso a banco de dados SQLite. As principais bibliotecas disponíveis são:

- a) *Freetype*: utilizada para renderização de fontes e bitmaps;
- b) *System C library*: customizada e otimizada para uso embutido no Android;
- c) *Webkit*: renderizador de páginas para navegadores, com suporte para CSS e JavaScript;
- d) *SQLite*: banco de dados relacional implementado em C, leve e embutido, com suporte à base de dados acima de 2 terabytes. Esta biblioteca implementa a maioria do SQL92;
- e) *SGL*: responsável pelos gráficos 2D;
- f) *Surface Manager*: acesso para as camadas de aplicações 2D e 3D, os subsistemas de exibição;
- g) *Media Libraries*: suportam os mais populares formatos de áudio, vídeo, e também, hardware e software de *plugins* da codec;
- h) *LibWebCore*: um *engine* de web browser utilizado tanto no Android Browser quanto para exibições web;
- i) *3D libraries*: utilizam aceleração 3D via hardware ou software de renderização 3D altamente otimizado.

Todos os recursos, das bibliotecas, estão disponibilizados na camada de framework, para desenvolvimento (PEREIRA; SILVA, 2009).

2.2.4 Camada de runtime

Bibliotecas chamadas de *runtime*, automatizam tarefas comuns realizadas por linguagens diferentes de programação. Com a utilização do tempo de execução é possível fornecer eficiência de modo a reduzir tanto o número de passos necessários para a execução de programas quanto de determinados recursos como o espaçamento de disco, de memória e a utilização de CPU. Este tempo de execução consiste de funções, variáveis, constantes comuns e operações, desta forma é possível efetuar cálculos, vídeos, imagens, áudio, texto e até mesmo outras bibliotecas em tempo de execução (SIMÕES; PEREIRA, 2014).

2.2.5 Camada de kernel

O Kernel, que é responsável pela comunicação entre o hardware e software, é considerado o núcleo do sistema operacional e, tem como objetivo gerenciar o computador e permitir a execução de aplicativos com o uso dos recursos existentes (SIMÕES; PEREIRA, 2014)

O Android utiliza a versão 2.6 do Kernel do Linux para serviços centrais do sistema, tais como segurança, gestão de memória, de processos, pilha de protocolos de rede e modelo de drivers. Nesta camada atua o Binder (IPC), responsável por obter e enviar interfaces de serviços de uma aplicação para outra criando assim a comunicação *Inter Processos* (IPC) (PEREIRA; SILVA, 2009).

2.2.6 Recursos

Segundo (PEREIRA; SILVA, 2009), o Android possui diversos recursos que o tornam um sistema operacional completo, estes recursos estão presentes nas camadas de sua arquitetura e seus objetivos atendem diversas demandas, são eles:

- a) framework de aplicação: torna possível a incorporação, reutilização e substituição de recursos;
- b) máquina virtual *Dalvik*: máquina virtual otimizada para smartphones;
- c) navegador web integrado: navegador baseado no *engine open source*;

- d) gráficos otimizados: gráficos e tratamento de imagens baseados na especificação *OpenGL ES 1.0*;
- e) *SQLite*: gerenciador de banco de dados;
- f) suporte multimídia: suporte para determinados formatos de som (MP3, AAC, AMR), vídeo (MPEG4 e H.264) e imagens (JPG, PNG e GIF);
- g) telefonia GSM: fácil integração com tecnologias GSM;
- h) protocolos de comunicação *Wireless*: com tecnologia *Bluetooth* EDGE, 3G e Wi-Fi;
- i) integração com câmera, GPS, bússola e acelerômetro;
- j) ambiente de desenvolvimento: ferramenta de depuração, emulador de dispositivo, analisador de memória e desempenho, e *plugin* para IDE *Eclipse* (ADT).

É possível modificar as APIs do Android por meio da programação de seu conteúdo para tornar possível a otimização das APIs básicas e focar na utilidade dos pacotes. É possível implementar controles de fluxo de pacotes para aumentar o aproveitamento e remover pacotes pesados e indesejáveis. Em determinados APIs, o administrador possui privilégios de criar interfaces para limitar a visualização e restringir acesso do usuário final (PEREIRA; SILVA, 2009).

2.3 LINGUAGEM HTML

Desenvolvida em 1992, por Tim Berners e Robert Caillau o HTML (*Hypertext Markup Language*) é a linguagem padrão utilizada para o acesso e exibição de páginas Web onde as linhas de código são interpretadas pelo browser e então apresentadas como resultado. Constituído de textos e códigos denominados *tags*, o HTML não precisa de um compilador para obtenção de resultados, apenas um editor de textos e um browser são o suficiente (COSTA, 2007).

Segundo (COSTA, 2007), com os formulários HTML é possível efetuar a troca de dados ou informações entre o utilizador e o servidor. Iniciado pela *tag* `<form>` e finalizado pela *tag* `</form>`, os campos de entrada de dados (*input*, *textarea* e *select*), caso possuam um atributo *name* associado aos mesmos, serão recebidos pelo método (*method*) especificado, podendo ser tanto *get*, onde os dados fazem parte da URL enviada para o servidor (limitados a 128 caracteres), quanto *post* que, além

de possibilitar grandes transferências de dados os mesmos não são apresentados na URL.

2.4 FRAMEWORK BOOTSTRAP

Desenvolvido pela Twitter e de código aberto, o *framework* Bootstrap foi criado para auxiliar na criação de aplicações Web responsivas que, segundo (FREITAS, 2014) uma aplicação Web responsiva se refere a uma técnica de estruturação (HTML e CSS) que visa a adaptação de páginas Web em diversos dispositivos e resoluções. Na Figura 2 são apresentados os navegadores e, os sistemas operacionais, que suportam a tecnologia Bootstrap.

Figura 2: Relação de Navegadores X Sistemas Operacionais suportados

	Chrome	Firefox	Internet Explorer	Opera	Safari
Android	✓	✗	N/A	✗	N/A
iOS	✓	N/A		✗	✓
Mac OS X	✓	✓		✓	✓
Windows	✓	✓		✓	✗

Fonte: Site oficial do Bootstrap (2014)

O *framework* Bootstrap opera por meio da utilização de fontes JavaScript e CSS, o mesmo utiliza uma determinada técnica para separar o layout da tela em doze colunas que podem ser mescladas ou até mesmo combinadas conforme a necessidade do desenvolvedor. Na Figura 3 é apresentado a divisão de colunas e exemplos de códigos a serem utilizados fornecer o design responsivo (FREITAS, 2014).

Figura 3: Divisão de Colunas no Bootstrap

.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1	.col-md-1
.col-md-8									.col-md-4			
.col-md-4				.col-md-4				.col-md-4				
.col-md-6						.col-md-6						

Fonte: Freitas (2014)

Segundo (FREITAS, 2014), a utilização do Bootstrap em navegadores convencionais (como Chrome, Safari, IE, Firefox e Opera) proporciona diversas vantagens, como a documentação simples e ampla, o código aberto, diversidade de componentes à disposição, estrutura padronizada e, a mais importante, a operação em layouts responsivos.

2.5 LINGUAGEM PHP E MYSQL

Muitos usuários se questionam na hora de escolher entre as diversas tecnologias existentes então é necessário se perguntar “Por que utilizar o PHP juntamente com MySQL”, segundo (BENTO, 2013), algumas vantagens da dupla:

- a) são ágeis para o desenvolvimento de aplicações Web dinâmicas;
- b) são tecnologias livres;
- c) muitos serviços de hospedagem oferecem suporte a PHP e MySQL;
- d) são linguagens simples de serem implementadas;
- e) possuem uma curva de aprendizado consideravelmente suave, quando comparadas com outras tecnologias.

A dupla PHP e MySQL apresenta ao usuário uma forma mais fácil, mais flexível e também, mais prática quando o objetivo é o desenvolvimento de uma aplicação Web dinâmica.

PHP é uma das linguagens de programação mais utilizadas em aplicações Web e, além de possibilitar o pré-processamento de dados de páginas HTML alterando o conteúdo de uma página antes de enviar a mesma para o navegador, também permite a captura de dados dos elementos HTML como, por exemplo, dados de formulários preenchidos pelo usuário (BENTO, 2013).

As principais características da linguagem PHP, quando utilizada para pré-processamento de dados em aplicações Web, segundo (NIEDERAUER, 2008), são:

- a) gratuito e de código aberto: Além de ser um software de código aberto o seu conteúdo pode ser obtido no site www.php.net;
- b) embutido no HTML: as linguagens HTML e PHP podem ser mescladas, ou seja, é possível a inserção de código PHP seguido de código HTML;

c) baseado no Servidor: todo código é executado no servidor e somente seu resultado final é apresentado ao usuário;

d) banco de dados: diversos bancos de dados suportam a linguagem PHP e o próprio PHP possui funções específicas de cada banco de dados desde suportem o formato SQL;

e) portabilidade: é possível executar PHP em Linux, Unix e Windows NT.

MySQL é um tipo de SGBD (Sistema Gerenciador de Banco de Dados) relacional, que armazena as informações em estruturas no formato de tabelas e utiliza dados do tipo SQL (*Structured Query Language*). Além de ser mais o popular entre os bancos de dados e um dos mais utilizados em aplicações Web, também possui o custo extremamente baixo, mesmo sendo uma tecnologia complexa. (NIEDERAUER, 2008)

Algumas vantagens providas da utilização de um SGBD (Sistema Gerenciador de Banco de Dados) do tipo MySQL, segundo (NIEDERAUER, 2008), são:

a) número ilimitado de acessos simultâneos;

b) grande capacidade de manipulação de tabelas (mais de 50.000.000 registros);

c) alta velocidade no quesito de execução de comandos e de velocidade de acesso no qual destaca o MySQL dos demais tipos de bancos de dados;

d) controle de privilégios fácil e eficiente.

2.6 LINGUAGEM JAVASCRIPT

Segundo (COSTA, 2007) JavaScript é uma linguagem de programação Web originalmente criada pela Netscape que permite a criação de páginas interativas. Mesmo sendo uma linguagem baseada em objetos, não é possível a criação de classes nem a existência de herança e polimorfismo pois sua estrutura já possui uma hierarquia pré-definida (os objetos mais abaixo são propriedades daqueles localizados mais acima). Quando o assunto é a inserção de códigos do tipo JavaScript, existem três alternativas funcionais:

a) no corpo da página HTML;

b) no cabeçalho da página HTML, mais precisamente, como função;

c) armazenado em um ficheiro com extensão (sufixo) ".js" que será chamado na página HTML.

Segundo (COSTA, 2007), para utilização de JavaScript é necessário a utilização das tags `<script>` e `</script>` que delimitam o do código. Já os eventos, apresentados a partir de ações realizadas pelo utilizador, estão associados a event-handlers, que são scripts executados quando determinados eventos ocorrem, são eles:

- a) *onClick*: executado no momento de clique em determinado elemento;
- b) *onChange*: executado após a alteração de valores de determinado elemento;
- c) *onBlur*: executado após o utilizador retirar o foco do determinado elemento;
- d) *onFocus*: executado no momento em que o elemento for focado;
- e) *onLoad*: executado a partir do carregamento da página do browser;
- f) *onMouseOver*: executado quando o utilizador posiciona o ponteiro do mouse sobre determinado elemento;
- g) *onSelect*: executado quando selecionamos um valor de um elemento input;
- h) *onSubmit*: executado quando é enviado um formulário;
- i) *onUnload*: executado quando é abandonado a página.

Com o passar dos anos, o JavaScript tem se tornado uma linguagem muito mais robusta e eficiente, deixando de ser apenas uma linguagem de script e atendendo a demanda por recursos para desenvolvimento de softwares em grande escala. Uma curiosidade, que acaba confundindo muitos usuários, seria que, apesar da semelhança sintética superficial, o JavaScript é completamente diferente da linguagem de programação Java (FLANAGAN, 2013).

Conhecida popularmente entre os desenvolvedores por ser uma biblioteca JavaScript simples e fácil de utilização, com o JQuery é possível manipular os elementos das páginas HTML, ou seja, tanto adicionar, quanto modificar atributos HTML e propriedades de estilo. Outro grande benefício desta biblioteca é o suporte à tecnologia AJAX, que possibilita o envio de requisições HTTP com informações em diversos formatos (CHERUBIN, 2011).

Segundo (SILVA, 2012), AJAX (*Asynchronous JavaScript and XML*) é uma técnica para carregamento dos conteúdos de páginas Web a partir da utilização de JavaScript e XML. Criado pelos desenvolvedores do cliente de e-mail Web Access 2000, da Microsoft, com a finalidade de permitir a troca de informações entre o cliente de e-mail e seu respectivo servidor.

A tecnologia AJAX tem como responsável por seu funcionamento o objeto XMLHttpRequest, que se trata de requisições XML com utilização do protocolo HTTP

para transferência de dados. Embora este objeto XMLHttpRequest não faça parte do DOM (*Document Object Model*), o mesmo pode ser instanciado em uma simples linha de código JavaScript (SILVA, 2012).

2.7 LINGUAGEM REACT

Criada em 2013, a React é uma biblioteca JavaScript, do próprio Facebook, para interfaces de aplicações Web. Não só criada pelo Facebook mas também utilizada para o mesmo e também presente em outras aplicações de larga escala como AirBnb, Instagram e Twitter (LOPES, 2017).

Segundo (LOPES, 2017) A React atua na camada View do modelo de MVC e, proporciona, ao desenvolvedor, a tomada de decisões em relação ao restante da arquitetura da aplicação, em outras palavras, o *Model* e o *Controller* da aplicação.

Por meio de interfaces interativas, é possível o desenvolvimento de *Views* que são renderizadas conforme a demanda da aplicação. Seu código é estruturado de forma declarativa que resulta numa maneira mais fácil e mais previsível na hora de debugar (LOPES, 2017).

Segundo (LOPES, 2017), a lógica dos componentes é escrita, completamente, em JavaScript no qual permite que o desenvolvedor mantenha o estado da aplicação separado do modelo de objetos (DOM), conforme apresentado na Figura 4.

Figura 4: Estrutura do código em React

```
1 import React from 'react';
2 import Button from './Button'
3
4 const GroupButton = ({ children }) => (
5   <div className='btn-group'>
6     {children}
7   </div>
8 );
9
10 ReactDOM.render(
11   <GroupButton>
12     <Button>Left</Button>
13     <Button>Middle</Button>
14     <Button>Right</Button>
15   </GroupButton>
16 ,
17   document.getElementById('root')
18 );
```

Fonte: Lopes (2017)

Segundo (LOPES, 2017), a React se apresenta como uma escolha para desenvolvimento de aplicações robustas nos quais dependem tanto de manutenção quanto de escalabilidade e performance e, devido a sua natureza síncrona e de propagação de dados unidirecional, ele pode ser facilmente integrado a outras bibliotecas.

2.7.1 Versão React Native

Segundo (Facebook ,2017), o React Native é uma versão do React. Embora seja semelhante ao React, o React Native dispensa componentes web para então utilizar componentes nativos, ou seja da sua própria biblioteca, na criação de blocos de construção das aplicações. A Figura 5 apresenta a relação de componentes entre o React, para aplicações Web, e o React Native, para aplicações mobiles que possuem funções semelhantes, porém nomenclaturas diferentes.

Figura 5: Componentes React X React Native

React	React Native
<code><div></code>	<code><View></code>
<code></code>	<code><Text></code>
<code>, </code>	<code><ListView></code>
<code></code>	<code><Image></code>

Fonte: (Facebook, 2017)

O desenvolvedor não cria uma aplicação web móvel, um aplicativo HTML5 ou um aplicativo híbrido, mas sim uma aplicação móvel real (que não necessite da utilização de um browser). Por meio da utilização de JavaScript, o React Native permite a criação de aplicações regulares tanto para IOS quanto para Android (FACEBOOK, 2017).

Segundo (FACEBOOK, 2017) o Hot Reloading é uma funcionalidade, que aumenta tanto a praticidade quanto a produtividade do desenvolvimento por meio da execução da aplicação em fase de desenvolvimento. Caso seja efetuado uma

alteração no código, uma nova versão do arquivo modificado será gerada automaticamente para então ser inserida na aplicação.

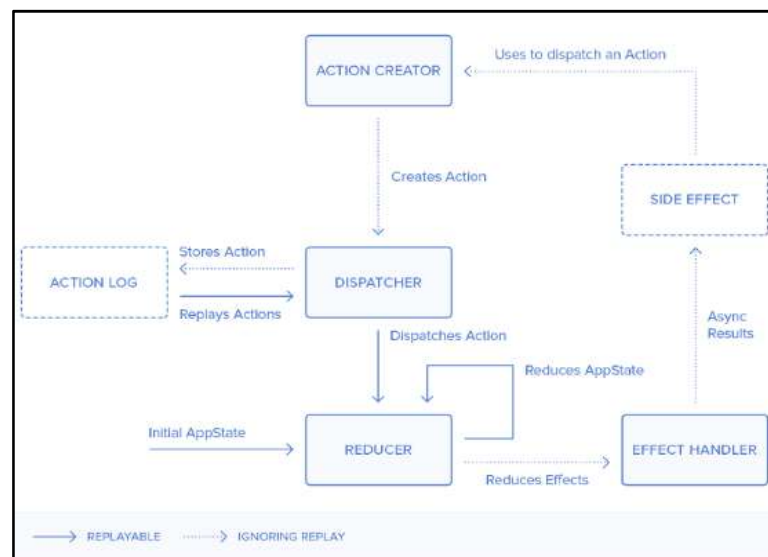
2.7.2 Arquitetura Redux

O Redux, criado por Dan Abramov, possui características semelhantes ao Flux, onde ambos possuem a mesma arquitetura, porém o Redux torna as mudanças de estado mais previsíveis e é capaz de tornar as abstrações mais fáceis (IANAKIARA, 2016).

Flux é uma arquitetura de interfaces criada a partir da necessidade de gerenciamento de estados das aplicações. Por meio da utilização de fluxo de dados unidirecional, o Flux agrupa componentes combináveis em uma interface (IANAKIARA, 2016).

Segundo (IANAKIARA, 2016), o Redux é basicamente um container de mudança de estado previsível que utiliza das melhores características do Flux para uma conclusão lógica e eficiente. Em outras palavras, Redux é uma otimização dos obstáculos e simplificação de implementação do Flux.

Figura 6: Árvore de objetos Redux



Fonte: Ianakiara (2016)

Conforme apresentado na Figura 6, o Redux armazena o estado da aplicação em uma árvore de objetos dentro do *store* onde este será o responsável por manter o

state e efetuar os disparos dos *actions* por meio da utilização do *dispatcher* (IANAKIARA, 2016).

Segundo (IANAKIARA, 2016), uma forma de efetuar uma manutenção no *state* é emitindo uma ação com informações da alteração. Para que ocorra a transformação de um estado por uma *action*, é necessário a escrita de um *reducer*. Esta estrutura do Redux oferece um padrão mais sustentável para desenvolvimento de aplicações tanto web quanto mobile.

2.8 LINGUAGEM JAVA

Java é uma linguagem de programação orientada a objetos que segue padrões de linguagens tradicionais como C e C++ e dispensa muitos elementos que tornam a programação em si, complexa. Seu código fonte pode ser compilado em qualquer sistema operacional desde que possua uma máquina virtual instalada como ambiente de execução para aplicações em Java (Santos, 2013).

Segundo (Santos, 2013) a linguagem Java dispõe de um mecanismo de exceções que trata, com eficiência, erros durante a execução dos programas. Por meio da utilização de bibliotecas de classes, disponibilizadas na própria máquina virtual Java, é possível obter diversos mecanismos como:

- a) entrada e saída;
- b) acesso à internet;
- c) manipulação de strings em alto nível;
- d) grandes e poderosas estruturas de dados;
- e) diversos utilitários;
- f) classes para implementação de interfaces gráficas.

Sistemas desenvolvidos em Java possuem restrições de acesso à memória de modo a aumentar tanto a segurança quanto robustez do sistema sem diminuir a utilidade do mesmo. Existem, segundo (Santos, 2013), três categorias de softwares desenvolvidos em Java:

- a) classes para representação de modelo: nesta categoria, as classes são utilizadas como forma de representação de modelos e abstração de dados. Essas classes de representação não são executadas diretamente, mas suas instâncias podem ser utilizadas dentro de aplicações;

- b) aplicações ou programas: executados por um sistema operacional, esta categoria permite ao software uma forma maior de interação, ou não, com o usuário final e, pode possuir uma interface gráfica.
- c) classes como conjuntos de rotinas: as classes, nesta categoria, podem conter somente métodos (rotinas) e, simplesmente, dispensar a representação de dados.

2.9 ARQUITETURA REST

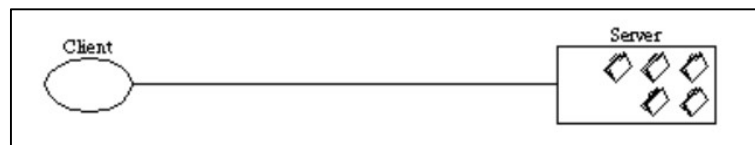
Segundo (FIELDING, 2010), a arquitetura REST, em conjunto com o protocolo HTTP, possibilita a construção de sistemas flexíveis e eficientes. O protocolo HTTP define oito métodos que podem ser trabalhados pelo REST, são eles:

- a) *GET*: obter uma representação;
- b) *PUT*: criar um novo recurso ou até mesmo alterar um já existente;
- c) *POST*: criar um novo recurso;
- d) *DELETE*: excluir um recurso existente;
- e) *HEAD*: obter apenas a representação pertinente aos dados de um recurso;
- f) *OPTIONS*: obter uma relação dos métodos suportados por um determinado recurso.

REST é um estilo arquitetural híbrido para sistemas distribuídos que ignora implementação de componentes e sintaxe de protocolos com finalidade de focar nas funções, interações e interpretações exercidas pelos componentes do sistema (FIELDING, 2010).

Segundo (FIELDING, 2010), o REST possibilita a comunicação por meio de transferências de representações em diversos formatos que são selecionados dinamicamente a partir da capacidade e necessidade do componente de receber informações de um conjunto de tipos de dados padrões. Estas representações podem ser no formato bruto ou derivado, mas, os dados sempre são representados de forma transparente.

Por meio do estilo Cliente-Servidor, tanto o cliente quanto o servidor se desenvolvem de forma independente pois o conteúdo do servidor se encontra separado do conteúdo do cliente. Um sistema pode suportar requisições de diversos domínios organizacionais, conforme apresenta a Figura 7 (FIELDING, 2010).

Figura 7: Arquitetura REST Cliente-Servidor

Fonte:Fielding (2010)

No estilo de Cliente-Servidor, cada requisição efetuada pelo cliente, deve possuir todos os atributos necessários para que seja possível o processamento por parte do servidor (FIELDING, 2010).

O lado do cliente deve conhecer totalmente a informação de estado, ou seja, as comunicações não devem depender de estados controlados pelo servidor. Este estilo pode resultar na perda de desempenho uma vez que é aumentada a quantidade de dados repetidos enviados nas requisições (FIELDING, 2010).

Para melhorar a eficiência da comunicação entre o cliente e o servidor é possível o armazenamento em memória cache, das informações, para que não seja necessário o acesso ao banco de dados a todo o momento, ou seja, permite a reutilização de informações (FIELDING, 2010).

O estilo de Cliente-Servidor com cache elimina determinadas interações que, além de melhorar a eficiência da comunicação, também reduz a média de latências na percepção do usuário. Apesar de suas vantagens, este sistema pode prejudicar a confiabilidade da aplicação pois em determinados contextos um dos lados (cliente ou servidor) pode receber dados inconsistentes (FIELDING, 2010).

Com a utilização de uma interface uniforme, a arquitetura é simplificada e a visibilidade melhorada. Define princípios de generalização de software à interface dos componentes como a identificação de recursos, diferentes representações, mensagens auto descritivas e recursos hipermídia (links). Contudo, com esta uniformização, a troca de informações é feita genericamente, ou seja, as necessidades não são especificadas (FIELDING, 2010).

Segundo (FIELDING, 2010), por meio de um sistema arquitetural REST divididos em camadas, é possível restringir acesso a determinadas camadas por meio de uma divisão em níveis hierárquicos. Com este estilo de camadas é possível proteger serviços a serem implementados e a melhorar a escalabilidade devido a distribuição de cargas entre redes e processadores.

2.10 CÓDIGO QR CODE

Segundo (Kuan-Chieh Liao, 2010) o QR Code é um sistema bidimensional de código de barras, apresentado pela companhia Japonesa Denso-Wave em 1994. Comparado ao código de barras o QR Code consegue armazenar uma quantidade maior de dados além de possuir um sistema para correção de aptidão, ou seja, dados podem ser recuperados mesmo que uma parte substancial do código esteja distorcido ou danificado (LIAO; LEE, 2010).

O primeiro propósito do QR Code, foi de acompanhar os hábitos humanos de efetuar operações e convivência com produtos do dia-a-dia por meio de uma forma prática de autenticação utilizando o smartphone e, que não deixará de efetuar a verificação na tabela de senhas (LIAO; LEE, 2010).

A necessidade de proteção contra acesso não autorizado a informações compartilhadas aumenta de acordo com a conectividade entre dispositivos. A validação de usuários, que possuem acesso a determinadas informações, é essencial. Desta forma, um método de validação, baseado em senhas, deve ser implantado para garantir melhor segurança na comunicação entre os dispositivos (LIAO; LEE, 2010).

A senha *One-time*, que é uma senha válida para uma única sessão de *login* ou transação, evita determinados tipos de ataques associados a forma comum de efetuar, como por exemplo, ataques de replay, de dicionário e de *phishing*. Sendo assim caso algum indivíduo mal-intencionado, consiga uma cópia da senha *One-time*, que já foi utilizada, não será possível o acesso ao sistema ou a condução da transação (LIAO; LEE, 2010).

Segundo (LIAO; LEE, 2010) as tecnologias mais comuns de senha *One-Time* são as baseadas em cálculos matemáticos, em *smart cards*, *tokens* e SMS. Devido a problemas na utilização de chaves encadeadas, muitos especialistas sugerem o uso de *smart cards* para o aumento da segurança em um sistema, pois além de apresentarem resistência, também mantém o arquivo de senha armazenado (LIAO; LEE, 2010)

O método por *token* resolve o problema de ter um componente que leia a chave de acesso. Para geração da chave o *token* contém um sistema de *clock* muito preciso que está sincronizado com o servidor de autenticação, porém além de conter alguns

dos pontos negativos do cartão, o *token* também requer um sistema e infraestrutura (LIAO; LEE, 2010).

2.11 SISTEMAS DE RECOMENDAÇÃO

Devido a quantidade de informações disponibilizada pela internet, é inevitável não se deparar com uma grande variedade de opções na hora de escolher um produto ou um serviço. Muitas vezes o usuário não possui experiência suficiente para escolher entre as diversas opções apresentadas e, para auxiliar neste processo é comum buscar opiniões de antigos consumidores por meio de revistas, artigos, filmes ou livros. O sistema de recomendação ingressa neste cenário com o objetivo de aumentar a capacidade e eficácia do processo de indicação, por meio da combinação adequada entre a expectativa dos usuários e os produtos a serem recomendados aos mesmos. A grande dificuldade resultante, é a descoberta do relacionamento de interesses do usuário (CAZELLA; NUNES; REATEGUI, 2010).

Segundo (CAZELLA; NUNES; REATEGUI, 2010), é possível afirmar que personalização é o ato de adequar um produto ou serviço para atender as necessidades de um indivíduo. Um sistema personalizado pode ser a diferença entre perder ou cativar um novo cliente pois, o mesmo ajuda a direcionar o usuário para o conteúdo de seu interesse e, conseqüentemente, resulta na redução do tempo gasto para encontrar informações relevantes. O sistema pode, também, sugerir produtos relacionados àquele que está sendo comprado, visualizado ou que possua alguma interação tanto direta, quanto indireta, com o usuário

Segundo (LUCAS et al., 2009), para a realização de recomendações é necessário a coleta de dados e, para isto, é possível a utilização de duas formas de coletas, sendo elas:

- a) a coleta implícita: realizada por meio do monitoramento das atividades do usuário enquanto o mesmo está dentro do sistema armazenando informações como, por exemplo, produtos que foram demonstrados interesses, sites visitados, dentre outros. A vantagem desse método é a descrição e o conforto ao usuário;

- b) coleta explícita: o usuário expressa sua opinião sobre um produto ou serviço por meio de uma pontuação, sendo assim, são gerados dados mais confiáveis e mais precisos com relação a coleta implícita.

Após a coleta dos dados é necessário que sejam efetuadas as manipulações de informações das diferentes técnicas. Uma destas, é a filtragem da informação, que pode ser colaborativa, híbrida ou baseada em conteúdo. Estas técnicas têm como objetivo refinar tanto os resultados das recomendações quanto o tempo para geração das recomendações (LUCAS et al., 2009).

O tempo para geração é um extremamente impactante no processo pois, em algumas situações, necessitam de recomendação em tempo real, desta forma, é necessário a utilização de técnicas que demandam menos tempo de processamento. Quando o tempo não é apresentado como impactante, é possível a utilização de técnicas que demandam mais tempo de processamento, mas que também resultam em recomendações mais precisas aos usuários (LUCAS et al., 2009).

Com a utilização da tecnologia *Push*, o sistema recomenda serviços independente da interação direta com o usuário, já, a tecnologia *Pull*, só são apresentadas as recomendações quando as mesmas são solicitadas, explicitamente. (LUCAS et al., 2009).

Segundo (LUCAS et al., 2009), para mais descrição, é necessário a utilização da apresentação passiva, onde, as recomendações são apresentadas dentro de um contexto no qual o usuário está localizado. A partir desta, é possível efetuar a integração com sistema de maneira mais suave e fluida. Estes sistemas não devem restringir o ambiente do usuário, de modo que obriguem o mesmo a seguir um único caminho, e sim, a sugerir direções, mesmo que existam chances de não serem escolhidas (LUCAS et al., 2009).

2.11.1 Sistema de recomendação baseado em perfil mais próximo

O sistema de recomendação baseado em perfil do mais próximo, como o nome sugere, utiliza dados referentes às preferências, ou gostos, de diversos usuários da comunidade como base para, então, efetuar uma sugestão ou recomendação (GORAKALA, 2016).

Segundo (GORAKALA, 2016), por meio da utilização dos dados coletados, são calculados valores de similaridade de outros usuários da comunidade juntamente com o usuário ativo. Conforme exemplo apresentado pela Amazon, na Figura 8, a partir dos valores coletados são identificados os usuários da comunidade que possuem mais similaridades com o usuário ativo. Com isso, é obtido uma lista de itens no qual os usuários da comunidade se interessaram no passado, porém o usuário ativo ainda não teve contato.

Figura 8: Sistema de recomendação da Amazon



Fonte: Gorakala (2016)

O método de colaboração supera a dificuldade de recomendar itens em que a descrição é vaga ou genérica. Além disso, filtros de colaboração permitem recomendar itens diferentes, que já possuam uma avaliação, para um usuário. Para isso, é necessário que outros usuários demonstrem interesse por estes itens (LINDEN; SMITH; YORK, 2003).

Segundo (GORAKALA, 2016), o sistema de recomendação baseado em perfil mais próximo também é conhecido como sistema de recomendação por similaridade ou sistema de recomendação de filtragem por colaboração.

Baseado nas presunções de que pessoas com as preferências similares no passado possuem a mesma preferência no futuro e, preferência das pessoas permanece estável e consistente no futuro, este método heurístico é dividido em duas categorias, segundo (GORAKALA, 2016):

- a) filtragem baseada em usuários;
- b) filtragem baseada em produtos.

Na Tabela 1, é apresentado um exemplo de dados coletados a partir de um sistema de streaming onde, são relacionados filmes com usuários e as notas coletadas a partir de avaliações efetuadas por estes mesmos usuários. (GORAKALA, 2016).

Tabela 1: Notas de usuários

Movie/User	Claudia Puig	Gene Seymour	Jack Mathews	Lisa Rose	Mick LaSalle	Toby
Just My Luck	3	1.5		3	2	
Lady in the Water		3	3	2.5	3	
Snakes on a Plane	3.5	3.5	4	3.5	4	4.5
Superman Returns	4	5	5	3.5	3	4
The Night Listener	4.5	3	3	3	3	
You Me and Dupree	2.5	3.5	3.5	2.5	2	1

Fonte: Gorakala (2016)

Segundo (GORAKALA, 2016), para diminuir a complexidade de entendimento, o método necessário para cálculo de recomendação pode ser resumido em três passos:

- calcular as similaridades dos usuários usando a tabela de notas dos filmes;
- para cada usuário ativo, são considerados os filmes que não foram avaliados por estes usuários e sim, por outros usuários.
- prever a nota para os filmes não assistidos de acordo com cada usuário.

Para um sistema de recomendação baseado em perfil mais próximo, conforme (GORAKALA, 2016), é utilizado a Equação 1, de distância euclidiana, para obtenção da similaridade entre os usuários.

Equação 1: Distância de Euclidean

$$(x, y) = \sqrt{\sum_{i=1}^n |x_i - y_i|^2}$$

Fonte: Gorakala (2016)

A intenção de se utilizar a fórmula de distância de Euclidean, é para que seja possível a representação dos filmes, usuários e classificações em um determinado vetor no espaço. Para interpretação da fórmula, os usuários são representados no eixo x, já os filmes no eixo y e os pontos de classificações são, nada mais nada menos que, pontos no vetor (GORAKALA, 2016).

Tabela 2: Similaridades entre usuários

	Claudia Puig	Gene Seymour	Jack Mathews	Lisa Rose	Mick LaSalle	Toby
Claudia Puig	1	0.7559289	0.9285714	0.9449112	0.6546537	0.8934051
Gene Seymour	0.7559289	1	0.9449112	0.5	0	0.3812464
Jack Mathews	0.9285714	0.9449112	1	0.7559289	0.3273268	0.662849
Lisa Rose	0.9449112	0.5	0.7559289	1	0.8660254	0.9912407
Mick LaSalle	0.6546537	0	0.3273268	0.8660254	1	0.9244735
Toby	0.8934051	0.3812464	0.662849	0.9912407	0.9244735	1

Fonte: Gorakala (2016)

De acordo com a Tabela 2, ao definir Toby como usuário ativo, é possível a verificação de que Lisa Rose, com relação aos outros usuários, possui maior similaridade com Toby. A partir desta informação, é possível avaliar, por meio da Equação 2, um filme que ainda não foi visto pelo usuário Toby e então analisar a classificação do mesmo para definir a possível classificação do filme 'Just My Luck' para o usuário Jack (GORAKALA, 2016).

Equação 2: Previsão de nota

$$\frac{(3 * 0,9285 + 1,5 * 0,944 + 3 * 0,755 + 2 * 0,327)}{(0,9285 + 0,944 + 0,755 + 0,327)} = 2,41$$

Fonte: Gorakala (2016)

Os valores obtidos, de cada usuário comparados a Jack, multiplicados por suas respectivas notas, são somados todos os resultados e dividido pelo peso das notas, para então resultar na previsão de nota do filme para Jack. Conforme equação apresentada na Equação 2, a nota prevista para o filme, considerando Jack, seria de 2,41 (GORAKALA, 2016).

A abordagem baseada em itens auxilia na caracterização de preferências do usuário com os fatores latentes. Utilizando como exemplo um sistema de recomendação de filmes, a filtragem por item, determina que um usuário é fã de filmes de comédias românticas (mesmo sem ter a noção exata de “comédia” e “romance”) e recomenda um filme do mesmo gênero, mas que ainda não foi visto. Já a filtragem por vizinhança, recomenda filmes com base em um dos vizinhos mais próximo que atribuiu uma nota alta para determinado item. Esta abordagem, certas vezes, pode resultar em uma recomendação não precisa. Segundo (DEROSIERS; KARPIS, 2009), as principais vantagens, da recomendação por vizinhança, são:

- a) a simplicidade: métodos fáceis e intuitivos de implementar onde, na sua forma mais simples, apenas um parâmetro (número vizinhos a serem considerados na predição) necessita ser afinado.
- b) a eficiência: além de ocupar pouca memória, o modelo de proximidade por vizinhança armazena as recomendações de forma off-line de forma a gerar recomendações quase que instantâneas.
- c) a estabilidade: a entrada de itens e notas, de forma constante, não afetam no desempenho e no consumo do sistema, pois não existe a necessidade de um novo treinamento.

2.11.2 Sistema de recomendação baseada em filtragem colaborativa por item

Na filtragem colaborativa por itens, é utilizada a similaridade entre um conjunto de itens e suas respectivas informações. O conceito básico do sistema de recomendação é de que se o usuário gostou de um item, no passado, pode então gostar de outro item que possui conteúdo similar. (GORAKALA, 2016).

O sistema de recomendação por itens pode ser usado para recomendações baseadas nos itens que o cliente já possui em seu “carrinho”. Esta característica é semelhante ao posicionamento de itens em um corredor de supermercado, onde os itens são posicionados estrategicamente para que, caso o usuário adquira um item, o mesmo já se sinta impulsionado a adquirir outro (LINDEN; SMITH; YORK, 2003).

Segundo (GORAKALA, 2016), a recomendação baseada em perfis mais próximos, quando comparado à filtragem baseada em item, possui determinados pontos negativos como:

- a) a classificação dos usuários normalmente é muito esparsa, o que é muito comum no mundo real onde os usuários classificam apenas determinados itens, em um largo catálogo, de modo a afetar diretamente na performance do sistema.
- b) se a quantidade de dados armazenados for, consideravelmente grande, o custo de processamento para calcular a similaridade dos valores para todos os usuários será elevado.

c) caso o perfil do usuário mude rapidamente, o custo computacional será impactado diretamente devido ao reprocessamento de valores similares.

A fórmula de medição mais comum para este tipo de abordagem é a similaridade do cosseno, que calcula a similaridade entre dois vetores de n dimensões através do ângulo no espaço entre eles. A fórmula, apresentada na Equação 3, demonstra a da similaridade do cosseno (GORAKALA, 2016).

Equação 3: Similaridade do cosseno

$$\text{sim}(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

Fonte: Gorakala (2016)

Ao utilizar como exemplo a situação citada anteriormente, onde são analisados os dados de um sistema de streaming, é possível a criação de uma tabela, apresentada na Tabela 3, de similaridade dos filmes e, em seguida, a utilização da equação do cosseno para definir todos os valores. (GORAKALA, 2016).

Tabela 3: Similaridade com do cosseno

	Just My L...	Lady in t...	Snakes on...	Superman ...	The Night...	You Me a...
Just My Luck	1	0.6339001	0.7372414	0.7194516	0.8935046	0.7898559
Lady in the Water	0.6339001	1	0.7950515	0.8149529	0.7977412	0.8897565
Snakes on a Plane	0.7372414	0.7950515	1	0.9779829	0.8585983	0.9200319
Superman Returns	0.7194516	0.8149529	0.9779829	1	0.8857221	0.9680784
The Night Listener	0.8935046	0.7977412	0.8585983	0.8857221	1	0.9412504
You Me and Dupree	0.7598559	0.8897565	0.9200319	0.9680784	0.9412504	1

Fonte: Gorakala (2016)

Para prever a classificação do filme ‘Lady in the Water’ do usuário Toby, é preciso identificar um filme similar, que no caso da tabela 3, é o filme ‘You, me and Dupree’ com 0,8897565. Multiplicando a soma da classificação dos filmes similares já feitas pelo Toby, pelos pesos de seus valores, e no final dividindo pelos valores similares do ‘Lady in the Water’ é obtido a previsão da similaridade do filme ‘Lady in the Water’ para Toby. Conforme Equação 4 (GORAKALA, 2016).

Equação 4: Previsão de similaridade

$$\frac{(0,795 * 4,5 + 0,814 * 4 + 0,889 * 1)}{(0,795 + 0,814 + 1,889)} = 3,09$$

Fonte: Gorakala (2016)

O sistema de recomendação, baseada em filtragem colaborativa por item, possui como vantagem a sua facilidade de implantação, porém, quando os alvos são novos usuários, o mesmo pode ocasionar em previsões menos precisas devido ao seu 'cold start', ou seja, a ausência de informações para a criação de recomendações (GORAKALA, 2016).

2.11.3 Estimação Bayesiana

A Teoria Bayesiana, desenvolvida por Thomas Bayes no século XVIII, é um tipo de inferência estatística que descreve as incertezas sobre quantidades invisíveis de forma probabilística. A interferência Bayesiana é o processo que, por meio da distribuição de probabilidade e quantidades não observadas, torna possível agrupar um conjunto de dados e resumir o resultado. As informações coletadas são dados amostrais (função de verossimilhança) de um conhecimento prévio a respeito dos parâmetros (distribuição a priori) e então o cálculo posteriori dos parâmetros (NASCIMENTO, 2014)

Segundo (NASCIMENTO, 2014), o teorema de Bayes envolve probabilidades condicionais, modificando a atitude inicial conforme as causas, hipóteses ou estados após ter conhecimento de um acontecimento observado, em outras palavras, representar o desconhecimento (a priori) sobre os parâmetros por meio de probabilidades.

Um exemplo do teorema de Bayes na prática e, segundo (NASCIMENTO, 2014), seria a probabilidade de queda da bolsa com fundo de investimento, onde a previsão inicial é de 0, 10 e após encerrado, as novas informações buscadas de dados históricos apresentam que o dólar está em alta frente ao real onde 20% das vezes que a bolsa caiu, foi devido a este fator. A probabilidade a priori é $P(E) = 0, 10\%$, enquanto a probabilidade de alta é $P(Ec) = 0, 90$, e as verossimilhanças são dadas por $P(B|E) = 0, 20$ e $(B|Ec) = 0, 05$. Com o Teorema de Bayes, teremos que $P(B|B) = \text{Resultado}$, ou seja, $P(E|B) = 0, 31$. Com este dado, concluímos que a probabilidade de ocorrer uma queda na bolsa é aumentada de 10% para 31%.

2.12 INTELIGÊNCIA ARTIFICIAL

A inteligência artificial é um ramo da ciência da computação que, por meio da utilização de técnicas avançadas, se propõe a simular capacidade humana de raciocinar. Os termos *Machine Learning* e *Deep Learning* são formas de inteligência artificial que surgiram na década de 1960 e hoje se apresentam como grandes avanços tecnológicos (PEREIRA, 2017).

Segundo (PEREIRA, 2017), *Machine Learning*, ou aprendizado de máquina, é um tipo de IA (Inteligência Artificial) utilizado para facilitar a capacidade em que um computador possui de aprender e ensinar o mesmo a evoluir à medida que é exposto a novos dados e funcionalidades. Existem muitas aplicações atualmente que utilizam *machine learning*, uma delas é o próprio Facebook que usa para personalizar o *feed* de cada usuário com base nos acessos e informações dos mesmos. Os algoritmos de *machine learning* utilizam análises estatísticas e preditivas para a detecção de padrões.

Segundo (SONNENBURG et al., 2007), existem grandes variedades de algoritmos de *machine learning* dos quais possuem o objetivo de solucionar problemas do mundo real. Esta área, embora esteja crescendo rapidamente, apresenta uma escassez de algoritmos com código aberto afetando diretamente no progresso científico da mesma.

Segundo (SONNENBURG et al., 2007), quando o assunto é um software open source para *machine learning*, são combinadas as necessidades de um esforço científico tanto para um software consumidor quanto para um produtor. As vantagens específicas de um software open source para *machine learning* são:

- a) reprodutibilidade e comparação de algoritmos;
- b) encontrar problemas;
- c) basear em recursos existentes;
- d) acessar ferramentas científicas sem pausas;
- e) combinação de vantagens;
- f) adoção de métodos mais rápidos;
- g) emergência colaborativa de padrões.

A facilidade para detectar bugs, em um software de código aberto, tem se apresentado como uma grande vantagem pois todos possuem permissão para

inspecionar o código e então enviar a falha aos responsáveis pela manutenção do mesmo (SONNENBURG et al., 2007).

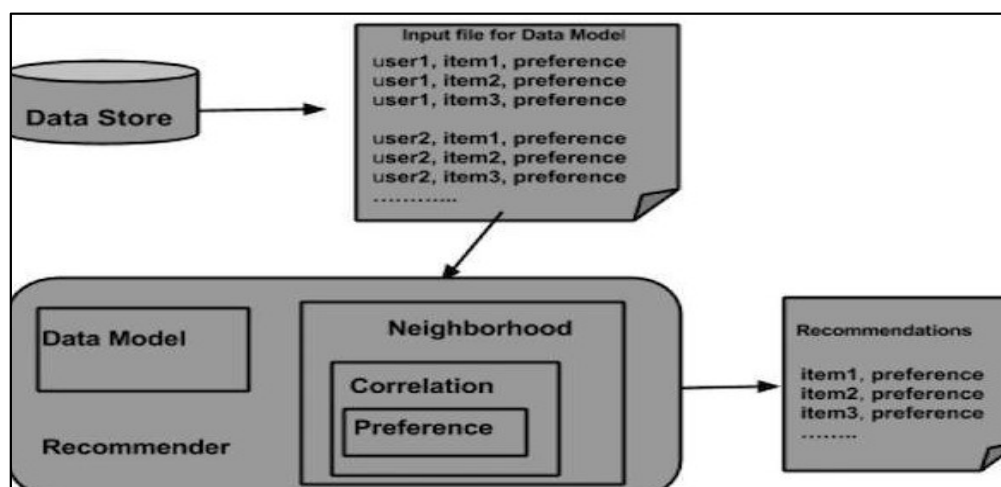
2.13 APACHE MAHOUT

Segundo (PEREIRA; CHARÃO; ROSE, 2011), a biblioteca Apache Mahout tem como objetivo primário, a criação de algoritmos de aprendizagem de máquina e, para isto, ela suporta quatro tipos de classes de operações, sendo elas:

- a) recomendação: onde itens são recomendados a partir do comportamento dos usuários;
- b) agrupamento: onde os registros são divididos em classes (*clusters*) tendo como base sua semelhança;
- c) classificação: onde são classificados os itens de acordo com uma categorização já existente;
- d) mineração por frequência de grupos de itens: onde são identificados padrões e relação entre os grupos de itens.

O Apache Mahout é uma biblioteca de algoritmos de mineração de dados que utiliza a linguagem, de orientação a objetos, java. Seus algoritmos, além de serem escritos conforme o modelo MapReduce, também utilizam a ferramenta Apache Hadoop e implementam Naive Bayes (PEREIRA; CHARÃO; ROSE, 2011).

Figura 9: Arquitetura motor Recommender



Fonte: W3ii (2017)

Segundo (PEREIRA; CHARÃO; ROSE, 2011) o motor de recomendação é o mecanismo responsável por gerar as recomendações para um usuário específico. Este motor analisa dados passados pelo usuário e então recomenda novos produtos. Sua arquitetura é apresentada na Figura 9.

Conforme apresentado na Figura 9, o modelo de dados é preparado a partir dos dados armazenados no banco de dados. Com a criação do modelo de dados, o mecanismo de recomendação recebe uma entrada e, após análise, conclui seu processo por meio do envio de recomendações (PEREIRA; CHARÃO; ROSE, 2011).

Inspirados nas primitivas *map* e *reduce*, das linguagens convencionais, o MapReduce é um modelo de programação no qual os algoritmos do Apache Mahout são escritos. Primeiramente, é efetuada a divisão dos dados de entrada em conjuntos independentes e, para cada conjunto, é aplicado uma operação *map* que resulta no conjunto de pares intermediários chave/valor. Logo após a geração dos pares intermediários, é aplicada uma operação *reduce*, que agrupa os valores onde seus pares possuem chaves iguais (PEREIRA; CHARÃO; ROSE, 2011).

Como o Apache Hadoop implementa o modelo MapReduce, é possível distribuir os dados analisados com a utilização do Hadoop Distributed File System (HDFS) por meio do Apache Mahout. O Apache Hadoop trata-se de um conjunto de programas para computação distribuída e escalável. Esta ferramenta dispõe do sistema de arquivos distribuídos Hadoop Distributed File System (HDFS) e o framework MapReduce, nos quais serão utilizados pelo Mahout (PEREIRA; CHARÃO; ROSE, 2011).

2.13.1 Fórmula de erro médio absoluto (MAE)

O Apache Mahout implementa em seu algoritmo de filtragem colaborativa baseada em perfil de usuário mais próximo, a fórmula MAE (Mean Absolute Error), utilizada para calcular a média dos erros absolutos resultante da diferença entre duas variáveis contínuas conforme apresentado na Equação 5 (SEMINARIO; WILSON, 2012).

Equação 5: Erro médio absoluto

$$MAE = \frac{\sum_{i=1}^n | ActualRating_i - PredictedRating_i |}{n}$$

Fonte: Seminario e Wilson (2012)

A fórmula MAE, como apresenta a Figura 18, é composta pelas variáveis *ActualRating_i*, *PredictedRating_i* e *n*, que dizem respeito a, respectivamente, classificação real (valor verdadeiro), classificação prevista (previsão) e ao número total de classificações previstas. A fórmula MAE, em outras palavras, pode ser descrita como uma medida comum de erro de previsão em análises temporais (SEMINARIO; WILSON, 2012).

2.13.2 Iniciando com o Apache Mahout

Segundo (INGERSOLL, 2009), o Apache Mahout foi criado pela Apache Software Foundation (ASF) com o interesse em aprendizagem por máquina e implementações robustas escaláveis. o Apache Mahout visa:

- a) a construção de uma comunidade de usuários;
- b) a concentração em casos de uso práticos;
- c) o fornecimento da documentação de qualidade e exemplos.

O próprio Apache Mahout fornece ferramentas para a construção de um motor de recomendação (rápido e flexível) por meio da utilização da biblioteca Taste que suporta tanto recomendações com base em item quanto recomendações com base em usuário, além disso, a mesma dispõe de diversas opções para realização das recomendações. Segundo (INGERSOLL, 2009), a biblioteca Taste consiste em cinco componentes, sendo eles:

- a) *Data Model*: consiste no armazenamento de usuários, itens e preferências;
- b) *User Similarity*: consiste em uma interface que define a similaridade entre dois usuários;
- c) *Item Similarity*: consiste em uma interface que define a similaridade entre dois itens;

- d) *Recommender*: consiste em uma interface que fornece recomendações;
- e) *User Neighborhood*: consiste em uma interface com finalidade de calcular uma vizinhança de usuários similares e então utilizar o Recommender.

Segundo (INGERSOLL, 2009), os componentes da biblioteca Taste, e suas respectivas implementações, possibilitam criações tanto de sistemas de recomendações simples, quanto de sistemas de recomendações complexos, onde podem ser baseados em recomendações em tempo real ou em recomendações offline.

2.13.3 Semelhança colaborativa de usuário com Apache Mahout

O algoritmo de semelhança colaborativa de usuário (PCS) é bem conhecido e utilizado para encontrar semelhanças de nós. Para iniciar com sua utilização, é necessária uma base de conhecimento de modo que seja possível calcular similaridade a partir das comparações entre interações de usuários. (ÇAPRAS, 2016)

Segundo (ÇAPRAS, 2016), existem diversos métodos para calcular a similaridade entre os usuários. Um destes métodos é pela utilização do coeficiente de correlação entre as interações dos usuários com o sistema. Na Figura 10 é apresentado o algoritmo, escrito em Java, de Pearson Correlation Similarity (PCS).

Figura 10: Algoritmo de semelhança colaborativa

```

1 UserSimilarity userSimilarity = new PearsonCorrelationSimilarity(model);
2 UserNeighborhood neighborhood = new NearestNUserNeighborhood(3, UserSimilarity, model);
3
4 Recommender recommender = new GenericUserBasedRecommender(model, neighborhood, userSimilarity);
5 Recommender cachingRecommender = new CachingRecommender(recommender);
6
7 List<RecommendedItem> recommendations = cachingRecommender.
8
9 recommend(1, 10);

```

Fonte: Çapras (2016)

Como apresentado na Figura 10, é necessário definir quais similaridades dos usuários o motor de recomendação deve levar em consideração para então inserir quantos itens devem ser recomendados por usuários. No caso da Figura 12, são recomendados dez itens para um único usuário por meio do comando `recommend(1,10)` e, com a utilização do `NearestNUserNeighborhood(3, userSimilarity, model)`, o

motor de recomendação passa a considerar apenas três usuários similares (ÇAPRAS, 2016).

O PCS mensura a correlação entre duas variáveis em uma escala de -1 até 1 já a similaridade da distância euclidiana, mensura em uma escala de 0 até 1. Quando o PCS resulta em 1, significa que os dados do objeto estão perfeitamente correlacionados, já quando o PCS resulta em -1, significa que os dados não estão, de forma alguma, correlacionados. Com isto, é possível identificar a taxa de desvio padrão entre ambos os objetos que é matematicamente expressa na Equação 6. (ÇAPRAS, 2016).

Equação 6: Semelhança colaborativa

$$Pearson(x, y) = \frac{\sum xy - \frac{\sum x \sum y}{N}}{\sqrt{(\sum x^2 - \frac{(\sum x)^2}{N})(\sum y^2 - \frac{(\sum y)^2}{N})}}.$$

Fonte: Çapras (2016)

O termo similaridade indica que, para recomendar algo ao usuário em questão, é necessário encontrar usuários na vizinhança que possuem um perfil semelhante e, por meio deste inserir na filtragem colaborativa (ÇAPRAS, 2016)

2.13.4 Limite de vizinhança de usuário

A técnica de limite de vizinha de usuário é utilizada para quando o resultado é apresentado com um limite de 0.1, ou seja, quando as recomendações geradas a partir de outro algoritmo de filtro colaborativo, não são precisas devido a quantidade de dados armazenados (ÇAPRAS, 2016).

De uma forma resumida, a técnica de limite de vizinhança considera outros filtros para então gerar recomendações diferentes, mas com uma precisão lógica. Como por exemplo, o resultado de uma amostragem de dados levantadas pelo tipo de filme que um usuário goste não foi o suficiente, porém é possível gerar recomendações mais precisas a partir das avaliações inseridas por outros usuários,

que assistiram a determinados filmes e avaliaram os mesmos com notas similares, de forma a desconsiderar os tipos de filmes (ÇAPRAS, 2016).

3 DESENVOLVIMENTO

Neste capítulo são apresentadas as características sobre a metodologia utilizada, as formas nos quais as tecnologias foram aplicadas e, os resultados alcançados com o desenvolvimento desta aplicação.


3.1 METODOLOGIA

Como metodologia, foi considerado a pesquisa exploratória e aplicada onde, a pesquisa exploratória proporcionou a identificação das necessidades existentes para desenvolvimento de um sistema de gerenciamento de comanda eletrônica com foco em recomendação de pedidos.

O levantamento dos dados, a partir dos experimentos efetuados, possuem o objetivo de sugerir pedidos (produtos) por meio do conhecimento de notas (atribuídas por usuários) e de grupos de perfis (de usuários).

3.2 DESENVOLVIMENTO DA APLICAÇÃO WEB

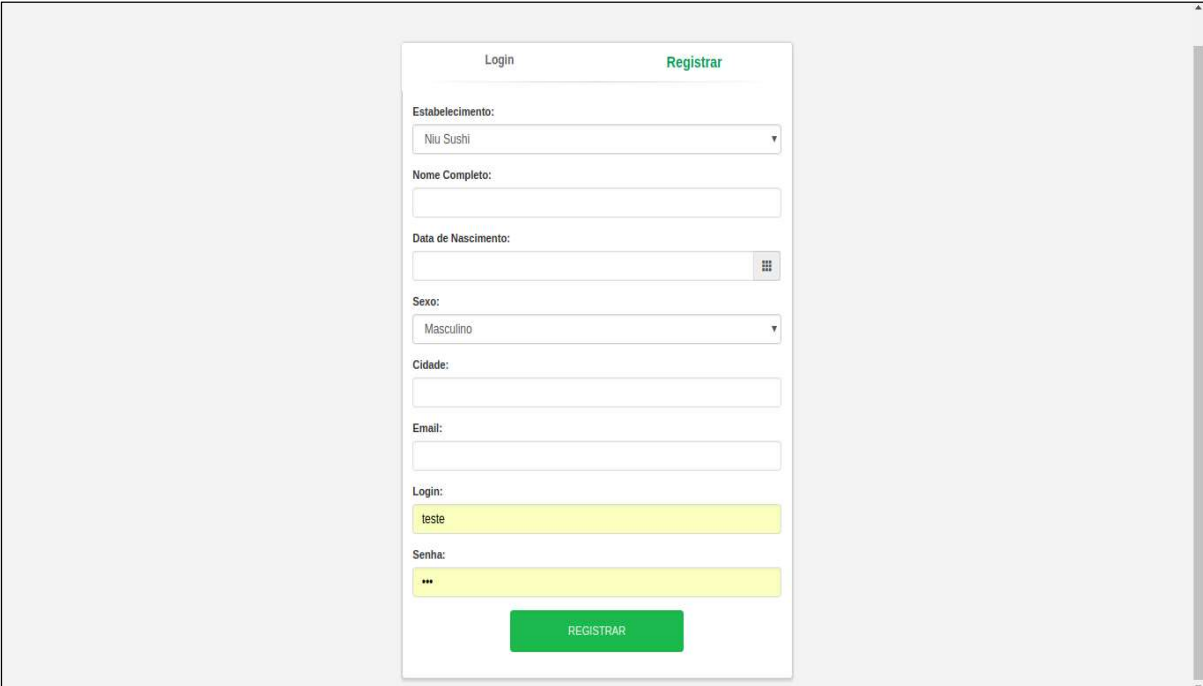
Primeiramente foi desenvolvida a aplicação Web para o estabelecimento utilizar como plataforma de atendimento aos pedidos efetuados pelos clientes. Para que existisse uma forma de autenticação, foi criado uma tela com formulário de login e, com a utilização de JavaScript, nesta mesma tela foi inserido um formulário para o usuário efetuar cadastro em seu respectivo estabelecimento conforme apresentado pelas Figuras 11 e 12.

Figura 11: Formulário de login do atendenteA interface de login para o atendente. No topo, há dois links: "Login" (em verde) e "Registrar". Abaixo, há dois campos de entrada: o primeiro contém o texto "yurilongaray" e o segundo contém pontos para mascarar a senha. No final, há um botão verde com o texto "ENTRAR".

Fonte: Os autores

A Figura 11 apresenta o formulário de login para usuário do tipo atendente do estabelecimento. Para ingressar no sistema o usuário deve inserir um login e senha cadastrados no banco de dados, caso tente ingressar em outra tela do sistema sem autenticação, o usuário será redirecionado para esta tela de login. Para segurança, foram adicionadas funções nativas do PHP, como `mysqli_real_escape_string()` para evitar ataques de SQL injection e, `md5()` para cifrar a senha do usuário.

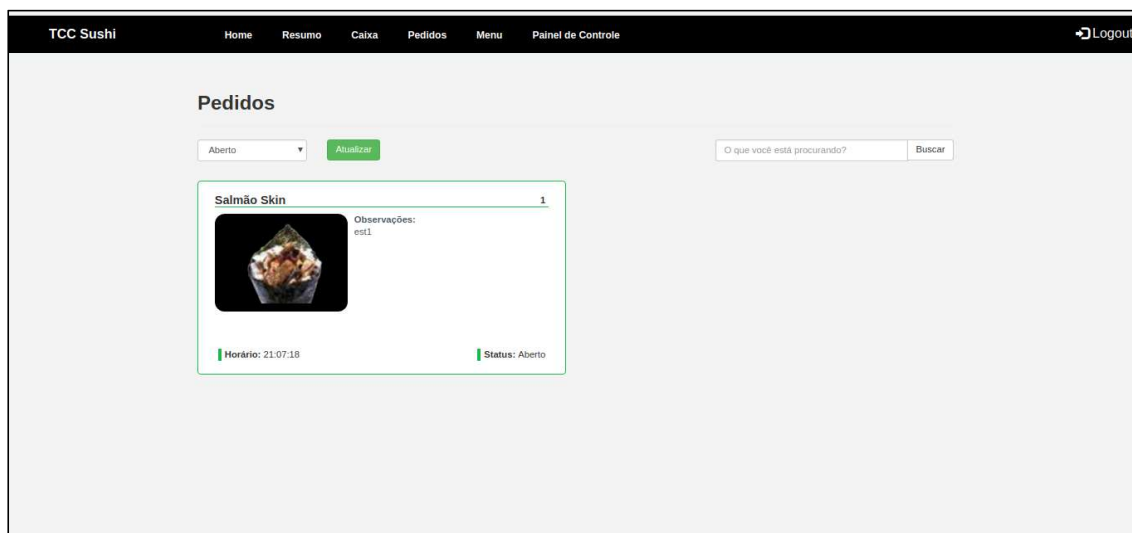
Na Figura 12 é apresentado o formulário de registro, onde um usuário do tipo atendente informará suas credenciais que serão armazenadas no banco de dados MySQL. Este formulário, é compartilhado com a tela de login que por meio da utilização de JavaScript foi criado um evento para ocultar o login quando o botão Registrar for pressionado, e vice-versa.

Figura 12: Formulário de registroA interface de registro para o atendente. No topo, há dois links: "Login" e "Registrar" (em verde). O formulário contém os seguintes campos: "Estabelecimento:" com uma lista suspensa selecionando "Niu Sushi"; "Nome Completo:" com um campo de texto; "Data de Nascimento:" com um campo de data; "Sexo:" com uma lista suspensa selecionando "Masculino"; "Cidade:" com um campo de texto; "Email:" com um campo de texto; "Login:" com um campo de texto contendo "teste"; e "Senha:" com um campo de texto contendo pontos. No final, há um botão verde com o texto "REGISTRAR".

Fonte: Os autores

Após o atendente efetuar login, o mesmo será direcionado para a tela de pedidos, conforme apresentado na Figura 13, onde estão apresentados os pedidos solicitados pelos clientes. A tela de pedidos possibilita, ao atendente, o filtro tanto pelo nome do pedido (que é o mesmo do item do menu), quanto pelo status do mesmo.

Figura 13: Listagem de pedidos



Fonte: Os autores

Criado modal com a utilização de JQuery e Bootstrap para a alteração de status dos pedidos. Esta model, apresentada na Figura 14, lista os status cadastrados no banco de dados e apresentados por meio de uma *query* (pesquisa) em SQL efetuada pela linguagem PHP.

Figura 14: Alteração do status do pedido

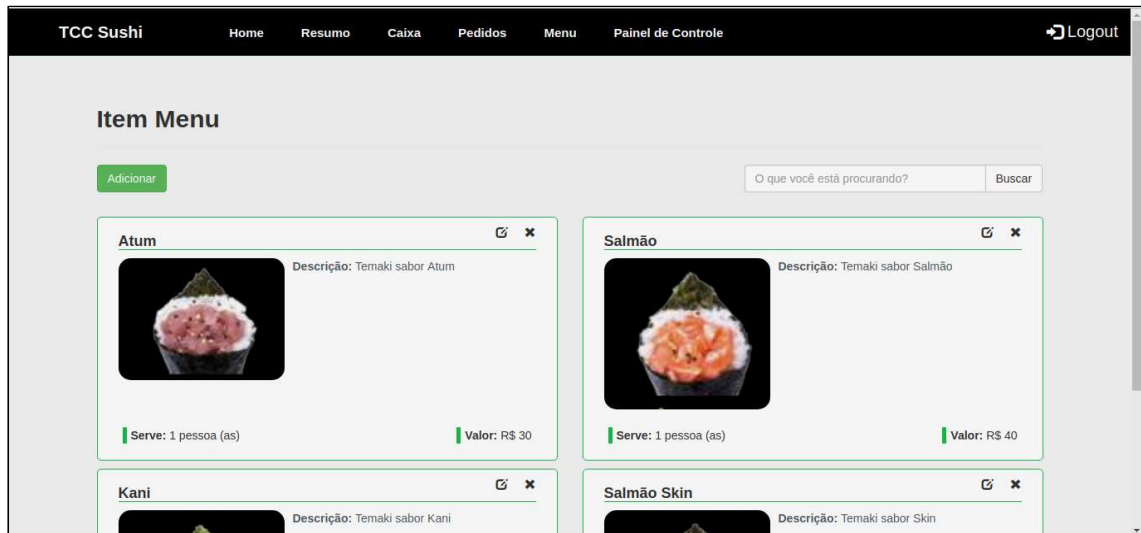


Fonte: Os autores

Para que um cliente efetue um pedido, é necessário optar por um item do menu, desta forma, foi desenvolvido uma página para a criação destes itens. A página,

chamada de item_menu.php, é apresentada na Figura 15 e, além de permitir que o atendente cadastre, edite ou exclua, também permite a filtragem pelo nome de determinado item do menu.

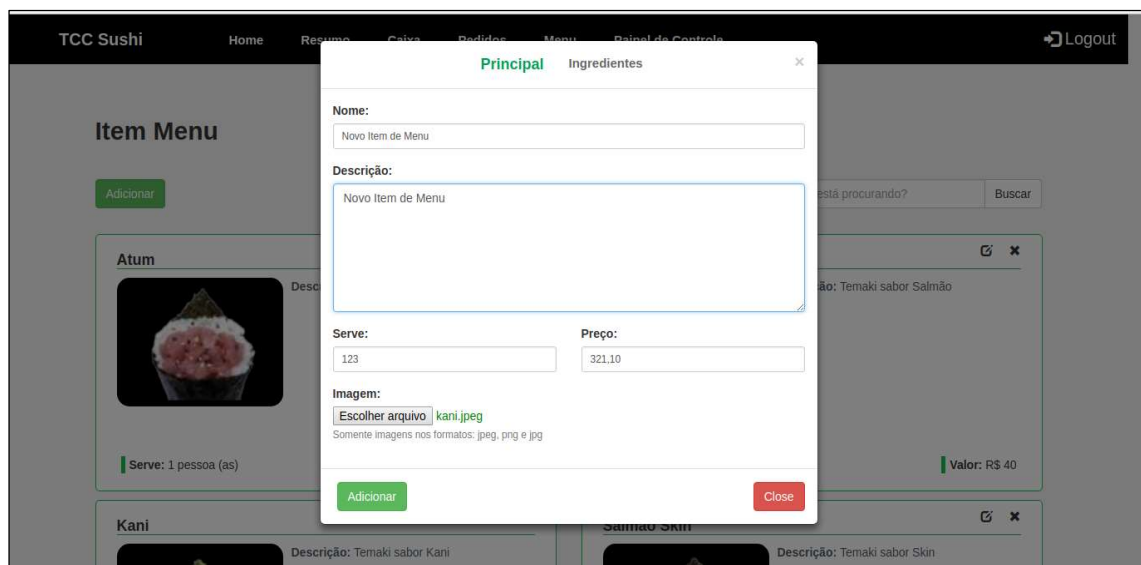
Figura 15: Itens do Menu



Fonte: Os autores

Para uma interface inteligente e responsiva, foi implementado o *framework* Bootstrap em todos os elementos HTML possíveis de forma que os mesmos se adaptem a resolução e ao dispositivo utilizado pelo usuário.

Figura 16: Cadastro do item de menu



Fonte: Os autores

Na aplicação Web, todos os componentes de filtros, cadastros, edições e exclusões, utilizam as tecnologias Ajax e JQuery para o envio de requisições HTTP de uma determinada página PHP para outra. Com esta implementação, o usuário poderá efetuar operações sem precisar atualizar ou sair da página atual. Os resultados, da requisição solicitada, serão aplicados no elemento HTML informado no código desenvolvido. A Figura 16 apresenta um exemplo desta implementação.

3.3 DESENVOLVIMENTO DA APLICAÇÃO MOBILE

O desenvolvimento da aplicação mobile, destinado ao cliente, é efetuado a partir dos componentes nativos do React Native e JavaScript. A aplicação mobile dispõe de um formulário com *inputs* para cadastro de cliente, conforme apresentado na Figura 17, que é armazenado diretamente no banco de dados MySQL e então consultado por demais funcionalidades da aplicação.

Figura 17: Cadastro de cliente

A imagem mostra a interface de usuário para o cadastro de um cliente em uma aplicação móvel. O formulário é composto por campos de texto para 'Nome', 'E-mail', 'Senha' e 'Cidade'. Abaixo desses, há um campo para 'Data de Nascimento' com um ícone de calendário e um campo para 'Sexo' com a opção 'Masculino' selecionada. No fundo do formulário, há uma ilustração de uma bicicleta. Na base da tela, há um botão vermelho com o texto 'CADASTRAR' em branco. O status bar no topo da tela indica o tempo 7:05.

Fonte: Os autores

Para que o usuário ingresse no sistema, é efetuado uma comparação entre os dados do banco de dados e os dados inseridos na tela de login. Uma listagem de estabelecimentos é apresentada assim que o usuário ingressa no sistema conforme Figura 19. Toda tela desenvolvida na aplicação mobile, sem exceções, buscam

informações no banco de dados por meio de requisições HTTP efetuadas pelo servidor REST.

Figura 18: Listagem de estabelecimentos



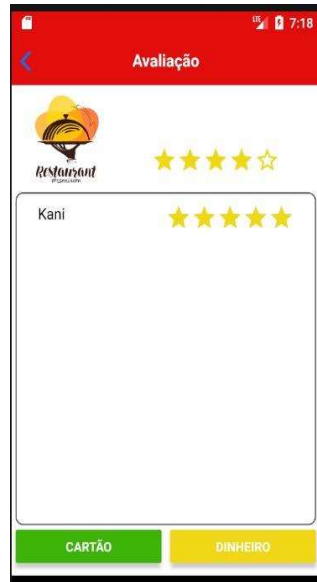
Fontes: Os autores

Por meio da utilização do padrão Redux, são gerenciadas todas as variáveis do sistema. Conforme apresentado anteriormente, no capítulo de Redux, as alterações e obtenções de variáveis são efetuadas por meio do disparo de *actions* para então iniciar o tratamento pelos *Reducers*.

3.2 DESENVOLVIMENTO DO SISTEMA DE RECOMENDAÇÃO

Para criação do sistema de recomendação, foram desenvolvidas as telas de avaliação do item de menu, menu do restaurante e a tela de perguntas para o cliente. Desta forma é possível capturar de informações necessários para que o sistema de recomendação possa operar e então sugerir itens posteriormente.

A tela de avaliação do item de menu, apresentada na Figura 19, disponibiliza ao cliente uma forma de avaliar o item do menu por meio da seleção de uma nota entre um e cinco, representada por estrelas douradas.

Figura 19: Avaliação do item

Fonte: Os autores

A nota de um a cinco, informada na Figura 19, é armazenada na tabela de avaliação para que então o sistema de recomendação efetue a leitura desta tabela e gere recomendações a partir da nota inserida.

Na tela de perguntas ao cliente, apresentada na Figura 20, é disponibilizado um formulário para conhecimento das preferências do cliente. As respostas inseridas nas perguntas são utilizadas pelo sistema de recomendação com o propósito de sugerir itens para quando o cliente ainda não efetuou nenhuma avaliação. As respostas são convertidas em uma escala de 0 (zero) a 1 (um) a fim de gerar uma margem de erro.

Figura 20: Perguntas ao cliente

Deixe-nos conhecer um pouco sobre você

1 - Você prefere comidas doces ou salgadas?

Doce

2 - Possui algum tipo de restrição alimentar?

Vegano

3 - Qual seu tipo de restaurante favorito?

Alemã

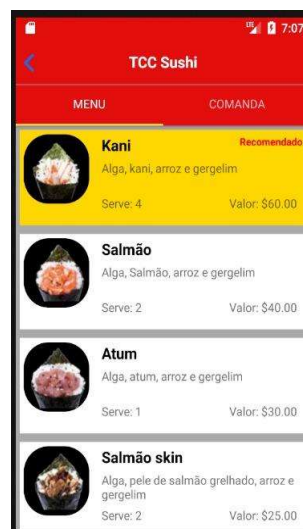
FINALIZAR CADASTRO

Fonte: Os autores

Por mais importante que sejam as informações referentes às avaliações efetuadas pelo usuário, é interessante o armazenamento das respostas provindas das perguntas apresentadas na Figura 20 para que o motor de recomendação (*Recommender*) perceba um padrão e o implemente em seu algoritmo.

A partir do uso da tabela chamada recomendação, que está presente no banco de dados, é gerado uma quantidade determinada de recomendações por cliente e então levantados os itens com maior avaliação. Conforme apresentado na Figura 21, os itens levantados são apresentados na tela de menu do estabelecimento em formato destacado.

Figura 21: Menu do estabelecimento



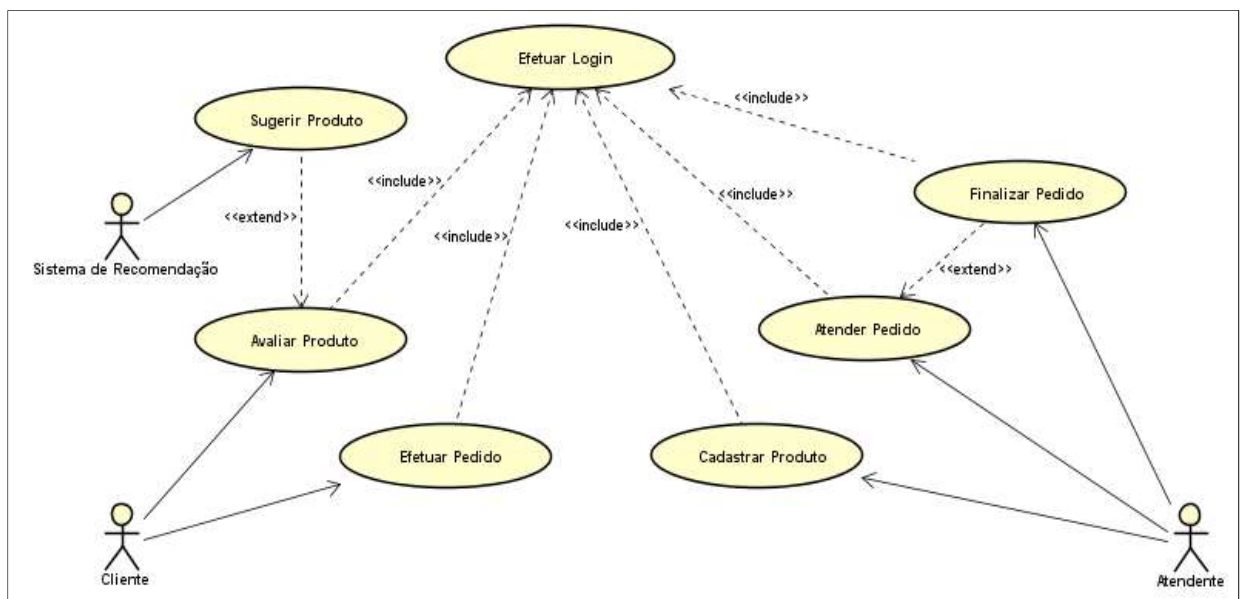
Fonte: Os autores

Para que os produtos não se repitam, o sistema de recomendação exclui, com uma determinada frequência, todos os registros da tabela de recomendação e então preenche a mesma novamente de modo que novos itens sejam sugeridos ao cliente.

3.3 MODELAGEM DO SISTEMA

No diagrama representado na Figura 22, são demonstradas as ações em que os usuários (atores) cliente, atendente e sistema de recomendação, podem executar no sistema.

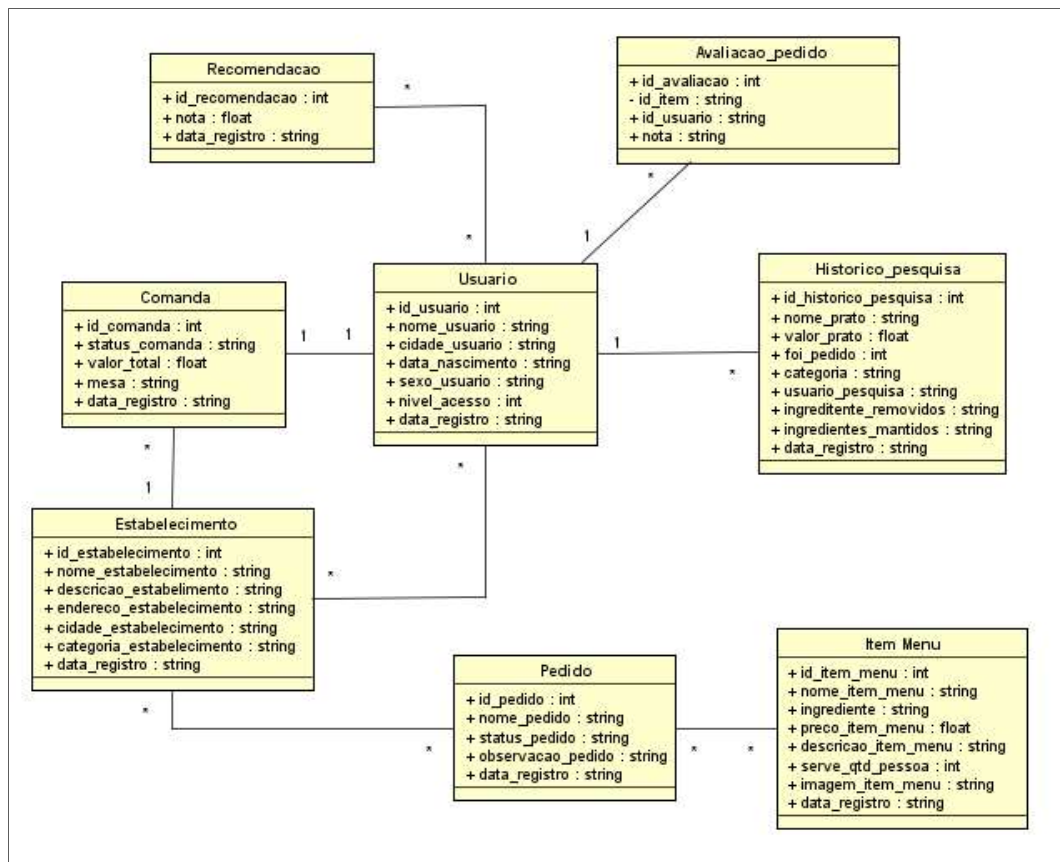
Figura 22: Diagrama de casos de uso



Fonte: Os autores

Após a definição supracitadas dos diagramas de casos de uso, são especificadas as características da estrutura de classes, representado na Figura 23 pelo diagrama de classes.

Figura 23: Diagrama de classes

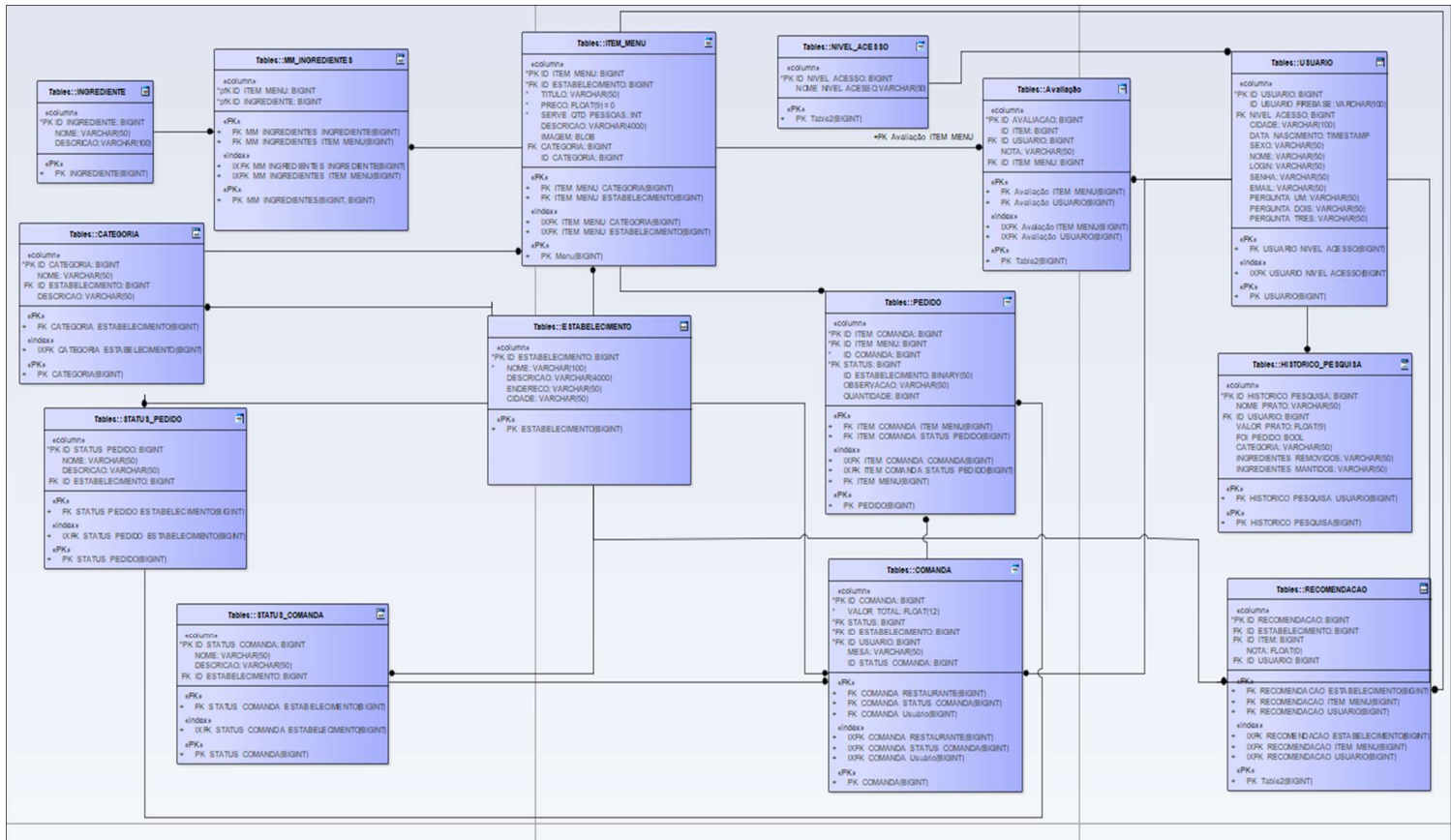


Fonte: Os autores

No diagrama, representado na Figura 23, são demonstradas as classes presentes no software desenvolvido, bem como suas devidas interações aos quais buscam demonstrar as funcionalidades do software.

Após a definição tanto do diagrama de casos de uso quanto do diagrama de classes, são especificadas as características da estrutura do banco de dados, ou seja, as tabelas, colunas e relacionamentos, conforme representado na Figura 24.

Figura 24: Modelo entidade relacionamento



Fonte: Os autores

Os dados inseridos e manipulados pelo software, são armazenados no banco de dados em um total de 14 (quatorze) tabelas. Os dados são de extrema importância para o funcionamento correto do sistema e para que seja possível realizar a sugestão de produtos a partir da execução do sistema de recomendações.

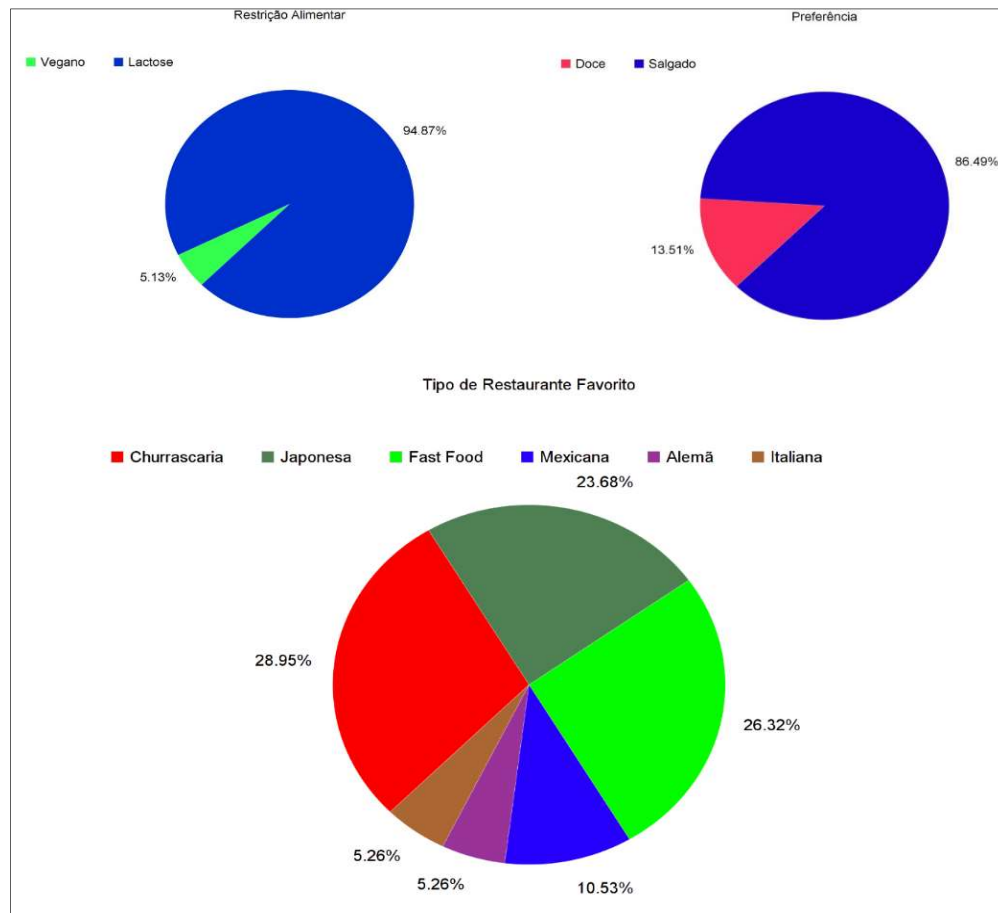
3.4 RESULTADOS ALCANÇADOS

Para validação do sistema de recomendação, foi necessário a realização de uma pesquisa de campo por meio de um questionário (formulário) a partir da utilização da ferramenta Google Forms. O questionário possui o objetivo de analisar e segmentar grupos de perfis. As perguntas inseridas inicialmente foram referentes:

- a restrição alimentar do usuário;
- a preferência de sabores do usuário;
- a preferência de estabelecimentos do usuário.

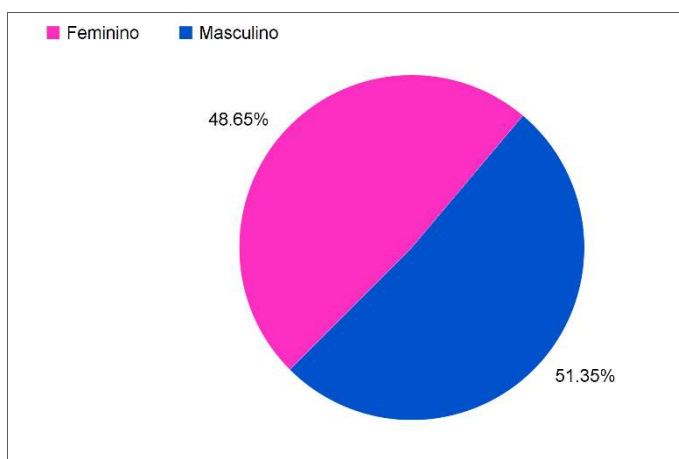
Conforme os gráficos apresentados na Figura 25, apenas uma, das três perguntas efetuadas, mais precisamente a pergunta referente a preferência de estabelecimento, obteve um determinado equilíbrio de resultado enquanto as demais não possibilitaram uma análise mais avançada devido à falta de diversidade nas respostas inseridas pelos usuários.

Figura 25: Perguntas preliminares



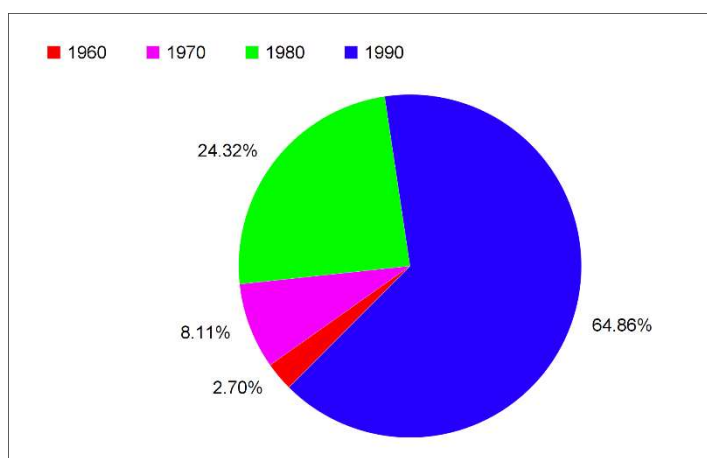
Fonte: Os autores

Embora perguntas como sexo e data de nascimento, estejam presentes no questionário, as mesmas não influenciaram diretamente no sistema de recomendação. Conforme o gráfico apresentado na Figura 26, por meio da pergunta referente ao sexo do indivíduo, é possível a identificação de um equilíbrio entre homens e mulheres que, posteriormente, pode ser considerada nos filtros para o algoritmo de recomendação.

Figura 26: Sexo dos usuários

Fonte: Os autores

Conforme o gráfico apresentado na Figura 27, com a utilização da pergunta referente a idade do usuário, foi possível identificar a faixa etária dos usuários que são apresentados jovens entre 17 a 27 anos de idade.

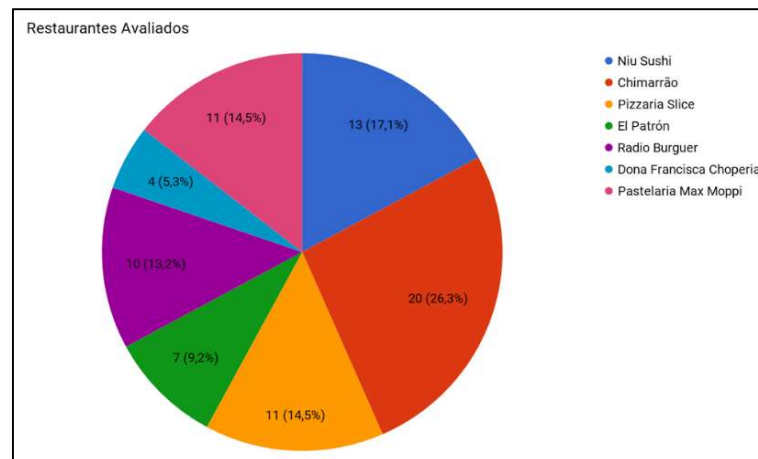
Figura 27: Faixa etária

Fonte: Os autores

Para a pesquisa de opinião foram selecionados determinados estabelecimentos de Joinville, estes foram avaliados levando em consideração sua popularidade e seu ramo de atuação. O objetivo desta seleção de estabelecimentos, é de comprovar que, por meio do sistema de recomendação por perfil, é possível a recomendação de itens totalmente diferentes. Uma vez que, para efetuar uma recomendação, não é necessário a similaridade entre os produtos consumidos e sim entre os usuários que efetuaram determinada interação com o estabelecimento.

Conforme o gráfico apresentado na Figura 28, cada estabelecimento foi avaliado, no mínimo, quatro vezes para que fosse possível a recomendação de produtos para um determinado usuário, mesmo que este ele não tenha comparecido no estabelecimento ainda.

Figura 28: Estabelecimentos avaliados



Fonte: Os autores

Para recomendação de produtos e validação de resultados, foram criados oito experimentos onde cada experimento agrupa três variáveis. A primeira variável diz respeito ao banco de dados, onde estão contidas as avaliações dos produtos de todos os usuários e as respostas dos mesmos. A segunda variável representa o algoritmo de definição de correlação, que, no projeto foram abordados os algoritmos de Correlação de Pearson (Pearson Correlation) e o de Distância de Euclidean (Euclidean Distance). Já a terceira variável se refere ao método utilizado para definir a porcentagem/quantidade de usuários próximos onde foram abordados os métodos de limite por vizinhança de usuário (Threshold User Neighborhood) e Proximidade de N usuários (Nearest N User Neighborhood).

Para a averiguação dos dados e garantir que os resultados gerados eram reais, todos os algoritmos foram testados com a fórmula MAE, conforme as Tabelas 4 e 5. O resultado foi gerado 10 vezes, para cada algoritmo, e então gerado a média e o desvio padrão para garantir que os resultados não foram obtidos por sorte.

Tabela 4: Média e desvio padrão sem perguntas preliminares

Média Fórmula MAE (Desvio Padrão)		
	Distância de Euclidean	Correlação de Pearson
Proximidade de N Usuarios	0,5 (0,2690)	0,6666667461 (0,4180)
Limite por vizinhança de usuário	0,7305633277 (0,1479)	0,757081151 (0,5271)

Fonte: Os autores

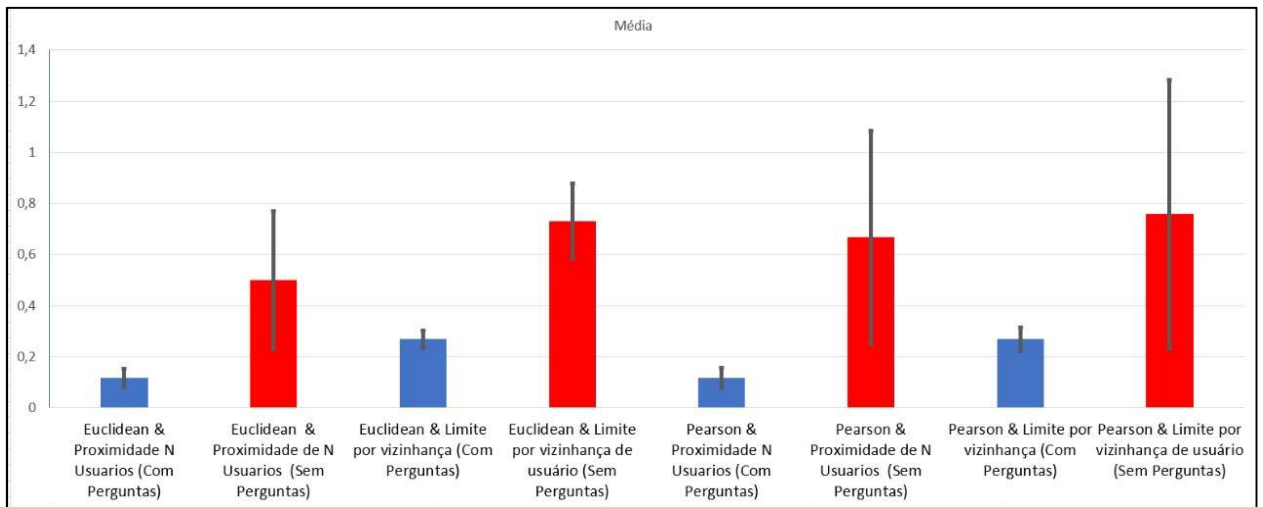
Os algoritmos de recomendação, apresentados nos apêndices R, S, T e U, foram selecionados para implementação nos experimentos devido a sua portabilidade (uma vez que os mesmos são suportados pelo sistema de recomendação Apache Mahout e, consequentemente, codificados em Java) e, a popularidade no mercado.

Tabela 5: Média e desvio padrão com perguntas preliminares

Média Fórmula MAE (Desvio Padrão)		
	Correlação de Pearson	Distância de Eclidean
Limite por vizinhança de usuário	0,2687563345 (0,0470)	0,2684511032 (0,0338)
Proximidade de N usuários	0,1172447669 (0,0390)	0,1166978781 (0,0348)

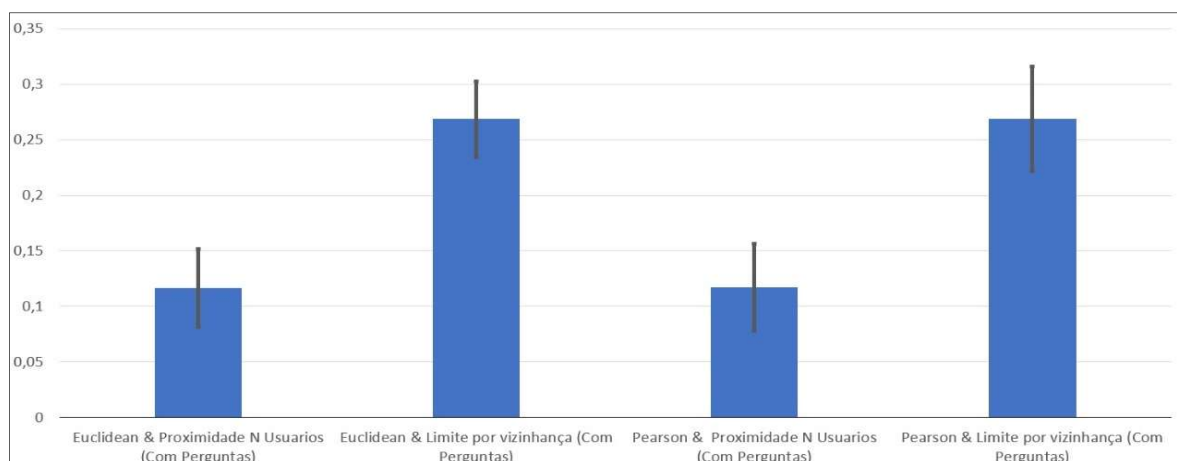
Fonte: Os autores

Conforme o gráfico apresentado na Figura 29, é possível identificar o aumento no erro médio a partir da utilização das perguntas preliminares. Os itens com perguntas preliminares são representados pela cor azul, já os em vermelho, não possuem perguntas preliminares. É necessário ressaltar que ambos os experimentos possuem perguntas avaliativas.

Figura 29: Relação de algoritmos com/sem perguntas preliminares

Fonte: Os autores

Como apresenta a Figura 30, a partir da utilização das perguntas preliminares, os dados resultaram em informações mais precisas uma vez que, tanto a taxa de erro, quanto a taxa de desvio padrão se apresentaram baixas. Vale ressaltar que 90% (noventa por cento) dos dados são testados e os 10% (dez por cento) restantes, são utilizados para treinamento.

Figura 30: Média e desvio padrão com perguntas preliminares e avaliativas

Fonte: Os autores

Conforme a Figura 30, é possível identificar que os algoritmos de limite por vizinhança para cálculo de correlação, possuem taxa de erro maior do que os dados

gerados a partir da utilização do algoritmo de vizinhança por N usuários. Os algoritmos, que utilizam a fórmula de proximidade por N Usuários, possuem taxa de erro de 0,11, em outras palavras, resultados mais precisos.

3.5 TRABALHOS RELACIONADOS

Muitos sistemas pesquisados possuem formas de recomendação, mas todos recomendam itens de forma aleatória ou caso exista uma determinada especialidade, como por exemplo, um desconto em determinado produto.

É possível citar, como exemplo, o sistema Soft Restaurant que está disponível no link: <http://www.programasparaempresas.com/softrestaurant>. O Soft Restaurant é um sistema de comanda eletrônica originado na Costa Rica e, que está no mercado desde 2006. Embora possua diversas funcionalidades para a gestão de um estabelecimento, o mesmo não implementa nenhum tipo de algoritmo de recomendação e sim, formas especiais (como mencionado anteriormente) para tratativas de descontos, ou semelhantes, em determinados produtos.

O Hunglion é um software, gratuito, de comanda eletrônica para restaurantes. O Hunglion dispõe de uma interface web e mobile, e possui um projeto desktop em desenvolvimento, entre suas principais funcionalidades estão controle de caixa, cadastro e gestão de clientes e fornecedores, conciliação bancária e controle de estoque. O Hunglion está disponível no link: <https://www.hunglion.com/sistema-para-restaurant>.

O Você Comanda é um aplicativo voltado para resolver o problema de tempo gasto em filas. Possui somente uma interface mobile com design simples e poucas funcionalidades. O mesmo foca apenas no controle de comandas. Você Comanda está disponível no link: <http://vocecomandaapp.com.br/index.html#services>.

O SisChef, que é um sistema pago, foca em diversos segmentos de restaurantes (desde bares, restaurantes a quilo até sistemas de delivery). O SisChef possui tanto uma aplicação web quanto uma aplicação mobile. Suas principais funcionalidades são: o reconhecimento de chamadas por bina, o reconhecimento do cliente pelo número de telefone e a impressão de pedidos. O SisChef está disponível no link: <https://sischef.com/>.

Tabela 6: Relação de Softwares X Funcionalidades

Softwares	Aplicação web	Aplicação mobile	Cnciliação bancária	Controle de estoque	Controle de Caixa	Sistema de Recomendação
SisChef	x	x			x	
Você Comanda		x				
Hunglioni	x	x	x	x	x	
Soft Restaurant	x	x			x	
<i>Sistema Desenvolvido</i>	x	x			x	x

Fonte: Os autores

4 CONCLUSÃO

Este trabalho apresentou o desenvolvimento de uma aplicação com o fim de sugerir produtos baseados no perfil de usuário mais próximo. Além disso, a aplicação também resultou na agilidade de atendimento, na agregação de valor para os estabelecimentos e na solução para perdas, fraudes e adulterações de comandas.

Os erros, a partir da utilização do banco de dados do projeto, variam entre 0,11 (no melhor dos casos) e 0,26 (no pior dos casos). Desta forma, é concluso que os resultados obtidos são confiáveis. A taxa de erro tende a diminuir com a aplicação dos algoritmos citados para casos reais juntamente com um banco de dados mais populado. Os algoritmos possibilitam a segmentação de forma mais detalhada, referente aos padrões dos usuários.

Na avaliação da MAE, em ambos os casos de testes, foram utilizados 90% (noventa por cento) dos casos para treinamento e 10% (dez por cento) para validação de dados, durante a validação foi verificado que, cada vez que era utilizado uma partição diferente para testes, o resultado gerado era diferente. Desta forma, é necessário que o algoritmo opere várias vezes de modo a obter a média e o desvio padrão necessário para avaliar se o resultado era consistente. As utilizações das perguntas introdutórias possibilitaram a criação de um algoritmo confiável juntamente com uma taxa de erro consideravelmente pequena.

Os resultados demonstram que a aplicação otimiza o processo de qualquer estabelecimento de modo a diminuir a indecisão por parte dos usuários, uma vez que serão recomendados itens nos quais usuário nunca experimentou.

A ausência de informação para recomendações iniciais devido as respostas dos usuários, nas perguntas preliminares, dificultam as pesquisas desta forma é necessário a adição de perguntas genéricas para uma melhor segmentação. Posteriormente, o projeto pode agrupar em sua estrutura, funcionalidades como relatórios, recomendação de restaurantes para os usuários, implementações de segurança com QR Code e sistemas de localização por meio da utilização do Google Maps.

Para finalizar, o algoritmo que utiliza a correlação por proximidade de N usuários é o que mais se adequa ao nosso projeto pois, nos experimentos efetuados, o mesmo obteve uma taxa de erro muito baixa comparada com as notas geradas pelo sistema.

5 REFERÊNCIAS

PEDROSA, Paulo & NOGUEIRA, Tiago. **Computação em Nuvem**. 2011. Disponível em: <<http://www.ic.unicamp.br/~ducatte/mo401/1s2011/T2/Artigos/G04-095352-120531-t2.pdf>>. Acesso em: 20 nov. 2017.

OLIVEIRA, Tiago & PEREIRA, Júlio. **Uma introdução significativa sobre computação nas nuvens (cloud computing)**. 2015. Disponível em: <<http://docplayer.com.br/2615275-Uma-introducao-significativa-sobre-computacao-nas-nuvens-cloud-computing.html>>. Acesso em: 20 nov. 2017.

CAZELLA, Silvio & NUNES, Maria & REATEGUI, Eliseo. **A ciência da opinião: Estado da arte em sistemas de recomendação**. 2010. Disponível em: <<http://meninasnacomputacao.com.br/gutanunes/publications/JAI4.pdf>>. Acesso em: 20 nov. 2017.

ÇAPRAS. **A Content Boosted Hybrid Recommendation System**. 2016. Disponível em: <<http://etd.lib.metu.edu.tr/upload/12619752/index.pdf>>. Acesso em: 20 nov. 2017.

GORAKALA, Suresh. **Building Recommendation Engines With R**. 2016. Disponível em: <https://books.google.com.br/books?hl=pt-BR&lr=&id=V9VOCwAAQBAJ&oi=fnd&pg=PP1&dq=Building+Recommendation+gorakala+&ots=xXEJXlZfyf&sig=SylgK-mnLg92sLycZdsH_641sHA#v=onepage&q&f=false>. Acesso em: 20 nov. 2017.

LIAO, Kuan-Chieh & LEE, Wei-Hsun. **A Novel User Authentication Scheme Based on QR-Code**. 2010. Disponível em: <<http://ai2-s2-pdfs.s3.amazonaws.com/9df2/e4cecc9b28befd513542ee38276b23d1fd41.pdf>>. Acesso em: 20 nov. 2017.

LUCAS, Adriano. **Personalização para televisão digital utilizando a estratégia de sistema de recomendação para ambientes multiusuário**. 2009. Disponível em: <<https://repositorio.ufscar.br/bitstream/handle/ufscar/452/3290.pdf?sequence=1&isAllowed=y>>. Acesso em: 20 nov. 2017.

NASCIMENTO, Denis. **Estimação Bayesiana**. 2014. Disponível em: <http://www.ufpa.br/heliton/arquivos/aplicada/seminarios/M2_03_Estimacao_Bayesiana_Denis.pdf>. Acesso em: 20 nov. 2017.

PEREIRA, Lúcio & SILVA, Michel. **Android para desenvolvedores**. 2009. Disponível em : <[https://books.google.com.br/books?hl=pt-BR&lr=&id=8u9wJowXfdUC&oi=fnd&pg=PA1&dq=Android+para+desenvolvedores&ots=LUdhZ22lm1&sig=Vq\\$FF3sY9WorLMRB7GknuvcNZuo#v=onepage&q&f=false](https://books.google.com.br/books?hl=pt-BR&lr=&id=8u9wJowXfdUC&oi=fnd&pg=PA1&dq=Android+para+desenvolvedores&ots=LUdhZ22lm1&sig=Vq$FF3sY9WorLMRB7GknuvcNZuo#v=onepage&q&f=false)>. Acesso em: 20 nov. 2017.

PEREIRA, Tiago. **Diferença entre machine learning e deep learning**. 2017. Disponível em: <<http://computerworld.com.br/qual-diferenca-entre-machine-learning-e-deep-learning>>. Acesso em: 20 nov. 2017.

SIMÕES, Danielle & PEREIRA, César. **Sistemas operacionais móveis**. 2014. Disponível em: <https://azslide.com/sistemas-operacionais-moveis-android-x-ios_59e30ebf1723ddec036e66b5.html>. Acesso em: 20 nov. 2017.

SONNENBURG, Soeren. **The Need for Open Source Software in Machine Learning**. 2007. Disponível em: <<http://www.jmlr.org/papers/volume8/sonnenburg07a/sonnenburg07a.pdf>>. Acesso em: 20 nov. 2017.

CAMPOS, Jonathan. **RESTMB: API RESTful para Android**. 2013. Disponível em: <<http://aberto.univem.edu.br/bitstream/handle/11077/988/Monografia%20Final.pdf?sequence=1>>. Acesso em: 20 nov. 2017.

COSTA, Fernando. **Design responsivo para WEB com Bootstrap**. 2014. Disponível em: <<http://blog.unifimes.edu.br/fernando/wp->

content/uploads/sites/2/2014/05/Design-responsivo-para-WEB-com-Bootstrap.pdf>. Acesso em: 20 nov. 2017.

TOBALDINI, Ricardo. **Aplicação do Estilo Arquitetural REST a um Sistema de Congressos**. 2008. Disponível em:

<https://projetos.inf.ufsc.br/arquivos_projetos/projeto_673/TCCRicardoGhisiTobaldini_final.pdf>.

Acesso em: 20 nov. 2017.

FIELDING, Roy. **Representational State Transfer (REST)**. 2010. Disponível em:

<http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm>. Acesso em: 20 nov. 2017.

COSTA, Carlos. **Desenvolvimento para Web**. 2007. Disponível em:

<https://books.google.com.br/books?hl=pt-BR&lr=lang_pt&id=Jn6dTDF-wcsC&oi=fnd&pg=PT3&dq=related:SG4Bc7oKAeIJ:scholar.google.com/&ots=wM9FUM30Wf&sig=I4RW_43vgmpUC4CjYfaEYev8qmw#v=onepage&q&f=false>. Acesso em: 20 nov. 2017.

FLANAGAN, David. **JavaScript**. 2013. Disponível em:

<https://books.google.com.br/books?hl=pt-BR&lr=lang_pt&id=zWNYDgAAQBAJ&oi=fnd&pg=PR1&dq=java+script&ots=IzvbE5K7iK&sig=r6GKZA0_W0U81T4x3zYKvSu5Vts#v=onepage&q&f=false>. Acesso em: 20 nov. 2017.

BENTO, Evaldo. **PHP e MySQL**. 2013. Disponível em:

<https://books.google.com.br/books?hl=pt-BR&lr=&id=xG2CCwAAQBAJ&oi=fnd&pg=PP7&dq=desenvolvimento+web&ots=_YXf_KDeSz&sig=r8OIrD6REsNKgpC7el4hcnZ5co8#v=onepage&q&f=false>. Acesso em: 20 nov. 2017.

NIEDERAUER, Juliano. **Desenvolvendo Websites com PHP**. 2008. Disponível em:

<<http://www.martinsfontespaulista.com.br/anexos/produtos/capitulos/650595.pdf>>. Acesso em: 20 nov. 2017.

SILVA, Maurício. **Ajax com JQuery**. 2012. Disponível em:

<<https://books.google.com.br/books?hl=pt->

BR&lr=&id=iFGiDQAAQBAJ&oi=fnd&pg=PA23&dq=jquery+introdu%C3%A7%C3%A3o&ots=o_GHnV-h_g&sig=ns4uxhRlt-6dJ_cdQdAVG5GYIVY#v=onepage&q&f=false>. Acesso em: 20 nov. 2017.

CHERUBINI, Edivaldo. **Desenvolvimento em ASP.NET MVC, JQuery e Ajax.**

2011. Disponível em:

<http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/585/1/MD_COADS_2011_1_01.pdf>.

Acesso em: 20 nov. 2017.

SANTOS, Rafael. **Introdução à programação orientada a objetos usando JAVA.**

2013. Disponível em: <[https://books.google.com.br/books?hl=pt-](https://books.google.com.br/books?hl=pt-BR&lr=&id=2pfpCgAAQBAJ&oi=fnd&pg=PT5&dq=linguagem+java+introdu%C3%A7%C3%A3o&ots=sn5v5stVhh&sig=exhvFoIEzgMU7Lc3Zm8uxdj4kn4#v=onepage&q&f=false)

BR&lr=&id=2pfpCgAAQBAJ&oi=fnd&pg=PT5&dq=linguagem+java+introdu%C3%A7%C3%A3o&ots=sn5v5stVhh&sig=exhvFoIEzgMU7Lc3Zm8uxdj4kn4#v=onepage&q&f=false>. Acesso em: 20 nov.

2017.

INGERSOLL, Grant. **Introdução ao Apache Mahout.** 2009. Disponível em:

<<https://www.ibm.com/developerworks/br/java/library/j-mahout/index.html>>. Acesso em: 20 nov. 2017.

PEREIRA, Adriano & CHARÃO, Andrea & ROSE, César. **Análise de Desempenho da Ferramenta Apache Mahout para Mineração de Dados Distribuída.** 2011.

Disponível em: <<http://www.lbd.dcc.ufmg.br/colecoes/erad-rs/2011/0014.pdf>>. Acesso em: 20 nov. 2017.

LOPES, Vinicius. **Iniciando com React.** 2017. Disponível em:

<<https://blog.coderockr.com/iniciando-com-react-parte-1-a79d74fe8f8c>>. Acesso em: 20 nov. 2017.

FACEBOOK. **Getting started with React Native.** 2017. Disponível em:

<<http://facebook.github.io/react-native/docs/getting-started.html>>. Acesso em: 20 nov. 2017.

IANAKIARA. **5 Motivos para utilizar Redux.** 2016 Disponível em:

<<https://blog.getty.io/5-motivos-para-aprender-redux-6ac730f3f1f2>>. Acesso em: 20 nov. 2017.

SEMARIO, WILSON. **Case Study Evaluation of Mahout as a Recommender Platform**. 2012. Disponível em: <

https://www.researchgate.net/profile/Carlos_Seminario/publication/259261455_Case_study_evaluation_of_Mahout_as_a_recommender_platform/links/560309c708ae596d2591c57d.pdf>. Acesso em: 29 nov. 2017.

LINDEN, SMITH, YORK. **Amazon.com Recommendations Item-to-Item Collaborative Filtering**. 2003. Disponível em: <

<https://www.cs.umd.edu/~samir/498/Amazon-Recommendations.pdf>> Acesso em: 29 nov. 2017.

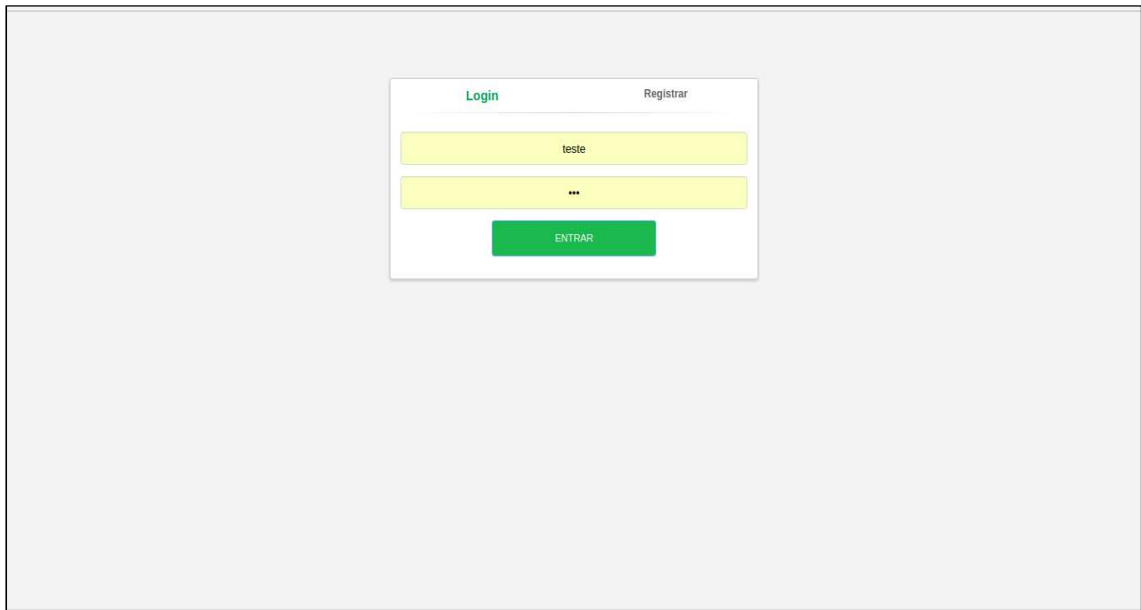
DEROSIERS, KARPIS. **A comprehensive survey of neighborhood-based recommendation methods**. 2009. Disponível em: <

<https://www.google.com.br/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0ahUKEwio38TO1ZnYAhXHPCYKHY2VApcQFgg2MAI&url=https%3A%2F%2Fetsmtl.ca%2Fgetattachment%2FProfesseurs%2Fcdesrosiers%2FPublications%2FRecHB-SurveyOfNeighborhoodBasedRecommendation.pdf.aspx&usg=AOvVaw3lc1-tFRn2j-PvEjHSNM7z>> Acesso em: 29 nov. 2017.

6 APÊNDICE

APÊNDICE A

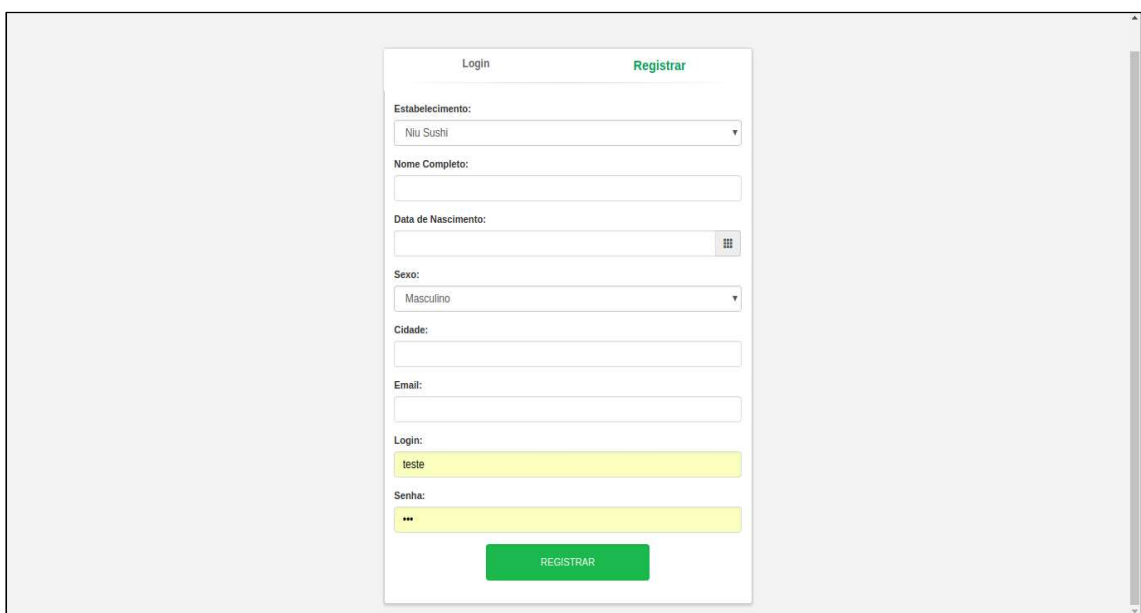
Tela de login do atendente



A screenshot of a login screen for an attendant. The screen has a light gray background. In the center, there is a white rectangular box with a thin gray border. At the top of this box, there are two tabs: 'Login' (highlighted in green) and 'Registrar'. Below the tabs, there are two yellow input fields. The first field contains the text 'teste'. The second field contains three dots '...'. Below these fields is a green button with the text 'ENTRAR' in white capital letters.

APÊNDICE B

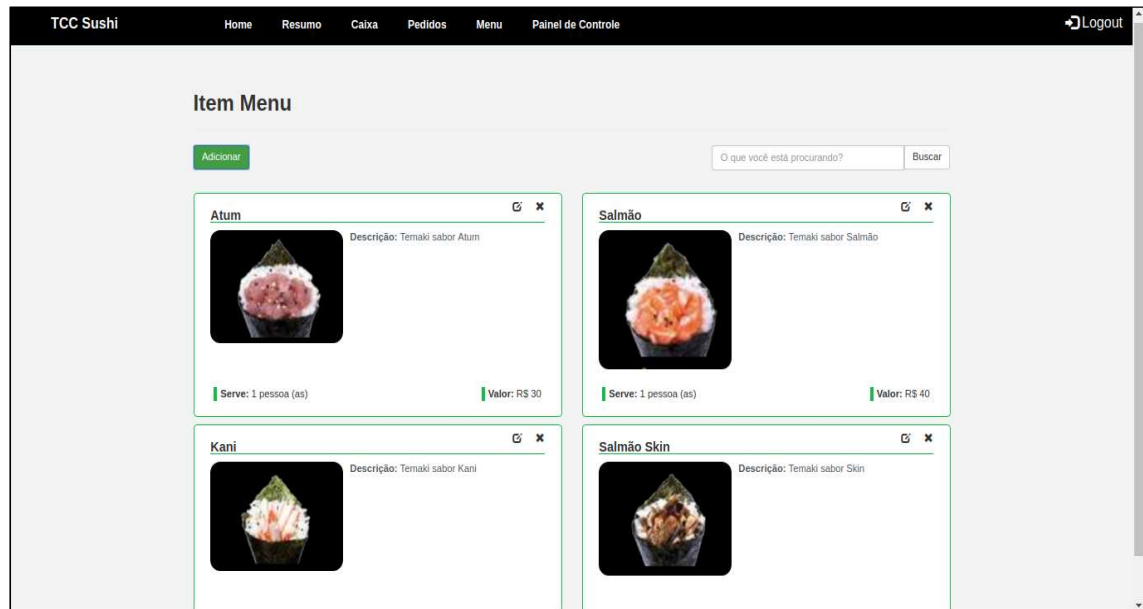
Tela de cadastro do atendente



A screenshot of a registration screen for an attendant. The screen has a light gray background. In the center, there is a white rectangular box with a thin gray border. At the top of this box, there are two tabs: 'Login' and 'Registrar' (highlighted in green). Below the tabs, there are several form fields: a dropdown menu for 'Estabelecimento:' with 'Niu Sushi' selected; a text field for 'Nome Completo:'; a date picker for 'Data de Nascimento:'; a dropdown menu for 'Sexo:' with 'Masculino' selected; a text field for 'Cidade:'; a text field for 'Email:'. Below these fields, there are two yellow input fields. The first field is labeled 'Login:' and contains the text 'teste'. The second field is labeled 'Senha:' and contains three dots '...'. Below these fields is a green button with the text 'REGISTRAR' in white capital letters.

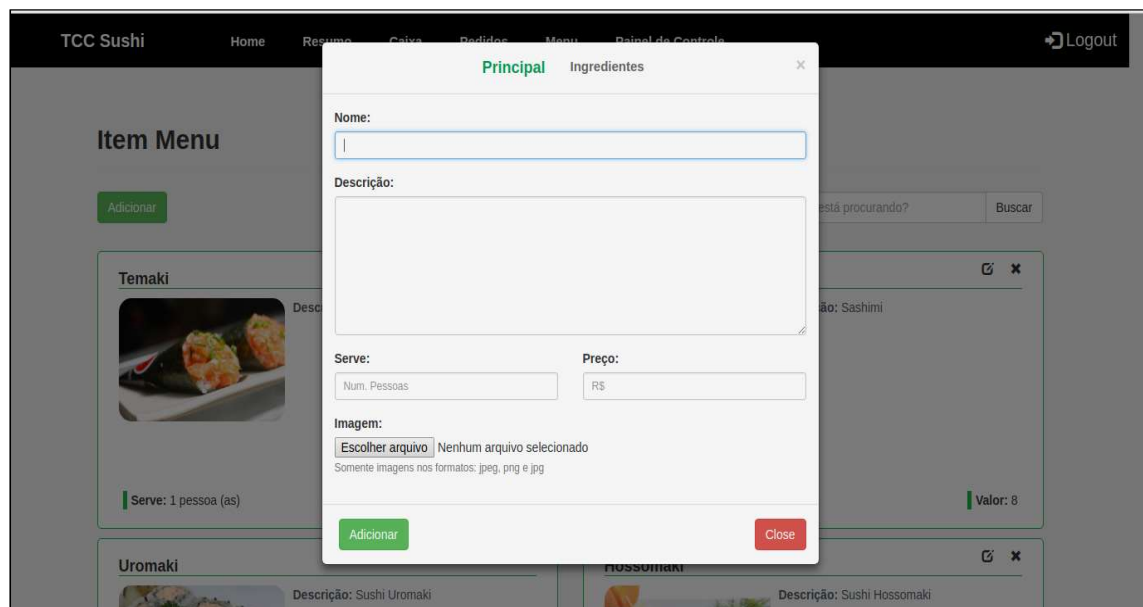
APÊNDICE C

Itens do menu do atendente



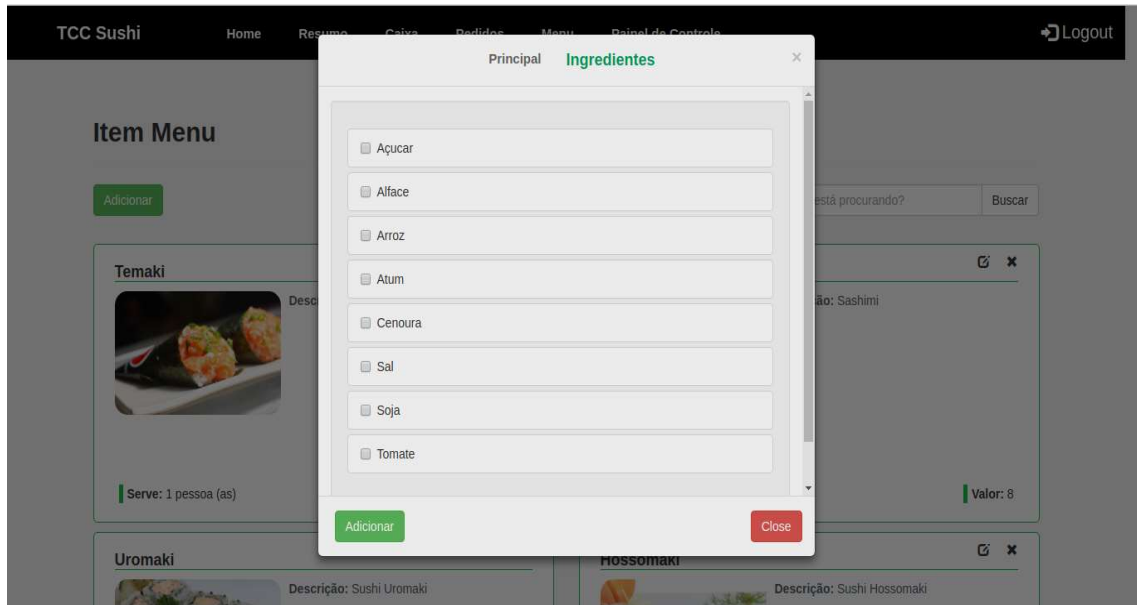
APÊNDICE D

Cadastro de item do menu do atendente



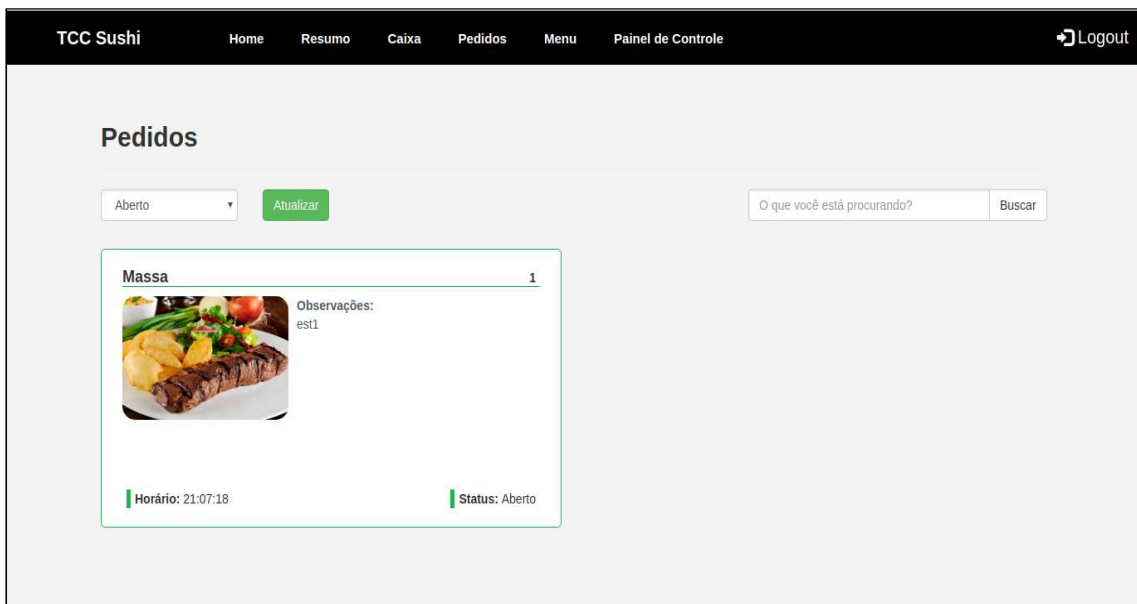
APÊNDICE E

Vinculação entre item menu e ingredientes do atendente



APÊNDICE F

Pedidos do atendente



APÊNDICE G

Cadastros gerais (painel de controle) do atendente

TCC Sushi
Home
Resumo
Caixa
Pedidos
Menu
Painel de Controle
Logout

Painel de Controle

Status de Pedido
Aberto Fechado Pendente
Cadastrar

Status de Comanda
Aberto Fechado Pendente
Cadastrar

Ingredientes
Açúcar Alface Arroz Atum Cenoura Sal Soja Tomate
Cadastrar

APÊNDICE H

Caixa / comandas do atendente

TCC Sushi
Home
Resumo
Caixa
Pedidos
Menu
Painel de Controle
Logout

Caixa / Comandas

Comanda	Status da Comanda	Nome do Cliente	Código da Mesa	Valor a Pagar	Encerrar
1	Aberto	Yuri	47	33	✕
4	Aberto	Carlos	41	15	✕
2	Fechado	Joao	22	500	✕
5	Fechado	Suzi	32	79	✕
3	Pendente	Vilma	51	120	✕
6	Pendente	Pedro	71	110	✕

APÊNDICE I

Cadastro de cliente

Nome

E-mail

Senha

Cidade

Data de Nascimento

Masculino

CADASTRAR

APÊNDICE J

Segunda etapa de cadastro do cliente

Deixe-nos conhecer um pouco sobre você

1 - Você prefere comidas doces ou salgadas?

Doce

2 - Possui algum tipo de restrição alimentar?

Vegano

3 - Qual seu tipo de restaurante favorito?

Alemã

FINALIZAR CADASTRO

APÊNDICE K

Login do cliente

E-mail

Senha

Ainda não tem cadastro? Cadastre-se

ACESSAR

APÊNDICE L

Avaliação de item do cliente

Avaliação

Restaurant

Kani

CARTÃO

DINHEIRO

APÊNDICE M

Ingredientes de pedido do cliente



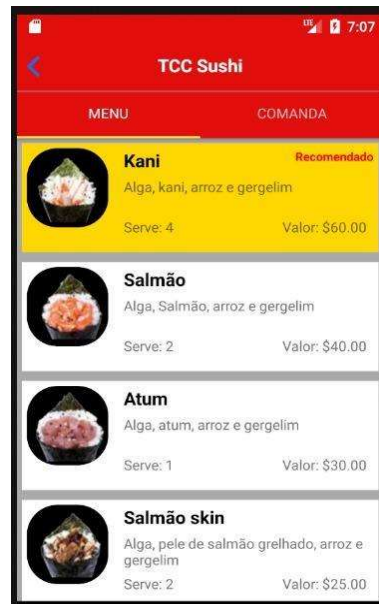
APÊNDICE N

Listagem de estabelecimento do cliente



APÊNDICE O

Menu do estabelecimento do cliente



APÊNDICE P

Realização de pedido do cliente



APÊNDICE Q

Início de comanda do cliente



Qty	Pedido	Valor
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
2	Kani	60,00
Total		1080,00

ENCERRAR

APÊNDICE R

Algoritmo de distância de Euclidean

```

1 package br.com.alura;
2
3 import org.apache.mahout.cf.taste.common.TasteException;
4
15 public class RecomendadorDeProdutosBuilder implements RecommenderBuilder {
16
17     public Recommender buildRecommender(DataModel model) throws TasteException {
18         UserSimilarity similarity = new EuclideanDistanceSimilarity(model);
19
20         UserNeighborhood neighborhood = new NearestUserNeighborhood(3, similarity, model);
21         UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
22
23         return recommender;
24     }
25 }
26
27 }

```


APÊNDICE S

Algoritmo de vizinhança de usuário mais próximo

```

1 package br.com.alura;
2
3 import org.apache.mahout.cf.taste.common.TasteException;
15
16 public class RecomendadorDeProdutosBuilder implements RecommenderBuilder {
17
18     public Recommender buildRecommender(DataModel model) throws TasteException {
19         UserSimilarity similarity = new PearsonCorrelationSimilarity(model);
20
21         UserNeighborhood neighborhood = new NearestUserNeighborhood(3, similarity, model);
22         UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
23
24         return recommender;
25     }
26
27 }

```

APÊNDICE T

Algoritmo de semelhança colaborativa de usuário

```

1 package br.com.alura;
2
3 import org.apache.mahout.cf.taste.common.TasteException;
15
16 public class RecomendadorDeProdutosBuilder implements RecommenderBuilder {
17
18     public Recommender buildRecommender(DataModel model) throws TasteException {
19         UserSimilarity similarity = new PearsonCorrelationSimilarity(model);
20
21         UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, model);
22         UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
23
24         return recommender;
25     }
26
27 }

```

APÊNDICE U

Algoritmo de limite de vizinhança

```

1 package br.com.alura;
2
3 import org.apache.mahout.cf.taste.common.TasteException;
15
16 public class RecomendadorDeProdutosBuilder implements RecommenderBuilder {
17
18     public Recommender buildRecommender(DataModel model) throws TasteException {
19         UserSimilarity similarity = new EuclideanDistanceSimilarity(model);
20
21         UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.1, similarity, model);
22         UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
23
24         return recommender;
25     }
26
27 }

```