Documentação e Relatório

Documento de Estratégia de Testes

Objetivos dos Testes

O objetivo dos testes é garantir a **qualidade e robustez** da aplicação Flask desenvolvida, validando:

- O correto funcionamento das rotas da API.
- A integração com o banco de dados MongoDB.
- O tratamento adequado de dados válidos e inválidos.
- A resiliência da aplicação a falhas de conexão com serviços externos (ex: banco de dados).

Escopo e Funcionalidades Testadas

As funcionalidades cobertas pelos testes foram:

- **Conexão com o MongoDB**:
  - Simulação de falha de conexão para validar o tratamento de exceções.
- **Rotas da Aplicação**:
  - `/`: rota raiz.
  - `/dashboard`: dashboard de visualização.
  - `/dashboard_data`: dados provenientes do MongoDB para exibição no dashboard.
  - `/predict`: rota de predição via Machine Learning, com testes para:
    - Dados válidos.
    - Dados inválidos (menos features que o modelo espera).

Critérios de Aceitação

- Todas as rotas devem responder com status HTTP **200 OK** quando chamadas com parâmetros válidos.
- A rota `/predict` deve:
  - Retornar status **200** com predição e salvar no banco de dados quando os dados forem válidos.
  - Retornar status **400** quando os dados forem incompletos ou inválidos.

- A simulação de falha de conexão com o MongoDB deve ser detectada e tratada corretamente.

Ferramentas Utilizadas

- **Linguagem:** Python 3.12
- **Framework de Testes:** `unittest`
- **Banco de Dados:** MongoDB (local)
- **Bibliotecas auxiliares:** `pymongo`, `pandas`, `joblib` (para modelo ML), `Flask`, `logging`
- **Ambiente:** Windows 11, execução local via terminal

Responsáveis

- **Desenvolvedor e responsável pelos testes:** João Victor, Cauã Romero, Tanus, Rodrigo, Caio
- **Data dos testes:** Abril de **2025**

Relatório dos Testes

Plano de Testes

ID	Objetivo do Teste	Tipo	Resultado Esperado
T1	Verificar resposta da rota `/`	Unitário	Status 200 OK
T2	Verificar resposta da rota `/dashboard`	Unitário	Status 200 OK
T3	Verificar resposta da rota `/dashboard_data`	Unitário	Retornar JSON com dados
T4	Testar `/predict` com dados válidos	Integração	Status 200, predição e gravação no MongoDB
T5	Testar `/predict` com dados inválidos	Unitário	Status 400 com mensagem de erro
T6	Simular falha na conexão com MongoDB	Unitário	Detectar e registrar falha

Caso de testes desenvolvidos

Caso	Descrição	Entrada	Resultado Esperado
CT01	Acessar rota `/`	GET	HTTP 200

Caso	Descrição	Entrada	Resultado Esperado
CT02	Acessar rota `/dashboard`	GET	HTTP 200
CT03	Acessar rota `/dashboard_data`	GET	JSON com dados do banco
CT04	Predição com dados válidos	JSON com 12 features	Predição retornada + inserção no banco
CT05	Predição com dados inválidos	JSON com 2 features	Erro HTTP 400
CT06	Conexão MongoDB inválida	URI incorreta	Falha simulada corretamente

Resultados dos testes executados

Caso	Resultado	Observações
CT01	✅ OK	Página inicial carregada
CT02	✅ OK	Dashboard acessado
CT03	✅ OK	Dados retornados do MongoDB
CT04	✅ OK	Predição gerada e armazenada com sucesso
CT05	✅ OK	Erro 400 retornado como esperado
CT06	✅ OK	Falha na conexão tratada com sucesso

Resumo de falhas

Falha	Descrição	Correção Sugerida
ResourceWarning	Arquivo `index.html` não foi fechado corretamente durante o teste	Utilizar `with open()` para garantir o fechamento do arquivo