Software Requirements Specification for Mumbai Trails:

An AI-Powered Travel Recommendation System

Batch: A2

Prepared by :

Shreeya Inamdar	**2023800032**	**shreeya.inamdar23@spit.ac.in**
Rachel Fernandes	**2023800021**	**rachel.fernandes23@spit.ac.in**
Divit Gupta	**2023800029**	**divit.gupta23@spit.ac.in**

Instructor: Prof. Swati Vyas
Course: Software Engineering

Date: 1/09/2022

Index

1 INTRODUCTION 3
1.1 Document Purpose 3
1.2 Product Scope 3
1.3 Intended Audience and Document Overview 3
1.4 Definitions, Acronyms and Abbreviations 4
1.5 Document Conventions 4
1.6 References and Acknowledgments 5
2 OVERALL DESCRIPTION 6
2.1 Product Perspective 6
2.2 Product Functionality 6
2.3 Users and Characteristics 7
2.4 Operating Environment 7
2.5 Design and Implementation Constraints 8
2.6 User Documentation 8
2.7 Assumptions and Dependencies 8
3 SPECIFIC REQUIREMENTS 9
3.1 External Interface Requirements 10
3.2 Functional Requirements 11
4 OTHER NON-FUNCTIONAL REQUIREMENTS 13
4.1 Performance Requirements 13
4.2 Safety and Security Requirements 13
4.3 Software Quality Attributes 14

1 Introduction

1.1 Document Purpose

This Software Requirements Specification (SRS) provides a description of all the functions and constraints of the **MumbAI Trails System**, developed as a smart tourism platform for Mumbai.
It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate, and how the system will react to the internal and external environment. This document is intended for both the stakeholders and the developers of the system.

1.2 Product Scope

The MumbAI Trails project is aimed at developing an online AI-powered tourism application for travelers exploring Mumbai. The system is a web-based application that can be accessed both locally and remotely with proper login credentials provided.
The main aim is to reduce the manual effort involved in travel planning by providing personalized, mood-aware itineraries and optimized routes. In order to avoid the hassles of traditional planning, MumbAI Trails will generate itineraries automatically using Artificial Intelligence, considering user preferences, time, budget, and real-time weather/traffic data.
The system will also store essential details such as:

1. Tourist profiles and preferences.

2. Points of Interest (POIs) including heritage sites, eateries, and attractions.

3. Generated itineraries with travel time predictions.

These details can be updated, modified, or deleted by users or system administrators to keep the information accurate and up to date.

1.3 Intended Audience and Document Overview

The intended audience of this document includes:

1. Tourists/Users who will use the platform to create, customize, and share itineraries.

2. Developers and designers who will implement, maintain, and enhance the platform.

3. Faculty members and evaluators who will assess the project in terms of software engineering principles.

4. Future contributors who may extend the system with additional features or datasets.

The audience precisely will be:

1. Tourists/Travelers who sign up, register, and generate itineraries using the system.
2. Platform administrators who maintain Points of Interest (POIs) and ensure reliable performance.
3. Academic evaluators who use the SRS to understand the project scope and requirements.

Document Overview:

- Section 1 introduces the product purpose, scope, intended users, and conventions.
- Section 2 provides an overall description of the system context and characteristics.
- Section 3 details functional and interface requirements.
- Section 4 covers non-functional requirements including performance, security, and quality attributes.

1.4 Definitions, Acronyms and Abbreviations:

SR. NO	TERM	DESCRIPTION
1	SRS	Software Requirement Specification
2	POI	Point of Interest (tourist attraction, restaurant, heritage site, etc.)
3	ETA	Estimated Time of Arrival
4	API	Application Programming Interface
5	ML/DL	Machine Learning / Deep Learning (used for recommendations & predictions)

1.5 Document Conventions

- **Heading:** Font Size: 26, Font Style: Bold, Font: Times New Roman
- **Sub-Heading:** Font Size: 12, Font Style: Bold, Font: Arial
- **Content:** Font Size: 11, Font: Arial
- Numbering follows **IEEE standard SRS format** (1, 1.1, 1.1.1)

1.6 References and Acknowledgements

References

1. OpenAI. (2025). GPT Models for Natural Language Understanding and Assistance.
2. Google Maps API – For geolocation and navigation data.
3. OpenWeatherMap API – For real-time weather information.
4. Airbnb API – For accommodation recommendations in Mumbai.
5. Kaggle Tourism & Travel Datasets – For user preference analysis and ML model training.
6. Government of Maharashtra Tourism Portal – For authentic details about tourist spots in Mumbai.
7. Research literature on Recommender Systems and AI-driven travel planning.

Acknowledgments

We would like to express our sincere gratitude to our project mentors and faculty members for their constant guidance and support during the development of the Mumbai Trails project.

We extend our appreciation to the creators of APIs and datasets, which played a key role in integrating real-time travel data. Special thanks to OpenAI tools for assisting in research, ideation, and documentation.

Finally, we would like to acknowledge our peers and teammates for their active collaboration, motivation, and contributions, without which this project would not have been possible.

2 Overall Description

2.1 Product Perspective

Currently, travelers who wish to explore Mumbai must manually search for tourist spots, dining options, and activities through different platforms. They also have to manually plan routes, estimate travel times, and adjust schedules according to weather and traffic.

Moreover, travelers often face difficulty in selecting points of interest (POIs) suited to their mood, budget, and time. They also have to constantly search for updated information such as operating hours, travel delays, or local events.

Hence, the MumbAI Trails system would maintain a structured and intelligent database of POIs, tourist profiles, itineraries, and contextual information (weather, traffic) that would save travelers time, effort, and confusion. The system leverages AI to personalize itineraries, predict travel times, and recommend optimized travel routes, making the tourism experience smarter and more enjoyable.

2.2 Product Functionality

The MumbAI Trails system is to be developed as an attempt to maintain records of tourist preferences and POIs by building a comprehensive AI-powered platform.
The system will provide the following functionality:

1. Tourists can register and create their personal travel profiles including mood, budget, preferred categories, and available time.

2. The system will generate personalized itineraries suggesting attractions, dining spots, and activities.

3. The system will also be able to search for attractions based on user specifications such as mood (relaxed, adventurous, foodie, heritage), location, and time.

4. The generated itineraries will include travel time predictions and optimized travel routes considering real-time weather and traffic.

5. Tourists will also be able to save itineraries, share them via links/QR codes, or export them as PDF files.

6. A chatbot assistant will allow tourists to plan using natural language commands.

7. The system will provide statistics and analytics of popular spots, frequently chosen itineraries, and past user feedback to improve recommendations.

2.3 Users and Characteristics

The major User classes in the System would be:

Tourist/User

1. A new tourist must register/sign up to provide travel preferences and details.
2. They can create and update their personal profile (preferred themes, budget, pace of travel).
3. They can generate itineraries, adjust plans manually, and save or share them.
4. They can provide feedback and ratings after using the system.

Administrator

1. Administrators have full control over the system.
2. They can add, update, or delete Points of Interest (POIs).
3. They are responsible for maintaining and updating the system's database.
4. They may analyze user feedback to improve recommendations.

Company/Service Provider (future scope)

1. Hotels, restaurants, and event organizers can register on the platform.
2. They can notify the Administrator about upcoming events, offers, or attractions.
3. They may promote or recommend activities that could be integrated into itineraries.

2.4 Operating Environment

The **MumbAI Trails** web application can be deployed on **Linux or Windows servers**.

- **Server Requirements:**
    - Minimum RAM: 2 GB
    - Storage: 50 GB
    - Quad-Core Processor

- Internet Connectivity with API keys configured

- **Client Requirements:**
  - Device: Smartphone, Tablet, or Desktop with Internet access
  - Browser: Chrome (v90+), Firefox (v80+), Safari (v14+), or Edge (v90+) with HTML5 & JavaScript support
  - Responsive design for both desktop and mobile devices

2.5 Design and Implementation Constraints

Design Constraints:

1. **Security:** All user data, itineraries, and POI information must be secured against unauthorized access or modification.
2. **Fault Tolerance:** In case of system crash or power failure, saved itineraries and user data must not be lost.
3. **API Quotas:** The system depends on third-party APIs like Google Maps and OpenWeather, which may limit request frequency.

2.6 User Documentation

User documentation components such as user manuals, online help, and tutorials will be delivered along with the software. A “How It Works” guide will be available for first-time users.
Support forms and bug-reporting features will also be provided. Documentation will be available as:

1. HTML help pages integrated into the system
2. Downloadable user manual in PDF format
3. FAQs and tutorials accessible via the website

2.7 Assumptions and Dependencies

1. It is assumed that the user has basic knowledge of using web applications.
2. Travelers must have an active internet connection to generate itineraries and access live data (traffic, weather).
3. The system depends on external APIs (Google Maps, OpenWeather, Firebase) for accurate functioning.
4. Users are assumed to have devices (smartphones/desktops) capable of accessing modern browsers.

3 Specific Requirements

3.1 External Interface Requirements

User Interfaces

- **Registration Window:**

1. **User:** Tourist / Administrator

2. **Properties:**
   This window is used for the entry of tourist details and registering a new tourist in the MumbAI Trails system.
   The window has text fields to take the information such as name, email ID, age, gender, preferred travel categories (heritage, food, shopping, adventure, relaxation), mood preferences, budget, and time availability.

- **Login Window:**

1. **User:** Tourist, Administrator

2. **Properties:**
   This window has two fields for username and password, and two buttons for Login or Sign Up.
   For the correct username and password, it opens the appropriate homepage (Tourist Dashboard / Admin Dashboard).
   It also has a “Sign Up” button to let new tourists register by redirecting to the Registration window.

- **Tourist Homepage:**

1. **User:** Registered Tourist

2. **Properties:**
   This dashboard opens when a tourist logs in.

   - Option to create/update their profile (preferences, mood, budget, etc.).
   - Generate AI-powered personalized itineraries.
   - Modify itineraries manually (add/remove attractions).
   - Save itineraries and export them as PDF or share via QR code/link.
   - Chatbot interface for natural language itinerary planning.
   - View weather, traffic predictions, and estimated travel times.

- **Administrator Homepage:**

1. **User:** Administrator of the platform
2. **Properties:**
   This dashboard opens when the administrator logs in.

    - Administrator provides approval for new Points of Interest (POIs) added.
    - Responsible for maintaining and updating the system database (attractions, restaurants, events).
    - Update upcoming events, local festivals, or special offers.
    - Notify tourists regarding new POIs or travel advisories.
    - Analyze user feedback and usage patterns.

- **Hardware Interfaces**

The program will communicate with the hard drive (filesystem and database) via the backend server. The user will interact through a web browser using keyboard/mouse/touchscreen, with the graphical interface displayed on the user's screen.

The system will run on a server with at least 2 GB RAM, 50 GB storage, stable Internet connectivity, and API access configured. Client devices include smartphones, tablets, or desktops.

- **Software Interfaces**

1. The software runs under Linux or Windows operating systems.
2. Database: Centralized storage for user profiles, POIs, itineraries, and logs.
3. APIs: Google Maps API (routes), OpenWeather API (weather), Firebase/SQL (storage).
4. Backend: Node.js / Python Flask.
5. Frontend: React-based responsive web application.

- **Communication Interfaces**

The system communicates over the Internet using HTTPS/HTTP protocols.

1. Communication between frontend and backend via REST APIs.
2. Authentication handled via OAuth/JWT tokens.

3. Communication security with SSL/TLS encryption.

4. Email notifications sent via integrated mail services.

3.2 Functional Requirements

Register Tourist:

Collect Personal Information of Tourist

1. **Input:** Name, age, gender, email ID, password, contact number.

2. **Output:** Updated tourist details stored temporarily in the database.

3. **Description:** The entered details must not completely match with an existing entry in the tourist database.

Collect Preference Details of Tourist:

1. **Input:** Preferred categories (heritage, adventure, food, shopping), budget, time availability, travel pace, mood.

2. **Output:** Updated preference details stored in the database, converted from temporary to permanent record.

3. **Description:** These details are permanently stored for generating personalized itineraries.

Generate Itinerary:

AI-based Recommendation

1. **Input:** Tourist preferences (categories, mood, budget, time).

2. **Output:** List of recommended Points of Interest (POIs).

3. **Description:** The AI model selects appropriate POIs using user profile data.

Route Optimization

1. **Input:** Selected POIs, Google Maps API data, real-time traffic.

2. **Output:** Optimized travel route with ETA (Estimated Time of Arrival).

3. **Description:** Travel sequence is adjusted to minimize travel time and maximize experience.

Save and Share Itinerary

1. **Input:** Tourist's selected itinerary.

2. **Output:** Saved itinerary displayed in dashboard, option to download PDF or share via QR code.
3. **Description:** Allows tourists to store and reuse itineraries across trips.

Update Upcoming Events:

Updating City Events Information

1. **Input:** Event name, description, organizer, time, date, venue, category.
2. **Output:** Updated event information displayed on the dashboard.
3. **Description:** Admin stores the relevant data in the events database.

Chatbot Interaction:

1. **Input:** Tourist query in natural language (e.g., “Plan me a relaxing day with heritage and street food”).
2. **Output:** Recommended itinerary displayed.
3. **Description:** NLP engine processes the request and integrates with itinerary generation.

Apply Filters and Adjust Plans

1. **Input:** Tourist applies filters (budget, category, distance, opening hours).
2. **Output:** Adjusted itinerary with new recommendations.
3. **Description:** The system dynamically updates recommendations.

Feedback and Rating

1. **Input:** Tourist feedback/rating for visited attractions.
2. **Output:** Stored feedback in database, shown in analytics.
3. **Description:** Used to improve recommendations and highlight popular POIs.

4 Other Non-functional Requirements

4.1 Performance Requirements

1. The users must get responses within **a few seconds**; i.e., the response time of itinerary generation, route optimization, and event search should be minimized.

2. The system would exhibit **high performance** because it will be optimized with efficient AI models and caching strategies for frequently used data such as POIs and routes.

3. The business logic is completely separated from the user interface on the server side, ensuring smooth processing and scalability.

4. API calls (Google Maps, OpenWeather) must be handled asynchronously to avoid delays in user interaction.

4.2 Safety and Security Requirements

1. This software offers **password protection and secure login** so that tourists can access their accounts, update preferences, and save itineraries privately.

2. User authentication will be handled with **encrypted tokens (JWT/OAuth)**, preventing unauthorized access.

3. Sensitive data such as passwords will be **encrypted before storage** in the database.

4. All communication between client and server will be secured with **SSL/TLS encryption**.

5. The system will also implement **role-based access control** where only administrators can update POIs and events.

4.3 Software Quality Attributes

1. The software must have a **simple and user-friendly interface** that is intuitive for tourists of all age groups.

2. The navigation to various features (registration, itinerary generation, event updates, profile management) should be **smooth, quick, and consistent**.

3. The system must be **responsive and mobile-friendly**, allowing access on both smartphones and desktops.

4. The design will ensure **consistency in fonts, layouts, and icons** for better usability.

5. The platform should be **maintainable and extensible**, so future contributors can easily add new POIs, categories, or features.