

CENTRO UNIVERSITÁRIO ESTÁCIO DE SÁ

GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

GEVERSON JOSE DE SOUZA

**MyStore**

Fortaleza-Ceará

2021

GEVERSON JOSE DE SOUZA

**MyStore**

Trabalho apresentado ao curso de Sistemas de Informação do Centro Universitário Estácio do Ceará como requisito para obtenção do título de Bacharel.

Orientador: Prof. Dr. Wellington Sousa Aguiar.

Aprovado em \_\_\_\_\_\_/\_\_\_\_\_\_/\_\_\_\_\_\_\_

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr. Wellington Sousa Aguiar

Centro Universitário Estácio do Ceará (Estácio - FIC)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Cassio Pinheiro

Centro Universitário Estácio do Ceará (Estácio - FIC)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Me. Henrique Nogueira da Gama Mota

Centro Universitário Estácio do Ceará (Estácio - FIC)

**RESUMO**

Atualmente, a quantidade de sistemas desenvolvidos aumenta exponencialmente, sendo que uma parcela significativa desses sistemas se encontra destinado as médias e grandes empresas. O objetivo desta pesquisa é aplicar o conhecimento adquirido no decorrer do curso visando solucionar o problema detectado através de experiências vívidas pelos empreendedores informais, pois a maior parte deles relatam experiências que torna a aquisição de sistemas inviável para o seu negócio. A metodologia utilizada nessa pesquisa foi a pesquisa aplicada e o resultado obtido beneficia os empreendedores informais, foi utilizada também a pesquisa bibliográfica. A partir desse cenário foi possível tirar várias conclusões quanto ao grande volume de sistemas disponíveis, seja destinado ao comércio ou para outros fins, tal tarefa constitui-se um desafio no mundo dos negócios, pois a finalidade é sempre gerar um produto para um usuário que por sua vez tem personalidade única. Os problemas citados foram extraídos de uma pesquisa realizada por meio de entrevista com quatro pequenos empreendedores, quando relataram conviver com os mesmos problemas, sendo que um deles nunca pensou em usar esta solução, pois não estava habituado a tecnologia. Concluindo, após o desenvolvimento desse projeto foi detectado um mercado a explorar com investimento de baixo custo para tratar problemas em um nicho de mercado “esquecido” e lançar um produto com fins lucrativos e um bom potencial de retorno financeiro.

**Palavras-chave**: Gestão, Tecnologia, Sistemas de informação, App mobile.

ABSTRACT

Currently, the amount of developed systems increases exponentially, and a significant portion of these systems is aimed at medium and large companies. The objective of this research is to apply the knowledge acquired during the course in order to solve the problem detected through vivid experiences by informal entrepreneurs, as most of them report experiences that make the acquisition of systems unfeasible for their business. The methodology used in this research was applied research and the result obtained benefits informal entrepreneurs, bibliographical research was also used. From this scenario, it was possible to draw several conclusions about the large volume of systems available, whether for commerce or for other purposes, such a task is a challenge in the business world, as the purpose is always to generate a product for a user who in turn it has a unique personality. The problems mentioned were extracted from a survey conducted through interviews with four small entrepreneurs, when they reported living with the same problems, one of which never thought of using this solution, as he was not used to the technology. In conclusion, after the development of this project, a market was detected to be explored with a low-cost investment to deal with problems in a “forgotten” market niche and launch a product for profit and with a good potential for financial return.

Keywords: Management, Technology, Information systems, App mobile.

Aos meus familiares pelo apoio e compreensão para a realização desse trabalho, e aos amigos, em especial um amigo que infelizmente não está presente neste momento da minha vida, porém foi ele que me apresentou a informática e todos professores do curso que sempre estiveram ao meu lado contribuindo para minha formação.

# **AGRADECIMENTOS**

Agradeço em primeiro lugar a Deus, pelo fôlego de vida, e força para superar as dificuldades enfrentadas no dia a dia.

Aos meus familiares em especial a minha vó Maria Julia de Araújo Vidal, e meu Tio José Cleide Vidal pelo apoio e por sempre me motivar, e a um amigo Anderson Amorim, já falecido, mas não posso esquecer dele, pois ele me apresentou a informática e sempre me incentivou motivou nos estudos.

Ao prof. Dr. Wellington Sousa Aguiar pela paciência e dedicação na orientação deste trabalho. E a todos que direta ou indiretamente fizeram parte da minha formação.

“Perder tempo em aprender coisas que não interessam priva-nos de descobrir coisas interessantes. ”

(Carlos Drummond de Andrade)

Sumário

[AGRADECIMENTOS 6](#_Toc88775816)

[Sumário 8](#_Toc88775817)

[1 INTRODUÇÃO 11](#_Toc88775818)

[1.1 Objetivo Geral 13](#_Toc88775819)

[1.2 Objetivos Específicos 13](#_Toc88775820)

[2 FUNDAMENTAÇÃO TEÓRICA 14](#_Toc88775821)

[2.3 API RESTful 14](#_Toc88775822)

[2.4 Protocolo HTTP 16](#_Toc88775823)

[2.5 JSON 16](#_Toc88775824)

[2.7 Android 17](#_Toc88775825)

[2.8 IOS 17](#_Toc88775826)

[2.11 Java 18](#_Toc88775827)

[2.12 OAuth2 19](#_Toc88775828)

[3 METODOLOGIA 20](#_Toc88775829)

[4 DESENVOLVIMENTO E RESULTADO 21](#_Toc88775830)

[4.1 Analise de Requisitos 21](#_Toc88775831)

[4.1.1 Requisitos Funcionais 21](#_Toc88775832)

[1. RF001 – Menu “Cadastrar Fornecedor”. 21](#_Toc88775833)

[2. RF002 – Menu “Cadastrar Forma de Pagamento”. 21](#_Toc88775834)

[3. RF003 – Menu “Cadastrar Produto”. 21](#_Toc88775835)

[4. RF004 – Menu “Cadastrar Cliente”. 21](#_Toc88775836)

[5. RF005 – Menu “Realizar Venda”. 22](#_Toc88775837)

[6. RF006 – Menu “Registrar Conta Paga”. 22](#_Toc88775838)

[7. RF007 – Menu “Registrar Conta Recebida”. 22](#_Toc88775839)

[8. RF008 – Menu “Visualizar Resultados”. 22](#_Toc88775840)

[4.1.2 Requisitos Não Funcionais 22](#_Toc88775841)

[1. RNF001 – Aplicativo Multiplataforma. 23](#_Toc88775842)

[2. RNF002 – Registrar conta a receber offline. 23](#_Toc88775843)

[3. RNF003 – Design e layout do site devem ser autoexplicativos. 23](#_Toc88775844)

[4. RNF004 – Toda informação do aplicativo é de total responsabilidade do usuário. 23](#_Toc88775845)

[5. RNF005 – Design responsivo. 23](#_Toc88775846)

[4.2 Diagramas 23](#_Toc88775847)

[4.2.1 BPMN 23](#_Toc88775848)

[4.2.2 Diagrama de Atividade 24](#_Toc88775849)

[4.2.3 Diagrama de Sequência 25](#_Toc88775850)

[4.2.4 Diagrama de Classe 26](#_Toc88775851)

[4.2.5 Diagrama de Entidade Relacionamento 28](#_Toc88775852)

[4.3 Ferramentas Utilizadas 28](#_Toc88775853)

[4.3.1 Visual Studio Code 28](#_Toc88775854)

[4.3.2 yEd - Graph Editor 29](#_Toc88775855)

[4.3.3 Github 30](#_Toc88775856)

[4.3.4 Spring Tool Suite 30](#_Toc88775857)

[4.3.5 MySQL Workbanch 31](#_Toc88775858)

[4.4 Resultados 32](#_Toc88775859)

[4.4.1 Apresentação do Aplicativo 32](#_Toc88775860)

[4.4.2 Tela de Login 32](#_Toc88775861)

[4.4.3 Tela de Principal 32](#_Toc88775862)

[4.4.4 OpenAPI 2.0 33](#_Toc88775863)

[4.4.5 Análises e Resultados 34](#_Toc88775864)

[5 CONSLUSÃO 35](#_Toc88775865)

[REFERÊNCIAS BIBLIOGRÁFICAS 36](#_Toc88775866)

**LISTA DE ABREVIATURAS E SIGLAS**

API Application Programming Interface

BPMN Business Process Model and Notation

CNPJ Cadastro Nacional de Pessoa Jurídica

CSS Cascading Style Sheets

DER Diagrama de Entidade de Relaciuonamento

GPS Global Position System

HTML HyperText Markup Language

IDE Integrated Development Environment

JSON JavaScript Object Notation

REST Representational state transfer

RFC Request for Comment

SGBD Sistema de Gerenciamento de Banco de Dados

TI Tecnologia da Informação

UML Unified Modeling Language

# INTRODUÇÃO

Com a evolução da tecnologia o crescimento de sistemas aumenta exponencialmente e junto a esse aumento vem também as dúvidas para a escolha de um sistema que atenda às necessidades de um negócio independente da atividade que este negócio exerce. Este problema é encontrado nas grandes, médias e pequenas empresas. Quando se trata de empreendedor informal este problema aumenta mais ainda, pois além de ter o problema da dúvida há também outros problemas como ambiente reduzido e poder financeiro baixo para a aquisição de ativos de TI (Tecnologia da Informação).

Tendo em vista o que foi mencionado anteriormente, foi estudado os meios como os envolvidos nesse cenário trabalham atualmente para controlar e acompanhar a evolução e os resultados do seu negócio. Os meios utilizados são: planilhas; agendas; notas promissórias; no pior cenário o meio utilizado é a própria mente para saber o que comprou, vendeu e tem a receber;

As pessoas entrevistadas têm atividades comerciais no ramo de moda, acessórios, confecção de roupas e produtos de limpeza. No dia-a-dia de cada um deles as atividades operacionais do seu respectivo negócio o procedimento é parecido, onde o mesmo vende para conhecidos ou indicações de seus amigos. E as pessoas que realiza estas compras, ou seja, o cliente faz o pedido do produto escolhido escolhe a forma de pagamento que em alguns casos é fiado, porém com a facilidade que atualmente existe de possuir cartão de crédito isso tem reduzido bastante. e em seguida o empreendedor realiza as suas anotações.

Para o desenvolvimento deste projeto foi criado dois subprojetos onde um subprojeto consiste na iteração do usuário com o dispositivo, utilizando a biblioteca React Native criada pelo Facebook, esta biblioteca permite aos desenvolvedores criarem aplicativos para as plataformas iOS e Android com as melhores partes do desenvolvimento nativo. O segundo subprojeto é responsável por centralizar as informações geradas pelo usuário e a regra do negócio utilizando a linguagem Java para a construção de uma API RESTful seguindo os padrões da RFC (Request for Comment) 6749 para o tráfego de dados na internet. Para os dados trafegarem pela internet com segurança foi utilizado o padrão OAuth que também segue o padrão da RFC 6749.

Em uma pesquisa domiciliar realizada pelo SEBRAE-SP no período de 11 de novembro de 2002 à 5 de março de 2003 a partir de uma amostra com 1.600 empreendedores e 300 candidatos a empreendedores, foi chegado a uma conclusão que em termos de gestão empresarial, as principais necessidades estão na área financeira (p.ex. custos/ preços/ prazos/ fluxo de caixa/ margem de lucro) e relação com clientes/ mercado (p. ex. conquistar clientes/ imagem/ oportunidades). Nesta mesma pesquisa concluiu-se que os principais obstáculos à formalização dos negócios são: a burocracia e o peso da carga tributária. Na pesquisa também é possível observar a grande diferença nos tipos de ocupação conforme mostra a figura 1 abaixo, a porcentagem refere-se a 358 empreendedores formais e 1242 empreendedores informais(https://www.sebrae.com.br/Sebrae/Portal%20Sebrae/UFs/SP/Pesquisas/empreendedores.pdf).

Figura 1 – Empreendedores por tipo de ocupação.



Fonte: Sebrae-SP.

Empreendedor informal é o empregador que trabalha por conta-própria ou que não possui registro regulado pelo governo e sem CNPJ (Cadastro Nacional de Pessoa Jurídica), ou seja, não paga os encargos tributários devidos pois para o governo o mesmo não existe. Estes são alguns exemplos de empreendedores informais: bancas de calçada; lojas de garagem; vendas diretas.

O contrário do empreendedor informal o empreendedor formal é o empregador que possui registro regulado pelo governo ou com CNPJ, sendo assim o mesmo é obrigado a pagar os encargos tributários devidos.

O grande número de empreendedores informais requer uma atenção, pois o sistema existente tem o foco em grandes, médias e pequenas empresas, e pode levar em conta a complexidade e o custo de manter um ambiente de TI, que acaba sendo um dos fatores que tornam inviável os envolvidos deste mercado pensar em uma solução para o perfil do negócio.

Considerando esses problemas observados e citados por parte dos usuários, foi pensada a criação de um aplicativo que mostre o resultado de forma objetiva, através dos dados informados pelos usuários do sistema e o aplicativo mostrará os resultados do negócio. Este aplicativo irá registrar as vendas e os recebimentos das vendas realizadas e também será realizado o cadastro de clientes sendo obrigatório o cadastro do mesmo em caso de venda que seja realizado com a modalidade de pagamento a prazo.

# Objetivo Geral

Desenvolver um aplicativo que automatize o processo de vendas, controle financeiro e cadastro de cliente facilitando o dia-a-dia dos empreendedores informais.

# Objetivos Específicos

1. Registrar as vendas realizadas para que possa gerar ´relatórios gerencias do negócio.
2. Registrar as contas a receber de acordo com as vendas realizadas.
3. Controlar e analisar o cadastro de clientes.
4. Aplicar técnicas de desenvolvimento de software estudadas na graduação.
5. Apresentar um aplicativo mobile usando as técnicas de levantamento de requisitos, diagramas UML, React, React Native, Javascript, HTML CSS e Java.

# FUNDAMENTAÇÃO TEÓRICA

* 1. Sistemas de Informação (SI)

Sistemas de Informação é um termo utilizado para descrever um sistema seja ele de computadores, pessoas ou métodos que tenham o objetivo de armazenar, processar ou distribuir informações para apoiar a tomada de decisões, a coordenação e o controle de uma organização.

* 1. Interface de Programação de Aplicações (API)

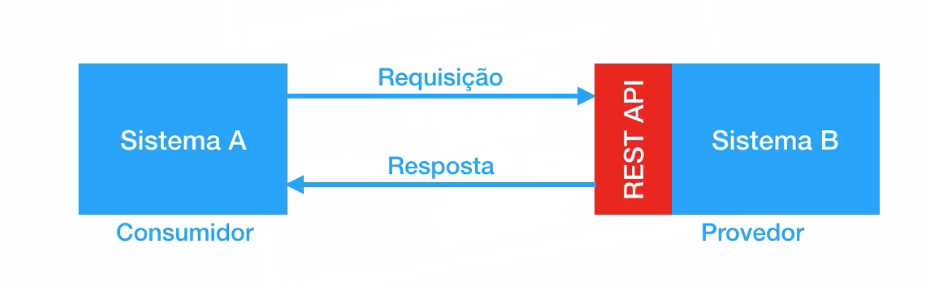
APIs é um software que possui um conjunto de funções que intermedia o acesso as funcionalidades de um sistema, componentes de sistema ou serviço, ou seja, uma API tem como função prover suas funcionalidades para que algum outro sistema consiga usa-la, cada dia que se passa além de sua utilização o termo também vem sendo bastante citado e solicitado para o mercado de tecnologia, quem vai construir uma API tem que sempre pensar em quem vai usa-la pois o uma API bem elaborada atrai os consumidores. O crescimento do uso das APIs deve-se ao crescimento dos dispositivos móveis onde cada dia que se passa o acesso a eles está cada vez mais fácil, sendo assim este crescimento faz com que mais serviços sejam disponibilizados (GALVÃO; SOARES; KAI, 2017).

# API RESTful

API é um termo muito amplo, para esta pesquisa usaremos API REST onde esta tem por objetivo expor a as funcionalidades de um sistema de vendas e controle financeiro onde a comunicação com esta API será feita através de um aplicativo. REST (REpresetantion State Transfer) é um modelo arquitetural para o desenvolvimento de webservices, estes que transmitem dados pela internet por meio dos protocolos http e independente da linguagem de programação em que o mesmo foi desenvolvido.

REST surgiu no início dos anos 2000 a partir da tese de Ph.D de um cientista chamado Roy Fielding com o intuito de criar um conjunto de melhores práticas e regras para o desenvolvimento de webservices, e estas melhores práticas são chamadas de constraints (restrições). A figura 2 descreve a comunicação utilizando esta arquitetura (FIELDING, 2000).

Figura 2 – Descrição do modelo REST.



Fonte: Google Imagens, 2021.

A figura 2 demonstra da forma mais resumida possível de como é o funcionamento de uma API REST, sempre vai haver quem prover a API REST nesta situação chamamos de aplicação servidor (Provedor) que é quem vai disponibilizar as funcionalidades, e do outro lado temos o consumidor que são as aplicações clientes, que é o que vai utilizar as funcionalidades disponíveis pelo servidor.

Mais porque a utilização deste modelo arquitetural REST? Segue alguns dos vários motivos para este modelo arquitetural está em crescimento: Separação entre aplicações cliente servidor; Escalabilidade; Independência da linguagem; E a necessidade de integração com os mais diversos tipos de sistemas existentes.

As constraints do REST são : Cliente-servidor essa constraint diz que as aplicações devem poder evoluir separadamente sem qualquer dependência desde que sua interface continue inalterada; Outra constraint é a stateless nesta diz que toda requisição deve ser sem estado, ou seja, tudo que a requisição precisar para aquela operação tem que está contida na mesma requisição; Outra constraint é a de cache que é a possibilidade de fazer cache das respostas para melhorar escalabilidade, consumindo menos recurso computacional; Interface uniforme e bem definida onde uma vez definida devemos seguir rigorosamente para esta constraint os verbos http nos ajudam a seguir as interfaces que uma vez foi definida; Outra constraint é sistema em camada, como entre as aplicações cliente e servidor podem ter outros servidores(proxy, balanceador de carga, etc), estes não devem afetar a requisição; E por fim a última constraint é código sobre demanda essa trata que o servidor pode retornar como resposta junto a uma requisição um código que deve ser executado no cliente (Fielding, 2000).

Segundo Roy Fielding existem quatro princípios de diretrizes que nos ajuda a seguir essas constraints são eles:

1. Baseado em recursos: os recursos individuais são identificados nas solicitações. Por exemplo: URL/clientes.
2. Manipulação de Recursos Através de Representações: O cliente possui representação de recurso e contém informações suficientes para modificar ou excluir o recurso no servidor, desde que tenha permissão para tal. Exemplo: Normalmente, o usuário obtém um ID de cliente quando solicita uma lista de usuários e, em seguida, usa esse ID para excluir ou modificar esse usuário específico.
3. Mensagens auto descritivas: cada mensagem inclui informações suficientes para descrever como processar a mensagem para que o servidor possa analisar facilmente a solicitação.
4. Hipermídia como Motor de Estado da Aplicação (HATEOAS): Necessita incluir links para cada resposta para que o cliente descubra facilmente outros recursos.

# Protocolo HTTP

O protocolo de transferência usado sobre a World Wide Web é o HTTP (Hypertext Transfer Protocol). Ele especifica quais mensagens podem ser enviadas para o servidor e quais respostas serão retornadas ao cliente [(JUNIOR, 2008)](#_bookmark42). Principais métodos/verbos e funções existentes na requisição do protocolo HTTP:

* GET: Solicita que seja retornado o recurso identificado pela URL
* PUT: Atualiza os dados do recurso existente.
* POST: Envia dados para serem processados para o recurso especificado.
* DELETE: Remove recurso conforme a solicitação.

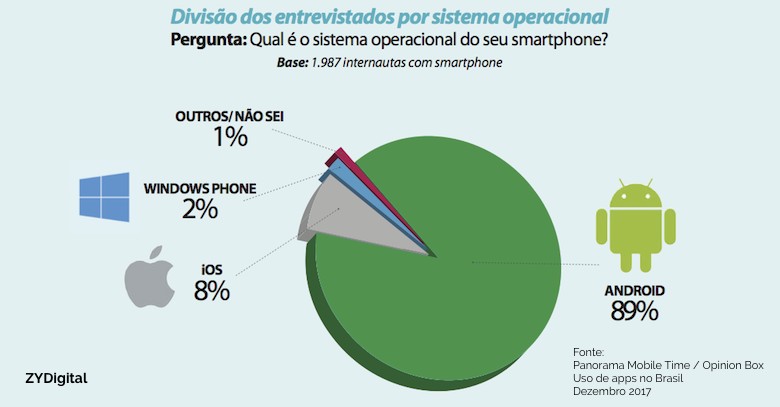
# JSON

O JSON (Javascript Object Notation) é um mecanismo de codificação/decodificação de valores para intercâmbio de dados. É uma linguagem textual, e um subconjunto de Javascript, leve e muito fácil de analisar. Sendo assim JSON tem sido amplamente adotado no meio acadêmico e por diversas empresas, principalmente devido ao fato de ser altamente produtivo em aplicações distribuídas e um formato ideal para transmissão de dados orientado a objetos através da rede [(PULUCENO, 2012)](#_bookmark49).

* 1. Sistemas Operacionais para Dispositivos Móveis

A Panorama Mobile Time e a Opinion Box realizaram uma pesquisa em 2017 sobre o uso de aplicativos no Brasil. Segundo a pesquisa, o Android é o sistema operacional de 89% dos smartphones em funcionamento no Brasil. O iOS (iPhone) representa 8%, e o Windows Phone apenas 2%. A pesquisa foi realizada com 1987 pessoas. Como mostra na figura 3 a seguir (https://www.mobiletime.com.br/pesquisas/).

Figura 3 – Uso de aplicativos no Brasil – dezembro de 2017.



Fonte:[(TIME; BOX, 2017)](#_bookmark54)

# Android

O Android começou com uma pequena empresa com o nome do próprio Android Inc, em Palo Alto, Califórnia, EUA, criada por quatro sócios, chamados Andy Rubin, Rich Miner, Nick Sears e Chris White. Somente em 2006 o Google tomou posse do projeto comprando a Android Inc e assumindo a direção do desenvolvimento do sistema. Este sistema é baseado em uma vesão modificada do kernel Linux projetado para dispositivos móveis e com tela de toque, é um software gratuito e de código aberto (NICOLAI, 2012).

# IOS

O iOS é o sistema operacional para dispositivos móveis fabricados pela Apple, é derivado do Mac OS X (sistema operacional utilizado nos desktops e notebooks do mesmo fabricante). Lançado oficialmente em junho de 2007 primeiramente para iPhone, o sistema também é a base para três outros sistemas operacionais feitos pela Apple: iPadOS, [tvOS](https://en.wikipedia.org/wiki/TvOS" \o "TvOS) e [watchOS](https://en.wikipedia.org/wiki/WatchOS). É um software proprietário, embora algumas partes sejam de código aberto sob a [Licença de Código Público](https://en.wikipedia.org/wiki/Apple_Public_Source_License) da [Apple](https://en.wikipedia.org/wiki/Apple_Public_Source_License) e outras licenças, este sistema é compatível com outros aparelhos da empresa, como iPod, iPad, iPod Touch e Apple TV [(RECH, 2013)](#_bookmark50).

* 1. Aplicativos Híbridos

Os aplicativos híbridos são aplicativos que utilizam tecnologias web padrão (HTML, CSS e JavaScript) e podem acessar recursos nativos de cada plataforma mobile, como por exemplo: câmera, GPS, acelerômetro, etc.

No desenvolvimento híbrido, ao invés de dar manutenção em Java (Android), Swift (iOS) e C# (Windows Phone), o desenvolvedor irá se dedicar a apenas uma base de código. Para que alguns recursos nativos dos dispositivos possam ser acessados em diferentes plataformas (GONÇALVES, 2017).

“O desenvolvimento de aplicativos híbridos pode ser a melhor alternativa para o mercado atual, pois reduz os custos com o tempo de desenvolvimento e os recursos humanos necessários, visto que nesta modalidade não são necessários profissionais dedicados ao desenvolvimento específico para cada plataforma de software disponível no mercado” (Gonçalves, 2017).

* 1. React Native

O React Native foi criado em 2015 pelo Facebook. O foco do projeto foi a construção de uma estrutura que se concentrasse no desempenho com os padrões modernos da web de forma que seus componentes sejam renderizados de forma independente, ou seja, cada componente é associado ao estado de um determinado dado, quando este dado é modificado o componente é atualizado. Desde então foram lançadas algumas versões com mudanças significativas entre a primeira versão e a versão atual (EISENMAN; BONNIE, 2015).

O React Native foi criado utilizando o framework React, com apenas um código é possível criar aplicativos para as plataformas Android e iOS, sendo estas as mais utilizadas atualmente. O React Native conta ainda com um conjunto de ferramentas para auxiliar na criação de aplicativos com interface gráfica responsiva e interações muito semelhantes aos dispositivos nativos, de modo que a experiência de usuário não seja diferente no uso de uma aplicação nativa e multiplataforma (MATOS; SILVA, 2016).

O framework React foi criado por Jordan Walke, um engenheiro de software do Facebook e implementado pela primeira vez no feed notícias do Facebook em 2011 e em 2012 no Instagram, e tem como linguagem de programação o Javascript.

Javascript é uma linguagem de programação de alto nível criada em 1995 Brendan Eich a serviço da Netscape, seu propósito era resolver um problema que havia na época em que as páginas eram estáticas e as aplicações web apesar de terem surgido na época havia um crescente cenário de desenvolvimento web (EISENMAN; BONNIE, 2015).

# Java

Java é uma linguagem de programação que surgiu em 1995 em um projeto do grupo de funcionários da Sun Microsystems e teve como coordenador James Gosling. O intuito era possibilitar a criação de programas para dispositivos móveis, além disso essa linguagem se tornou bastante popular por conta que prometia os programadores escrevesse apenas um código e o mesmo fosse possível executar em qualquer lugar sem a necessidade de recompilar o código fonte (ROSSI; 2011).

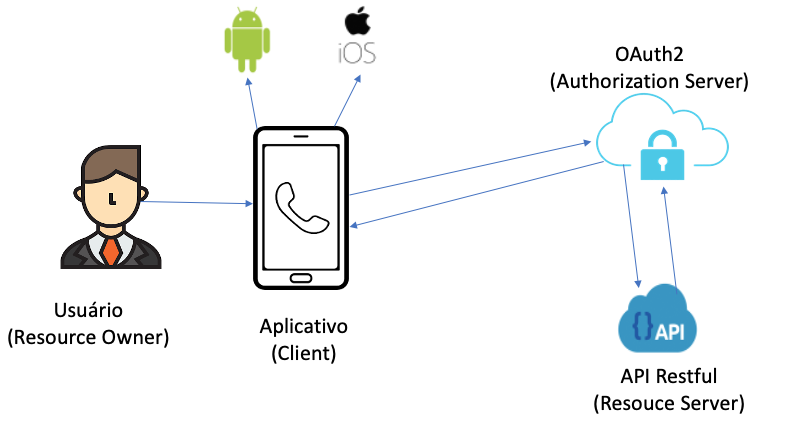
# OAuth2

O OAuth é um protocolo padrão para autorização de aplicação, e o OAuth2 é o OAuth na versão 2.0 seguindo a especificação da RFC 6749 (<https://datatracker.ietf.org/doc/html/rfc6749>), em outras palavras este protocolo é o responsável por garantir o acesso seguro a API RESTful. O OAth2 define 4 roles (papéis) sendo eles:

1. Resource Owner: O usuário proprietário dos recursos que serão utilizados.
2. Client: A aplicação que irá acessar os recursos protegidos em nome do resource owner, ou em seu próprio nome.
3. Authorization Server: Servidor que autentica o servidor qualquer recurso protegido.
4. Resource Server: Servidor que hospeda os recursos protegidos, ou seja, a API RESTful.

A figura 4 a seguir mostra como é este fluxo no ambiente do mundo real.

Figura 4 – Fluxo do consumo de uma API RESTful.



Fonte:Própria, 2021.

# METODOLOGIA

O estudo do projeto caracteriza-se como uma pesquisa aplicada, onde os conceitos estudados são aplicados visando a utilização no mundo real. Segundo Gil [(2010)](#_bookmark39) Pesquisa Aplicada “é voltado à aquisição de conhecimentos com vistas à aplicação numa situação específica”.

A natureza desta pesquisa é de cunho qualitativa. Segundo Martinelli (1999), uma consideração importante é que a pesquisa qualitativa é participante, ou seja, nós também somos sujeitos da pesquisa.

A pesquisa qualitativa, pode ser definida como uma metodologia que produz dados a partir de observações extraídas diretamente do estudo de pessoas, lugares ou processos com os quais o pesquisador procura estabelecer uma interação direta para compreender os fenômenos estudados [(GUILHOTO, 2002)](#_bookmark41).

O estudo utilizou a metodologia de pesquisa bibliográfica, por meio de livros, artigos, teses, dissertações, anais de congressos, sites e etc. visando criar um conteúdo completo e seguro sobre o assunto estudado.

Para a elaboração deste trabalho foi desenvolvido um aplicativo e disponibilizado na Apple Store e Google Play para a registrar as vendas realizadas, contas a receber e contas a pagar, baseado na regra de negócio validada com o usuário do aplicativo.

# DESENVOLVIMENTO E RESULTADO

# Analise de Requisitos

# Requisitos Funcionais

Os requisitos funcionais especificam as funções que o sistema ou seus componentes deverão ser capazes de realizar, podem ser definidos como atividade que o software ou sistema faz ou fará com o objetivo de capacitar os usuários a realizar suas funções. Esses requisitos devem ser bem definidos e descritos claramente para que não exista ambiguidade na sua funcionalidade (OLIVEIRA SALMON, 2017). A seguir será apresentado os requisitos funcionais definidos para o aplicativo desenvolvido.

## RF001 – Menu “Cadastrar Fornecedor”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode cadastrar o fornecedor.

**Pré-requisitos:**

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de cadastro, inserir as informações do fornecedor.

**Fluxo principal:**

1. RF001 - Selecionar o menu de cadastrar fornecedor.

## RF002 – Menu “Cadastrar Forma de Pagamento”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode cadastrar a forma de pagamento.

**Pré-requisitos:**

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de cadastro, inserir as informações da forma de pagamento.

**Fluxo principal:**

1. RF002 - Selecionar o menu de cadastrar forma de pagamento.

## RF003 – Menu “Cadastrar Produto”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode cadastrar o produto, fazendo com que o mesmo exiba as demais opções referentes aos requisitos: RF001 e RF002.

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de cadastro de produto, inserir as informações do produto.

**Fluxo principal:**

1. RF003 - Selecionar o menu de cadastrar produto.

## RF004 – Menu “Cadastrar Cliente”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode cadastrar o cliente, fazendo com que o mesmo exiba as demais opções referentes aos requisitos: RF001, RF002 e RF003.

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de cadastro de cliente, inserir as informações do cliente.

**Fluxo principal:**

1. RF004 - Selecionar o menu de cadastrar cliente.

## RF005 – Menu “Realizar Venda”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode registrar a venda, caso o usuário deseje o mesmo exiba as demais opções referentes aos requisitos: RF002, RF003 e RF004.

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de realizar venda, inserir as informações da venda.

**Fluxo principal:**

1. RF005 - Selecionar o menu de realizar venda.

## RF006 – Menu “Registrar Conta Paga”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode registrar uma conta paga, fazendo com que o mesmo exiba as demais opções referentes aos requisitos: RF002.

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de registrar conta paga, inserir as informações da conta que foi paga.

**Fluxo principal:**

1. RF006 - Selecionar o menu de registrar conta paga.

## RF007 – Menu “Registrar Conta Recebida”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode registrar uma conta recebida, fazendo com que o mesmo exiba as demais opções referentes aos requisitos: RF002.

* Realizar o login no aplicativo.

**Regra de negócio:**

* Na tela de registrar conta recebida, inserir as informações da conta que foi paga.

**Fluxo principal:**

1. RF007 - Selecionar o menu de registrar conta recebida.

## RF008 – Menu “Visualizar Resultados”.

Funcionalidade que através de um menu localizado na página principal onde usuário pode visualizar os resultados referentes aos requisitos: RF007 e RF008.

* Realizar o login no APP.

**Regra de negócio:**

* Na tela de visualizar resultados, inserir as informações do filtro dos resultados.

**Fluxo principal:**

1. RF008 - Selecionar o menu de visualizar resultados.

# Requisitos Não-Funcionais

Os requisitos Não-Funcionais descrevem as restrições e condições dos requisitos funcionais, como por exemplo: adaptabilidade, confiabilidade, eficiência, performance, portabilidade, flexibilidade, usabilidade, entre outros. Os requisitos não funcionais são mensuráveis, e podem estar associados a uma medida ou referência. A seguir será apresentado os requisitos não funcionais definidos para o aplicativo desenvolvido.

## RNF001 – Aplicativo Multiplataforma.

Permitir que o aplicativo seja instalado nos sistemas operacionais Android e iOS.

## RNF002 – Registrar conta a receber offline.

O aplicativo deve permitir registrar uma conta a pagar mesmo estando sem internet.

## RNF003 – Design e layout do site devem ser autoexplicativos.

O Requisito para facilitar o entendimento dos usuários em relação ao aplicativo, deve fazer uso de ícones intuitivos e legendas e até mesmo vídeos, facilitando o uso e a aprendizagem do aplicativo. Podendo até mesmo ser descartado a necessidade de algum tipo de treinamento prévio.

## RNF004 – Toda informação do aplicativo é de total responsabilidade do usuário.

O sistema deverá atender às normas legais, tais como padrões, leis, etc. Assim como as informações inseridas será de inteira responsabilidade do usuário.

## RNF005 – Design responsivo.

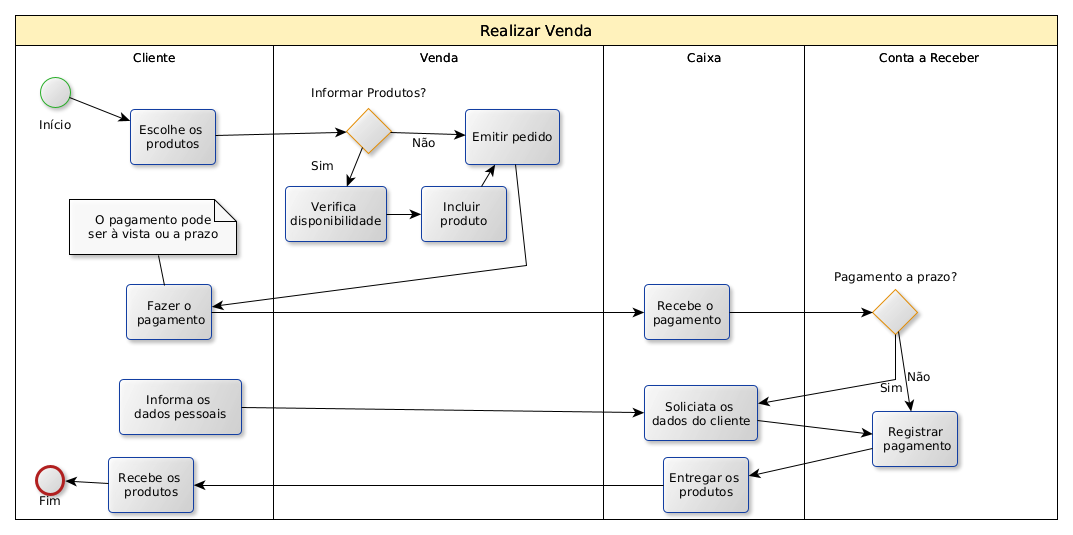
É uma técnica de estruturação do layout para que o aplicativo se adapte de acordo com a resolução em que ele está sendo visualizado, ou seja, o layout deve manter coesão e uma boa experiência ao usuário independentemente do dispositivo. Mesmo que seja em dispositivos menores.

# Diagramas

# BPMN

O BPMN (Business Process Model and Notation) é um padrão de modelagem importante no projeto de sistemas de software orientados a processos, serviços da web e arquiteturas orientadas a serviços. Tem uma absorção significativa na comunidade de analistas de sistema, negócios e processos, e é usado para áreas de aplicação do modelo de processo, como análise de negócios, especificação de fluxo de trabalho, análise de requisitos e configuração de sistemas (RECKER, 2011). A figura 5 a seguir mostra como o processo de venda é realizado pelo usuário.

Figura 5 – Diagrama de BPMN.



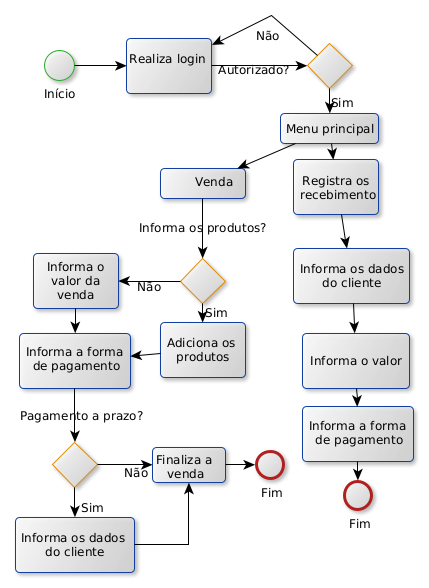
Fonte:Própria, 2021.

# Diagrama de Atividade

O diagrama de atividades é um modelo gráfico que analisa a estrutura e as etapas sequenciais em um processo computacional, dando ênfase ao fluxo de controle de uma etapa para outra (RODRIGUES, 2013).

O usuário ao entrar no aplicativo será apresentado o menu principal do aplicativo, a figura 6 a seguir mostra as 2 principais funcionalidades do aplicativo.

Figura 6 – Diagrama de Atividade.

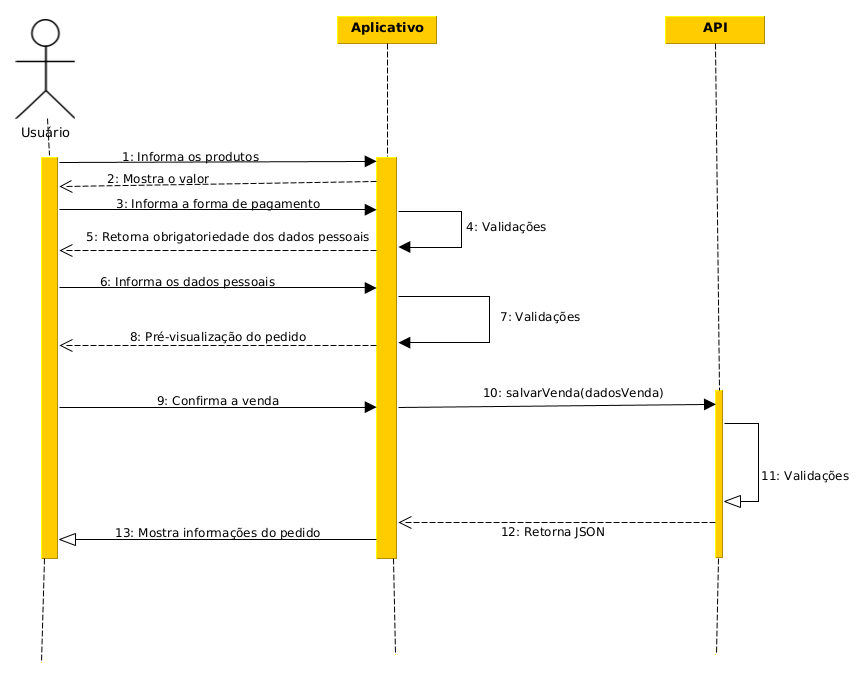


Fonte:Própria, 2021.

# Diagrama de Sequência

O diagrama de sequência é comportamental e dá importância à ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo. Um diagrama de sequência costuma identificar o evento gerador do processo modelado, bem como o ator responsável por esse evento, determina também como o processo deve se desenvolver e ser concluído por meio da chamada de métodos disparados por mensagens enviadas entre objetos (WILCZEK, 2016). A figura 7 apresenta o diagrama e a sequência que os objetos seguem entre si.

Figura 7 – Diagrama de Sequência.



Fonte:Própria, 2021.

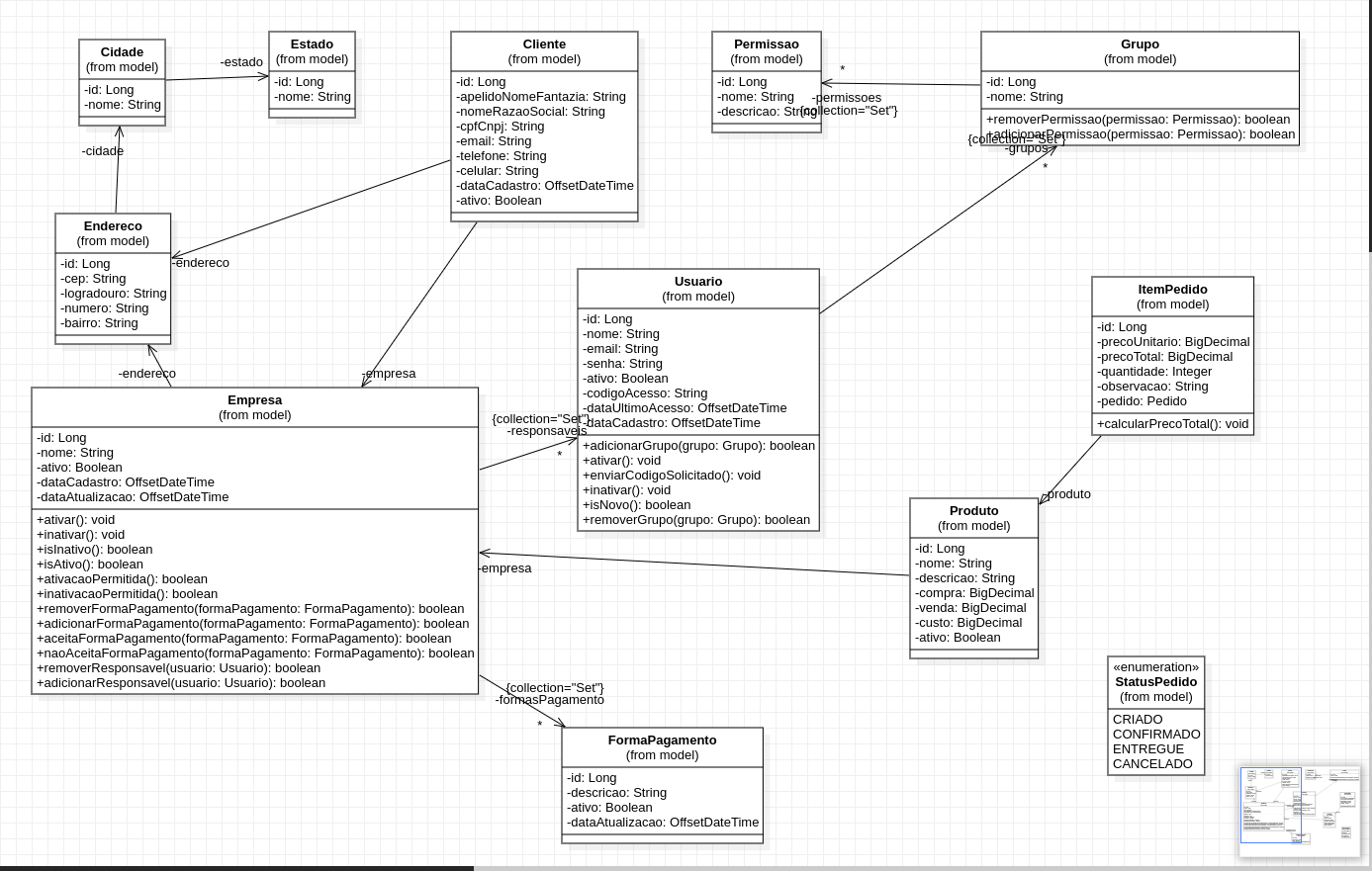
# Diagrama de Classe

O diagrama de classes é fundamental na modelagem de um software, como demonstra (Melo; 2010), afirmando que o referido é a estrela principal de um sistema orientado a objetos. Tal diagrama tem como principal finalidade detalhar as classes pertencentes ao modelo e identificar seus relacionamentos.

A figura 8 a seguir – Diagrama de Classe, mostra as entidades do sistema, sendo elas: Estado, Cidade, Endereço, Empresa, Cliente, Forma de Pagamento, Produto, Pedido, Grupo, Usuário e Permissão.

A classe Estado possui as propriedades básicas para manipular os dados dos estados que serão vinculados a classe Cidade. A classe Cidade possui as propriedades básicas para armazenar os dados das cidades que serão cadastradas e que serão vinculadas a classe Endereço. A classe Endereço possui as propriedades para armazenar os dados referentes aos endereços que serão vinculados as classes Empresa e Cliente. A classe Empresa possui as propriedades que armazenará os dados das empresas e estas Empresas serão vinculadas aos usuários cadastrados. A classe Cliente possui as propriedades para armazenar e manipular os dados dos clientes que serão utilizados na classe Pedido. A classe pedidos possui as propriedades que armazenará os dados das vendas realizadas. A classe Usuário possui as propriedades que o usuário utilizará para realizar o login no aplicativo, este usuário estará associado a classe Grupo. A classe grupo possui as propriedades para armazenar as permissões dos usuários. A classe Permissão possui as permissões que o aplicativo disponibilizará.

Figura 8 – Diagrama de Classe.

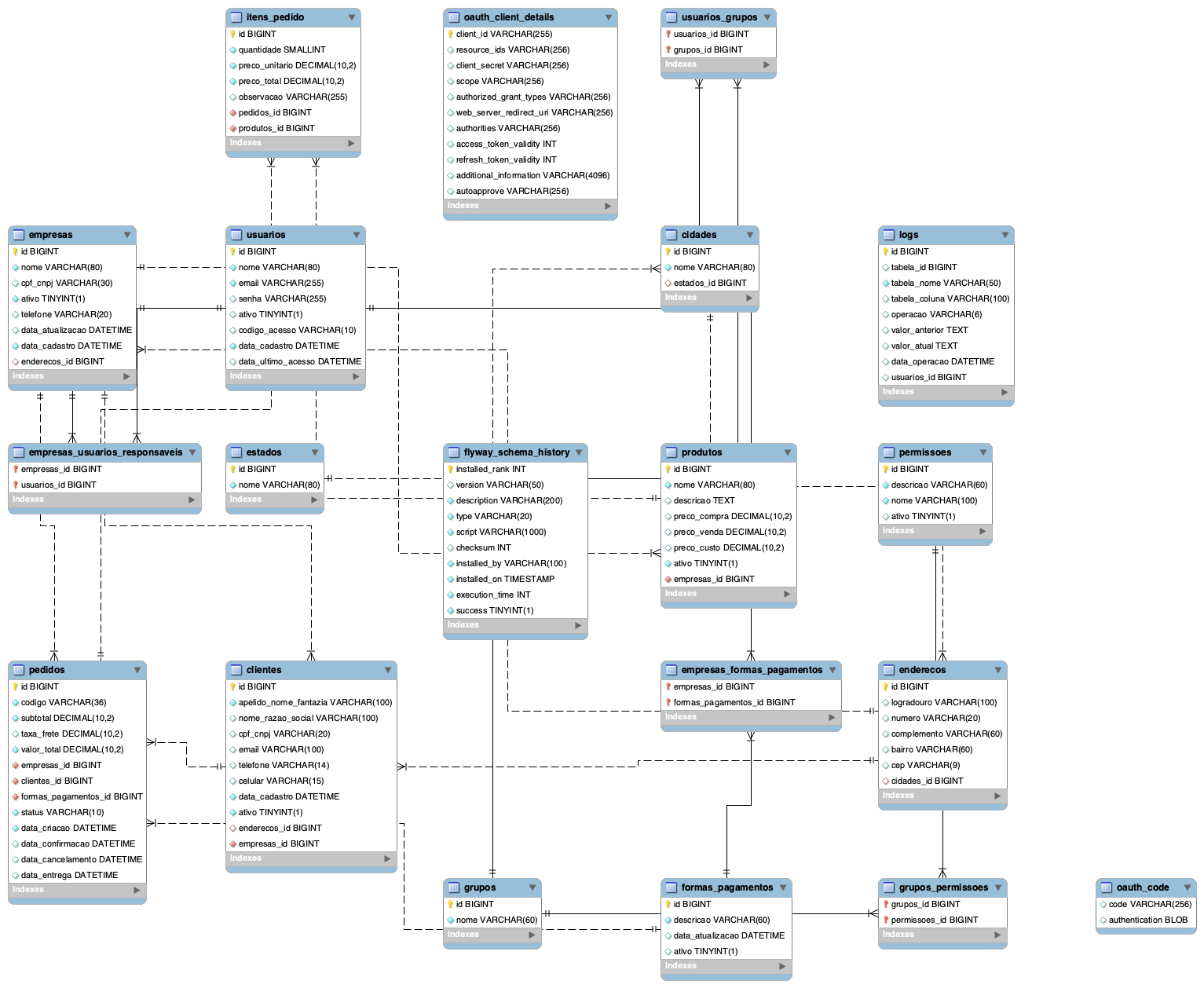


Fonte:Própria, 2021.

# DER (Diagrama de Entidade Relacionamento)

Diagrama de Entidade Relacionamento é um modelo conceitual que descreve o banco de dados de forma independente de implementação em um SGBD (Sistema de Gerenciamento de Banco de Dados). Este modelo registra que dados podem aparecer no banco de dados, mas não registra como estes dados estão armazenados em um SGBD (Rubio, 2012). A figura 8 mostrada apresenta como ficou este o diagrama deste projeto com seus respectivos relacionamentos.

Figura 8 – Diagrama de Classe.



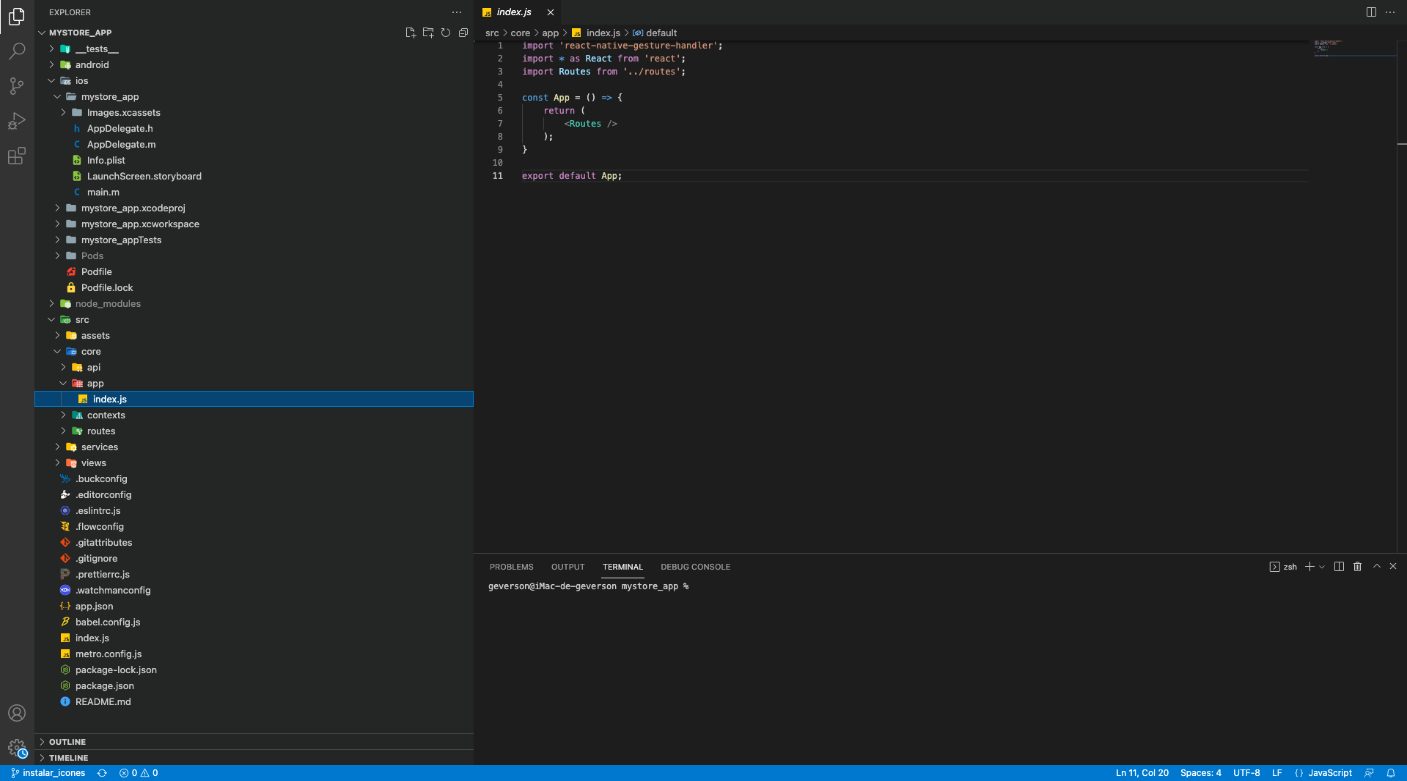
Fonte:Própria, 2021.

# Ferramentas Utilizadas

# Visual Studio Code

O Visual Studio Code é uma IDE (Integrated Development Environment) ​open source de desenvolvimento de aplicações web, lançado pela Microsoft em 2015 com o objetivo de fornecer suporte a mais de 30 linguagens de programação. Sendo multiplataforma ele pode ser executado em qualquer sistema operacional (Mac OS X, Linux e Windows). Além disso, o seu código está disponibilizado no GitHub, e isso permite contribuições com seu desenvolvimento (MACORATTI, 2016). A figura 9 apresenta uma breve ilustração da utilização desta IDE durante o desenvolvimento do aplicativo.

Figura 9 – Visual Studio Code.

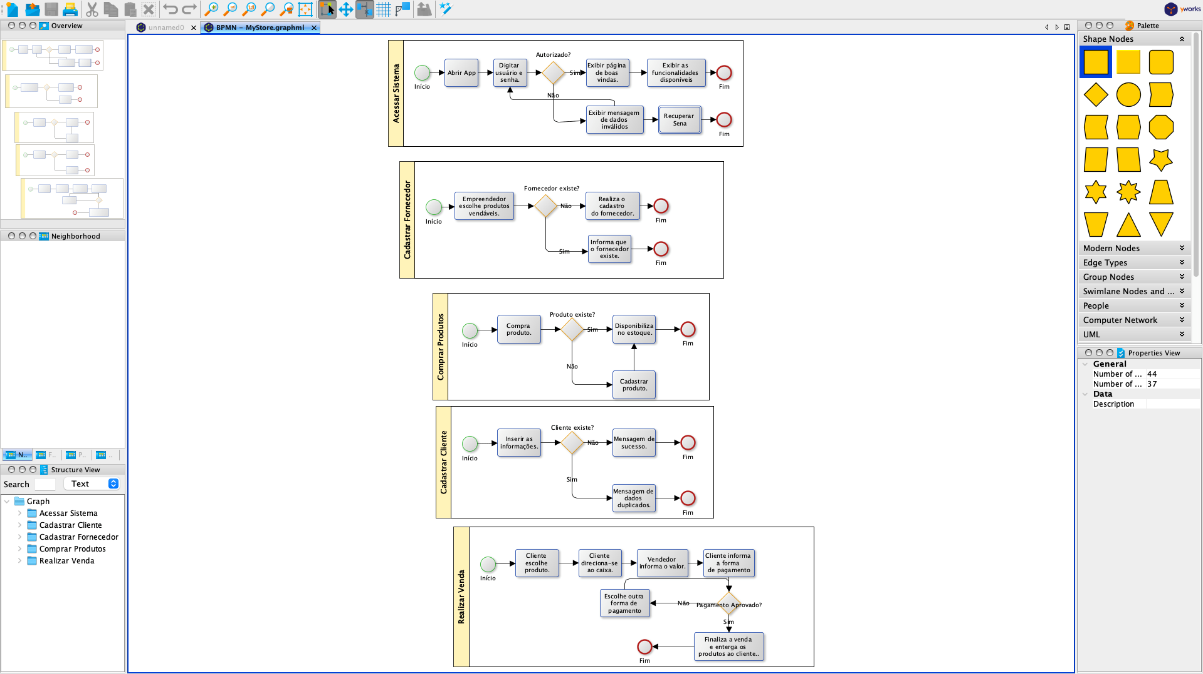


Fonte:Própria, 2021.

# yEd - Graph Editor

Os diagramas apresentados foram desenvolvidos utilizando o Software yEd - Graph Editor. Trata-se de uma plataforma que utiliza a linguagem UML 2.0 em sua notação. Tal programa foi escolhido por apresentar a possibilidade de criação de todos os diagramas necessários neste projeto, além de ser um software de fácil acesso e possuir uma versão gratuita. A figura 10 apresenta uma breve ilustração da utilização desta plataforma durante o desenvolvimento do diagrama BPMN.

Figura 10 – yEd - Graph Editor.



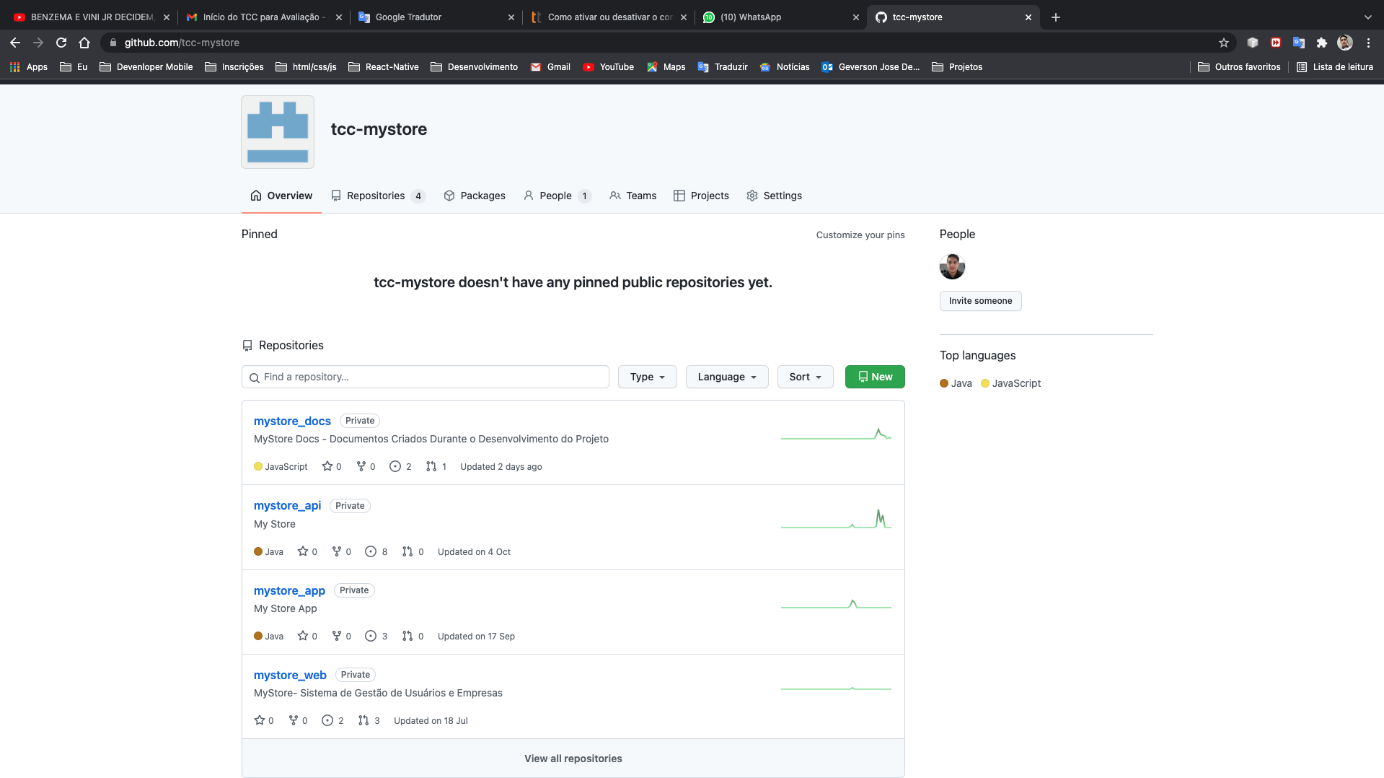
Fonte:Própria, 2021.

# Github

O GitHub é uma ferramenta muito importante que foi utilizada no desenvolvimento desta aplicação. Ele colabora e hospeda arquivos com controle de versão utilizando o Git. O sistema Git, por sua vez, controla versões distribuídas. O GitHub é utilizado principalmente para o desenvolvimento de software ao armazenar o código-fonte de um projeto e rastrear o histórico completo de todas as alterações (GURGEL, 2019).

A utilização do GitHub foi de extrema relevância, pois estamos suscetíveis a imprevistos que podem ocasionar a perda do código-fonte, ou realizar alterações indesejadas, e tendo acesso ao controle de versionamento do código, esses imprevistos podem ser amenizados. A figura 11 expõe o controle de versionamento da aplicação no Github.

Figura 11 – Github.

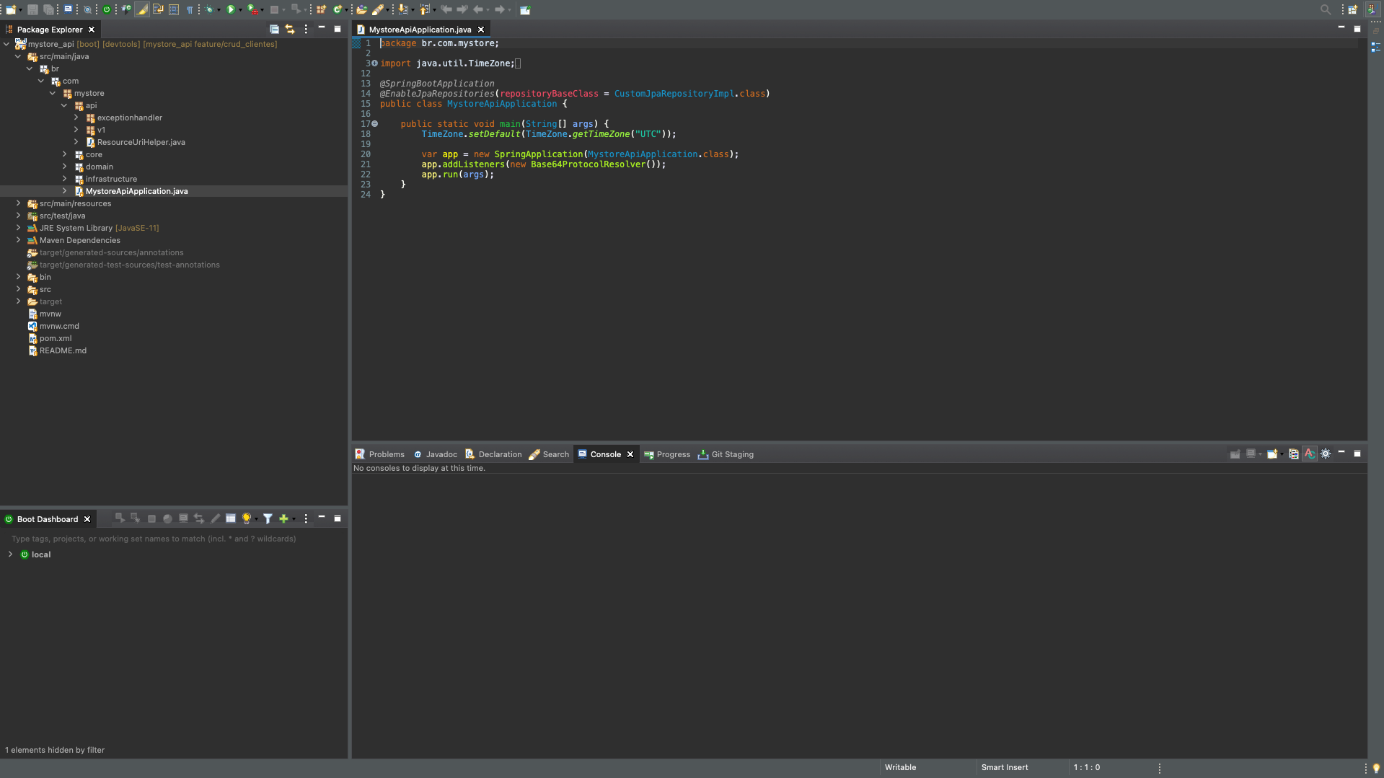


Fonte:Própria, 2021.

# Spring Tool Suite

Spring Tool Suite é um IDE para desenvolver aplicativos Spring. É um ambiente de desenvolvimento baseado em Eclipse. Ele fornece um ambiente pronto para uso para implementar, executar, implantar e depurar o aplicativo (JUNIOR e AFONSO, 2017). A figura 12 apresenta uma breve ilustração da utilização desta IDE durante o desenvolvimento da API RESTful.

Figura 12 – Spring Tool Suite.

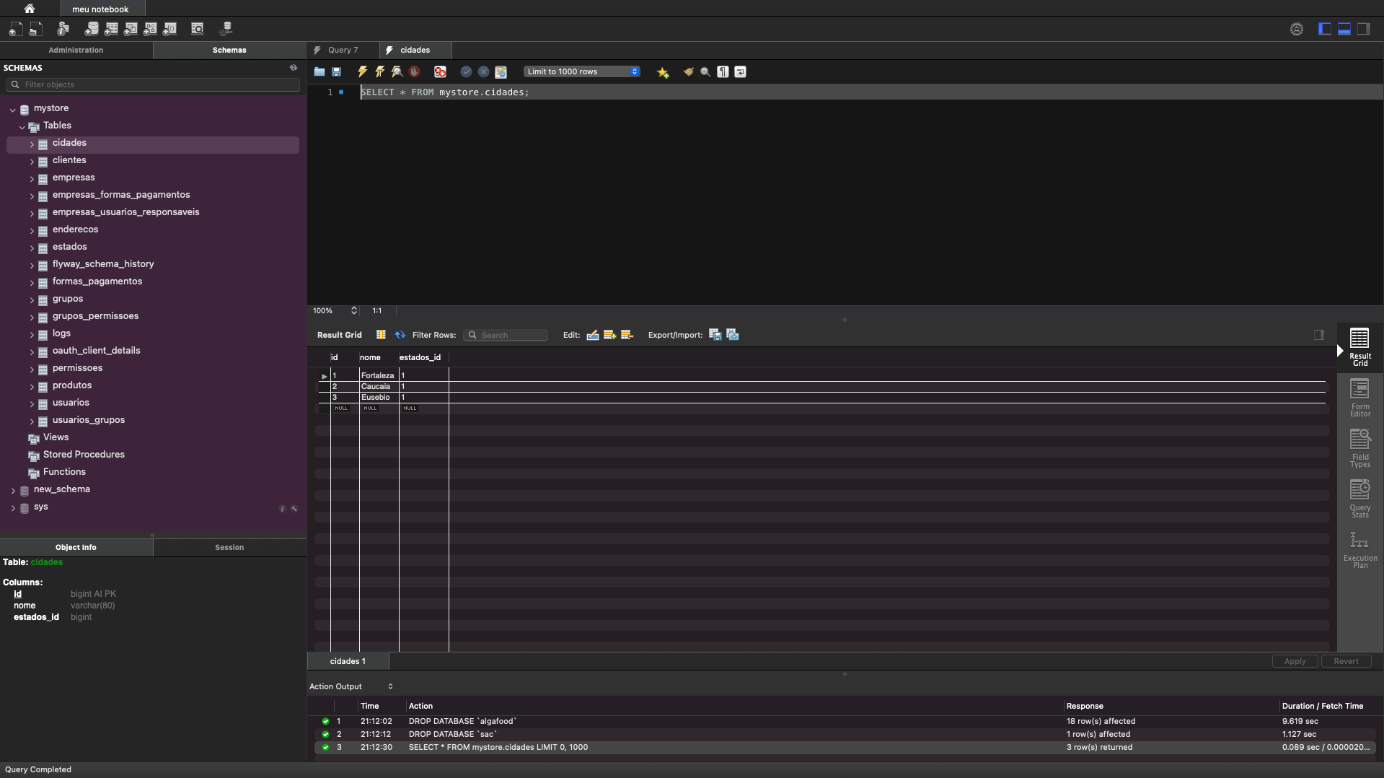


Fonte:Própria, 2021.

# MySQL Workbanch

# MySQL Workbench é uma ferramenta de design de banco de dados visual que integra desenvolvimento, administração, design de banco de dados, criação e manutenção de SQL em um único ambiente de desenvolvimento integrado para o sistema de banco de dados MySQL. É o sucessor do DBDesigner 4 da fabFORCE.net e substitui o pacote de software anterior, MySQL GUI Tools Bundle. A figura 13 apresenta ilustra a interface desta ferramenta durante a modelagem do banco de dados.

Figura 13 – MySQL Workbanch.



Fonte:Própria, 2021.

# Resultados da Execução do Projeto

Então, fundamentado em entrevistas e pesquisas bibliográficas, foi criado um aplicativo em React Native para registrar as vendas e recebimentos dos vendedores informais para lhes mostrar o real cenário do seu negócio.

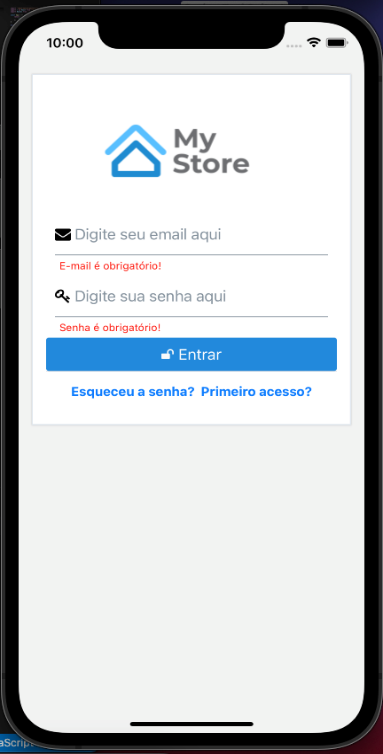
Para a fase inicial de execução do projeto foi escolhido os dados reais de um dos usuários, no qual este usuário possuía a menor quantidade de dados comparado aos demais. Os dados utilizados para a demonstração do aplicativo foram extraídos de anotações em planilhas e agendas por escrito no período de janeiro de 2020 a junho de 2021. Estes dados são os resultados das vendas e recebimentos do mesmo período.

# Apresentação do Aplicativo

# Tela de Login

A figura 14 apresenta como ficou a tela de login do aplicativo construído.

Figura 14 – Tela Login.

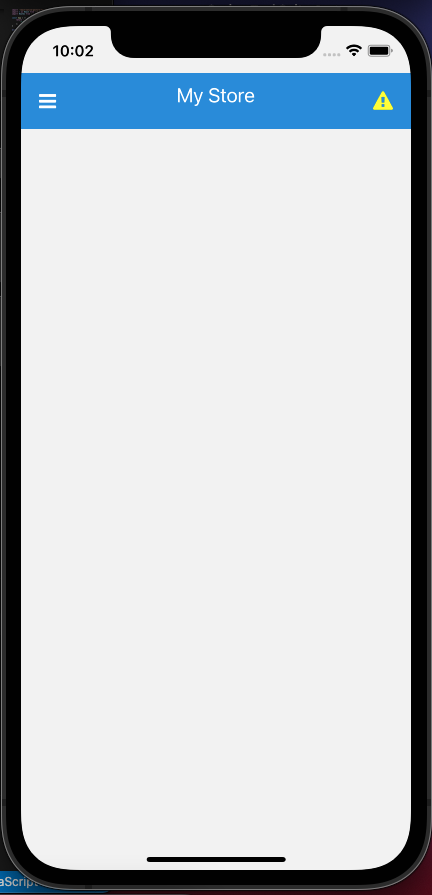


Fonte:Própria, 2021.

# Tela de Principal

A figura 14 apresenta como ficou a tela após o usuário realizar o login no aplicativo construído, esta tela contém um botão que quando acionado apresenta as funcionalidades disponíveis.

Figura 15 – Tela Principal.

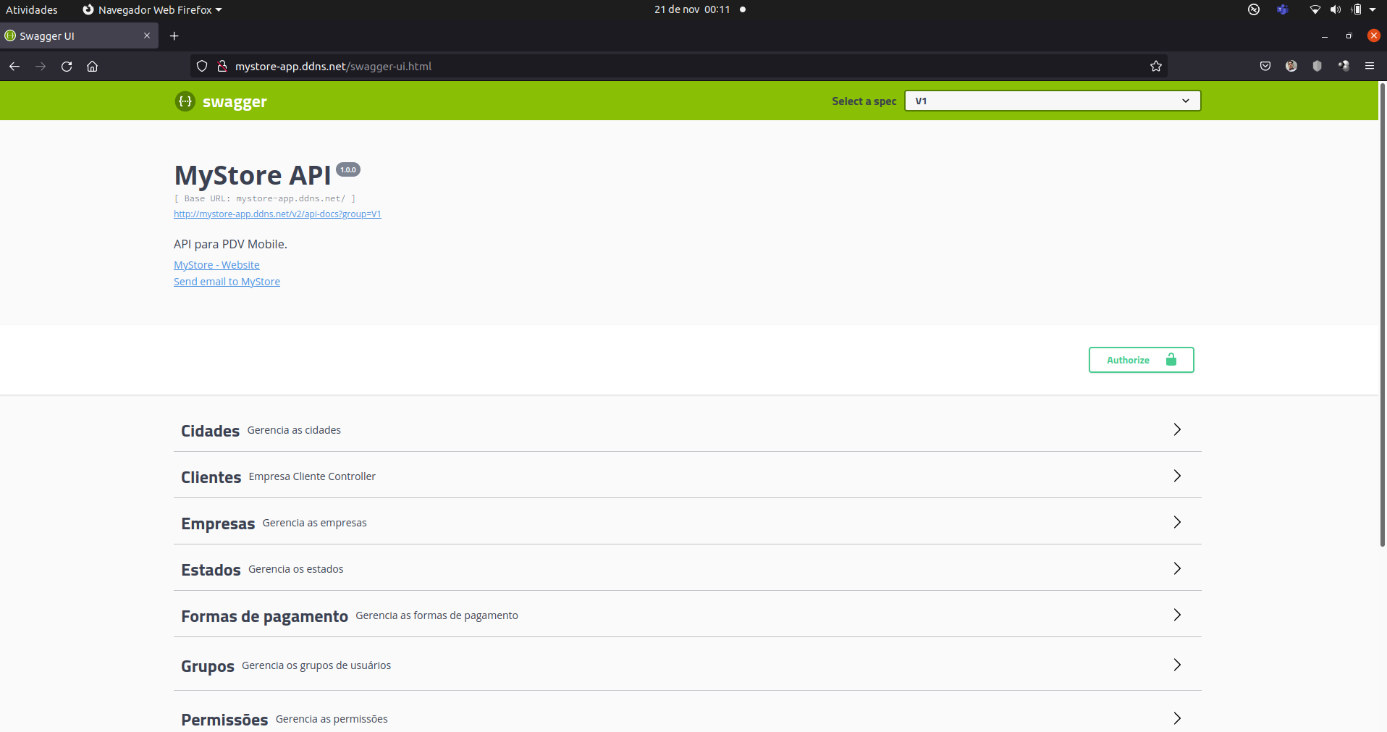


Fonte:Própria, 2021.

# OpenAPI 2.0

A especificação OpenAPI , anteriormente conhecida como especificação Swagger , é uma [especificação](https://en.wikipedia.org/wiki/Specification_(technical_standard)) para arquivos de [interface](https://en.wikipedia.org/wiki/Interface_(computing))[legíveis](https://en.wikipedia.org/wiki/Machine-readable_medium) por [máquina](https://en.wikipedia.org/wiki/Machine-readable_medium) para descrever, produzir, consumir e visualizar [serviços da](https://en.wikipedia.org/wiki/Representational_state_transfer)[web](https://en.wikipedia.org/wiki/Web_API)[RESTful](https://en.wikipedia.org/wiki/Representational_state_transfer" \o "Representational state transfer) . Anteriormente parte da estrutura [Swagger](https://en.wikipedia.org/wiki/Swagger_(software)) , tornou-se um projeto separado em 2016, supervisionado pela OpenAPI Initiative, um projeto de colaboração de código aberto da [Linux Foundation](https://en.wikipedia.org/wiki/Linux_Foundation) . Swagger e algumas outras ferramentas podem gerar código, documentação e casos de teste em um arquivo de interface. (<https://spec.openapis.org/oas/latest.html>). A figura 16 mostra a página em HTML gerada pelo Swagger da documentação da API RESTful deste projeto no ambiente de produção.

Figura 16 – Swagger.



Fonte:http://mystore-app.ddns.net/swagger-ui.html, 2021.

# Análises e Resultados

Com o objetivo de criar melhorias ou uma possível mudança no escopo do projeto seguem abaixo as avaliações dos usuários que deram início ao surgimento da ideia do projeto e estão sendo voluntários do aplicativo:

Usuário de vendas de produtos de limpeza - “​Excelente experiência com as funcionalidades do aplicativo! ”

Usuário que confecciona e vende modinha – “Muito bom, porém seria interessante integrar com o WhatsApp! ”

Usuário que vende roupas e acessórios femininos e masculino – “Vou precisar de treinamento para me adaptar com o uso da tecnologia! ”

Usuário que vende roupas e acessórios masculinos – “Vou usar quando o aplicativo estiver 100% concluído! ”

Com base na proposta deste projeto e com os comentários dos envolvidos, o resultado foi satisfatório, pois foi possível ver o empenho dos usuários a usarem e sugerirem novas ideias para que a automatização do processo do negócio aumente facilitando o dia-a-dia no trabalho.

# CONSLUSÃO

A realização do trabalho envolveu o conhecimento técnico acerca do desenvolvimento de aplicativos móveis. Para atingir os objetivos foi necessário passar pelas etapas de levantamento de requisitos e modelagem de sistemas de forma sistemática, consolidando assim os conhecimentos e habilidades adquiridos nessa área. Deste modo, pode-se afirmar que foi possível aplicar os diversos conhecimentos teóricos e tecnológicos obtidos no curso de Sistemas de Informação, ao longo das diversas disciplinas, de uma forma integrada para poder desenvolver o aplicativo proposto.

Apesar de necessitar de algumas funcionalidades extras, tais como ler o código de barras dos produtos para evitar que o usuário digite o nome ou o código dos produtos, e também uma opção integrar com o Whatsapp que poderiam ser implementadas para melhorar cada vez mais a experiência do usuário, o aplicativo conseguiu executar o objetivo para o qual foi construído com êxito.

Desta forma, espera-se que a continuação deste projeto continue pois durante o desenvolvimento foram surgindo novas ideias bastante interessante e que será avaliada para futuras versões do aplicativo podendo até ser criado uma versão web deste aplicativo.

# REFERÊNCIAS BIBLIOGRÁFICAS

EISENMAN, BONNIE. Learning React Native — Published by O'Reilly Media, Inc. 1005 Gravenstein Highway North, Sebastopol, CA, 2015.

FIELDING, ROY. Architectural Styles and the Design of Network-based Software Architectures. Dissertation (Doctorate) — UNIVERSITY OF CALIFORNIA, IRVINE. Information and Computer Science, 2000.

GALVÃO, J.; SOARES, F.; KAI, P. M. Modelo de API REST para integração de sistemas biométricos agnósticos. XIII Simpósio Brasileiro de Automação Inteligente, 2017.

GIL, A. C. Como Elaborar Projetos de Pesquisa. [S.l.]: Atlas, 2010. v. 5a ed.

GUILHOTO, L. D. O Uso da Internet como Ferramenta para a Oferta Diferenciada de Serviços a Clientes Corporativos: Um Estudo Exploratório no Setor de Telecomunicações. Dissertação (Mestrado) — Universidade de São Paulo: Faculdade de Economia, 2002.

Gonçalves, A. J. R. Desenvolvimento de aplicativos híbridos com o ionic framework. III Escola Regional de Informática do Piauí, 2017.

GURGEL, T. K. S. ​RepositORE​: Um Repositório de Objetos de Aprendizagem para Robótica Educacional. Dissertação (Graduação) – Universidade do Estado do Rio Grande do Norte, 2019.

JUNIOR, G. G. Modelo de Servidor Web com Quatro Módulos de Atendimento de Requisições (SWMAR). Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação - USP, 2008.

JUNIOR, NORMANDES e AFONSO, ALEXANDRE. Produtividade no Desenvolvimento de Aplicações Web com Spring Boot**:** AlgaWorks, Treinamentos e Serviços Ltda, 2017.

MACORATTI, J. C. ​Se conectando a um banco de dados remoto​. il. color. Disponível em: <​http://www.macoratti.net/18/05/xf\_conr bd1.htm​>. Acesso em 15 jul. 2019.

MATOS, B. R. D.; SILVA, J. G. de B. Estudo comparativo entre o desenvolvimento de aplicativos móveis utilizando plataformas nativas e multiplataforma. Dissertação (Graduação) - Universidade de Brasília - UnB, 2016.

MELO, Ana Cristina. Desenvolvendo aplicações com UML 2.2 do conceito à implementação​. 3° edição. Rio de Janeiro: Brasport, 2010.

MENDES, G. Detecção de adulterações em gasolina e previsão de parâmetros físico-químicos a partir de curvas de destilação associadas a ferramentas quimiométricas. Dissertação (Doutorado) — Universidade Federal de Minas Gerais. Departamento de Química, 2012.

MILANI, ANDRÉ. MySQL: Guia do Programador: Novatec, 2006.

MOREIRA, D. L. d. J. Panorama sobre a utilização de Dados Governamentais Abertos no Brasil: um estudo a partir dos aplicativos desenvolvidos. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, Instituto Brasileiro de Informação em Ciência e Tecnologia, 2015.

NASCIMENTO, H. A. D. do; FERREIRA, C. B. R. Uma introdução à visualização de informações. Universidade Federal de Goiás -UFG, 2011.

NICOLAI, B. B.; OLIVEIRA, D. M. de; MORAES, N.; SILVA, W. L. da. Google Android: A plataforma, seus componentes e suas versões. Dissertação (Graduação) — Faculdade Anhanguera de Santa Bárbara.

PULUCENO, T. V. Estudo de caso sobre uma API REST utilizando a abordagem de pro- gramação orientada e eventos com a plataforma Node.js. Dissertação (Graduação) — Uni- versidade Federal de Santa Catarina, 2012.

RECKER, JAN. Evaluations of Process Modeling Grammars: Springer, 2011.

ROSSI, RUI. Programação de Computadores em Java: Nova Terra, 2011. v. 2a ed.

Rubio, Ewerton Julian; Cruz, Juan Pablo Fabro Sistema Integrado de Compras Coletivas - SICC. 204 p. Trabalho de Diplomação – Universidade Tecnológica Federal do Paraná. Curso de Tecnologia em Sistemas para Internet, 2012.

SALMON, ARIANNA ZOILA OLIVEIRA Modelagem e Análise de Requisitos de Sistemas Automatizados Usando UML e Rede Petri / A. Z. O. Salmon – versão corr. – São Paulo 2017.