

PÓS-GRADUAÇÃO EM DESENVOLVIMENTO FULL STACK

**BLUEY – PLATAFORMA ONLINE PARA
DIVULGAÇÃO DE INFORMAÇÕES
SOBRE ANIMAIS DE ESTIMAÇÃO**

MIDGEL RIBEIRO BORGES

Orientador: Nome do Orientador

2025



SUMÁRIO

1. CONTEXTUALIZAÇÃO DA PROPOSTA	3
2. OBJETIVOS DA CONSTRUÇÃO DA SOLUÇÃO	4
2.1. OBJETIVOS ESTRATÉGICOS	4
2.2. OBJETIVOS ESPECÍFICOS.....	4
3. ELABORAÇÃO DA JORNADA DO USUÁRIO	5
4. APELO MERCADOLÓGICO DA SOLUÇÃO.....	7
5. CICLO DE DESENVOLVIMENTO DA SOLUÇÃO.....	9
6. MOCKUP DA PROPOSTA DE SOLUÇÃO.....	11
7. ARQUITETURA DE SOFTWARE	17
7.1. Componentes	17
7.1.1. <i>Front-end</i>	17
7.1.2. <i>Back-end</i>	17
7.1.3. <i>Camada de dados</i>	17
7.2. Infraestrutura.....	18
7.2.1. <i>Containerização</i>	18
7.2.2. <i>Infraestrutura em Nuvem (AWS)</i>	18
7.3. Pipeline CI/CD.....	18
7.3.1. <i>Git/Github</i>	19
7.4. Design Patterns	19
7.4.1. <i>MVC Laravel</i>	19
7.5. Integrações	19
7.5.1. <i>Back-end e Front-end</i>	19
7.5.2. <i>Back-end e Banco de Dados (Mysql)</i>	19
7.5.3. <i>BLUEY e Serviços AWS.</i>	20
7.6. Segurança.....	20
7.7. Diagramas.....	20
7.7.1. <i>Diagrama de arquitetura em camadas</i>	20

7.7.2. Diagrama de infraestrutura em nuvem.	20
7.7.3. Diagrama de pipeline ci/cd	21
8. VALIDAÇÃO DA SOLUÇÃO	22
9. REGISTROS DAS EVIDÊNCIAS DO PROJETO	27
10. CONSIDERAÇÕES FINAIS E EXPECTATIVAS.....	35
REFERÊNCIAS.....	36

1. CONTEXTUALIZAÇÃO DA PROPOSTA

Os animais de estimação estão presentes em quase todas as residências brasileiras, segundo uma pesquisa divulgada pela ABINPET – ASSOCIAÇÃO BRASILEIRA DA INDÚSTRIA DE PRODUTOS PARA ANIMAIS DE ESTIMAÇÃO em 2024, em todo o país são pelo menos 167 milhões de animais compartilhando o lar com os seres humanos, dos quais 67 milhões são cães, 33 milhões são gatos, 41 milhões são aves e 22 milhões são peixes. Seguindo essa linha, podemos refletir sobre a criação de um ambiente compartilhado para que os donos possam divulgar informações úteis ou histórias sobre seus animais de estimação, afinal todos podemos ter histórias incríveis para compartilhar com outros donos de “PET’S”.

O nome “BLUEY” escolhido para representar a aplicação, vem de um incrível cão reconhecido pelo Guinness Book Records como “O cachorro mais velho do Mundo”, Bluey foi um cão que viveu por 29 anos e 5 meses em uma fazenda na Austrália. Então para homenagearmos a longevidade desse animal e buscar instruir donos de animais a cuidar bem de seus companheiros para que vivam mais, cria-se a “BLUEY - PLATAFORMA ONLINE PARA DIVULGAÇÃO DE INFORMAÇÕES SOBRE ANIMAIS DE ESTIMAÇÃO”.

2. OBJETIVOS DA CONSTRUÇÃO DA SOLUÇÃO

2.1. OBJETIVOS ESTRATÉGICOS

- a. Aprimorar conhecimento sobre a cultura DevOps.
- b. Aplicar conceitos de computação em nuvem na prática.
- c. Aprimorar Design UI com responsividade.
- d. Desenvolver uma aplicação em formato full-stack.
- e. Melhorar conhecimento sobre a linguagem PHP com framework Laravel.

2.2. OBJETIVOS ESPECÍFICOS

- a. Realizar configuração de um ambiente devops com containers utilizando a infraestrutura da AWS com integração ao Github.
- b. Efetuar configuração de ambiente AWS para alocação de imagens com o S3 bucket.
- c. Utilizar de bibliotecas voltadas ao PHP e Laravel que auxiliem na responsividade da aplicação.
- d. Conhecer e aplicar todos os passos para construção de uma aplicação desde seu início ao fim, contemplando desenvolvimento back-end e front-end com PHP, manipulação de banco de dados Mysql, versionamento utilizando o Git/Github, armazenamento de imagens com Amazon Aws e hospedagem de aplicação via Load Balancer AWS.
- e. Estudar uso da linguagem PHP tanto em back-end quanto em front-end com uso de responsividade para dispositivos móveis, aplicando conceitos junto ao framework Laravel.

3. ELABORAÇÃO DA JORNADA DO USUÁRIO

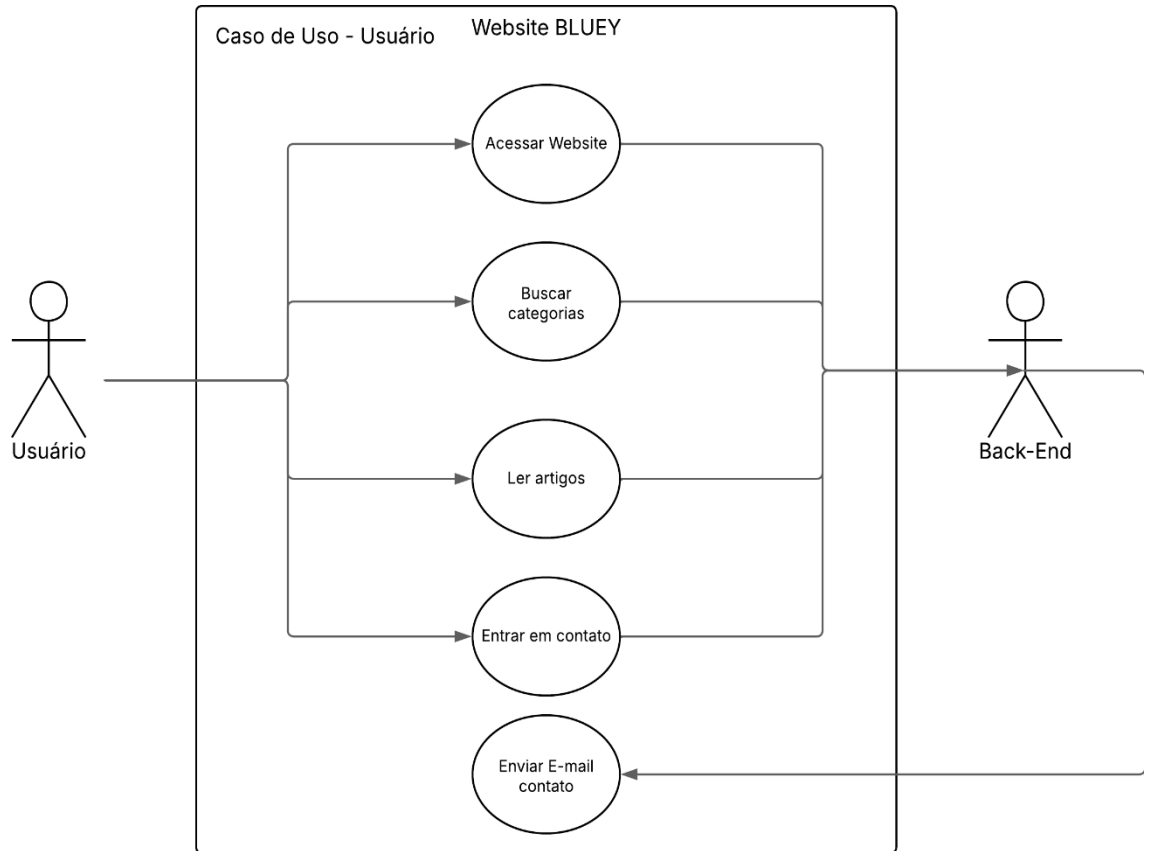
A jornada do usuário é imprescindível para descrição do uso da aplicação, ou seja, nela descrevemos como os usuários do sistema irão se comportar frente as opções disponíveis dentro do escopo definido.

Neste projeto trazemos três atores principais, são eles:

- Usuário – membro ao qual a aplicação é destinada, este possui as características de acessar o site, buscar por categorias, ler artigos e entrar em contato através de e-mail.
- Administrador – membro responsável por efetuar administração sistêmica da aplicação, efetuando manutenção da plataforma internamente, efetuando postagens, criando categorias e efetuando contato com os usuários.
- Back-end – parte da aplicação que processa as informações vinda das solicitações de usuários e administradores.

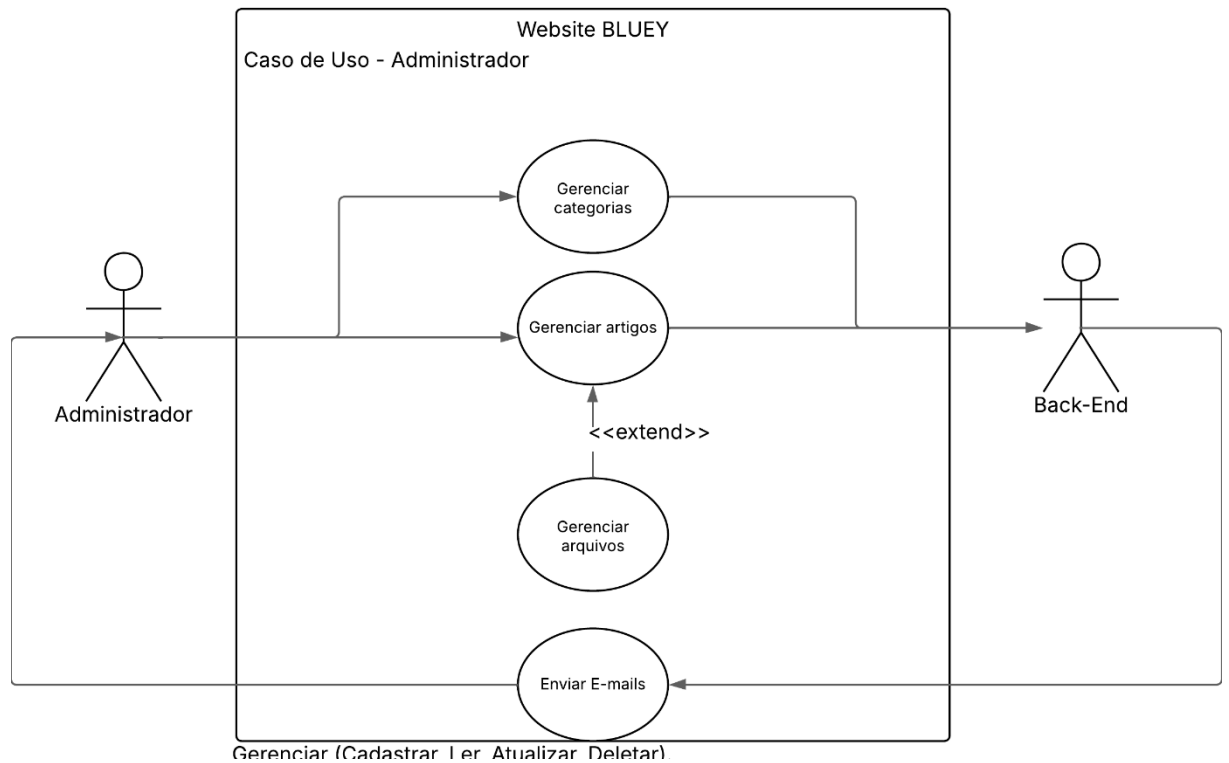
Abaixo é possível ter uma visualização gráfica expressa em imagem, sobre como os atores se relacionam com a aplicação BLUEY.

Casos de uso de usuário:



Fonte: o autor.

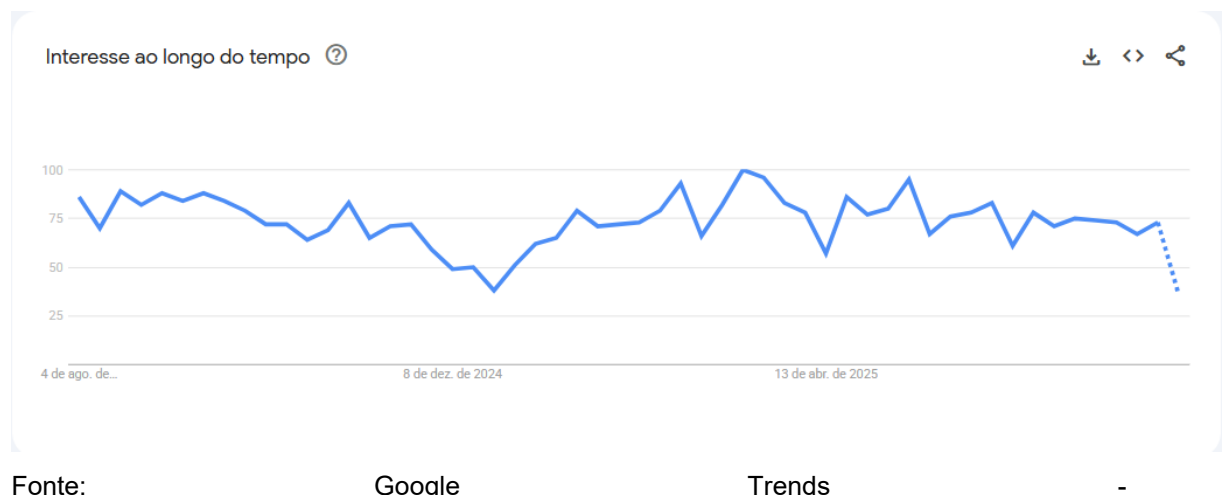
Caso de uso administrador:



Fonte: o autor.

4. APELO MERCADOLÓGICO DA SOLUÇÃO

Conforme citado anteriormente, o mercado PET no Brasil tem crescido e as dúvidas sobre a melhor forma de tratar as mascotes pode surgir em seus donos. Por exemplo a palavra “PET” vinda da sigla inglês utilizada para designar um animal carinhoso próximo, teve em território brasileiro no motor de busca Google, mais de quatro mil pesquisas entre agosto de 2024 e agosto de 2025.



<https://trends.google.com.br/trends/explore?geo=BR&q=bem%20estar%20animal&hl=pt-BR>

Ao optarmos por trabalhar com base em uma solução voltada para o público dono de animais domésticos, buscamos disseminar informações relevantes para possibilitar a longevidade dos companheiros. Há no mercado brasileiro e mundial diversos portais com foco em bem estar animal, por exemplo podemos citar o SPECIAL DOG, este possui abas relacionadas a nutrição animal, saúde e guias para melhorar sua relação com os animais domésticos; Sua aba de nutrição conta com diversos artigos, mencionando dietas específicas para diversos tipos de animais, informando também quais alimentos são nocivos à espécies específicas de cães; A aba de saúde promove artigos relacionados a hábitos e precauções sobre o mundo animal, mencionando formas de proteger e precaver doenças muito comuns em estações ou locais, a ideia geral é repassar informações sobre cuidados alimentares,

hábitos e estresse; A aba de guia do pet é bastante abrangente, contando com artigos que varia entre diversas áreas, sendo eles relacionado desde o cuidado e até ao meio jurídico ensinando como registrar seu animal.

Em se tratando de formas de divulgação, podemos citar o canal do Youtube denominado Perito Animal, o perfil na rede da companhia Google, conta com mais de um milhão de inscritos, mais de mil vídeos e cerca de duzentas milhões de visualizações em seu conteúdo. Os vídeos variam de informações relevantes, curiosidades, brinquedos, produtos, alimentação, nutrição, diversão, entretenimento, sentimentos, histórias e várias outras formas de interação com as mascotes; também é possível encontrar o conteúdo em redes sociais como Instagram, Facebook, Twitter e Pinterest. Este perfil é um exemplo de como as pessoas se interessam pelo conteúdo atrelado a animais domésticos e sua vasta gama de informações, associando essa necessidade à criação do BLUEY, podemos validar a importância da disseminação de conteúdo para pessoas pertencentes ao público alvo.

Podemos definir o público alvo da aplicação por:

- Tutores – pessoas que já possuem animais domésticos e buscam informações sobre como cuidar destes;
- Futuros tutores – pessoas que ainda não possuem o animal, mas que desejam tê-lo e, portanto, estão buscando informações para planejamento e estudo.
- Voluntários – pessoas que desejam disseminar informações e conteúdos relacionados e encontrar outros de seu grupo.
- Profissionais – veterinários, adestradores e outros especialistas quais almejam de alguma forma contribuir para o bem estar animal.

Em relação ao modelo de negócio da aplicação, podemos pensar em marcas relevantes que possam atuar em parceria com a aplicação, trabalhando no formato de marketing de afiliados ou conteúdo patrocinado, por exemplo, divulgação de ração, produtos para entretenimento, serviços e outros. Assim os interessados poderiam entrar em contato com administração e ambos poderiam beneficiar-se da vitrine virtual que o site dispõe.

5. CICLO DE DESENVOLVIMENTO DA SOLUÇÃO

O ciclo de desenvolvimento de um software é relevante para termos assertividade na conclusão do projeto como um todo, dessa forma podemos definir as seguintes etapas:

- Planejamento – essa etapa consiste na investigação e organização das atividades a serem elaboradas atingirmos nosso objetivo, para tal, faremos uso da plataforma online Trello, esta permite uma separação clara de tarefas e prazos para que sejam executadas, possibilitando descrições detalhadas das atividades.
- Desenvolvimento – utilizando como IDE (Ambiente de Desenvolvimento Integrado) o Visual Studio Code fornecido pela Microsoft, faremos também uso da linguagem PHP (PHP Processador de Hipertexto) para Back-End com integração de jQuery, uma biblioteca Javascript e o Bootstrap uma biblioteca PHP para auxílio em responsividade, como framework para auxílio no desenvolvimento junto ao PHP valer-se-á do Laravel.
- Validação – consiste na camada de testes, buscando falhas no desenvolvimento para que possam ser corrigidos e disponibilizados da melhor forma ao usuário final.
- Implantação – é a etapa final, após executado todo o desenvolvimento e testes na aplicação, fazemos o deploy para ambiente de produção a fim de lançar o produto.

No tangente à gestão de versões do projeto será efetuado através do Git utilizando o Github, este nos permite ter um controle de código extremamente eficiente, tornando o conjunto de desenvolvimento mais auditável e seguro. Para tal criaremos Branches, canais diferentes para cada ambiente, sendo uma para ambiente de homologação onde efetuaremos criação e teste da aplicação, e outra para ambiente de produção, este é a versão final disponível para o consumidor. O Github

possui histórico de uploads de atualizações denominados “Commit’s”, ou seja, toda vez que submetemos um código geramos um commit, essa funcionalidade auxilia no controle de versão facilitando a identificação dos códigos injetados. Sua funcionalidade “Pull Request” serve para que migremos código de homologação para produção, tornando o deploy mais fluido.

O desenvolvimento inicial do projeto será feito localmente utilizando Apache, este simula um ambiente HTTP na máquina do usuário, permitindo testes com maior agilidade e praticidade. A fim de simularmos o ambiente de produção será introduzido o Docker Desktop, esta ferramenta permite criação de cluster Docker local, separando quase totalmente o ambiente onde a aplicação está rodando, da máquina física do usuário. O uso do deploy containerizado do PHP busca trazer maior consistência e escalabilidade para aplicação, tornando o ambiente independente para manutenções facilitadas.

A estratégia utilizada para deploy da aplicação será o “Code Review”, esta técnica consiste na revisão do código submetido, avaliando suas alterações e como ele irá impactar na aplicação, caso aprovado o código será enviado para produção, caso reprovado, o código deverá ser revisado, alterado e novamente enviado para posterior avaliação. Vale lembrar que este método é internamente implementado pelo Github através de sua estrutura de “Pull Request” na qual o código submetido para envio a produção é previamente avaliado por um responsável que pode fazer suas colocações para posterior “Merge” em produção.

6. MOCKUP DA PROPOSTA DE SOLUÇÃO

A idealização de uma interface visual conta com diversos elementos gráficos distribuídos em uma página Web. Seguindo o padrão informado pela MDN Web Docs, uma construção digital pode conter os seguintes recursos:

- Cabeçalho (header) - parte inicial do site situada ao topo e onde emprega-se a logotipo com informações essenciais sobre a aplicação.
- Barra de navegação - aqui se situam os itens de navegação do usuário, por onde este deverá transitar entre as possíveis abas.
- Conteúdo principal (body) - aqui se concentram as informações alvo, é onde o usuário de fato irá atuar, sendo efetuando leitura ou executando ações como o preenchimento de formulários por exemplo.
- Barra lateral (sidebar) - guia extra situada nas laterais do corpo principal, podendo ser alguma informação útil ao contexto do body.
- Rodapé (footer) - situado ao fim da aplicação, o footer é usado para alocar informações gerais, como por exemplo contatos e link úteis.

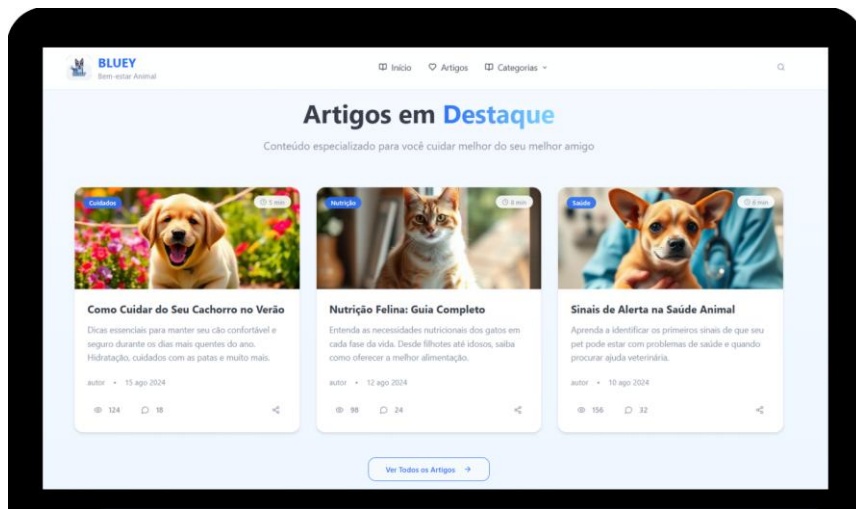
Estes são os elementos principais, mas que podem variar de acordo com a natureza do site. Aqui apresento o mockup inicial do BLUEY, disponibilizando uma visão primária do que o projeto virá a ser, em um tamanho de tela para computadores (desktop) e telas móveis (mobile), a prototipação de telas foi construída utilizando as ferramentas online Figma, Canva e Lovable.

Home page BLUEY – Página inicial, versão desktop.



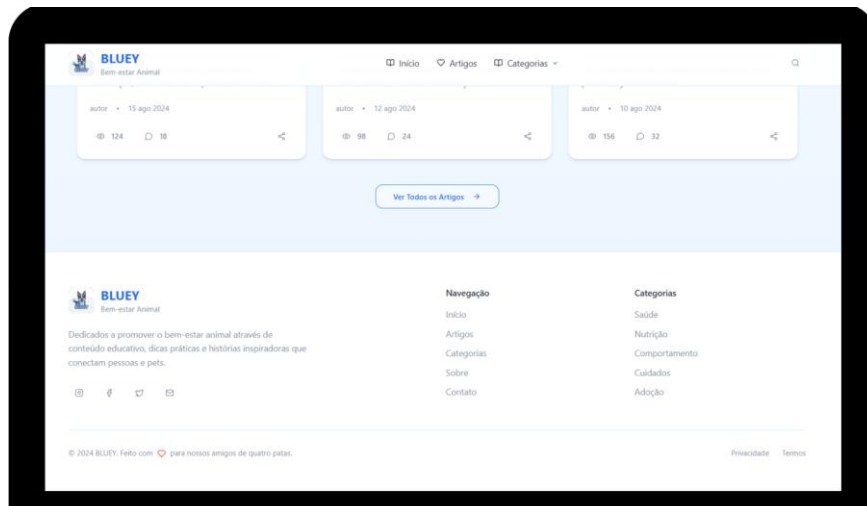
Fonte: o autor.

Body BLUEY – Corpo principal, versão desktop.



Fonte: o autor.

Footer BLUEY – Rodapé, versão desktop.



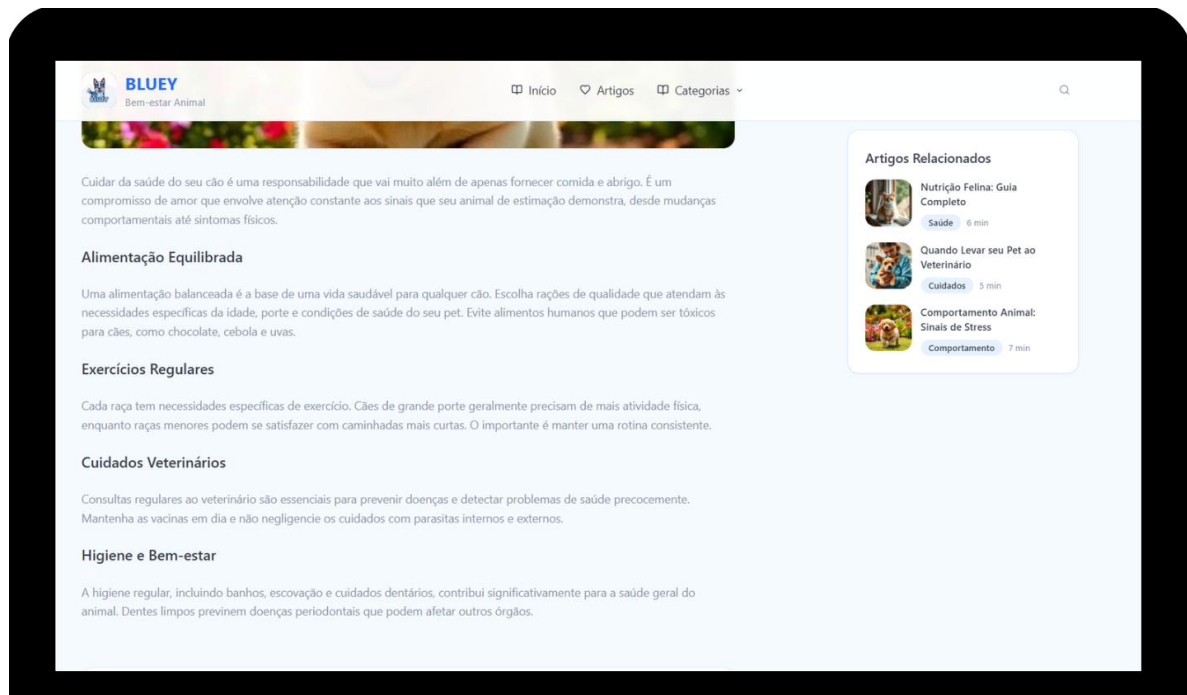
Fonte: o autor.

Body de leitura BLUEY – Corpo para leitura, versão desktop.



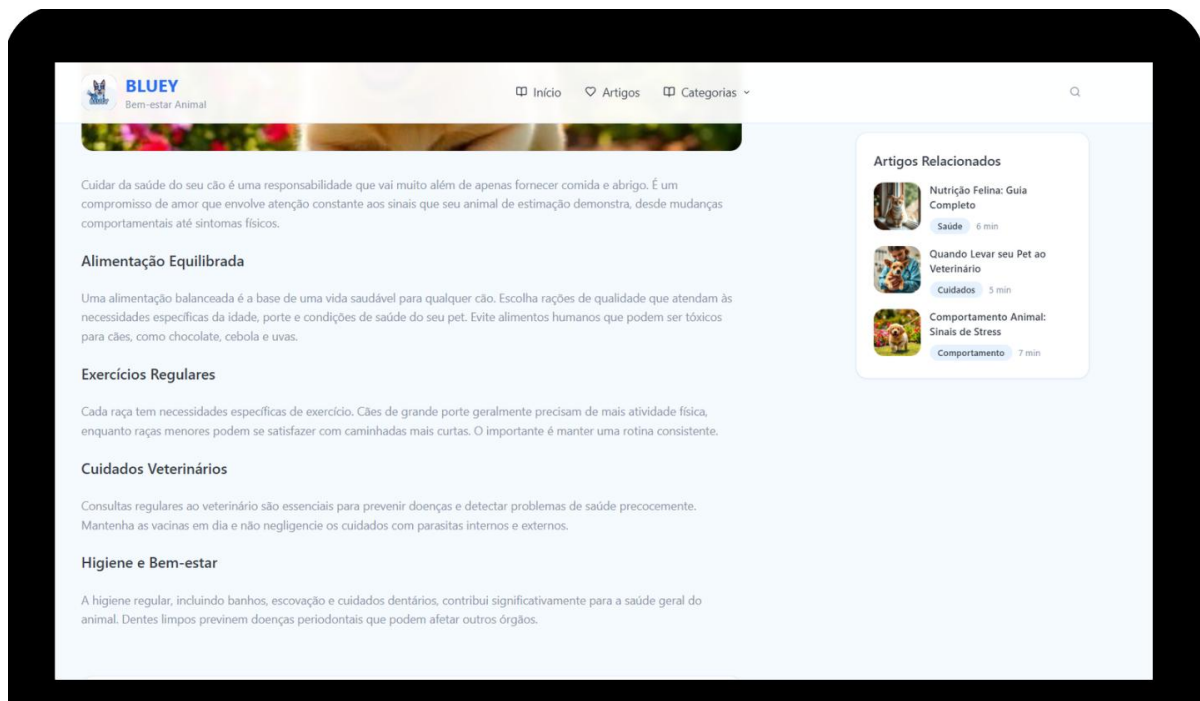
Fonte: o autor.

Body de leitura BLUEY – Corpo para leitura parte 2, versão desktop.



Fonte: o autor.

Body contato BLUEY – Corpo para envio de e-mail, versão desktop.



Fonte: o autor.

Home page BLUEY – Página inicial, versão mobile.



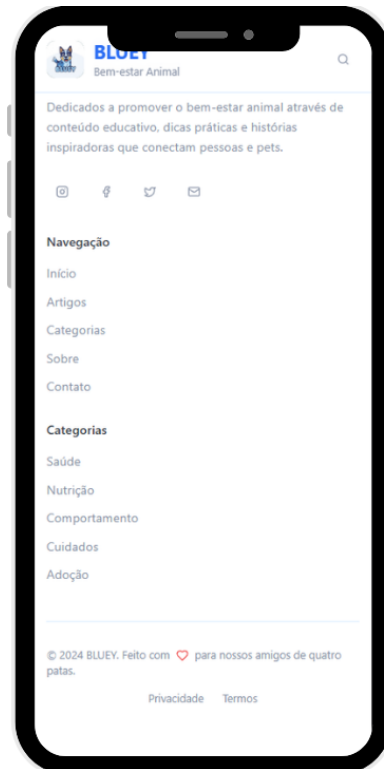
Fonte: o autor.

Body BLUEY – Corpo principal, versão mobile.



Fonte: o autor.

Footer BLUEY – Rodapé, versão mobile



Fonte: o autor.

Body de leitura BLUEY – Corpo para leitura, versão desktop.



Fonte: o autor.

Body de leitura BLUEY – Corpo para leitura parte 2, versão mobile.



Fonte: o autor.

Body contato BLUEY – Corpo para envio de e-mail, versão mobile.



Fonte: o autor.

7. ARQUITETURA DE SOFTWARE

Utilizando as tecnologias de arquitetura em camadas, arquitetura orientada a serviços e modelos MVC (Model-View-Controller) fornecida pelo framework Laravel, o projeto BLUEY conta com uma infraestrutura extremamente atual para sua disponibilização. Seu modelo de desenvolvimento conta com 3 camadas, são elas:

- Camada de Apresentação - interface para usuário, aqui o utilitário, seja ele administrador ou cliente, interage com a solução. Esta foi desenvolvida utilizando a linguagem PHP com integração da biblioteca Bootstrap, framework Tailwind CSS e a linguagem Javascript;
- Camada de Aplicação – a camada de aplicação consiste na lógica de negócio que envolve o projeto, aqui são definidas todas as regras de funcionamento Back-End e através da qual outras camadas se relacionam. Para tal utilizamos a linguagem server-side PHP com Laravel.
- Camada de Dados – nesta camada estão presentes as informações mutáveis do projeto, ou seja, o armazenamento de dados, desta forma utilizamos o banco de dados Mysql com hospedagem nos serviços da Amazon RDS (Relational Database Service), a hospedagem de imagens se vale do serviço S3 fornecido pela Amazon.

7.1. Componentes

7.1.1. *Front-end*

- PHP (8.2.12) – linguagem para renderização server-side das views criadas com o Laravel (12.28).
- Bootstrap (5.0) – biblioteca css para estilização das páginas.
- Tailwind CSS (3.4) – framework css para customizações específicas.
- Vite (7.0) – ferramenta de build para testes de layout otimizados.
- Javascript – interatividade entre cliente e aplicação.

7.1.2. *Back-end*

- PHP (8.2.12) – linguagem de programação principal da aplicação.
- Laravel (12.28) – framework MVC (Model-View-Controller) para auxílio do desenvolvimento com PHP.
- Composer (2.8) – gerenciamento de dependências do PHP.

7.1.3. *Camada de dados*

- MySQL – banco de dados relacional padrão do PHP, hospedado junto ao Amazon RDS para garantir persistência de dados e melhor gerenciamento dos eventos do banco.
- Amazon S3 – armazenamento de imagens utilizadas no projeto, o serviço fornecido pela Amazon traz possibilidade de backup para recuperação de dados em caso de perda.
- Eloquent ORM – mapeamento objeto-relacional oferecido pelo framework Laravel, facilitando as operações com banco de dados, sem necessidade de trabalhar com Queries diretas em banco.

7.2. Infraestrutura

7.2.1. Containerização

- Amazon ECR – o componente Elastic Container Registry da Amazon, permite o armazenamento de imagens Docker que são usadas para rodar a aplicação PHP+Laravel e o servidor Nginx, o versionamento das imagens facilita também a implementação da pipeline CI/CD com o Github.

7.2.2. Infraestrutura em Nuvem (AWS)

- EC2 – instancia Amazon para fornecimento de recurso computacional na nuvem.
 - Load Balancer – responsável pela distribuição de carga entre as definições de tarefa, traz alta disponibilidade e hospedagem WEB da aplicação (serviço de hosting).
 - Destiny group – gerenciamento de trafego dentro de um load balancer.
 - Security group – aqui define-se a estrutura de segurança de toda a aplicação, trabalhando com um firewall virtual para comunicação interna entre as aplicações na AWS e externa com a rede mundial.
- ECS – o Elastic Container Service é responsável por efetuar a orquestração dos contêineres, ele é quem dita seu mapeamento de rede, segurança e administração de recursos, neste caso utilizamos o AWS Fargate.
 - Cluster – local de agrupamento lógico das configurações de tarefas e serviços.
 - Service – configuração que atuará dentro do cluster para executar tarefas.
 - Task Definition – aqui define-se as configurações de uma tarefa para organizar as atividades que o container irá precisar, variáveis de ambiente, configurações de rede, segurança, monitoramento manipulação de imagens e outros.
- ECR – dentro do Elastic Container Registry efetuamos o versionamento e gerenciamento de imagens do container.
- IAM – o Identity and Access Management permite que façamos o controle de perfis de usuário com configurações específicas para que cada aplicação esteja isolada e somente possa efetuar atividades pertinentes.
- RDS – com o Relational Database Service, garantimos um banco de dados de alta disponibilidade, além de facilitar a implantação e comunicação com outros serviços da AWS.
- S3 Bucket – o s3 permite que o armazenamento de arquivos seja eficiente e traz facilidade no gerenciamento de backups.

7.3. Pipeline CI/CD

7.3.1. *Git/Github*

O controle de versionamento de código é efetuado através do Github, este proporciona uma estratégia de “Branch” qual permite a divisão do código em caminhos, neste caso homologação e produção, assim garantimos que antes de implementarmos os códigos desenvolvidos estes podem ser testados em um ambiente controlado com acesso restrito, posteriormente após validado que não há problemas com a nova atualização, efetua-se o pull request da nova versão para o ambiente de produção. Além disso o Github traz sua funcionalidade de “Actions” que serve para facilitar o direcionamento de código para sua hospedagem e disponibilização. Consiste na inserção de um arquivo .yaml que direciona ao github sobre como deve lidar com um deploy efetuado na aplicação, no caso do BLUEY, ele efetua sequencialmente:

1. Checkout de código – verifica todos os arquivos contidos no deploy.
2. Configuração de credenciais – define variáveis para conexão com a AWS.
3. Login no ECR – efetua login no serviço de imagens da AWS.
4. Definição de variáveis da imagem – definir tag da imagem.
5. Verificação de repositório – valida a existência da imagem referenciada.
6. Construção da imagem – comando Docker Build para criar a nova imagem.
7. Teste de imagem – efetuar health check na rota definida para verificar funcionamento da nova imagem.
8. Envio da imagem – comando Docker Push para enviar a imagem para o ECR.
9. Atualização do serviço ECS – efetua forçadamente a inicialização do serviço ECS com a nova imagem enviada referenciando seu cluster.
10. Verificação – testar status do cluster e serviço para garantir que está correto.

Além das etapas no Github, também efetuamos configuração do arquivo Dockerfile para especificar as configurações da imagem ECR, dentro da qual rodamos um servidor Node para utilização do Vite, um servidor PHP para processamento server-side e um servidor Nginx para Web.

7.4. **Design Patterns**

7.4.1. *MVC Laravel*

O Laravel como framework PHP trabalha com o padrão Model-View-Controller para construir uma arquitetura lógica, o Model é usado para representar a lógica de negócio efetuando operações e enviar informações para o Controller, o View é a interface pela qual o usuário interage com a aplicação utilizando os templates Blade, e o Controller é responsável por fazer a mediação entre Model e View, processando as requisições e agindo conforme necessário.

7.5. **Integrações**

7.5.1. *Back-end e Front-end*

As requisições efetuadas pelo front-end são enviadas ao back-end através de requisições HTTP, são elas: GET, POST, PUT e DELETE, as validações de segurança ficam por conta do CSRF com tokens que validam cada requisição, o processamento das requisições é feita no lado do servidor.

7.5.2. *Back-end e Banco de Dados (Mysql)*

O Eloquent ORM do Laravel é uma abstração para operações com banco de dados, assim toda a lógica que efetua queries em banco é feita através de classes pré-definidas com padrões de utilização, isso facilita a migração dos dados e consultas para o Back-end.

7.5.3. BLUEY e Serviços AWS.

Integração efetuada através da AWS SDK para PHP, esta implementa diversas funções e configurações necessárias para comunicar-se com serviços da Amazon, também necessário configurações em variáveis de ambiente (.ENV) para conexão com RDS.

7.6. Segurança

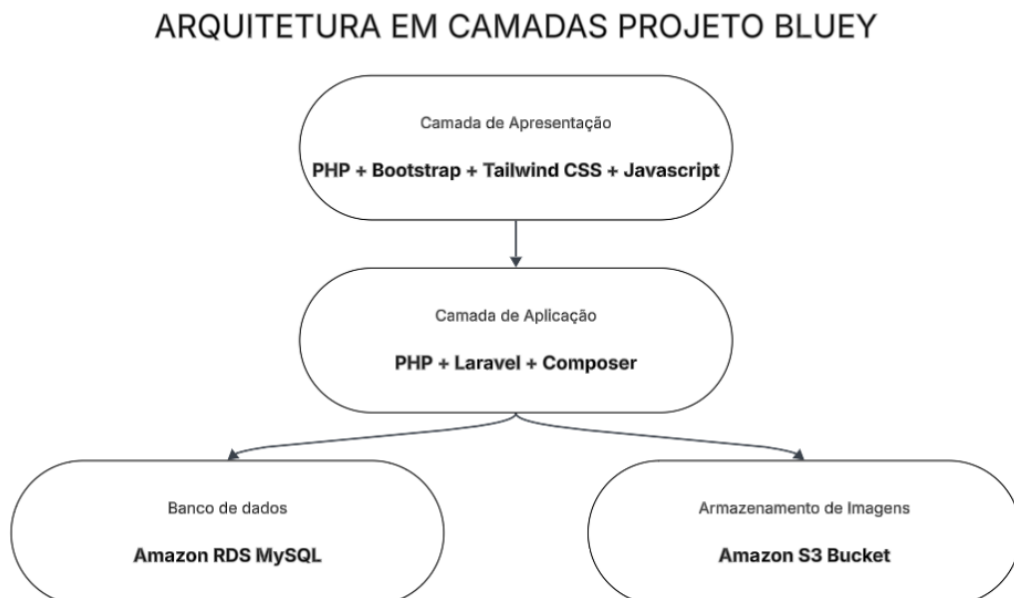
O Laravel por padrão já implementa diversas vertentes de segurança úteis para proteger a aplicação, a utilização do CSRF (Cross-Site Request Forgery) em todas as requisições efetuadas pela aplicação previne injeção de comandos maliciosos que possam afetar o comportamento padrão, a prevenção de XSS (Cross-site Scripting) é encarregada automaticamente pelo template Blade do Laravel, o Eloquent ORM protege a aplicação contra SQL Injection utilizando estrutura padrão e validação de campos para manipulação de banco de dados.

Os serviços AWS por sua vez tem um controle rigoroso de acesso, utilizando VPC, grupos de segurança, IAM e variáveis de ambiente.

7.7. Diagramas

Os diagramas são uma representação visual para simplificar uma estrutura e trazer clareza para o leitor, aqui apresentamos os diagramas arquiteturais do BLUEY.

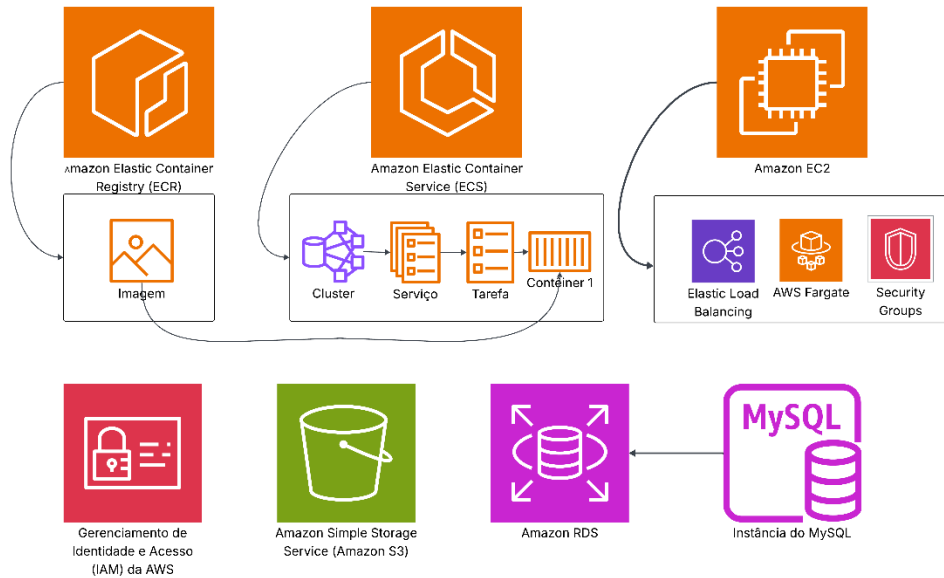
7.7.1. Diagrama de arquitetura em camadas.



Fonte: o autor.

7.7.2. Diagrama de infraestrutura em nuvem.

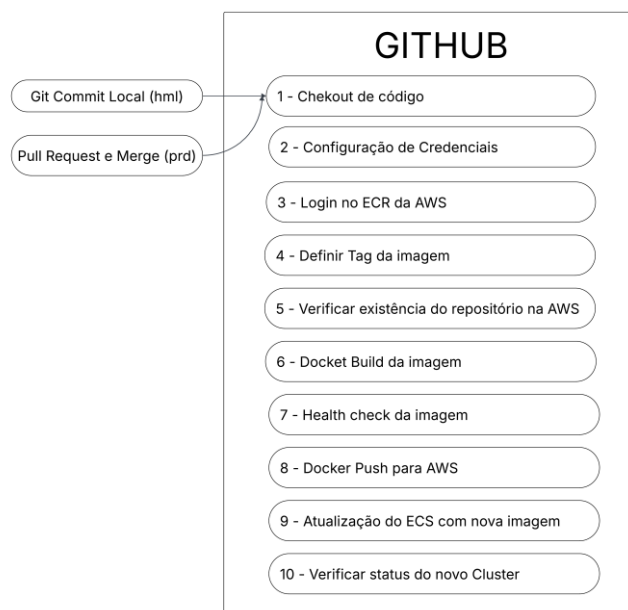
ARQUITETURA DE INFRAESTRUTURA EM NUVEM PROJETO BLUEY



Fonte: o autor.

7.7.3. Diagrama de pipeline ci/cd

PIPELINE CI/CD PROJETO BLUEY



Fonte: o autor.

8. VALIDAÇÃO DA SOLUÇÃO

Para validação do projeto BLUEY utilizamos a biblioteca PHPUnit, muito conceituada na escrita de testes de software, contando com uma estrutura de aplicação simples e eficaz, nos permite automatizar o código para simular um ambiente no qual a aplicação é executada, assim garantindo que o ambiente esteja em suas melhores condições para disponibilização ao usuário final. Para exemplificar o uso do PHPUnit com o Laravel Testing Suite, foi necessário criar uma nova base de dados denominada “bluey_testing” no MySQL, também utilizamos um armazenamento simulado na Amazon S3 utilizando o “Storage::fake('s3')”.

Considerando o conceito vital da aplicação de criação de artigos, escrevemos quatro testes para essa funcionalidade:

- Caminho perfeito – aqui recriamos o cenário ideal onde é enviado os campos necessários para criação de um artigo normalmente e validamos se de fato houve input no banco.
- Faltam parâmetros – buscamos validar a criação de um artigo sem os campos obrigatórios como título, este não deve ser permitido e retornar erro.
- Título duplicado – uma das premissas aplicadas no BLUEY é que não se pode criar um artigo com título que já exista em banco de dados, esse teste busca validar exatamente essa premissa.
- Falta token validação – este teste busca validar se a aplicação permite criar um artigo sem enviar o token definido em Middleware, ação essa que não deve salvar no banco e retornar erro ao usuário.

Abaixo descrevemos o teste “test_it_blocks_access_without_valid_token”, descrito anteriormente como “Falta token validação” na linguagem PHP:

Teste automatizado – teste_it_blocks_access_without_valid_token

```
public function test_it_blocks_access_without_valid_token(): void
{
    $category = Category::create(attributes: ['description' => 'Teste']);
    $file = UploadedFile::fake()->image(name: 'test.jpg');

    // Ação - Tentar acessar sem token
    $response = $this->post(uri: '/admin/articles', data: [
        'title' => 'Teste sem token',
        'body' => 'Conteúdo...',
        'author' => 'Autor',
        'id_category' => $category->id,
        'image' => $file,
        'qt_views' => 0,
        'qt_emails' => 0,
    ]);

    // Assertividade - Deve retornar erro 401
    $response->assertStatus(status: 401);
    $response->assertSeeText(value: 'Acesso negado. Token de segurança não encontrado ou inválido.');
```

Fonte: o autor.

Através dos testes, buscamos deixar a aplicação mais sólida para nossos usuários, prevenindo precocemente eventuais bugs que pudessem ocorrer, validando integrações com serviços externos como a AWS e também servindo como base de documentação para descrever o comportamento esperado da funcionalidade. Considerando as funcionalidades descritas, obtivemos aproveitamento de 100% sobre os testes, garantindo que a criação de artigos ocorre conforme planejado, abaixo descrevemos o impacto executado do código:

Teste automatizado – resumo sobre teste executado e sucesso.

```
G:\xampp\htdocs\bluey>php artisan test tests/Feature/CreateArticleTest.php

PASS Tests\Feature\CreateArticleTest
✓ it can create article with image successfully
✓ it validates required fields when creating article
✓ it prevents creating article with duplicate title
✓ it blocks access without valid token

Tests: 4 passed (13 assertions)
Duration: 2.77s
```

Fonte: o autor.

Em se tratando de testes manuais e de usabilidade podemos incluir as validações de interface, performance e disponibilidade. A responsividade da aplicação foi testada em navegador WEB utilizando-se do recurso para desenvolvedores “barra de ferramentas do dispositivo” que permite emular o uso da aplicação em diversos

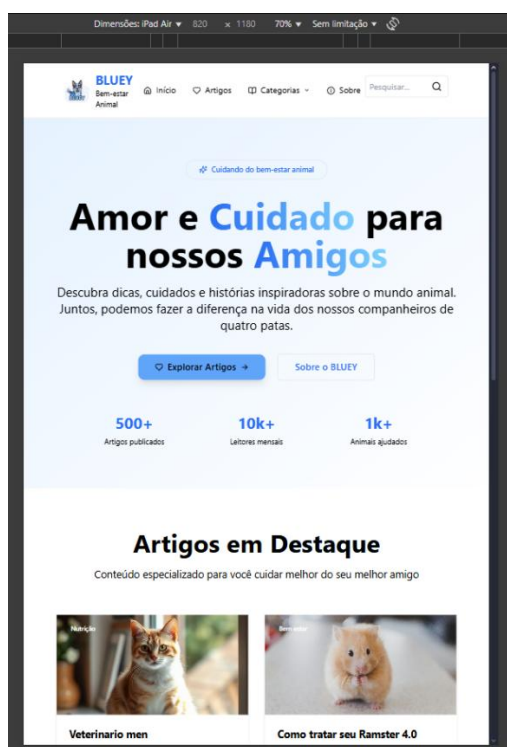
tipos de dispositivos, a primeira demonstração foi testada utilizando dispositivo móvel “iPhone 14 Pro Max” com resolução de 430x932 pixels, a segunda demonstração utilizou-se do modelo tablet “iPad Air” com resolução de 820x1180 pixels e a terceira demonstração usa a resolução padrão desktop de 1280x1024 pixels.

Teste responsividade – iPhone 14 Pro Max



Fonte: o autor.

Teste responsividade – iPhone Air



Fonte: o autor.

Teste de responsividade - Desktop



Fonte: o autor.

A disponibilidade da aplicação é buscada através do processo de integração contínua implementado usando o Github Actions e as funcionalidades nativas da AWS, a validação de cada deploy é executada através das etapas número sete e dez em nosso arquivo de configuração yaml.

Teste de disponibilidade – passo 7 health check da aplicação local

```
# Etapa 7: Testar a imagem (health check)
- name: Testar imagem Docker
  run: |
    # Iniciar container em background
    docker run -d --name test-container -p 8080:80 ${ steps.image-tag.outputs.FULL_IMAGE_URI }

    # Aguardar container estar pronto
    sleep 30

    # Testar health check
    curl -f http://localhost:8080/health || exit 1

    # Limpar
    docker stop test-container
    docker rm test-container
```

Fonte: o autor.

Teste de disponibilidade – passo 10 verificar o deploy na AWS.

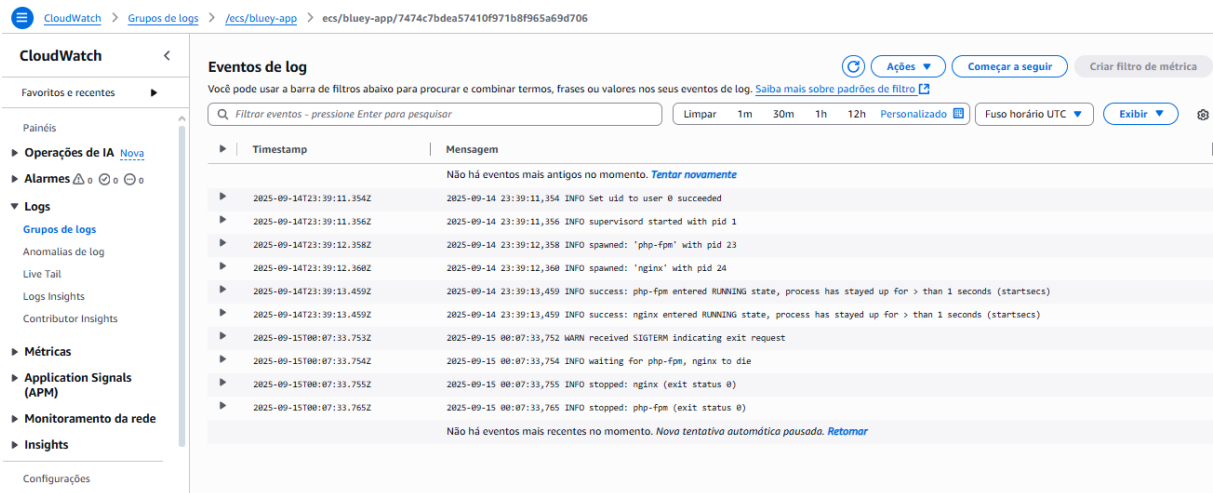
```
# Etapa 10: Verificar deploy
- name: Verificar deployment
  run: |
    # Obter status do serviço
    SERVICE_STATUS=$(aws ecs describe-services \
      --cluster ${env.ECS_CLUSTER} \
      --services ${env.ECS_SERVICE} \
      --query 'services[0].deployments[0].status' \
      --output text)

    if [ "$SERVICE_STATUS" != "PRIMARY" ]; then
      echo "❌ Deployment falhou! Status: $SERVICE_STATUS"
      exit 1
    else
      echo "✅ Deployment realizado com sucesso!"
    fi
```

Fonte: o autor.

O Amazon Cloud Watch também nos permite que verifiquemos logs em tempo real conforme configurado em seus serviços, no caso do BLUEY, possuímos log sobre toda atualização de imagem efetuada no ECR (Elastic Container Service).

Teste de disponibilidade – Amazon Cloud Watch.



The screenshot shows the Amazon CloudWatch console. The breadcrumb navigation at the top indicates the path: CloudWatch > Grupos de logs > /ecs/bluey-app > ecs/bluey-app/7474c7bdea57410f971b8f965a69d706. The left sidebar contains a 'Painéis' section with links to 'Operações de IA', 'Alarmes', 'Logs', 'Métricas', and 'Insights'. The 'Logs' section is expanded, showing 'Grupos de logs', 'Anomalias de log', 'Live Tail', 'Logs Insights', and 'Contributor Insights'. The main area is titled 'Eventos de log' and contains a search bar and a table of log events. The table has two columns: 'Timestamp' and 'Mensagem'. The events show the application's startup and running status, including messages like 'INFO Set uid to user 0 succeeded', 'INFO supervisord started with pid 1', 'INFO spawned: 'php-fpm' with pid 23', and 'INFO spawned: 'nginx' with pid 24'.

Fonte: o autor.

O projeto BLUEY está disponibilizado externamente através do serviço EC2 da Amazon e abaixo de um domínio privado com endereço HTTP na porta 80 (<http://bluey.app.br>), considerando o contexto acadêmico da aplicação, não possui certificado SSL (Secure Sockets Layer) para comunicação HTTPs na porta 443, mas pode ser acessado normalmente para validação. A aplicação conta também com um ambiente de homologação (<http://bluey-lb-603001560.us-east-2.elb.amazonaws.com>) disponível apenas para o domínio específico de rede do desenvolvedor do projeto.

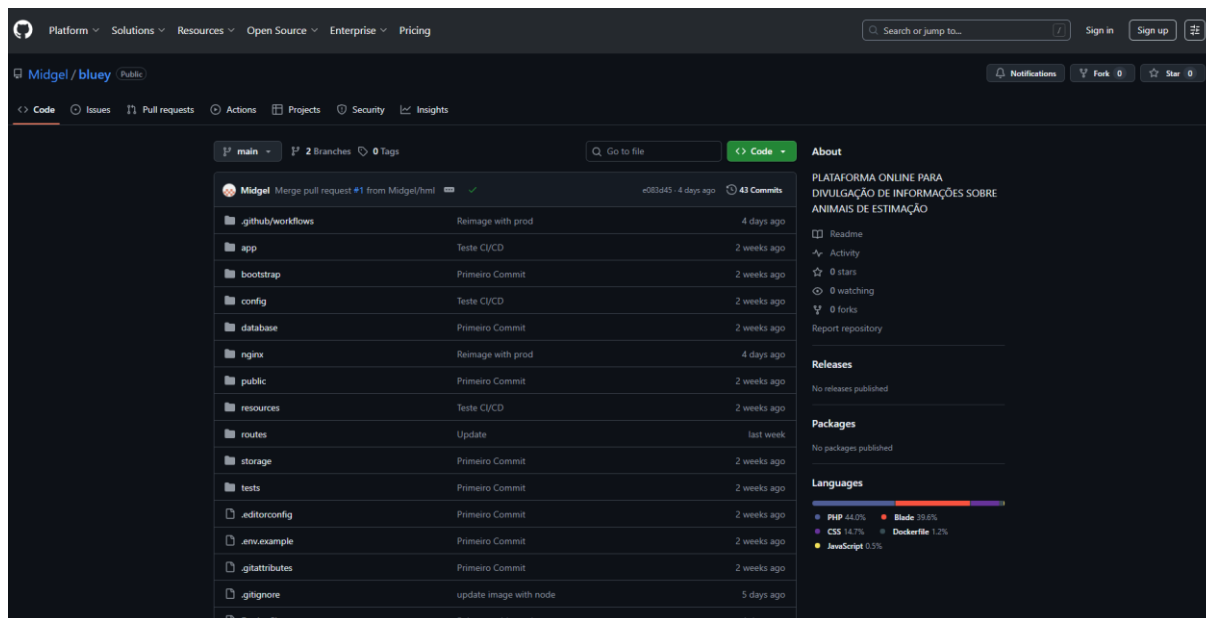
Dessa forma podemos constatar a implementação prática do projeto, demonstrando maturidade e capacidade de desenvolvimento de uma aplicação real com metologias de mercado e integração com serviços complexos.

9. REGISTROS DAS EVIDÊNCIAS DO PROJETO

9.1. Código fonte

O projeto de código do BLUEY está disponível em repositório público da plataforma Github e pode ser acessando através do endereço <https://github.com/Midgel/bluey>. É possível verificar a arquitetura de arquivos utilizados para funcionamento da aplicação, respeitando boas práticas de desenvolvimento estruturados automaticamente pelo Laravel, também é possível verificar histórico de commit's efetuados e a estratégia de divisão de ambientes homologação (hml) e produção (main) com diferentes Branch's.

Código fonte – projeto BLUEY no Github.



Fonte: o autor.

9.2. Aplicação WEB

A verificação do projeto implementado pode ser efetuada acessando a página web <http://bluey.app.br>, aqui estão presentes todas as funcionalidades anteriormente apresentadas e que podem ser testadas conforme interesse do usuário, é possível ler

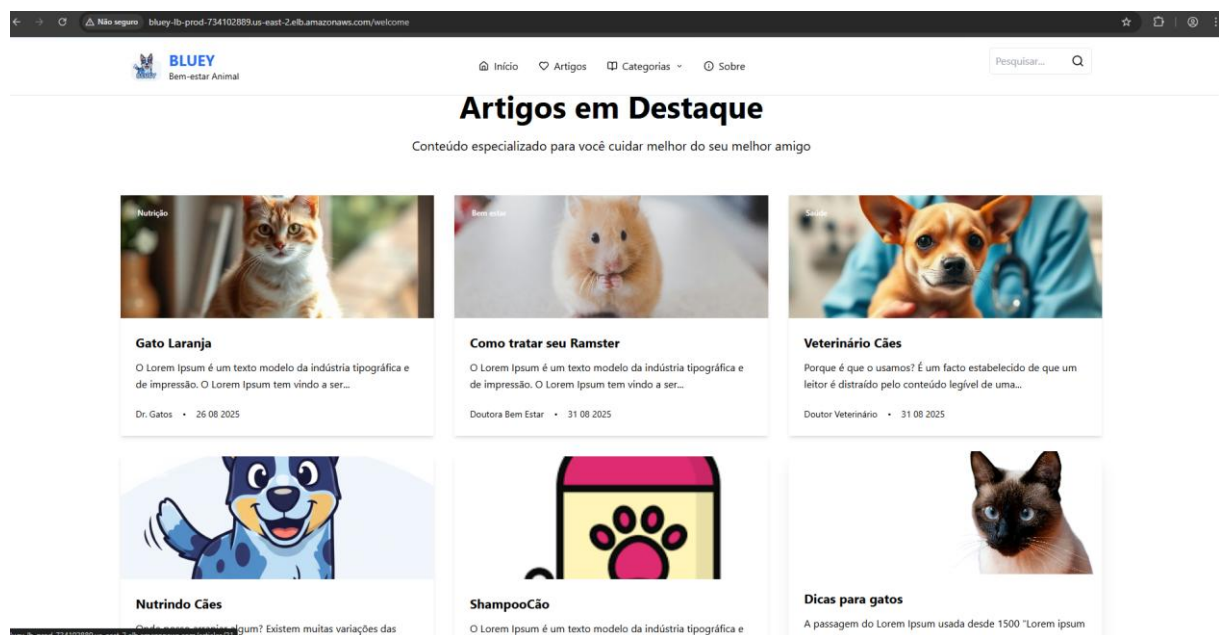
os artigos, filtrar por categoria, navegar entre páginas, enviar e-mail de contato, verificar redes sociais, ver informações relevantes sobre o projeto e outros, a manipulação de artigos (CRUD) está reservada ao ambiente local, buscando através deste prover segurança na manutenção de recursos computacionais armazenados em nuvem. A disponibilização do ambiente é completamente garantida pelo serviço de Load Balancer da Amazon permitindo acesso integral ao site a qualquer momento. Aplicação WEB BLUEY – página Home/Welcome.



Artigos em Destaque

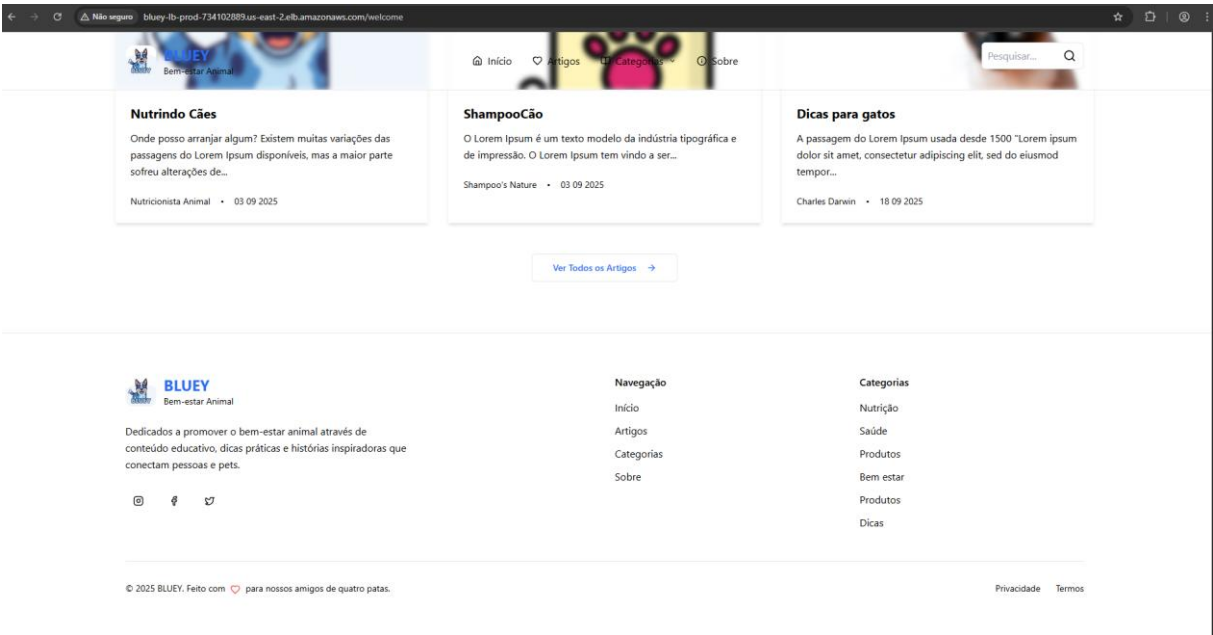
Fonte: o autor.

Aplicação WEB BLUEY – página Home/Welcome.



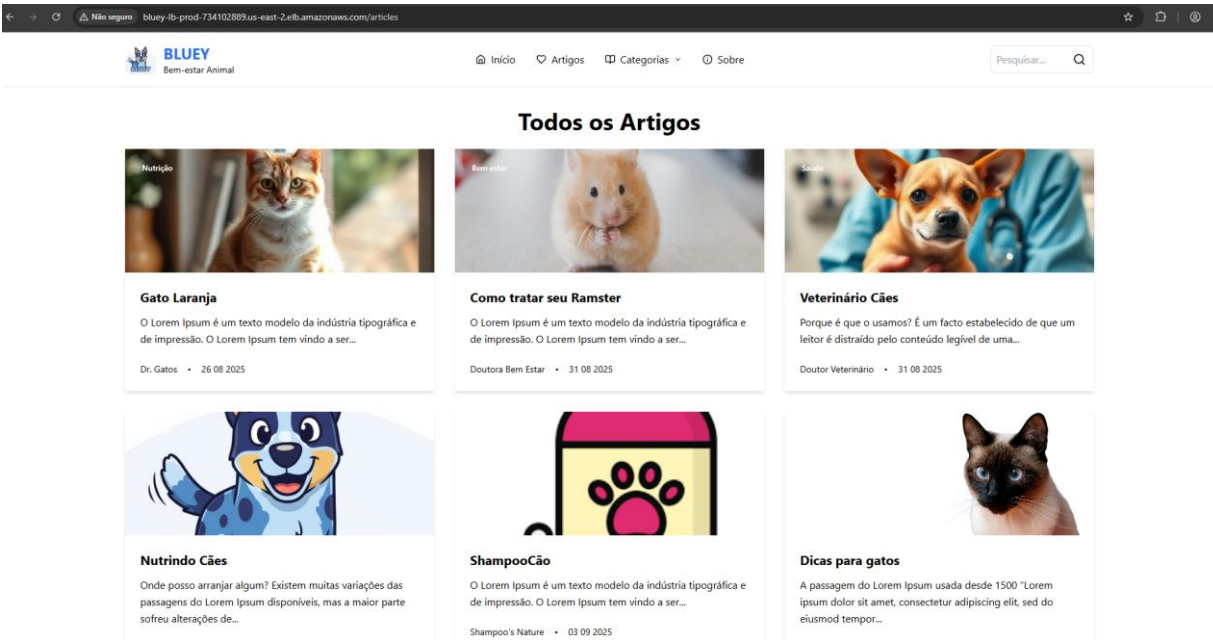
Fonte: o autor.

Aplicação WEB BLUEY – página Home/Footer.



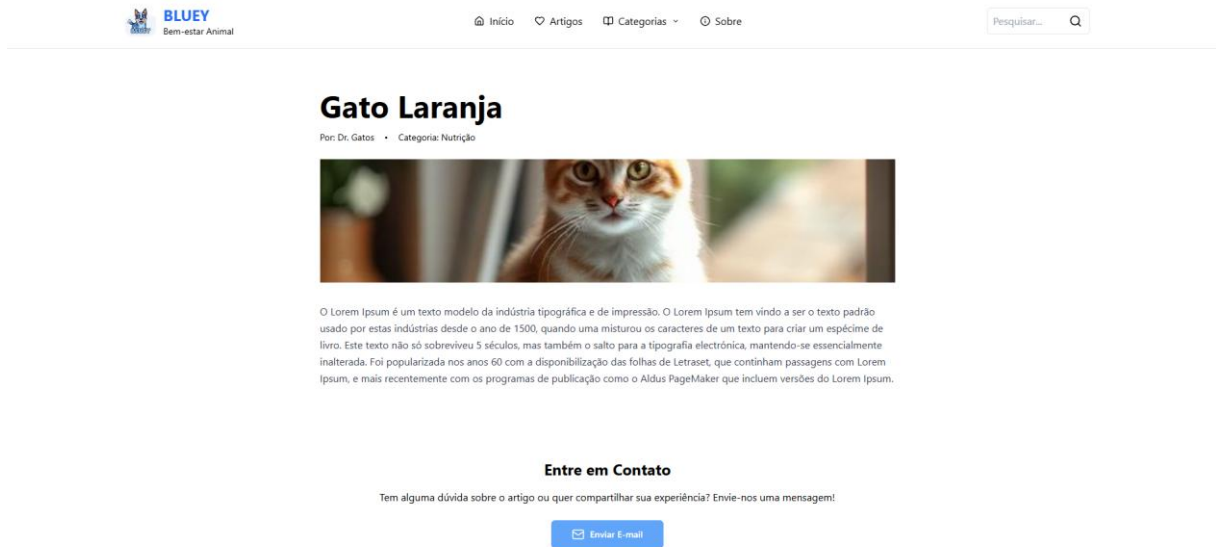
Fonte: o autor.

Aplicação WEB BLUEY – página Artigos.



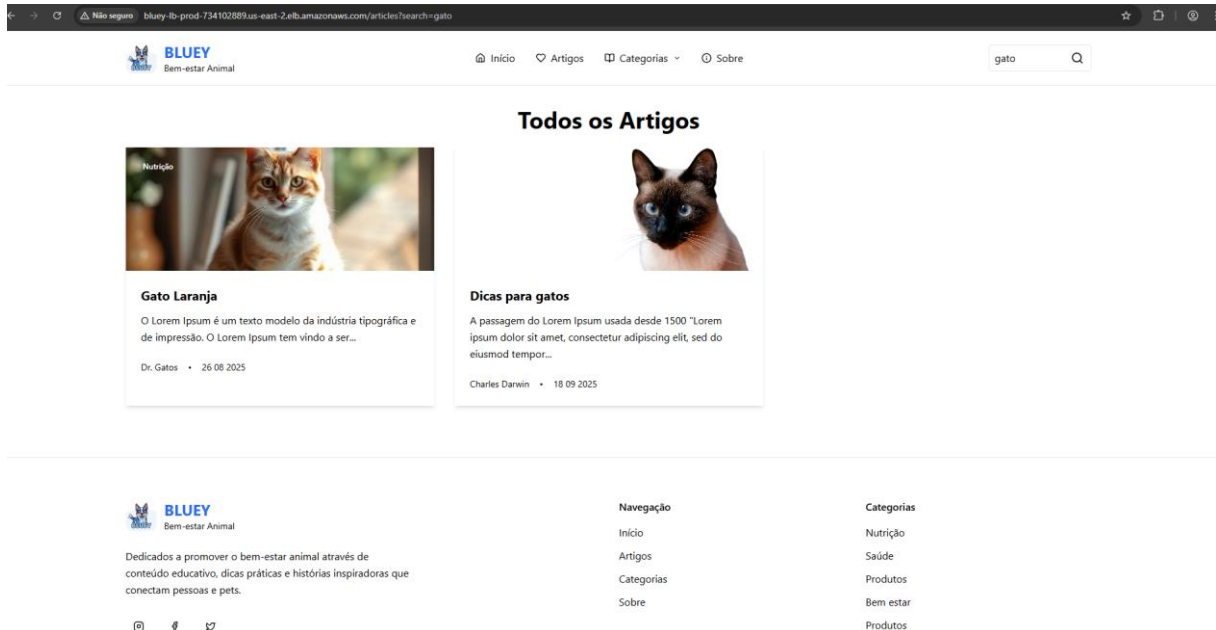
Fonte: o autor.

Aplicação WEB BLUEY – página Ler Artigo.



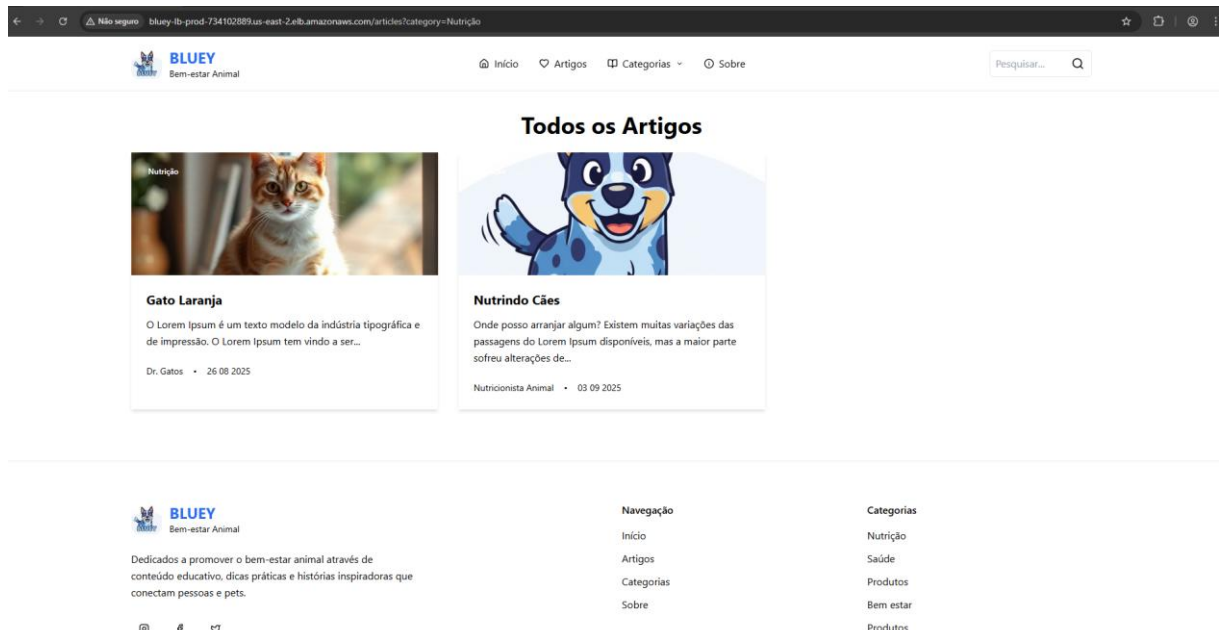
Fonte: o autor.

Aplicação WEB BLUEY – página Busca por texto.



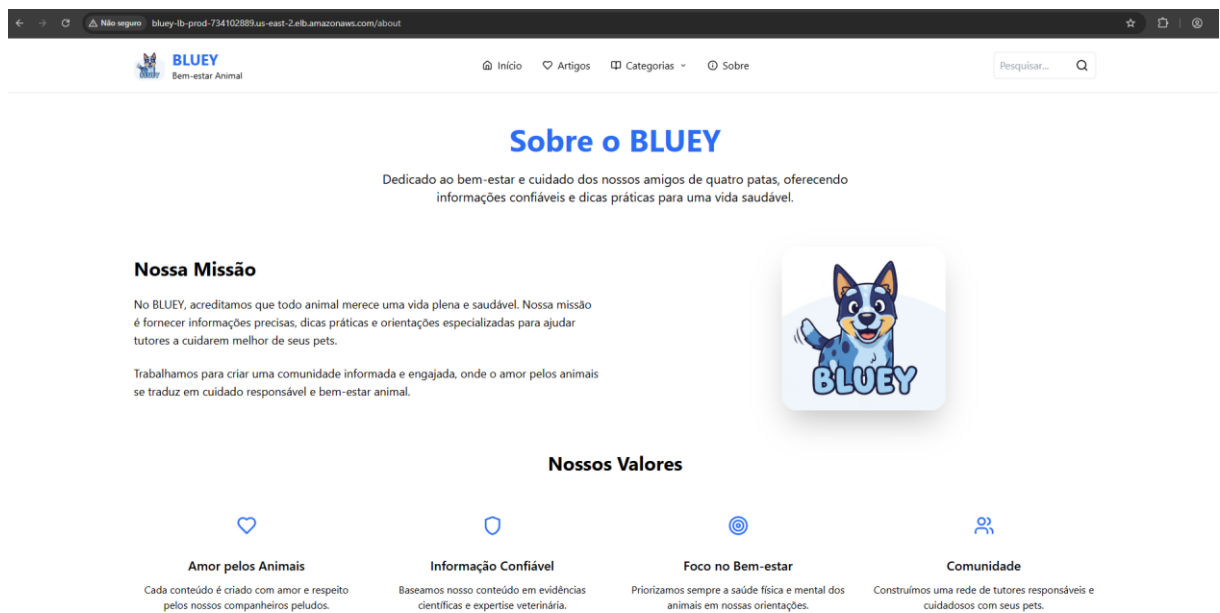
Fonte: o autor.

Aplicação WEB BLUEY – página Busca por categoria.



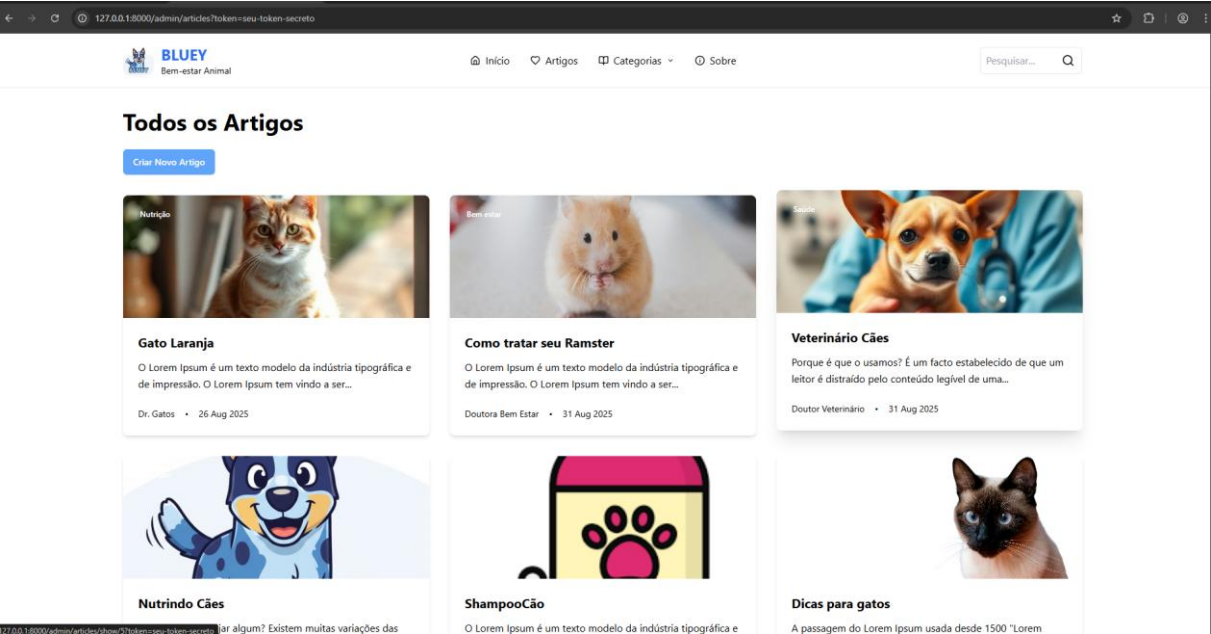
Fonte: o autor.

Aplicação WEB BLUEY – página Sobre.



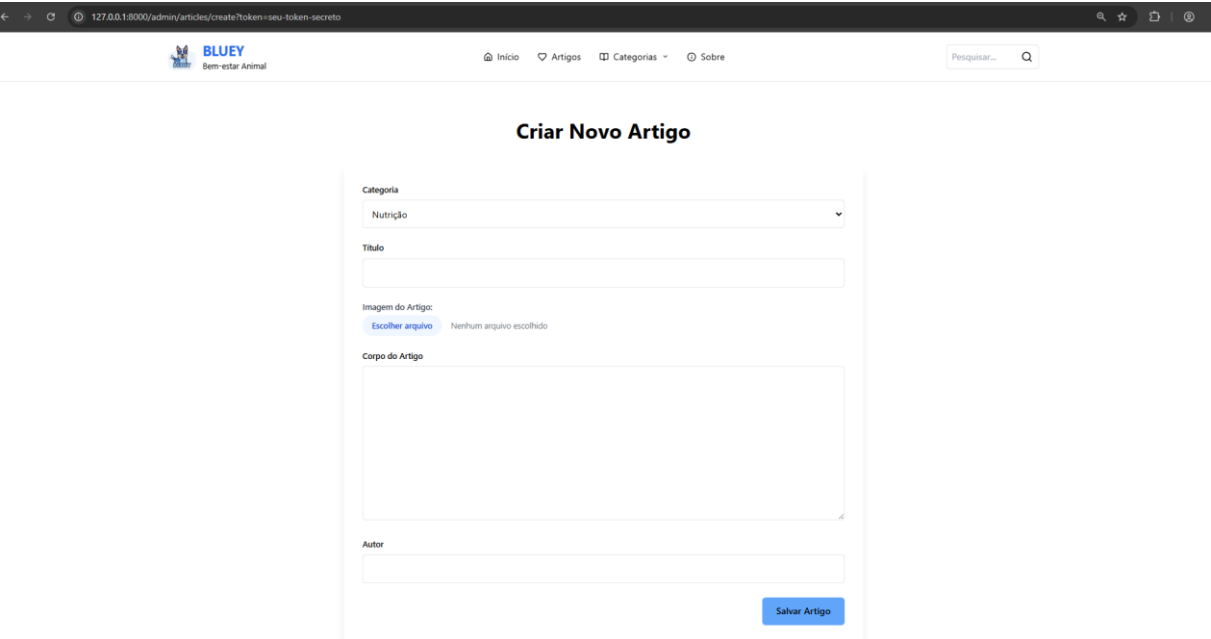
Fonte: o autor.

Aplicação WEB BLUEY – página Artigos admin.



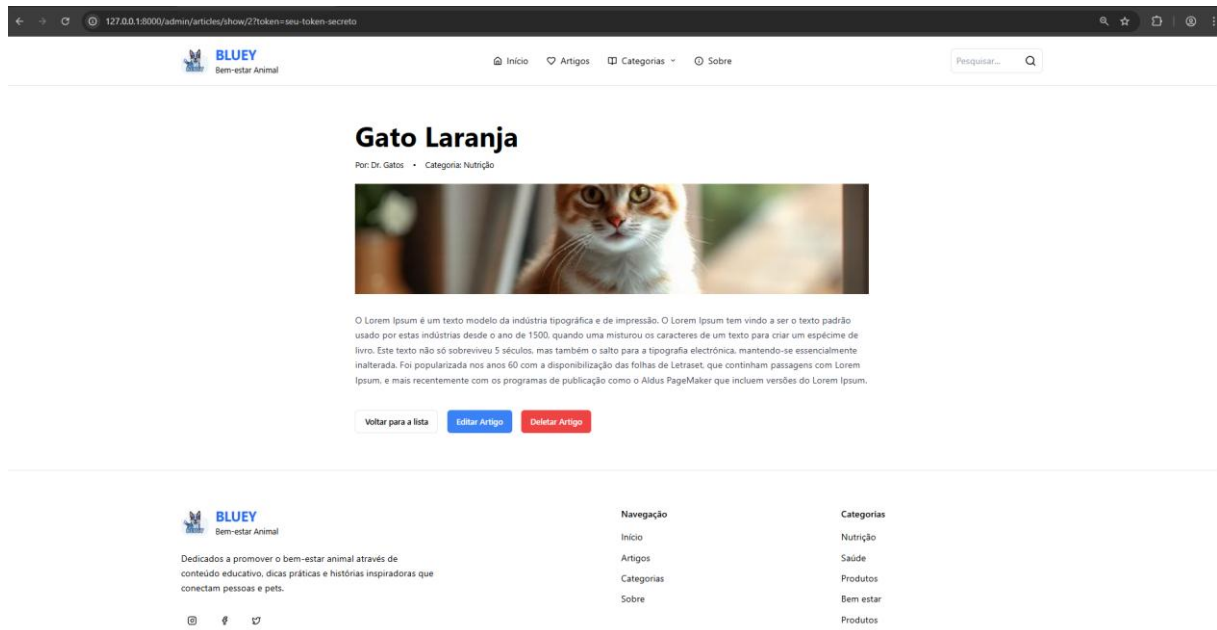
Fonte: o autor.

Aplicação WEB BLUEY – página Artigos admin criar.



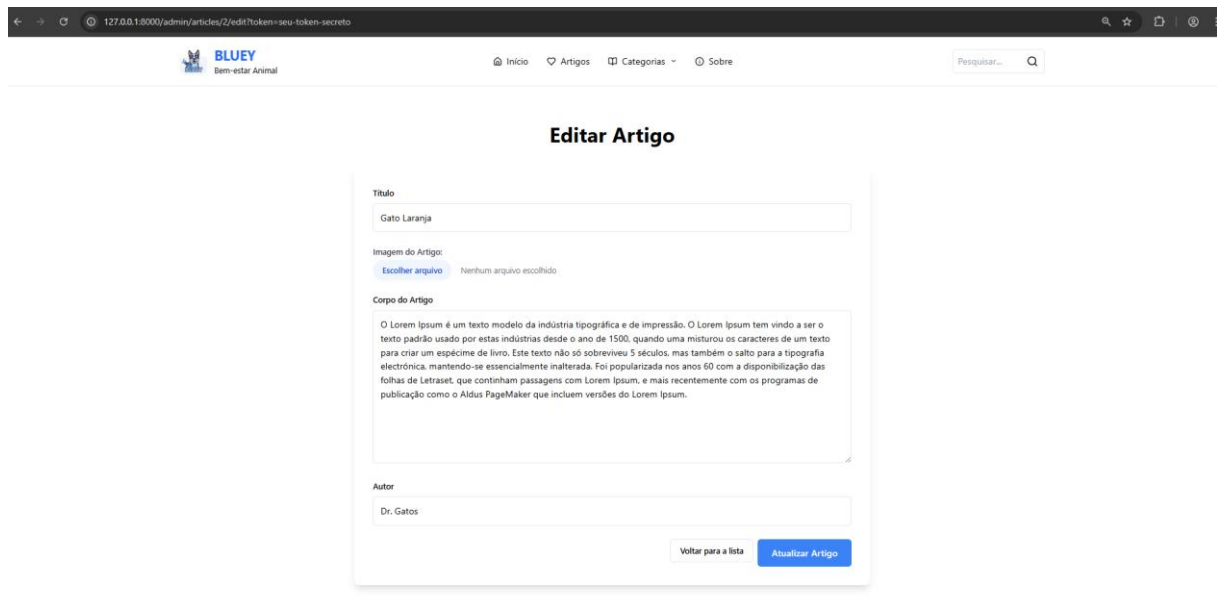
Fonte: o autor.

Aplicação WEB BLUEY – página Artigos admin ver.



Fonte: o autor.

Aplicação WEB BLUEY – página Artigos admin editar.



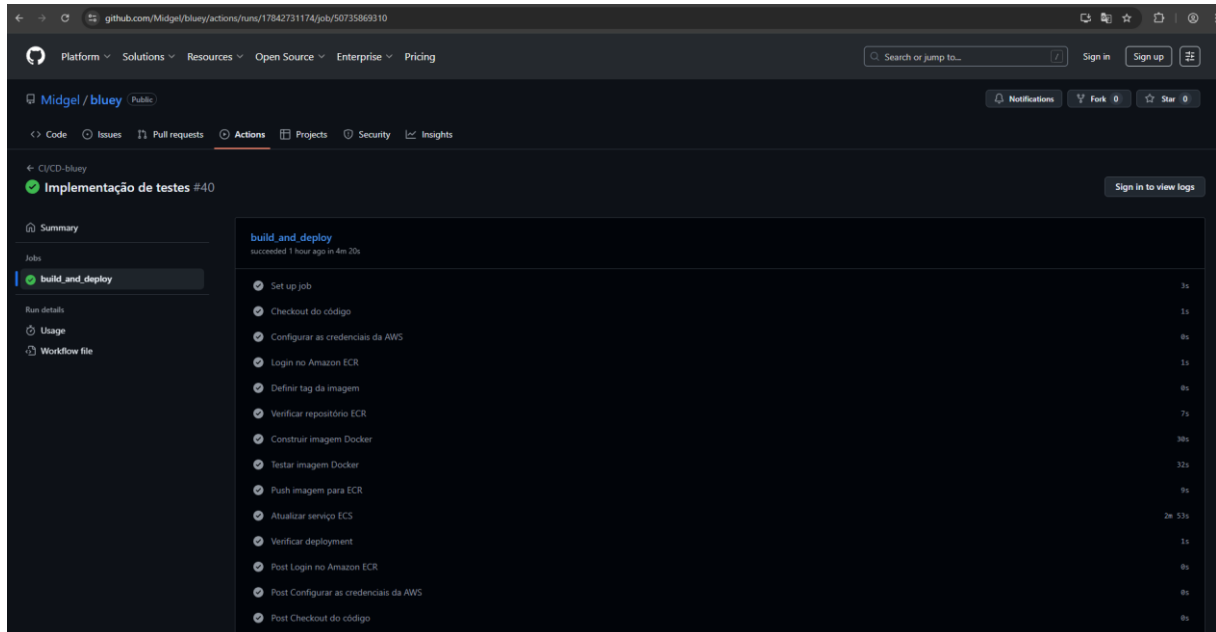
Fonte: o autor.

9.3. Pipeline CI/CD

Através do recurso Github Actions podemos verificar a esteira de integração e implantação contínua implantada no projeto BLUEY, sua configuração é executada dos através de passos definidos nos arquivos `.github/workflows/hml.yaml` e `.github/workflows/main.yaml`. A pipeline descrita para o BLUEY leva cerca de cinco minutos para ser finalizada e a maior parte do tempo é destinada a atualização do serviço na AWS e testes sobre a imagem enviada, abaixo exemplificamos o deploy de

número 40 onde testamos a implementação de testes automatizados, neste é possível verificar todas as etapas descritas anteriormente assim como o tempo de execução, para acessar esse teste em específico utilize o endereço <https://github.com/Midgel/bluey/actions/runs/17842731174/job/50735869310>.

Pipeline CI/CD – deploy #40



Fonte: o autor.

9.4. Documentação principal

O documento denominado “BLUEY – PLATAFORMA ONLINE PARA DIVULGAÇÃO DE INFORMAÇÕES SOBRE ANIMAIS DE ESTIMAÇÃO – MIDGEL RIBEIRO BORGES – PUCRS 2025.pdf” é a principal fonte de informações sobre o projeto BLUEY, para acessá-lo basta ir ao endereço

10. CONSIDERAÇÕES FINAIS E EXPECTATIVAS

A trajetória do curso e o fechamento deste com a elaboração do trabalho aqui apresentado certamente são motivo de exaltação. Como fechamento deste documento, apresente um pouco da sua trajetória ao longo do curso e quais expectativas imagina surgir a partir de tal formação.

REFERÊNCIAS

Citar todas as referências utilizadas no trabalho (seguir as normas da ABNT).

REDAÇÃO E LAYOUT

A redação precisa ser clara e fluida, prezando pelo layout do template previsto. O Trabalho de Conclusão de Curso deve ser autoral e inédito. A originalidade é um requisito essencial à aprovação do TCC, neste sentido, não serão aceitos trabalhos já apresentados ou submetidos a avaliações anteriores, para obtenção de grau em outros cursos ou níveis de ensino, publicações em periódicos, revistas ou outros meios, bem como, obras derivadas, reproduções totais ou parciais de produções do próprio autor ou de terceiros.

Todas as referências a trabalhos e obras de terceiros ou do próprio autor, somente podem ser incorporadas ao texto por meio de citações devidamente referenciadas como citação direta ou indireta, segundo as normas da ABNT.

Quaisquer trechos oriundos de outros materiais, obras de terceiros ou do próprio autor inseridos no TCC não documentados como citação, caracterizarão trechos com similaridade, falta de originalidade e consequentemente poderão ensejar a reprovação do aluno.