Desafio Técnico
Catálogo de Países + Agência de Viagens

Imagine que você foi contratado por uma pequena agência de viagens online que deseja modernizar sua plataforma. Hoje eles possuem apenas uma lista desorganizada de destinos, mas querem um sistema simples e funcional onde possam:
- Mostrar países de forma organizada, a partir de uma API pública de países;
- Criar roteiros de viagens personalizados, com informações de datas, detalhes, preço e imagens;
- Permitir que qualquer visitante navegue entre países e roteiros sem precisar de login.

Seu papel é desenvolver essa aplicação em Django, garantindo que a equipe de marketing consiga facilmente cadastrar novos roteiros e que os clientes tenham uma boa experiência ao visualizar os destinos disponíveis.

1. Objetivo

Construir uma aplicação web em Django que:
1. Consome uma API pública de países (sugestão: REST Countries) para manter um catálogo local;
2. Permite cadastrar roteiros de viagens associados a países, com imagens, detalhes, data de início e data de partida;
3. Exibe listagens e detalhes com busca, filtros e ordenação, usando HTML/CSS/JS;
4. Sobe com Docker/docker-compose;
5. Demonstra boas práticas de Git e um README claro.

2. Requisitos Funcionais

- Criar comando `manage.py sync_countries` para sincronizar países da API.
- Campos mínimos de país: name, cca2, region, subregion, capital, population, flag_url.
- Incluir fallback com countries_sample.json.
- Implementar CRUD de Itinerary (roteiros), associado a Country: título, país, datas de início e partida, descrição, preço (opcional) e imagem.
- Exibir:
  * /countries/: listagem com busca, filtro e ordenação;
  * /countries/<cca2>/: detalhe do país e seus roteiros;
  * /itineraries/: listagem com filtros por país e período;
  * /itineraries/<id>/: detalhe do roteiro;
  * Formulários para criar/editar roteiros.
- Front-end em HTML/CSS/JS vanilla, Tailwind , responsivo e simples.

3. Requisitos Técnicos

- Python 3.11+, Django 4/5+;
- PostgreSQL via Docker;
- Subida do projeto via `docker-compose up --build` em http://localhost:8000;
- Uso de variáveis em .env.example;
- Git com histórico de commits claros e pequenos.

4. Diferenciais

- Suporte a múltiplas imagens por roteiro.

5. Data-limite

Entrega até 01 de setembro de 2025.