CRUD – TypeScript / PostgreSQL

Professor: André Olímpio

Disciplina: Programação Web

Desenvolver uma aplicação **web completa** (front-end e back-end) utilizando **TypeScript** e **PostgreSQL**, com o propósito de gerenciar dados relacionados a **cidades, países e continentes**. O sistema deve permitir **operações CRUD completas** (Create, Read, Update, Delete) e realizar **integração com pelo menos duas APIs externas** relevantes ao contexto geográfico.

Descrição do Projeto:

O projeto consiste em criar uma **aplicação web interativa e funcional** que possibilite ao usuário cadastrar, consultar, editar e excluir informações sobre **cidades, países e continentes**.

Cada cidade deve estar associada a um país, e cada país a um continente.

Além disso, a aplicação deverá consumir **duas APIs externas** que complementem os dados ou funcionalidades do sistema, como por exemplo:

- API de informações geográficas (**Ex:** REST Countries)
- API de clima (**Ex:** OpenWeatherMap)
- API de bandeiras, mapas ou dados econômicos.

Requisitos Funcionais:

1. **Cadastro de Continentes**
    - Permitir criar, visualizar, atualizar e excluir continentes.
    - Campos obrigatórios: *id, nome, descrição.*
2. **Cadastro de Países**
    - Cada país deve estar vinculado a um continente.
    - Campos obrigatórios: *id, nome, população, idioma oficial, moeda, id_continente.*
    - Deve ser possível listar países por continente.
3. **Cadastro de Cidades**
    - Cada cidade deve estar vinculada a um país.
    - Campos obrigatórios: *id, nome, população, latitude, longitude, id_pais.*
    - Deve ser possível listar cidades por país e/ou continente.

4. **Integração com APIs**
    - Integrar **duas APIs externas**, sendo:
        - Uma para **obter dados complementares** (**Ex:** informações de países, moedas, clima, ou fusos horários);
        - Outra para **enriquecer a interface** (**Ex:** exibir bandeiras, mapas, imagens ou dados econômicos).
    - Os dados obtidos das APIs devem ser exibidos dinamicamente na aplicação.
5. **Interface Web**
    - Desenvolver uma **interface gráfica amigável e responsiva**, utilizando HTML, CSS e/ou frameworks como React, Angular ou outro compatível com TypeScript.
    - Incluir **telas** de:
        - Login (opcional);
        - Cadastro/edição de continentes, países e cidades;
        - Consulta e listagem com filtros e paginação;
        - Exibição de dados adicionais vindos das APIs (como bandeira, clima ou mapa).
6. **Banco de Dados**
    - Utilizar **PostgreSQL** para armazenar os dados locais do sistema.
    - Criar **tabelas relacionadas** com chaves estrangeiras (continente → país → cidade).
    - O acesso ao banco deve ser feito via **Prisma ORM** ou **pg** (biblioteca nativa do Node).