

MAYARA CAROLINE CAVALIN NICOCHELLI

**CIDADE SUSTENTÁVEL: Aplicativo para Incentivo a Reciclagem e
Sustentabilidade em Cidades Inteligentes**

Curitiba

Dezembro/2022

RESUMO

O crescimento dos centros urbanos e os avanços da tecnologia criaram um novo conceito para descrever as cidades da atualidade, que agora possui uma adaptação de dinâmicas tradicionais que tornaram-se mais eficientes através de soluções digitais, as Cidades Inteligentes. Em contraste com os avanços tecnológicos surgiu um panorama de crescimento exponencial dos resíduos produzidos e descartados de maneira inadequada. Acompanhando o cenário descrito, existe a desinformação da população sobre como lidar com o lixo gerado desenfreadamente, que acaba prejudicando o meio ambiente inconscientemente.

Para auxiliar na solução da problemática do lixo utilizando as novas tecnologias, o estudo propõe a construção de um software para celulares com o objetivo de promover a conscientização sobre a reciclagem, agregando em uma plataforma informações e locais para ajudar a população na reciclagem e descarte correto dos resíduos.

A metodologia ágil aplicada no desenvolvimento do aplicativo baseou-se na Extreme Programming e propõe requisitos de histórias para serem implementados diretamente como uma série de tarefas baseadas no levantamento de requisitos do software.

Em resumo, o projeto resultará em uma aplicação disponível para dispositivos móveis que além de contribuir para conscientização do descarte correto de lixo nas Cidades Inteligentes, promoverá através de análises de qualidade de código uma aplicação estável e segura para seus usuários.

Palavras-chave: CIDADES INTELIGENTES; RECICLAGEM; APLICATIVO.

1. INTRODUÇÃO

Nas últimas décadas, a taxa da população brasileira vivendo no meio urbano cresceu exponencialmente, estabelecendo a população urbana em 84,72% de acordo com a Pesquisa Nacional por Amostra de Domicílios (PNAD) realizada em 2015. (PNAD,2015). Com esse crescimento dos centros urbanos e os avanços tecnológicos surge o conceito da Cidade Inteligente, um lugar onde as redes e serviços tradicionais tornam-se mais eficientes através de soluções digitais em benefício de seus habitantes de acordo com a União Europeia. (UNIÃO EUROPÉIA, 2018)

Como consequência, nesses centros urbanos existe o aumento da produção de resíduos, como é o caso do lixo eletrônico. Segundo a pesquisa feita pela Green Eletron, o Brasil descartou mais de 2 milhões de toneladas de resíduos eletrônicos em 2019, porém menos de 3% foi destinado a reciclagem, e embora existam lugares próprios para coleta do lixo eletrônico, constatou-se que 33% da população nunca ouviu falar em pontos ou locais de descarte correto para lixo eletrônico. (GREEN ELETRON, 2021)

Torna-se evidente a falta de informação sobre a reciclagem, como demonstra a pesquisa realizada em 2019 pela consultoria global Ipsos “A Throwaway World: The Challenge of Plastic Packaging and Waste” (Um mundo descartável: o desafio das embalagens plásticas e resíduos, em tradução livre), onde aponta que cerca de 65% dos entrevistados acreditavam que todos os plásticos podem ser reciclados, no entanto alguns tipos, como embalagens metalizadas, adesivos e papel celofane, não podem ir para a coleta seletiva. (IPSOS, 2019)

Ao contrário do panorama de reciclagem no Brasil, o uso de celulares aumenta, segundo dados divulgados na 30ª Pesquisa Anual de Administração e Uso de Tecnologia da Informação nas Empresas, realizada pela Fundação Getúlio Vargas de São Paulo (FGV – SP) realizada em 2019, no país haviam cerca de 230 milhões de smartphones em uso no país. (MEIRELLES, 2019). A maior parte da

população utiliza o celular como principal meio de acesso à internet, segundo a pesquisa Impactos do Android no Brasil feita em parceria com a Google, cerca de 51% dos brasileiros acessam a internet exclusivamente pelo celular, e aproximadamente 24 milhões de brasileiros foram introduzidos à internet por meio de um dispositivo Android. (CAMARGO; MOURA, 2019)

Considerando o aumento preocupante da produção do lixo e a falta de informações sobre reciclagem por parte da população, este trabalho tem como objetivo desenvolver um aplicativo colaborativo para dispositivos móveis para facilitar o acesso à informações sobre pontos de coleta de lixo eletrônico, reciclagem e coleta seletiva. A tecnologia Android foi escolhida no desenvolvimento para alcançar grande parte da população e o público é voltado para as pessoas interessadas em reciclagem e outras iniciativas de sustentabilidade nas Cidades Inteligentes.

- **OBJETIVO GERAL**

O objetivo geral deste trabalho consiste em desenvolver um aplicativo mobile Android para ajudar a localizar pontos de coleta de lixo eletrônico e lixo reciclável nas Cidades Inteligentes.

- **OBJETIVOS ESPECÍFICOS**

1. Criar um aplicativo mobile que promova a consulta de informações e dicas sobre reciclagem e pontos de coleta de lixo eletrônico e lixo reciclável.
2. Permitir o acesso a informações e dicas sobre reciclagem de lixo eletrônico, lixos recicláveis e não recicláveis e descarte correto do lixo.

3. Possibilitar a visualização dos pontos de coleta de lixo eletrônico e lixo reciclável no mapa da Cidade Inteligente desejada.
4. Permitir que as prefeituras, população e ONGs possam contribuir e divulgar iniciativas sustentáveis através de um formulário no aplicativo.
5. Prototipar interfaces da aplicação.
6. Documentar a aplicação.

- **JUSTIFICATIVA**

Num contexto global, o Brasil é um dos países que mais gera resíduos sólidos, cujo destino final não é o ideal, carecendo de tratamento e sendo despejados a céu aberto. Após 20 anos de discussão, o governo federal estabeleceu em 2010 a lei 12.305, que define a PNRS (Política Nacional de Resíduos Sólidos), que prevê o gerenciamento de resíduos sólidos, responsabilizando os municípios pelos resíduos gerados. Embora tenha expirado em 2014 o prazo inicial, os dados demonstram que 53% das cidades do país, ainda não se adequaram ao determinado pela lei.

Para a reversão deste cenário, é fundamental a adoção de tecnologias que proporcionem o desenvolvimento sustentável e auxiliem na criação de alternativas para correta destinação dos resíduos, aproveitando-os e evitando que eles cheguem aos aterros. (ANTENOR, SZIGETHY, 2020)

Neste cenário, a geração de resíduos e sua destinação, são fatores preocupantes e as informações sobre reciclagem e descarte correto nem sempre atingem toda a população, portanto uma das maneiras identificadas para auxiliar na conscientização é desenvolver uma plataforma que traga as informações que contribuam na compreensão sobre o tema.

Para suprir esta demanda, o desenvolvimento de aplicativo mobile propaga a informação para seus usuários, aumentando assim o acesso a informação sobre reciclagem tanto dos resíduos domésticos como dos eletrônicos, assim mais pessoas podem contribuir com a coleta seletiva e o meio ambiente, além de aumentar o alcance das iniciativas sustentáveis.

2. ESPECIFICAÇÕES INICIAIS DO SOFTWARE

Este capítulo apresenta os conceitos sobre o desenvolvimento das especificações iniciais da proposta de desenvolvimento do aplicativo móvel Android Cidade Sustentável.

2.1 Escopo do Produto

O escopo do produto consiste em desenvolver um aplicativo Android que permita aos usuários acessar informações sobre reciclagem e localizar pontos de coleta de lixo eletrônico, além de iniciativas sustentáveis de acordo com a sua localização integrada ao GPS do celular. A aplicação também permitirá sugestões de melhorias feitas pelos usuários através de uma função de formulário.

O aplicativo mobile também conta com integração de testes unitários e possui Banco de Dados em nuvem.

2.2 PÚBLICO-ALVO

O público alvo consiste em residentes de centros urbanos, de faixa etária adulta e com hábito de consumir conteúdos sobre reciclagem e sustentabilidade a partir de aplicativos para celulares inteligentes com sistema operacional Android.

2.3 Definições, Acrônimos e Abreviações

As abreviações explicando os conceitos de termos importantes citados no durante o desenvolvimento do projeto de software, constam no quadro abaixo:

Quadro 1: Abreviações e Definições:

Sigla	Significado	Definição
ONGs	Organizações não governamentais	Instituições que não pertencem a iniciativas públicas ou governamentais e não possuem fins lucrativos.
XP	Extreme Programming	Metodologia ágil de desenvolvimento de Software.
GPS	Global Positioning System	Sistema com navegação de medidas precisas de localização geográfica.
NoSQL	Not Only Standard Query Language	Linguagem para trabalhar com bancos de dados não relacionais.
Release	Lançamento de software	Release é uma versão de um Sistema de Software distribuído. Podem ser divididas em principais, que fornecem nova e significativa funcionalidade, e menores que reparam bugs e corrigem problemas de clientes que foram

		relatados.
RN	Regra de Negócio	Políticas referentes ao negócio.
SO	Sistema Operacional	Administrador de recursos de software e hardware. Estrutura que sustenta e administra todos os programas.
MySQL	Structured Query Language	Código aberto desenvolvido para a gestão de banco de dados.
Mb	Megabyte	Unidade de medida de informação que equivale a 1024 Kilobytes.
IDE	Integrated Development Environment	Ambiente de desenvolvimento integrado.
JSON	JavaScript Object Notation	Formato para troca de dados simples baseado em um atributo-valor.
BD	Banco de Dados	Conjuntos de registros relacionados.

Fonte: Autora.

2.4 Convenções

- Cidade Sustentável: nome do aplicativo desenvolvido.
- Activity: Elemento de interface do Android, em português significa atividade.

2.5. Requisitos de Produto de Software

Os requisitos de produto de software utilizados no durante o desenvolvimento do projeto de software, constam no quadro abaixo:

Quadro 2: Convenção de prioridades dos Requisitos:

Convenção de prioridades
Essencial - Requisito imprescindível para o funcionamento do software
Importante - Requisito sem o qual o software não funciona de forma satisfatória
Desejável - Requisito que não compromete as funcionalidades do sistema

Fonte: Autora.

Diagrama 1: Requisito Funcional 01:

[REQUISITO FUNCIONAL] RF 01 - ACESSO AO APLICATIVO
DESCRÍÇÃO:
Permitir que o usuário acesse o aplicativo sem credenciais, o celular do usuário precisa estar conectado a Internet.
PRIORIDADE: Essencial [] Importante [X] Desejável []

Fonte: Autora.

Diagrama 2: Requisito Funcional 02:

[REQUISITO FUNCIONAL] RF 02 - MENU DE PESQUISA	
Descrição: Permitir que o usuário acesse o aplicativo a visualize informações sobre como reciclar diferentes tipos de lixo.	
Prioridade: Essencial [X] Importante [] Desejável []	

Fonte: Autora.

Diagrama 3: Requisito Funcional 03:

[REQUISITO FUNCIONAL] RF 03 - LOCALIZAÇÃO	
Descrição: Permitir que o usuário acesse o Maps mesmo com seu GPS desligado, assim poderá visualizar os pontos de coleta de resíduos recicláveis e eletrônicos de sua cidade que estão armazenados na aplicação.	
Prioridade: Essencial [X] Importante [] Desejável []	

Fonte: Autora.

Diagrama 4: Requisito Funcional 04:

[REQUISITO FUNCIONAL] RF 04 - FORMULÁRIO
DESCRIÇÃO: <p>Permitir que o usuário preencha um formulário com sugestões de novas localizações de pontos de reciclagem e sua opinião sobre o aplicativo.</p>
PRIORIDADE: Essencial <input checked="" type="checkbox"/> Importante <input type="checkbox"/> Desejável <input type="checkbox"/>

Fonte: Autora.

Diagrama 5: Requisito Não Funcional 01:

[REQUISITO NÃO FUNCIONAL] RNF 01 - CONFIABILIDADE
DESCRIÇÃO: <p>O aplicativo deve permanecer disponível sem interrupções durante o uso, garantindo a conectividade do usuário.</p>
PRIORIDADE: Essencial <input type="checkbox"/> Importante <input checked="" type="checkbox"/> Desejável <input type="checkbox"/>

Fonte: Autora.

Diagrama 6: Requisito Não Funcional 02:

[REQUISITO NÃO FUNCIONAL] RNF 02 - USABILIDADE	
DESCRIÇÃO:	
<p>As interfaces do aplicativo devem ser simples e intuitivas para facilitar o acesso as informações que o usuário deseja, muitos usuários não possuem tempo para aprender a utilizar sistemas complexos.</p>	
PRIORIDADE: Essencial [] Importante [X] Desejável []	

Fonte: Autora.

Diagrama 7: Requisito de Domínio (Negócio) 01:

[REQUISITO DE NEGÓCIO] RN 01
DESCRIÇÃO:
A aprovação das sugestões e adições de localidades no Maps do aplicativo é de responsabilidade do desenvolvedor que possui acesso ao Banco de Dados.

Fonte: Autora.

Diagrama 8: Requisito de Domínio (Negócio) 02:

[REQUISITO DE NEGÓCIO] RN 02
<p>DESCRIÇÃO:</p> <p>A responsabilidade de manutenção do aplicativo é do desenvolvedor do aplicativo.</p>

Fonte: Autora.

2.6 Perspectiva do Produto

A perspectiva do aplicativo visa a conscientização sobre reciclagem e iniciativas voltadas à sustentabilidade dos residentes de Cidades Inteligentes, propondo uma aplicação interativa com interfaces amigáveis e de fácil acesso contendo informações sobre resíduos e seu descarte adequado, a ferramenta também possibilita interação da população aceitando contribuições sobre os temas citados anteriormente.

2.7 Funcionalidade do Produto

Função de Dicas: Interface mostrando dicas de reciclagem de resíduos orgânicos, plásticos, eletrônicos, metálicos.

Função de Localização no mapa: Interface integrada com mapa baseado no Google Maps, permitindo ao usuário navegar livremente.

Função de preenchimento de formulário: Interface contendo campo nome no qual pode ser digitado até 100 caracteres, campo email no qual pode ser digitado

até 100 caracteres e comentário aceitando até 400 caracteres, além de botão enviar formulário.

2.8 Usuários

Os seguintes usuários irão utilizar o software:

- Consumidor do aplicativo: sua função consiste em visualizar localizações, informações e interagir através de formulário.
- Desenvolvedor: sua função consiste em atualizar o software conforme necessidades de manutenção e gerenciar adições de novas localizações e informações.

2.9 Ambiente Operacional

A plataforma proposta utiliza o Android, baseado em SO Linux. Integração com Banco de Dados NoSql em nuvem.

2.10 Restrições de Projeto e Implementação

O aplicativo estará disponível apenas para celulares inteligentes com sistema Android. A linguagem Java e Kotlin. Espaço disponível de 80 MB em disco.

2.11 Documentação do Usuário

Tutorial para usuários:

1. Se for sua primeira vez acessando o aplicativo, ele não necessita de cadastro.

2. Para realizar consultar informações sobre sustentabilidade e reciclagem existe o campo Sustentabilidade.
3. Não é necessário inserir sua localização ou estar com o GPS ligado, basta clicar no mapa para navegar.
4. Para deixar um comentário ou sugestão clique no botão Sugestões e surgirá uma página contendo os campos nome, email e sugestão para digitar livremente.

2.12 Suposições e Dependências

A aplicação pode ser afetada pela falta de Internet do celular e caso a memória em disco for menor que 80 MB o sistema apresentará lentidão no carregamento das informações.

3. METODOLOGIA

A construção do produto de software baseou-se na metodologia XP para desenvolvimento e gestão do projeto. O Extreme Programming propõe requisitos expressos como histórias e implementados diretamente como uma série de tarefas. Além de boas práticas como Integração Contínua, Planejamento Incremental e Refatoração do código (SOMMERVILLE, 2011, p. 44-47), e Clean Code propondo desenvolvimento com tratamento de erros e testes de unidade (MARTIN, 2008).

Quadro 2 - Atividades para Desenvolvimento de Projeto de Software

Atividade	Artefatos resultantes
1. Engenharia de Requisitos e pesquisa teórica	Pesquisa sobre a temática, público-alvo e material teórico. Levantamento de requisitos funcionais e não funcionais do Sistema.
2. Elaboração da base Desenvolvimento do Aplicativo e metodologia ágil	Criação de Diagramas de caso de uso, diagrama de classes e modelo para banco de dados. Refinamento e criação de histórias(requisitos divididos em tarefas) para início do desenvolvimento.
3. Tecnologias utilizadas durante o Desenvolvimento	Prototipação de Interfaces. Desenvolvimento de Back-end Java/Kotlin e Interfaces criadas no Android Studio.
4. Criação de testes na aplicação, criação de documentação.	Criação de testes unitários e análise sobre refatoração e boas práticas seguindo a plataforma de código aberto SonarQube. Criação da documentação da aplicação.

Fonte: Autora.

3.1 Cronograma

O cronograma contém o planejamento de atividades a serem realizadas para alcançar os objetivos propostos na criação do Aplicativo Cidade +Sustentável.

Quadro 3 - Cronograma de Atividades

Atividades	J A N	F E V	M A R	A B R	M A I	J U N	J U L	A G O	S E T	O U T	N O V	D E Z
1. Engenharia de Requisitos e pesquisa teórica	X	X	X	X								
2. Elaboração da base Desenvolvimento do Aplicativo e metodologia ágil				X	X	X						
3. Tecnologias utilizadas durante o Desenvolvimento e produção do app mobile							X	X	X	X	X	
4. Criação de testes na aplicação											X	X

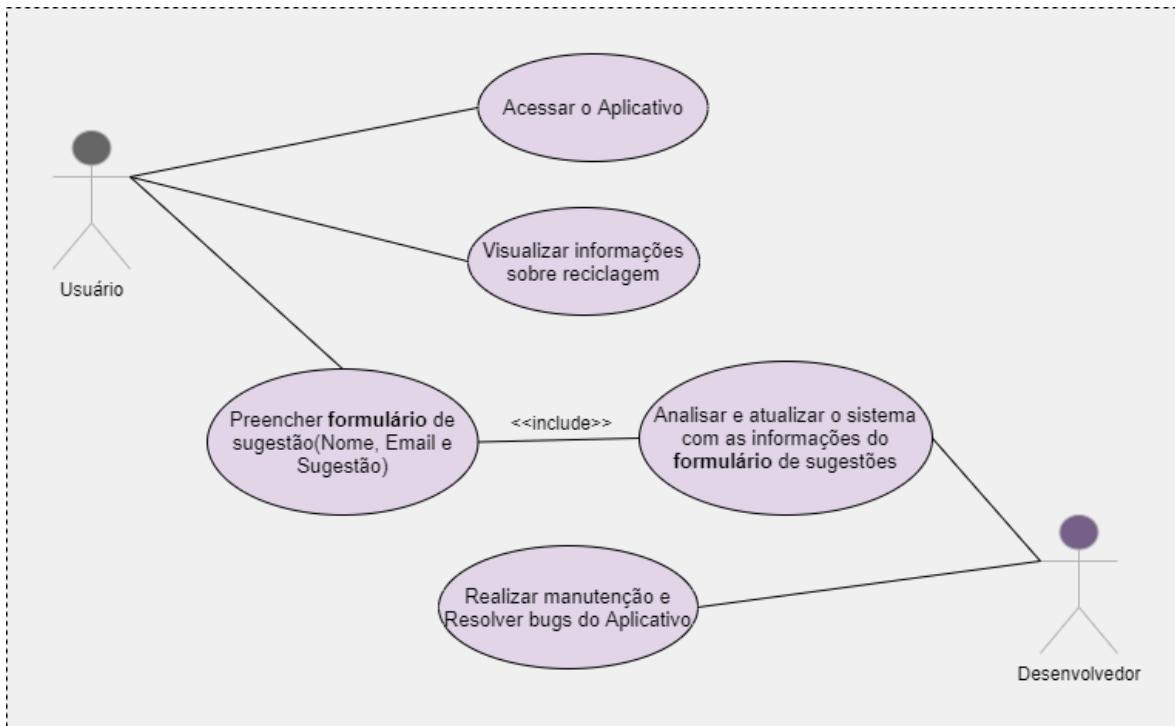
Fonte: a Autora.

4. DESENVOLVIMENTO

4.1. Diagrama de Caso de Uso

O diagrama de Caso de uso representa a interação entre o usuário e a aplicação.

Diagrama 9: Diagrama de Caso de Uso.

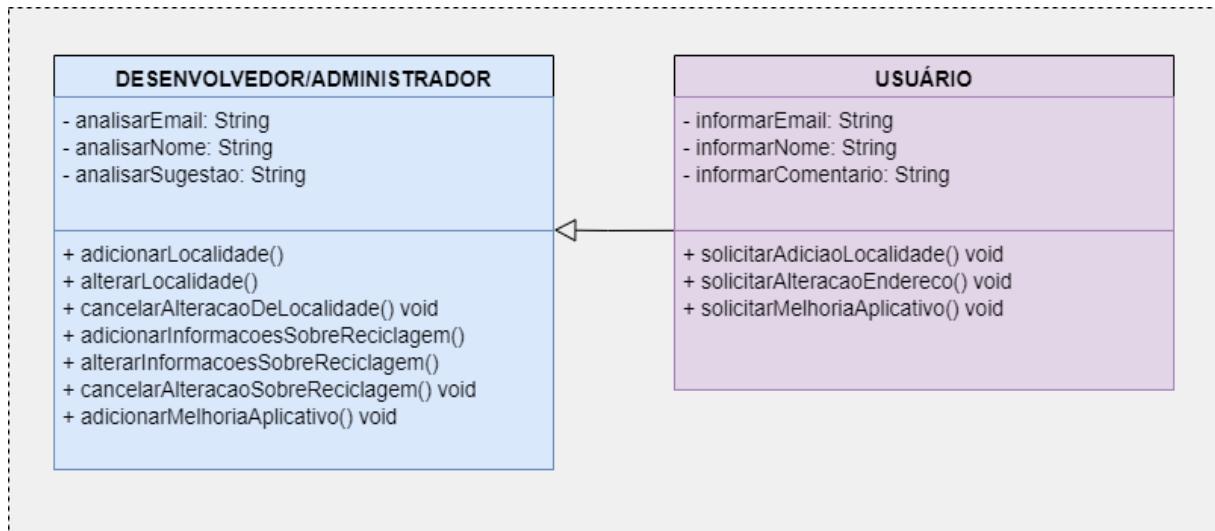


Fonte: Autora.

4.2. Diagrama de Classes

O diagrama de Classes representa a estrutura e relação das classes Desenvolvedor/Administrador e Usuário.

Diagrama 10: Diagrama de Classes:



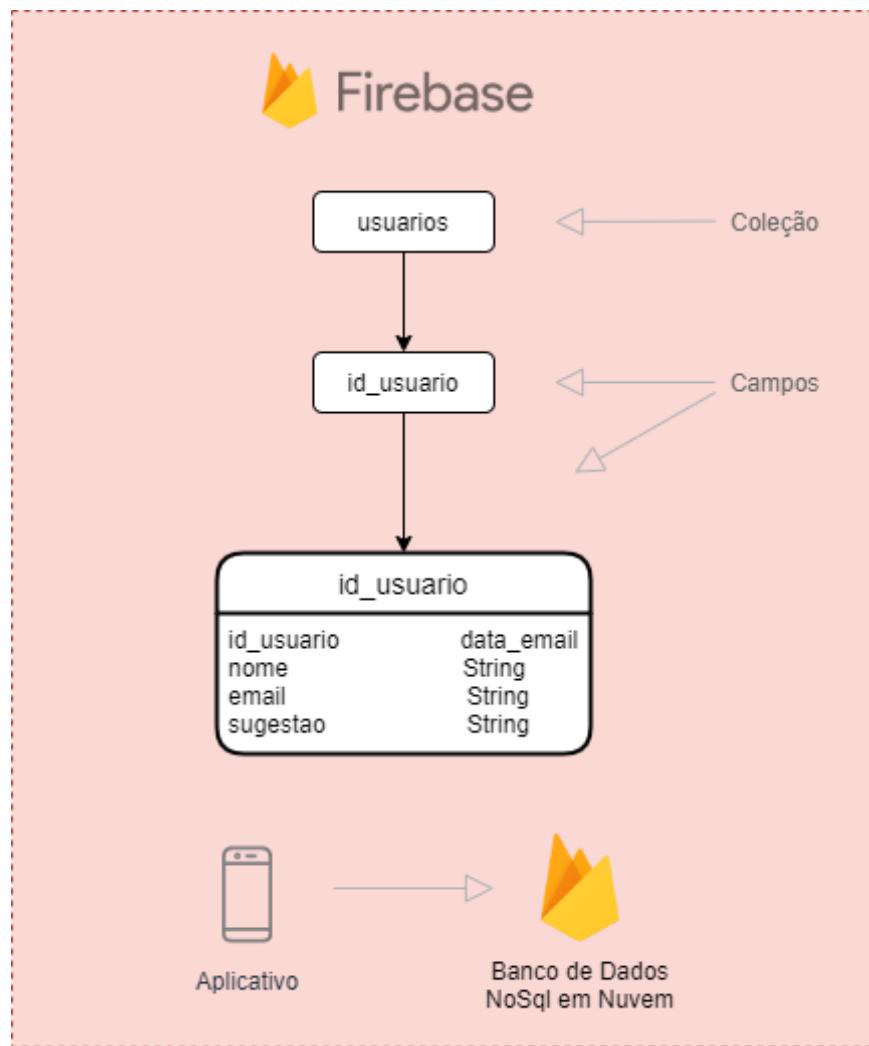
Fonte: Autora.

4.3. Modelo de Banco de Dados

Diagrama Entidade-Relacionamento(DER) em um Banco de Dados NoSql, as informações são gravadas por par de valores-chave compostos por valores String como um JSON(JavaScript Object Notation).

A plataforma utilizada foi o Firebase, pois contém um conjunto de serviços de hospedagem NoSQL para aplicativos em tempo real. (FIREBASE,2021)

Diagrama 11: Diagrama Entidade-Relacionamento do banco de dados na plataforma do Firebase(Cloud Firestore).



Fonte: Autora.

4.4. Arquitetura

A arquitetura de software implementada baseia-se na Arquitetura Android, a convenção de código segue a Atividade(Activity), a mesma serve como ponto de entrada para a interação de um aplicativo com seus usuários, uma atividade fornece a janela de interface, suas interações e lógicas de programação. (ANDROID, 2021)

A IDE empregada no desenvolvimento foi o Android Studio e o projeto contém as linguagens de programação Java e Kotlin, o Kotlin foi a linguagem predominante

pois possui suporte para os desenvolvedores segundo a documentação do Android.
(ANDROID, 2022)

Os requisitos mínimos para implantação:

1. O dispositivo deve possuir Sistema Android.
2. O dispositivo deve conter 30 MB de espaço livre em disco para instalação.
3. O dispositivo deve estar conectado à rede para acessar as informações da função Mapa.

4.5. Códigos-Fontes

Link do repositório no GitHub(plataforma de hospedagem de código-fonte) contendo os códigos-fonte e o APK: <https://github.com/maynicochelli/Aplicativo>

4.6. Telas do Aplicativo

Todas as telas e suas interações estão inseridas a seguir.

4.6.1. Interface do Menu Inicial

Na tela inicial, o usuário pode escolher qual funcionalidade do menu deseja acessar.

Imagen 1: Interface do Menu Inicial

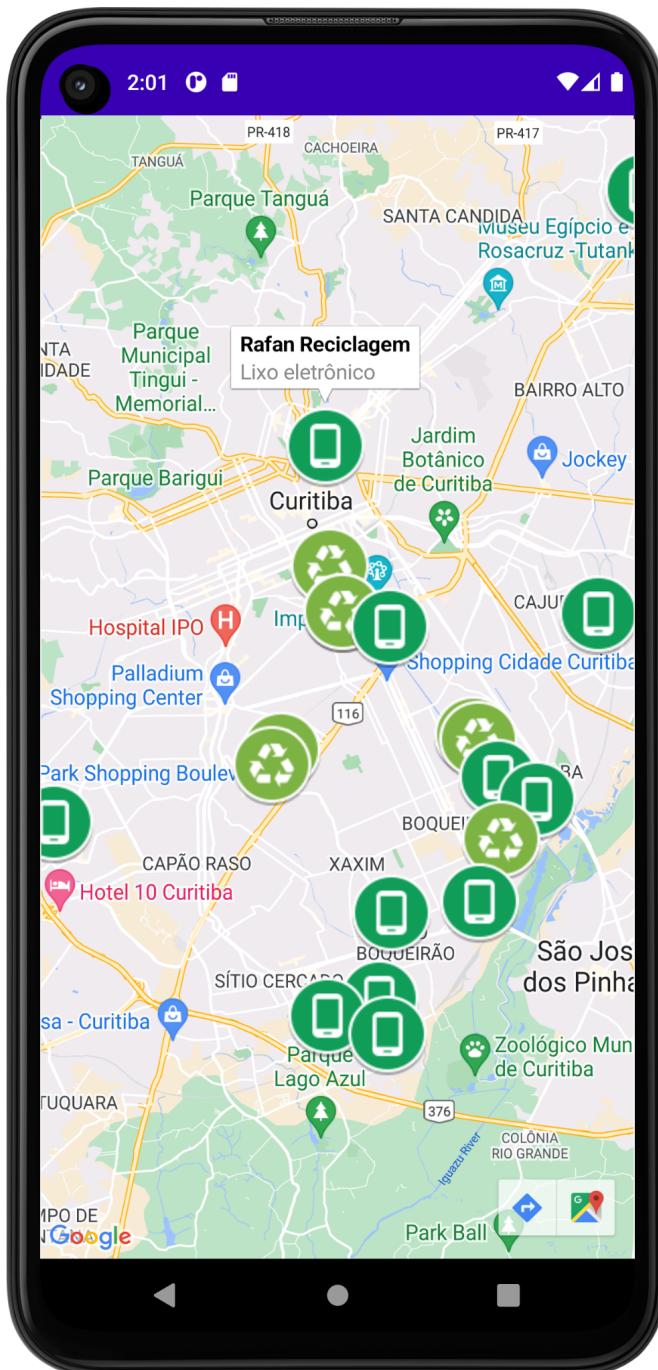


Fonte: Autora.

4.6.2. Interface do Maps

A função acionada quando o usuário clicar na opção “PONTOS DE COLETA”, a aplicação redireciona para o mapa, no mapa o usuário pode navegar e visualizar pontos de coleta de eletrônicos e recicláveis. A cidade mapeada para construção do aplicativo foi a de Curitiba, no estado do Paraná.

Imagen 2: Interface do Maps

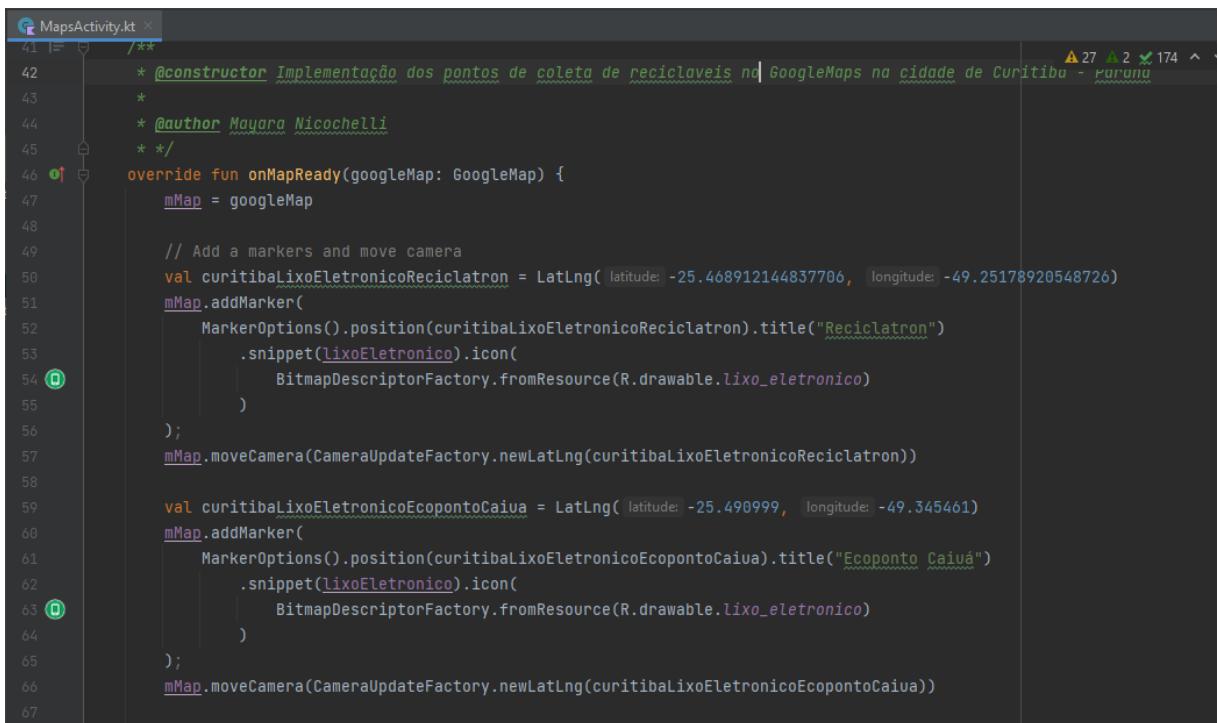


Fonte: Autora

4.6.3. Código de Atividade do mapa

O código corresponde a Atividade do mapa, novos pontos podem ser inseridos pelo desenvolvedor com as informações de latitude, longitude, título e o tipo (lixo eletrônico ou reciclável).

Imagen 3: Código de Atividade do mapa



The screenshot shows the code editor for the `MapsActivity.kt` file. The code is written in Kotlin and implements a Google Map activity. It adds two markers to the map: one for a recycling point in Curitiba and another for an electronic waste collection point in Caiuá. The code includes comments explaining the purpose of each section and the author's name.

```
41  /**
42   * @constructor Implementação dos pontos de coleta de recicláveis na GoogleMaps na cidade de Curitiba - Paraná
43   *
44   * @author Mayara Nicochelli
45   */
46 override fun onMapReady(googleMap: GoogleMap) {
47     mMap = googleMap
48
49     // Add a markers and move camera
50     val curitibaLixoEletronicoReciclatron = LatLng( latitude: -25.468912144837706, longitude: -49.25178920548726)
51     mMap.addMarker(
52       MarkerOptions().position(curitibaLixoEletronicoReciclatron).title("Reciclatron")
53         .snippet(LixoEletronico).icon(
54           BitmapDescriptorFactory.fromResource(R.drawable.lixo_eletronico)
55         )
56     );
57     mMap.moveCamera(CameraUpdateFactory.newLatLng(curitibaLixoEletronicoReciclatron))
58
59     val curitibaLixoEletronicoEcopontoCaiua = LatLng( latitude: -25.490999, longitude: -49.345461)
60     mMap.addMarker(
61       MarkerOptions().position(curitibaLixoEletronicoEcopontoCaiua).title("Ecoponto Caiuá")
62         .snippet(LixoEletronico).icon(
63           BitmapDescriptorFactory.fromResource(R.drawable.lixo_eletronico)
64         )
65     );
66     mMap.moveCamera(CameraUpdateFactory.newLatLng(curitibaLixoEletronicoEcopontoCaiua))
67 }
```

Fonte: Autora

4.6.4. Interface de Sustentabilidade:

A função acionada quando o usuário clica no menu inicial na opção “SUSTENTABILIDADE”, a aplicação redireciona para uma tela com rolagem horizontal contendo informações sobre reciclagem.

Imagen 4: Informações sobre a reciclagem de plásticos



Fonte: Autora

Imagen 5: Informações sobre a reciclagem de vidro



Fonte: Autora

Imagen 6: Informações sobre a reciclagem de metal



Fonte: Autora

Imagen 7: Informações sobre a reciclagem de papel



Fonte: autora

Imagen 8: Informações sobre o lixo orgânico



Fonte: Autora

Imagen 10: Informações sobre a reciclagem de lixo eletrônico



Fonte: Autora

4.6.5. Interface de Formulário:

A função acionada quando o usuário clica no menu inicial para "SUGESTÕES", a aplicação redireciona para a tela do formulário, contendo os campos nome, email e sugestões que podem ser preenchidos.

Imagen 11: Interface de Formulário



Fonte: Autora

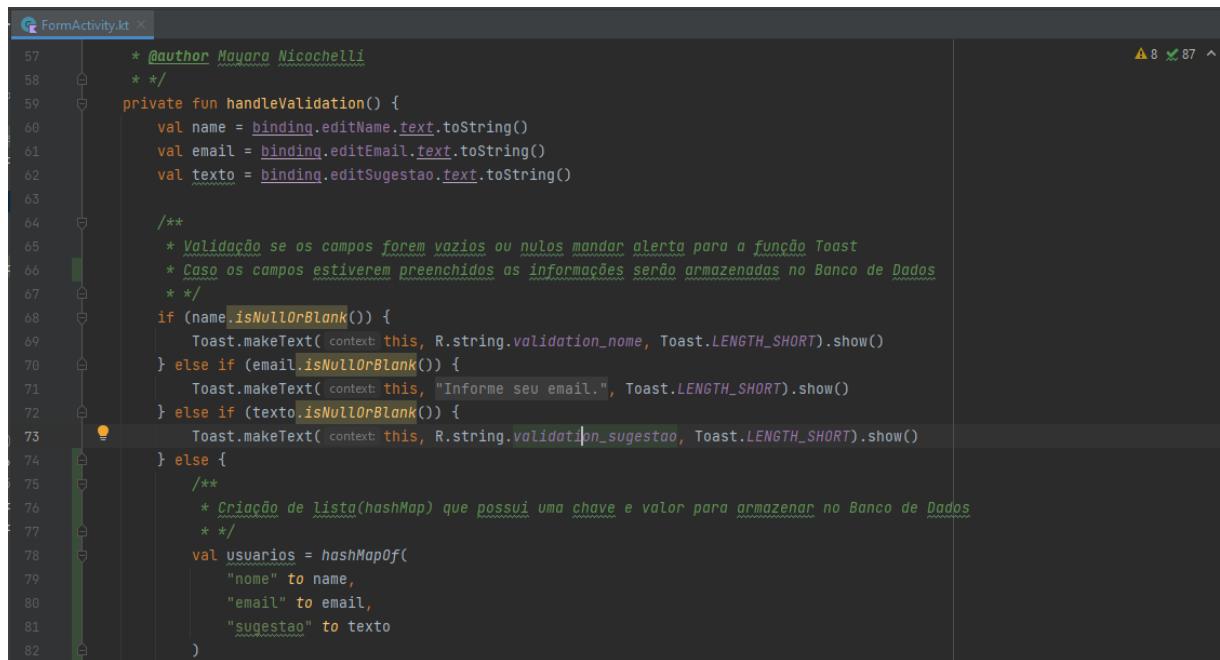
Imagen 12: Função de notificação



Fonte: Autora

A função de notificação dos campos não preenchidos caso o botão de enviar seja acionado pelo usuário.

Imagen 13: Código da atividade formulário



```
FormActivity.kt
57 * @author Mayara Nicchelli
58 */
59 private fun handleValidation() {
60     val name = binding.editName.text.toString()
61     val email = binding.editEmail.text.toString()
62     val texto = binding.editSugestao.text.toString()
63
64     /**
65      * Validação se os campos forem vazios ou nulos mandar alerta para a função Toast
66      * Caso os campos estiverem preenchidos as informações serão armazenadas no Banco de Dados
67      */
68     if (name.isNullOrEmpty()) {
69         Toast.makeText(context: this, R.string.validation_nome, Toast.LENGTH_SHORT).show()
70     } else if (email.isNullOrEmpty()) {
71         Toast.makeText(context: this, "Informe seu email.", Toast.LENGTH_SHORT).show()
72     } else if (texto.isNullOrEmpty()) {
73         Toast.makeText(context: this, R.string.validation_sugestao, Toast.LENGTH_SHORT).show()
74     } else {
75         /**
76          * Criação de lista(hashMap) que possui uma chave e valor para armazenar no Banco de Dados
77          */
78         val usuarios = hashMapOf(
79             "nome" to name,
80             "email" to email,
81             "sugestao" to texto
82         )
    }
```

Fonte: Autora

O código corresponde a Atividade do formulário, o método *handleValidation* contém a lógica para validação dos campos nome, e-mail e sugestão.

Imagen 14: Exemplo de preenchimento de formulário de sugestão



Fonte: Autora

Exemplo de tela com informações inseridas pelo usuário.

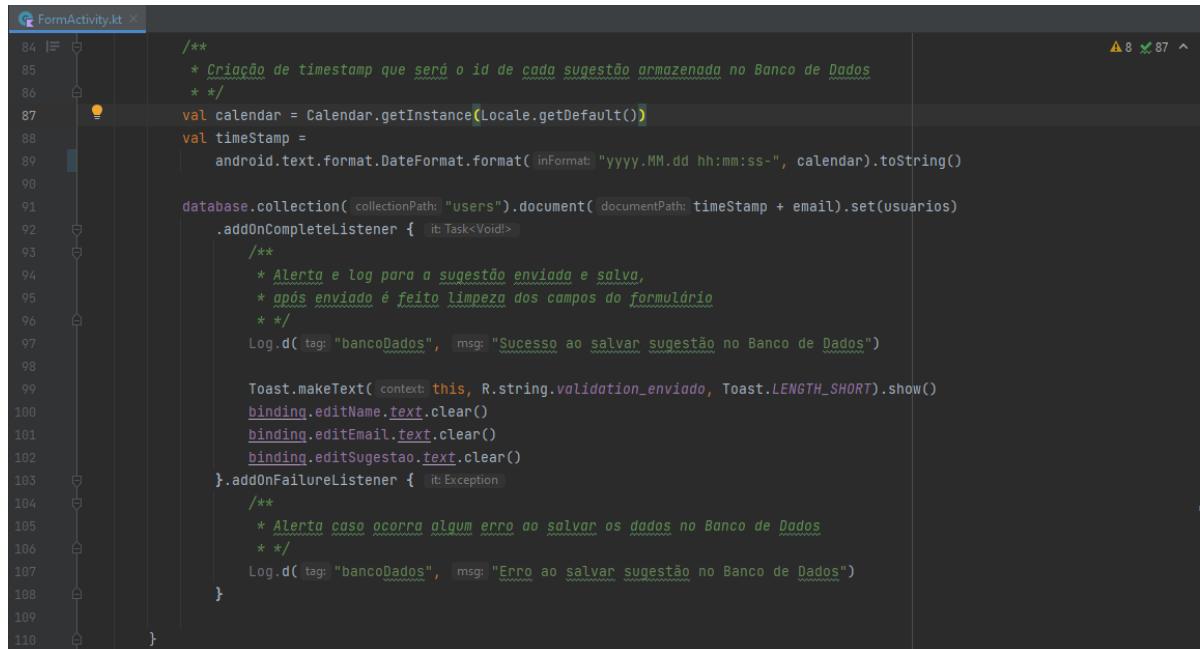
Imagen 15: Notificação de envio de formulário



Fonte: Autora

A função de notificação confirmado o envio da sugestão do usuário quando o botão enviar é acionado.

Imagen 16: Código da atividade de formulário de sugestão com a implementação da lógica para armazenamento no BD



The screenshot shows the Android Studio code editor with the file `MainActivity.kt` open. The code is written in Kotlin and handles the submission of a suggestion form. It includes logic to generate a timestamp, save the data to a Firestore database, and provide feedback to the user via Toast notifications and field clearances. The code is annotated with comments in Portuguese explaining its purpose.

```
84 /**
85 * Criação de timestamp que será o id de cada sugestão armazenada no Banco de Dados
86 */
87 val calendar = Calendar.getInstance(Locale.getDefault())
88 val timeStamp =
89     android.text.format.DateFormat.format("yyyy.MM.dd hh:mm:ss-", calendar).toString()
90
91 database.collection("users").document("documentPath: timeStamp + email").set(usuarios)
92     .addOnCompleteListener { it: Task<Void!>
93         /**
94          * Alerta e log para a sugestão enviada e salva,
95          * após enviado é feito limpeza dos campos do formulário
96          */
97         Log.d("bancoDados", "Sucesso ao salvar sugestão no Banco de Dados")
98
99         Toast.makeText(context, R.string.validation_enviado, Toast.LENGTH_SHORT).show()
100        binding.editName.text.clear()
101        binding.editEmail.text.clear()
102        binding.editSugestao.text.clear()
103    }.addOnFailureListener { it: Exception
104        /**
105          * Alerta caso ocorra algum erro ao salvar os dados no Banco de Dados
106          */
107        Log.d("bancoDados", "Erro ao salvar sugestão no Banco de Dados")
108    }
109
110 }
```

Fonte: Autora

O código corresponde a Atividade do formulário, o registro no banco de dados contém o id(composto pela data mais o e-mail do usuário), nome, e-mail e sugestão. Os dados localizam-se armazenados em um BD na nuvem e disponíveis para consulta pelo desenvolvedor/administrador.

Imagen 17: Banco de Dados NoSql

The screenshot shows the Firebase Realtime Database interface. The path is 'users > 2022.12.03-01:54:40-may@gmail.com'. The left sidebar shows a collection named 'users' with a single item. The main panel displays a document with fields: 'email: "may@gmail.com"', 'nome: "Mayara"', and 'sugestao: "Sugestao..."'. There are also buttons for 'Iniciar coleção' (Create Collection), 'Adicionar documento' (Add Document), and 'Adicionar campo' (Add Field).

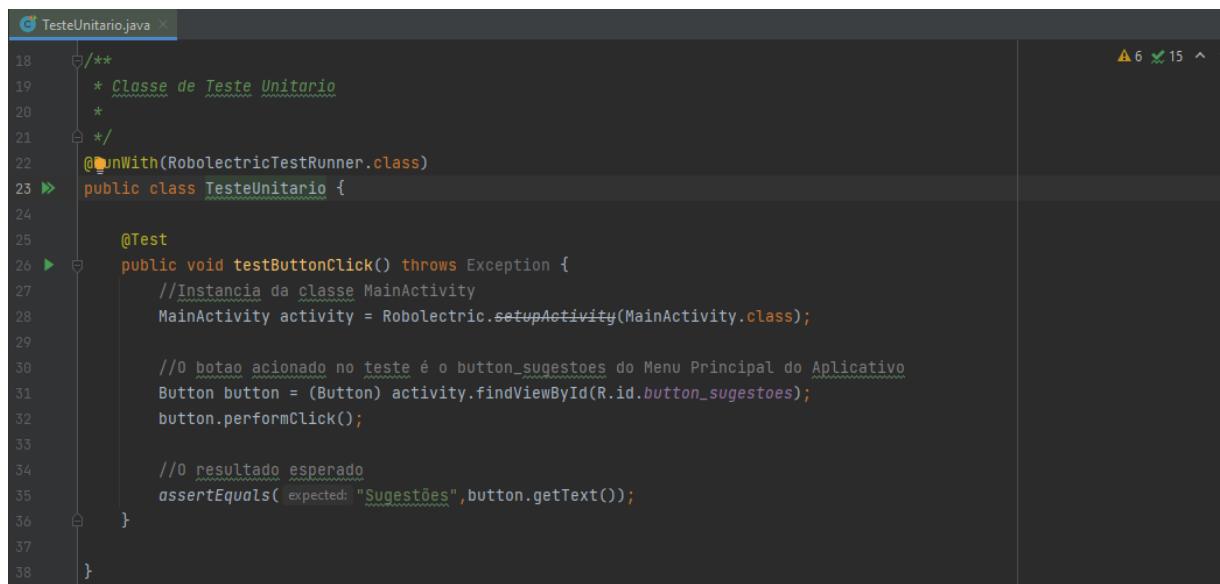
Fonte: Autora

Os dados são armazenados em um Banco de Dados NoSql e as informações são gravadas por valores String como um JSON na plataforma do Firebase, os dados podem ser consultados em tempo real através do console da plataforma.

4.6.6. Classes de testes

Os testes unitários garantem o funcionamento dos componentes de um programa, como métodos e classes. Os testes contribuem para o bom funcionamento e evitam falhas de programação. (SOMMERVILLE, 2011, p. 156)

Imagen 18: Teste unitário



```
18  */
19  * Classe de Teste Unitario
20  *
21  */
22 @RunWith(RobolectricTestRunner.class)
23 public class TesteUnitario {
24
25     @Test
26     public void testButtonClick() throws Exception {
27         //Instancia da classe MainActivity
28         MainActivity activity = Robolectric.setupActivity(MainActivity.class);
29
30         //O botao acionado no teste é o button_sugestoes do Menu Principal do Aplicativo
31         Button button = (Button) activity.findViewById(R.id.button_sugestoes);
32         button.performClick();
33
34         //O resultado esperado
35         assertEquals("Sugestões",button.getText());
36     }
37
38 }
```

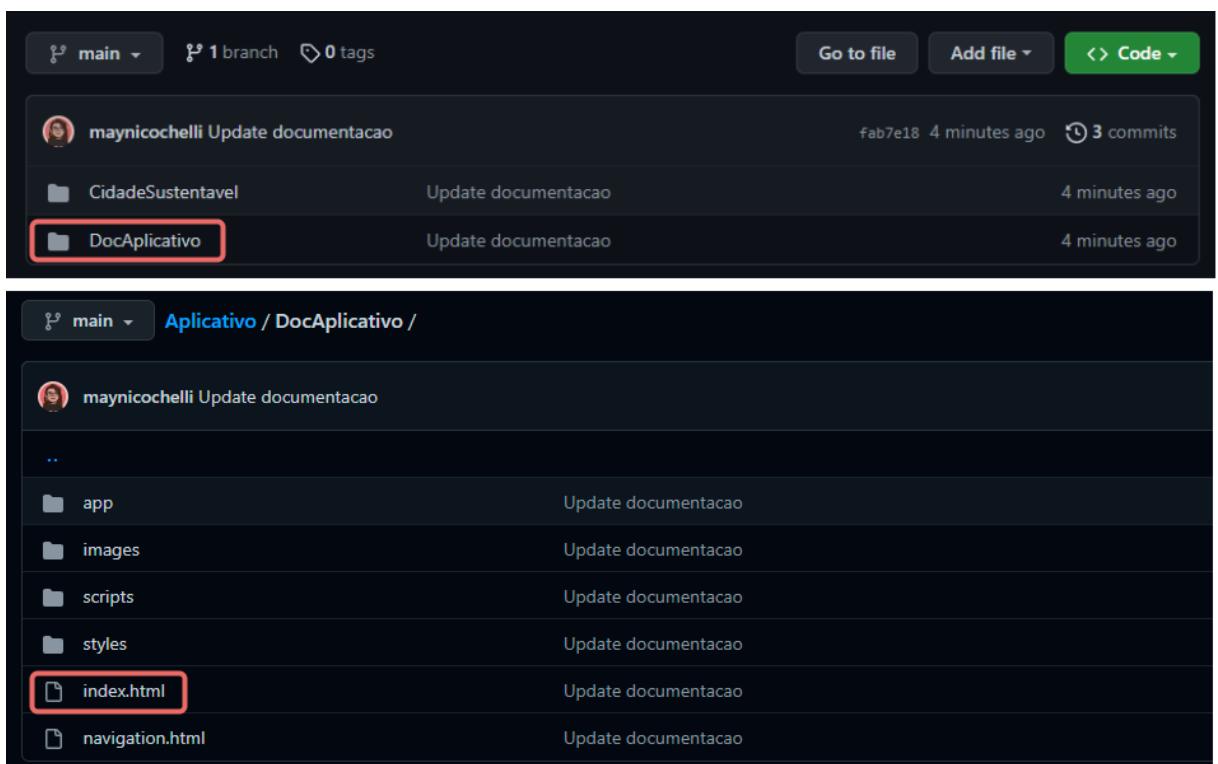
Fonte: Autora

O código corresponde a um teste unitário da função do botão “Sugestões” do menu inicial do aplicativo.

4.6.7. Documentação do Aplicativo

A Documentação do Aplicativo permite uma visualização das classes e métodos desenvolvidos para o funcionamento do Aplicativo, esta documentação foi gerada pela ferramenta gratuita Dokka(documentação para linguagem de programação Kotlin). (KOTLIN..., 2020)

Imagen 19: Localização da documentação no repositório



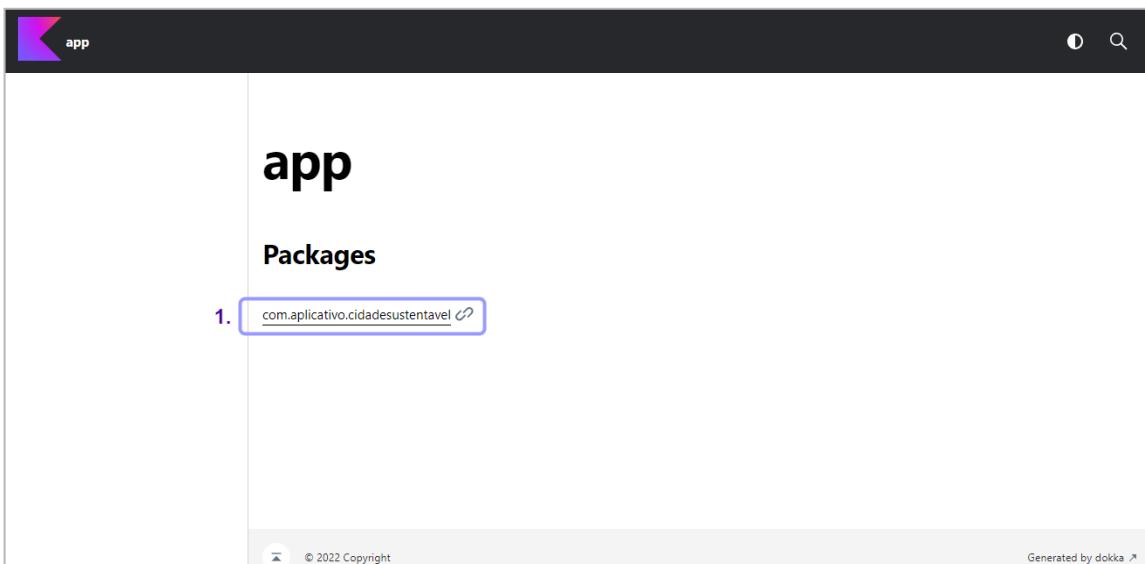
Fonte: Autora

A imagem apresenta a localização da documentação no repositório, o index.xml, que pode ser executado em qualquer navegador para internet.

A documentação pode ser visualizada utilizando o seguinte tutorial:

1. Clicando na pasta do aplicativo pode-se visualizar a documentação completa

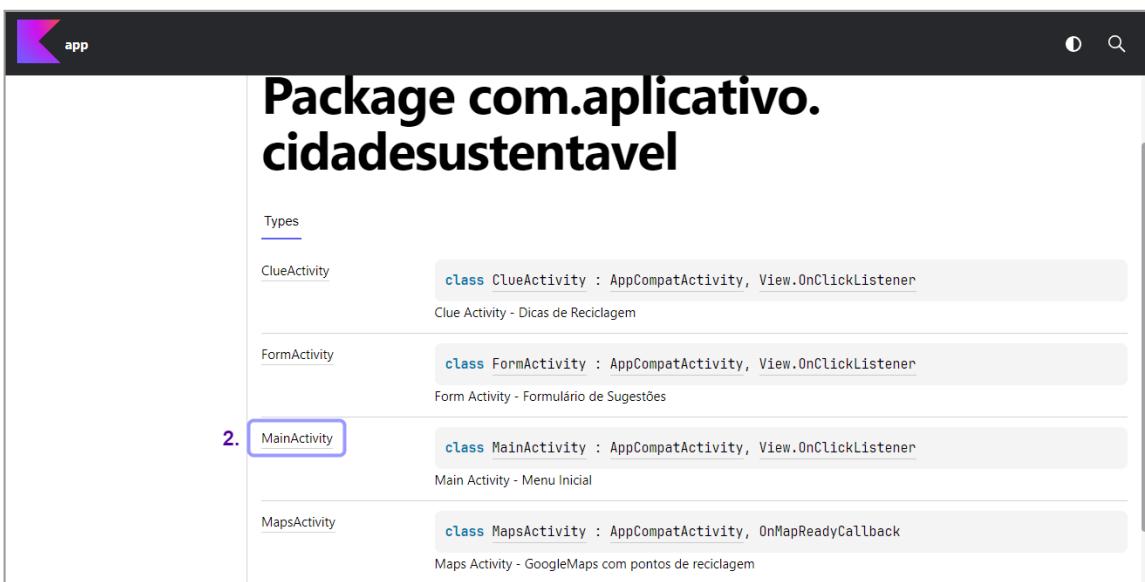
Imagen 20: Passo 1



Fonte: Autora

2. Clicando na classe MainActivy pode-se visualizar os métodos usados nesta classe.

Imagen 21: Passo 2



Fonte: Autora

3. Construtores usados na classe MainActivity e sua descrição

Imagen 22: Passo 3

```
class MainActivity : AppCompatActivity, View.OnClickListener
```

Main Activity - Menu Inicial

3. Constructors Functions Properties

MainActivity

fun MainActivity()

Activity do Menu inicial do Aplicativo

© 2022 Copyright Generated by dokka

Fonte: Autora

4. Funções utilizadas na classe MainActivity

Imagen 23: Passo 4

```
class MainActivity : AppCompatActivity, View.OnClickListener
```

Main Activity - Menu Inicial

4. Constructors Functions Properties

addContentView

open override fun addContentView(p0: View, p1: ViewGroup.LayoutParams)

addMenuProvider

open override fun addMenuProvider(@NonNull p0: MenuProvider)

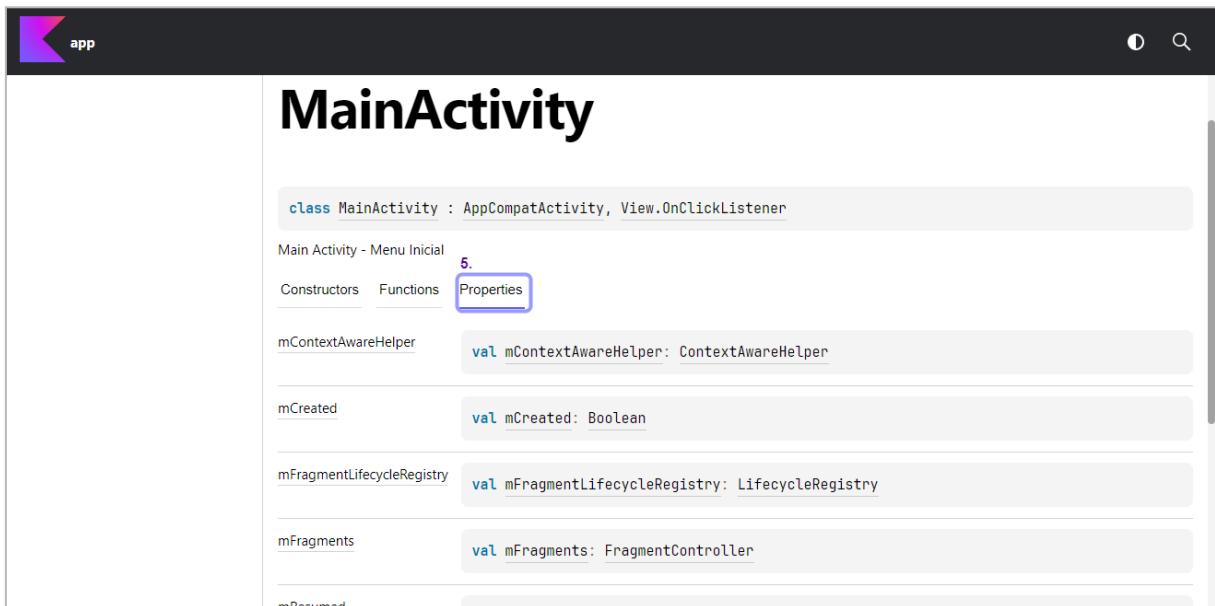
open override fun addMenuProvider(@NonNull p0: MenuProvider, @NonNull p1: LifecycleOwner)

open override fun addMenuProvider(@NonNull p0: MenuProvider, @NonNull p1: LifecycleOwner, @NonNull p2: Lifecycle.State)

Fonte: Autora

5. Propriedades utilizadas na classe MainActivity

Imagen 24: Passo 5



The screenshot shows the Android Studio code editor with the file `MainActivity.kt` open. The title bar says "MainActivity". The code editor has a dark theme with a purple header bar. The main area displays the following Kotlin code:

```
class MainActivity : AppCompatActivity, View.OnClickListener
```

Below the code, there is a navigation bar with tabs: "Constructors", "Functions", and "Properties". The "Properties" tab is highlighted with a blue border. A small number "5." is displayed above the tab. The properties listed are:

- `mContextAwareHelper`: `val mContextAwareHelper: ContextAwareHelper`
- `mCreated`: `val mCreated: Boolean`
- `mFragmentLifecycleRegistry`: `val mFragmentLifecycleRegistry: LifecycleRegistry`
- `mFragments`: `val mFragments: FragmentController`
- `mResumed`

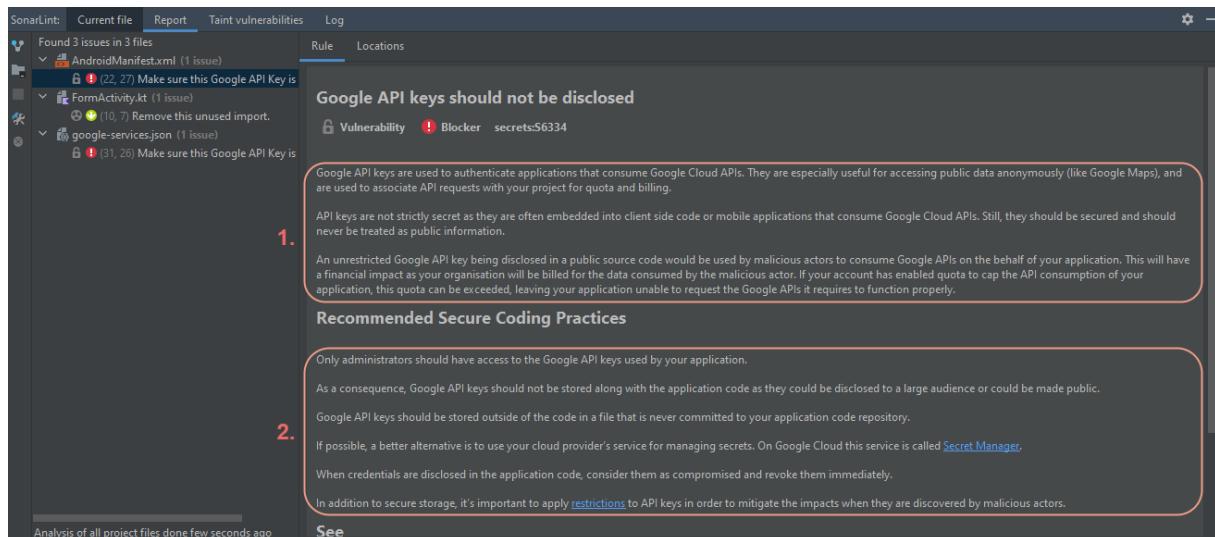
Fonte: Autora

4.6.8. Segurança e Qualidade de Software

O Sonar é uma plataforma de código aberto cuja função é inspecionar a qualidade de código para detectar bugs e code smells em variadas linguagens de programação. (SONARQUBE..., 2008)

Os arquivos do aplicativo foram analisados pelo plugin do Sonar na IDE do Android Studio e gerou sugestões de melhorias quanto à segurança e boas práticas de programação da linguagem Kotlin.

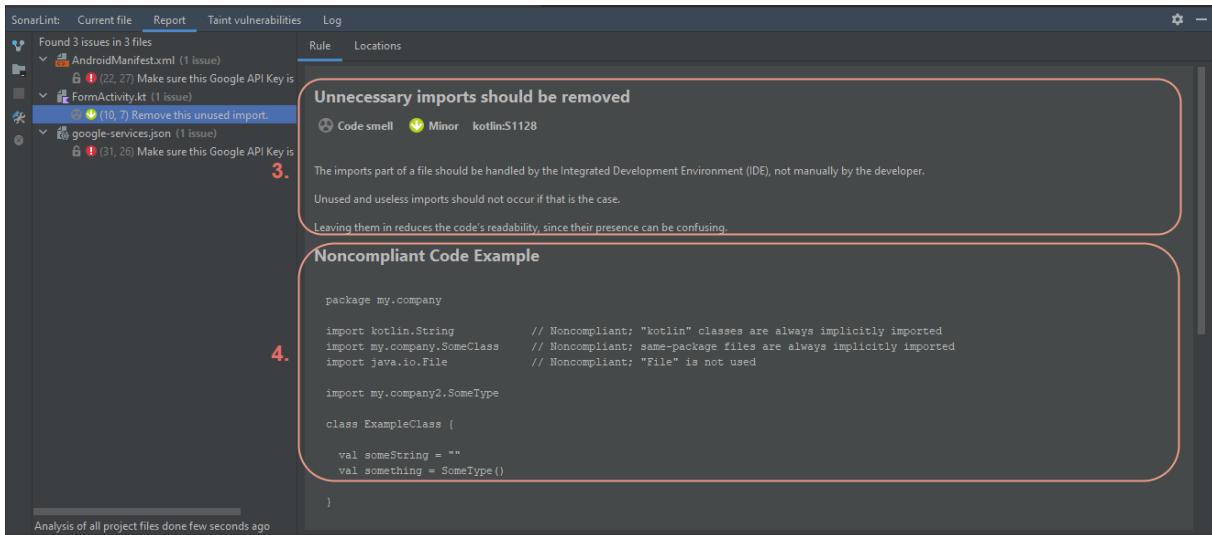
Imagen 25: Resultado da análise plugin SonarLint 01



Fonte: Autora

1. Uma vulnerabilidade de segurança foi apontada pelo Sonar, uma chave exclusiva do Google Maps utilizada na construção da função Mapa do Aplicativo está exposta no código e pode ser consumida por algum ator malicioso e o Google API poderá restringir a aplicação.
2. O sonar sugeriu que a chave em questão deve ser armazenada fora do código.

Imagen 26: Resultado da análise plugin SonarLint 02



Fonte: Autora

3. Uma prática inadequada de programação apontada foi uma dependência não usada no projeto, a sugestão é retirar a biblioteca não utilizada.
4. O Sonar gerou um exemplo da prática inadequada analisada.

5. CONSIDERAÇÕES FINAIS

O desenvolvimento do Aplicativo Android para cidades inteligentes visou fortalecer a conscientização de grande parte da população brasileira sobre temas relacionados à reciclagem e descarte correto do lixo, o aplicativo contém informações de como e onde reciclar o lixo eletrônico e outros resíduos.

A problemática do descarte de lixo nos centros urbanos atuais e a desinformação sobre a importância da reciclagem convergem para reflexão de possíveis soluções para o cenário, neste contexto a solução digital abordada consiste em uma aplicação disponível para usuários de celulares que obtendo as informações corretas poderão minimizar o acúmulo dos resíduos produzidos diariamente em suas comunidades.

As melhorias futuras para o aplicativo consistem em seguir o cadastramento de novas cidades e pontos de coleta de lixo eletrônico e reciclável, além das melhorias sugeridas pela análise do projeto pela plataforma do Sonar que visam melhorias de segurança e qualidade de código, fortalecendo a segurança para o aplicativo e seus usuários.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDROID Developer: **Introdução a atividades.** [S. I.], 2021. Disponível em: <https://developer.android.com/guide/components/activities/intro-activities?hl=pt-b>. Acesso em: 09 nov. 2022.

ANDROID Developer: **Desenvolver apps Android com o Kotlin.** [S. I.], 2022. Disponível em: <https://developer.android.com/kotlin>. Acesso em: 12 nov. 2022.

ANTENOR, Samuel; SZIGETHY, Leonardo. **Resíduos sólidos urbanos no Brasil: desafios tecnológicos, políticos e econômicos.** IPEA, 2020. Disponível em: <https://www.ipea.gov.br/cts/pt/central-de-conteudo/artigos/artigos/217-residuos-solidos-urbanos-no-brasil-desafios-tecnologicos-politicos-e-economicos>. Acesso em: 26 abr. 2022.

CAMARGO, Gustavo; MOURA, Lívia. **Impacto econômico e social do Android no Brasil.** Bain & Company, 2019. Disponível em: <https://www.bain.com/contentassets/a9200a057a0241b8963c05a9b09e33fe/impactos-do-android-no-brasil.pdf>. Acesso em: 15 abr. 2022.

FIREBASE for Android: **Adicione o Firebase ao seu projeto Android.** [S. I.], 2021. Disponível em: <https://firebase.google.com/docs/android/setup>. Acesso em: 30 nov. 2022.

GREEN ELETRON. **Resíduos Eletrônicos no Brasil - 2021,** 2021. Disponível em: https://greeneletron.org.br/download/RELATORIO_DE_DADOS.pdf. Acesso em: 18 abr. 2022.

IPSOS. **A Throwaway World: The Challenge of Plastic Packaging and Waste,** 2019. Disponível em: <https://www.ipsos.com/sites/default/files/ct/news/documents/2019-11/a-throwaway-world-global-advisor.pdf>

MARTIN, Robert C. **Código Limpo: Habilidades Práticas do Agile Software.** p 104. p 121. Prentice Hall, 2008. 431 p.

MEIRELLES, F. S. **30ª Pesquisa Anual FGcia do Uso de TI. FGV-EAESP**, 2019. Disponível em: https://eaesp.fgv.br/sites/eaesp.fgv.br/files/noticias2019fgcia_2019.pdf. Acesso em: 19 abr. 2022.

PNAD. **Pesquisa Nacional por Amostra de Domicílios de 2015**. Disponível em <https://biblioteca.ibge.gov.br/visualizacao/livros/liv98887.pdf>. Acesso em 16 abr. 2022.

SOMMERVILLE, Ian. **Engenharia de Software**. 9º edição. ed. São Paulo: Pearson Education do Brasil, 2011. 548 p.

SONARQUBE Documentation: **Developing with Sonar**. [S. l.], 2008. Disponível em: <https://docs.sonarqube.org/latest/>. Acesso em: 14 nov. 2022.

UNIÃO EUROPÉIA. **WHAT are smart cities?**, 2018. Disponível em: https://ec.europa.eu/info/eu-regional-and-urban-development/topics/cities-and-urban-development/city-initiatives/smart-cities_en. Acesso em: 16 abr. 2022.