Uma imagem com texto, Tipo de letra, logótipo, Azul elétrico

Os conteúdos gerados por IA poderão estar incorretos.

**PassaPraFrente**

**Proposta de Sistema**

**Grupo No. 4**

No.27970 – Hugo Pereira

No.27963 – Hugo Especial

No.27966 – Paulo Gonçalves

No.27959 – Duarte Pereira

No.27969 – Marco Cardoso

**Licenciatura em Engenharia Sistemas Informáticos**

**2ºano**

Barcelos | Março, 2025

# 1. Lista de Abreviaturas e Siglas

IPCA – Instituto Politécnico do Cávado e do Ave

UC – Unidade Curricular

BPMN – *Business Process Model and Notation*

HTTP – *Hypertext Transfer Protocol*

API – *Application Programming Interface*

JWT - *Json Web Token*

SEO - *Search Engine Optimization*

URL – *Uniform Resource Locator*

CSS – *Cascading Style Sheets*

DOM *– Document Object Model*

JS *– JavaScript*

US – *User Story*

XSS – *Cross-Site Scripting*

SQL – *Structured Query Language*

DDoS – Distributed Denial-of-Service

M – Must Have

S – Should Have

C – Could Have

W – Would Have

CdU – Casos de Uso

UML – *Unified Modeling Language*

Índice

[1. Lista de Abreviaturas e Siglas 2](#_Toc192429142)

[2. Introdução 7](#_Toc192429143)

[2.1. Caracterização do Grupo 8](#_Toc192429144)

[2.2. Metodologia de trabalho utilizada 9](#_Toc192429145)

[3. Proposta de Sistema 10](#_Toc192429146)

[3.1. Objetivos de negócio 10](#_Toc192429147)

[3.2. Operações a realizar pelo sistema 11](#_Toc192429148)

[3.3. Especificação do Sistema 11](#_Toc192429149)

[3.3.1. Requisitos Funcionais 12](#_Toc192429150)

[3.3.2. Requisitos Não Funcionais 13](#_Toc192429151)

[3.3.3. User Stories 14](#_Toc192429152)

[4. Tecnologias 16](#_Toc192429153)

[4.1. Bibliotecas utilizadas no *Front-End* 18](#_Toc192429154)

[4.2. Bibliotecas utilizadas no *Back-end* 19](#_Toc192429155)

[5. Processos de Negócio 20](#_Toc192429156)

[5.1. Diagrama BPMN – Venda 21](#_Toc192429157)

[5.2. Diagrama BPMN – Empréstimo 22](#_Toc192429158)

[5.3. Diagrama BPMN – Sorteio de Doação 23](#_Toc192429159)

[5.4. Diagrama de Casos de Uso 24](#_Toc192429160)

[6. Diagrama Entidade-Relação 34](#_Toc192429161)

[7. Diagrama de Classes 37](#_Toc192429162)

[7.1. Principais Classes e Métodos 38](#_Toc192429163)

[8. Diagramas de Sequência 43](#_Toc192429164)

[8.1. Diagrama de sequência de venda 44](#_Toc192429165)

[8.2. Diagrama de sequência de empréstimo 46](#_Toc192429166)

[8.3. Diagrama de sequência de sorteio 48](#_Toc192429167)

[9. Mockup’s 50](#_Toc192429168)

[10. Melhorias futuras 73](#_Toc192429169)

[11. Conclusão 74](#_Toc192429170)

[12. Anexos 75](#_Toc192429171)

[13. Webgrafia 76](#_Toc192429172)

Índice de Tabelas

[Tabela 1 – Requisitos Funcionais 12](#_Toc192429139)

[Tabela 2 – Requisitos não Funcionais 13](#_Toc192429140)

[Tabela 3 – User stories 15](#_Toc192429141)

Índice de Figuras

[Figura 1 - PassaPraFrente 7](#_Toc192430592)

[Figura 2 – React vs. outras tecnologias 16](#_Toc192430593)

[Figura 3 - Diagrama BPMN – Venda 21](#_Toc192430594)

[Figura 4 - Diagrama BPMN – Empréstimo 22](#_Toc192430595)

[Figura 5 - Diagrama BPMN - Sorteio de Doação 23](#_Toc192430596)

[Figura 6 – Diagrama de CdU – Pacotes 24](#_Toc192430597)

[Figura 7 - Diagrama CdU – Venda 26](#_Toc192430598)

[Figura 8 - Diagrama CdU - Doação 28](#_Toc192430599)

[Figura 9 - Diagrama CdU – Empréstimo 30](#_Toc192430600)

[Figura 10 - Diagrama CdU - Gestão de Utilizadores 32](#_Toc192430601)

[Figura 11 - Diagrama ER 34](#_Toc192430602)

[Figura 12 – Diagrama de classes 37](#_Toc192430603)

[Figura 13 – Diagrama de sequência - Venda 44](#_Toc192430604)

[Figura 14 – Diagrama de sequência - Empréstimo 46](#_Toc192430605)

[Figura 15 – Diagrama de sequência - Sorteio 48](#_Toc192430606)

[Figura 16 - Página de Login 50](#_Toc192430607)

[Figura 17 - Página Sign In 51](#_Toc192430608)

[Figura 18 - Página de Alterar-Passe 52](#_Toc192430609)

[Figura 19 - Página de Alterar-Passe 53](#_Toc192430610)

[Figura 20 - Ecrã Inicial 54](#_Toc192430611)

[Figura 21 - Página de Vendas 55](#_Toc192430612)

[Figura 22 - Detalhes venda 56](#_Toc192430613)

[Figura 23 - Detalhes empréstimo 57](#_Toc192430614)

[Figura 24 - Detalhes sorteio 57](#_Toc192430615)

[Figura 25 - Proposta venda 58](#_Toc192430616)

[Figura 26 - Proposta empréstimo 59](#_Toc192430617)

[Figura 27 - Página favoritos 60](#_Toc192430618)

[Figura 28 – Página sem favoritos 60](#_Toc192430619)

[Figura 29 - Adicionar venda 61](#_Toc192430620)

[Figura 30 - Adicionar empréstimo 62](#_Toc192430621)

[Figura 31 - Adicionar sorteio 62](#_Toc192430622)

[Figura 32 - Página da conta 63](#_Toc192430623)

[Figura 33 - Página da conta 2 64](#_Toc192430624)

[Figura 34 - Editar conta 65](#_Toc192430625)

[Figura 35 - Proposta 66](#_Toc192430626)

[Figura 36 - Editar venda 67](#_Toc192430627)

[Figura 37 - Editar empréstimo 68](#_Toc192430628)

[Figura 38 - Editar sorteio 68](#_Toc192430629)

[Figura 39 - Avaliação 1 69](#_Toc192430630)

[Figura 40 - Avaliação 2 69](#_Toc192430631)

[Figura 41 - Página de administrador 70](#_Toc192430632)

# 2. Introdução

Uma imagem com logótipo, Tipo de letra, símbolo, Gráficos

Os conteúdos gerados por IA poderão estar incorretos.**Contextualização do Relatório**

O presente relatório documenta o desenvolvimento de um sistema inovador para a venda, empréstimo e doação de artigos dentro de uma comunidade local. O objetivo principal do projeto é criar uma plataforma intuitiva que conecte moradores interessados em comercializar produtos usados. Dessa forma, o sistema contribui para a economia colaborativa, promovendo a reutilização de bens e reduzindo o desperdício material, que é um problema comum em muitas áreas urbanas. Além disso, a iniciativa incentiva a interação entre os membros da vizinhança, fortalecendo o senso de comunidade e criando uma rede de apoio baseada na sustentabilidade.

Figura 1 - PassaPraFrente

**Motivação e Objetivos no Âmbito da UC do Projeto**

A motivação principal do projeto é desenvolver uma solução prática para um problema real comum em muitas áreas urbanas: o desperdício material. O projeto faz parte da UC de Projeto de Desenvolvimento de *Software* e visa aplicar conceitos teóricos de modelação, análise e design de sistemas assim como a sua implementação desenvolvendo uma aplicação *web* que suporte o projeto.

## 2.1. Caracterização do Grupo

O grupo é composto por cinco elementos, todos estudantes do 2º ano da Licenciatura em Engenharia de Sistemas Informáticos do IPCA. O objetivo do grupo é desenvolver um projeto no âmbito da unidade curricular de Projeto de Desenvolvimento de *Software*.

O grupo utilizou uma metodologia chamada *SCRUM*, uma metodologia que facilita a gestão e a execução de projetos de forma iterativa e incremental. O grupo trabalhou em ciclos curtos chamados *sprints*, nos quais tarefas priorizadas foram desenvolvidas e revisadas continuamente. Papéis como *Scrum* *Master* e *Product Owner* ajudaram a manter o foco nos objetivos e na entrega de valor contínuo.

Cada membro tem um respetivo cargo que deve ser respeitado para existir uma maior eficiência no desenvolvimento do projeto, sendo estes:

* *Product Owner*: Marco Cardoso – Responsável por definir e gerir os requisitos do projeto, assegurando que o produto final corresponde às necessidades dos utilizadores e aos objetivos propostos.
* *Scrum Master*: Hugo Pereira – Garante que a metodologia *Scrum* é seguida corretamente, removendo impedimentos que possam surgir no desenvolvimento e facilitando a comunicação entre os membros da equipa.
* *Development team member*: Duarte Pereira, Paulo Gonçalves, Hugo Especial – Encarregado do desenvolvimento e implementação de funcionalidades, trabalhando na programação e integração de diferentes componentes do sistema.

## 2.2. Metodologia de trabalho utilizada

Para a realização deste projeto, foi utilizado uma ferramenta que permite a interação de todos os membros do grupo onde são reunidas informações importantes e algumas notas para termos noção de como o projeto está a ser desenvolvido. A ferramenta chama-se *Milanote*.

*Milanote* é uma ferramenta de organização visual voltada para profissionais, que permite planear, desenvolver e organizar ideias em um espaço intuitivo. Com uma interface baseada em painéis, ela facilita a criação de listas de tarefas e diagramas de forma colaborativa. É especialmente útil para equipas, pois combina elementos de um quadro branco digital com a flexibilidade de armazenar e conectar informações, imagens e *links*, promovendo o *brainstorming* e a colaboração em tempo real.

Utilizámos o *Jira* para a planificação do trabalho, organizar tarefas, prazos e prioridades de maneira eficiente. A ferramenta permite acompanhar o progresso de cada etapa do projeto, distribuindo responsabilidades entre os membros da equipa.

Para garantir um design organizado, foi utilizada a plataforma *Figma* para a criação e prototipagem das *mockups* do sistema. A ferramenta permitiu que todos os membros da equipa trabalhassem simultaneamente no design, facilitando a troca de *feedbacks* e ajustes em tempo real.

Para garantir um desenvolvimento colaborativo, foi utilizada a plataforma *Git* para controlo de versões e gestão do código-fonte. O repositório foi criado no *GitHub*, permitindo que cada membro da equipa contribuísse de forma eficiente.

# 3. Proposta de Sistema

A proposta do sistema, denominada “PassaPraFrente”, é uma solução tecnológica voltada para facilitar a venda, o empréstimo e a doação de artigos dentro de comunidades locais. O sistema busca solucionar desafios como a dificuldade de conectar doadores e interessados, a subutilização de bens em bom estado e a falta de uma comunicação eficiente entre os moradores.

Através de uma plataforma intuitiva, os utilizadores podem registar artigos para venda, empréstimo e doação, e encontrar ofertas próximas, promovendo uma economia mais sustentável e fortalecendo os laços comunitários.

## 3.1. Objetivos de negócio

Este negócio tem como objetivo resolver a dificuldade de conexão entre doadores, vendedores, emprestadores e aqueles que necessitam de artigos, um desafio comum nas comunidades. Muitas vezes, bens em bom estado são descartados devido à falta de um canal eficiente para doação, venda ou empréstimo, enquanto outras pessoas enfrentam dificuldades em acessar esses artigos de forma simples.

A falta de visibilidade sobre os artigos disponíveis e a ausência de um sistema estruturado para facilitar esses processos resulta em desperdício de recursos e subutilização de bens. Além disso, a comunicação ineficaz entre os membros da comunidade dificulta a criação de uma rede ativa e sustentável para o compartilhamento de artigos.

## 3.2. Operações a realizar pelo sistema

**Método MoSCoW**

O método MoSCoW é uma técnica de priorização que classifica requisitos em *Must have* (essenciais), *Should have* (importantes, mas não críticos), *Could have* (desejáveis, mas opcionais) e *Won't have* (não incluídos no momento). Este método ajuda as equipas a focarem no que é essencial para o sucesso do projeto dentro dos prazos e recursos disponíveis.

**Funcionalidades com a prioridade definida:**

* Catálogo de artigos - M
* Sistema de avaliações – M
* Sistema de vendas – M
* Sistema de empréstimo - M
* Sistema de sorteio - M
* Sistema de aprovação de publicações - S
* Função de anúncio temporário - C
* Sistema de chat - W
* Sistema de favoritos - W
* Sistema de reserva de objeto – W

## 3.3. Especificação do Sistema

As especificações de sistema são essenciais no desenvolvimento de *software*, pois definem requisitos técnicos detalhados, garantindo que o sistema atenda às necessidades do cliente. As especificações servem como um guia para os programadores, ajudando a estruturar as funcionalidades, o desempenho e as características esperadas do *software* ao longo do processo de criação.

### 3.3.1. Requisitos Funcionais

Requisitos funcionais são especificações que definem o que o sistema deve fazer, descrevendo as suas funcionalidades e comportamentos esperados. Estes requisitos determinam as interações entre o utilizador e o sistema, como registo, *login*, pesquisa e envio de propostas. Esses requisitos são essenciais para garantir que o *software* cumpra os seus objetivos e atenda às necessidades dos utilizadores.

|  |  |
| --- | --- |
| **Código** | **Descrição** |
| RF01 | O sistema deve permitir que moradores se registem, façam login e editem os seus perfis. |
| RF02 | Os utilizadores devem poder participar em sorteios. |
| RF03 | Após uma venda, empréstimo ou sorteio, os utilizadores devem poder avaliar uns aos outros. |
| RF04 | O sistema deve permitir que os utilizadores façam proposta para negociar o preço da venda e negociar o preço e datas do empréstimo. |
| RF05 | Os utilizadores devem poder ver o perfil de outros utilizadores para ver avaliações. |
| RF06 | O sistema deve permitir que o utilizador recupere sua palavra-passe caso a esqueça. |
| RF07 | Os utilizadores devem poder visualizar o seu histórico de vendas e empréstimos. |

Tabela 1 – Requisitos Funcionais

### 3.3.2. Requisitos Não Funcionais

Requisitos não funcionais são especificações que definem como o sistema deve comportar-se, focando em aspetos como desempenho, segurança, usabilidade e fiabilidade, em vez das funcionalidades específicas. Estes requisitos determinam a qualidade e as restrições do sistema, como tempo de resposta, disponibilidade e escalabilidade.

|  |  |  |
| --- | --- | --- |
| **Código** | **Categoria** | **Descrição** |
| RNF01 | Segurança | Na base de dados, a palavra-passe não deve ser texto limpo, mas sim uma *hash*. |
| RNF02 | Segurança | O sistema deve ser protegido contra injeções de SQL. |
| RNF03 | Portabilidade | O sistema deve garantir que os utilizadores possam acessar os dados ou realizar tarefas independentemente do dispositivo. |
| RNF04 | Usabilidade | A interface do sistema deve ser intuitiva e acessível, permitindo que a maioria de utilizadores com diferentes níveis de experiência tecnológica consigam utilizá-lo facilmente. |
| RNF05 | Segurança | O sistema deve restringir o acesso aos seus recursos de forma que apenas origens confiáveis (domínios específicos) possam interagir com a API. |
| RNF06 | Segurança | |  |  | | --- | --- | | O sistema deve limitar o número de pedidos feitos por um mesmo utilizador em um curto período, a fim de evitar abusos e proteger contra-ataques DDoS. | | |  | |
| RNF07 | Desempenho | |  | | --- | | O sistema deve garantir bom desempenho, ao controlar o número de pedidos e evitando que o servidor esteja sobrecarregado. |  |  | | --- | |  | |

Tabela 2 – Requisitos não Funcionais

### 3.3.3. User Stories

*User stories* são descrições curtas e simples de funcionalidades ou requisitos de um produto do ponto de vista do utilizador final. Estas descrições são amplamente utilizadas em metodologias ágeis para capturar necessidades e orientar o desenvolvimento.

Normalmente seguem um formato padrão: "Como [tipo de utilizador], quero [ação] para [benefício]". Esse formato ajuda a garantir que o foco esteja na experiência do utilizador e no valor entregue pelo artigo.

|  |  |
| --- | --- |
| **Código** | ***User Story*** |
| **US01** | Eu como comprador quero ver todos os artigos para os poder comprar. |
| **US02** | Eu como comprador quero avaliar os vendedores dos artigos que compro para que outros compradores saibam se é de confiança. |
| **US03** | Eu como vendedor quero ver a avaliação dos compradores para ver se gostaram do artigo. |
| **US04** | Eu como vendedor quero fazer anúncios temporários para fazer empréstimos/vendas. |
| **US05** | Eu como vendedor quero emprestar artigos que não uso para vizinhos que possam precisar. |
| **US06** | Eu como comprador quero pedir artigos emprestados a vizinhos para uso numa determinada situação. |
| **US07** | Eu como vendedor quero criar sorteios para doar artigos que não preciso. |
| **US08** | Eu como comprador quero participar em sorteios de doações para conseguir artigos que esteja a precisar. |
| **US09** | Eu como comprador quero enviar propostas a vendas/empréstimos para conseguir o melhor preço. |
| **US10** | Eu como vendedor quero aceitar/rejeitar propostas para conseguir arranjar o melhor preço para mim. |
| **US11** | Eu como admin quero poder aceitar ou rejeitar anúncios de venda, empréstimo e doação. |

Tabela 3 – User stories

# 4. Tecnologias

O projeto será desenvolvido utilizando uma *stack* moderna e robusta, garantindo escalabilidade e desempenho.

***Front-end***

Para a interface do utilizador, utilizaremos *ReactJS*, permitindo a criação de uma experiência dinâmica e interativa. O *ReactJS* possibilita o desenvolvimento de componentes reutilizáveis e melhora o desempenho da aplicação por meio da virtual *DOM*.

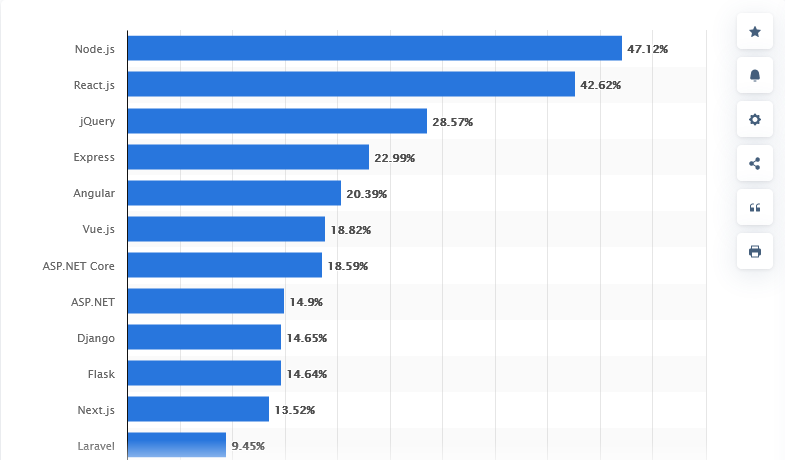
“De acordo com um relatório publicado no “Statista”, o *ReactJS* é o segundo *framework* web mais preferido, com 42,64% dos programadores enfatizando o uso do *React* em relação a outros *frameworks*.”[[[1]](#footnote-2)]

Figura 2 – React vs. outras tecnologias

De acordo com diversas pesquisas e comparações realizadas em *websites* especializados em programação *web*, o *React* costuma figurar entre os frameworks mais populares e preferidos pelos programadores, ficando frequentemente no topo das listas de desempenho e adoção, em comparação com outras alternativas como *AngularJS* e *VueJS*.

**Back-end**

No lado do servidor, utilizaremos *NodeJS* com o *framework ExpressJS*, que proporciona uma estrutura ágil e eficiente para a construção de APIs *RESTful*. O *ExpressJS* facilita o roteamento, manipulação de pedidos e integração com base de dados.

Escolhemos estas tecnologias devido a três membros da equipa já as terem utilizado num projeto anterior, o que facilitará o processo de implementação. Além disso, o uso do npm possibilita uma gestão eficiente dos pacotes. Como destacado pelo próprio website do npm: "O Registo npm gratuito tornou-se o centro da partilha de código JavaScript e, com mais de dois milhões de pacotes, o maior registo de *software* do mundo"[[[2]](#footnote-3)].

**Base de Dados**

A base de dados será *SQLServer*. A biblioteca mssql será utilizada para a conexão entre o *back-end* e a base de dados. Todos os membros da equipa já têm experiência prévia com o *SQLServer*, o que facilitará a integração e otimização da base de dados.

## 4.1. Bibliotecas utilizadas no *Front-End*

Neste projeto, planeamos utilizar algumas bibliotecas essenciais para garantir uma experiência de desenvolvimento eficiente e otimizada:

* **react-router-dom**: Para o roteamento e navegação entre páginas.
* **Axios**: Para a comunicação com a API.
* **Zustand**: Uma opção conhecida para gestão de estado, e é a escolha ideal para o nosso projeto específico, pois é simples, leve e fácil de integrar, o que o torna adequado para projetos de médio porte como o nosso.
* **Formik**: Para a gestão de formulários e validações de dados.
* **tailwindcss**: É um *framework* de classes utilitárias que torna a personalização rápida e flexível. Este *framework* facilita a responsividade, melhora a produtividade e gera apenas o CSS necessário, mantendo o projeto leve e eficiente.

Embora tenhamos outras bibliotecas em nosso projeto, estas são provavelmente as mais importantes, pois desempenham um papel central no desenvolvimento da aplicação.

## 4.2. Bibliotecas utilizadas no *Back-end*

Para garantir uma aplicação segura, modular e bem estruturada, utilizaremos as seguintes bibliotecas:

* **Jest**: Para testes, garantindo a qualidade do código.
* **Mssql**: Para integração eficiente com o base de dados *SQLServer*.
* **Nodemailer**: Para envio de emails, nomeadamente para confirmação de conta e recuperação de palavra-passe.
* **Ejs**: Para criação de *templates* de email dinâmicos, permitindo mensagens personalizadas para os utilizadores.
* **Jsonwebtoken**: Para autenticação e gestão de sessões por meio de *tokens JWT*.
* **Config**: Facilita a gestão de variáveis ambientais, permitindo configurações dinâmicas para diferentes ambientes.
* **Cookie-Parser**: Analisa *cookies* enviados pelo cliente, facilitando o acesso a dados armazenados para autenticação e outras funcionalidades.
* **Helmet**: Configura *headers* de segurança para proteger a aplicação contra-ataques como XSS e *Clickjacking*.
* **Express-rate-limiter:** Para limitar o número de pedidos de um mesmo cliente, protegendo a aplicação contra abusos e ataques DDoS.

# 5. Processos de Negócio

Os processos de negócio no contexto do sistema “PassaPraFrente” são os fluxos de atividades principais que garantem a gestão eficiente da venda, doação e empréstimo de artigos dentro de uma comunidade de vizinhos. Estes processos envolvem a interação entre os diferentes utilizadores e o uso do sistema para facilitar a comunicação, a negociação e a entrega dos artigos.

Neste documento estão presentes 3 processos de negócio:

* Venda
* Empréstimo
* Sorteio de doação

## 5.1. Diagrama BPMN – Venda

[**Ver anexo 1**](#_Anexos)

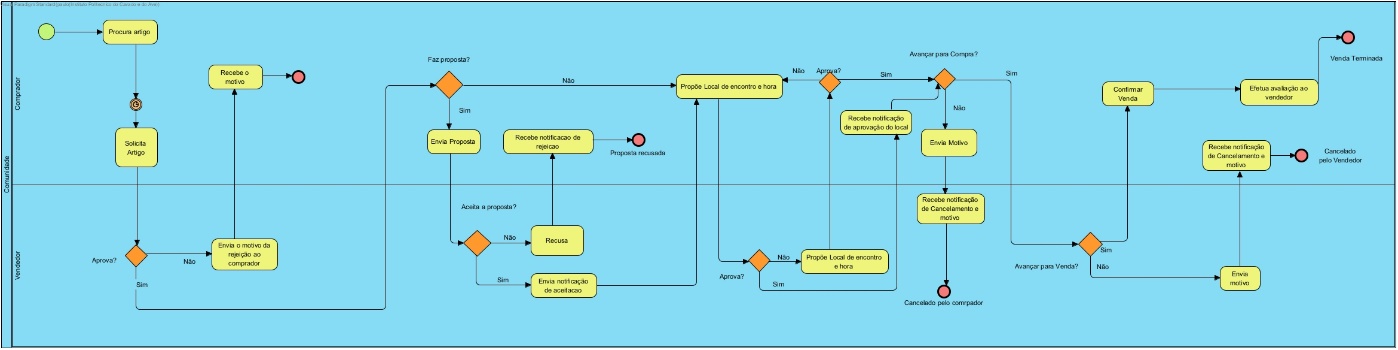


Figura 3 - Diagrama BPMN – Venda

O diagrama ilustra o processo de venda entre um comprador e um vendedor, abrangendo todas as etapas desde a procura do artigo até à conclusão da transação.

O fluxo inclui a solicitação e aprovação da compra, a possibilidade de propostas de preço, a definição do local de encontro e a confirmação da venda. Além disso, considera cenários de cancelamento por ambas as partes, garantindo um registo completo das interações e decisões ao longo da negociação.

## 5.2. Diagrama BPMN – Empréstimo

[**Ver anexo 2**](#_Anexos)

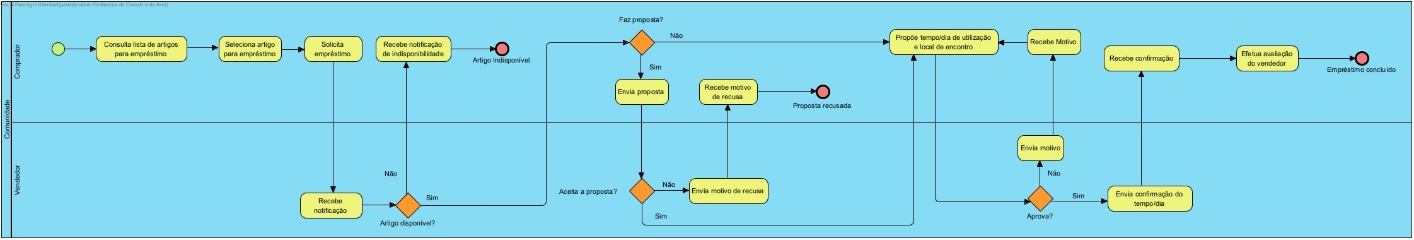


Figura 4 - Diagrama BPMN – Empréstimo

O diagrama representa o fluxo de um processo de empréstimo entre um comprador e um vendedor. O processo inicia-se com a consulta e seleção de um artigo para empréstimo, seguida do pedido feito pelo comprador.

O vendedor verifica a disponibilidade do artigo. Se não estiver disponível, o comprador é notificado. Caso esteja disponível, o comprador tem a opção de fazer uma proposta referente ao preço e às datas. Se não fizer, define-se diretamente o dia e o local do encontro. Se optar por fazer uma proposta, o vendedor pode aceitá-la ou recusá-la. Em caso de recusa, ele informa o motivo ao comprador.

Na etapa de definição do dia e local do encontro, o vendedor pode aprovar ou rejeitar a proposta, devendo justificar a rejeição, se for o caso. Após a aprovação, a confirmação do horário e do local é enviada e é feita a avaliação do vendedor por parte do comprador, concluindo assim o processo de empréstimo.

## 5.3. Diagrama BPMN – Sorteio de Doação

[**Ver anexo 3**](#_Anexos)

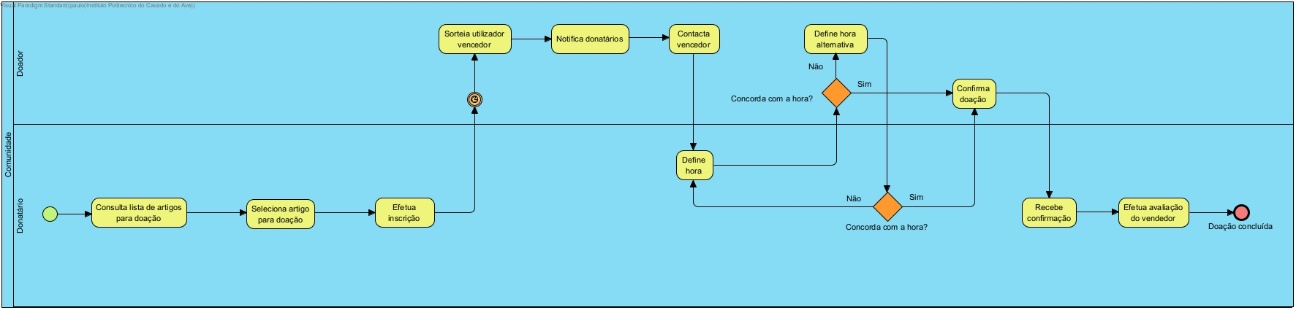


Figura 5 - Diagrama BPMN - Sorteio de Doação

O diagrama representa o fluxo de um processo de doação entre um comprador e um vendedor, realizado por meio de um sorteio. O processo inicia-se pela parte do donatário com a consulta e seleção de um artigo listado para doação.

Os interessados em participar do sorteio inscrevem-se para concorrer à doação do artigo. Após o encerramento das inscrições, o vendedor notifica todos os participantes sobre o resultado e realiza o sorteio para selecionar o vencedor, que é posteriormente contactado.

O vencedor define um horário e um local para o encontro, sendo necessário confirmar a disponibilidade. Caso haja discordância, pode-se sugerir uma alternativa. Após a confirmação do horário e do local, a doação é validada, e o vencedor recebe a confirmação.

Por fim, o vencedor avalia o doador, encerrando o processo de doação.

## 5.4. Diagrama de Casos de Uso

Uma imagem com diagrama, texto, file, Esquema

Os conteúdos gerados por IA poderão estar incorretos.Os casos de uso são uma técnica de modelação que descreve como os utilizadores interagem com o sistema para atingir objetivos específicos. No contexto do “PassaPraFrente”, os casos de uso capturam as interações entre os moradores e o sistema para facilitar a venda, empréstimo e doação de artigos de maneira eficiente e organizada.

Figura 6 – Diagrama de CdU – Pacotes

**Visão Geral**

Este diagrama representa um Diagrama de Caso de Uso (CdU) com Pacotes, que organiza diferentes subsistemas dentro de um sistema maior chamado “PassaPraFrente”.

**Atores e as suas interações**

**Administrador:**

Interage com o Sistema de Gestão de Utilizadores, Venda, Doação e Empréstimo.

**Utilizador:**

Interage com os sistemas de Venda, Doação, Empréstimo e também com o Sistema de Gestão de Utilizadores.

**Pacotes/Sistemas:**

* Sistema de Venda
* Sistema de Doação
* Sistema de Gestão de Utilizadores
* Sistema de Empréstimo

**Uma imagem com texto, diagrama, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.**

Figura 7 - Diagrama CdU – Venda

**Visão Geral**

Este diagrama representa o Sistema de Venda e mostra os casos de uso relacionados às interações dos atores “Utilizador” e “Administrador”.

O diagrama destaca as ações que cada ator pode realizar dentro do sistema.

**Atores e as suas interações**

**Utilizador:**

Pode pesquisar venda, criar venda, fazer proposta, solicitar venda, cancelar venda e avaliar vendedor.

O caso de uso “Criar venda” pode incluir o caso de uso “Criar artigo”, indicando que, para criar uma venda, pode ser necessário criar um artigo.

**Administrador:**

Possui permissão para gerir vendas, indicando um nível maior de controlo sobre o sistema de vendas.

**Relações específicas**

A relação "*extend*" entre “Criar venda” e “Criar artigo” significa que sempre que um utilizador cria uma venda, pode ser necessário criar um artigo.

Uma imagem com texto, diagrama, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Figura 8 - Diagrama CdU - Doação

**Visão Geral**

Este diagrama representa o Sistema de Doação, destacando as interações dos atores “Utilizador” e “Administrador” com os principais casos de uso do sistema.

**Atores e as suas interações**

**Utilizador:**

Pode pesquisar sorteio, criar sorteio, cancelar sorteio, sortear vencedor, efetuar inscrição e avaliar vendedor.

O caso de uso “Criar sorteio” pode incluir “Criar artigo”, indicando que para criar um sorteio pode ser necessário criar um artigo.

**Administrador:**

Tem permissão para gerir sorteios, indicando que pode administrar as doações e sorteios realizados no sistema.

**Relações específicas**

A relação "*extend*" entre “Criar sorteio” e “Criar artigo” significa que ao criar o sorteio pode precisar de criar o artigo.

Uma imagem com texto, diagrama, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

Figura 9 - Diagrama CdU – Empréstimo

**Visão Geral**

Este diagrama representa o “Sistema de Empréstimo”, destacando as interações dos atores “Utilizador” e “Administrador” com os principais casos de uso do sistema.

**Atores e as suas interações**

**Utilizador:**

Pode pesquisar empréstimo, solicitar artigo para empréstimo, criar empréstimo, fazer proposta, cancelar empréstimo e avaliar vendedor.

O caso de uso “Criar empréstimo” pode incluir “Criar artigo”, indicando que, para criar um empréstimo, pode ser necessário criar um artigo.

**Administrador:**

Tem permissão para gerir empréstimos, indicando que pode administrar os empréstimos realizados no sistema.

**Relações Específicas**

A relação "*extend*" entre “Criar empréstimo” e “Criar artigo” significa que sempre que um utilizador cria um empréstimo, pode precisar de criar um artigo.

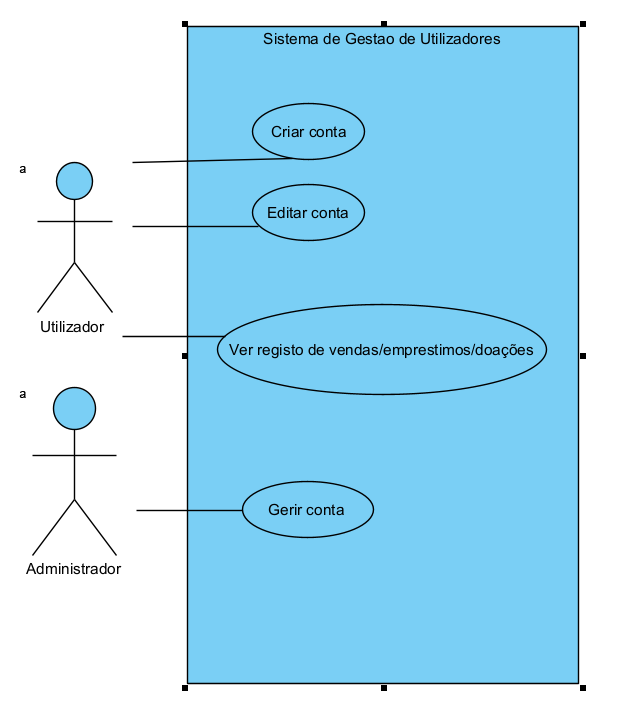


Figura 10 - Diagrama CdU - Gestão de Utilizadores

**Visão Geral**

Este diagrama representa o Sistema de Gestão de Utilizadores, destacando as interações dos atores “Utilizador” e “Administrador” com os principais casos de uso do sistema.

**Atores e as suas interações**

**Utilizador:**

Pode criar conta, editar conta e ver registo de vendas, empréstimos e doações.

Essas funcionalidades permitem que o utilizador mantenha a sua conta atualizada e acompanhe as suas atividades no sistema.

**Administrador:**

Tem permissão para gerir conta, o que sugere que pode administrar contas de utilizadores, possivelmente alterando informações, desativando contas, entre outras ações de moderação.

**Relações Específicas**

O caso de uso **"**Ver registo de vendas e empréstimos**"** indica que o utilizador pode visualizar suas transações no sistema, o que pode ser essencial para controlo e histórico de atividades.

“Gerir conta”, atribuído ao administrador, sugere que este pode realizar ações administrativas sobre as contas dos utilizadores.

# 6. Diagrama Entidade-Relação

O diagrama Entidade-Relação é utilizado para modelar a estrutura de uma base de dados, representando visualmente as entidades (tabelas), os seus atributos (campos) e as relações entre eles. Este diagrama ajuda no planeamento e na organização dos dados, facilitando a compreensão do sistema e garantindo que as relações entre os elementos estejam bem definidas antes da implementação.

**Uma imagem com texto, captura de ecrã, Retângulo, Paralelo

Os conteúdos gerados por IA poderão estar incorretos.**[**Ver an****exo 4**](#_Anexos)

Figura 11 - Diagrama ER

**Principais Entidades**

1. **Utilizador**

* Representa os utilizadores do sistema.
* Contém atributos como nome, data de nascimento, imagem e um tipo de utilizador (que pode ser utilizador ou administrador).

1. **Autenticação**

* Armazena credenciais dos utilizadores.
* Inclui email, password e um campo para confirmar o email.

1. **Venda, Proposta de Venda e Transação de Venda**

* “Venda” representa um artigo colocado à venda por um utilizador.
* “PropostaVenda” permite que outro utilizador proponha um novo valor para a compra do artigo.
* “TransacaoVenda” regista a venda final de um artigo, incluindo valores e o utilizador como comprador.

1. **Empréstimo, Proposta de Empréstimo e Transação de Empréstimo**

* “Emprestimo” representa um artigo disponibilizado para empréstimo.
* “PropostaEmprestimo” permite que outro utilizador proponha novas datas e preço para o empréstimo.
* “TransacaoEmprestimo” regista o acordo final de um empréstimo entre utilizadores.

1. **Sorteio**

* “Sorteio” é um evento onde um artigo pode ser sorteado entre utilizadores inscritos.
* “InscricaoSorteio” permite que utilizadores participem do sorteio.
* “VencedorSorteio” armazena o vencedor do sorteio.

1. **Estado**

* “Estado” indica o estado da “Venda”, “Emprestimo” e do “Sorteio” (ex: concluído, aberto).

1. **Artigo, Categoria e Condição**

* “Artigo” representa os artigos disponíveis no sistema, ligados a categorias e condições.
* “Categoria” define a classificação dos artigos.
* “Condicao” indica a condição do artigo (ex: novo, usado).

1. **Imagem**

* Associa imagens aos artigos para melhor visualização.

# 7. Diagrama de Classes

Um diagrama de classes é um dos principais diagramas da UML e é utilizado para representar a estrutura estática de um sistema orientado a objetos. Ele descreve as classes presentes no sistema, os seus atributos, métodos e as relações entre as classes.

Uma imagem com texto, diagrama, Esquema, captura de ecrã

Os conteúdos gerados por IA poderão estar incorretos.[**Ver anexo 5**](#_Anexos)

Figura 12 – Diagrama de classes

### 7.1. Principais Classes e Métodos

1. **Autenticação e Utilizador**
   * A classe “Autenticacao” gere o login e registo dos utilizadores e inclui métodos como:
     + createActivationToken(utilizador: Utilizador): string – Cria um *token* para ativação da conta.
     + activateUser(user: Utilizador): boolean – Ativa a conta de um utilizador.
     + loginUser(): string – Realiza a autenticação do utilizador.
     + registrationUser(): Utilizador – Regista um utilizador no sistema.
     + logoutUser(): boolean – Encerra a sessão do utilizador.
   * A classe “Utilizador” gere perfis e contém métodos como:
     + getAllUsers(): Utilizador[] – Retorna a lista de utilizadores.
     + getUserById(id: int): Utilizador – Obtém um utilizador pelo ID.
     + getUserByEmail (email: string): Utilizador – Obtém um utilizador pelo email.
     + updateUserInfo(): Utilizador – Atualiza informações do utilizador.
     + requestPasswordReset(): string – Envia um URL para o utilizador conseguir alterar a palavra-passe.
     + getUserInfo(): Utilizador – Obtém os dados do utilizador que se encontra logado.
     + deleteUser(): boolean – Elimina um utilizador do sistema.
2. **Transações e Propostas**
   * A classe “Transacao” lida com operações finais dos processos de negócio e inclui métodos como:
     + createTransaction(): Transacao – Cria uma nova transação.
     + cancelTransaction(): boolean – Cancela uma transação ativa.
     + getTransactions(): Transacao[] – Obtém todas as transações registadas.
     + getTransactionById(): Transacao – Obtém uma transação pelo ID.
   * A classe “Proposta” permite que os utilizadores façam propostas para venda ou empréstimo de artigos, com métodos como:
     + createProposal(): Proposta – Cria uma nova proposta.
     + acceptProposal(): boolean – Aceita uma proposta existente.
     + denyProposal(): boolean – Rejeita uma proposta.
     + getProposalById() – Obtém uma proposta pelo ID.
3. **Venda e empréstimo**
   * A classe “Venda” representa artigos à venda e inclui métodos como:
     + createSale(): Venda – Regista um novo artigo para venda.
     + editSale(): Venda – Edita informações da venda.
     + getSaleById(): Venda[] – Obtém venda pelo ID.
     + deleteSale(): boolean – Elimina um anúncio de venda.
     + getAllSales(): Venda[] – Obtém todos os anúncios de venda.
     + getAvailableSales(): Venda[] – Obtém todos os anúncios de venda disponíveis.
     + getUserSales(): Venda[] – Obtém o histórico de vendas de um utilizador específico.
   * A classe “Emprestimo” representa os empréstimos, contendo métodos como:
     + createLoan(): Emprestimo – Cria um novo empréstimo.
     + editLoan(): Emprestimo – Edita informações de um empréstimo.
     + getLoanById(): Emprestimo – Obtém um empréstimo pelo ID.
     + deleteLoan(): boolean – Elimina um anúncio de empréstimo.
     + getAllLoans(): Emprestimo[] – Obtém todos os anúncios de empréstimo.
     + getAvailableLoans(): Emprestimo[] – Obtém todos os anúncios de empréstimo disponíveis.
     + getUserLoans(): Emprestimo [] – Obtém o histórico de empréstimos de um utilizador específico.
4. **Sorteios e Inscrições**
   * A classe “Sorteio” representa eventos de sorteio no sistema e possui métodos como:
     + createDraw(): Sorteio – Cria um novo sorteio.
     + getDrawById(): Sorteio[] – Obtém um sorteio pelo ID.
     + deleteDraw(): boolean – Elimina um sorteio.
     + getAllDraws(): Sorteio[] – Obtém todos os anúncios de sorteio.
     + wonDraw(): boolean – Verifica se um utilizador venceu um sorteio.
     + lostDraw(): boolean – Verifica se um utilizador perdeu um sorteio.
     + getAvailableDraws(): Sorteio[] – Obtém todos os anúncios de sorteio disponíveis.
     + getUserDraws(): Sorteio[] – Obtém o histórico de sorteios de um utilizador específico.
   * A classe “InscricaoSorteio” permite que utilizadores participem de sorteios e contém os métodos:
     + createRegistration(): InscricaoSorteio – Regista um utilizador em um sorteio.
     + getRegistrations(): InscricaoSorteio – Obtém todas as inscrições de um determinado sorteio.
   * A classe “VencedorSorteio” representa um vencedor de determinado sorteio e contém os seguintes métodos:
     + drawWinner(): Utilizador – Sortear um utilizador vencedor do sorteio.
     + getWinner(): Utilizador – Obtém o utilizador que venceu o sorteio.
5. **Estado**
   * A classe “Estado” indica o estado da “Venda”, “Emprestimo” e do “Sorteio”.
6. **Artigos e características**
   * A classe “Artigo” representa os artigos no sistema.
   * A classe “Categoria” classifica os artigos.
   * A classe “Condicao” define a condição do artigo.
   * A classe “Imagem” contém uma ou mais imagens associada a um artigo.

# 8. Diagramas de Sequência

Os diagramas de sequência são utilizados para representar a interação entre objetos ao longo do tempo e mostram como diferentes atores e sistemas interagem para atingir um objetivo específico. Cada diagrama foi elaborado para um processo de negócio específico, detalhando as trocas de mensagens, os eventos e as ações executadas ao longo do fluxo.

Foram criados diagramas de sequência para os processos de negócio de doação, venda e empréstimo. Estes diagramas ajudaram a identificar ineficiências, facilitaram a documentação e auxiliarão na implementação do sistema.

## 8.1. Diagrama de sequência de venda

Este diagrama de sequência representa o fluxo de um processo de venda, detalhando a interação entre o utilizador, o sistema de venda, a proposta de venda e a transação de venda. Em baixo encontra-se uma descrição detalhada.

[**Ver an****exo 6**](#_Anexos)Uma imagem com texto, Paralelo, diagrama, file

Os conteúdos gerados por IA poderão estar incorretos.

Figura 13 – Diagrama de sequência - Venda

**1.** O utilizador inicia a interação ao pedir e ao ver as vendas disponíveis (getAvailableSales()).

**2.** O sistema responde com a lista de vendas disponíveis (availableSales).

**3.** O utilizador seleciona uma venda específica (getSaleById()).

**4.** O sistema retorna a venda escolhida (chosenSale).

A partir deste ponto, o fluxo divide-se em diferentes possibilidades:

* **Se o utilizador faz uma proposta:**

**5.** A proposta é criada (createPropostal())

* + - **Se o vendedor aceita a proposta (acceptProposal()):**

**6.** É enviada a notificação de aceitação de proposta ao utilizador (acceptProposal()).

**7.** É criada uma transação (createTransaction()).

**8.** O objeto “TransacaoVenda” já não é mais usado a partir deste ponto.

* + - **Se a proposta for recusada (denyProposal()):**

**9.** É enviada a notificação de recusa de proposta ao utilizador (denyProposal()).

**10.** O objeto “PropostaVenda” já não é mais usado a partir deste ponto.

* **Se o utilizador não fizer uma proposta:**

**11.** O utilizador cria uma transação (createTransaction()).

**12.** O objeto “TransacaoVenda” já não é mais usado a partir deste ponto.

## 8.2. Diagrama de sequência de empréstimo

**[Ver anexo 7](#_Anexos)**

Este diagrama de sequência tem uma estrutura bastante semelhante ao diagrama de sequência da venda, envolvendo quatro elementos cruciais: utilizador, venda/empréstimo, proposta e transação.Uma imagem com texto, Paralelo, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.

Figura 14 – Diagrama de sequência - Empréstimo

**1.** O utilizador inicia a interação ao pedir e ao ver os empréstimos disponíveis (getAvailableLoans()).

**2.** O sistema responde com a lista de empréstimos disponíveis (availableLoans).

**3.** O utilizador seleciona um empréstimo específica (getLoanById()).

**4.** O sistema retorna o empréstimo escolhido (chosenLoan).

A partir deste ponto, o fluxo divide-se em diferentes possibilidades:

* **Se o utilizador faz uma proposta:**

**5.** A proposta é criada (createPropostal())

* + - **Se o vendedor aceita a proposta (acceptProposal()):**

**6.** É enviada a notificação de aceitação de proposta ao utilizador (acceptProposal()).

**7.** É criada uma transação (createTransaction()).

**8.** O objeto “TransacaoEmprestimo” já não é mais usado a partir deste ponto.

* + - **Se a proposta for recusada (denyProposal()):**

**9.** É enviada a notificação de recusa de proposta ao utilizador (denyProposal()).

**10.** O objeto “PropostaVenda” já não é mais usado a partir deste ponto.

* **Se o utilizador não fizer uma proposta:**

**11.** O utilizador cria uma transação (createTransaction()).

**12.** O objeto “TransacaoEmprestimo” já não é mais usado a partir deste ponto.

## 8.3. Diagrama de sequência de sorteio

Este diagrama de sequência representa o fluxo de um processo de sorteio de doação, detalhando a interação entre o utilizador, o sistema de sorteio, inscrição no sorteio e vencedor do sorteio. Em baixo encontra-se uma descrição mais detalhada.

**[Ver anexo Uma imagem com texto, Paralelo, captura de ecrã, file

Os conteúdos gerados por IA poderão estar incorretos.8](#_Anexos)**

Figura 15 – Diagrama de sequência - Sorteio

**1.** O utilizador inicia a interação ao pedir e ao ver os sorteios abertos (getAvailableDraws()).

**2.** O sistema responde com a lista de sorteios abertos (openDraws).

**3.** O utilizador seleciona um sorteio específico (getDrawById()).

**4.** O sistema retorna o sorteio escolhido (chosenDraw).

**5.** O utilizador inscreve-se nesse sorteio (createRegistration()).

**6.** Quando pelo menos um utilizador se inscreve no sorteio o objeto “VencedorSorteio” é criado uma única vez.

**7.** No fim do sorteio é sorteado o vencedor (drawWinner()).

A partir deste ponto, o fluxo divide-se em diferentes possibilidades:

* **Se o utilizador ganhou o sorteio:**

**8.** É enviada a notificação ao utilizador a informar sobre a vitória (wonDraw()).

**9.** O objeto “VencedorSorteio” já não é mais usado a partir deste ponto.

* **Se o utilizador não ganhou o sorteio:**

**10.** É enviada a notificação ao utilizador ao informar sobre a derrota. (loseDraw()).

**11.** O objeto “VencedorSorteio” já não é mais usado a partir deste ponto.

# *9.* *Mockup’s*

*Mockup’s* são representações visuais, usadas para demonstrar o *design* antes da produção, ajudando a mostrar o aspeto final sem interatividade. Permitem visualizar ideias, testar conceitos e fazer ajustes antes da versão final. São úteis para apresentar projetos a clientes e equipas.

Uma imagem com texto, captura de ecrã, Tipo de letra, design

Os conteúdos gerados por IA poderão estar incorretos.Começando pela página de *login* permite a autenticação dos utilizadores, garantindo acesso seguro à aplicação. Inclui campos para email e palavra-passe. Também disponibiliza *links* para "Alterar Palavra-passe" e "Criar Conta" para facilitar a gestão de acesso.

Figura 16 - Página de Login

A página de “Criar conta” permite que novos utilizadores criem uma conta na aplicação. Inclui campos para introdução de dados essenciais, email, palavra-passe e confirmar palavra-passe, além de um botão para concluir o registo. Contém um *link* para a página de Login, caso o utilizador já tenha conta.

Uma imagem com texto, captura de ecrã, Tipo de letra, design

Os conteúdos gerados por IA poderão estar incorretos.

Figura 17 - Página Sign In

Uma imagem com texto, captura de ecrã, Tipo de letra, Retângulo

Os conteúdos gerados por IA poderão estar incorretos.A página de recuperação de palavra-passe permite que os utilizadores redefinam o seu acesso. Contém um campo para introduzir o e-mail associado à conta e um botão para enviar o pedido de recuperação. Após o envio, o utilizador recebe um e-mail com o URL para a próxima página.

Figura 18 - Página de Alterar-Passe

Nesta página, o utilizador pode definir uma nova palavra-passe após receber um *link* de recuperação por email. Inclui campos para introduzir e confirmar a nova palavra-passe, garantindo segurança no processo. Após a submissão, a palavra-passe é atualizada e o utilizador pode aceder novamente à conta.

Uma imagem com texto, captura de ecrã, Tipo de letra, Retângulo

Os conteúdos gerados por IA poderão estar incorretos.

Figura 19 - Página de Alterar-Passe

 A página inicial apresenta os artigos disponíveis para venda, empréstimo e sorteios. No topo, há uma barra de pesquisa para facilitar a pesquisa de artigos. No menu lateral, encontram-se atalhos para a página inicial, favoritos e conta do utilizador, além de um botão para adicionar novos artigos na aplicação.

Figura 20 - Ecrã Inicial

A página de vendas mostra todos os artigos para venda que existem na aplicação. Isto acontece também para os empréstimos e sorteios. Na barra lateral, há atalhos para categorias, permitindo uma navegação rápida e eficiente.



Figura 21 - Página de Vendas

Na página de detalhes do artigo, são apresentados o título, descrição, valor e condição do artigo, acompanhados de imagens para uma visualização completa. Também são exibidos o número de contato para comunicação direta e dois botões: "Fazer Proposta" e "Comprar Agora", permitindo ao utilizador interagir conforme sua intenção de negociação.

Se o artigo for um empréstimo, a data de início e término do empréstimo são claramente indicadas. Já no caso de sorteio, os botões de proposta e aceitação são substituídos por um botão "Inscrever no Sorteio", e o número de contacto permanece disponível para dúvidas ou esclarecimentos.

Uma imagem com texto, captura de ecrã, Website

Os conteúdos gerados por IA poderão estar incorretos.

Figura 22 - Detalhes venda

Uma imagem com texto, captura de ecrã, Website, Página web

Os conteúdos gerados por IA poderão estar incorretos.Uma imagem com texto, captura de ecrã, design, ferramenta

Os conteúdos gerados por IA poderão estar incorretos.Figura 23 - Detalhes empréstimo

Figura 24 - Detalhes sorteio

Uma imagem com texto, captura de ecrã, Website, Página web

Os conteúdos gerados por IA poderão estar incorretos.Ao clicar no botão "Fazer Proposta", um pop-up é exibido, permitindo ao utilizador inserir o valor que deseja propor para o artigo. O pop-up contém um campo de entrada para o valor e um botão "Enviar" para confirmar a proposta.

Figura - Proposta venda

Uma imagem com texto, captura de ecrã, software, Página web

Os conteúdos gerados por IA poderão estar incorretos.Ao clicar no botão "Fazer Proposta" para um empréstimo, um pop-up será exibido, permitindo ao utilizador inserir o valor desejado, bem como as horas de início e término do empréstimo. O utilizador poderá definir esses horários em campos específicos. Um botão "Enviar" estará disponível para confirmar a proposta.

Uma imagem com texto, captura de ecrã, software, Website

Os conteúdos gerados por IA poderão estar incorretos.

Figura - Proposta empréstimo

Uma imagem com captura de ecrã, texto, software, Ícone de computador

Os conteúdos gerados por IA poderão estar incorretos.A Página de favoritos mostra os artigos guardados pelo utilizador. A interface permite uma navegação simples, com a possibilidade de acessar rapidamente os artigos desejados. Caso o utilizador não tenha artigos marcados como favorito, um texto centralizado no ecrã informa: "Sem favoritos".

Figura 27 - Página favoritosUma imagem com captura de ecrã, texto, software

Os conteúdos gerados por IA poderão estar incorretos.

Figura 28 – Página sem favoritos

Para adicionar um artigo à venda, o utilizador preenche informações como título, descrição, valor, condição e categoria do artigo, além de carregar as imagens correspondentes. Um botão "Publicar" estará disponível para concluir a inserção do artigo.

No caso de um empréstimo, além dos campos mencionados para a venda, o utilizador também deve informar a data de início e término do empréstimo. Para sorteio, o processo é semelhante ao de empréstimo, mas sem o campo valor.

O utilizador apenas insere o título, descrição, imagens, categoria e as datas de início e término do sorteio, podendo publicar o artigo diretamente com o botão “Publicar".

Uma imagem com texto, captura de ecrã, Website, design

Os conteúdos gerados por IA poderão estar incorretos.

Figura 29 - Adicionar venda

Uma imagem com texto, captura de ecrã, design

Os conteúdos gerados por IA poderão estar incorretos.

Figura 30 - Adicionar empréstimo

Uma imagem com texto, captura de ecrã, Website, Página web

Os conteúdos gerados por IA poderão estar incorretos.

Figura 31 - Adicionar sorteio

Na secção da conta, o utilizador pode visualizar todos os artigos publicados, ordenados por tipo de transação (venda, empréstimo ou sorteio). Caso o utilizador não tenha artigos publicados, será exibida uma mensagem centralizada a dizer "Publique algo!" como incentivo para adicionar novos artigos.

Além disso, o painel oferece botões para "Editar Perfil", onde o utilizador pode atualizar as suas informações pessoais, "Ver Propostas", para aceder às ofertas recebidas, e “*Logout”*, para terminar a sessão.

Também é possível visualizar a avaliação como vendedor, com uma classificação baseada nas interações e avaliações de outros utilizadores.

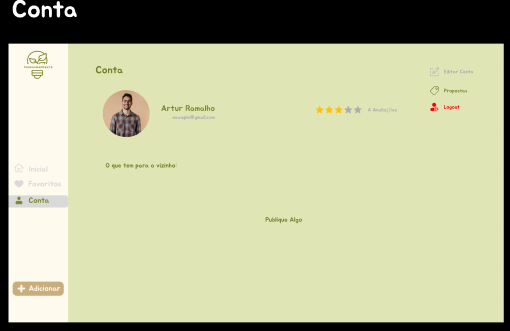


Figura 32 - Página da conta

Uma imagem com texto, captura de ecrã, multimédia, software

Os conteúdos gerados por IA poderão estar incorretos.

Figura 33 - Página da conta 2

O utilizador pode alterar a imagem de perfil, nome, telefone e morada. Após as alterações, o utilizador pode clicar no botão "Guardar" para salvar as modificações ou no botão "Voltar" caso deseje cancelar as alterações e regressar à página anterior.

Uma imagem com texto, captura de ecrã, software, Website

Os conteúdos gerados por IA poderão estar incorretos.

Figura 34 - Editar conta

Ao clicar no botão "Ver Propostas", será exibido um *pop-up* onde o utilizador pode visualizar todas as propostas recebidas. Cada proposta terá a opção de ser aceite ou rejeitada, permitindo uma resposta rápida às ofertas. Além disso, o *pop-up* também apresenta as propostas feitas pelo utilizador, indicando se foram aceites ou não.

Uma imagem com texto, captura de ecrã, software, Página web

Os conteúdos gerados por IA poderão estar incorretos.

Figura 35 - Proposta

Ao editar um artigo, o utilizador pode modificar todas as informações relacionadas com a venda, incluindo título, descrição, valor, condição, categoria e imagens. Além disso, existe uma opção para definir o estado do artigo, podendo escolher entre "Visível", "Vendido" ou "Não Visível", permitindo controlar a visibilidade do anúncio. Para os artigos de empréstimo, o utilizador pode alterar os dados relacionados com o empréstimo, como datas de início e término. Já os artigos de sorteio não têm a opção de definir o estado, mas permitem editar o título, descrição, categoria, imagens e datas de início e término do sorteio. Em todos os casos, existe a opção de "Apagar Artigo", caso o utilizador queira remover o anúncio da aplicação.

Uma imagem com texto, captura de ecrã, software, Website

Os conteúdos gerados por IA poderão estar incorretos.

Figura 36 - Editar venda

Uma imagem com texto, captura de ecrã, Website, software

Os conteúdos gerados por IA poderão estar incorretos.

Figura 37 - Editar empréstimo

Uma imagem com texto, captura de ecrã, Website, software

Os conteúdos gerados por IA poderão estar incorretos.

Figura 38 - Editar sorteio

Uma imagem com texto, captura de ecrã, software, Página web

Os conteúdos gerados por IA poderão estar incorretos.Após a conclusão de uma transação, seja venda ou empréstimo, o utilizador tem a opção de avaliar o vendedor. Um pop-up é exibido, permitindo ao comprador inserir o melai do comprador que fez o empréstimo ou a venda atribuir uma classificação.

Uma imagem com texto, captura de ecrã, Sistema operativo, software

Os conteúdos gerados por IA poderão estar incorretos.Figura 39 - Avaliação 1

Figura 40 - Avaliação 2

Todos os artigos publicados, sejam de venda, empréstimo ou sorteio, passam primeiro por uma revisão. O administrador pode visualizar todos os detalhes do artigo, incluindo título, descrição, valor, condição, categoria, imagens e qualquer outra informação relevante.

Após a revisão, o administrador tem a opção de aprovar ou reprovar o artigo. Caso o artigo seja aprovado, ele ficará visível na plataforma para os utilizadores.

Se for reprovado, o administrador pode fornecer uma razão ou instruções para que o vendedor faça as correções necessárias. Este processo de aprovação garante que todos os artigos publicados na plataforma cumpram com as políticas e normas estabelecidas, mantendo a qualidade e a conformidade do conteúdo.

Uma imagem com texto, captura de ecrã, software, multimédia

Os conteúdos gerados por IA poderão estar incorretos.

Figura 41 - Página de administrador

Uma imagem com texto, captura de ecrã, Website, Página web

Os conteúdos gerados por IA poderão estar incorretos.

Figura 43 - Página de administrador 2

# 10. Melhorias futuras

No futuro, pretendemos implementar novas funcionalidades que tornem o sistema mais eficiente, intuitivo e adaptável às necessidades dos utilizadores. Entre as possíveis melhorias, destacamos a otimização do desempenho e a adaptação do sistema a novas tecnologias.

Além disso, será essencial realizar revisões contínuas para identificar e corrigir eventuais falhas, garantindo maior segurança e usabilidade. Com essas melhorias, buscamos assegurar que o sistema continue a evoluir e a atender às expectativas dos utilizadores.

# 11. Conclusão

Com este trabalho, esperamos expandir o nosso conhecimento ao explorar novos conceitos e tecnologias relevantes para o desenvolvimento do sistema. Esperamos que este aprendizado contribua significativamente para a nossa formação académica e profissional, preparando-nos para desafios futuros na área da tecnologia da informação.

Além disso, ao trabalhar com novas tecnologias, teremos a oportunidade de aprimorar a nossa capacidade de resolver problemas, adaptar-nos a diferentes cenários e aplicar os conhecimentos adquiridos de forma prática. Esse processo ajudará a desenvolver um pensamento crítico e analítico, essencial para a criação de soluções inovadoras e eficientes. Com isso, estaremos mais preparados para enfrentar os desafios do mercado de trabalho e acompanhar a constante evolução do setor tecnológico.

# 12. Anexos

* **[Ane](Anexo1.jpg)****[xo 1](Anexo1.jpg)**
* **[Anexo 2](Anexo2.jpg)**
* **[Anexo 3](Anexo3.jpg)**
* **[Anexo](Anexo4.jpg)** **[4](Anexo4.jpg)**
* **[Anexo 5](Anexo5.jpg)**
* **[Anexo 6](Anexo6.jpg)**
* **[Anexo 7](Anexo7.jpg)**
* [**Anexo 8**](Anexo8.jpg)

# 13. Webgrafia

big264903. (26 de 02 de 2025). *Diagrama de Classes*. Obtido de Brainly: https://brainly.com.br/tarefa/59811172

Carlos, W. (04 de 03 de 2025). *Diagramas de Sequência*. Obtido de PasseiDireto: https://www.passeidireto.com/pergunta/150311173/os-diagramas-de-sequencia-na-uml-sao-usados-para-representar-interacoes-entre-ob

Habbema, H. (15 de 02 de 2025). *Node.JS: Módulo Express*. Obtido de Medium: https://medium.com/@habbema/node-js-m%C3%B3dulo-express-aeaea83c853c

*O que são especificações de sistema?* (25 de 02 de 2025). Obtido de tiencontreinaweb: https://tiencontreinaweb.com.br/blog/glossario/o-que-e-especificacoes-de-sistema/

Pires, R. (20 de 02 de 2025). *Método MoSCoW*. Obtido de RockContent: https://rockcontent.com/br/blog/metodo-moscow/

Printi, E. (27 de 02 de 2025). *O que é mockup?* Obtido de Printi: https://www.printi.com.br/blog/o-que-e-mockup

*Sete etapas para começar a usar o Jira*. (14 de 02 de 2025). Obtido de Atlassian: https://www.atlassian.com/br/software/jira/guides/getting-started/basics#step-1-create-a-project

Team, A. (23 de 02 de 2025). *User Stories in Agile - How To Write With Examples*. Obtido de ActiveCollab: https://activecollab.com/blog/project-management/user-stories-in-agile

1. https://www.konstantinfo.com/blog/react-frameworks/ [↑](#footnote-ref-2)
2. https://www.npmjs.com/ [↑](#footnote-ref-3)