

# Sistema de coleta de dados de Estações Meteorológicas



# Sumário

- 1. Introdução**
- 2. Finalidade**
- 3. Objetivo**
- 4. Escopo**
- 5. Visão Geral**
- 6. Visão de Casos de Uso**
- 7. Documentação API**
  - 7.1 Objetivo**
  - 7.2 Tags**
  - 7.3 Endpoints**

## 1. Introdução

Este documento descreve o desenvolvimento do sistema de monitoramento ambiental da Tecsus, voltado para coleta, processamento, armazenamento e visualização de dados meteorológicos em tempo real. O sistema será utilizado por administradores e usuários interessados em dados climáticos, promovendo a conscientização ambiental, apoio à prevenção de desastres naturais e incentivo ao aprendizado escolar.

## 2. Finalidade

A finalidade deste projeto é fornecer uma solução tecnológica escalável, acessível e educativa, que integre sensores ambientais com uma plataforma digital robusta, permitindo:

- Acompanhamento em tempo real de condições meteorológicas.
- Tomada de decisão rápida diante de alertas climáticos.
- Uso pedagógico por escolas do ensino médio.

### 3. Objetivo

O objetivo principal deste projeto é desenvolver uma plataforma moderna e eficiente para o monitoramento ambiental por meio de sensores IoT, possibilitando a coleta, o processamento e a visualização de dados meteorológicos em tempo real. Além disso, o sistema busca promover a educação ambiental, oferecendo materiais didáticos sobre os parâmetros climáticos monitorados, e apoiar ações preventivas relacionadas a desastres naturais por meio de alertas automáticos. Dessa forma, alia-se inovação tecnológica à responsabilidade social e educacional, atendendo tanto às necessidades de gestão ambiental quanto ao incentivo ao aprendizado escolar.

### 4. Escopo

O sistema será desenvolvido para abranger as seguintes funcionalidades:

- Cadastro e gerenciamento de **estações meteorológicas, sensores, parâmetros climáticos, usuários e alertas.**
- **Recepção de dados** em tempo real das estações.
- **Dashboards interativos** com gráficos e relatórios.
- **Geração automática de alertas** com base em parâmetros definidos.
- **Registro de dados históricos** com datalogger.
- **Tutoriais explicativos** sobre conceitos meteorológicos.
- **Montagem física da estação meteorológica** com sensores.

### 5. Visão Geral

O sistema de monitoramento ambiental da Tecsus será composto por uma plataforma web acessível, dividida em dois ambientes principais: o painel administrativo e o portal público. O painel administrativo será destinado aos responsáveis pela gestão das estações e sensores, permitindo o cadastro, edição e monitoramento em tempo real dos dados coletados. Já o portal público será voltado à visualização de informações meteorológicas por qualquer usuário, com foco em acessibilidade e uso educacional. Os dados serão exibidos por meio de dashboards interativos e gráficos atualizados, possibilitando análises detalhadas. A arquitetura adotada será escalável e modular, garantindo flexibilidade para futuras expansões e integrações com outras ferramentas ou sensores ambientais.

### 6. Visão de Casos de Uso

Este sistema será utilizado por dois tipos principais de usuários: administradores e usuários públicos.

#### **Administrador:**

- Gerenciar estações, parâmetros, usuários e alertas.
- Configurar sensores e montar estações.
- Monitorar dados em tempo real e gerar relatórios.
- Gerar alertas e visualizar logs.

#### **Usuário:**

- Acessar tutoriais meteorológicos.
- Visualizar dados ambientais em tempo real.

## 7. Documentação API

### 7.1 Objetivo

O objetivo deste documento tem como informar ao leitor sobre as mecânicas que foram desenvolvidas em nossa aplicação.

### 7.2 Tags

Abaixo estão listadas quais foram as seguintes tags utilizadas:

- **auth**
- **user**
- **station**
- **emailStation**
- **typeAlert**
- **alert**
- **typeParameter**
- **parameter**
- **measure**
- **measureAverage**
- **receiveJson**
- **dashboard**

### 7.3 Endpoints

## Equipe: Sync

---

- /auth/login  
Método: **POST**  
Objetivo: Autenticação de usuário  
**Parâmetros:**  

```
{  
  "email": "user@example.com",  
  "password": "string"  
}
```

  
Response: (200, OK)
- /auth/register  
Método: **POST**  
Objetivo: Registro de um novo usuário  
**Parâmetros:**  

```
{  
  "name": "string",  
  "email": "user@example.com",  
  "role": "admin"  
}
```

  
Response: (200, OK)
- /auth/createpassword  
Método: **POST**  
Objetivo: Registro de um novo usuário  
**Parâmetros:**  

```
{  
  "email": "user@example.com",  
  "newPassword": "string"  
}
```

  
Response: (200, OK)

## Equipe: Sync

---

- /user/update

Método: **PUT**

Objetivo: Atualiza um usuário

**Parâmetros:**

```
{  
  "id": "string",  
  "name": "string",  
  "email": "user@example.com",  
  "role": "admin"  
}
```

Response: (200, OK)

- /user/delete/{id}

Método: **DELETE**

Objetivo: Deleta um usuário

**Parâmetros:** *ID do usuário*

Response: (200, OK)

- /user/list

Método: **GET**

Objetivo: Lista todos os usuários

**Parâmetros:** Nenhum

Response: (200, OK)

- /user/read/{id}

Método: **GET**

Objetivo: Obtém um usuário específico

**Parâmetros:** *ID do usuário*

Response: (200, OK)



## Equipe: Sync

---

- /user/change-password

Método: **PUT**

Objetivo: Altera a senha do usuário

**Parâmetros:**

```
{  
  "currentPassword": "string",  
  "newPassword": "string"  
}
```

Response: (200, OK)

- /station/create

Método: **POST**

Objetivo: Cria uma nova estação

**Parâmetros:**

```
{  
  "name": "string",  
  "location": "string",  
  "description": "string"  
}
```

Response: (200, OK)

## Equipe: Sync

---

- /station/update  
Método: **PUT**  
Objetivo: Atualiza uma estação  
**Parâmetros:**  

```
{  
  "id": "string",  
  "name": "string",  
  "location": "string",  
  "description": "string"  
}
```

  
Response: (200, OK)
- /station/delete/{id}  
Método: **DELETE**  
Objetivo: Deleta uma estação  
**Parâmetros:** *ID da estação*  
Response: (200, OK)
- /station/list  
Método: **GET**  
Objetivo: Lista todas as estações  
**Parâmetros:** Nenhum  
Response: (200, OK)
- /station/read/{id}  
Método: **GET**  
Objetivo: Obtém uma estação específica  
**Parâmetros:** *ID da estação*  
Response: (200, OK)

## Equipe: Sync

---

- /emailStation/create

Método: **POST**

Objetivo: Registra um email para uma estação

**Parâmetros:**

```
{  
  "email": "user@example.com",  
  "stationId": "string"  
}
```

Response: (200, OK)

- /typeAlert

Método: **GET**

Objetivo: Lista todos os tipos de alerta

**Parâmetros:** Nenhum

Response: (200, OK)

- /typeAlert

Método: **POST**

Objetivo: Cria um novo tipo de alerta

**Parâmetros:**

```
{  
  "name": "string",  
  "description": "string",  
  "parameters": [  
    "string"  
  ]  
}
```

Response: (200, OK)

## Equipe: Sync

---

- /typeAlert

Método: **PUT**

Objetivo: Atualiza um tipo de alerta

**Parâmetros:**

```
{  
  "id": "string",  
  "name": "string",  
  "description": "string",  
  "parameters": [  
    "string"  
  ]  
}
```

Response: (200, OK)

- /typeAlert/{id}

Método: **GET**

Objetivo: Obtém um tipo de alerta específico

**Parâmetros:** *ID do tipo de alerta*

Response: (200, OK)

- /typeAlert/{id}

Método: **DELETE**

Objetivo: Remove um tipo de alerta

**Parâmetros:** *ID do tipo de alerta*

Response: (200, OK)

## Equipe: Sync

---

- /alert/create

Método: **POST**

Objetivo: Cria um novo alerta

**Parâmetros:**

```
{  
  "typeAlertId": "string",  
  "stationId": "string",  
  "parameterId": "string",  
  "condition": "greater",  
  "value": 0  
}
```

Response: (200, OK)

- /alert/update

Método: **PUT**

Objetivo: Atualiza um alerta

**Parâmetros:**

```
{  
  "id": "string",  
  "typeAlertId": "string",  
  "stationId": "string",  
  "parameterId": "string",  
  "condition": "greater",  
  "value": 0  
}
```

Response: (200, OK)

## Equipe: Sync

---

- /alert/delete/{id}  
Método: **DELETE**  
Objetivo: Deleta um alerta  
**Parâmetros:** *ID do alerta*  
Response: (200, OK)
- /alert/list  
Método: **GET**  
Objetivo: Lista todos os alertas  
**Parâmetros:** Nenhum  
Response: (200, OK)
- /alert/read/{id}  
Método: **GET**  
Objetivo: Obtém um alerta específico  
**Parâmetros:** *ID do alerta*  
Response: (200, OK)
- /typeParameter/create  
Método: **POST**  
Objetivo: Cria um novo tipo de parâmetro  
**Parâmetros:**  

```
{  
  "name": "string",  
  "description": "string"  
}
```

  
Response: (200, OK)

## Equipe: Sync

---

- /typeParameter/update  
Método: **PUT**  
Objetivo: Atualiza um tipo de parâmetro  
**Parâmetros:**  

```
{  
  "id": "string",  
  "name": "string",  
  "description": "string"  
}
```

  
Response: (200, OK)
- /typeParameter/delete/{id}  
Método: **DELETE**  
Objetivo: Remove um tipo de parâmetro  
**Parâmetros:** *ID do tipo de parâmetro*  
Response: (200, OK)
- /typeParameter/read/{id}  
Método: **GET**  
Objetivo: Obtém um tipo de parâmetro específico  
**Parâmetros:** *ID do tipo de parâmetro*  
Response: (200, OK)
- /typeParameter/list  
Método: **GET**  
Objetivo: Lista de todos os parâmetros  
**Parâmetros:** Nenhum  
Response: (200, OK)

## Equipe: Sync

---

- /parameter/create  
Método: **POST**  
Objetivo: Cria um novo parâmetro  
**Parâmetros:**  

```
{  
  "name": "string",  
  "unit": "string",  
  "typeParameterId": "string"  
}
```

  
Response: (200, OK)
- /parameter/list  
Método: **GET**  
Objetivo: Lista todos os parâmetros  
**Parâmetros:** Nenhum  
Response: (200, OK)
- /parameter/update  
Método: **PUT**  
Objetivo: Atualiza um parâmetro existente  
**Parâmetros:**  

```
{  
  "id": "string",  
  "name": "string",  
  "unit": "string",  
  "typeParameterId": "string"  
}
```

  
Response: (200, OK)



## Equipe: Sync

---

- /parameter/delete/{id}

Método: **DELETE**

Objetivo: Remove um pâmetro

**Parâmetros:** ID do parâmetro

Response: (200, OK)

- /measure/create

Método: **POST**

Objetivo: Cria uma nova medição

**Parâmetros:**

```
{  
  "value": 0,  
  "parameterId": "string",  
  "stationId": "string"  
}
```

Response: (200, OK)

- /measure/update

Método: **PUT**

Objetivo: Atualiza uma medição

**Parâmetros:**

```
{  
  "id": "string",  
  "value": 0,  
  "parameterId": "string",  
  "stationId": "string"  
}
```

Response: (200, OK)

## Equipe: Sync

---

- /measure/delete/{id}  
Método: **DELETE**  
Objetivo: Deleta uma medição  
**Parâmetros:** *ID da medida*  
Response: (200, OK)
- /measure/list  
Método: **GET**  
Objetivo: Lista todas as medições  
**Parâmetros:** Nenhum  
Response: (200, OK)
- /measure/read/{id}  
Método: **GET**  
Objetivo: Obtém uma medição específica  
**Parâmetros:** *ID da medida*  
Response: (200, OK)
- /measureAverage/{stationId}  
Método: **GET**  
Objetivo: Obtém a média das medições de uma estação  
**Parâmetros:** ID da estação  
Response: (200, OK)

## Equipe: Sync

---

- /receiverJson

Método: **POST**

Objetivo: Recebe dados JSON de medições

**Parâmetros:**

```
{  
  "stationId": "string",  
  "measures": [  
    {  
      "parameterId": "string",  
      "value": 0,  
      "timestamp": "2025-04-27T20:27:33.695Z"  
    }  
  ]  
}
```

Response: (200, OK)

- /receiverJson/sync

Método: **POST**

Objetivo: Sincroniza os dados com o mongoDB

**Parâmetros:** Nenhum

Response: (200, OK)

- /dashboard/public

Método: **GET**

Objetivo: Obtém dados públicos do dashboard (últimos 7 dias)

**Parâmetros:** Nenhum

Response: (200, OK)

## Equipe: Sync

---

- /dashboard/list

Método: **GET**

Objetivo: Obtém dados do dashboard (com autenticação)

**Parâmetros:** Nenhum

Response: (200, OK)