SRS for Railway Reservation System Project

Table of Contents

1. Introduction

1.1 Purpose

The Railway Reservation System is designed to facilitate ticket booking, cancellation, and passenger management for a train service. The system provides functionalities for booking tickets, checking seat availability, displaying passenger details, and managing a waiting list.

1.2 Scope

This system is intended for use in a simple railway booking scenario where users can interact via a console interface. It handles a single train's reservations with features such as booking, cancellation, and a waiting list mechanism.

1.3 Definitions

- **Passenger:** A person traveling on the train.
- **Ticket:** A reservation record with a seat number.
- **Waiting List:** A queue for passengers who are waiting for seat allocation when all seats are booked.

2. Overall Description

2.1 Product Perspective

This is a standalone system developed in C++ and operates on a command-line interface. It is independent of external systems and focuses on managing seat reservations for a single train.

2.2 Product Features

- Book a ticket if seats are available.
- Add passengers to the waiting list if seats are full.
- Cancel tickets and allocate canceled seats to passengers on the waiting list.
- Display all booked passengers' details.
- Check seat availability.

2.3 User Characteristics

Users are expected to have basic knowledge of operating a console-based system.

2.4 Constraints

- The system handles only one train.
- Input is restricted to the console interface.
- Passenger names and ages must be entered manually.
- Only integer seat numbers are managed, formatted as "S1", "S2", etc.

2.5 Assumptions and Dependencies

- Passenger names are unique in the context of the waiting list.
- Age is optional for waiting list passengers.

3. Functional Requirements

3.1 Book Ticket

- Input: Passenger name and age.
- Process:
    - Assign a seat if available.
    - Add the passenger to the waiting list if no seats are available.
- Output: Display the allocated seat number or waiting list confirmation.

3.2 Cancel Ticket

- Input: Seat number.
- Process:
    - Remove the ticket from the booking list.
    - Allocate the vacant seat to the first passenger in the waiting list.
- Output: Display cancellation confirmation and reallocation status.

3.3 Check Seat Availability

- Input: None.
- Process: Calculate remaining seats by subtracting booked seats from total seats.
- Output: Display the number of available seats.

3.4 Display Passengers

- Input: None.
- Process: Retrieve passenger details from the booking list.
- Output: Display all passengers' names, ages, and seat numbers.

4. Non-Functional Requirements

4.1 Performance Requirements

- The system should handle operations efficiently for up to the total number of train seats.
- Waiting list operations should work seamlessly for small queues.

4.2 Usability Requirements

- The console interface must be user-friendly, with clear prompts and error messages.

4.3 Reliability Requirements

- The system should maintain data integrity between operations (e.g., no seat duplication).

4.4 Maintainability Requirements

- The code should be modular to allow future extension (e.g., adding multiple trains).

5. System Design

5.1 Data Structures

- **Passenger**: Struct to hold passenger information (name, age, seat number).
- **Train**: Class to manage train operations, including passenger bookings and cancellations.
- **Data Containers**:
    - map<string, Passenger>: Stores booked passengers with seat numbers as keys.
    - queue<string>: Manages the waiting list using passenger names.

5.2 Operations

- Methods in the Train class:
    - bookTicket(name, age)
    - cancelTicket(seatNo)
    - displayPassengers()
    - checkAvailability()

6. User Interface Design

- **Menu-driven Interface:**
    - Options for booking, canceling, viewing availability, displaying passengers, and exiting.
    - Clear prompts for input and concise error messages for invalid operations.

7. Testing Requirements

- Test cases to verify the following:
    - Booking tickets until the train is full.
    - Adding passengers to the waiting list.
    - Canceling tickets and verifying reassignment.
    - Checking seat availability at different stages.
    - Displaying accurate passenger details.

8. Future Enhancements

- Support for multiple trains.
- Graphical User Interface (GUI).
- Integration with a database for persistent storage.
- Enhanced error handling and input validation.