**MOVIE TICKET BOOKING SIMULATOR**

package frontdesk.moviebooking;

import frontdesk.moviebooking.exceptions.InvalidTicketRangeException;

import frontdesk.moviebooking.exceptions.TicketAlreadyBookedException;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class MovieBookingSystem {

    private final BookingData bookingData;

    private final Theatre theatre;

    public MovieBookingSystem() {

        bookingData = new BookingData();

        theatre = new Theatre();

    }

    public void displayShowTimes() {

        System.out.print("Available times: ");

        String showTimes = String.join("    ", theatre.showTimes);

        System.out.println(showTimes);

    }

    public void displaySeatingArrangement(String selectedDate, String selectedShowTime) {

        System.out.println("Seating Arrangement for " + selectedDate + " at " + selectedShowTime + ":");

        for (int i = 0; i < theatre.seats.length; i++) {

            for (int j = 0; j < theatre.seats[i].length; j++) {

                String seat = theatre.seats[i][j];

                boolean isBooked = bookingData.isSeatBooked(selectedDate, selectedShowTime, seat);

                if(isBooked) {

                    System.out.print("\u001B[31m" + seat + " " + "\u001B[0m");

                }

                else{

                    System.out.print(seat+" ");

                }

            }

            System.out.println();

        }

    }

    public List<String> getSeats(String seatsInput) throws  InvalidTicketRangeException {

        List<String> allSeats=new ArrayList<>();

        String[] seatRanges = seatsInput.split(",");

        for (String seatRange : seatRanges) {

            String[] seats = seatRange.split("-");

            if (seats.length == 1) {

                allSeats.add(seats[0]);

            } else if (seats.length == 2) {

                String startSeat=seats[0];

                String endSeat=seats[1];

                char startRow = startSeat.charAt(0);

                char endRow = endSeat.charAt(0);

                if(startRow!=endRow){

                    throw new InvalidTicketRangeException("Invalid Ticket Range "+seatRange);

                }

                int startCol = Integer.parseInt(startSeat.substring(1));

                int endCol = Integer.parseInt(endSeat.substring(1));

                if (startCol>4 || endCol<=startCol){

                    throw new InvalidTicketRangeException("Invalid Ticket Range "+seatRange);

                }

                for(int i=startCol;i<=endCol;i++){

                    allSeats.add(String.valueOf(startRow)+i);

                }

            }

        }

        return allSeats;

    }

    private List<String> checkAvailability(String selectedDate, String selectedShowTime, String seatInp) throws TicketAlreadyBookedException, InvalidTicketRangeException {

        List<String> selectedSeats = getSeats(seatInp);

        for(String seat : selectedSeats){

            if(bookingData.isSeatBooked(selectedDate,selectedShowTime,seat)){

                throw new TicketAlreadyBookedException("Ticket :"+seat+" is already Booked");

            }

        }

        return selectedSeats;

    }

    public List<String> bookSeats(String selectedDate, String selectedShowTime, String seatInp) throws TicketAlreadyBookedException, InvalidTicketRangeException {

        List<String> selectedSeats = checkAvailability(selectedDate,selectedShowTime,seatInp);

        for(String seat:selectedSeats){

            bookingData.bookSeat(selectedDate,selectedShowTime,seat);

        }

        return selectedSeats;

    }

    public List<String> getShowTimes(){

        return Arrays.stream(theatre.showTimes).collect(Collectors.toList());

    }

}

package frontdesk.moviebooking;

import frontdesk.moviebooking.exceptions.InvalidTicketRangeException;

import frontdesk.moviebooking.exceptions.TicketAlreadyBookedException;

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.time.format.DateTimeParseException;

import java.util.List;

import java.util.Scanner;

public class ConsoleApp {

    static MovieBookingSystem bookingSystem = new MovieBookingSystem();

    static Scanner scanner = new Scanner(System.in);

    static DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd");

    private static String getValidDate(){

        boolean isValidDate=false;

        String selectedDate=null;

        while (!isValidDate) {

            System.out.print("Enter the date (e.g., 2024-01-03) :- ");

            selectedDate = scanner.next();

            try {

                LocalDate date = LocalDate.parse(selectedDate, dtf);

                isValidDate=true;

            } catch (DateTimeParseException e) {

                System.out.println("Invalid date format. Please enter a date in the format yyyy-MM-dd.");

            }

        }

        return selectedDate;

    }

    private static String getValidShowTime(List<String> availableShowTimes){

        boolean isValidShowTime = false;

        String validShowTime = null;

        while (!isValidShowTime) {

            System.out.print("Enter the show time (e.g., 10:00AM) :- ");

            validShowTime= scanner.next();

            if(availableShowTimes.contains(validShowTime)){

                isValidShowTime = true;

            }

            else {

                System.out.println("Invalid show time");

                System.out.print("Please ");

            }

        }

        return validShowTime;

    }

    private static void showAvailableSeats(){

        String selectedDate=getValidDate();

        System.out.println("Selected Date :- "+selectedDate);

        bookingSystem.displayShowTimes();

        String selectedTime=getValidShowTime(bookingSystem.getShowTimes());

        System.out.println("Selected Time :- "+selectedTime);

        bookingSystem.displaySeatingArrangement(selectedDate, selectedTime);

    }

    public static void bookSeats(){

        System.out.print("Please Enter Customer Name :- ");

        String customerName=scanner.nextLine();

        String selectedDate=getValidDate();

        bookingSystem.displayShowTimes();

        String selectedTime=getValidShowTime(bookingSystem.getShowTimes());

        System.out.print("\nEnter the seat to book (e.g., A1 or B1-B4 or A1-A3,B1) :- ");

        String selectedSeats = scanner.next();

        try {

            List<String> seatsBooked = bookingSystem.bookSeats(selectedDate, selectedTime, selectedSeats);

            System.out.println("Booking Confirmation");

            System.out.println("\nSelected Date  :- "+selectedDate);

            System.out.println("Selected Time  :- "+selectedTime);

            System.out.println("Selected Seats :- "+String.join(",",seatsBooked));

            System.out.print("\nEnter Price of Each Ticket :- ");

            double price=scanner.nextDouble();

            String[] seats=new String[seatsBooked.size()];

            Tickets ticket=new Tickets(customerName,seatsBooked.size(),price,LocalDate.parse(selectedDate,dtf),selectedTime,seatsBooked.toArray(seats));

            System.out.println("\nNo Of Tickets :- "+ticket.noOfTickets);

            System.out.println("Amount To Paid :- "+(ticket.getNoOfTickets()\*ticket.getTicketPrice()));

            System.out.println("Waiting for amount to be paid!! Hit Any key To Gen Ticket");

            scanner.nextLine();

            System.out.println(ticket);

        }

        catch (TicketAlreadyBookedException e) {

            System.err.println(e.getMessage());

            System.out.println("Please Try Again");

        }

        catch (InvalidTicketRangeException e) {

            System.out.println(e.getMessage());

            System.out.println("Please Choose Valid Tickets");

        }

        catch (Exception e){

            System.out.println(e.getMessage());

        }

    }

    public static void main(String[] args) {

        System.out.println("Welcome to the Movie Booking System!");

        while (true){

            System.out.println("\nWhat Do You want to do?");

            System.out.println("1. Show Available Seats as well as Show Date and Time");

            System.out.println("2. Book Tickets");

            System.out.println("3. Exit");

            System.out.print("\nPlease enter your choice (1 or 2 or any key to exit) :- ");

            int choice=scanner.nextInt();scanner.nextLine();

            if(choice!=1 && choice!=2) {

                scanner.close();

                System.exit(0);

                break;

            }

            switch(choice){

                case 1:

                    showAvailableSeats();

                    break;

                case 2:

                    bookSeats();

                    break;

            }

        }

        // Close the scanner

        scanner.close();

    }

}

package frontdesk.moviebooking.exceptions;

public class TicketAlreadyBookedException extends Exception {

    public TicketAlreadyBookedException(String message){

        super(message);

    }

}

package frontdesk.moviebooking;

import java.util.\*;

class BookingData {

    private Map<String, Map<String, List<String>>> bookedSeatsByDate;

    public BookingData() {

        bookedSeatsByDate = new HashMap<>();

    }

    public void bookSeat(String date, String showTime, String seat) {

        bookedSeatsByDate

                .computeIfAbsent(date, k -> new HashMap<>())

                .computeIfAbsent(showTime, k -> new ArrayList<>())

                .add(seat);

    }

    public boolean isSeatBooked(String date, String showTime, String seat) {

        return bookedSeatsByDate

                .getOrDefault(date, Collections.emptyMap())

                .getOrDefault(showTime, Collections.emptyList())

                .contains(seat);

    }

}

package frontdesk.moviebooking;

import frontdesk.moviebooking.exceptions.InvalidTicketRangeException;

import frontdesk.moviebooking.exceptions.TicketAlreadyBookedException;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class MovieBookingSystem {

    private final BookingData bookingData;

    private final Theatre theatre;

    public MovieBookingSystem() {

        bookingData = new BookingData();

        theatre = new Theatre();

    }

    public void displayShowTimes() {

        System.out.print("Available times: ");

        String showTimes = String.join("    ", theatre.showTimes);

        System.out.println(showTimes);

    }

    public void displaySeatingArrangement(String selectedDate, String selectedShowTime) {

        System.out.println("Seating Arrangement for " + selectedDate + " at " + selectedShowTime + ":");

        for (int i = 0; i < theatre.seats.length; i++) {

            for (int j = 0; j < theatre.seats[i].length; j++) {

                String seat = theatre.seats[i][j];

                boolean isBooked = bookingData.isSeatBooked(selectedDate, selectedShowTime, seat);

                if(isBooked) {

                    System.out.print("\u001B[31m" + seat + " " + "\u001B[0m");

                }

                else{

                    System.out.print(seat+" ");

                }

            }

            System.out.println();

        }

    }

    public List<String> getSeats(String seatsInput) throws  InvalidTicketRangeException {

        List<String> allSeats=new ArrayList<>();

        String[] seatRanges = seatsInput.split(",");

        for (String seatRange : seatRanges) {

            String[] seats = seatRange.split("-");

            if (seats.length == 1) {

                allSeats.add(seats[0]);

            } else if (seats.length == 2) {

                String startSeat=seats[0];

                String endSeat=seats[1];

                char startRow = startSeat.charAt(0);

                char endRow = endSeat.charAt(0);

                if(startRow!=endRow){

                    throw new InvalidTicketRangeException("Invalid Ticket Range "+seatRange);

                }

                int startCol = Integer.parseInt(startSeat.substring(1));

                int endCol = Integer.parseInt(endSeat.substring(1));

                if (startCol>4 || endCol<=startCol){

                    throw new InvalidTicketRangeException("Invalid Ticket Range "+seatRange);

                }

                for(int i=startCol;i<=endCol;i++){

                    allSeats.add(String.valueOf(startRow)+i);

                }

            }

        }

        return allSeats;

    }

    private List<String> checkAvailability(String selectedDate, String selectedShowTime, String seatInp) throws TicketAlreadyBookedException, InvalidTicketRangeException {

        List<String> selectedSeats = getSeats(seatInp);

        for(String seat : selectedSeats){

            if(bookingData.isSeatBooked(selectedDate,selectedShowTime,seat)){

                throw new TicketAlreadyBookedException("Ticket :"+seat+" is already Booked");

            }

        }

        return selectedSeats;

    }

    public List<String> bookSeats(String selectedDate, String selectedShowTime, String seatInp) throws TicketAlreadyBookedException, InvalidTicketRangeException {

        List<String> selectedSeats = checkAvailability(selectedDate,selectedShowTime,seatInp);

        for(String seat:selectedSeats){

            bookingData.bookSeat(selectedDate,selectedShowTime,seat);

        }

        return selectedSeats;

    }

    public List<String> getShowTimes(){

        return Arrays.stream(theatre.showTimes).collect(Collectors.toList());

    }

}

package frontdesk.moviebooking;

import java.util.\*;

class BookingData {

    private Map<String, Map<String, List<String>>> bookedSeatsByDate;

    public BookingData() {

        bookedSeatsByDate = new HashMap<>();

    }

    public void bookSeat(String date, String showTime, String seat) {

        bookedSeatsByDate

                .computeIfAbsent(date, k -> new HashMap<>())

                .computeIfAbsent(showTime, k -> new ArrayList<>())

                .add(seat);

    }

    public boolean isSeatBooked(String date, String showTime, String seat) {

        return bookedSeatsByDate

                .getOrDefault(date, Collections.emptyMap())

                .getOrDefault(showTime, Collections.emptyList())

                .contains(seat);

    }

}