SOURCE CODE

public class Train {

private int trainNumber;

private String trainName;

private String source;

private String destination;

private int availableSeats;

// Constructor

public Train(int trainNumber, String trainName, String source, String destination, int availableSeats) {

this.trainNumber = trainNumber;

this.trainName = trainName;

this.source = source;

this.destination = destination;

this.availableSeats = availableSeats;

}

// Getters and setters

public int getTrainNumber() {

return trainNumber;

}

public String getTrainName() {

return trainName;

}

public String getSource() {

return source;

}

public String getDestination() {

return destination;

}

public int getAvailableSeats() {

return availableSeats;

}

// Method to book a ticket

public void bookTicket() {

if (availableSeats > 0) {

availableSeats--;

System.out.println("Ticket booked successfully!");

} else {

System.out.println("Sorry, no seats available on this train.");

}

}

// Method to cancel a ticket

public void cancelTicket() {

availableSeats++;

System.out.println("Ticket cancelled successfully.");

}

}

public class Station {

private String stationName;

// Constructor

public Station(String stationName) {

this.stationName = stationName;

}

// Getter

public String getStationName() {

return stationName;

}

}

public class Ticket {

private int ticketNumber;

private String passengerName;

private Train train;

// Constructor

public Ticket(int ticketNumber, String passengerName, Train train) {

this.ticketNumber = ticketNumber;

this.passengerName = passengerName;

this.train = train;

}

// Getters

public int getTicketNumber() {

return ticketNumber;

}

public String getPassengerName() {

return passengerName;

}

public Train getTrain() {

return train;

}

}

public class User {

private String username;

private String password;

// Constructor

public User(String username, String password) {

this.username = username;

this.password = password;

}

// Getters

public String getUsername() {

return username;

}

public String getPassword() {

return password;

}

}

import java.util.ArrayList;

import java.util.List;

public class ReservationSystem {

private List<Train> trains;

private List<Station> stations;

private List<Ticket> tickets;

// Constructor

public ReservationSystem() {

this.trains = new ArrayList<>();

this.stations = new ArrayList<>();

this.tickets = new ArrayList<>();

}

// Method to add a train

public void addTrain(Train train) {

trains.add(train);

}

// Method to add a station

public void addStation(Station station) {

stations.add(station);

}

// Method to display available trains

public void displayAvailableTrains() {

System.out.println("Available Trains:");

for (Train train : trains) {

System.out.println(train.getTrainNumber() + " - " + train.getTrainName() + " from " + train.getSource() + " to " + train.getDestination() + " (" + train.getAvailableSeats() + " seats available)");

}

}

// Method to book a ticket

public void bookTicket(User user, Train train) {

if (train.getAvailableSeats() > 0) {

Ticket ticket = new Ticket(tickets.size() + 1, user.getUsername(), train);

tickets.add(ticket);

train.bookTicket();

System.out.println("Ticket booked successfully for " + user.getUsername());

} else {

System.out.println("Sorry, no seats available on this train.");

}

}

// Method to cancel a ticket

public void cancelTicket(User user, Train train) {

for (Ticket ticket : tickets) {

if (ticket.getPassengerName().equals(user.getUsername()) && ticket.getTrain().equals(train)) {

train.cancelTicket();

tickets.remove(ticket);

System.out.println("Ticket cancelled successfully for " + user.getUsername());

return;

}

}

System.out.println("Ticket not found for " + user.getUsername() + " on train " + train.getTrainNumber());

}

}

public class Main {

public static void main(String[] args) {

ReservationSystem reservationSystem = new ReservationSystem();

// Adding stations

Station stationA = new Station("Station A");

Station stationB = new Station("Station B");

reservationSystem.addStation(stationA);

reservationSystem.addStation(stationB);

// Adding trains

Train train1 = new Train(101, "Express One", "Station A", "Station B", 50);

Train train2 = new Train(102, "Express Two", "Station B", "Station A", 40);

reservationSystem.addTrain(train1);

reservationSystem.addTrain(train2);

// Display available trains

reservationSystem.displayAvailableTrains();

// Sample user

User user = new User("john.doe", "password");

// Booking a ticket

reservationSystem.bookTicket(user, train1);

reservationSystem.bookTicket(user, train1); // This will show no seats available

// Canceling a ticket

reservationSystem.cancelTicket(user, train1);

// Display available trains again

reservationSystem.displayAvailableTrains();

}

}