UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

GABRIEL SOUZA PEPLINSKI

ÁGORA: PARTICIPAÇÃO CIDADÃ NA MANUTENÇÃO URBANA

GUARAPUAVA

2023

GABRIEL SOUZA PEPLINSKI

ÁGORA: PARTICIPAÇÃO CIDADÃ NA MANUTENÇÃO URBANA

Ágora: Citizen Participation in Urban Maintenance

Projeto de Trabalho de Conclusão de Curso de Graduação apresentado à disciplina de Trabalho de Conclusão de Curso 1, do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná, Campus Guarapuava, como requisito parcial para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet.

Orientador: Me. Dênis Lucas Silva

GUARAPUAVA

2023

RESUMO

Nos ambientes urbanos, muitas vezes, o cidadão se depara com problemas relacionados à cidade onde reside, seja por falta de planejamento prévio ou por outro motivo, sempre existem diferentes tipos de ocorrências que afetam o dia a dia dos seus habitantes, reduzindo e impactando a qualidade de vida dessas pessoas. Este trabalho propõe o desenvolvimento de um sistema onde o próprio cidadão, através de seu celular, realiza registro dessas ocorrências que o afetam, além disso, outros usuários também poderão ter acesso a esses dados e acompanhar informações como: quais os problemas mais comuns, quais os locais mais afetados e o que está sendo realizado para resolver cada um. Com a implementação deste sistema, o objetivo é fazer o cidadão ocupar seu lugar na sociedade e exercer seu papel de forma ativa, como um agente fiscalizador dentro do ambiente que está inserido.

Palavras-chave: aplicativo; participação cidadã; problemas urbanos.

ABSTRACT

In urban environments, citizens often encounter problems related to the city where they live in, whether due to lack of prior planning or for other reasons. There are always different types of occurrences that affect the daily lives of its inhabitants, reducing and impacting the quality of life for these people. This work aims to propose the development of a system that the citizen oneself, through a cell phone, can register occurrences that affect his or her life, furthermore, other users will also be able to access this data and follow information such as: the most common problems, the most affected places and what is being done to resolve each one. With the implementation of this system, the main objective is to make citizens assume their role in society and play an active part as a monitoring agent within their surroundings.

Keywords: application; citizen participation; urban problems.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

Siglas

API	*Application Programming Interface*
MER	*Modelo Entidade Relacionamento*
ORM	*Object-Relational Mapping*
SGBD	*Sistema de Gerenciamento de Banco de Dados*
SPOJ	*Sphere Online Judge*
TDD	*Test Driven Development*

SUMÁRIO

1 INTRODUÇÃO

Nesse capítulo será apresentado uma visão geral sobre o trabalho desenvolvido, mostrando o cenário onde ele se aplica, bem como a motivação e justificativas do seu desenvolvimento. Também, os objetivos que precisam ser cumpridos para que o trabalho seja concluído com êxito.

1.1 Considerações iniciais

A qualidade de vida pode ser determinada pela observação de uma série de fatores na vida de uma pessoa, um conjunto de condições que contribuem para o seu bem-estar. Sendo assim, pode incluir, por exemplo, auto-estima, finanças, relações pessoais e satisfação com atividades rotineiras. Porém, é difícil medir de forma geral, a qualidade de vida de uma pessoa, pois é preciso entender quais são os fatores que ela pondera na sua vida, para que se possa dizer se ela tem ou não qualidade de vida.

Contudo, é possível dizer que alguns aspectos são comuns a todas as pessoas, afetando a qualidade de vida. Aspectos profissionais e relacionados à saúde, são exemplos, também, os aspectos sociais, que estão além das relações com outras pessoas, fazendo com que o meio ambiente, ou seja, tudo o que está ao redor, tenha impacto no bem-estar geral das pessoas.

> A título de exemplo, imagine uma cidade com muito lixo, muita poluição e sem espaços verdes. Certamente que serão fatores, por um lado, suscetíveis de causar doenças e por outro, não produzem sentimentos de bem-estar nas pessoas[1].

Todas as cidades, independentemente de seu tamanho, enfrentam desafios que afetam seus cidadãos diariamente. Dentre os problemas mais comuns enfrentados no dia a dia da maioria dos residentes de áreas urbanas, podemos citar: a) Problemas de mobilidade, como ruas em condições precárias, falta de sinalização em ruas ou aveninas perigosas e movimentadas, problemas relacionados à acessibilidade de pessoas com deficiência ou mobilidade reduzida (cadeirantes ou pessoas com algum grau de deficiência visual), por exemplo, falta de rampas e de calçamento com piso tátil direcional; b) Problemas relacionados ao ambiente urbano: Ruas sem iluminação, ambientes públicos em más condições de preservação (falta de manutenção em áreas gramadas como parques e descarte indevido de lixo e resíduos); c) Problemas relacionados a qualidade de vida, como falta de saneamento básico e moradia em áreas impróprias para a habitação; d) E muitos outros. Independentemente do tipo de problema, todos esses desafios colocam em risco a vida e o bem-estar de inúmeras pessoas todos os dias. De acordo com um estudo realizado em 1993 na cidade de São Paulo, foi constatado que a maioria das queixas de problemas na infraestrutura urbana provinha da população de baixa renda, que

[1] https://www.saudebemestar.pt/pt/blog-saude/qualidade-de-vida/

acaba sendo a mais prejudicada devido às condições precárias de urbanização dos lugares onde moram (JACOBI, 2013).

A paisagem urbana está constantemente em crescimento e evolução, à medida que cada vez mais pessoas escolhem viver e trabalhar em áreas urbanas. Infelizmente, em raras ocasiões, esse processo de urbanização é conduzido de forma elaborada e planejada, o que contribui para o surgimento de problemas urbanos. Independentemente de quem são os causadores diretos de cada um dos problemas encontrados nas cidades, é preciso que o poder público, neste caso, as prefeituras, se antevejam aos incômodos causados e comecem a agir. Contudo, pode-se imaginar que tomar conhecimento sobre os problemas de uma cidade não é uma tarefa fácil, principalmente se a cidade é grande e populosa. Para contornar esta situação, muitas prefeituras têm estabelecido canais para abordar esse desafio. Um exemplo é o Portal Carioca Digital[2], o SP156[3] e, em Guarapuava, o Atende Net[4].

Dois grandes problemas podem ser notados neste cenário descrito até o momento, primeiro, a sensação do cidadão que um problema encontrado por ele demorará para ser "visto" pelas autoridades municipais e, segundo, que os canais disponibilizados para "informar" sobre um problema, não funcionam e/ou não são transparentes. Essa falta de transparência faz com que um cidadão fique na dúvida se um problema já foi informado ou não, resultando, muitas vezes, na invisibilidade do problema, pois ninguém relata, achando que alguém já o fez. Mas também, quando o relato é feito, fica a sensação que nada será feito, pois ao passar do tempo, não se vê nenhuma ação por parte do poder público e seus responsáveis.

Consequentemente, as pessoas se acostumam com essas situações adversas sem buscar soluções, muitas vezes por falta de conhecimento sobre a quem recorrer ou por desacreditarem que algo será feito. Porém, em qualquer sociedade, o cidadão desempenha um papel de extrema importância. Além de constituir a sua base, o cidadão detém o poder de escolher seus representantes e, igualmente significativo, atuar como agente fiscalizador. Na Grécia Antiga, considerada o berço da democracia, os cidadãos se reuniam em uma praça pública onde debatiam uma ampla variedade de tópicos, abrangendo desde política, comércio, justiça até administração pública. Essa participação ativa na tomada de decisões e discussões era essencial para cumprir o papel de cidadão. Este lugar era chamado de Ágora, e inspira o nome do sistema proposto neste trabalho. A alusão deste nome com a palavra "agora"[5], encaixa no propósito de uso do sistema, ou seja, no momento que cada cidadão encontrar um problema na sua cidade, seja uma calçada com buracos ou um terreno abandonado servindo de depósito de lixos, ele poderá relatá-lo.

O maior diferencial do *Ágora* frente aos sistema similares mencionados anteriormente, é que todos os relatos ficarão públicos. Desta forma, estarão acessíveis, aumentando a transparência desse processo. Além disso, o sistema *Ágora* pretende contribuir com a participação

[2] https://home.carioca.rio/

[3] https://sp156.prefeitura.sp.gov.br/portal/servicos

[4] https://guarapuava.atende.net/autoatendimento/servicos/ouvidoria-municipal

[5] Nesta hora; neste exato momento; já (https://www.dicio.com.br/agora/).

da sociedade no desenvolvimento urbano, sendo este inclusive um dos 8 objetivos estratégicos da Carta Brasileira para Cidades Inteligentes, emitida em 2021:

> Objetivo Estratégico 7: Fomentar um movimento massivo e inovador de educação e comunicação públicas para maior engajamento da sociedade no processo de transformação digital e de desenvolvimento urbano sustentáveis (CARTA. . . , 2020).

Também, na Primeira Conferência Internacional sobre Promoção da Saúde, realizada em Ottawa, Canadá, em novembro de 1986, os princípios fundamentais da promoção da saúde e as diretrizes para ações e políticas para melhorar a saúde das populações foram discutidas (EUROPE, 1986). Logo, enquanto política pública de promoção da saúde, para melhorar a qualidade de vida da população, é preciso envolver as pessoas no cuidado e zelo das cidades, por meio de um...

> (...) processo de capacitação da comunidade para atuar na melhoria de sua qualidade de vida e saúde, incluindo uma maior participação no controle deste processo. Para atingir um estado de completo bem-estar físico, mental e social os indivíduos e grupos devem saber identificar aspirações, satisfazer necessidades e modificar favoravelmente o meio ambiente. A saúde deve ser vista como um recurso para a vida, e não como objetivo de viver. Nesse sentido, a saúde é um conceito positivo, que enfatiza os recursos sociais e pessoais, bem como as capacidades físicas. Assim, a promoção da saúde não é responsabilidade exclusiva do setor saúde, e vai para além de um estilo de vida saudável, na direção de um bem estar global (EUROPE, 1986)

Com a adoção do sistema, espera-se conscientizar o cidadão da sua responsabilidade na manutenção urbana. De forma a entender que ao zelar pelo seu entorno, está criando uma cultura de cuidados, que resulta em um sentimento de pertencimento e, por sua vez, em qualidade de vida. Na seção 3.1 pode ser encontrado mais detalhes sobre o sistema *Ágora*.

1.2 Objetivos

Nessa seção serão apresentados os objetivos com o desenvolvimento e implantação do projeto Ágora

1.2.1 Objetivo geral

Desenvolvimento de um sistema, composto por uma *Application Programming Interface* (API) e um aplicativo móvel, para registro e consultas de solicitações de manutenção urbanas no município de Guarapuava-PR.

1.2.2 Objetivos específicos

- Desenvolver uma API, com a qual se possa realizar:
  - Cadastro e autenticação de usuários;
  - Cadastro, Edição e Exclusão de solicitações de manutenções urbanas.
- Desenvolver uma aplicação móvel, com a qual o usuário cidadão possa realizar:
  - O cadastro de novas solicitações para manutenção urbana utilizando a sua atual localização, pelo GPS do seu dispositivo móvel;
  - Registrar que suas solicitações de manutenção urbana foram solucionadas;
  - O apoio as solicitações de outros usuários, como uma forma de aumentar o peso de determinada solicitação, por exemplo, ao ver que existe uma anterior relativa ao mesmo problema, o usuário pode apoia-la no lugar de criar uma nova;
  - Consultas/visualização de solicitações de manutenção urbana, com ranqueamento, e a partir dessas informações, gerar relatórios que podem ser mensais ou semestrais por exemplo para exibir os problemas com solicitações mais recorrentes ou que demoraram mais tempo para serem solucionados;
- Integrar a API com o aplicativo móvel, que será a plataforma utilizada pelo usuário;

1.3 Justificativa

Todas as cidades enfrentam problemas no processo de urbanização, cidadãos sofrem diariamente com estes, que muitas vezes, acabam reduzindo a sua qualidade de vida, ou até mesmo, em casos extremos, os colocando em risco de vida. Ademais, ser cidadão, quer dizer participar ativamente da sociedade. A possibilidade de registrar situações indesejadas encontradas no seu entorno ou em qualquer outro ponto da cidade, e outros cidadãos poderem ter acesso a esse tipo de informação, de forma transparente, contribui para a sociedade, pois o cidadão poderá cumprir seu papel de agente fiscalizador.

A partir disso, pode-se concluir que ao disponibilizar o sistema *Ágora* na forma de aplicativo, ficará mais acessível a população. Pois, nos dias de hoje, o acesso a Internet e a dispositivos móveis como celulares está bem difundido. Segundo uma pesquisa realizada pelo Centro Regional de Estudos para o Desenvolvimento da Sociedade da Informação (Cetic.br) do Núcleo de Informação e Coordenação do Ponto BR (NIC.br)[6], 92 milhões de brasileiros tem

[6] https://nic.br/noticia/releases/92-milhoes-de-brasileiros-acessam-a-internet-apenas-pelo-telefone-celular-aponta-tic-domicilios-2022/

acesso a internet somente pelo celular, representando 62% da população. Outra vantagem é a possibilidade de poder registrar uma solicitação no momento exato da ocorrência.

Ainda, o desenvolvimento do sistema *Ágora* é uma oportunidade de colocar em prática conhecimentos e tecnologias estudadas ao longo do curso na implementação de uma solução para um problema real.

1.4 Estrutura do trabalho

O restante deste trabalho está organizado da seguinte maneira: no Capítulo 2 apresentamos um referencial teórico para sustentar o desenvolvimento dos capítulos seguintes; no Capítulo 3 delineamos o ambiente utilizado para as implementações; no Capítulo 4 apresentamos as implementações realizadas, seus resultados e considerações particulares; no Capítulo 5 apresentamos resultados práticos obtidos ao submeter os códigos ao *Sphere Online Judge* (SPOJ), visando comprovar o desempenho obtido quando comparado com soluções apresentadas por usuários de todo o mundo; por fim, no Capítulo 6 apresentamos a conclusão do trabalho e apontamos uma possível continuidade na pesquisa buscando soluções cada vez mais eficientes.

2 TRABALHOS RELACIONADOS

Atualmente, existem sistemas com funções análogas ao objetivo do sistema *Ágora*, várias prefeituras disponibilizam aplicações onde o usuário pode registrar suas solicitações de manutenção urbana, porém, a maioria não são sistemas focados exclusivamente para atender essas solicitações, mas sim sistemas mais generalizados, onde agrupam diversas funcionalidades. Em decorrência desse fato, eles não possuem muitas funções para esse objetivo, e além disso, todos não mostram informações de forma transparente ao usuário, como foi previamente mencionado na seção 1.3.

Dentre esses sistemas similares, está o Portal Carioca Digital, um sistema implementado pela Prefeitura da cidade do Rio de Janeiro, que por meio da ouvidoria oferece aos cidadãos a oportunidade de registrar suas solicitações de manutenção urbanas e acompanhar seu progresso por meio dos números de protocolo. Além disso, o sistema fornece gráficos informativos que detalham a quantidade de reclamações por setor, fornecendo uma visão abrangente da situação. A Figura 1 apresenta a tela deste sistema com a sumarização das solicitações.

Figura 1 – Captura de tela da interface do site Carioca Digital mostrando o painel, onde contém um relatório geral das solicitações registradas e atendidas por categoria
Fonte: Site Carioca Digital.

Outra ferramenta semelhante está em operação no município de São Paulo, conhecida como SP156, que permite aos moradores registrar solicitações de manutenção urbana e obter respostas de operadores, caso mais informações sejam necessárias ou para saber sobre os resultados obtidos com as reclamações. Em Guarapuava, essa funcionalidade está presente também através da ouvidoria municipal, no sistema Atende Net. No entanto, vale ressaltar que este sistema é mais simples em comparação com os sistemas mencionados anteriormente, limitando-se ao acompanhamento das reclamações apenas por meio do número de protocolo.

Figura 2 – Captura de tela da interface do aplicativo Atende Net mostrando a sua tela principal
Fonte: Aplicativo Atende Net.

Todos os três sistemas mencionados, possuem uma versão para celular. Porém nenhum dos aplicativos, de acordo com a loja de aplicativos para dispositivos Android, Google Play, não tem uma boa avaliação de seus usuários. Dentre todos, o mais bem avaliado é o Atende Net (Figura 2). Ainda assim, o processo de registrar solicitações em Guarapuava, é desconfortável devido a várias limitações do aplicativo e do sistema WEB. No sistema WEB, a interface não é muito amigável, as cores usadas são desagradáveis, e tanto na versão mobile como na WEB, os usuários têm apenas algumas categorias e subcategorias limitadas para escolher ao fazer suas reclamações. Além disso, o sistema não oferece recursos para a visualização de gráficos ou a geração de relatórios sobre as solicitações.

3 SISTEMA *ÁGORA*

Neste capítulo será apresentado as funcionalidades do sistema (seção 3.1) e como o uso destas auxilia o usuário no domínio (seção 3.2).

3.1 Escopo

O sistema *Ágora*, é direcionado para incentivar a fiscalização cidadã e visa fornecer ao cidadão, um sistema que seja focado em solicitações de melhorias urbanas, que além de ser prático, seja transparente, de forma que todas as solicitações podem ser acessadas, sem restrição para outros usuários, como normalmente ocorre com os sistemas discutidos no Capítulo 2. Desta forma, não será necessário ter uma conta ou estar autenticado para poder ver os registros do sistema, e sim somente caso o usuário deseje cadastrar novas solicitações ou editar as que foram previamente registradas por ele mesmo.

O objetivo do sistema é promover uma forma prática e eficiente de registrar e acompanhar relatos de problemas urbanos de diversos tipos, através do celular do usuário. Bem como, evidenciar o impacto de uma situação na cidade, por meio de uma funcionalidade, na qual o usuário vai poder “marcar” a situação, quando esta o impactar ou incomodar.

O sistema é um produto voltado para o público em geral, para poder ajudar no cuidado de sua cidade. Contudo, pode ser usado por entidades públicas ou privadas, como forma de mapeamento de problemas urbanos dentro de uma determinada localidade. Pois desta forma, poderá ser possível entender quais os problemas mais comuns enfrentados pelos cidadãos e quais bairros ou locais mais afetados por essas situações.

Visando o tratamento das necessidades identificadas, o sistema apresenta as seguintes funcionalidades no que diz respeito às solicitações de manutenção urbana:(a) Registro; (b) Acompanhamento; (c) Reforço e priorização; e (d) Mapeamento.

3.2 Perspectivas

Os cidadãos, de todas as cidades, enfrentam problemas urbanos diariamente, e a ausência de um canal centralizado e especializado para informar essas situações acaba deixando o cidadão sem saber a quem recorrer. O sistema *Ágora*, proposto neste documento, proverá as funcionalidades necessárias para que o cidadão possa ser um agente fiscalizador, relatando situações inconvenientes encontradas na sua cidade ou em outras cidades por ocasião de uma viagem.

Com o uso do sistema *Ágora*, no seu celular, o usuário pode registrar solicitações de manutenção para problemas urbanos. Durante o cadastro da sua solicitação, o usuário pode escolher a categoria mais adequada ao tipo da ocorrência e adicionar fotos para complementar

o cadastro, após isso, com o cadastro da solicitação, ela é adicionada ao mapa usando a localização do dispositivo via GPS, de forma que outros usuários possam ver onde, na cidade, fica localizada a situação relatada na solicitação do usuário. Quando uma solicitação é atendida, o usuário poderá finalizá-la.

Em alguns casos, pode acontecer que uma solicitação já esteja registrada no sistema, nesses casos, o usuário não precisa criar uma nova solicitação, ele pode simplesmente deixar seu apoio para a solicitação existente. Desta forma, a solicitação terá um peso maior através de um ranqueamento, na qual solicitações com maior apoio tendem a ser problemas que causam transtornos para uma maior quantidade de pessoas, e por isso devem ser consideradas prioritárias.

Para o acompanhamento das solicitações, usuários com boa reputação no uso do sistema, adquirida através do ganho de pontos, por participar de forma ativa, podem indicar seu fechamento/conclusão, para que os registros fiquem atualizados. Também, podem registrar que uma solicitação está sendo atendida, quando na sua rotina identificam que um problema está sendo resolvido, seja pela prefeitura ou não. Além disso, as solicitações possuem categorias diferentes, desta forma diversos tipos de problemas urbanos podem ser relatados, pois qualquer tipo de problema merece atenção.

4 MATERIAIS E MÉTODOS

Neste capítulo, será apresentada as tecnologias, ferramentas e a metodologia utilizada para desenvolver o sistema *Ágora*. Na seção 4.1 as ferramentas, bibliotecas, *frameworks*, plataformas de desenvolvimento e editores de código utilizados. Por fim, na seção 4.2, as atividades e tarefas de engenharia de software envolvidas no desenvolvimento do sistema.

4.1 Materiais

Sobre as tecnologias utilizadas, é possível separá-las em três categorias, que são as tecnologias do *backend*, *frontend* e outras ferramentas de uso geral.

4.1.1 Tecnologias Utilizadas para o Desenvolvimento do *Backend*

O desenvolvimento do *backend* do sistema *Ágora* será realizado com a implementação de uma API, utilizando o *framework* PHP Laravel[1]. Um *framework* escalável e progressivo, que possui um ambiente para o desenvolvimento de monólitos a API's, com documentação e comunidade ativa (LARAVEL, 2023a). Para o armazenamento de dados e informações no sistema será utilizado o sistema de gerenciamento de banco de dados PostgreSQL[2] (POSTGRESQL, 2023).

O ambiente de desenvolvimento que a API utilizará, será desenvolvido utilizado o *Docker*, uma ferramenta de permite implantar uma aplicação utilizando-se de contêineres, trazendo vantagens para o desenvolvimento e *deploy* (DOCKERINC, 2023a). Para poder aproveitar mais essas possibilidades do *Docker*, será usado em conjunto, o *Docker-compose*, que possibilita a utilização de múltiplos contêineres de forma simultânea, essa configuração é feita em um arquivo com a extensão YAML, que contém todas as imagens, contêineres e suas respectivas configurações (DOCKERINC, 2023b).

Para realizar os testes *backend*, serão utilizadas as próprias ferramentas do Laravel, que conta por si só, com um grande suporte para testes, sejam unitários, que devem testar partes pequenas do código, ou de funcionalidades que por sua vez, testam o funcionamento de pequenas partes como um todo, geralmente os mais implementados em uma aplicação, promovendo grande segurança ao código (LARAVEL, 2023b).

[1] https://laravel.com/

[2] https://www.postgresql.org/

4.1.2 Tecnologias Utilizadas para o Desenvolvimento do Aplicativo Mobile

Além da API, será desenvolvido um aplicativo *mobile* com o framework React Native[3], uma ferramenta Javascript baseada na própria biblioteca React[4], desenvolvida para criar aplicativos nativos Android e IOS utilizando código fonte na linguagem Javascript (META, 2023a). Juntamente com o React Native será utilizada a plataforma e conjunto de ferramentas, Expo[5], que facilita o acesso a recursos nativos dos aparelhos.

Da mesma forma que serão testados os recursos desenvolvidos no *backend*, como foi mencionado no subseção 4.1.1, serão realizados para o *frontend* também. Para os testes de componentes, a biblioteca utilizada será a React Native Testing Library, uma ferramenta mantida pela comunidade e recomendada pela própria documentação do React Native, que além de ser leve, dispõe de múltiplas funções úteis ao realizar testes (META, 2023b). Já para os testes *end-to-end*(ponta a ponta, testes que validam o funcionamento do sistema de forma completa, através da sua própria interface), será a ferramenta *Maestro*. De acordo com a sua documentação, é um *framework* para testes de interface com aplicabilidade em aplicações React Native, podendo executar tanto em um simulador Android como também para IOS, através do *Maestro* é possível poder testar fluxos inteiros do sistema, um fluxo consiste em uma sequência de passos, por exemplo, o login de um usuário no sistema, para fazer o login, ele deve seguir uma sequência de passos em telas diferentes. Os testes devem ser configurados em um arquivo com extensão YAML, e são escritos através de uma linguagem com sintaxe própria, porém bem intuitiva (MOBILE.DEV, 2023).

4.1.3 Ferramentas para Versionamento de Código e Prototipação

Ademais as tecnologias de desenvolvimento, serão utilizadas como ferramentas de versionamento de código fonte o *Git* e o *Github*. A primeira, é um sistema de versionamento *open-source* e gratuita, que como uma das principais vantagens possui a possibilidade de trabalhar com ramificações dentro do mesmo projeto. E o uso da plataforma de hospedagem de código em nuvem *Github*, onde é possível acompanhar o projeto, gerenciar as suas *branches* (ramificações) e ainda utilizar ferramentas, como por exemplo, testes e *deploys* automáticos (LONGEN, 2023). Juntamente com a plataforma *Github*, será utilizado a funcionalidade *Projects*, que de acordo com documentação do *Github*, permite criar um quadro *kanban*, para organizar as tarefas e o fluxo de trabalho dentro do próprio repositório do projeto (GITHUB, 2023).

Para poder realizar a prototipação das interfaces do sistema, será utilizado a plataforma Figma[6], que disponibiliza recursos simples e avançados para criação de protótipos. Segundo

[3] https://reactnative.dev/
[4] https://react.dev/
[5] https://expo.dev/
[6] https://www.figma.com

Santana (2023), o Figma apresenta ótimas vantagens ao ser utilizado para a criação de interfaces e protótipos, pois é uma ferramenta online multiplataforma, utilizada pelo navegador de qualquer computador com acesso a internet, permite também o colaboração entre uma equipe pessoas, além disso, o Figma, dispõe de ferramentas que auxiliam para criação de layouts responsivos (SANTANA, 2023).

4.1.4 Ferramentas para Modelagem de Banco de Dados

Para a modelagem do banco de dados, será utilizado o Miro[7], com esta ferramenta será desenvolvido o MER do sistema *Ágora*, da mesma forma que os requisitos, como será destacado na seção 4.2, o MER pode ser atualizado durante a etapa desenvolvimento, para ir se adaptando a novas demandas ou melhorias que podem ser feitas durante este processo.

O uso da ferramenta Miro, segundo a sua documentação, tem como vantagens, que além de poder ser utilizado de forma gratuita, tem funcionalidades que possibilitam a elaboração de um MER de forma fácil, ademais essas funcionalidades, e tem a disposição diversos modelos pré definidos para diferentes tipos de documentos (MIRO, 2023).

4.1.5 Ferramentas para Documentação do Software

O processo de documentar, é uma ótima prática durante o processo de desenvolvimento de *software*, porém nem sempre é uma prática adotada. Segundo um artigo disponibilizado pela Atlassian (2023a), elaborar a documentação pode trazer vantagens como, por exemplo, economia de tempo na procurar de uma determinada funcionalidade ou, ainda, entender algum fluxo. Outra vantagem é poder ter maior controle na qualidade do seu software, evitar duplicação de tarefas, facilitar a integração de novos desenvolvedores e aumentar o nível de conhecimento coletivo da equipe, pois através da documentação, novas informações podem ser partilhadas de forma que todos podem ter acesso a elas (ATLASSIAN, 2023a).

Para documentar o desenvolvimento do sistema *Ágora*, será utilizada uma ferramenta chamada *Scribe*[8]. Esta ferramenta, segundo a sua documentação, gera uma página HTML, com interface agradável com a funcionalidade de poder testar os *endpoints* da API pelo navegador (*Try it out*), podendo inclusive testar filtros. Além dessas funcionalidades, é possível adicionar informações gerais do software que está sendo documentado (SCRIBE, 2023).

[7] https://miro.com

[8] https://scribe.knuckles.wtf/laravel/

4.2 Métodos

Nesta seção será apresentada a metodologia e as práticas que irão ser adotadas para o desenvolvimento do sistema *Ágora*, e é resultado do esforço colaborativo do professor orientador deste trabalho, em conjunto com seus orientandos de TCC, logo este trabalho e outros compartilharão da mesma metodologia, inclusive os textos e/ou as imagens são muito similares, com pouquíssimas diferenças.

Uma meta importante para o desenvolvimento do sistema *Ágora*, é desenvolver um software com um código limpo e rigor de testes. A metodologia de desenvolvimento, como ilustrada na Figura 3, será composta das seguintes atividades: levantamento de requisitos, prototipação, testes e codificação.

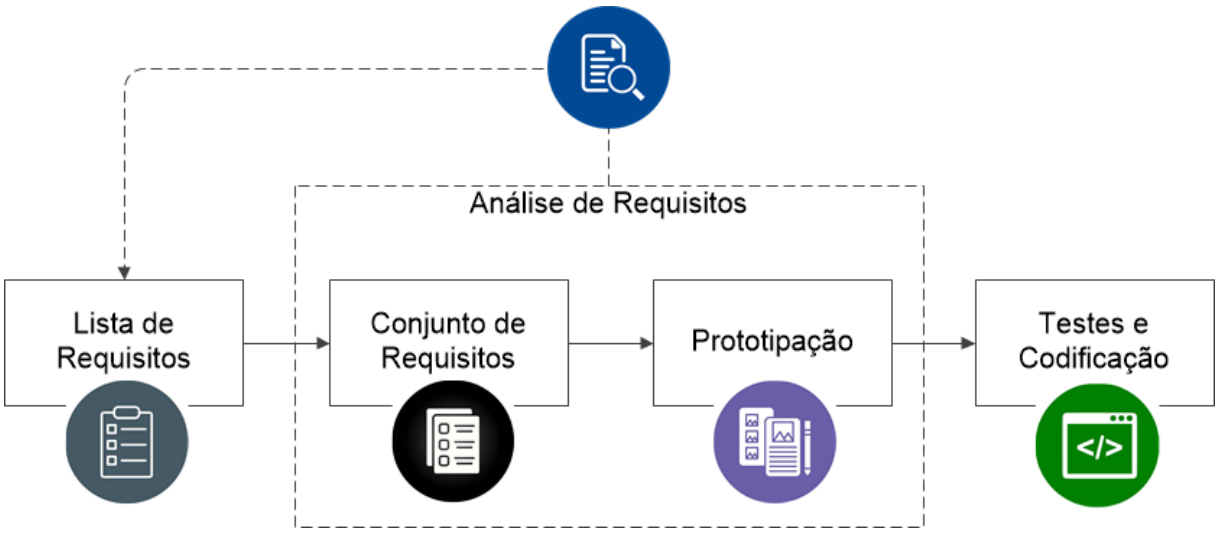

Figura 3 – Metodologia de desenvolvimento
Fonte: Autoria própria.

Uma parte essencial no desenvolvimento de software, é a engenharia de requisitos. “Os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços que oferece e as restrições a seu funcionamento” (SOMMERVILLE, 2011a). Assim sendo, primeiramente será feito o levantamento dos requisitos mais tangíveis (**Lista de Requisitos**), que serão necessários para o sistema, sendo divididos em requisitos funcionais e não funcionais.

Os requisitos do sistema são separados em dois tipos, os requisitos funcionais e os não funcionais. Segundo Sommerville (2011b), podemos descrever que os requisitos funcionais, são os requisitos necessários para a funcionalidade do sistema proposto, a forma que ele deve reagir, como deve receber dados ou até mesmo o que o sistema não pode fazer, já os requisitos não funcionais são aqueles que não interferem de forma direta nas funcionalidades do sistema, são funções ou restrições que estão no sistema, podendo melhorar a usabilidade e muitas vezes são aplicados no sistema todo. De uma forma geral, o levantamento dos requisitos, em um primeiro momento, como não foi realizada nenhuma forma de entrevista com possíveis futuros usuários, todos os requisitos foram delineados pelo o que autor do trabalho juntamente com o

professor orientador esperam do sistema. Contudo, à medida que o sistema torne-se acessível para os usuários, será verificado se atende ao proposto e ao que o usuário espera da aplicação. Para os requisitos funcionais foi utilizado as funcionalidades descritas na subseção 1.2.2, e cada objetivo específico, pode gerar ou não mais de um requisito na lista, isso depende do grau de complexidade da funcionalidade em si. Já para os requisitos não funcionais, foram utilizadas informações obtidas através dos sistemas parecidos discutidos no Capítulo 2, que serão implementadas como melhorias e também funcionalidades que trariam uma melhor experiência de uso para o usuário. Com essas informações definidas, será possível quantificar o número de tarefas e o tempo de desenvolvimento. Também, será igualmente importante como auxílio para o processo de modelagem do banco de dados.

Para organizar o desenvolvimento, serão utilizados ciclos de desenvolvimento, cada ciclo terá 14 dias de duração, e nesse período, serão desenvolvidas as funcionalidades, que a princípio, serão os requisitos levantados na Tabela 1, então após esse período destinado ao desenvolvimento, será discutido, em forma de reunião, para organizar para o próximo ciclo que se iniciará.

A implementação do sistema começará durante as atividades de verificação e elaboração de requisitos (**Conjunto de Requisitos**), onde por meio de reuniões, do entendimento do domínio, abrangência de aplicação e uso, um conjunto de requisitos da Lista de Requisitos será trabalhado, de forma a entender as funcionalidades e o que será necessário de ser desenvolvido. Após estas atividades, será iniciada a etapa de prototipação das telas para o sistema (**Prototipação**). Criar protótipos permite testar/validar ideias antes que sejam realmente implementadas num sistema computacional, essa técnica permite também descobrir falhas ou requisitos que anteriormente não haviam sido considerados relevantes (JUHL; SøRENSEN, 2023).

Por fim, considerando o apresentado na Figura 3, a implementação do sistema terá continuidade por meio da codificação de testes e funcionalidades (**Testes e Codificação**). Para esta etapa de desenvolvimento do sistema, será adotado a metodologia *Test Driven Development* (TDD), que diz que o código deve ser feito baseado em testes unitários, de forma que o código fonte, seja feito baseado em um teste que falhou previamente, e então o código deve ser aprimorado até que o teste seja executado com sucesso, desta forma, o teste irá determinar o comportamento esperado. Usar desta metodologia, traz grandes vantagens ao projeto, pois cria-se um código robusto e flexível, menos sujeito a falhas e a problemas com refatorações ou alterações. Outro conceito trabalhado no TDD, é que o código deve ser fácil de testar (ACOSTA; GAJDA, 2023). Dessa maneira, a codificação *backend* será iniciada a partir do desenvolvimento de testes unitários, utilizando as técnicas de TDD, após a elaboração desses testes, será construído um código que faça os testes passarem com sucesso. Já as atividades referentes ao *frontend*, serão iniciadas a partir do desenvolvimento da tarefa em si, e então somente depois será validado com testes unitários.

Partindo desse ponto, outra metodologia trabalhada será o SOLID. O próprio nome dessa metodologia, representa cada uma das iniciais dos seus 5 princípios. Segundo Oloruntoba (2021), seus princípios são: Princípio da Responsabilidade Única (***S**ingle Responsiblity Principle*), que afirma que uma classe deve ter somente funções diretamente inerentes a entidade representada por ela; Princípio Aberto-Fechado (***O**pen-Closed Principle*), uma classe deve ser aberta a expansões, porém fechada a modificações; Princípio da Substituição de Liskov (***L**iskov Substitution Principle*), que diz que uma classe derivada, pode ser substituída por sua classe base; Princípio de Segregação da Interface (***I**nterface Segregation Principle*), uma classe não deve implementar um função que não será usada, por isso é melhor criar interfaces mais especializadas; e o Princípio da Inversão de Dependência (***D**ependency Inversion Principle*), "entidades devem depender de abstração, não de implementações, modelos de alto nível não devem depender de modelos de baixo nível, mas sim depender de abstrações".

A gestão dos requisitos acontecerá principalmente durante as atividades de verificação, elaboração e prototipação (**Análise de Requisito**), mas também durante a etapa de implementação do sistema (**Testes e Codificação**), fomentando o amadurecimento e evolução dos requisitos para o sistema. Consequentemente, alguns requisitos podem ser atualizados ou substituídos ao longo do andamento do projeto de acordo com a necessidade. Entende-se que esta abordagem é válida, proporcionando a definição clara e correta dos requisitos e consequentemente a evolução do sistema.

O desenvolvimento do sistema, suas funcionalidades e testes, compreendidos na etapa de **Testes e Codificação**, será conduzido utilizando o Git e o GitHub, como mencionado anteriormente na subseção 4.1.3, ferramenta e plataforma de versionamento de código. Será utilizado o fluxo de trabalho conhecido como *Gitflow*, descrito mais a frente nesta seção. Para este momento, a criação de códigos será organizada conforme apresentado na Figura 4, um micro fluxo para o gerenciamento e controle desta etapa, onde estão presentes os conceitos de funcionalidade (*feature*) e desenvolvimento (*develop*). São estas:

- *Feature*: De acordo com o Atlassian, as ramificações de *feature*, tem como pai a ramificação *develop*, cada uma delas, vai conter o desenvolvimento de uma ou um conjunto de funcionalidades inerentes, que quando é concluído é realizado *merge*[9] para a *develop* (ATLASSIAN, 2023b). Desta forma, o código da *develop* fica atualizado com as novas funcionalidades implementadas na ramificação de desenvolvimento;

- *Develop*: É a ramificação de desenvolvimento dentro do fluxo do *Gitflow*. Segundo acordo com o Atlassian, está ramificação é alimentada pelas *features*, quando já possui recursos suficientes para uma nova versão do software, é realizado uma bifurcação a partir desta ramificação para uma *branch release*, a qual conterá o conteúdo na nova versão disponibilizada (ATLASSIAN, 2023b).

[9] Processo de mesclar duas diferentes ramificações

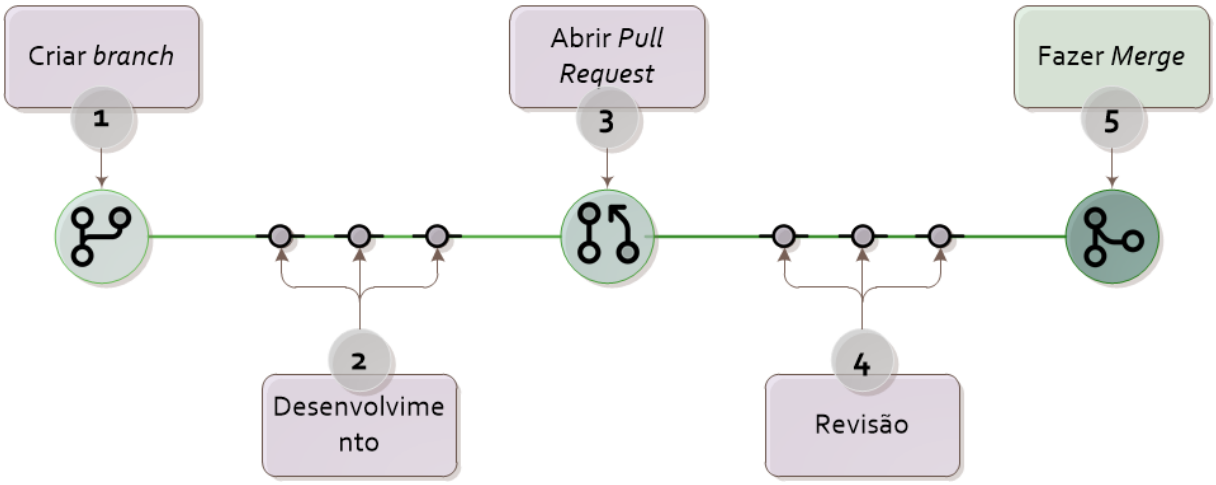

Figura 4 – Fluxo de desenvolvimento
Fonte: Autoria própria.

Após a elaboração do desenvolvimento da atividade e da validação com testes unitários, a próxima etapa é referente aos testes de fluxo, esse termo contemplará, os testes de funcionalidade para a API e também do aplicativo. O objetivo deste tipo de teste é poder validar funcionalidades de forma completa.

Com os testes implementados, e o código correspondendo de acordo com o desejado, será feito uma validação geral da atividade, para verificar se: (a) O requisito foi realmente cumprido; (b) Essa atividade pode gerar impacto em alguma outra área ou regra do sistema; (c) Verificação de estilização do código que desenvolvido, para executar esta função será utilizada uma nova ferramenta chamada de *Pint*, a qual a primeira versão estável foi lançada em 2022, segundo a documentação do Laravel, o *Pint* é um formatador simplificado e minimalista opinado para código PHP, que não necessita de dependências externas, desenvolvido a partir da popular ferramenta *The PHP Coding Standards Fixer*[10], que é amplamente utilizada (LARAVEL, 2023c), além disso, será integrado com a plataforma GitHub, para que todo o código novo seja verificado de acordo com as regras do formatador; (d) Revisão de código, nesta etapa final, como em todo ciclo de desenvolvimento que acontecer, será gerada uma *pull request*, ou seja, uma solicitação para realizar o *merge* de código novo no projeto, então este código será revisado pelo professor orientador para verificar a padronização de codificação e qualquer outra questão relacionada à arquitetura e desempenho da aplicação, bem como será verificada pela execução dos testes e, posteriormente, teste de uso do sistema.

Após a conclusão dessas etapas, a etapa final tarefa pode então ser considerada completa e apta para ser realizado o *merge*. Porém, é importante ressaltar que esse fluxo não é uma estrutura rígida e absoluta, pois se em algum ponto durante a etapa de desenvolvimento da tarefa, for notado algum erro, inconsistência ou má descrição, pode-se então retornar para alguma etapa anterior, a fim de que a atividade possa ser desenvolvida da melhor forma possível.

[10] https://cs.symfony.com/

O micro fluxo da Figura 4 está inserido dentro de um fluxo maior, o *Gitflow*, representado na Figura 5.

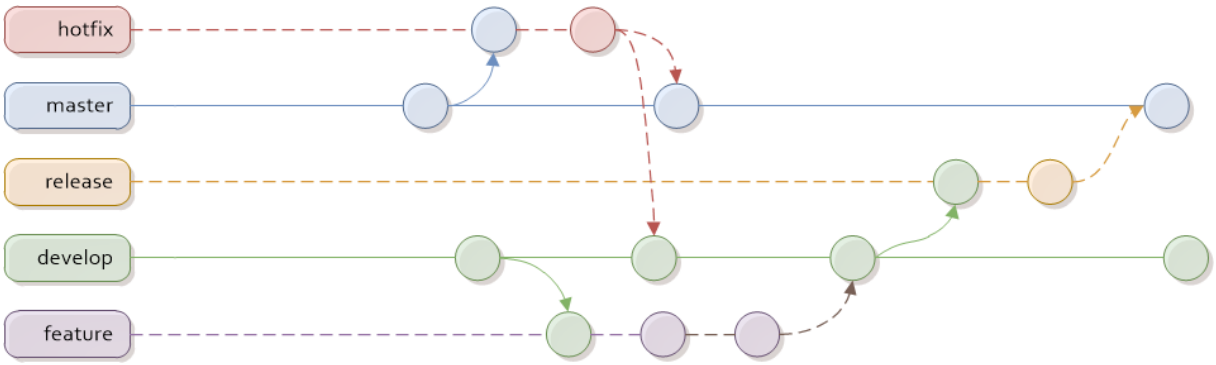

Figura 5 – Fluxo de Ramificações
Fonte: Autoria própria.

A Figura 5, representa as ramificações existentes dentro do fluxo do *Gitflow*. Como foi previamente mencionado, é criado uma nova ramificação a partir da *develop*. De acordo com o Atlassian (2023b), esta nova *branch* é uma de *release*, então quando está nova versão estiver pronta para ser lançada, será feito o *merge* desta ramificação com a *main*, a principal *branch* repositório, representando a versão mais estável do software. Caso, sejam encontrados erros de código, para realizar as correções necessárias, devem ser utilizadas ramificações com esse propósito, se o erro for encontrado na *develop*, deve ser criado uma ramificação a partir desta *branch*, do tipo *bugfix* e após a correção ser feita, é feito o *merge* na *develop*. Porém, se o erro for encontrado na versão estável do software, ou seja, em um código que esteja na *main*, é criada então uma ramificação do tipo *hotfix*, que são *branches* elaboradas para correção de erros que estão acontecendo na versão disponibilizada do software, que pode estar muitas vezes em ambiente de produção, e em um cenário onde coexistissem uma *hotfix* com uma *release*, a correção deve ser aplicada na *release* também (ATLASSIAN, 2023b).

5 RESULTADOS PARCIAIS

Neste capítulo serão apresentados os resultados parciais obtidos com a aplicação da metodologia mencionada na seção 4.2. Como resultados foram produzidos os seguintes itens: lista de requisitos (seção 5.1) e o MER (seção 5.2).

5.1 Lista de Requisitos

Após a análise das funcionalidades esperadas do sistema, foram criados, neste primeiro momento, 18 requisitos funcionais. Na Tabela 1 estão todos estes requisitos, cada item na tabela contém colunas para exibir seu número, título, descrição e se são requisitos funcionais ou não.

Tabela 1 – Lista de Requisitos do Sistema

Número	Título	Descrição	Funcional
RF 001	Cadastro de novos usuários.	Como futuro usuário do sistema, quero poder realizar meu cadastro, para ter acesso ao aplicativo.	Sim
RF 002	Edição de Usuários	Como usuário, quero poder atualizar as minhas informações pessoais e de acesso, para manter a integridade das minhas informações.	Sim
RF 003	Login de usuário.	Como usuário, quero poder utilizar minhas credenciais, para realizar o login.	Sim
RF 004	Logout de usuário.	Como usuário, quero poder encerrar a minha sessão, para evitar acessos indevidos em meu nome.	Sim
RF 005	Cadastro de solicitações de manutenção urbana.	Como usuário, quero poder registrar solicitações de manutenção urbana, para tornar público um problema ou demanda na cidade.	Sim
RF 006	Utilizar a localização atual no momento de cadastrar uma solicitação de manutenção urbana.	Como usuário, quero que minha localização seja considerada no momento do registro de uma solicitação de manutenção urbana, para que o problema ou demanda sejam encontrados com maior facilidade.	Sim
RF 007	Adição de imagens ao realizar o cadastro de uma solicitação de manutenção urbana.	Como usuário, quero adicionar fotos a uma solicitação de manutenção urbana, para mostrar ou validar o estado atual do problema ou demanda.	Sim
RF 008	Categorizar as solicitações de manutenção urbanas.	Como usuário, quero categorizar a solicitação de manutenção urbana, para organização do problema ou demanda.	Sim
RF 009	Edição de solicitações de manutenção urbanas.	Como usuário, eu quero poder editar as informações das solicitações de manutenção urbana criadas por mim.	Sim
RF 010	Exclusão de solicitações de manutenção urbanas.	Como usuário, eu quero poder excluir solicitações de manutenção urbana criadas por mim.	Sim

Continua na próxima página

Tabela 1 – Continuação da Página Anterior

Número	Título	Descrição	Funcional
RF 011	Exibição de solicitações de manutenção urbanas no mapa.	Como usuário, eu quero poder visualizar solicitações de manutenção urbana criadas por mim e por outros usuários.	Sim
RF 012	Exibição de uma solicitação de manutenção urbana em particular.	Como usuário, eu quero poder visualizar uma solicitação de manutenção urbana, podendo ver seus detalhes, como por exemplo, imagens e histórico.	Sim
RF 013	Poder solicitar o fechamento de uma solicitação de manutenção urbana.	Como usuário, eu quero poder realizar o fechamento de uma solicitação de manutenção urbana criada por mim, ou caso seja criada por outro usuário, quero poder solicitar o seu fechamento caso esta seja atendida.	Sim
RF 014	Visualização de relatórios gerais de registros das solicitações de manutenção urbana.	Como usuário, eu quero poder ver dashboards com informações gerais sobre o tipo de registros mais comuns ou solucionados, etc.	Sim
RF 015	Funcionalidade de reforço a solicitações de manutenção urbana já registradas.	Como usuário, quero poder reforçar determinada solicitação, para que esse problema ganhe um grau maior de visibilidade e poder demonstrar que sou afetado pela mesma ocorrência.	Sim
RF 016	Aplicação de filtros para exibição de registros no mapa.	Como usuário, quero poder aplicar filtros nas listagens, para poder ver somente solicitações de manutenção urbana específicas de acordo com os critérios dos filtros.	Sim
RF 017	Aplicação de filtros nas dashboards de relatório.	Como usuário, quero poder aplicar filtros nas listagens das dashboards de relatório de informações gerais.	Sim
RF 018	Casos possíveis para deletar uma solicitação de manutenção urbana	Como usuário, quero poder excluir minhas solicitações de manutenção urbanas, somente caso ela não ter sido alterada por outro usuário	Sim

5.2 Modelo Entidade Relacionamento (MER)

Neste primeiro momento, o MER do sistema *Ágora*, representado na Figura 6, contém as principais entidades, porém será atualizado e incrementado ao longo do desenvolvimento do projeto. Além disso, o MER é independente de tecnologia, e serve para modelagem das entidades e dados, no momento de implementar o banco em um *Sistema de Gerenciamento de Banco de Dados* (SGBD), será feito de acordo com as particularidades e regras do banco de dados escolhido e do framework *Laravel*, para que funcione de acordo o *Object-Relational Mapping* (ORM) do *framework*, o *Eloquent*[1].

[1] https://laravel.com/docs/10.x/eloquent

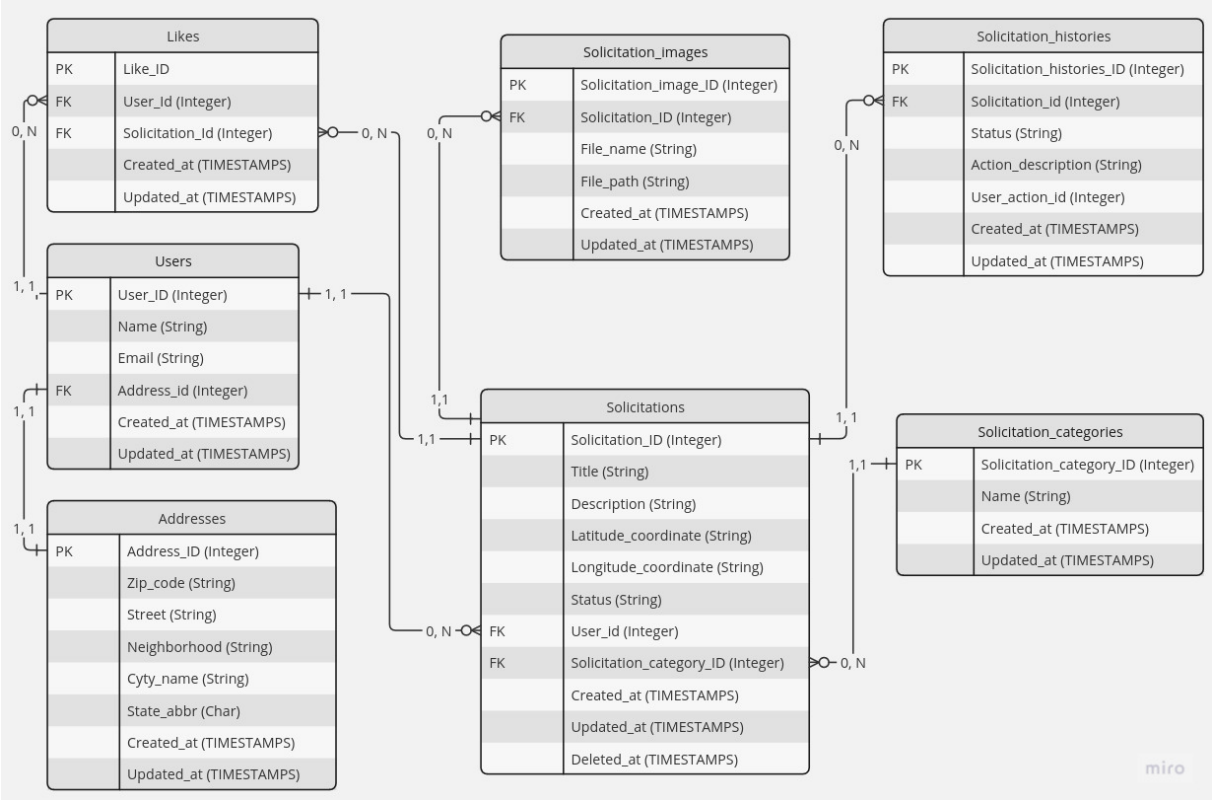

Figura 6 – Diagrama MER do sistema
Fonte: Autoria própria.

No MER desenvolvido, temos a tabela chamada *users*, que será responsável pelo armazenamento de informações referentes a autenticação dos usuários, logo, todo usuário novo da plataforma terá um registro nesta tabela. Campos com informações sensíveis tais como o campo de senha (*password*), como medida de segurança, serão registrados de forma criptografada. A tabela de usuários, interage com a tabela de *addresses*, que armazenará informações referentes ao atual endereço dos usuários, este tipo de informação será utilizado somente para fins de relatórios gerais sobre o uso do sistema, todo usuário deve ter somente um endereço cadastrado.

A principal entidade no MER desenvolvido, corresponde a tabela de *solicitations*, tabela na qual serão registrados as solicitações de manutenção urbana, como seu título, sua descrição e também estarão presentes os campos que controlarão a exibição dos solicitações no mapa e o fluxo dos processos, para esta última função será utilizado o campo *status*, que irá diferenciar em qual estado está cada solicitação, por exemplo, ativa ou resolvida, para gerenciar estes possíveis valores será utilizado um *Enum*, que basicamente é uma classe utilizada para controlar um conjunto de valores possíveis para um tipo, os valores deste *Enum*, para controlar o *status* serão definidos em uma etapa futura deste projeto. Além disso, está tabela fará relação com a tabela *solicitation_categories*, que será utilizada para separar os diferentes tipos de solicitações que podem ser registradas.

Cada ação que for realizada com as solicitações de manutenção urbanas, resultará em um novo registro na tabela *solicitation_histories*, de forma que será possível traçar todas as ações que foram tomadas a respeito de uma solicitação, por exemplo, ao usuário sugerir o fechamento de uma solicitação, por ter sido atendida, essa ação será registrada, com o *status* que estava nesse momento, com a descrição da ação realizada e o identificador do usuário que realizou tal ação, as solicitações podem ter vários registros deste tipo no seu histórico ao mesmo tempo. Como parte importante no processo de registro de novas solicitações de manutenção urbana, o MER, conta com uma entidade responsável por armazenar e relacionar imagens a uma solicitação, no diagrama, está entidade corresponde a tabela *solicitation_images*. Por último, a tabela *likes*, será responsável por registrar os reforços que as solicitações podem receber dos usuários do sistema, cada usuário pode registrar seu reforço somente uma vez para cada solicitação.

6 CONSIDERAÇÕES FINAIS

Esse trabalho, propõe e descreve o desenvolvimento de um sistema para consulta e registro de solicitações de manutenção para problemas urbanos, através de um aplicativo *mobile* chamado *Ágora*. Que será desenvolvido utilizando PHP com Laravel para a API do sistema, e React Native para a aplicação *mobile*, a escolha dessas tecnologias e ferramentas de desenvolvimento foram pautadas nas vantagens que as mesmas oferecem.

Além disso, foram abordadas as metodologias que serão utilizadas para o desenvolvimento do sistema *Ágora*. Desde a fase inicial de análise e levantamento de requisitos, prototipação, até a etapa de desenvolvimento. Uma grande meta do projeto, é que ao final do desenvolvimento, será possível ter um software que além de cumprir todos os seus objetivos funcionais e não funcionais, seja um software robusto, de código limpo e organizado, com testes que cubram todas as suas funcionalidades, possibilitando melhores condições para possíveis continuações e atualizações futuras para o sistema.

Conclui-se que a implementação do sistema *Ágora*, pode trazer vantagens, como mais transparência para o cidadão, podendo ter acesso a mais informações sobre os problemas urbanos que sua cidade enfrenta, e quais ações estão sendo tomadas para resolver essas ocorrências. Além disso, essa solução tem como principal objetivo incentivar o cidadão a exercer seu papel como agente fiscalizador.

REFERÊNCIAS

ACOSTA, J.; GAJDA, K. **Test-driven development**. 2023. Disponível em: https://www.ibm.com/garage/method/practices/code/practice_test_driven_development/.

ATLASSIAN. 2023. Disponível em: https://www.atlassian.com/work-management/knowledge-sharing/documentation/importance-of-documentation.

ATLASSIAN. 2023. Disponível em: https://www.atlassian.com/br/git/tutorials/comparing-workflows/gitflow-workflow.

CARTA Brasileira: Cidades Inteligentes. 2020. Disponível em: https://www.gov.br/mdr/pt-br/assuntos/desenvolvimento-urbano/carta-brasileira-para-cidades-inteligentes/CartaBrasileiraparaCidadesInteligentes2.pd.

DOCKERINC. 2023. Disponível em: https://www.docker.com/why-docker/.

DOCKERINC. 2023. Disponível em: https://docs.docker.com/compose/.

EUROPE, W. H. O. R. O. for. Ottawa charter for health promotion, 1986. World Health Organization. Regional Office for Europe, p. 5 p., 1986.

GITHUB. 2023. Disponível em: https://docs.github.com/en/issues/planning-and-tracking-with-projects/learning-about-projects/about-projects.

JACOBI, P. A percepção dos problemas ambientais urbanos em são paulo. **Lua Nova: Revista de Cultura e Política**, 2013.

JUHL, J.; SøRENSEN, A. S. **Prototyping: Getting the design right**. 2023. Disponível em: https://www.ibm.com/blogs/nordic-msp/prototyping/.

LARAVEL. 2023. Disponível em: https://laravel.com/docs/10.x#why-laravel.

LARAVEL. 2023. Disponível em: https://laravel.com/docs/10.x/testing#introduction.

LARAVEL. 2023. Disponível em: https://laravel.com/docs/10.x/pint.

LONGEN, A. S. **O Que é GitHub, Para Que Serve e Como Usar**. 2023. Disponível em: https://www.hostinger.com.br/tutoriais/o-que-github.

META. 2023. Disponível em: https://reactnative.dev/.

META. 2023. Disponível em: https://callstack.github.io/react-native-testing-library/docs/getting-started.

MIRO. 2023. Disponível em: https://miro.com/pt/diagrama/diagrama-entidade-relacionamento/.

MOBILE.DEV. 2023. Disponível em: https://maestro.mobile.dev/.

OLORUNTOBA, S. **SOLID: The First 5 Principles of Object Oriented Design**. 2021. Disponível em: https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design.

POSTGRESQL. 2023. Disponível em: https://www.postgresql.org/about/.

SANTANA, B. **Eleve Seus Designs: O Que é Figma e Como Usá-lo Corretamente!** 2023. Disponível em: https://www.hostinger.com.br/tutoriais/figma-o-que-e#:~:text=Aprendemos%20que%20o%20Figma%20%C3%A9,aplicativos%20e%20sites%2C%20por%20exemplo.

SCRIBE. 2023. Disponível em: https://scribe.knuckles.wtf/laravel/.

SOMMERVILLE, I. **Engenharia de Software**. [*S.l.*]: Pearson Education do Brasil Ltda., 2011. 57 p.

SOMMERVILLE, I. **Engenharia de Software**. [*S.l.*]: Pearson Education do Brasil Ltda., 2011. 59 p.