**JERONIMO BARBIERI JUNIOR**
**MIGUEL MIRANDA MORANDINI**
**PEDRO FERREIRA LEITE**
**RAFAEL ROMANO SILVA**

# Cidade Unida

2024

**SUMÁRIO**

## 1. SOBRE A EMPRESA

A Cidade Unida, visa ajudar o desenvolvimento de uma cidade extremamente bem-organizada, onde todas as estruturas funcionem de maneira eficaz e eficiente. Deseja promover a sustentabilidade em todos os aspectos, seja no uso consciente de recursos naturais ou na implementação de soluções ecológicas. Busca uma cidade limpa, não apenas em termos de lixo, mas também no que diz respeito à poluição do ar e da água. O objetivo da empresa é minimizar os problemas que afetam a qualidade de vida dos cidadãos, através de melhores políticas públicas, de uma administração eficiente e de uma maior participação dos cidadãos nas denúncias de problemas que afetam a comunidade.

## 2. PROJETO

### A. IDEIA INICIAL

A ideia inicial do software Cidade Unida é promover a sustentabilidade e melhorar a qualidade de vida em áreas urbanas. Pretende-se alcançar isso facilitando a denúncia de problemas ambientais, promovendo a participação dos cidadãos na tomada de decisões e implementando políticas públicas eficazes relacionadas à qualidade de vida.

### B. TECNOLOGIAS

O projeto Cidade Unida utilizou e continuará a usar as seguintes tecnologias para seu desenvolvimento: HTML, CSS, Javascript (Web); *Flutter* (iOS, Android); *Figma* (Protótipo Interativo); Sebrae Canvas (*Business Model Canvas*); *Dia Diagram* (Modelagem de Banco de Dados).

### C. SOLUÇÃO PROPOSTA

O software Cidade Unida oferecerá funcionalidades como um sistema de gerenciamento de denúncias de problemas ambientais, uma plataforma de participação cidadã para engajar os moradores na tomada de decisões, um sistema de monitoramento ambiental para acompanhar a qualidade do ar, da água e outros aspectos ambientais, e um sistema de gestão de políticas públicas para facilitar a implementação de iniciativas de sustentabilidade.

## I. CICLO DE VIDA

O ciclo de vida do software é uma abordagem sistemática para o desenvolvimento e manutenção de sistemas de software ao longo de seu tempo de vida. Ele descreve as diferentes fases pelas quais um projeto de software passa desde sua concepção até sua desativação.

O modelo de ciclo de vida utilizado no desenvolvimento deste software foi o Modelo de Prototipagem, que de forma consiste em desenvolver um protótipo do software, testá-lo e retrabalhá-lo, até que se alcançasse o nível desejado do sistema. Também se cria uma base sólida desenvolvendo o software através de etapas dentro da prototipagem, como ilustra a Imagem 1.

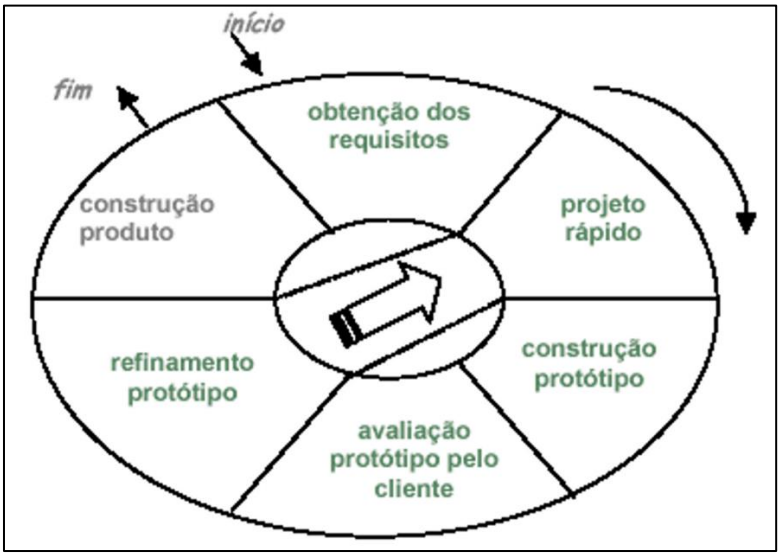


*Imagem 1. Etapas do Modelo Prototipagem*

Este modelo foi utilizado pois, permite que os desenvolvedores obtenham feedback dos usuários mais cedo no processo de desenvolvimento. Isso ajuda a identificar e corrigir problemas de usabilidade, requisitos mal compreendidos ou falta de recursos desejados pelos usuários antes mesmo da implementação completa do sistema. Ao criar protótipos iterativos do software, os desenvolvedores podem identificar e mitigar riscos técnicos e de negócios de forma mais eficaz. Isso permite que problemas sejam abordados em estágios iniciais do projeto, reduzindo o risco de falhas graves durante a implementação.

Também a abordagem de prototipagem permite que as equipes de desenvolvimento economizem tempo e recursos, pois podem validar e refinar conceitos e funcionalidades rapidamente, sem a necessidade de investir em implementações completas e detalhadas desde o início do projeto.

A criação de protótipos visuais e interativos facilita a comunicação entre os membros da equipe de desenvolvimento, *stakeholders* e usuários finais. Isso ajuda a garantir que todos tenham uma compreensão clara das expectativas e requisitos do projeto, facilitando a colaboração e o alinhamento de objetivos.

Além deste modelo ser altamente adaptável e flexível às mudanças de requisitos e feedback dos usuários. As iterações frequentes permitem que o software seja ajustado e refinado ao longo do tempo para atender às necessidades em constante evolução do negócio e dos usuários. Ao focar o desenvolvimento em torno de protótipos que são testados e validados pelos usuários finais, o modelo de prototipagem garante que o software final atenda às necessidades e expectativas dos usuários, resultando em uma melhor experiência do usuário e maior satisfação.

Para o desenvolvimento do Cidade Unida, utilizou-se a metodologia Kanban. De forma suscinta o Kanban é um sistema visual de gestão de trabalho ou uma metodologia ágil, que busca conduzir cada tarefa por um fluxo predefinido de trabalho. Ou seja, um fluxo de trabalho que busca indicar (e limitar) o trabalho em andamento ou WIP (*Work In Progress*).

## II. ETAPAS

O Modelo de Prototipagem é dividido em 7 etapas de trabalho, desde os requisitos que devem estar presentes até a implementação final do software:

- **Etapa 1 – Análise de Requisitos**

A análise de requisitos trata-se do processo de compreensão e identificação das necessidades que o cliente espera ser solucionado pelo sistema que será desenvolvido, definindo a função que o software vai desempenhar.

Os requisitos podem ser classificados em três: Requisitos Funcionais, Requisitos Não Funcionais e Regras de Negócios.

✓ **Requisitos Funcionais**

Um Requisito Funcional é uma descrição das funções que um sistema ou software deve executar. Em outras palavras, ele especifica o que o sistema deve fazer. Esses requisitos são geralmente declarados em termos de entradas, saídas e comportamentos esperados do sistema em resposta a várias entradas. Eles são importantes porque formam a base para o design e

desenvolvimento do software, ajudando a garantir que o produto final atenda às necessidades e expectativas dos usuários. Para o software desenvolvido levantou-se os seguintes Requisitos Funcionais:

- **RF01 – Cadastro de Usuário**

  O sistema deve permitir que os usuários se cadastrem, fornecendo nome, e-mail, senha e telefone.

- **RF02 – Login de Usuário:**

  O sistema deve permitir que os usuários façam login usando e-mail e senha.

- **RF03 – Recuperação de Senha**

  O sistema deve permitir que os usuários recuperem suas senhas por meio de um e-mail de recuperação.

- **RF04 – Denúncia Urbana**

  O sistema deve permitir que usuários logados façam denúncias de problemas urbanos por meio de um formulário.

- **RF05 – Denúncia Anônima**

  O sistema deve permitir que usuários ocultem seus dados pessoais ao fazer uma denúncia, desde que estejam logados.

- **RF06 – Visualização de Perfil**

  O sistema deve permitir que usuários visualizem e editem seus dados pessoais em um perfil após o login.

- **RF07 – FAQ e Suporte**

  O sistema deve oferecer uma seção de ajuda com perguntas frequentes e formas de contato para suporte.

- **RF08 – Contato com a Equipe**

  O sistema deve permitir que os usuários enviem mensagens ou solicitem suporte

por meio de um formulário de contato.

- **RF09 – Listagem de Denúncias para Administrador (Painel de Administração)**

O sistema deve permitir que o administrador visualize todas as denúncias registradas por todos os usuários. Essa funcionalidade tem como objetivo fornecer ao administrador uma visão geral das denúncias e facilitar a gestão e análise das informações reportadas.

- **RF10 – Administrador poderá atualizar o Status da Denúncia (Painel de Administração)**

O sistema deve permitir que o administrador altere o status da denúncia pode ser atualizado pelo administrador de acordo com os seguintes estados pré-definidos:

  - Em Análise;
  - Em Andamento;
  - Encaminhado/Concluído.

✓ **Requisitos Não Funcionais**

Um Requisito Não-Funcional na engenharia de software refere-se a critérios que não estão diretamente relacionados com a funcionalidade específica do sistema, mas sim com as qualidades que o sistema deve ter. Esses requisitos são igualmente cruciais para o sucesso do software, pois afetam diretamente a experiência do usuário, a eficiência do sistema e a capacidade de manutenção e evolução a longo prazo. Para o software desenvolvido levantou-se os seguintes Requisitos Não-Funcionais:

- **RNF01 – Segurança**

O sistema deve garantir a segurança dos dados dos usuários, especialmente durante o cadastro, login e denúncias.

- **RNF02 – Acessibilidade**

O site deve ser acessível para todos os usuários, incluindo aqueles com deficiências, utilizando práticas de design inclusivo.

- **RNF03 – Performance**

O sistema deve ser capaz de lidar com um número significativo de usuários simultaneamente, sem comprometer a performance.

- **RNF04 – Responsividade**

O site deve ser totalmente responsivo e funcionar adequadamente em dispositivos móveis e desktops.

- **RNF05 – Usabilidade**

O sistema deve ser fácil de usar, com navegação intuitiva e informações claras para todos os tipos de usuários.

- **RNF06 – Desempenho**

A listagem e a filtragem de denúncias no painel de administração devem ser rápidas.

- **RNF07 – Segurança**

O acesso ao painel de administração deve ser protegido por autenticação (login e senha), com o uso de conexões seguras (HTTPS). As credenciais do administrador devem ser armazenadas de forma segura no banco de dados.

✓ **Regras de Negócio**

Já as Regras de Negócio são declarações que definem ou restringem algum aspecto do comportamento do negócio. Elas representam as políticas, diretrizes, práticas ou restrições que uma organização estabelece para guiar suas operações e processos. Essas regras são derivadas dos objetivos estratégicos, regulamentos governamentais, práticas do setor e requisitos dos stakeholders. Para o software desenvolvido levantou-se as seguintes Regras de Negócio:

- **RN01 – Login Obrigatório para Denúncias**

Todas as denúncias devem ser feitas por usuários logados, mesmo que optem por fazer a denúncia de forma anônima.

- **RN02 – Denúncia Anônima**

Ao optar por uma denúncia anônima, os dados pessoais cadastrados não devem ser vinculados à denúncia, mas o usuário ainda precisa estar logado para fazê-la.

- **RN03 – Publicação de Denúncias**

As denúncias devem ser públicas, mas os dados pessoais dos denunciantes podem ser ocultados caso a opção anônima seja escolhida.

- **RN04 – Recuperação de Senha**

O processo de recuperação de senha deve ser feito por meio de um link enviado ao e-mail registrado do usuário.

- **RN05 – Edição de Perfil**

Os usuários devem ter a opção de editar suas informações pessoais a qualquer momento após o login.

- **RN06 – Armazenamento de Dados**

As informações fornecidas pelos usuários devem ser armazenadas de forma segura, seguindo as diretrizes de privacidade e segurança.

- **RN07 – Validação de Formulários**

Todos os formulários devem ser validados para garantir que as informações fornecidas sejam completas e corretas antes de serem enviadas.

- **RN08 – Acesso Exclusivo ao Painel de Administração**

Apenas o administrador registrado pode acessar o painel de denúncias. Usuários comuns não devem ter acesso a essa área do sistema.

- **RN09 – Status da Denúncia**

O administrador não pode criar ou excluir novos estados de status.

- **RN10 – Exclusão de Denúncias**

O administrador pode excluir denúncias, mas apenas se elas não estiverem no

estado "Encaminhado/Concluído". O sistema deve solicitar confirmação ao administrador antes de uma denúncia ser permanentemente removida.

- **RN11 – Registro de Ações Administrativas**

Toda alteração feita pelo administrador no painel de denúncias (como alterar o status ou excluir uma denúncia) deve ser registrada em um log de auditoria, com data e hora da ação, identificação do administrador e descrição da mudança.

O log de auditoria não pode ser modificado nem removido pelo administrador.

- ***LOGGING*:**

**Descrição:** O sistema deve registrar eventos importantes em arquivos de log ou banco de dados, permitindo a auditoria, rastreamento de ações e análise de erros. O log deve conter informações detalhadas sobre operações críticas, como tentativas de login, criação de denúncias, modificações de status e ações de administradores.

**Critérios de Aceitação:** O sistema deve registrar os seguintes eventos:

- Acesso de usuários ao sistema (logins, logouts).
- Falhas de autenticação (tentativas de login inválidas).
- Ações de administradores (criação, modificação ou exclusão de denúncias, usuários, categorias).
- Criação e edição de denúncias pelos usuários.
- Mensagens de erro (falhas no sistema, erros de banco de dados, exceções).

**Formato do Log:** Cada entrada no log deve incluir:

- Data e hora do evento.
- Descrição do evento.
- Identificação do usuário (ID, nome ou e-mail, se disponível).
- Endereço IP do usuário (se aplicável).
- Tipo de evento (info, aviso, erro).

**Requisitos Técnicos:** O log pode ser armazenado em:

- Arquivos locais (por exemplo: .log).
- Uma tabela específica no banco de dados.
- Os logs devem ser acessíveis para consulta e análise pelos

administradores do sistema.

- Logs de erros críticos devem ser enviados para os desenvolvedores (se configurado) ou para um serviço externo de monitoramento (por exemplo: Sentry).

**Segurança e Privacidade:**

- Os dados sensíveis registrados nos logs, como senhas, devem ser mascarados ou omitidos.
- A retenção dos logs deve seguir as diretrizes de privacidade, sendo removidos ou anonimizados após um período específico (por exemplo, 6 meses).

**Performance:** O processo de *logging* não deve impactar significativamente o desempenho do sistema.

- **Etapa 2 – PLANEJAMENTO**

Durante esta etapa, o plano geral do projeto é elaborado. Isso inclui a definição de metas e objetivos do projeto, a alocação de recursos (como equipe, tempo e orçamento) e o estabelecimento de um cronograma de trabalho. No contexto da prototipagem, o planejamento também pode incluir a definição da abordagem de prototipagem a ser utilizada e a seleção das ferramentas e tecnologias apropriadas.

✓ **PRODUCT BACKLOG**

O *Product Backlog* é uma coleção viva de necessidades do cliente e requisitos de negócios para um produto de software. Ele é constantemente atualizado e priorizado pelo *Product Owner*, que é responsável por representar os interesses dos *stakeholders* e garantir que o time de desenvolvimento esteja trabalhando nas funcionalidades mais importantes e de maior valor para o negócio.

- **Etapa 1 - Análise de Requisitos:**

**Levantamento de Requisitos:** Processo de coleta de informações sobre o sistema a ser desenvolvido.

**Observação:** Observar o ambiente e os usuários para entender suas necessidades.

**Questionários*:*** Formulação de perguntas específicas para obter informações dos

*stakeholders*.

**Entrevistas:** Conversas diretas com stakeholders para obter informações detalhadas.

**Classificação dos Requisitos:** Organização dos requisitos em categorias.

**Requisitos Funcionais:** Descrição das funcionalidades que o sistema deve fornecer.

**Requisitos Não Funcionais:** Requisitos de desempenho, segurança, usabilidade, entre outros.

**Regras de Negócio:** Regras específicas que o sistema deve seguir.

- **Etapa 2 - Planejamento:**

**Ciclos de Vida:** Definição do modelo de desenvolvimento a ser seguido (exemplo: Cascata, Agile, etc.).

**Metodologia:** Estratégia geral para conduzir o projeto.

**Cronograma:** Sequenciamento das atividades ao longo do tempo.

***Business Model Canvas:*** Ferramenta para visualizar e desenvolver modelos de negócios.

- **Etapa 3 - Design e Projeto:**

**Fluxograma:** Representação visual do fluxo de trabalho ou processo.

**Sitemap:** Representação das páginas HTMLs de um site.

***Storyboard:*** Sequência de imagens representando as principais telas ou interfaces de usuário.

***Wireframes:*** Esboços de baixa fidelidade das interfaces de usuário.

**Modelagem de Banco de Dados:** Desenho da estrutura de dados do sistema.

- **Etapa 4 - Codificação:**

**Linguagem:** Escolha da linguagem de programação para implementar o sistema.

- **Etapa 5 - Testes:**

**Automação de Testes:** Utilização de ferramentas para automatizar o processo de teste.

**Caixa Branca:** Teste baseado no código-fonte interno.

**Caixa Preta:** Teste baseado nos requisitos externos, sem conhecimento interno do código.

**Caixa Cinza:** Combinação de técnicas de caixa branca e caixa preta.

- **Etapa 6 - Implementação:**

**Implementação:** Processo de traduzir o projeto e o código em um sistema funcional e operacional.

### ✓ Cronograma – Gráfico De Gantt

O Gráfico de Gantt é uma ferramenta visual para controlar o cronograma de um projeto ou de uma programação de produção, ajudando a avaliar os prazos de entrega e os recursos críticos.

Em nosso projeto foi utilizado, para ajudar os integrantes do grupo a visualizarem as tarefas que já haviam sido concluídas, bem como as que estavam em andamento e as futuras. A imagem X, mostra o Gráfico de Gantt do desenvolvimento do Cidade Unida.

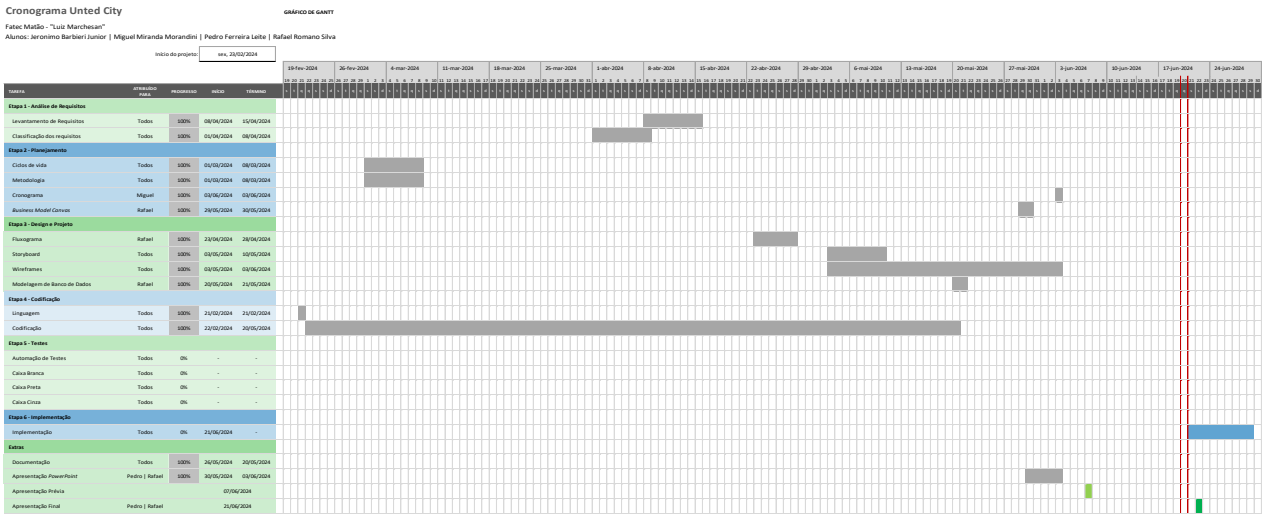


*Tabela 1. Gráfico de Gantt - Cidade Unida*

A visualização pode estar um pouco dificultada devido ao formato do arquivo, então, aqui também consta o link do Gráfico de Gantt do Cidade Unida: *Gráfico de Gantt - Cidade Unida.*

### ✓ Business Model Canvas

O *Business Model Canvas* é uma ferramenta de planejamento estratégico que permite que as empresas descrevam, visualizem, avaliem e modifiquem seu modelo de negócios

existente ou proposto.

Consiste em nove blocos que representam os principais elementos de um modelo de negócios, como segmentos de clientes, propostas de valor, canais de distribuição, relacionamento com clientes, fontes de receita, entre outros. É frequentemente utilizado no desenvolvimento de produtos de software para garantir que o produto atenda às necessidades do cliente e seja comercialmente viável.

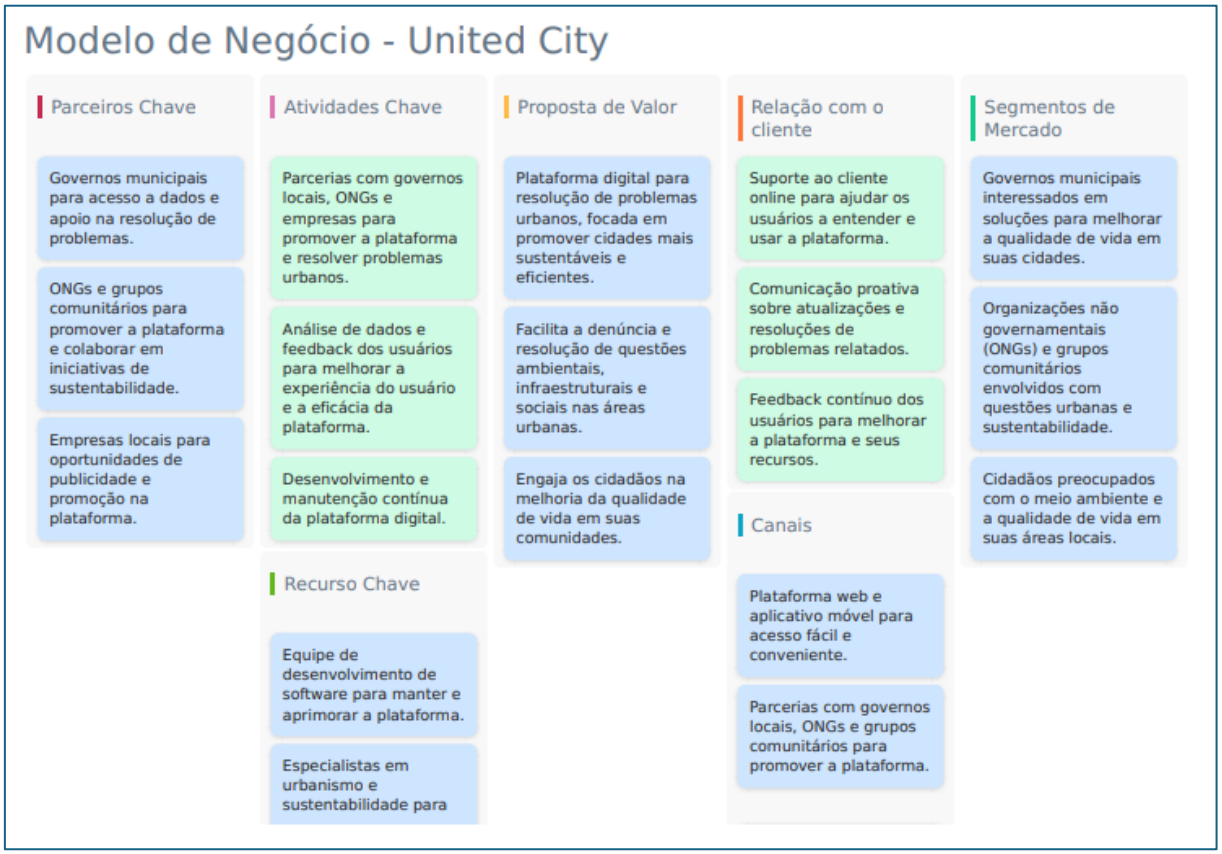


*Imagem 2. Modelo de Negócio Canvas da Cidade Unida*

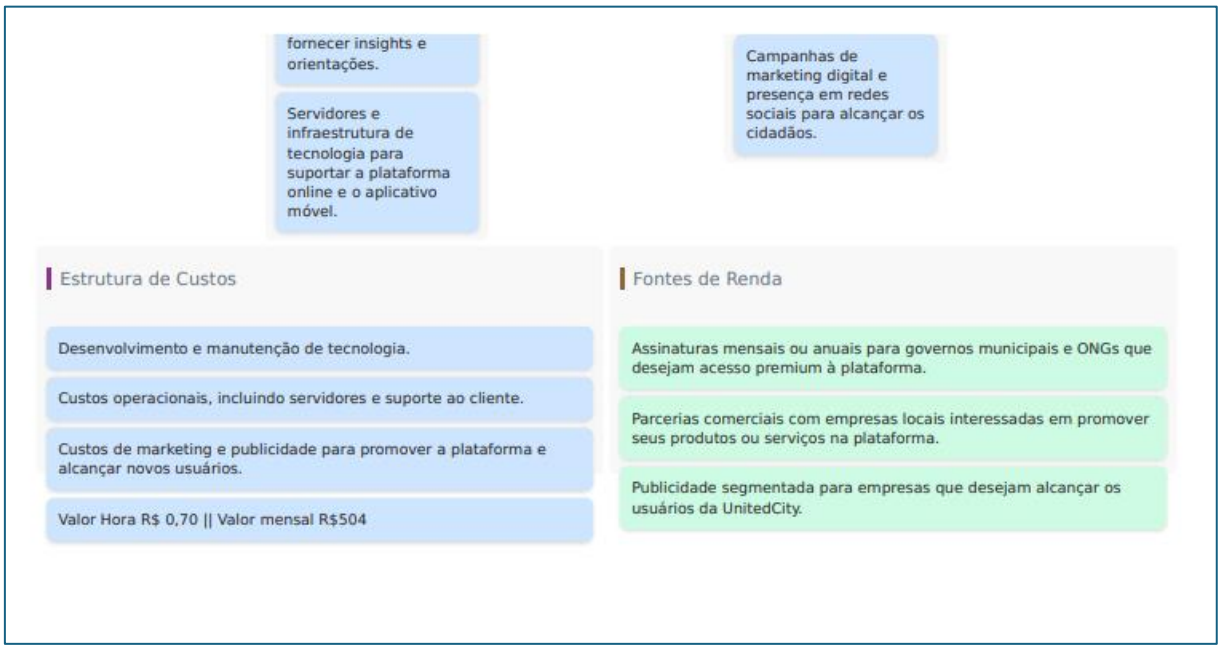


*Imagem 3. Modelo de Negócio Canvas da Cidade Unida*

- **Etapa 3 – DESIGN E PROJETO**

Nesta etapa, os conceitos identificados na análise de requisitos são transformados em um design de sistema detalhado. Isso inclui a criação de esboços de interface do usuário, fluxos de trabalho e arquitetura de sistema. Na prototipagem, o foco está em desenvolver protótipos de baixa fidelidade ou protótipos de software que demonstrem visualmente como o sistema funcionará e como os usuários interagirão com ele.

✓ **Fluxograma:**

Um fluxograma é uma representação gráfica de um processo ou sistema, mostrando a sequência de etapas ou atividades e as relações entre elas. Na engenharia de software, fluxogramas são frequentemente utilizados para modelar o fluxo de informações ou o fluxo de controle dentro de um sistema de software. O Cidade Unida utilizou fluxogramas para representar os processos do site, como mostra Imagem 4.

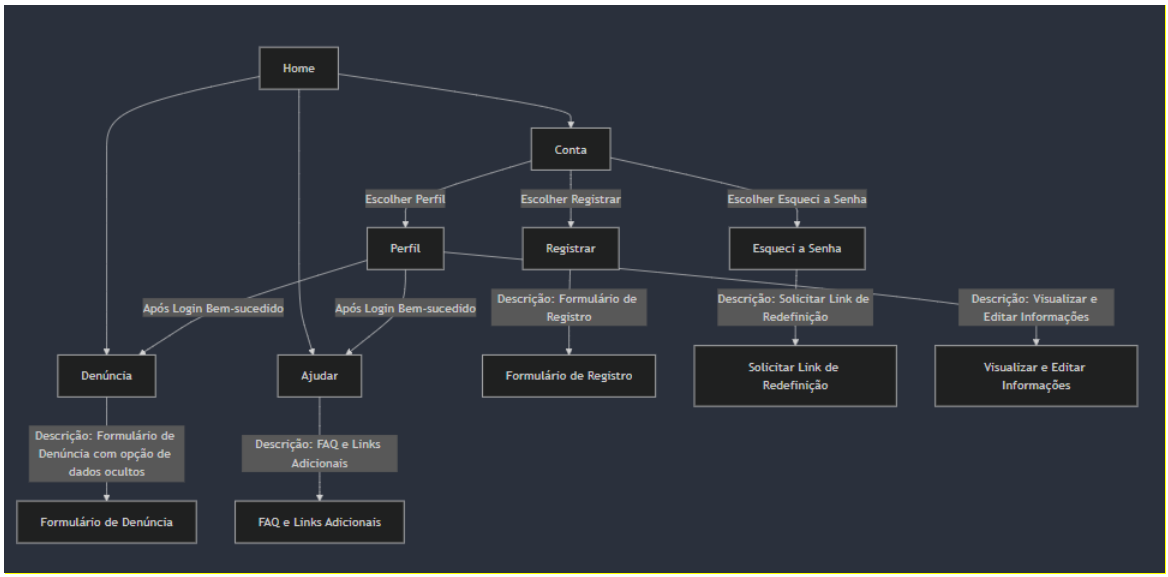


*Imagem 4. Fluxograma do Cidade Unida*

✓ ***Sitemap*:**

Um *sitemap* lista todas as páginas de um site, fornecendo informações sobre a organização e a estrutura do conteúdo. Como mostra Imagem 5, o *sitemap* do Cidade Unida, traz todas as páginas HTMLs do projeto.

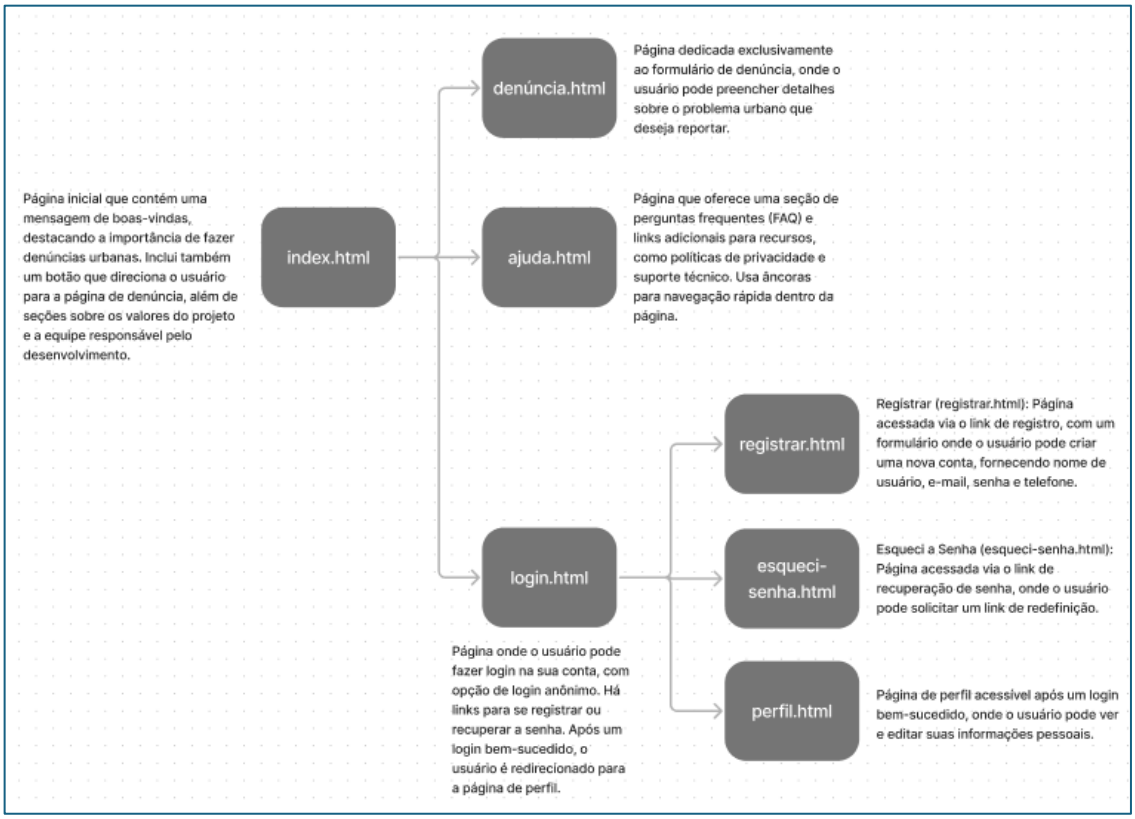


*Imagem 5. Sitemap do Cidade Unida*

✓ **Storyboard:**

Um *storyboard* é uma série de ilustrações sequenciais ou esboços que representam as principais cenas ou eventos de um software. Na engenharia de software, *storyboards* são frequentemente usados para visualizar e comunicar as interações do usuário e o fluxo de tela em um aplicativo ou sistema de software.

✓ **Wireframes:**

Um *wireframe* é um esboço visual de uma interface de usuário, mostrando a disposição geral dos elementos da interface sem detalhes de design. Ele representa a estrutura básica e o layout de uma página ou tela de aplicativo, incluindo elementos como botões, campos de texto e áreas de conteúdo. Na engenharia de software, *wireframes* são usados para planejar e prototipar a interface de usuário de um sistema de software.

Baixa definição: Refere-se a protótipos ou representações que têm um nível de detalhe mínimo ou uma resolução visual baixa. Eles são usados para fornecer uma visão geral ou conceitual de um sistema de software, sem se preocupar com detalhes de design ou implementação. Protótipos de baixa definição são frequentemente usados nas fases iniciais de um projeto de software para explorar conceitos e requisitos. As imagens 6 a 10 demonstram alguns exemplos do *wireframes* de baixa definição do Cidade Unida.

Alta definição: Refere-se a protótipos ou representações que têm um nível de detalhe máximo ou uma resolução visual alta. Eles são usados para fornecer uma representação precisa e detalhada de um sistema de software, incluindo todos os elementos de design e interações de usuário. Protótipos de alta definição são frequentemente usados nas fases posteriores de um projeto de software para validar o design e a usabilidade antes da implementação final.

Links dos Protótipos Interativos do Cidade Unida no *Figma*: Mobile: *Wireframe - Cidade Unida - Mobile*. Tablet: *Wireframe - Cidade Unida - Tablet*. Desktop: *Wireframe - Cidade Unida - Desktop*.

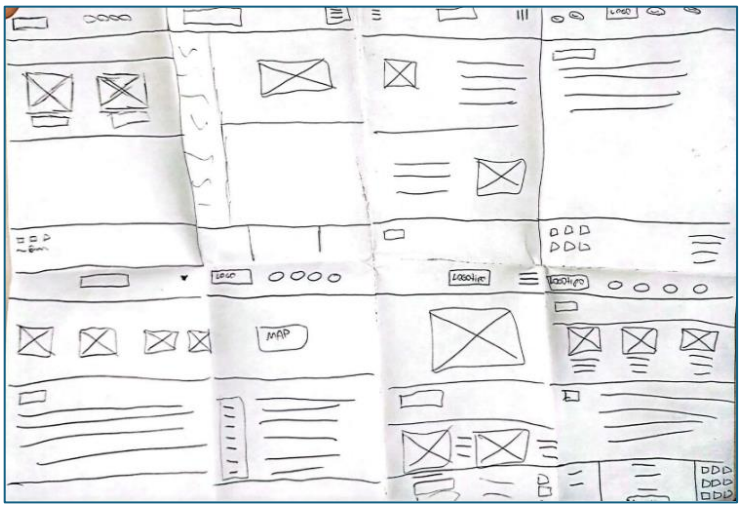


*Imagem 6. Wireframe - Interface Tela Inicial*

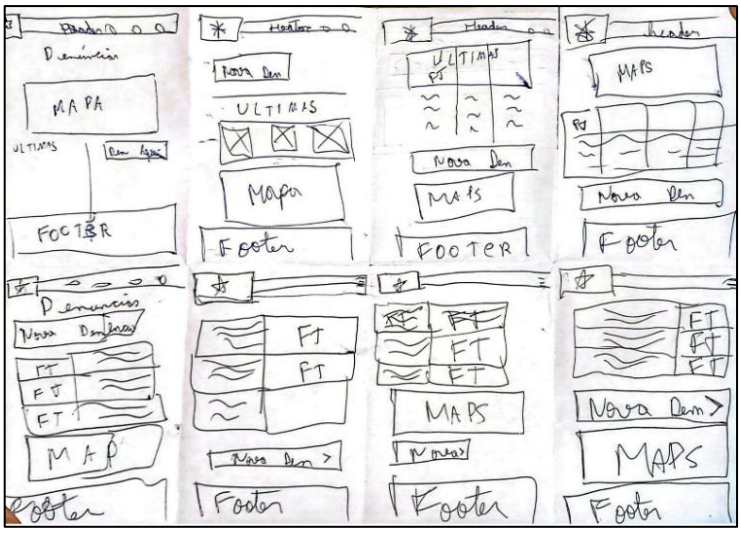


*Imagem 7. Wireframes – Interface de Denúncias*

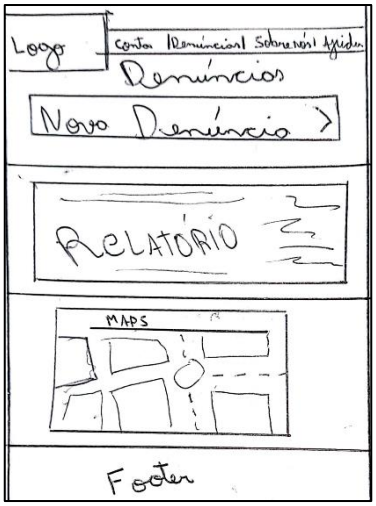


*Imagem 8. Wireframes – Interface de Denúncias*

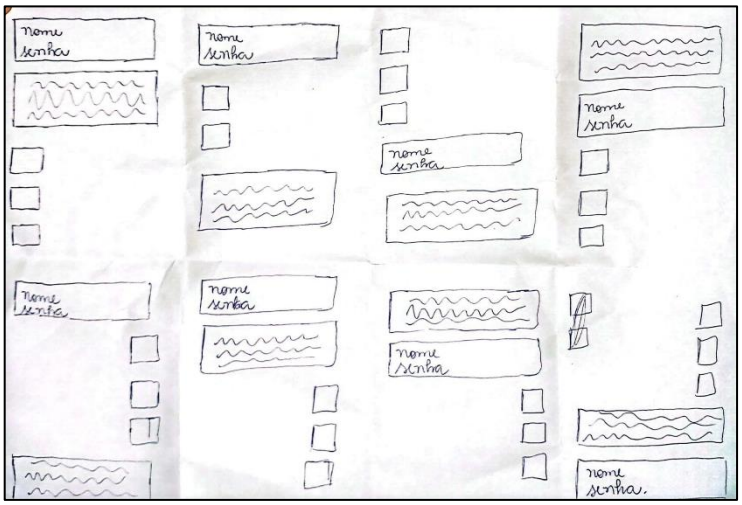


*Imagem 9. Wireframe - Interface de Contas*

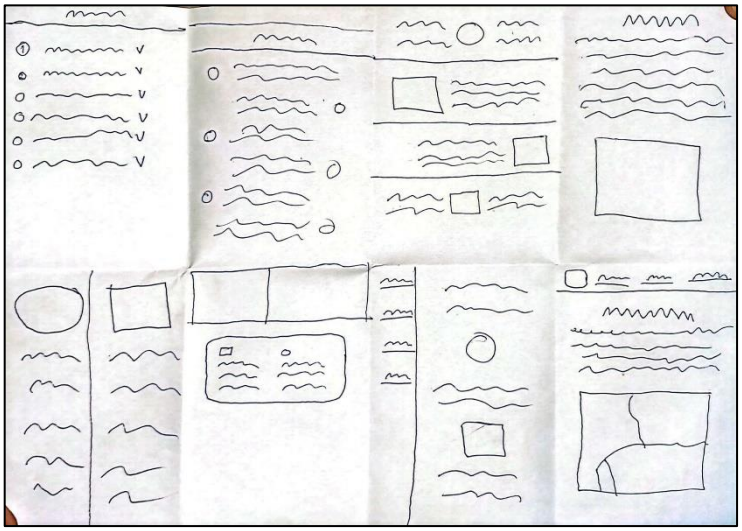

*Imagem 10. Wireframe - Interface de Ajuda*

✓ **Modelagem do Banco de Dados:**

A modelagem do banco de dados envolve o processo de projetar a estrutura e as relações dos dados que serão armazenados em um banco de dados. Isso inclui a identificação das entidades (tabelas), atributos (campos) e relacionamentos entre as entidades. Modelos de banco de dados são frequentemente representados por diagramas de entidade-relacionamento (ER) ou modelos de dados normalizados.

Requisito: Requisitos para um software de denúncia urbana.

a) É necessário armazenar o código exclusivo, nome (composto por Primeiro e Último Nome), e-mail e senha do cidadão que fará a denúncia. Um cidadão realiza várias Denúncias.

b) É necessário armazenar a data, categoria e a descrição da denúncia, bem como o

número de protocolo exclusivo e a localização (composta por Rua, CEP, Bairro, Complemento, Número e Cidade) do problema denunciado. Um problema pode ser denunciado/reportado por vários cidadãos.

c) Uma denúncia deve ser encaminhada para a empresa terceirizada responsável pela resolução do problema. Deve-se guardar o código exclusivo de identificação da empresa terceirizada, além do seu nome, descrição e categorias. Deve-se saber quando a denúncia foi resolvida e qual é a sua prioridade. Uma denúncia pode ser encaminhada para várias empresas terceirizadas. Uma empresa terceirizada pode receber várias denúncias.

Modelo Conceitual: O modelo de dados conceitual é um diagrama de alto nível que representa os conceitos de dados que suportam o negócio de uma empresa, uma área de negócio ou, por exemplo, um sistema de informações. A imagem 11 ilustra o Modelo Conceitual do Banco de Dados do Cidade Unida.

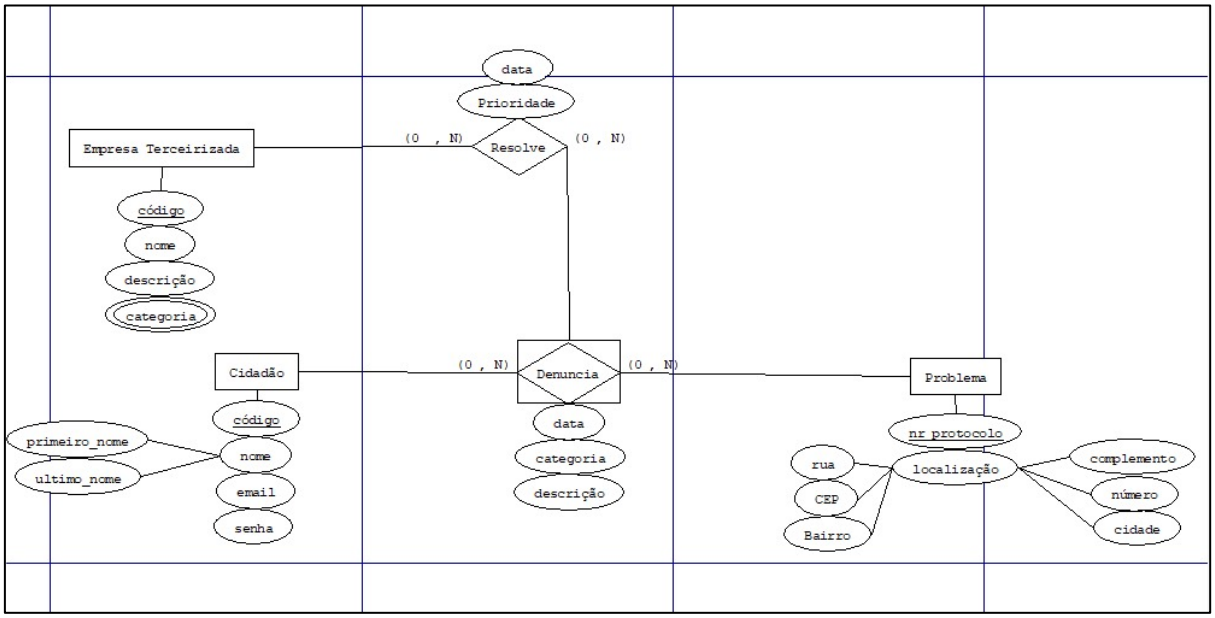


*Imagem 11. Modelo Conceitual do Bando de Dados do Cidade Unida*

- **Etapa 4 – CODIFICAÇÃO**

Durante esta etapa, o software é desenvolvido com base no design e nos protótipos criados nas etapas anteriores. Os programadores escrevem o código-fonte do sistema, seguindo as diretrizes estabelecidas no design. Na prototipagem, a codificação pode envolver o desenvolvimento de protótipos de alta fidelidade ou a criação de versões simplificadas do software que se concentram em demonstrar funcionalidades específicas.

O código fonte do Cidade Unida está disponível no GitHub, através do link: *GitHub - Cidade Unida.*

- **Etapa 5 – TESTES**

Após a codificação, o software é testado para garantir que funcione conforme o esperado e atenda aos requisitos definidos na análise de requisitos. Isso inclui a realização de testes de unidade, testes de integração e testes de sistema para identificar e corrigir quaisquer defeitos ou problemas de funcionamento. Na prototipagem, os testes podem ser usados para validar os conceitos e funcionalidades apresentados nos protótipos, bem como para coletar *feedback* dos usuários para refinamento adicional.

- **Etapa 6 – INSTALAÇÃO E IMPLEMENTAÇÃO**

Na etapa final, o software é instalado e implantado no ambiente de produção. Isso pode envolver a configuração de servidores, a instalação de software em dispositivos de usuários finais e a migração de dados, conforme necessário. Na prototipagem, esta etapa pode incluir a distribuição dos protótipos para os usuários finais testarem e fornecerem *feedback* adicional, ou pode envolver a transição do protótipo para uma versão completa e funcional do sistema.

## III. CRONOGRAMA

Utilizou-se uma lista de atividades denominada Sprint Backlog, onde um conjunto de tarefas devem ser realizadas durante um período específico, sprint.

## IV. EQUIPE

A equipe de trabalho da Cidade Unida é composta pelos seguintes integrantes:

- **Jeronimo Barbieri Junior**

  (jeronimo.barbieri@fatec.sp.gov.br | GitHub: jeronimobarbieri | (16) 99465-5603)

  Desempenhou um papel importante na pesquisa dos concorrentes diretos e indiretos da empresa, na criação e decisão da identidade visual por meio do desenvolvimento do logotipo, na elaboração do *storyboard* e do *wireframe* (mobile, tablet e desktop) e na implementação da codificação do projeto. Adicionalmente, participou na definição do *sitemap* e tomei decisões importantes juntamente ao grupo relacionadas à documentação do projeto.

**Qualificações:** Engenharia de Software I - Nível Básico; Design Digital - Nível Básico; Desenvolvimento Web I - Nível Básico; Modelagem de Banco de Dados - Nível Básico; Algoritmo e Lógica de Programação - Nível Básico.

- **Miguel Miranda Morandini**

(miguel.morandini@fatec.sp.gov.br | GitHub: miguelmorandini | (16) 99227-9896)

Auxiliou o grupo em diversos campos, desde a pesquisa de concorrentes diretos e indiretos, bem com a seleção da paleta de cores do site, a criação do logotipo, além do desenvolvimento de *storyboards* e *wireframe* do Cidade Unida no *Figma*.

Também contribuiu para formação do mapa do site, dando ideias e novos ponto de vistas. Além disso, participou diretamente na formatação e escrita da documentação, junto com os outros integrantes do grupo.

**Qualificações:** Engenharia de Software I - Nível Básico; Design Digital - Nível Básico; Desenvolvimento Web I - Nível Básico; Modelagem de Banco de Dados - Nível Básico; Algoritmo e Lógica de Programação - Nível Básico.

- **Pedro Ferreira Leite**

(pedro.leite16@fatec.sp.gov.br | GitHub: PedroFerreiraLeite | (16) 99395-1471)

Contribuiu para a pesquisa dos sites concorrentes de maneira direta e indireta, participou na decisão do *sitemap*, auxiliou em algumas decisões do conteúdo da documentação, colaborou na escolha da paleta de cores, desenvolveu um dos logotipos, contribuiu na criação do cartaz, além de participar diretamente no desenvolvimento do *wireframe* do site no *Figma*.

**Qualificações:** Engenharia de Software I - Nível Básico; Design Digital - Nível Básico; Desenvolvimento Web I - Nível Básico; Modelagem de Banco de Dados - Nível Básico; Algoritmo e Lógica de Programação - Nível Básico.

- **Rafael Romano Silva**

(rafael.silva741@fatec.sp.gov.br | GitHub: rafaelromwno | (16) 99768-0906)

Durante o projeto em grupo, foi responsável por alguns fatores. Incluindo a pesquisa dos concorrentes diretos e indiretos da empresa, juntamente com meus colegas de equipe. Contribuiu para o desenvolvimento do logotipo e do *wireframe* de baixa e alta fidelidade no *Figma*, abrangendo as versões para mobile, desktop e tablet, novamente com a ajuda do grupo.

Além disso, participou ativamente do desenvolvimento do *sitemap* e da definição da paleta de cores do sistema, sempre colaborando com os outros membros da equipe. Elaborou a modelagem do banco de dados usando o *Dia Diagram*, além de ajudar no levantamento e na classificação dos requisitos do projeto.

Utilizou o *Notion* para fazer a documentação prévia do progresso do grupo e para manter a equipe organizada. Além disso, assumiu um papel fundamental na implementação prática do projeto, ajudando na codificação do sistema. Colaborou na elaboração da apresentação em *PowerPoint*, do *moodboard* e do *Business Model Canvas.*

**Qualificações:** Engenharia de Software I - Nível Básico; Design Digital - Nível Básico; Desenvolvimento Web I - Nível Básico; Modelagem de Banco de Dados - Nível Básico; Algoritmo e Lógica de Programação - Nível Básico.

## 3. CONSIDERAÇÕES FINAIS

Em resumo, o projeto Cidade Unida desenvolveu, em primeira instancia, uma plataforma de intermediação entre os cidadãos e órgãos responsáveis por serviços públicos.

Ademais, contribuiu efetivamente para o desenvolvimento de todos os seus colaboradores, principalmente na área de criação de *wireframes* e prototipação, bem como também na área de codificação HTML e CSS.

## 4. BIBLIOGRAFIAS

ABREU, Leandro. Rock Content. O que é e como criar Sitemaps: guia prático para otimizar seu site. Disponível em: https://rockcontent.com/br/blog/sitemaps/. Acesso em: 12 maio 2024.

BIZZDESIGN. 7 Applications of the Business Model Canvas. Bizzdesign, [s.d.]. Disponível em: https://bizzdesign.com. Acesso em: 02 maio 2024.

DevMedia. Testes de Software - Parte 01. Disponível em: https://www.devmedia.com.br/testes-de-software-parte-01/9409. Acesso em: 28 maio 2024.

GUEDES, Marylene. Ciclo de vida do software: por que é importante saber? [s. d.]. TREINAWEB. Disponível em: https://www.treinaweb.com.br/blog/ciclo-de-vida-software-por-que-e-importante-saber Acesso em: 06 maio 2024.

INTERACTION DESIGN FOUNDATION. What is a Business Model Canvas?. Interaction Design Foundation, 2024. Disponível em: https://www.interaction-design.org. Acesso em: 02 maio 2024.

LEÃO, Thiago. Gráfico de Gantt: o que é, como funciona e como fazer. 3 jul. 2018. Nomus. Disponível em: https://www.nomus.com.br/blog-industrial/grafico-de-gantt/. Acesso em: 3 jun. 2024.

MDN contributors. Mozilla Developer Network (MDN). CSS. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/CSS/. Acesso em: 06 junho 2024.

MDN contributors. Mozilla Developer Network (MDN). HTML5. Disponível em: https://developer.mozilla.org/en-US/docs/Glossary/HTML5. Acesso em: 06 junho 2024.

MEDEIROS, Higor. Introdução a Requisitos de Software. Disponível em: https://www.devmedia.com.br/introducao-a-requisitos-de-software/29580. Acesso em: 14 maio. 2024.

NIELSEN, Jakob. Designing Web Usability: The Practice of Simplicity. New Riders Publishing, 1999. Acesso em: 15 maio 2024.

NORMAN, Donald A. The Design of Everyday Things. Basic Books, 2013. Acesso em: 15 maio 2024.

SABINO, Roberto. Alura. Método Kanban: como funciona e suas vantagens. Disponível em: https://www.alura.com.br/artigos/metodo-kanban. Acesso em: 13 maio 2024.

Semrush. HTML Sitemap: The Benefits for SEO and Users. Disponível em:

https://www.semrush.com/blog/html-sitemap/. Acesso em: 15 maio 2024.

Semrush. What Is a Sitemap? Website Sitemaps Explained. Disponível em: https://www.semrush.com/blog/what-is-a-sitemap/. Acesso em: 15 maio 2024.

Slickplan. How to Create a Sitemap for a Website: HTML, XML, or Visual. Disponível em: https://slickplan.com/blog/how-to-create-a-sitemap. Acesso em: 15 maio 2024.

SMASHING MAGAZINE. Articles on Typography. Disponível em: https://www.smashingmagazine.com/. Acesso em: 02 maio 2024.

STRATEGIC MANAGEMENT INSIGHT. Business Model Canvas: The Ultimate Guide. Strategic Management Insight, [s.d.]. Disponível em: https://strategicmanagementinsight.com. Acesso em: 02 maio 2024.

XML Sitemaps. How to create HTML Sitemaps and how does it help a website. Disponível em: https://www.xml-sitemaps.com/html-sitemap.html. Acesso em: 15 maio 2024.