

Desenvolvimento de uma Plataforma Web para Catálogo de Produtos e Realização de Pedidos

Márlon da Rosa¹, Ana Paula Canal¹

¹Curso de Sistemas de Informação – Universidade Franciscana
CEP 97010-032 – Santa Maria – RS – Brasil

marlon.rosa@ufn.edu.br, apc@ufn.edu.br

Abstract. *This work presents a proposal for a web application to catalog and receive product orders. To reach the proposed objective, the Feature Driven Development methodology was adopted, aiming at the elaboration of the developed project. The application had the programming language NodeJS using the Adonis framework for the backend and VueJS for the frontend. As for data storage, MySQL was chosen. As a result, there is a web application for viewing the product catalog and making purchase orders.*

Resumo. *O presente trabalho apresenta a proposta de uma aplicação web para catalogar e receber pedidos de produtos. Para alcançar o objetivo proposto, foi adotada a metodologia Feature Driven Development, visando a elaboração do projeto desenvolvido. A aplicação contou com a linguagem de programação NodeJS utilizando o framework Adonis para o backend e VueJS para o frontend. Quanto ao armazenamento de dados foi escolhido o MySQL. Como resultado tem-se a aplicação web para visualização do catálogo de produtos e realização dos pedidos para compra.*

1. Introdução

O acesso à informação é realidade hoje, sendo utilizada em empresas, auxiliando na organização em tomada de decisões bem como para uso pessoal em tarefas como estudo e lazer. Devido ao uso da tecnologia de forma ativa, foi acumulado um grande volume de informações sobre os mais diversos assuntos.

Para Lessing (2002), saber a necessidade de cada cliente é uma tarefa árdua, dado que o cliente deseja algo que não tem conhecimento. Em muitos desses casos, são empregados termos pouco objetivos que acabam dificultando a comunicação.

Segundo Oliveira e Martiniuk (2021), com o avanço da computação, a comunidade tecnológica propôs possíveis soluções para suprir essas necessidades, já que dados podem agregar valor aos negócios. Dessa maneira, foram criadas aplicações para automatização de tarefas e, principalmente, auxiliar em tomadas de decisões. Isso pode refletir em alguns aspectos, como o aumento do lucro nas empresas.

Este trabalho tem o objetivo de criar uma plataforma de produtos para diferentes nichos de negócios, possibilitando a criação de catálogo de produtos, sua visualização e consulta pelos potenciais clientes e solicitação de produtos.

1.1. Objetivo geral

Este trabalho tem como objetivo a criação de uma plataforma onde é possível a disponibilização de produtos ao público. Esta plataforma permitirá a criação de categorias e produtos para seu negócio como também permite a visualização e pedidos pelos seus respectivos clientes.

1.2. Objetivos específicos

São objetivos específicos deste trabalho:

- Estudar linguagem de programação NodeJS, JavaScript e CSS.
- Estudar e aplicar o uso de *frameworks* no desenvolvimento do sistema.
- Aplicação do SGBD (Sistema Gerenciador de Banco de Dados) *MySQL* para o gerenciamento do banco de dados da aplicação.
- Planejar um software com sua respectiva documentação e processos, aplicando a metodologia FDD (*Feature Driven Development*).

A Seção 2 apresenta o referencial teórico e as tecnologias que têm participação neste trabalho. Já na Seção 3 estão descritos os trabalhos correlatos pertinentes ao tema apresentado que são de suma importância para a compreensão deste projeto. A Seção 4 abordará conceitos a respeito da metodologia de desenvolvimento. A Seção 5 apresentará as considerações finais.

2. Referencial Teórico

Nesta seção serão apresentados os assuntos necessários ao entendimento e construção deste trabalho. Serão tratados assuntos como plataformas web e tecnologias a serem utilizadas para o desenvolvimento web.

2.1. Plataforma Web

A chegada do Covid-19 trouxe um grande impacto para economia e como também o isolamento social e refletiu nas vendas e faturamento das empresas. Com isso, os consumidores que são os principais indivíduos para movimentação deste tipo de comércio econômico viram-se obrigados a deixar o consumo alternativo de lado por questões econômicas, pela legislação imposta pelo governo e principalmente para prevenir o risco de contágio próprio e de seus familiares.

O comércio também obrigou-se a criar maneiras alternativas para poder dar continuidade a suas atividades para continuar gerando renda para seus negócios visto que o modo tradicional diminuiria consideravelmente sua economia [Rezende e Marcelino e Miyaji 2020].

Outro ponto positivo para o mercado eletrônico são suas despesas com local físico, a necessidade de um maior número de funcionários, entre outros. As plataformas *web* também oferecem um maior alcance de clientes, o qual é um dos pontos negativos oferecidos pelas lojas físicas. Contudo é altamente indicado que mesmo após a pandemia estar erradicada ou ao menos chegar em um nível aceitável como outras doenças de menor periculosidade o comércio opte também por continuar seus negócios *online*.

Este trabalho visou estudar, aplicar e aliar os problemas do distanciamento social juntamente com a tecnologia. Pensando nisso, foram pesquisados temas relacionados ao *e-commerce* que já exploravam o uso da *Internet* para venda de produtos.

2.2. Tecnologias

Nesta seção serão apresentadas as tecnologias necessárias para o desenvolvimento do objetivo proposto: NodeJS, *frameworks* Adonis, VueJS e o sistema gerenciador de banco de dados MySQL. As tecnologias escolhidas possuem uma integração, além de contarem com uma vasta comunidade e documentação.

O NodeJS é uma linguagem de programação assíncrona e orientada a eventos criada para ser utilizada no lado do servidor. Devido ser uma linguagem assíncrona não existem *threads* pois não ocorre bloqueio como nas demais linguagens. Conta com uma grande quantidade de recursos e *frameworks* [Adonis 2021].

O NodeJS foi escolhido devido a sua versatilidade para trabalhar com web sendo bem utilizada no mercado devido sua facilidade em reaproveitamento de código além de poder concentrar seu conhecimento em uma só linguagem. Ainda conta com *framework* Adonis para o *back end*, o qual possui funcionamento muito semelhante ao Laravel *framework* usado com a linguagem PHP.

O NodeJS conta com um *framework* chamado Express, o qual é responsável por criar as rotas que permite a criação de rotas entre outras funcionalidades de forma simples e rápida.

O Adonis é um *framework* com uma série de funcionalidades já implementadas, que tem como base outros frameworks como Laravel, Rails e Django. Ao contrário do ExpressJS que trabalha apenas com rotas, o Adonis conta com uma lista de funcionalidades como *migrations*, *seeders*, ORM (*Object Relational Mapping*), autenticação, validação entre outros [Adonis 2021].

O Vue JS é um *framework* JavaScript, de *open source*, responsável pela criação de *interfaces*. Seu desenvolvimento se deu em 2014 onde seu desenvolvedor trouxe o que gostou do AngularJS com a ideia de ser algo leve (CROMWELL, 2016). Ele permite uma migração gradual e de fácil compatibilidade com outras tecnologias já existentes no projeto [Mendes 2018]. Está tecnologia foi escolhida devido sua possibilidade de processamento de dados no lado do cliente possibilitando minimizar o processamento do servidor. Também abrange o uso de muitas ferramentas de *front-end* com pouca ou nenhuma configuração, permite a aplicação de testes unitários, roteamento (VueJS 2021).

O MySQL é um SGBD (Sistema Gerenciador de Banco de Dados) relacional de que possui licença GPL (General Public License) para uso comercial e não-comercial. A versão comercial é voltada para projetos comerciais, que conta com suporte algo ausente na versão gratuita não possui. Criado por David Axmark, Allan Larsson e Michael Monty, nos anos 90, devido a uma necessidade específica, dado que o que havia na época não satisfazia suas necessidades. Inicialmente, surgiu com ideia de funcionar em projetos de pequeno e médio porte, já em sua versão mais atual, possui suporte para grandes aplicações, empresas renomadas no mercado da tecnologia utilizam esse SGBD [Oracle 2018].

3. Trabalhos Correlatos

Nesta seção são apresentados os trabalhos relacionados ao objetivo proposto acima, os quais contemplam características semelhantes a este trabalho, cooperando com uma base de conhecimento efetivo elaborando e organizando este projeto.

3.1 SIGAT (Sistema de gestão e agendamento de TCCs)

A proposta deste projeto é o desenvolvimento de uma aplicação para agendamento de TCC's, onde alunos podem realizar o agendamento de sua apresentação. A ideia ocorreu após uma análise sobre como era realizado o processo, no campus da UFC de Quixadá. O objetivo era uma ferramenta de automatização e organização [Viana 2017].

Para tal projeto foram utilizadas as seguintes tecnologias: Spring Boot, MySQL, HTML, CSS, JavaScript e Materialize.

A proposta se relaciona com o trabalho em questão, pelo conceito de utilizar tecnologias web para desenvolver uma solução para agilizar um processo que ainda é feito de forma manual. Desta forma o foco deste trabalho relacionado é de facilitar o agendamento dos TCCs.

3.2. Sistema de recomendação mobile: um estudo de caso para *delivery*

Garbin e Silveira (2017) propõem o desenvolvimento de uma aplicação com sistema de recomendação para *delivery*. Essa aplicação permite ao usuário realizar o pedido, acompanhar o status e consultar seu histórico de pedidos e recomendações para novas compras.

Para o projeto, foi escolhido o desenvolvimento de um protótipo a fim de validar a aplicação. O teste foi realizado na pizzaria Fratello, em Rodeio Bonito/RS. O acompanhamento foi realizado pelo gerente.

O primeiro teste foi realizado localmente apenas, então, após a aprovação, foi requisitado um domínio para a empresa e serviço de hospedagem na web. Já para o aplicativo, foi criado um cadastro no Google Play para sua publicação. O autor conclui que os objetivos iniciais obtiveram sucesso em sua prova final.

3.3. Considerações de Trabalhos Correlatos

Os trabalhos citados acima possuem relação com este devido ao planejamento, desenvolvimento e testes de uma plataforma que disponibiliza sistemas de vendas e agendamentos aos seus usuários. Os trabalhos acima foram usados como base e para fins de comparação de ideias. O primeiro caso apresenta a parte de agendamentos para TCC's, enquanto no segundo foi desenvolvido um aplicativo para Android já o presente trabalho apresentou a proposta de um sistema *web* para disponibilização e pedidos de produtos sem a necessidade de nenhum tipo de instalação.

4. Feature Driven Development (FDD)

O FDD reúne as melhores práticas de outras metodologias, permitindo um equilíbrio entre as filosofias tradicionais, propondo, desse modo, uma entrega ágil, com menor chance de falhas, já que o cliente acompanha todo o projeto durante seu desenvolvimento. O FDD foi escolhido para este trabalho devido a sua fácil adaptação a pequenos projetos [FDD 2018].

O FDD torna-se interessante devido às características apresentadas como por exemplo seu *sprint* a cada duas semanas ou menos, pequenos blocos de funcionalidades que permitem um acompanhamento do que está sendo realizado, planejamento bem detalhado, rastreabilidade, monitoramento através de resumos para clientes e gerentes, tudo em termos de negócio e permite uma verificação do projeto, estimando se o planejamento é sólido ou não. As seguintes etapas foram desenvolvidas:

- Desenvolver um modelo abrangente: fase responsável pelo escopo do projeto.
- Listar funcionalidades: apresenta uma lista de requisitos funcionais e não funcionais.
- Planejar por funcionalidades: nesta fase são listadas e ordenadas as funcionalidades de acordo com as prioridades do projeto.
- Detalhar por funcionalidade: conta com o diagrama de atividades o qual é responsável por apresentar o fluxo do sistema.
- Construir por funcionalidade: fase responsável pela implementação do sistema.

4.1. Desenvolver um modelo abrangente

Nesta fase foi realizado um estudo detalhado sobre o domínio e escopo do projeto, junto com uma lista de requisitos a serem desenvolvidos e, então, tem-se a visualização do projeto de forma abrangente [FDD 2018].

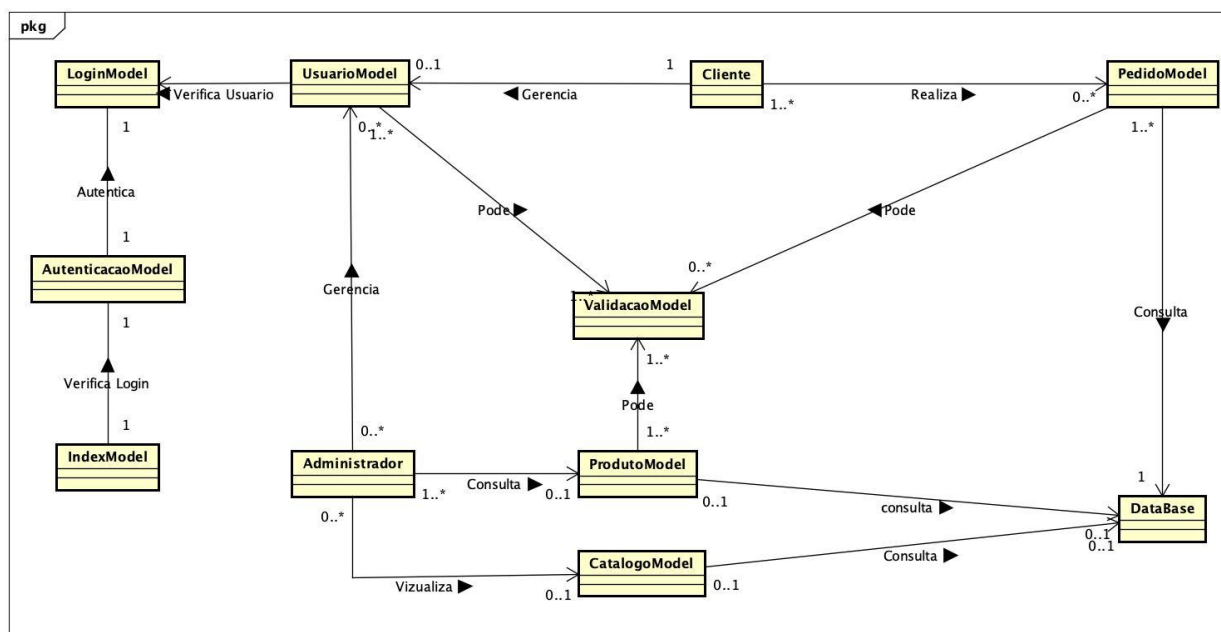


Figura 1. Diagrama de domínio

O Diagrama de Domínio, representado pela Figura 1, apresenta as entidades de domínio do sistema a ser desenvolvido. Como exemplo, há os produtos cadastrados pelo administrador do sistema do sistema, onde realizará a ação de adicionar ou remover produtos e manipulação de suas regras de negócio. Já os clientes podem visualizar e realizar pedidos de produtos.

4.2. Listar funcionalidades

Neste processo foi dado início à construção da lista de funcionalidades baseada na documentação anterior. A parte mais desafiadora é a descoberta dos requisitos, pois isso define a qualidade e manutenção do produto após ser finalizado [FDD 2018]. A seguir, nos Quadros 1 e 2, respectivamente, estão listados os requisitos funcionais e não funcionais que são necessários para execução do projeto.

Quadro 1. Lista de requisitos funcionais

| RF01 - Gerenciar usuário | |
|----------------------------------|---|
| RF1.1 | O sistema deve permitir ao administrador alterar os dados dos usuários. |
| RF1.2 | O sistema deve permitir que o administrador seja capaz de cadastrar, alterar, excluir, consultar produtos. |
| RF1.3 | O sistema deve permitir que o administrador seja capaz de gerenciar o catálogo. |
| RF1.4 | O sistema deve permitir que o administrador possa cadastrar outros usuários como administrador. |
| RF1.5 | O sistema deve permitir ao cliente que crie seu cadastro, preenchendo formulário ou usando sua conta do Gmail ou Facebook. |
| RF1.6 | O sistema deve permitir ao cliente realizar pedidos de um prato e/ou bebidas. |
| RF02 - Validar Formulário | |
| RF2.1 | O sistema deve verificar se o usuário está logado antes de realizar qualquer ação. |
| RF2.2 | O sistema deve validar os campos “Nome”, “E-mail”, “Senha”, “Confirmar Senha” do formulário Cadastro. |
| RF2.3 | O sistema deve validar os campos “E-mail”, “Senha”, “Confirmar Senha” do formulário Editar Dados. |
| RF2.4 | O sistema deve validar os campos “E-mail”, “Senha” do formulário Login. |
| RF2.5 | O sistema deve validar os campos “Senha” do formulário Recuperar Senha. |
| RF2.6 | O sistema deve validar os campos “Nome do Produto”, “Descrição” do formulário Cadastrar Produto. |
| RF2.7 | O sistema deve validar os campos, “Valor” do Formulário de Catálogo. |
| RF03 - Recuperar Senha | |
| RF3.1 | O sistema deve fornecer um campo para recuperação de senha. |
| RF3.2 | O sistema deve enviar um e-mail com o link para recuperar senha. |
| RF04 - Criar Catálogo | |
| RF4.1 | O sistema deve consultar os produtos que estão cadastrados no sistema e, a partir disso, permitir ativar para ficar visível |

| | |
|---------------------------------|---|
| | aos clientes ou desativar caso deseje que não esteja mais disponível no catálogo. |
| RF4.2 | O sistema deve permitir alterar, atribuir ou alterar o valor de um produto. |
| RF4.3 | O sistema deve validar os campos “Mensagem” do formulário Contato. |
| RF05 - Criar Produto | |
| RF5.1 | O sistema deve permitir alterar, atribuir ou alterar o valor de um produto. |
| RF5.2 | O sistema deve validar os campos do formulário Produto. |
| RF06 - Criar Pedido | |
| RF6.1 | O sistema deve permitir ao cliente realizar um pedido do que estiver disponível no catálogo |
| RF07 - Visualizar Pedido | |
| RF7.1 | O Sistema deve permitir a visualização dos pedidos ao responsável por repassar os pedidos. |

A seguir, é apresentado o Quadro 2, onde constam os requisitos não funcionais. Requisitos não funcionais descrevem os tipos de plataforma em que a aplicação irá executar, descrição da parte técnica como linguagem de programação, banco de dados e *frameworks*.

Quadro 2. Lista de requisitos não funcionais

| Requisito Não Funcional | Descrição |
|--------------------------------|---|
| RNF1 | A aplicação será desenvolvida para plataforma web. |
| RNF2 | Será utilizada NodeJS no <i>back-end</i> em conjunto com o <i>framework</i> Adonis. |
| RNF3 | Será utilizado VueJS para o <i>front-end</i> . |
| RNF4 | Será feito o uso do SGBD MySQL para o armazenamento e gerenciamento dos dados da aplicação. |

4.3. Planejar por funcionalidades

Nesta fase foi realizado o planejamento por funcionalidades, ou seja, são ordenadas as prioridades das funcionalidades listadas na etapa anterior (Quadro 3). Isso pode variar dependendo do prazo, dependências e, ainda, apresentando o tempo de desenvolvimento de cada funcionalidade [FDD 2018].

Quadro 3. Planejamento por funcionalidades

| Requisito funcional | Tempo | Complexidade | Relevância |
|----------------------------|--------------|---------------------|-------------------|
| RF01: Gerenciar Usuário | 13 Dias. | Média | Alta |
| RF02: Validar Formulário | 11 Dias. | Baixa | Média |
| RF03: Recuperar Senha | 1 Dias. | Média | Média |
| RF04: Criar Catálogo | 15 Dias. | Média | Média |
| RF05: Criar Produto | 15 Dias. | Média | Alta |
| RF06: Criar Pedido | 15 Dias. | Média | Alta |

| | | | |
|-------------------------|----------|-------|-------|
| RF07: Visualizar Pedido | 10 Dias. | Baixa | Média |
|-------------------------|----------|-------|-------|

A seguir, na Figura 2, é apresentado o Diagrama de Casos de Uso, onde estão as principais funcionalidades do sistema. Também é mostrada a interação dos usuários (atores) com o sistema.

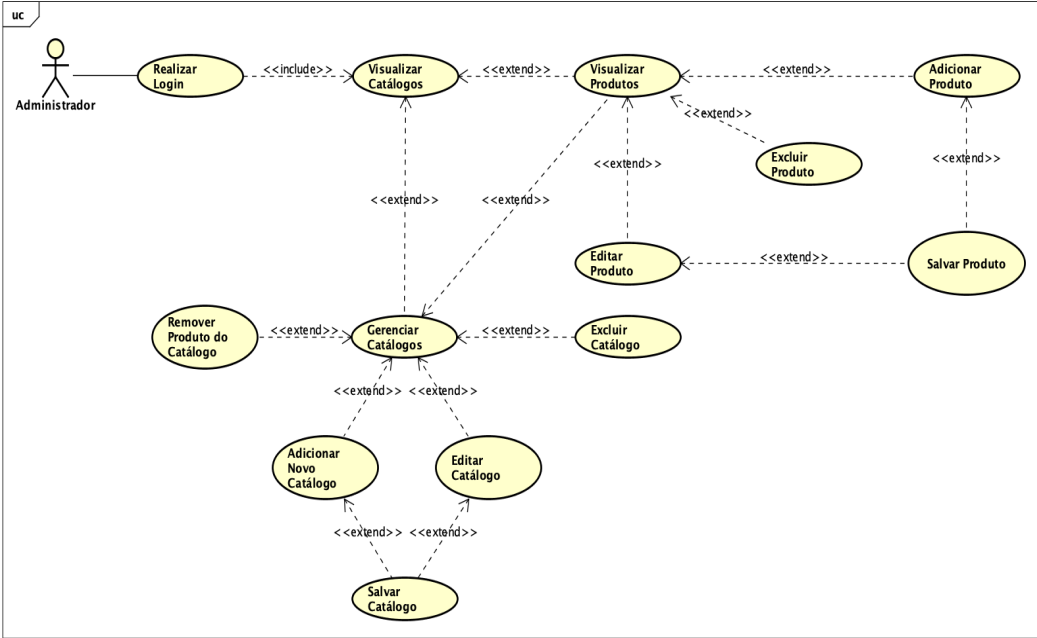


Figura 2. Diagrama de casos de uso Administrador

Nesta fase são apresentados os principais detalhes que irão compor as principais funcionalidades do sistema, mostrando a interação destas com os usuário e não se importando com os detalhes técnicos que explicam como o sistema realiza o processo [FDD 2018].

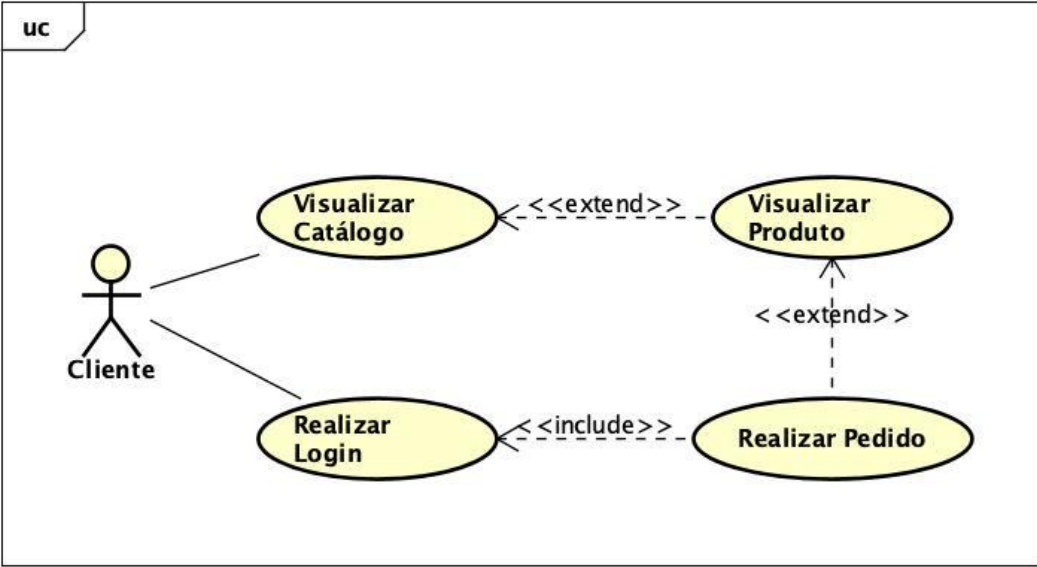


Figura 3. Diagrama de casos de uso Cliente

O diagrama seguinte, presente na Figura 3, descreve o procedimento que deve ser realizado pelo cliente do sistema para visualizar e realizar pedidos de produtos.

4.4. Detalhar por funcionalidade

Esta fase possibilita o entendimento do funcionamento do sistema, responsável por mostrar a possibilidades que podem ser realizadas [FDD 2018].

O diagrama de atividades mostra o fluxo do software. Iniciando pela tela de *login*, realiza a validação dos dados, então, se a consulta for positiva, será logado no sistema e redirecionado para página *home*. Caso contrário, ele mostrará uma mensagem de erro.

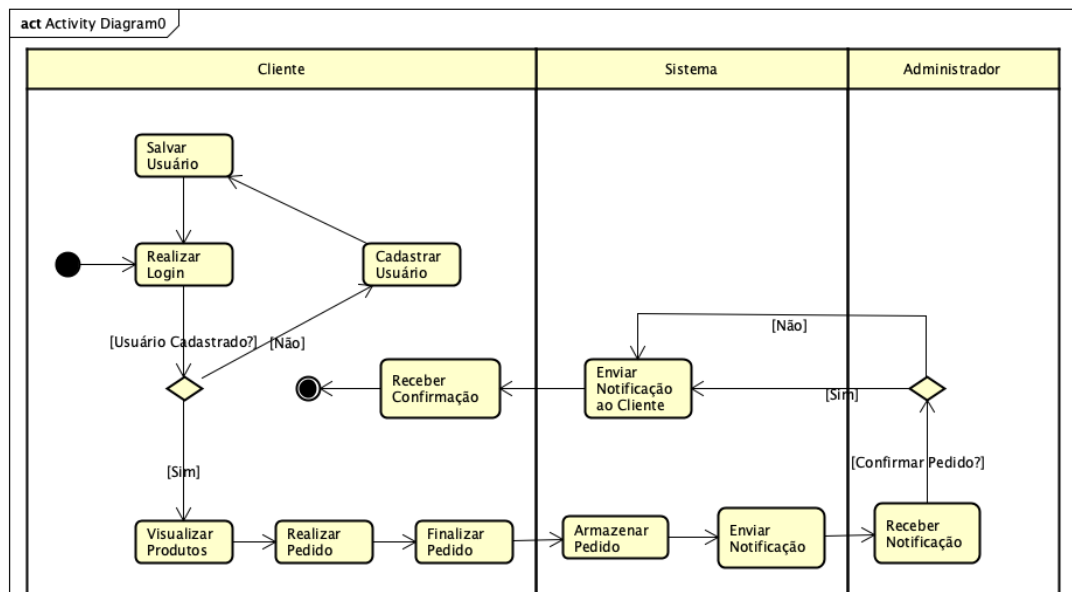


Figura 4. Diagrama de atividades – Cliente Pedido

Na Figura 4, o diagrama de atividades apresenta o fluxo de execução do sistema que se inicia quando um pedido é realizado até sua confirmação.

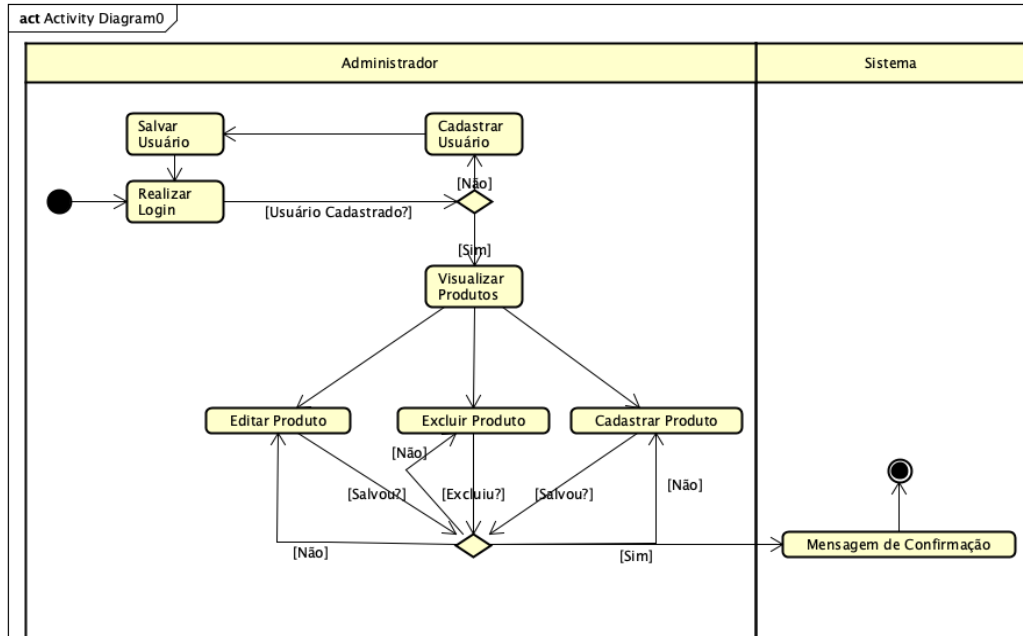


Figura 5. Diagrama de atividades – Administrador Cadastro, Edição e Exclusão de Produto

O Diagrama de Classes, representado pela Figura 6, apresenta todas as classes que foram utilizadas no sistema com seus respectivos métodos e atributos. Para exemplo, será utilizada a ação de pedido. Um pedido é realizado pelo cliente e este acessa a classe “Ordem” que é a classe responsável pelos pedidos unindo-se com que possui uma relação com a classe “Product”, que tem as informações sobre os produtos. Para realizar o relacionamento criou-se métodos usando o Lucid ORM (Object Relational Mapping) do Adonis *Framework* este oferece uma forma muito mais simplificada de relacionar as classes e qualquer operação relacionada a banco de dados.

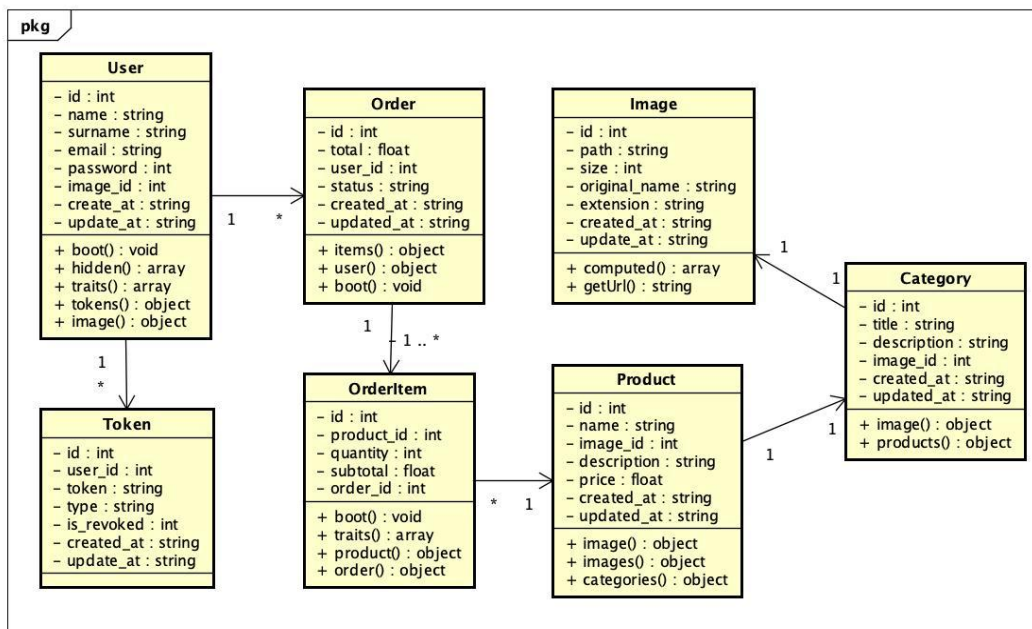


Figura 6. Diagrama de classes

Representando a estrutura do Banco de Dados deste projeto, o Diagrama Entidade e Relacionamento (DER), mostrado na Figura 6, faz com que seja possível visualizar as entidades existentes na estrutura do banco de dados do software implementado.

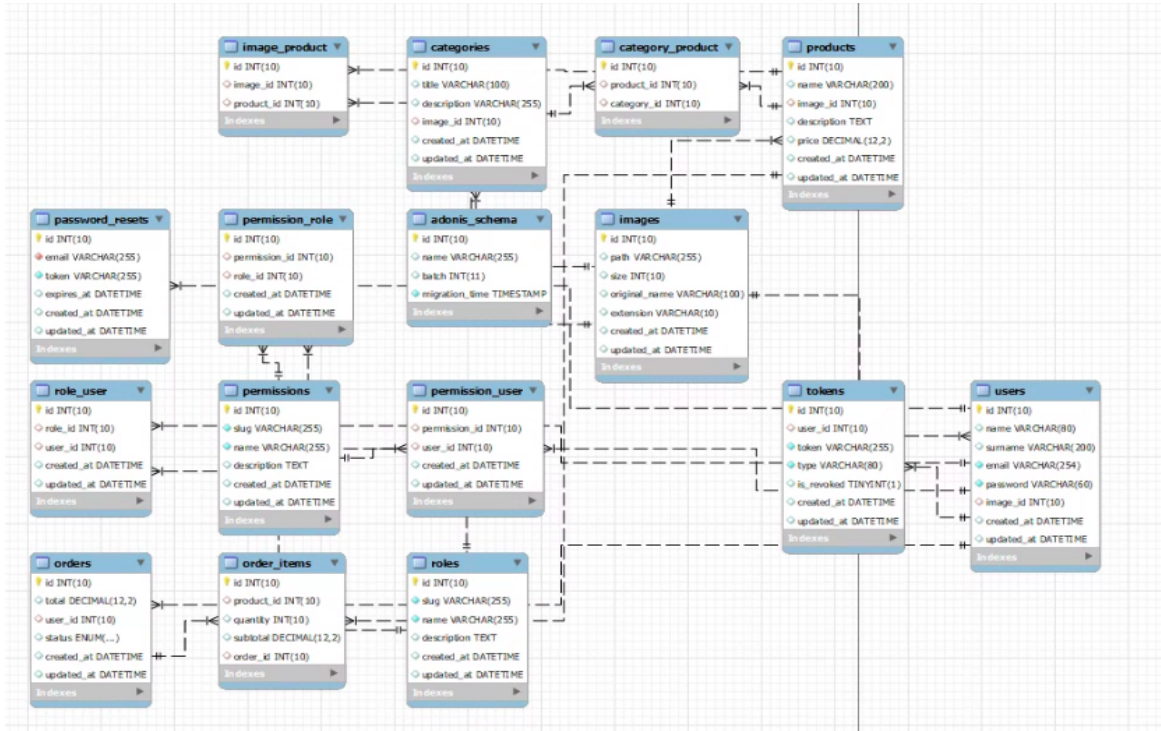


Figura 7. Diagrama entidade relacionamento (DER)

O diagrama DER (Diagrama entidade relacionamento) é responsável por apresentar a modelagem do banco de dados organização das entidades, seus atributos, tipos, chaves e relacionamentos.

4.5 Construir por Funcionalidade

A proposta principal deste trabalho é a criação de uma plataforma web para auxiliar o comércio na simplificação de rotinas e uma maior exposição de produtos, transparência e confiabilidade para seus clientes. O sistema não possui uma aplicação *mobile* até o presente momento, entretanto sua *interface* é totalmente responsiva.

O projeto foi dividido de duas formas: o *back-end* e o *front-end*. No *back end* foi desenvolvida uma API (*Application Programming Interface*) utilizando o *framework* Adonis para que mais tarde pudesse ser reaproveitada para projetos futuros como na criação de uma aplicação *mobile*. O *front-end* foi desenvolvido com o *framework* VueJS utilizando *template* Vuemmerce - Ecommerce Template o qual é responsável por consumir a API e criar as *interfaces*.

Para armazenamento de dados foi escolhido o banco de dados MySQL devido sua grande popularidade e alta compatibilidade com sistemas *web* e *stack* escolhidos.

```

methods: {
  addToCart (id) {
    this.$store.commit('addToCart', id);
    this.$store.commit('setAddedBtn', {id: id, status: true});
  },
  removeFromCart (id) {
    this.$store.commit('removeFromCart', id);
    this.$store.commit('setAddedBtn', {id: id, status: true});
  },
  onSelectQuantity (id) {
    this.$store.commit('quantity', {id: id, quantity: this.selected});
  }
}
}

```

Figura 8. Métodos da Tela Produto

O código acima apresenta uma parte correspondente a *view* Products utilizando VueJS que é escrito utilizando JS (JavaScript). O objeto *methods* recebe todos os métodos responsáveis por executar alguma função dentro da *interface* e são eles: *addToCart* (Adicionar ao Carrinho), *removeFromCart* (Remover do Carrinho) e *onSelectQuantity* (Selecionar Quantidade) passando seus respectivos parâmetros como *id* (identificador), *status* (status) e *quantity* (quantidade).

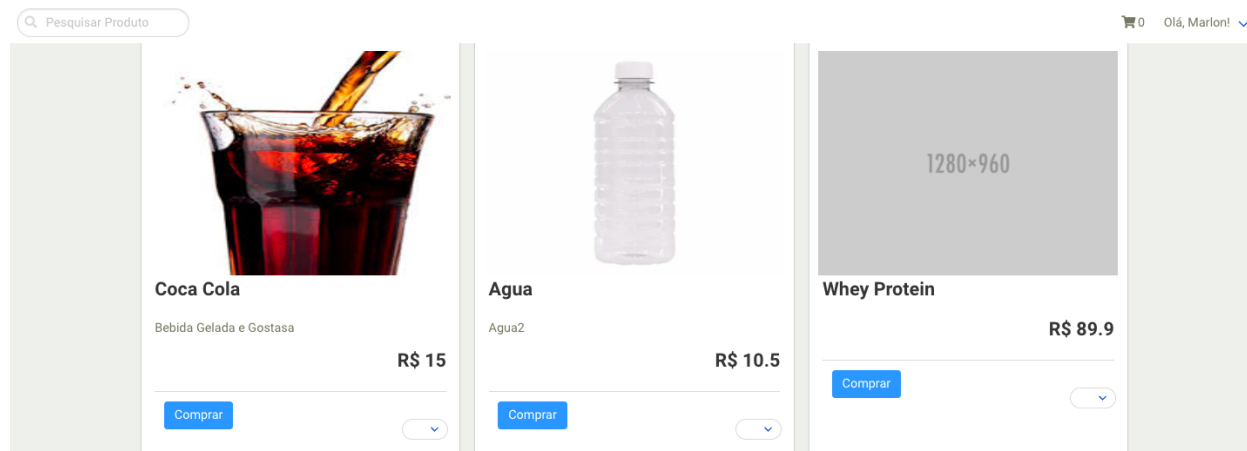


Figura 9. Lista da Interface Produto Cliente

A *interface* acima pode ser visualizada pelo cliente estando ou não logado. Caso não esteja logado será necessário realizar o *login* para realizar o pedido. A imagem que apresenta uma imagem escura é um exemplo de imagem padrão caso não tenha escolhido uma imagem para o produto.

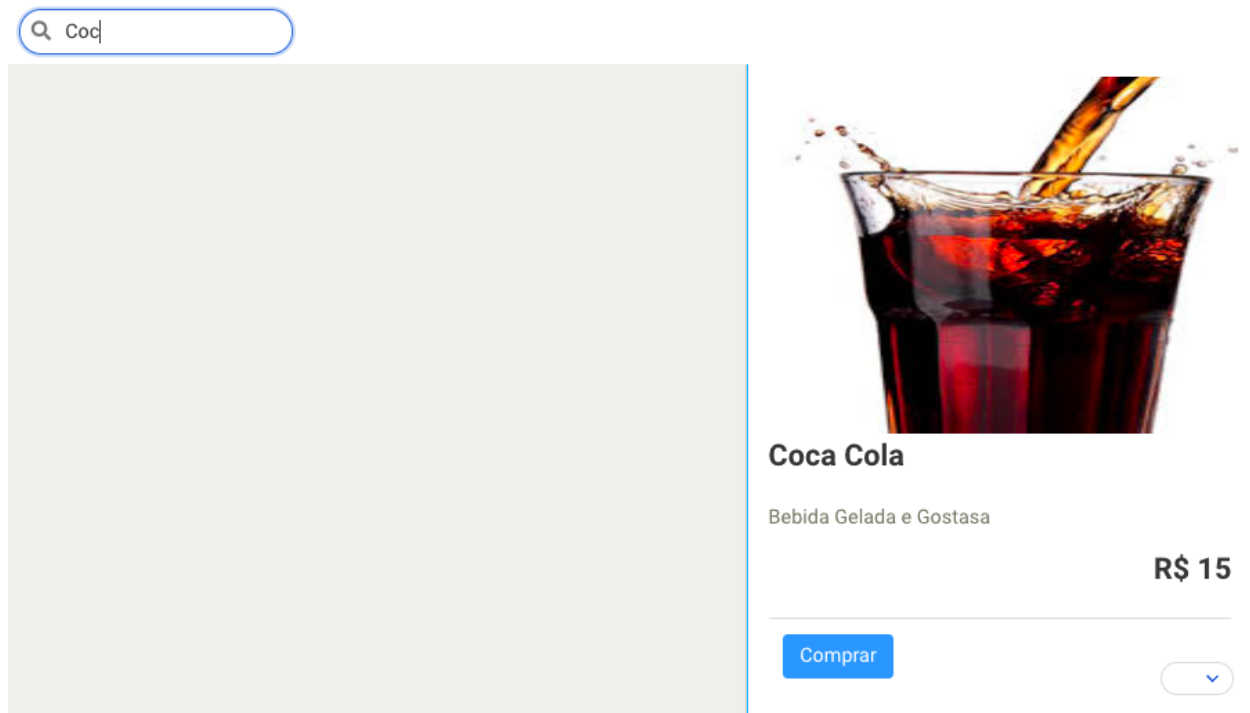


Figura 10. Administrador *Interface* com Caixa de pesquisa na *interface* de listagem

A Figura 10 apresenta a *interface* da lista de produtos após utilizar o campo de filtro. O filtro mostra os produtos de acordo com a descrição do campo de busca em tempo real sem a necessidade de recarregar a página para obter o resultado.



Figura 11. Cliente *Interface* Adicionar Produto ao Carrinho

Na Figura 11 temos a *interface* do produto escolhido caso exista uma descrição e valor. Também conta com um botão para alterar a quantidade de itens ou remover o produto do carrinho caso seja adicionado de forma errônea.

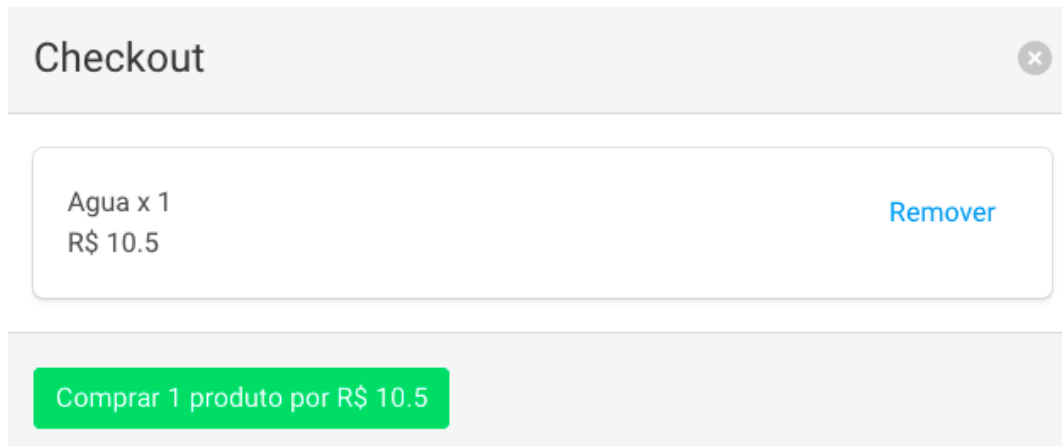





Figura 12. Cliente *Interface* Visualizar Pedidos do Carrinho

A *interface* da Figura 12 apresenta o carrinho do sistema, este é o último passo antes de confirmar o pedido. Este conta com uma breve descrição e valor. Também conta com três botões, um para fechar a janela do carrinho e outro para remover os produtos caso acidentalmente tenha errado na quantidade pedida e por último o botão de confirmação do pedido.

The image shows a product list interface for an administrator. It features a table with the following data:

| Cód. | Nome | Preço | Imagem | |
|------|--------------|-------|---|------------------------|
| 57 | Coca Cola | 15 |  | Editar |
| 60 | Agua | 10.5 |  | Editar |
| 61 | Whey Protein | 89.9 |  | Editar |

At the top left, there is a blue button labeled 'Novo Produto'. At the top right, there is a search box labeled 'Filtrar'. At the bottom, there is a pagination control showing 'Total 3', a dropdown for '10/page', and a 'Go to' field with the value '1'.

Figura 13. Lista da *Interface* Produto Administrador

A Figura 13 apresenta a tela de produtos na visão do administrador. A *interface* dispõe de botão para cadastro, edição, paginação. Os botões de paginação são anteriores, próximo e um *dropdown* para escolher a quantidade de registros por página. Também há um campo para escolher a página que deseja visualizar.

| Cód. | Cliente | Status | Total | Data | |
|------|----------------|-----------|-------|---------------------|--------|
| 1 | Marlon da Rosa | finished | 20 | 2021-11-07 21:55:48 | Editar |
| 2 | Marlon da Rosa | cancelled | 12.5 | 2021-11-07 21:56:27 | Editar |
| 3 | Marlon da Rosa | | 10 | 2021-11-07 21:59:09 | Editar |
| 4 | Marlon da Rosa | finished | 30 | 2021-11-09 19:40:48 | Editar |
| 5 | Marlon da Rosa | | 10.5 | 2021-11-23 23:16:54 | Editar |

Figura 14. Administrador *Interface Pedidos*

A Figura 14 apresenta a lista de pedidos onde encontra-se disponível informações como código, cliente, status, valor e data do pedido.

| ID | Imagem | Nome | Preço | Quantidade | subtotal |
|----|--------|------|-----------|------------|-----------|
| 60 | | Agua | R\$ 10.50 | 1 | R\$ 10.50 |

Informações do Pedido
 Cliente: Marlon da Rosa
 Status: Enviado
 Subtotal: R\$ 10.50
 Descontos: R\$ 0
 Total: R\$ 10.50
 Excluir Salvar

Figura 15. Administrador *Interface Editar Pedido*

A Figura 15 apresenta a *interface* de edição de pedidos onde é possível alterar o status do pedido como para *finished* (Finalizado) por exemplo, além de outros dados em caso de erro no pedido. Também é possível cadastrar um pedido pelo lado do cliente caso haja necessidade.

As demais *interfaces* de administrador devido a uma maior quantidade encontram-se disponíveis no apêndice deste artigo.

5. Conclusões e Trabalhos Futuros

Este estudo levou à realização de pesquisas relacionadas ao tema e apresentou a proposta de desenvolvimento de uma plataforma *web* para ligação entre comerciante e cliente. No lado do

comerciante é possível disponibilizar seus produtos enquanto o cliente pode realizar pedido dos mesmos. A metodologia escolhida para o projeto foi a FDD, devido a sua capacidade em amparar pequenos projetos e seu processo ágil. Quanto aos trabalhos correlatos, buscou-se a identificação de propostas e metodologias semelhantes, para ter como base os quais apresentaram alguns pontos de reaproveitamento de ideias.


Contudo foi concluído que este projeto obteve sucesso já que sua implementação foi bem sucedida. Com o software desenvolvido, espera-se o colocar em teste o software para validar suas funcionalidades e verificar a satisfação por parte de seus usuários. Funcionalidades poderão ser serão adicionadas como por exemplo a geração de relatórios, aplicação *mobile*, meio de pagamento dentro da plataforma, entre outras conforme surgirem.

Referências

- Adonis. (2021) “AdonisJS at a glance”, <https://adonisjs.com/adonisjs-at-a-glance>, Novembro.
- Adonis. (2021) “AdonisJS at a glance”, <https://docs.adonisjs.com/guides/introduction>, Novembro.
- FDD. (2018) “Feature Driven Development”, <http://www.featuredrivendevelopment.com/>, Setembro.
- Garbin, .; Silveira, . (2017) “Sistema de Recomendação Mobile: Um Estudo de Caso para Delivery”, <https://periodicos.unifacef.com.br/index.php/resiget/article/view/1463/1038>, Novembro.
- Lessing, L. (2002) “The Architecture of Innovation”, <https://scholarship.law.duke.edu/cgi/viewcontent.cgi?article=1162&context=dlj>, Setembro.
- Mendes, R. V. (2018) “Desenvolvimento de uma ferramenta para organização e gerenciamento de atividades de docentes”, <https://repositorio.ufu.br/bitstream/123456789/22185/1/DesenvolvimentoFerramentaOrganizacao.pdf>, Novembro.
- Oracle. (2018) “MySQL 8.0 Reference Manual”, <https://dev.mysql.com/doc/refman/8.0/en/>, Agosto.
- Oliveira, W. C., Martiniuk, V. C. (2021) “A Importância do Sistema de Informação nas Empresas”, http://fait.revista.inf.br/imagens_arquivos/arquivos_destaque/XDymmXAY62VisuQ_2021-6-8-16-34-45.pdf, Outubro.
- Rezende, A. A. de.; Marcelino, J. A. .; Miyaji, M. . (2020) “A Reinvenção das Vendas: As Estratégias das Empresas Brasileiras Para Gerar Receitas na Pandemia de Covid-19”, <https://revista.ioles.com.br/boca/index.php/revista/article/view/113>, Novembro.
- VIANA, B. (2017) “SIGAT (Sistema de gestão e agendamento de TCCs)”, <https://www.quixada.ufc.br/wp-content/uploads/2015/06/EncontrosUniversitarios-2017.pdf>, Novembro.
- VueJS. (2021) “Vue.js Server-Side Rendering Guide”, <https://ssr.vuejs.org/#what-is-server-side-rendering-ssr>, Dezembro.

APÊNDICE A – Demais *interfaces* do sistema (Dados Fictícios)

Novo Produto

| Cód. | Nome | Preço | Imagem | Aquí |
|------|------|-------|---|------------------------|
| 60 | Água | 10.5 |  | Editar |

Total 3 < 1 > Go to

Figura 16. Cliente *Interface* Caixa de pesquisa na interface de listagem

dashboard / Usuários / Todos os Usuários

Novo Usuário

| Cód. | Nome | E-mail | Cadastro | Filtrar |
|------|--------------------|---------------------|----------|------------------------|
| 1 | Marlon da Rosa | dev@dev.com.br | | Editar |
| 7 | Joshua Vincent | kite@dev.com.br | | Editar |
| 10 | Joel Petrucci | ohu@dev.com.br | | Editar |
| 11 | Fannie Santoni | silolcof@dev.com.br | | Editar |
| 12 | Jordan Tapinassi | ho@dev.com.br | | Editar |
| 69 | geek republica 000 | rg@gmail.com | | Editar |
| 70 | José Moraes | ze@gmail.com | | Editar |
| 71 | User 8 | 8@gmail.com | | Editar |
| 72 | User 9 | 9@gmail.com | | Editar |
| 73 | User 10 | 10@gmail.com | | Editar |

Total 13 < 1 2 > Go to

Figura 17. Administrador *Interface* Lista de Usuários

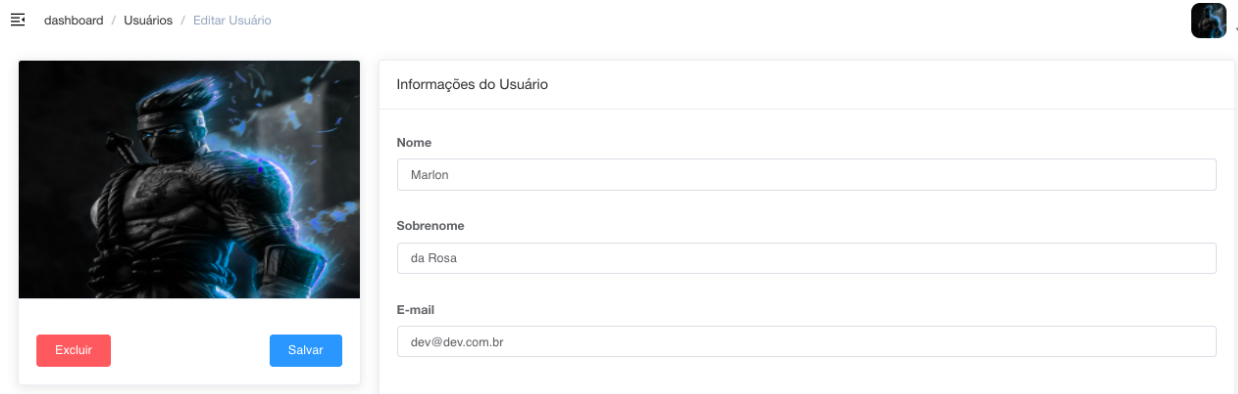


Figura 18. Administrador *interface* Editar Usuário

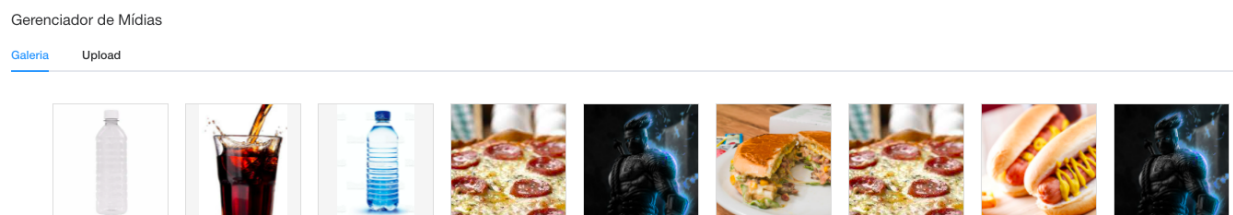


Figura 19. Administrador *Interface* da Galeria de Mídias

E-commerce Admin

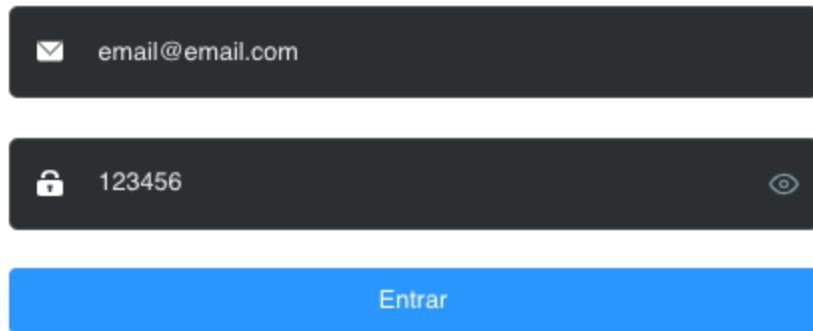
✉ email@email.com

🔒 🔓

Entrar

Figura 20. Interface de Login

E-commerce Admin



The image shows a login interface for an E-commerce Admin. It consists of three main elements: a title, an email input field, a password input field, and a login button. The title 'E-commerce Admin' is centered at the top. Below it is a dark grey rounded rectangle containing an envelope icon and the text 'email@email.com'. The next element is another dark grey rounded rectangle containing a lock icon, the text '123456', and an eye icon on the right side. At the bottom is a bright blue rounded rectangle with the text 'Entrar' centered inside.

Figura 21. *Interface de Login com Campo de Senha Visível*