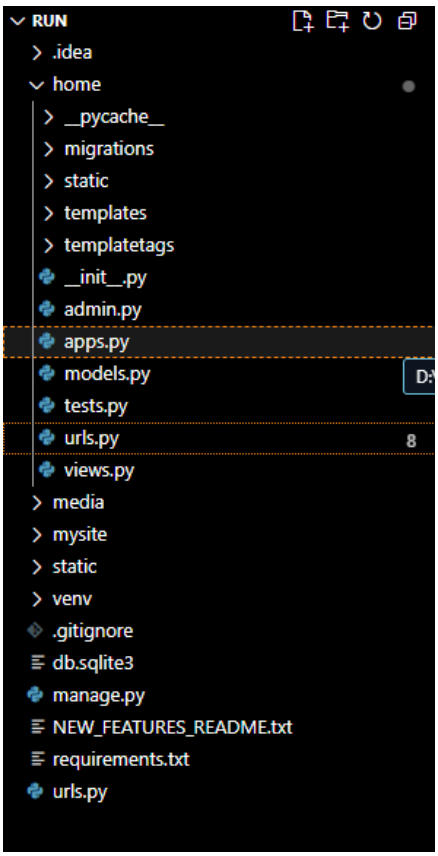
Implementation and Technical Documentation: Travel Assistant Web Application

1. Introduction

The Travel Assistant web application is developed to simplify travel planning by integrating tour packages, hotel locators, and city-specific recommendations. Users can search for destinations, view hotel listings, and explore tour packages with a user-friendly interface.

The project is implemented using **Python**, **Django**, **HTML, CSS, JavaScript**, and **SQLite**. The application emphasizes modularity, maintainability, and seamless integration across components.

2. Code Structure and Organization



3. Implementation Details

3.1 Technologies Used

- **Backend:** Python 3.10, Django 4.x
- **Frontend:** HTML5, CSS3, JavaScript, Bootstrap
- **Database:** SQLite
- **Libraries/Tools:** Django

3.2 System Components

1. **User Interface (UI):**
    - Templates provide interactive pages for search, hotel listings, and tour packages.
    - Responsive design via Bootstrap ensures compatibility across devices.
2. **Backend (Django views):**
    - Handles search queries, hotel listings, and tour package logic.
    - `views.py` contains all request-handling functions for each page.
3. **Database (SQLite):**
    - Stores structured data on destinations, hotels, and tour packages.
    - `models.py` defines database schemas.

3.3 Key Processes

- **Hotel Listing:** Based on selected city, hotels are displayed with details like name, rating, and availability.
- **Tour Package Recommendations:** Packages are filtered according to city and user preferences.

4. Integration Across Components

- **Frontend - Backend:** Django views process user input from forms and render dynamic HTML templates.
- **Backend - Database:** Django with SQLite, ensuring data retrieval, insertion, and filtering.
- **UI - Logic:** JavaScript enhances user experience with dynamic behavior.

5. Testing and Verification

5.1 Testing Procedures

- **Unit Tests:** Verify individual functions (e.g., database queries for hotels, filtering logic for packages).
- **Integration Tests:** Ensure correct communication between frontend, backend, and database.
- **Manual Testing:** Functional testing of search, hotel listing, and tour package recommendations.

5.2 Evidence of Functionality

- Screenshots of:
    - Home page search bar
    - Hotel listings for a city
    - Tour package recommendations
- Logs showing successful database queries and user reques