**Backend**

* **app.py:** This is the main application file for TravelMate. Likely initializes the web application, sets up the routes, and configures any necessary settings for the backend server. This file could also handle the app’s startup sequence and manage interactions between the user and database through controllers.
* **Init\_db.py:** This script is responsible for setting up the initial database. Includes code to create the database if it doesn’t exist, executes the schema.sql file, and possibly adds default or initial data to the database. Utility for initializing and setting up the database structure before the application if fully functional.
* **models.py:** Defines data models (representation of tables in the database) for the application. Likely includes classes that map to database tables, with attributes corresponding to table columns and relationships defined for interactions between tables (e.g., trips and activities)
* **controllers**
  + **activity\_controller.py:** Manages the routes and logic related to activities in a trip. It likely handles requests to add, update, delete, or retrieve activities within a trip, interacting with the models.py file to perform CRUD (Create, Read, Update, Delete) operations on activities
  + **trip\_controller.py:** Focuses on managing trips. Likely includes functions for creating, updating, deleting, and retricing trips. This file facilitates interactions between the backend and the frontend regarding trip data.
  + **user\_controller.py:** Manages user-related functionality, such as registration, login, profile management, and possibly authentication. It includes routes and logic for interacting with user data, handling operations like creating user accounts and managing user sessions.
* **utils**
  + **db\_connection.py**: Responsible for managing the database connection in TravelMate. Likely includes functions or configurations to establish and close connections to the SQL database, enabling the application to interact with and query the database. It may use a library like sqlite3 or SQLAlchemy to facilitate secure and efficient database access, ensuring that data operations can be performed reliably throughout the app. This file is foundational for any data retrieval, insertion, and updates within the application.

**Frontend**

* **SRC**
  + **App.css:** This CSS file contains general styles for the TravelMate application, including fonts, background colors, and basic layout properties. It applies styles to elements like the body, headings, paragraphs, and the container, giving the app a consistent look and feel. It also includes styling for the navigation bar, using a light blue color scheme to match the apple theme.
  + **App.js:** This is the main React component for the frontend, acting as the root of the application’s component tree. It sets up routing using react-router-dom to manage navigation between different pages, such as the login, registration, trip manager, itinerary creation, and profile views. App.js integrates various components like the Navbar, Login, Register, and other, serving as the base for the entire user interface.
  + **index.js:** This is the entry point for the React application. It renders the App component within the root element in index.html, initializing the React app and linking it to the DOM. index.js sets up the foundational structure for the frontend and is essential for starting the app.
  + **components**
    - **ItineraryManager.jsx:** Handles the creation and management of trip itineraries. It provides functionality for users to add, edit, and organize activities within an itinerary, linking the frontend with the backend to store and retrieve itinerary data.
    - **Login.jsx:** Handles user login functionality. Includes a form for users to enter their credentials, send login requests to the backend, and processes authentication responses to allow users access to their accounts.
    - **Register.jsx:** Handles the user registration functionality. Includes a form for entering user information, sends registration data to the backend, and handles responses to confirm successful account creation or display errors.
    - **TripManager.jsx:** Responsible for managing trips within the app. It allows users to create, view, edit, and delete trips, interacting with the backend to retrieve and update trip data. This is a core part of the app’s functionality, enabling users to organize their travel plans.
    - **Profile.jsx:** Manages the user profile page, allowing users to view and edit their personal information. It may also display details about the user’s trips or saved itineraries, providing a personalized space for each user.
    - **Navbar.jsx:** Renders the navigation bar at the top of the page, allowing users to access different parts of the app, such as login, registration, trips, and profile pages. It likely includes links and may conditionally display options based on user login status.
    - **Navbar.css:** Styles the navigation bar used throughout the app. Defines properties such as background color, padding, and hover effects for links, ensuring a cohesive and user-friendly navigation experience across the application.
  + **public**
    - **index.html:** The main HTML file that serves as the entry point for the front end. This file provides the basic structure for the web page, setting up a root <div> element where the React.js frontend can be rendered dynamically. It also includes meta tags and basic HTML setup for the TravelMate user interface.

**Database**

* **Schema.sql**: This SQL file defines the structure of the database, including tables for trips, activities, users, and any other entities required by the application. Contains SQL commands to create tables, set up relationships, and enforce constraints, making it essential for initializing database schema.