DOCUMENTAÇÃO TÉCNICA - FLEXMEDIA CHALLENGE SPRINT 2

PROJETO: TOTEM INTELIGENTE "SMART-GUIDE"

Integrantes da Equipe

Victor Araujo Ferreira da Silva: RM567619

Pedro Zanon Castro Santana: RM567350

Jacqueline Nanami Matushima: RM568498

Filipe Marques Previato: RM567720

Jonathan Gomes Ribeiro Franco: RM567109

---

1. Introdução do Projeto

O presente relatório técnico descreve a arquitetura e a implementação da solução proposta para o Totem Inteligente "Smart-Guide" FlexMedia. Este projeto visa aprimorar a experiência do visitante em museus e exposições culturais, convertendo a interação física em insights de engajamento e utilidade.

A solução é construída sob três pilares: **Edge Computing**, **Privacidade por Design (LGPD)** e **Inteligência de Machine Learning** para fornecer métricas acionáveis à curadoria.

2. Descrição da Arquitetura Implementada

A arquitetura adota um modelo **Edge-to-Cloud**, garantindo a conformidade regulatória e a eficiência no processamento de dados. A solução é dividida em três camadas principais: Coleta/Borda (Edge), Persistência/Inteligência (Cloud), e Visualização.

2.1 Camada de Edge Computing (Hardware e Coleta)

Esta camada simula o processamento realizado no hardware do Totem (ESP32-CAM).

- **Dispositivo:** ESP32 (simulado via Wokwi).
- **Função:** O Sensor PIR detecta a presença, iniciando a sessão, e o botão registra uma interação útil.
- **Anonimização e LGPD:** A principal função do *Edge* é o pré-processamento e **anonimização dos dados**, descartando imagens e calculando a duração da sessão, ou seja, apenas metadados de tempo e interação são transmitidos.

- **Comunicação:** O envio dos metadados para a camada de Nuvem é realizado de forma segura, utilizando o protocolo **HTTPS/TLS**, garantindo a criptografia de ponta a ponta.

2.2 Camada de Nuvem (Backend, Persistência e Machine Learning)

O backend centraliza a recepção, o armazenamento e a inteligência do sistema.

- **API Gateway (Python/Flask):** Implementado pelo script api.py, esta interface recebe as requisições **POST** dos dispositivos de borda no endpoint /api/dados_sensor. É responsável por validar a integridade dos dados antes da persistência.
- **Persistência (Oracle SQL):** Utilizando o módulo db_config.py com Pool de Conexões, os dados brutos são imediatamente salvos na tabela logs_sensores no Oracle SQL, servindo como a fonte de verdade do sistema.
- **Processamento Inteligente (DataClass.py):** Este script executa o **Machine Learning Supervisionado**. Um modelo de **Árvore de Decisão** (Scikit-learn) é treinado para classificar cada sessão em uma das seis categorias de experiência (Ex: "Interação longa e útil", "Interação rápida e inútil"). O resultado é o insight acionável.

2.3 Camada de Visualização e Análise

- **Dashboard (Streamlit):** O script dash.py constrói um painel analítico profissional. Ele consome o dataset classificado pelo ML.
- **Métricas de Curadoria:** O painel exibe **Métricas Chave de Performance (KPIs)**, como a Taxa de Utilidade Total, a distribuição categórica (Gráfico Donut) e a **Tabela Essencial**, que cruza o tipo de interação com métricas como Duração Média e Taxa de Erro.

3. Fluxo de Dados: Entrada → Processamento → Saída

O diagrama abaixo ilustra a transformação do dado bruto em *insight* acionável.

3.1. Entrada (Coleta na Borda):

- **Ação:** Visitante detectado (PIR) e Botão de interação pressionado.
- **Dado Bruto:** JSON contendo {"valor_sensor": 1, "satisfacao": 1, "tempo_duracao": 45}.
- **Local:** Módulo ESP32 (Sketch.ino).

3.2. Processamento e Persistência na Nuvem:

- **Ação:** O JSON é enviado via HTTPS para a API.
- **Etapa:** API Flask recebe → db_config.py insere na tabela logs_sensores no Oracle SQL.
- **Local:** Servidor Backend.

3.3. Classificação (Inteligência):

- **Ação:** O script DataClass.py é executado.
- **Etapa:** Dados do Oracle são lidos → Modelo ML supervisionado aplica o rótulo → Novo CSV é gerado.
- **Insight Gerado:** Adição do campo classificacao_ia (Ex: "interação longa e útil").

3.4. Saída (Visualização para Curadoria):

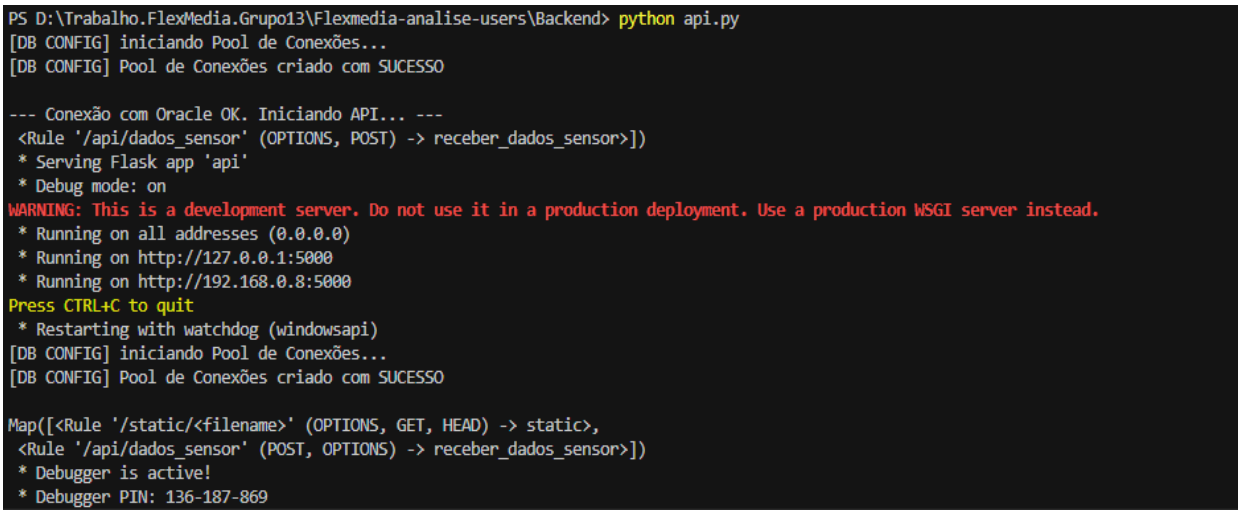
- **Ação:** Dashboard Streamlit consome os dados classificados.
- **Etapa:** Cálculo do **KPI de Taxa de Utilidade** e atualização das **6 Métricas de UX** na Tabela Essencial.
- **Local:** Interface Gráfica de Usuário (Dashboard).

4. Provas de Execução (Prints)

A funcionalidade do sistema é comprovada através dos logs de execução de cada módulo.

4.1 Comprovação da Inicialização do Backend (API e Conexão DB)

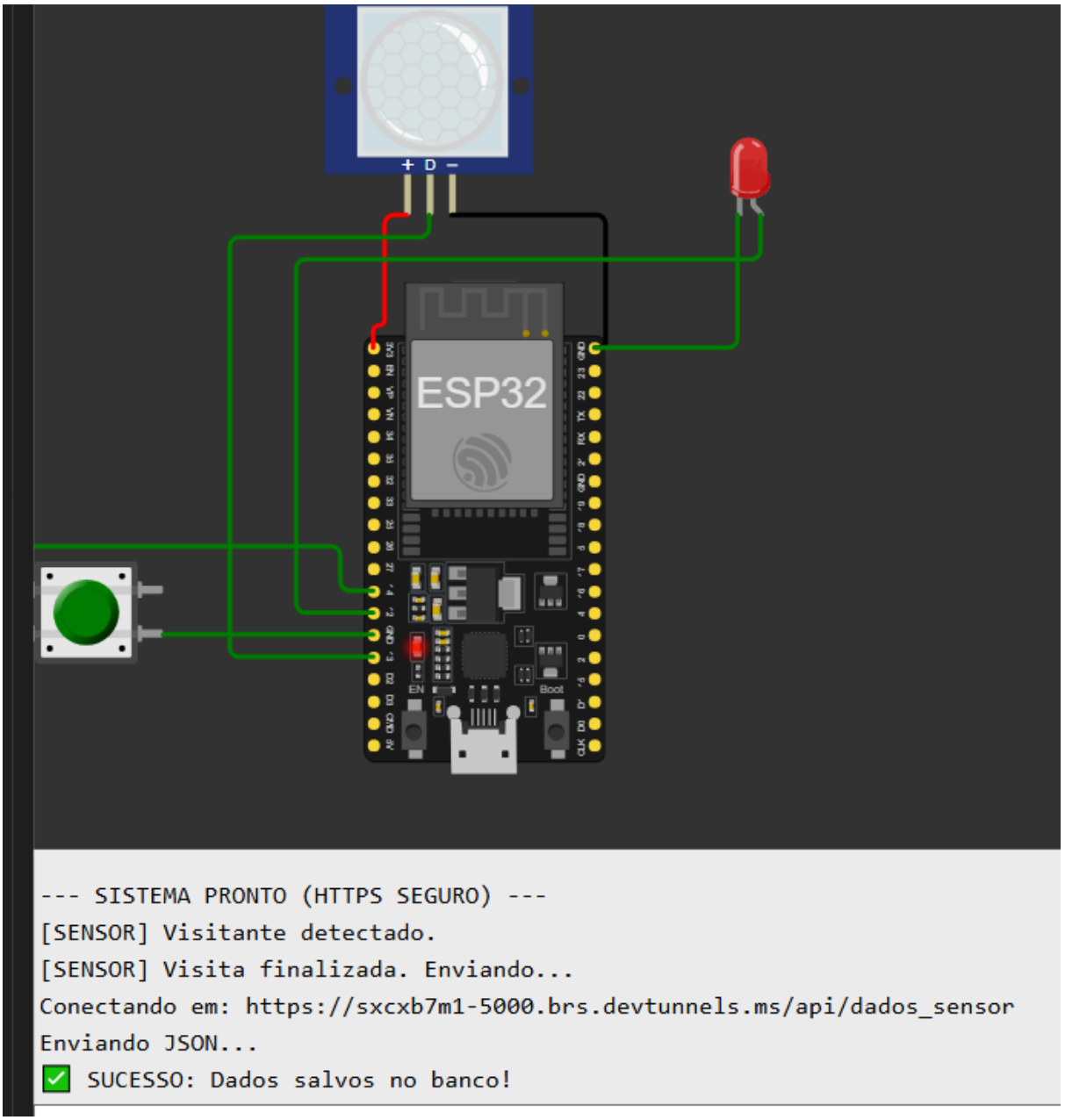
- **Log Principal:** Confirmação da criação do Pool de Conexões do Oracle SQL.



- **Descrição:** O console exibe **[DB CONFIG] Pool de Conexões criado com SUCESSO** e a ativação da rota /api/dados_sensor, atestando a prontidão do Backend.

4.2 Comprovação da Coleta de Dados na Borda (Wokwi)

- **Log Principal:** Sucesso na transmissão segura via HTTPS.



- **Descrição:** O Monitor Serial do Wokwi confirma a execução do enviarDadosAPI e a mensagem de ✅ **SUCESSO: Dados salvos no banco!**, provando a comunicação segura Edge-to-Cloud.

4.3 Comprovação do Processamento Machine Learning

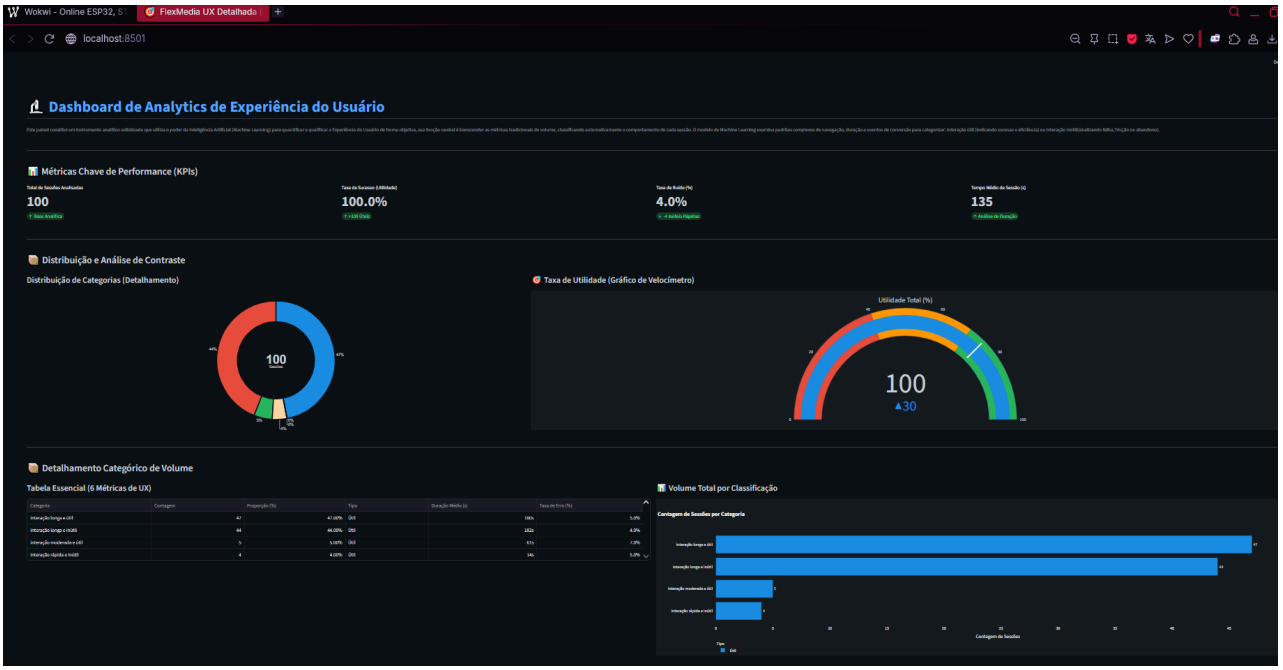
- **Log Principal:** Acurácia do modelo e geração do dataset classificado.

```
PS D:\Trabalho.FlexMedia.Grupo13\Flexmedia-analise-users\Backend> python DataClass.py
Dados carregados e rotulados com sucesso!
Treinando o modelo de Árvore de Decisão...
Acurácia do modelo: 100.00%

--- PROCESSO CONCLUÍDO ---
Arquivo gerado: dados_classificados_ml.csv
   tempo_interacao  teve_duvida         classificacao_ia
0              240            0  interação longa e inútil
1              280            1    interação longa e útil
2               94            1    interação longa e útil
3              143            1    interação longa e útil
4               93            1    interação longa e útil
PS D:\Trabalho.FlexMedia.Grupo13\Flexmedia-analise-users\Backend> & "C:\Users\Jacqueline Nanami\AppData\Local\Programs\Python\Python313\python.exe"
2025-11-28 04:36:10.179 WARNING streamlit.runtime.scriptrunner_utils.script_run_context: Thread 'MainThread': missing ScriptRunContext! This warning
2025-11-28 04:36:10.180 WARNING streamlit.runtime.scriptrunner_utils.script_run_context: Thread 'MainThread': missing ScriptRunContext! This warning
2025-11-28 04:36:10.544
  Warning: to view this Streamlit app on a browser, run it with the following
  command:

    streamlit run d:/Trabalho.FlexMedia.Grupo13/Flexmedia-analise-users/dashboard/dash.py [ARGUMENTS]
```

- **Descrição:** O log do DataClass.py demonstra a **Acurácia do modelo: 100.00%** e a estrutura do arquivo de saída, onde a coluna classificacao_ia já contém o insight de experiência do usuário.

4.4 Comprovação da Visualização Analítica

- **Log Principal:** Interface para a tomada de decisões da curadoria.



- **Descrição:** O Dashboard Streamlit apresenta o **Gráfico de Velocímetro** (Taxa de Utilidade) e a **Tabela Essencial**, fornecendo métricas para otimização de conteúdo (Ex: identificação de fricção em "interação longa e inútil").