REPÚBLICA FEDERATIVA DO BRASIL
MINISTÉRIO DA ECONOMIA
INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL
DIRETORIA DE PATENTES, PROGRAMAS DE COMPUTADOR E TOPOGRAFIAS DE CIRCUITOS INTEGRADOS

Certificado de Registro de Programa de Computador

Processo Nº: **BR512022000940-4**

O Instituto Nacional da Propriedade Industrial expede o presente certificado de registro de programa de computador, válido por 50 anos a partir de 1° de janeiro subsequente à data de 22/04/2022, em conformidade com o §2°, art. 2° da Lei 9.609, de 19 de Fevereiro de 1998.

Título: Cardápio Digital: Doce&Cia

Data de publicação: 22/04/2022

Data de criação: 21/04/2022

Titular(es): SEBASTIÃO EMIDIO ALVES FILHO; CLAUDIONOR AMÂNCIO DE OLIVEIRA JÚNIOR

Autor(es): SEBASTIÃO EMIDIO ALVES FILHO; CLAUDIONOR AMÂNCIO DE OLIVEIRA JÚNIOR

Linguagem: HTML; JAVA SCRIPT; CSS

Campo de aplicação: EC-04; SV-01; SV-03

Tipo de programa: AP-01; GI-01

Algoritmo hash: SHA-512

Resumo digital hash:
bc57ae4e47150386d7af78479cf879e4ddbbce1294a3b43bc11ababc4750169f38779c4ea9dbb5c6d2dabd25ed8b5a72281b695b5dcbe70599ce9b7d27e201e8

Expedido em: 10/05/2022

Aprovado por:
Joelson Gomes Pequeno
Chefe Substituto da DIPTO - PORTARIA/INPI/DIRPA Nº 02, DE 10 DE FEVEREIRO DE 2021

UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN

FACULDADE DE CIÊNCIAS EXATAS E NATURAIS – FANAT

DEPARTAMENTO DE INFORMÁTICA – DI

CLAUDIONOR AMÂNCIO DE OLIVEIRA JÚNIOR

CARDÁPIO DIGITAL DOCE&CIA

MOSSORÓ - RN

2022

CLAUDIONOR AMÂNCIO DE OLIVEIRA JÚNIOR

CARDÁPIO DIGITAL DOCE&CIA

Relatório apresentado à Universidade do Estado do Rio Grande do Norte como um dos pré-requisitos para obtenção do grau de bacharel em Ciência da Computação, sob orientação do Prof. Dr. Sebastião Emidio Alves Filho.

MOSSORÓ - RN

2022

CLAUDIONOR AMÂNCIO DE OLIVEIRA JÚNIOR

CARDÁPIO DIGITAL DOCE&CIA

Registro de software apresentado como pré-requisito para a obtenção do título de Bacharel em Ciência da Computação da Universidade do Estado do Rio Grande do Norte – UERN, submetida à aprovação da banca examinadora composta pelos seguintes membros:

Aprovada em: 25/04/2022

Banca Examinadora

Prof. Dr. Sebastião Emídio Alves Filho
Universidade do Estado do Rio Grande do Norte - UERN

Prof. Dr. Maximiliano Araújo da Silva Lopes
Universidade do Estado do Rio Grande do Norte - UERN

Prof. Dr. Francisco Chagas de Lima Júnior
Universidade do Estado do Rio Grande do Norte - UERN

RESUMO

Os avanços tecnológicos abrangem cada vez mais áreas da vida humana, progredindo para o surgimento de novas ferramentas, novas carreiras de trabalho, influenciando na dinâmica de outras profissões, como ocorre no mercado alimentício. A explosão do uso de aplicativos de entrega, e os mais recentes cardápios digitais, ganharam ainda mais força em sua adesão pela ação do isolamento causado pela COVID-19. O presente trabalho é uma aplicação *web* de um cardápio digital que proporciona o autoatendimento do cliente na realização dos pedidos, para um empreendimento de pequeno porte, no ramo de bolos e doces. A implementação do projeto se desenvolve com o uso de ferramentas e conceitos atuais e pouco complexos, porém poderosos no desenvolvimento *web*, como o uso da biblioteca *React* e dos serviços do *Firebase*, além da aplicação do conceito de *Single Page Application* (*SPA*).

Palavras-chave: *website*, cardápio digital, *React*, *Firebase*, *SPA*.

ABSTRACT

Technological advances influence more and more areas of human life, progressing towards the emergency of new tools and new careers, influencing the dynamic of other professions, as in the food market. The increase in use of delivery apps, and the latest digital menus, have become even more popular due to the seclusion caused by the COVID-19 pandemic. This present work is a web application of a digital menu that allows the customer to order his requests for a small-sized establishment in the cake and confectionery business. The application of the project occurs with the use of tools and simple modern concepts, but useful in web development, such as the use of React libraries and Firebase services, as well as the application of the Single Page Application (SPA) concept.

Keywords: website, digital menu, React, Firebase, SPA.

LISTA DE SIGLAS

API	*Application Programming Interface*
BaaS	*Backend as a Service*
CSS	*Cascading Style Sheets*
HTML	*HyperText Markup Language*
IDE	*Integrated Development Environment*
JSON	*JavaScript Object Notation*
NoSQL	*Not Only SQL*
Sass	*Syntactically Awesome Style Sheets*
SDK	*Software Development Kit*
SPA	*Single Page Application*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A tecnologia está progressivamente mais habitual no meio social, na maneira como nos comportamos, comunicamos, compramos, aprendemos; encontra-se presente nas mais diversas atividades comuns ao nosso dia-a-dia (Prado, 2005). A exemplo da ação da compra, algo realizado somente de forma presencial no Brasil, até o início do século XXI, atualmente podemos fazer de maneira muito simples, em qualquer lugar que estivermos, por um dispositivo móvel que cabe na palma da mão.

As inovações tecnológicas impactam na rotina da sociedade e apresentam soluções que agregam qualidade e praticidade de uso (Manso, 2019), revolucionam nossos hábitos, repercutindo inclusive em tarefas pequenas e aparentemente não tão importantes. Há pouco tempo, era comum realizar ligações para fazer algum pedido de lanche individual ou coletivo, no final de semana, ou para a compra de uma marmita para o almoço de um dia cansativo ou corriqueiro, estas e outras ainda são necessidades da população, contudo a forma de realizá-las atualmente evoluiu junto com a tecnologia.

A utilização de smartphones e/ou computadores para a compra de diversos produtos já era uma realidade, mas agora temos aplicativos/*websites* com uma interface mais atrativa e amigável ao cliente, trazendo facilidade para sua experiência de compra em poucos cliques, assim como agilidade para o comerciante/empreendedor na logística e atendimento de seus serviços (Monty, 2018; Manso, 2019). São esses os chamados aplicativos de entrega.

Além do contínuo avanço desses sistemas, um fator externo ao mercado de *food services* impulsionou ainda mais o surgimento e uso de tais mecanismos digitais. Em 30 de janeiro de 2020, com o mundo assolado pelo vírus SARS-CoV-2, acarretando na pandemia da COVID-19, a Organização Mundial de Saúde (OMS) declarou estado de emergência internacional (OPAS, 2020). O distanciamento social se fez necessário como uma medida preventiva ao contágio entre a população mundial.

Devido à necessidade do isolamento, a população se viu forçada a buscar meios alternativos de consumo, tornando os aplicativos de entrega a medida mais viável. Desse modo, os serviços nos quais antes traziam facilidade, comodidade, agora também ofereciam

segurança aos usuários, ao realizarem seus pedidos de alimentos pela internet, em casa, de forma segura, respeitando o distanciamento.

1.1 Justificativa

As plataformas de pedidos são um exemplo claro do impacto do digital no social. Essas ferramentas estão mais frequentes no cotidiano e tornam-se habituais ao uso do indivíduo. Ao promover uma experiência mais simples, célere e positiva ao mercado de *food services*, as referidas plataformas resultaram em avanços, tanto na visão do cliente consumidor quanto na do cliente empreendedor.

Um cenário crescente e propício, segundo pesquisa realizada pela Confederação Nacional de Dirigentes Lojistas (CNDL) e do Serviço de Proteção ao Crédito (SPC Brasil), nos últimos 12 meses, 54,80% (cinquenta e quatro vírgula oitenta por cento) dos internautas brasileiros realizaram compra de comida pela internet, um aumento de 14,40% (quatorze vírgula quarenta por cento) em comparação ao ano de 2019 (G1, 2021). Esse mercado demanda pessoal qualificado para o desenvolvimento e uso das tecnologias e suas potencialidades.

Os pequenos negócios têm suas atribuições centralizadas em poucos funcionários, em alguns casos em um único funcionário, responsável por todo o serviço, o qual se sobrecarrega com diversas atividades, como a realização do serviço em si e o atendimento com o cliente. Além disso, se tratando de pequenos empreendedores, especialmente os de pequenas áreas urbanas, muitos não têm domínio sobre a tecnologia e buscam auxílio para o negócio em ferramentas *online*, porém estas não provêem o apoio de um suporte local, viabilizando uma atenção presencial para sanar as dúvidas sobre o uso.

Ciente desse avanço do ambiente virtual favorável ao comércio, disponibilizar um sistema que forneça funcionalidades para tornar as atividades do negócio acessíveis, pode impulsionar o empreendimento. Nesse contexto, automatizar a ação da encomenda pelo consumidor, provendo um suporte local ao comerciante, apresenta-se como uma ferramenta atrativa para aquecer um negócio familiar, trazendo a facilidade e comodidade do atendimento *online*, a otimização, a interface atrativa e o uso acessível aos clientes.

1.2 Objetivos

O objetivo do sistema é a elaboração de um cardápio digital que funcione através de um *website,* para a automação dos pedidos feitos pelos clientes a um empreendimento local de pequeno porte.

Os aplicativos de entrega seguem um nicho de mercado de alimentos mais amplo, ofertando diferentes tipos de comida. A finalidade do sistema apresentado é focar em um mercado mais específico do segmento alimentício, como o mercado de doces e bolos.

Apesar de muito semelhante a um aplicativo de entrega, o sistema implementado se caracteriza como um cardápio digital, uma funcionalidade existente nos aplicativos de entrega, porém sem a logística da entrega através do sistema. Divergindo também em singularidade, já que está voltado a um único empreendimento. Para tanto, constitui-se, basicamente, de uma versão virtual de cardápio do estabelecimento, que proporciona uma experiência de autoatendimento ao cliente.

O sistema consiste em uma aplicação *web* onde os clientes podem acessar a página da confeitaria, através de seus *browsers*, tanto de um dispositivo *mobile* quanto de um *desktop*, para realizar os seus pedidos dentro da interface do *website*, escolhendo os produtos, selecionando os sabores de sua preferência, conferindo preços e quantidades. Ao final, o pedido será encaminhado, como uma mensagem pronta e estruturada, através do aplicativo mensageiro *Whatsapp*.

O propósito do sistema é que o cliente possa fazer suas escolhas e encaminhar o pedido já pronto para o lojista. Essa ferramenta contribui para que o empreendedor não desperdice muito tempo com o atendimento ao cliente. Dessa forma, ao permitir um fácil acesso e atualização ao cardápio da loja, há uma otimização na escolha dos produtos, comunicação com o comerciante e confirmação do pedido. Além de possibilitar o registro e armazenamento dos pedidos realizados para um controle de vendas dentro da aplicação.

1.3 Trabalhos relacionados

Em busca de realizações semelhantes que estão em atuação no mercado, para o embasamento do presente trabalho, foi possível criar um comparativo com a proposta de projeto aqui apresentada. A seguir temos uma descrição de cada critério escolhido.

- Cardápio digital: uma versão virtual do menu de opções prestadas pelo empreendimento.
- Montar pedido: dar a liberdade de um pedido mais customizado, com escolha de tipo, tamanho, massa, recheio.
- Comunicação via *Whatsapp*: comunicação terceirizada, sendo direcionado para o *Whatsapp*, que é o mensageiro mais utilizado no país (Valor, 2022), trazendo comodidade ao usuário na utilização de uma ferramenta já conhecida, além de prover segurança na troca de mensagem, atendendo bem a necessidade de comunicação entre ambas as partes (cliente/empreendedor).
- *Delivery*: sistema de entregas integrado à aplicação.
- Retirada no local: possibilitar ao usuário cliente a opção de retirar sua encomenda no local de comércio.
- Integração direta com meio de pagamento: integrado a um ferramenta de pagamento online para que este possa ser feito dentro da aplicação.
- Cobrança de taxas: é cobrada alguma taxa ou mensalidade por parte da aplicação para a utilização da mesma.
- Suporte local: suporte local técnico e de dúvidas para o empreendedor.
- Suporte remoto: suporte remoto técnico e de dúvidas para o empreendedor.
- Nicho de mercado: indústria no qual é voltada a utilização do programa.

Os comparativos foram feitos entre a ideia (Doce&Cia) com os concorrentes: Goomer, Cardápio Digital IFood, Kyte e oimenu, sendo, estas últimas, aplicações que tem por finalidade oferecer uma experiência de logística melhorada na realização dos pedidos. Nesse contexto, a Tabela 1, abaixo descrita, demonstra as funcionalidades usadas como critérios para a comparação, expondo os pontos fortes e fracos e quais das propostas atendem a cada uma delas.

Tabela 1 – Comparação entre a ideia e propostas semelhantes.

Similares / **Funcionalidades**	**Goomer**	**Cardápio Digital IFood**	**Kyte**	**oimenu**	**Proposta: Doce&Cia**
Cardápio digital	✔	✔	✔	✔	✔
Montar pedido	-	-	-	-	✔

Comunicação via Whatsapp	✔	✔	✔	-	✔
Delivery	-	✔	-	-	-
Retirada no local	✔	✔	✔	-	✔
Integração direta com meio de pagamento	✔	✔	✔	✔	-
Cobrança de taxas	✔	✔	✔	✔	-
Suporte Local	-	-	-	-	✔
Suporte Remoto	✔	✔	✔	✔	✔
Nicho de mercado	Alimentício	Alimentício	Variado	Alimentício	Alimentício

Fonte – Autoria própria (2022).

Este trabalho proposto se destaca nos quesitos “montar pedido” e “suporte local”, respectivamente, pelas características de oferecer uma experiência mais customizada para o cliente, no ato de formar o pedido, e por se tratar de um serviço familiar pessoal que irá atender a uma demanda local, na cidade de Grossos/RN, onde o negócio está estabelecido, possibilitando auxiliar às necessidades técnicas. Contudo, o trabalho desenvolvido pode ser expandido para novos tipos de negócios e em outras localidades, necessitando da adequação dos produtos disponíveis e suas possíveis customizações.

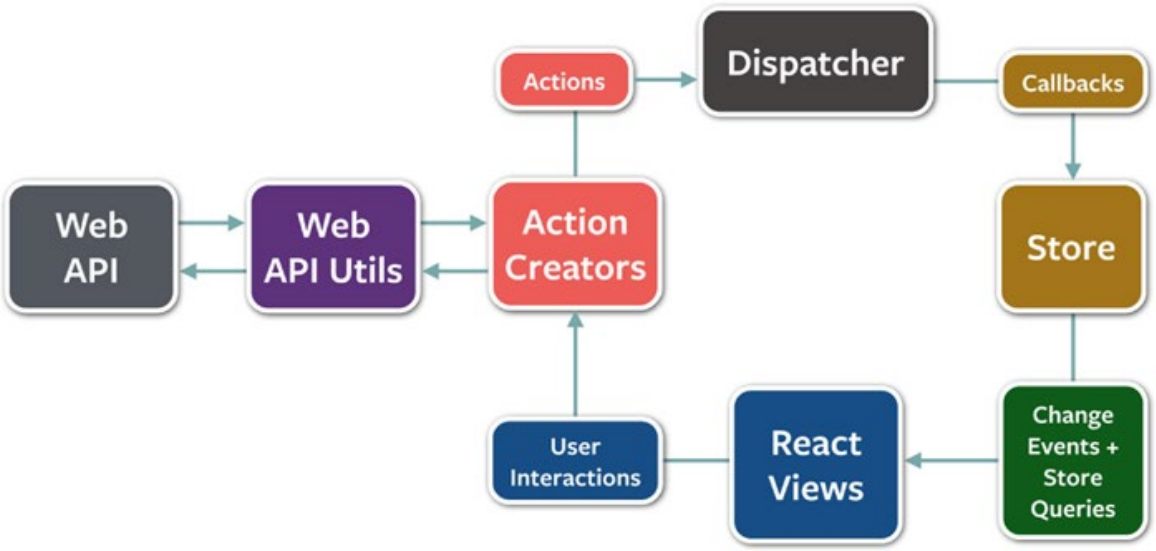
2 ARQUITETURA DO SISTEMA

A arquitetura utilizada na implementação do sistema é o *Flux*, desenvolvida pela Meta, a qual está focada em aplicações *web client-side,* utilizando de um fluxo de dados unidirecional. O *Flux,* por sua vez, foi elaborado especificamente para aplicações *React*, que, especificamente, é utilizada neste projeto (ver seção 3.2). Por isso, os elementos principais dessa arquitetura são três: *dispatcher, store* e *view*, este último nada mais é do que um componente *React* (Gackenheimer, 2015).

Na Figura 1 (a seguir), podemos observar o fluxo inicial de dados nesse tipo de arquitetura, no qual começa com uma ação, muitas vezes correspondente à interação do usuário a uma *view*, cuja ação é transferida para o *dispatcher*, que funciona como um

gerenciador central do fluxo de dados. É nele que estão registradas as *callbacks* das *stores*. Já as *stores* são responsáveis por registrar os dados, em outras palavras, o estado da *view* e emitir os eventos. Desse modo, quando uma *callback* é invocada por uma ação, emitem o evento da *store* para alteração das *views*, acarretando em uma nova renderização delas mesmas e de todos os seus descendentes (*Flux*, 2022).

Figura 1 – Fluxo de dados na arquitetura *Flux*.

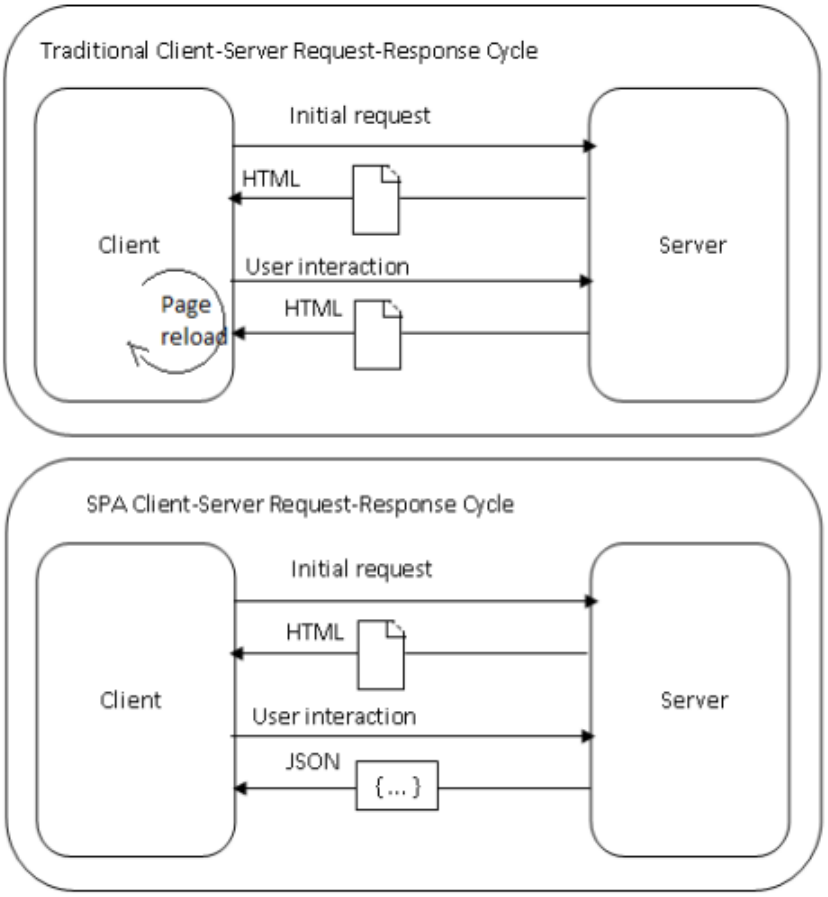


Fonte – *Introduction to React* (2015).

Diante disso, o sistema compreende-se por um *website* construído, utilizando do conceito de *Single Page Application* (*SPA*), na qual a aplicação *web* consiste de componentes individuais, independentes e reutilizáveis. Na requisição inicial ao servidor, são retornados todos os recursos, como *HTML*, estilização em *CSS*, imagens, *JavaScript* e dados, para a composição de cada componente e renderização da página, podendo estes, respondendo as interações do usuário, serem atualizados/substituídos de forma autônoma (Jadhav et al, 2015).

A cada ação realizada pelo usuário, somente o componente manipulado tem suas informações atualizadas – em alguns casos não sendo necessário a renderização do componente por completo, somente os dados retornados de uma nova chamada ao banco de dados –. A Figura 2 ilustra como distingue-se uma aplicação *web* tradicional, na qual retorna toda a estrutura *HTML,* em comparação a uma *SPA,* onde o servidor retorna somente os dados em formato *JSON*, não necessitando do recarregamento de todo o conteúdo da página, reduzindo o uso de rede, acarretando em melhora de performance.

Figura 2 – Comparação do ciclo de requisição e resposta entre modelo tradicional e *SPA*.



Fonte – *Single Page Application using AngularJS* (2015).

O servidor funciona através de um *Backend as a Service* (*BaaS*), que provê as funcionalidades necessárias ao *back-end* do *website*. *BaaS* é a ideia de fornecer o *back-end* de uma aplicação como um serviço terceirizado, através do qual pode-se aproveitar de mecanismos, como autenticação de usuário, gerenciamento de banco de dados, armazenamento em nuvem, hospedagem, dentre outros disponíveis para o uso. Tal ferramenta é uma facilitadora para a construção de aplicações, reduzindo o trabalho, já que não é necessário construir seu próprio *back-end*, assim, otimizando o processo de desenvolvimento (Cloudflare, 2022).

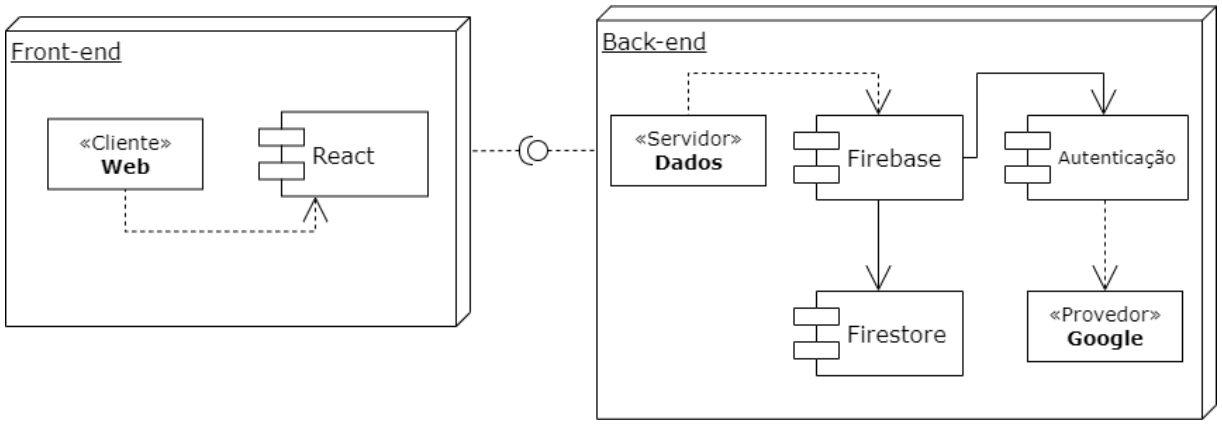
2.1 Diagrama de Componentes

Segundo Guedes (2009), o diagrama representa os componentes implementados no sistema, em termos de código-fonte, bibliotecas, formulários, arquivos de ajuda, dentre

outros. Ele demonstra como são estruturados, quais e de que forma interagem para o funcionamento do sistema.

No sistema aqui apresentado, os componentes estão separados em 2 blocos: a interface, acessível ao usuário, construída por meio do *React*, diz respeito ao *front-end*, que usufrui dos serviços prestados pelo *Firebase*, mecanismos correspondentes a um *back-end*. Vide Figura 3.

Figura 3 – Diagrama de Componentes do sistema.



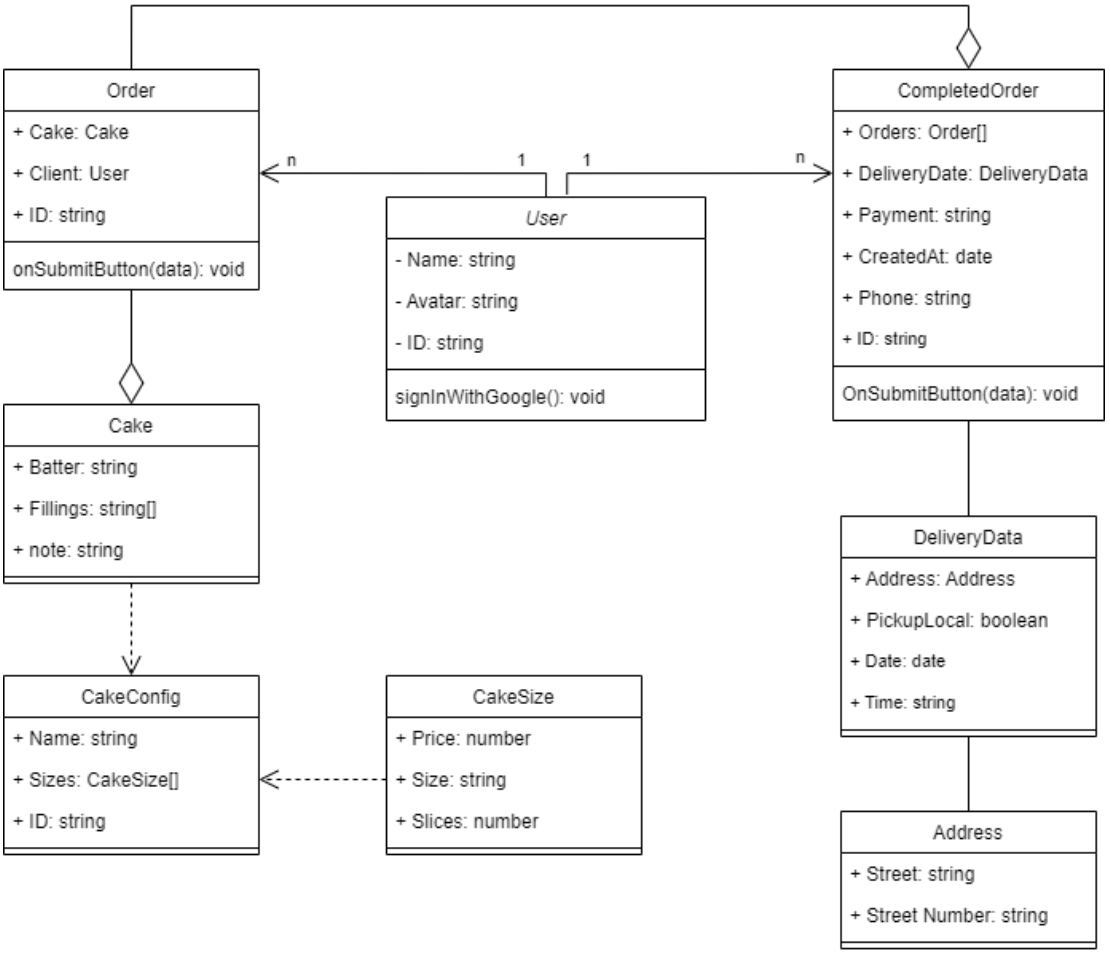
Fonte – Autoria própria (2022).

2.2 Diagrama de Classes

Diagrama usado para o desenvolvimento de um modelo de sistema orientado a objetos, representando a estrutura de classes presentes no sistema, determinando os atributos e métodos nelas presentes, e como se associam (Sommerville, 2011).

O diagrama de classes, apresentado pela Figura 4, expressa as classes presentes no sistema, contendo cada atributos e métodos necessários ao seu funcionamento. O usuário é uma figura central, podendo um mesmo usuário criar e realizar inúmeros pedidos. Nesse contexto, cada pedido criado conta com as informações selecionadas para o bolo, e, na ação de realizar o pedido, ilustrada pela classe *CompletedOrder*, contém os pedidos criados e dados precisos para a entrega.

Figura 4 – Diagrama de Classes do sistema.



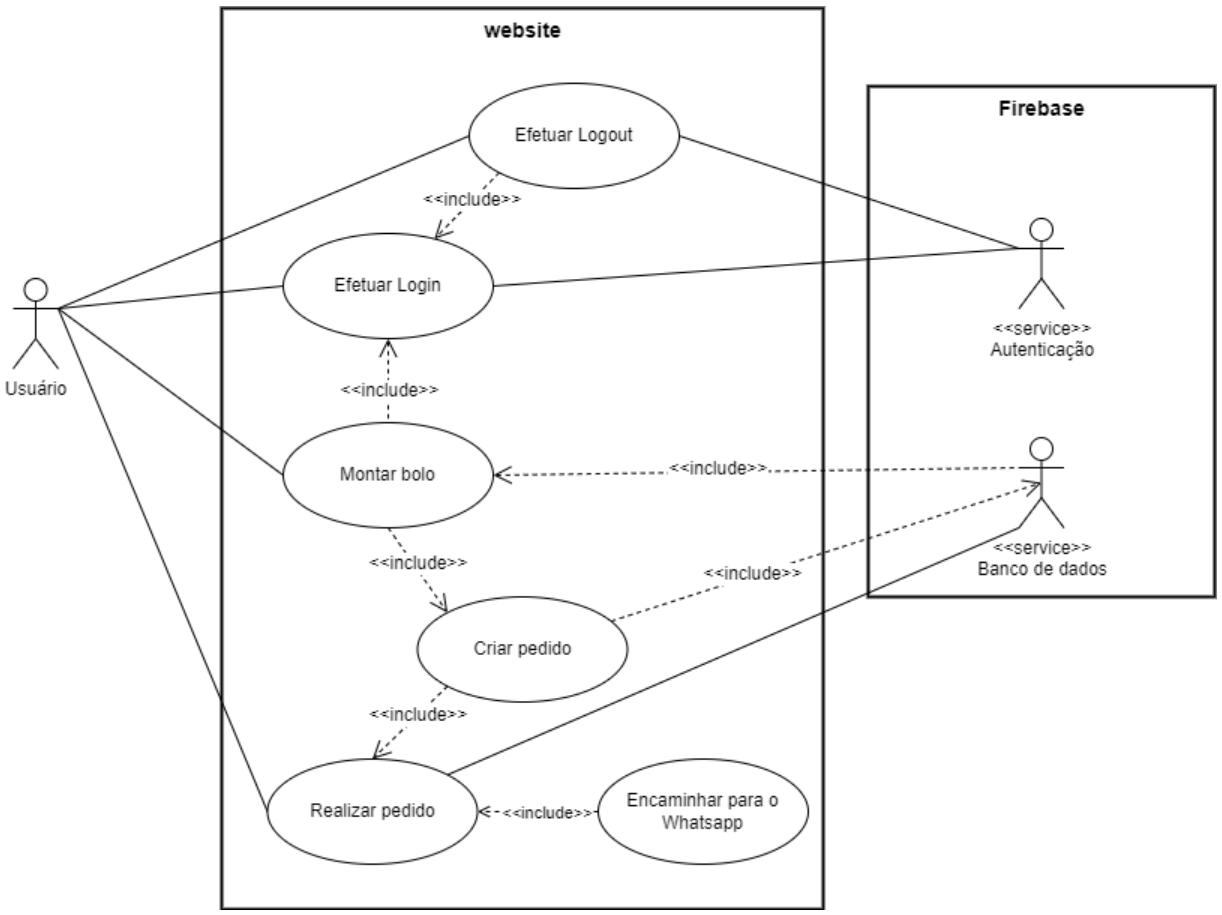
Fonte – Autoria própria (2022).

2.3 Diagrama de Casos de Uso

Uma representação mais geral e informal, com uma abordagem simples e de fácil entendimento do comportamento do sistema. O diagrama de casos de uso é utilizado para o levantamento e análise dos requisitos do mesmo, identificando os atores que interagem com os serviços presentes no sistema (Guedes, 2009).

O ator principal da aplicação é o usuário, um cliente da loja, no qual pode efetuar o *login* no site, bastando que tenha uma conta Google (necessária para autenticação) e a partir de então é possível fazer a seleção das opções para o bolo, criar o pedido e encaminhá-lo para o *Whatsapp* do comerciante. A Figura 5 demonstra como cada ator pode atuar dentro do sistema.

Figura 5 – Diagrama de Casos de Uso do sistema.



Fonte – Autoria própria (2022).

2.4 Descrição dos principais requisitos

Nesta subseção são especificados e definidos todos os atores e serviços requeridos ao sistema. Os atores existentes e suas funções na aplicação são descritos na Tabela 2.

Tabela 2 – Atores do sistema.

Usuário	**Descrição**
Cliente	Efetua o *login*; gera e realiza os pedidos dentro do sistema.
Firebase	Serviço de *backend* responsável pela autenticação do usuário e armazenamento das informações.

Fonte – autoria própria (2022).

Os requisitos funcionais do sistema descrevem os serviços presentes no mesmo, como se comporta em dada situação e de que forma deve reagir frente a atuação do usuário

(Sommerville, 2011). A Tabela 3 lista quais são os requisitos funcionais que o sistema atende e retrata suas ações.

Tabela 3 – Requisitos funcionais do sistema (RF).

ID	**Nome**	**Descrição**
RF01	Efetuar *Login/Logout*	Autenticação do usuário no sistema.
RF02	Montar bolo	Seleção de opções dos pedidos, como tipo, tamanho, sabor, por parte do usuário.
RF03	Criar pedido	Gravação das escolhas feitas pelo usuário.
RF04	Realizar pedido	Executar pedido(s) criado(s) pelo usuário.

Fonte – autoria própria (2022).

De acordo com Sommerville (2011), os requisitos não funcionais “são os quais não estão diretamente relacionados com os serviços específicos oferecidos pelo sistema a seus usuários”. A Tabela 4 descreve os requisitos não funcionais presentes na aplicação.

Tabela 4 – Requisitos não funcionais (RNF).

ID	**Nome**	**Descrição**
RNF01	Interface acessível	O *website* possui uma interface simples de fácil entendimento, para trazer comodidade ao usuário.
RNF02	Responsividade	A interface do *website* é pensada para se adaptar a diferentes tamanhos de tela.

Fonte – autoria própria (2022).

A Tabela 5 relata a especificação dos Casos de Uso do sistema, o processo que ocorre em cada atividade e quais são os atores que estão envolvidos.

Tabela 5 – Especificação de Casos de Uso do sistema (CDU).

CDU01 - Efetuar Login/Logout
Descrição: usuários se autenticam através de uma conta Google para poder

realizar os pedidos no sistema.

Requisitos atendidos: RF01.

Pré-condição: possuir conta Google.

Pós-condição: usuário poderá montar bolos e realizar pedidos.

Atores: cliente e *Firebase*.

Condição de entrada: acessar o sistema e clicar em *login* ou montar bolo.

Processo padrão:

a) cliente acessa o *website*;

b) cliente clica no botão de *login* ou de montar bolo;

c) cliente fornece e-mail Google válido;

d) *Firebase* faz o reconhecimento e retorna os dados do usuário.

CDU02 - Montar bolo

Descrição: o usuário deve selecionar uma dentre as opções de tipos e tamanhos de bolos disponíveis para então escolher sabores de massa e recheio para o bolo; além de ter a opção de adicionar uma observação no pedido.

Requisitos atendidos: RF01 e RF02.

Pré-condição: estar logado no sistema.

Pós-condição: salvando no banco de dados o pedido com as devidas seleções feitas.

Atores: cliente.

Condição de entrada: clicar no botão “montar bolo”.

Processo padrão:
a) cliente loga no sistema;
b) seleciona tipo e tamanho de bolo;
c) escolhe sabores de massa e recheios;
d) pode escrever uma observação;

CDU03 - Criar pedido

Descrição: o usuário salva as preferências escolhidas para o bolo como um pedido.

Requisitos atendidos: RF01, RF02 e RF03.

Pré-condição: estar logado na aplicação; ter selecionado as opções de sua escolha.

Pós-condição: visualização dos pedidos criados na página de sacola.

Atores: cliente e *Firebase*.

Condição de entrada: clicar no botão “adicionar à sacola”.

Processo padrão: a) usuário clica no botão “adicionar à sacola"; b) o pedido é salvo no banco de dados do *Firebase*.
CDU04 - Realizar pedido
Descrição: na página de sacola o cliente visualiza os pedidos que foram criados por ele, além de adicionar outras informações como endereço e data de entrega, telefone para contato e forma de pagamento como informações necessárias para a realização do pedido. **Requisitos atendidos:** RF01, RF03 e RF04. **Pré-condição:** estar logado e ter algum pedido criado pelo usuário. **Pós-condição:** pedido finalizado é salvo no banco de dados e uma mensagem estruturada com as informações do pedido são enviadas para o *Whatsapp* da loja. **Atores:** cliente e *Firebase*. **Condição de entrada:** clicar no botão “enviar pedido”. **Processo padrão:** a) cliente fornece as informações para entrega; b) o pedido finalizado é salvo no banco de dados do *Firebase*; c) mensagem contendo as informações do pedido e entrega são enviadas ao *Whatsapp* da loja.

Fonte – autoria própria (2022).

3 IMPLEMENTAÇÃO

Como ponto de partida para a produção, foi realizada uma reunião com o dono do negócio com a finalidade de definição do escopo do processo, a fim de conhecer quais necessidades o cliente gostaria de serem atendidas, a partir daí construindo o *backlog* do produto. Após esse primeiro contato foi feita a modelagem básica da estrutura do *layout* do *website.* Discutida a lista de aplicabilidades e tendo o *layout* base, definiu-se a precedência para implementação.

Uma explicação mais descritiva sobre o funcionamento de cada parte da aplicação é feita na seção 3.3, juntamente com a demonstração da interface.

Metodologia de desenvolvimento

O gerenciamento das atividades de elaboração do projeto se deu por meio do *framework* ágil Scrum, mais especificamente o Scrum Solo (Pagotto et al, 2016), uma variação do método voltada para o desenvolvimento individual. Nesse modelo, elementos do Scrum original, como o *backlog* do produto, uma lista de funcionalidades necessárias para a implementação, e o *sprint backlog*, uma lista derivada do *backlog* do produto com os itens selecionados para serem realizados no *sprint*, permanecem os mesmos, diferindo na redução da duração dos *sprints* (ciclos de desenvolvimento) para uma semana e na retirada das reuniões diárias. O Scrum Solo possui quatro atividades: *requeriments, sprint, deployment e management*, descritas abaixo:

- *Requeriments*: definição do escopo do produto, identificação do cliente do produto e definição do *backlog* do produto.
- *Sprint*: desenvolvimento do conjunto de itens selecionados da lista de *backlog* do produto.
- *Deployment*: disponibilização do produto para uso do cliente.
- *Management*: planejamento, monitoramento e desenvolvimento do produto.

3.1 Ferramentas e bibliotecas utilizadas

O projeto proposto foi implementado utilizando *React*, uma biblioteca *JavaScript open source* criada pelos engenheiros da Meta, focada na produção de interfaces de usuário. Tal mecanismo é uma biblioteca baseada em componentes declarativos, sendo estes análogos a funções *JavaScript*, nas quais podem receber entradas e retornam elementos *React* responsáveis pela composição do que deve ser apresentado em tela. Já os componentes são partes independentes e reutilizáveis que juntos constroem a interface (*React*, 2022).

Para o desenvolvimento foi utilizado o *Create React App*, uma *toolchain* oficialmente compatível com *React*, também criada pela Meta, recomendada pela própria comunidade *React* para a construção de *SPAs*, oferece um ambiente configurado e atualizado para a utilização da biblioteca (*React*, 2022). Por trás utiliza-se de um gerenciador de pacotes, para este caso foi escolhido o *Yarn*, uma solução *open source* de compartilhamento de código de terceiros através de pacotes, na qual se sobressai no desempenho e segurança em relação ao gerenciador padrão do *Node.js*, o *NPM* (Chodorowski, 2021). A *toolchain* compõe-se também de um compilador, o Babel, que converte o código ECMAScript 2015 ou superior

em uma versão de código *JavaScript* compatível com os *browsers* (Babel, 2022) e o *Webpack*, um empacotador de módulo estático para aplicativos *JavaScript* (*Webpack*, 2022).

O *React* é uma biblioteca que trabalha em cima da linguagem *JavaScript*, porém é utilizada a linguagem *TypeScript* para este projeto. O *Create React App* fornece um *template* que permite a utilização sem conflitos do *Typescript*, um *superset* que adiciona tipagem a sintaxe do *JavaScript*, integrando-se melhor com os editores de código, permitindo utilizar dos recursos de *intelliSense*, como edição e sugestão de código, dando um suporte melhor para o desenvolvimento (*TypeScript*, 2022).

Na estilização da interface do *website* é empregado o *Sass*, um pré-processador de *CSS* que permite adicionar mais funcionalidades à linguagem de estilização. O *Sass* faz uso de variáveis, funções, e os chamados *mixins*, que possibilitam a reutilização de estilos, além de possuir uma estrutura de regras aninhadas, funcionalidade que mais atrai para uso do pré-processador, por dar mais organização, legibilidade e redução de código (*Sass*, 2022).

Algumas das bibliotecas de terceiros aplicadas ao sistema foram o *React Router*, o *React Hook Form* e o *Yup*, o primeiro trata-se de uma coleção de componentes, *hooks* e utilitários para definição e manipulação de rotas em aplicações *React* (*React Router*, 2022); enquanto estes dois últimos trabalham em conjunto, respectivamente, um sendo responsável por oferecer mecanismos para a criação de formulários de alto desempenho, flexíveis e extensíveis dentro do *React* e o outro responsável por prover ferramentas para criação de esquemas de análise e validação dos valores adicionados aos campos dos formulários.

O *Visual Studio Code* foi a *IDE* utilizada. É um editor de código-fonte leve, mas poderoso, que é executado na área de trabalho, e está disponível para diferentes sistemas operacionais. Ele vem com suporte integrado para *JavaScript*, *TypeScript* e *Node.js* e possui um rico ecossistema de extensões para outras linguagens (como C++, C#, Java, *Python*, PHP, *Go*) e ambientes de execução (como .NET e *Unity*) (*Visual Studio Code*, 2022).

Como repositório para o compartilhamento e controle de versão do código, é utilizado o Github, uma plataforma de hospedagem de código em nuvem para acesso remoto, de controle de versão e colaboração. Para tanto, permite o trabalho em conjunto em projetos e o gerenciamento do progresso de desenvolvimento (GitHub, 2022).

O *Firebase* atua como o *back-end* da aplicação, por ser um *BaaS*, propicia utilidades, como autenticação e armazenamento em nuvem – que foram empregues ao sistema –, além

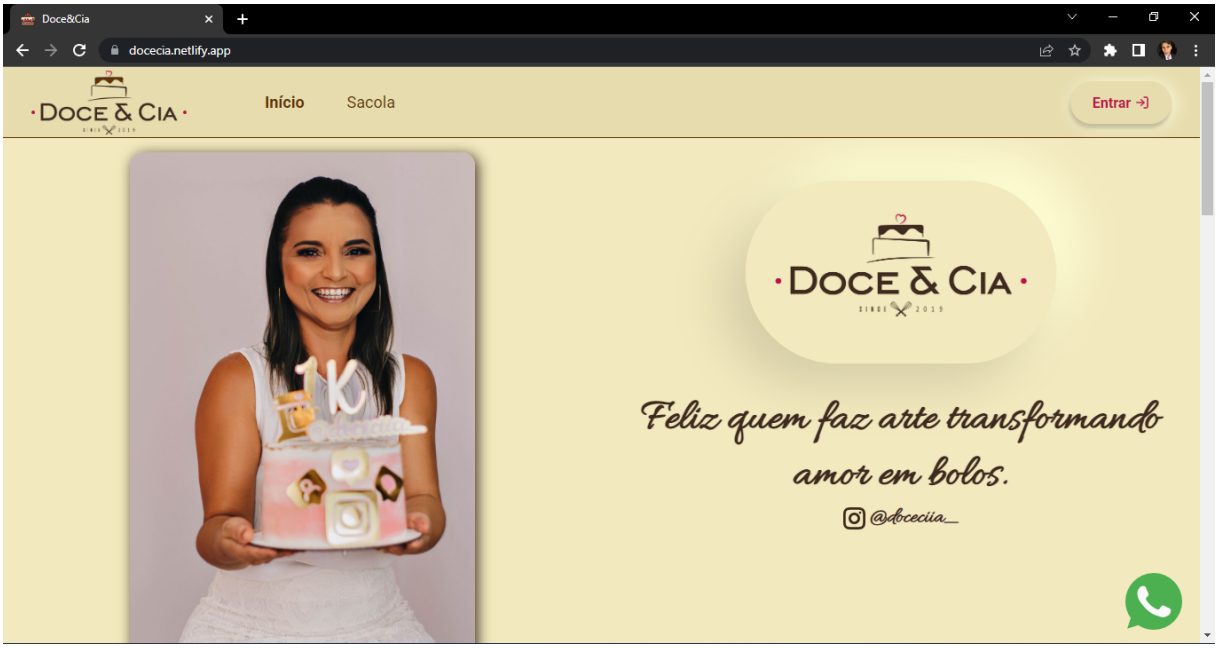
de outras. É possível integrá-lo a diferentes tipos de tecnologias, como *Android, Apple, Unity* e o *JavaScript*, através das *SDK*s correspondentes. A sua plataforma é de fácil acesso, por ser desenvolvido pela Google, basta que possua uma conta no domínio da empresa para utilizar dele, também ofertando as funcionalidades em planos, incluso um gratuito.

O *Firebase Authentication* provê os mecanismos de autenticação do usuário, como autenticação por *email* e senha, números de telefones ou provedores de *login* social, como o próprio Google, *Facebook*, Twitter, dentre outros. E o *Cloud Firestore*, um banco de dados em nuvem *NoSQL*, organiza os dados nas chamadas coleções, que são contêineres de documentos nos quais mapeiam os campos de valores em uma estrutura *JSON* (*Firebase*, 2022).

3.2 Resultados

A tela de Início (Figura 6) é o ponto de partida do *website*, uma apresentação inicial do cliente ao empreendimento. Nesta tela está presente um cabeçalho contendo a barra de navegação entre as demais telas, além do botão para *login/logout* do cliente na aplicação, que funciona através do serviço de autenticação do *Firebase* com o Google como *login* social. Um ponto interessante no cabeçalho é que, por ser um componente *React*, por assim independente, se repete entre outras páginas sem necessidade de seu recarregamento.

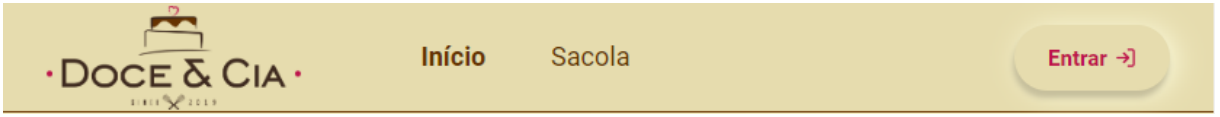
Figura 6 – Tela inicial.



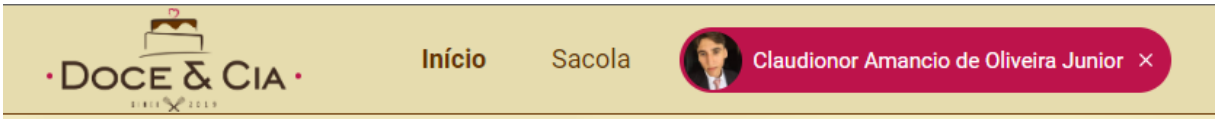
Fonte – Autoria própria (2022).

O botão de *login* também é um componente, encapsulado dentro do cabeçalho. Nele é exibido o *status* do usuário, alterando sua apresentação na interface a depender do usuário, se não está logado, Figura 7a, ou se está, Figura 7b.

Figura 7 – Interfaces do botão de *login/logout*.



(a)



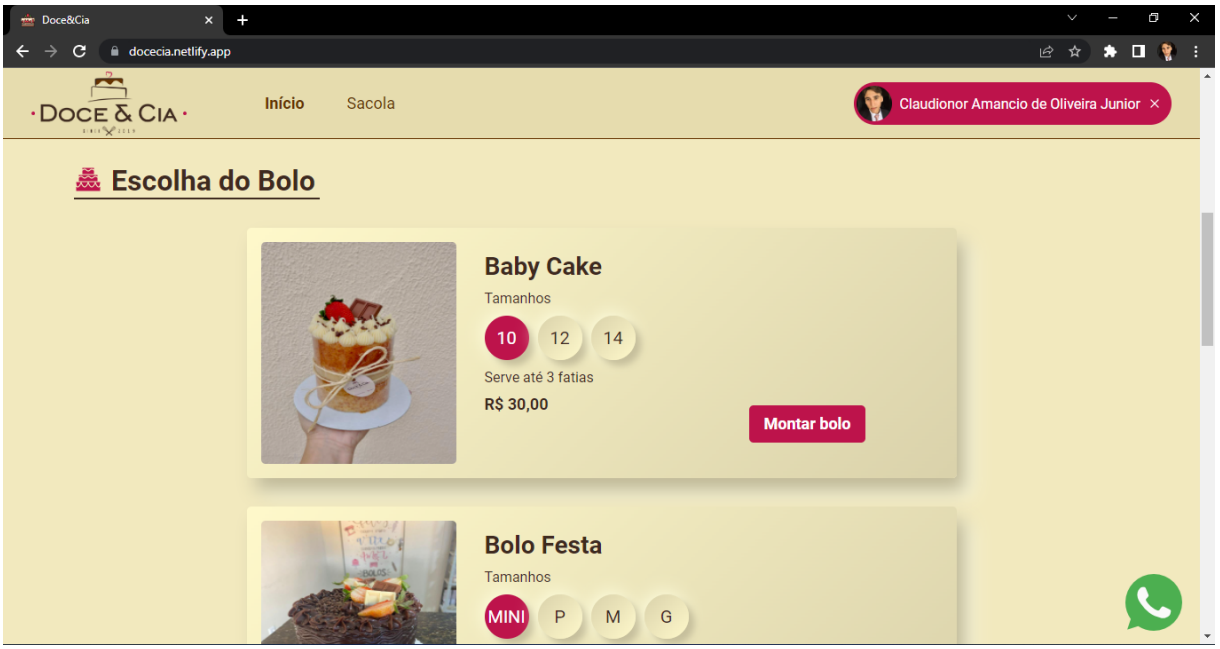
(b)

Fonte – Autoria própria (2022).

Logo abaixo é exibida a tela de Menu, representada pela Figura 8, é acessível pela rolagem do site, onde está presente o cardápio de opções de bolos e seus respectivos tamanhos disponíveis ao usuário. Foi pensado em um *layout* simples, com *cards* correspondentes a cada item, para deixar já visível ao usuário suas opções e efetuar suas escolhas. Este pode ver qual item o agrada e selecionar o tamanho desejado, assim, se

logado, clicando no botão “Montar bolo”. Caso não esteja logado, acionando o botão, este irá abrir um *pop-up* para autenticação, pois só é permitida a montagem do bolo se estiver autenticado.

Figura 8 – Tela de menu.



Fonte – Autoria própria (2022).

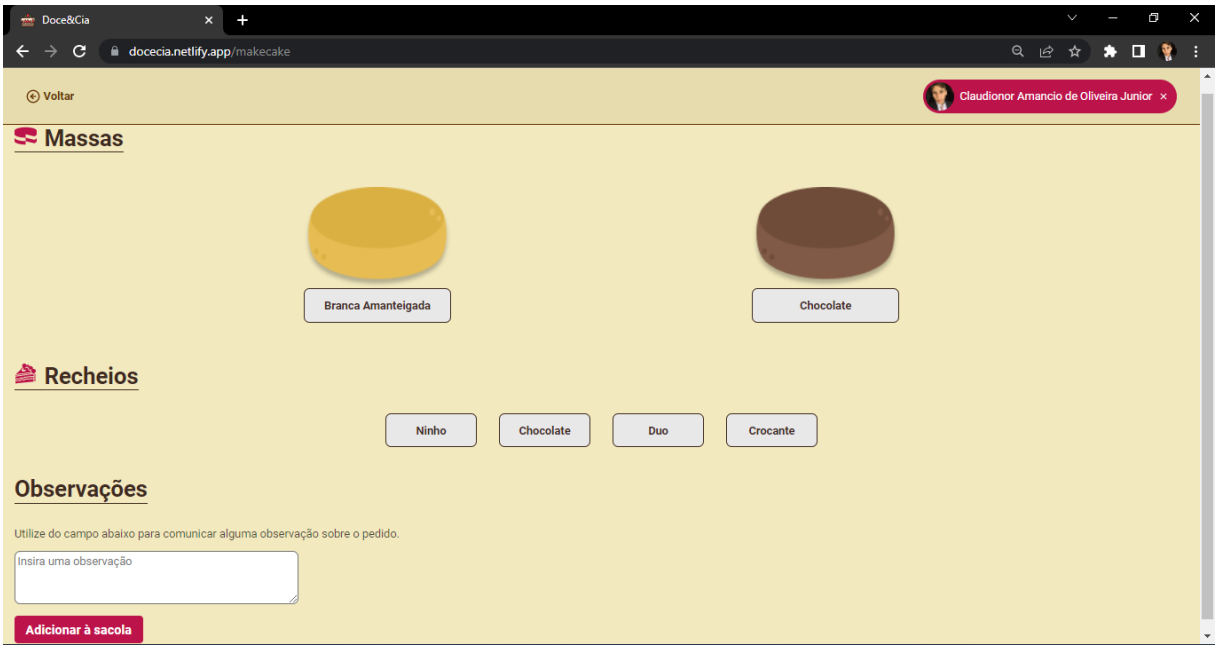
No componente de *card*, visível pela Figura 9, são retornadas informações do bolo, como preço e média de quantas fatias servem, a depender do tamanho escolhido.

Figura 9 – *Card* de opção de bolo.

Fonte – Autoria própria (2022).

Clicando em “Montar bolo” é redirecionado para página de montagem (Figura 10) onde faz a personalização de seu pedido, escolhendo sabores de massa e recheios, podendo adicionar alguma observação ao pedido. Selecionadas suas preferências, ao clicar em “Adicionar à sacola” os dados do pedido são salvos no banco de dados.

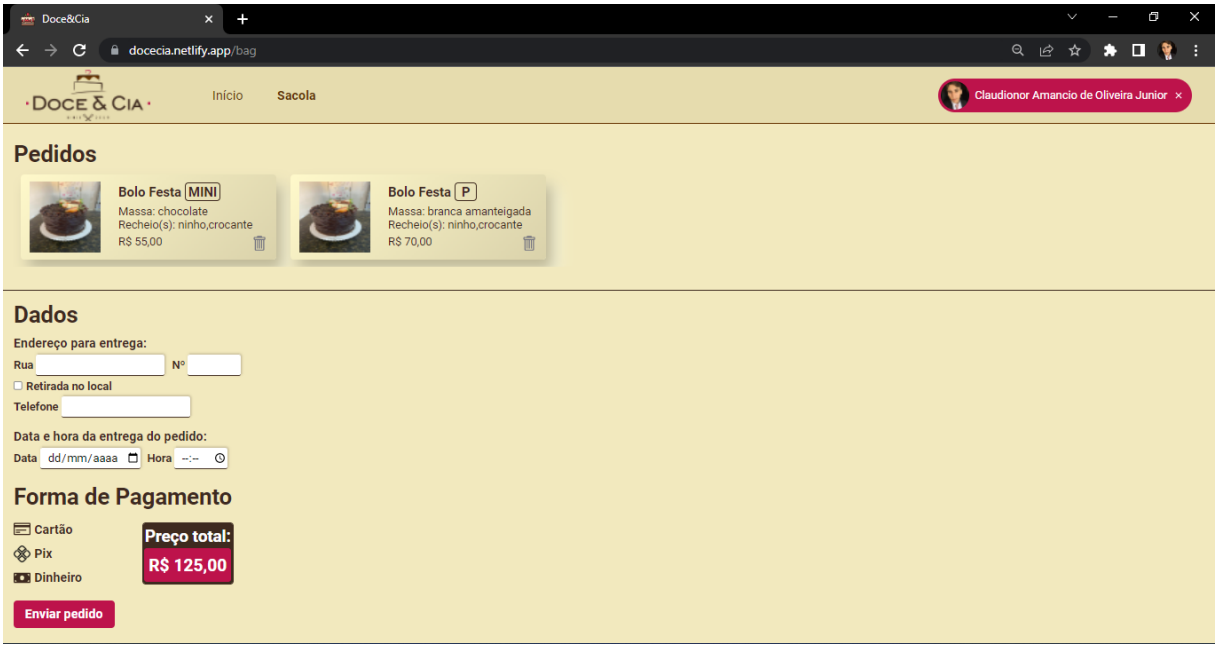
Figura 10 – Tela para montagem do bolo.



Fonte – Autoria própria (2022).

Adicionando o pedido à sacola de compras o usuário é redirecionado para a tela Sacola, onde são mostrados todos os pedidos criados por ele. Também na mesma tela são pedidas algumas informações necessárias para a entrega, como endereço, data e hora de entrega, telefone para contato e forma de pagamento, além do preço total dos pedidos. Veja a Figura 11 abaixo:

Figura 11 – Tela de sacola.



Fonte – Autoria própria (2022).

Os pedidos que foram criados pelo usuário são retornados do banco de dados e listados na seção de pedidos, para que o mesmo possa conferir as informações antes de realizá-los, podendo então, a seu critério, realizar mais de um item em um mesmo pedido. Caso deseje excluir um item basta que clique no ícone de lixeira no canto inferior direito do *card* para que este seja apagado. A Figura 12 demonstra essa seção.

Figura 12 – Seção de pedidos criados.

Fonte – Autoria própria (2022).

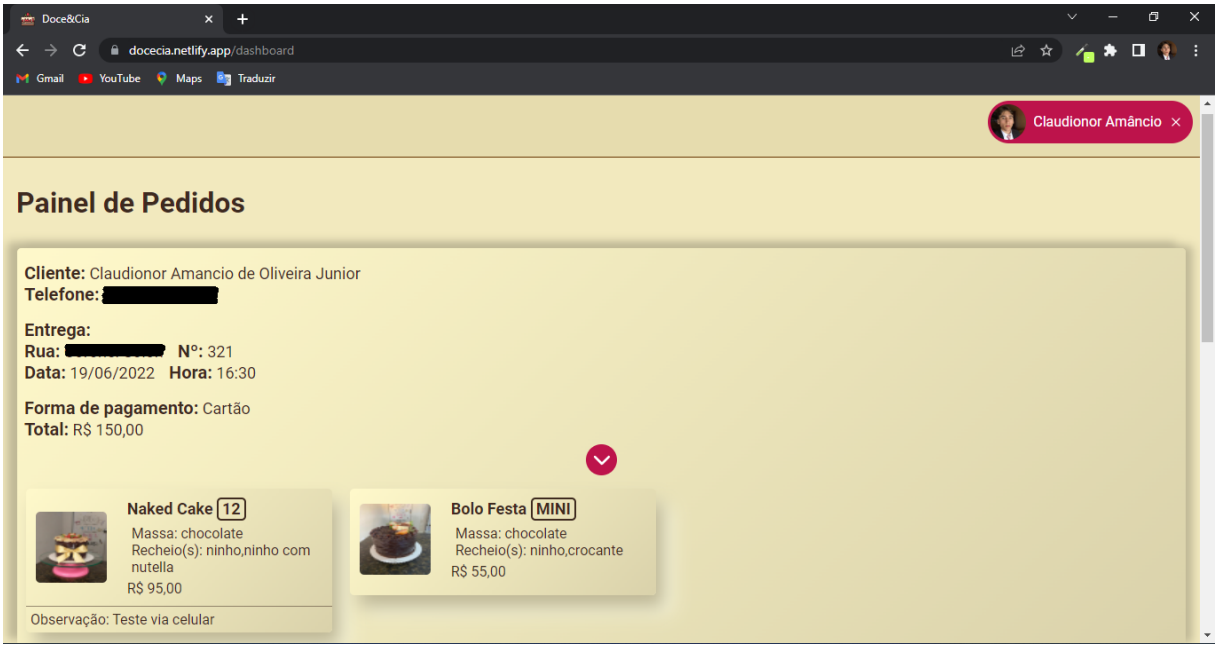
Preenchidas as informações que faltam e pressionando o botão “Enviar pedido” é estruturada uma mensagem com os dados do pedido e encaminhada, utilizando-se da *API* do *Whatsapp*, para o contato do comerciante.

Figura 13 – Tela do *Whatsapp*.

Fonte – Autoria própria (2022).

O empreendedor, através de um usuário administrador, quando logado na aplicação é direcionado para a tela de dashboard (Figura 14). Nesta tela, o administrador pode visualizar os pedidos que foram realizados pelos clientes, retornando quais os itens escolhidos, observações sobres os pedidos, informações para a entrega.

Figura 14 – Tela de dashboard.



Fonte – Autoria própria (2022).

4 CONCLUSÕES E PERSPECTIVAS FUTURAS

Diante da aplicação dos conceitos, tecnologias e desenvolvimento deste trabalho, foi possível a construção de uma aplicação *web front-end*. O Cardápio Digital Doce&Cia, como seu título já enfatiza, caracteriza-se como um cardápio digital, acessível através dos *browsers* de dispositivos *desktop* e móveis, na qual proporciona o autoatendimento ao cliente, podendo autenticar-se na aplicação por meio de uma conta Google, visualizar o cardápio, com suas opções e preços correspondentes, e selecionar o item desejado, escolhendo, dentre as alternativas, os sabores de seu agrado para compor o pedido.

O *software* consegue atender ao principal ponto ao qual foi desenvolvido, que é dar assistência para o empreendedor no atendimento ao cliente, economizando tempo e esforço para esta tarefa, além de também viabilizar o controle dos pedidos realizados, os quais podem ser visualizados dentro da interface da aplicação.

Vale ressaltar que para a escolha do tipo de programa cogitava-se de um aplicativo ou *website*. Logo, pensando na comodidade do cliente e aceitação do uso da proposta, a opção de um *website* se demonstrou mais viável para este caso. Por ser um *software* pensado para uma atividade mais específica, torna seu uso mais limitado, o que leva em consideração a recusa dos clientes em instalar o aplicativo em seus dispositivos, por ocupar

espaço de armazenamento e ter uma utilização mais baixa, além das atualizações mais frequentes necessárias a essas aplicações.

Um *website*, sendo assim, se sobressai, por não haver a necessidade de instalação, não ocupar armazenamento interno do dispositivo e retornar sempre a versão mais atual a cada acesso do usuário.

Visando a continuação, manutenção e aprimoramento do projeto, é pensada a aplicação de futuras funcionalidades. Dentre as melhorias, tem-se a integração de um meio de pagamento dentro do sistema, para que os pagamentos sejam realizados por meio de cartão de crédito e pix, através do *website*, trazendo mais comodidade ao uso e automatizando mais uma tarefa do negócio.

A tela de *dashboard* possibilita ao administrador visualizar dentro da aplicação os pedidos que foram solicitados; uma utilidade a mais pensada para esta página é a implementação de um mecanismo de aceitação ou recusa dos pedidos. Ainda, oferecer também ao administrador a opção de alterar o cardápio, por meio de uma tela de edição, para que o menu possa estar sempre atualizado, proporcionando mais autonomia para o empreendedor.

Outras funcionalidades futuras almejadas são a generalização do cardápio para atender a outros tipos de empreendimentos culinários, salvar os endereços já utilizados pelos clientes e integrar um *bot* para *Whatsapp* que retorna o *status* do pedido.

A princípio, o projeto, em sua versão inicial, visa atender a demanda de um empreendedor familiar-pessoal. Contudo, conforme forem empregados novos mecanismos, oferecendo mais robustez, que esteja aderido a utilização pelos clientes, pretende-se estabelecer uma taxa por pedido ou assinatura para o uso do sistema pelo empreendedor, como forma de monetização, para viabilidade financeira e expansão do negócio.

REFERÊNCIAS

Comércio eletrônico: comida por delivery e supermercados são categorias que mais crescem na pandemia. **G1**, 2021. Disponível em: <https://g1.globo.com/economia/noticia/2021/05/26/comercio-eletronico-comida-por-delivery-e-supermercados-sao-categorias-que-mais-crescem-na-pandemia.ghtml>. Acesso em: 13, dez de 2021.

Conheça a história do e-commerce! **Agência FG**, 2021. Disponível em: <https://agenciafg.com.br/blog/e-commerce/historia-do-e-commerce>. Acesso em: 10, dez de 2021.

CHODOROWSKI, M. (2021). **Comparative analysis of JavaScript package managers - yarn and npm.** *Journal of Computer Sciences Institute*, *19*, 75-80.

Facebook, Inc. Flux: docs, c2022. In-Depth Overview. Disponível em: <https://facebook.github.io/flux/>. Acesso em: 22, mar de 2022.

GitHub, Inc. GitHub Docs: QuickStart, c2022. Hello World. Disponível em: <https://docs.github.com/pt/get-started/quickstart/hello-world>. Acesso em: 25, mar 2022.

Google. Firebase, [s.d.]. Disponível em: <https://firebase.google.com/docs/web/setup>. Acesso em: 29, mar e 2022.

Goomer, c2021. Disponível em: <https://goomer.com.br/precos>. Acesso em: 20, dez 2021.

GUEDES, Gilleanes TA. UML 2. **Uma Abordagem Prática, São Paulo, Novatec**, 2009.

Histórico da pandemia de COVID-19. **OPAS**. Disponível em: <https://www.paho.org/pt/covid19/historico-da-pandemia-covid-19>. Acesso em: 13, dez de 2021.

Ian Sommerville. 2011. **Engenharia de Software.** 9. ed. São Paulo: Pearson Education.

IFood, c2021. Disponível em: <https://menu.ifood.com.br/cadastro?utm_source=Google&utm_medium=Ads&utm_campaign=CardapioDigital_Search&utm_content=cardapiodigital_responsivo_restaurante&utm_term=search09nov&gclid=EAIaIQobChMIgoW1vJrz9AIVDg2RCh3WlgLqEAAYASAAEgJZ6_D_BwE>. Acesso em: 20, dez 2021

JADHAV, Madhuri A.; SAWANT, Balkrishna R.; DESHMUKH, Anushree. Single page application using angularjs. **International Journal of Computer Science and Information Technologies**, v. 6, n. 3, p. 2876-2879, 2015.

Kyte Tecnologia de Software LTDA. **Kyte**. Disponível em: <https://www.kyte.com.br/fidelize/cardapio-digital?gclid=EAIaIQobChMIqOr529Lz9AIVjYKRCh28JQCdEAAYASABEgIsrfD_BwE>. Acesso em: 20, dez 2021

MANSO, Afonso Henrique Dutra. Aplicativos de delivery: análise da percepção dos consumidores e entregadores sobre seus possíveis impactos positivos. 2019.

Meta Platforms, Inc. React, c2022. Disponível em: <https://pt-br.reactjs.org/>. Acesso em: 24, mar 2022.

Microsoft. TypeScript: TypeScript for the New Programmer, c2022. Get Started. Disponível em: <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>. Acesso em: 24, mar. de 2022.

Microsoft. Visual Studio Code: docs, c2022. Getting Started. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 25, mar e 2022.

MONTY, Renata. Consumo de comida por aplicativos: os impactos das materialidades da comunicação em Uber Eats e iFood.

oimenu, c2022. Disponível em: <https://www.oimenu.com.br/cardapio-qr-code>. Acesso em: 20, dez 2021.

O que é BaaS? **Cloudflare**. Disponível em: <https://www.cloudflare.com/pt-br/learning/serverless/glossary/backend-as-a-service-baas/>. Acesso em:22, mar. de 2022.

PAGOTTO, T. et al. **Scrum solo: Software process for individual development.** In: 2016 11th Iberian Conference on Information Systems and Technologies (CISTI). [S.l.]: IEEE, 2016. p. 1–6. ISBN 978-9-8998-4346-2.

PRADO, Gilbertto. Redes e espaços artísticos de intervenção. **MEDEIROS, Maria Beatriz de. Arte em pesquisa: especificidades–Anais do**, v. 13, p. 258-263, 2004.

Remix, React Router: docs, c2022. API Reference. Disponível em: <https://reactrouter.com/docs/en/v6/api>. Acesso em: 25, mar de 2022.

Sass. Sass: documentation, c2022. Disponível em: <https://sass-lang.com/documentation>. Acesso em: 24, mar de 2022.

Whatsapp é o app mais usado pelos brasileiros; veja lista. **Valor**, 2022. Disponível em: <https://valor.globo.com/empresas/noticia/2022/01/11/whatsapp-e-o-aplicativo-mais-utilizado-pelos-brasileiros-veja-lista-de-apps.ghtml>. Acesso em: 22, mar. de 2022.