# Problem Statement

The Online Reservation System aims to provide a centralized system where users can log in, make reservations for trains, and cancel their bookings. The system reduces manual workload, ensures easy access to reservation data, and enhances the user experience by providing a simple interface.

# Algorithm

1. Start the system.  
2. User logs in using Login ID and Password.  
3. Display menu options: Make Reservation, View Reservations, Cancel Reservation, Exit.  
4. If Make Reservation is selected:  
 a. Enter passenger details (Name, Train No., Train Name, Class, Journey Date, Source, Destination).  
 b. Generate a unique PNR and save details.  
5. If View Reservations is selected:  
 a. Display list of all reservations with details.  
6. If Cancel Reservation is selected:  
 a. Enter PNR number.  
 b. Search and remove reservation if found.  
7. If Exit is selected: terminate the system.  
8. Stop.

# Source Code

import java.util.\*;  
  
class Reservation {  
 private int pnr;  
 private String passengerName;  
 private String trainNo;  
 private String trainName;  
 private String classType;  
 private String journeyDate;  
 private String source;  
 private String destination;  
  
 public Reservation(int pnr, String passengerName, String trainNo, String trainName, String classType, String journeyDate, String source, String destination) {  
 this.pnr = pnr;  
 this.passengerName = passengerName;  
 this.trainNo = trainNo;  
 this.trainName = trainName;  
 this.classType = classType;  
 this.journeyDate = journeyDate;  
 this.source = source;  
 this.destination = destination;  
 }  
  
 public int getPnr() {  
 return pnr;  
 }  
  
 public String toString() {  
 return "PNR: " + pnr + ", Name: " + passengerName + ", Train: " + trainNo + " - " + trainName +   
 ", Class: " + classType + ", Date: " + journeyDate + ", From: " + source + " To: " + destination;  
 }  
}  
  
class ReservationSystem {  
 private List<Reservation> reservations = new ArrayList<>();  
 private int nextPnr = 1001;  
  
 public Reservation makeReservation(String name, String trainNo, String trainName, String classType, String date, String source, String destination) {  
 Reservation reservation = new Reservation(nextPnr++, name, trainNo, trainName, classType, date, source, destination);  
 reservations.add(reservation);  
 return reservation;  
 }  
  
 public List<Reservation> getReservations() {  
 return reservations;  
 }  
  
 public Reservation getReservationByPnr(int pnr) {  
 for (Reservation r : reservations) {  
 if (r.getPnr() == pnr) return r;  
 }  
 return null;  
 }  
  
 public boolean cancelReservation(int pnr) {  
 Reservation reservation = getReservationByPnr(pnr);  
 if (reservation != null) {  
 reservations.remove(reservation);  
 return true;  
 }  
 return false;  
 }  
}  
  
class ReservationSystemUI {  
 private ReservationSystem reservationSystem = new ReservationSystem();  
 private Scanner scanner = new Scanner(System.in);  
  
 public void login() {  
 System.out.print("Enter Login ID: ");  
 String loginId = scanner.nextLine();  
 System.out.print("Enter Password: ");  
 String password = scanner.nextLine();  
 if (loginId.equals("admin") && password.equals("admin")) {  
 start();  
 } else {  
 System.out.println("Invalid Login. Try again.");  
 login();  
 }  
 }  
  
 public void start() {  
 while (true) {  
 System.out.println("1. Make a reservation");  
 System.out.println("2. View all reservations");  
 System.out.println("3. Cancel a reservation");  
 System.out.println("4. Exit");  
 int choice = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (choice) {  
 case 1:  
 System.out.print("Passenger Name: ");  
 String name = scanner.nextLine();  
 System.out.print("Train No: ");  
 String trainNo = scanner.nextLine();  
 System.out.print("Train Name: ");  
 String trainName = scanner.nextLine();  
 System.out.print("Class Type: ");  
 String classType = scanner.nextLine();  
 System.out.print("Journey Date: ");  
 String date = scanner.nextLine();  
 System.out.print("Source: ");  
 String source = scanner.nextLine();  
 System.out.print("Destination: ");  
 String destination = scanner.nextLine();  
 Reservation r = reservationSystem.makeReservation(name, trainNo, trainName, classType, date, source, destination);  
 System.out.println("Reservation made successfully. PNR: " + r.getPnr());  
 break;  
 case 2:  
 for (Reservation res : reservationSystem.getReservations()) {  
 System.out.println(res);  
 }  
 break;  
 case 3:  
 System.out.print("Enter PNR to cancel: ");  
 int pnr = scanner.nextInt();  
 if (reservationSystem.cancelReservation(pnr)) {  
 System.out.println("Reservation Cancelled Successfully.");  
 } else {  
 System.out.println("PNR not found.");  
 }  
 break;  
 case 4:  
 return;  
 default:  
 System.out.println("Invalid Choice.");  
 }  
 }  
 }  
  
 public static void main(String[] args) {  
 ReservationSystemUI ui = new ReservationSystemUI();  
 ui.login();  
 }  
}

# Output

Sample Run:  
Enter Login ID: admin  
Enter Password: admin  
  
1. Make a reservation  
2. View all reservations  
3. Cancel a reservation  
4. Exit  
  
>> 1  
Passenger Name: John  
Train No: 12345  
Train Name: Express  
Class Type: Sleeper  
Journey Date: 25-09-2025  
Source: Delhi  
Destination: Mumbai  
Reservation made successfully. PNR: 1001  
  
>> 2  
PNR: 1001, Name: John, Train: 12345 - Express, Class: Sleeper, Date: 25-09-2025, From: Delhi To: Mumbai  
  
>> 3  
Enter PNR to cancel: 1001  
Reservation Cancelled Successfully.  
  
>> 2  
(no reservations displayed)

# Explanation of Code

1. Reservation class stores individual booking details with attributes like PNR, Passenger Name, Train No, etc.  
2. ReservationSystem manages the list of reservations, provides methods to add, search, and cancel bookings.  
3. ReservationSystemUI handles user interaction, login validation, and menu-driven options.  
4. Login validation ensures only authorized users can access the system.

# Features to be Added in Future

- Online payment integration  
- Email/SMS confirmation  
- Multiple user roles (Admin, Passenger)  
- Real-time train availability  
- Graphical User Interface (GUI)

# Conclusion

The Online Reservation System simplifies the process of booking and cancelling train tickets. It provides an easy-to-use console-based interface and ensures data consistency with a centralized structure. Further improvements can enhance the user experience with real-time data, GUI, and integration with online services.