Personalized Travel Booking Platform (FlyMe2DaMoon)

Software Requirements Specification

V1

31 – 10 - 2024

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to outline the requirements for developing a Personalized Travel Booking Platform. This platform will provide a seamless, user-friendly experience for travelers to search, compare, and book various travel services, including flights, hotels, and car rentals. The SRS document serves as a guide for developers, project managers, and stakeholders to ensure alignment with user and business requirements.

1.2 Scope

The Personalized Travel Booking Platform aims to offer a comprehensive, personalized online booking experience that caters to diverse travel needs. The system will support functions like search, booking, user profile management, and payment processing. The primary goals are to enhance user satisfaction, increase revenue for the company, and strengthen brand presence.

1.3 Definitions, Acronyms, and Abbreviations

***SRS*: Software Requirements Specification**

***UI*: User Interface**

***UX*: User Experience**

***API*: Application Programming Interface**

1.4 References

-Software Development and Engineering Standards

- *IEEE Standard for Software Requirements Specifications (IEEE Std 830-1998).*
- *IEEE Standard for Software and System Test Documentation (IEEE Std 829-2008).*

-API Integration and Web Services

- *Official documentation for Google Maps API, Amadeus Travel API, or other travel-related APIs.*

- Security and Data Protection

- *General Data Protection Regulation (GDPR) guidelines.*
- *Payment Card Industry Data Security Standard (PCI DSS) for handling payment information.*

2. General Description

2.1 Product Perspective

The Personalized Travel Booking Platform is a standalone web application accessible via desktops, tablets, and mobile devices. It will integrate with third-party service providers through APIs for real-time information on flights, hotels, and car rentals.

2.2 Product Functions

- *Search and Browse: Users can search and browse travel options based on dates, budget, destination, and other filters.*
- *Booking and Reservation Management: Users can book a ticket for the travel with all related entities.*
- *User Profile Management: Users can create and manage their profiles, including saving travel preferences and booking history.*
- *Payment Processing: Secure, integrated payment gateway for booking transactions.*
- *Notifications and Alerts: The system will send booking confirmations, updates, and reminders via email OR SMS.*

2.3 User Characteristics

-End Users: Travelers who use the platform for booking and managing travel arrangements.

-Admin: Internal users who manage content, handle customer service, and oversee booking transactions.

-Others : Viewers and Stakeholders

2.4 General Constraints

-Security: Compliance with security protocols and data protection laws.

-Scalability: The system must support up to 1000 concurrent users.

-Payment Processing: Integration with secure, PCI-compliant payment gateways.

2.5 Assumptions and Dependencies

-Assumes that all users have internet access.

-Depends on the availability and reliability of third-party APIs for booking services.

3. Specific Requirements.

3.1 External Interface Requirements

3.2 Functional Requirements

Search and Browse for Travel Options

- Description: Allows users to search for travel options using filters like destination, budget, and travel dates.
- Inputs: User-specified filters for search criteria.
- Outputs: List of matching travel options with pricing and availability.

Booking Management

- Description: Users can book travel services and manage their reservations.
- Inputs: Booking details, user preferences, and payment information.
- Outputs: Booking confirmation and itinerary.

User Profile Management

- Description: Users can create, update, and manage personal profiles.
- Inputs: Profile information such as name, contact details, and travel preferences.
- Outputs: Updated user profile and personalized booking recommendations.

Payment Processing

- Description: Secure payment gateway for completing bookings.
- Inputs: Payment details from the user.
- Outputs: Payment confirmation and booking finalization.

Notifications and Alerts

- Description: Sends users notifications for booking confirmations, cancellations, and reminders.
- Inputs: Booking details and user contact information.
- Outputs: Notifications via email or SMS.

3.3 Use Case

3.4 Classes / Objects

. User Class :

Handles user registration, login, profile management, and travel recommendations. ***(Related to: User Profile Management, Search & Browse)*****.**

. Booking Class :

Manages booking creation, updates, cancellations, and payment confirmations. ***(Related to: Booking Management, Payment Processing)*****.**

. Payment Class :

Processes payments, refunds, and generates invoices securely. ***(Related to: Payment Processing, Security Requirements)*****.**

. Notification Class :

Sends booking alerts and payment reminders via email/SMS. ***(Related to: Notifications & Alerts)*****.**

.

3.5 Non-Functional Requirements

- **Performance Requirements**
  - Description: The system must handle up to 1000 concurrent users without delay, with page load times of less than 3 seconds.
- **Security Requirements**
  - Description: Secure login, encrypted data transmission, and compliance with data protection standards.
- **Usability Requirements**
  - Description: User-friendly and accessible UI that caters to a wide range of user backgrounds.
- **Reliability Requirements**
  - Description: Ensures 99%uptime for consistent service availability.

- Availability Requirements
- Description: The platform should be available 24/7.
- Exception : Off if there is maintenance .

- **Maintainability Requirements**
  - Description: Code should be modular and easy to update without major overhauls.

3.6 Inverse Requirements
3.7 Design Constraint
3.8 Logical Database Requiremens
3.9 Other Requirements

4. Analysis Models

4.1 Sequence Diagrams

4.3 Data Flow Diagrams (DFD)

4.2 State-Transition Diagrams (STD)

5. Change Management Process

A. Appendices

A.1 Appendix 1
A.2 Appendix 2

Requirements Engineering Process

1-Feasibility Study

1. Technical Feasibility

- **Modern Technology: The platform can be developed using current web technologies such as HTML5, CSS3, JavaScript frameworks (e.g., React, Vue.js), and server-side technologies like Node.js or Django.**
- **API Integration: Integration with third-party APIs (e.g., for flights, hotels, and car rentals) is achievable. Challenges like data consistency and API rate limits can be mitigated with proper planning and by using API management tools.**
- **Resources Needed: The team will consist of both front-end and back-end developers. Given that the platform is not for real-world use, open-source tools and free-tier cloud services (such as AWS Free Tier or Heroku) will be utilized to reduce costs.**
- **Database Requirements: The platform will use a relational database (like MySQL or PostgreSQL) or a NoSQL option (such as MongoDB) to manage user data, bookings, and other platform information.**

2. Economic Feasibility

- **Initial Investment: The initial investment is minimal and primarily for domain registration, cloud hosting, and incidental expenses. Estimated between $X - $X.**
- **Revenue Model: Since the project is not for real-world use, no actual revenue will be generated. However, a simulated revenue model can be used to demonstrate how income could be generated through booking commissions, premium subscriptions, and advertisements.**

3. Operational Feasibility

- **Training and Skill Development: The development team needs to stay updated on the latest technologies and be trained on the platform's features. Online resources and training programs can be leveraged to enhance the team's skills.**
- **Team Requirements: The project will require a team of developers skilled in web development, databases, and UI/UX design, all of whom will be guided by instructors. Project management tools such as GitHub and Trello will help organize tasks and ensure smooth development.**
- **Partnerships: No real-world partnerships with travel providers will be required. Instead, mock data or free APIs will be used to simulate the booking process.**
- **Infrastructure and Scalability: The platform will be hosted on cloud services like AWS, Google Cloud, or Azure using free tiers, which will provide enough resources for the project's workload. Scalability options and basic disaster recovery plans will be considered in case the project is later scaled or deployed in a real-world setting.**

2. Requirements Elicitation and Analysis

2.1. Functional Requirements

- **Search and Browse for Travel Options**

  Description: Users can search for travel options based on filters such as destination, budget, and travel dates.
  Inputs:

  - Destination
  - Travel dates
  - Budget range
  - Filters for travel preferences (e.g., hotel type, transportation mode) **Outputs**:
  - List of available travel options that match the user's criteria. Pricing and availability information for each option.

- **Booking Management**

  Description: Users can book travel services (e.g., flights, hotels, car rentals) and manage reservations.
  Inputs:

  - User's selected travel options (e.g., destination, dates).
  - User preferences (e.g., seat class, hotel amenities).
  - Payment information for booking.

  Outputs:

  - Confirmation of booking.
  - A detailed itinerary.
  - A summary of booked services.

- **User Profile Management**

  Description: Users can create and manage their personal profiles, update contact details, and save travel preferences.
  Inputs:

  - User's personal information (e.g., name, contact details).
  - Travel preferences (e.g., preferred airlines, seat class).

  Outputs:

  - A customized profile.
  - Personalized travel recommendations.

- **Payment Processing**

  Description: Secure payment gateway for processing bookings.
  Inputs:

  - Payment details (e.g., credit card or debit card information). Billing information.

  Outputs:

  - Payment confirmation. Booking finalized.

- **Notifications and Alerts**

  Description: Sends notifications to users for important updates, such as booking confirmations, cancellations, and reminders.

  Inputs:

  - Booking details.
  - User contact information (e.g., email or phone number).

  Outputs:

  - Email or SMS notifications for booking status updates.
  - Alerts for upcoming travel dates, cancellations, or changes to the booking.

2.2. Non-Functional Requirements

The non-functional requirements describe how the system should perform in terms of attributes like performance, security, and usability:

- **Performance Requirements**:

  The system must be able to handle up to 1000 concurrent users without performance degradation.

  Page load times must be under 3 seconds to ensure a smooth user experience.

- **Security Requirements**:

  The platform must implement secure login procedures.

  All data (especially payment and user information) must be encrypted during transmission.

  The platform should comply with relevant data protection and privacy standards (e.g., GDPR or similar regulations).

- **Usability Requirements**:

  The system's interface should be user-friendly and accessible to a wide range of users, including those with limited technical knowledge.

  The platform should be designed to accommodate diverse user backgrounds, ensuring accessibility and ease of navigation.

- **Reliability Requirements**:

  The platform should ensure a 99% uptime to provide consistent service availability to users.

  Backup systems must be in place to prevent data loss.

- **Availability Requirements**:

  The platform should be available 24/7, with downtime only during scheduled maintenance.

  Maintenance windows should be communicated in advance to users.

- **Maintainability Requirements**:

  The system's codebase should be modular and easy to update.

  Future changes or updates to the platform should not require major overhauls, ensuring flexibility and long-term sustainability.

2.3. User Requirements

1. **User Registration and Login**
   Users should be able to create an account with a username, password, and email.
   Users should be able to log in using their credentials.
   Users should have the option to reset their passwords if forgotten.
2. **Profile Management**
   Users should be able to create and update their personal profiles with details like name, contact information, and travel preferences.
   Users should have the ability to view and edit their saved travel preferences.
3. **Search and Browse for Travel Options**
   Users should be able to search for travel options using filters such as destination, budget, and travel dates.
   Users should be able to view a list of available travel options with details like pricing, availability, and reviews.
   Users should have the ability to sort and filter search results based on criteria like price, rating, and availability.
4. **Booking Management**
   Users should be able to book flights, hotels, or car rentals through the platform.
   Users should be able to view, modify, or cancel their bookings.
   Users should receive a booking confirmation and itinerary after completing a reservation.
5. **Payment Processing**
   Users should be able to securely pay for their bookings using credit/debit cards or other payment methods.
   Users should receive a payment confirmation and receipt after completing a transaction.
6. **Notifications and Alerts**
   Users should receive notifications for booking confirmations, cancellations, and reminders via email or SMS.
   Users should have the option to enable or disable notifications.

7. **User-Friendly Interface**

   The platform should be easy to navigate with a clear and intuitive user interface. Users should have access to customer support or a help section for assistance.

2.4. System Requirements

. Functional System Requirements

1. **Authentication and Authorization**

   The system must support secure user authentication (e.g., via password hashing and secure session management).
   The system should implement role-based access control (e.g., users vs. administrators).

2. **Data Management**

   The system must store user profiles, booking details, and payment information in a secure database. The platform should use a relational database (e.g., MySQL, PostgreSQL) or NoSQL database (e.g., MongoDB) to manage data.

3. **API Integration**

   The system must integrate with third-party APIs for fetching travel data (e.g., flights, hotels, car rentals). The system should handle API rate limits and data synchronization to ensure data consistency.

4. **Payment Gateway**

   The system must integrate with a secure payment gateway for processing transactions.
   The system should support multiple payment methods and ensure PCI DSS compliance for secure transactions.

5. **Notification Service**

   The system should have a notification module to send emails or SMS alerts to users regarding booking confirmations, cancellations, and reminders.
   The platform should support customizable notification templates.

6. **Search and Filter Functionality**

   The system should support efficient search algorithms to handle user queries and filters.
   The system must provide sorting and filtering options to improve the user search experience.

. Non-Functional System Requirements

1. **Performance Requirements**

   The platform must handle up to 1000 concurrent users with a response time of less than 3 seconds. The system should optimize page load times, aiming for less than 3 seconds per page.

2. **Security Requirements**

   All user data must be encrypted during transmission using SSL/TLS protocols.
   The platform should implement secure coding practices to protect against common vulnerabilities (e.g., SQL injection, XSS attacks).
   The system should include logging and monitoring to detect and respond to security breaches.

3. **Usability Requirements**

   The user interface must be intuitive, accessible, and responsive across different devices (desktop, tablet, mobile).
   The platform should follow best practices for UX design to ensure a positive user experience.

4. **Reliability and Availability Requirements**

   The platform must achieve 99% uptime, ensuring continuous availability for users.
   The system should have a backup and disaster recovery plan in place to prevent data loss.

5. **Scalability Requirements**

   The system architecture should support horizontal and vertical scaling to accommodate future growth. The platform should be designed to handle increased traffic and data volume as user demand grows.

6. **Maintainability Requirements**

   The codebase should be modular and well-documented to facilitate easy updates and maintenance.
   The system should support automated testing and deployment processes to streamline development.

7. **Compliance Requirements**

   The platform must comply with data privacy regulations (e.g., GDPR) to protect user information. The system should have user consent mechanisms for data collection and processing.

3. Requirement Specification

1. Functional Requirements:

✧ Home Page

Description: The main landing page provides users with an overview of available travel options and promotions.

Features:

Header: Contains navigation links to other sections of the platform.

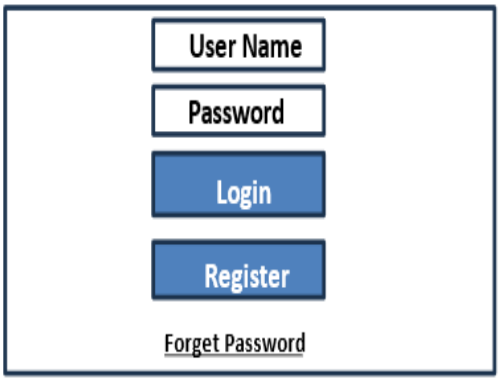
Search Box: Allows users to search for travel options by destination, travel date, and type (flight, hotel, package).

Promotions Section: Displays featured travel deals and special offers.

Footer: Contains links to other pages on the platform, along with social media links.

✧ User Registration & Login

Diagram:



Components:

Registration Form: Collects details like email (username), password, and phone number.
Login Form: Requires email (username) and password.
Account Management Allows updating information and password reset.

✧ Travel Search & Booking

. Package Content (e.g..) :

Offer	Travel Type	Destination	Date	Price
1	Flight	Paris	12/1/2024	$300
2	Hotel	New York	25/2/2024	$150
3	Package	Tokyo	30/1/2024	$500
4	Flight	London	4/2/2024	$250
5	Hotel	Dubai	3/5/2024	$200

BOOK

Booking Steps:

1. **Search Filters:**Refine by criteria (e.g., destination, date).
2. **Select Offer** Review travel option details.
3. **Proceed to Booking:**Complete booking form.
4. **Confirmation:** Displays booking summary.

✧ Payment Gateway

. **Components:**

Payment Options: Supports multiple payment methods like credit card, debit card, and digital wallets

Security features: Ensures user payment information is handled securely, with SSL encryption.

Transaction History: Users can view past payment transactions and download receipts.

Payment Method	Security features	Transaction History
Credit Card	SSL Encryption	-
Debit Card	SSL Encryption	-
Digital Wallet	SSL Encryption	-

Submit

2. Non-Functional Requirements:

✧ Usability

Description: The platform must be user-friendly and accessible for users of various ages and tech proficiency.
Details: Intuitive Navigation: Easy access to all sections of the platform through a clear, structured layout .
Responsive Design Compatible across devices (desktop, tablet, mobile).

✧ Performance

Description: The platform should offer fast and efficient performance.
Details : Load Time: Pages should load within 2 seconds under standard internet conditions .
Scalability: Capable of handling up to 10,000 users concurrently without downtime.

✧ Security

Description: User data and transactions must be secure .
Details: Data Protection: Uses encryption for sensitive data, including passwords and payment details.
Compliance: Adheres to GDPR for data privacy.

✧ Reliability

Description: The platform must function reliably under all conditions .
Details : Availability: 99.9% uptime, except for scheduled maintenance .
Backup: Daily data backup to ensure recovery in case of failure.

4-Requirements validation for a Personalized Travel Booking Platform

• Purpose

- Verify all requirements are well-defined, consistent, and achievable with available resources.
- Involves stakeholders and technical experts to review and approve each requirement

•

. Key Steps in Requirements Validation

- Review Requirements - Meeting with stakeholders to confirm details and completeness.
- Check Consistency - Ensuring there are no conflicts or unclear points.
- Feasibility Check - Verify that each requirement is possible to build.
- Resolve Conflicts - Clear up any issues or resolving ambiguities
- Output: Validated requirements, ready for the final document.

Validation of Functional Requirements

- The functional requirements focus on the essential operations and user interactions within the platform. Each functional requirement has been checked for clarity, feasibility, and alignment with user needs.

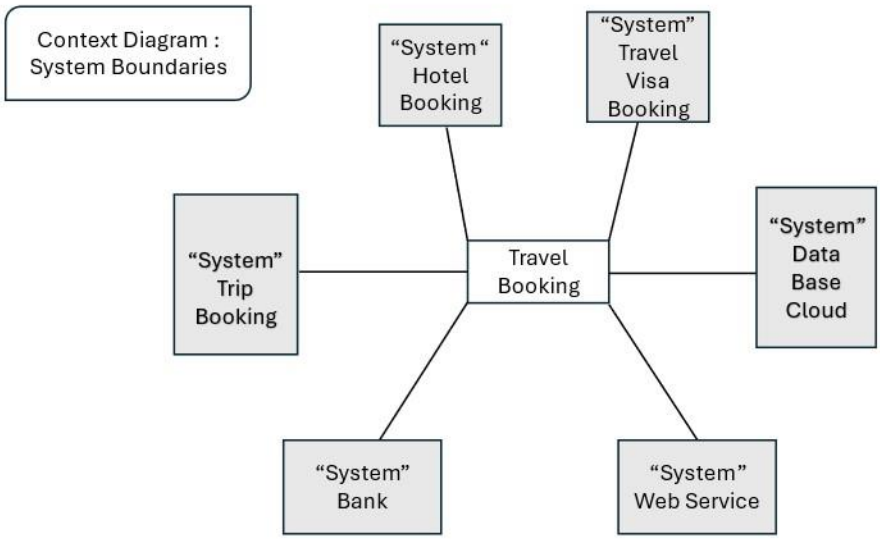
- User Registration and Authentication: Users should be able to sign up using either an email address or social media accounts.
- Personalized User Dashboard: Allows users to save and manage travel preferences
- Search and Booking : Allow users to filter and sort search results by options like price, ratings, and availability.
- Notifications and Alerts : Send real-time notifications for price changes, booking confirmations.

Validation of Non-Functional Requirements

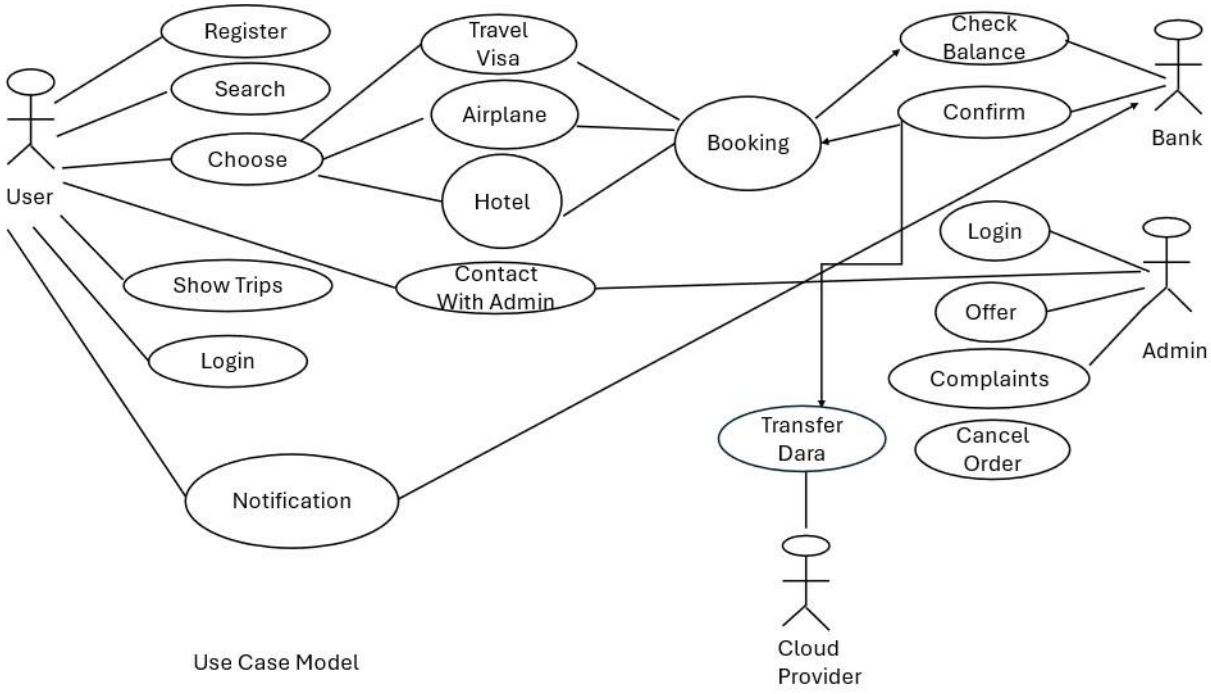
- Performance: Ensuring fast page load times under 3 seconds.
- Scalability: The platform should handle up to 10,000 concurrent users without degradation.
- Security: Implementation of SSL encryption and GDPR compliance for data protection.
- Reliability: Aiming for 99.9% uptime and maintaining daily backups.
- Usability: Get user feedback to make sure the platform is easy to use.
- Compatibility: Ensure the platform works on all main browsers and devices.
- Maintainability: Review code to make sure it’s easy to update

System Model

Context Diagram

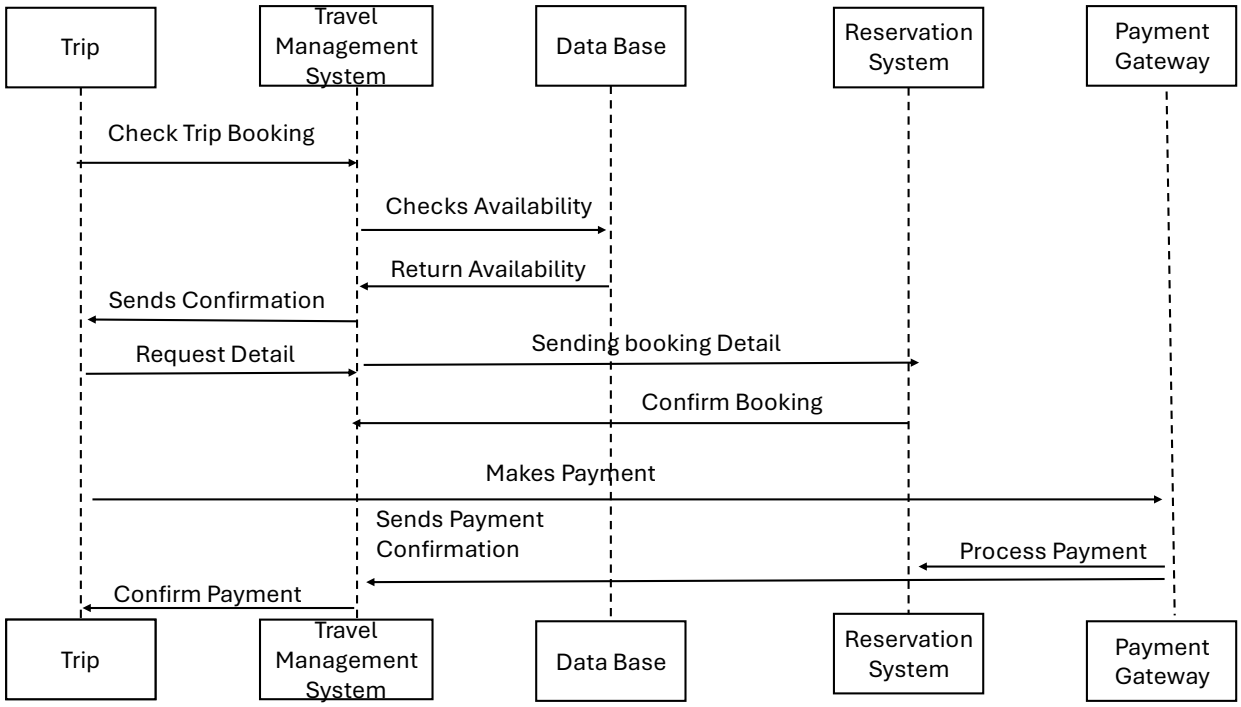


Use Case Diagram

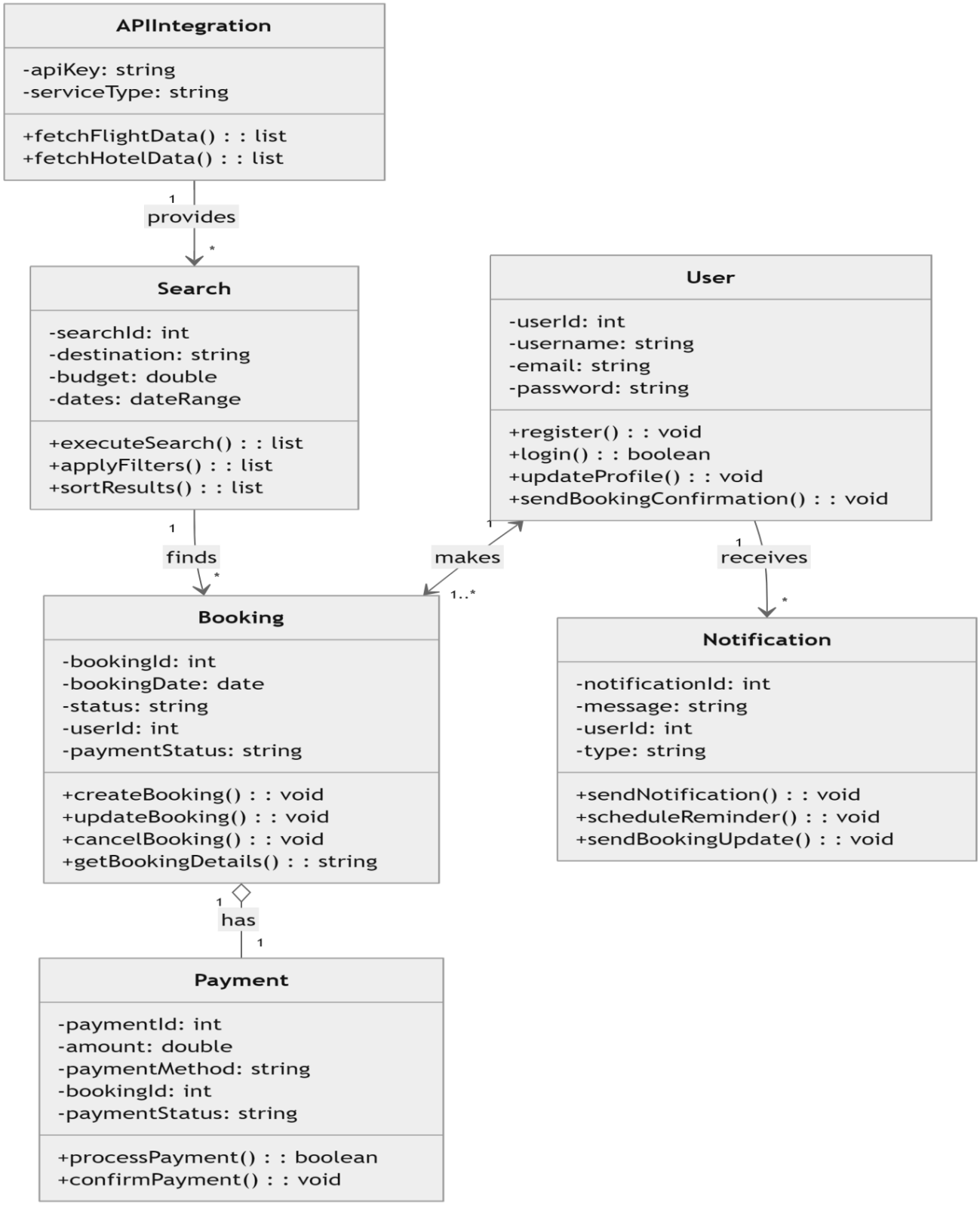


Use Case Model

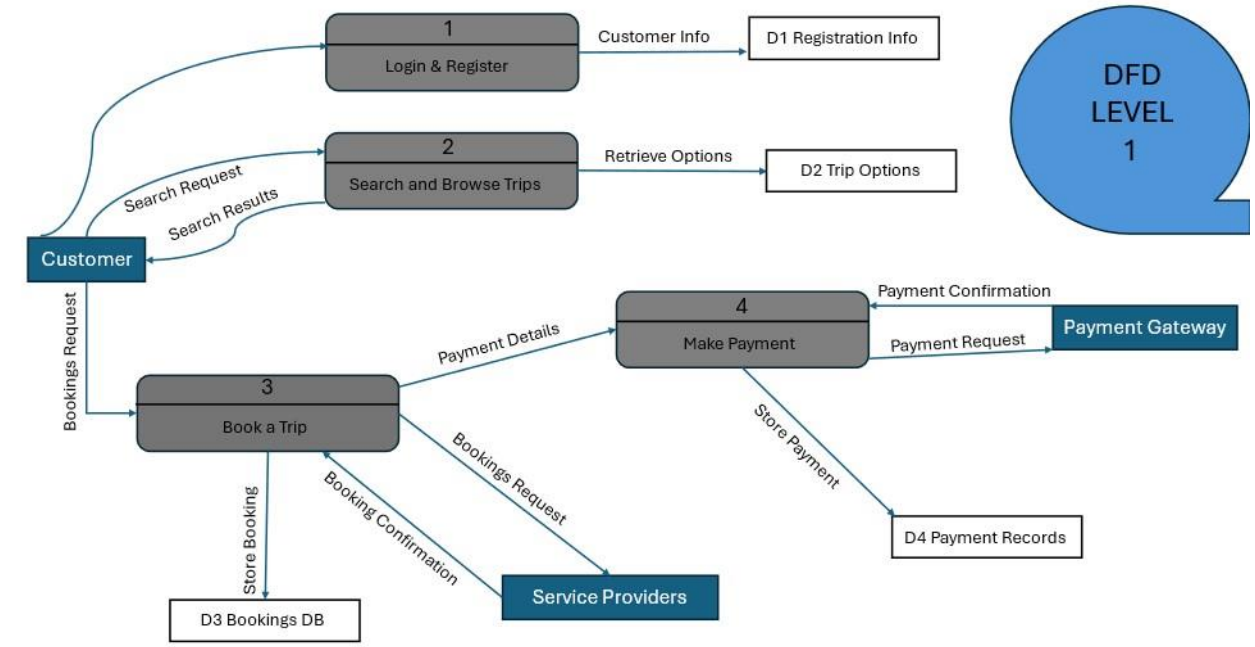
Sequence Diagram

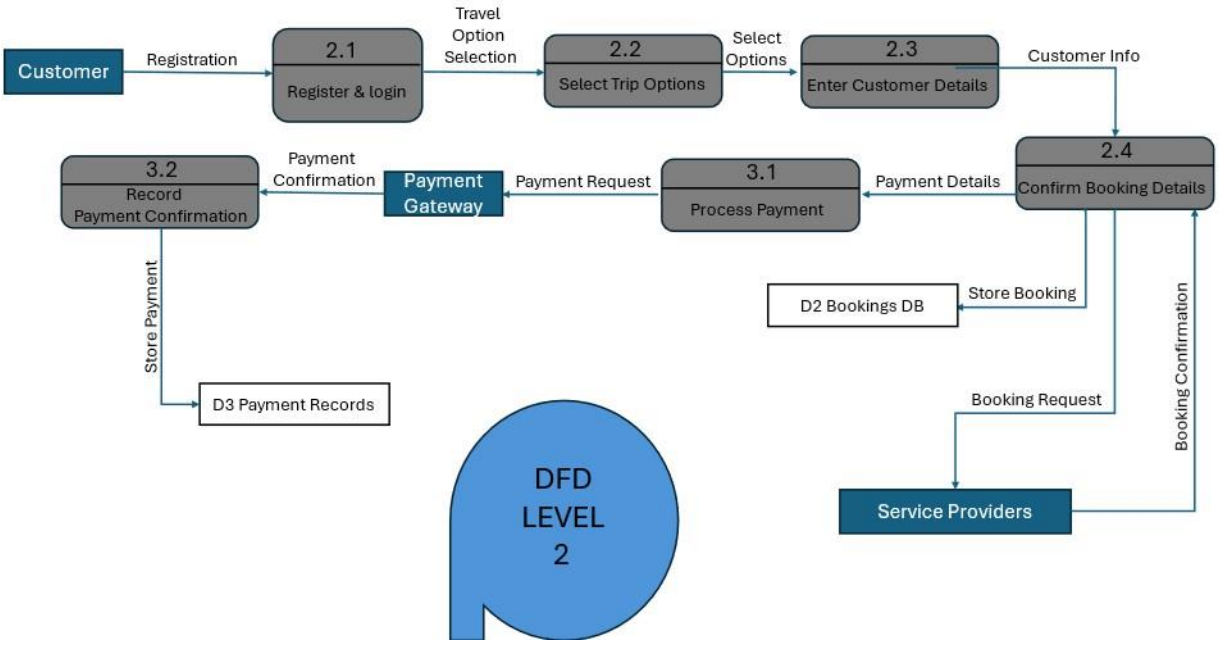


UML Diagram


APIIntegration
-apiKey: string
-serviceType: string
+fetchFlightData() : : list
+fetchHotelData() : : list
1
provides
*
Search
-searchId: int
-destination: string
-budget: double
-dates: dateRange
+executeSearch() : : list
+applyFilters() : : list
+sortResults() : : list
1
finds
*
User
-userId: int
-username: string
-email: string
-password: string
+register() : : void
+login() : : boolean
+updateProfile() : : void
+sendBookingConfirmation() : : void
1
makes
1..*
1
receives
*
Booking
-bookingId: int
-bookingDate: date
-status: string
-userId: int
-paymentStatus: string
+createBooking() : : void
+updateBooking() : : void
+cancelBooking() : : void
+getBookingDetails() : : string
Notification
-notificationId: int
-message: string
-userId: int
-type: string
+sendNotification() : : void
+scheduleReminder() : : void
+sendBookingUpdate() : : void
1
has
1
Payment
-paymentId: int
-amount: double
-paymentMethod: string
-bookingId: int
-paymentStatus: string
+processPayment() : : boolean
+confirmPayment() : : void

DFD


1
Login & Register
Customer Info
D1 Registration Info
DFD
LEVEL
1
2
Search and Browse Trips
Retrieve Options
D2 Trip Options
Search Request
Search Results
Customer
Bookings Request
3
Book a Trip
Payment Details
4
Make Payment
Payment Confirmation
Payment Request
Payment Gateway
Store Payment
D4 Payment Records
Bookings Request
Booking Confirmation
Store Booking
D3 Bookings DB
Service Providers


Customer
Registration
2.1
Register & login
Travel
Option
Selection
2.2
Select Trip Options
Select
Options
2.3
Enter Customer Details
Customer Info
2.4
Confirm Booking Details
Payment Details
3.1
Process Payment
Payment Request
Payment
Gateway
Payment
Confirmation
3.2
Record
Payment Confirmation
Store Payment
D3 Payment Records
D2 Bookings DB
Store Booking
Booking Request
Booking Confirmation
Service Providers
DFD
LEVEL
2