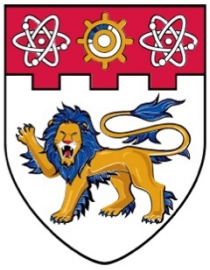


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

OODP Assignment Report

Written By:

Ng Mu Rong

Ong Jing Xuan U2122081B

Prasad Shubhangam Rajesh

Saw Thida Htut U2122487B

Somesh Sahu

1. Introduction

Our object-oriented app is a console-based application that models a movie booking and listing application. It provides a streamlined platform for movie-goers to make movie bookings and admins to manage listings of movies and sales reporting.

2. Class Diagram

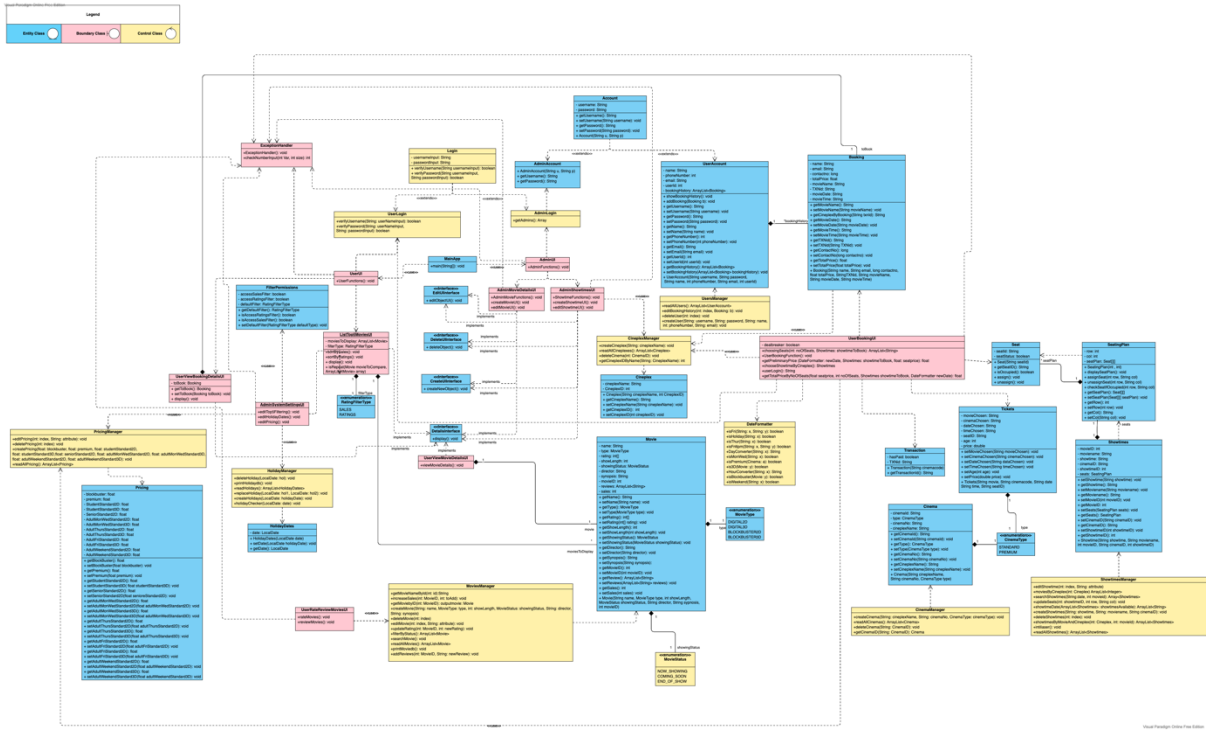


Figure 1: Class Diagram

3. Design Considerations

3.1 SOLID Design Principles

3.1.1 Single Responsibility Principle

The Single Responsibility Principle (SRP) states that a class should only have one responsibility and therefore should only have one reason to change. Based on this principle, we implemented the following in our MOBLIMA application.

1. Each manager classes in our app manages one database only. For instance, MoviesManager only manages the AllMovies.dat file and ShowtimesManager only manages the AllShowtimes.dat file.
2. Separated classes for accounts of different users. We have a UserAccount and AdminAccount class that extends the Account class.

3. Separated classes for accounts of different users. We have a UserLogin and AdminLogin class that extends the Login class.

3.1.2 Open-Closed Responsibility Principle

The Open-Closed Responsibility Principle (OCP) states that a class should be open for extension but closed for modification. In other words, we should be able to add new functionality without touching the existing code of the class. Currently, the UserAccount and AdminAccount extends Account class. This allows for easy extension of a GuestAccount class that allows users to book without signing in, while not risking modification of the other two classes.

3.1.3 Liskov Substitution Principle

The Liskov Substitution Principle (LSP) states that subclasses should be able to substitute their super classes. There should be no disruptions to the functionality of the program if any object of a subclass is passed to a method which expects an object of its super class. This was implemented in our code by ensuring that all overridden methods have the same behaviour as their respective super class methods, have the same return types and take in the same number of parameters as their super class methods.

For instance, Login class is the super class of the classes AdminLogin and UserLogin where the methods verifyUsername(String usernameInput) and verify(String usernameInput, String passwordInput) are overridden in the subclass UserLogin.

3.1.4 Interface Segregation Principle

The Interface Segregation Principle (ISP) keeps the interfaces separated such that classes are not required to implement functions they do not need. We implemented ISP in our OO application through the following:

1. DetailsInterface is implemented by 5 of the boundary classes, but the EditUIInterface, DeleteUIInterface, CreateUIInterface interfaces are only implemented by 2 of the 5 classes, as the other 3 do not require the relevant methods. Instead of having a general purpose UI Interface, we made specific interfaces to ensure that the classes are only required to implement the methods they need.

3.1.5 Dependency Injection Principle

The Dependency Injection Principle (DIP) states that high level modules should not depend on low level modules, instead, both should depend on abstractions. Abstractions should not depend on details, details should depend on abstractions instead. This is displayed in our application through:

1. High level classes such as Movie, Cinema and Seat that control the logic is independent of low-level classes such as MoviesManager, CinemaManager and SeatsManager. The low-level classes help the high-level classes perform their commands.

3.1.6 Extensibility & Reusability

Extensibility is the ability to extend a system with little effort. For instance, we created CinemaManager and CineplexManager to ensure easy extensibility for more cineplexes and cinemas under each cineplex.

Reusability is the ease in which portions of a class can be reused in the development of new classes. In our application, we reused multiple methods in the MovieManager class to get data. Examples of such methods are getMoviebyID() and readAllMovies(). Additionally, we UI functions in UserBookingUI for view booking history module to remove repetitive code.

3.2 OO Design Concepts

3.2.1 Abstraction

Abstraction is the process of filtering out the attributes of objects to display only the relevant attributes and hide the unnecessary details. Furthermore, control abstraction simplifies the program and removes unnecessary execution details. In our application, the UserAccount and AdminAccount classes extends the Account class. The reusability of the classes for its child classes, allows us to abstract them to reuse.

3.2.2 Encapsulation

Encapsulation is wrapping attributes and code acting on the methods together as a single unit, to control the visibility of the class. In our application, we applied this concept by making all attributes of the different objects private. These attributes can only be retrieved through

public getter methods or changed through public setter methods, enhancing the security of the app.

3.2.3 Inheritance

Inheritance is an important OO feature that allow derivation of new classes from existing classes by absorbing their attributes and behaviours and adding new capabilities in the new class. We applied this concept to build more efficient code through reusing old code. It is shown in the following examples:

1. We have the classes AdminAccount and UserAccount that extend the abstract class Account. This is because both classes share the same common attributes and methods.
2. Classes AdminLogin and UserLogin that extend Login
3. EditUIInterface, DeleteUIInterface, CreateUIInterface and DetailsInterface interfaces are implemented by the UI classes.

3.2.4 Polymorphism

Polymorphism is the ability of an object to take on many forms. Our group uses polymorphism through downcasting. Whenever we read an object from our .dat files, the object returned is of class Object. In order to use each Object in our functions, we downcast the Object read from the .dat file to the required class. For instance, the readAllCineplexes() function in CineplexManager reads the object from the AllCineplexes.dat file and downcasts the array list of Objects returned to an array list of Cineplexes.

4. Future Enhancements

The first feature for further enhancement involves the possibility of booking as a guest user without including any necessary details. For users that do not wish to provide their personal information to book, they can easily make use of our CreatePerson class. The create person class currently extends UserLogin and AdminLogin. However, by following SOLID principle, for the Open-Closed concept, we can create a guest login feature which does not involve the inclusion of necessary details.

Another feature that can add to our application is to add an enumeration function to increase our code reusability. As time goes by and the cineplex expands, we foresee an increase in types of movies showcased, ranging from Dolby Atmos to IMAX. Cinema types can potentially also change to have different tiers apart from premium and digital. By adding the

enumeration function, users can easily see all the movie types and cinema types offered to them once such movies, cinemas and showtimes can also be created with minimal addition of code. This is the result of using good coding features, enhancing code reusability.

5. Testing

5.1 Admin Test Cases

ID	Test Cases
Login Module	
1	Verify that admin can only log in and gain access to editing movies, showtimes and configure settings after their username and password are verified (UI for successful login does not work)
2	Verify that admin is prompted to enter username or password again if verification for either one fails
Create/Update/Remove Movie Listing	
3	Verify that admin can add new movie by entering movie name, movie type, show length, show status, director and synopsis
4	Verify that admin can delete any movie by selecting the movie they wish to delete
5	Verify that admin can edit any movie by selecting the movie they wish to edit and edit either movie name, movie type, show length, show status, director or synopsis
6	Verify that admin can edit the status of any movie to “End of Showing” and the specific movie will not be listed for bookings
7	Verify that admin can view current list of all the movies in the cinema
Create/Update/Remove Showtime or Movie to be shown	
8	Verify that admin can add new showtime by entering movie, date, showtime slot and cinema ID
9	Verify that admin can delete any showtime by selecting the showtime they wish to delete
10	Verify that admin can edit any showtime slot, by either selecting the movie or cinema/cineplex ID.
11	Verify that admin can view current list of all the showtimes of different movies in the cinema

Configure System Settings	
12	Verify that admin can add holiday dates by entering the date in format DD/MM/YYYY
13	Verify that admin can delete a holiday date by entering the date in format DD/MM/YYYY
14	Verify that admin can edit holiday date by entering the date in format DD/MM/YYYY
15	Verify that admin can view a list of all current holiday dates in format DD/MM/YYYY
16	Verify that admin is prompted with an error message when holiday date is entered in wrong format
17	Verify that admin can edit movie prices by selecting the movie type or age group they wish to edit and subsequently entering a new price.
18	Verify that admin can edit top 5 movies filters by editing user access
19	Verify that admin can view top 5 movies by ratings or ticket sales
20	Verify that overall ratings are only displayed if there are more than one rating made for it (display NA if don't have rating)

5.2 Movie-goer Test Cases

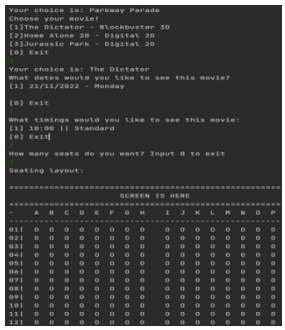
ID	Test Cases
Book a Movie	
1	Verify that movie-goer can book a movie by choosing the movie, date, showtime slot, seat, type of movie and age group
2	Verify that movie-goer can only book movies that have "Preview" or "Now Showing" status
3	Verify that pricing is accurate for a student booking a digital 2D movie after 6pm on Wednesday
4	Verify that pricing is accurate for a booking of digital 2D on a holiday
5	Verify that seating is not updated when movie-goer exits booking before transaction
Search/List Movie	

6	Verify that movie-goer can search for a movie by entering the movie name to obtain details such as movie name, movie type, show length, show status, director and synopsis
7	Verify that movie-goer is prompted to search or view list movies if search input is does not match any of the “Now Showing” and “Preview” movies
8	Verify that movie-goer can only view movies that are “Now Showing” and “Preview”
9	Movie-goer can choose a movie from the list of “Now Showing” and “Preview” movies to view details such as movie name, movie type, show length, show status, director and synopsis
Check Seat Availability	
10	Verify that movie-goer can choose the movie on a specific date and showtime to view seating
List Top 5 Movies	
11	Verify that movie-goer can choose to view top 5 movies by rating or ticket sales and 5 movies appear with movie name and ratings
12	Verify that movie-goer can choose to view top 5 movies by rating or ticket sales and the total number of movies will appear if there are less than 5 existing movies
13	Verify that the top 5 movies are updated when a rating is changed or a ticket has been sold
View Booking History	
14	Verify that movie-goer can view booking history details (transaction ID, location, movie name, movie date, show length) after signing in/logging in and choosing the specific booking date to view
15	Verify that movie-goer is prompted to enter username or password again if verification for either one fails
16	Verify that movie-goer who do not have any existing booking history is prompted an error message
Rate/Review Movies	
17	Verify that movie-goer can choose movie to rate or review and the rating/review is updated
18	Verify that movie-goer can only rate/review movies that have “Preview” or “Now Showing” status

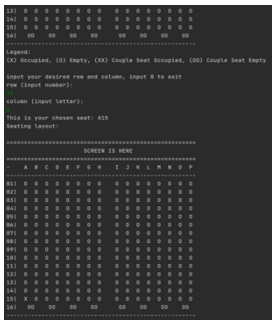
5.3 Test Cases with Results Recorded

Test Case 1: Adult 3D Digital Standard on Weekday (blockbuster type) - Price shown to be $12

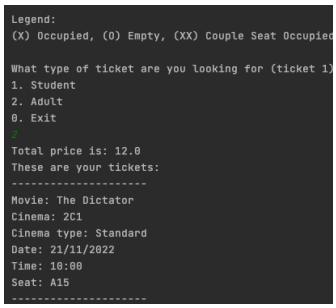
Choosing Cineplex



Selecting 3D blockbuster movie on a weekday

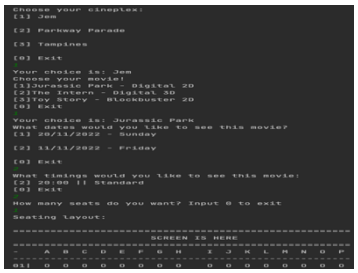


Seats chosen

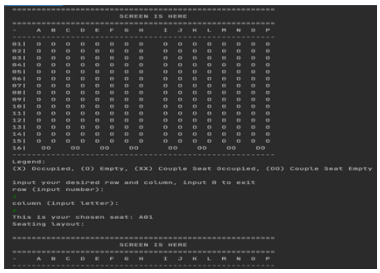


Price shown is $12 with $1 from blockbuster nature

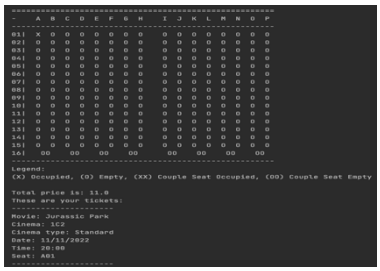
Test Case 2: Student Digital Standard on Friday after 6pm – Option for student not available, automatically charged $11



Choosing Cineplex



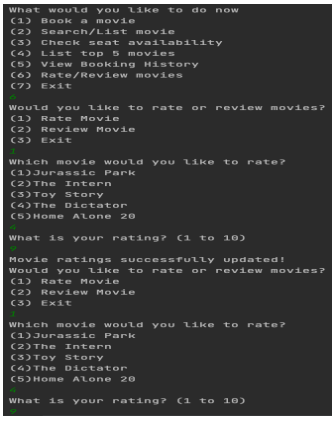
Selecting 3D blockbuster movie on a weekday



Seats chosen

Test Case 3 and 4: Ratings will not sure if there is one or less review available – Dictator Movie went up from 0 to 2 allowing it to be reflected

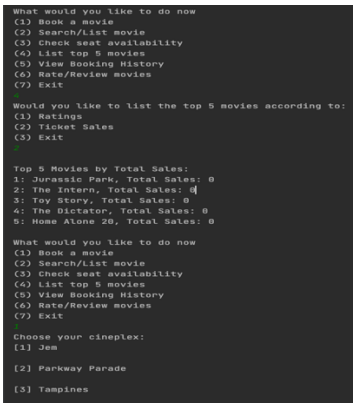
Choosing to rate movie, currently only 4 movies seen



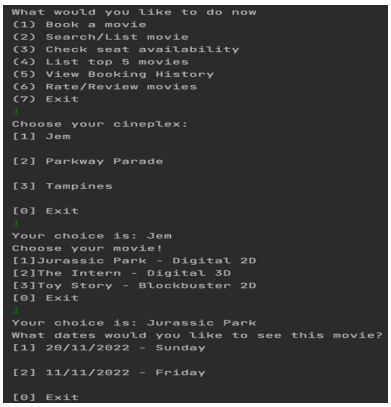
Add two ratings to movie with no ratings

New rating can now be clearly seen

Test Case 5: Verifying that the top 5 movies change after the purchase of a ticket

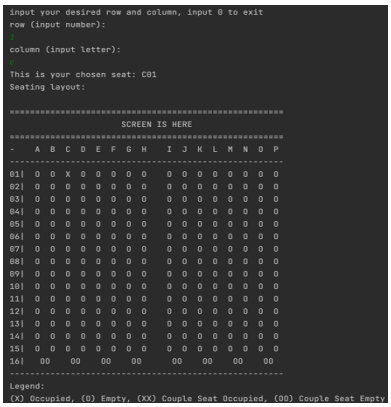


Currently shows all 5 with no sales in rankings

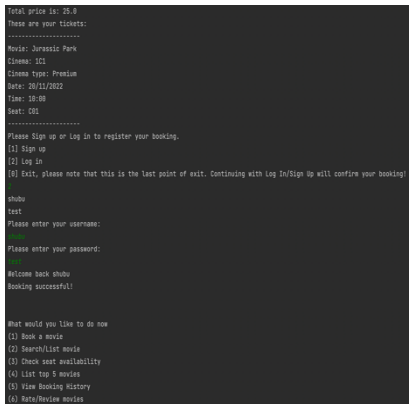


Process of chosing the correct cinemplex

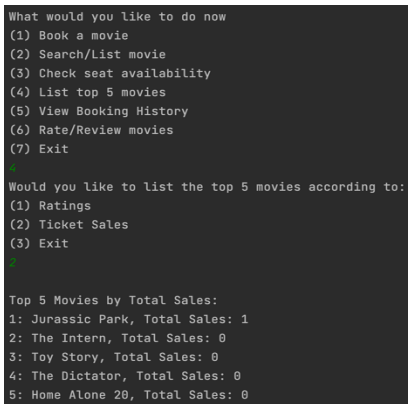
Choosing movie timings



Choosing seats



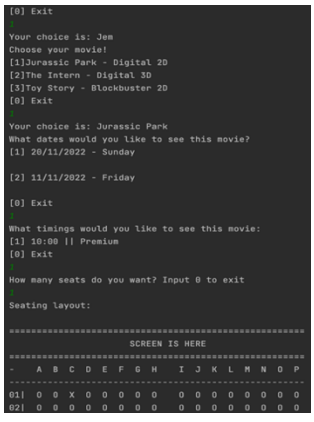
Ticket displayed with prompt to login



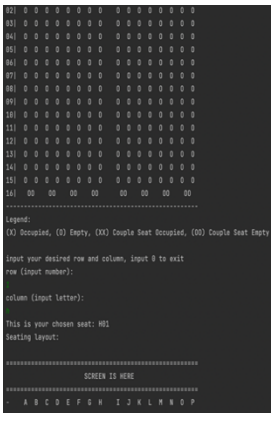
Jurassic Park updated as top most movie with highest sales

Test Case 6: Ticket price for a premium cinema on a weekday

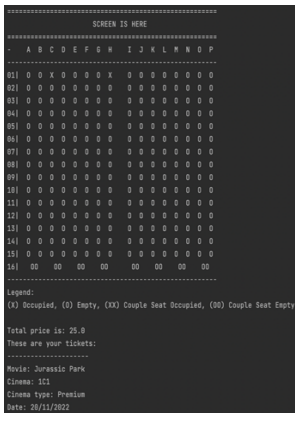
Process of booking a movie



Selecting a premium cinema timing



Selecting the specific seat



Price shows $25 indicating premium price with no student prompt

Declaration of Original Work for SC/CE/CZ2002 Assignment

We hereby declare that the attached group assignment has been researched, undertaken, completed and submitted as a collective effort by the group members listed below.

We have honoured the principles of academic integrity and have upheld Student Code of Academic Conduct in the completion of this work.

We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work. In addition, disciplinary actions may be taken.

Name	Course (CE2002 or CZ2002)	Lab Group	Signature/Date
Ng Mu Rong	SC2002	REP	NMR/ 13 Nov 2022
Ong Jing Xuan	SC2002	REP	OJX/ 13 Nov 2022
Prasad Shubhangam Rajesh	SC2002	REP	PSR/ 13 Nov 2022
Saw Thida Htut	SC2002	REP	STH/ 13 Nov 2022
Somesh Sahu	SC2002	REP	SS/ 13 Nov 2022

Important notes: 1. Name must **EXACTLY MATCH** the one printed on your Matriculation Card.xs