Documentação do Projeto Interdisciplinar - Gestão Ágil de Projetos de Software

SaneaSP

Davy Oliveira Ribeiro

Matheus Augusto Santos Gueff

Pedro Silva Martins

Ryan Carlo Negretti Pereira

Sumário

LISTA DE FIGURAS

LISTA DE QUADROS

Introdução

Resumo do projeto

O projeto SaneaSP tem como objetivo criar um site para fortalecer a comunicação entre a população e os responsáveis pela área de saneamento em Sorocaba/Votorantim, contribuindo para duas ODS: Água Potável e Saneamento (ODS6) e Saúde e Bem-Estar (ODS3). O projeto permitirá que os cidadãos relatem problemas de forma estruturada e fácil e acompanhem as soluções para esses problemas, além de disponibilizar informações organizadas sobre doenças relacionadas e ações corretivas e outras notícias relacionadas à área de saneamento básico. Dessa forma, buscamos contribuir para a eficiência e transparência das iniciativas já existentes.

Tema Central

O tema central do projeto integra três eixos fundamentais que se complementam: Tecnologia e Desenvolvimento Web, com foco na criação de uma aplicação web moderna, acessível e funcional; Saneamento e Água Potável (ODS 6), destacando a importância da infraestrutura adequada e do acesso universal à água de qualidade; e Saúde Pública (ODS 3), promovendo a prevenção de doenças por meio da informação confiável, da participação cidadã e da melhoria das condições sanitárias nos bairros. Esses elementos convergem para fortalecer a relação entre população e poder público, incentivando ações mais eficazes e transparentes na área de saneamento básico.

Objetivo Geral do Projeto

Desenvolver uma plataforma web inovadora e funcional, denominada SaneaSP, com o propósito de fortalecer a comunicação entre a população e os órgãos responsáveis pelo saneamento básico, promovendo o acesso à informação e incentivando a participação cidadã na identificação e denúncia de problemas relacionados à água potável e à infraestrutura sanitária. A aplicação será concebida com base nos seguintes pilares técnicos:

- Arquitetura Full-Stack: Estruturação completa da aplicação, unindo a experiência do usuário no front-end com a lógica de negócios e gestão de

dados no back-end, assegurando um sistema coeso, eficiente e responsivo.

- Tecnologia de Ponta: Emprego de ferramentas e frameworks modernos, que proporcionem alta performance, escalabilidade e segurança, facilitando futuras expansões e manutenções.
- Serviços Web RESTful: Criação de uma API baseada em padrões REST, para garantir a comunicação fluida entre os componentes da aplicação e viabilizar a integração com outras soluções digitais.
- Acessibilidade e Usabilidade: Desenvolvimento de uma interface clara, acessível e amigável, seguindo as normas internacionais de acessibilidade digital (WCAG), com foco em inclusão e facilidade de uso em diferentes dispositivos.
- Gestão Colaborativa do Código: Aplicação de práticas de versionamento e controle de código por meio de plataformas colaborativas, permitindo rastreamento de alterações e trabalho em equipe eficiente durante todo o ciclo de desenvolvimento.

Levantamento de Requisitos

Descrição da técnica utilizada

O levantamento de requisitos para o projeto SaneaSP foi feito de forma simples e direta. Trabalhamos apenas com nossa percepção da realidade e do que imaginamos que os usuários precisariam, sem a aplicação de técnicas profissionais formais.

Baseamo-nos em conversas informais entre os integrantes da equipe, nas experiências pessoais com problemas de saneamento e em suposições sobre o que a população gostaria de ver em um site desse tipo. Também levamos em conta a estrutura básica que percebemos em portais de serviços e em notícias relacionadas à área.

Esse levantamento inicial nos permitiu listar os requisitos funcionais e não funcionais que julgamos importantes para o projeto, com o objetivo de que o sistema fosse simples, prático e atendesse aos propósitos que estabelecemos.

Requisitos funcionais

Nº Requisito	Nome	Descrição
RF001	Login e autenticação	O sistema deve permitir que os usuários façam login e autenticação pelo sistema ou com redes sociais para acessar funcionalidades conforme seu perfil.
RF002	Gerenciamento e cadastro de reclamações	O usuário deve ser capaz de criar, editar e excluir reclamações, que serão públicas para visualização de outros usuários e administradores.
RF003	Criação de Comentários e chat de respostas	O administrador pode responder reclamações e iniciar um chat para comunicação com o usuário que criou a reclamação.
RF004	Gerenciamento e cadastro de doenças	O administrador pode criar, editar e excluir doenças relacionadas ao saneamento básico, que serão visíveis para todos os usuários
RF005	Gerenciamento e cadastro de notícias	O administrador pode criar, editar e excluir notícias relacionadas a saúde pública e saneamento básico, que serão visíveis para todos os usuários
RF006	Filtragem de reclamações	A tabela de reclamações deve ser filtrável por pontuação, categoria e localização geográfica.
RF007	Upload de imagens	O sistema deve permitir o upload de imagens relacionadas aos registros para exibição no site.
RF008	Logging de comentários e reclamações	Ações de criação, edição e exclusão de comentários e reclamações devem ser registradas em uma tabela de log.
RF009	Geração de pontuação para reclamações	As reclamações devem ser classificadas por pontuação, baseada no detalhamento da reclamação e no perfil do usuário criador, para aumentar a confiabilidade.

Quadro 1 - Requisitos funcionais

Requisitos não funcionais

Nº Requisito	Nome	Descrição
RNF001	Acesso à internet	O usuário deve dispor de conexão ativa à internet para acessar o site.
RNF002	Banco de dados	O sistema deve utilizar banco de dados relacional para armazenamento e gerenciamento dos dados.
RNF003	Responsividade	O site deverá ser responsivo, adaptando-se a diferentes resoluções de tela, com foco em uma boa experiência tanto em dispositivos móveis quanto em desktops.
RNF004	Hardwares mínimos	O sistema poderá ser acessado por dispositivos com navegador moderno (Chrome 90+, Firefox 90+, Edge 90+, Safari 14+), e resolução mínima de 360x640. Recomendado uso em telas maiores (1366x768+) para administradores.
RNF005	Hospedagem	O site deve estar hospedado em servidor que ofereça suporte a banco de dados e recursos web necessários para seu funcionamento.
RNF006	Acessibilidade	O site deverá seguir diretrizes de acessibilidade (como WCAG), garantindo uso por pessoas com deficiência visual, incluindo cegos e daltônicos, além de oferecer recursos que melhorem a usabilidade para todos os usuários.
RNF007	Feedback ao usuário	O sistema deve exibir notificações visuais ou sonoras para informar o usuário sobre erros, exceções ou a conclusão bem-sucedida de operações.
RNF008	Backend e API	O sistema deve possuir backend robusto, expor API RESTful e integrar-se com banco de dados em nuvem, garantindo segurança e desempenho adequados.

RNF009	Documentação	O projeto deve conter documentação atualizada, seguindo práticas das metodologias ágeis SCRUM e Kanban.

Quadro 2 - Requisitos não funcionais

Planejamento de sprints

Relatórios das sprints

O desenvolvimento do SaneaSP foi dividido em 7 sprints, com foco inicial na interface e posterior integração com backend, banco de dados e autenticação. As funcionalidades foram implementadas gradualmente, incluindo acessibilidade, feedback ao usuário e documentação. A entrega final consolidou o sistema funcional e bem documentado até 20/06.

O relatório detalhado das sprints se encontra no repositório principal em

- https://github.com/MathGueff/saneasp-documentation/

Para cada sprint foram criadas branchs contendo o relatório detalhado da sprint e como contribuíram para os requisitos do nosso projeto.

Sprint	Data	Relatório (link)
1	17/03/2025 - 01/04/2025 (15 dias)	Ver relatório da sprint 1
2	01/04/2025 - 16/04/2025 (15 dias)	Ver relatório da sprint 2
3	17/04/2025 - 28/04/2025 (11 dias)	Ver relatório da sprint 3
4	28/04/2025 - 12/05/2025 (14 dias)	Ver relatório da sprint 4
5	13/05/2025 - 27/05/2025 (14 dias)	Ver relatório da sprint 5
6	28/05/2025 - 10/06/2025 (13 dias)	Ver relatório da sprint 6
7	10/06/2025 - 24/06/2025 (14 dias)	Ver relatório da sprint 7

Quadro 3 - Relatório das sprints

Apresentação final

Link para o protótipo funcional:

- https://www.figma.com/proto/MG7Q5GWGymhm1LSMIoI72c/SaneaSP

Vídeo do produto final:

- https://youtu.be/EcFkEs_aKWA

Protótipo

Descrição do protótipo

O protótipo do nosso site abrange todas as telas e fluxos de navegação, representando fielmente a estrutura e a experiência esperadas no produto final. Priorizamos o uso de medições consistentes e padronização dos espaçamentos entre os elementos, garantindo um visual harmonioso e coeso.

Para ações como cadastro, exclusão ou edição de elementos, utilizamos frames duplicados com a simulação de um fundo escurecido e o modal centralizado, ilustrando de forma clara o funcionamento do sistema de alertas e modais da aplicação.

Link para o protótipo

Link para o protótipo:

- https://www.figma.com/design/MG7Q5GWGymhm1LSMIoI72c/SaneaSP

Link para a apresentação do protótipo (preview)

- https://www.figma.com/proto/MG7Q5GWGymhm1LSMIoI72c/SaneaSP

Documentação do projeto

Diagrama de caso de uso

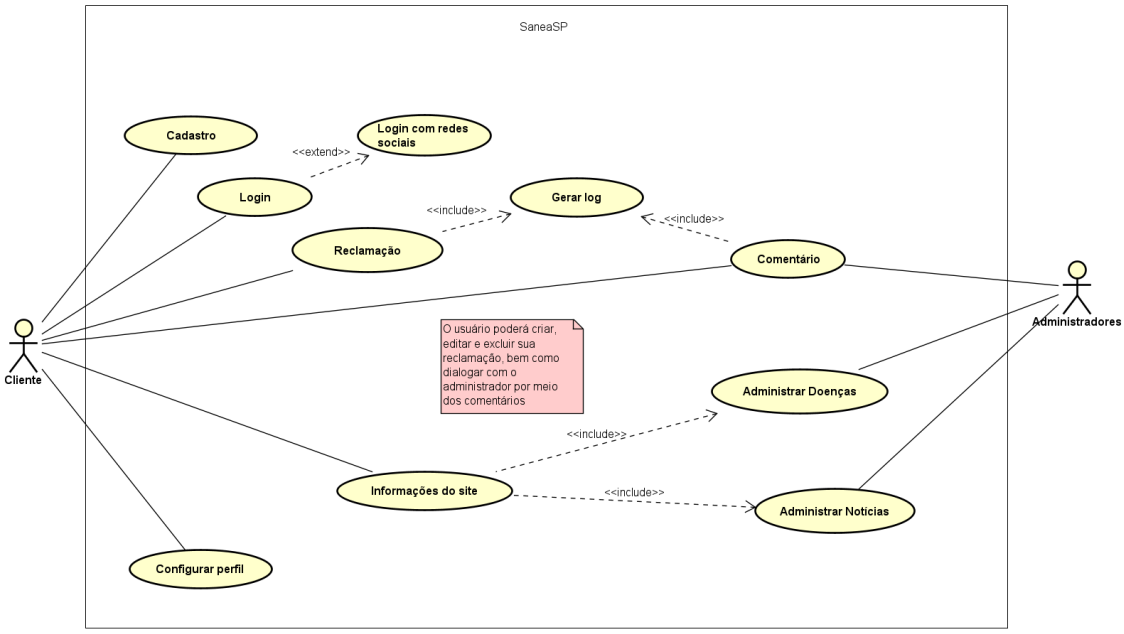

Figura 1 - Diagrama de caso de uso

Backlog do produto

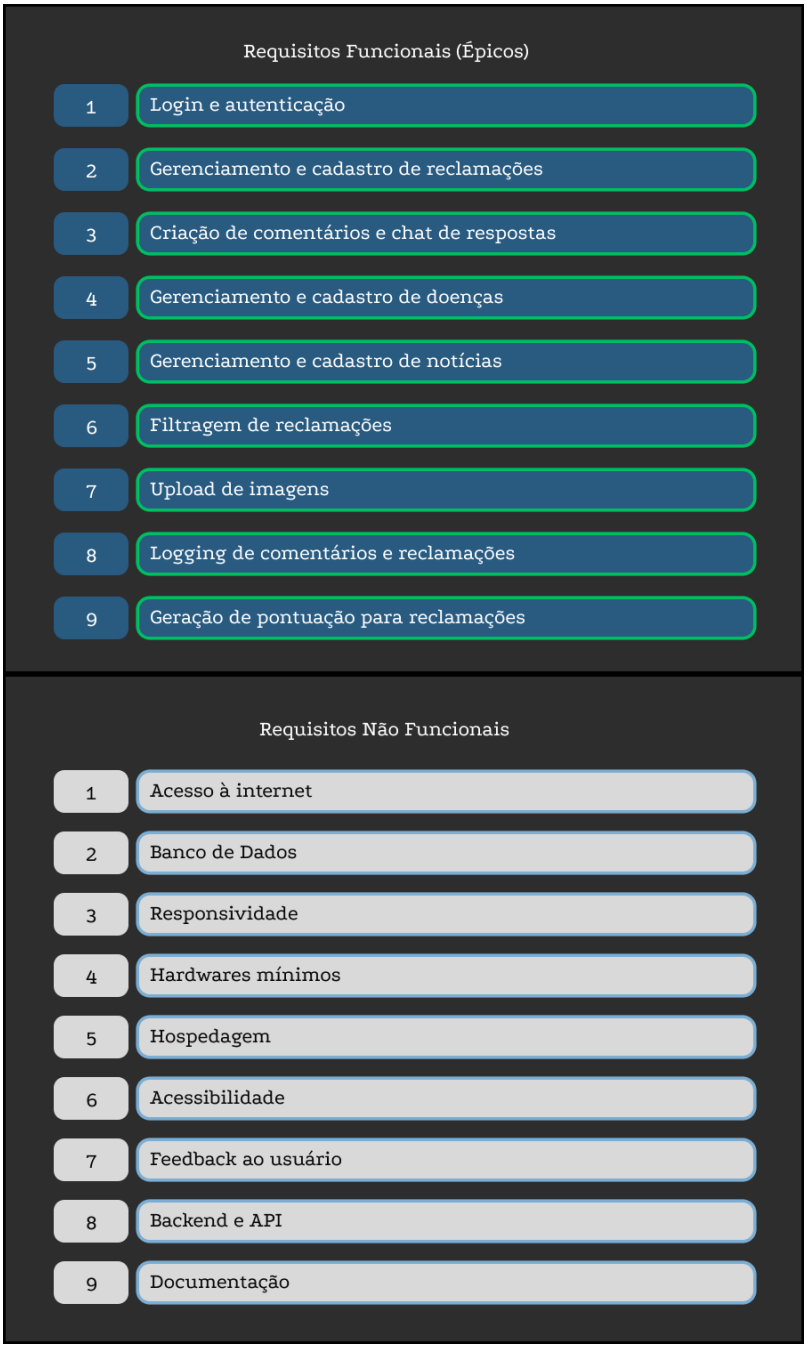

Figura 2 - Backlog do produto

Backlog das sprints

Figura 3 - Backlog das sprints

User Stories

Épico	ID	Ator	Ação	Motivo
1	1	Cidadão	Fazer login	Para acessar o perfil e as funcionalidades básicas da aplicação, também podendo realizar o login através das redes sociais.
1	2	Administrador	Fazer login	Para acessar o perfil e as funcionalidades da aplicação.
2	3	Cidadão	Criar reclamação.	Para relatar um ocorrido na página de reclamações da aplicação.
2	4	Cidadão	Atualizar reclamação.	Para atualizar uma reclamação que tenha registrado.
2	5	Cidadão.	Visualizar reclamações.	Para visualizar suas reclamações e as de outros usuários na página de reclamações da aplicação.
2	6	Cidadão.	Excluir reclamação.	Para excluir suas reclamações da página de reclamação da aplicação.
2	7	Administrador.	Visualizar reclamações.	Para visualizar as reclamações na página de reclamações da aplicação.
3	8	Cidadão.	Responder comentário.	Para responder a um comentário iniciado pelo administrador.
3	9	Administrador.	Fazer comentário.	Para iniciar uma conversa com o usuário que fez uma reclamação.
4	10	Cidadão.	Visualizar doenças.	Para visualizar as doenças na página de doenças da aplicação.
4	11	Administrador.	Cadastrar doença.	Para cadastrar uma nova doença na página de doenças da aplicação.
4	12	Administrador.	Atualizar doença.	Para atualizar uma doença na página de doenças da aplicação.
4	13	Administrador.	Visualizar doenças.	Para visualizar doenças na página de doenças da aplicação.
4	14	Administrador.	Excluir doença.	Para excluir uma doença na página de doenças da aplicação.
5	15	Cidadão.	Visualizar notícias.	Para visualizar as notícias na página de notícias da aplicação.
5	16	Administrador.	Cadastrar notícia.	Para cadastrar uma notícia na página de notícias da aplicação.
5	17	Administrador.	Atualizar notícia.	Para atualizar uma notícia na página de notícias da aplicação.
5	18	Administrador.	Visualizar notícias.	Para visualizar as notícias na página de notícias da aplicação.
5	19	Administrador.	Excluir notícia.	Para excluir uma notícia na página de notícias da aplicação.
8	20	Administrador.	Gerenciamento de log.	Para ter o registro de todos os cadastros, atualizações e exclusões de reclamações e comentários para controle e visualização dos administradores.
9	21	Cidadão.	Visualizar pontuação de reclamação.	Para visualizar as reclamações com maior pontuação na página de reclamações.
9	22	Administrador.	Visualizar pontuação de reclamação.	Para visualizar as reclamações com maior pontuação na página de reclamações.

Figura 4 - User Stories

Projeto do Software

Arquitetura da aplicação

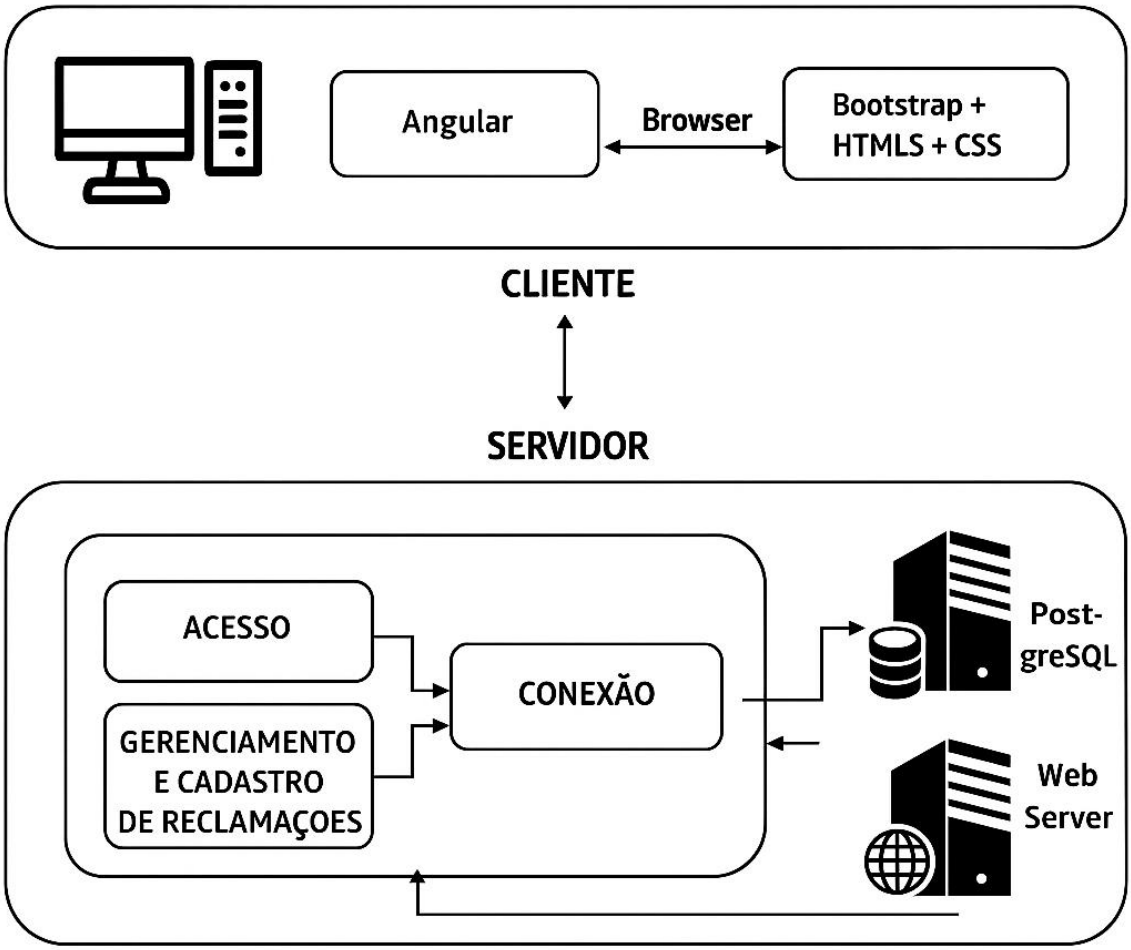

Figura 5 - Arquitetura da aplicação

Tecnologias utilizadas

Front-end (Cliente):

- Angular – Framework SPA para construção da interface.
- TypeScript – Linguagem utilizada no Angular, que oferece tipagem estática e recursos avançados sobre JavaScript.
- Bootstrap – Framework CSS para design responsivo.
- HTML5 – Linguagem de marcação para estrutura da interface.

- CSS – Linguagem de estilo para personalização visual.

Back-end (Servidor):

- Node.js – Ambiente de execução JavaScript no lado do servidor.
- Express.js – Framework web para gerenciamento de rotas, middleware e APIs REST.
- Sequelize – ORM utilizado em todas as etapas do projeto para mapear entidades e interagir com o banco de dados.
- SQLite – Banco de dados leve utilizado durante o desenvolvimento e testes iniciais.
- PostgreSQL – Banco de dados relacional utilizado na versão final do sistema, em conjunto com o Sequelize, oferecendo maior robustez e escalabilidade.

Durante o desenvolvimento, o Sequelize foi utilizado desde o início como ORM principal. Inicialmente, optamos por SQLite para facilitar a prototipação e os testes. Após a definição da estrutura final, migramos o banco de dados para PostgreSQL, mantendo o Sequelize como camada de persistência.

Integração e papéis da equipe

Nossa equipe foi dividida para trabalhar os diferentes desafios no desenvolvimento do projeto, em uma análise geral, a maioria dos integrantes trabalhou tanto no desenvolvimento frontend quanto no backend. A documentação em sua maioria foi feita por um único integrante com ajuda de documentos necessários pelo resto da equipe, e a prototipagem foi realizada por todos do grupo. Abaixo a separação de responsabilidades:

Davy Oliveira Ribeiro (desenvolvedor e PO)

- Desenvolveu as interfaces da funcionalidade de Reclamações, prezando pela clareza e usabilidade.
- Propôs melhorias no design e na qualidade do código ao longo do projeto.

- Contribuiu com o protótipo no Figma
- Desenvolveu a API de Reclamações, incluindo a integração com as tabelas de TagReclamacao e ImagemReclamacao.
- Foi responsável pela criação da tabela de Reclamações no Sequelize e pela implementação das operações CRUD relacionadas.
- Foi responsável pela integração das rotas da API de Reclamações com o front-end.
- Atuou como Product Owner, organizando prioridades e mantendo o foco da equipe nos objetivos principais.
- Desenvolveu a aplicação em Java para a documentação de Técnicas de Programação II

Matheus Augusto Santos Gueff (desenvolvedor e Scrum Master)

- Organizou o conteúdo de cada sprint, contribuindo para o planejamento ágil do projeto.
- Participou ativamente do desenvolvimento do protótipo funcional no Figma.
- Responsável pela organização das documentações técnicas e funcionais.
- Desenvolveu as interfaces do dashboard do administrador, incluindo modais de cadastro, edição e exclusão de tags, bem como a página com a tabela de exibição de tags.
- Implementou o back-end das funcionalidades de Tags, Imagens da Reclamação e Tags da Reclamação utilizando Sequelize e Express.
- Criou as tabelas tag, imagemReclamacao e tagReclamacao no Sequelize, implementando as operações CRUD e os relacionamentos entre essas entidades.
- Desenvolveu o sistema de autenticação com JWT, incluindo login e retorno do usuário autenticado.
- Realizou a integração entre o front-end e as rotas de Tags e Autenticação, garantindo o funcionamento completo dessas funcionalidades.

- Realizou a hospedagem do back-end na plataforma “Render”, adaptando o Sequelize para funcionar com o banco de dados PostgreSQL fornecido pelo serviço.
- Criou a documentação Swagger da API, facilitando o entendimento e a integração com os serviços desenvolvidos.

Pedro Silva Martins (desenvolvedor)

- Contribuiu para a elaboração da documentação do projeto.
- Auxiliou na criação da tabela de Tags no banco de dados.
- Contribuiu com o desenvolvimento do protótipo.

Ryan Carlo Negretti Pereira (desenvolvedor)

- Desenvolveu as interfaces de exclusão de notícias no sistema.
- Criou a API de Usuários, permitindo o cadastro e gerenciamento de contas.
- Responsável pela criação da tabela de usuários no Sequelize e pela implementação das operações CRUD com o banco de dados SQLite.
- Realizou a integração do cadastro de usuários com o front-end.
- Contribuiu com o desenvolvimento do protótipo funcional.
- Prestou suporte na criação de documentações técnicas.
- Desenvolveu a aplicação em Java para a documentação de Técnicas de Programação II