Atividade: Integração de APIs — Clima, Qualidade do Ar e Informações de Países

Objetivo Geral

O objetivo desta atividade é desenvolver um pequeno pipeline de dados integrando informações de três APIs distintas:

- **WeatherAPI**: dados de clima atual e previsão.
- **AirVisual API**: dados de qualidade do ar.
- **REST Countries API**: informações detalhadas sobre países.

Além disso, você irá trabalhar práticas de **programação segura**, **boas práticas de organização de código** e **tratamento de erros**, simulando um ambiente mais próximo de projetos profissionais.

Parte 1 — Preparação de Ambiente

Antes de começar a programar, é necessário preparar o seu ambiente:

- Crie um ambiente virtual ou use seu ambiente Python instalado.
- Instale as bibliotecas necessárias: requests, pandas e python-dotenv.
- Crie uma estrutura básica de projeto, separando dados e scripts em pastas.

Também é importante configurar suas **API Keys** de forma segura:

- Crie um arquivo .env para armazenar suas chaves de API.
- Utilize a biblioteca dotenv para carregar essas chaves no seu script, evitando que fiquem expostas no código.

Parte 2 — Comunicação com APIs

Você irá interagir com três APIs:

- Fazer chamadas HTTP (GET) para buscar dados de clima, qualidade do ar e informações de países.
- Entender a estrutura das respostas (normalmente no formato JSON).
- Realizar o **tratamento de erros**: a comunicação com APIs pode falhar e seu código deve estar preparado para isso.

Dicas:

- Leia a documentação oficial de cada API para entender os parâmetros necessários.
- Utilize try/except para capturar falhas de conexão e erros de resposta.

Parte 3 — Armazenamento de Dados

Após receber os dados das APIs:

- Inicialmente, salve os dados **em bruto** (sem formatação) em arquivos .txt.
- Depois, processe os dados e extraia apenas as informações relevantes.
- Salve os dados processados em **arquivos CSV**, organizando as colunas adequadamente.

Esse processo simula o comportamento de pipelines de dados reais, onde primeiro se coleta a informação e depois ela é tratada para análises posteriores.

Parte 4 — Integração de Dados

Você irá integrar informações de diferentes fontes:

- Ao consultar o clima de uma cidade, recupere o país retornado.
- Com base no nome do país, faça uma nova consulta na API REST Countries para buscar informações detalhadas (ex: população, capital, moeda).
- Salve esses dados integrados em um único arquivo final.

Este passo desenvolve a habilidade de **relacionar fontes de dados distintas**, um conceito fundamental em engenharia de dados.

Conceitos que você aplicará na prática:

- Estrutura de um projeto com organização de arquivos.
- Boas práticas de segurança com API Keys.
- Requisições HTTP e parsing de JSON.
- Tratamento de exceções.
- Manipulação de dados com pandas.
- Integração de dados de múltiplas APIs.

Importante:

Você **não deve apenas copiar e colar código**. Entenda:

- O que a API espera de você (parâmetros).
- Como os dados retornam (analisando o JSON).
- Como estruturar as informações para salvar corretamente.

Dicas para Atividade: Integração de APIs

Dica 1 — Como organizar o projeto

Estruture suas pastas dessa forma:

```
meu_projeto_api/
│
├── .env          # Arquivo com suas chaves de API
├── main.py       # Arquivo principal de execução
├── requirements.txt
├── /data/        # Pasta onde ficarão os arquivos de saída
```

Dica 2 — Como carregar variáveis do

.env

Pesquise como utilizar a biblioteca dotenv para acessar variáveis no seu código Python.

Você precisará carregar as variáveis de ambiente e utilizá-las para montar as URLs das APIs.

Dica 3 — Como fazer uma requisição básica

Estude a documentação da biblioteca requests.

Você irá usar o método requests.get(url) para fazer chamadas às APIs.

Lembre-se de verificar se a resposta foi bem-sucedida (status code 200) antes de tentar acessar os dados.

Dica: utilize response.json() para trabalhar mais facilmente com o resultado.

Dica 4 — Como salvar um arquivo de texto

Após receber a resposta JSON de uma API, você pode converter esse dicionário em uma string e gravá-lo em um .txt.

Pesquise como usar o comando open em modo de escrita.

Dica 5 — Como transformar JSON em CSV

Analise o formato dos dados que você receberá (eles estarão em dicionário dentro de listas ou vice-versa).

Utilize a biblioteca pandas para criar um DataFrame a partir dos dados extraídos.

Sugestão: use o método DataFrame.to_csv('caminho.csv').

Dica 6 — Como tratar falhas

Em todas as chamadas HTTP, envolva seu requests.get dentro de um try/except.

Dica: aprenda como capturar exceções específicas, como requests.exceptions.RequestException.

Dica 7 — Como integrar diferentes APIs

Após buscar os dados do clima, extraia o nome do país da resposta e use esse valor para construir a chamada à API REST Countries.

Sugestão: observe que algumas APIs exigem formatação especial no nome do país (como substituir espaços por %20 ou usar o formato correto no endpoint).

Resultado Esperado

Ao final, você deverá ter:

- Um arquivo .txt com dados brutos de clima e qualidade do ar.
- Arquivos .csv individuais para clima e país.
- Um arquivo .csv final integrando todos os dados (cidade, clima, qualidade do ar, país).