

**FATEC DOM AMAURY CASTANHO**  
**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

ANGELO FERRAZ MODANEZ

JOÃO VITOR DE PAULA OLIVEIRA

JÚLIA CANAVEZZI DE OLIVEIRA

MARIA EDUARDA DO CARMO BRITO

MURILO DOS SANTOS

PEDRO VINICIUS SILVA

RAFAEL SCALET DE LIMA

YUUYA OKATANI

**ATIVIDADE PRÁTICA “O BAIRRO INTELIGENTE”**

Itu

2025

ANGELO FERRAZ MODANEZ

JOÃO VITOR DE PAULA OLIVEIRA

JÚLIA CANAVEZZI DE OLIVEIRA

MARIA EDUARDA DO CARMO BRITO

MURILO DOS SANTOS

PEDRO VINICIUS SILVA

RAFAEL SCALET DE LIMA

YUUYA OKATANI

### **ATIVIDADE PRÁTICA “O BAIRRO INTELIGENTE”**

Trabalho em grupo, apresentado ao Professor Sérgio Salgado, como requisito para a obtenção de nota nas disciplinas Sistemas de Informação e Tecnologias Emergentes e Sistemas Distribuídos a Internet das Coisas com o tema: Atividade Prática “O Bairro Inteligente”

Itu

2025

## SUMÁRIO

<b>RESUMO .....</b>	<b>4</b>
<b>1. DEFINIÇÃO DO PROBLEMA .....</b>	<b>5</b>
<b>2. INTRODUÇÃO .....</b>	<b>5</b>
<b>3. STAKEHOLDERS .....</b>	<b>5</b>
<b>4. REQUISITOS FUNCIONAIS .....</b>	<b>6</b>
<b>4.1 REQUISITOS NÃO FUNCIONAIS .....</b>	<b>6-7</b>
<b>5. DADOS COLETADOS E FORNECIDOS .....</b>	<b>7</b>
<b>6. DESENHO DA ARQUITETURA .....</b>	<b>8</b>
6.1 Camada de Percepção (IoT) .....	8
6.2 Camada de Rede (Comunicação) .....	8
6.3 Camada de Processamento (Cloud / Distribuído) .....	8
6.4 Camada de Aplicação (Apresentação) .....	8
<b>7. DIAGRAMA .....</b>	<b>9</b>

## RESUMO

O projeto consiste no desenvolvimento de um sistema inteligente para monitoramento de lixeiras, utilizando tecnologias de IoT e aplicações web. O objetivo é modernizar a gestão de resíduos, permitindo que gestores acompanhem, em tempo real, o nível de preenchimento das lixeiras distribuídas pela cidade.

Cada lixeira conta com um sensor ultrassônico HC-SR04 conectado a um micro controlador ESP32, responsável por medir a distância do lixo até o topo e converter esse valor em porcentagem de ocupação. O ESP32 envia periodicamente esses dados para a nuvem por meio de comunicação Wi-Fi utilizando requisições HTTP direcionadas ao Firebase Realtime Database.

Possuímos uma interface desenvolvida em React que consome os dados do Firebase via WebSocket, exibindo um dashboard atualizado em tempo real, permitindo que gestores visualizem o status de todas as lixeiras, recebam alertas quando atingem níveis críticos e cadastrem novas lixeiras.

## **1. DEFINIÇÃO DO PROBLEMA**

A gestão de resíduos sólidos urbanos é ineficiente, com acúmulo de lixo, coleta mal planejada e falta de monitoramento em tempo real, impactando a saúde pública, o meio ambiente e a qualidade de vida. Entre os principais problemas estão: desperdício de recursos, proliferação de agentes transmissores, queda na qualidade de vida e redução, da qualidade de vida da população. Nesse contexto, um sistema IoT pode monitorar lixeiras, otimizar rotas e fornecer dados estratégicos para gestores, coletores e população. Nesse contexto, um sistema de IoT para a gestão de resíduos.

Lixeiras sem monitoramento podem estar vazias na coleta, cheias antes do horário ou danificadas, provocando sujeira, saúde pública prejudicada, reclamações, congestionamentos e altos custos operacionais.

## **2. INTRODUÇÃO**

O projeto objetiva desenhar e simular uma arquitetura IoT para um “bairro inteligente” que resolva problemas urbanos reais em particular a gestão ineficiente de resíduos sólidos por meio da coleta de dados em campo, processamento distribuído e interfaces de visualização para stakeholders (cidadãos, prefeitura, empresas de coleta, manutenção e administradores).

## **3. STAKEHOLDERS**

Como stakeholder poderíamos citar:

- Cidadãos (querem as ruas limpas e informações sobre coleta)
- Prefeitura (exige monitoramento e redução de custos)
- Empresa de Coleta (ordens de serviço e atualização em tempo real)
- Equipe de TI (responsável por implantação, operação e garantia de segurança)
- Gestores (controle e configuração de acessos.)

## 4. REQUISITOS FUNCIONAIS

**RF01 – Coletar nível da lixeira:** O sistema deve realizar a leitura do sensor ultrassônico HC-SR04 instalado em cada lixeira, obtendo a distância até o lixo.

**RF02 – Calcular a porcentagem de ocupação:** O ESP32 deve enviar essa leitura em cm e a aplicação irá transformar a leitura em porcentagem (%).

**RF03 – Enviar dados ao Firebase:** O ESP32 deve enviar periodicamente o valor da porcentagem para o Firebase via HTTP.

**RF04 – Armazenar dados em tempo real:** O Firebase Realtime Database deve registrar e atualizar os dados de cada lixeira.

**RF05 – Atualizar a dashboard automaticamente:** A aplicação React deve receber os dados em tempo real via WebSocket.

**RF06 – Exibir status de cada lixeira:** A dashboard deve mostrar a porcentagem atual de cada lixeira monitorada.

**RF07 – Emitir alerta de lixeira cheia:** Quando a porcentagem atingir ou ultrapassar um limite de 80% avisando de que o lixo está cheio, a dashboard deve exibir um alerta visual.

**RF08 – Identificar cada lixeira:** O sistema deve exibir um identificador (id, localização) para cada lixeira cadastrada.

**RF09 – Cadastro de novas lixeiras:** O administrador deverá ser capaz de cadastrar novas lixeiras no sistema, criando uma nova entrada no Firebase com seu respectivo ID.

**RF10 – Exibir lixeiras em um mapa visual:** Na aplicação React deve haver um mapa onde exibe as lixeiras, mostrando a localização de cada uma. (O mapa será apenas visual).

### 4.1 REQUISITOS NÃO FUNCIONAIS

**RNF01 – Disponibilidade:** O sistema deve estar disponível continuamente, garantindo comunicação entre sensores, Firebase e dashboard.

**RNF02 – Baixa latência:** Os dados devem aparecer no dashboard com atraso mínimo.

**RNF03 – Confiabilidade das medições:** As leituras do sensor ultrassônico devem ser estáveis e próximas do valor real.

**RNF04 – Escalabilidade:** A arquitetura deve permitir aumentar facilmente o número de lixeiras cadastradas.

**RNF05 – Responsividade:** A dashboard deve se adaptar bem a diferentes tamanhos de tela.

## 5. DADOS COLETADOS E FORNECIDOS

Os dados coletados pelo sistema são provenientes dos sensores ultrassônicos instalados em cada lixeira. O ESP32 realiza leituras contínuas da distância interna da lixeira em centímetros (cm), identificando o espaço ainda disponível. Esses valores brutos são enviados ao Firebase, onde são interpretados pela aplicação web que irá converter automaticamente essa medida em uma porcentagem de ocupação, permitindo visualizar o nível de enchimento de cada lixeira.

As informações processadas são apresentadas na aplicação web destinada ao administrador municipal. Esse dashboard exibe um mapa visual, mostrando a localização de cada lixeira cadastrada. Quando a porcentagem de enchimento atinge um limite pré-estabelecido (80%), o sistema emite um alerta diretamente na interface administrativa, indicando que a lixeira precisa de coleta, caso chegue em 100% o sistema exibe outro alerta informando que a lixeira está transbordando e precisa de coleta. O administrador pode cadastrar novas lixeiras no sistema, permitindo que o mapa seja atualizado.

## 6. DESENHO DA ARQUITETURA

### 6.1 Camada de Percepção (IoT)

Sensor utilizado: O sensor utilizado é o Sensor Ultrassônico HC-SR04, pois ele é o de melhor custo benefício.

Micro controlador: O micro controlador utilizado é o ESP32 DevKit V1, pois já vem com suporte nativo a Wi-Fi, utilizando um sensor por lixeira conectados ao ESP32 por jumpers.

### 6.2 Camada de Rede (Comunicação)

Como enviará os dados: Protocolo Wi-Fi, escolhemos ele ao invés do LoRaWAN, pois o ESP32 já vem com ele embutido, e não iríamos nos beneficiar de um módulo LoRaWAN em uma Smart City.

Protocolo de mensagem: O protocolo HTTP será usado por nosso ESP32 para enviar as informações captadas pelos sensores diretamente para o nosso Realtime Database no Firebase, enquanto o WebSocket será utilizado por nosso frontend para receber as informações do Firebase, assim, mantendo a UI atualizada sem necessidade de recarregar.

### 6.3 Camada de Processamento (Cloud / Distribuído)

Usaremos o Firebase como plataforma de nuvem, onde será hospedado nosso banco de dados (Realtime Database), escolhemos ele por conta de suas features pré-prontas, como autenticação e um banco de dados em realtime, e por não ser necessário o uso do protocolo MQTT, muito útil para projetos IoT. O ESP32 fará conexão direta com o Firebase, enviando as informações para ele via HTTP. Estas informações serão consumidas por uma aplicação React hospedada na AWS.

### 6.4 Camada de Aplicação (Apresentação)

A informação coletada pelos sensores será consumida por meio de uma aplicação web, que utilizará WebSocket para manter a interface sempre atualizada em tempo real. Dessa forma, sempre que o Firebase receber novos dados enviados pelo ESP32, o frontend refletirá imediatamente o nível de preenchimento das lixeiras, sem necessidade de recarregar a página.

## 7. DIAGRAMA

