Ryerson University

Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Title:	**Object Oriented Eng Analysis and Design**
Course Number:	**COE528**
Semester/Year (e.g.F2016)	**W2021**

Instructor:	**Dr. Olivia Das**

Assignment/Lab Number:	**Project**
Assignment/Lab Title:	**Project**

Submission Date:	**April 4, 2021**
Due Date:	**April 4, 2021**

Student LAST Name	Student FIRST Name	Student Number	Section	Signature*
Patel	Manav	500967756	6	MP
Alvi	Rania	500814930	6	R.A
Gurram	Karthik	500880724	6	KG
Jalali	Waqas	500899545	6	WJ

Use-Case Description;

A use case represents a class of functionality provided by the system. A textual representation of the use case can be described in 7 parts as follows:

1. Unique Name: BookStoreApplication
2. Participating Actors: Owner and Customer
3. Flow of Events:
    3.1. To add a new row for a book into the book table , the owner enters the title and price of the book by clicking the [Add] button. The owner deletes a book from the book table by selecting the specific row and clicking the [Delete] button.
    3.2. To add a new customer into the customer table, the owner enters the customer's username, password by clicking on the [Add] button. The owner deletes the customer from the customer table by selecting the corresponding row and clicking the [Delete] button.
    3.3. To buy one or more books, the customer will select the checkboxes present on the corresponding row(s) of the book(s) and click [Buy] or [Redeem and Buy]. If the customer chooses to redeem their points they will simply click [Redeem and Buy], for which 1 CAD will be deducted from the total cost for every 100 points redeemed. If the customer chooses to simply buy a book without redeeming any points, they will click[Buy].
    3.4. After either [Buy] or [Redeem and Buy] is clicked, the total transaction cost is calculated. If the transaction was made without redeeming points, the new points are updated such that for every 1 CAD spent, the customer receives 10 points. Depending on the points the customer has, the status is updated to either Silver or Gold.
4. Entry Condition: This use case starts when the Owner and Customer are logged in to the system.
5. Exit Condition: The use case terminates when the Owner and Customer are logged out of the system.
6. Exceptions: First exception would be when an incorrect username and/or password is

entered. If this happens, it will print “invalid” and an exception should be thrown. If the username or password does not exist, the User is notified immediately. Next, if any sort of error occurs at runtime and if any connection is lost between the admin and user or customer and User, that would be an exception and it should be thrown.

7. Special Requirements: It is assumed that there is one copy of each book. After a customer purchases a book, the book should be deleted from the book table.

Rationale Behind Using the State Design Pattern

While designing and implementing the bookstore application, the state design pattern was used. The state design pattern is used when the behaviour of a class or method is dependent on a state. This change must occur at runtime. Also when states are changed, the current class is not affected, meaning the state object would make the state change, but the current state or class won’t change itself.

The current application will be in one of two states as a User: Owner and Customer. The owner state will have different functionalities compared to the customer state. While implementing the state pattern, each state was implemented in a separate class. Based on the general diagram of a state machine, the User class would be the ‘client’, the GUI would be the ‘Context’, and there would be two states as mentioned above that would each have multiple sub-states. The first state would be the Admin. This state has three of its own sub-states: Books, customers and logout. If the state books is clicked, it will go to the list of books; if the state customers is clicked, it will go to the list of customers, and finally if the logout state is clicked, it will logout and go back to the original login screen. Similarly, for the state ‘customers’, it has three other states: the buy state which would take you to another window showing the total cost and status; the redeem and buy state which would again take you to another window and show the cost and status; and finally the logout state which will log the customer out of the application.