

48024

Applications Programming

Assignment 2

Topics:

OO design, GUIs, MVC, tables, lists

Learning Outcomes:

This assignment supports objectives 3 - 5

Due date:

14th May 2021 - 11:59PM (Friday Week 11)

Weight:

25%

Individual Work

All work is individual. You must write every line of code yourself except for code copied from study module sample code, lecture sample code, tutor demos or lab code.

In most cases, you may discuss ideas, approaches and problems. However, if an assignment task is labeled as "Advanced", you must not discuss ideas, approaches and problems. Advanced tasks are designed to test your ability to think on your own.

You **MUST NOT** let another student see your solution code, and you **MUST NOT** look at another student's solution code. Sharing your code on public forums such as the discussion board, or Internet forums such as stackoverflow.com is not permitted. More information about Academic Misconduct can be found at:

<http://www.gsu.uts.edu.au/rules/student/section-16.html>

Skeleton Code

As a starting point for this assignment, you must use the skeleton code provided on ED (<https://edstem.org/courses/5167/lessons/10525/slides/76638>) in Assignment 2 Description. There are two options, a NetBeans version, which includes the structure required to import directly into NetBeans, and a plain version for other IDEs (which tend to be less rigid about their import requirements).

The skeleton code contains a file called progress.txt which you must fill in and submit with your project to ED as you progress on the assignment (read Submission to ED for further details). You are NOT allowed to modify the model or remove any of the starting classes or FXML files.

Expected workload

The time to do the assignment to a distinction level (i.e. a mark between 75% to 84%) has been estimated at 25 hours for a student of average ability who has completed all the tutorial and lab exercises.

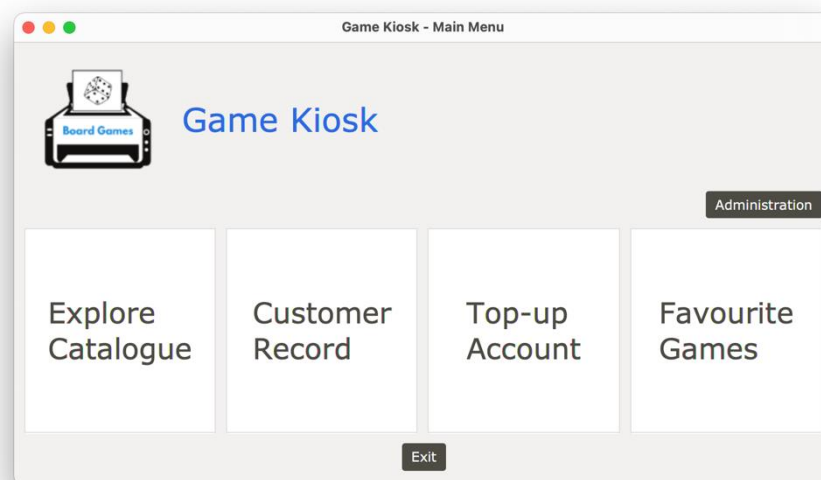
Specification

The specification is presented in several parts. In this document is given a series of screen shots and textual descriptions for visual reference.

A demonstration is also presented in the video found on ED in Assignment 2 <https://edstem.org/courses/5167/lessons/10525/slides/76640>. This demonstration video is considered part of the specification and contains important details about the dynamic function of the assignment.

The screens presented below are given in no particular order, other than vague logical grouping, and in particular the order should not be construed as an indication of difficulty or recommended order of implementation.

Main Menu (10%)

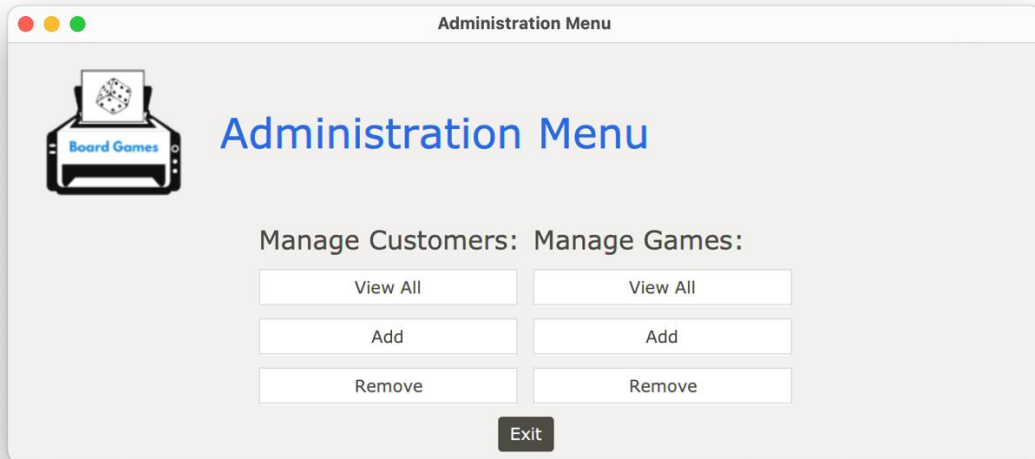


The main menu is opened when the application launches. It has buttons to access the catalogue, view customer records, top-up an account, view customers' favorite games and the administration menu. It also has an exit button which shuts the entire application down. The main menu has a header section with a small logo and the title "Game Kiosk".

A little hint for multi-line text area:

```
<TextArea prefHeight=" " prefWidth=" " text="{ 'Multi\nLine\tTab' } " />
```

Administration Menu (5%)



Administration Menu

Board Games

Administration Menu

Manage Customers: Manage Games:

View All View All

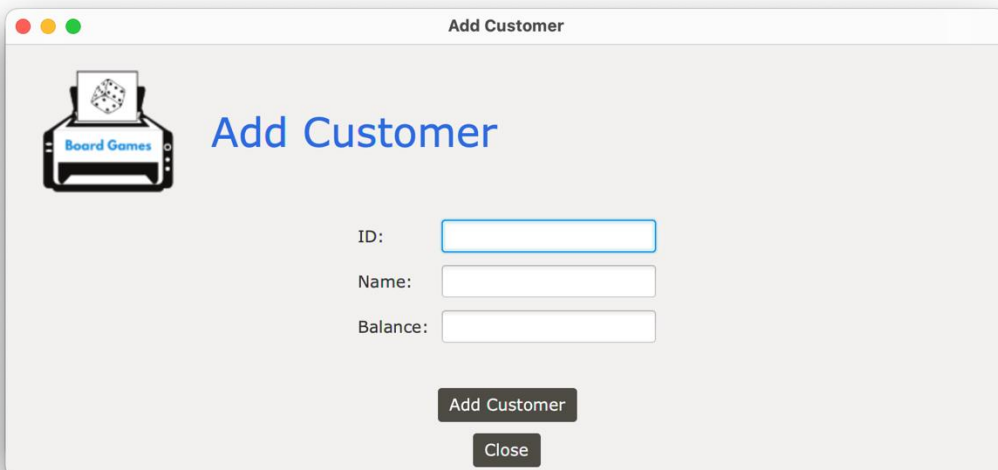
Add Add

Remove Remove

Exit

The administration menu is launched from the main menu by opening a separate window. It has buttons to add, list and remove customers, and add, list and remove games. The Exit button closes the administration menu window. The administration menu includes a similar header as the main menu (and catalogue menu).

Add Customer and Add Game (10%)



Add Customer

Board Games

Add Customer

ID:

Name:

Balance:

Add Customer

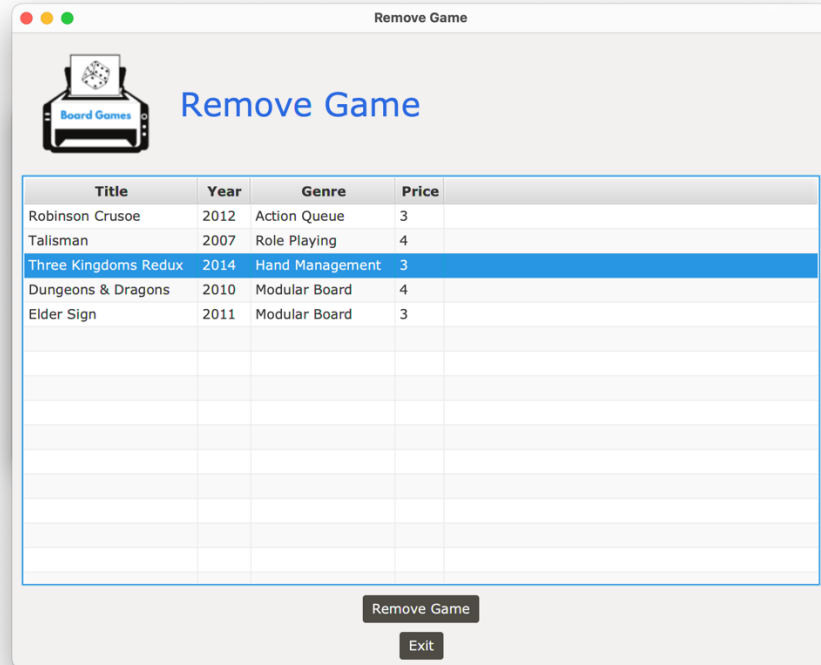
Close

These two windows add a customer to the system, and a game to the catalogue respectively. Each has text fields for data entry, each with an appropriate label. The Customer ID and Balance fields expect an integer, all others expect strings. The Add buttons handle the addition of the customer/game to the appropriate component of the model. They also both have areas to give feedback to the user if the action was successful or not.

Remove Customer and Remove Game (10%)

These two windows handle the removal of Customer and Games. Remove Customer shows a table with the current Customers, remove Game shows a table of the games available to be removed (which may not be all the games in the kiosk). Each remove button removes the currently selected item in its window one item at a time. The close buttons close their windows.


ID	Name	Balance
101	Jaime	10
102	Luke	10
103	William	1



Customer Record and Favourite Games (10%)

These two windows display the customer record and the customer's favourite games. They both include display areas for the related lists, suitably labelled. They both also include a text field for the entry of a Customer ID, along with a label and a button to refresh the tables based on the entered Customer ID. There is an area showing feedback text indicating which customer is selected.





Customer Record

Customer ID: Select Customer

101 Jaime \$ 4

Rented Games


Title	Year	Genre	Price
Three Kingdoms Redux	2014	Hand Management	3

Renting History

Title	Year	Genre	Price
Three Kingdoms Redux	2014	Hand Management	3
Elder Sign	2011	Modular Board	3

Close

Catalogue Menu (5%)



Catalogue

View Games:

All Games

Available Games

Games by Genre

Games by Year

Rent a Game

Return a Game

Exit

The catalogue menu presents the options for interacting with the catalogue. It includes the menu header (as with the main menu and the administration menu).

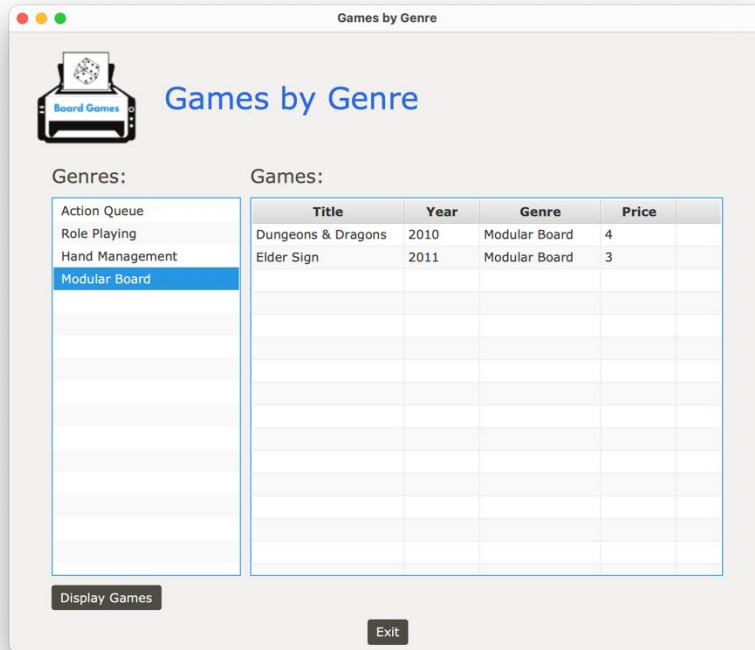
Show All Games and Show Available Games (10%)

The complete catalogue (accessed from Show All Games) and the available games menu show the lists of all the games, and the games available for someone to rent, respectively.

[illegible][illegible]

Browse by Genre and Browse by Year (10%)

These two windows display labelled lists of years/genres from which the selected item is used to display the suitably filtered list of games in the table display area.



Games by Genre

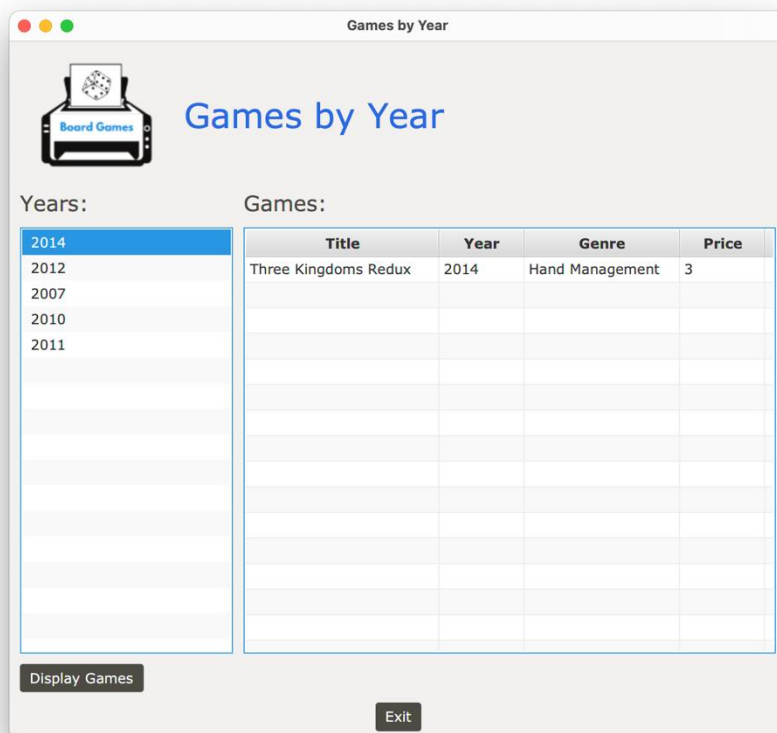
Genres:

- Action Queue
- Role Playing
- Hand Management
- Modular Board**

Games:

Title	Year	Genre	Price
Dungeons & Dragons	2010	Modular Board	4
Elder Sign	2011	Modular Board	3

Buttons: Display Games, Exit



Games by Year

Years:

- 2014**
- 2012
- 2007
- 2010
- 2011

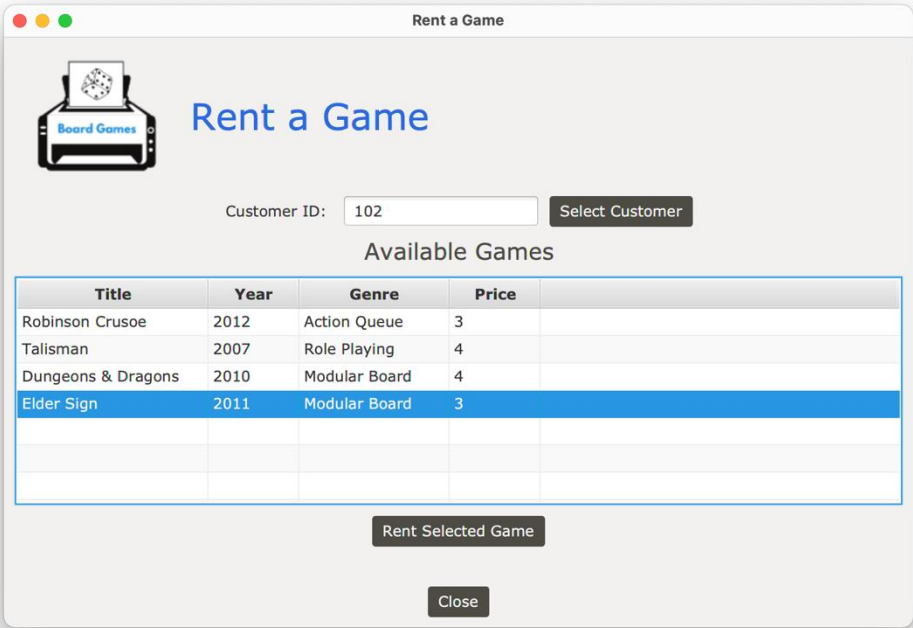
Games:

Title	Year	Genre	Price
Three Kingdoms Redux	2014	Hand Management	3

Buttons: Display Games, Exit

Rent a Game (10%)

The Rent a Game window allows the entry of a Customer ID, which when the Select Customer button is clicked, shows a list of games that are available to borrow. This list should exclude games that were rented by other customers. Note that Select Customer button is disabled if there is no text in the Customer ID field, and the Rent Selected Game button is disabled if no game is selected.



Rent a Game

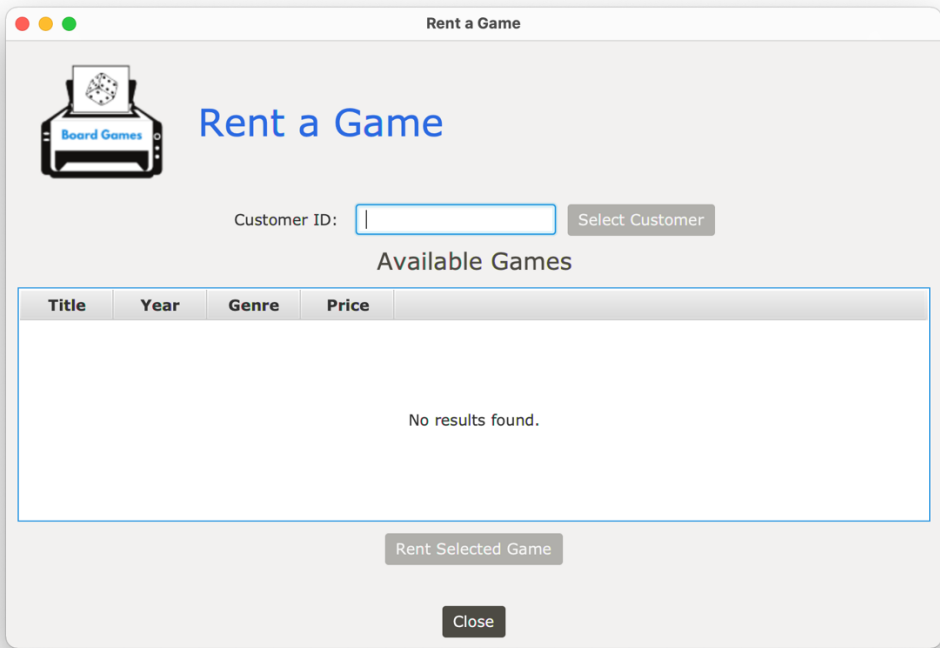
Customer ID: Select Customer

Available Games

Title	Year	Genre	Price
Robinson Crusoe	2012	Action Queue	3
Talisman	2007	Role Playing	4
Dungeons & Dragons	2010	Modular Board	4
Elder Sign	2011	Modular Board	3

Rent Selected Game

Close



Rent a Game

Customer ID: Select Customer

Available Games

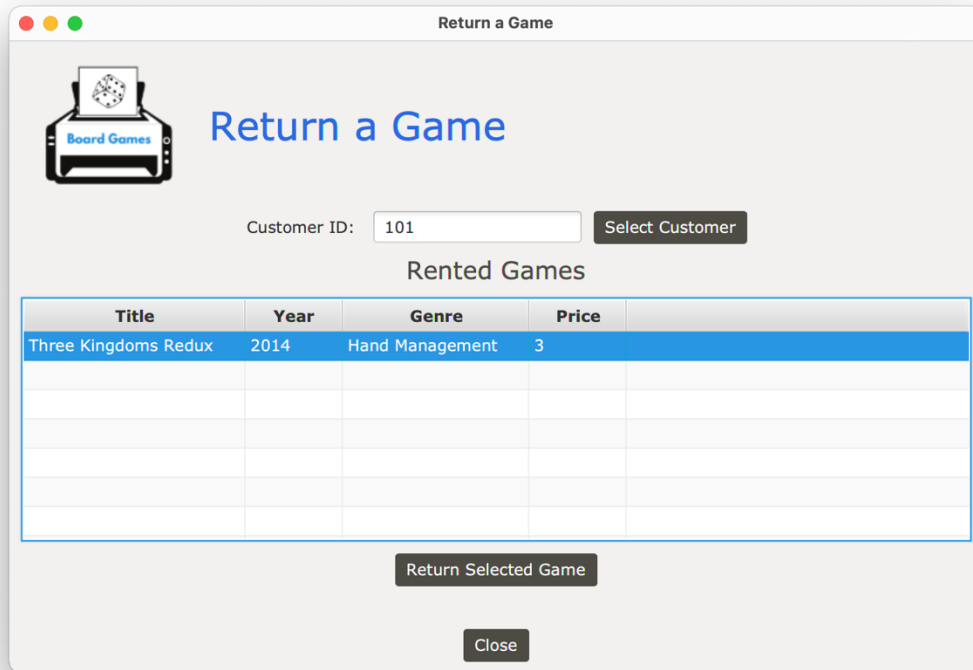
Title	Year	Genre	Price
No results found.			

Rent Selected Game

Close

Return a Game (10%)

The return a game window operates similarly to the rent a game window but shows the games that the customer has currently rented (and are thus available to return). The buttons in this window have similar restrictions to the rent a game window (text in Customer ID field, game selected).

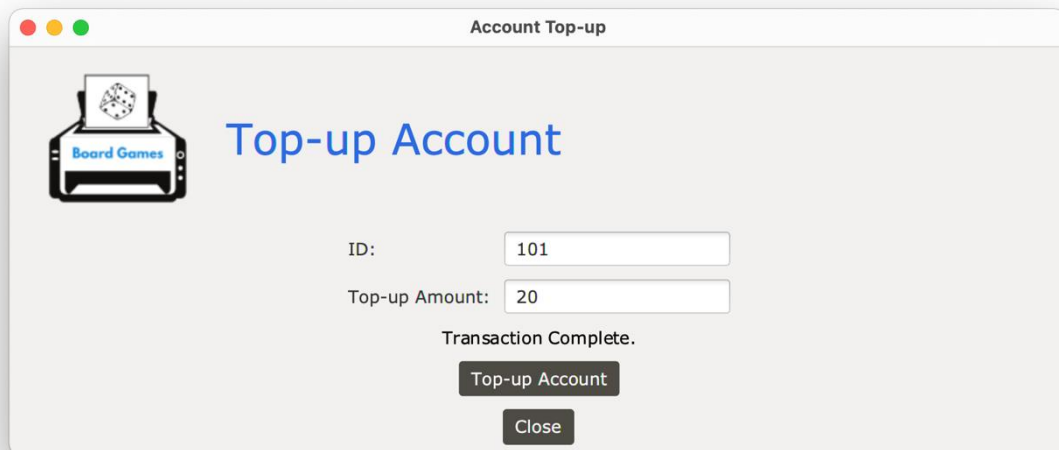


The screenshot shows a window titled "Return a Game". It features a "Board Games" logo on the left. The main heading is "Return a Game". Below this, there is a "Customer ID:" label followed by a text input field containing "101" and a "Select Customer" button. A table titled "Rented Games" is displayed, with columns for Title, Year, Genre, and Price. The first row is highlighted in blue and contains the text "Three Kingdoms Redux", "2014", "Hand Management", and "3". Below the table is a "Return Selected Game" button. At the bottom of the window is a "Close" button.

Title	Year	Genre	Price
Three Kingdoms Redux	2014	Hand Management	3

Top-Up Account (10%)

The top-up account window allows the entry of a Customer ID and a top-up amount (an integer). The account top-up only takes place when the user clicks on the Top-Up Account button. The window also includes an area for text feedback as to whether a top-up was successfully completed.



The screenshot shows a window titled "Account Top-up". It features a "Board Games" logo on the left. The main heading is "Top-up Account". Below this, there is an "ID:" label followed by a text input field containing "101". Below that is a "Top-up Amount:" label followed by a text input field containing "20". A message "Transaction Complete." is displayed. Below the message is a "Top-up Account" button. At the bottom of the window is a "Close" button.



Requirements

Layout

To get full marks, you should layout your windows to look as close as possible to the screenshots. This means that you should try to duplicate the spacing between and around nodes that is shown in the screenshots, and the width and height of the nodes, and the alignment of the nodes. In the benchmark solution, all hgap, vgap and spacing properties for GridPanes, HBoxes and VBoxes were set to 10.

Style

A CSS file is included in the skeleton code which provides all the styles used in the assignment.

Code

Your solution must satisfy the following code requirements:

- Your solution must follow the MVC architecture.
- Your solution must keep the package structure and class names that were provided in the skeleton code.
- The models must notify the views of changes by correctly applying the JavaFX property patterns and observable lists. Model data that can change must be observable. Model data that never changes need not be observable.
- The views must be laid out in FXML.

Peer marking and demonstration

In your scheduled week 12 lab class you must be prepared to demonstrate your assignment to your tutor and to explain parts of your code to your tutor if requested. If you are unable to explain your code, it may impact your marks. Your presence is required at this class. Any student who is not present without being granted prior permission may have up to 50% of their marks for this assignment deducted.

In addition to demonstrating your assignment, you will also be assigned two other students to peer mark, and two other students will be assigned to peer mark you. The purpose of this peer marking is to mark the functionality of your application which cannot be tested by ED. Your marks for functionality will be based on these peer marks after they are moderated by the subject coordinator. Aside from marks for the functionality, the subject coordinator will also mark your code to ensure that all code requirements have been met. Your final mark will be a combination of marks for functionality and marks for code (See "Marking scheme"). Note that you can only be marked for features that can be demonstrated to work.

Marking the code and analyzing spoofing, cheating and plagiarism is done in the two weeks following the due date. If you are suspected of Academic Misconduct, I will forward your case to the Misconduct Committee and will notify you by manual feedback in ED. Your mark will be finalized within 2 weeks of the due date.



Submission to ED

READ THIS ENTIRE SECTION CAREFULLY

Included in the skeleton code is a file called progress.txt which you must fill out as you progress through the assignment. This file will contain lines such as these:

```
[?] The Main menu window is at least partially done.
[?] The Main menu window is done.
[?] The Catalogue menu window is at least partially done.
...etc...
```

As you make progress on your assignment, you must edit this file by changing each [?] into a [y] and then submit your progress to ED. Don't forget to save this file before submitting. For example, after you get the main menu window partially done (even if you have only done a small amount), you edit this file as follows:

```
[y] The Main menu window is at least partially done.
[?] The Main menu window is done.
[?] The Catalogue menu window is at least partially done.
...etc...
```

Then you submit your project to ED so that there is a record of what your code looked like when you first started to make progress on your Main menu window. After you complete the Main window feature, you should again update this file as follows:

```
[y] The Main menu window is at least partially done.
[y] The Main menu window is done.
[?] The Catalogue menu window is at least partially done.
...etc...
```

Then you submit your project to ED again so that there is a record of what your code looked like when you completed this feature.

It is not always required that you complete a feature before moving onto the next feature. For example, your progress.txt file may read:

```
[y] The Main menu window is at least partially done.
[?] The Main menu window is done.
[y] The Catalogue menu window is at least partially done.
...etc...
```

This would indicate that you partially completed the Main menu window, then moved on to the Catalogue menu window. This is allowed, as long as you have completed at least enough of the Main menu window that will allow you to correctly open the Catalogue menu window.

Important: If you don't submit your progress on a particular feature, then your marks for that feature won't count! That is, you are only marked for those features where you submit evidence of your progress. Be very careful to always submit your progress as soon as you make progress so that you don't lose any marks unnecessarily.

Your solution is to be submitted to ED at <https://edstem.org/courses/5167/lessons/10525/slides/76638>. You must submit your progress to ED in increments of no larger than 25 marks. The penalty for making a submission to ED that is N marks higher than your previous best mark, where $N > 25$ is N marks. To avoid a penalty, submit your progress in smaller increments (2-3 windows). That is, you should not introduce more than 4 new [y] answers in your progress.txt file in a single submission.

Submission to Canvas

Canvas Entry for assignment 2 submission will become available in Lab 11 session. For peer marking allocation, you must submit your assignment to Canvas before the due date. Peer marking process will start directly after the due date and conclude in lab 12 session. Your assignment should be submitted as a JAR file that includes:

- All FXML, CSS and image files required to run your assignment.
- The progress.txt file at the top level of your project directory structure.

You can check the JAR file on a lab PC to make sure it works. The JAR file should **NOT** include:

- Any identical information such as student ID or student name.
- Any Java source files required to compile your assignment.

There is no scheduled late submission period. No extension will be given before or after the due date. If you have documentary evidence for special consideration, you will be assigned an alternative assignment instead of getting an extension.

To apply for special consideration for reasons including unexpected health, family or work problems, you can refer to more information about how to apply for special consideration at <http://www.sau.uts.edu.au/assessment/consideration.html>

Online support

The Assignment 2 discussion board has been set up so that students can ask questions, and other students can reply. The teaching team may post a reply only if they think the student response was wrong, or in the case of correcting a mistake in the assignment specification.

You must not post Java code to the discussion board. The board is there to help you, not to provide the solution. Posting your code is academic misconduct and will be reported. Each time this rule is violated, I will delete the code and post a comment of the form: "Strike 1: Posting code". After 3 strikes, the discussion board will be deleted because it did not work.

A dynamic FAQs (Frequently Asked Questions) has been pinned as the megathread in the channel and their answers will be posted on ED alongside the question. If you have a question, check the FAQ (megathread) first, it may already be answered there. You should read the FAQ (megathread) at least once before you hand in your solution, but to be safe check it every couple of days. Anything posted on the FAQ (megathread) is considered to be part of the assignment specification. The FAQ will be frozen (no new entries) two days before the due date; no questions will be answered after it is frozen.

If anything about the specification is unclear or inconsistent, check the FAQ (megathread) first, then contact the subject coordinator who will try to make it clearer by replying to you directly and posting the common questions and answers to the FAQ. This is similar to working on the job, where you ask your client if you are unsure what has to be done, but then you write all the code to do the task. Email huan.huo@uts.edu.au to ask for any clarifications or corrections to the assignment.

Test Cases

For testing purposes, please make sure that you have a database of games and customers as follows:

ID	Name	Balance
101	Jaime	10
102	Luke	10
103	William	1

Title	Year	Genre	Price
Robinson Crusoe	2012	Action Queue	3
Talisman	2007	Role Playing	4
Three Kingdoms Redux	2014	Hand Management	3
Dungeons & Dragons	2010	Modular Board	4
Elder Sign	2011	Modular Board	3

Marking Scheme

Task	Mark
Main Menu	10
All nodes are shown and FXML is used	4
The layout is correct	2
Fonts and colours are correct	1
The buttons open the correct stages	1
The exit button works	2
Administration Menu	5
All nodes are shown and FXML is used	1
The layout is correct	1
Fonts and colours are correct	1
The buttons open the correct stages	1
The close button works	1
Catalogue Menu	5
All nodes are shown and FXML is used	1
The layout is correct	1
Fonts and colours are correct	1
The buttons open the correct stages	1
The close button works	1
Add Customer and Add Game	10
All nodes are shown and FXML is used	4
The layout is correct	4
The close buttons work	2
Remove Customer and Remove Game	10
All nodes are shown and FXML is used	1
The layout is correct	1
The customers and games are displayed as a table	4
The tables are displayed correctly using the observer pattern (*)	3
The close buttons work	1



Customer Record and Favourite Games	10
All nodes are shown and FXML is used	1
The layout is correct	1
The buttons are correctly disabled and enable based on input	2
The text feedback displays the correct information for valid and invalid Customer IDs	2
The customer record is correctly updated using the observer pattern (*)	3
The close buttons work	1

Show All Games and Show Available Games	10
All nodes are shown and FXML is used	1
The layout is correct	1
The columns are correctly labelled	4
The tables are correctly updated using the observer pattern (*)	3
The close buttons work	1

Show Games By Genre and Show Games By Year	10
All nodes are shown and FXML is used	1
The layout is correct	1
The tables and lists display the correct information (and only the correct information)	3
The tables and lists are correctly updated using the observer pattern (*)	4
The close buttons work	1

Rent a Game	10
All nodes are shown and FXML is used	1
The layout is correct	1
The buttons are correctly disabled and enabled according to the conditions in the specification	3
The table is correctly refreshed when a game is rented	3
The table displays the correct set of available games	1
The close buttons work	1

Return a Game	10
All nodes are shown and FXML is used	1
The layout is correct	1
The buttons are correctly disabled and enabled according to the conditions in the specification	3
The list is correctly updated using the observer pattern (*)	4
The close buttons work	1



Top-up Account	10
All nodes are shown and FXML is used	1
The layout is correct	1
The buttons are correctly disabled and enabled according to the conditions in the specification	3
The text feedback correctly displays the result of attempting to top-up an account	4
The close buttons work	1

(*) The code will be checked by the subject coordinator in the 2 weeks following the due date