📘 Pipeline de Cotações Cambiais com Python + LLM

Alexandre Roberto - 1520357

bianca pereira de lima lourenço - 2301749

Vinícius Julio - 2500392

Henrique Ferreira Santos - 2502307

Silvio Cezar Teles - 2502503

Marcio Soares de Oliveira - 2500306

1. Definição e objetivos

O projeto tem como objetivo construir um pipeline completo de ETL (Extract, Transform, Load) para ingestão e processamento de cotações cambiais, aplicando técnicas de normalização e enriquecimento com insights de um LLM (Large Language Model). A entrega final inclui não apenas a parte técnica (código, testes e automação), mas também os artefatos executivos (documentação e apresentação para diretoria).

2. Estrutura inicial

A arquitetura do projeto foi organizada da seguinte forma:

```
exchange-rate-pipelne-llm/
├── src/              # código-fonte principal
│   ├── ingest.py       # captura dados da API
│   ├── transform.py    # normalização e limpeza
│   ├── load.py         # exporta para Parquet e DB
│   ├── llm_insights.py   # persistência dos insights do LLM
│   └── __init__.py
├── data/
│   ├── raw/          # dados brutos da API
│   ├── silver/         # dados normalizados
```

```
│   └── gold/          # dados finais em Parquet e insights
├── tests/             # testes unitários e de integração
├── logs/              # logging estruturado
├── exchange_rate.db   # banco SQLite para persistência
├── README.md          # documentação
├── requirements.txt   # dependências
├── .env               # variáveis locais (não versionadas)
└── .env.example       # modelo de configuração
```

3. Implementações técnicas

Ingestão dos dados (/data/raw)

- Dados brutos salvos em JSON/CSV
- Uso de requests com timeout e tratamento de erros HTTP
- Retry exponencial usando tenacity
- Salvamento do JSON bruto em /data/raw com nome padronizado YYYY-MM-DD.json e sufixo _HHMMSS se houver múltiplas coletas no mesmo dia
- Inclusão de metadados no JSON (timestamp, status HTTP, URL consultada)
- Logging estruturado em JSON (INFO/ERROR) com run_id e arquivo
- Escrita atômica (salvar em tmp e depois renomear)

Transformação (/data/silver)

- Normalização de colunas e tipos
- Remoção de taxas inválidas
- Leitura do JSON bruto correspondente à data
- Normalizar em DataFrame com colunas obrigatórias (base_currency, target_currency, rate, retrieved_at, date)
- Validação: nenhuma taxa nula, zero ou negativa; base_currency correto
- Conversão de tipos (rate como float/Decimal, arredondamento 6 casas)
- Remoção de duplicatas (target_currency + retrieved_at)
  Arquivo de rejeitados em /data/raw/rejects com motivo
- Logging detalhado de erros
- Escrita em /data/silver/YYYY-MM-DD.parquet (engine pyarrow, compressão snappy, index=False)

Carga (/data/gold)

- Salvamento em Parquet e SQLite
- Agregar arquivos /data/silver e gerar /data/gold/YYYY-MM-DD.parquet
- Inserir metadados (pipeline_version, run_id, run_timestamp)
- Opcional: carregar em banco via SQLAlchemy, if_exists='append', chunksize=1000
- Garantir índice único (date + base_currency + target_currency) para evitar duplicatas

Enriquecimento

- Calcular métricas: pct_change, volatilidade, top movers
- Criar resumo compacto (JSON ou tabela com top 5 moedas)

- Chamar LLM (OpenAI API) com template de prompt
- Logging do prompt e resposta em /logs/llm_prompts.log

Testes e Logging (/tests, /logs)

- Unitários: ingest, transform, load
- Garantir que pytest roda sem falhas
- Logging estruturado: eventos registrados em logs
- Cobrem ingestão, transformação e carga
-

Integração com LLM

- Persistência de insights em JSON no /data/gold

Variáveis de ambiente

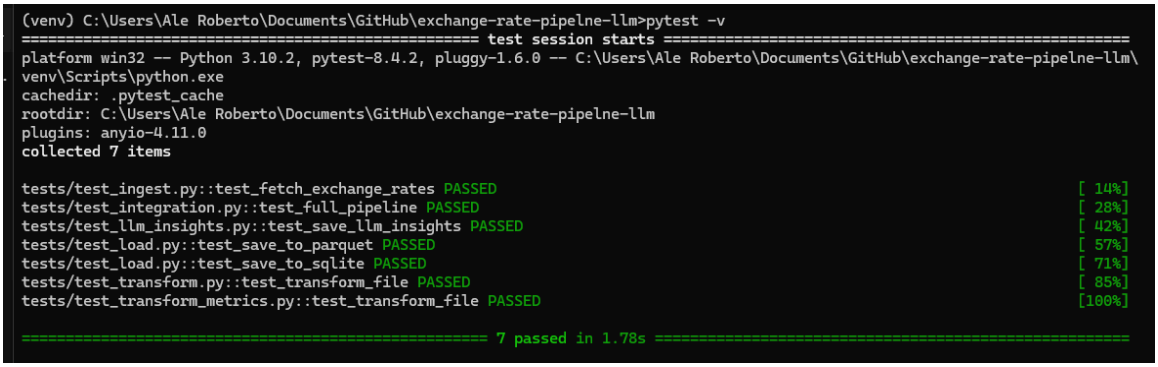
- Criado .env.example com placeholders

5. Resumo final

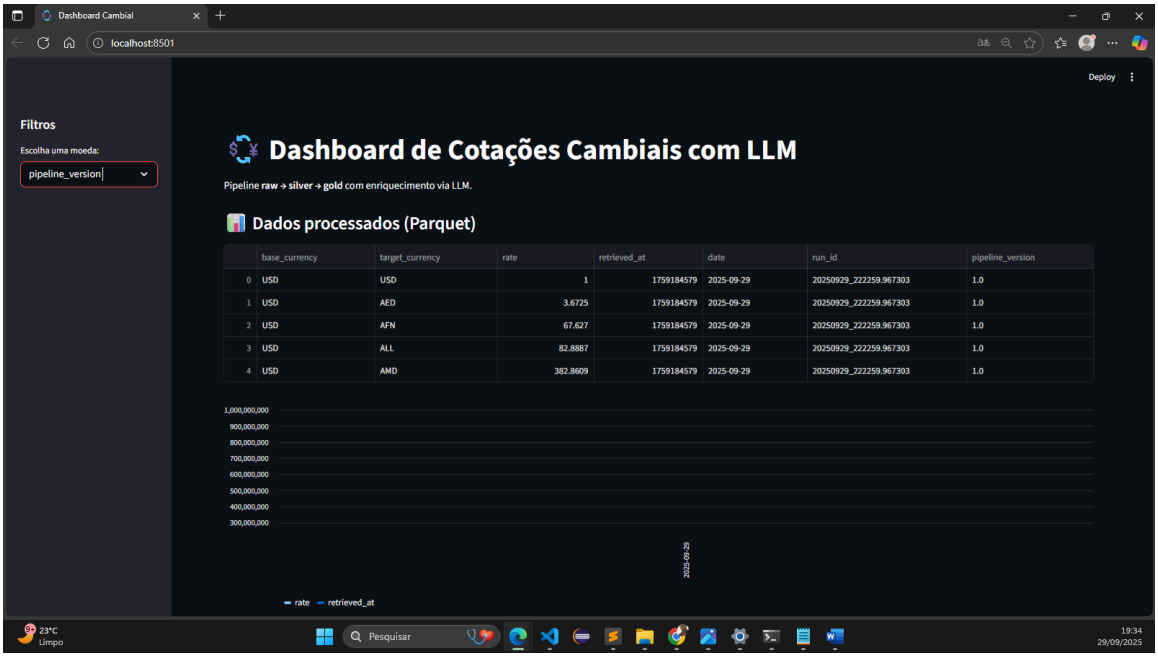
- Pipeline ETL concluído.
- Integração com LLM implementada.
- Testes automatizados todos passando.
- Boas práticas aplicadas (.env.example, logging estruturado).

6. Imagens

Testes:



Dashboard


Dashboard Cambial
localhost:8501
Deploy
Filtros
Escolha uma moeda:
pipeline_version
Dashboard de Cotações Cambiais com LLM
Pipeline raw → silver → gold com enriquecimento via LLM.
Dados processados (Parquet)
base_currency target_currency rate retrieved_at date run_id pipeline_version
0 USD USD 1 1759184579 2025-09-29 20250929_222259.967303 1.0
1 USD AED 3.6725 1759184579 2025-09-29 20250929_222259.967303 1.0
2 USD AFN 67.627 1759184579 2025-09-29 20250929_222259.967303 1.0
3 USD ALL 82.8887 1759184579 2025-09-29 20250929_222259.967303 1.0
4 USD AMD 382.8609 1759184579 2025-09-29 20250929_222259.967303 1.0
1,000,000,000
900,000,000
800,000,000
700,000,000
600,000,000
500,000,000
400,000,000
300,000,000
2025-09-29
rate
retrieved_at
23°C
Limpo
Pesquisar
19:34
29/09/2025


Dashboard Cambial
localhost:8501
Deploy
Filtros
Escolha uma moeda:
pipeline_version
2025-09-29
rate
retrieved_at
Insights do LLM
{
"date" : "2025-09-29"
"summary" : "Mercado de câmbio apresentou forte volatilidade."
"top_risks" : [
0 : "Inflação EUA"
1 : "Política monetária Brasil"
]
"source" : "OpenAI GPT-4"
}


Dashboard Cambial
localhost:8501
Deploy
Filtros
Escolha uma moeda:
pipeline_version
2025-09-29
rate
retrieved_at
Insights do LLM
{
"date" : "2025-09-29"
"summary" : "Mercado de câmbio apresentou forte volatilidade."
"top_risks" : [
0 : "Inflação EUA"
1 : "Política monetária Brasil"
]
"source" : "OpenAI GPT-4"
}


Deploy
Filtros
Escolha uma moeda:
run_id
2025-09-29
rate
retrieved_at
Insights do LLM
{
"date" : "2025-09-29"
"summary" : "Mercado de câmbio apresentou forte volatilidade."
"top_risks" : [
0 : "Inflação EUA"
1 : "Política monetária Brasil"
]
"source" : "OpenAI GPT-4"
}
Evolução da moeda: run_id
20250929_22225...
2025-09-29