

# Pet Denuncie: Sistema de denúncias on-line de maus-tratos aos animais

Wendel M. A. Barbosa<sup>1</sup>, André C. da Silva<sup>1</sup>

<sup>1</sup>Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas  
Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP)  
Hortolândia - SP - Brasil

wendel.barbosa@aluno.ifsp.edu.br, andre.constantino@ifsp.edu.br

**Abstract.** *With the increase in various types of crime, it would be no different in the case of mistreatment of animals, giving rise to the need for new ways to combat these acts. With this in mind, this project was designed to reduce such criminal behavior. The main aim of this work is to develop a system for reporting mistreatment of animals that will be used by an administrator and the respective users. The users' role is to write and send reports of violence, while the administrator's role is to receive these reports in order to take the best possible action. Currently, mobile devices are the main means of communication, so the site will be responsive and will work on a wide variety of screen sizes.*

**Resumo.** *Com o aumento dos diversos tipos de crimes, não seria diferente no caso de maus-tratos a animais, surgindo a necessidade de novas maneiras de combater esses atos. Diante dessas perspectivas, esse projeto foi planejado para diminuir tais condutas criminosas. O objetivo central deste trabalho é desenvolver um sistema para denúncias de maus-tratos a animais que será utilizado por um administrador e pelos respectivos usuários. O papel dos usuários é escrever e enviar as ocorrências de violência, enquanto a função do administrador é receber essas denúncias para tomar as melhores medidas possíveis. Atualmente, os dispositivos móveis são o principal meio de comunicação, sendo assim, o site será responsivo e funcionará nos mais variados tamanhos de telas.*

## 1. Introdução

Os animais e os seres humanos sempre se relacionaram desde o início da humanidade, no entanto, há uma questão preocupante nessa relação: os maus-tratos contra os bichos. Maus-tratos são definidos como ato que causam dor, sofrimento ou até mesmo a morte dos animais [Crimlab 2024]. Exemplos comuns incluem abandono, espancamento, queimaduras, privação de água e comida, entre outras formas de crueldade.

Outro fator que torna a questão ainda mais preocupante é que os demais mamíferos possuem uma estrutura cerebral semelhante à dos seres humanos, o que lhes permite sentir emoções [PETZ 2024]. Apesar da resposta emocional dos animais ser diferente da resposta dos humanos, ela ainda existe - por exemplo, não é incomum que um animal de estimação desenvolva um vínculo maior com algum membro da família.

Os maus-tratos a animais representam uma forma de violência que afeta muitos seres vivos ao redor do mundo. De acordo com [Borba 2020], a violência contra animais foi o 5º crime mais cometido no Brasil no ano de 2020, sendo que a maioria dos brasileiros

que presencia essa conduta criminoso não faz a devida denúncia, muitas vezes por não saberem o que fazer e onde fazer. Segundo [G1 SP 2022], uma pesquisa do Ibope com 2000 internautas revelou que 92% já presenciaram alguma agressão, mas apenas 17% denunciaram.

Segundo [Rodrigues 2021], houve mais de mil denúncias de maus-tratos a animais em 2021. A Equipe Cães e Gatos, em 2022, relatou que a OMS registrou cerca de 30 milhões de animais abandonados no Brasil, reforçando a ideia de que esse é um crime muito comum.

Esses atos de violência são contra seres que muitas vezes não conseguem se defender. Segundo [Flávio 2022], a Lei Federal nº 14.064/20 ampliou a pena de 1 para 5 anos para indivíduos que praticarem a infração de maus-tratos a animais. Diante de tais perspectivas, além das medidas já existentes, fica claro a necessidade do desenvolvimento de novos modos para realizar denúncias e prevenir tais casos, visando uma sociedade mais justa.

Os índices alarmantes de denúncias de maus-tratos a animais em São Paulo em 2021, com um aumento significativo de 15% em relação ao ano anterior, evidenciam a urgência de medidas eficazes para combater esse problema crescente [G1 SP 2022]. A publicação de uma lei em dezembro de 2021 determinando que todos os condomínios, sejam comerciais ou residenciais, devem denunciar crimes dessa natureza, reflete um passo importante na conscientização e responsabilização coletiva.

Neste cenário, é evidente a urgência da realização de medidas práticas para promover o bem-estar dos animais e alcançar uma convivência saudável e empática entre pessoas e animais. Para abordar esses desafios, este trabalho propõe o desenvolvimento de um sistema *web* dedicado à facilitação da denúncia de maus-tratos a animais. O objetivo e diferencial é centralizar e simplificar o processo de denúncia, proporcionando uma plataforma acessível onde qualquer cidadão possa reportar incidentes de maneira eficiente e segura. A iniciativa visa não apenas aumentar o número de denúncias, mas também assegurar uma resposta rápida e adequada por parte das autoridades competentes, contribuindo assim para a proteção e o bem-estar animal.

O presente trabalho abordará os objetivos expostos com o desenvolvimento de um sistema web de denúncias de maus-tratos a animais, e está organizado em sete seções ordenadas em: introdução, referencial teórico, trabalhos correlatos, metodologias e materiais, desenvolvimento, conclusão e as referências bibliográficas utilizadas para compor a fundamentação teórica e metodologias.

## **2. Referencial Teórico**

Esta Seção aborda a pesquisa bibliográfica realizada para gerar o embasamento teórico sobre os conceitos e das tecnologias aplicadas.

### **2.1. Sistemas *web***

Nos primeiros dias da *World Wide Web*, os sites consistiam apenas em um conjunto de arquivos de hipertexto. Com o passar dos anos, a HTML foi complementada por diversas ferramentas e tecnologias de desenvolvimento, aumentando a complexidade do desenvolvimento *web*. No entanto, os usuários passaram a ter acesso a mais funcionalidades,

tornando a *web* indispensável para diversos setores, como negócios, comércios, educação, indústrias, entre outros.

Segundo [Beder 2012], um sistema *web* é um software de computador que oferece uma coleção de instruções executáveis e dados aos usuários - esses sistemas são projetados para serem acessíveis em diferentes dispositivos e plataformas, bastando uma conexão com a internet, proporcionando uma experiência consistente e intuitiva para os usuários.

## 2.2. Engenharia *web*

Como citado anteriormente, o desenvolvimento de sistemas *web* foi ficando cada vez mais complexo com o passar do tempo, e os sistemas precisam passar pelo processo de Engenharia *Web*.

Muitos processos da Engenharia de Software podem ser aplicados no desenvolvimento *web*, mas adaptando-se às respectivas necessidades. [Beder 2012] define a Engenharia *Web* como "a utilização de princípios, conceitos e métodos da Engenharia de Software, de uma maneira que se adaptem às características inerentes de aplicações *web*, tais como Desempenho, Disponibilidade, Evolução Contínua, Urgência e Segurança".

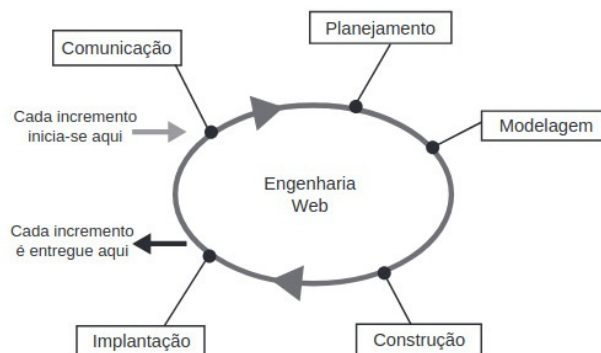
Há dois conceitos chaves para entender bem o que vem a ser a Engenharia *Web*, são eles:

### 2.2.1. Agilidade de processo

Os sistemas *web* necessitam de agilidade em seu desenvolvimento, pois a demanda por atualizações frequentes e a evolução contínua dos requisitos são fatores críticos nesse tipo de aplicação. Basicamente se resume a ciclos rápidos de desenvolvimento, onde a cada ciclo há um incremento no sistema.

### 2.2.2. Arcabouço de processos

Cada ciclo de desenvolvimento precisa passar por um conjunto de atividades, e arcabouço de processos é justamente isso. A Figura 1 - que é adaptada do livro Engenharia *web* [BEDER 2012] - mostra todas essas atividades.



**Figura 1. Arcabouço de Processos. Fonte: Beder (2012)**

A primeira etapa é a comunicação, e consiste no completo entendimento do(os) problema(as) que o sistema se propõe a resolver, além de chegar a um acordo entre as partes interessadas. A etapa seguinte é o planejamento, que consiste em traçar um plano de desenvolvimento. A terceira etapa é a modelagem, que consistem em criar os respectivos diagramas, tais como os diagramas de caso de uso, de classe, do banco de dados e entre outros de acordo com a necessidade. A quarta etapa construção, que é a codificação em si e testes. E finalmente, a quinta etapa é a disponibilização do sistema para os usuários para *feedbacks*, tanto positivos quanto negativos.

### **2.3. Autenticação e segurança em sistemas *web***

A segurança é um dos pilares fundamentais no desenvolvimento de sistemas *web*, especialmente em plataformas que lidam com informações sensíveis. De acordo com [Maziero 2019], a autenticação visa verificar a autenticidade de uma entidade no sistema computacional. Em outras palavras, comprovar que as informações dessa entidade correspondem às informações do mundo real - como exemplo: em uma aplicação *web* com sistema de login, é essencial garantir que a pessoa que está tentando acessar a conta seja realmente ela mesma.

### **2.4. Kanban**

O Kanban surgiu na montadora Toyota [Espinha 2023] com a proposta do "*just in time*" - antes, muitos produtos eram produzidos de uma vez e sem a necessidade do mercado, causando desperdícios não só de tempo, mas principalmente de recursos. Com o surgimento do Kanban, os produtos passaram a serem produzidos por demanda, ou seja, apenas quando havia necessidade - poupando tempo e recursos.

Segundo [Espinha 2023], em japonês, Kanban significa "cartão", e consiste em controlar o fluxo de trabalho de uma maneira visual com o uso de cartões e colunas. O fluxo de trabalho é organizado em três estágios: tarefas a fazer, tarefas em andamento e tarefas finalizadas. As tarefas são movimentadas entre esses estágios à medida que são concluídas.

## **3. Trabalhos Correlatos**

Nesta seção serão apresentadas duas aplicações desenvolvidas na plataforma *web*, sendo abordadas as suas principais características, conforme o estudo feito entre elas.

### **3.1. Cãopanha: uma aplicação *web* para gerenciar denúncias contra maus-tratos em animais**

A autora [Coutinho 2022] traz como objetivo o desenvolvimento de uma aplicação *web* que gerencia denúncias de maus-tratos a animais, o processo é simples e conciso e visa minimizar os efeitos dessas práticas. Os métodos que foram utilizados estavam relacionados com a Engenharia de Software.

Uma das semelhanças desse trabalho com outros do gênero é o desenvolvimento de um sistema, desde a prototipação e a documentação até o desenvolvimento. A autora ainda destacou os materiais (dispositivos) utilizados. Também foram realizadas pesquisas sobre a temática de maus-tratos para levantar os requisitos iniciais. Essencialmente, o objetivo do site é facilitar o registro de denúncias.

Diferentemente do "Pet Denuncie" que além das funcionalidades da denúncia, possui um blog para a veiculação de informações, o "Cãopanha" busca apenas aplicar as funcionalidades da denúncia.

### 3.2. Análise e desenvolvimento de um sistema para registros, acompanhamentos e gerenciamentos de denúncias contra crimes ambientais do município de Paraúna – GO

A autora [Ferreira 2022] se diferencia deste trabalho por sua proposta de um sistema *web* voltado para o gerenciamento de denúncias contra crimes ambientais. O ponto central do sistema de crimes ambientais é promover uma maior facilidade de realizar uma denúncia - que é o mesmo intuito do "Pet Denuncie".

Há uma preocupação da autora com criar uma diagramação e levantamento de requisitos adequados. É presente também regras de negócio e um escopo de projeto, além da realização de testes com usuários finais - apesar dessa etapa ter ficado para "trabalhos futuros".

A Tabela 1 mostra uma comparação entre os três trabalhos.

**Tabela 1. Tabela comparativa de trabalhos correlatos.**

| Aspecto            | Pet Denuncie                                                                                                                            | Cãopanha                                                               | Sistema de gestão de crimes ambientais                               |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|----------------------------------------------------------------------|
| <b>Objetivo</b>    | Gerenciamento de denúncias contra casos de maus-tratos contra animais e um blog para veiculação de notícias e informações sobre o tema. | Gerenciamento de denúncias contra casos de maus-tratos contra animais. | Gerenciamento de denúncias contra crimes ambientais.                 |
| <b>Metodologia</b> | Metodologia Kanban.                                                                                                                     | Metodologia Scrum.                                                     | Não informado.                                                       |
| <b>Tecnologias</b> | NextJS com TypeScript, TailwindCSS, PostgreSQL, Postman, Autenticação JWT, Gimp, Canva, Kanban,                                         | HTML, CSS, PHP, MySQL, Visual Paradigm, Db Diagram e Notion.           | Astah UML, HTML, CSS, JS, MySQL, Bootstrap, PHP, Unsplash e Freepik. |

## 4. Metodologia e Materiais

Nesta seção, serão apresentadas as etapas para o desenvolvimento do projeto, além dos materiais que serão utilizados. Essa etapa refere-se ao segundo passo do arcabouço da Engenharia *Web*: o planejamento.

### 4.1. Extração e análise de requisitos

A definição dos requisitos foi feita considerando os trabalhos correlatos estudados. Foi feita uma comparação entre eles e uma análise sobre quais as dificuldades das pessoas em relação a fazer uma denúncia.

Mesmo não tendo uma interação com usuários reais, foi possível identificar essas questões para estabelecer os requisitos essenciais e não funcionais do projeto. Esses requisitos serão apresentados na próxima Seção.

## 4.2. Definição do gerenciador do Kanban

A metodologia Kanban foi escolhida para gerenciar o processo de desenvolvimento do sistema. O software escolhido para aplicá-lo foi o Obsidian juntamente com o plugin "Kanban", que oferece funcionalidades para criar colunas, criar tarefas, mover essas tarefas entre as colunas e criar anotações.

A Figura 2 destaca essas colunas com as respectivas tarefas feitas durante o desenvolvimento.

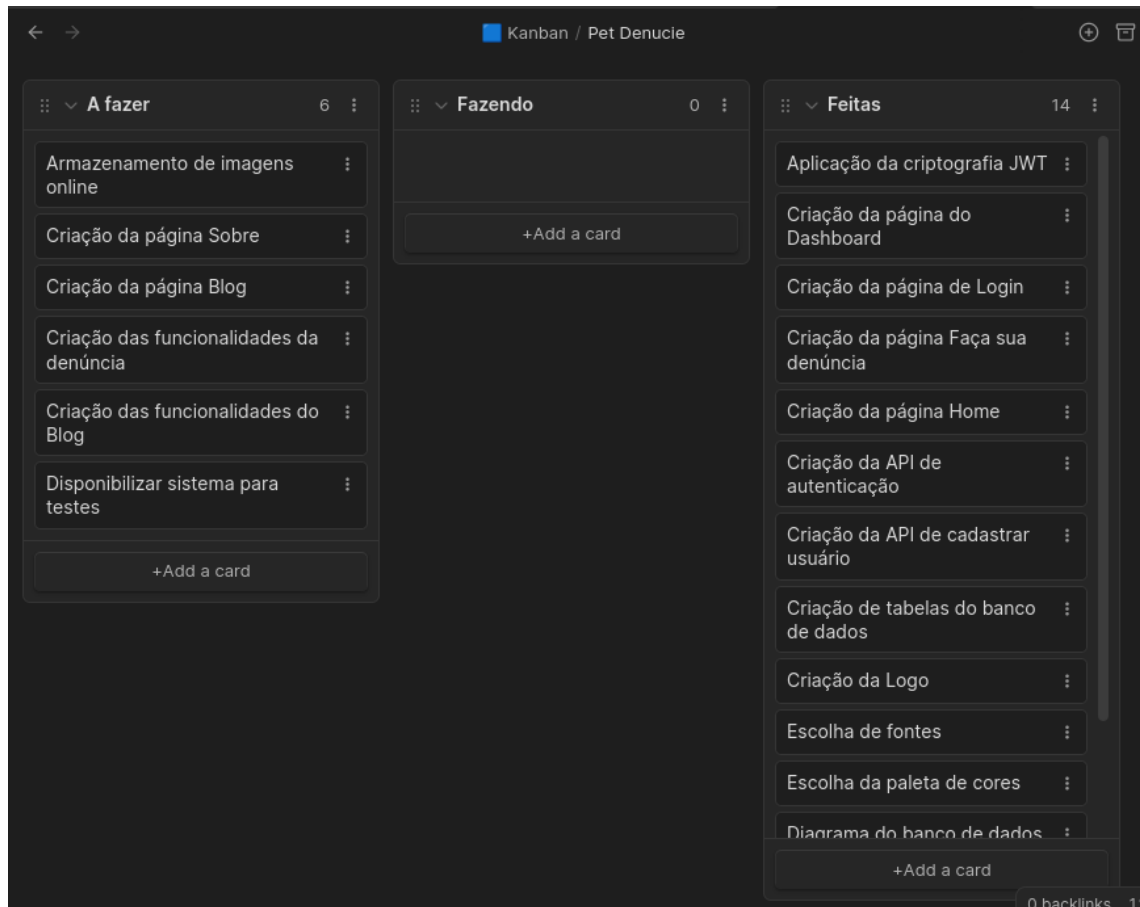


Figura 2. *Printfscreen* do Software Obsidian durante o desenvolvimento.

## 4.3. Escolha de tecnologias

Foi feita uma análise das tecnologias disponíveis que mais se adequam para o desenvolvimento do projeto, considerando aspectos positivos e negativos de cada uma. A Tabela 2 apresenta as tecnologias escolhidas.

**Tabela 2. Tecnologias escolhidas para o desenvolvimento do projeto.**

| Tecnologia            | Descrição                                           |
|-----------------------|-----------------------------------------------------|
| Canva e Gimp          | Para a criação da logo.                             |
| NextJS com TypeScript | Para o desenvolvimento do sistema web.              |
| TailwindCSS           | Para o design responsivo e estilização das páginas. |
| PostgreSQL            | Para a gestão e armazenamento dos dados.            |
| Postman               | Para teste e validação das APIs.                    |
| Obsidian              | Para aplicação do Kanban.                           |

É importante destacar que o estudo dos trabalhos correlatos e pesquisa bibliográfica permitiu a análise da viabilidade da execução do projeto.

## 5. Desenvolvimento

Esta Seção retrata as fases do desenvolvimento do projeto como um todo, sendo elas: levantamento de requisitos, criação dos diagramas de caso de uso e do banco de dados, escolha das fontes e paleta de cores, criação da logo, codificação das funcionalidades da API, criação das funcionalidades da denúncia e do blog.

### 5.1. Levantamento de requisitos

Conforme exposto anteriormente, foi realizado o levantamento dos requisitos funcionais e não funcionais para o desenvolvimento do projeto. Esta tarefa faz parte da comunicação do arcabouço de processo. As Tabelas 3 e 4 descrevem cada um deles, respectivamente.

**Tabela 3. Requisitos Funcionais para a aplicação Pet Denuncie.**

| Requisito     | Descrição                                                                      |
|---------------|--------------------------------------------------------------------------------|
| <b>RF - 1</b> | Permitir que um administrador tenha email e senha para logar na plataforma.    |
| <b>RF - 2</b> | Permitir que os usuários enviem denúncias, sejam anônimas ou identificando-se. |
| <b>RF - 3</b> | Permitir que os usuários acompanhem o status da denúncia.                      |
| <b>RF - 4</b> | Permitir que os usuários leiam artigos do blog.                                |
| <b>RF - 5</b> | Permitir que o administrador receba denúncias dos usuários.                    |
| <b>RF - 6</b> | Permitir que o administrador possa responder essas denúncias.                  |
| <b>RF - 7</b> | Permitir que o administrador altere o status dessas denúncias.                 |
| <b>RF - 8</b> | Permitir que o administrador escreva artigos para o blog.                      |

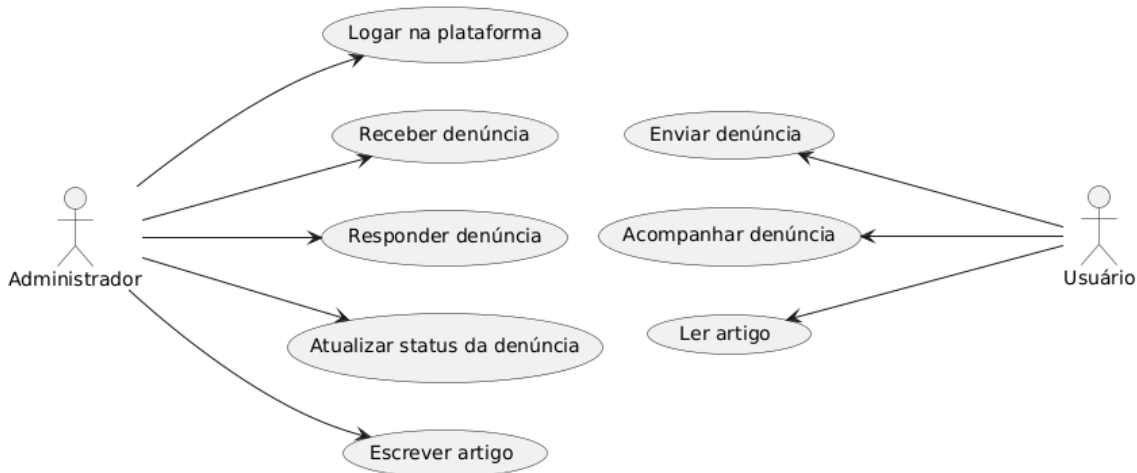
**Tabela 4. Requisitos Não Funcionais para a aplicação Pet Denuncie.**

| Requisito      | Descrição                                                                                                                                  |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>RNF - 1</b> | Interface do usuário deve ser simples, intuitiva e transmitir segurança.                                                                   |
| <b>RNF - 2</b> | O armazenamento de informações sensíveis do administrador deve ser feito de maneira segura, utilizando técnicas adequadas de criptografia. |

## 5.2. Modelagem

Nesta subseção serão apresentados os diagramas, sendo o diagrama de caso de uso e o do banco de dados. Essa tarefa faz parte da etapa de modelagem do arcabouço de processo.

A Figura 3 aborda o diagrama de caso de uso, com os atores e suas respectivas funções. Foram identificados dois atores: o administrador e o usuário do sistema, enquanto o administrador pode logar na plataforma, receber e responder denúncias, atualizar o status da denúncia e escrever artigos, o usuário pode enviar e acompanhar as denúncias, além de ler os artigos.



**Figura 3. Diagrama de caso de uso para a aplicação Pet Denuncie.**

A Figura 4 apresenta o diagrama do banco de dados, com suas tabelas, colunas, chaves e relacionamentos. A tabela "administrador" representa os dados do administrador do sistema, que tem o e-mail como chave primária. Vale destacar os campos de senha e *salt*. A senha é criptografada com o algoritmo de criptografia sha256 - que gera uma string de 64 caracteres - por isso o tamanho do varchar de 64 no banco. O *salt* é uma camada de criptografia extra gerada que será explicada na seção do desenvolvimento das API's.

A tabela "artigos" representa os artigos do blog, enquanto que a tabela "denuncias" representa as denúncias realizadas pelos usuários. A tabela "denunciante" representa os usuários que fizeram a denúncia. Vale destacar que, caso o usuário seja anônimo, os

campos de "email" e "nome" serão armazenados com a string "anônimo". Essa funcionalidade será explicada na subseção do desenvolvimento das páginas - mais precisamente na página de login.

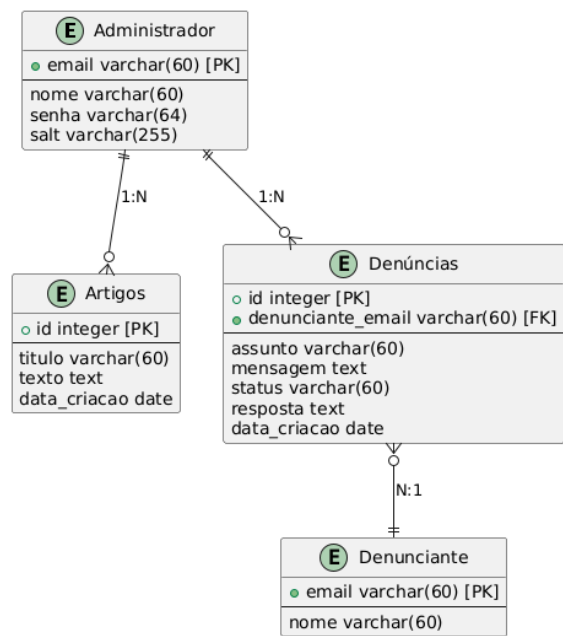


Figura 4. Diagrama do banco de dados para a aplicação Pet Denuncie.

Em relação aos relacionamentos, foi identificado que um administrador pode escrever vários artigos, mas um artigo só pode ser escrito por um administrador. Um administrador pode receber várias denúncias, mas uma denúncia só pode ser enviada a um administrador. E um denunciante pode escrever várias denúncias, mas uma denúncia só pode ser escrita por um denunciante.

5.3. Escolha das fontes e paleta de cores

Essa tarefa faz parte da etapa de planejamento do arcabouço da engenharia *web*, e como citado anteriormente, um dos requisitos do projeto é trazer segurança aos usuários. Nesse intuito, foi montada uma paleta de cores com tons de azul e cores sólidas - para [Criap 2024], o azul representa segurança, profissionalismo e estabilidade. As fontes escolhidas também foram escolhidas com esse propósito - sendo letras neutras. A Figura 5 expõe as cores e fontes escolhidas.

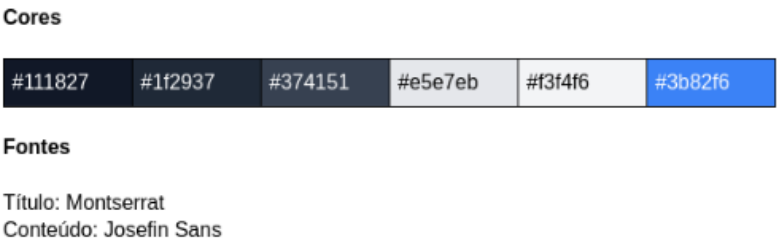


Figura 5. Paleta de cores e fontes do projeto.

## 5.4. Criação da logo

Essa tarefa faz parte da etapa de planejamento do arcabouço da engenharia *web*, como trata-se de um projeto para receber denúncias sobre maus-tratos a animais, foi criada uma logo com uma pata de um cachorro e um *headphone* - para trazer a ideia de que a denuncia está sendo recebida por alguém. A Figura 6 mostra essa logo.



Figura 6. Logo do projeto.

## 5.5. Codificação das funcionalidades da API

A partir dessa tarefa, a etapa do arcabouço de processos refere-se à construção. Nesta Subseção serão apresentadas duas funcionalidades da API: a funcionalidade de cadastro de um administrador e a funcionalidade para autenticação. A Figura 7 mostra o código da funcionalidade de cadastro.

```
1 export default async function handler(req: NextApiRequest, res: NextApiResponse) {
2
3   const { nome, email, senha } = req.body;
4
5   const conexao = await AppDataSource;
6   if (conexao) {
7     if (verificarDados(nome, email, senha)) {
8
9       const salt = Array.from(crypto.randomBytes(16));
10      const data = `${senha}${salt}`;
11      const hashSalt: string = crypto.createHash("sha256").update(data).digest("base64");
12
13      const repositorio = conexao.getRepository(Administrador);
14      const salvar = repositorio.create({
15        name: nome,
16        email: email,
17        senha: hashSalt,
18        salt: salt
19      });
20      try {
21        await repositorio.insert(salvar);
22        res.status(200).end();
23      } catch (erro) {
24        if (erro instanceof QueryFailedError && erro.message
25          .includes("duplicate key value violates unique constraint")) {
26          res.status(409).end();
27        } else {
28          throw erro;
29        }
30      }
31    } else {
32      res.status(400).end();
33    }
34  } else {
35    res.status(500).end();
36  }
37 }
```

Figura 7. Codificação do serviço de cadastro.

Basicamente, ela recebe como requisição o nome, e-mail e senha (linha 3) e tenta fazer uma conexão com o banco de dados (linha 5) e, caso não consiga, retorna o erro 500 *Internal Server Error* (linha 35). Caso consiga, chama a função `verificarDados` (linha 7), que apenas verifica se as informações recebidas são diferentes de vazio e, se não forem, retorna o erro 400 *Bad Request* (linha 32). Se forem, ele cria o *salt* (linha 9), que é um *hash* feito para aumentar a segurança. Em seguida, na linha 10, a senha digitada pelo usuário é concatenada com esse *salt*, e então é feita a criptografia com o algoritmo sha256 (linha 11). Após isso, é feita uma tentativa de salvar aqueles dados na tabela de usuário (linhas 14 a 21), retornando o status 200 *OK* caso consiga (linha 22). Caso não consiga salvar, o sistema verifica se o usuário já é cadastrado no banco e, se for esse o caso, é retornado o erro 409 *Conflict* (linha 26). Caso não seja o caso, o erro é jogado para frente com o status 500 (linha 28).

O segundo serviço programado na API é a de autenticação (Figura 8), que segue uma estrutura parecida com a funcionalidade anterior.

```
1 export default async function handler(req: NextApiRequest, res: NextApiResponse) {
2
3   const { email: emailUsuario, senha: senhaUsuario } = req.body;
4
5   const conexao = await AppDataSource;
6   if (conexao) {
7     if (verificarDados(emailUsuario, senhaUsuario)) {
8
9       const repositório = conexao.getRepository(Administrador);
10      const user = await repositório.findOneBy({ email: emailUsuario });
11
12      if (user == null) res.status(404).end();
13      else {
14        const dataUsuario = `${senhaUsuario}${user.salt?.map((item) => item)}`;
15        const hashSaltUsuario: string = crypto.createHash("sha256").update(dataUsuario)
16          .digest("base64");
17
18        if (hashSaltUsuario == user.senha) {
19
20          const chaveSecreta = process.env.CHAVE_SECRETA;
21
22          const token = jwt.sign(
23            { nome: user.name,
24              email: user.email
25            },
26            chaveSecreta!,
27            { expiresIn: "1h" });
28
29          res.status(200).json({ token, nome: user.name, email: user.email });
30        } else res.status(401).end();
31      }
32    } else {
33      res.status(400).end();
34    }
35  } else {
36    res.status(500).end();
37  }
38 }
```

**Figura 8. Codificação do serviço de autenticação.**

Como requisição, ela recebe o e-mail e senha do administrador (linha 3) - e segue lógica semelhante a da funcionalidade de cadastrar. Inicialmente tenta fazer a conexão com o banco de dados (linha 5), chama a função `verificarDados` (linha 7) e busca os dados do administrador na tabela de usuários (linha 9). Então, é verificado se aquele e-mail está cadastrado (linha 10) e, caso não esteja, é retornado o erro 404 *Not Found* (linha 12). Caso esteja, é feito o mesmo processo de criptografia feito na API de cadastro (linhas 15 e 16) e, caso o resultado da criptografia seja diferente do que está salvo no banco, é retornado o status 401 *Unauthorized* (linha 30). Caso o resultado seja o mesmo, é criado um token JWT e retornado o status 200 *OK* com o token, nome e e-mail do usuário (linha 29).

Além dessas, também foram desenvolvidas outras que seguem a mesma estrutura: funcionalidades para cadastro de denúncia, de denunciante, de artigo e para enviar e-mail. Porém, vale destacar a funcionalidade de enviar e-mail que foi utilizada o pacote `Nodemailer` do NPM (Figura 9). Inicialmente foi instalado o pacote `Nodemailer` com o comando `npm install nodemailer`. Depois disso, configurado uma senha para apps da conta do Google e colocando no arquivo `.env.development`, um arquivo ignorado pelo git na hora de enviar para o repositório online. Essa API recebe como parâmetro os valores `to` (para quem o e-mail será enviado), `subject` (o assunto) e `text` (- que é o texto enviado no corpo do e-mail) (linha 2). Então, é chamada a função `createTransport` do `Nodemailer` (linhas 7 a 13) que recebe o provedor, e-mail e senha do e-mail remetente. Após isso, é chamada a função `sendEmail` (linhas 16 a 21) que recebe o remetente, destinatário, assunto e texto e, se der certo, o e-mail é enviado e retornado o status 200 na API (linha 23). Se não, o erro é jogado para frente com o uso do `try-catch` (linhas 24 a 27).

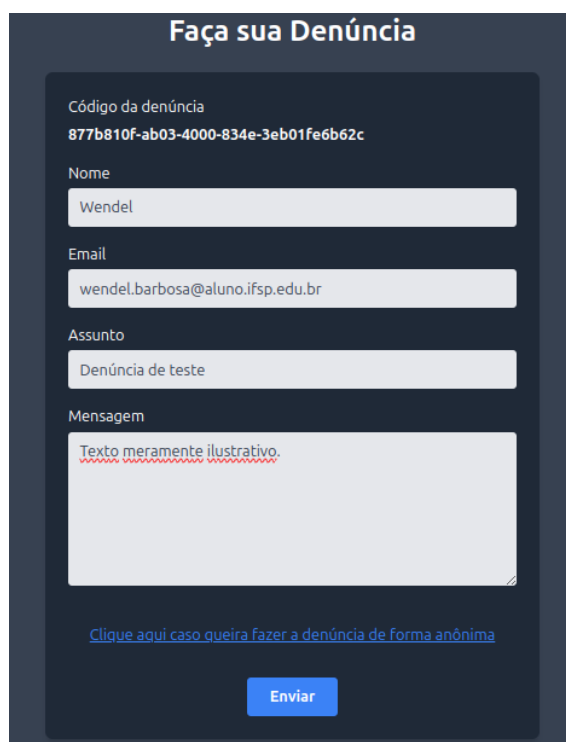
```
1 export default async function handler(req: NextApiRequest, res: NextApiResponse) {
2   const { to, subject, text } = req.body;
3   if (!to || !subject || !text) {
4     return res.status(400).json({ message: 'Parâmetros de email incompletos' });
5   }
6
7   const transporter = nodemailer.createTransport({
8     service: 'gmail',
9     auth: {
10      user: process.env.EMAIL_USER,
11      pass: process.env.EMAIL_PASSWORD,
12    },
13  });
14
15  try {
16    await transporter.sendMail({
17      from: process.env.EMAIL_USER,
18      to,
19      subject,
20      text,
21    });
22
23    res.status(200).json({ message: 'Email enviado com sucesso!' });
24  } catch (error) {
25    console.error('Erro ao enviar email:', error);
26    res.status(500).json({ message: 'Erro ao enviar email', error });
27  }
28 }
```

Figura 9. API de enviar e-mail.

## 5.6. Funcionalidades da denúncia

Como citado anteriormente, foram desenvolvidas três API's para as funcionalidades da denúncia: cadastro de denúncia, de denunciante e de envio de e-mail. Além dessas, foram desenvolvidas outras duas: uma para listar todas as denúncias, e outra para trazer uma denúncia pelo *id*.

Na página "faça sua denúncia"(Figura 10), há um formulário com um botão de rótulo "Enviar"para cadastrar a denúncia. Ao clicar no botão, os dados escritos pelo usuário são enviados para a API de cadastro de denúncia e, caso seja concluída com sucesso, o status 200 é retornado e um *feedback* para o usuário é o informando. Também aparecerá no formulário o código daquela denúncia. Caso o usuário queira fazer a denúncia de forma anônima, há um botão escrito "Clique aqui caso queira fazer a denúncia de forma anônima". Ao clicar nesse botão, os campos "nome"e "email"desaparecem. Porém, ao optar por realizar a denúncia dessa maneira, o usuário não receberá e-mail em nenhum momento. Caso o usuário opte por se identificar, será enviado um e-mail com os dados da denúncia.

A imagem mostra a interface de uma página web intitulada "Faça sua Denúncia". O formulário é preenchido com os seguintes dados: "Código da denúncia" com o valor "877b810f-ab03-4000-834e-3eb01fe6b62c"; "Nome" com "Wendel"; "Email" com "wendel.barbosa@aluno.ifsp.edu.br"; "Assunto" com "Denúncia de teste"; e "Mensagem" com "Texto meramente ilustrativo.". Abaixo do campo de mensagem, há um link azul que diz "Clique aqui caso queira fazer a denúncia de forma anônima". No rodapé do formulário, há um botão azul com o texto "Enviar".

**Figura 10. Página faça sua denúncia.**

Dentro do *dashboard*, na primeira página há um informativo de quantas denúncias foram recebidas e quantos artigos escritos (o assunto dos artigos será tratado na próxima Subseção). Foi criada a página "ver denúncias"(Figuras 11 e 12), que lista todas as denúncias cadastradas no banco de dados (separadas em denúncias recebidas, respondidas e aguardando aprovação, para facilitar o gerenciamento). Ao clicar em uma denúncia, o administrador é redirecionado para uma página para ver mais detalhes dessa denúncia, onde é possível responder, além de um botão para enviar um e-mail para o(a) denunciante.

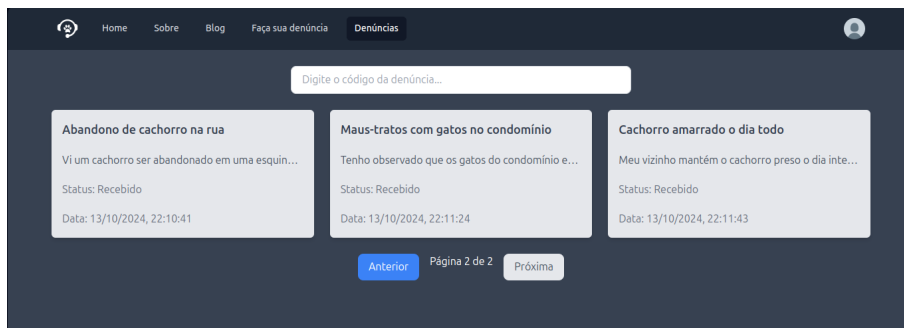


**Figura 11. Página de ver denúncias exibindo denúncias recebidas.**



**Figura 12. Página de ver denúncias exibindo denúncias respondidas e aguardando aprovação.**

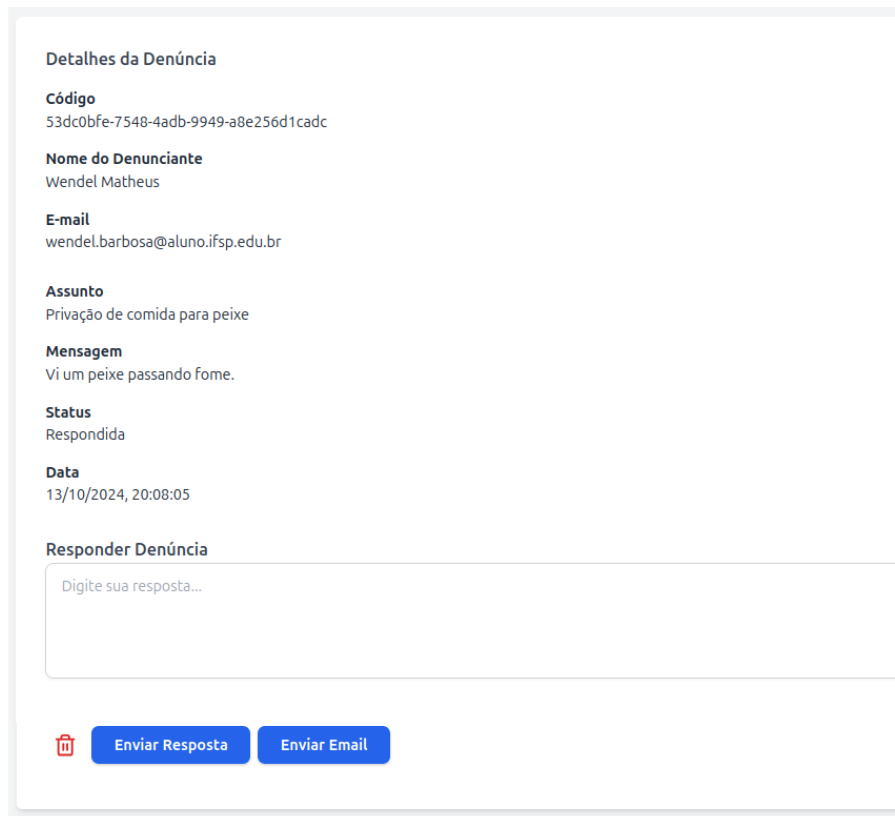
Finalmente, a página denúncias exibe todas as denúncias feitas (utilizando a mesma API para buscar no banco). Essa página contém uma paginação, permitindo que no máximo 9 denúncias apareçam em tela (Figura 13). Há uma barra de pesquisa no topo da página para inserir o código da denúncia e encontrá-la rapidamente. Ao clicar em uma denúncia, o usuário é redirecionado para uma página com mais detalhes. O id da denúncia é passado como parâmetro na URL.



**Figura 13. Página denúncias exibindo paginação e barra de pesquisa.**

Vale lembrar que uma denúncia pode ter 3 status: recebida, respondida e aguardando aprovação. Quando uma denúncia é feita, tem seu status como "aguardando

aprovação”e não aparece na página de denúncia, aparecendo apenas para o administrador, que pode aprovar ou apagar. Ao aprovar, a denúncia tem seu status alterado para ”recebida”e já aparece na página de denúncias para os usuários. Então, o administrador pode responder essa denúncia e mandar um e-mail para o usuário com a resposta (por meio de um botão que aparece na página para responder a denúncia) - Figura 14.



Detalhes da Denúncia

**Código**  
53dc0bfe-7548-4adb-9949-a8e256d1cadc

**Nome do Denunciante**  
Wendel Matheus

**E-mail**  
wendel.barbosa@aluno.ifsp.edu.br

**Assunto**  
Privação de comida para peixe

**Mensagem**  
Vi um peixe passando fome.

**Status**  
Respondida

**Data**  
13/10/2024, 20:08:05

**Responder Denúncia**

Digite sua resposta...

 **Enviar Resposta** **Enviar Email**

Figura 14. Página para responder denúncia.

## 5.7. Funcionalidades do blog

Na página ”Blog”é possível visualizar todos os artigos escritos pelo administrador (Figura 15), que possui também um recurso de paginação (Figura 16) que permite que até 9 artigos apareçam na tela por vez para evitar que o site fique lento por carregar muitas informações do banco de dados.



Figura 15. Página do blog exibindo artigos.



Figura 16. Paginação do blog.

Dentro do *dashboard*, o administrador pode visualizar os artigos (Figura 17), apagá-los (Figura 18), além de escrever um novo artigo (Figura 19).

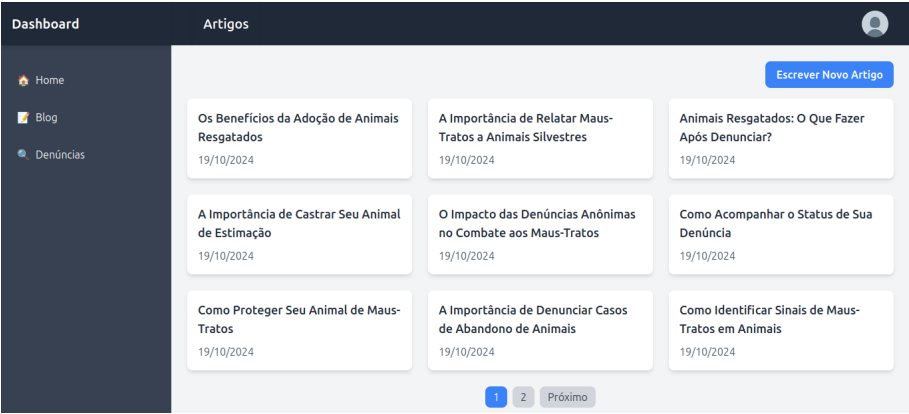


Figura 17. Gerenciador de artigos do dashboard.

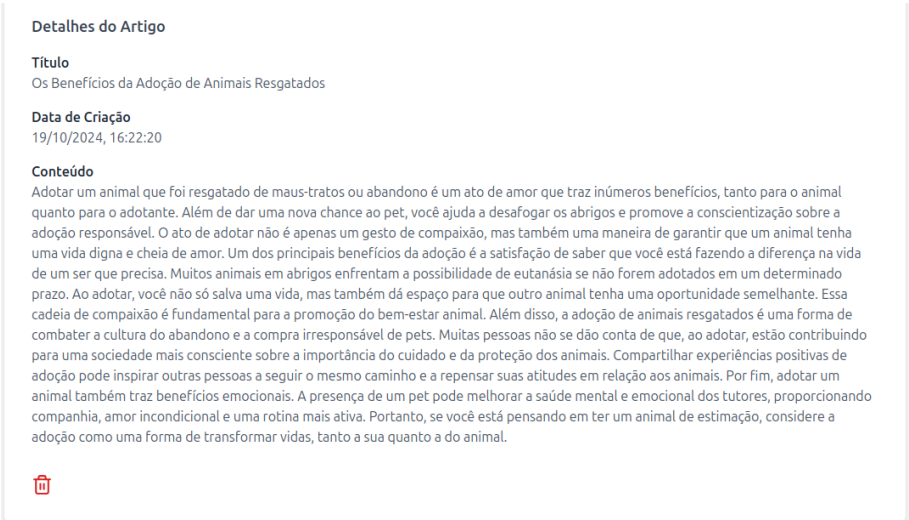
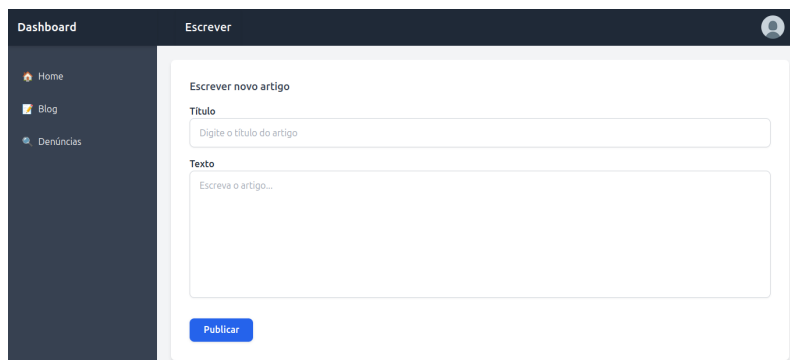


Figura 18. Página para ler ou apagar um artigo.



**Figura 19. Página para escrever um novo artigo.**

## 6. Resultados

Esta Seção apresenta os resultados obtidos ao longo do desenvolvimento do projeto.

Os objetivos propostos para esse trabalho foram concluídos com sucesso.

### 6.1. Checklist de Funcionalidades Atendidas

- Login no sistema;
- Realização de denúncias;
- Receber informações da denúncia por e-mail;
- Pesquisar denúncia realizada;
- Administrador receber denúncias;
- Administrador escrever artigos;
- Administrador aprovar, apagar, responder denúncias e enviar e-mail;
- Administrador apagar artigos.

O framework NextJS foi escolhido para o desenvolvimento do sistema por ser capaz de contruir tanto as páginas quanto as API's, trazendo modernidade e desempenho para a aplicação. O TailwindCSS foi escolhido por ter boa integração com o NextJS e ser recomendado pela equipe do framework.

## 7. Conclusão

O objetivo ao longo do desenvolvimento deste projeto foi criar uma plataforma digital que possibilitasse o processo de denúncia de casos de maus-tratos a animais. Para isso, foi optado por centralizar e simplificar esse processo em um site com o uso de tecnologias atuais para garantir uma boa experiência aos usuários.

Destaca-se a estrutura do projeto que atendeu perfeitamente aos objetivos propostos. Adicionar testes beta é uma boa forma de melhorar o projeto - algo que não foi feito.

### 7.1. Trabalhos Futuros

Para os próximos passos, é importante implementar funcionalidades adicionais que enriqueçam a experiência do usuário, como a possibilidade de adicionar comentários nos artigos do blog pelos usuários. Essa função proporcionaria uma oportunidade de criar discussões e propagar ainda mais informações.

Outra função é possibilitar a criação de logins para diversas autoridades utilizarem o sistema. Ou seja, cada administrador cadastrado terá seu próprio domínio e informações do banco de dados, como denúncias e artigos - além de uma possibilidade de personalização do site dentro do painel administrativo. Essa funcionalidade possibilitaria escalar o sistema para outros lugares.

Seguindo a linha de raciocínio de escalar o sistema, outro fator que vale a pena ser considerado é a criação de um aplicativo para dispositivos móveis - que facilitaria ainda mais a gestão das denúncias e dos artigos.

Vale ressaltar a importância de buscar um aprimoramento contínuo das funções e detecção das necessidades do mercado e dos usuários, além do estudo das tecnologias que surgem ao longo do tempo.

Uma próxima etapa para a continuidade desse projeto é a implantação de um teste beta - que consiste em disponibilizar o projeto para um grande número de pessoas, ao invés de apenas testes informais [Carvalho 2019].

Durante a formação do Curso Superior em Análise e Desenvolvimento de Sistemas, a aplicação de conceitos das disciplinas: Linguagem de Programação I e II, Desenvolvimento Web I e II, Engenharia de Software, Metodologias Ágeis, Análise Orientada a Objetos, Programação Orientada a Objetos e Banco de Dados forneceram os conhecimentos necessários para o desenvolvimento de todas as etapas desse projeto.

## 8. Referências

BEDER, Delano Medeiros. 2012. Engenharia Web: uma abordagem sistemática para o desenvolvimento de aplicações web. São Carlos: EdUFSCar, 2012.

BORBA, Juliana. 2020. Combate aos maus-tratos a animais domésticos. Disponível em: <https://www.folhaunica.com.br/unico-educacional/2020/12/juliana-borba/combate-aos-maus-tratos-a-animais-domesticos/> . Acesso em 11 de maio de 2024.

CARVALHO, Ingrid. Quais são as funções do Teste Alfa, Beta, e Regressão?. Disponível em: <https://medium.com/@ingrid.carvalho.mo/quais-s>

COUTINHO, Evelyn. CÃOPANHA: uma aplicação web para gerenciar denúncias contra mau-tratos em animais. Disponível em: <https://josif.ifsuldeminas.edu.br/ojs/index.php/anais/article/view/73/209>. Acesso em 24 de março de 2024.

CRIMLAB. 2024. Maus-tratos aos animais. Disponível em: <https://www.crimlab.com/dicionario-criminologico/maus-tratos-aos-animais/47>. Acesso em 11 de maio de 2024.

ESPINHA, Roberto Gil. 2023. Kanban – Guia completo para gerenciar fluxos de trabalho. Disponível em: <https://artia.com/blog/kanban/>. Acesso em 24 de julho de 2024.

FERREIRA, Jéssica. Análise e Desenvolvimento de Um Sistema para Registros, Acompanhamentos e Gerenciamentos de Denúncias contra Crimes Ambientais do Município de Paraúna - Go. Disponível em: <https://repositorio.ifgoiano.edu.br/handle/prefix/2369>. Acesso em 24 de março de 2024.

2024.

FLÁVIO, Lúcio. Maltratar animais é crime, veja como denunciar. Disponível em: <https://www.agenciabrasilia.df.gov.br/2022/10/08/maltratar-animais-e-crime-veja-como-denunciar/>. Acesso em 24 de março de 2024.

G1 SP. Denúncias de maus-tratos a animais crescem 15,6% em 2021, em SP. Disponível em: <https://g1.globo.com/sp/sao-paulo/noticia/2022/01/10/denuncias-de-maus-tratos-a-animais-crescem-156percent-em-2021-em-sp.ghtml>. Acesso em 24 de março de 2024.

INSTITUTO CRIAP. 2024. O que é a Psicologia das Cores?. Disponível em: <https://www.institutocriap.com/blog/psicologia/psicologia-cores>. Acesso em 11 de dezembro de 2024.

MAZIERO, Carlos. 2019. Autenticação. Disponível em: <https://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=sc:seg-texto-05.pdf>. Acesso em 4 de outubro de 2024

NIELSEN, Jakob. Usability Engineering. Morgan Kaufmann, 2012.

PETZ. 2024. Cachorro tem sentimento? Veja o que a ciência diz sobre o assunto. Disponível em <https://www.petz.com.br/blog/bem-estar/caes-bem-estar/cachorro-tem-sentimentos/>. Acesso em 11 de maio de 2024.

RODRIGUES, Gizella. 2021. Maus-tratos a animais: mais de mil denúncias em 2021. Disponível em: <https://www.agenciabrasilia.df.gov.br/2021/03/15/maus-tratos-a-animais-mais-de-mil-denuncias-em-2021/>. Acesso em 2 de junho de 2024.

# Documento Digitalizado Público

## Anexo I - artigo - TCC

**Assunto:** Anexo I - artigo - TCC  
**Assinado por:** Andre Constantino  
**Tipo do Documento:** Relatório Externo  
**Situação:** Finalizado  
**Nível de Acesso:** Público  
**Tipo do Conferência:** Documento Digital

Documento assinado eletronicamente por:

- Andre Constantino da Silva, PROFESSOR ENS BASICO TECN TECNOLOGICO, em 26/02/2025 00:22:26.

Este documento foi armazenado no SUAP em 26/02/2025. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse <https://suap.ifsp.edu.br/verificar-documento-externo/> e forneça os dados abaixo:

**Código Verificador:** 1951280  
**Código de Autenticação:** abfd4b78f8

