AI project

Tourist Assistant System Components

Project Overview

Project Name: Rahal – Tourist Assistant

Type: Intelligent Travel Planning System

Goal: To provide users (tourists) with personalized and intelligent recommendations for places to visit (POIs) based on their preferences and contextual data using an LLM-powered assistant.

✅ 2. Functional Requirements

A. User Management

- Users can register and log in with authentication.
- Each user has a unique profile including:
  - Name, email, username, password, role.
- Roles: Admin, Tourist

B. Places (POIs) Management

- Admin can add, update, delete POIs.
- POIs include:
  - `uuid`, `poi_name`, `lat`, `lng`, `reviews_no`, `price_range`
- Categories may include: Restaurant, Museum, Cafe, Park, etc.

C. Trip Planning (ToVisit List)

- Users can add places to their **ToVisit list** with:

    - Name, Description, Priority, Category
- Users can rate or mark places as visited (Visited_Places).

D. Visited Places

- Track previously visited places.
- Allow feedback (rating) on visited places.

E. LLM-Based Tourist Assistant

- **Recommendation**:
    - LLM suggests personalized POIs based on:
        - User history
        - Preferences (budget, type of places)
        - Current location (optional)
- **Search Assistant**:
    - LLM helps user by fetching relevant POIs using online sources or available data.
    - Natural language queries like:
        - *“Find me a cheap seafood restaurant near downtown”*
        - *“Suggest 3 historical sites for tomorrow”*

F. Data Filtering and Sorting

- POIs can be filtered by:
    - Category, Rating, Price Range, Distance
- Sort by: priority, rating, reviews number

📦 3. Non-Functional Requirements

A. Performance

- Fast response for LLM-based recommendations.

- Efficient database queries for POIs.

B. Scalability

- Database schema supports new POI types or extended categories.
- LLM module should be modular to allow future model upgrades (e.g., OpenAI, Mistral).

C. Security

- User authentication and authorization (hashed passwords).
- Secure API access (JWT or OAuth2 suggested).

D. Maintainability

- Clean and modular codebase using FastAPI/Flask backend and SQLAlchemy ORM.
- Dockerized deployment for reproducibility.

E. Accessibility

- UI is mobile-responsive (if web app).
- Arabic language support optional for tourists in Arabic-speaking regions.

🧠 4. AI Integration Requirements (LLM)

A. Recommendation Logic

- Input: User preferences, category, location, time
- Output: List of recommended POIs with justifications (reasoning)
- Use techniques like:
  - In-context learning (few-shot prompts)
  - Embedding similarity (optional for semantic search)

B. Chat Assistant Interface

- Natural language interface that:
  - Understands complex queries
  - Handles context (follow-up questions)
  - Can retrieve external data if needed (search plugins/RAG)

🗃️ 5. Database Design (Summary)

Tables:

- `Users`: Tourist data
- `Places`: POI master table
- `ToVisit`: Planned itinerary
- `Visited_Places`: History with rating

Suggested Additions:

- `Categories`: To normalize place categories
- `User_Preferences`: Optional table to store preferred types, budget, languages

---

🛠️ 6. Tech Stack Suggestion

- **Backend**: Python (FastAPI) + SQLAlchemy ORM
- **LLM Integration**: OpenAI API / HuggingFace Transformers
- **Frontend**: React / Vue (or basic HTML for MVP)
- **Database**: PostgreSQL / SQLite (for dev)
- **Geolocation**: Leaflet / Google Maps API
- **LLM RAG Option**: LangChain or Haystack (if using local embedding search)

📈 7. User Journey Example

1. Tourist signs up and enters preferences (e.g., low-budget food).
2. Tourist chats with Rahal assistant: *"Where should I eat tonight?"*

3. LLM recommends 3 nearby restaurants using Places data.
4. User adds one to ToVisit list.
5. After visit, marks it as visited and leaves a rating.
6. LLM uses visit history for better future suggestions.

📌 8. Future Scope

- Voice-enabled assistant (multimodal support)
- Itinerary optimization (e.g., minimize travel time)
- Social features: Group planning, friend recommendations
- Local language translator

Notion