Software Requirements Specification (SRS)

Contents

1. Introduction

a. Purpose:

The purpose of this document is to provide a comprehensive overview of the requirements and specifications for the Thomas Cook Website Clone Project. It defines the scope, features, and architecture of the software system.

b. Document Conventions:

i. All requirements are stated in a clear and concise manner.

ii. Functional requirements are denoted with an "FR" prefix, and non-functional requirements with an "NFR" prefix.

c. References:

i. *Thomascook.in offers Tours and Travels, Flight Bookings, Hotels, Forex, Visa & lot more!* (n.d.). https://www.thomascook.in/

ii. *Spring boot*. (n.d.). Spring Boot. https://spring.io/projects/spring-boot

iii. *Spring Security*. (n.d.). Spring Security. https://spring.io/projects/spring-security

iv. *React*. (n.d.). https://react.dev/

v. GeeksforGeeks. (2023). Axios in React A Guide for Beginners. *GeeksforGeeks*. https://www.geeksforgeeks.org/axios-in-react-a-guide-for-beginners/

vi. Atlassian. (n.d.). *DevOps Pipeline | Atlassian*. https://www.atlassian.com/devops/devops-tools/devops-pipeline

2. System Overview

a. **System Description:**

i. The Thomas Cook Website Clone is a web-based travel booking platform that aims to replicate the functionality and user experience of the original Thomas Cook website. It enables users to search and book hotels, flights, holiday packages, and access static pages for information about the company and gift card purchases.

b. **System Features**:

The main features of the system include:

a. Hotel Booking
b. Flight Booking
c. Holiday Booking
d. User Registration and Login
e. Static Pages (About Us and Gift Cards)
f. Integration with Payment Gateway (RazorPay)
g. User Profiles and Bookings

c.

3. Functional Requirements

a. **Use Case Descriptions:**

Use Case 1: Hotel Booking

- **Actors**:
  - Registered User: The user who is logged in and wants to book a hotel.
  - Guest User: A user who is not logged in but wants to search for hotels.
- **Preconditions**:
  - The user is on the hotel booking section of the website.
- **Postconditions**:
  - A hotel room is booked, and the user receives a booking confirmation.
- **Steps**:
  1. The user selects the desired location, check-in and check-out dates, and room preferences.
  2. The system verifies the availability of rooms for the specified dates and displays results.
  3. The user selects a hotel and room type.
  4. The user provides personal information, including name, contact details, and payment information.

5. The system confirms the booking and generates a booking confirmation for the user.

Use Case 2: Flight Booking

- **Actors**:
  - Registered User: The user who is logged in and wants to book a flight.
  - Guest User: A user who is not logged in but wants to search for flights.
- **Preconditions**:
  - The user is on the flight booking section of the website.
- **Postconditions**:
  - A flight is booked, and the user receives an electronic ticket.
- **Steps**:
  1. The user specifies the origin, destination, travel dates, and passenger details.
  2. The system searches for available flights and displays options.
  3. The user selects a flight and seat preferences.
  4. The system confirms the booking and issues an electronic ticket to the user.

Use Case 3: Holiday Booking

- **Actors**:
  - Registered User: The user who is logged in and wants to book a holiday package.
  - Guest User: A user who is not logged in but wants to search for holiday packages.
- **Preconditions**:
  - The user is on the holiday booking section of the website.
- **Postconditions**:
  - A holiday package is booked, and the user receives a detailed itinerary.
- **Steps**:
  1. The user specifies the destination, duration.
  2. The system searches for available holiday packages and displays options.
  3. The user selects a package.
  4. The user provides personal information, including name and contact details.
  5. The system confirms the booking and generates a detailed itinerary for the user.

Use Case 4: User Registration and Login

- **Actors**:
  - New User: A user who wants to create a new account.
  - Registered User: An existing user who wants to log in.
- **Preconditions**:
  - The user is on the registration or login page.
- **Postconditions**:
  - For registration, a new user account is created. For login, the user gains access to their account.
- **Steps**:
  1. For Registration:
     - The new user provides required registration information.
  2. For Login:
     - The user enters their login credentials.
     - The system verifies the credentials and grants access.

Use Case 5: Static Pages (About Us and Gift Cards)

- **Actors**:
  - Website Visitor: Any user or guest visiting the website.
- **Preconditions**:
  - The user is on the "About Us" or "Gift Cards" page.
- **Postconditions**:
  - For "About Us," the user gains information about the company. For "Gift Cards," the user can check gift card offers.
- **Steps**:
  1. For "About Us":
     - The user clicks on the "About Us" link.
     - The system displays information about the company, including its history, values, and mission.
  2. For "Gift Cards":
     - The user clicks on the "Gift Cards" link.
     - The system displays information about different gift cards.

b. Functional Requirements:

Functional requirements are enumerated as follows:

Requirement ID	Requirement Description
FR001	Users must be able to search for hotels by location, check-in/out dates, and preferences.
FR002	The system must verify the availability of hotel rooms for selected dates.
FR003	Users must be able to select and book hotel rooms, specifying room type.
FR004	Users should have the option to book holiday packages.
FR005	The system must generate success message for booked holiday packages.
FR006	Users must be able to register for accounts.
FR007	Users should be able to log in securely to access their profiles and booking history.

c. Non-functional Requirements:

Performance Requirements:

- **NFR001: Response Time**: The system must respond to user interactions within a few seconds for all critical functions, such as searching for hotels and booking flights.
- **NFR002: Scalability**: The system should be capable of handling a concurrent load of at least 10 users without significant degradation in performance.
- **NFR003: Availability**: The system must be available 99.9% of the time, allowing for scheduled maintenance and updates during off-peak hours.

Security Requirements:

- **NFR004: Data Encryption**: All sensitive user data, passwords, must be encrypted during transmission using industry-standard encryption protocols.
- **NFR005: Authentication**: User authentication must be secure, with password hashing and salting for storing user credentials.
- **NFR006: Access Control**: Role-based access control (RBAC) must be implemented to ensure that users only have access to authorized functionalities.

Usability Requirements:

- **NFR008: User-Friendly Interface**: The user interface must be intuitive and user-friendly, with clear navigation and responsive design for mobile devices.

- **NFR009: Accessibility**: The system must comply with web accessibility standards (e.g., WCAG) to ensure that it can be used by individuals with disabilities.

4. System Architecture

a. High-Level Architecture:

The high-level architecture of the Thomas Cook Website Clone Project is designed to provide a scalable, robust, and modular foundation for the entire system. It encompasses various components, each responsible for specific functionalities. The primary architectural components include:

Web Application: The front-end component responsible for user interactions. It includes the user interface (UI) and client-side logic developed using React.

Application Server: The back-end component that handles user requests, processes data, and communicates with the database. It is implemented using Java 17 and Spring Boot 3.1.3.

Database: The data storage component using MySQL 8.0.34. It stores user profiles, booking information, product data, and other essential information.

External Services Integration: The system integrates with external services for functionalities like payment processing.

Authentication and Authorization: This component ensures secure access to the system, handling user authentication and authorization using industry-standard protocols.

b. Component Descriptions:

Web Application (Front-end):

- **Description**: The web application is built using React, providing an interactive and responsive user interface. It communicates with the application server to fetch data and handle user interactions.

Application Server (Back-end):

- **Description**: The application server is implemented in Java 17 and Spring Boot 3.1.3. It serves as the core of the system, handling user requests, processing data, and ensuring secure communication with external services.

Database (MySQL):

- **Description**: MySQL 8.0.34 is used as the relational database management system. It stores user profiles, booking data, product information, and other relevant data. The database schema is designed for efficient data retrieval and storage.

External Services Integration:

- **Description**: The system integrates with external services, including payment gateway like razorpay.

Authentication and Authorization:

- **Description**: This component is responsible for user authentication and authorization. It ensures that users have appropriate access permissions based on their roles and manages secure user sessions.

5. Data Requirements

a. Data Entities:

The Thomas Cook Website Clone Project relies on several data entities to store and manage user information, booking details, product information, and more. The primary data entities include:

1. **User Profile**: This entity stores user information such as name, email address, contact number, and authentication credentials. It includes fields for user preferences and settings.
2. **Hotel**: The hotel entity contains information about various hotels available for booking. It includes details like hotel name, location, room types, prices, and availability.
3. **Flight**: This entity stores information about flights, including flight numbers, departure and arrival locations, schedules, available seats, and prices.
4. **Holiday Package**: The holiday package entity contains details about various holiday packages, including destinations, durations, inclusions, prices, and customizable options.
5. **Booking**: This entity represents user bookings, whether for hotels, flights, or holiday packages. It includes booking IDs, user references, dates, and associated product details.

b. Data Dictionary

Below is a detailed description of each data entity, including its attributes and data types:

User Profile:

- **Attributes**:
  - User ID (Primary Key)

- First Name (String)
- Last Name (String)
- Email Address (String)
- Contact Number (String)
- Username (String)
- Password (Hashed String)
- Address (String)

Hotel:

- **Attributes**:
  - Hotel ID (Primary Key)
  - Hotel Name (String)
  - Location (String)
  - Prices (Text)
  - Availability (Boolean)
  - Description (Text)
  - Ratings (Float)

Flight:

- **Attributes**:
  - Flight ID (Primary Key)
  - Flight Number (String)
  - Departure Location (String)
  - Arrival Location (String)
  - Departure Date and Time (Timestamp)
  - Arrival Date and Time (Timestamp)
  - Available Seats (Integer)
  - Prices (Text)

Holiday Package:

- **Attributes**:
  - Package ID (Primary Key)
  - Destination (String)
  - Duration (String)
  - Description (Text)
  - Inclusions (Text)
  - Prices (Text)

Booking:

- **Attributes**:
  - Booking ID (Primary Key)
  - User ID (Foreign Key)
  - Product Type (String - Hotel, Flight, or Holiday Package)
  - Product ID (Foreign Key)
  - Booking Date (Timestamp)
  - Check-In Date (Timestamp - Hotel only)

- Check-Out Date (Timestamp - Hotel only)
- Passenger Details (Text - Flight and Holiday Package)
- Total Amount (Float)

6. User Interface Design

a. User Interface Mockups

1. **User-Friendly Design**: The user interface (UI) is designed with a focus on user-friendliness, ensuring that users of all levels of computer literacy can easily navigate and interact with the system.
2. **Responsive Design**: The UI is responsive and adapts seamlessly to different screen sizes and devices, including desktops, tablets, and smartphones, providing a consistent user experience.
3. **Intuitive Navigation**: Navigation menus and controls are intuitively placed to help users find what they need quickly. A user-friendly menu structure ensures efficient movement between different sections of the website.
4. **Search Functionality**: Prominent search functionality is provided, allowing users to search for hotels, flights, and holiday packages based on their preferences and travel plans.
5. **Interactive Booking Process**: The booking process for hotels, flights, and holiday packages is interactive, guiding users step by step and providing clear instructions to complete bookings.

b. **User Interaction**

1. **User Registration**: New users can easily register by providing required information.
2. **User Login**: Registered users can log in securely using their credentials. Password hashing and salting mechanisms are employed to protect user data.
3. **Browsing and Search**: Users, whether logged in or as guests, can browse and search for hotels, flights, and holiday packages based on various criteria such as location, dates, and preferences.
4. **Booking Process**: Booking a hotel room, flight, or holiday package involves a clear and guided process. Users provide necessary details, and the system validates information for accuracy.
5. **Payment**: Secure payment processing allows users to complete transactions confidently. Payment options, including credit card and other methods, are available.
6. **User Profiles**: Registered users have access to personalized profiles where they can view booking history, preferences, and account settings.
7. **Static Pages**: Users can access static pages, including "About Us" and "Gift Cards," to learn about the company and view available gift cards.

8. **Logout**: Users can log out securely when they have completed their interactions, ensuring the security of their accounts.

7. System Interfaces

i. External Interfaces:

The Thomas Cook Website Clone Project interfaces with external systems, APIs, and services to enhance its functionalities and provide real-time data to users. These external interfaces are integral to the system's performance and capabilities. Key external interfaces include:

1. **Payment Gateway Integration**:
    - **Description**: The system integrates with payment gateway (RazorPay) to facilitate secure and convenient payment processing for users. This includes credit card payments and other payment methods.
    - **Protocols**: HTTPS, Payment Gateway APIs (RazorPay)
    - **Data Formats**: JSON

ii. Communication Protocols

The following communication protocols and data formats are used for interactions with external systems and services:

1. **HTTPS (Hypertext Transfer Protocol Secure)**:
    - **Description**: HTTPS is used to secure data transmission between the system and external services. It ensures data privacy and encryption during communication.
2. **RESTful API (Representational State Transfer)**:
    - **Description**: RESTful APIs are employed for interactions with external services like flight availability and hotel booking. They allow for efficient, stateless communication through HTTP requests.
3. **JSON (JavaScript Object Notation)**:
    - **Description**: JSON is a widely used data format for exchanging information between the system and external services. It offers a lightweight and human-readable format for data transmission.