```
import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

import java.util.*;

import java.io.PrintWriter;

import java.io.FileWriter;


/*Functional Requirements


1. Movie data loading: The system should read movie details(movie code, show time, available seats,
total seats, ticket prices, language, genre) from a CSV file.

2. Movie Selection: Users should be able to select a movie by inputting the relevant movie code.

3. Date and Showtime Selection: Users should be able to select a date and showtime by inputting the
data and showtime that user wish to watch the movie.

4. Ticket Booking: Users should be able to input the number of tickets they wish to book.

5. Error handling: The system must give an error invalid movie codes, incorrect showtimes and invalid
number of tickets

5. Total Price Calculation: The system should calculate the total price of the tickets.

6. Email Notification: Generate the PDF including all the data and automatically email the PDF invoice to
the user.

7. CSV File as Database: system must properly parse the CSV and store movie data in memory for rapid
lookup and validation. Each column contains moviecode, moviename, date, showtime, totalsseats,
availableseats, ticketprices, language, genre.

8. Payment Integration: Simulate a payment process before completing the booking.


Non-Functional Requirements


1. Performance: system must be able to process moderate number of movies or a booking request in less
than 2 seconds under average load. Response time and search time must be quick.

2. Security: Sensitive data like personal or payment details(e.g., user email) should be encrypted before it
is stored or transmitted.
```

3. Scalability: The system should be design to handle larger numbers of movies, showtimes, and bookings records without reducing performance and without significant slowdowns.

4. Usability: The system should have a minimal, easy-to-use command-line user friendly interface for easy interaction. Input prompts should be clear and error messages should be understandable.

5. Reliability: The system must be reliable, if there are no errors then it works correctly. When generate any exception, should handle without crashing the system. for a example, If an email fails to send, the user should be notified and allowed to re-enter their email.

6. Maintainability: The code should be modula and easy to maintain the system. Support to add different movie details. The code has to be OOP compliant to improve readabality and reusability.

7.Portability: The below system should be able to run in different interfaces(Windows, MacOS, Linux) without significant change in the code.*/

```
class InvalidMovieCodeException extends Exception {
    public InvalidMovieCodeException(String message) {
        super(message);
    }
}


class InvalidDateException extends Exception {
    public InvalidDateException(String message) {
        super(message);
    }
}


class InvalidShowtimeException extends Exception {
    public InvalidShowtimeException(String message) {
        super(message);
    }
}


class InsufficientSeatsException extends Exception {
    public InsufficientSeatsException(String message) {
        super(message);
    }
}
public class MovieTicketReservationGroup_SYSNTAX_error{
    private List<Movie> movies = new ArrayList<>();

    public void loadMoviesFromCSV(String filePath) {
        try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
            String line;
            br.readLine(); // Skip the header row

            while ((line = br.readLine()) != null) {
                String[] data = line.split(",");
                if (data.length < 9) continue;

                Movie movie = new Movie(
                    data[0], data[1], data[2], data[3],
                    Integer.parseInt(data[4]), Integer.parseInt(data[5]),
                    Double.parseDouble(data[6]), data[7], data[8]
                );
                movies.add(movie);
            }
        } catch (IOException e) {
            System.err.println("Error reading CSV file: " + e.getMessage());
        }
    }

    public Set<String> getAvailableMovies() {
        Set<String> movies1 = new HashSet<>();
        for (Movie movie : movies) {
            movies1.add(movie.movieCode +"-" + movie.movieName);
        }
        return movies1;
    }


            public Set<String> getAvailableMovieCodes() throws InvalidMovieCodeException {
                                Set<String> movieCodes=new HashSet<>();
                                for(Movie movie: movies){
                                        movieCodes.add(movie.movieCode);
                                }
                                return movieCodes;
            }


    public Set<String> getAvailableShowtimes(String movieCode, String date) throws InvalidDateException
{
        Set<String> availableShowtimes = new HashSet<>();
        for (Movie movie : movies) {
            if (movie.movieCode.equalsIgnoreCase(movieCode) && movie.date.equals(date)) {
                availableShowtimes.add(movie.showtime);
            }
        }
        if (availableShowtimes.isEmpty()) {
            throw new InvalidDateException("No shows available on this date.");
        }
        return availableShowtimes;
    }


    public Set<String> getAvailableDates(String movieCode) throws InvalidMovieCodeException {
        Set<String> availableDates = new HashSet<>();
        for (Movie movie : movies) {
            if (movie.movieCode.equalsIgnoreCase(movieCode)) {
                availableDates.add(movie.date);
            }
        }
        if (availableDates.isEmpty()) {
            throw new InvalidMovieCodeException("Movie code not found.");
        }
        return availableDates;
    }

    public int getAvailableSeats(String movieCode, String date, String showtime) {
        for (Movie movie : movies) {
            if (movie.movieCode.equalsIgnoreCase(movieCode) &&
                movie.date.equals(date) &&
                movie.showtime.equalsIgnoreCase(showtime)) {
                return movie.availableSeats;
            }
        }
        return -1; // Return -1 if no matching movie found
    }

    public double bookTickets(String movieCode, String date, String showtime, int requestedSeats) throws
InsufficientSeatsException {
        for (Movie movie : movies) {
            if (movie.movieCode.equalsIgnoreCase(movieCode) &&
                movie.date.equals(date) &&
                movie.showtime.equalsIgnoreCase(showtime)) {
```

```
            if (movie.isAvailable(requestedSeats)) {
                movie.bookTickets(requestedSeats);
                return movie.ticketPrice * requestedSeats;
            } else {
                throw new InsufficientSeatsException("Not enough seats available.");
            }
        }
    }
    return -1; // Movie not found
}

public static void main(String[] args) {
    MovieReservationSystem system = new MovieReservationSystem();
    system.loadMoviesFromCSV("Movie Reservation Dataset.csv");

    Scanner scanner = new Scanner(System.in);

    try {

        Set<String> availableMovies = system.getAvailableMovies();
        System.out.println("Available Movies : ");
        for (String name: availableMovies ){
                                System.out.println(name);
                            }

                            Set<String> availableMovieCodes=system.getAvailableMovieCodes();

        String movieCode = null;
```

```
while (movieCode == null) {
    System.out.print("Enter movie code: ");
    String input = scanner.nextLine();
    if (availableMovieCodes.contains(input)) {
        movieCode = input;
    } else {
        System.out.println("Invalid Movie Code. Please choose from available codes.");
    }
}

// Get valid date
Set<String> availableDates = system.getAvailableDates(movieCode);
System.out.println("Available Dates: " + availableDates);

String date = null;
while (date == null) {
    System.out.print("Enter Date (YYYY-MM-DD): ");
    String input = scanner.nextLine();
    if (availableDates.contains(input)) {
        date = input;
    } else {
        System.out.println("Invalid Date. Please choose from available dates.");
    }
}

// Get valid showtime
Set<String> availableShowtimes = system.getAvailableShowtimes(movieCode, date);
System.out.println("Available Showtimes: " + availableShowtimes);
```

```
String showtime = null;
while (showtime == null) {
   System.out.print("Enter Showtime (Morning, Afternoon, Evening): ");
   String input = scanner.nextLine();
   if (availableShowtimes.contains(input)) {
      showtime = input;
   } else {
      System.out.println("Invalid Showtime. Please choose from available showtimes.");
   }
}

// Get valid number of seats
int availableSeats = system.getAvailableSeats(movieCode, date, showtime);
System.out.println("Available Seats: " + availableSeats);

int requestedSeats = 0;

while (requestedSeats <= 0 || requestedSeats > availableSeats) {
   System.out.print("Enter number of seats: ");
   requestedSeats = scanner.nextInt();
   if(requestedSeats<0){
                                             System.out.println("Please Enter Valid number.");
                                  }
   if (requestedSeats > availableSeats) {
      System.out.println("Not enough seats available. Please choose a smaller number.");
   }
}

// Book tickets and show total price
```

```
            double totalPrice = system.bookTickets(movieCode, date, showtime, requestedSeats);
            System.out.println("Booking successful! " + requestedSeats + " tickets booked for " + movieCode +
                    " on " + date + " at " + showtime + ". Total Price: Rs" + totalPrice +"/=");
            System.out.print("Enter your name : ");
            String customer_name = scanner.nextLine();
            System.out.print("Enter your email :");
            String email = scanner.nextLine();
                                String filename = customer_name + ".pdf";
                                try (PrintWriter writer = new PrintWriter(new FileWriter(filename, true))) {
                                        writer.println(String.format("Name: %s", customer_name));
                                        writer.println(String.format("Email: %s", email));
                                        writer.println(String.format("Movie Code: %s", movieCode));
                                        writer.println(String.format("Date: %s", date));
                                        writer.println(String.format("Time: %s", showtime));
                                        writer.println(String.format("Tickets: %d", requestedSeats));
                                        writer.println(String.format("Price: %.2f", totalPrice ));
                                }catch(IOException e){
                                        System.out.println("Invalid file.");


                                }
                                //System.out.println("Your bill is Sent to your email " + email);
        } catch (InvalidMovieCodeException | InvalidDateException |InsufficientSeatsException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```