T3 - ENG SOFT - PEDRO LUCAS

Atividade T3: Back-end (API + Persistência XML + Validação XSD)

1) Arquitetura e Fluxo de Dados

Este back-end atua como o ponto central de ingestão e consulta. O fluxo principal de dados é o seguinte:

1. **Ingestão (POST):**
    - Um dispositivo IoT (ou o T4 simulando) envia um documento XML (`Content-Type: application/xml`) para o endpoint `POST /api/xml`.
    - O back-end intercepta a requisição.
    - **Validação (Camada 1 - XSD):** O XML é validado contra o `model/estufa.xsd` (da T2). Isso garante a estrutura, tipos de dados, enums e integridade referencial (`xs:ID` / `xs:IDREF`). Se falhar, retorna `400`.
    - **Validação (Camada 2 - Regras de Negócio):** O back-end analisa o XML (agora estruturalmente válido) e aplica regras de negócio (ex: faixas de valores de sensores). Se falhar, retorna `400`.
    - **Persistência:** Se ambas as validações passarem, o back-end gera um ID único (ex: `leitura_UUID.xml`) e salva o documento XML *original* no diretório `backend/data/`.
    - **Resposta:** O back-end retorna `201 Created` com o ID do arquivo salvo.
2. **Consulta (GET):**
    - O T4 (Front-end) precisa de dados para os dashboards.
    - `GET /api/xml/:id`: Retorna o XML bruto específico (ex: para auditoria).
    - `GET /api/consulta`: O back-end lê *todos* os XMLs em `backend/data/`, aplica os filtros de query (ex: `sensorId`, `dataInicio`), agrega os resultados e retorna um *JSON* consolidado, pronto para o T4 renderizar gráficos.

---

2) Validação XSD e Regras de Negócio

A validação é um processo de duas etapas, crucial para a integridade dos dados.

2.1 Validação XSD (Estrutural)

- **Arquivo:** `backend/model/estufa.xsd` (o mesmo definido na T2).
- **O que valida:**
  - Estrutura hierárquica (`estufa > sensores > sensor`).
  - Tipos de dados (ex: `dataHora` é um `xs:dateTime` válido).
  - Obrigatoriedade (ex: `@id` do sensor é `use="required"`).
  - Enumerações (ex: `@tipo` deve ser 'temperatura', 'umidadeAr', etc.).
  - Integridade Referencial (ex: `leitura/@sensorRef` deve apontar para um `sensor/@id` existente *no mesmo documento*).
- **Erro (Exemplo XSD):**JSON

```
{
 "code": "XSD_VALIDATION_ERROR",
 "message": "XML falhou na validação XSD.",
 "details": "Element '{http://www.w3.org/2001/XMLSchema}leitura', attribute 'sensorRef': 'id_inexistente' is not a valid value of the atomic type 'xs:IDREF'."
}
```

2.2 Validação de Regras de Negócio (Semântica)

- **Lógica:** Implementada no código da API (ex: `api.py`). Ocorre *após* a validação XSD.
- **O que valida:**
  - **Faixas de Valor (T2):**
    - `temperatura`: [-10.0, 60.0] °C
    - `umidadeAr` / `umidadeSolo`: [0.0, 100.0] %
    - `luminosidade`: [0, 200000] lux
- **Erro (Exemplo Regra de Negócio):**JSON

```
{
 "code": "BUSINESS_RULE_ERROR",
 "message": "Valor '99.0' para o sensor 'tNorte' (tipo: temperatura) está fora da faixa permitida [-10.0, 60.0].",
 "xpath": "/estufa/leituras/leitura[1]/valor"
}
```

---

3) Endpoints REST (Documentação)

POST /api/xml

- **Descrição:** Recebe, valida e persiste um novo documento XML da estufa.
- **Request Body:** `application/xml` (O XML bruto `dadosEstufa.xml` da T2).
- **Resposta (Sucesso 201 Created):**JSON

```
{
 "id": "leitura_a8b3e1c0.xml",
 "message": "XML válido e armazenado com sucesso."
}
* **Resposta (Erro 400 Bad Request):**
* Retorna o JSON de erro (conforme Seção 2) se a validação XSD ou de Regra de Negócio falhar.
```

GET /api/xml/:id

- **Descrição:** Recupera um documento XML específico que foi persistido, usando o ID retornado pelo POST.
- **Parâmetro de Path:** `id` (ex: `leitura_a8b3e1c0.xml`).
- **Resposta (Sucesso 200 OK):**
  - Retorna o XML bruto (ex: `Content-Type: application/xml`).
- **Resposta (Erro 404 Not Found):**JSON

```
{
 "code": "NOT_FOUND",
 "message": "O arquivo com ID 'id_nao_existe.xml' não foi encontrado."
}
```

GET /api/consulta

- **Descrição:** Consulta e agrega *todas* as leituras de *todos* os arquivos XML persistidos, aplicando filtros. Este é o endpoint principal para o T4 (Dashboard).
- **Query Params:**
  - `sensorId` (string, ex: "tNorte"): Filtra leituras por um ID de sensor.
  - `dataInicio` (string, format: `date-time`): Data/hora inicial (ISO 8601).
  - `dataFim` (string, format: `date-time`): Data/hora final (ISO 8601).
- **Resposta (Sucesso 200 OK):**JSON

```
{
 "totalResultados": 2,
 "filtrosAplicados": {
```

```
    "sensorId": "tNorte"
  },
  "leituras": [
    {
      "sensorId": "tNorte",
      "tipo": "temperatura",
      "unidade": "C",
      "localizacao": "Parede Norte, 1.5m altura",
      "dataHora": "2025-10-30T10:30:00-03:00",
      "valor": 24.5,
      "arquivoOrigem": "leitura_a8b3e1c0.xml"
    },
    {
      "sensorId": "tNorte",
      "tipo": "temperatura",
      "unidade": "C",
      "localizacao": "Parede Norte, 1.5m altura",
      "dataHora": "2025-10-30T10:35:00-03:00",
      "valor": 24.7,
      "arquivoOrigem": "leitura_a8b3e1c0.xml"
    }
  ]
}
```


4) Estrutura de Diretórios

A estrutura de diretórios implementada foi:

```
meu-projeto-t3/
├── backend/
│   ├── app/
│   │   └── api.py          # Lógica da API (Flask) e validação
│   ├── data/
│   │   └── (XMLs salvos aqui) # Ex: leitura_a8b3e1c0.xml
│   └── model/
│       └── estufa.xsd      # O Schema (da T2) usado para validação
│
├── test_data/
│   ├── valido.xml          # XML para teste de sucesso
│   └── invalido.xml        # XML para teste de falha (regra)
│
└── venv/                   # Ambiente virtual Python
```

5) Documentação da API (OpenAPI)

```
openapi: 3.0.3
info:
  title: API Estufa de Tomates IoT (T3)
  description: API para receber, validar e consultar dados de sensores da e
stufa no formato XML.
  version: "1.0.0"
servers:
  - url: http://localhost:5000
    description: Servidor de Desenvolvimento

paths:
  /api/xml:
    post:
      summary: Envia dados de leitura da estufa
      description: Recebe, valida (XSD + Regras) e persiste um documento X
ML de leituras.
      requestBody:
        required: true
        content:
          application/xml:
            schema:
              type: string
              format: xml
            example: |
              <?xml version="1.0" encoding="UTF-8"?>
              <estufa id="estufaPrincipal" ...>
                <!-- ... (XML completo da T2) ... -->
              </estufa>
      responses:
        '201':
          description: XML válido e armazenado com sucesso.
          content:
            application/json:
              schema:
                type: object
                properties:
```

```
              id: { type: string, example: "leitura_a8b3e1c0.xml" }
              message: { type: string, example: "XML válido e armazenado co
m sucesso." }
      '400':
        description: Erro de validação (XSD ou Regra de Negócio).
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/ErroValidacao'

  /api/xml/{id}:
    get:
      summary: Recupera um XML persistido
      description: Retorna o conteúdo de um arquivo XML específico salvo an
teriormente, pelo seu ID.
      parameters:
        - name: id
          in: path
          required: true
          description: O ID do arquivo XML (retornado pelo POST).
          schema:
            type: string
            example: "leitura_a8b3e1c0.xml"
      responses:
        '200':
          description: Sucesso. Retorna application/xml por padrão.
          content:
            application/xml:
              schema:
                type: string
                format: xml
        '404':
          description: Recurso não encontrado.
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ErroPadrao'
```

```
  /api/consulta:
    get:
      summary: Consulta leituras agregadas
      description: Consulta e filtra dados de leituras de todos os XMLs persis
tidos. Retorna sempre JSON.
      parameters:
        - name: sensorId
          in: query
          description: Filtra por um ID de sensor específico.
          schema:
            type: string
            example: "tNorte"
        - name: dataInicio
          in: query
          description: Data/hora inicial (ISO 8601).
          schema:
            type: string
            format: date-time
        - name: dataFim
          in: query
          description: Data/hora final (ISO 8601).
          schema:
            type: string
            format: date-time
      responses:
        '200':
          description: Sucesso na consulta.
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/ResultadoConsulta'

components:
  schemas:
    ErroValidacao:
      type: object
      properties:
        code:
```

```
          type: string
          example: "BUSINESS_RULE_ERROR"
        message:
          type: string
          example: "Valor '99.0' para o sensor de temperatura está fora da faix
a permitida [-10.0, 60.0]."
        xpath:
          type: string
          example: "/estufa/leituras/leitura[1]/valor"
        idReferencia:
          type: string
          example: "tNorte"

    ErroPadrao:
      type: object
      properties:
        code:
          type: string
          example: "NOT_FOUND"
        message:
          type: string
          example: "O arquivo com ID 'id_nao_existe.xml' não foi encontrado."

    LeituraAgregada:
      type: object
      properties:
        sensorId: { type: string, example: "tNorte" }
        tipo: { type: string, example: "temperatura" }
        unidade: { type: string, example: "C" }
        localizacao: { type: string, example: "Parede Norte, 1.5m altura" }
        dataHora: { type: string, format: date-time, example: "2025-10-30T10:
30:00-03:00" }
        valor: { type: number, example: 24.5 }
        arquivoOrigem: { type: string, example: "leitura_a8b3e1c0.xml" }

    ResultadoConsulta:
      type: object
      properties:
```

```
      totalResultados:
        type: integer
        example: 1
      filtrosAplicados:
        type: object
        example: { "sensorId": "tNorte" }
      leituras:
        type: array
        items:
          $ref: '#/components/schemas/LeituraAgregada'
```

6) Exemplos de Arquivos XML Persistidos

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  ID: leitura_a8b3e1c0.xml
  Recebido em: 2025-10-30T10:35:01-03:00
  Origem: Device_Estufa_01
-->
<estufa id="estufaPrincipal"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="../model/estufa.xsd">

    <sensores>
        <sensor id="tNorte" tipo="temperatura" unidade="C">
            <localizacao>Parede Norte, 1.5m altura</localizacao>
        </sensor>
        <sensor id="tSul" tipo="temperatura" unidade="C">
            <localizacao>Parede Sul, 1.5m altura</localizacao>
        </sensor>
        <sensor id="uArCentro" tipo="umidadeAr" unidade="%">
            <localizacao>Centro da estufa (pendurado)</localizacao>
        </sensor>
        <sensor id="uSoloB1" tipo="umidadeSolo" unidade="%">
            <localizacao>Bancada 1, Vaso 3</localizacao>
        </sensor>
        <sensor id="luzTeto" tipo="luminosidade" unidade="lux">
```

```
        <localizacao>Acima da Bancada 1</localizacao>
      </sensor>
    </sensores>

    <leituras>
      <!-- Pacote de leituras das 10:30 -->
      <leitura sensorRef="tNorte">
        <dataHora>2025-10-30T10:30:00-03:00</dataHora>
        <valor>24.5</valor>
      </leitura>
      <leitura sensorRef="tSul">
        <dataHora>2025-10-30T10:30:00-03:00</dataHora>
        <valor>25.1</valor>
      </leitura>
      <leitura sensorRef="uArCentro">
        <dataHora>2025-10-30T10:30:00-03:00</dataHora>
        <valor>78.0</valor>
      </leitura>

      <!-- Pacote de leituras das 10:35 -->
      <leitura sensorRef="tNorte">
        <dataHora>2025-10-30T10:35:00-03:00</dataHora>
        <valor>24.7</valor>
      </leitura>
    </leituras>

</estufa>
```

7) Testes (Descrição dos Cenários)

Os seguintes testes foram executados via `curl` para validar o back-end:

- ✅ **Ping (Sanidade):**
  - `GET /api/ping`
  - **Resultado:** `200 OK` com a mensagem "Pong!".
- ✅ **XML Válido (Happy Path):**
  - `POST /api/xml` com o `test_data/valido.xml`.

    - **Resultado:** `HTTP 201 Created`. JSON de sucesso (`{"id": ..., "message": ...}`). Arquivo `.xml` criado em `backend/data/`.
- ❌ **XSD Inválido (Sintaxe/Estrutura):**
    - `POST /api/xml` com XML faltando um atributo obrigatório (ex: `sensorRef`).
    - **Resultado:** `HTTP 400 Bad Request`. JSON de erro `{ "code": "XSD_VALIDATION_ERROR", ... }`.
- ❌ **Regra de Negócio Inválida (Faixa):**
    - `POST /api/xml` com `test_data/invalido.xml` (temperatura = 99.0).
    - **Resultado:** `HTTP 400 Bad Request`. JSON de erro `{ "code": "BUSINESS_RULE_ERROR", "message": "Valor '99.0' está fora da faixa...", ... }`.
- ✅ **Consulta de Arquivo (GET ID):**
    - `GET /api/xml/{id_retornado_do_teste_valido}`
    - **Resultado:** `HTTP 200 OK` com o conteúdo `application/xml` do arquivo.
- ✅ **Consulta Filtrada (GET Consulta):**
    - `GET /api/consulta?sensorId=tNorte`
    - **Resultado:** `HTTP 200 OK`. JSON com `totalResultados: ...` e a lista de leituras apenas do sensor `tNorte`.

---

8) Execução/Deploy (Notas)

- **Stack:** Python 3, Flask (Framework Web), LXML (Validação XSD).
- **Dependências:** `pip install Flask lxml`
- **Variáveis de Ambiente (implícito):** Os caminhos para `DATA_PATH` (`backend/data/`) e `SCHEMA_PATH` (`backend/model/estufa.xsd`) estão definidos no `api.py`.
- **Execução (Desenvolvimento):**Bash

  ```
  # (Ativar o venv)
  source venv/bin/activate
  # Rodar o servidor
  python backend/app/api.py
  ```

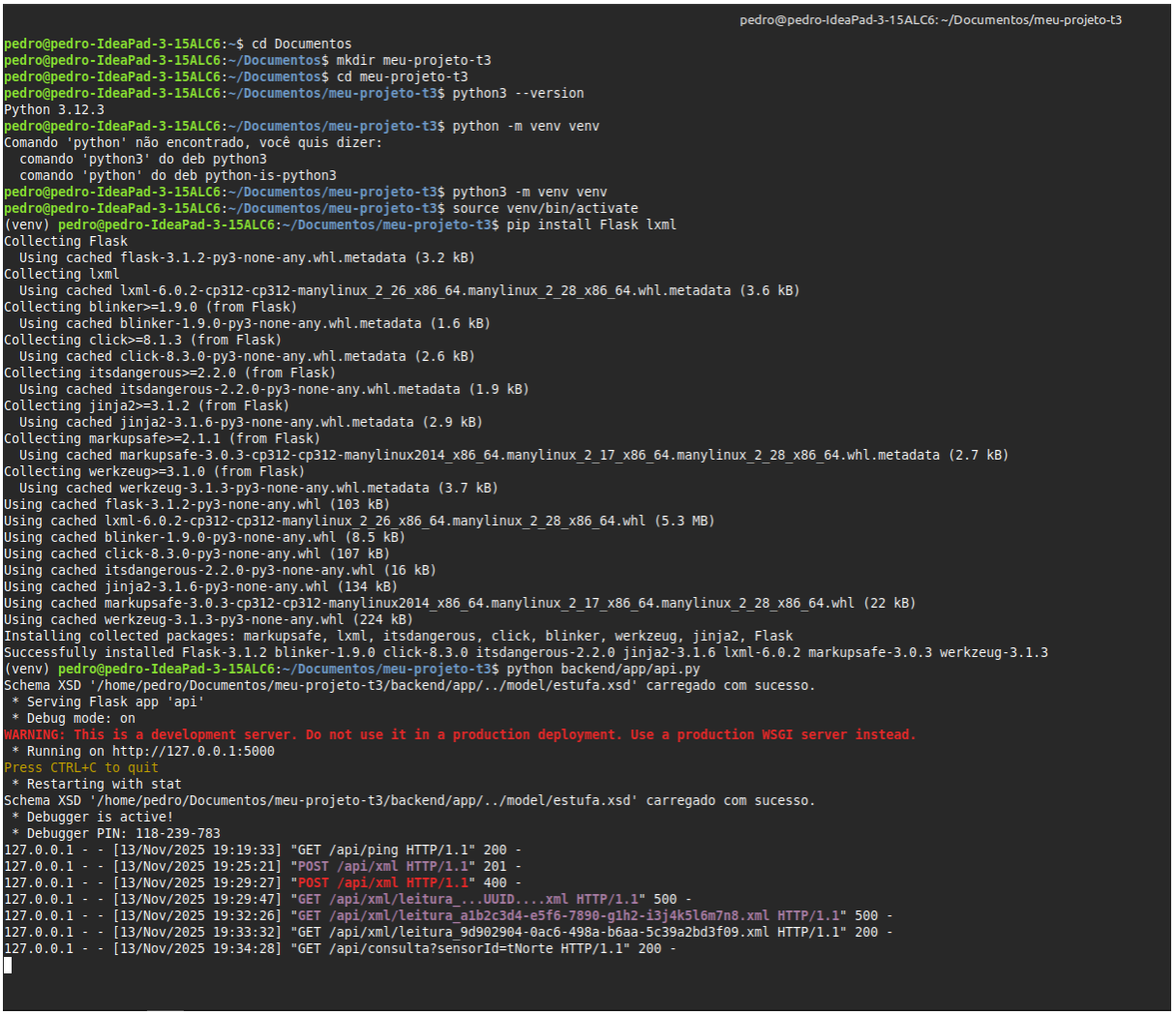
---

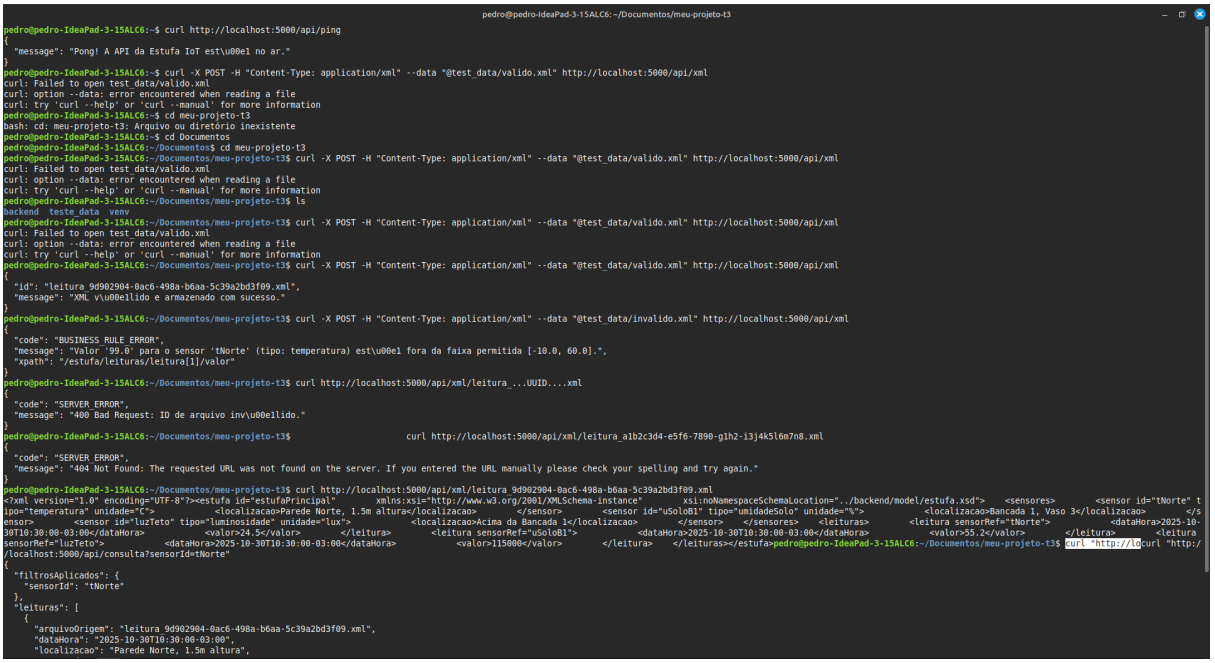
9) Checklist

- ☑ ~~Endpoints documentados (Seção 3)~~

- [x] ~~Padrões de erro definidos (Seção 2)~~
- [x] ~~Validação antes da persistência (XSD + Regras de Negócio)~~
- [x] ~~Exemplos de XML armazenados (Seção 6)~~
- [x] ~~Testes descritos (Seção 7)~~
- [x] ~~OpenAPI colado (Seção 5)~~

Prints dos testes no terminal:




pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3
pedro@pedro-IdeaPad-3-15ALC6:~$ curl http://localhost:5000/api/ping
{
"message": "Pong! A API da Estufa IoT est\u00e1 no ar."
}
pedro@pedro-IdeaPad-3-15ALC6:~$ curl -X POST -H "Content-Type: application/xml" --data "@test_data/valido.xml" http://localhost:5000/api/xml
curl: Failed to open test_data/valido.xml
curl: option --data: error encountered when reading a file
curl: try 'curl --help' or 'curl --manual' for more information
pedro@pedro-IdeaPad-3-15ALC6:~$ cd meu-projeto-t3
bash: cd: meu-projeto-t3: Arquivo ou diretório inexistente
pedro@pedro-IdeaPad-3-15ALC6:~$ cd Documentos
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos$ cd meu-projeto-t3
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl -X POST -H "Content-Type: application/xml" --data "@test_data/valido.xml" http://localhost:5000/api/xml
curl: Failed to open test_data/valido.xml
curl: option --data: error encountered when reading a file
curl: try 'curl --help' or 'curl --manual' for more information
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ ls
backend teste_data venv
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl -X POST -H "Content-Type: application/xml" --data "@test_data/valido.xml" http://localhost:5000/api/xml
curl: Failed to open test_data/valido.xml
curl: option --data: error encountered when reading a file
curl: try 'curl --help' or 'curl --manual' for more information
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl -X POST -H "Content-Type: application/xml" --data "@test_data/valido.xml" http://localhost:5000/api/xml
{
"id": "leitura_9d902904-0ac6-498a-b6aa-5c39a2bd3f09.xml",
"message": "XML v\u00e1lido e armazenado com sucesso."
}
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl -X POST -H "Content-Type: application/xml" --data "@test_data/invalido.xml" http://localhost:5000/api/xml
{
"code": "BUSINESS_RULE_ERROR",
"message": "Valor '99.0' para o sensor 'tNorte' (tipo: temperatura) est\u00e1 fora da faixa permitida [-10.0, 60.0].",
"xpath": "/estufa/leituras/leitura[1]/valor"
}
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl http://localhost:5000/api/xml/leitura_...UUID....xml
{
"code": "SERVER_ERROR",
"message": "400 Bad Request: ID de arquivo inv\u00e1lido."
}
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl http://localhost:5000/api/xml/leitura_a1b2c3d4-e5f6-7890-g1h2-i3j4k5l6m7n8.xml
{
"code": "SERVER_ERROR",
"message": "404 Not Found: The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again."
}
pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl http://localhost:5000/api/xml/leitura_9d902904-0ac6-498a-b6aa-5c39a2bd3f09.xml
<?xml version="1.0" encoding="UTF-8"?><estufa id="estufaPrincipal" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="../backend/model/estufa.xsd"> <sensores> <sensor id="tNorte" t
ipo="temperatura" unidade="C"> <localizacao>Parede Norte, 1.5m altura</localizacao> </sensor> <sensor id="uSoloB1" tipo="umidadeSolo" unidade="%"> <localizacao>Bancada 1, Vaso 3</localizacao> </s
ensor> <sensor id="luzTeto" tipo="luminosidade" unidade="lux"> <localizacao>Acima da Bancada 1</localizacao> </sensor> </sensores> <leituras> <leitura sensorRef="tNorte"> <dataHora>2025-10-
30T10:30:00-03:00</dataHora> <valor>24.5</valor> </leitura> <leitura sensorRef="uSoloB1"> <dataHora>2025-10-30T10:30:00-03:00</dataHora> <valor>55.2</valor> </leitura> <leitura
sensorRef="luzTeto"> <dataHora>2025-10-30T10:30:00-03:00</dataHora> <valor>115000</valor> </leitura> </leituras></estufa>pedro@pedro-IdeaPad-3-15ALC6:~/Documentos/meu-projeto-t3$ curl "http://localhost:5000/api/consulta?sensorId=tNorte"
{
"filtrosAplicados": {
"sensorId": "tNorte"
},
"leituras": [
{
"arquivoOrigem": "leitura_9d902904-0ac6-498a-b6aa-5c39a2bd3f09.xml",
"dataHora": "2025-10-30T10:30:00-03:00",
"localizacao": "Parede Norte, 1.5m altura",