



## **22S1 SC2002 Object Oriented Design and Programming MOBLIMA Report**

### **SS3 Assignment Group 5**

#### **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.

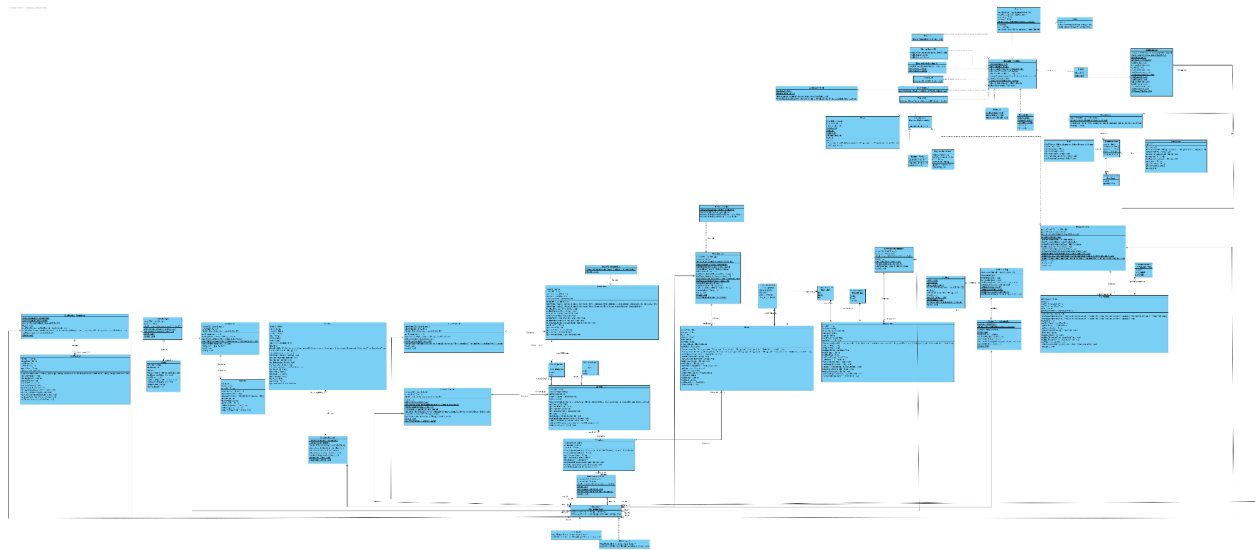
We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work. In addition, disciplinary actions may be taken.

Name	Course	Lab Group	Signature/ Date
Lee Kar Jun Lester	SC2002	SS3	Lester / 11-11-2022
Lee Yu Xuan	SC2002	SS3	Yu Xuan / 11-11-2022
Lim Boon Hian	SC2002	SS3	Boon Hian / 11-11-2022
Lim Yao Xian	SC2002	SS3	Yao Xian / 11-11-2022
Loke Yong Jian	SC2002	SS3	Yong Jian / 11-11-2022

## **Table of Content:**

UML Class Diagram .....	3
Approach Taken .....	4
Design Consideration .....	6
Moblina Functions .....	8
Testings .....	x
Two New Features .....	y

## UML Class Diagram:



A clear diagram is attached in the zip file.

## **Approach Taken:**

During the planning of the MOBLIMA application, we wanted the system to be user friendly and simple to upgrade. As our application aims to be flexible and with intuitive inputs. Unlike the existing real life movie booking system which already has an existing database of customers and movies, we had to allow admin users to create movies, movie time slots as well as user creation for the system.

The system also allows registered movie-goers to see the various movie showtimes and see what movies are currently available. If the user has a specific movie in mind, they can search for it. If not, they can view the rankings of the movie ranked by either ticket sales or movie-goer ratings. When the users want to book a movie ticket for a certain movie at a certain time slot, they will be brought to the cineplex layout. From the cineplex layout, users will be able to view the seating plan and see which seats are vacant. Users will also receive a 2 dollar discount (except on public holidays and weekends) when they buy seats from the first 2 rows. Once they want to purchase the ticket, users will be prompted if they want to use a promo code. Users then can choose to pay via credit/debit card or through PayLah. Once payment has been confirmed, movie-goers can check booking history to confirm that the purchase has been made.

As for users who do not want to be registered with the application, they will still be able to access the same features as registered movie-goers. However, they would need to key in their email address, so that the confirmation of payment can be sent to their email instead.

Admin users of the system have access to more features. Admin users are able to add, remove and update movies in the cinema's movie database. They should include the movie title, director, cast, rating (PG13, NC16 etc) as well as the synopsis into the system. They are able to also view the top 5 movies, either in terms of ticket sales or movie-goer rating. Admin users can also add or remove showtimes to specific movies at specific cineplexes. Additionally, they are able to add in promo-codes. They are able to set the name of the promo codes, the start/end date, quantity of the promo code as well as the discount offered. Admin users are also allowed to edit the system configuration. Firstly, they are able to input which dates of the year are public holidays or special events. This is important as it will affect the pricing of the ticket (Public Holidays dates will not provide any discount). They are also able to view a list of holidays as well as remove any if they wanted to. Secondly, they are able to modify the base price of the tickets as well as the different price modifiers for the different cinema classes. They can also view the current multipliers for each class of cinemas. Lastly, Admin users are able to add in more cinemas into a specific cineplex or add more cineplexes. For the cinemas, they are able to input which branch it belongs to, cinemaID, class

of cinema, number of seats and which layout type they want. As for the cineplex, admin users can key in the location of where the cineplex is located.

The MOBLIMA application aims to provide a pleasant user experience to both admin staff and movie goers. Should any features need to be updated or implemented, MOBLIMA will be able to do so seamlessly.

## **Design Consideration:**

Design principles have been considered in every possible aspect in the project. Developing the project with the SOLID principles in mind help us to achieve loose coupling and high cohesion, increasing the reusability and extensibility of the whole project.

### **Single Responsibility Principle:**

SRP states that a single class should only have a single responsibility and a single reason to change. In our project, for the case of 'Movie' class, instead of having all the functions (adding of movie, updating of movie etc) inside the 'Movie.java' file, we create separate controllers to handle those functions. Hence classes that link to 'Movie.java' file does not have to be edited and recompiled when changes are made to the 'Movie.java'

### **Open-Closed Principle:**

OCP advocates that a module should be closed for modification but open for extension. In our project, we have various Controller Classes responsible for different functions (i.e addMovie(), removeMovie()). But what if in the future there's a class that requires more interactions with the Movie Class but we do not want to alter existing codes as it might affect current users of the controller (i.e Staff Class uses Movie Controller to add movies)? We can use the Open-Closed Principle and let the new Controller Class inherit the functions of the current Movie Controller Class and give it new functions such as selecting some movies to be featured. By doing this, we not only maintain the functions of the Staff Class but we also introduced a new Controller Class that can select movies to be featured

### **Liskov Substitution Principle:**

LSP states that if arguments meant for a base class is instead passed to a subclass of the base class, it should be able to perform normally or even better. In our project we have a Staff Class. If we need to implement a new class such as a Supervisor Class, this can be easily done by inheriting from the Staff Class as the Supervisor Class should have the same features as the Staff Class or even more features.

**Interface Segregation Principle:**

ISP states that having multiple specific interfaces is better than having one general interface that does all the work. In our project, for our PaymentMethodInterface. We have two abstract methods: pay() and validation(). Hence we can implement various payment methods that do payment differently and validate the transaction differently.

**Dependency Injection Principle:**

DIP states that high level modules should not depend upon low level modules. Rather, both should depend upon abstractions. In our project, to access different file types, we have File\_IOInterface which all other file mutators and readers would inherit from. Such as JSONFile\_IO and CSVFile\_IO. Hence, allowing different file types input and output to be used based upon abstraction and not fixate on a particular file type allowing for dependency injection.

## **MOBLIMA Functions:**

- : Basic Features
- : Additional Features

### **Admin Module:**

- Login
  - Hashed Password (Demonstrated in video)
  - Automatically differentiate Staff from MovieGoer (Demonstrated in video)
- Create/Update/Remove movie listing (Demonstrated in video)
  - Create/Update/Remove movie details (Demonstrated in video)
  - Movie status is automatically updated (Pg. (YJ / BH))
- Create/Update/Remove cinema showtimes and the movies (Demonstrated in Video)
- Create/Update/Remove cinema (Demonstrated in Video)
  - Create/Update/Remove cineplex (Pg. xyz (YaoXian))
- Configure system settings (Demonstrated in Video)
  - Update Public Holiday dates (Demonstrated in Video)
  - Change Base Price of Tickets (Demonstrated in Video)
  - Change Price difference between cinema class (Demonstrated in Video)
  - Supports promotion code/ discount coupons with automatic updates (Pg. (YJ / BH))

### **Movie-goer Module:**

- Search/List movie (Demonstrated in video)
  - Search by movie title (Demonstrated in video)
- View movie details – including reviews and ratings (Demonstrated in video)
- Check seat availability and selection of seat/s (Demonstrated in video)
  - Seat Selection layout print (Demonstrated in video)
  - Prevention of duplicate bookings (Demonstrated in video)
- Book and purchase ticket (Demonstrated in video)
  - Supports multiple payment systems (Demonstrated in video)
  - Automatic price calculation based on multiple factors (Pg. xyz (Lester))
  - Price deduction by inputting promotion code (Demonstrated in video)
- View booking history (Demonstrated in video)
  - Individual booking history for every user (Pg. xyz (Y))
  - Able to categorised the booking history into ‘Upcoming’ or ‘Past’ based on current dateTime and the showTime stated on the movieTicket (Demonstrated in video)



- List the Top 5 ranking by ticket sales OR by overall ratings (Demonstrated in video)
  - Supports guest booking (Demonstrated in video)
  - Exception handling, error input will be prompted for correct input. (Pg. xyz (YJ/BH

## **How we handle the data from CSV**

With the OOP concept in mind, we decided to stay away from reading and editing the CSV directly, as it will defeat the purpose of this project - that is to practise the concept of OOP. Hence our choice of implementation in this project is by extracting the data from the CSV and using them to create instances for the classes that we have made. In this way, we are able to practise inheritance, abstraction, polymorphism etc.

To do this, we have 'loadAllClass' and 'saveAllClass' functions.

- 'loadAllClass' will be called when starting the application and is responsible for creating the instances from the data in respective CSV
- 'saveAllClass' will be called when exiting the application and is responsible for saving all the instances edited back to the respective CSV

(\*we have included 'class'.save() in some parts of the projects to reflect the real time changes for the purpose of the demo video)

```
public static void loadAllClass() {  
    MovieListing.load();  
    ReviewList.load();  
    ShowTimeList.load();  
    SeatBooked_Controller.load();  
    MovieTicketController.load();  
    SalesManager.load();  
    Holiday.load();  
    CinemaController.load();  
    SystemConfigController.load();  
    PromoCodeList.load();  
    CineplexController.load();  
}
```

```
public static void saveAllClass() {  
    MovieListing.save();  
    ReviewList.save();  
    ShowTimeList.save();  
    SeatBooked_Controller.save();  
    MovieTicketController.save();  
    SalesManager.save();  
    Holiday.save();  
    CinemaController.save();  
    SystemConfigController.save();  
    PromoCodeList.save();  
    CineplexController.save();  
}
```

## **Exception Handling**

To have a tidy way of dealing with exception handling, we have developed a java file called 'ExceptionHandling' which is responsible for handling all the data inputs. Whenever we want to get inputs from the user, we will do 'String input = ExceptionHandling.StringScanner()'.

Some of the examples are as follows:

```

public static int IntegerScannerRangeOfFunction(int upperBound) {
    Scanner scanner = new Scanner(System.in);
    int num = 0;
    try {
        num = scanner.nextInt();
    } catch (InputMismatchException e) {
        System.out.println(x: "You did not enter an integer.");
    }

    while (num <= 0 || num > upperBound) {
        System.out.println("Please enter a number between 1 and " + upperBound);
        num = ExceptionHandling.IntegerScannerRangeOfFunction(upperBound);
    }

    return num;
}

```

```

public static String StringScanner() {
    Scanner scanner = new Scanner(System.in);
    String str = "";
    try {
        str = scanner.nextLine();
        if(str.equals(anObject: "")){
            throw new InputMismatchException();
        }
    } catch (InputMismatchException e) {
        System.out.println(x: "You did not enter a string.");
    }

    return str;
}

```

```

public static LocalDateTime checkDateTime() {
    Scanner sc = new Scanner(System.in);
    String temp = "";
    LocalDateTime dateTime;
    try {
        temp = sc.nextLine();
        // convert string to datetime
        dateTime = LocalDateTime.parse(temp, _DateTimeFormatter.formatter);
    } catch (Exception e) {
        System.out.println(x: "Please enter according to the format 'yyyy-MM-dd HH:mm'!");
        dateTime = ExceptionHandling.checkDateTime();
    }
    return dateTime;
}

```

Apart from having exception handling for input such as integer, string and DateTime we have also done exception handling for each of our enumeration classes.

Some of the examples are:

```

public static Layout checkLayout() {
    Scanner scanner = new Scanner(System.in);
    Layout layout;
    try {
        layout = Layout.valueOf(scanner.nextLine());
    } catch (IllegalArgumentException e) {
        System.out.println(x: "Please enter one of the following: ");
        System.out.println(x: "small, medium, large");
        layout = ExceptionHandling.checkLayout();
    }
    return layout;
}

```

```
public static ShowingStatus checkShowingStatus() {  
    Scanner scanner = new Scanner(System.in);  
    ShowingStatus showingStatus = null;  
    try {  
        showingStatus = ShowingStatus.valueOf(scanner.nextLine());  
        // System.out.println("You entered: " + num);  
    } catch (IllegalArgumentException e) {  
        System.out.println(x: "Please enter one of the following: ");  
        System.out.println(x: "COMING_SOON, PREVIEW, NOW_SHOWING, END_OF_SHOWING");  
        showingStatus = ExceptionHandling.checkShowingStatus();  
    }  
    return showingStatus;  
}
```

## Promotion Code Auto Refresh

The promo code is automatically sets to available when the two condition is reached:

- PromoCodeStatus is not blocked
  - Blocked: The promo code will not be available until admin change its status to ready/available
- Current time past promo code start time

The invalid promo code is automatically deleted when the following condition is reached:

- Current time past promo code end time

The promo code status is updated when the program starts or when other classes try to get the info of the promocode by invoking functions(e.g. getAvailableCodeList, checkPromoCode).

The refresh function is shown below:

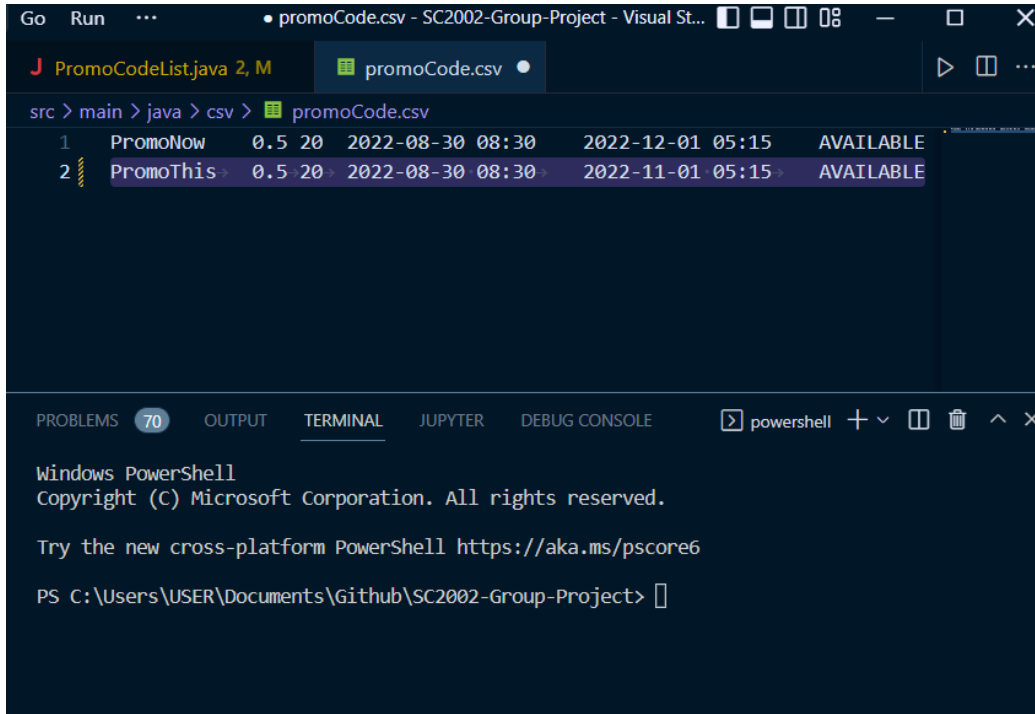
```
public static void refresh(){
    //get current time
    LocalDateTime now = LocalDateTime.now();

    for(int i = 0; i < promoCodeList.size();i++){
        //set to valid for incoming promocode
        if(now.isAfter(promoCodeList.get(i).getStartTime()) &&
            (promoCodeList.get(i).getPromoCodeStatus() != PromoCodeStatus.BLOCKED)){
            promoCodeList.get(i).setPromoCodeStatus(PromoCodeStatus.AVAILABLE);
        }
        //remove outdated promocode
        if(now.isAfter(promoCodeList.get(i).getEndTime())
            || promoCodeList.get(i).getCount() <= 0){
            promoCodeList.remove(i);
            i--;
        }
    }

    save();
}
```

## Screenshots:

Before the program starts:



The screenshot shows the Visual Studio Code editor with the file `promoCode.csv` open. The file contains two lines of CSV data:

	PromoNow	0.5	20	2022-08-30 08:30	2022-12-01 05:15	AVAILABLE
2	PromoThis	0.5	20	2022-08-30 08:30	2022-11-01 05:15	AVAILABLE

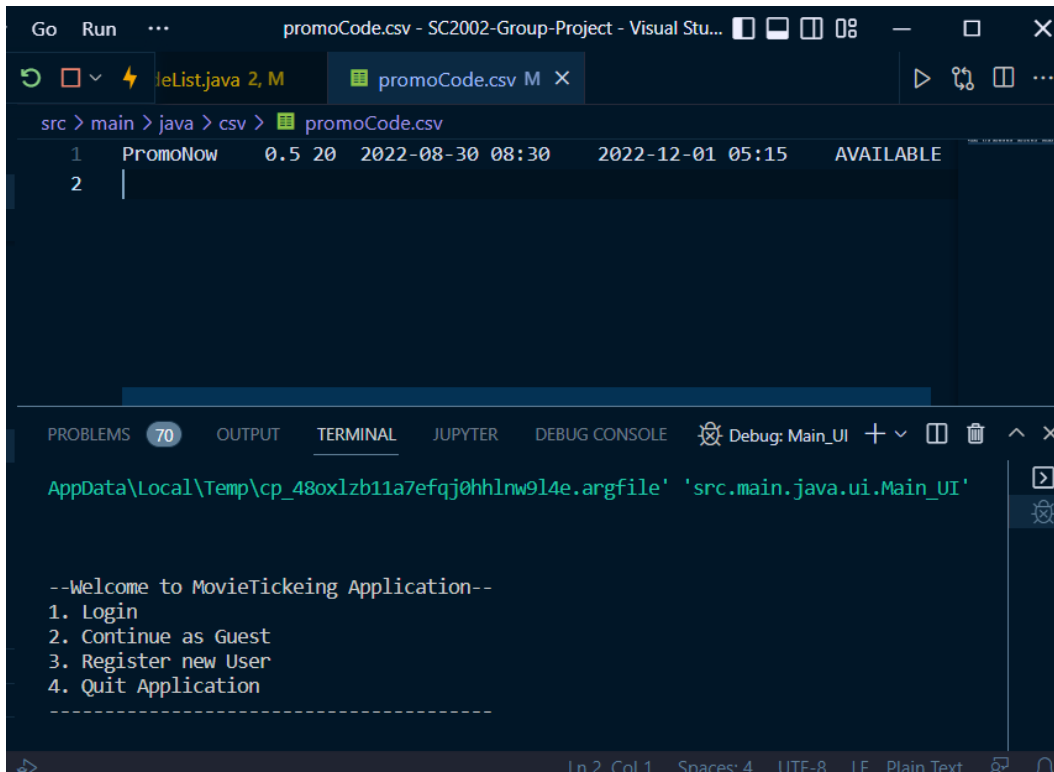
The bottom panel shows the Windows PowerShell terminal with the following text:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\USER\Documents\Github\SC2002-Group-Project>
```

After the program starts:



The screenshot shows the Visual Studio Code editor with the file `promoCode.csv` open. The file contains two lines of CSV data:

	PromoNow	0.5	20	2022-08-30 08:30	2022-12-01 05:15	AVAILABLE
2						

The bottom panel shows the Windows PowerShell terminal with the following text:

```
AppData\Local\Temp\cp_48ox1zb11a7efqj0hhlw914e.argfile' 'src.main.java.ui.Main_UI'

--Welcome to MovieTickeing Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
-----
```

## **Movie Status Auto Refresh**

The movie is automatically sets to NOW\_SHOWING through the following sequences:

- Check through the entire show time list
- If there exist an entries where current time is later than the **start** time of the showing,
- The movie status is automatically updated to NOW\_SHOWING

The movie is automatically sets to END\_OF\_SHOWING through the following sequences:

- Check through the entire show time list
- If there exist an entries where current time is later than the **end** time of the showing,
- The showtime is removed
- After removing the showtime, if there exist no showtime for the corresponding movie
- The movie status is automatically updated to END\_OF\_SHOWING

The movie status is updated when the program starts.



The refresh function is shown below:

```
public static void refresh(){
    //debug
    //LocalDateTime now = LocalDateTime.parse("2023-01-03 12:00",_DateTimeFormatter.formatter);

    LocalDateTime now = LocalDateTime.now();
    List<ShowTime> showTime = ShowTimeList.getShowTimeList();

    for(int i = 0;i<showTime.size();i++){

        int diff = now.compareTo(showTime.get(i).getStartTime());

        if(diff > 0){ // current Time already past start Time
            String currentMovieID = showTime.get(i).getMovieID();

            if(ShowTimeList.getShowTimeByID(currentMovieID).size() != 0){
                int index = MovieListing.getMovieIndexByID(currentMovieID);
                if(index != -1){
                    MovieListing.getMovieList().get(index).setStatus(ShowingStatus.NOW_SHOWING);
                }
            }
            //debug message
            //System.out.println("This show is over.");
        }
    }
}
```

```
for(int i = 0;i<showTime.size();i++){

    int diff = now.compareTo(showTime.get(i).getEndTime());

    if(diff > 0){ // current Time already past End Time
        String currentMovieID = showTime.get(i).getMovieID();
        showTime.remove(i);

        if(ShowTimeList.getShowTimeByID(currentMovieID).size() == 0){
            int index = MovieListing.getMovieIndexByID(currentMovieID);
            if(index != -1){
                MovieListing.getMovieList().get(index).setStatus(ShowingStatus.END_OF_SHOWING);
            }
        }
        i--;

        //debug message
        //System.out.println("This show is over.");
    }
}

MovieListing.save();
ShowTimeList.save();
```

## Screenshots:

Notice the following entries:

- movieList.csv - MovieTitle: Prince (Line 9) - ShowingStatus: COMING\_SOON
- showTime.csv - MovieID : S0001(Line 1)

The screenshot shows the Visual Studio Code interface with two CSV files open: showTime.csv and movieList.csv. The showTime.csv file contains a table with columns for MovieID, MovieTitle, ShowingStatus, and ShowingTime. The movieList.csv file contains a table with columns for MovieID, MovieTitle, ShowingStatus, and ShowingTime. The terminal window at the bottom shows the output of the application, including a welcome message and a list of options.

```
src > main > java > csv > showTime.csv
1 S0001 A3 2022-10-05 17:30 2022-12-05 19:15 Gold
2 S0001 A1 2022-12-25 08:30 2022-12-25 10:30 Silver
3 S0001 A2 2022-12-25 08:45 2022-12-25 10:45 Platinum
4 S0002 A2 2022-11-12 08:30 2022-11-13 10:30 Platinum
5 S0003 A2 2022-12-20 08:30 2022-12-20 10:30 Platinum
6 S0006 A3 2022-12-20 12:30 2022-12-20 02:30 Gold
7 S0007 A3 2022-12-20 09:30 2022-12-20 10:30 Silver
8 S0007 A3 2022-12-21 09:30 2022-12-21 12:30 Gold
9

src > main > java > csv > movieList.csv
1 Bee Movie COMING_SOON Fresh out 99 Simon Jerry PG
2 Talbis Iblis NOW_SHOWING To protect their family's name
3 Family Carnival: Puss In Boots: The Last Wish PREVIEW Th
4 Life Is Beautiful PREVIEW 'Se-yeon', who has devoted her
5 Black Adam NOW_SHOWING Nearly 5,000 years after he was be
6 Top Gun: Maverick NOW_SHOWING After more than thirty yea
7 ANFF22: Disney And Pixar's Turning Red COMING_SOON Disney
8 Minions 2: The Rise Of Gru NOW_SHOWING From the biggest a
9 Prince COMING_SOON Anbu a social science teacher in the t
10 Jurassic World Dominion NOW_SHOWING In the epic conclusion
11 Black Panther: Wakanda Forever COMING_SOON In Marvel Stud
12

--Welcome to MovieTickeing Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
Exception in thread "main" java.util.NoSuchElementException
PS C:\Users\USER\Documents\Github\SC2002-Group-Project>
```

As there exist entrie(s) where the current time is later than the **start** time of the showing, the movie status of Prince will be set to NOW\_SHOWING.

The screenshot shows the Visual Studio Code interface with two CSV files open: showTime.csv and movieList.csv. The showTime.csv file contains a table with columns for MovieID, MovieTitle, ShowingStatus, and ShowingTime. The movieList.csv file contains a table with columns for MovieID, MovieTitle, ShowingStatus, and ShowingTime. The terminal window at the bottom shows the output of the application, including a welcome message and a list of options.

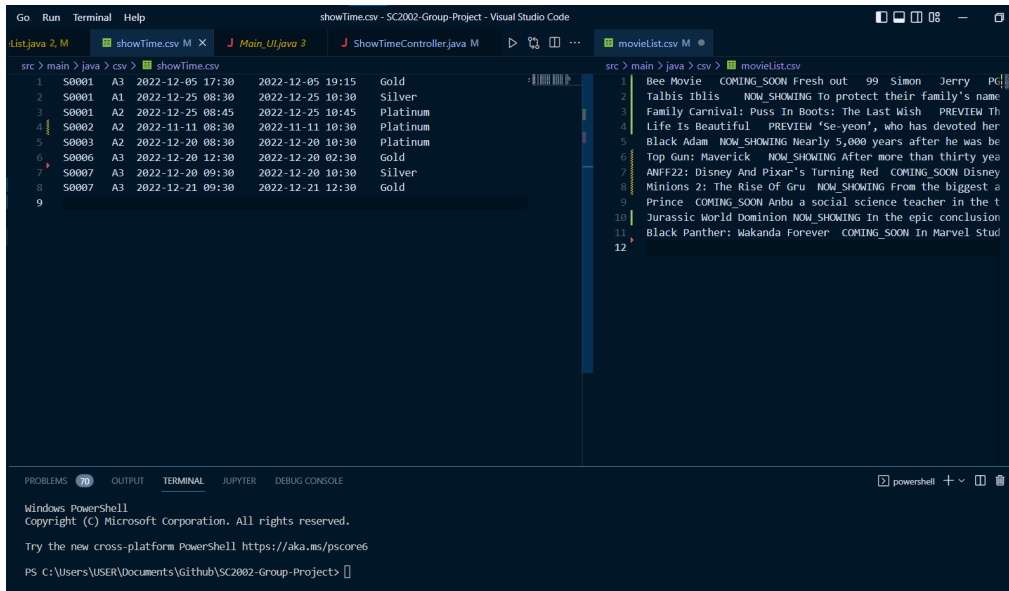
```
src > main > java > csv > showTime.csv
1 S0001 A3 2022-10-05 17:30 2022-12-05 19:15 Gold
2 S0001 A1 2022-12-25 08:30 2022-12-25 10:30 Silver
3 S0001 A2 2022-12-25 08:45 2022-12-25 10:45 Platinum
4 S0002 A2 2022-11-12 08:30 2022-11-13 10:30 Platinum
5 S0003 A2 2022-12-20 08:30 2022-12-20 10:30 Platinum
6 S0006 A3 2022-12-20 12:30 2022-12-20 02:30 Gold
7 S0007 A3 2022-12-20 09:30 2022-12-20 10:30 Silver
8 S0007 A3 2022-12-21 09:30 2022-12-21 12:30 Gold
9

src > main > java > csv > movieList.csv
1 Bee Movie COMING_SOON Fresh out 99 Simon Jerry PG
2 Talbis Iblis NOW_SHOWING To protect their family's name
3 Family Carnival: Puss In Boots: The Last Wish PREVIEW Th
4 Life Is Beautiful PREVIEW 'Se-yeon', who has devoted her
5 Black Adam NOW_SHOWING Nearly 5,000 years after he was be
6 Top Gun: Maverick NOW_SHOWING After more than thirty yea
7 ANFF22: Disney And Pixar's Turning Red COMING_SOON Disney
8 Minions 2: The Rise Of Gru NOW_SHOWING From the biggest a
9 Prince NOW_SHOWING Anbu a social science teacher in the t
10 Jurassic World Dominion NOW_SHOWING In the epic conclusion
11 Black Panther: Wakanda Forever COMING_SOON In Marvel Stud
12

--Welcome to MovieTickeing Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
PS C:\Users\USER\Documents\Github\SC2002-Group-Project>
```

Notice the following entries:

- movieList.csv - MovieTitle: Talbis Iblis (Line 2)
- showTime.csv - MovieID : S0002 (Line 4)



```
Go Run Terminal Help
showTime.csv - SC2002-Group-Project - Visual Studio Code

src > main > java > csv > showTime.csv
1 S0001 A3 2022-12-05 17:30 2022-12-05 19:15 Gold
2 S0001 A1 2022-12-25 08:30 2022-12-25 10:30 Silver
3 S0001 A2 2022-12-25 08:45 2022-12-25 10:45 Platinum
4 S0002 A2 2022-11-11 08:30 2022-11-11 10:30 Platinum
5 S0003 A2 2022-12-20 08:30 2022-12-20 10:30 Platinum
6 S0006 A3 2022-12-20 12:30 2022-12-20 02:30 Gold
7 S0007 A3 2022-12-20 09:30 2022-12-20 10:30 Silver
8 S0007 A3 2022-12-21 09:30 2022-12-21 12:30 Gold
9

movieList.csv M
src > main > java > csv > movieList.csv
1 Bee Movie COMING_SOON Fresh out 99 Simon Jerry PC
2 Talbis Iblis NOW_SHOWING To protect their family's name
3 Family Carnival: Puss In Boots: The Last Wish PREVIEW Th
4 Life Is Beautiful PREVIEW 'Se-yeon', who has devoted her
5 Black Adam NOW_SHOWING Nearly 5,000 years after he was be
6 Top Gun: Maverick NOW_SHOWING After more than thirty yea
7 ANFF22: Disney And Pixar's Turning Red COMING_SOON Disney
8 Minions 2: The Rise Of Gru NOW_SHOWING From the biggest a
9 Prince COMING_SOON Anbu a social science teacher in the t
10 Jurassic World Dominion NOW_SHOWING In the epic conclusion
11 Black Panther: Wakanda Forever COMING_SOON In Marvel Stud
12

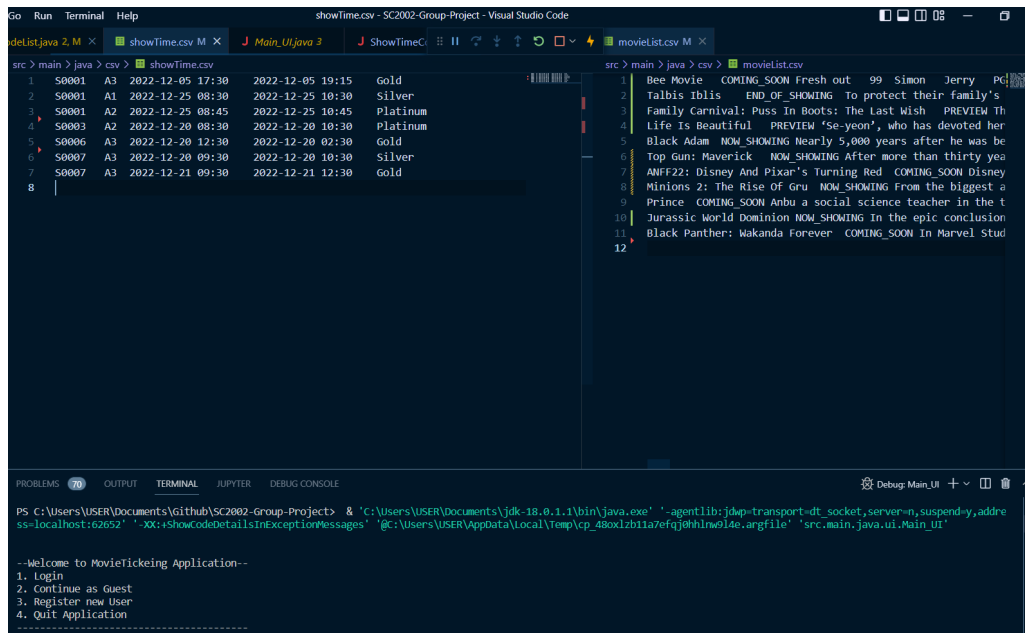
PROBLEMS 70 OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\USER\Documents\Github\SC2002-Group-Project>
```

As there exists an entry where the current time is later than the **end** time of the showing, the entry (Line 4 in showTime.csv) is removed.

As there exist no entries for after deleting the showtime, the movie status of Talbis Iblis will be set to **END\_OF\_SHOWING**.



```
Go Run Terminal Help
showTime.csv - SC2002-Group-Project - Visual Studio Code

src > main > java > csv > showTime.csv
1 S0001 A3 2022-12-05 17:30 2022-12-05 19:15 Gold
2 S0001 A1 2022-12-25 08:30 2022-12-25 10:30 Silver
3 S0001 A2 2022-12-25 08:45 2022-12-25 10:45 Platinum
4 S0003 A2 2022-12-20 08:30 2022-12-20 10:30 Platinum
5 S0006 A3 2022-12-20 12:30 2022-12-20 02:30 Gold
6 S0007 A3 2022-12-20 09:30 2022-12-20 10:30 Silver
7 S0007 A3 2022-12-21 09:30 2022-12-21 12:30 Gold
8

movieList.csv M
src > main > java > csv > movieList.csv
1 Bee Movie COMING_SOON Fresh out 99 Simon Jerry PC
2 Talbis Iblis END_OF_SHOWING To protect their family's name
3 Family Carnival: Puss In Boots: The Last Wish PREVIEW Th
4 Life Is Beautiful PREVIEW 'Se-yeon', who has devoted her
5 Black Adam NOW_SHOWING Nearly 5,000 years after he was be
6 Top Gun: Maverick NOW_SHOWING After more than thirty yea
7 ANFF22: Disney And Pixar's Turning Red COMING_SOON Disney
8 Minions 2: The Rise Of Gru NOW_SHOWING From the biggest a
9 Prince COMING_SOON Anbu a social science teacher in the t
10 Jurassic World Dominion NOW_SHOWING In the epic conclusion
11 Black Panther: Wakanda Forever COMING_SOON In Marvel Stud
12

PROBLEMS 70 OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
PS C:\Users\USER\Documents\Github\SC2002-Group-Project> & 'C:\Users\USER\Documents\jdk-18.0.1\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:62692' '-XX:+ShowCodeDetailsInExceptionMessages' '@C:\Users\USER\AppData\Local\Temp\cp_48ox1zbi1a7efqj0hhlw0140.argfile' 'src.main.java.ui.Main_UI'

--Welcome to MovieIckee Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
-----
```

## Automatic Price Calculation

Platinum (20 Dec 22), Adult: \$50.

```
Movies
-----
1. Jurassic World Dominion
2. Black Adam
3. Minions 2: The Rise Of Gru
4. Top Gun: Maverick
5. Life Is Beautiful
2

Showtimes
-----
1. A2 2022-12-20T08:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | 0 | 0 | 0 | 0 | 0 |
row: 2 | 0 | 0 | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | X | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | 0 |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
5
Select the column of your preferred seat
5

row: 5
col: 5
Please enter the age of the movieGoer
21
Please enter your email address, so that we can send you the ticket after the payment.
thosmail@gmail.com
Your price for the movie is: 50.00
```

Platinum (20 Dec 22), Senior: \$ 34.00.

```
Showtimes
-----
1. A2 2022-12-20T08:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | 0 | 0 | 0 | 0 | 0 |
row: 2 | 0 | X | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | X | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | X |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
5
Select the column of your preferred seat
4

row: 5
col: 4
Please enter the age of the movieGoer
75
Please enter your email address, so that we can send you the ticket after the payment.
senior@gmail.com
Your price for the movie is: 34.00
```

Gold (20 Dec 22), Adult: \$25.

```
Enter movie title:
Minions 2: The Rise Of Gru

Showtimes
-----
1. A3 2022-12-20T12:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | X | 0 | 0 | 0 | 0 |
row: 2 | 0 | 0 | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | X | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | 0 |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
5
Select the column of your preferred seat
5

row: 5
col: 5
Please enter the age of the movieGoer
21
Please enter your email address, so that we can send you the ticket after the payment.
lee@gmail.com
Your price for the movie is: 25.00
```

Gold (20 Dec 22), Senior: \$15.00

(Second row, less premium seats)

```
Showtimes
-----
1. A3 2022-12-20T12:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | X | 0 | 0 | 0 | 0 |
row: 2 | 0 | 0 | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | X | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | X |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
1
Select the column of your preferred seat
5

row: 1
col: 5
Please enter the age of the movieGoer
70
Please enter your email address, so that we can send you the ticket after the payment.
senior@mail.com
Your price for the movie is: 15.00
```

Silver(20 Dec 22), Adult: \$12.50.

```
Showtimes
-----
1. A3 2022-12-20T09:30
2. A3 2022-12-21T09:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | 0 | X | 0 | 0 | 0 |
row: 2 | 0 | 0 | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | 0 | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | 0 |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
5
Select the column of your preferred seat
5

row: 5
col: 5
Please enter the age of the movieGoer
21
Please enter your email address, so that we can send you the ticket after the payment.
lee
Your price for the movie is: 12.50
```

Silver (20 Dec 22), Adult: \$10.50

(second row, less premium seats)

```
Showtimes
-----
1. A3 2022-12-20T09:30
2. A3 2022-12-21T09:30
Select your preferred cinema and timing
1

Cinema layout
-----
col:      1  2  3  4  5
row: 1 | 0 | X | 0 | 0 | 0 |
row: 2 | 0 | 0 | 0 | 0 | 0 |
row: 3 | 0 | 0 | 0 | 0 | 0 |
row: 4 | 0 | 0 | 0 | 0 | 0 |
row: 5 | 0 | 0 | 0 | 0 | X |
*Note that the first 2 rows will have a price deduction of $2
Select the row of your preferred seat
1
Select the column of your preferred seat
5

row: 1
col: 5
Please enter the age of the movieGoer
22
Your price for the movie is: 10.50
```

→ The Ticket price takes into consideration, **Age** of MovieGoer, Premium of **Seat**, **Time** of the day, **Class** of Cinema, **Type** of movie, **Day** of the week and eve/**public holiday** to calculate the price of a particular movie screening. As seen in Appendix.

## Individual Booking History

### Booking History of Yu Xuan

```
--Welcome to MovieTickeing Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
-----
1

Enter your Username:
yx
Enter your Password:
1234

Login successful...

---Welcome to Our User application---
1. List the Movies
2. Book Tickets
3. View Booking History
4. Exit
-----
3
Your upcoming booking:
You have a movie: Black Adam at 2022-12-20T08:30 and your seat number is

You have no past booking
```

### Booking History of Robin

```
--Welcome to MovieTickeing Application--
1. Login
2. Continue as Guest
3. Register new User
4. Quit Application
-----
1

Enter your Username:
Robin
Enter your Password:
55555555

Login successful...

---Welcome to Our User application---
1. List the Movies
2. Book Tickets
3. View Booking History
4. Exit
-----
3
Your upcoming booking:
You have a movie: Prince at 2022-12-25T08:30 and your seat number
You have a movie: Prince at 2022-12-25T08:45 and your seat number

You have no past booking
```

→ **Every user** has their own **dedicated booking history**. They will only be able to view their own booking history and no one else can view it. We have also **separated the bookings** into **upcoming bookings** and **past bookings**. Allowing the user to differentiate bookings that are already older and upcoming bookings

## Create/Remove cineplex

Create a new cineplex

```
Please enter your option
1) Add Movie
2) Remove Movie
3) Update Movie
4) List top 5 movie
5) Add ShowTime
6) Remove ShowTime
7) View Promo Codes
8) Add Promo Codes
9) Edit System Configuration
10) Add new cinema
11) Remove cinema
12) Add new cineplex
13) Remove cineplex
14) Log out of staff account
```

```
12
Please enter the name of the new cineplex that you wish to add:
Cathay Cineplex
Cineplex has been added successfully!
```

```
1  JurongPointBranch
2  ChangiBranch
3  King
4  Cathay Cineplex
5
```

Remove cineplex

```
Please enter your option
1) Add Movie
2) Remove Movie
3) Update Movie
4) List top 5 movie
5) Add ShowTime
6) Remove ShowTime
7) View Promo Codes
8) Add Promo Codes
9) Edit System Configuration
10) Add new cinema
11) Remove cinema
12) Add new cineplex
13) Remove cineplex
14) Log out of staff account
```

```
13
Enter the cineplex name that you want to remove:
Cathay Cineplex
Cineplex deleted!
```

```
1  JurongPointBranch
2  ChangiBranch
3  King
4
```

→ Administrator will be able to add or remove any cineplex.

## Test Cases

### Admin Login Function

Username	Password	Output
staff1	staffpass	Login successful...
staff	staff1	Login Fail, Please try again...

### Admin UI

Input	Output
1	addMovie()
2	removeMovie()
3	updateMovie()
4	topFiveMovie()
5	addShowTime()
6	removeShowTime()
7	viewPromoCodes()
8	addPromoCodes()
9	editSysCon()
10	Logging out
11	Error, Logging out

### Add Movie

Name	Synopsis	Type	Director	Cast	Rating	MovieID	Output
Bee Movie	Fresh out of college, Barry ...	Cartoon	Simon J. Smith	Jerry Seinfeld, Renée Zellweger,	PG	542	Success
Bee Movie	Fresh out of college,	67483	Simon J. Smith	Jerry Seinfeld,	PG	542	Movie type is not

	Barry ...			Renée Zellweger,			a number!
Bee Movie	Fresh out of college, Barry ...	Cartoon	23498	Jerry Seinfeld, Renée Zellweger,	PG	542	Director is not a number!
Bee Movie	Fresh out of college, Barry ...	Cartoon	Simon J. Smith	Jerry Seinfeld, Renée Zellweger,	4.5	542	Please enter the correct rating!
Bee Movie	Fresh out of college, Barry ...	Cartoon	Simon J. Smith	Jerry Seinfeld, Renée Zellweger,	PG	movieID	MovieID must be a number!

Remove Movie

Input	Output
1	deleteMovieByTitle
2	deleteMovieByID
3	Thank you for using our application! Quit
a	You did not enter an integer. Wrong option. Quit



### Update Movie

Input	Output
1	updateMovieByTitle
2	updateMovieByID
3	Thank you for using our application! Quit
a	You did not enter an integer. Wrong option. Quit

### List top 5 movie

Input	Output
1	rankedByTicketSales()
2	rankedByReviewersRating()
3	Please choose one of the option

### Add ShowTime

Movie ID	Cinema ID	startTime	endTime	Cinema Class	Output
S0001	A1	2022-11-08 10:30	2022-11-08 12:00	Platinum	ShowTime successfully added
00001	A1	2022-11-08 10:30	2022-11-08 12:00	Platinum	Invalid MovieID
S0001	11	2022-11-08 10:30	2022-11-08 12:00	Platinum	Invalid CinemaID
S0001	A1	2022-11-09	2022-11-08 12:00	Platinum	Invalid startTime
S0001	A1	2022-11-08 10:30	2022-11-09	Platinum	Invalid startTime
S0001	A1	2022-11-08 10:30	2022-11-08 12:00	Bronze	Invalid Cinema Class

### Remove ShowTime

Input	Output
1	removeShowTimeByMovieIndex()
2	removeShowTimeByMovieInfo()
3	Error, Exit process

removeShowTimeByMovieIndex()

Input	Output
1	Success
S	Please enter in integer!

removeShowTimeByMovieInfo()

Movie ID	Cinema ID	startTime	endTime	Cinema Class	Output
S0001	A1	2022-11-08 10:30	2022-11-08 12:00	Platinum	ShowTime successfully removed
00001	A1	2022-11-08 10:30	2022-11-08 12:00	Platinum	Invalid MovieID
S0001	11	2022-11-08 10:30	2022-11-08 12:00	Platinum	Invalid CinemaID
S0001	A1	2022-11-09	2022-11-08 12:00	Platinum	Invalid startTime
S0001	A1	2022-11-08 10:30	2022-11-09	Platinum	Invalid endTime
S0001	A1	2022-11-08 10:30	2022-11-08 12:00	Bronze	Invalid Cinema Class

addPromoCodes()

Promo	Count	startTime	endTime	Status	Offer	Output
S0001	5	2022-11-08 10:30	2022-11-08 12:00	Ready	0.50	Promo Code Successfully Added
464564	5	2022-11-08 10:30	2022-11-08 12:00	Ready	0.50	Please enter only words
S0001	sggs	2022-11-08 10:30	2022-11-08 12:00	Ready	0.50	Please enter integer
S0001	5	2022-11-09	2022-11-08 12:00	Ready	0.50	Invalid startTime
S0001	5	2022-11-08 10:30	2022-11-09	Ready	0.50	Invalid endTime
S0001	5	2022-11-08 10:30	2022-11-08 12:00	Not Ready	0.50	Invalid Status
S0001	5	2022-11-08 10:30	2022-11-08 12:00	Ready	dfgfdg	Please enter number

addCinema()

CinemaID	Branch	No of seats	Class of cinema	Layout Type	Output
SC2002	OODP	500	Gold	Medium	Cinema has been added successfully!
SC2002	OODP	Five Hundred	Gold	Medium	Please enter an integer.
SC2002	OODP	500	ffff	Medium	Please enter one of the followings:  Platinum,Gold,Silver
SC2002	OODP	500	Gold	ggg	Please enter one of the following: small, medium, large

removeCinema()

CinemaID	Output
SC2002	Cinema deleted!
C	Cinema not found!

addCineplex()

Name of Cineplex	Output
Cathay	Cinema has been added successfully!
Cathay Village	Cinema has been added successfully!

removeCineplex()

Name of Cineplex	Output
Cathay	Cinema deleted!
Hi	Cineplex not found!

### **Movie Goers Login Function**

Username	Password	Output
yj	1234	Login successful...
yj1234	1234	Login Fail, Please try again...

MovieGoers UI

Input	Output
1	DisplayMovie_UI.displayInformation(userName)
2	DisplayMovieBooking_UI.displayInformation(userName)
3	BookingManager.getBookingHistory();

4	exit
5	Please enter a number between 1 and 4

List the movies

Input	Output
1	listBySales();
2	listByRating()
3	MovieListing.getMovieID(movieTitle)
4	Please enter a number between 1 and 3

Book Tickets

Input	Output
1	listBySales();
2	listByRating()
3	MovieListing.getAvailableMovieID(movieTitle)
4	Please enter a number between 1 and 3

List top 5 movies by sales

Input	Output
1	listBySales();
ccc	Please enter a number between 1 and 3

Choose showTime

Input	Output
1	Success
ccc	Please enter a number between 1 and 3

Choose Row & Column

Row	Column	Output
1	1	SeatBooked_Controller.updateSeatBooked(row, col, cinemaID, time)
1	2	Seat is taken, please choose another seat.

Enter Age

Age	Output
10	new Price(time, age, 1, SystemConfigController.getClassMulti(tmp.get(choice).getClassOfCinema().toString()), row)
fff	You did not enter numerical number. Please enter it again.

Enter Promo Code

Promo Code	Output
PromoNow	Success
ff	The promo code is invalid!

Payment

Payment Method	Output
1. Paylah	Success
2. Credit Card	Success
3	Please enter a number between 1 and 2

List top 5 movies by rating

Input	Output
1	Success
ccc	Please enter a number between 1 and 3

Search by movie title

Input	Output
Black Adam	Success
Bee	Movie not found

## **Propose of New Features:**

### **New feature 1:**

To enhance functionality of MOBLIMA, the application should be able to cater for purchases other than movie tickets, such as food and beverages sold in the cinema. Hence, users can purchase food and beverage as a set with movie tickets which is often seen in real world scenarios.

#### → Single Responsibility Principle:

Every class is singly responsible, such as File\_IO is only responsible for reading and writing of the files. Hence, to load and save data into a particular file, File\_IO is called instead of having a read-write method for every class.

#### → Open-Closed Principle:

Items sold have individual classes. To expand the number of items sold, our design can easily be expanded to include more items to be sold such as having a popcorn class to be inherited by an item superclass.

#### → Reusability:

To process the payment, Payment\_UI can be called and reused to process the payment.

Thereby, achieving reusability of code and extensibility with low coupling to increase the number of items sold readily.

### **New feature 2:**

With the hastening of the finTech industry, it is inevitable that more payment functions will be developed beyond conventional visa, nets or cash payment. One such example is the use of cryptocurrency. Hence expandability to accept newer payment methods should be allowed with minimal changes.

#### → Liskov's Substitution Principle:

With implementation of a new payment method, procedure of payment remains largely unchanged as pre-conditions are no stronger, while post-conditions are no weaker than base class.



→ Dependency Injection Principle:

Our current payment method is invoked using a payment interface, such that a new payment method can be easily supported with a new payment class inheriting from `PaymentMethodInterface` which is both dependent upon abstraction.

Thereby, allowing reusability of code and extensibility to expand supported payment methods.