

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**

**ANA BRUNA BELO**

**COTTONNECT: UM SISTEMA PARA DOAÇÃO DE PRODUTOS COM  
GRATIFICAÇÃO VIA MOEDA VIRTUAL**

**GUARAPUAVA**

**2022**

**ANA BRUNA BELO**

**COTTONNECT: UM SISTEMA PARA DOAÇÃO DE PRODUTOS COM  
GRATIFICAÇÃO VIA MOEDA VIRTUAL**

**Cottonnect: A system for donation of products with gratification via virtual  
currency.**

Trabalho de Conclusão de Curso de Graduação  
apresentado como requisito para obtenção do  
título de Tecnólogo em Tecnologia em Sistemas  
para Internet do Curso Superior de Tecnologia  
em Sistemas para Internet da Universidade  
Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Roni Fabio Banaszewski

Coorientador: Prof. Dr. Diego Marczal

**GUARAPUAVA**

**2022**



[4.0 Internacional](https://creativecommons.org/licenses/by/4.0/)

Esta licença permite compartilhamento, remixe, adaptação e criação a partir do trabalho, mesmo para fins comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

**ANA BRUNA BELO**

**COTTONNECT: UM SISTEMA PARA DOAÇÃO DE PRODUTOS COM GRATIFICAÇÃO VIA  
MOEDA VIRTUAL**

Trabalho de Conclusão de Curso de Graduação  
apresentado como requisito para obtenção do título  
de Tecnólogo em Sistemas para Internet do Curso  
de Tecnologia em Sistemas para Internet da  
Universidade Tecnológica Federal do Paraná  
(UTFPR).

Data da aprovação: 30/junho/2022

---

Prof. Roni Fabio Banaszewski  
Doutor  
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Prof. Dênis Lucas Silva  
Mestre  
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

---

Prof. Andres Jessé Porfirio  
Doutor  
Universidade Tecnológica Federal do Paraná - Campus Guarapuava

GUARAPUAVA

2022

## **AGRADECIMENTOS**

Agradeço primeiramente a minha família, pelo incentivo e apoio em toda minha caminhada acadêmica.

Aos meus professores da Universidade Tecnológica Federal do Paraná, em especial ao meu orientador Prof. Dr. Roni Fabio Banaszewski, pela dedicação, auxílio e incentivo durante a realização deste trabalho.

Ao meu namorado Higor Silva Chaves, pelo demasiado incentivo, motivação e auxílio nos momentos mais difíceis.

Enfim, a todos os que por algum motivo contribuíram para a realização deste trabalho.



Há um prazer nas florestas desconhecidas; Um entusiasmo na costa solitária; Uma sociedade onde ninguém penetra; Pelo mar profundo e música em seu rugir; Amo não menos o homem, mas mais a natureza. (Lord Byron, 1826).

## RESUMO

Atualmente, as discussões sobre questões ambientais vem se tornando cada vez mais frequentes, proporcionando uma reflexão sobre os impactos originados das práticas sociais exercidas. Diante da crescente evidência do impacto causado ao meio ambiente, destaca-se o atual padrão de consumo, muitas vezes praticado inconscientemente, que acaba por contribuir para um maior acúmulo de lixo e mau uso da matéria-prima. Neste contexto, o presente projeto consiste no desenvolvimento de uma aplicação web que visa estimular doações de produtos com gratificações quantificadas por uma moeda virtual, que simboliza a gratidão ao receber uma doação. Desta forma, a intenção é estimular nos indivíduos de diferentes comunidades uma relação mais ampla de troca de produtos, fomentando a concepção do consumo consciente a fim de contribuir com a redução da degradação gradual do meio ambiente.

**Palavras-chave:** doações; consumismo; impacto ambiental; desenvolvimento web.

## ABSTRACT

Currently, discussions on environmental issues are becoming increasingly frequent, reflecting the impacts originated from the social practices exercised. Given the growing evidence of the impact caused to the environment, we highlight the current pattern of consumption, often practiced unconsciously, which ends up contributing to a greater accumulation of garbage and misuse of the raw material. In this context, this project consists of the development of a web application that aims to stimulate donations of products with bonuses quantified by a virtual currency, which symbolizes gratitude when receiving a donation. Thus, it is intended to stimulate in individuals from different communities a broader relationship of exchange of products, fostering the concept of conscious consumption to contribute to avoid the gradual degradation of the environment.

**Keywords:** donations; consumerism; environmental impact; web development.

## LISTA DE FIGURAS

Figura 1 – Captura de tela da opção de escolha para tipo de anúncio no <i>alguémquer?</i> .	18
Figura 2 – Captura de tela inicial do aplicativo <i>QUEM QUER</i> . . . . .	19
Figura 3 – Captura de tela inicial e de doação do <i>DOARFÁCIL</i> . . . . .	20
Figura 4 – Imagem da dinâmica do <i>Scrum</i> . . . . .	29
Figura 5 – Imagem do gráfico <i>Burndown</i> aplicado a uma <i>Sprint</i> . . . . .	30
Figura 6 – Imagem do ciclo do <i>TDD</i> . . . . .	31
Figura 7 – Captura de tela do quadro do projeto <i>Cottonnect</i> no <i>Trello</i> . . . . .	32
Figura 8 – Imagem do fluxo de trabalho do <i>Gitflow</i> . . . . .	33
Figura 9 – Imagem do fluxo de trabalho de um projeto utilizando CI/CD. . . . .	34
Figura 10 – Imagem da arquitetura da aplicação <i>Cottonnect</i> . . . . .	40
Figura 11 – Imagem da modelagem do banco de dados da aplicação <i>Cottonnect</i> . . . . .	42
Figura 12 – Imagem da modelagem final do banco de dados da aplicação <i>Cottonnect</i> . . . . .	43
Figura 13 – Imagem que apresenta a primeira etapa na criação da conta. . . . .	44
Figura 14 – Imagem que apresenta a segunda etapa na criação da conta. . . . .	45
Figura 15 – Imagem que apresenta a terceira etapa na criação da conta. . . . .	45
Figura 16 – Imagem que apresenta a tela de sucesso na criação de uma conta. . . . .	46
Figura 17 – Imagem da tela inicial que lista as doações. . . . .	46
Figura 18 – Imagem da tela do modal que permite dar um lance para uma doação. . . . .	47
Figura 19 – Imagem da tela onde é realizado o cadastro de uma doação. . . . .	48
Figura 20 – Imagem que apresenta todas as doações de um usuário. . . . .	49
Figura 21 – Imagem da tela do passo 01 para criação de conta. . . . .	51
Figura 22 – Imagem da tela do passo 02 para criação de conta. . . . .	51
Figura 23 – Imagem da tela do passo 03 para criação de conta. . . . .	52
Figura 24 – Imagem da tela do passo 03 para criação de conta. . . . .	52
Figura 25 – Imagem dos testes da API passados com sucesso. . . . .	53
Figura 26 – Imagem da criação de um leilão. . . . .	54
Figura 27 – Imagem dos detalhes de um leilão. . . . .	55
Figura 28 – Imagem da página que lista os leilões abertos. . . . .	56
Figura 29 – Imagem da página que lista os leilões do usuário logado. . . . .	56
Figura 30 – Imagem do gráfico de <i>burndown</i> da <i>sprint 06</i> . . . . .	57

Figura 31 – Imagem da página que lista os leilões que foram gratificados pelo usuário logado. . . . .	58
Figura 32 – Imagem que mostra a visualização do proprietário ou ganhador do leilão. . .	58
Figura 33 – Imagem que mostra os botões da ação de aceite ou rejeição da doação. . .	59
Figura 34 – Imagem da página de visualização/edição das informações pessoais do usuário logado. . . . .	59
Figura 35 – Imagem dos botões para ação do reenvio de códigos para confirmação do <i>e-mail</i> e número de celular. . . . .	60
Figura 36 – Imagem que permite mudar a senha para recuperação da conta . . . . .	61
Figura 37 – Imagem da nova tabela que salva o código de verificação . . . . .	62
Figura 38 – Imagem do <i>template</i> enviado quando um leilão é ganhado . . . . .	62
Figura 39 – Imagem do <i>template</i> enviado quando um leilão é ganhado . . . . .	63
Figura 40 – Imagem que demonstra detalhes e tutorias da aplicação . . . . .	63
Figura 41 – Resultado da primeira pergunta . . . . .	66
Figura 42 – Resultado da segunda pergunta . . . . .	66
Figura 43 – Resultado da terceira pergunta . . . . .	67
Figura 44 – Resultado da quarta pergunta . . . . .	67
Figura 45 – Resultado da quinta pergunta . . . . .	67
Figura 46 – Resultado da sexta pergunta . . . . .	68
Figura 47 – Resultado da sétima pergunta . . . . .	68
Figura 48 – Resultado da oitava pergunta . . . . .	68
Figura 49 – Resultado da nona pergunta . . . . .	69
Figura 50 – Resultado da décima pergunta . . . . .	69

## LISTA DE QUADROS

Quadro 1 – Comparativos entre os Sistemas. . . . .	21
Quadro 2 – Histórias de Usuário . . . . .	38
Quadro 3 – Novas Histórias de Usuário Mapeadas . . . . .	39
Quadro 4 – Histórias de Usuário Removidas . . . . .	39
Quadro 5 – Histórias de Usuário . . . . .	71

## LISTA DE ABREVIATURAS E SIGLAS

RSU	Resíduos Sólidos Urbanos
TDD	Test Driven Development (em português Desenvolvimento Orientado por Testes)
TCC	Trabalho de Conclusão de Curso
SQL	Structured Query Language (em português Linguagem de Consulta Estruturada)
IDE	Integrated Development Environment (em português Ambiente de Desenvolvimento Integrado)
API	Application Programming Interface (em português Interface de Programação de Aplicações)
PaaS	Platform as a Service (em português Plataforma como Serviço)
ORM	Object Relational Mapper (em português Mapeamento objeto-relacional)
HTTP	Hypertext Transfer Protocol (em português Protocolo de Transferência de Hipertexto)
URL	Uniform Resource Locator (em português Localizador Padrão de Recursos)
CSS	Cascading Style Sheets (em português Folhas de Estilo em Cascatas)
HTML	HyperText Markup Language (em português Linguagem de Marcação de Hipertexto)
UI	User Interface (em português Interface de Usuário)
JWT	JSON Web Token
ONG	Organização não governamental
JSON	JavaScript Object Notation (em português Notação de Objetos JavaScript)
REST	Representational State Transfer (em português Transferência Representacional de Estado)
ACID	Atomicidade, Consistência, Isolamento e Durabilidade
PWA	Progressive Web App (em português Aplicação Web Progressiva)

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>14</b>
<b>1.1</b>	<b>Objetivos</b>	<b>15</b>
1.1.1	Objetivo geral	15
1.1.2	Objetivos específicos	15
<b>1.2</b>	<b>Organização do trabalho</b>	<b>16</b>
<b>2</b>	<b>SISTEMAS SIMILARES</b>	<b>17</b>
2.0.1	<i>Alguémquer?</i>	17
2.0.2	<i>Quem Quer</i>	17
2.0.3	<i>DoarFácil</i>	19
2.0.4	Estudo Comparativo	19
<b>3</b>	<b>REFERENCIAL TEÓRICO</b>	<b>22</b>
<b>3.1</b>	<b>TECNOLOGIAS DO LADO CLIENTE</b>	<b>22</b>
3.1.1	<i>HTML, CSS e JavaScript</i>	22
3.1.2	<i>TypeScript</i>	23
3.1.3	<i>React.js</i>	24
3.1.4	<i>PWA</i>	24
<b>3.2</b>	<b>TECNOLOGIAS DO LADO SERVIDOR</b>	<b>25</b>
3.2.1	<i>Node.js</i>	25
3.2.2	<i>REST, JSON e JWT</i>	25
3.2.3	<i>ORM e TypeORM</i>	26
3.2.4	<i>Express.js</i>	26
3.2.5	<i>Twilio, SMS e Verify</i>	27
3.2.6	<i>Cloudinary</i>	27
3.2.7	<i>PostgreSQL</i>	27
<b>3.3</b>	<b>PROCESSO DE DESENVOLVIMENTO</b>	<b>28</b>
3.3.1	Metodologia de desenvolvimento ágil <i>Scrum</i>	28
3.3.2	Gráfico <i>Burndown</i>	29
3.3.3	<i>Kanban</i>	30
3.3.4	Desenvolvimento Orientado por Testes	31
<b>3.4</b>	<b>FERRAMENTAS DE DESENVOLVIMENTO</b>	<b>31</b>



3.4.1	<i>Trello</i> . . . . .	31
3.4.2	<i>Git, GitHub, Gitflow e GitHub Actions</i> . . . . .	32
3.4.3	<i>Heroku</i> . . . . .	33
3.4.4	<i>Jest</i> . . . . .	34
3.4.5	<i>ESLint</i> . . . . .	34
<b>3.5</b>	<b>MECANISMOS DE LEILÕES</b> . . . . .	<b>35</b>
<b>4</b>	<b>ANÁLISE E PROJETO DO SISTEMA</b> . . . . .	<b>36</b>
<b>4.1</b>	<b>ANÁLISE DO SISTEMA</b> . . . . .	<b>37</b>
<b>4.2</b>	<b>ARQUITETURA DO SISTEMA</b> . . . . .	<b>40</b>
<b>4.3</b>	<b>MODELAGEM E DESENVOLVIMENTOS DE PROTÓTIPOS</b> . . . . .	<b>41</b>
4.3.1	Modelagem do Banco de Dados . . . . .	41
4.3.2	Protótipos das Telas . . . . .	44
<b>5</b>	<b>DESENVOLVIMENTO DO SISTEMA</b> . . . . .	<b>50</b>
<b>5.1</b>	<b><i>SPRINT 01</i></b> . . . . .	<b>50</b>
<b>5.2</b>	<b><i>SPRINT 02</i></b> . . . . .	<b>50</b>
<b>5.3</b>	<b><i>SPRINT 03</i></b> . . . . .	<b>52</b>
<b>5.4</b>	<b><i>SPRINT 04</i></b> . . . . .	<b>53</b>
<b>5.5</b>	<b><i>SPRINT 05</i></b> . . . . .	<b>54</b>
<b>5.6</b>	<b><i>SPRINT 06</i></b> . . . . .	<b>55</b>
<b>5.7</b>	<b><i>SPRINT 07</i></b> . . . . .	<b>57</b>
<b>5.8</b>	<b><i>SPRINT 08</i></b> . . . . .	<b>58</b>
<b>5.9</b>	<b><i>SPRINT 09</i></b> . . . . .	<b>59</b>
<b>5.10</b>	<b><i>SPRINT 10</i></b> . . . . .	<b>60</b>
<b>5.11</b>	<b><i>SPRINT 11</i></b> . . . . .	<b>61</b>
<b>5.12</b>	<b><i>SPRINT 12</i></b> . . . . .	<b>62</b>
<b>5.13</b>	<b>CONSIDERAÇÕES SOBRE O DESENVOLVIMENTO</b> . . . . .	<b>64</b>
<b>6</b>	<b>RESULTADOS</b> . . . . .	<b>65</b>
<b>7</b>	<b>CONCLUSÃO</b> . . . . .	<b>70</b>
<b>7.1</b>	<b>TRABALHOS FUTUROS</b> . . . . .	<b>70</b>
	<b>REFERÊNCIAS</b> . . . . .	<b>72</b>

## 1 INTRODUÇÃO

Nos primórdios da civilização humana, a prática da troca direta de objetos e alimentos sem o uso da moeda foi o sistema comercial original e mais básico utilizado para negociações. Mais recentemente, durante o início da colonização portuguesa do Brasil, esta prática de troca de produtos ainda foi amplamente utilizada para negociação com os índios brasileiros. Basicamente, trocavam-se produtos excedentes ou desnecessários, por outros de maior interesse ou necessidade, sem a intervenção de instrumentos monetários. Esta prática ficou conhecida como “escambo” ou “permuta” e foi uma forma elementar de comércio utilizada por muito tempo (ROSSETTI, 2013, p. 171).

Em decorrência da necessidade de superar obstáculos encontrados no desenvolvimento do sistema de trocas diretas, como as necessidades em comum e a definição quantitativa em relação ao valor aplicado ao produto, deu-se início à prática do sistema de trocas indiretas, o qual envolve a empregabilidade de uma moeda para intermediar a aquisição de produtos. Desta forma, as relações comerciais foram facilitadas e as trocas, mesmo que de forma indireta, tornaram-se mais eficientes e fáceis de se realizarem (ROSSETTI, 2013, p. 171).

Contudo, com a evolução da indústria e comércio, ocorreu uma diversificação maior da oferta de produtos e também, um aumento da qualidade de vida e do poder aquisitivo da população. Neste sentido, as necessidades básicas como alimentação e vestuário, e de satisfação individual como joias e perfumes aumentaram de forma significativa, contribuindo para um consumismo exacerbado ao criar necessidades ou desejos de consumo.

O consumo inconsciente, gera um impacto negativo no meio ambiente. Segundo a WWF-Brasil (2019), o atual padrão de consumo está diretamente relacionado com o processo de degradação do planeta, sendo que cada vez mais consome-se uma quantidade maior de recursos que a natureza consegue oferecer. De fato, relatórios ambientais indicam que no modelo atual de consumo são extraídos mais recursos naturais do que a capacidade do planeta tem de regenerar. Em 29 de julho de 2019, foi considerada a data mais recente desde a primeira vez em que se ultrapassou a capacidade de renovação de recursos naturais (O Dia da Sobrecarga da Terra) (WWF-BRASIL, 2019).

O consumo promovido por meio de impulsos, sem que de fato haja necessidade de compra, torna o bem adquirido rapidamente descartável, contribuindo para o uso da matéria-prima de maneira ineficiente, onde é retirada do ambiente natural para retornar a ele em forma de resíduos. Segundo o relatório anual da ABRELPE (2020), entre os anos 2010 e 2019, a geração de resíduos sólidos urbanos (RSU) no Brasil registrou aumento, passando de 67 para 79 milhões de toneladas por ano. Estima-se que até 2050, o país terá um acréscimo de quase 50% na geração de RSU comparado ao ano de 2019.

O consumo inconsciente acarreta em um maior acúmulo de bens, onde rapidamente estes podem ser substituídos pelas tendências atuais do mercado. Esse fato contribui para uma maior produção de lixo, em que muitas vezes ocorre o descarte de itens que ainda poderiam

ser utilizados, juntamente com o descarte em locais inapropriados. Estima-se que no Brasil, a quantidade de resíduos que têm destinação inadequada aumentou cerca de 30% (ABRELPE, 2020).

A tomada de consciência destes fatos (consumo desnecessário e lixo), juntamente com a percepção de que o atual padrão de consumo está diretamente relacionado com a crise ambiental, tem contribuído para a construção de uma sociedade mais sustentável (Consumers International *et al.*, 2005, p. 17). De acordo com a pesquisa de perfil dos consumidores brasileiros, realizada por Nielsen (2019), 42% deles informaram que estão mudando seus hábitos de consumo para reduzir o impacto causado no meio ambiente.

Desta forma, com base nas questões ambientais abordadas, buscando soluções a estimular a concepção de consumo consciente e contribuir para redução do descarte de produtos, produção de lixo e mau uso da matéria-prima, em um planeta que dispõe recursos limitados, apresenta-se viável o desenvolvimento de um sistema, que facilite e estimule doações de objetos. No entanto, as doações efetuadas serão convertidas em moedas virtuais, denominadas *flocos de algodão*, que simbolizam o ato de gratidão ao receber uma doação. Os *flocos de algodão* são adquiridos por meio de doações ou através do saldo inicial quando uma nova conta é criada e tem como função exclusiva ser utilizado nos leilões, permitindo a aquisição de novos produtos no sistema, promovendo uma movimentação de produtos via trocas, porém de forma indireta.

Enfim, com o advento desta aplicação computacional, buscar-se-á estimular nos indivíduos de diferentes comunidades uma relação mais ampla de troca de produtos, fomentando a concepção do consumo consciente, de forma a contribuir no combate ao descarte desnecessário, ao consumo inconsciente e irresponsável e à degradação gradual do meio ambiente.

## 1.1 Objetivos

### 1.1.1 Objetivo geral

Desenvolver um sistema para estimular doações de objetos com gratificação via moedas virtuais.

### 1.1.2 Objetivos específicos

- Definir um mecanismo de leilão com lances baseados no ranking de moedas virtuais a fim de escolher o participante com maior prioridade em receber uma dada doação de interesse do mesmo;
- Implementar a funcionalidade de “reserva” de um item pelo tempo necessário para o participante vencedor ir analisar fisicamente e retirar o produto;

- Implementar operação de transferência das moedas da conta do recebedor para o doador quando a doação for efetuada;
- Obter informações de localização dos usuários para que as negociações ocorram em uma região próxima a fim de evitar gastos com fretes intermunicipais e principalmente para estimular o contato pessoal e análise do produto antes da aquisição.
- Definir e implementar medidas de segurança a fim de evitar fraudes na aquisição de moedas virtuais.

## **1.2 Organização do trabalho**

Este trabalho está organizado em sete capítulos, sendo estes: no Capítulo 2, Sistemas Similares, serão abordados os sistemas que possuem objetivo similar ao deste projeto; O Capítulo 3, Referencial teórico será apresentado a fundamentação teórica que embasa o trabalho, juntamente com as tecnologias que serão utilizadas no desenvolvimento e na gestão; O Capítulo 4, Análise e projeto do sistema, serão apresentadas as análises e considerações a respeito do projeto; O Capítulo 5, Desenvolvimento do Sistema será apresentado o processo de desenvolvimento do sistema de acordo com a metodologia de desenvolvimento escolhida; Por sua vez, o Capítulo 6, Resultados será apresentado os resultados obtidos nos testes aplicados para a solução proposta; Por fim, no Capítulo 7, Conclusão serão apresentadas as considerações finais acerca do tema e do projeto.

## 2 SISTEMAS SIMILARES

Atualmente, no mercado são encontradas ferramentas que têm como objetivo a doação direta entre bens de consumo, não havendo como comparação ferramentas que utilizam intermediação monetária como recompensa das doações, onde as moedas são trocadas por produtos dentro do próprio sistema. A seguir serão apresentados alguns sistemas disponíveis no mercado que possuem um propósito similar ao objetivo desse projeto.

Vale ressaltar que no mercado existem alternativas que possibilitam trocas e doações de produtos como Mercado Livre, Olx, Enjoei e *Facebook Marketplace*, porém estes não foram implementadas com o mesmo fim do presente projeto, pensando na retribuição por meio de moedas virtuais das doação para troca posterior por outro produto.

### 2.0.1 *Alguémquer?*

É um aplicativo, disponibilizado para *Android* e *iOS*, voltado a ser uma rede de doações de objetos, sem restrições de categorias. O aplicativo possibilita duas opções, estas, podendo ser doações ou pedidos de doações, conforme é ilustrado na Figura 1. As doações são criadas por meio da publicação de anúncios, permitindo aos usuários doarem ou receberem objetos de qualquer tipo de categoria, como livros, roupas, eletrônicos, brinquedos e inclui uma categoria que são itens com defeitos. Os anúncios possuem restrição de visualização para usuários que tem a localização cadastrada apenas mesma cidade em que o anuncio foi publicado.

Cada anúncio contém informações do item a ser doado, como título, descrição, categorias, fotos e localização por cidade na qual poderá ocorrer a retirada do objeto, juntamente com um contato do doador, que é o seu número de telefone. O aplicativo implementa filtros, que auxiliam os usuários a encontrarem anúncios de maior preferência, sendo possível a seleção de anúncios por país, estado e cidade, através da localização da conta do usuário, assim como também por categorias a qual os objetos doados pertencem.

O aplicativo possibilita a criação de grupos, deste modo, na publicação de um anúncio para um doação, pode-se definir se ela será publicada sem restrições de visibilidade de usuário ou restringida para determinados grupos que o usuário é membro.

### 2.0.2 *Quem Quer*

*Quem Quer* é um aplicativo para dispositivos móveis, disponibilizado para plataforma *Android* e *iOS* e tem como função possibilitar doações ou vendas de produtos e serviços. Para acesso ao aplicativo é necessário possuir uma conta, que pode ser criada rapidamente com as credencias do *Facebook* do usuário. Logado ao aplicativo, para realizar doações ou vendas é



**Figura 1 – Captura de tela da opção de escolha para tipo de anúncio no *alguémquer?***

**Fonte: A autora.**

necessário que o usuário confirme seu número de telefone através de um código enviado por SMS.

Com o sucesso na confirmação do número de telefone, o usuário tem disponível a funcionalidade da publicação das suas doações ou vendas, que podem ser criadas através de um anúncio. Os anúncios possuem informações do objeto a ser doado/vendido, como título, descrição, categorias, localização em que o objeto poderá ser retirado e uma data de início e fim que limita o período de visibilidade do anúncio.

Todos os anúncios, antes de serem publicados passam por uma análise de verificação, validando se estão de acordo com as políticas e regras da aplicação. Para comunicação das negociações com os anunciantes, o aplicativo possui bate-papo integrado. Ele também oferece o recurso para salvar os anúncios favoritos e permite que os usuários compartilhem anúncios que não são seus, ajudando no alcance da visibilidade dos objetos de outros usuários.

O aplicativo conta com um sistema de pontuação, em que os pontos são trocados por destaques nos anúncios, esses destaques aumentam o alcance de visibilidade dos objetos oferecidos. Os pontos podem ser adquiridos anunciando produtos, compartilhando o próprio anúncio, compartilhando o anúncio alheio ou finalizando uma negociação.

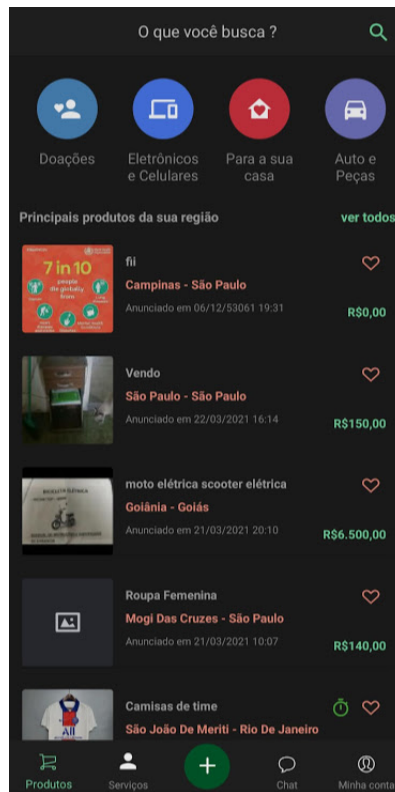


Figura 2 – Captura de tela inicial do aplicativo **QUEM QUER**.

Fonte: A autora.

### 2.0.3 DoarFácil

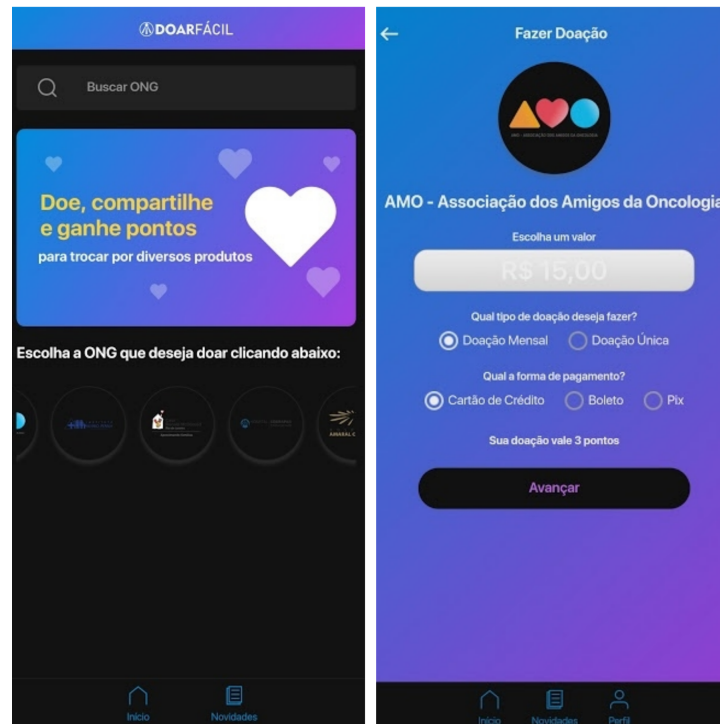
*DoarFácil* é uma aplicação que permite realizar doações únicas ou mensais em dinheiro para ONGs de todo o Brasil por meio de um aplicativo disponível para plataforma *Android* e *iOS* ou diretamente pelo site. Como recompensa, cada doação efetuada por um usuário é convertida em pontos que viram prêmios das instituições parceiras.

Para realizar uma doação é necessário estar logado ao sistema, e posterior, escolher uma ONG, conforme a Figura 3a. A etapa seguinte após a escolha da ONG é de fato realizar a doação, onde são inseridas as suas respectivas informações, como valor, método de pagamento e seu tipo, sendo possível doações mensais ou únicas. A Figura 3b, mostra a tela do aplicativo na qual é possível realizar a doação.

### 2.0.4 Estudo Comparativo

Com base na análise apresentada na Capítulo 2, os aplicativos *Alguémquer?* e *QUEM QUER* possuem como o intuito a doação de objetos entre indivíduos, já o aplicativo *DOARFÁCIL* foi apresentado para especificar a doação, neste caso por meio de dinheiro a ONGs, possuindo uma recompensa em forma de pontos, que posteriormente, podem ser trocados por prêmios.

O sistema proposto no desenvolvimento deste trabalho apresenta a união das características de doações de objetos com o fator de recompensa, tal fator é caracterizado pelas



(a) Tela Inicial

(b) Tela Doação

**Figura 3 – Captura de tela inicial e de doação do DOARFÁCIL.****Fonte: A autora.**

moedas virtuais. Para cada doação realizada, é estabelecido um valor monetário, o qual a pessoa recebedora da doação vai enviar ao doador como forma de agradecimento/gratidão. Conseqüentemente, as moedas recebidas poderão ser trocadas por outro objeto anunciado no sistema.

O Quadro 1 faz um comparativo entre os sistemas abordados, visando evidenciar as principais diferenças entre os referidos sistemas.



**Quadro 1 – Comparativos entre os Sistemas.**

	<b>alguémquer?</b>	<b>quemquer</b>	<b>doarfácil</b>	<b>cottonnect</b>
1. Doações de objetos	X	X		X
2. Doações em dinheiro a ONGs			X	
3. Recompensa por doações		X	X	X
4. Movimentação dos produtos através do sistema monetário do sistema e trocas indiretas de objetos por meio de moedas				X
5. Doações encontradas por meio de localização	X	X		X
6. Confirmação da conta por meio do número de telefone		X		X
7. Chat Integrado		X		
8. Mobile	X	X	X	

### 3 REFERENCIAL TEÓRICO

O desenvolvimento de sistemas é uma área multidisciplinar que necessita organização e escolhas adequadas das ferramentas para ser bem sucedida. Desta forma, fazer uso de um *framework* de gerenciamento de projetos como o *Scrum* e seus artefatos relacionados, que auxiliem na organização dos processos, assim como nas atividades a serem desenvolvidas, é essencial para se ter sucesso no desenvolvimento de sistemas. Outrossim, o uso de práticas ágeis pode auxiliar no atendimento de requisitos e nas entregas das funcionalidades implementadas no tempo mais próximo possível do esperado.

Quando refere-se ao desenvolvimento de sistemas para web, são inúmeras as ferramentas disponíveis no mercado, desde linguagens de programação, *frameworks*, e bibliotecas que auxiliam os programadores no desenvolvimento de suas aplicações. Tanto no lado do cliente como no lado do servidor, a escolha dessas tecnologias impacta diretamente no processo de desenvolvimento, na qualidade e no tempo de desenvolvimento. Portanto, definir quais tecnologias serão utilizadas é uma parte que possui grande importância no sucesso de uma aplicação.

Assim como a definição de tecnologias adequadas, é importante manter boas práticas de programação e fazer uso de técnicas que auxiliem na manutenção e na integridade da aplicação, como práticas de implementação de teste e *TDD*, além de validadores da qualidade do código, como o *ESLint*, *JSHint*, *Prettier*, *RuboCop*, entre outros.

Nesse capítulo, algumas características e particularidades de uso das tecnologias mencionadas acima que contribuirão para o desenvolvimento do projeto serão apresentadas.

#### 3.1 TECNOLOGIAS DO LADO CLIENTE

O lado cliente, ou *front-end*, pode ser classificado como a parte visual de uma aplicação. Por meio dela torna-se possível a iteração entre um usuário e o sistema. Nesta seção, serão abordadas algumas tecnologias de desenvolvimento de aplicações que são utilizadas no lado cliente.

##### 3.1.1 HTML, CSS e JavaScript

*HTML (HyperText Markup Language)* é uma linguagem para marcação de hipertexto. De maneira sucinta, pode-se definir hipertexto como todo o conteúdo inserido em um documento para a web e que tem como principal característica a possibilidade de se interligar a outros documentos (SILVA, 2011). Basicamente, o *HTML* é utilizado para a construção da estrutura das páginas web. Ele consiste em uma série de elementos que identificam cada parte da página, informando ao navegador como exibir o conteúdo, podendo corresponder a títulos, parágrafos, imagens, etc (W3SCHOOLS, 2021).

Como o *HTML* tem seu uso exclusivo na construção das estruturas que formam uma página na web, é necessário fazer uso de outra tecnologia que permite a *design* dessas páginas. Para este fim, é utilizado a linguagem de estilo *CSS*. *CSS* é a abreviação para o termo em inglês *Cascading Style Sheet*, traduzido para o português como folhas de estilo em cascata e cabe a ele todas as funções de estilo de um documento (SILVA, 2011). *CSS* é uma linguagem de estilo utilizada para definir a aparência de uma página, descrevendo como os elementos *HTML* devem ser exibidos em tela, incluindo o *design*, *layout*, cores e variações de exibição para diferentes dispositivos e tamanhos de tela.

Como os elementos *HTML* são construídos com conteúdo estáticos, ou seja, não possibilitam iteração com funcionalidades dinâmicas, é necessário fazer uso de uma linguagem que permita a criação de conteúdos que se atualizam de forma dinâmica, dando “vida” às aplicações que antes eram apenas estruturadas estática. O *JavaScript* é uma linguagem que pertence a essa tríade do desenvolvimento *front-end*, onde permite realizar funcionalidades dinâmicas na construção de páginas da web. *JavaScript* é uma linguagem de programação da Web, utilizada na ampla maioria dos sites, navegadores, computadores de mesa, consoles de jogos, *tablets* e *smartphones*, tornando-a a linguagem de programação mais onipresente da história. Pode ser utilizada no lado cliente, dando “movimento” as páginas, assim como pode ser utilizada no lado servidor, na construção de sistemas *back-end*, desde *APIs*, *webservices*, até servidores de aplicação. *JavaScript* é uma linguagem de alto nível, dinâmica, interpretada e não tipada, conveniente para estilos de programação orientados a objetos e funcionais (FLANAGAN, 2013).

### 3.1.2 *TypeScript*

*TypeScript* é uma linguagem de código aberto com base na linguagem *JavaScript*, oferecendo entre outras características, a possibilidade de definir tipos de dados. O código *TypeScript* é transformado em código *JavaScript* por meio do compilador *TypeScript*. Este código compilado pode ser executado em qualquer ambiente que o *JavaScript* possa ser executado (TYPESCRIPT, 2021).

A linguagem *TypeScript* pode ser considerada um superconjunto tipado de *JavaScript*, que é compilado para uma dada versão do *JavaScript*. Isto faz programas escritos em *TypeScript* altamente portáteis, pois podem ser executados em quase qualquer máquina. A verificação de tipo estático reduz erros causados pelo uso indevido acidental de tipos, assim como permite que as ferramentas de desenvolvimento forneçam mecanismos de autocompletar inteligente, aumentando o nível de produtividades dos programadores (FENTON, 2017).

### 3.1.3 *React.js*

*React.js* é frequentemente referido como *ReactJS* ou apenas *React*. O *React.js* é uma biblioteca *JavaScript* de código aberto, criada pela equipe do *Facebook* para construir interfaces de usuário (UI - *User Interface*). Seu desenvolvimento é baseado em componentes, isso permite a divisão das UI em partes independentes, reutilizáveis e isoladas, onde cada componente possui estados. Mesmo não sendo obrigatório, o *React* recomenda o uso do *JSX* para criar elementos para utilização nos *templates*. O *JSX* é uma extensão de sintaxe para *JavaScript*, em que pode lembrar uma linguagem de *template*, como *HTML*, mas vem com todo o poder do *JavaScript* (REACTJS, 2021). O *React* faz uso do paradigma de programação declarativa, que concentra-se em descrever o que deve-se fazer e não em como seus procedimentos funcionam.

Para manipulação da DOM (*Document Object Model*), que é a representações dos dados que compõem a estrutura e o conteúdo de uma pagina na *Web*, ele faz uso de um DOM virtual. Isso permite atualizar e renderizar apenas os componentes necessários na medida em que os dados mudam, permitindo um desempenho mais eficiente (HAMEDANI, 2021).

### 3.1.4 *PWA*

*PWA* (*Progressive Web App*) é um modelo de aplicação web que oferece tecnologias que permitem a experiência de uso muito próxima da oferecida pelos aplicativos móveis. A principal vantagem em utiliza-lo é a de não ser necessário a instalação da aplicação nos dispositivos, as aplicações são leves e sua construção é relativamente simples quando o sistema web segue os padrões requeridos para ser um *PWA*. As características de um *PWA* são: ser progressivo, responsivo, independente de conectividade, semelhante a aplicativos nativos, estar sempre atualizado, ser seguro devido ao *HTTPS*, descobrível, envolvente por utilizar recursos em que é possível engajar com o usuário, simular o estado de instalado ao inserir um ícone da aplicação nas áreas de aplicativos dos dispositivos e permitir acesso e compartilhamento através de links (ANTUNES, 2021).

Existem algumas tecnologias Web por trás do conceito de *PWA*, que são as seguintes: *Service Worker*, *Cache API* e *Web App Manifest*. Basicamente, o *Service Worker* é que torna a tecnologia de *PWA* possível. Ele intercepta as requisições da aplicação e guarda os resultados no lado do cliente, possibilitando um consumo dos dados extremamente veloz, e permitindo que a aplicação funcione offline. Por sua vez, o *Cache API* é uma forma de se armazenar as requisições feitas localmente e por fim, o *Web App Manifest* é um documento que tem como finalidade padronizar as aplicações Web (ANTUNES, 2021).

## 3.2 TECNOLOGIAS DO LADO SERVIDOR

O lado servidor, ou *back-end*, é responsável por receber requisições e realizar processamentos que estão diretamente ligadas as regras de negócios, como persistir dados, obter informações, realizar cálculos, entre outras. O lado servidor está diretamente interligado com o lado cliente, o qual requisita informações a ele e aguarda o processamento dessas requisições. Por sua vez, o lado servidor deverá devolver o resultado dessas informações para o cliente que as requisitou. Nesta seção, serão abordadas as tecnologias de desenvolvimento de aplicações que tem seu uso no lado servidor.

### 3.2.1 *Node.js*

*Node.js* é um ambiente de execução *JavaScript* assíncrono orientado a eventos. Ele foi construído com base na *engine V8*. O *engine V8* é um interpretador de código *JavaScript* desenvolvido pelo *Google* que executa a linguagem em navegadores. Desta forma, o uso do *Node.js* permite que a execução da linguagem *JavaScript* seja possível fora dos navegadores, tornando o uso da linguagem viável tanto no lado do cliente quanto no lado do servidor.

Sua execução é *single-thread*, diferente das outras tecnologias disponíveis no mercado, ou seja, no modelo *Node.js* apenas uma *thread* é responsável por tratar as requisições (NODE, 2021).

### 3.2.2 *REST, JSON e JWT*

*REST (Representational State Transfer)* é um conjunto de princípios de arquitetura de software para a criação de *web services*. O *REST* teve origem na tese de doutorado de Roy Fielding, e visa desempenho de interação, simplicidade, modificabilidade, portabilidade e confiabilidade (FIELDING, 2000). O protocolo *REST* é baseado nos conceitos do protocolo HTTP, que é o protocolo de camada de aplicação para transmissão de documentos hipermídia, como o *HTML*.

Basicamente, o *REST* é guiado pelo que seriam as boas práticas de uso do protocolo HTTP. Estas boas práticas caracterizam-se pelo uso adequado dos métodos HTTP (*GET, POST, PUT, DELETE, OPTIONS, HEAD, TRACE, CONNECT*), usando-os de maneira a respeitar o objetivo original para o qual realmente foram criados. Conforme as práticas do *REST*, as *URLs* escolhidas devem ser padronizadas dentro do sistema e devem ser únicas por recurso. Também, o uso de códigos de status devem ser padronizados para representação de sucessos ou falhas e, os cabeçalhos HTTP devem ser usados de forma adequada para atender o objetivo no qual foram realmente criados. Desta forma, esta arquitetura permite a criação de sistemas distribuídos, eficientes, confiáveis e escaláveis (SAUDATE, 2013).

Por sua vez, o JSON (*JavaScript Object Notation*) é um formato para armazenamento e transmissão de informações em texto, leve, fácil para desenvolvedores lerem e escreverem e fácil para as máquinas analisarem. É baseado em um subconjunto do padrão de linguagem de programação JavaScript (especificada na ECMA-262) e é um formato de texto completamente independente da linguagem. A representação do formato é composto por nome/valor, ou seja, para cada valor representado, atribui-se um nome que descreve o seu significado (JSON.ORG, 2021).

Por fim, o *JSON Web Token (JWT)* é um padrão aberto que define uma forma compacta e independente para transmitir informações com segurança entre aplicações. Na autenticação, quando o usuário efetua login, um *JSON Web Token* é retornado, e sempre que o usuário requisitar acesso à uma rota ou recurso protegido, o agente do usuário deve enviar esse *Token*. Desta forma, se o *Token* estiver correto, o recurso é liberado (JWT, 2021).

### 3.2.3 ORM e TypeORM

Com o utilização da orientação a objetos, surge uma técnica de programação denominada *ORM (Object Relational Mapping)*. O *ORM* consiste na persistência automatizada (e transparente) de dados que representa como objetos as tabelas de um banco de dados relacional, utilizando metadados que descrevem o mapeamento entre os objetos e banco de dados. A utilização de um *ORM* permite um aumento de produtividade, fazendo com que os desenvolvedores se concentrem nos problemas de negócios e facilitando a manutenção nos projetos, tornando-os mais compreensíveis e portáteis para outros sistemas de bancos de dados (BAUER; KING, 2005).

*TypeORM* é um *ORM* que pode ser executado em diversas plataformas e pode ser usado com as linguagens *TypeScript* e *JavaScript*. Seu objetivo é fornecer recursos adicionais que ajudem a desenvolver qualquer tipo de aplicativo que use bancos de dados, através da técnica de mapeamento objeto relacional (TYPEORM, 2021).

### 3.2.4 Express.js

*Express* é um *framework* web de roteamento e *middleware* para aplicações que usam *Node.js* e tem como a funcionalidade mínima fornecer uma série de chamadas de funções de *middleware*. As funções de *middleware* são funções que tem acesso ao objeto de solicitação (*request*), ao objeto de resposta (*response*) e a próxima função de *middleware* no ciclo solicitação-resposta do aplicativo. O *Express.js* permite a manipulação de roteamento, determinando como um aplicativo responde a uma solicitação do cliente, que é requisitada por meio de uma caminho e um método de solicitação HTTP específico (e.g. GET, POST, etc) (EXPRESS, 2021).

O *Express* vem com um manipulador de erros integrado, que cuida de qualquer erro que possa ser encontrado no aplicativo. Ele permite a definição de funções de *middleware* de manipulação personalizada de erros. Estas possuem quatro argumentos, a saber: *err*, *req*, *res*, *next*. Ele também permite realizar configurações comuns de aplicação web, como a porta a ser usada para conexão e a localização dos modelos que são usados para renderizar a resposta (EXPRESS, 2021).

### 3.2.5 Twilio, SMS e Verify

*Twilio* é uma plataforma unificada que fornece ferramentas de comunicação através de mensagens, voz, vídeo e *e-mail*. Possibilitando a comunicação por meio de serviços como o *SMS* e *WhatsApp*. (TWILIO, 2022a)

*SMS* (Serviço de Mensagens Curtas) é um serviço de telecomunicações de rede móvel que permite enviar e receber mensagens de texto através de números de celulares. (CONCEITO.DE., 2014).

Em conjunto com o envio de mensagens de texto por *SMS* o *Twilio* dispõe da ferramenta *Verify* que permite a validação de contas através do número de celular, enviando um *SMS* com um código para validação. Caso o usuário retorne com o código correto, o *Verify* irá registrar o número de telefone como aprovado. Dessa forma, ajudando a combater fraudes e protegendo as contas de usuários. (TWILIO, 2022b)

### 3.2.6 Cloudinary

*Cloudinary* é uma plataforma que fornece experiências de mídia, permitindo armazenamento, gerenciamento e transformação de imagens e vídeos.

Por meio do uso de ferramentas como inteligência artificial, automação e recursos avançados de processamento de imagem e vídeo, é possível eliminar processos manuais de mídia digital, como definição de tamanhos padrões, cortes, redirecionamentos etc. (CLOUDINARY, 2022)

### 3.2.7 PostgreSQL

*PostgreSQL* é um sistema gerenciador de banco de dados (SGBD) objeto relacional, gratuito e de código aberto com mais de 30 anos de desenvolvimento ativo, utilizado no armazenamento de informações (POSTGRESQL.ORG, 2021). Além de controlar o armazenamento de dados, um *SGBD* deve ser responsável por controlar o acesso ao banco de dados, gerenciando quem pode visualizar e alterar os dados. O *PostgreSQL* como sendo um (*SGBD*), oferece estas funções.

O *PostgreSQL* usa e estende a linguagem *SQL*, que é a linguagem de pesquisa declarativa padrão utilizada para os banco de dados relacionais. O *PostgreSQL* é executado em todos os principais sistemas operacionais e é compatível com o conjunto de propriedades de transação *ACID*, garantindo atomicidade, consistência, isolamento e durabilidade. Assim como os demais (*SGBDs*), ele oferece recursos necessários para realizar replicações em servidores. Ele também permite gerenciar várias conexões com o banco de dados ao mesmo tempo, por meio do recurso de *multithread* oferecidos pelos sistemas operacionais. Ainda, tem suporte nativo à *SSL*, que é um protocolo de segurança projetado para fornecer segurança nas comunicações em uma rede de computadores. E ainda vale mencionar que ele também suporta de forma eficiente e confiável grandes tamanhos de dados em suas tabelas. (MILANI, 2008).

### 3.3 PROCESSO DE DESENVOLVIMENTO

O processo de desenvolvimento de uma aplicação constitui todo o período em que a aplicação estará em desenvolvimento. Neste período é importante fazer uso de práticas e metodologias que auxiliam a gerenciar e organizar todo processo, principalmente em projetos que são desenvolvidos em equipes. Ao se trabalhar em equipes, há necessidade de uma maior organização para que todos os membros saibam exatamente o que é necessário fazer, e como resultado final, possam entregar produtos com qualidade e dentro dos prazos estipulados. Neste sentido, a corrente seção apresenta algumas práticas e metodologias que auxiliam no processo de gerenciamento, organização e desenvolvimento de uma aplicação.

#### 3.3.1 Metodologia de desenvolvimento ágil *Scrum*

O *Scrum* é um *framework* estrutural que vem sendo usado desde o início de 1990. Seu principal evento são as Sprints, elas podem ser determinadas por um espaço de tempo (de uma a duas semanas, geralmente) em que uma versão potencialmente utilizável do produto irá ser criada. A metodologia *Scrum* define também um *Time Scrum*, o qual é composto pelo *Product Owner*, um *Scrum Master* e o *Time de Desenvolvimento*. O *Product Owner* é o responsável por fornecer novos requisitos juntamente com a ordem que devem ser executados, além de avaliar a qualidade final das entregas. O *Scrum Master* é responsável por garantir que o *Scrum* seja entendido e aplicado. O *Time de Desenvolvimento* consiste nos profissionais que realizam o trabalho de entregar uma versão usável que incrementa o produto ao final de cada *Sprint*. A Figura 4 apresenta a dinâmica do processo do *Scrum*. Para fornecer transparência ao processo, assegurando que todas as informações sejam transmitidas às equipes e que todos tenham o mesmo entendimento, são aplicados alguns artefatos (SCHWABER; SUTHERLAND, 2013). Segundo Schwaber e Sutherland (2013), os artefatos do *Scrum* são:



- **Product Backlog:** é uma lista de tudo que deve ser necessário no produto, e é a única origem dos requisitos para qualquer mudança a ser feita no produto.
- **Sprint Backlog:** é um conjunto de itens do *Backlog* do Produto selecionados para uma dada *Sprint*, juntamente com o plano para entregar o incremento do produto e atingir o objetivo da *Sprint*.

Na composição da lista do *Product Backlog* é comum utilizar para descrever os itens que devem ser realizados as Histórias de Usuário. Uma História de Usuário descreve de forma natural e informal uma funcionalidade que será valiosa para um usuário de um sistema. Uma História pode utilizar um *template* que pode ser descrito da seguinte forma: COMO ator QUERO algo PARA um motivo (COHN, 2004), onde primeiramente define-se o usuário que executará tal funcionalidade, seguido do que ele quer realizar e qual o objetivo a atingir com a implementação. O uso das Histórias de Usuário facilita a criação de sentido e a comunicação no que deve ser desenvolvido, ajudando as equipes na compreensão do sistema.

Para auxiliar na gestão de tempo e divisão das *Sprints*, é possível definir pontos às Histórias de Usuário. Os pontos são unidades de medida utilizados para expressar uma estimativa do esforço necessário para implementar uma funcionalidade requerida. Desta forma, os desenvolvedores com base em complexidade e quantidade de trabalho, atribuem pontos às Histórias. Com a quantidade de pontos, pode-se definir quais e quantas Histórias podem fazer parte de cada *Sprint*.

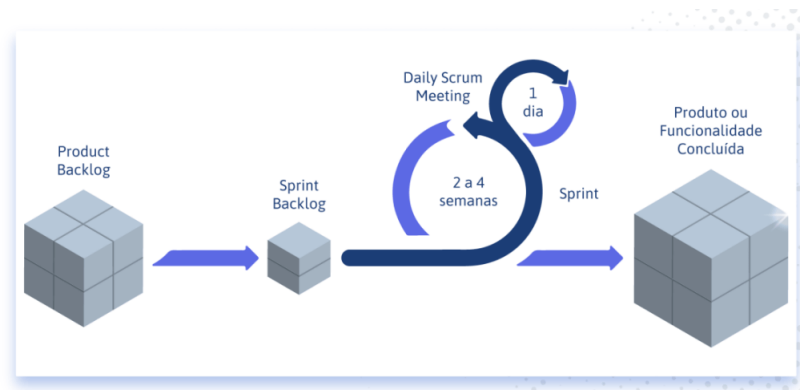


Figura 4 – Imagem da dinâmica do Scrum.

Fonte: (JUSTO, 2017)..

### 3.3.2 Gráfico *Burndown*

O gráfico de *Burndown* é uma ferramenta utilizada pelas equipes *Scrum* para apresentar o trabalho concluído por dia em relação ao valor de conclusão estipulado para uma *Sprint*. Ele é composto por dois eixos: Y, que vai representar o trabalho que precisa ser realizado, e X, que representa o tempo ou quantidade de trabalho, estipulada em dias ou horas, para concluir a demanda. Um exemplo do gráfico pode ser observado na Figura 5. Basicamente, o gráfico

inicialmente começa com um total de pontos planejados na *Sprint* e a medida que as tarefas são executadas, a pontuação no gráfico vai sendo “queimada”. No final do último dia de duração, espera-se que tenha um total de 0 pontos planejados, caracterizando que todas as tarefas definidas na *Sprint* foram realizadas. Desta forma, o gráfico irá ajudar a equipe a visualizar o progresso na conclusão de uma *Sprint*, sendo possível verificar se a equipe está adiantada, dentro do cronograma ou em atraso (CAMARGO, 2020).

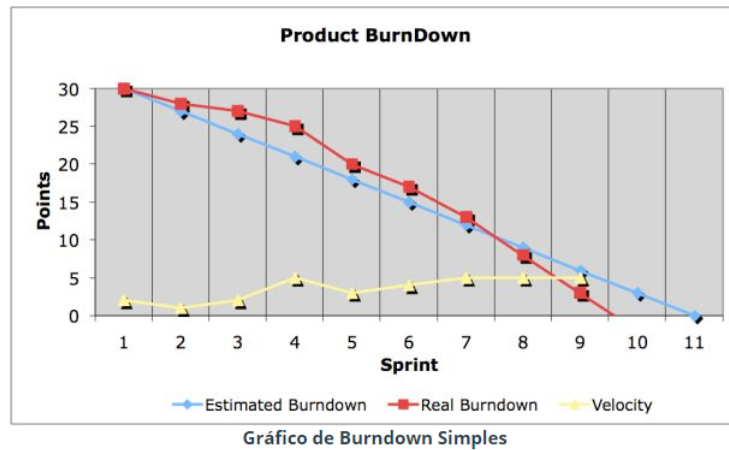


Figura 5 – Imagem do gráfico *Burndown* aplicado a uma *Sprint*.  
Fonte: (CAMARGO, 2020)..

### 3.3.3 Kanban

*Kanban* é um sistema ágil e visual para controle de produção ou gestão de tarefas. Ele possui 3 (três) principais componentes, os quais são denominados de: cartões, colunas e quadros. O cartão representa uma tarefa ou ação que precisa ser tomada para que o resultado final seja entregue. Por sua vez, as colunas representam os status dos cartões que geralmente são representados por três colunas: A Fazer, Em Execução e Feito. Porém, estas podem ser alteradas de acordo com a necessidade do projeto. Por fim, o quadro é o *Kanban* como um todo, organizado em colunas e cartões. Desta forma, a organização se dá pela movimentação dos cartões pelas colunas, indicando o seu atual estado no momento. Esses componentes podem ser visualizados na Figura 7. A união entre o modelo visual apresentado pelo *Kanban* e os métodos definidos pelo *Scrum* trazem vantagens as equipes, permitindo que as histórias de usuário sejam organizadas em cartões e as Sprints definidas em colunas, facilitando a visualização dos riscos do projeto. Assim, o *Product Owner* e o Time de Desenvolvimento podem visualizar possíveis gargalos que estão atrasando entregas e indicando a produtividade individual de cada membro, já que cada cartão entregue pode ter um responsável definido. (ARTIA, 2021).

### 3.3.4 Desenvolvimento Orientado por Testes

Do inglês *Test Driven Development*, ou simplesmente TDD, é uma prática na qual se escrevem os testes antes mesmo de escrever o código de produção. Ao escrever os testes, o desenvolvedor garante que boa parte do sistema tem um teste que assegura o seu funcionamento. Desta forma, consegue-se verificar a integridade do sistema quando novas funcionalidades são adicionadas. O mecanismo de testes consiste em escrever um teste que falha e em seguida o fazer passar da maneira mais simples possível e, por fim, refatorar o código (ANICHE, 2014). Esse ciclo é conhecido como Ciclo Vermelho-Verde-Refatora e pode ser observado na Figura 6.

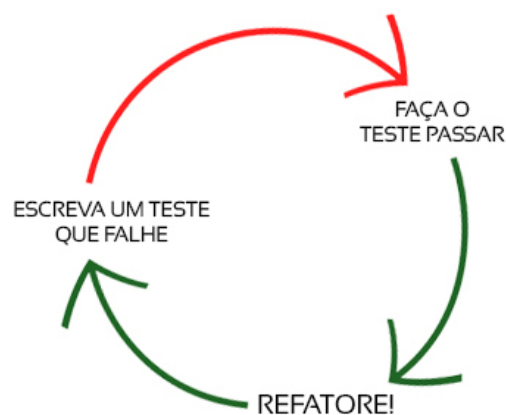


Figura 6 – Imagem do ciclo do TDD.

Fonte: (ANICHE, 2014)..

## 3.4 FERRAMENTAS DE DESENVOLVIMENTO

As ferramentas de desenvolvimento são requisitos crucias para auxiliar os desenvolvedores na construção de sistemas computacionais. Com o uso de ferramentas adequadas, pode-se aumentar a produtividade do desenvolvimento, minimizando a necessidade de efetuar práticas manuais, assim como permitir uma organização e qualidade no sistema à ser desenvolvido. Nesta seção, serão apresentadas algumas ferramentas que auxiliam na construção de aplicações, ajudando na manutenção, organização e melhores práticas de desenvolvimento.

### 3.4.1 Trello

O *Trello* é uma ferramenta de gerenciamento de projetos online. Cada projeto criado é denominado de quadro e pode ser compartilhado com vários usuários. Cada quadro possui colunas que contêm cartões. Os cartões são os elementos essenciais para a organização do trabalho em um quadro, utilizados para representar tarefas e histórias do usuário das metodolo-

gias ágeis. O *Trello* permite integrar ferramentas de trabalho, conectando *power-ups* que ajudam a atender a uma necessidade específica. Como exemplo, há o *power-up Burndown Chat*, que permite visualizar gráficos de *Burndown* para as *Sprints* e o *Card Priority by Screenful* que permite adicionar prioridades as Histórias de Usuário. Também há a possibilidade de uso de extensões, tal como a *Scrum for Trello*, que permite trabalhar com as práticas do *Scrum* e *Card Numbers for Trello* que permite adicionar números de identificação (*IDs*) nos cartões do quadro (TRELLO, 2021). A Figura 7 apresenta a tela da organização de um quadro.

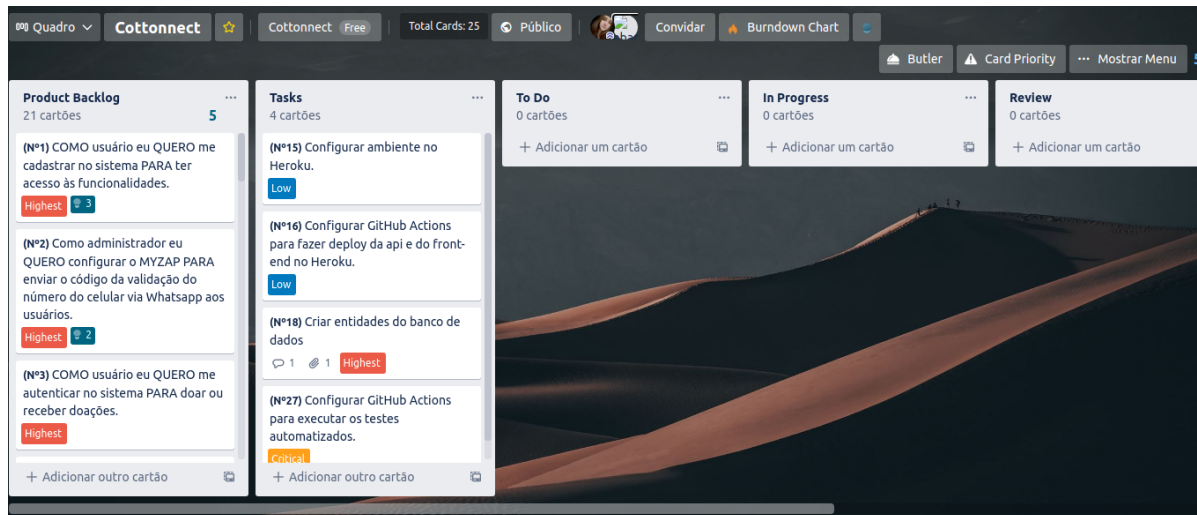


Figura 7 – Captura de tela do quadro do projeto *Cottonnect* no *Trello*.

Fonte: A autora..

### 3.4.2 *Git*, *GitHub*, *Gitflow* e *GitHub Actions*

*Git* é um sistema de controle de versão de arquivos, distribuído, gratuito e de código aberto. Ele foi projetado para hospedar desde projetos pequenos a projetos grandes mantendo a mesma velocidade e eficiência (GIT, 2021). Para gerenciar o fluxo de trabalho no *Git*, pode-se seguir um modelo de trabalho denominado *Gitflow*. O *Gitflow* é uma ideia abstrata do fluxo de trabalho do *Git*, em que define um modelo de ramificação rigoroso projetado com base no lançamento do projeto. Em termos técnicos, o *Gitflow* define um fluxo de trabalho para gerenciar as *branches*, que são ramificações do projeto nos repositórios.

Em vez da única *branch* principal, este fluxo de trabalho usa duas ramificações para registrar o histórico do projeto. A *branch* principal (chamada de main) armazena o histórico do lançamento oficial e a ramificação de desenvolvimento (chamada de develop) contem o código para a próxima versão de lançamento. Cada nova funcionalidade deve residir na própria ramificação, que deve ser criada a partir da *branch* de desenvolvimento. Quando uma funcionalidade é concluída, ela é mesclada na ramificação de desenvolvimento e quando a ramificação de desenvolvimento adquiriu recursos o suficiente para um lançamento, ela é bifurcada em uma ramificação de lançamento (chamada de release). Quando estiver pronta para ser lançada, essa

ramificação de lançamento é mesclada com o *branch* principal e de desenvolvimento. Para corrigir com rapidez problemas que ocorrem na versão que está em produção, são criadas as ramificações de manutenção ou de “*hotfix*”, em que são baseadas a partir da *branch* principal, uma vez que esta *branch* reflete o código em produção (ATLASSIAN, 2021). Pode-se visualizar o fluxo de trabalho do *Gitflow* na Figura 8

O *GitHub* é uma plataforma online de código aberto que permite a hospedagem de projetos gratuitamente com repositórios públicos e privados de forma ilimitada. Em conjunto, ele oferece uma série de funcionalidades que ajudam na revisão e gerenciamento dos projetos (GITHUB, 2021).

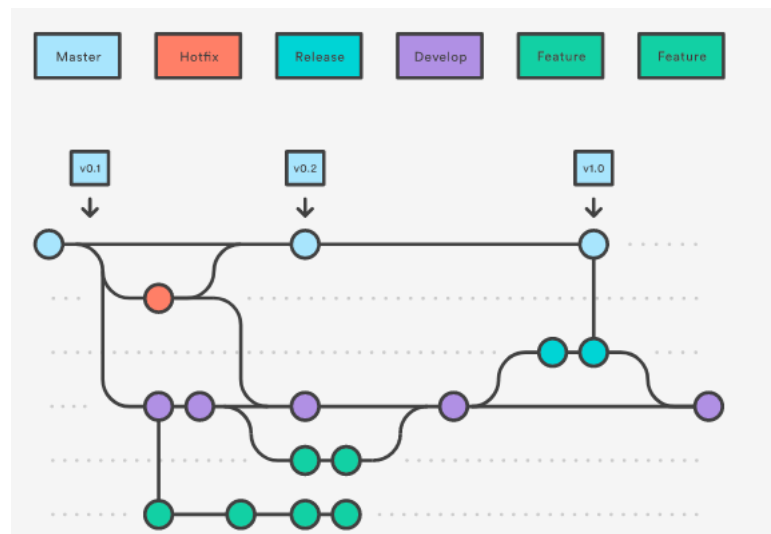


Figura 8 – Imagem do fluxo de trabalho do *Gitflow*.

Fonte: (ATLASSIAN, 2021)..

Por sua vez, o *GitHub Actions* é utilizado para automatizar, personalizar e executar fluxos de trabalho de desenvolvimento em integração contínua (CI) e implantação contínua (CD), diretamente em projetos que estão hospedados em repositórios do *GitHub*. O CI que usa *GitHub Actions* oferece fluxos de trabalho que podem ser configurados quando ocorre um evento no *GitHub* (por exemplo, quando um novo código é enviado para o repositório), desta forma ele pode executar os testes da aplicação (*GitHub Actions*, 2021). Já o CD, se refere ao lançamento automático das mudanças do repositório à produção, que podem ser configuradas através de uma mesclagem a *branch* principal, por exemplo. Na Figura 9 é possível visualizar um fluxo de trabalho utilizando os conceitos de CI/CD.

### 3.4.3 Heroku

*Heroku* é uma plataforma como serviço (PaaS) em nuvem baseada em contêiner, ou seja, oferece serviços de hospedagem de aplicações. É utilizado para implantar, gerenciar e dimensionar aplicativos. É flexível e de fácil utilização, oferecendo aos desenvolvedores o caminho mais simples para colocar os aplicativos em produção. *Heroku* é totalmente gerenciável,

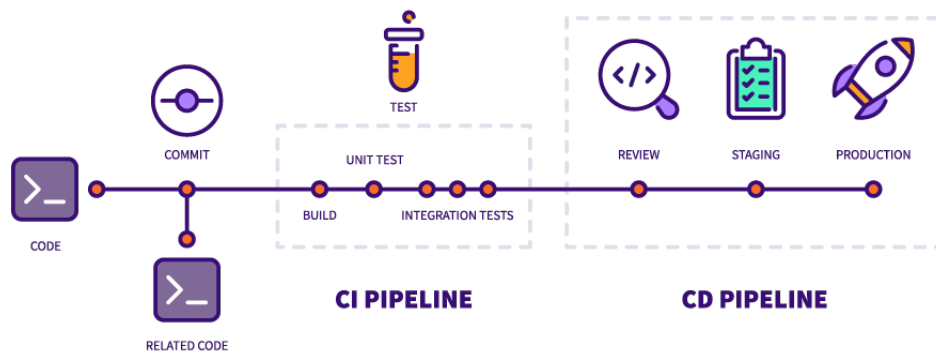


Figura 9 – Imagem do fluxo de trabalho de um projeto utilizando CI/CD.

Fonte: (GITLAB, 2021)..

dando aos desenvolvedores a liberdade de se concentrar em seu produto sem a distração de manter servidores, hardware ou infraestrutura. Ele também fornece serviços, ferramentas, fluxos de trabalho, projetados para aumentar a produtividade do desenvolvedor HEROKU (2021b).

O *Heroku* oferece o recurso *Heroku Developer Experience* que é uma abordagem para entrega de software em modo produção, para que os desenvolvedores possam se concentrar na criação e entrega contínua de aplicativos, sem se distrair com servidores ou infraestrutura. Os desenvolvedores implantam as aplicações diretamente de ferramentas populares como *Git*, *GitHub* ou sistemas de integração contínua (CI) HEROKU (2021a).

#### 3.4.4 Jest

*Jest* é um *Framework* de testes projetado para garantir a correção de qualquer código *Javascript* (JEST, 2021). Foi criado inicialmente para testar aplicações que foram construídas com *React*, porém sua implementação se tornou muito mais ampla, sendo atualmente utilizado também em aplicações que utilizam *Node.js* e *Redux*, assim como em plataformas que utilizam *TypeScript* como *Angular* e *Ionic*.

É caracterizado por ser de fácil utilização, tendo “zero configuração” na maioria dos projetos, possuindo agilidade, devido a execução paralela dos testes de forma confiável. Ele também possui a opção de gerar relatórios de cobertura de código e fornece um contexto rico do motivo dos testes terem falhados, como: *toBe*, *toBeCloseTo*, *toEqual*, *toStrictEqual*, *toHaveProperty*, *toMatchSnapshot* e *toThrowError* (JEST, 2021).

#### 3.4.5 ESLint

*ESLint* é um utilitário *linting* para a linguagem *JavaScript*, de código aberto que realiza análise estática de códigos, frequentemente utilizada para encontrar padrões problemáticos ou códigos que não obedecem a certas regras de estilo (ESLINT, 2021). A análise estática é o processo que tem como objetivo produzir uma lista de erros, advertências e pontos de melhoria

no código de um projeto. Assim, o *ESLint* permite que desenvolvedores descubram problemas ou apliquem melhorias que ajudam na legibilidade do código e estimulam boas práticas de programação. Ele permite que os desenvolvedores criem suas próprias regras de *linting* e é projetado para ter todas as regras completamente plugáveis. Ele também pode ser executado como parte de um *pipeline* de uma integração contínua (ESLINT, 2021).

### 3.5 MECANISMOS DE LEILÕES

Um leilão é um mecanismo que tem como objetivo promover a aquisição competitiva de produtos entre pessoas, que em um leilão são denominadas licitantes. Leilões são mecanismos dinâmicos e eficientes utilizados na comercialização de bens em mercados, principalmente quando não existe uma referência estável de preço. Os produtos são adquiridos por meio de um conjunto de regras pré-definidas conhecidas pelos licitantes antes do início do leilão e um critério de avaliação preciso especificado pelo leiloeiro, que é a pessoa responsável por mediar os eventos de leilão, oferecendo os bens que serão colocados à venda e recebendo as ofertas. Um leilão é um mecanismo transparente que deve se alcançar por meio de um resultado justo, reduzindo as oportunidades de corrupção (MAURER; BARROSO, 2011).

Nos últimos anos, houve um grande aumento no destaque e variedade de leilões na economia, permitindo a existência de vários tipos de leilão, sendo de múltiplos ou único itens, preços crescente ou decrescentes, lances abertos ou fechados, etc. Para Ausubel (2003), os principais tipos de leilão de um único item são os seguintes:

- **Leilões com lances fechados** É considerado um leilão estático, onde são enviados os lances sem saber os lances dos oponentes. Depois de acabar o prazo do envio, o leiloeiro abre os lances e define o ganhador. Podem ser considerados leilões de lances fechados o leilão de primeiro preço e o leilão de segundo preço. No leilão de primeiro preço, o maior lance ganha o item e paga exatamente o valor do seu lance. No leilão de segundo preço, o maior lance ganha o item, mas paga o valor do segundo maior lance.
- **Leilões dinâmicos de um único item** Os participantes fazem seus lances e com isso aprendem uma estratégia a partir dos lances de seus oponentes durante o leilão. Podem ser considerados leilões dinâmicos de um único item o leilão inglês e o leilão holandês. No leilão inglês, os licitantes submetem sucessivamente lances mais altos para o item, onde o licitante final ganha o item. No leilão holandês, o leiloeiro começa com um preço alto e sucessivamente anuncia preços mais baixos até alguém demonstrar interesse no item, ganha quem dar o primeiro lance no item.

## 4 ANÁLISE E PROJETO DO SISTEMA

Atualmente, a aplicação apresentada neste trabalho é caracterizada por ser uma aplicação web, mas futuramente, a intenção é transformá-la em uma aplicação em conformidade com as características de uma *PWA*. Basicamente, o sistema tem como objetivo permitir que usuários anunciem doações de objetos em leilões virtuais para que outros usuários possam dar lances com moedas virtuais para obter o respectivo objeto.

O tipo de leilão escolhido para uso na aplicação é o leilão de primeiro preço. A escolha deste modelo se deu pela quantificação do valor que o objeto possui para o usuário que demonstra interesse nele. Assim, segundo a análise do participante, ele irá definir o preço que julga mais justo para adquirir o produto e recompensar o doador sem a visualização dos lances dos adversários. Esta omissão evita influência na definição do valor do lance. Desta forma, os usuários participantes poderão dar um lance único por leilão, e ao final do mesmo, o vencedor será o que deu o maior lance. Em caso de empate, ganhará o participante que possuir mais moedas no sistema. A justificativa para uso desta estratégia é a de que o participante com mais moedas supostamente contribuiu com mais doações. Por fim, se novamente houver empate, o ganhador será o que deu o lance primeiro.

Para os lances com moedas virtuais, será utilizado um sistema monetário próprio, definido como flocos de algodão. Um floco de algodão simboliza o ato de agradecimento por uma doação, por ser leve, macio e remeter a sensações agradáveis. Basicamente, a função desta moeda virtual é intermediar as doações, ou seja, quando uma doação é efetuada, o doador receberá sua gratificação em flocos de algodão e este pode enviar lances para requisitar uma doação de outro item.

O nome da aplicação desenvolvida é *Cottonnect*. Este nome é originado da junção das palavras *Cotton* (Algodão) e *Connect* (conectar), trazendo o significado da aplicação ser uma comunidade de doações conectada pelos flocos de algodão, pois é por meio deles que as doações são possíveis.

Uma particularidade que vale salientar é a que envolve a criação da conta e saldo inicial de um usuário. Cada nova conta criada será iniciada com uma quantidade de flocos de algodão configurável pelo administrador do sistema, para que novos usuários comecem a interagir com a comunidade. Com o intuito de evitar fraudes na aquisição de flocos de algodão, as contas são criadas e integradas a um número de telefone, utilizando a *API Twilio*, descrita na subseção 3.2.5. Com isso, haverá certa dificuldade na criação de novas contas *fakes*, visto que, não se torna vantajoso a aquisição de novos números de telefone apenas para obter alguns flocos de algodão.

As características do sistema proposto são detalhadas nas subseções a seguir. Na seção 4.1, são apresentados os artefatos gerados na análise do sistema e na seção 4.2, é abordada a arquitetura da aplicação. Por fim, na seção 4.3 é apresentada a modelagem do banco de dados e a prototipação das principais telas.



## 4.1 ANÁLISE DO SISTEMA

Esta seção apresenta a análise necessária para que haja um melhor entendimento de como ocorreu o desenvolvimento do sistema, utilizando as tecnologias e conceitos abordados no Capítulo 3.

Para o planejamento e gestão no desenvolvimento do sistema, o projeto seguiu as práticas da metodologia ágil *Scrum*, definida na subseção 3.3.1. Os papéis de *Product Owner* e *Scrum Master* foram realizados pelo Professor Orientador e o time de desenvolvimento foi composto apenas pelo aluno autor deste projeto. As Histórias de Usuário<sup>1</sup> que compõem o *Backlog* são apresentadas no Quadro 2, que foram previamente levantadas pelo aluno em conversas com o professor orientador no papel de *Product Owner*. Desta forma, conseguiu-se a identificação das principais funcionalidades que o sistema propõe-se a fazer.

Para manter um padrão na definição das Histórias de Usuário, foi utilizado o *template* COMO...QUERO...PARA e definido números de identificação (*IDs*) para cada História de Usuário, utilizando a extensão *Card Numbers for Trello*, a fim de facilitar a identificação e referencição no texto. Também, foi adicionado pesos as Histórias com o *power-up* do *Trello* chamado *Card Priority by Screenful*, sendo definidas prioridades de 1 a 5, em que 5 tem o maior peso. A duração para a entrega de uma *Sprint* inicialmente foi definida com período de uma semana e cada respectivo ponto de uma História tem a duração de 1 dia, que representará 3 horas. Porém, ao decorrer das entregas o prazo foi aumentado passando de 1 para 2 semanas, afim de conseguir entregar as funcionalidades tanto da *API* como do *front-end* completas para que o *Product Owner* tivesse uma experiência melhor para conseguir validar as funcionalidades entregues. Mais precisamente, o aluno autor deste projeto teve a disponibilidade de trabalhar 3 horas por dia, que foram aplicadas nos 5 dias da semana, caracterizando 15 horas semanais de desenvolvimento.

Para a organização e gerenciamento das Histórias do *Backlog* foi utilizado o sistema *Trello*, definido na subseção 3.4.1, utilizando o recurso *Scrum for Trello* integrado juntamente com a visualização em modelo de *Kanban*, apresentado na subseção 3.3.3, onde as colunas definidas foram nomeadas como *Product Backlog*, *Tasks*, *To Do*, *In Progress*, *Review* e *Done* (por *Sprint*), e podem ser visualizadas na Figura 7, sendo:

- **Product Backlog:** são Histórias de Usuário que devem ser implementadas;
- **Tasks:** são tarefas com funcionalidades que não estão ligadas diretamente a uma requisição de usuário;
- **To Do:** são as Histórias da *Sprint* corrente;
- **In Progress:** são Histórias que estão sendo desenvolvidas no momento;
- **Review:** são Histórias que estão aguardando revisão para conclusão;

- **Done (por Sprint):** são Histórias que já foram revisadas e então concluídas na *Sprint* atual. Cada *Sprint* finalizada terá a sua própria coluna.

Mais precisamente, a movimentação dos *cards* é dada do *Product Backlog* para o *To Do* quando a data da História entrar na *Sprint* corrente. Quando a tarefa for iniciada, ela é movida para *In Progress*. Por sua vez, quando for finalizada, ela é movida para o *Review*. Por fim, se a tarefa for aprovada, ela é movida para a coluna *DONE* da sua respectiva *Sprint*.

Para monitoramento do andamento do projeto, a fim de identificar se o desenvolvimento ocorreu dentro do prazo estipulado, foi utilizado o gráfico de *Burndown*, descrito na subseção 3.3.2. Ele foi utilizado integrado ao *Trello*, que dispõe a ferramenta *Burndown for Trello* para acompanhamento do gráfico de maneira automatizada.

A primeira *Sprint* definida para o início do desenvolvimento do projeto compõe as História de Usuário de *IDs* 01 e 02, considerando que no início do desenvolvimento também foram destinadas algumas horas para instalação e configurações de ferramentas.

**Quadro 2 – Histórias de Usuário**

ID	Peso	Descrição da História
01	5	COMO usuário eu QUERO me cadastrar no sistema PARA ter acesso às funcionalidades
03	5	COMO usuário eu QUERO me autenticar no sistema PARA doar ou receber doações
04	5	COMO usuário eu QUERO cadastrar doações PARA doá-las
09	4	COMO usuário eu QUERO visualizar as doações ofertadas PARA encontrar algo para aquisição
11	4	COMO usuário eu QUERO dar lances com flocos de algodão em doações PARA ganhar uma doação
07	4	COMO usuário eu QUERO visualizar minhas doações PARA verificar quais doações possuo
12	3	COMO usuário eu QUERO saber quem é o ganhador da minha doação PARA concluí-la
21	3	COMO usuário eu QUERO marcar como recusada ou recebida a doação que ganhei PARA finalizar a doação
13	3	COMO usuário eu QUERO receber os flocos de algodão de uma doação concluída PARA utilizá-los na aquisição de outras doações
05	2	COMO usuário eu QUERO editar a quantidade de flocos de algodão e o prazo limite de término de uma doação PARA alterar a doação
19	2	COMO usuário eu QUERO receber uma notificação PARA que eu possa saber se ganhei ou perdi um leilão de uma doação
06	2	COMO usuário eu QUERO deletar doações PARA não mais ofertá-las para doação
10	2	COMO usuário eu QUERO filtrar as doações por categoria PARA visualizar doações de maior interesse
14	1	COMO usuário eu QUERO alterar os dados da minha conta PARA atualizar informações
20	1	COMO usuário eu QUERO visualizar as doações no qual efetuei lances PARA visualizar meus lances
22	1	COMO Administrador QUERO cadastrar novas categorias PARA adicionar novos tipos de categorias
23	1	COMO Administrador QUERO cadastrar novos status PARA adicionar novos tipos de privilégio
24	1	COMO Administrador QUERO cadastrar novos papéis de usuário PARA adicionar novos tipos de usuário.
25	1	COMO Administrador QUERO cadastrar novos usuários de maior privilegio PARA ajudar na manutenção da aplicação
26	1	COMO Administrador QUERO definir a quantidade de moedas virtuais iniciais das contas PARA que quando um usuário crie sua conta ele tenha disponível uma quantidade de moedas limitadas

Porém, além das histórias levantadas inicialmente, com o avanço das *Sprints* e maior entendimento sobre o domínio da aplicação novas histórias foram levantadas e outras foram

removidas juntamente com o *Product Owner*. Estas histórias estão apresentadas na Quadro 3 e Quadro 4, e serão melhor detalhadas no Capítulo 5 Desenvolvimento do Sistema.

Percebe-se que os *IDs* das tarefas não estão sequenciais, isso deve-se ao fato de ao decorrer do desenvolvimento do projeto julgou-se melhor separar as tarefas de *back-end* e *front-end*, então novas tarefas foram criadas, porém numa análise melhor e mais detalhada considerou-se que a primeira forma que havia sido organizado atendia melhor o que era proposto, e isso acarretou na geração de novas tarefas que foram removidas posteriormente, causando um espaçamento maior entre os números identificadores.

**Quadro 3 – Novas Histórias de Usuário Mapeadas**

ID	Peso	Descrição da História
44	2	COMO usuário QUERO ver os detalhes de um leilão PARA analisa-lo melhor
45	1	COMO usuário eu QUERO saber quem é o proprietário da minha doação que ganhei PARA entrar em contato
46	4	COMO usuário eu QUERO saber quantas pessoas deram lances em um leilão PARA me ajudar definir um valor
51	2	COMO usuário eu QUERO reabrir um leilão caso PARA casos onde o ganhador recuse a doação e não tenha outros lances
52	1	COMO sistema eu QUERO gerar um novo ganhador PARA casos em que o ganhador atual recuse a doação
53	3	COMO usuário eu QUERO recuperar minha conta PARA caso eu esqueça a senha
54	3	COMO usuário eu QUERO visualizar uma pagina com informações sobre a aplicação PARA entender como ela funciona
55	1	COMO sistema eu QUERO identificar o maior lance de um leilão em sua data de fechamento PARA definir o ganhador do leilão
56	1	COMO usuário eu QUERO confirmar meu e-mail PARA ter acesso as funcionalidades da aplicação
57	1	COMO usuário eu QUERO confirmar meu número de celular PARA ter acesso as funcionalidades da aplicação

**Quadro 4 – Histórias de Usuário Removidas**

ID	Descrição da História
02	COMO administrador eu QUERO configurar o MYZAP PARA enviar o código da validação do número do celular via Whatsapp aos usuários
08	COMO usuário eu QUERO informar minha geolocalização PARA visualizar doações próximas.

## 4.2 ARQUITETURA DO SISTEMA

A respeito do desenvolvimento do sistema, este foi dividido em dois projetos. A comunicação entre eles é definida pelo padrão cliente/servidor apresentado pelo *REST*, descrito na subseção 3.2.2 e pode ser analisado na Figura 10

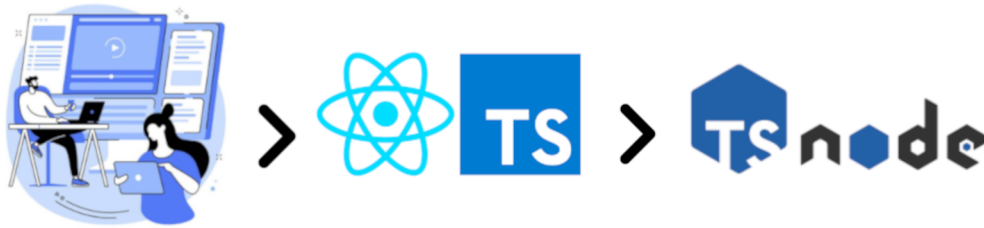


Figura 10 – Imagem da arquitetura da aplicação *Cottonnect*.

Fonte: A autora..

- **Servidor/API:** é o *back-end* do sistema. Esta camada é responsável, em termos gerais, por abrigar as implementações das regras de negócio. A *API* foi implementada no estilo de arquitetura *REST* e seguiu o padrão *JSON Web Token (JWT)* para a autenticação. Ela foi desenvolvida utilizando a biblioteca *Node.js* descrita na subseção 3.2.1, permitindo o uso do *TypeScript*, que foi a linguagem de desenvolvimento utilizada, citada na subseção 3.1.2. Para auxiliar no controle de rotas da aplicação, foi utilizado o *framework Express*, descrito na subseção 3.2.4. A fim de garantir integridade ao sistema, a *API* foi desenvolvida utilizando as práticas do *TDD*, definidos na subseção 3.3.4 a fim de realizar os teste unitários. Para a implementação dos testes foi utilizado a biblioteca *Jest* definida na subseção 3.4.4. Os testes foram integrados juntamente com a ferramenta *GitHub Actions*, que os executa de forma automatizada.
- **Cliente:** é o *front-end* da aplicação. Esta camada é responsável por implementar a interface gráfica para os usuários. Para o seu desenvolvimento foi utilizada a tecnologia *React.js*, definida na subseção 3.1.3, juntamente com a utilização da linguagem *TypeScript*. Basicamente, a aplicação *front-end* obtém os dados acessando a *API* remota da aplicação, sendo que para cada requisição é necessário o envio do *Token* para permissão de acesso e as informações são obtidas em formato *JSON*.

Ademais, os códigos dos projetos tiveram o seu controle de versionamento gerenciados via *GIT* e hospedados no *GITHUB* referenciados na subseção 3.4.2. As aplicações foram hospedadas no *Heroku*, citado na subseção 3.4.3 usando a automação de *deploy* automático com a ferramenta *GitHub Actions*. E a fim de garantir a qualidade do código escrito, foi utilizado o *linter ESLint* descrito na subseção 3.4.5, para a *API* e o *front-end*.

### 4.3 MODELAGEM E DESENVOLVIMENTOS DE PROTÓTIPOS

Nesta seção são apresentadas as modelagens e os protótipos gerados para a elaboração do projeto do sistema. A subseção 4.3.2 apresenta o protótipo das principais telas do sistema e as descrições de suas funcionalidades.

#### 4.3.1 Modelagem do Banco de Dados

O sistema gerenciador de banco de dados definido para o desenvolvimento da aplicação foi o *PostgreSQL*, descrito na subseção 3.2.7. Para facilitar sua utilização, foi empregado o uso da ferramenta *ORM TypeORM*, descrita na subseção 3.2.3.

Para representar o banco de dados foi criado o diagrama de modelagem<sup>2</sup> de dados com os atributos e relacionamentos das principais entidades que o sistema possui, como pode ser analisado na Figura 11. O modelo apresentado corresponde a um protótipo das tabelas que serão implementadas no sistema, com os relacionamentos, e as colunas com seus respectivos tipos.

No diagrama, pode-se visualizar a existência da tabela chamada *users*, que é responsável por conter os dados de acesso do usuário, como nome, informações adicionais, número de telefone, *e-mail*, senha, data de nascimento e o status da verificação do número de telefone. Também armazenará a quantidade de flocos de algodão que um usuário possui. Como o sistema possui dois tipo de usuários, foi definida a tabela *Roles* para indicar qual tipo de usuário a conta pertence. O usuário possui também a informação de sua localidade, sendo uma informação importante para a aplicação, pois é por meio dela que os usuários poderão encontrar doações, permitindo obter doações próximas a ele, essa localização é definida pela cidade e estado em que ele reside, sendo respectivamente armazenadas nas tabelas de *cities* e *states*.

A *auctions* é a tabela principal da aplicação, a qual representa os leilões. Os leilões possuem o dia do fechamento, o usuário que está doando o objeto, assim como o lance ganhador, definido após o fechamento do leilão. O principal artefato dessa tabela são as doações, representadas na tabela *donations-objects*. Os objetos possuem título, descrição, fotos e categorias. Como um objeto pode possuir múltiplas categorias foi criada a tabela *donation-category* que referencia quais categorias, armazenadas na tabela *categories*, o objeto de doação possui.

Para representar os lances do leilão foi definida a tabela *biddings*, que possui o valor do lance e em qual leilão ele será efetuado. Os leilões e as doações possuem um *status*. O leilão tem como *status* 'aberto', 'fechado' ou 'concluído'. Os lances só podem ser efetuados quando o status do leilão está aberto. Devido ao fato de que o ganhador pode recusar o objeto em análise, e ele é o único usuário que disputou o leilão, será necessário abrir novamente o leilão para os lances, passando do *status* de fechado novamente para aberto. Quando o ganhador aceitar a doação, o *status* do leilão passa a ser concluído, não podendo retornar aos *status* anteriores.

<sup>2</sup> A imagem completa do modelo físico do banco de dados pode ser encontrada clicando aqui

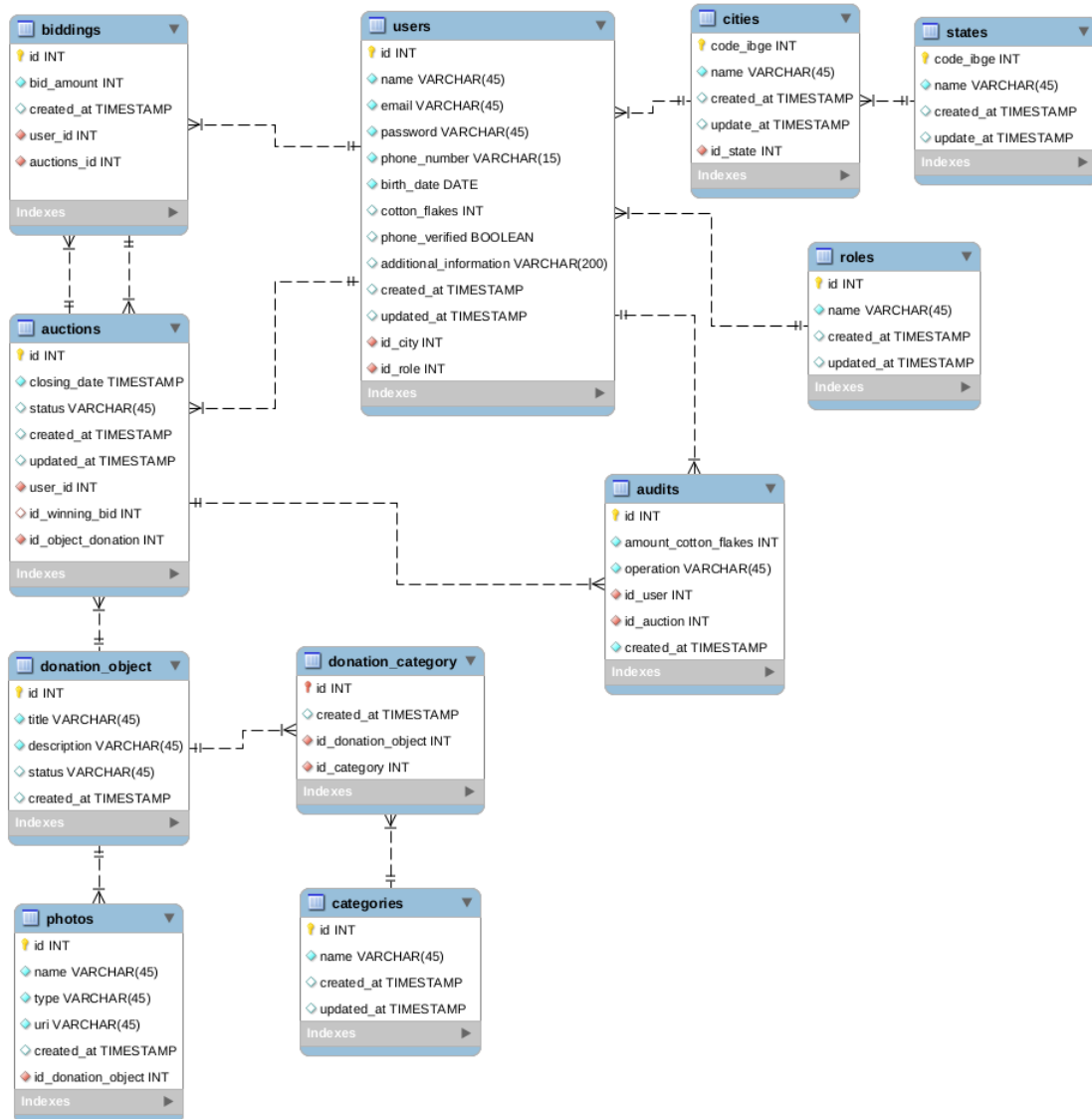


Figura 11 – Imagem da modelagem do banco de dados da aplicação *Cottonnect*.

Fonte: A autora..

Já as doações possuem *status* de em análise, recusada e recebida. O *status* estará em análise quando está aguardando o usuário ganhador receber o produto. Caso ele recuse, o *status* passará para recusado, onde deverá ser gerado outro ganhador. Caso não haja outros usuários, o leilão será aberto para lances novamente. Quando a doação for concluída, o seu *status* será definido como recebido, que é o momento que ocorre a transferência das moedas para a conta do doador.

No decorrer da evolução do desenvolvimento das *Sprints*, algumas tabelas e colunas foram adicionadas, e outras foram removidas, como por exemplo a tabela *audits* que foi decidido juntamente com o *Product Owner* a não implementação nesse primeiro momento devido a sua complexidade. Também foi adicionada uma nova coluna denominada *password-verification-code* que é usada pra guardar o código de verificação para recuperação de conta quando ocorre o esquecimento da senha. Estes casos são mencionados com maiores detalhes no Capítulo 5

Desenvolvimento do Sistema. Na Figura 12 é apresentada a versão final da modelagem de banco de dados, ou seja, após a finalização da versão da aplicação apresentada neste trabalho.

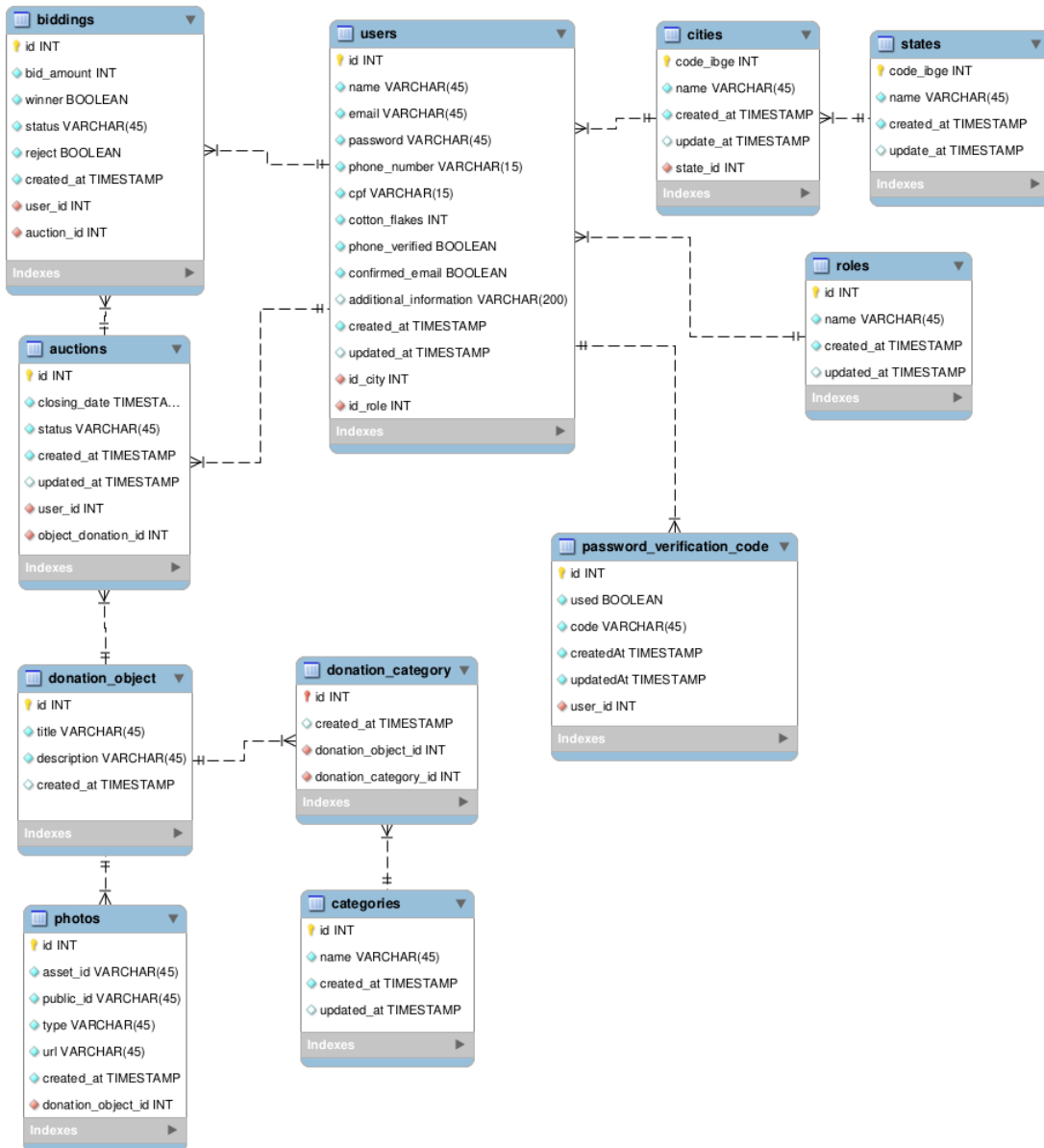


Figura 12 – Imagem da modelagem final do banco de dados da aplicação Cottonnect.

Fonte: A autora..

### 4.3.2 Protótipos das Telas

O processo de prototipação das telas<sup>3</sup> foi realizado utilizando a ferramenta *Figma*, com o auxílio de *templates* disponibilizados pela comunidade. A seguir são apresentadas as prototipações das principais telas que compõem o sistema.

A criação da conta do usuário se dá por meio da construção de quatro telas principais. A primeira representada na Figura 13, dispõe as informações de nome, *e-mail*, senha, telefone, data de nascimento e um campo para informações adicionais que o usuário poderá inserir à respeito do seus contatos. Este campo é importante para facilitar que o doador entre em contato com o ganhador, pois será disponibilizado ao doador, quando uma doação for concluída. A Figura 14 apresenta a segunda etapa do cadastro, onde o usuário deve informar sua localização, por meio da cidade e estado. Na Figura 15 é apresentada a terceira etapa do cadastro, onde o usuário deverá informar o código que foi enviado para confirmação do seu número de telefone e por fim, a Figura 16 apresenta a tela de sucesso quando uma conta é criada.

**Figura 13 – Imagem que apresenta a primeira etapa na criação da conta.**

**Fonte: A autora..**

A Figura 17 apresenta a tela principal da aplicação, onde ocorre a listagem de todas as doações disponíveis para o usuário por meio do cadastro da sua localização. Há um menu lateral, onde os usuários poderão navegar pelo sistema, sendo possível, acessar as páginas para anunciar uma doação, visualizar suas próprias doações, visualizar as doações ganhas, filtrar as doações por categorias e localização assim como também é possível filtrar as doações por meio do filtro na tela principal. Logo acima, no menu principal, o usuário poderá acessar seu perfil

<sup>3</sup> O layout com as imagens pode ser encontrado em: <https://www.figma.com/file/v6l75nZj6EBvds8WtapVbj/Cottonnect>





Figura 14 – Imagem que apresenta a segunda etapa na criação da conta.

Fonte: A autora..



Figura 15 – Imagem que apresenta a terceira etapa na criação da conta.

Fonte: A autora..

e também poderá receber notificações quando ganhar um produto, que serão apresentadas no ícone de sino, ao lado do identificador do usuário.

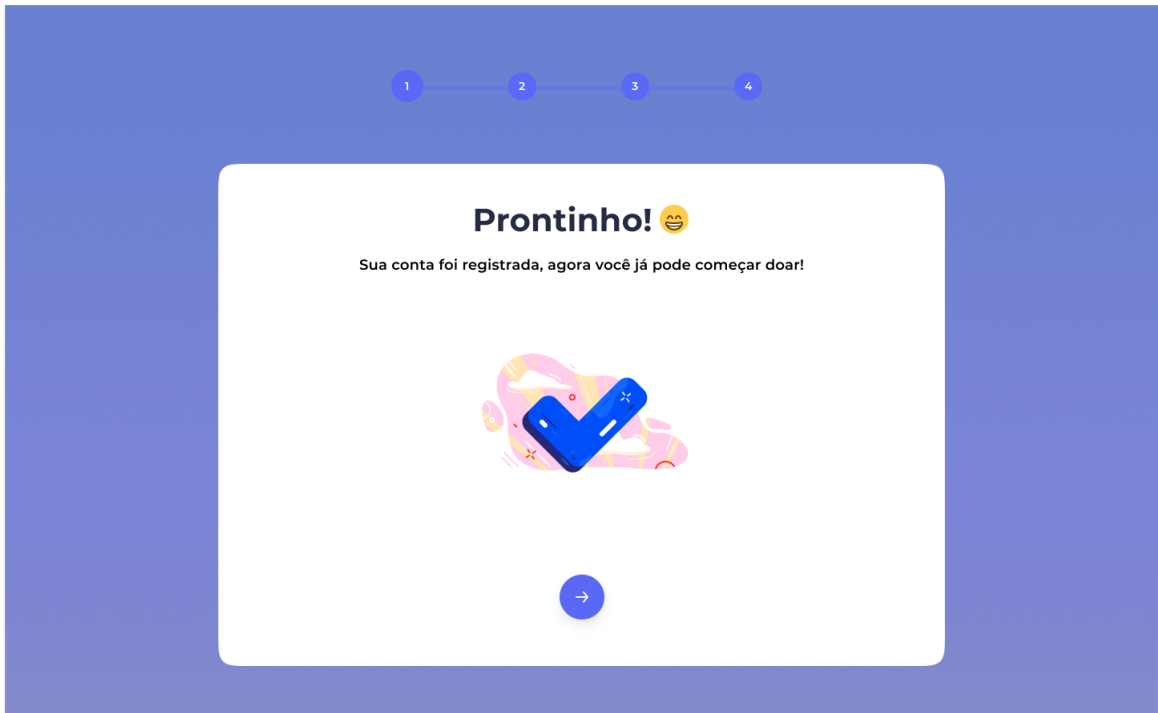


Figura 16 – Imagem que apresenta a tela de sucesso na criação de uma conta.

Fonte: A autora..

Para efetuar um lance em uma doação é disponibilizado o botão chamado "eu quero", clicando nele abrirá um modal, que pode ser visualizado na Figura 18 em que o usuário poderá informar a quantidade de moedas virtuais que ele quer dar de lance para a doação escolhida.

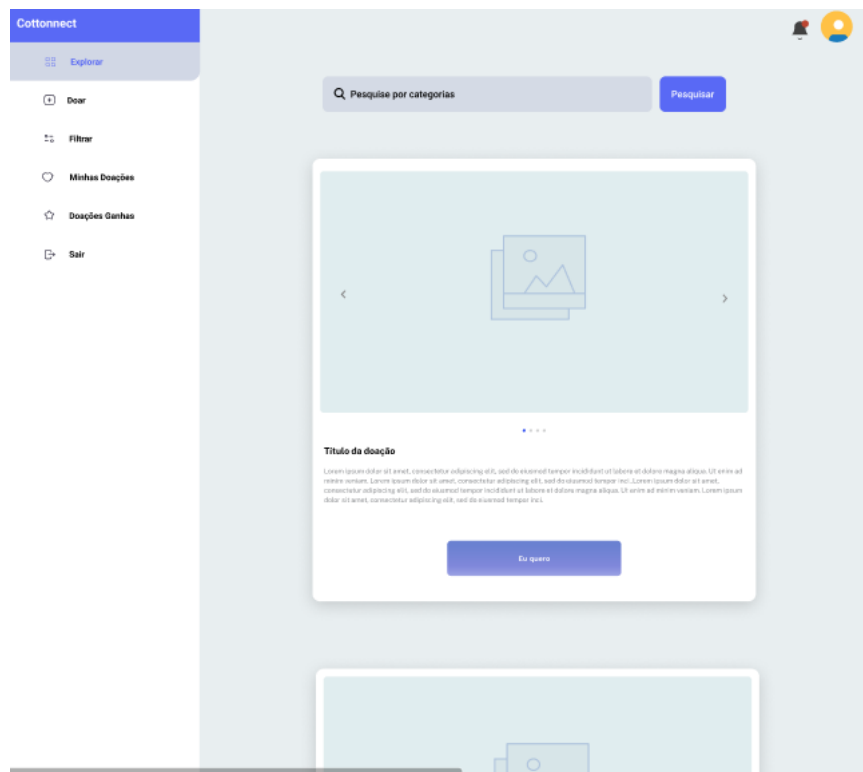


Figura 17 – Imagem da tela inicial que lista as doações.

Fonte: A autora..

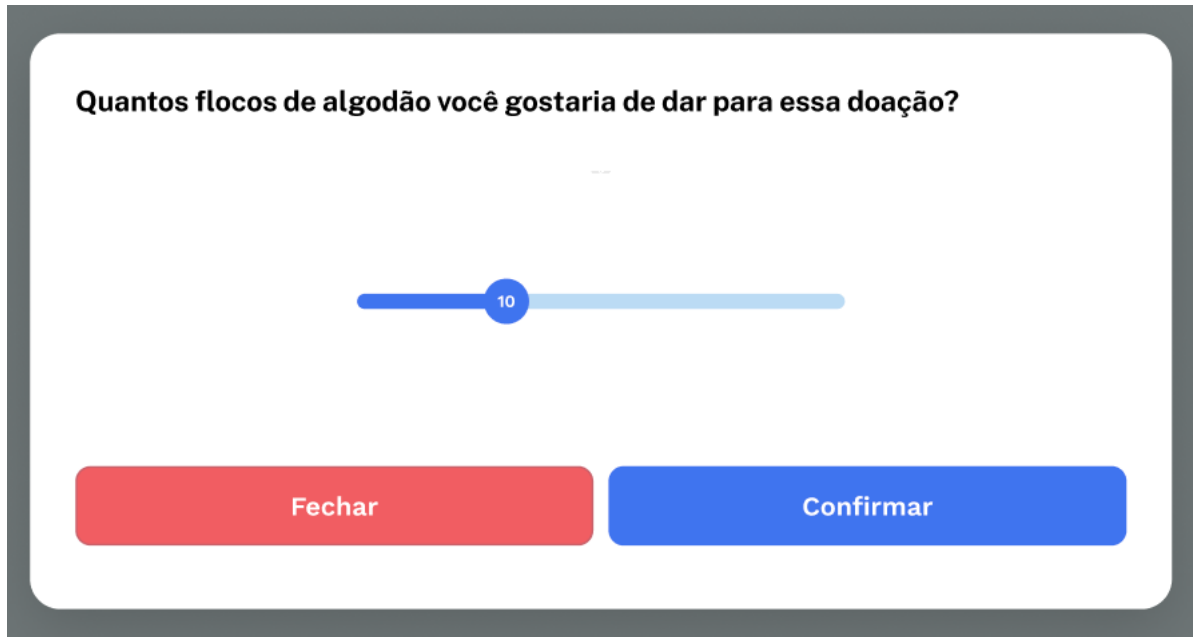


Figura 18 – Imagem da tela do modal que permite dar um lance para uma doação.  
Fonte: A autora..

A Figura 19 apresenta a página em que é possível cadastrar uma doação. Por meio dela é possível inserir as informações do objeto a ser doado como título, descrição, categorias, fotos e a data de fechamento do leilão, no qual o sistema irá definir o ganhador da doação.

A imagem mostra a interface de usuário para o cadastro de uma doação. No topo, há o logotipo "Cottonnect" e o slogan "Doe algo!" com um emoji de sorriso. À esquerda, há um menu lateral com opções: Explorar, Doar (destacado), Filtrar, Minhas Doações, Doações Ganhas e Sair. O formulário principal contém os seguintes campos:

- Título\***: Campo de texto para o título da doação.
- Data de fechamento \***: Campo de data com ícone de calendário. Abaixo dele, há o texto: "Essa será a data para fechar a doação e gerar o ganhador".
- Categorias \***: Menu suspenso com a opção "Selecione".
- Fotos \***: Botão "Choose File" com ícone de upload e um botão "Procurar".
- Descreva sua doação**: Área de texto grande para a descrição da doação.

Na base do formulário, há um botão azul arredondado com o texto "Doar".

Figura 19 – Imagem da tela onde é realizado o cadastro de uma doação.

Fonte: A autora..

A Figura 20 apresenta todas as doações que foram cadastradas pelo usuário dono da conta. Elas podem ser visualizadas de acordo com seu status, e por meio delas é possível visualizar o ganhador quando esta for concluída.

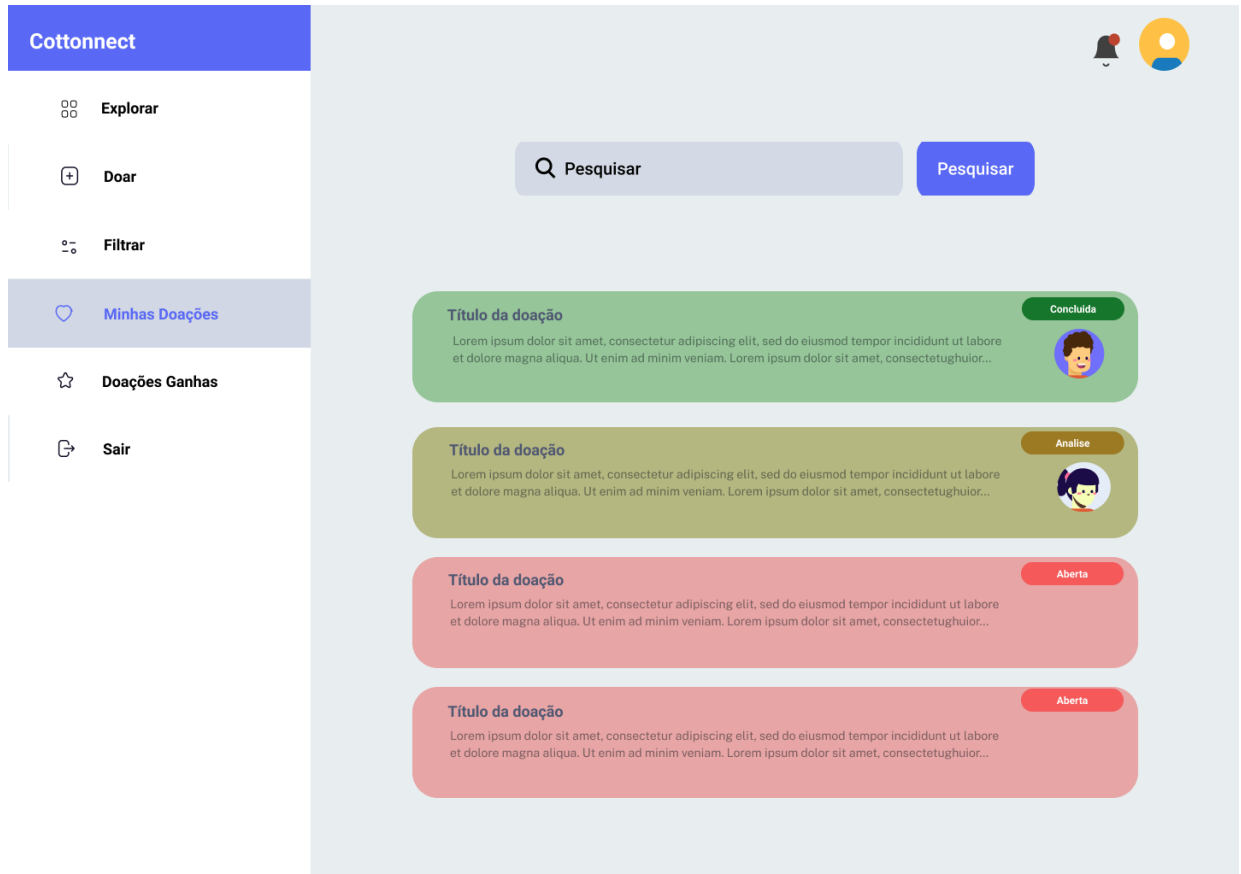


Figura 20 – Imagem que apresenta todas as doações de um usuário.

Fonte: A autora..

## 5 DESENVOLVIMENTO DO SISTEMA

Neste capítulo serão apresentadas as fases do desenvolvimento da aplicação em forma de *Sprints* conforme a metodologia *Scrum*, as quais estão divididas em seções.

### 5.1 SPRINT 01

O desenvolvimento da primeira *Sprint* consistiu em tarefas de cunho mais técnico, como as configurações do ambiente de desenvolvimento, envolvendo a criação da estrutura do sistema. Em seguida, deu-se início à criação da estrutura do banco de dados, criando todas as entidades pré-definidas, exceto a tabela denominada auditoria, sendo que, em análise junto ao *Product Owner*, foi decidido pela não implementação neste primeiro momento.

Ainda, na primeira *Sprint*, também foi iniciada a configuração do ambiente para testes na *API*, juntamente com a configuração das *Factories* para as entidades. Inicialmente, a ideia seria encontrar uma biblioteca já existente que facilitasse a criação, porém não foi encontrada uma biblioteca que tivesse uma boa confiabilidade. Por conta disto, as *Factories* foram geradas de forma manual, o que acabou levando um pouco mais de tempo do que previamente estimado.

Tarefas:

- Criar a estrutura de pastas da *API*;
- Configurar o projeto;
- Criar as entidades no banco de dados;
- Configuração da biblioteca *Jest*;
- Criação das *Factories* para as entidades.

### 5.2 SPRINT 02

Na *Sprint 02* foi iniciado o desenvolvimento da história de número 01, permitindo a criação das contas dos usuários. Porém, foram aplicados alguns refinamentos. Em conjunto com o *Product Owner*, foi definida a remoção da coluna de data de nascimento e adicionada a obrigatoriedade de envio do CPF pelo usuário. A decisão da adição do campo de CPF se deu pelo fato de dificultar a criação de contas falsas e ter uma garantia maior da segurança da aplicação. Também foi definido juntamente com o *Product Owner* que inicialmente as contas criadas começaram com a quantidade de 5 moedas e que posteriormente isso poderá ser alterado pelo administrador do sistema.

O cadastro do usuário foi separado em 3 passos para tornar a experiência do usuário mais intuitiva. No passo 1, foram adicionados os campos de informações pessoais e de contato

do usuário, como pode ser analisado na Figura 21. No passo 2, foi adicionado os campos para inserção da localização, demonstrado na Figura 22. E no passo 3, Figura 23, foi adicionado o campo para inserir o código da confirmação do número de celular.



**Figura 21 – Imagem da tela do passo 01 para criação de conta.**

**Fonte: Autora.**



**Figura 22 – Imagem da tela do passo 02 para criação de conta.**

**Fonte: Autora.**

Como o cadastro da conta de um usuário envolve previamente o cadastro de cidades e estados, foi adicionada uma nova tarefa para realizar o cadastro manual através de uma *Seed* para persistir no banco de dados as cidades habilitadas. A decisão de quais cidades poderão fazer uso da aplicação é definida manualmente pelo administrador, liberando aos poucos para as cidades em que ele deseje que a aplicação seja lançada.

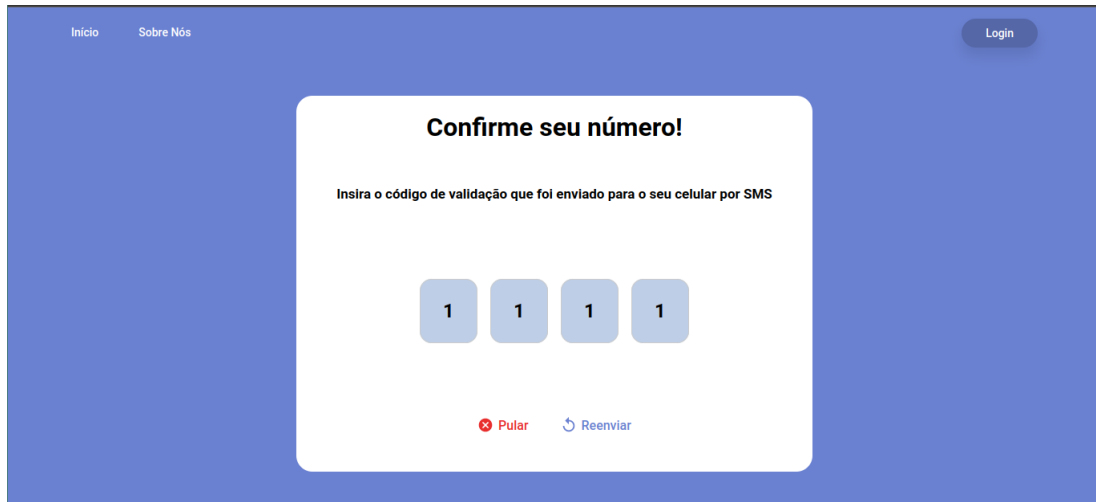


Figura 23 – Imagem da tela do passo 03 para criação de conta.

Fonte: Autora.

### 5.3 SPRINT 03

Na *Sprint* 03, foi iniciado o desenvolvimento da história de número 03, a qual contempla a página de login e a implementação da autenticação do usuário por meio do *JWT* na API. A Figura 24 demonstra a tela que o usuário utiliza para se logar na aplicação.

Nesta *Sprint* não foi implementada a recuperação da conta em caso de esquecimento da senha. Esta funcionalidade foi deixada para ser implementada em uma *Sprint* posterior. Esta decisão foi tomada juntamente com o *Product Owner*, considerando que a ação envolveria o envio de *e-mails*. Desta forma, a intenção foi focar neste primeiro momento nas principais funcionalidades do sistema.

Também, nesta *Sprint*, foi configurado o *GitHub Actions* para executar, de forma automática no *GitHub*, os testes que estavam sendo construídos. Na Figura 25 é demonstrado o *Pipeline* com todos os testes passando com sucesso.

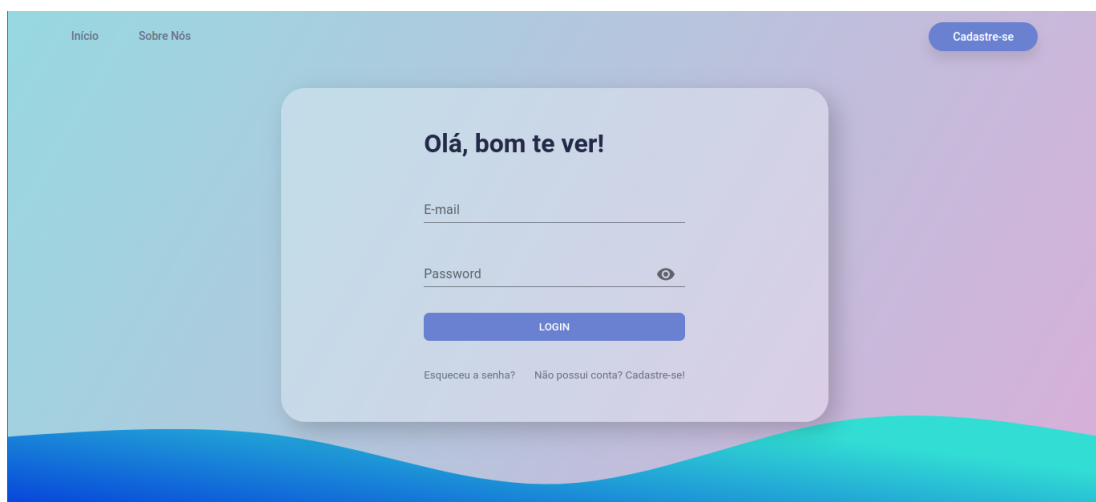


Figura 24 – Imagem da tela do passo 03 para criação de conta.

Fonte: Autora.



```

✓ ✓ Run Tests
 8 PASS src/__test__/controller/user-controller-test.spec.ts (14.915 s)
 9 PASS src/__test__/controller/auction-controller-test.spec.ts (9.414 s)
10 PASS src/__test__/services/auctions/generate-winner-service.spec.ts
11 PASS src/__test__/controller/bidding-controller-test.spec.ts (5.99 s)
12 PASS src/__test__/unit/user-unit-test.spec.ts
13 PASS src/__test__/unit/state-unit-test.spec.ts
14 PASS src/__test__/unit/city-unit-test.spec.ts
15 PASS src/__test__/controller/state-controller-test.spec.ts
16 PASS src/__test__/unit/role-unit-test.spec.ts
17 PASS src/__test__/unit/auction-unit-test.spec.ts
18 PASS src/__test__/unit/donation-object-unit-test.spec.ts
19 PASS src/__test__/unit/category-unit-test.spec.ts
20 PASS src/__test__/controller/category-controller-test.spec.ts
21 PASS src/__test__/unit/password-verification-code-test.spec.ts
22 PASS src/__test__/unit/bidding-unit-test.spec.ts
23 PASS src/__test__/unit/photo-unit-test.spec.ts
24
25 Test Suites: 16 passed, 16 total
26 Tests:      81 passed, 81 total

```

Figura 25 – Imagem dos testes da API passados com sucesso.

Fonte: Autora.

#### 5.4 SPRINT 04

Na *Sprint* 04 foi iniciada a história de número 04, a qual contempla o desenvolvimento da página de cadastro de leilão. Primeiramente, foi desenvolvida a ação de criação de categorias, onde, em um primeiro momento, foram geradas manualmente através de uma *Seed* pelo administrador da aplicação. Depois de finalizar o cadastro de categorias, foi iniciado de fato o desenvolvimento da tela de cadastro de leilões. Na Figura 26 é apresentado a tela que possibilita a ação.

A respeito da definição da data de fechamento dos leilões, surgiram algumas dúvidas naquele momento, principalmente se a estratégia a ser adotada seria deixar o usuário dono do leilão escolher uma data, ou se a própria aplicação atribuiria uma data para o fechamento, ou ainda se o usuário poderia escolher uma data e hora ou apenas a data. Em conjunto com o *Product Owner*, foi decidido deixar a cargo do usuário escolher qual data o leilão irá fechar e gerar o ganhador e o horário foi definido sendo fixo para gerar sempre as 23:59.

Ainda nesta *Sprint*, foi desenvolvido o cadastro de leilões na *API*, a criação dos respectivos testes e também configurações e implementação da biblioteca do *Cloudinary* para o armazenamento das imagens do objeto doado.

Os atributos de criação do leilão passaram por um processo de refinamento. Na primeira análise havia-se decidido que o usuário dono do objeto poderia atribuir um preço mínimo de lance para o leilão. Esse comportamento foi removido, deixando o lance aberto para que o usuário que irá dá-lo defina qual valor ele julga justo por aquela doação.

Também foi definido quais os tipos de status um leilão pode possuir:

- Aberto: Significa que o leilão está disponível para receber lances;
- Fechado: Significa que o leilão está indisponível para receber lances.

**Figura 26 – Imagem da criação de um leilão.**  
**Fonte: Autora.**

## 5.5 SPRINT 05

Na *Sprint* 05 foi desenvolvida a história de número 44, a qual contempla a página para visualização dos detalhes de um leilão. Inicialmente, esta tarefa não havia sido mapeada, mas ao decorrer do desenvolvimento, juntamente com o *Product Owner*, julgou-se necessário a visualização com informações mais detalhadas para o usuário.

Neste primeiro momento, foi definido a exibição do título, descrição, data de fechamento e suas respectivas fotos, como pode-se visualizar na Figura 27. E na parte da *API* foi criado o *end-point* para retorno das informações necessárias para página de detalhes do leilão e seus respectivos testes.

Ainda nessa *Sprint*, foi desenvolvida a história de número 11, que contempla a ação de dar uma gratificação em um leilão. Esta ação foi amplamente discutida com o *Product Owner* em relação a deixar a aplicação mais segura e dificultar fraudes. Algumas ideias abordadas foram as de congelar o valor dos flocos de algodão usados no lance para evitar que o usuário de lances e ganhe muito leilões e depois não tenha como gratificar e também se o usuário poderia excluir um lance já dado.

Nesse primeiro momento foi decidido não congelar o valor do lance e apenas verificar na hora de gerar o ganhador, se o usuário em questão tem a quantidade de moedas suficientes para ganhar o leilão. Caso ele não tenha, a aplicação irá gerar um novo ganhador. Foi decidido implementar apenas a verificação de quantidades de moedas suficientes quando o usuário tenta

dar um lance maior que a quantidade de moedas que ele possui no momento. E a ação de excluir algum lance dado, foi discutido, e a mesma poderá ser implementada como um trabalho futuro.



Figura 27 – Imagem dos detalhes de um leilão.  
Fonte: Autora.

## 5.6 SPRINT 06

Na *Sprint* 06 foi realizado o desenvolvimento da página de explorar, relacionada a história de número 09, a qual lista os leilões abertos de acordo com a cidade em que o usuário reside. Foi decidido trazer os leilões abertos de acordo com a cidade e estado que o usuário inicialmente informou no cadastro da sua conta, por esse motivo a tarefa de número 08 foi removida, visto que ela seria utilizada para obtenção da geolocalização em tempo real do usuário logado. A página de explorar pode ser visualizada na Figura 28.

A pedido do *Product Owner*, também foi realizada a história de número 46, a fim de mostrar a quantidade de gratificações que o leilão em questão possui, com o objetivo de ajudar o usuário que quer dar um lance se basear em um valor que ele acha que faz mais sentido.

Também, nesta *Sprint*, foi desenvolvida a história de número 10 a fim de facilitar a busca por alguma doação específica, adicionando filtros para os leilões. Em conjunto com o *Product Owner*, foi decidido neste primeiro momento, adicionar apenas os filtros por categoria e pelo título da doação.

Ainda nesta *Sprint*, foi implementada a história de número 17, que contempla a página que lista as doações do usuário que está logado, onde foram subdivididas em 4 estados:

- Aberto: São leilões que ainda estão abertos para receber gratificações;
- Em Andamento: São leilões fechados que estão aguardando aceite ou recusa do ganhador;

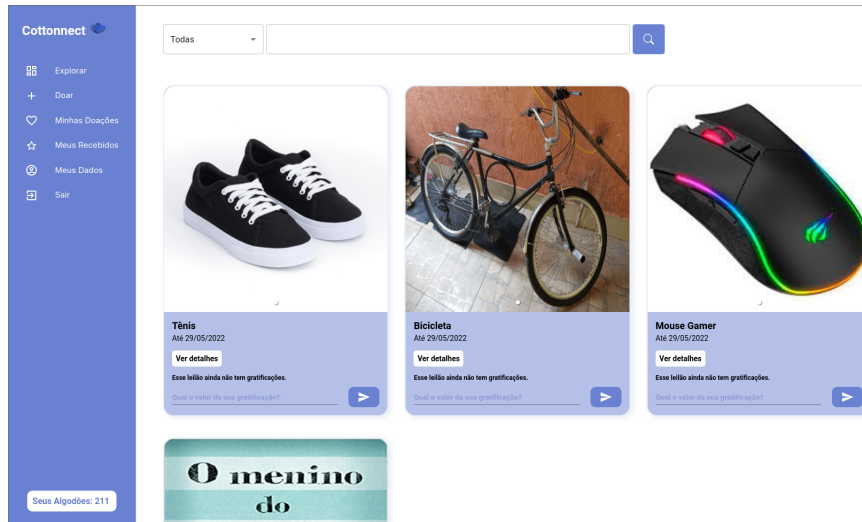


Figura 28 – Imagem da página que lista os leilões abertos.

Fonte: Autora.

- Concluído: São leilões que foram aceitos pelo ganhador;
- Sem Ganhador: São leilões fechados que não tiveram nenhum lance ganhador, portanto, o dono do leilão poderá abri-lo novamente para receber gratificações.

Inicialmente, a página havia sido construída com a identificação de estado em cada *card* de leilão, porém na análise do *Product Owner*, foi identificado que uma visualização através de abas ficaria mais intuitivo. Então, a tela foi reconstruída para se adaptar a abas, tendo como resultado final a Figura 29.

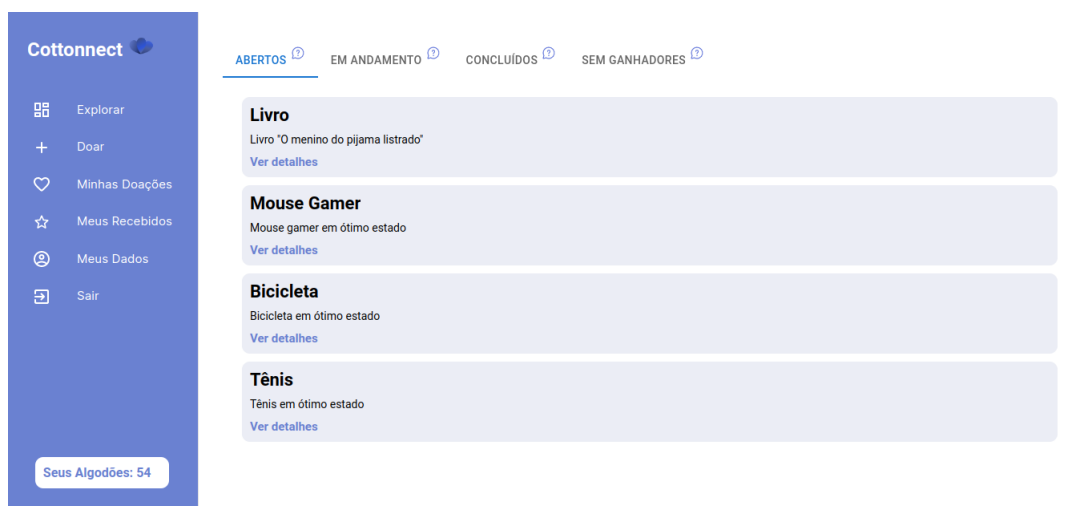


Figura 29 – Imagem da página que lista os leilões do usuário logado.

Fonte: Autora.

A Figura 30 apresenta o gráfico de *burndown* gerado para a *sprint* em questão. Nessa *sprint* foram desenvolvidas 4 tarefas, que inicialmente foram planejados 25 pontos para a conclusão. Na primeira tarefa, o valor estimado foi um pouco menor do que o realizado, o que ocasionou uma baixa. Porém, na tarefa seguinte, o valor estimado foi maior do que o realizado,

isso ajudou a estabilizar a *sprint*, pois o ponto que faltou na tarefa anterior pode ser usado com o que sobrou da segunda tarefa. E nas últimas tarefas, a pontuação realizada correspondeu a estimativa inicial.



Figura 30 – Imagem do gráfico de *burndown* da *sprint* 06

Fonte: Autora.

## 5.7 SPRINT 07

Na *Sprint* 07 foi desenvolvida a história de número 20, que contém a página que lista as doações que o usuário que está logado deu gratificações, onde foram subdivididas em 4 estados:

- Participando: são leilões que ainda estão abertos para receber gratificações;
- Em Andamento: são leilões fechados que estão aguardando aceite ou recusa do usuário logado;
- Concluído: são leilões que foram aceitos pelo usuário logado;
- Rejeitado: são leilões fechados em que o usuário logado foi o ganhador mas rejeitou a doação.

Nessa *Sprint* também foi realizada as histórias de números 12 e 45, adicionando a visualização do proprietário para o ganhador e do ganhador para o proprietário do leilão, com o objetivo de promover o contato entre ambas as partes. A Figura 32 mostra a visualização das informações do usuário.



Figura 31 – Imagem da página que lista os leilões que foram gratificados pelo usuário logado.

Fonte: Autora.

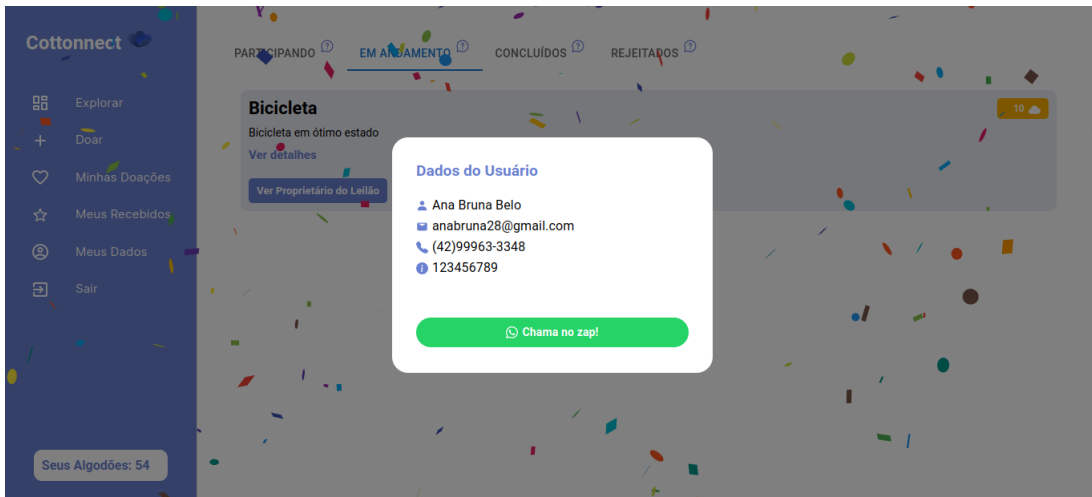


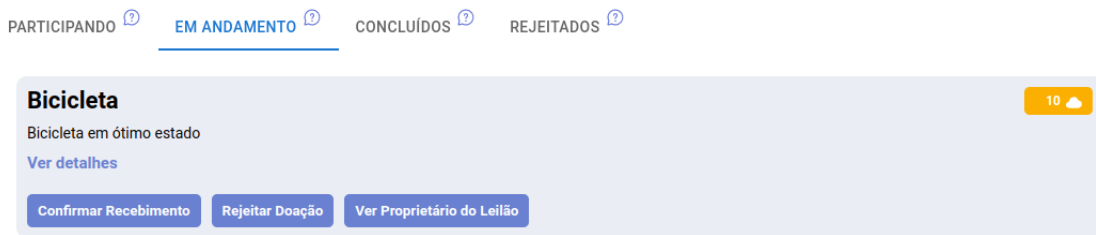
Figura 32 – Imagem que mostra a visualização do proprietário ou ganhador do leilão.

Fonte: Autora.

## 5.8 SPRINT 08

Nessa *Sprint* foi realizada as histórias de números 21 e 13, adicionado as ações de aceitar ou recusar da doação por parte do ganhador. Em conjunto com o *Product Owner*, foi definido que apenas o usuário ganhador tem acesso a ação de rejeitar ou aceitar a doação em questão, trazendo para o doador apenas a mentalidade de como a transação dos *cottons flakes* fosse parecida com um *pix* (Meio de pagamento eletrônico instantâneo). Então, para evitar golpes é necessário a entrega do objeto pessoalmente onde ele entregará o objeto e o ganhador clicará no botão de ação de aceite, o qual irá transferir as moedas para conta. A Figura 33 mostra os botões da ação.

Nessa *Sprint* também foi desenvolvida a história de número 52, que gera um novo ganhador para o leilão nos casos em que o ganhador atual rejeitou a doação. Como a ação de



**Figura 33 – Imagem que mostra os botões da ação de aceite ou rejeição da doação.**

**Fonte: Autora.**

rejeitar não havia sido mapeada na definição inicial da base de dados, uma nova coluna que guarda a informação se o usuário rejeitou ou não a doação foi adicionada na tabela de lances.

## 5.9 SPRINT 09

Na *Sprint 09* foi desenvolvida a história de número 14 com foco nos dados de cadastro do usuário. Portanto, foi criada uma página para visualização e edição dos dados do usuário logado, ilustrada na Figura 34. As informações que estão para edição foram previamente definidas em conjunto com o *Product Owner*.



**Figura 34 – Imagem da página de visualização/edição das informações pessoais do usuário logado.**

**Fonte: Autora.**


Também foi necessário criar uma ação na *API* para quando o *e-mail* ou número de celular for alterado, a conta do usuário em questão volte a ficar com verificação pendente até o momento em que ele confirme o novo *e-mail* ou o novo número de celular. Por conta disso, foi necessário adicionar dois novos botões que são demonstrados na Figura 35 para que o usuário consiga validar sua conta.

**Meus Dados** 

Email * <input type="text" value="anabruna28@gmail.com"/>	Celular * <input type="text" value="(42)99963-3348"/>
Estado * <input type="text" value="Rio De Janeiro"/>	Cidade * <input type="text" value="Rio De Janeiro"/>
Informações Adicionais * <input type="text" value="123456789"/>	

SALVAR 

Reenviar email de confirmação 

Reenviar SMS de confirmação 

**Figura 35 – Imagem dos botões para ação do reenvio de códigos para confirmação do e-mail e número de celular.**

**Fonte: Autora.**

## 5.10 SPRINT 10

Na *Sprint* 10 foi realizado o desenvolvimento de confirmações de conta, através do e-mail e número de celular, as quais são identificadas pelas histórias de número 56 e 57. Para a confirmação do número de celular inicialmente havia sido mapeado a utilização da biblioteca MYZAP, porém por conta de instabilidades foi decidido a sua não utilização, o que acarretou na remoção da tarefa de número 02. Substituindo o MYZAP foi utilizado a plataforma do *Twilio* que disponibiliza inicialmente um valor gratuito que satisfaz o uso da aplicação neste primeiro momento e é de grande confiabilidade. Com a *Twilio* foi feita uma requisição para gerar um código e outra para validar se o código enviado está válido. E então, foi adicionada essa nova ação para o passo 3 do formulário de criação de conta e para o botão de reenvio de código de SMS na ação de edição da conta.

Posterior a confirmação do número de celular, foi iniciado o desenvolvimento da confirmação do e-mail. Para isso, foi necessária a criação de um *template* que é enviado para o e-mail cadastrado, com um botão para confirmação de e-mail. Quando clicado no botão, é enviado um código para validação. Para conseguir efetuar essa validação foi necessário a criação de uma nova tabela no banco de dados que salva o código gerado e depois o compara para verificação da validade do código.

Ainda nessa *Sprint* foi desenvolvida a história de número 53, em que possui a ação de recuperação da conta em casos de esquecimento da senha. Para isso, foi criado um novo *template* para ser enviado para o e-mail informado, com um link que o usuário poderá alterar sua senha, como mostra a Figura 36. Também foi necessária a criação de uma nova tabela para



salvar o código de validação gerado para recuperar a conta, a fim de validar a veracidade do usuário. Para cada ação de recuperação de senha é gerado um novo código e enviado para o *e-mail* do usuário que solicitou a ação e esse código é reenviado novamente para a *API* quando o usuário de fato manda a nova senha, então o código é checado e se for válido, a senha é alterada.



O formulário apresenta um ícone de cadeado azul no topo. Abaixo dele, o título "Definir nova senha" está centralizado. Há dois campos de entrada de texto: "Nova senha \*" e "Confirmar nova senha \*", cada um com uma linha de base. Abaixo dos campos, há uma linha de texto que diz: "\*Certifique-se de que tenha pelo menos 8 caracteres.". No final, há um botão azul arredondado com o texto "ALTERAR" em branco.

Figura 36 – Imagem que permite mudar a senha para recuperação da conta  
Fonte: Autora.

## 5.11 SPRINT 11

Na *Sprint* 11 foi realizada a história de número 55, referente ao agendamento de um evento para executar todos os dias às 23:59h buscando leilões com a data de fechamento com a mesma data do dia corrente para gerar seus respectivos ganhadores. Para isso, foi utilizada a biblioteca *job-schedule* que permite configurar um horário e dias específicos para executar uma ação desejada.

Também nessa *Sprint* foi realizada a história de número 19, criando um novo *template* para enviar no *e-mail* do ganhador e do proprietário de um leilão quando o mesmo é fechado e gerado um ganhador. Assim, os usuários sempre estão sendo notificados quando ocorre o fechamento de dado leilão. Esse *template* pode ser visualizado na Figura 38.

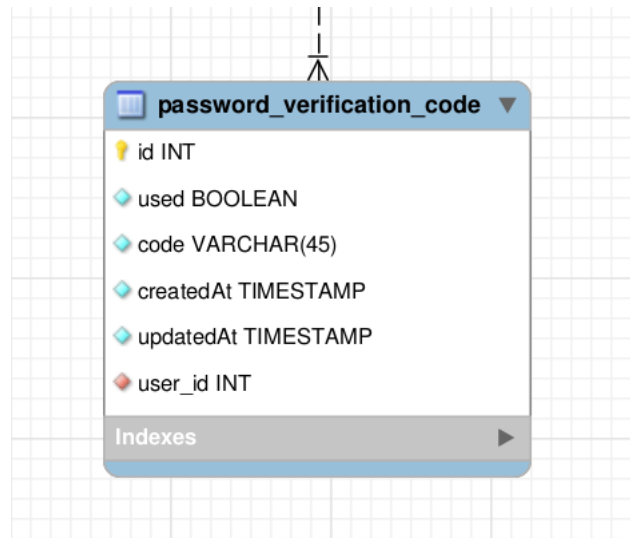


Figura 37 – Imagem da nova tabela que salva o código de verificação

Fonte: Autora.



Figura 38 – Imagem do *template* enviado quando um leilão é ganhado

Fonte: Autora.

## 5.12 SPRINT 12

Na *Sprint* 12 foi desenvolvida a história de número 51, que contempla a ação para reabrir um leilão nos casos em que nenhum ganhador foi gerado pela falta de lance ou recusa dos ganhadores. Então, em reunião com o *Product Owner*, foi decidido que o proprietário do leilão poderia querer colocar o leilão aberto a gratificações novamente, para isso, foi adicionada a ação para reabrir o leilão. A Figura 39 demonstra a tela para reativação do leilão.

Ainda nessa *Sprint* foi realizada a história de número 54, criando a página que contém as principais informações da aplicação, explicando o propósito da aplicação e dando algumas dicas a fim de ensinar o passo a passo para executar determinadas ações. Esta página tem o objetivo de ajudar os usuários da aplicação a entender melhor como a aplicação funciona e melhorar a sua experiência com ela. A página pode ser visualizada na Figura 40.

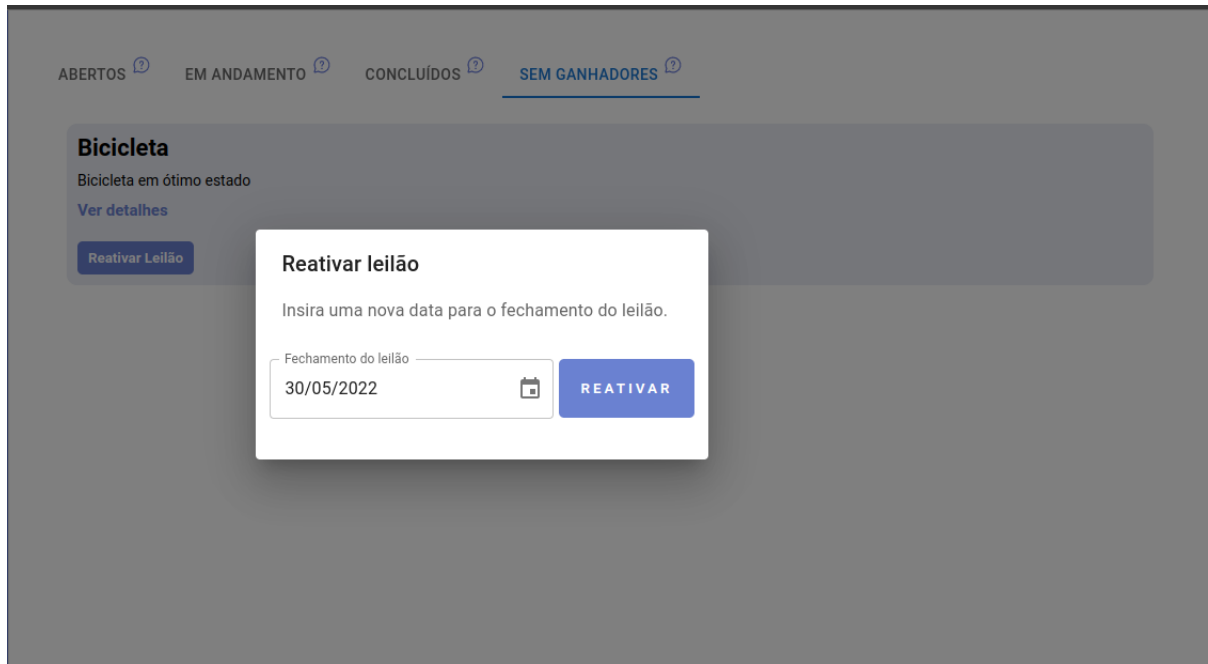


Figura 39 – Imagem do *template* enviado quando um leilão é ganhado

Fonte: Autora.



Figura 40 – Imagem que demonstra detalhes e tutorias da aplicação

Fonte: Autora.

E por fim, foi realizada a hospedagem da aplicação tanto da *API* quando do *front-end* para um ambiente no *Heroku*. Inicialmente, foi detalhado que seria configurado um ambiente para fazer o *deploy* contínuo da aplicação, porém as telas acabaram sendo apresentadas no próprio ambiente local para o *Product Owner*, o que ocasionou a não utilização de um ambiente de entrega contínua.

### 5.13 CONSIDERAÇÕES SOBRE O DESENVOLVIMENTO

Vale ressaltar que o desenvolvimento desse presente trabalho teve as suas primeiras *sprints* iniciadas em outro período letivo, devido ao momento pandêmico global enfrentado, afetando portanto, a finalização deste trabalho em um único período letivo.

Também, ao decorrer do desenvolvimento não foi empregada, por completo, a metodologia sugerida pelo *Git Flow*, pelo fato do trabalho ser realizado apenas por um único desenvolvedor. Assim, acabou mais comodo utilizar o método tradicional para o versionamento do código.

Também é importante ressaltar que a metodologia TDD não foi aplicada para a construção de todos os testes de *back-end*, apesar de ser empregada na construção da maioria deles. Em alguns casos específicos de funcionalidades que foram construídas inicialmente pelo *front-end*, acabou-se desenvolvendo os testes após a criação da ação.

Em relação a utilização da tecnologia *Typescript* ao invés do *Javascrit*, a autora considera que no geral foi uma experiência positiva, aumentando a produtividade na maioria dos casos. Porém, é valido ressaltar que em determinados momentos quando a tipagem era desconhecida, como por exemplo em algumas bibliotecas externas, acabava-se levando um pouco mais de tempo para conclusão. E por esse motivo, em alguns casos, foi decidido utilizar uma tipagem genérica.

## 6 RESULTADOS

Neste capítulo são apresentados os resultados de uma pesquisa de avaliação da aplicação por um conjunto de pessoas. Esta pesquisa foi realizada com o intuito de avaliar o grau de aceitação da aplicação pelo público, receber sugestões e verificar possíveis falhas para propor melhorias como trabalhos futuros.

O experimento contou com a participação de 9 usuários aleatórios, uma vez que a divulgação ocorreu nos perfis das redes sociais do curso de TSI. Vale salientar, que mais usuários acessaram a aplicação, porém, não responderam o formulário para contabilizar como uma avaliação. Portanto, não houve uma solicitação direta, nominal e direcionada para que usuários avaliassem a aplicação, justamente para coletar a opinião real do público, o qual supostamente não teria vínculo direto com a autora do trabalho.

Os usuários receberam algumas orientações pelo texto de divulgação para solicitação da avaliação. Primeiramente, foi orientado para que os usuários lessem a página "sobre nós". Também, foi salientado em relação ao desempenho da aplicação, pois em alguns momentos, poderia ser lenta devido a utilização de um plano gratuito de hospedagem.

Após os experimentos, os usuários foram convidados a responder um questionário disponível na Internet feito pela autora. Para construção e disponibilização do questionário foi utilizada a ferramenta *Google Forms*. O questionário teve como intuito identificar a experiência dos usuários ao utilizarem a aplicação. Portanto, as perguntas foram focadas no desempenho e usabilidade, assim como no propósito e valor da aplicação.

Desta forma, são demonstrados sequencialmente nas Figura 41 até Figura 50 os gráficos com as perguntas e respectivas respostas. Foram realizadas 10 perguntas, sendo que as respostas estão apresentadas de forma agregada em gráficos no formato de pizza e de barras. Particularmente, os gráficos de barras possuem uma avaliação de 1 a 5, sendo que 1, indica muito fácil e o 5, indica muito difícil. Assim sendo, de um modo geral, após analisar as respostas, nota-se que a grande maioria das perguntas foram respondidas de forma positiva. Mesmo que não tenha havido um número significativo de avaliações, pode se considerar este grupo de usuários, como uma amostra de um público real que usariam a aplicação. Por fim, esta pesquisa com os usuários finais foi importante para se ter uma noção do aceite da corrente versão da aplicação. A mesma pesquisa pode ser realizada em situações futuras, se novas versões da aplicação forem lançadas, a fim de verificar o melhoramento dos resultados.

Ademais, no mesmo formulário, foi adicionado um campo para opinião, apontamentos ou sugestões sobre a aplicação. De um modo geral, foram recebidos comentários positivos como:

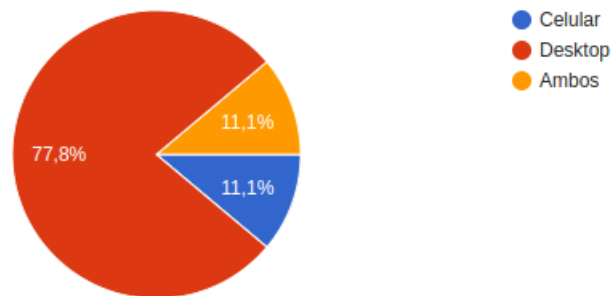
- Simples e fácil de usar;
- Achei super fácil de utilizar e gostei do design;
- Aplicação bem intuitiva e fácil de usar.

E também foram levantados alguns pontos de melhorias, que podem ficar como trabalhos futuros, ligados a melhorias de layout e usabilidade, como:

- Ajustar alguns ícones que ficaram sobrepostos no modo de visualização para aplicativo;
- Adicionar uma forma mais fácil de trocar a senha;
- Adicionar um botão para voltar para tela principal na tela de "esqueci minha senha", para casos em que seja clicado sem querer ou queira apenas voltar.

Você utilizou a aplicação via:

9 respostas

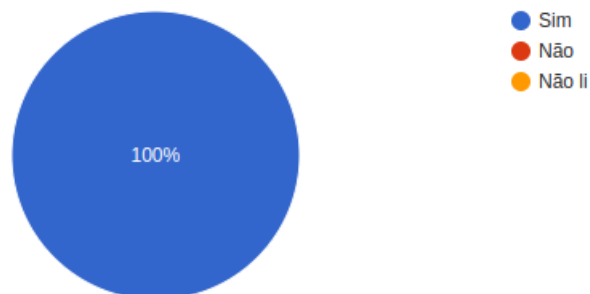


**Figura 41 – Resultado da primeira pergunta**

Fonte: Autora.

O tutorial descrito na página "sobre nós" te ajudou a entender melhor a aplicação?

9 respostas

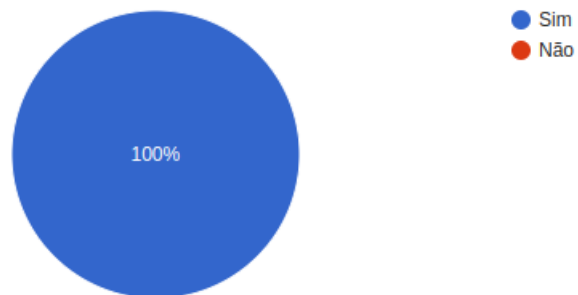


**Figura 42 – Resultado da segunda pergunta**

Fonte: Autora.

Você entendeu o propósito da aplicação?

9 respostas

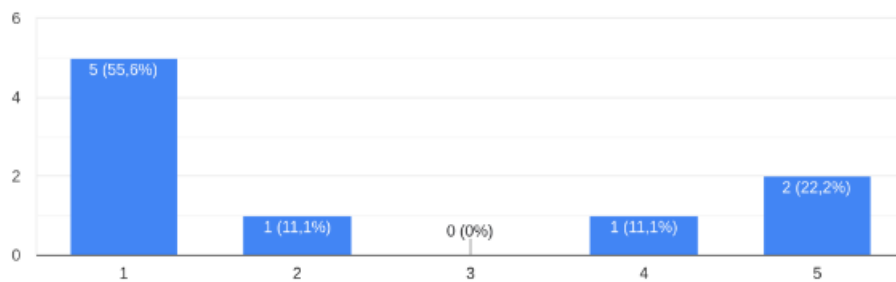


**Figura 43 – Resultado da terceira pergunta**

Fonte: Autora.

Como você considera a ação de cadastro de sua conta?

9 respostas

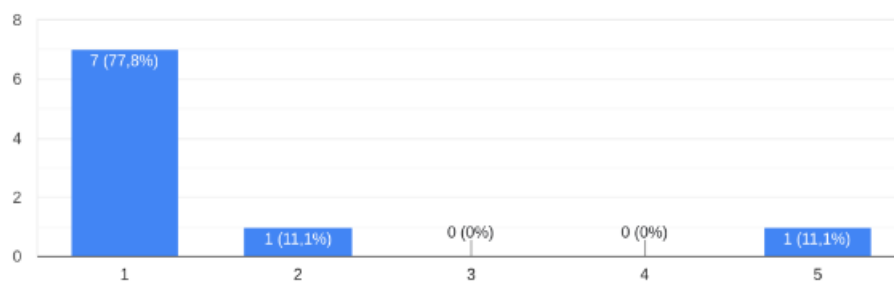


**Figura 44 – Resultado da quarta pergunta**

Fonte: Autora.

Como você considera a ação de cadastrar uma doação?

9 respostas

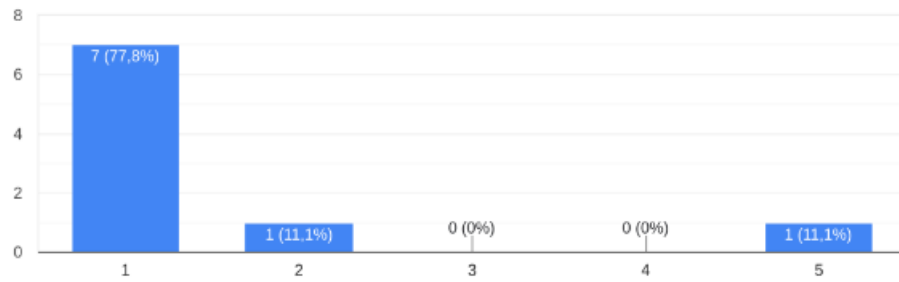


**Figura 45 – Resultado da quinta pergunta**

Fonte: Autora.

Como você considera a ação de dar uma gratificação (lance) ?

9 respostas

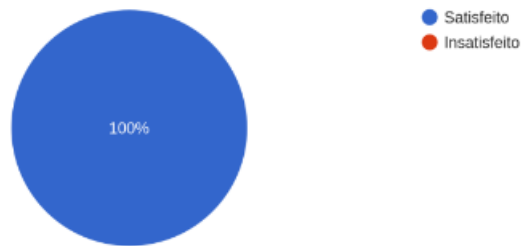


**Figura 46 – Resultado da sexta pergunta**

Fonte: Autora.

Qual sua opinião sobre o desempenho da aplicação?

9 respostas

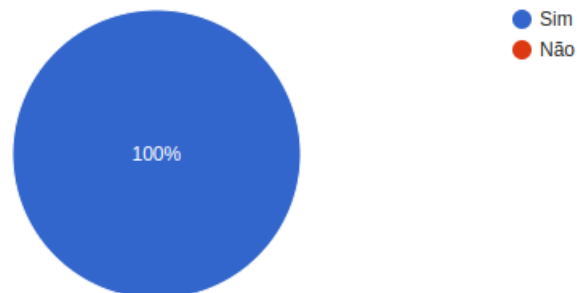


**Figura 47 – Resultado da sétima pergunta**

Fonte: Autora.

Você achou a utilização intuitiva?

9 respostas



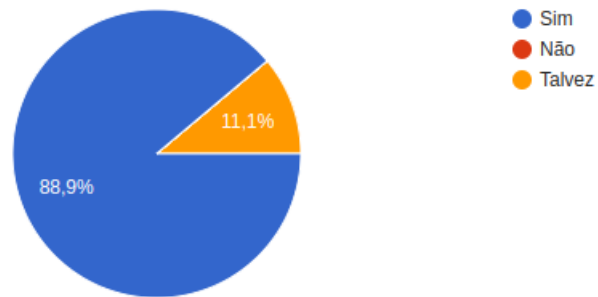
**Figura 48 – Resultado da oitava pergunta**

Fonte: Autora.



Você usaria a aplicação para doar ou receber produtos?

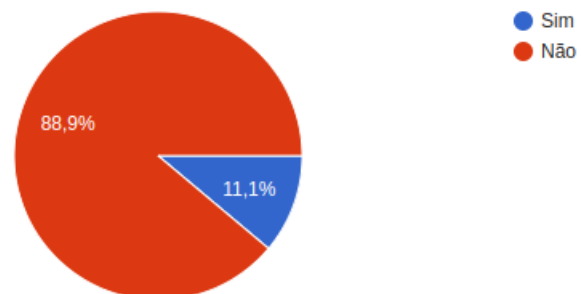
9 respostas



**Figura 49 – Resultado da nona pergunta**  
Fonte: Autora.

Você teve dificuldade em confirmar seu e-mail ou número de celular?

9 respostas



**Figura 50 – Resultado da décima pergunta**  
Fonte: Autora.

## 7 CONCLUSÃO

O resultado final pretendido com o presente trabalho foi o desenvolvimento de uma aplicação *web* para estimular doações de objetos entre pessoas em alguma comunidade, a fim de promover um consumo consciente nos indivíduos da sociedade, com o intuito de ajudar ao combate da degradação gradual do meio ambiente.

A aplicação permite que indivíduos ofereçam seus objetos e em troca sejam retribuídos com moedas virtuais. Tais moedas são representadas simbolicamente por flocos de algodão. Desta forma, em posse das moedas, o sistema torna possível requisitar outras doações, na qual disputará o objeto por meio de um leilão *online* fechado.

Por meio dos resultados obtidos através da avaliação por usuários que utilizaram a aplicação e responderam o formulário apresentado no Capítulo 6, conclui-se que a aplicação cumpre com as funcionalidades que se designa a realizar e também satisfaz os objetivos geral e específicos do presente trabalho.

Enfim, a maior dificuldade encontrada durante o desenvolvimento do sistema foi relacionada ao desenvolvimento de tarefas do *front-end*, pelo fato de não ser algo usual no dia a dia da autora. Estas tarefas demandaram bastante estudo e posteriormente, ações de refatoração a fim de melhorar o que já estava feito, à medida em que se aprendia mais sobre o assunto. Porém, destaca-se o grande aprendizado adquirido ao estar em contato com outras tecnologias e também na gestão de tempo empregadas para o desenvolvimento do trabalho.

### 7.1 TRABALHOS FUTUROS

Um dos trabalhos a ser implementado futuramente está diretamente relacionando com uma maior segurança ao sistema e aos usuários. Com a intenção de evitar fraudes na aquisição de moedas por doações falsas através de perfis falsos, futuramente, poderá ser implementado um método de validação de veracidade de usuários através de verificação humana de fotos com algum documento de identificação. Desta forma, pode-se aumentar o nível de confiança sobre a real identidade do usuário e dificultar a criação de perfis falsos que fazem mau uso da aplicação.

Um outro trabalho futuro seria o desenvolvimento de uma ação para penalizar usuários que ganharam um leilão e não foram ver o produto. Isso ajudaria a evitar que os usuários se cadastrassem em leilões onde não querem realmente o item doado, aumentando o tempo de espera para aquisição da doação para usuários que realmente querem a doação, caso acabem ganhando o leilão em questão.

Também é válido salientar que, devido ao tempo, nesta primeira versão da aplicação não foi implementada a aplicação em formato *PWA*. Porém, as páginas foram construídas respeitando o formato responsivo, para que futuramente a aplicação esteja apta a ser convertida para este formato.

Ainda, como trabalho futuro, é importante o desenvolvimento de todas as histórias mapeadas no *backlog* que não foram implementadas no decorrer deste trabalho. Basicamente, estas histórias estão relacionadas a área do administrador do sistema, a fim de evitar processos manuais, como o cadastro de novas categorias, cidades, estados e perfis. As histórias que formam parte dos trabalhos futuros podem ser vistas no Quadro 5.

**Quadro 5 – Histórias de Usuário**

<b>ID</b>	<b>Peso</b>	<b>Descrição da História</b>
22	3	COMO Administrador QUERO cadastrar novas categorias PARA adicionar novos tipos de categorias.
23	3	COMO Administrador QUERO cadastrar novos status PARA adicionar novos tipos de privilégio.
24	3	COMO Administrador QUERO cadastrar novos papéis de usuário PARA adicionar novos tipos de usuário.
25	3	COMO Administrador QUERO cadastrar novos usuários de maior privilegio PARA ajudar na manutenção da aplicação.
26	3	COMO Administrador QUERO definir a quantidade de moedas virtuais iniciais das contas PARA que quando um usuário crie sua conta ele tenha disponível uma quantidade de moedas limitadas.

## REFERÊNCIAS

- ABRELPE. **Panorama de Resíduos Sólidos no Brasil 2020**. [S.l.], 2020.
- ANICHE, M. Test-driven development. **Caelum**, 2014. Disponível em: <https://tdd.caelum.com.br>. Acesso em: 28 de março de 2021.
- ANTUNES, A. Pwa: O que são progressive web apps e por que usar? **GoBacklog**, 2021. Disponível em: <https://gobacklog.com/blog/progressive-web-apps>. Acesso em: 07 de maio de 2021.
- ARTIA. Kanban. o que é e tudo sobre como gerenciar fluxos de trabalho. 2021. Disponível em: <https://artia.com/kanban>. Acesso em: 05 de maio de 2021.
- ATLASSIAN. Fluxo de trabalho de gitflow. 2021. Disponível em: <https://www.atlassian.com/br/git/tutorials/comparing-workflows/gitflow-workflow>. Acesso em: 11 de maio de 2021.
- AUSUBEL, L. M. Auction theory for the new economy. 2003. Disponível em: <https://www.ausubel.com/auction-papers/auction-theory-new-economy.pdf>. Acesso em: 24 de abril de 2021.
- BAUER, C.; KING, G. **Hibernate in Action**. 6. ed. [S.l.]: Manning Publications, 2005.
- CAMARGO, R. Burndown: conheça o gráfico que mede produtividade. **ROBSON CAMARGO Projetos e Negócios**, 2020. Disponível em: <https://robsoncamargo.com.br/blog/Burndown>. Acesso em: 27 de março de 2021.
- CLOUDINARY. Plataforma clouinary media experience. 2022. Disponível em: [https://cloudinary.com/solutions/why\\_clouinary](https://cloudinary.com/solutions/why_clouinary). Acesso em: 28 de maio de 2022.
- COHN, M. **User Stories Applied: For Agile Software Development**. [S.l.]: Pearson Education, Inc., 2004.
- CONCEITO.DE., E. editorial de. Conceito de sms. 2014. Disponível em: <https://conceito.de/sms>. Acesso em: 25 de maio de 2022.
- Consumers International *et al.* **Manual de Educação para o Consumo Sustentável**. 2. ed. Brasília, 2005.
- ESLINT. About. 2021. Disponível em: <https://eslint.org/docs/about/>. Acesso em: 28 de abril de 2021.
- EXPRESS. Express. 2021. Disponível em: <https://expressjs.com>. Acesso em: 28 de abril de 2021.
- FENTON, S. **Pro TypeScript: Application-Scale JavaScript Development**. 2. ed. [S.l.]: Apress, 2017.
- FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**: Tese (phd in computer science) — university of california. [s.n.], 2000. Disponível em: <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>. Acesso em: 06 de maio de 2021.
- FLANAGAN, D. **JavaScript: O guia definitivo**. 6. ed. [S.l.]: Bookman, 2013.
- GIT. Git. 2021. Disponível em: <https://git-scm.com>. Acesso em: 28 de abril de 2021.

- GITHUB. Github. 2021. Disponível em: <https://github.com>. Acesso em: 28 de abril de 2021.
- GitHub Actions. Sobre integração contínua. 2021. Disponível em: <https://docs.github.com/pt/actions/guides/about-continuous-integration>. Acesso em: 18 de maio de 2021.
- GITLAB. 2021. Disponível em: <https://about.gitlab.com/>. Acesso em: 18 de maio de 2021.
- HAMEDANI, M. React virtual dom explained in simple english. 2021. Disponível em: <https://programmingwithmosh.com/react/react-virtual-dom-explained>. Acesso em: 04 de maio de 2021.
- HEROKU. The heroku platform. 2021. Disponível em: <https://www.heroku.com/platform>. Acesso em: 18 de maio de 2021.
- HEROKU. What is heroku? 2021. Disponível em: <https://www.heroku.com/about>. Acesso em: 28 de abril de 2021.
- JEST. Jest. 2021. Disponível em: <https://jestjs.io/pt-BR>. Acesso em: 28 de abril de 2021.
- JSON.ORG. Introducing json. 2021. Disponível em: <https://www.json.org/json-en.html>. Acesso em: 06 de maio de 2021.
- JUSTO, A. S. 2017. Disponível em: <https://www.euaxcom.br/2017/05/metodologia-scrum-o-que-voce-precisa-saber-2>. Acesso em: 27 de março de 2021.
- JWT. Introduction to json web tokens. 2021. Disponível em: <https://jwt.io/introduction>. Acesso em: 06 de maio de 2021.
- MAURER, L. T.; BARROSO, L. A. Electricity auctions: An overview of efficient practices. The World Bank, 2011. Disponível em: <http://www.cgti.org.br/publicacoes/wp-content/uploads/2016/03/Teoria-dos-Leilao%CC%83es-Formulac%CC%A7o%CC%83es-e-Aplicac%CC%A7o%CC%83es-no-Setor-Ele%CC%81trico.pdf>. Acesso em: 11 de maio de 2021.
- MILANI, A. **PostgreSQL - Guia do Programador**. [S.l.]: Novatec, 2008.
- NIELSEN. Brasileiros estão cada vez mais sustentáveis e conscientes. In: . 2019. Disponível em: <https://www.nielsen.com/br/pt/insights/article/2019/brasileiros-estao-cada-vez-mais-sustentaveis-e-conscientes/>. Acesso em: 20 de março de 2021.
- NODE. About node.js. 2021. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 26 de abril de 2021.
- POSTGRESQL.ORG. About postgresql. 2021. Disponível em: <https://www.postgresql.org/about>. Acesso em: 06 de maio de 2021.
- REACTJS. React.js. 2021. Disponível em: <https://pt-br.reactjs.org>. Acesso em: 04 de maio de 2021.
- ROSSETTI, J. P. **Introdução à economia**. 20. ed. [S.l.]: Atlas, 2013.
- SAUDATE, A. **Rest - Construa APIs inteligentes de maneira simples**. 1. ed. [S.l.]: Casa do Código, 2013.
- SCHWABER, K.; SUTHERLAND, J. **Guia do Scrum**: Um guia definitivo para o scrum: As regras do jogo. [S.l.], 2013. Disponível em: <https://scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em: 27 de março de 2021.

SILVA, M. S. **HTML 5: A linguagem de marcação que revolucionou a web**. 1. ed. [S.l.]: Novatec Editora, LTDA, 2011.

TRELLO. Trello. 2021. Disponível em: <https://trello.com/pt-BR>. Acesso em: 28 de abril de 2021.

TWILIO. Twilio customer engagement platform. 2022. Disponível em: <https://www.twilio.com/platform>. Acesso em: 25 de maio de 2022.

TWILIO. Verify. 2022. Disponível em: <https://www.twilio.com/pt-br/verify>. Acesso em: 25 de maio de 2022.

TYPEORM. Typeorm. 2021. Disponível em: <https://typeorm.io/#/>. Acesso em: 28 de abril de 2021.

TYPESCRIPT. O que é typescript? 2021. Disponível em: <https://www.typescriptlang.org>. Acesso em: 25 de abril de 2021.

W3SCHOOLS. O que é html? 2021. Disponível em: [https://www.w3schools.com/html/html\\_intro.asp](https://www.w3schools.com/html/html_intro.asp). Acesso em: 10 de maio de 2021.

WWF-BRASIL. **Em 2019, Terra entra no cheque especial a partir de 29 de julho**. [S.l.], 2019. Disponível em: <https://www.wwf.org.br/?72262/Em-2019-Terra-entra-no-cheque-especial-a-partir-de-29-de-julho>. Acesso em: 01 de março de 2021.