Desafio 1: Integração com um Serviço de Terceiros

Consultar a API: Realizar chamadas para uma API de clima (ex.: OpenWeatherMap, Weatherstack, Open-Meteo ou similar) utilizando a cidade como parâmetro.
Processar e Exibir Dados: Processar a resposta da API e exibir informações relevantes como temperatura, umidade, e condições do tempo. Os dados devem ser formatados de forma amigável.
Cache de Resultados: Implementar um sistema de cache para armazenar as respostas da API por um período de tempo (ex.: 10 minutos) para reduzir o número de chamadas à API.
Tratar Erros: Garantir que erros de rede ou respostas inválidas da API sejam tratados de forma adequada, com mensagens de erro claras.

Desafio 2: Manipulação de Dados

Parte 1: Manipulação de Dados com Pandas

1. **Carregue os Dados:**
   - Carregue o arquivo Excel `vendas_ecommerce.xlsx` em um DataFrame.
2. **Análise Inicial de Vendas:**
   - Calcule o total de vendas por categoria de produto.
   - Filtre os dados para mostrar apenas as vendas realizadas no último mês.
   - Adicione uma nova coluna que indique o valor total da venda (`Quantidade Vendida * Preço Unitário`).
3. **Exportação de Resultados:**
   - Exporte os resultados filtrados para um novo arquivo Excel chamado `vendas_filtradas.xlsx`.
4. **Bônus - Geração de Insights:**
   - Extraia insights adicionais dos dados através de cruzamento de informações e storytelling. Explore relações entre categorias, sazonalidade, ou comportamento dos clientes.

Parte 2: Análise de Dados e Visualização

1. **Análise por Cliente:**
   - Calcule a receita total gerada por cada cliente.
   - Identifique os 5 clientes que geraram a maior receita e faça uma análise sobre o comportamento deles.
2. **Visualização dos Dados:**
   - Crie uma visualização que mostre a distribuição dos valores das transações.
   - Gere um gráfico que mostre a receita mensal do e-commerce ao longo do último ano.

Desafio 3: Desenvolvimento de Algoritmos

1. **Função de Algoritmo:**
   - Escreva uma função que receba uma lista de inteiros (representando valores de transações diárias) e retorne a maior soma de um subarray contínuo (subconjunto de transações).
2. **Otimização:**
   - Otimize a função para garantir que ela execute com a melhor complexidade temporal possível.
3. **Testes Unitários:**

    - Escreva testes unitários para validar a função com diferentes casos de teste, incluindo casos de borda e entradas grandes.

Entrega:

- Forneça o código Python em um Jupyter Notebook.
- O código deve estar bem documentado, explicando as decisões tomadas em cada etapa.
- Inclua as visualizações geradas, insights extraídos, e os resultados dos testes unitários.
- Deverá ser criado uma imagem de container para cada desafio
- Realizar o desafio no GitHub