## Railway Ticket Reservation System

## Abstract

This project implements a Railway Ticket Reservation System, designed as a command-line application in Python. It provides users with functionalities such as ticket booking, cancellation, checking seat availability, and managing user accounts. The system employs a monolithic architecture, utilizes Python dictionaries for in-memory data management, and persists data across sessions using the pickle module. Future enhancements include database integration, a graphical user interface, and scalability improvements.

## 1. Introduction

The Railway Ticket Reservation System is a Python-based application for managing train reservations. It offers users the ability to book tickets, cancel them, and check seat availability. The project uses object-oriented programming principles, enabling modular and scalable code design. Its goal is to demonstrate a functional ticket reservation system, which can be extended to more complex use cases in the future.

## 2. System Overview

### Architecture:

The application follows a Three-Tier Architecture concept:  
1. Presentation Layer: Command-line interface for user interactions.  
2. Application Layer: Core logic for ticket and user management.  
3. Data Layer: Uses dictionaries for in-memory storage and pickle for data persistence.

## 3. Key Features

1. User Management:  
- Account creation with unique user IDs.  
- User authentication through login credentials.  
  
2. Booking System:  
- Train search based on source and destination.  
- Seat availability checks and fare display.  
- Ticket booking with automatic PNR generation.  
  
3. Ticket Management:  
- Ticket cancellation with seat availability updates.  
- Retrieval of booking history and PNR details.

## 4. Design and Implementation

### Core Classes:

1. Train Class:  
- Attributes: Train number, name, source, destination, schedule, seat availability, and fare.  
- Methods: check\_availability, book\_ticket, and print\_seat\_availability.  
  
2. Ticket Class:  
- Attributes: PNR, train number, user details, and ticket details.  
- Functionality: Links train and user for each booking.  
  
3. User Class:  
- Attributes: User ID, name, contact information, and booking history.  
- Functionality: Manages user data and interaction history.

### Menu and Workflow:

The menu-driven structure uses Python dictionaries to map user options to corresponding functions, implementing the command pattern for modularity.

## 5. System Design Patterns

1. Singleton Pattern: Ensures a single instance of train, user, and ticket data storage.  
2. Factory Pattern: Dynamically creates user and ticket objects.  
3. Command Pattern: Delegates menu options to respective functionalities.

## 6. Results

The application supports the following operations:  
  
1. Ticket Booking:  
- Successfully books tickets based on seat availability and updates the system.  
- PNR generation demonstrated with details like train number, user ID, and ticket count.  
  
2. Ticket Cancellation:  
- Allows users to cancel tickets and updates seat availability.  
  
3. Seat Availability Check:  
- Displays train details and availability for selected source-destination pairs.

## 7. UML Diagrams

Use Case Diagram:  
- Actors: User (books tickets, cancels tickets, checks availability, and creates accounts).  
- Use Cases: Login, create account, book ticket, cancel ticket, check seat availability, and PNR history.  
  
Class Diagram:  
- Classes: Train, Ticket, User.  
- Relationships:  
 - Train ↔ Ticket: 1:N (Each train has multiple tickets).  
 - User ↔ Ticket: 1:N (Each user can book multiple tickets).  
  
Sequence Diagram:  
 - Booking Workflow: User → Menu → Acceptors → Train → Ticket → User.

## 8. Enhancements for Scalability

1. Database Integration: Replace pickle with SQL/NoSQL databases.  
2. GUI/Frontend Development: Implement a web-based or desktop application.  
3. Logging and Testing: Add robust logging and test automation.

## 9. Conclusion

This project demonstrates a working model of a railway ticket reservation system with essential features. Its modular design makes it a good candidate for enhancements such as GUI development and database integration, paving the way for real-world applications.

10. **Contributions**

**Renuka Pathlouth:**

* Designed and implemented the **core functionalities** of the Railway Reservation System, including ticket booking, cancellation, and seat availability checks.
* Developed the **class definitions** for train, ticket, and user, enabling object-oriented management of trains, tickets, and user data.
* Added methods for **booking tickets, validating availability, and generating PNRs**, ensuring smooth user operations.
* Integrated the **data persistence mechanism** using the pickle module for saving and loading the state of trains, users, and tickets across sessions.
* Defined **menu-driven functionality** using the command pattern to streamline user interaction.

**Hari Varma Nagaraju:**

* Structured the **system's architecture** and provided conceptual documentation for three-tier separation (presentation, application, and data layers).
* Introduced the use of **design patterns** like Singleton, Factory, and Command to enhance code maintainability and scalability.
* Conceptualized and documented **UML diagrams**, including use case, class, sequence, and activity diagrams, to visualize the system's design and workflows.
* Identified **enhancements for scalability**, such as database integration, GUI development, and decoupling of application layers for better maintainability.
* Reviewed and tested the codebase, ensuring robust performance and addressing edge cases in the booking and cancellation processes.