

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DO RIO GRANDE DO SUL
CAMPUS RESTINGA**

**MASTERMENU: UM SISTEMA DE AUTOATENDIMENTO
PARA CLIENTES DE RESTAURANTES E SIMILARES**

MATHEUS TABARES LOPES

**Porto Alegre
2017**

MATHEUS TABARES LOPES

**MASTERMENU: UM SISTEMA DE AUTOATENDIMENTO
PARA CLIENTES DE RESTAURANTES E SIMILARES**

Trabalho de Conclusão de Curso apresentado,
junto ao Curso de Análise e Desenvolvimento
de Sistemas do Instituto Federal Educação,
Ciência e Tecnologia do Rio Grande do Sul,
como requisito parcial para a obtenção do grau
de Tecnólogo em Análise e Desenvolvimento
de Sistemas.

Orientador: Prof. Rafael Pereira Esteves

**Porto Alegre
2017**

MATHEUS TABARES LOPES

MASTERMENU: UM SISTEMA DE AUTOATENDIMENTO PARA CLIENTES DE RESTAURANTES E SIMILARES

Trabalho de Conclusão de Curso apresentado
como requisito parcial para a obtenção do grau
de Tecnólogo em Análise e Desenvolvimento
de Sistemas.

Orientador: Prof. Rafael Pereira Esteves

Aprovado em MÊS, ANO.

Prof. Rafael Pereira Esteves (orientador)

Prof. Jean Carlo Hamerski – IFRS Campus Restinga

Profa. Eduarda Rodrigues Monteiro – IFRS Campus Restinga

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO SUL

Reitor: Prof. Osvaldo Casares Pinto

Pró-Reitora de Ensino: Prof. Clarice Monteiro Escott

Diretor do Campus Restinga: Prof. Gleison Samuel do Nascimento

Coordenador do CST em Análise e Desenvolvimento de Sistemas: Prof. Rafael Pereira Esteves

Bibliotecária-Chefe do Campus Restinga: Paula Porto Pedone

Dedico este trabalho aos meu filho e mulher,
que entenderam minha ausência e que me
apoiaram, compreendendo a importância desta
conquista para nossa família.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por ter me concedido a vida e com ela a saúde. Por ter me feito justo e forte para conseguir alcançar meus objetivos de forma honesta. Agradeço também a mulher que Ele colocou em minha vida, Caroline. Uma pessoa em quem me espelho por ser um exemplo de perseverança e dedicação em tudo que faz. Te agradeço por estar comigo em todos os momentos, principalmente nos difíceis, por motivar-me e não deixar-me desistir. Agradeço também ao meu filho Davi Henzo, que com seu amor e felicidade me contagia a cada dia, e ao professor Rafael Esteves e colegas Renan Tabares, meu irmão, e Paulo, que estiveram ao meu lado me auxiliando no decorrer do trabalho.

“Buscai, assim, em primeiro lugar, o Reino de Deus e a sua justiça, e todas essas coisas vos serão acrescentadas.”

(Livro de Matheus 6:33)

RESUMO

Muitos dos estabelecimentos alimentícios como restaurantes, bares, lancherias e similares possuem problemas relevantes no dia a dia de seus negócios como ineficiência no atendimento, demora no fechamento de contas e desorganização com os pedidos. Pensando nisso, a proposta deste trabalho é o desenvolvimento de um sistema de autoatendimento para clientes desses estabelecimentos, cujo principal objetivo é solucionar esses problemas rotineiros. Com a opção de controlar as solicitações através de um smartphone e visualizar o cardápio digital, o cliente do estabelecimento poderá personalizar seus pedidos de acordo com as suas necessidades, como também visualizar o status em tempo real de suas solicitações. O sistema também promoverá controle de comandas e setores no momento da produção para otimizar o gerenciamento da loja e segurança para os usuários de forma a criptografar os dados sensíveis, como a senha de autenticação. No final do consumo, o cliente poderá realizar o pagamento de sua conta sem precisar sair do lugar, apenas acessando o sistema pelo celular e acionando o fechamento da conta. O objetivo é promover a satisfação do cliente e uma melhor experiência para os funcionários do estabelecimento.

Palavra-chave: sistema, autoatendimento, restaurantes.

ABSTRACT

Many food businesses such as restaurants, bars, snack bars experience significant problems in their daily operation, such as inefficiency in service, delay in closing accounts, and disorganization with the orders. With this in mind, the main purpose of this work is to develop a self-service system to minimize these problems in food related businesses. With the option to control requests through a smartphone and view the digital menu, a customer can customize their requests according to their needs, as well as check in real-time the status of their requests. The system will also promote the control of orders and the involved sectors during production to optimize management and provide security for users by encrypting sensitive data. At the end of the service, the customer will be able to pay his/her orders at the table by accessing the system through the cell phone and performing the payment. The ultimate goal is to promote customer satisfaction and a better experience for the staff.

Keywords: system, self service, restaurant.

LISTA DE FIGURAS

Figura 1. Módulos do MasterMenu.....	18
Figura 2. Fluxo de funcionamento do MasterMenu.....	19
Figura 3. Diagrama de classes do MasterMenu.....	21
Figura 4. Arquitetura do MasterMenu.....	23
Figura 5. Resultado utilização da tag email do HTML5.....	23
Figura 6. Utilização da tag number do HTML5.....	24
Figura 7. Exemplo da utilização Bootstrap no Mastermenu.....	25
Figura 8. Exemplo da utilização do Bootstrap no Mastermenu.....	25
Figura 9. Utilização do AngularJS.....	26
Figura 10. Código de mapeamento do Controller Comanda do Mastermenu.....	27
Figura 11. Utilização da diretiva ng-app no arquivo index.html do Mastermenu.	27
Figura 12. Configuração do template padrão main.html no arquivo app.config.js	28
Figura 13. Implementação método POST do Spark Framework.....	28
Figura 14. Requisição para salvar comanda no método POST representado figura 12.....	29
Figura 15. Arquivo de configuração persistence.xml para conexão com banco de dados.....	30
Figura 16. Arquivo Connection.java.....	30
Figura 17. Utilização da biblioteca GSON para converter JSON em objeto Java.....	30
Figura 18. Utilização da biblioteca GSON para converter objeto Java em String JSON.....	31
Figura 19. Script de criação e utilização do banco de dados db_mastermenu.....	31
Figura 20. Repositório Mastermenu no Github.....	32
Figura 21. URL para copiar e clonar o repositório localmente.....	32
Figura 22. Comando para clonar repositório do Git na máquina local.....	33
Figura 23. Tela Mastermenu que lista estabelecimentos já cadastrados.....	33
Figura 24. Tela Mastermenu para cadastro de novo estabelecimento.....	34
Figura 25. Tela principal do estabelecimento.....	34
Figura 26. Tela de cardápio de um estabelecimento específico do sistema.....	35
Figura 27. Tela de lista de pedidos do consumidor.....	36
Figura 28. Tela acompanhamento do pedido.....	37
Figura 29. Tela pedidos cozinha.....	37
Figura 30. Tela para escolher mesa a reservar.....	38
Figura 31. Tela de confirmação de reserva.....	39
Figura 32. Tela que lista as mesas do estabelecimento.....	40
Figura 33. Tela que exibe detalhes da reserva do cliente.....	41
Figura 34. Tela que exibe lista de produtos com opção Happy Hour.....	42
Figura 35. Tela Happy Hour.....	42
Figura 36. Cardápio com promoção Happy Hour confirmada para produto Suco de laranja.....	43
Figura 37. Visualização do cardápio com data promoção vencida para produto Suco de laranja.....	44

LISTA DE TABELAS

TABELA 1. COMPARATIVO ENTRE SISTEMAS	17
--	----

SUMÁRIO

1. INTRODUÇÃO.....	13
1.1. PROBLEMA.....	14
1.2. PROPOSTA DE SOLUÇÃO.....	14
2. TRABALHOS RELACIONADOS.....	15
2.1. CARDÁPIO DIGITAL CONSUMER.....	15
2.2. SISTEMA ECOMANDA.....	16
2.3. MENU DIGITAL.....	16
2.4. COMPARATIVO ENTRE SISTEMAS.....	16
3. MODELAGEM DO SISTEMA	17
4. IMPLEMENTAÇÃO	22
4.1. CAMADA FRONT-END.....	23
4.1.1. <i>HTML5</i>	23
4.1.2. <i>BOOTSTRAP</i>	24
4.1.3. <i>ANGULARJS</i>	26
4.2. CAMADA API.....	28
4.2.1. <i>SPARK FRAMEWORK</i>	28
4.2.2. <i>JDK8</i>	29
4.2.3. <i>HIBERNATE JPA</i>	29
4.2.4. <i>GSON</i>	31
4.2.5. <i>BANCO DE DADOS MYSQL</i>	31
4.2.6. <i>REPOSITÓRIO REMOTO GITHUB</i>	32
4.3. PRINCIPAIS FUNCIONALIDADES.....	33
4.3.1. <i>CADASTRO DE ESTABELECIMENTO</i>	34
4.3.2. <i>GERENCIAMENTO DE PEDIDOS</i>	35
4.3.3. <i>RESERVA DE MESA</i>	38
4.3.4. <i>HAPPY HOUR</i>	41
4.4. METODOLOGIA DE DESENVOLVIMENTO.....	44
4.5. ARQUITETURA DO SISTEMA.....	45
4.6. METODOLOGIA DE REQUISITOS.....	45
6. CONSIDERAÇÕES FINAIS.....	46

1. INTRODUÇÃO

Nos últimos tempos, em virtude da crise econômica e do desemprego decorrente da mesma [16], os brasileiros estão despertando seu lado empreendedor como alternativa para superar a crise. Consequentemente novos empreendimentos no ramo alimentício como restaurantes, bares, lancherias e similares estão surgindo. Para se destacar frente à concorrência, esses estabelecimentos têm como foco promover uma experiência positiva para o consumidor, com o objetivo de fidelizar o mesmo e gerar lucro para o negócio.

Porém, muitos destes estabelecimentos encontram problemas relevantes no dia a dia de seus negócios como, por exemplo, a ineficiência no atendimento, demora no fechamento de contas e desorganização com os pedidos. Dessa forma, com o crescimento e popularização das tecnologias e pelo potencial que essas tecnologias possam contribuir para a qualidade do serviço oferecido, o objetivo deste trabalho é desenvolver um sistema voltado para este nicho do mercado, promovendo uma melhor experiência para os consumidores e para todos os envolvidos no negócio.

Já existem sistemas em produção que auxiliam no autoatendimento dos clientes de bares e restaurantes. Alguns desses sistemas oferecem um cardápio digital onde o cliente pode visualizar os produtos oferecidos, mas não pode personalizá-los, isto é, alterar o pedido de acordo com suas preferências. Outros sistemas oferecem um terminal no estabelecimento para controlar os pedidos solicitados. O diferencial do sistema proposto é combinar o que já existe, isto é, o cardápio digital, com o controle dos pedidos solicitados, em uma única plataforma, ou seja, o cliente pode visualizar os produtos e controlar seus pedidos em seu *smartphone* em uma única aplicação. Além disso, não é necessário acessar um sistema diferente para cada estabelecimento frequentado, já que o sistema proposto reúne na mesma plataforma vários estabelecimentos.

Portanto, a proposta deste trabalho é o desenvolvimento de um sistema de autoatendimento para clientes de estabelecimentos alimentícios, denominado MasterMenu, com o objetivo de minimizar os problemas tipicamente encontrados na operação desses estabelecimentos (exemplo: ineficiência no atendimento, desorganização nos pedidos, atrasos). Através da opção de poder controlar as solicitações através de um *smartphone* e visualizar o cardápio digital, o cliente do estabelecimento pode personalizar seus pedidos de acordo com as suas necessidades e pode também visualizar o *status* em tempo real de suas solicitações. O MasterMenu também permite o controle de comandas e setores (restaurante,

bar, etc.) no momento da produção dos pedidos, de forma a otimizar o gerenciamento dos pedidos e oferecer segurança para os usuários, criptografando os dados sensíveis.

Ao final do consumo o cliente pode realizar o pagamento de sua conta sem precisar sair de sua mesa, acessando uma funcionalidade de fechamento de conta. Dessa forma, pretende-se promover a satisfação do cliente e uma melhor experiência para os funcionários do estabelecimento.

1.1.Problema

O ano de 2016 foi um ano conturbado para bares e restaurantes. De acordo com dados da Abrasel (Associação Brasileira de Bares e Restaurantes) em Curitiba, por exemplo, houve redução de 30% no movimento deste tipo de estabelecimento e o ticket médio, ou seja, o valor médio que um cliente gasta em um restaurante também caiu [1].

Segundo Karina Muniz R. Dantas [2] consultora do Sebrae especializada no segmento de restaurantes, um dos motivos principais para esta queda no movimento é “a falta de pessoas qualificadas na equipe, pois em grande parte dos estabelecimentos no ramo alimentício e de bebidas, os clientes são dependentes dos funcionários (como garçons) para anotar seus pedidos e entregar a produção. Além disso, o cliente também não tem controle sobre suas solicitações, como por exemplo, se o seu pedido já está em andamento ou aguarda na fila e quanto tempo vai demorar para chegar em sua mesa”.

Essa falta de controle sobre os pedidos e a dependência de funcionários desqualificados para solicitar um serviço, vão de impacto na insatisfação dos clientes e por consequência no movimento desses estabelecimentos.

1.2.Proposta de Solução

A proposta de solução para o problema citado acima consiste no desenvolvimento de uma aplicação para que o consumidor de bares e restaurantes não dependa exclusivamente de um garçom para realizar a solicitação de seus pedidos. Esta aplicação foi denominada MasterMenu. Com o MasterMenu, o cliente pode obter informações sobre o andamento dos seus pedidos através do seu *smartphone* além de poder personalizá-los, característica que

muitas vezes é oferecida para o consumidor, porém de forma manual e demorada.

O consumidor pode realizar solicitações após realizar um cadastro simples e estar presencialmente dentro do estabelecimento, pois só assim ele terá acesso ao código do estabelecimento, que será oferecido por um funcionário do mesmo.

O cadastro do cliente é composto por e-mail e senha que serão salvos em uma base de dados. Para sua segurança a senha será criptografada.

Além deste capítulo introdutório esta monografia está organizada da seguinte forma. O Capítulo 2 apresenta os principais trabalhos relacionados ao sistema proposto. O Capítulo 3, mostra e descreve a modelagem do sistema, como também o fluxo do mínimo produto aceitável e sua descrição. No capítulo 4 está descrito a implementação e tecnologias utilizadas para o sistema proposto.

2. TRABALHOS RELACIONADOS

Como primeira etapa do trabalho, analisou-se uma série de sistemas na internet que atualmente estão disponíveis no mercado, que contemplam, pelo menos parcialmente, a proposta do trabalho.

Para realizar a pesquisa de sistemas já disponíveis na internet, que sejam semelhantes ao proposto, foi pesquisado primeiro sobre trabalhos acadêmicos relacionados ao assunto, porém não foi encontrado. Após essa pesquisa inicial, focou-se em sistemas utilizados por estabelecimentos da área alimentícia. A partir disso foi coletado uma série de informações sobre esses sistemas disponibilizados pelos próprios sites.

A seguir os três principais sistemas encontrados são discutidos e comparados entre si e com o MasterMenu.

2.1. Cardápio Digital Consumer

O Cardápio Digital Consumer é um módulo de uma plataforma digital para gerenciamento de bares e restaurantes [3]. Através de um *QR-Code*, o cliente o acessa na mesa do estabelecimento, pelo próprio celular, realiza pedidos, pode chamar o garçom e fazer uma avaliação ao final do atendimento.

O cardápio digital trabalha integrado ao programa *Consumer*, que permite o controle dos pedidos, controle financeiro, emissão de cupom fiscal em uma solução completa e focada na área da alimentação. Para adquirir o sistema o cliente tem direito a 30 dias grátis para testar as funcionalidades do programa. Após esse período ele deve escolher entre três planos:

- 1ºPlano: com direito a um computador e com preço de 12x de R\$29,90, o cliente (estabelecimento) tem direito a Controle de Pedidos, Controle do Caixa, Gerenciamento Financeiro, Controle do Fiado, Múltiplas Impressoras, Controle de Estoque, Bina Identificador Chamadas, Painel de Senha.
- 2ºPlano: por 12x de R\$39,00, o cliente (estabelecimento) tem os mesmos direitos do primeiro plano com um adicional de computadores ilimitados dentro da rede local.
- 3ºPlano: por 12x de R\$68,00 o cliente tem os mesmos direitos do segundo plano com um adicional de Comanda Mobile, Cardápio Digital, Monitor de Preparo.

2.2. Sistema EComanda

O EComanda [4] é um sistema digital que oferece gestão para restaurantes. Dentre suas funcionalidades, possui a Comanda Eletrônica para o funcionário do estabelecimento gerir os pedidos das mesas através de um smartphone.

Possui também o Cardápio Digital que tem a ideia de os clientes solicitarem seus pedidos pelo próprio aparelho, e através do sistema para restaurante, o estabelecimento terá acesso aos dados da clientela para realizar um pós-venda. Entretanto, o cliente terá que manter um aplicativo para cada estabelecimento em seu celular.

2.3. Menu Digital

Eleito pelo Jornal de São o melhor Cardápio Digital do Brasil, é uma empresa 100% brasileira que atua em 15 países.

Sem o cliente precisar instalar qualquer programa em seu celular, o Menu Digital [5] é acessado pelo leitor QR-Code que se encontra em cima das mesas do estabelecimento, e também pela URL do menudigital.com.br. Toda a customização do menu é feita por um painel do cliente.

Apesar de ser eleito o melhor cardápio digital do Brasil, com funcionalidades de harmonização e televisores *board*, que possibilitam o marketing dos produtos, onde a loja pode inserir propagandas. O Menu Digital não oferece ao cliente a opção de solicitar os pedidos, e tudo funciona de forma online.

Contudo o sistema oferece um cartão de visitas ao cliente, promovendo o conhecimento do espaço, dos integrantes do estabelecimento e seus produtos.

2.4. Comparativo entre sistemas

A Tabela 1 apresenta uma comparação entre os três sistemas citados e o MasterMenu.

Tabela 1. Comparativo entre sistemas

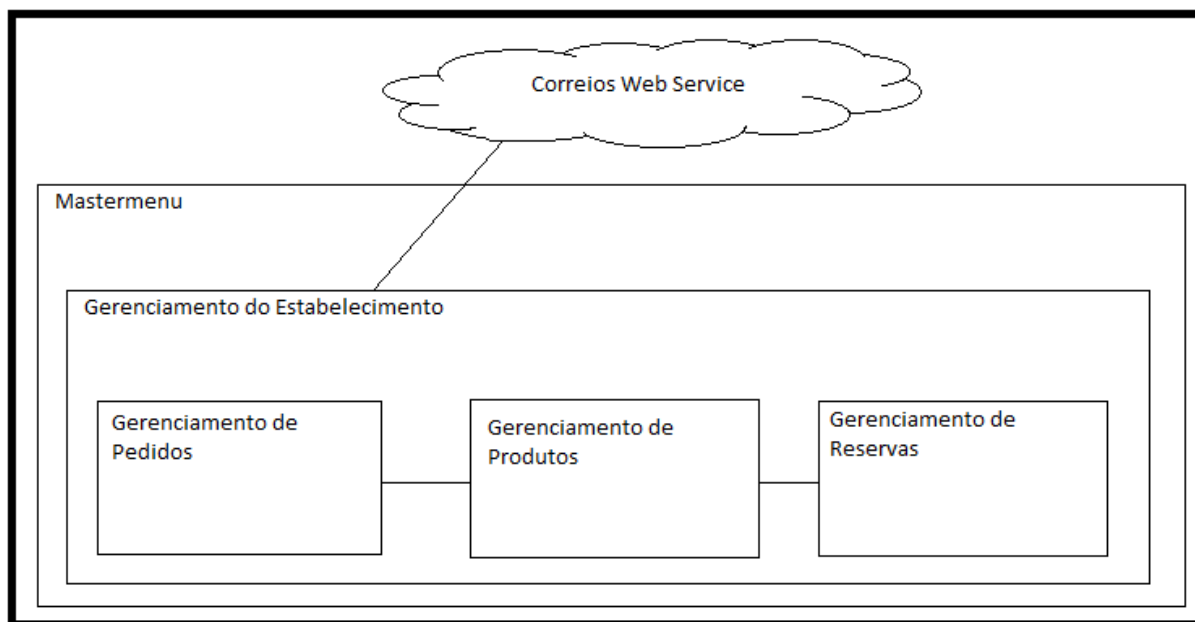
	MasterMenu	Menu Digital	Sistema eComanda	Cardápio Digital Consumer
Reserva do Cliente	SIM	NÃO	NÃO	NÃO
Plataforma Única	SIM	SIM	NÃO	SIM
Pedidos Offline	SIM	NÃO	NÃO	SIM
Opção Happy Hour	SIM	NÃO	NÃO	NÃO
Personalizar Pedido	SIM	NÃO	NÃO	NÃO
Busca por CEP	SIM	NÃO	SIM	SIM
Layout Responsivo	SIM	SIM	SIM	SIM
Acesso QR-Code	NÃO	SIM	NÃO	SIM
Status pedido	SIM	NÃO	NÃO	NÃO

Fonte: próprio autor.

Com base na Tabela 1, verificou-se que existem aplicações que satisfazem de certa forma a necessidade do consumidor gerenciar sua própria comanda.

Porém, pode-se perceber que, por exemplo, o Cardápio Digital Consumer, que é um dos sistemas mais completos, não contempla algumas funcionalidades que poderiam aumentar significativamente a satisfação do cliente, como a reserva do cliente, onde o cliente poderá reservar um local no estabelecimento com antecedência, como também já realizar o seu pedido e colocar nas observações em qual horário pretende chegar no estabelecimento. Essa funcionalidade é oferecida pelo MasterMenu. Outra função importante que o MasterMenu oferece é a personalização dos pedidos, onde o cliente pode escolher outras opções para alterar seu pedido (incluindo ou removendo itens), desde que habilitada pelos responsáveis pelo estabelecimento.

Além disso, o MasterMenu contém a opção *Happy Hour*, que permite a inclusão/remoção de promoções com duração específica e de forma automatizada, evitando que o estabelecimento tenha prejuízos financeiros caso o responsável pelo estabelecimento,



por algum motivo, esqueça de atualizar o valor original dos itens do cardápio.

3. MODELAGEM DO SISTEMA

A seguir, a Figura 1 ilustra os componentes principais do MasterMenu.

Figura 1. Módulos do MasterMenu

Fonte: Próprio autor.

Na Figura 1 é possível observar que o MasterMenu é composto basicamente por quatro módulos principais: Gerenciamento do Estabelecimento, Gerenciamento de Pedidos, Gerenciamento de Produtos e Gerenciamento de Reservas; além da integração com o *Web Service dos Correios*.

No módulo de Gerenciamento do Estabelecimento o usuário pode criar, editar, listar e excluir estabelecimentos. A integração com o *Web Service dos Correios* é feita neste módulo, no qual buscará por um determinado CEP o endereço do usuário. A partir da criação de um estabelecimento diversas outras funcionalidades são habilitadas. Essas funcionalidades são destacadas a seguir.

O módulo de Gerenciamento de Pedidos é o módulo principal do sistema onde o usuário pode, a partir de um cardápio criado no estabelecimento, incluir, editar, listar e excluir produtos em sua lista de pedidos. Após a inclusão dos produtos em sua lista, o mesmo pode enviar o pedido para a produção. Cada pedido faz parte de uma categoria, como, por exemplo, um prato feito faz parte da categoria “comidas”, que ao ser solicitada para produção, cairá na fila de execução dos pedidos da cozinha. Outra categoria seria “bebidas” cujos pedidos são

encaminhados para o bar (ou setor equivalente). Ao chegar a informação de um novo pedido solicitado na cozinha, o MasterMenu registra o horário de chegada do pedido e faz uma estimativa do tempo de conclusão do pedido conforme o tamanho da fila de pedidos. O responsável pelos pedidos da cozinha realizará sua tarefa e por fim entregará o produto ao responsável pela entrega até a mesa do cliente. Este funcionário terá a responsabilidade de finalizar o pedido no sistema, mas caso não o faça, o cliente ao realizar o fechamento da conta poderá encerrar o pedido. O cliente poderá acompanhar o fluxo do seu pedido em tempo real no seu *smartphone*. Além disso o cliente poderá também solicitar pedidos antes de chegar ao estabelecimento, mediante pagamento antecipado.

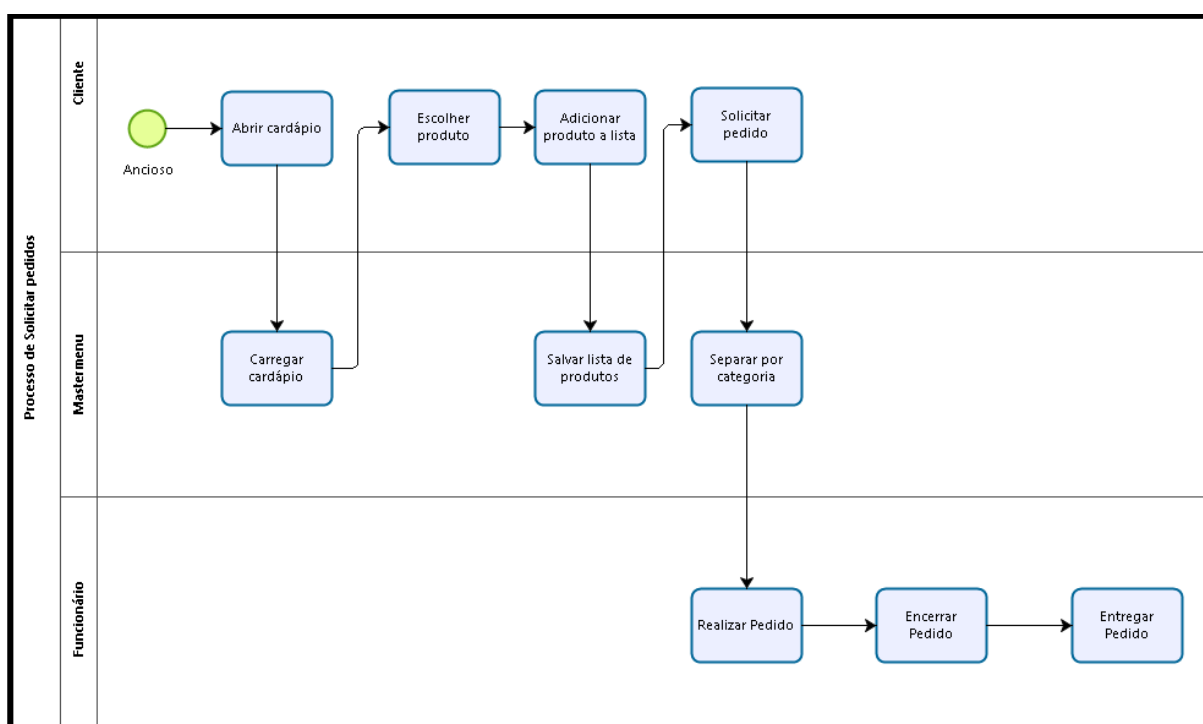


Figura 2. Fluxo de funcionamento do MasterMenu

Fonte: Próprio autor.

No módulo de Gerenciamento de Produtos, após a criação de um estabelecimento, o usuário poderá criar, editar, listar e excluir produtos e suas composições e também uma lista de opções para serem adicionadas a determinado produto, promovendo assim a personalização do pedido. Neste módulo, o usuário poderá também criar promoções específicas que serão publicadas somente em datas pré-determinadas.

No Gerenciamento de Reservas o cliente poderá criar uma reserva em uma mesa específica, antecipadamente, com data marcada e número de pessoas estimada.

A Figura 2 ilustra um diagrama de atividades que descreve o fluxo de funcionamento

básico do MasterMenu.

Como pode-se observar na Figura 2, o cliente só poderá iniciar o processo de solicitação de pedido depois de estar autenticado no MasterMenu. Após a autenticação, MasterMenu listará para o cliente os estabelecimentos cadastrados. Por sua vez, o cliente escolherá um dos estabelecimentos listados. Quando o cliente entrar na página do estabelecimento o MasterMenu mostrará o cardápio do mesmo. Assim, o cliente poderá escolher os produtos de sua preferência, além de poder personalizar o produto com as opções previamente inseridas pelo estabelecimento e adicionar os produtos à sua lista de pedidos. Ao adicionar os produtos à sua lista, a mesma é salva de forma temporária, expirando caso o cliente saia da página do estabelecimento. O cliente poderá especificar a quantidade dos produtos, tanto no cardápio quanto na sua lista de pedidos. Na lista, ele poderá também excluir seus pedidos antes de enviar para a produção. Após a lista estar de acordo com o desejo do cliente, o mesmo acionará o botão “solicitar” e seus pedidos serão enviados para produção. Após o envio do pedido, o MasterMenu classifica os produtos em categorias, por exemplo, “comida” ou “bebida” e envia cada produto para o seu respectivo setor (cozinha, bar). Chegando na produção, o pedido entrará em uma fila, cuja ordem é baseada no horário de cada solicitação. Ao ser encerrado o pedido pela produção, a mesma entregará o produto para o garçom, que por sua vez, encerrará o pedido no sistema e entregará ao seu respectivo cliente.

Os pedidos do cliente, quando enviados para produção, ficam em modo de visualização, que mostra o horário da solicitação, tempo estimado para o preparo, calculado com base na fila de produção, e um ícone de relógio que será visível também para os funcionários da produção. Caso o pedido não tenha sido finalizado antes do tempo estimado, o ícone ficará em modo urgente, até que o garçom encerre o pedido.

A seguir, a Figura 3 ilustra o diagrama de classes.

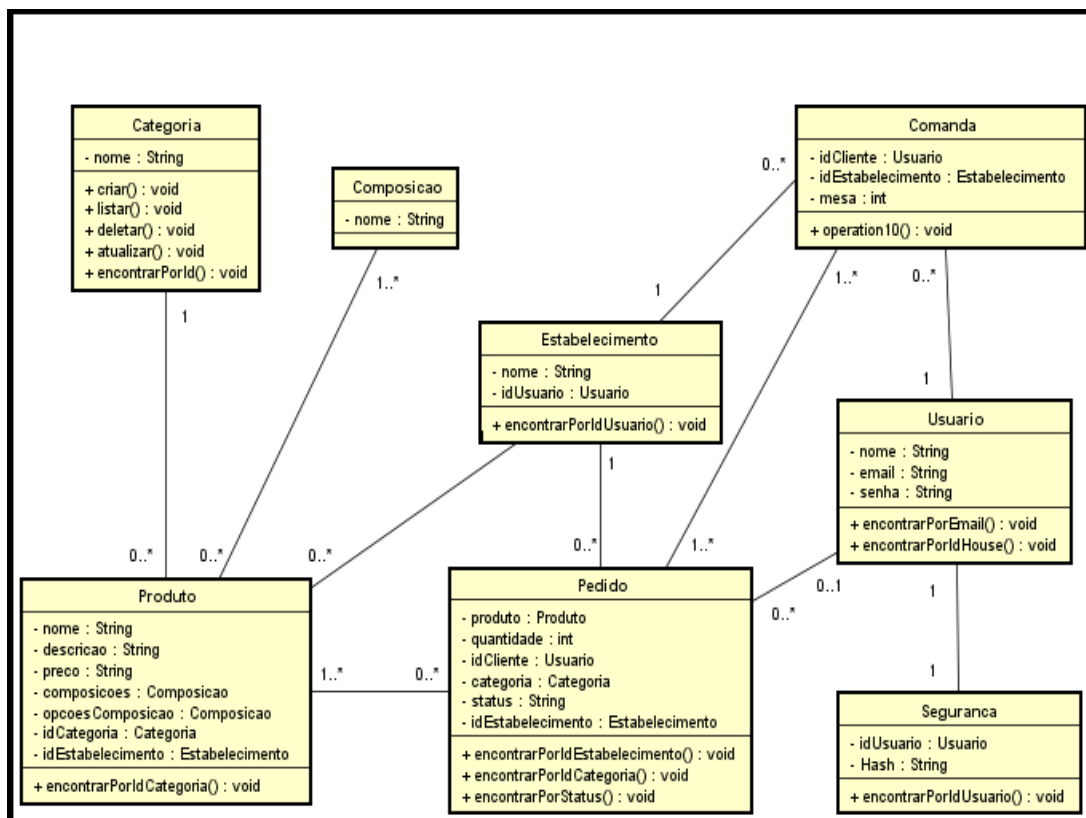


Figura 3. Diagrama de classes do MasterMenu

Fonte: Próprio autor.

O diagrama de classes é composto pelas principais classes do sistema, sendo elas, Produto, Pedido, Composição, Categoria, Estabelecimento, Comanda, Usuário e Segurança. Todas possuem identificador único e as funcionalidades de criar, listar, atualizar, excluir e encontrar por identificador único do objeto (id). Em cada classe foi inserido na parte inferior, onde é descrito os métodos, somente as funcionalidades diferentes das que já existem para cada classe, com o intuito de tornar o diagrama menor e mais limpo.

A classe Produto é o modelo para manter os dados dos serviços oferecidos pelo estabelecimento. Uma instância de um Produto deverá ter relacionamento somente com uma única Categoria e deverá ter pelo menos uma Composição, podendo ter mais se necessário. A classe Produto também tem relacionamento com as classes Estabelecimento e Pedido, que respectivamente, deverá ter relacionamento com um único Estabelecimento e poderá ter relacionamento com nenhum ou muitos Pedidos.

A classe Composição é o modelo para os itens que compõem um produto. Ela pode se relacionar com nenhum ou muitos Produtos.

A classe Categoria é o modelo para os agrupamentos de produtos. Ela pode se relacionar com nenhum ou muitos Produtos.

A classe Pedido é o modelo para as solicitações dos consumidores dos estabelecimentos. Ela pode ser relacionar com as classes Produto, Estabelecimento e Comanda. Uma instância de um Pedido deverá ter um único Estabelecimento, deverá se relacionar com pelo menos um Produto e uma Comanda e também poderá se relacionar com um único Usuário.

A classe Estabelecimento é o modelo responsável por manter os dados da loja física. Seu relacionamento com as classes Comanda, Pedido e Produto é de 0...*, ou seja, uma instância de estabelecimento pode ter nenhum ou muitos pedidos, comandas e produtos.

A classe Comanda é o modelo responsável basicamente por manter todos os pedidos dos consumidores. Uma instância de comanda deve ter um único usuário e um único estabelecimento, e pelo menos um pedido.

A classe Usuario é o modelo responsável por manter os dados dos usuários do sistema. Uma instância de usuário poderá ter nenhuma ou muitas comandas, nenhum ou muitos pedidos e deverá ter uma única segurança.

A classe Segurança é responsável por manter a chave de criptografia para cada usuário.

4. IMPLEMENTAÇÃO

Diferentemente de Web Services baseados no protocolo SOAP, que tem como objetivo estabelecer um protocolo para a comunicação de objetos e serviços, a arquitetura escolhida para o projeto foi a arquitetura REST [6], que nasceu juntamente com o protocolo HTTP 1.1 e, visa a utilização correta dos métodos deste protocolo (GET, POST, PUT, HEAD, OPTIONS e DELETE) para criar serviços que poderiam ser acessados por qualquer tipo de plataforma, tais como tablets, televisores, notebooks e smartphones.

A escolha por esta arquitetura também procura seguir as tendências mais atuais no mercado de trabalho da tecnologia da informação. A seguir, a Figura 4 ilustra de forma geral a arquitetura do MasterMenu, organizada em camadas.

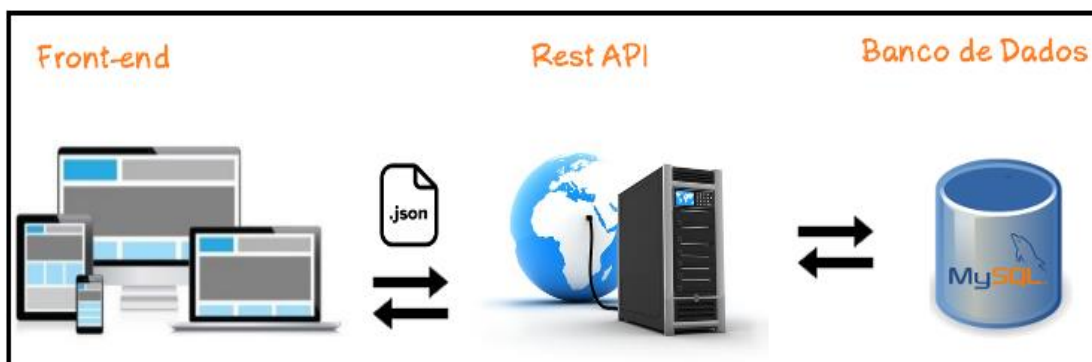


Figura 4. Arquitetura do MasterMenu

Fonte: Próprio autor.

4.1. Camada Front-end

Nesta camada, se concentra a parte visual do sistema (*front-end*) e as tecnologias escolhidas serão descritas a seguir.

4.1.1. HTML5

O HTML 5 [7] é a mais recente evolução que define o HTML com novos atributos e foi escolhida por ser a última versão e também por sua melhoria com validação e restrição com formulários web. Utilizada para a construção de páginas web.

Marcação semântica para utilização da tecnologia: `<!DOCTYPE html>`, escrita no arquivo *Mastermenu/scr/main/resources/index.html*.

A seguir, as Figuras 5 e 6 respectivamente, apresentam a utilização dos novos componentes `<input type="email">` e `<input type="number">`, como exemplo de utilização da tecnologia.

A imagem mostra a interface de login do MasterMenu. O formulário contém campos para "E-mail" e "Senha". O campo de E-mail utiliza a tag `<input type="email">` e contém o texto "matheus@gmail.com". O campo de Senha utiliza a tag `<input type="password">` e contém pontos para ocultar o texto. Abaixo dos campos, há dois botões: "Entrar" e "Voltar".

Figura 5. Resultado utilização da tag *email* do HTML5

Fonte: Próprio autor.

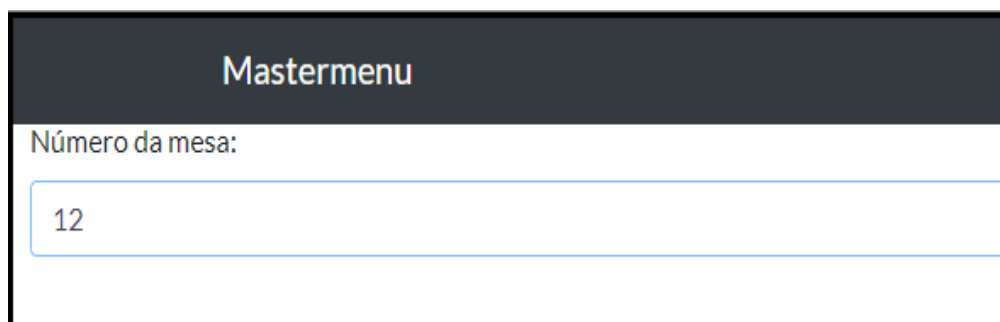
The image shows a web form with a dark header bar containing the text 'Mastermenu'. Below the header, there is a label 'Número da mesa:' followed by a text input field. The input field contains the number '12'. The form is styled with a light background and a thin border.

Figura 6. Utilização da tag *number* do HTML5

Fonte: Próprio autor.

A utilização das tags mencionadas auxiliam no aprimoramento da validação, que no caso das Figuras 5 e 6, respectivamente, seria a obrigatoriedade da inserção de um E-mail e números.

4.1.2. Bootstrap

O Bootstrap [8] é o mais popular entre os frameworks para HTML, CSS e Javascript. O objetivo é tornar a experiência visual do usuário mais agradável, pelo fato do framework ser focado para dispositivos móveis na web, ele tem como funcionalidade tornar as páginas HTML responsivas, ou seja, independentemente do tamanho da tela que o usuário acesse a aplicação se adequa perfeitamente.

Para iniciar com Bootstrap, foi realizado o download do pacote CSS no site oficial do Framework <http://getbootstrap.com/> e inserido numa pasta específica de folhas de estilo dentro da aplicação.

A seguir, a Figura 7 exemplifica a utilização dos componentes Bootstrap dentro da aplicação e o resultado na tela do navegador na Figura 8 respectivamente.

```

<table class="table table-striped">
  <tr>
    <th>Nome</th>
    <th>Opções</th>
  </tr>
  <tr ng-repeat="house in houses">
    <td>{{house.name}}</td>
    <td><button type="button" class="btn btn-info" ng-
click="actionUpdate(house)">Editar</button>
      <button type="button" class="btn btn-info" data-
toggle="modal" data-target="#modalDeleteProduct" ng-
click="actionDelete(house)">Excluir</button>
      <button type="button" class="btn btn-info" ng-
click="moreOptions(house)">+</button>
    </td>
  </tr>
</table>

```

Figura 7. Exemplo da utilização Bootstrap no Mastermenu

Fonte: Próprio autor.

Percebe-se a inserção da `class="table table-striped"` na tag `<table>` e da `class="btn btn-info"` dentro da tags `<button>`. Isso é a utilização do Bootstrap na prática. As propriedades `table`, `table-striped`, `btn` e `btn-info` são estilos próprios que geram o resultado mostrado na figura 8 a seguir.

Mastermenu	
Nome	Opções
Papa lanches	<div> <div>Editar</div> <div>Excluir</div> <div>+</div> </div>
<div> <div>Voltar</div> <div>Novo Estabelecimento</div> </div>	

Figura 8. Exemplo da utilização do Bootstrap no Mastermenu

Fonte: Próprio autor.

A tabela na figura a cima com distinção de cores cinza claro e branco entre as linhas e os botões levemente arredondados e azuis, que ao passar o cursor do mouse por cima altera sua cor para um azul mais forte, como mostra no *botão Editar* da segunda linha.

4.1.3. AngularJS

O AngularJS [9] é um Framework Javascript que roda dentro do navegador web, cujo principal objetivo de sua utilização é o desenvolvimento de interfaces dinâmicas. O mesmo foi escolhido, por possuir uma série de características interessantes. A primeira é que o mesmo foi desenvolvido para ser utilizado com aplicações que possuem arquitetura *RESTful*. Apesar de poder utilizá-lo com outras arquiteturas de aplicações, ele é todo otimizado para o tipo proposto do sistema desenvolvido. A seguir, a Figura 9 ilustra sua utilização.

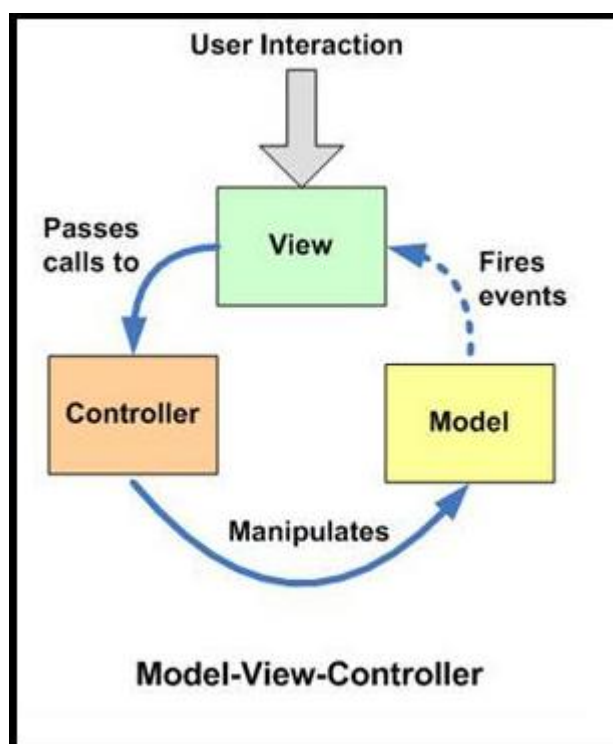


Figura 9. Utilização do AngularJS

Fonte: <https://stackoverflow.com/questions/21216729/combining-mvvm-and-mvc-in-a-picture>.

O AngularJS é baseado no modelo MVC (*Model*, *View*, *Controller*). Este framework possui uma forma de sincronizar simultaneamente os modelos de dados entre suas camadas (*View* e *Controller*) chamado *Two-way Data Bind*. Desta forma, as alterações na *View* são refletidas na fonte de dados e atualizações na fonte refletem na *View*, sem a necessidade de manipulação explícita do DOM (Document Object Model, ou em português, Modelo de Documento Objeto).

O *Controller* de cada página HTML na aplicação proposta é mapeado dentro do arquivo *app.config.js* através da funcionalidade *\$routeProvider* nativa do *AngularJS*, onde é nomeado o *template*, o *controller* e a rota para redirecionar.

A seguir, a Figura 10 mostra uma parte do código de mapeamento do *Controller* responsável por gerir os serviços para as comandas do estabelecimento.

```
angular.module('mastermenuModule', [ 'ngRoute',
  'mastermenuControllers', 'mastermenuServices' ])
  .config([ '$routeProvider', function($routeProvider)
    {
      $routeProvider.when('/commands/:idHouse', {
        templateUrl :
        '../view/registration/commands.html',
        controller : 'CommandsCtrl'
      })
    }
  ])
```

Figura 10. Código de mapeamento do Controller Comanda do Mastermenu

Fonte: Próprio autor.

Percebe-se no código mostrado na figura anterior, que a configuração da rota para o arquivo HTML *comanda.html* inicia-se com a chamada do método *module* do Angular, passando por parâmetro o nome do *app* que foi descrito na diretiva *ng-app* no arquivo HTML principal do sistema (Figura 11). Logo após, entre colchetes ([]), são injetadas as dependências, que são: *ngRoute* que é o modulo responsável por fornecer recursos para a configuração das rotas, *mastermenuControllers* e *mastermenuServices* que são módulos próprios do sistema, que contemplam a responsabilidade de controlar e gerir lógica de negócio e serviços para consultar a aplicação REST respectivamente.

```
<!DOCTYPE html>
<html ng-app="mastermenuModule">
<head>
<title>Master-menu</title>
<meta charset="UTF-8">
```

Figura 11. Utilização da diretiva *ng-app* no arquivo *index.html* do Mastermenu

Fonte: Próprio autor.

A aplicação desenvolvida é um SPA (*Single Page Application*), ou seja, existe uma página principal chamada *index.html* que carrega *templates* para seu *body* conforme a navegação do usuário. Entretanto, ao iniciar a aplicação é sempre carregado dentro deste, um *template default* chamado *main.html* que foi configurado no *app.config.js*, como ilustrado na Figura 12, a seguir.

```
.when('/main', {
  templateUrl : '../view/client/main.html',
  controller : 'MainCtrl'
})
.otherwise({
  redirectTo : '/main'
});
```

Figura 12. Configuração do template padrão main.html no arquivo app.config.js

Fonte: Próprio autor.

Quando o *template main.html* for renderizado, o responsável por sua lógica e regras de negócio, é o controlador *MainCtrl* mapeado na chave *controller* a baixo da chave *templateUrl* no código a cima.

O *MainCtrl* é implementado dentro do arquivo *resources/public/js/controllers/controller.js* na aplicação.

4.2. Camada API

As tecnologias utilizadas para esta camada são descritas a seguir.

4.2.1. Spark Framework

O Spark Framework [10] foi escolhido pela sua facilidade de mapear rotas com o protocolo HTTP, o que facilita a implementação de uma aplicação baseada nos princípios do desenvolvimento de uma aplicação REST.

São necessários somente dois passos para realizar o mapeamento das *URLs*. O primeiro passo é adicioná-lo como uma dependência dentro do arquivo *pom.xml*. O segundo passo é implementar o método, com base nos métodos do protocolo HTTP. Como exemplo, a seguir é mostrado a implementação de um dos métodos POST da aplicação.

```
post(mastermenu + "/commands", (req, res) -> {
  String body = req.body();
  Commands c = parseCommandsFromBody(body);
  if(c != null) {
    commandsService.create(c);
    return gson.toJson("Comanda adicionada a lista!");
  }
  res.status(404);
  return "Comanda não inserida!";
});
```

Figura 13. Implementação método POST do Spark Framework

Fonte: Próprio autor.

No código exibido na Figura 13 a URL */commands* é mapeada e, ao ser acessada, enviará uma requisição para este método.

A seguir, a Figura 14 mostra o código dentro de um *Controller* que faz a requisição para o método no *back-end* da aplicação.

```
$http.post("mastermenu/v1/commands", $scope.commands).success(
    function(data) {
    });
```

Figura 14. Requisição para salvar comanda no método POST representado figura 12.

Fonte: Próprio autor.

Esse método dentro do *Controller* é chamado através da tela *Meus Pedidos*. Ao acionar o botão *SOLICITAR* o método acima é chamado para criar uma comanda para o consumidor, consequentemente acionando o método mapeado através do *Spark*. Pela simplicidade da ferramenta, ela foi escolhida para o mapeamento de serviços na aplicação.

4.2.2. JDK 8

O JDK (Java Development Kit) versão 8 [11] foi escolhido essencialmente pela opção da ferramenta *SPARK FRAMEWORK* que trabalha com essa versão e todos seus tutoriais integrados com a linguagem Java, são focados com implementações de métodos *lambdas* (recurso do JDK 8), como demonstrado na Figura 13 em subseção 4.2.1. *SPARK FRAMEWORK*.

Presente em toda parte do *back-end*, o Java 8 será utilizado também quando trabalhado com datas, por possuir uma nova API de datas, disponibilizada no pacote *java.time*.

4.2.3. Hibernate JPA

O Hibernate [12] é responsável por realizar a persistência dos dados, já que ele se aplica a banco de dados relacionais via JDBC. O Hibernate é o líder no mercado de ferramentas que trabalham com *Mapeamento Objeto-Relacional*.

Para realizar a conexão com a base de dados utilizada, foi configurado um arquivo dentro da aplicação chamado *persistence.xml* mostrado na Figura 15 a seguir.

```

<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
    http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
  version="2.0">

  <persistence-unit name="db_mastermenu" transaction-type="RESOURCE_LOCAL">
    <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>

    <class>br.com.mastermenu.composition.model.Composition</class>

    <properties>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
      <property name="javax.persistence.jdbc.url"
        value="jdbc:mysql://localhost:3306/db_mastermenu?useTimezone=true&serverTimezone=UTC" />
      <property name="javax.persistence.jdbc.user" value="root" />
      <property name="javax.persistence.jdbc.password" value="" />

      <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL57Dialect" />
      <property name="hibernate.show_sql" value="true" />
      <property name="hibernate.format_sql" value="true" />
      <property name="hibernate.hbm2ddl.auto" value="update" />
    </properties>
  </persistence-unit>
</persistence>

```

Figura 15. Arquivo de configuração persistence.xml para conexão com banco de dados

Fonte: Próprio autor

Este arquivo é responsável basicamente por realizar a conexão com a base de dados. As propriedades desse arquivo de maneira geral é o Driver utilizado, a URL de acesso, usuário e senha para autenticação e tipo de dialeto utilizado.

A seguir, a Figura 16 apresenta a classe que realiza a inicialização do arquivo *persistence.xml*

```

package br.com.mastermenu.util;

import javax.persistence.EntityManager;

public class Connection {
    private static EntityManagerFactory ENTITY_MANAGER_FACTORY;

    public static EntityManager connection() {
        ENTITY_MANAGER_FACTORY = Persistence.createEntityManagerFactory("db_mastermenu");
        return ENTITY_MANAGER_FACTORY.createEntityManager();
    }
}

```

Figura 16. Arquivo Connection.java

Fonte: Próprio autor

Esta classe é responsável por criar a entidade de gerenciamento a partir do arquivo de conexão *persistence.xml* que é acessado a partir do nome “db_mastermenu” que é inserido

como parâmetro no método `createEntityManagerFactory("db_mastermenu")` da classe `Persistence`. O arquivo `persistence.xml` só é encontrado por esta classe porque seu nome foi explicitamente descrito na propriedade `name` da tag `<persistence-init>` no mesmo arquivo.

4.2.4. GSON

O GSON [13] é uma biblioteca do Google cuja finalidade é converter objetos em JSON e vice-versa. A seguir um exemplo da utilização dentro do sistema.

```
private static Commands parseCommandsFromBody(String body) {
    //log.info(body);
    Gson gson = new Gson();
    return gson.fromJson(body, Commands.class);
}
```

Figura 17. Utilização da biblioteca GSON para converter JSON em objeto Java

Fonte: Próprio autor

Este exemplo de utilização é para a conversão de um JSON em um objeto específico da aplicação Java chamado `Commands.java`

No exemplo a seguir é o processo inverso, onde a finalidade da biblioteca é converter o objeto Java em JSON para responder ao navegador web.

```
get(mastermenu + "/commands/:id", (req, res) -> {
    int id = Integer.parseInt(req.params(":id"));
    Optional<Commands> c = commandsService.readById(id);
    if (c.isPresent()) {
        return gson.toJson(c.get());
    } else {}
}
```

Figura 18. Utilização da biblioteca GSON para converter objeto Java em String JSON

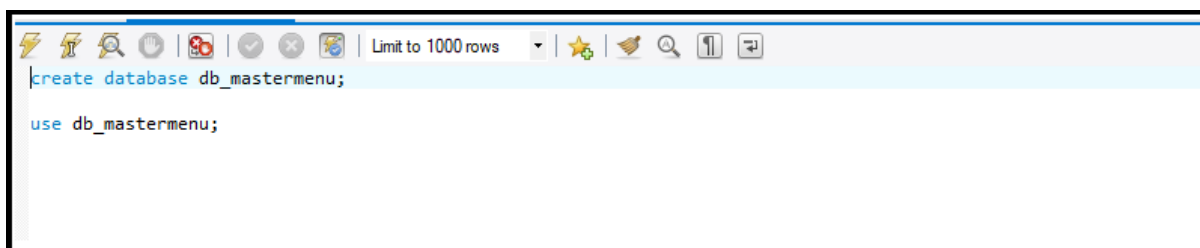
Fonte: Próprio autor

No `return` do `if` o objeto `c` do tipo `Commands` é convertido em `JSON`, caso o mesmo contenha algum conteúdo.

4.2.5. Bando de dados MySQL

A escolha pelo Sistema de Gerenciamento de Banco de Dados (SGBD) MySQL [14] foi motivada por se tratar de um SGBD gratuito, de fácil uso, multiplataforma e se adequar às necessidades do sistema proposto.

Foi necessária a implementação de um *script* para a criação da base de dados chamada *db_mastermenu*, a mesma que foi utilizada no arquivo *persistence.xml*. A seguir, o *script* de criação da base de dados *db_mastermenu*.



```

create database db_mastermenu;

use db_mastermenu;

```

Figura 19. Script de criação e utilização do banco de dados *db_mastermenu*

Fonte: Próprio autor

4.2.6. Repositório remoto GitHub

Todo o trabalho de conclusão de curso se encontra no repositório GitHub [15], que foi escolhido por oferecer facilidades de compartilhamento e controle de versão de arquivo.

Para criar um repositório no GitHub, foi preciso realizar o cadastro no site do GitHub (<https://github.com/>) e após o cadastro, entrar no sistema e realizar a criação de um novo repositório chamado Mastermenu, como mostrado na Figura 20 a seguir.

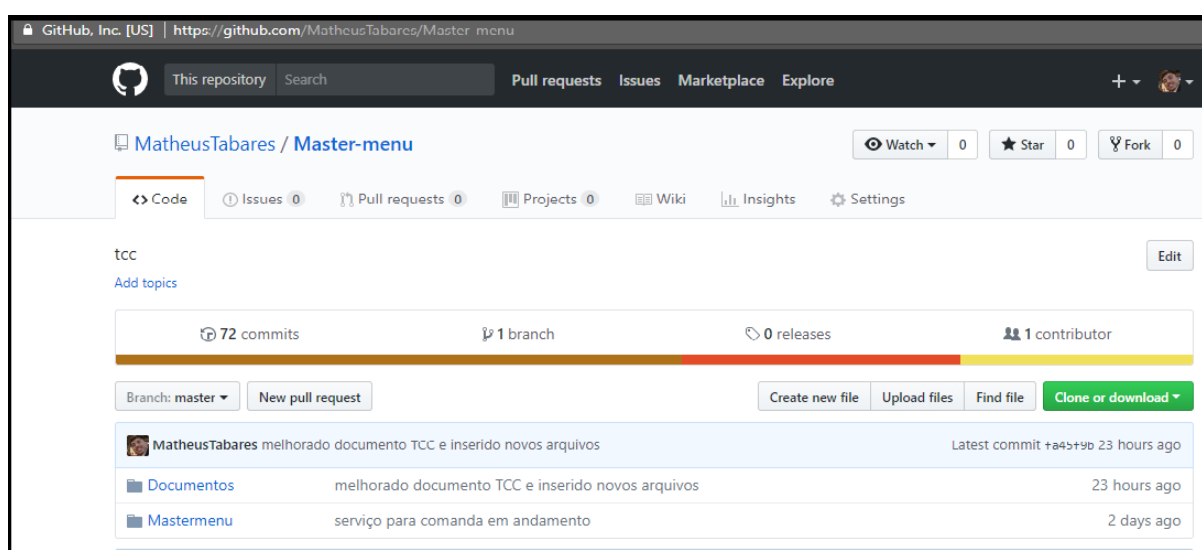


Figura 20. Repositório Mastermenu no Github

Fonte: Próprio autor

Após sua criação, foi necessário criar um repositório local através da ferramenta Git Bash que pode ser encontrada pelo link: <https://git-scm.com/downloads>.

Para realizar a criação do repositório local a partir do repositório remoto é necessário copiar a chave do repositório remoto como a mostra a figura a seguir.

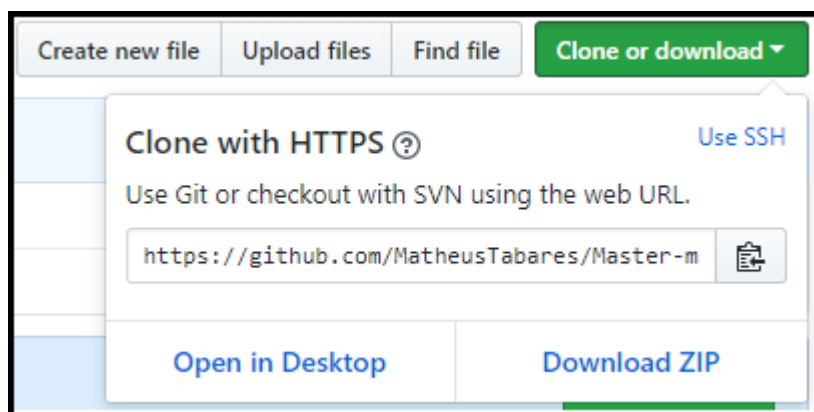


Figura 21. URL para copiar e clonar o repositório localmente

Fonte: Próprio autor

Para criar efetivamente o repositório local é necessário executar o seguinte comando no Git Bash (em um sistema Windows).

```
-git clone https://github.com/MatheusTabares/Master-menu.git
```

Figura 22. Comando para clonar repositório do Git na máquina local

Fonte: Próprio autor

A partir deste comando, o Git cria o repositório local. Caso haja alguma alteração no repositório local, estas podem ser verificadas através do comando *git status* que listará todas as alterações feitas. Para persistir essas alterações é necessário enviar as mesmas para um lugar temporário do git chamado *HEAD*. Para isso, o comando a ser executado no Git Bash é *git commit -m "descrição da alteração realizada"*. Para persistir suas alterações no repositório remoto na branch master é necessário executar o comando *git push origin master* no Git Bash.

4.3. Principais funcionalidades

O sistema Mastermenu tem como objetivo apresentar ao consumidor de estabelecimentos de alimentação uma alternativa de gerenciamento de pedidos. Nas subseções a seguir, as principais funcionalidades do MasterMenu são descritas.

4.3.1. Cadastro de Estabelecimento

A funcionalidade de cadastro de estabelecimento é a funcionalidade inicial do sistema, sendo que a partir de um estabelecimento o usuário pode acessar as demais funcionalidades do MasterMenu.

A princípio, foi implementado somente o mínimo necessário para se cadastrar um Estabelecimento, no caso o seu nome. As Figuras 23 e 24 a seguir mostram, respectivamente, a tela de estabelecimentos já cadastrados e a tela para cadastrar estabelecimentos do sistema Mastermenu.

Mastermenu	
Nome	Opções
Rei do Xis	<button>Editar</button> <button>Excluir</button> <button>+</button>
Pampa Burguer	<button>Editar</button> <button>Excluir</button> <button>+</button>
Oca de Savóia	<button>Editar</button> <button>Excluir</button> <button>+</button>
Frango e Cia.	<button>Editar</button> <button>Excluir</button> <button>+</button>
<button>Voltar</button> <button>Novo Estabelecimento</button>	

Figura 23. Tela Mastermenu que lista estabelecimentos já cadastrados

Fonte: Próprio autor

Mastermenu	
Nome:	<input type="text"/>
Campo com * são obrigatórios.	
<button>Salvar</button>	<button>Voltar</button>

Figura 24. Tela Mastermenu para cadastro de novo estabelecimento

Fonte: Próprio autor

Como apresentado na Figura 23, todos os estabelecimentos cadastrados podem ser excluídos e editados. Contudo, somente terá permissão para isso o usuário autenticado que cadastrou os mesmos. Além das funcionalidades de edição e exclusão para cada estabelecimento, existe também um botão com sinal de mais (+). Ao clicar nesse botão o usuário autenticado será redirecionado para uma tela que conterá as funcionalidades disponíveis para cada estabelecimento, incluindo as funcionalidades de manter produtos, manter mesas, gerir comandas e gerir pedidos. A Figura 25 apresentada logo a seguir exibe esta tela.

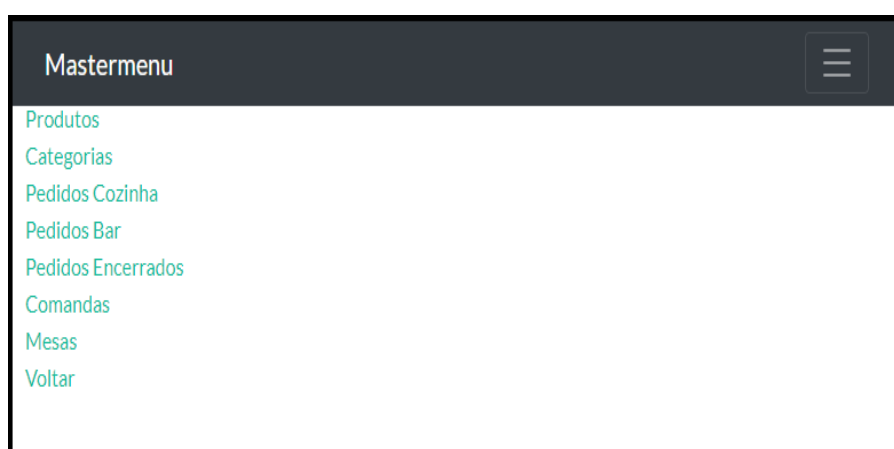


Figura 25. Tela principal do estabelecimento

Fonte: Próprio autor

4.3.2 Gerenciamento de Pedidos

Nesta funcionalidade, o usuário consumidor poderá, a partir da visualização do cardápio do estabelecimento no sistema proposto (exibido na Figura 26), adicionar os produtos a sua lista de pedidos e envia-los para produção (exibido na Figura 27). Após o envio do pedido, será apresentado para o usuário e para o setor de produção a data e horário de envio, bem como um tempo estimado para o pedido estar pronto para consumo, definido com base na fila de produção (exibido nas Figuras 28 e 29).

Mastermenu		Home
Nome	Preço R\$	Adicionar a Lista
Xis carne	16.9	Add
Xis Strogonoff	19.9	Add
Porção de Fritas	14.5	Add
Voltar		

Figura 26. Tela de cardápio de um estabelecimento específico do sistema

Fonte: Próprio autor

Mastermenu		
MEUS PEDIDOS		
Nome	Preço R\$	
Xis carne	16.9	
Porção de Fritas	14.5	
Suco de laranja	4.9	
Agua Mineral c/ gás	2.5	
Voltar	Solicitar	
ACOMPANHE SEU PEDIDO		
Nome	Hora de envio	Prazo estimado

Figura 27. Tela de lista de pedidos do consumidor

Fonte: Próprio autor

Mastermenu


MEUS PEDIDOS

Nome	Preço R\$
------	-----------

Voltar
Solicitar

ACOMPANHE SEU PEDIDO

Nome	Hora de envio	Prazo estimado
Porção de Fritas	14:39hs	14:46hs
Água Mineral c/ gás	14:39hs	14:43hs
Xis carne	14:39hs	14:53hs

Figura 28. Tela acompanhamento do pedido

Fonte: Próprio autor

Mastermenu


PEDIDOS COZINHA

Nome	Hora de envio	Prazo estimado
Porção de Fritas	14:39hs	14:46hs
Xis carne	14:39hs	14:53hs

VOLTAR
PRONTO
PRONTO

Figura 29. Tela pedidos cozinha.

Fonte: Próprio autor

Na Figura 28 é exibido a tela de pedidos da cozinha, onde existe a coluna prazo estimado, nessa coluna os valores estão em vermelho, pois se trata de uma validação do sistema, na qual muda a cor do valor de preto para vermelho quando a data corrente é igual ou maior que a data estimada. O objetivo dessa funcionalidade é policiar o funcionário a entregar seus pedidos dentro do tempo previsto.

O tempo pré-estabelecido para a execução de cada pedido foi baseado na experiência do orientando, que trabalhou cerca de um semestre em uma pizzeria situada no bairro Restinga, onde o tempo médio de realização dos pedidos era entre quinze e vinte minutos, com pouca demanda. Caso a demanda de pedidos aumentasse, o tempo estimado para cada pizza consequentemente aumentava.

4.3.3 Reserva de mesa

Esta funcionalidade oferece ao consumidor a possibilidade de reservar uma determinada mesa do estabelecimento pelo sistema. A seguir, é exibida na Figura 30 a tela em que o usuário que irá reservar escolhe a mesa em questão.

Mastermenu		Home
1 RESERVAR	2 RESERVAR	3 RESERVAR
4 RESERVAR	5 RESERVAR	6 RESERVAR
7 RESERVAR		
Voltar		

Figura 30. Tela para escolher mesa a reservar

Fonte: Próprio autor

Ao clicar no botão reservar será exibido ao cliente uma tela de confirmação da reserva, onde o usuário será obrigado a inserir o número de pessoas e a data e horário para reservar.



The image shows a web form titled "RESERVA MESA 2". It contains two input fields: "Nº de pessoas:" with the value "15" and "Data e hora:" with the value "10/12/2017 21:00". At the bottom right, there are two buttons: "Confirmar" and "Cancelar".

RESERVA MESA 2

Nº de pessoas:

15

Data e hora:

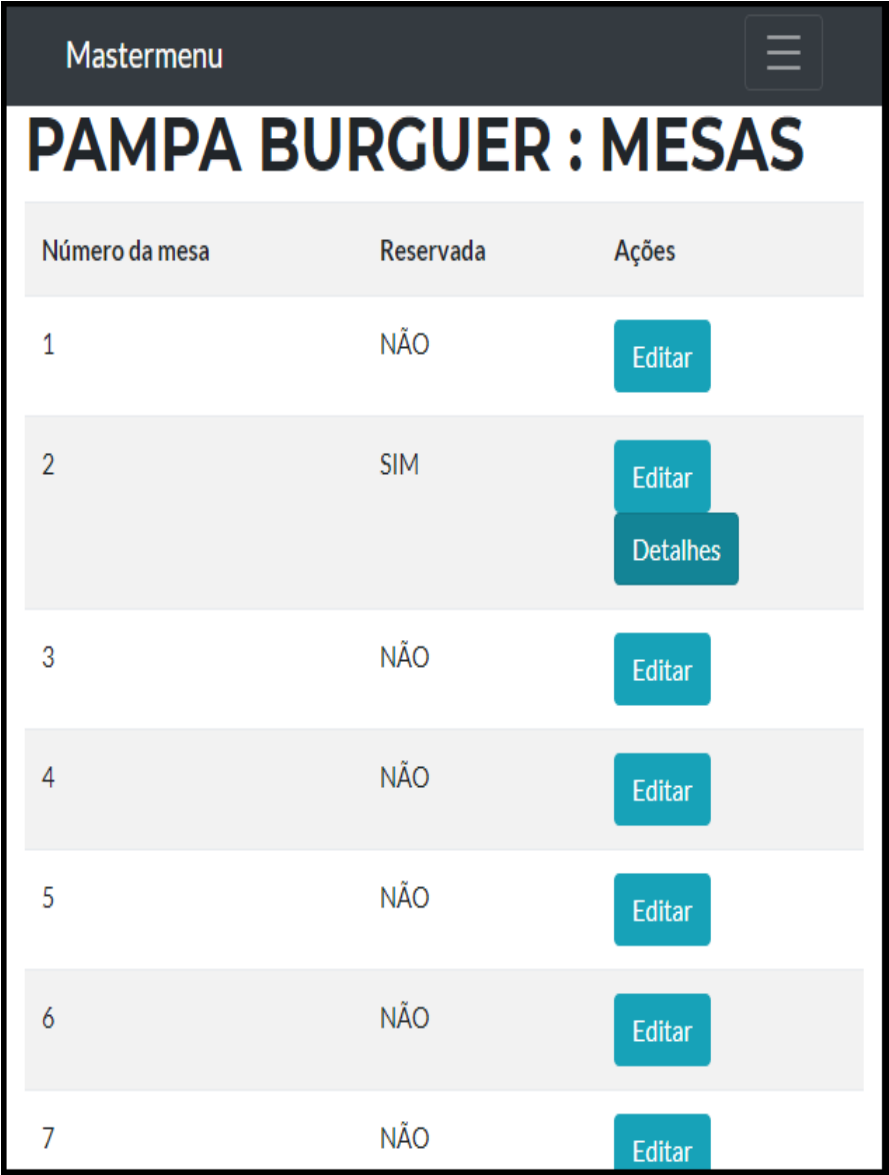
10/12/2017 21:00

Confirmar Cancelar

Figura 31. Tela de confirmação de reserva

Fonte: Próprio autor

Quando o consumidor confirma a reserva, o status da mesa do estabelecimento é automaticamente alterado como exibido na Figura 32 a seguir.



The screenshot shows a web interface for 'PAMPA BURGUER : MESAS'. At the top, there is a dark header with the text 'Mastermenu' and a hamburger menu icon. Below the header, the title 'PAMPA BURGUER : MESAS' is displayed in large, bold, black letters. The main content is a table with three columns: 'Número da mesa', 'Reservada', and 'Ações'. The table lists seven tables, numbered 1 through 7. Tables 1, 3, 5, and 7 are not reserved ('NÃO'), while tables 2, 4, and 6 are reserved ('SIM'). For reserved tables, an additional 'Detalhes' button is shown below the 'Editar' button. The buttons are teal with white text.

Número da mesa	Reservada	Ações
1	NÃO	Editar
2	SIM	Editar Detalhes
3	NÃO	Editar
4	NÃO	Editar
5	NÃO	Editar
6	NÃO	Editar
7	NÃO	Editar

Figura 32. Tela que lista as mesas do estabelecimento

Fonte: Próprio autor

O funcionário do estabelecimento, ao acessar a tela com a lista de mesas consegue visualizar o status de cada mesa. Caso alguma das mesas esteja reservada, um novo botão *Detalhes* será exibido. Ao clicar nesse botão, o usuário somente visualizará o número de pessoas e data e hora da reserva, podendo também cancelar a reserva. A Figura 33, a seguir, mostra a tela a ser exibida para o funcionário.



RESERVA: MESA 2

Nº de pessoas: 20

Data e hora: 2017-12-12 21:00hs

Cancelar Reserva Voltar

Figura 33. Tela que exibe detalhes da reserva do cliente

Fonte: Próprio autor

4.3.4 Happy Hour

Ao realizar a pesquisa de mercado e avaliando os sistemas já em execução, foi observado que um dos diferenciais para o sistema proposto seria a implementação da opção *Happy Hour* para o estabelecimento. Essa funcionalidade tem como objetivo apresentar ao usuário a alternativa de cadastrar promoções para datas e horários específicos, sem precisar ajustar preços por exemplo, após uma determinada promoção.

Ao cadastrar um novo produto, o mesmo será automaticamente exibido em uma lista de produtos, na qual conterà a opção de Happy Hour para cada produto (Figura 34). Ao clicar nessa opção, o usuário poderá informar a data e horário de início e fim da promoção, como também o preço estabelecido para este período (Figura 35).

O objetivo é mostrar para o consumidor a promoção somente no prazo especificado na sua criação.

LISTA DE PRODUTOS		
Nome	Preço R\$	Opções
Suco de laranja	5.6	<button>Editar</button> <button>Excluir</button> <button>Happy Hour</button>
Suco de uva	5	<button>Editar</button> <button>Excluir</button> <button>Happy Hour</button>
Água mineral	2.5	<button>Editar</button> <button>Excluir</button> <button>Happy Hour</button>
Suco de abacaxi	4.5	<button>Editar</button> <button>Excluir</button> <button>Happy Hour</button>
Mix de frutas	7.9	<button>Editar</button> <button>Excluir</button> <button>Happy Hour</button>

Figura 34. Tela que exibe lista de produtos com opção *Happy Hour*

Fonte: Próprio autor

HAPPY HOUR

Destinado a promoções para datas específicas.

Início:

Fim:

Preço Promocional:

ConfirmarCancelar

Figura 35. Tela Happy Hour

Fonte: Próprio autor

A seguir é exibida a tela do cardápio com a promoção Happy Hour confirmada para o

produto Suco de laranja e após o cardápio com a data da promoção expirada.



The screenshot shows a mobile application interface for a menu. At the top, there is a dark header with the text 'Mastermenu' and a hamburger menu icon. Below the header, the title 'CARDAPIO : BEBIDAS' is displayed in large, bold, black letters. The main content area is a table with three columns: 'Nome', 'Preço R\$', and 'Adicionar a Lista'. The table lists five items: 'Suco de laranja' (1.5 R\$, with a red '**PROMOÇÃO**' label), 'Suco de uva' (5 R\$), 'Agua mineral' (2.5 R\$), 'Suco de abacaxi' (4.5 R\$), and 'Mix de frutas' (7.9 R\$). Each item has a blue 'Add' button to its right. At the bottom left of the table area, there is a green 'Voltar' button. The bottom of the screen shows a black status bar with various icons (up arrow, speaker, battery, Wi-Fi) and the time '00:05' and date '12/12/2017'.

Nome	Preço R\$	Adicionar a Lista
Suco de laranja	1.5 **PROMOÇÃO**	Add
Suco de uva	5	Add
Agua mineral	2.5	Add
Suco de abacaxi	4.5	Add
Mix de frutas	7.9	Add

Voltar

Figura 36. Cardápio com promoção Happy Hour confirmada para produto *Suco de laranja*

Fonte: Próprio autor

Nome	Preço R\$	Adicionar a Lista
Suco de laranja	5.6	Add
Suco de uva	5	Add
Agua mineral	2.5	Add
Suco de abacaxi	4.5	Add
Mix de frutas	7.9	Add

Voltar

Figura 37. Visualização do cardápio com data promoção vencida para produto *Suco de laranja*

Fonte: Próprio autor

Nota-se que após o horário da promoção, como cadastrado na Figura 35, o preço retorna automaticamente para seu valor padrão, como exibido na Figura 37.

4.4 Metodologia de Desenvolvimento

O gerenciamento do projeto foi baseado na metodologia ágil SCRUM [17] e na utilização da ferramenta Trello [18], que utiliza o conceito Kanban [19].

Scrum é uma metodologia ágil para gestão e planejamento de projetos de software. No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. O Sprint representa um Time Box dentro do qual um conjunto de atividades deve ser executado.

Kanban é um conceito relacionado com a utilização de cartões (*post-it* e outros) para indicar o andamento dos fluxos de produção em empresas de fabricação em série. E o

Trello é uma ferramenta para Kanban, feita para promover a visualização do fluxo de trabalho.

A utilização de ambos são complementares, já que um dos pilares do SCRUM é a transparência. A ferramenta Trello promove isso, ficando fácil de visualizar as tarefas em execução e em qual situação está cada uma.

4.5 Arquitetura do Sistema

Após a análise dos problemas a serem resolvidos, foi pensando em como definir um padrão de projeto e como os componentes do sistema irão se organizar. Chegou-se a conclusão de implementar o padrão de projeto MVC (Model-View-Controller) [20].

O padrão MVC é uma solução já testada e documentada, utilizada por muitos projetos de software atualmente. Dividida em três camadas independentes, que são o modelo, a visão e o controlador. Essa divisão tem como objetivo desacoplar as partes independentes do código e deixá-lo mais coeso, facilitando assim a manutenção do software.

Na camada de modelo ficam os arquivos pertinentes a visualização do usuário do sistema, que conversa com a camada controladora, que por sua vez é responsável por realizar toda a parte de serviço oferecido. A camada de modelo é responsável por manter os arquivos de modelo representando objetos reais, como por exemplo, cardápio do estabelecimento.

4.6 Metodologia de Requisitos

Os requisitos para a implementação do sistema, teve embasamento na experiência profissional do orientando no período de 8 meses trabalhando em uma pizzaria, como atendente de telefone, gerenciador de pedidos e atendente ao cliente.

5. CONSIDERAÇÕES FINAIS

O trabalho proposto consisti no projeto e na implementação de um sistema de autoatendimento para consumidores de estabelecimentos alimentícios como bares e restaurantes.

Nesse sistema o consumidor pode personalizar, solicitar e acompanhar seus pedidos, através de uma única plataforma, como também, realizar a reserva de uma mesa. Para o estabelecimento, o sistema oferece as funcionalidades de gerenciar o cardápio digital, produtos e mesas do estabelecimento. O sistema também possui a função Happy Hour para cadastrar produtos com preços promocionais em datas específicas, que após a expiração dessa data o preço do produto, retornaria ao seu valor padrão.

Como sugestão de trabalhos futuros tem-se a autenticação do usuário com chave de acesso do estabelecimento, onde o consumidor somente poderá solicitar seus pedidos, se tiver autenticado com essa chave, que será disponibilizada no estabelecimento. A localização de bares e restaurantes por GPS, pagamento de conta via sistema e a implementação de um aplicativo mobile são outras sugestões para melhoria do sistema. Tais funcionalidades não são impeditivas para a utilização da plataforma, porém, acredita-se que agregaria qualidade ao serviço oferecido.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] ABRASEL: Associação Brasileira de Bares e Restaurantes. Disponível em: <<http://www.abrasel.com.br/component/content/article/7-noticias/4725-07102016-com-crise-movimento-nos-restaurantes-de-curitiba-cai-30-em-2016.html>> Acesso em: julho de 2017
- [2] Karyna Muniz Ramalho Dantas: Consultora do SEBRAE especializada no segmento de bares e restaurantes: Disponível em: <<https://clubesebrae.com.br/blog/os-principais-problemas-vivenciados-por-donos-de-restaurantes-nesse-momento-de-crise>> Acesso em: julho de 2017
- [3] Cardápio Digital Consumer. Disponível em: <<https://www.programaconsumer.com.br/cardapio-digital-restaurantes>> Acesso em: agosto de 2017
- [4] Sistema eComanda. Disponível em: <<http://www.ecomanda.com.br/sistema-de-restaurantes>> Acesso em: agosto de 2017
- [5] Menu Digital. Disponível em: <<http://www.digitalmenu.com.br>> Acesso em: agosto de 2017
- [6] Arquitetura SOAP e REST em: <<https://www.devmedia.com.br/web-services-rest-versus-soap/32451>> Acesso em: agosto de 2017.
- [7] HTML5 em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5>> Acesso em: setembro de 2017.
- [8] Bootstrap em: <<https://getbootstrap.com/>> Acesso em: setembro de 2017
- [9] AngularJS em: <<https://angularjs.org/>> Acesso em: setembro de 2017
- [10] Spark Framework em: <<http://sparkjava.com/>> Acesso em: setembro de 2017
- [11] Java Development Kit em: <<http://www.oracle.com/technetwork/pt/java/javase/documentation/index.html>> Acesso em: outubro de 2017
- [12] Hibernate em: <<http://hibernate.org/orm/documentation/5.2/>> Acesso em: outubro de 2017
- [13] Biblioteca GSON em: <<https://github.com/google/gson>> Acesso em: outubro de 2017
- [14] Mysql em: <<https://dev.mysql.com/doc/refman/8.0/en/>> Acesso em: outubro de 2017
- [15] Github em: <<https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/>> Acesso em: novembro de 2017.

[16] Crise Econômica Brasileira:

<https://pt.wikipedia.org/wiki/Crise_econ%C3%B4mica_no_Brasil_desde_2014>

Acesso em: janeiro de 2018.

[17] Metodologia SCRUM: <<http://www.desenvolvimentoagil.com.br/scrum/>> Acesso

em: janeiro de 2018.

[18] Ferramenta Trello: <[https://www.tecmundo.com.br/organizacao/75128-trello-](https://www.tecmundo.com.br/organizacao/75128-trello-ferramenta-ajudar-voce-organizar-vida.html)

[ferramenta-ajudar-voce-organizar-vida.html](https://www.tecmundo.com.br/organizacao/75128-trello-ferramenta-ajudar-voce-organizar-vida.html)> Acesso em: janeiro de 2018.

[19] Conceito Kanban: <<https://www.significados.com.br/kanban/>> Acesso em: janeiro

de 2018.

[20] Padrão MVC: <<https://www.devmedia.com.br/introducao-ao-padrao-mvc/29308>>

Acesso em: janeiro 2018.