Relatório de Erros – Tarefa 2.4: Testes Manuais

Projeto: Sistema de Concessão de Crédito com Regras de Negócio
Autores: Camila Cardoso Rehbein
Data: 06/05/2025

Objetivo

Realizar testes manuais na API desenvolvida para simular decisões de crédito com base em regras de negócio, utilizando 10 casos fictícios no Postman.

Etapas Executadas

1. Início do servidor Flask com o comando python app.py.

2. Envio de requisições via Postman para a rota /predict usando o método POST com um corpo JSON.

3. Análise das respostas da API, que deveriam retornar {"resultado": "Aprovado"} ou {"resultado": "Reprovado"}.

Erros Encontrados

1. Erro 500 - Internal Server Error

- Descrição: A API retornava erro 500 ao receber os dados via Postman.

- Causa: A variável resultado não estava definida antes de ser retornada.

- Solução: Corrigir a lógica da função para garantir que resultado seja definido antes do return.

---

2. Resposta com chave incorreta

- Descrição: A API retornava {"inadimplente": 0} em vez de {"resultado": "Aprovado"}.
- Causa: O dicionário retornado usava a chave errada.
- Solução: Ajustar o return jsonify({"resultado": resultado}) para que a resposta siga o padrão esperado.

---

3. Porta 5000 em uso

- Descrição: Ao tentar iniciar o Flask, recebemos Address already in use.
- Causa: O servidor Flask já estava em execução em segundo plano.
- Solução: Encerrar o processo ou reiniciar o ambiente. Alternativamente, usar outra porta com app.run(port=5001).

---

4. Dashboard HTML não renderizando

- Descrição: Ao abrir o dashboard no navegador, apareciam apenas os códigos e não o layout.
- Causa: O arquivo estava sendo salvo como .html, mas com conteúdo de código Python ou sem encoding apropriado.
- Solução: Garantir que o HTML seja salvo corretamente como dashboard.html e aberto diretamente no navegador.

---

5. Ambiente instável no Jupyter Notebook

- Descrição: Durante testes iniciais no Colab, diversas etapas não funcionavam (como curl ou Flask simultâneo).

- Solução: Migração para ambiente local com Visual Studio Code, Postman e Terminal no macOS. Após isso, todos os testes foram bem-sucedidos.

Conclusão

Apesar das dificuldades técnicas iniciais, os testes manuais foram concluídos com sucesso após os devidos ajustes. A API se mostrou funcional, retornando os resultados esperados com base nas regras de negócio implementadas.