Documento Técnico – Roteirizador Inteligente e Sustentável

🎯 Objetivo

Sistema que gera roteiros turísticos personalizados, equilibrando:

- **Orçamento**
- **Tempo (dias de viagem)**
- **Preferências (ex: lazer, gastronomia, natureza, cultura, aventura, etc.)**
- **Transporte** escolhido (com cálculo de custo e impacto ambiental)

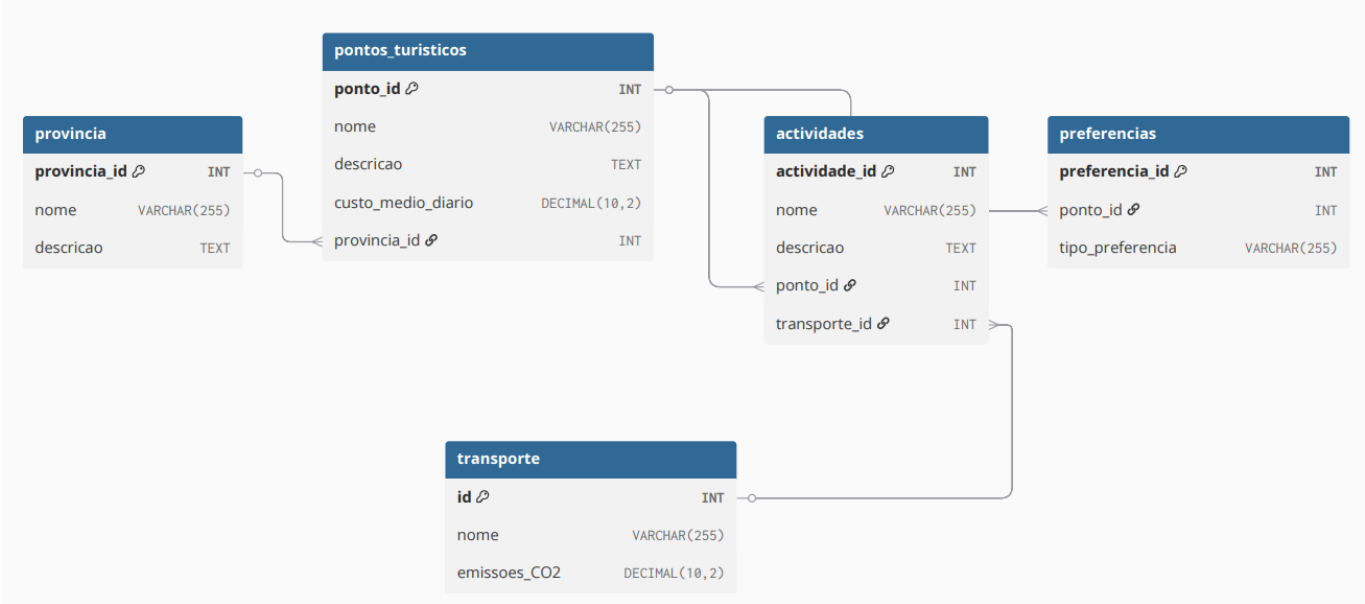
O sistema devolve **até 3 províncias** com pontos turísticos e atividades que se ajustam ao orçamento/dias, e no fim sugere o **roteiro mais sustentável**.

Banco de Dados (SQLite)

Tabelas

```
CREATE TABLE provincia (
    provincia_id INTEGER PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    descricao TEXT
);

CREATE TABLE pontos_turisticos (
    ponto_id INTEGER PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    descricao TEXT,
    custo_medio_diario DECIMAL(10, 2),
    provincia_id INTEGER,
    FOREIGN KEY (provincia_id) REFERENCES provincia(provincia_id)
);

CREATE TABLE actividades (
    actividade_id INTEGER PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    descricao TEXT,
    ponto_id INTEGER,
    transporte_id INTEGER,
    FOREIGN KEY (ponto_id) REFERENCES pontos_turisticos(ponto_id),
    FOREIGN KEY (transporte_id) REFERENCES transporte(id)
);

CREATE TABLE preferencias (
    preferencia_id INTEGER PRIMARY KEY,
```

```
    ponto_id INTEGER,
    tipo_preferencia VARCHAR(255),
    FOREIGN KEY (ponto_id) REFERENCES pontos_turisticos(ponto_id)
);

CREATE TABLE transporte (
    id INTEGER PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    emissoes_CO2 DECIMAL(10, 2)
);
```



Fluxo de Funcionamento

1. **Input do utilizador:**
   - Orçamento total
   - Tempo(dias pretendidos)
   - Preferências (ex: lazer, gastronomia, cultura)
   - Tipo de transporte
2. **Cálculo inicial:**
   - `orcamento / tempo` → **custo médio diário**
   - Filtrar pontos turísticos com `custo_medio_diario <= custo_médio_diario(do ponto turístico)`
   - Mapear pelas **preferências** escolhidas
3. **Seleção das províncias:**
   - Selecionar **até 3 províncias** que mais se ajustam às condições
   - Associar pontos turísticos e atividades disponíveis
     No final seria:

Provincia(1)

Pontos turísticos cujo custo medio se enquadrem ao custo medio calculado com base no orçamento / tempo.~

Actividades a fazer nestes pontos turísticos

Provincia(2)

Pontos turísticos cujo custo medio se enquadrem ao custo medio calculado com base no orçamento / tempo.~

Actividades a fazer nestes pontos turísticos

Provincia(3)

Pontos turísticos cujo custo medio se enquadrem ao custo medio calculado com base no orçamento / tempo.~

Actividades a fazer nestes pontos turísticos

4. **Cálculo de impacto ambiental:**
   - `impacto_CO2 = emissoes_CO2(transporte) * dias`
   - Selecionar o **roteiro mais sustentável**
5. **Sugestões de sustentabilidade:**
   - Trocar transporte por outro com menor CO_2
   - Usar menos plástico
   - Reduzir consumo de água
   - Escolher alimentação local

⚙ Tecnologias Sugeridas

- **Backend / Lógica:**
  - Python
  - Lógica com SQLite
- **Banco de Dados:**
  - SQLite (simples e leve)
- **Frontend:**
  - Opções:
    - **Streamlit** (mais rápido para MVPs e hackathons)
    - **React** (se acharem que vai ser simples)
    - Flask + templates (para versão simples web)
- **Bibliotecas Python úteis:**
  - `sqlite3` → integração com BD
  - `pandas` → manipulação de dados e cálculos
  - `matplotlib/plotly` → gráficos para impacto ambiental
  - `streamlit` → frontend rápido

📄 Estrutura de Diretórios (exemplo)

```
roteirizador/
├── app.py                # aplicação principal (Flask ou Streamlit)
├── db/
│   └── schema.sql        # criação do banco
│   └── seed.sql          # dados iniciais
├── models/
│   └── provincia.py
│   └── pontos_turisticos.py
│   └── actividades.py
│   └── transporte.py
│   └── preferencias.py
├── services/
│   └── roteiro_service.py # lógica principal do cálculo do roteiro
│   └── sustentabilidade.py # cálculo de CO2 e recomendações
├── static/               # CSS, JS, imagens (caso Flask/React)
├── templates/            # HTML (caso Flask)
```

🚀 Roadmap de Implementação (MVP)

1. **Discussão e modelagem do BD**
2. **Inserção de dados iniciais (seed)**
3. **Funções principais (roteiro_service):**
   - cálculo de custo médio diário
   - filtro por preferências
   - seleção das províncias e pontos turísticos
4. **Funções de sustentabilidade (sustentabilidade.py):**
   - cálculo de impacto ambiental
   - sugestões de redução
5. **Interface (Streamlit):**
   - formulários para input do utilizador
   - cards com os roteiros sugeridos
   - gráfico comparativo de emissões
6. **Testes e ajustes**