Metheia

Travel Agency Web Application

Software Requirements Specification 1.0

Prepared by

Student ID	Name
22201024	Sumaita Mahdiat
22201890	Atika Tabassum
22241011	Sanchita Sarker Puja
22201094	Rubaia Islam Shimu

1. Introduction

1.1 Purpose

The functional and non-functional requirements for Metheia, an online application that suggests trip destinations, are outlined in this Software Requirements Specification document. Through an interactive quiz, the system gathers user preferences, makes travel destination recommendations, and shows comprehensive travel details, including well-known sights, lodging, dining options, local guides, transportation choices, cultural insights, and estimated budgets that are dynamically retrieved from a database.

1.2 Scope

Metheia includes the design, development, testing, and deployment of a full-stack web application that:

- Presents an interactive quiz to collect user preferences (budget, region, activity type, climate, etc.).
- Suggests matching destinations dynamically using stored quiz logic.
- Displays comprehensive destination details, including attractions, restaurants, hotels, and transportation.
- Enables search and filtering of destinations.
- Supports hotel and guide booking features with secure payment options.
- Allows users to provide feedback for future improvements.

1.3 Definitions, Acronyms, and Abbreviations

- MERN - MongoDB, Express.js, React.js, Node.js
- UI/UX - User Interface / User Experience
- CRUD - Create, Read, Update, Delete

1.4 References

MERN Stack Documentation

- MongoDB
- Express.js
- React.js
- Node.js
- Agile

---

2. Overall Description

2.1 Product Perspective

Metheia is a standalone, full-stack web system built using the MERN stack. The React.js frontend communicates with a Node.js + Express.js backend, which interfaces with MongoDB to fetch, insert, and update travel-related data dynamically. All data, such as destinations, hotels, and guides, is stored in collections.

2.2 Product Features

1. Dynamic homepage with featured destinations and travel offers.
2. An interactive multi-question travel quiz
3. Destination overview with description, image, and key facts.
4. Display of top tourist attractions for each destination.
5. List of best-rated local restaurants.
6. Display of available transportation options.
7. Estimated travel budget and average cost details.
8. Recommended travel seasons and weather insights.
9. Local language and currency display.
10. Smart packing checklist for the chosen destination.
11. Cultural etiquette and do's/don'ts guide.
12. Quiz reset and reattempt option
13. Hotel listings with filters (price, rating, location, amenities).
14. Real-time availability check and booking confirmation.
15. Secure payment gateway (Visa Card).
16. Hotel booking management dashboard (view, modify and cancel bookings).
17. Search and filter travel guides by language, location, or experience.
18. View guide profiles with hourly rates and specialties.
19. Book guides for specific dates/times and get confirmation.
20. Search functionality for manual destination exploration.

2.3 User Classes and Characteristics

- **Traveler/User:** End-users who take quizzes, view destinations, and book hotels/guides.
- **Admin (Travel Agency Staff):** Users with privileges to manage destinations, hotels, guides, and quiz data through a secure admin panel.

2.4 Operating Environment

- **Frontend:** React.js web app running in the browser
- **Backend:** Node.js with Express.js

- **Database:** MongoDB
- **Hosting:** Deployment could be on AWS, Azure, Google Cloud, Render, or any equivalent production environment.

2.5 Constraints

- Must load quiz results within 2 seconds.
- The project is to be delivered in approximately 8–9 weeks with a team of four developers.
- No AI functionalities.

2.6 Assumptions and Dependencies

- MongoDB is accessible at all times.
- All destination data (spots, hotels, etc.) is pre-inserted into collections.
- An internet connection is required for real-time data fetching.

---

3. System Requirements

3.1 Functional Requirements

Homepage and Quiz

- FR-1: Display homepage with featured destinations and offers.
- FR-2: Fetch quiz questions and options from MongoDB.
- FR-3: Process user quiz responses and query matching destinations.
- FR-4: Allow users to retake or reset the quiz.

Destination & Information Display

- FR-5: Retrieve and display destination details, including images, description, and key facts.
- FR-6: Display top tourist attractions and local restaurants.
- FR-7: Show available transport modes and average costs.
- FR-8: Present hotel listings with filters (price, rating, etc.).
- FR-9: Show weather, travel seasons, and budget insights.
- FR-10: Display local language, currency, and cultural etiquette.
- FR-11: Provide packing checklist and safety tips.

Search & Booking

- FR-12: Allow search and filtering by climate, region, and budget.
- FR-13: Enable hotel and guide bookings with confirmation.

- FR-14: Support secure payment transactions.
- FR-15: Allow admin to add, update, or remove destinations, hotels, and guides.

3.2 Non-Functional Requirements

- Must return responses within 2 seconds under normal load.
- Must be simple to use and visually appealing.
- Allow the addition of new destinations, travel options, hotels, or quiz questions without code refactoring.

---

4. Technology Stack & Architectural Overview

4.1 MERN Stack Components

- **MongoDB:** Stores quiz data, destinations, hotels, restaurants, transportations, etc.
- **Express.js:** Handles routing
- **React.js:** Frontend for dynamic quiz, search, and results display.
- **Node.js:** Backend runtime

4.2 High-Level Architecture

- **Presentation Layer (React.js):** User-facing interface for quiz and travel display.
- **Business Logic Layer (Node.js/Express):** Processes quiz results
- **Data Layer (MongoDB):** Stores destinations, places, hotels, and restaurants

---

5. Acceptance Criteria

- At least one destination must be suggested based on user quiz responses.
- Destination pages must display attractions, hotels, and restaurants from the database.
- Search, filters, and booking features must function dynamically.
- All database operations (CRUD) must complete successfully with valid data

---

7. Conclusion

Metheia is a MERN-based travel destination web application that connects frontend and backend layers and uses MongoDB collections to deliver personalized, data-driven travel

experiences. It provides a responsive and user-friendly interface, ensuring smooth interaction, scalability, and easy database integration, ideal for a modern travel agency solution.