Movie Ticket Booking Simulator (CLI)

Phase-End Project Problem Statement

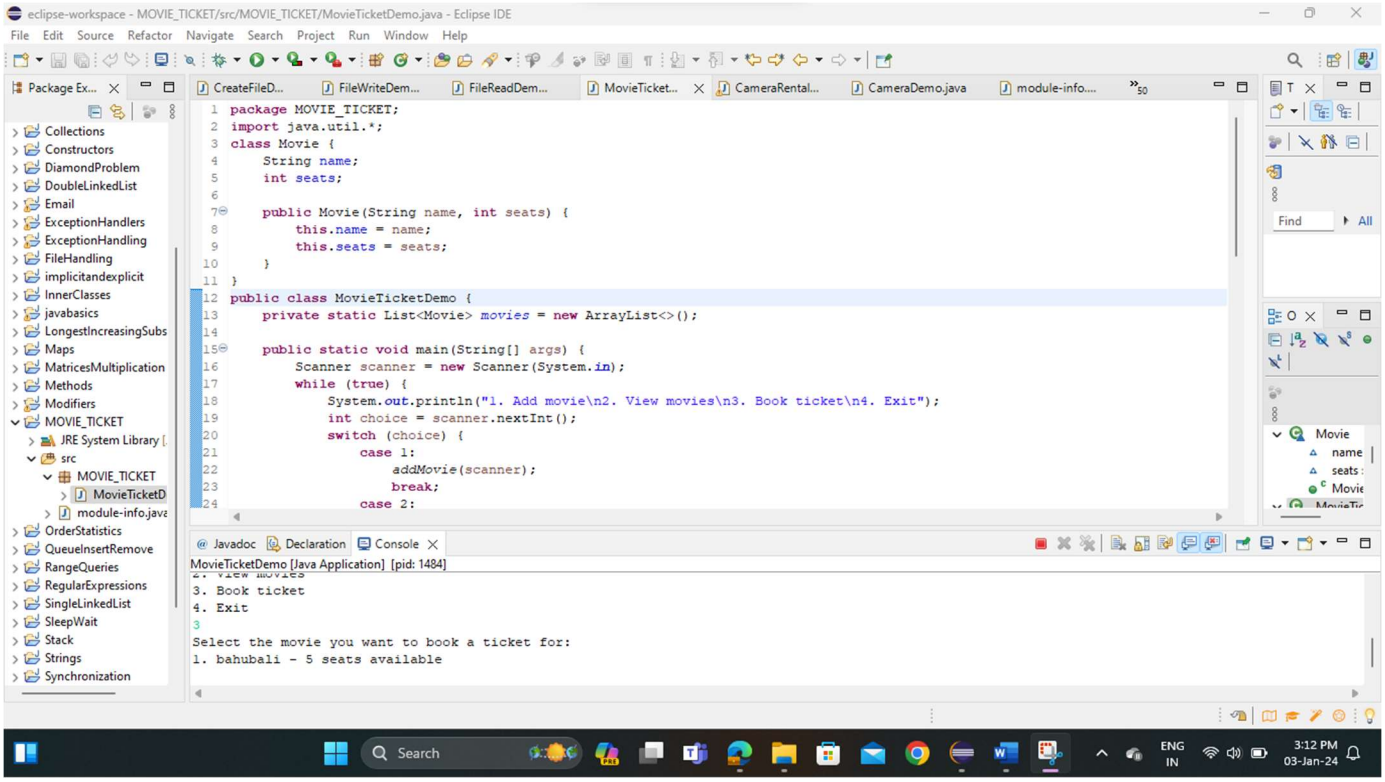
Project Agenda: Create a Movie Ticket Booking Simulator using command line interface, core java concepts, and data structures in Java.

Source Code:

```
package MOVIE_TICKET;
import java.util.*;
class Movie {
    String name;
    int seats;

    public Movie(String name, int seats) {
        this.name = name;
        this.seats = seats;
    }
}
public class MovieTicketDemo {
    private static List<Movie> movies = new ArrayList<>();

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("1. Add movie\n2. View movies\n3. Book
ticket\n4. Exit");
            int choice = scanner.nextInt();
            switch (choice) {
                case 1:
                    addMovie(scanner);
                    break;
                case 2:
                    viewMovies();
                    break;
                case 3:
                    bookTicket(scanner);
                    break;
                case 4:
                    System.exit(0);
                default:
                    System.out.println("Invalid choice");
            }
        }
    }

    private static void addMovie(Scanner scanner) {
        System.out.println("Enter movie name:");
        String name = scanner.next();
        System.out.println("Enter number of seats:");
        int seats = scanner.nextInt();
```

```
        movies.add(new Movie(name, seats));
    }

    private static void viewMovies() {
        for (int i = 0; i < movies.size(); i++) {
            System.out.println((i + 1) + ". " + movies.get(i).name + " - " +
movies.get(i).seats + " seats available");
        }
    }

    private static void bookTicket(Scanner scanner) {
        System.out.println("Select the movie you want to book a ticket
for:");
        viewMovies();
        int movieIndex = scanner.nextInt() - 1;
        if (movies.get(movieIndex).seats > 0) {
            movies.get(movieIndex).seats--;
            System.out.println("Ticket booked successfully!");
        } else {
            System.out.println("Sorry, no seats available for this movie.");
        }
    }
}
```

Output:



Algorithm:

1. Start
2. Display the login screen:
    1. Ask for username and password
    2. If credentials are correct, proceed to the main menu
    3. If not, display an error message and ask again
3. Display the main menu with options:
    1. Update Password
    2. Show Seating Arrangement
    3. Book Tickets
    4. Check Booking Status
    5. Exit
4. If user selects 'Update Password':
    1. Ask for the old password
    2. If the old password is correct, ask for the new password and update it
    3. If not, display an error message and go back to the main menu
5. If user selects 'Show Seating Arrangement':
    1. Ask for the date and show time
    2. Display the seating arrangement for the given date and time
    3. Go back to the main menu
6. If user selects 'Book Tickets':
    1. Ask for the booking date and show time
    2. Display the seating arrangement for the given date and time
    3. Ask for the preferred seat selection
    4. If the selected seats are available, display the total amount and ask for payment
    5. If the payment is successful, confirm the booking and update the seating arrangement
    6. If not, display an error message and go back to the main menu
7. If user selects 'Check Booking Status':

1. Ask for the booking ID
2. Display the booking status
3. Go back to the main menu

8. If user selects 'Exit', end the program
9. End

Description:

1)The Ticket Booking Simulator uses a HashMap to store the movie details and the number of tickets available for each movie. The HashMap is a part of Java's collections framework and it stores key-value pairs.

2)The bookTicket method is used to book tickets for a specific movie.

It checks if the movie is available and if there are enough tickets for the user's request.

3)The main method is the entry point of the application. It creates an instance of the MovieTicketBooking class and uses a Scanner to read the user's input from the command line.

4)At the output we need to enter the movie name and number of seats and view movies and select the movie we want to book a ticket for...

Introduction: The goal of this project is to create a command-line interface (CLI) application for a movie ticket booking system. The application will simulate the process of booking movie tickets at a theatre's front desk.

Requirements and Objectives: The application should allow the front desk to:

- Login using a username and password
- Update their password
- View the seating arrangement for a specific date and show time
- Book tickets by selecting seats
- View the auto-calculated amount for the booking
- Check or inquire about the booking status

Implementation Plan:

1. **Data Structures:** Use appropriate data structures like arrays or ArrayLists for storing movie details, seating arrangements, and bookings.

2. **Classes:** Create classes for Movie, Show, Seat, Booking, etc. to encapsulate related data and operations.
3. **User Interface:** Use the Scanner class for command line inputs and System.out.println for outputs.
4. **Authentication:** Implement a simple authentication system for the front desk login.
5. **Booking System:** Implement the logic for viewing seating arrangements, booking tickets, and checking booking status.

Testing and Deployment: Write unit tests for your methods and classes. Test the application thoroughly to ensure it works as expected. Once testing is complete, the application can be deployed.

Maintenance and Updates: Regularly update the application to add new features or fix bugs. Maintain the codebase by refactoring code when necessary and keeping the documentation up-to-date.

Conclusion: This project will help you understand the practical application of core Java concepts and data structures. It will also give you experience in developing a complete software solution from requirements gathering to deployment and maintenance.