Software Requirements and Design Document

for

WanderWide (Travel Management System)

Prepared by Maria Naeem (22i-0812), Ammar Hussain (22i-8222), Malaika Afzal (22i-0885)

CoreMpact

25/11/2024

Table of Contents

1. Introduction

1.1 Purpose

*This document specifies the software requirements for **WanderWide** (Version 1.0), a Travel Management System designed to streamline travel planning and management. The document provides a detailed overview of the system's features, functionality, and purpose. It outlines the scope of the project, focusing on the two primary user interfaces: **Service Providers** and **Customers**. This SRS describes the complete system, highlighting its capabilities to centralize travel services and bookings into a unified platform.*

1.2 Product Scope

***WanderWide** is a comprehensive travel management application that offers users a seamless platform for booking transportation (flights, trains, buses) and accommodation (hotels). By integrating multiple services into a single solution, it simplifies travel planning and enhances user convenience.*

The product's scope includes the following functionalities:

- ***Service Provider Interface:** Adding and managing services (transport and hotels), viewing bookings, and monitoring customer feedback.*
- ***Customer Interface:** Searching, booking, and managing travel and accommodation services through a centralized platform.*

The system aims to reduce the complexity of navigating multiple platforms for travel-related needs, saving time and providing personalized services. This project aligns with the corporate strategy of leveraging digital solutions to enhance customer experience and streamline agency operations.

1.3 Title

***WanderWide**: A Unified Travel Management System for Simplified and Streamlined Travel Planning*

1.4 Objectives

*The primary objectives of **WanderWide** are:*

- *To centralize travel and accommodation services into a unified platform for enhanced user experience.*
- *To enable service providers to easily manage their services, bookings, and customer feedback.*
- *To eliminate inefficiencies and time wastage associated with traditional travel planning methods.*
- *To enhance the overall travel experience through intuitive design, ease of access, and automation of booking processes.*
- *To foster trust and reliability by providing a transparent, customer-centric platform.*

1.5 Problem Statement

Travelers today face significant challenges in planning trips, including navigating through multiple platforms to gather information about destinations, check availability, and book transportation or accommodation. This fragmented approach leads to inefficiencies, confusion, and wasted time. Service providers also struggle to effectively manage bookings and gather insights from customer feedback due to the lack of streamlined solutions.
WanderWide *addresses these issues by centralizing services into a single platform that automates travel planning and management processes. With its dual interfaces for service providers and customers, the application ensures efficient operations for agencies and offers users a simplified and personalized booking experience.*

2. Overall Description

2.1 Product Perspective

WanderWide *is a self-contained, stand-alone product designed to address the need for a unified platform to manage travel bookings and services. It is not intended to replace any existing system but offers a fresh solution to the growing demand for streamlined travel management. The application is developed to provide an intuitive, user-friendly interface for both* ***Service Providers*** *(who add and manage travel-related services) and* ***Customers*** *(who book and manage travel plans).*
While there are many individual systems available for booking flights, trains, buses, and hotels, ***WanderWide*** *brings these services together into one cohesive platform, removing the need for users to interact with multiple, disjointed websites.*
The architecture is modular and scalable, allowing for future expansion and integrations with other services.

2.2 Product Functions

The ***WanderWide*** *system must support a variety of functions for both service providers and customers. Below is a high-level summary of the key functions:*

Service Provider Functions:

- ***Add/Update Services****: Service providers can add, modify, or delete travel-related services such as flights, buses, trains, and hotels.*
- ***View Bookings****: View customer bookings for transportation and accommodation.*
- ***View Customer Feedback****: Access customer feedback regarding services offered.*

Customer Functions:

- ***Search and Browse Services:*** *Customers can search for available services based on location, and preferences (flights, trains, buses, and hotels).*
- ***Book Travel Services****: Customers can book transportation and accommodation services directly through the platform.*
- ***View Booking History****: Customers can view past bookings, details, and statuses of their reservations.*
- ***Cancel Booking****: Customers can cancel bookings and refunded.*
- ***Provide Feedback****: Customers can rate and provide feedback for services they have used.*

- ***View Notifications****: Customers can view notifications regarding cancelled and completed services.*

Common Functions:
- ***User Authentication:*** *Secure user login and registration for both service providers and customers.*

2.3 List of Use Cases

Service Provider:
- Add Travel Service
- Add Hotel Listing
- Update Service
- Cancel Service
- View Feedbacks
- View Customer Bookings

Customer:
- Book a travel service
- Book a hotel
- Cancel Booking
- Give Feedback

2.4 Extended Use Cases

1) Book a Flight

Use Case Name	Book a flight
Scope	Travel Management System
Level	User goal
Primary Actor	Customer
Stakeholders and Interests	• Customer wants to book a flight at the best time and price. • Airlines want to sell tickets efficiently. • Travel Management System ensures the system runs smoothly.
Preconditions	Customer has a valid user account and is logged into the TMS. Flights are available for booking.
Success Guarantee (Postcondition)	Customer successfully books a flight and receives a confirmation.
Main Success Scenario	

Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to book a flight.	1. Displays the flight booking option.
3. Enters travel details (departure, destination, dates).	4. Accepts travel details and queries the flight database.
5. Searches for available flights.	6. Shows available flights.
7. Selects a flight from the available options	8. Books the selected flight.
9. Enters payment details and confirms booking.	10. Processes payment securely.
11. Receives flight booking confirmation	12. Sends confirmation email and stores the booking in the system.
Extensions	• No flights are available for the selected date or destination. • System displays an error and suggests alternate dates. • Payment fails. • System asks the customer to re-enter payment details or use another payment method.

2) Book a Train Ticket

Use Case Name	Book a Train Ticket
Scope	Travel Management System
Level	User Goal
Primary Actor	Customer
Stakeholders and Interests	• Customer wants to book a train ticket at the preferred time. • Train Operators need to fill train seats. • TMS facilitates bookings efficiently.
Preconditions	• Customer is logged in. • Train tickets are available.
Success Guarantee (Postcondition)	Customer successfully books a train ticket and receives a confirmation.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to book a train ticket.	1. Displays the train booking option.

3. Enters travel details (departure, destination, dates).	4. Accepts travel details and queries the train database.
5. Searches for available trains.	6. Shows available trains and ticket classes.
7. Selects a train and ticket class.	8. Books the selected train.
9. Enters payment details and confirms booking.	10. Processes payment securely.
11. Receives booking confirmation.	12. Sends confirmation email and stores the booking.
Extensions	• No trains are available for the selected time. • System displays an error or suggests alternate dates or times.

3) Book a Bus Ticket

Use Case Name	Book a Bus Ticket
Scope	Travel Management System
Level	User Goal
Primary Actor	Customer
Stakeholders and Interests	• Customer wants to book a bus ticket easily. • Bus Operators needs to fill bus seats. • TMS supports ticket sales.
Preconditions	• Customer is logged in. • Bus tickets are available.
Success Guarantee (Postcondition)	Malaika successfully books a bus ticket and receives confirmation.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to book a bus ticket.	1. Displays the bus booking option.
3. Enters travel details (departure, destination, dates).	4. Accepts travel details and queries the bus database.
5. Searches for available buses.	6. Shows available buses and seating options.
7. Selects a bus and seating class.	8. Books the selected bus.
9. Enters payment details and confirms booking.	10. Processes payment securely.
11. Receives booking confirmation.	12. Sends confirmation email and stores the booking.
Extensions	• No buses available. • System suggests alternative travel options.

4) Book a hotel

Use Case Name	Book a hotel
Scope	Travel Management System
Level	User Goal
Primary Actor	Customer
Stakeholders and Interests	• Customer wants to book a hotel that fits her preferences. • Hotel wants to fill rooms. • TMS facilitates bookings.
Preconditions	• Customer is logged in. • Hotels are available.
Success Guarantee (Postcondition)	Customer successfully books a hotel room and receives confirmation.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to book a hotel.	1. Displays hotel booking option.
3. Enters hotel search criteria (location, dates, preferences).	4. Accepts search criteria and queries hotel databases.
5. Searches for available hotels.	6. Shows available hotels.
7. Selects a hotel and room type.	8. Books the selected hotel.
9. Enters payment details and confirms booking.	10. Processes payment securely.
11. Receives booking confirmation.	12. Sends confirmation and stores the booking.
Extensions	• No hotels are available. • System suggests alternative options. • Payment fails. • System prompts to retry payment.

5) Cancel Booking

Use Case Name	Cancel Booking
Scope	Travel Management System
Level	User Goal
Primary Actor	Customer
Stakeholders and Interests	• Customer needs to cancel a booking easily. • Travel managers process cancellations efficiently. • TMS manages cancellations and refunds.
Preconditions	Customer has an existing booking.
Success Guarantee (Postcondition)	Customer successfully cancels the booking and receives confirmation.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to manage bookings.	1. Displays booking management options.
3. Chooses the booking to cancel.	4. Lists Customers bookings.
5. Confirms cancellation.	6. Processes the cancellation and issues a refund.
7. Receives cancellation confirmation.	8. Sends cancellation confirmation.
Extensions	• Booking is non-cancellable. • System displays a message with cancellation restrictions.

6) Manage Bookings

Use Case Name	Manage Bookings
Scope	Travel Management System
Level	User Goal
Primary Actor	Customer
Stakeholders and Interests	• Customer wants to view, update, or cancel bookings. • TMS maintains accurate records.
Preconditions	Customer has existing bookings.
Success Guarantee (Postcondition)	Customer successfully views, updates, or cancels bookings.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
2. Customer selects the option to manage bookings.	1. Displays booking management options.

3. Views her booking history.	4. Shows booking history.
5. Selects a booking to update or cancel.	6. Updates the selected booking.
7. Confirms changes.	8. Sends confirmation of any changes.
10. Receives confirmation of the update.	
Extensions	• Booking cannot be modified. • System displays an error message or alternative options.

7) Add Travel Service

Use Case Name	Add travel service
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to add new travel services to the system to offer customers more options. • **Customers:** Interested in having access to a variety of travel services for booking.
Preconditions	The service provider must be logged into the system.
Success Guarantee (Postcondition)	The new travel service is successfully added to the system and is visible to customers.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Add Travel Service' Section.	
	3. System loads the interface and displays relevant input fields.
4. Enters the details of the new service (e.g., service type, description, price, route).	
5. Submits the form.	
	6. Validates the input
	7. Adds the new service to the database
	8. Confirms the successful addition to the Service Provider
Extensions	• If the input data is invalid, the system displays an error message and prompts the service provider to correct the information.

8) Add Hotel Listing

Use Case Name	Add Hotel Listing
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to add hotel listings to provide more options to customers. • **Customers:** Interested in finding and booking hotels easily.
Preconditions	The service provider must be logged into the system.
Success Guarantee (Postcondition)	The new hotel listing is successfully added to the system and is visible to customers.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Add Hotel Listing' Section.	
	3. System loads the interface and displays relevant input fields.
4. Enters the details of the hotel (e.g., name, location, description, price, amenities).	
5. Submits the form.	
	6. Validates the input
	7. Adds the new hotel listing to the database
	8. Confirms the successful addition to the Service Provider
Extensions	• If the input data is invalid, the system displays an error message and prompts the service provider to correct the information.

9) Update Service

Use Case Name	Update Service
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to update existing services. • **Customers:** Interested in having access to the most current and accurate services.
Preconditions	• The service provider must be logged into the system. • The service to be updated exists in the system
Success Guarantee (Postcondition)	The updated service details are successfully saved in the system and are visible to customers.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Manage Services' Section.	
	3. Displays services
4. Selects the service to be updated (Travel or Hotel)	
	5. Displays details of service
4. Edits the details of the selected service (e.g., service type, description, price, availability).	
5. Submits the changes.	
	6. Validates the updated data
	7. Updates the service details in the database.
	8. Confirms the successful update to the service provider.
Extensions	• If the updated data is invalid, the system displays an error message and prompts the service provider to correct the information. • If the specified service does not exist, the system displays an error message indicating that it cannot be found.

10) Cancel/Remove Service

Use Case Name	Cancel/Remove Service
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to remove services that are no longer offered. • **Customers:** Interested in having access only to available services.
Preconditions	• The service provider must be logged into the system. • The service to be removed exists in the system.
Success Guarantee (Postcondition)	The specified service is successfully removed from the system and is no longer visible to customers.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Manage Services' Section.	
	3. System loads the interface and displays all services.
4. Selects the service to be removed.	
5. Confirms the intention to remove the selected service.	
	6. Validates the request for removal.
	7. Removes the service from the database
	8. Confirms the successful removal to the service provider.
Extensions	• If the service cannot be found, the system displays an error message indicating that it cannot be found for removal. • If there are pending bookings related to the service, the system displays a warning message indicating that removal is not possible until those bookings are addressed.

11) View Customer Bookings

Use Case Name	View Customer Bookings
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to view customer bookings to manage services effectively. • **Customers:** Interested in having their bookings visible and manageable by the service provider.
Preconditions	The service provider must be logged into the system.
Success Guarantee (Postcondition)	The service provider has access to the list of customer bookings, including details such as dates, services booked, and customer information.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Customer Booking' Section.	
	3. System loads the interface and displays services.
4. Selects the service for which they want to view the bookings.	
	5. Retrieves the list of customer bookings from the database.
	6. Displays booking details including customer names, dates, and services booked.
Extensions	• If there are no bookings available, the system displays a message that there are no current customer bookings.

12) View Customer Feedbacks

Use Case Name	View Customer Feedbacks
Scope	Travel Management System
Level	User Goal
Primary Actor	Service Provider
Stakeholders and Interests	• **Service Provider:** Wants to view customer feedback to improve services. • **Customers:** Interested in providing feedback on their experiences.
Preconditions	The service provider must be logged into the system.
Success Guarantee (Postcondition)	The service provider has access to customer feedback, including ratings and comments, to assess service quality.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
1. Service Provider logs in to the System	
2. Navigates to 'Feedback' Section.	
	3. System loads the interface and displays services.
4. Selects the service for which they want to view feedback.	
	5. Retrieves the list of feedbacks from the database.
	6. Shows feedback details including ratings, comments, and customer names.
Extensions	• If there are no feedback entries available for the selected service, the system displays a message indicating that there is no feedback to show.

13) Make Payment

Use Case Name	Make Payment
Scope	Travel Management System
Level	User goal level
Primary Actor	Customer
Stakeholders and Interests	● **Customer**: Wants to make secure payments for booked services. ● **Travel Service Providers**: Want to ensure payment confirmation and accurate recordkeeping. ● **Tax Agency**: Wants to collect tax on every transaction. ● **Payment Gateway**: Wants to facilitate smooth transactions and ensure security.
Preconditions	● Customer has successfully selected a travel service (hotel, flight, train, bus). ● Customer has valid payment details (credit/debit card, PayPal, etc.). ● The system is connected to a payment gateway for processing.
Success Guarantee (Postcondition)	The payment is processed successfully, and the Customer receives a confirmation (E-mail / SMS)of payment and booking.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
Customer selects the "Proceed to Payment" option after choosing a service.	
	- System redirects the Customer to the payment interface. - System displays relevant input fields.
- Customer enters payment details (credit card, debit card, etc.). - Customer submits the search request.	
	- System validates the input data and processes the payment.. - System sends a payment confirmation.
- Customer receive the confirmation and booking	

details.	
Extensions	● If payment details are invalid, the system prompts the Customer to correct the information. ● If the payment gateway is unavailable, the system shows an error message and suggests retrying. ● If payment is declined, the system informs the Customer and provides other payment options.

14) Give Feedback

Use Case Name	Give Feedback
Scope	Travel Management System
Level	User goal level
Primary Actor	Customer
Stakeholders and Interests	● **Customer**: Wants to share their experience (positive or negative) about a travel service. ● **Travel Service Providers**: Want feedback to improve services and boost ratings. ● **Travel Agency**: Wants to gather feedback to improve user experience and make informed decisions.
Preconditions	● Customer is logged into their account. ● Customer has completed a booking and used a travel service (e.g., hotel stay, flight, etc.).
Success Guarantee (Postcondition)	The feedback is successfully submitted and stored in the system.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
Customer navigate to the feedback section after completing their service.	
	System displays a feedback form with fields for rating (e.g., stars, numerical score) and optional comments.
- Customer provides a rating and writes comments. - Customer submits the feedback.	
	System stores the feedback in the database and associates it with the Customer’s booking.
Extensions	● If the feedback form is incomplete, the system prompts the Customer to fill required fields. ● If the system experiences an error, it shows an

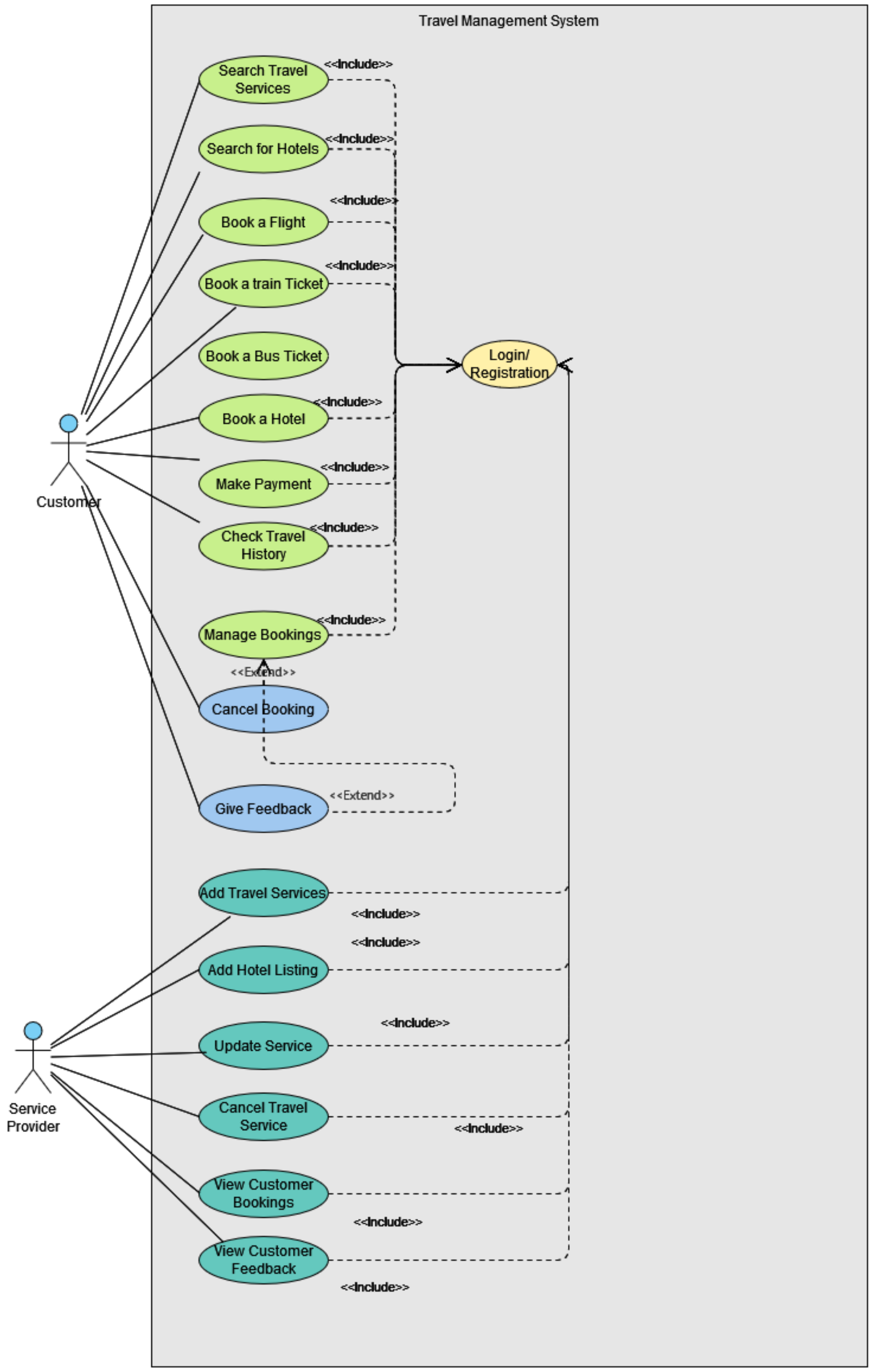
	error message and suggests retrying. ● If the feedback includes inappropriate content, the system flags it for review.

15) Check Travel History

Use Case Name	Check Travel History
Scope	Travel Management System
Level	User goal level
Primary Actor	Customer
Stakeholders and Interests	● **Customer**: Wants to review past bookings and travel details. ● **Travel Agency**: Wants to provide users with an easy way to access their booking history. ● **Travel Service Providers**: May use history data to offer personalized recommendations.
Preconditions	● Customer is logged into the system. ● Customer has previously completed bookings that are stored in the system.
Success Guarantee (Postcondition)	Customer successfully view their travel history with detailed booking information.
Main Success Scenario	
Actor Action (or Intention)	**System Responsibility**
Customer navigates to the "Travel History" section.	
	- System retrieves the Customer's booking history from the database. - System displays a list of past bookings with relevant details (dates, destinations, services used).
- Customer views the list of past bookings - Customer selects a specific booking for more detailed information.	
	System displays detailed information for the selected booking, including payment details and service usage.
Extensions	● If no booking history is found, the system displays a message indicating no past bookings. ● If the system cannot retrieve the history due to a technical issue, an error message is displayed and

	retry is suggested.

2.5 Use Case Diagram



Section K
Group Members
Maria Naeem (22i-0812)
Malaika Afzal (22i-0885)
Ammar Hussain (22i-8222)

3. Other Nonfunctional Requirements

3.1 Performance Requirements

- ***Response Time**: The system should provide a response to a customer booking request (for hotels, trains, buses, flights) within **3 seconds** during normal operation, with no more than **5 seconds** during peak times.*
- ***Concurrent Users**: The system must be able to handle up to **2,000 concurrent users** for booking services without performance degradation.*
- ***Scalability**: The system should scale horizontally to handle up to **50,000 concurrent users** as the user base grows.*
- ***Data Synchronization**: Changes made by service providers (e.g., adding or updating services) should be reflected in real time across all customer interfaces.*

3.2 Safety Requirements

- ***Data Backup and Recovery**: The system must have regular data backups, and the user's booking data should be recoverable within **30 minutes** in case of an unexpected shutdown or data loss.*
- ***Booking Cancellation**: The system must ensure that customer cancellations are processed securely and immediately and notify all affected parties (service provider and customer).*

3.3 Security Requirements

- ***User Authentication**: Customers and service providers must authenticate using **username/password**.*
- ***Authorization**: Service providers should only have access to their own services and feedback, while customers should only have access to their own bookings and feedback.*

3.4 Software Quality Attributes

- ***Reliability**: The system should have a minimum uptime of **99.9%**, ensuring it is available to users most of the time.*
- ***Usability**: The system should have an intuitive user interface that allows users to book travel and accommodation with no more than **5 clicks** from the homepage.*
- ***Maintainability**: The code should be modular and well-documented, allowing the system to be easily updated and maintained. This includes adding new services or improving the customer interface without disrupting the core functionality.*
- ***Flexibility**: The platform must be adaptable to future changes, such as the addition of new travel services or hotel chains.*

3.5 Business Rules

- ***Booking Limit**: A customer cannot book more than **1 travel service** (flight, train, bus) in a single transaction.*
- ***Feedback Submission**: Customers can only submit feedback for a service once the service has been completed (e.g., after the customer has completed their trip).*

3.6 Operating Environment

Hardware Platform:
*The application will be designed for desktop environments, compatible with modern operating systems (Windows, macOS, and Linux) running on typical consumer hardware (laptops, desktops). It should work well on machines with at least **4GB of RAM** and a **2.0 GHz** processor. The system is intended to run as a desktop application and does not require a web browser for interaction.*

Operating System:
The application will be compatible with the following operating systems:

- ***Windows**: Version 10 or higher.*
- ***macOS**: Version 10.14 (Mojave) or higher.*
- ***Linux**: Ubuntu 20.04 or higher*

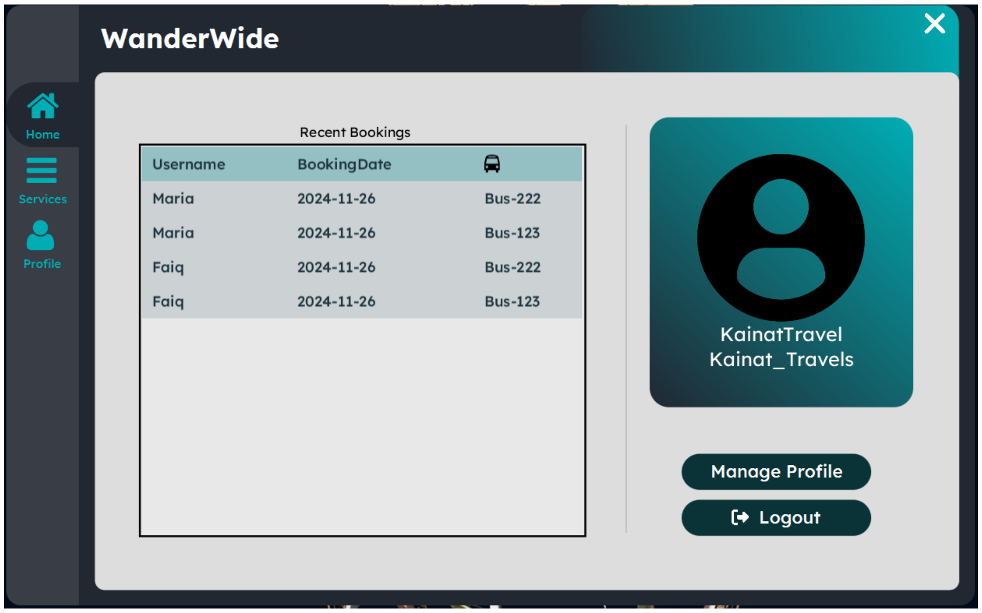
*The application will use **JavaFX** for the user interface and will require a **JDK** (Java Development Kit) version 8 or higher for full functionality.*

Dependencies:
*The system will be built using **Java** for the core backend logic and **JavaFX** for the graphical user interface (GUI). The database will be based on **MariaDB** for structured data storage, with **JDBC** (Java Database Connectivity) used to interact with the database. The application will not require an internet browser for usage, as it is a standalone desktop application.*

3.7 User Interfaces

Service Provider Interface:
The service provider's interface should have sections to add, update, or cancel travel services (flights, trains, buses) and hotels. The user interface should follow an intuitive layout with clear, labeled buttons, and responsive design.

- **Home Screen**: Displays a dashboard with recent customer bookings.

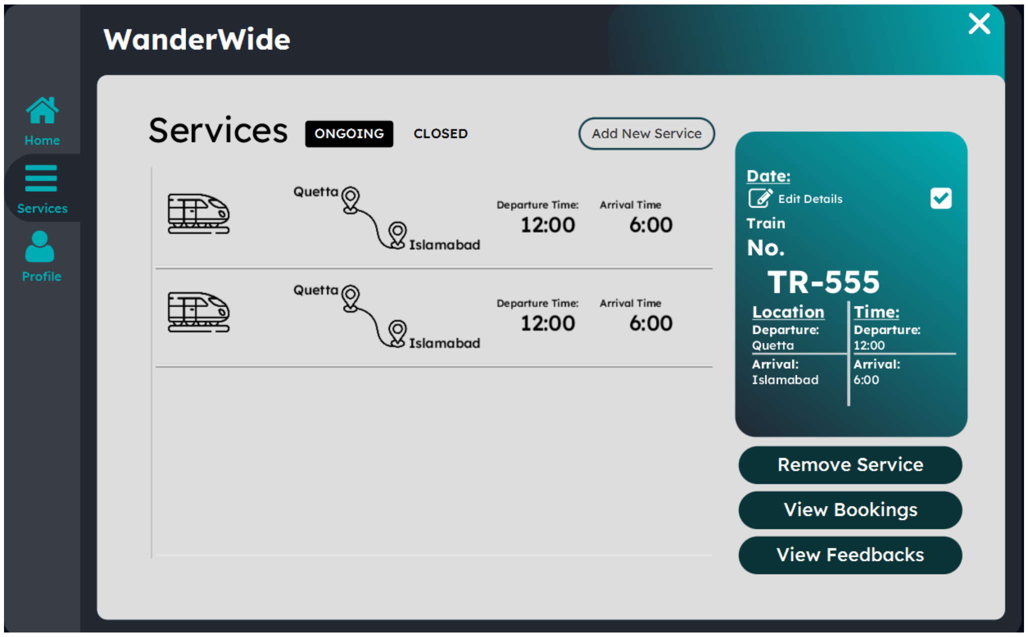


Recent Bookings

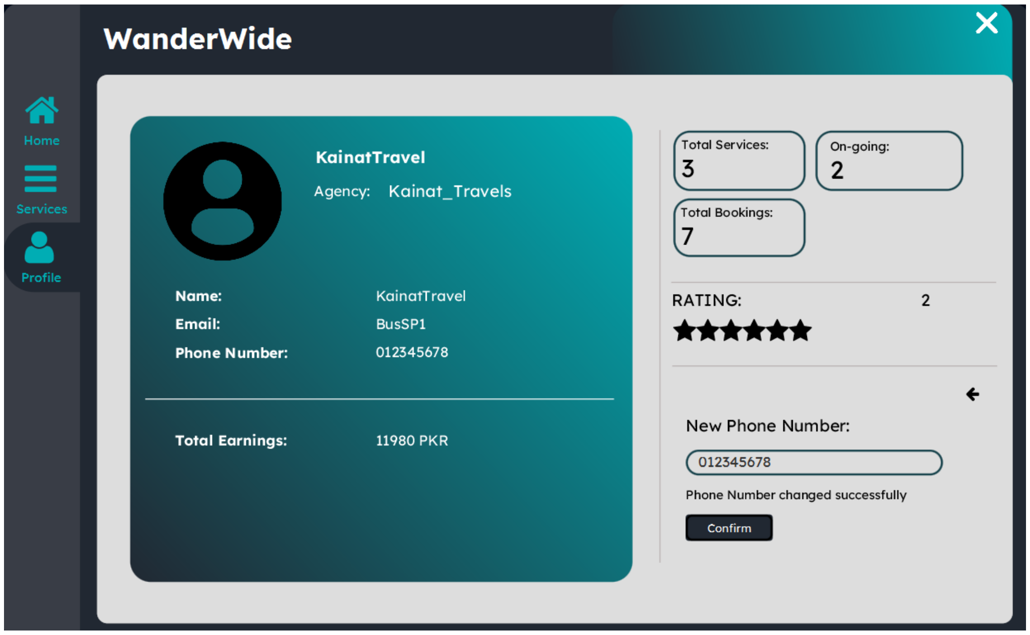
Username	BookingDate	🚌
Maria	2024-11-26	Bus-222
Maria	2024-11-26	Bus-123
Faiq	2024-11-26	Bus-222
Faiq	2024-11-26	Bus-123

KainatTravel
Kainat_Travels

- **Services Screen:** Displays current active and closed services. It should also allow the user to view customer bookings and feedbacks for a selected service. User will also be able to add, edit or cancel services through this interface.



- **Profile Screen:** This interface allows the service provider to view statistics, manage profile (changing password, or phone number)



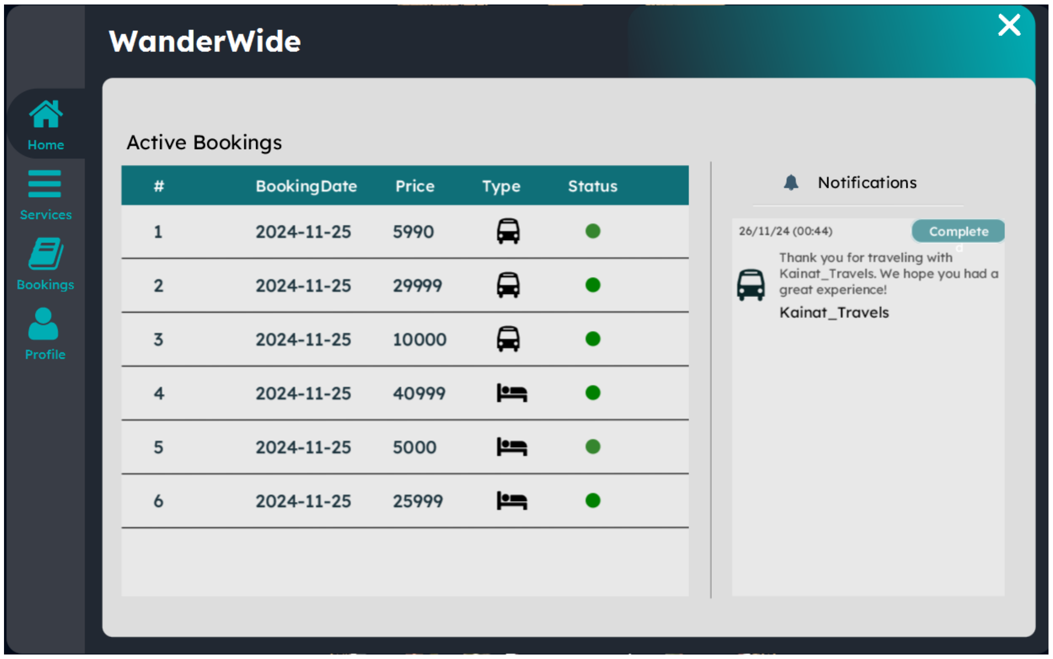
Name:	KainatTravel
Email:	BusSP1
Phone Number:	012345678
Total Earnings:	11980 PKR

New Phone Number: 012345678

Phone Number changed successfully

Customer Interface:
The customer interface should have sections to view, book services and manage bookings. The user interface should follow an intuitive layout with clear, labeled buttons, and responsive design.

- **Home Screen**: Displays a dashboard with current active bookings and notification.



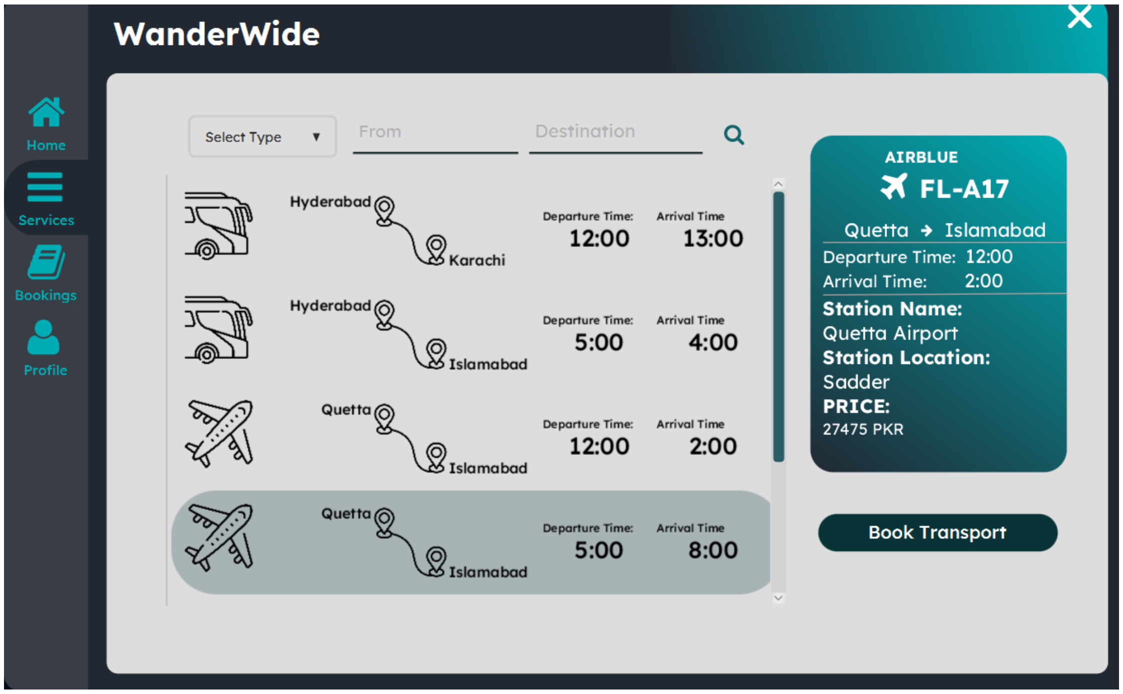
Active Bookings

#	BookingDate	Price	Type	Status
1	2024-11-25	5990	🚌	●
2	2024-11-25	29999	🚌	●
3	2024-11-25	10000	🚌	●
4	2024-11-25	40999	🛏	●
5	2024-11-25	5000	🛏	●
6	2024-11-25	25999	🛏	●

Notifications

26/11/24 (00:44) Complete
Thank you for traveling with Kainat_Travels. We hope you had a great experience!
Kainat_Travels

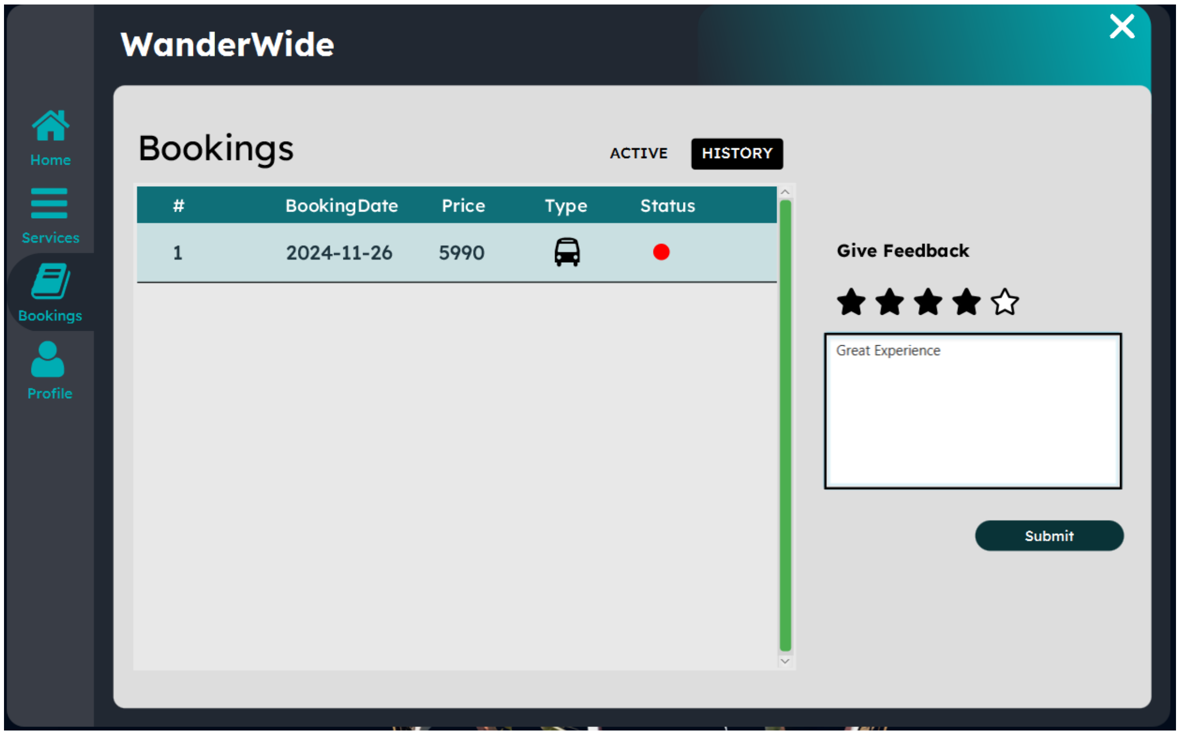
- **Services Screen:** Displays services available to book. It should also allow the user to filter services based on type and locations. User will be able to book the selected service through this interface.



Hyderabad	Karachi	Departure Time: 12:00	Arrival Time 13:00
Hyderabad	Islamabad	Departure Time: 5:00	Arrival Time 4:00
Quetta	Islamabad	Departure Time: 12:00	Arrival Time 2:00
Quetta	Islamabad	Departure Time: 5:00	Arrival Time 8:00

AIRBLUE
FL-A17
Quetta → Islamabad
Departure Time: 12:00
Arrival Time: 2:00
Station Name:
Quetta Airport
Station Location:
Sadder
PRICE:
27475 PKR

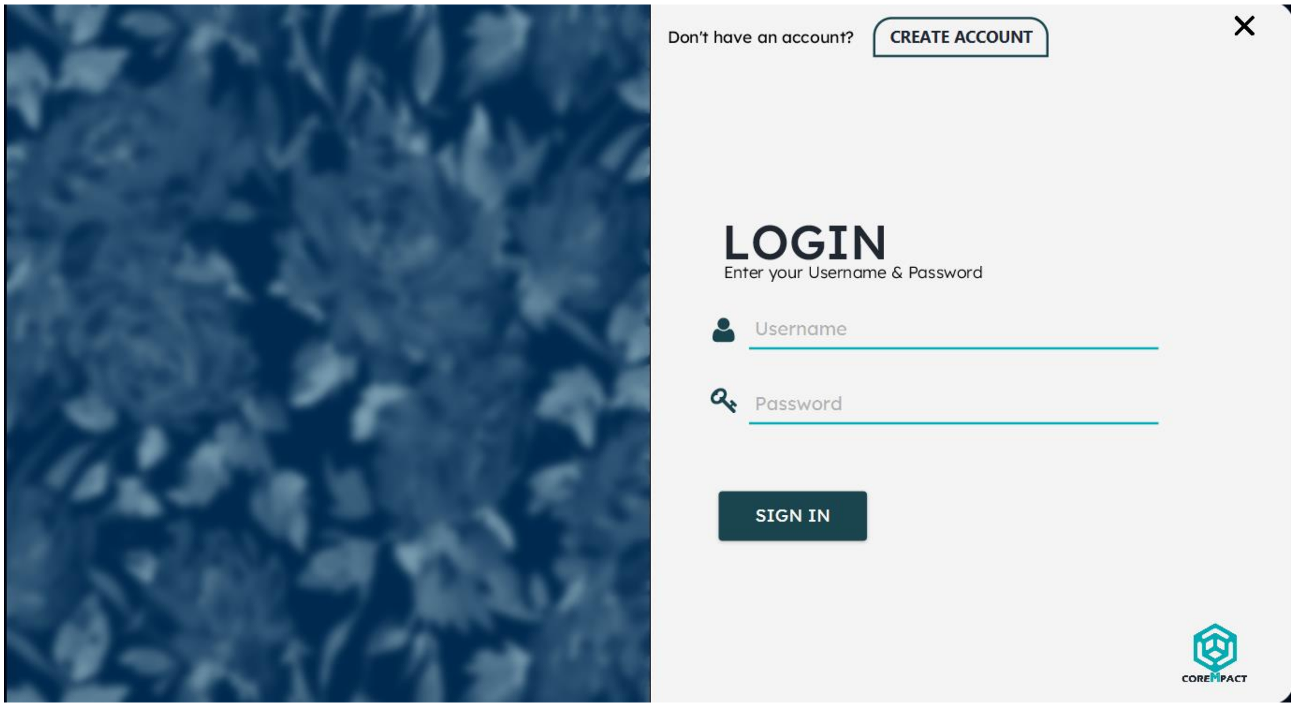
- **Bookings Screen:** Interface to manage customer bookings. It should allow the user to view current and booking history, with options to cancel booking and give feedback.

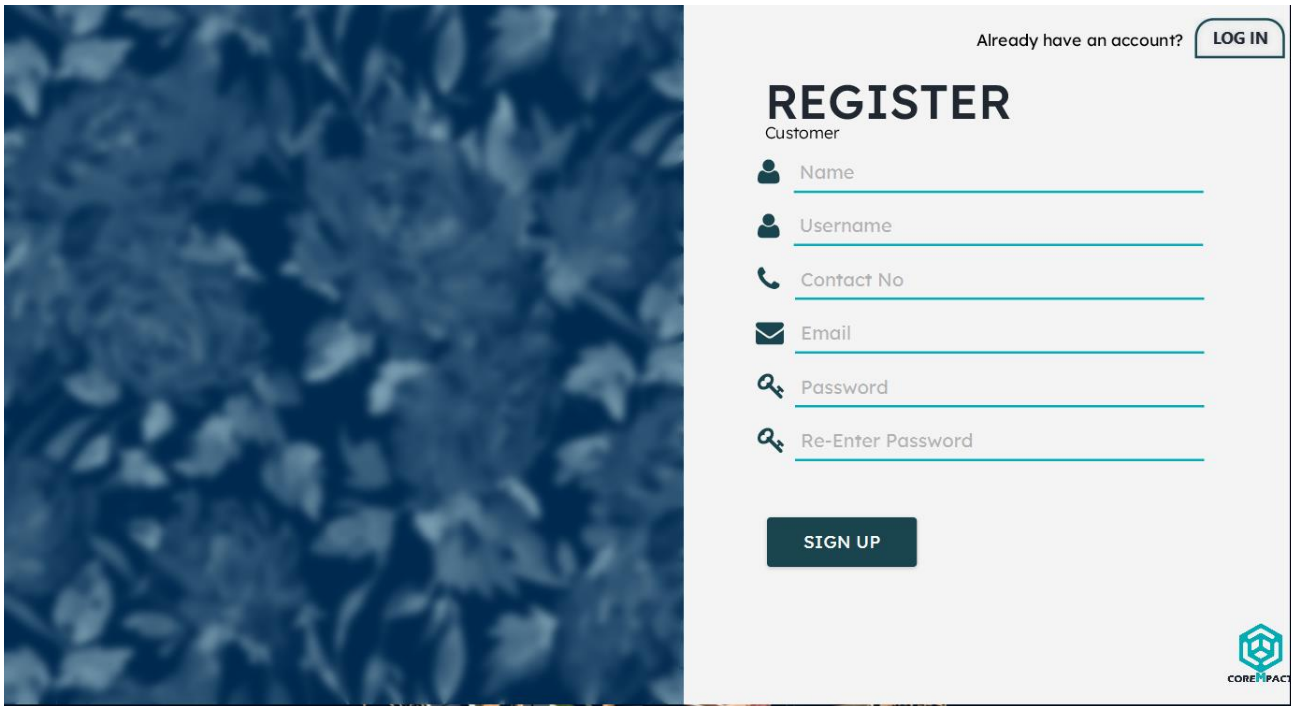


#	BookingDate	Price	Type	Status
1	2024-11-26	5990	🚌	🔴

Great Experience

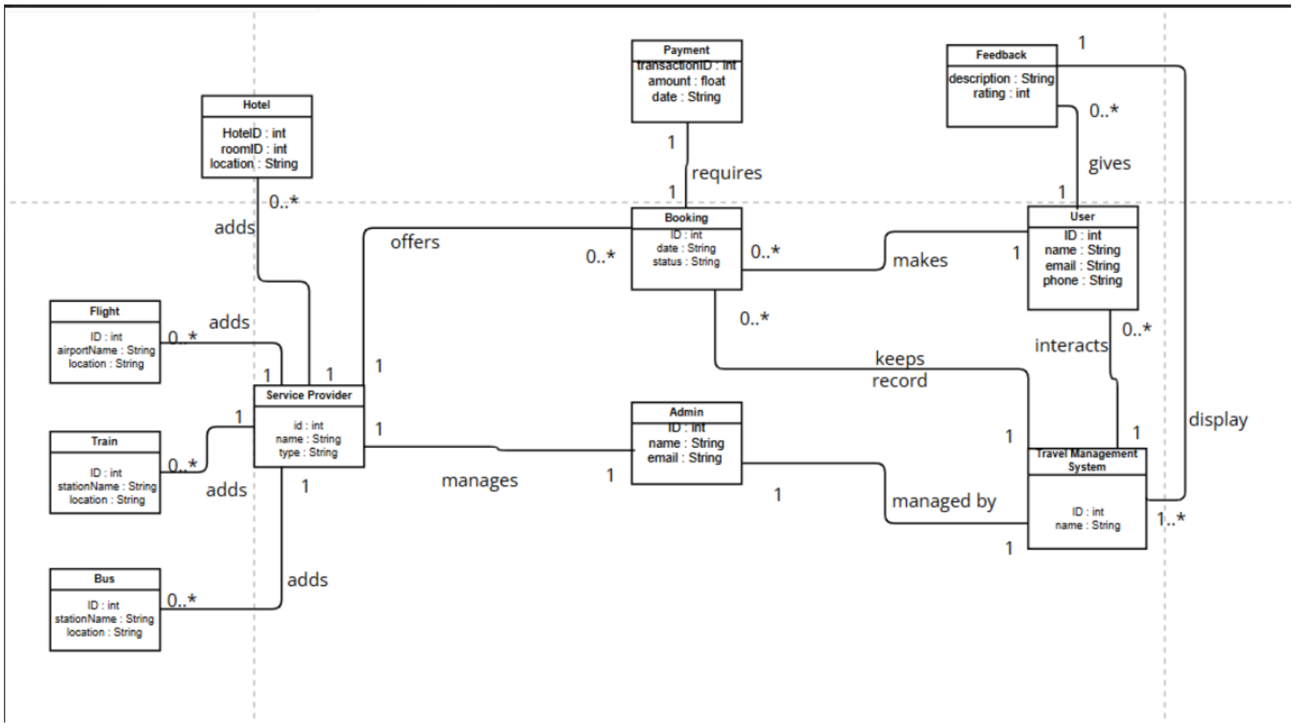
Common Interface (Register/Login Screen)



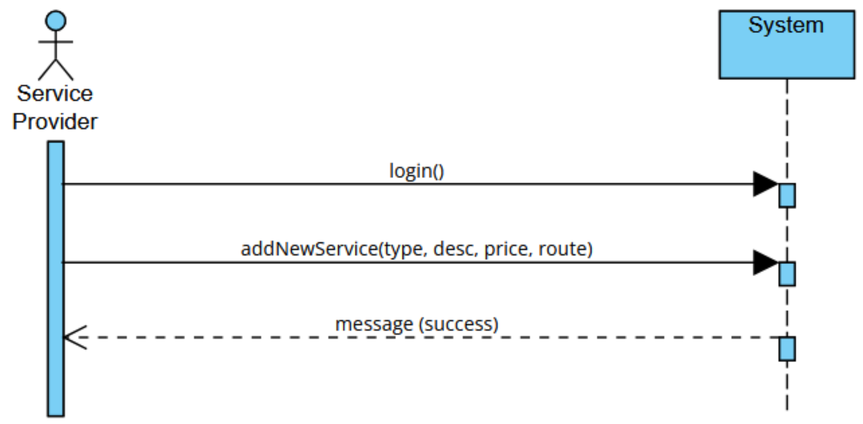


*We have used the fonts "**Lexend**" and "**Lexend Deca**" in the application to enhance readability, provide a modern aesthetic, and improve accessibility for extended reading sessions.*

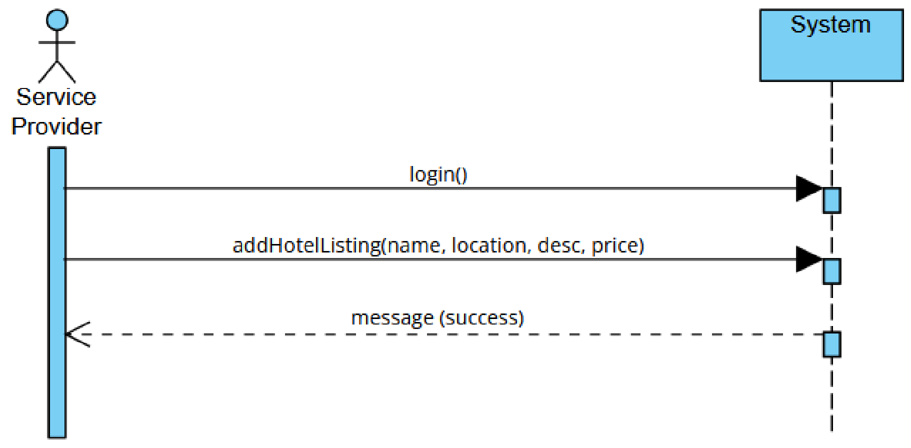
4. Domain Model



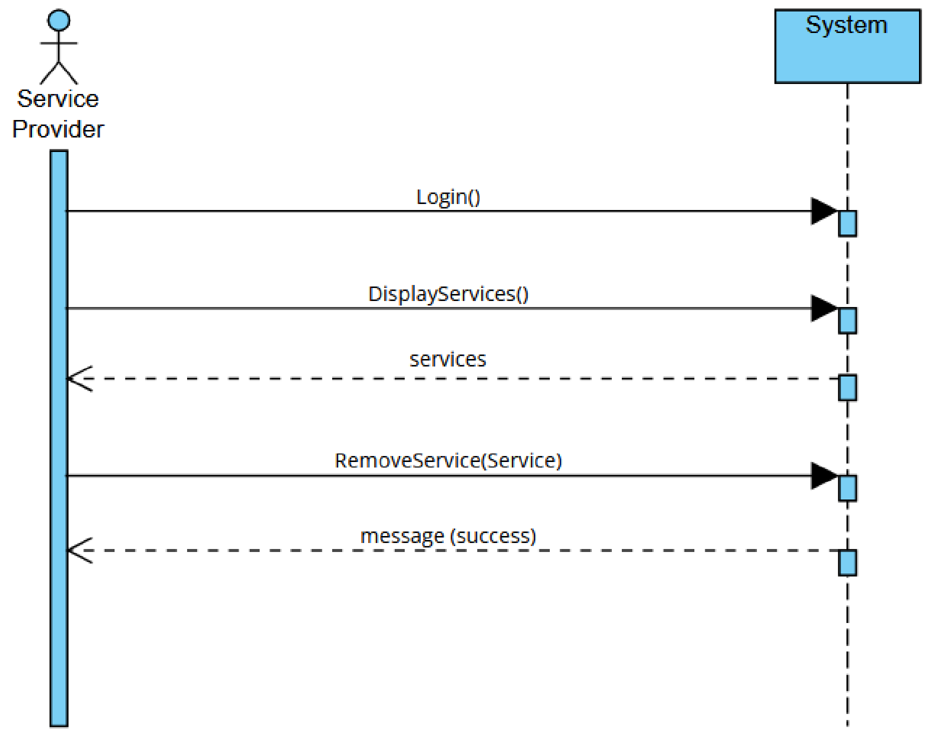
5. System Sequence Diagram



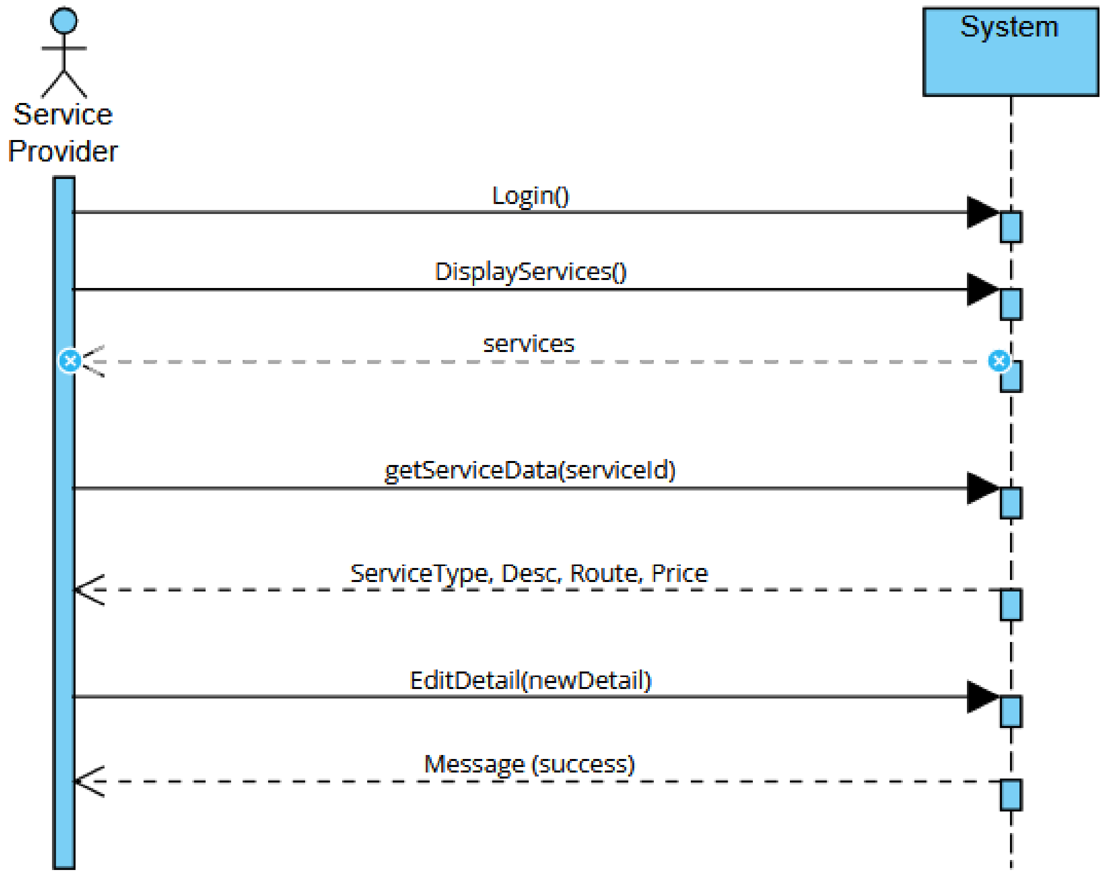
Add Travel Service



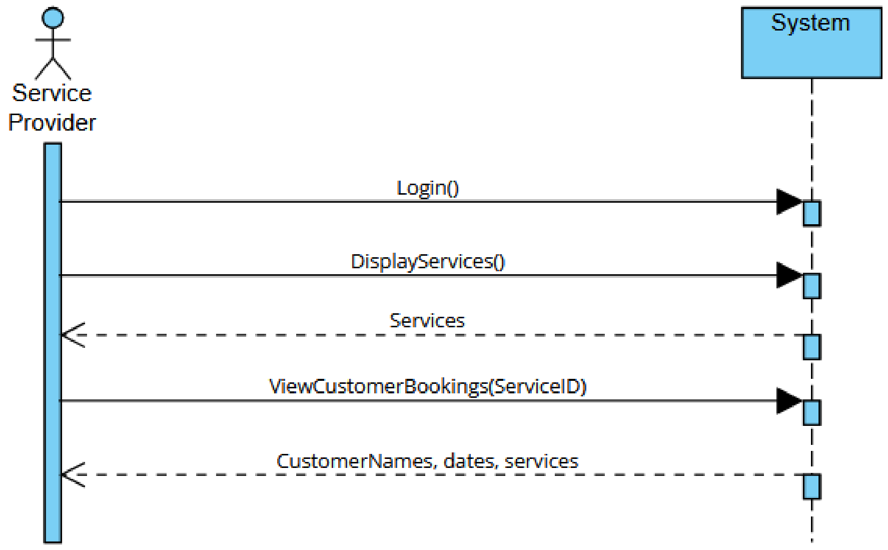
Add Hotel Listing



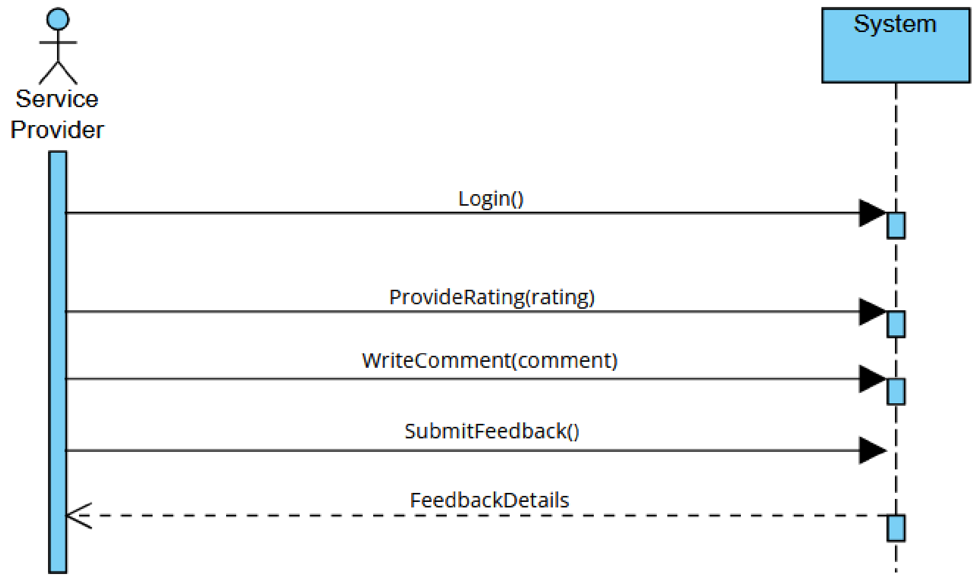
Remove/Cancel Service



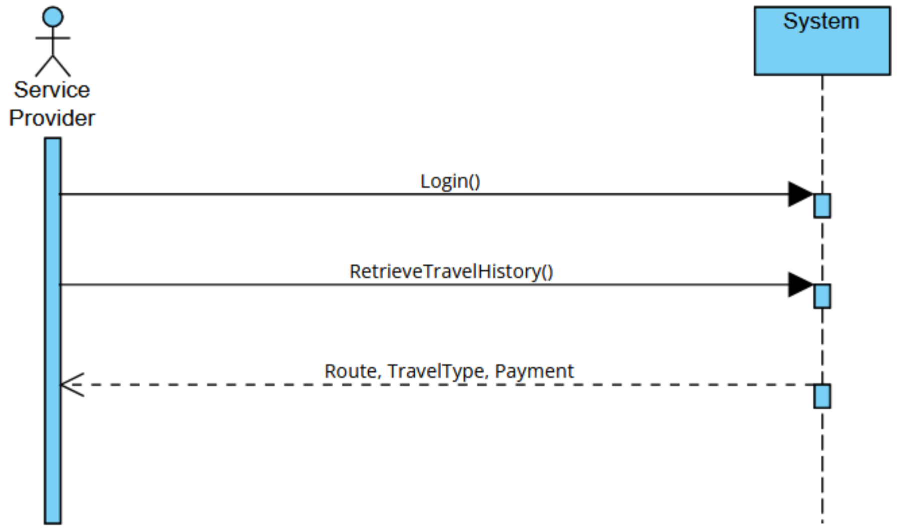
Update Services


Service Provider
System
Login()
DisplayServices()
Services
ViewCustomerBookings(ServiceID)
CustomerNames, dates, services

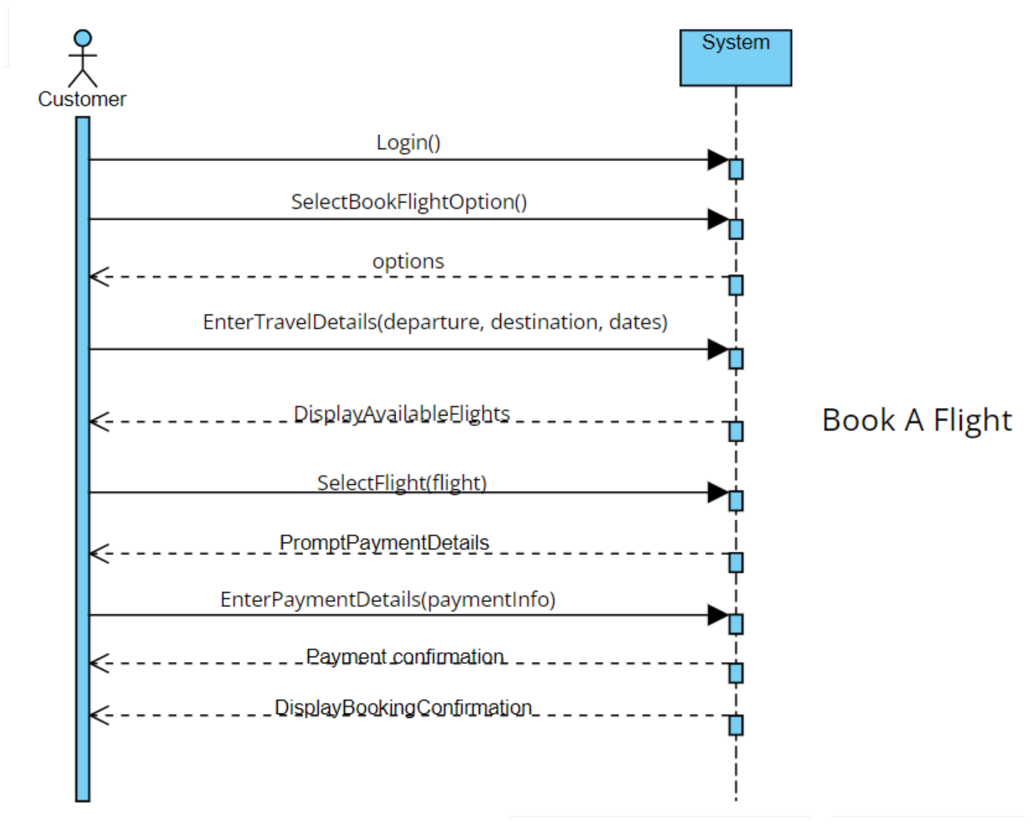
View Customer Bookings

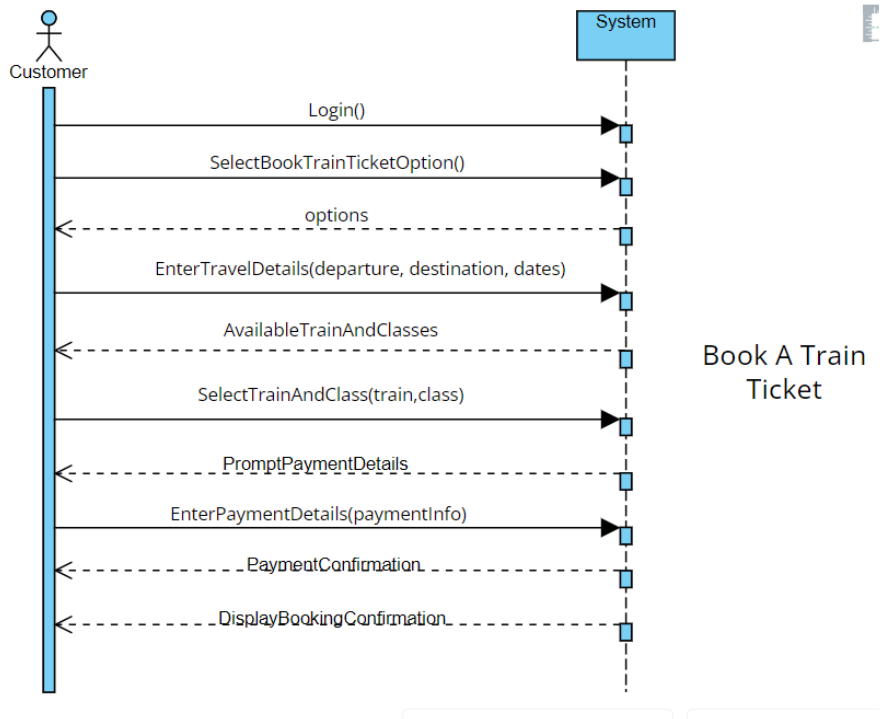


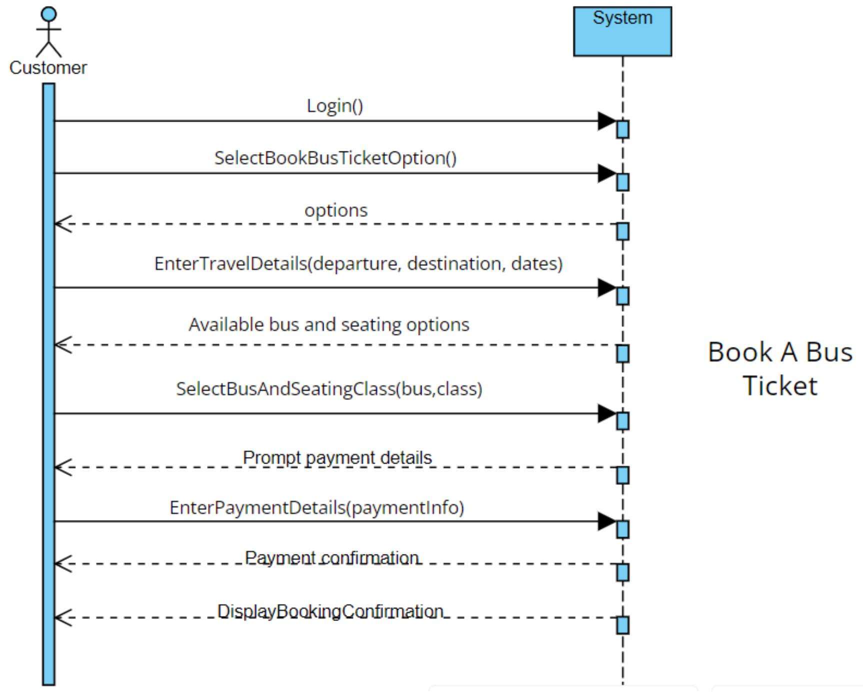
Give Feedback

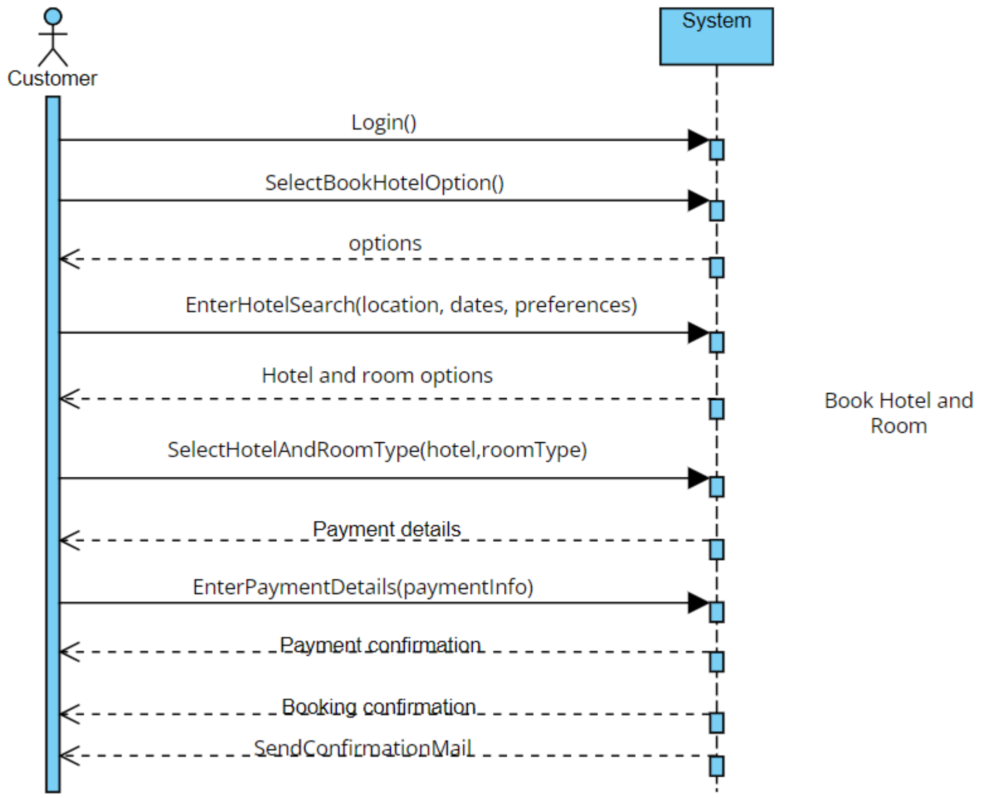


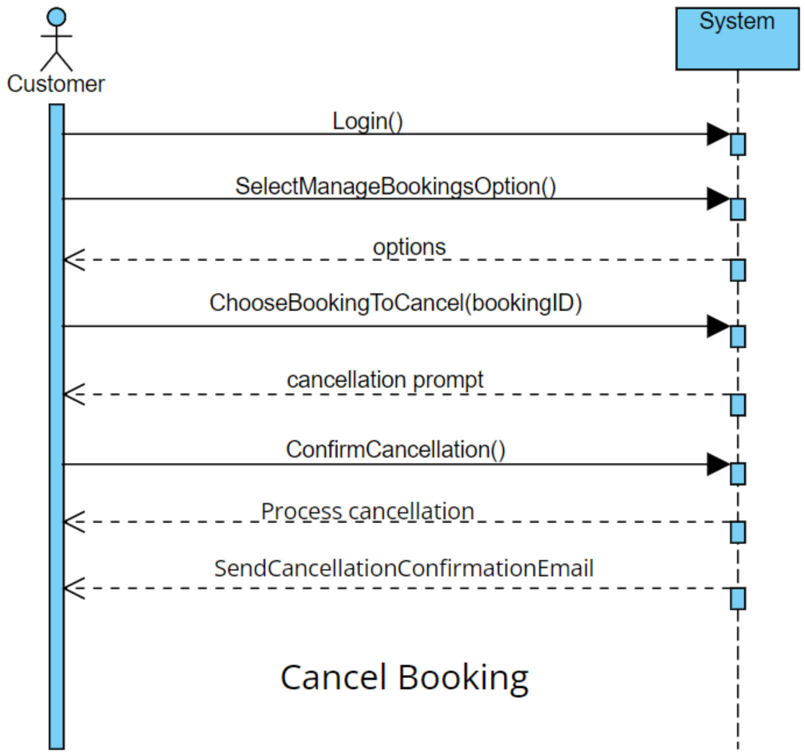
Check Travel History


Customer
System
Login()
SelectBookFlightOption()
options
EnterTravelDetails(departure, destination, dates)
DisplayAvailableFlights
SelectFlight(flight)
PromptPaymentDetails
EnterPaymentDetails(paymentInfo)
Payment confirmation
DisplayBookingConfirmation
Book A Flight

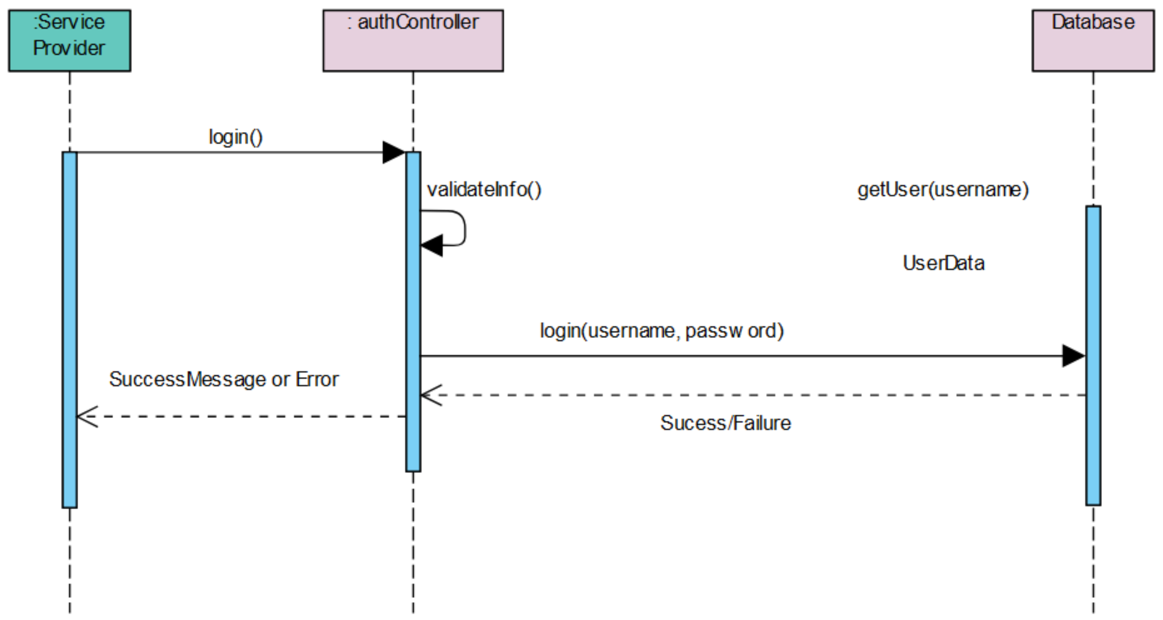

Customer
System
Login()
SelectBookTrainTicketOption()
options
EnterTravelDetails(departure, destination, dates)
AvailableTrainAndClasses
SelectTrainAndClass(train,class)
PromptPaymentDetails
EnterPaymentDetails(paymentInfo)
PaymentConfirmation
DisplayBookingConfirmation
Book A Train Ticket

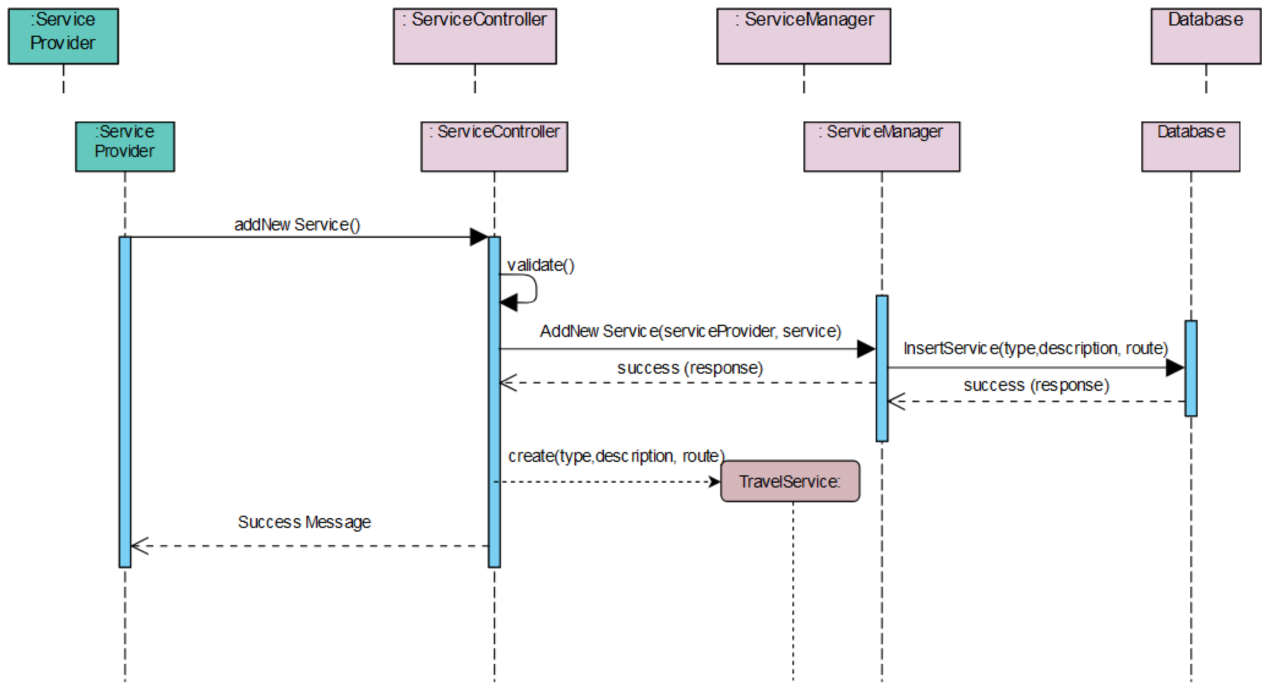

Customer
System
Login()
SelectBookBusTicketOption()
options
EnterTravelDetails(departure, destination, dates)
Available bus and seating options
SelectBusAndSeatingClass(bus,class)
Prompt payment details
EnterPaymentDetails(paymentInfo)
Payment confirmation
DisplayBookingConfirmation
Book A Bus Ticket


Customer
System
Login()
SelectBookHotelOption()
options
EnterHotelSearch(location, dates, preferences)
Hotel and room options
SelectHotelAndRoomType(hotel,roomType)
Payment details
EnterPaymentDetails(paymentInfo)
Payment confirmation
Booking confirmation
SendConfirmationMail
Book Hotel and Room

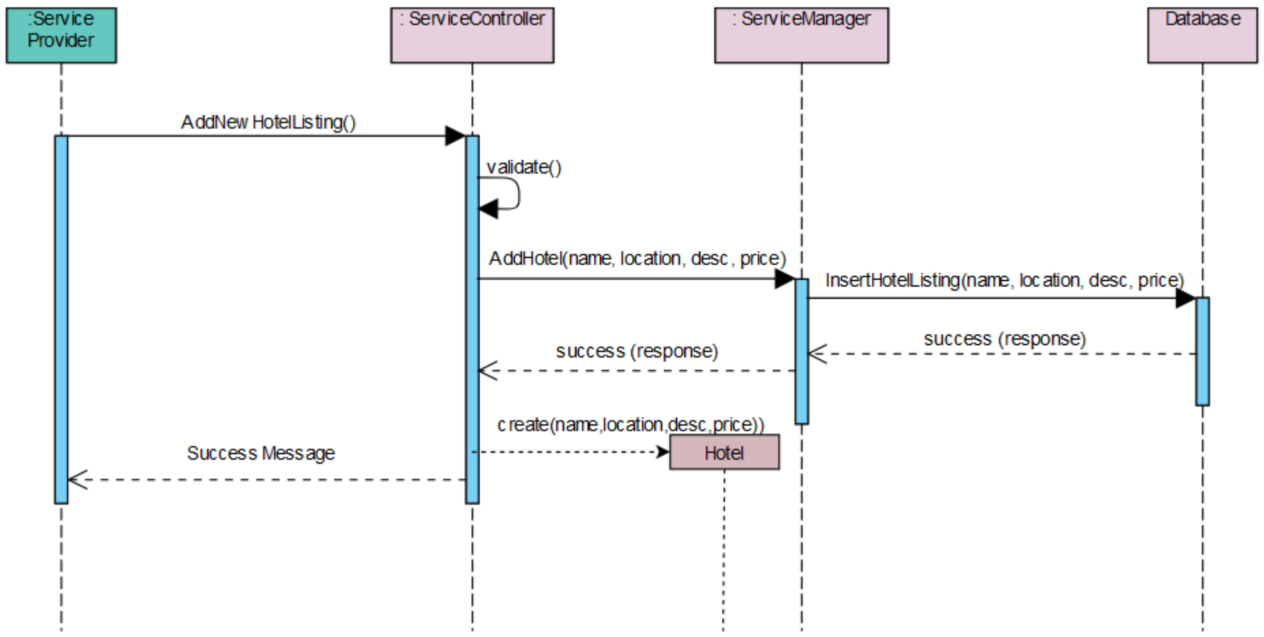

Customer
System
Login()
SelectManageBookingsOption()
options
ChooseBookingToCancel(bookingID)
cancellation prompt
ConfirmCancellation()
Process cancellation
SendCancellationConfirmationEmail
Cancel Booking

6. Sequence Diagram

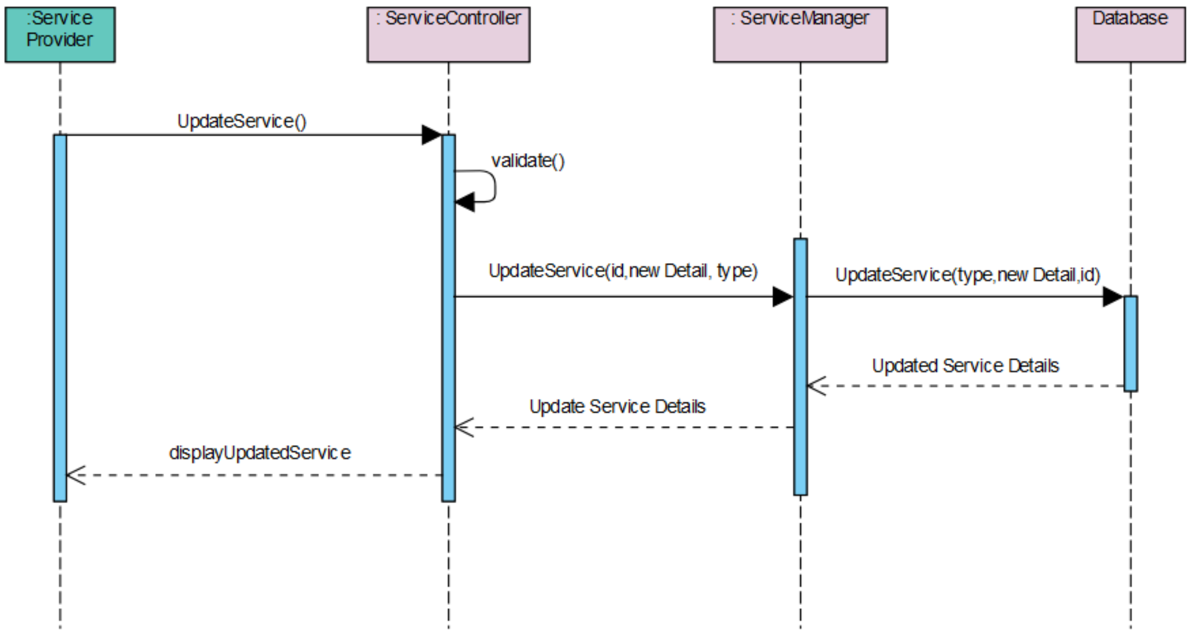




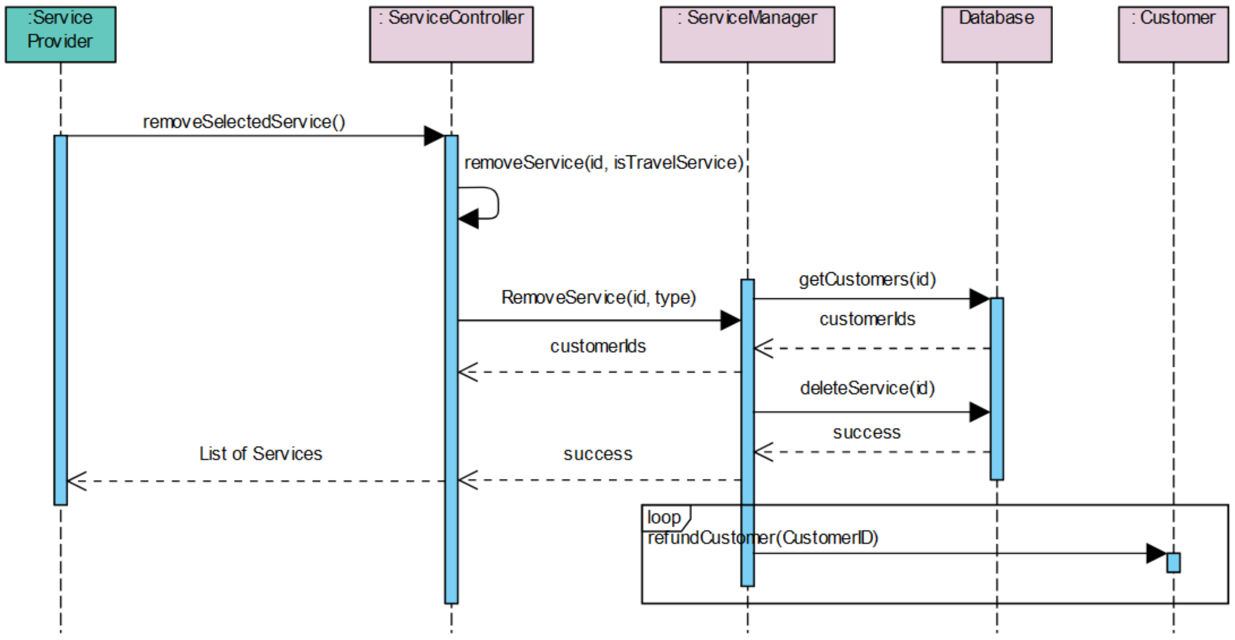
Add a New Service - AddNewService()



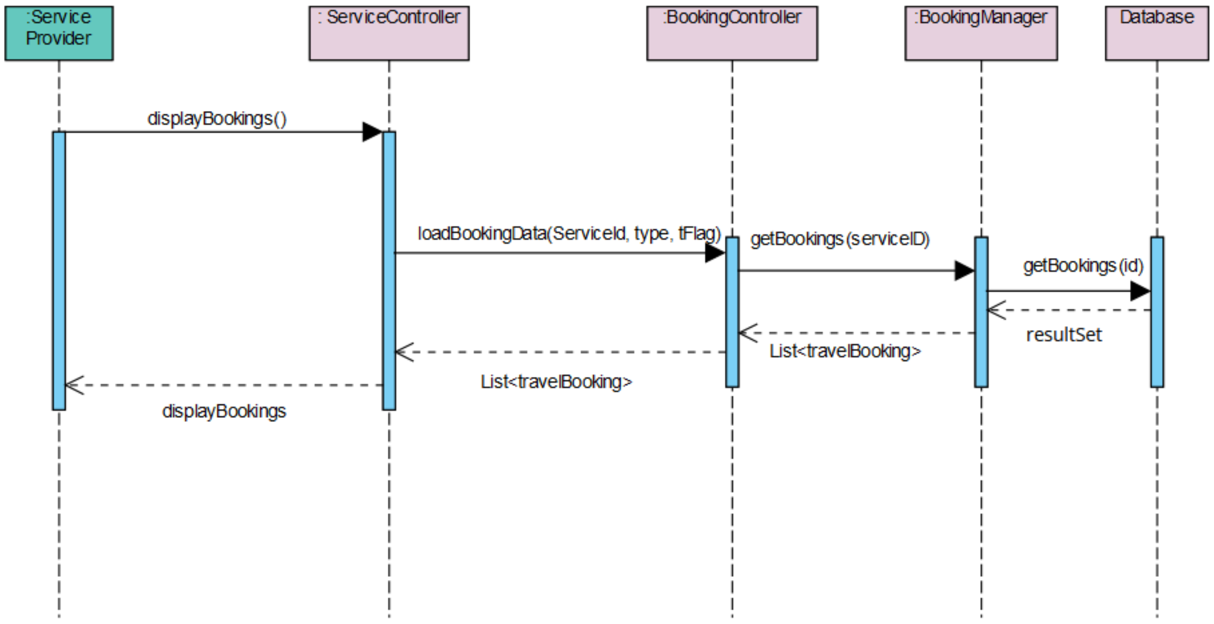
Add Hotel Listing - AddHotelListing()



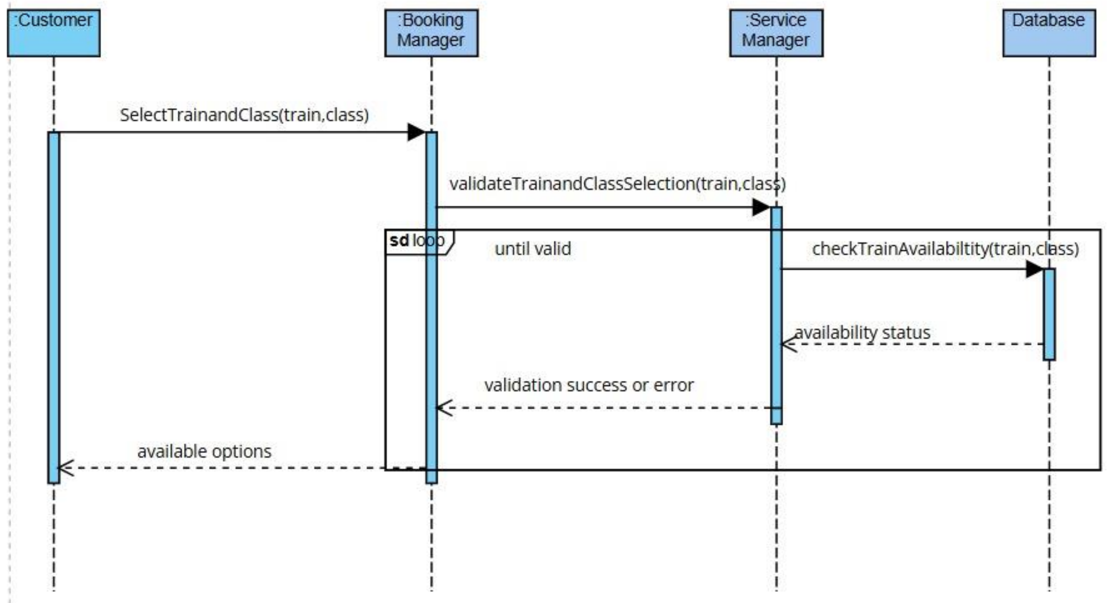
Edit Service Detail - EditDetail()



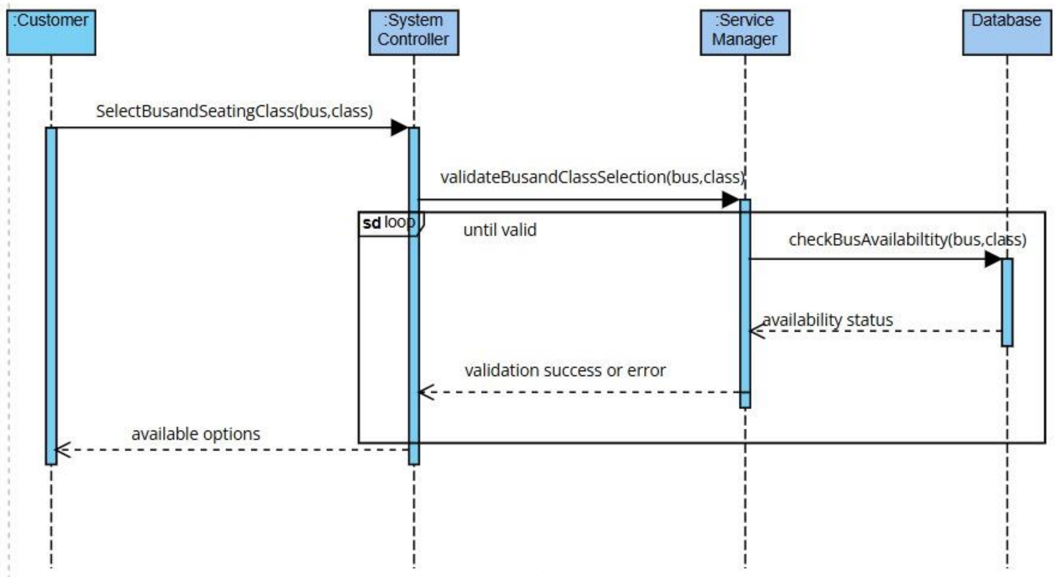
Remove/Cancel Service - removeService()



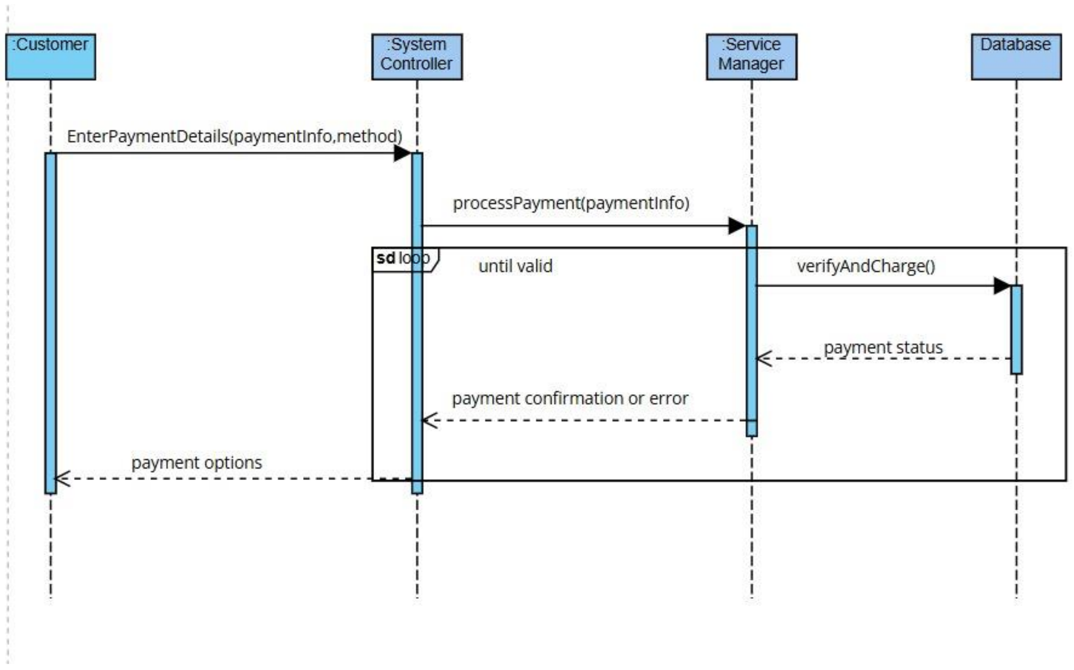
View Customer Bookings - ViewCustomerBookings()

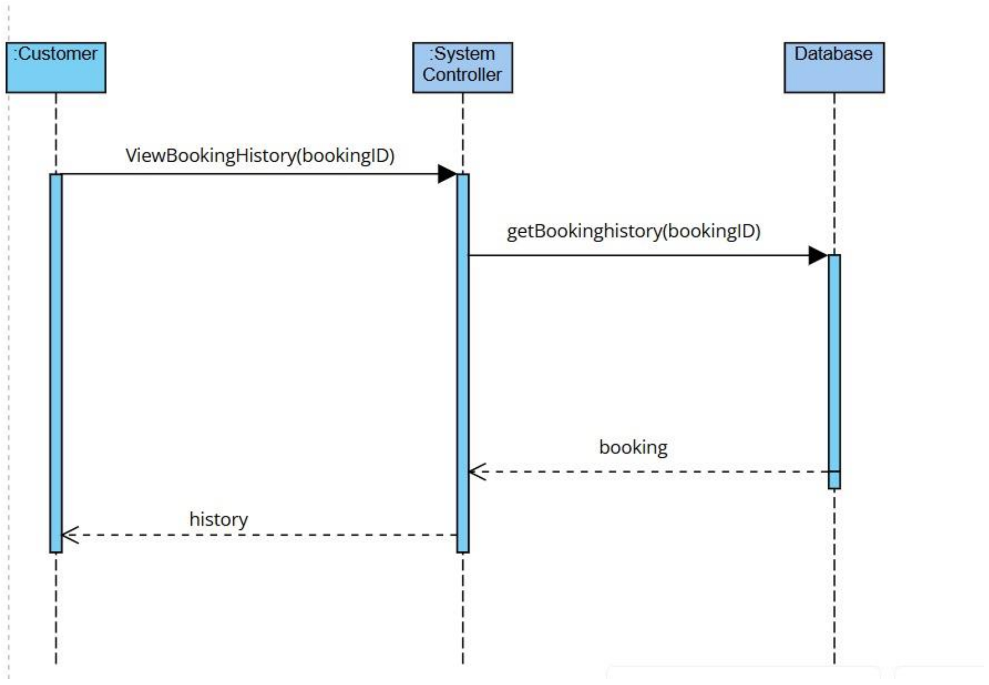

:Customer
:Booking Manager
:Service Manager
Database
SelectTrainandClass(train,class)
validateTrainandClassSelection(train,class)
sd loop
until valid
checkTrainAvailabiltity(train,class)
availability status
validation success or error
available options

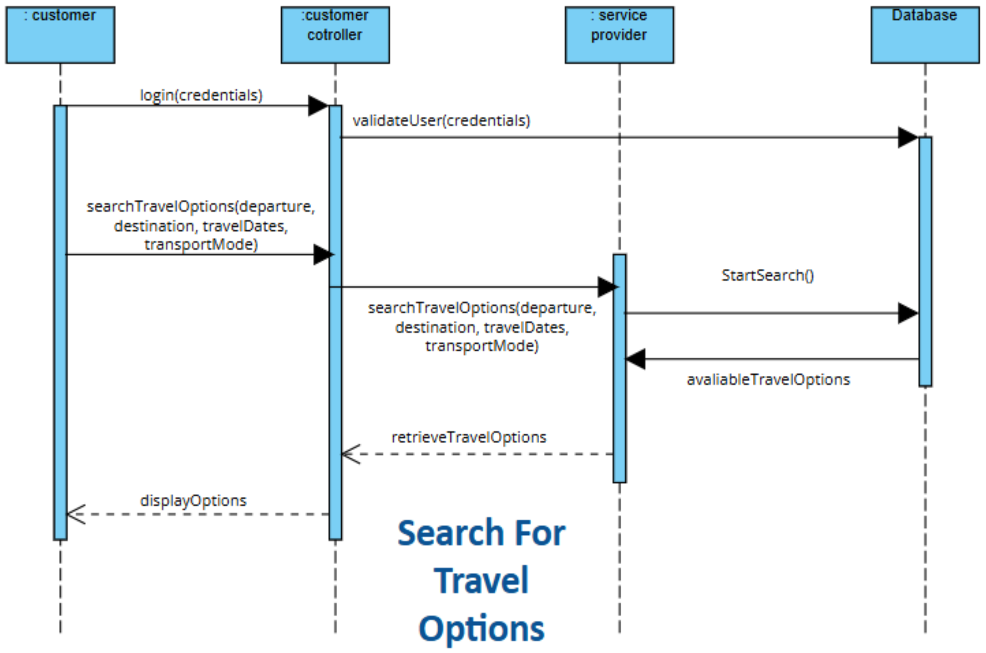
SelectTrainandClass() - for Booking a Train

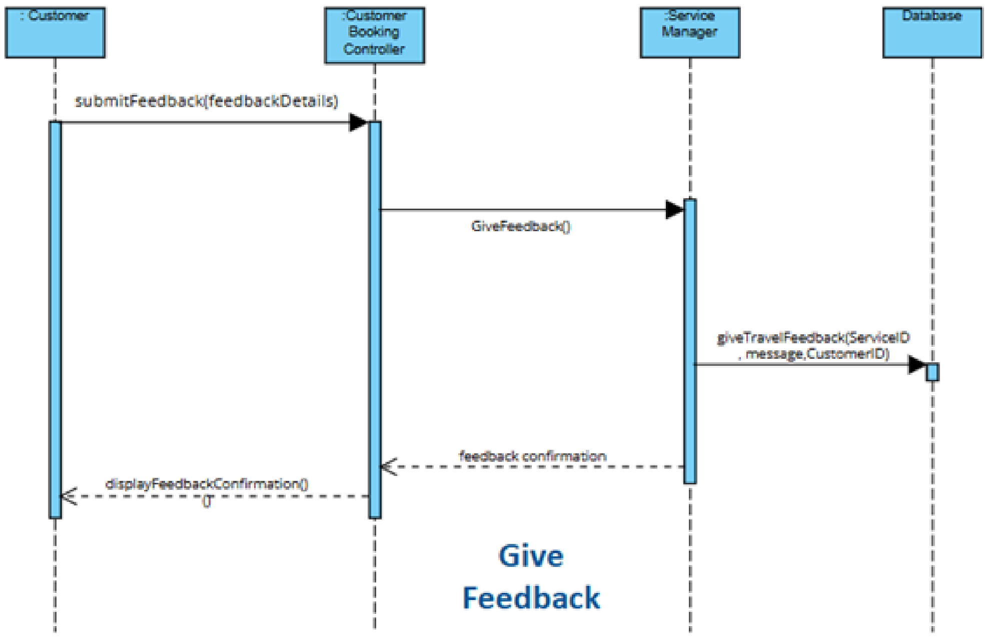

:Customer
:System Controller
:Service Manager
Database
SelectBusandSeatingClass(bus,class)
validateBusandClassSelection(bus,class)
sd loop
until valid
checkBusAvailabiltity(bus,class)
availability status
validation success or error
available options

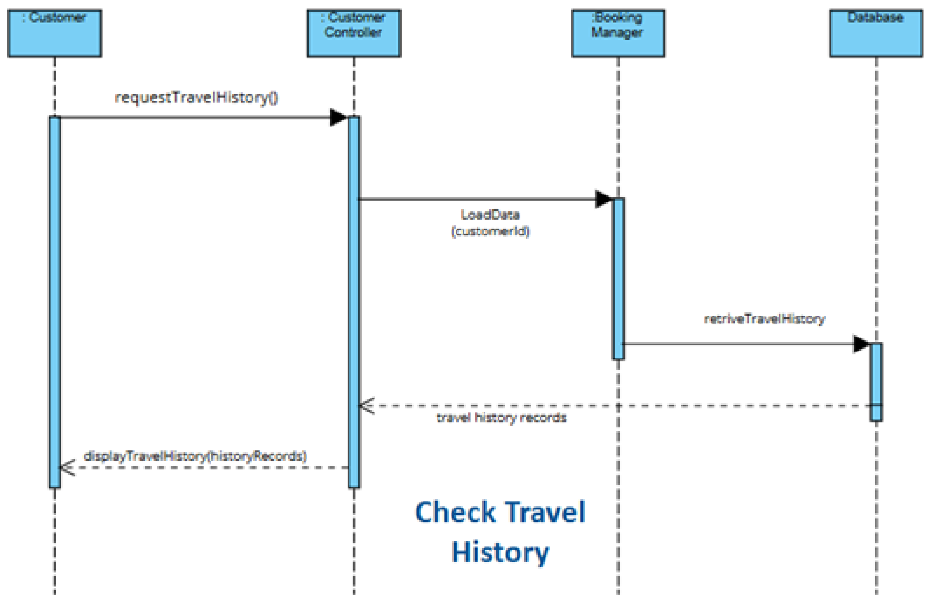
SelectBusAndSeatingClass() - Book a bus

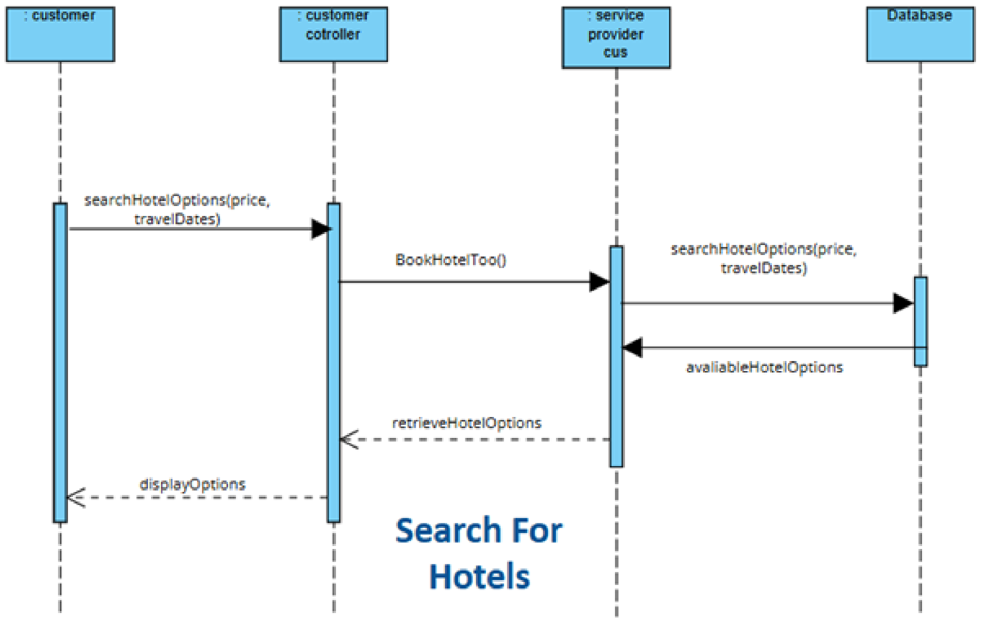

:Customer
:System Controller
:Service Manager
Database
EnterPaymentDetails(paymentInfo,method)
processPayment(paymentInfo)
sd loop
until valid
verifyAndCharge()
payment status
payment confirmation or error
payment options


:Customer
:System Controller
Database
ViewBookingHistory(bookingID)
getBookinghistory(bookingID)
booking
history

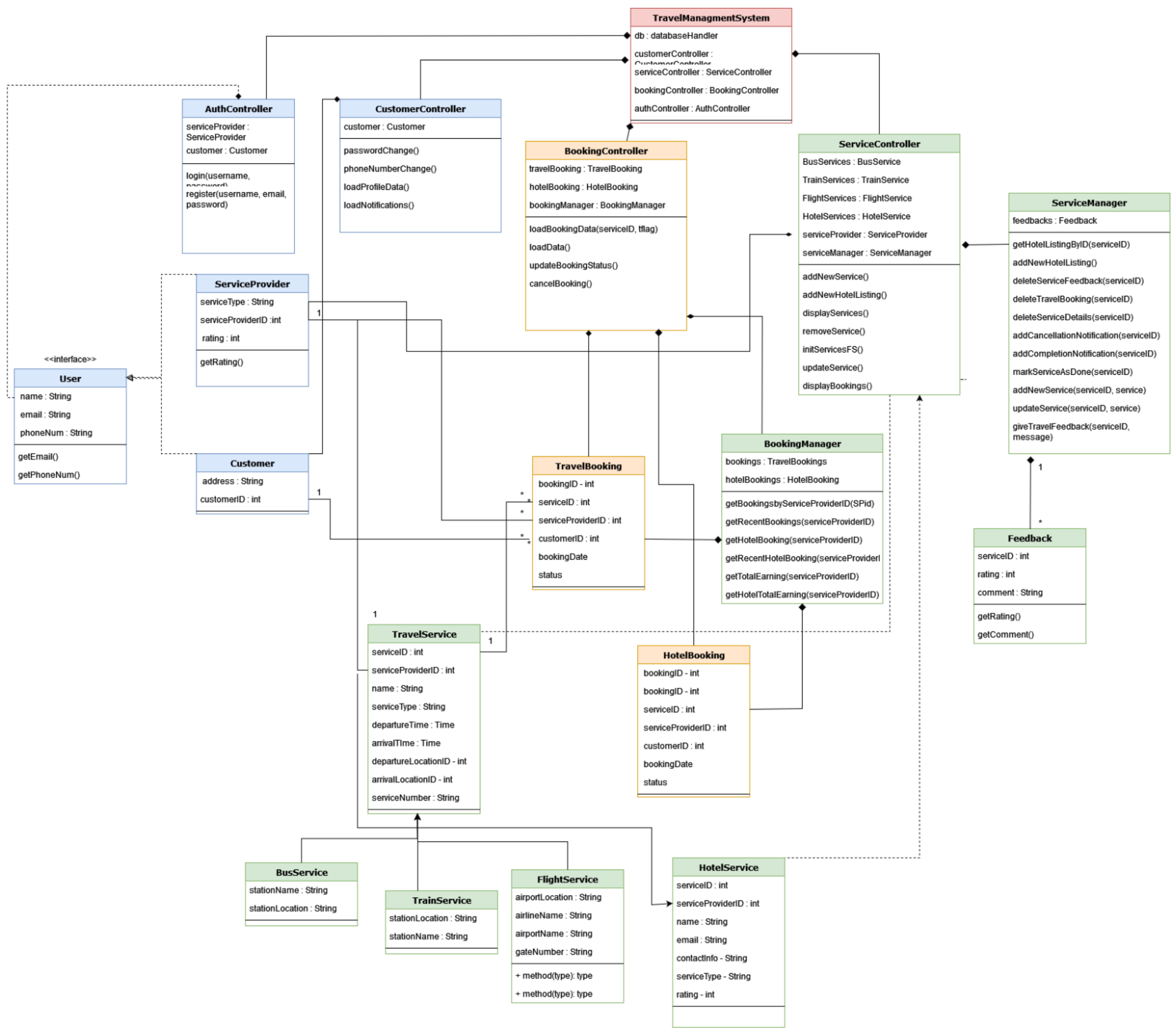

: customer
:customer cotroller
: service provider
Database
login(credentials)
validateUser(credentials)
searchTravelOptions(departure, destination, travelDates, transportMode)
StartSearch()
searchTravelOptions(departure, destination, travelDates, transportMode)
avaliableTravelOptions
retrieveTravelOptions
displayOptions
Search For Travel Options


: Customer
:Customer Booking Controller
:Service Manager
Database
submitFeedback(feedbackDetails)
GiveFeedback()
giveTravelFeedback(ServiceID , message,CustomerID)
feedback confirmation
displayFeedbackConfirmation() ()
Give Feedback

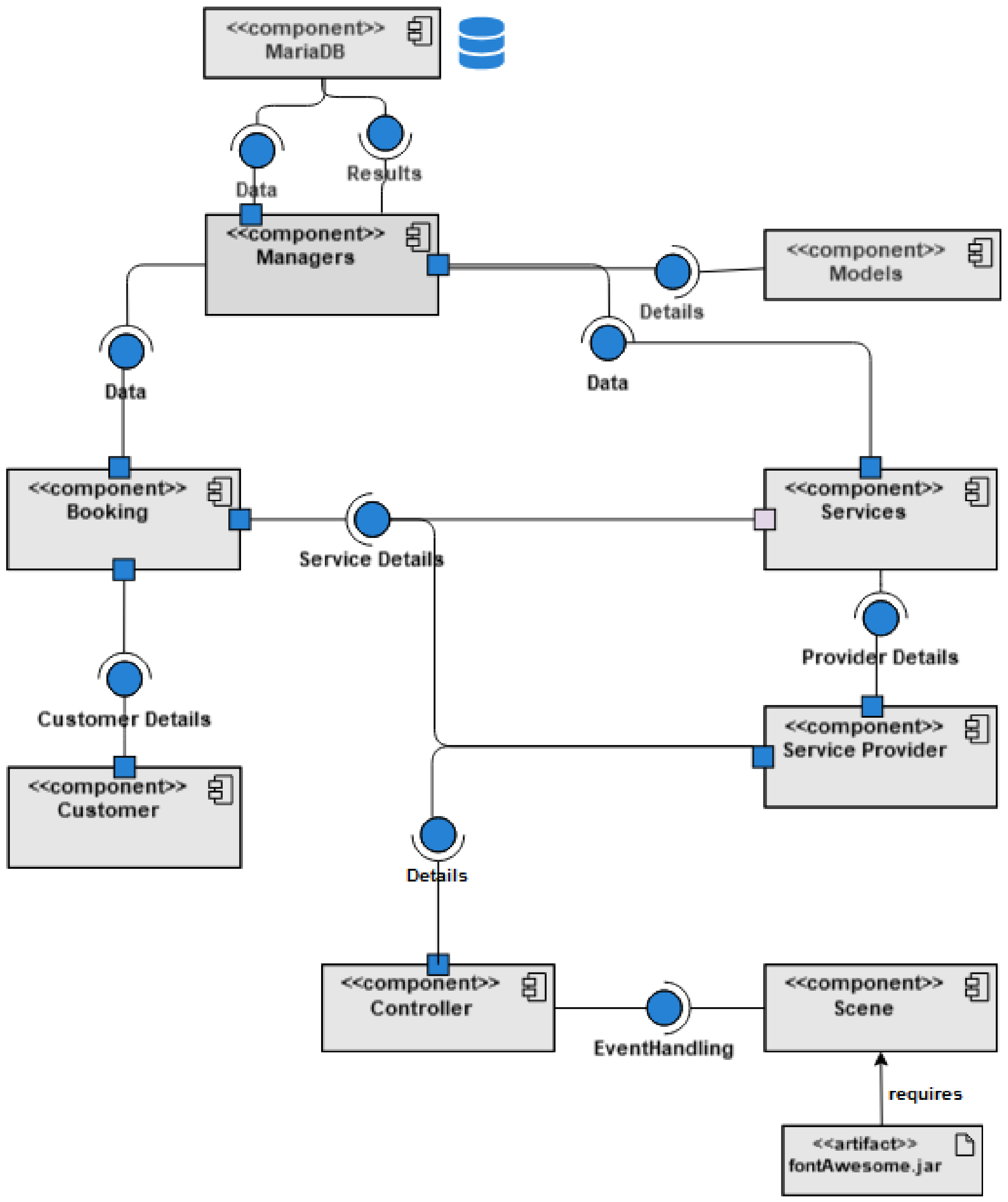

: Customer
: Customer Controller
:Booking Manager
Database
requestTravelHistory()
LoadData (customerId)
retriveTravelHistory
travel history records
displayTravelHistory(historyRecords)
Check Travel History


: customer
: customer cotroller
: service provider cus
Database
searchHotelOptions(price, travelDates)
BookHotelToo()
searchHotelOptions(price, travelDates)
avaliableHotelOptions
retrieveHotelOptions
displayOptions
Search For Hotels

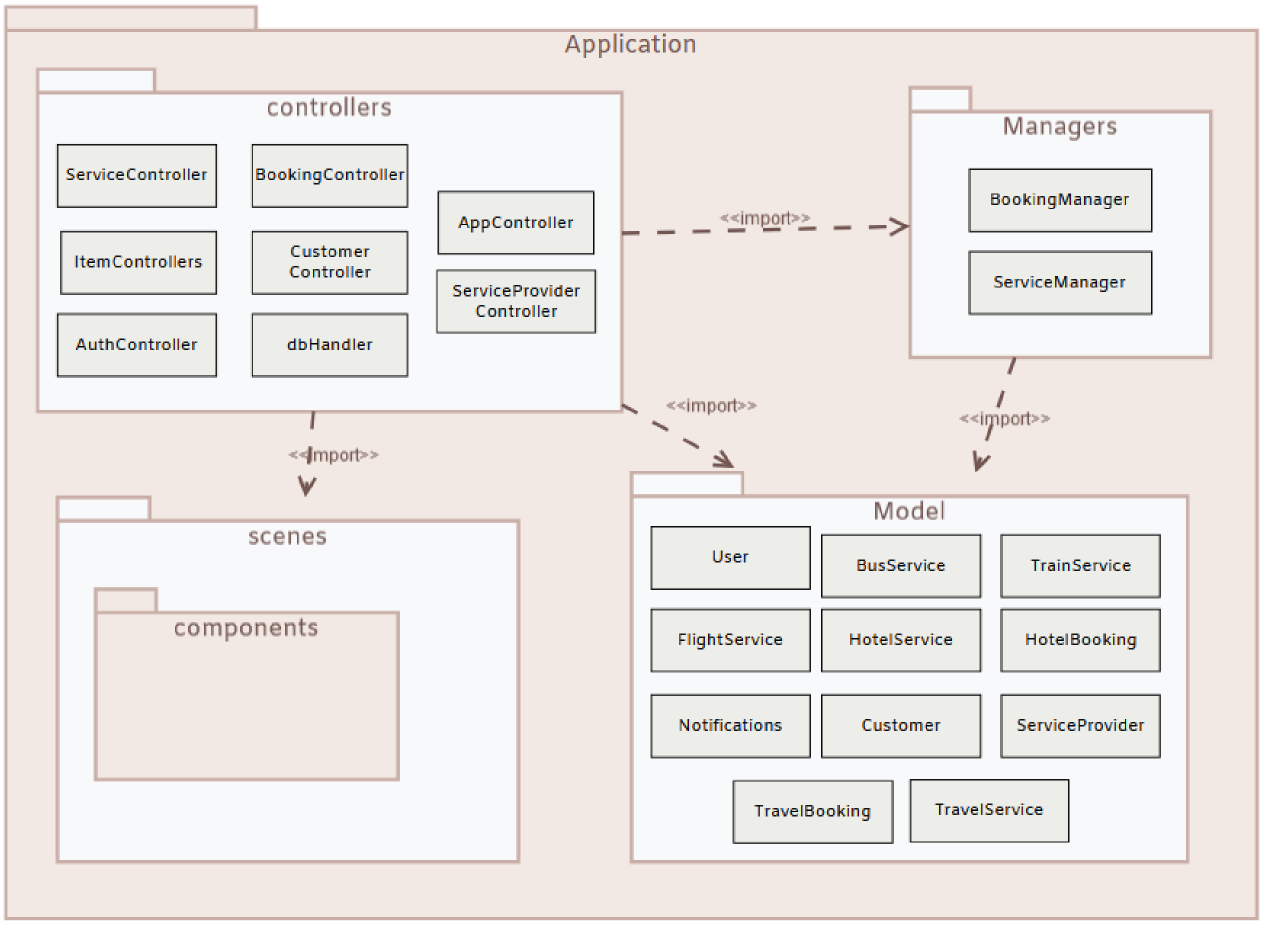
7. Class Diagram



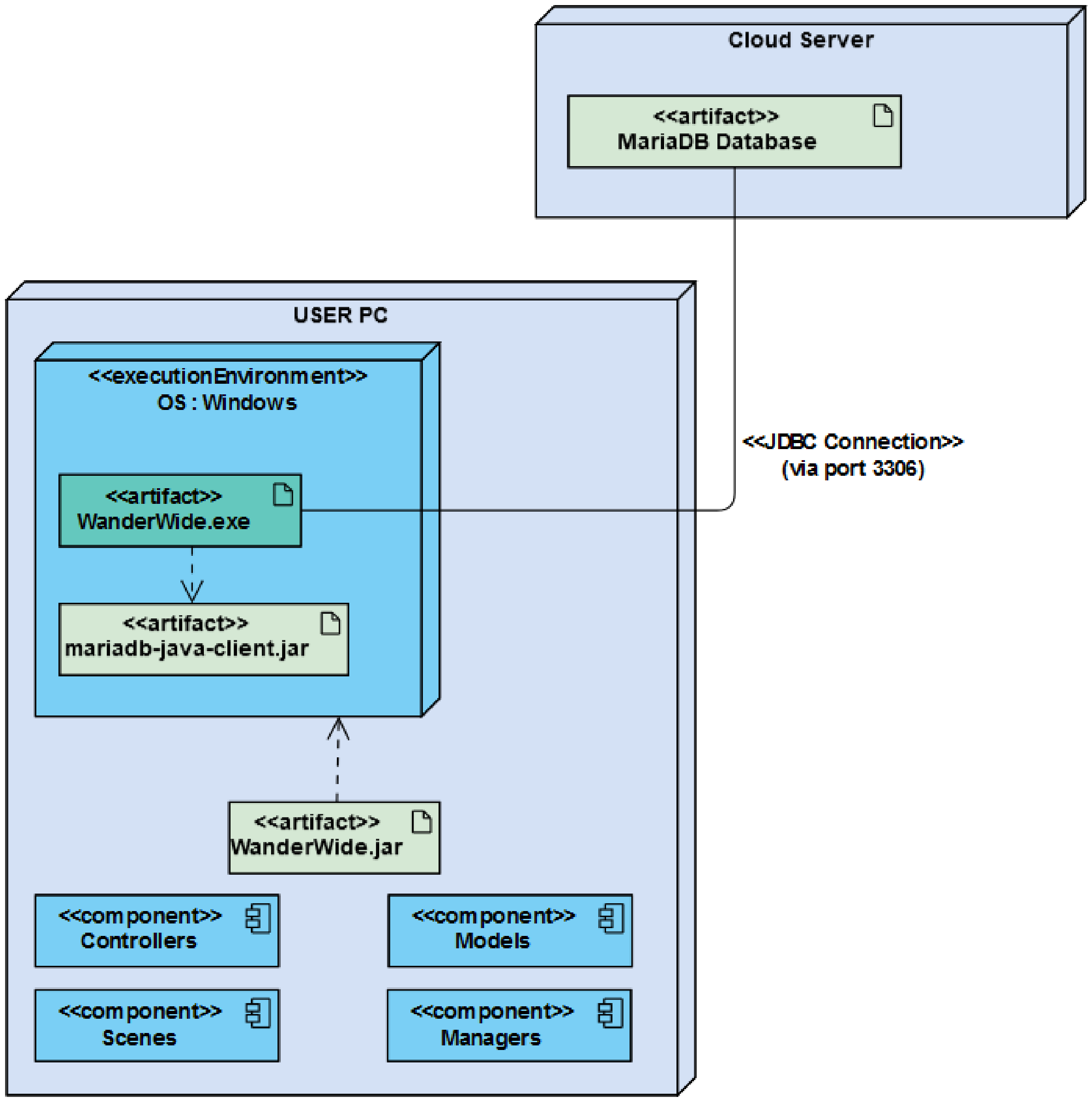
8. Component Diagram



9. Package Diagram



10. Deployment Diagram


Cloud Server
<<artifact>>
MariaDB Database
USER PC
<<executionEnvironment>>
OS : Windows
<<artifact>>
WanderWide.exe
<<artifact>>
mariadb-java-client.jar
<<JDBC Connection>>
(via port 3306)
<<artifact>>
WanderWide.jar
<<component>>
Controllers
<<component>>
Models
<<component>>
Scenes
<<component>>
Managers