# Acknowledgement

In preparation for my project, I needed to take the assistance and the direction of some regarded people, who merit my most profound appreciation. As the fulfillment of this project gave me much pleasure, I would like to appreciate and show my gratitude Mr. Sudeep Bajimaya, Course Instructor, on Softwarica College for giving me good guidelines and support for my project through various consultations. I would also like to expand my appreciation to every one whole guided me in this project.

In addition, I would like to thank Mr. Manish Khanal for introducing ADO.Net framework and Mr. Pratik Bhusal for introducing the MVC design pattern for C#.net to me.

# Abstract

Manually keeping tracks of stock and sales consume lots of effort and time. It also aids in human errors that complicates the work and causes a major problem in the long run. The purpose of this project is to develop a system to automate stock management and sales for the business. This project will allow the user to control the stock level and create a report to demand.

Liquor stock management system is a desktop-based application that automates the management of stock in an easier, faster way for the owner. It is a POS standalone application that allows the owner to record all added new stocks, manipulate stocks and create customer bills and stock reports. This application is developed using Object-oriented Analysis and Design Methodology and MVC design pattern with ado.net framework library(C#.net).

The application has a single user login system through pin protection. This system has the function to differentiate the liquor according to its category and manipulate the liquor accordingly. It allows to view all liquors, search and filter the liquor and restock liquor quantity. It allows adding threshold quantity which allows the user to get notified when the quantity is less than the threshold. It has an automated process of managing liquor quantity while creating customer bill. It also has a loyalty customer system to allow a discount for most often purchasing customers.

Conclusion, this the project will allow liquor store to record liquor stock data on database and automate their day to day sales and stock tracking process.

# Chapter 1: Introduction

## 1.1 Introduction to project

Stock management is one of the basic day to day task in a business. It allows the business to accurately measure their stock and their eating according to stock spent. Stock management is a common need in every business. Thus, liquor stock also needs to record their stock levels and daily sales. Human calculation and management are prone to make an error. The project is to build the system to automate the stock management for liquor store.

Liquor stock management system is a desktop-based application for managing the stock in a liquor store. It will automate the day to day process of stock management and sales record. The will make stock management easier, simpler and reliable for the store owner. The system will allow the user to create a bill of liquor transaction and record accordingly. The system will provide information about the stock level and will provide a function to control the stock level.

## 1.2 Background and problem statement

The system built is for a liquor stock that still operates all the stock management task manually by the owner. The daily sales are recorded on handwritten bills. Keeping track of stock levels and sales are susceptible to make an error. As owner needs to record day to day. This also takes lots of time and effort to maintain accurate records. Therefore the automated application will allow the owner to remove human error and makes stock management easier, less effort and simpler. The information is stored in a secured database thus there is less possibility of a loss of records. Therefore, from this project, the owner controlling the stock will be more efficient and effective.

## 1.3 Overview of project

The main aim of the project is to develop a desktop based application for the client to record and perform stock management process and billing and record transaction of the shop. The system will be time and cost efficient, easy to use and provide electronic way to store the data. It will also provide a feature such as loyalty discount service for customer (discount accord to customer spending) and stock level control and notification so that the shop will have sufficient stocks to meet customer needs. Measuring the change in stock will allow the store to plan for future inventory needs.

## 1.4 Aim and objectives

**Aims**

* To build desktop based standalone POS application for the liquor store to store liquor stocks information and manipulation of stocks.
* To make a billing process electronic and automated the stock control through bills.
* To make liquor store stock management activities easier, reliable, time-efficient and record accurate data.

**Objectives**

* To analyze and check whether the project is feasible or not for the shop.
* To make the system user-friendly and easy to learn.
* To store the information of stocks in the database systematically.
* To make a secure POS system.
* To allow a user to accurately control the stock level.
* To automate the stock control when spent.
* To help a user know the stock level to allow future stock prediction and planning.
* To show reports according to sales of the stock.
* To ensure the availability of the stock for meeting consumer demand.
* To provide billing service.
* To provide customer discount feature according to their spending.
* To accurately store sales transaction records.
* To test the system to check bugs or errors.

# Chapter 2: Analysis

## 2.1: Introduction to Analysis

An analysis is a process of determining the user needs and expectation for the new or upgraded products. These features are called requirements. In this process, a systematic examination and evaluation of information or data are performed to discover the important components to build the system.

The analysis is the first phase of the software development life cycle (SDLC). In this software development conceptual model, analysis focuses on the following parts:

* Gather, analyze, and ratify the information.
* Define the requirements and prototypes for new system.
* Evaluate the alternatives and prioritize the requirements.
* Examine the needs of end-user and enhance to meet the system goal.
* Prepare the Software Requirement Specification (SRS) document, which specifies the software, hardware, functional, and network requirements of the system.

## 2.2: Analysis Methodology

The project uses Object-Oriented Analysis and Design Methodology. It is a technical method of analyzing and designing a system based on their object models. An object is an instance of anything that represents a real-world object and has all the same types of characteristics (properties), behavior (methods), and states (data). This methodology not only focuses on processes or data of the system but outlook the system as a collection of object that can interact with each other to accomplish tasks.

Object-Oriented Analysis and Design (OOAD) often include stages i.e. requirements, planning, design, coding, testing, deployment, and maintenance. These stages are similar to the waterfall SDLC and does not require additional tasks for the project as the requirement are well defined. That’s why I have decided to use OOAD for this project.

In the Object-oriented Analysis, we undertake the following tasks as mentioned below:

1) Elicit requirements: Define what the problem the system is trying to solve, what the system needs to perform.

2) Specify requirements: Describe the requirements i.e. use cases or user stories.

3) Model: Identify the important objects, their relationships and functionality/behavior.

## 2.3: Feasibility Study

The feasibility study deals whether the project’s practical extent that can be performed successfully. Basically, feasibility study is performed to determine whether the solution to a problem is practical in real world scenario. There are different types of feasibility studies, i.e.:

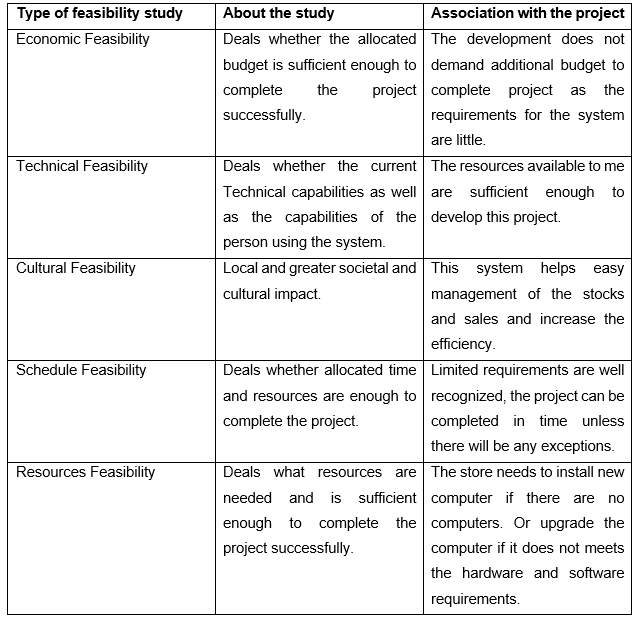


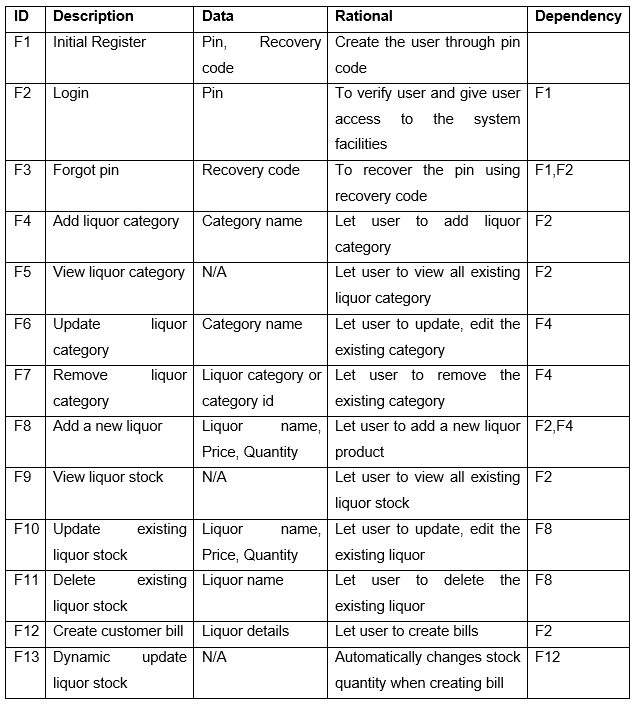
Table 1: Feasibility study

## 2.4: Requirement Analysis

The requirement analysis is the process of gathering, analyzing and defining the technical requirements of the users for the system.

## 2.4.1: Functional Requirement

The functional requirements generally defines tasks or processes of what a system should do.



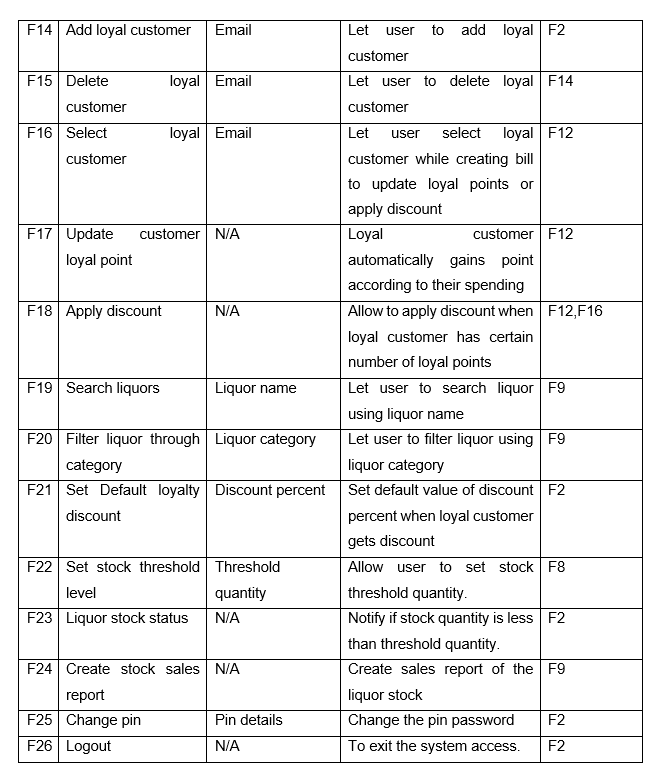
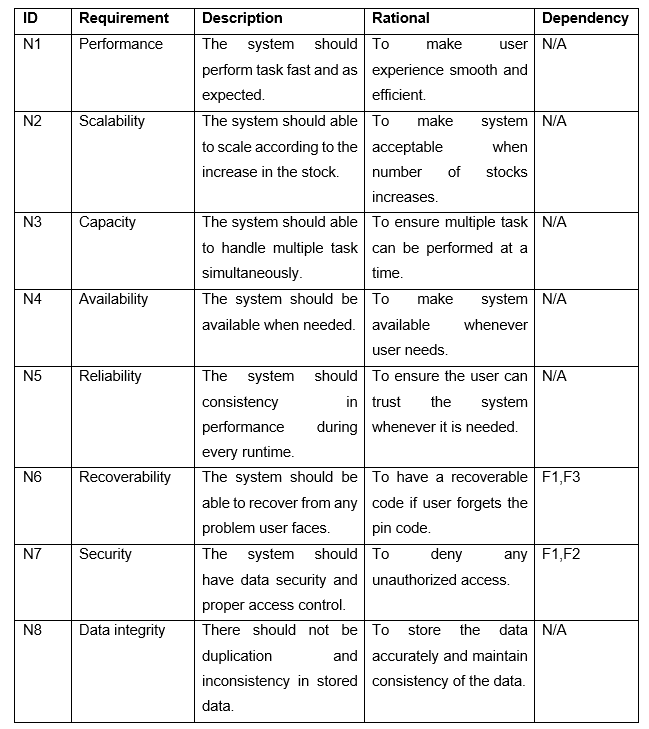


Table 2: Functional requirements

## 2.4.2: Non-functional Requirement

The non-functional requirements are important features that defines the quality of a system. It covers all the remaining requirements which are not covered by the functional requirements.



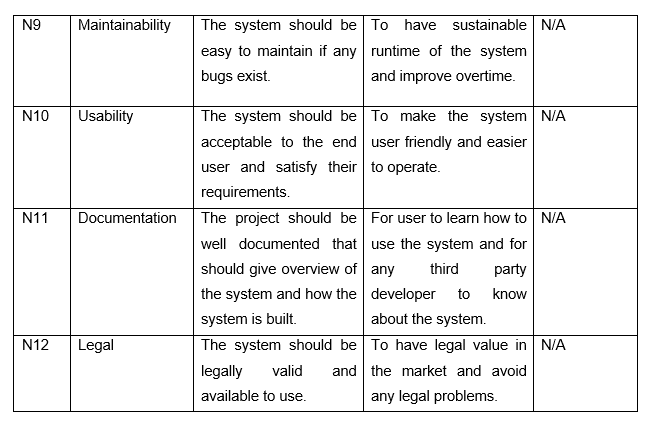
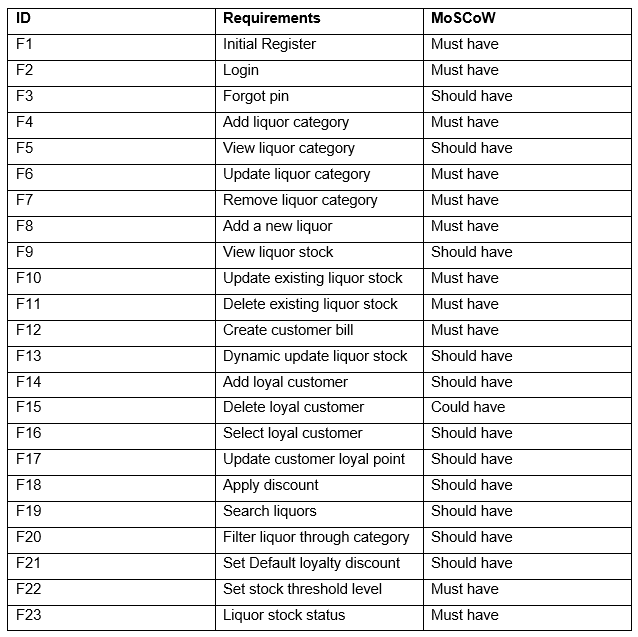


Table 3: Non-functional requirements

## 2.4.3: MOSCOW Prioritization

MoSCoW is a prioritization technique for assisting to understand and manage the priorities of the project. After gathering the requirements, they are divided based on their priorities to help stakeholders understand the importance of each requirement. The letters stands for

* Must Have
* Should Have
* Could Have
* Won’t Have this time



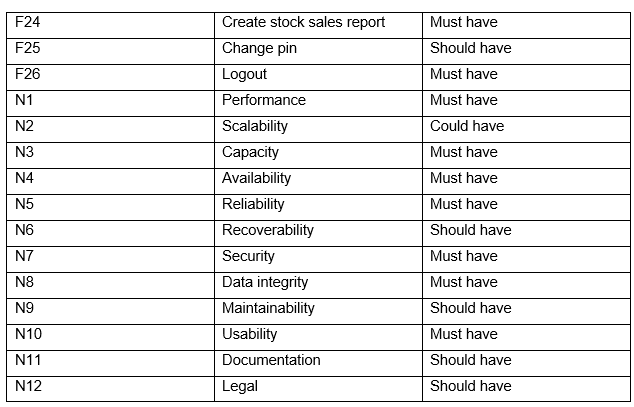


Table 4: MOSCOW prioritization

## 2.4.4: Software Requirement Specification

A Software Requirement Specification is a document, which specifies the software, hardware, functional, and network requirements of a system.

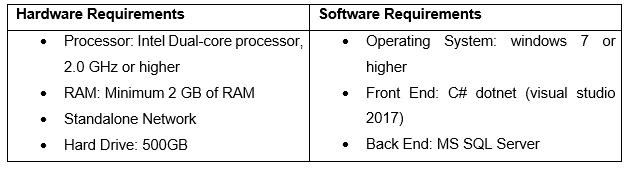


Table 5: Requirements specification

## 2.5: Use Case Diagram

The use case a diagram is a graphical representation used by an analyst to plan the boundaries of the business system that currently being analyzed, the expectations of the system and the potential user of the system.

It provides a high-level view of the solutions of the business problem and allows the analyst and designer to consider multiple implementation strategies. It helps to present the project scope to all the stakeholders. It allows people involved in the project to understand better about system processes and workflow. That’s why the use case diagram is made for this project analysis.

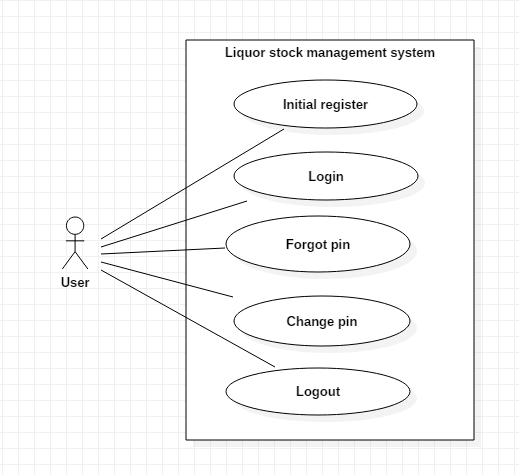


Figure 1: use case diagram 1

There is only a single actor for this system as the system is built for the standalone computer.

Scenario:

* User can register an account (pin) and recovery code is provided after registration.
* User can log in to the system using pin password.
* If the user forgets the password, they can recover the password using recovery code.
* User can change the pin password using the recovery code.
* User can log out from the system to stop access.

After logging in:

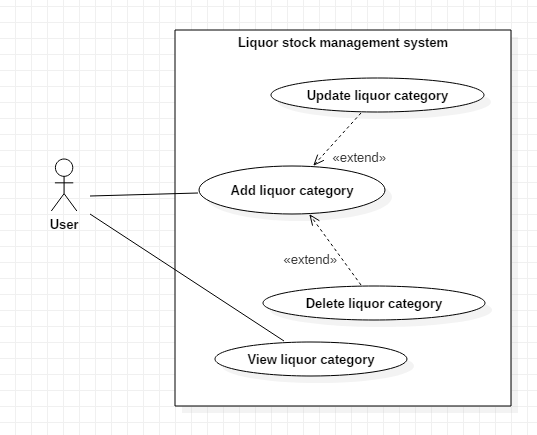


Figure 2: use case diagram 2

Scenario:

* User can add liquor category according to their wish.
* User can update or delete the existing liquor category.
* User can view all the existing liquor category.

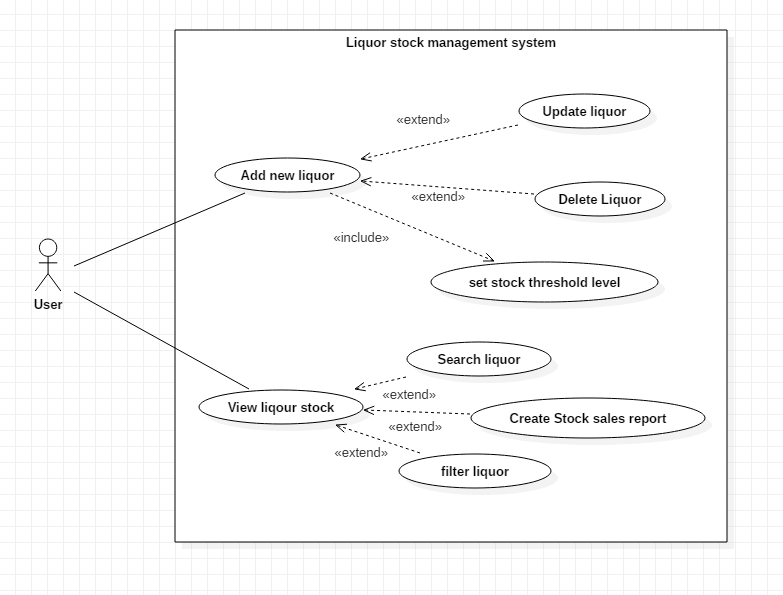


Figure 3: use case diagram 3

Scenario:

* User can add new liquor item on the database.
* User can update and delete the existing liquor stock.
* User can view all the existing liquor stock, filter using liquor category, search through liquor name and create stock sales report

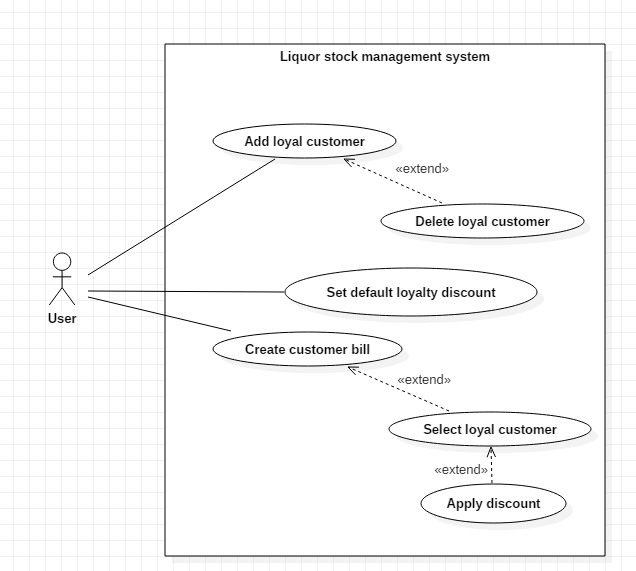


Figure 4: use case diagram 4

Scenario:

* User can add loyal customer for discount according to their points. The Loyal customer can obtain points when they buy any liquor and stacks with their previous purchase.
* User can set default loyalty discount to the loyal customer.
* User can delete the loyal customer.
* User can create customer bill and select loyal customer to update the loyal points and apply discount according to discount rate set by the user.

## 2.6: NLA and Initial Class Diagram

Liquor stock management system is a desktop based application system for stock management of a liquor store. The main aim of the system is to build an automated database system for managing the liquor stocks based on each item sold. The system allows to make work efficient and cut human errors.

The system should have a log-in system through a pin password. The system should also allow a user to change their log-in pin password.

The system must allow the user to add, edit and delete liquor items and its detail. The liquor item should contain its name, price, and quantity. The liquor has different categories so, the system should allow the user to view liquor stock according to the category. The user can also update and remove the liquor categories. The system should also allow the user to search liquor according to liquor name while viewing the stocks. The stock should have a threshold quantity level and should notify the user when the stock level is lower than the threshold.

The system should allow the user to create bills for the daily transaction of the store. The system should have a special feature for the customer through their emails to have a discount according to their earlier spending. The discount feature should be adjustable. The system should also create a report of sales.

From the scenario, the Natural Language Analysis (NLA) is performed and candidate classes, attributes and methods are picked up from the nouns, verbs and adjectives.

|  |  |  |
| --- | --- | --- |
| **Candidate classes** | **Candidate attributes** | **Candidate methods** |
| * User * Liquor * Category * Customer * Bill | * Pin * Liquor name * Price * Quantity * Category name * Threshold quantity * Discount * Email | * Login * Change * Add * Delete * Update * Search * View * Notify * Create report |

Table 6: NLA candidate table

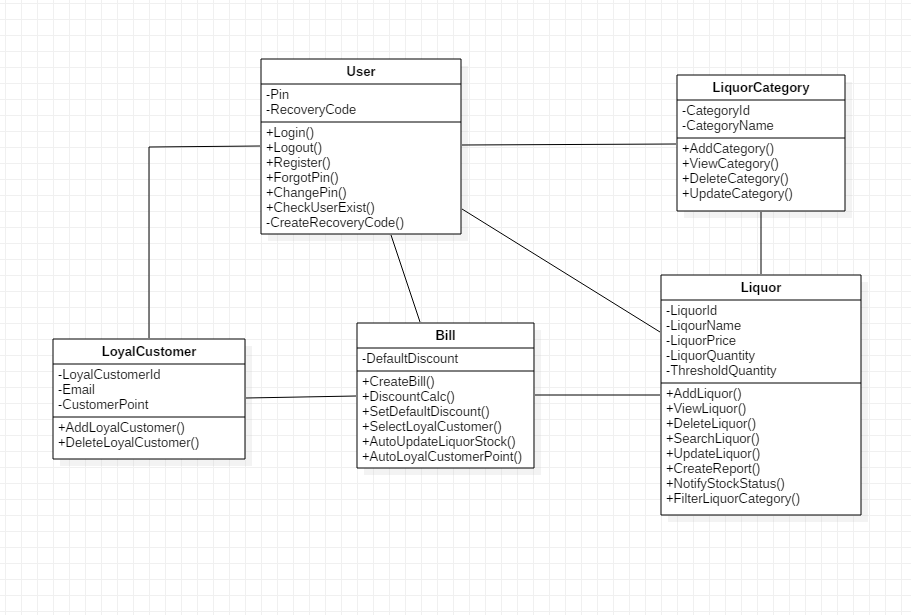


Figure 5: Initial Class Diagram

# Chapter 3: Design

After completion of the analysis, the design phase takes place. Basically, in design, the architecture of the project is built. In this phase, the requirements will be further broken down to estimate the needed effort and amount of resources. The design focuses on identifying specific workflows and designs for the application. It includes the design of an application, databases, network architecture, user interfaces, and system interfaces. This phase sets a standard to stick to it. It allows helping remove possible flaws.

Various tools are used to create design models. For this project, various structural and behavioral models are created to show the structure and workflow of the system. Database design is created to show the backend overview, architectural model made to show network structure for the system and user interface design prototype are made to show a frontend overview.

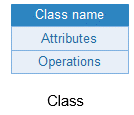
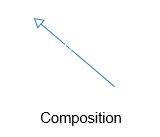
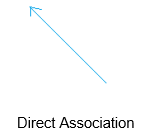
## 3.1 Structural Modeling

The structural modeling represents the static features of a system. This model represents the framework for the system. It never represents dynamic behavior of the system. The Class diagram is mostly used structural diagram.

## 3.1.1 Final Class Diagram

The class diagram is a static structure view of an application. It represents all the classes their properties and methods and the relationships between the objects. Class diagrams are widely used for object-oriented systems.

**Notation used:**

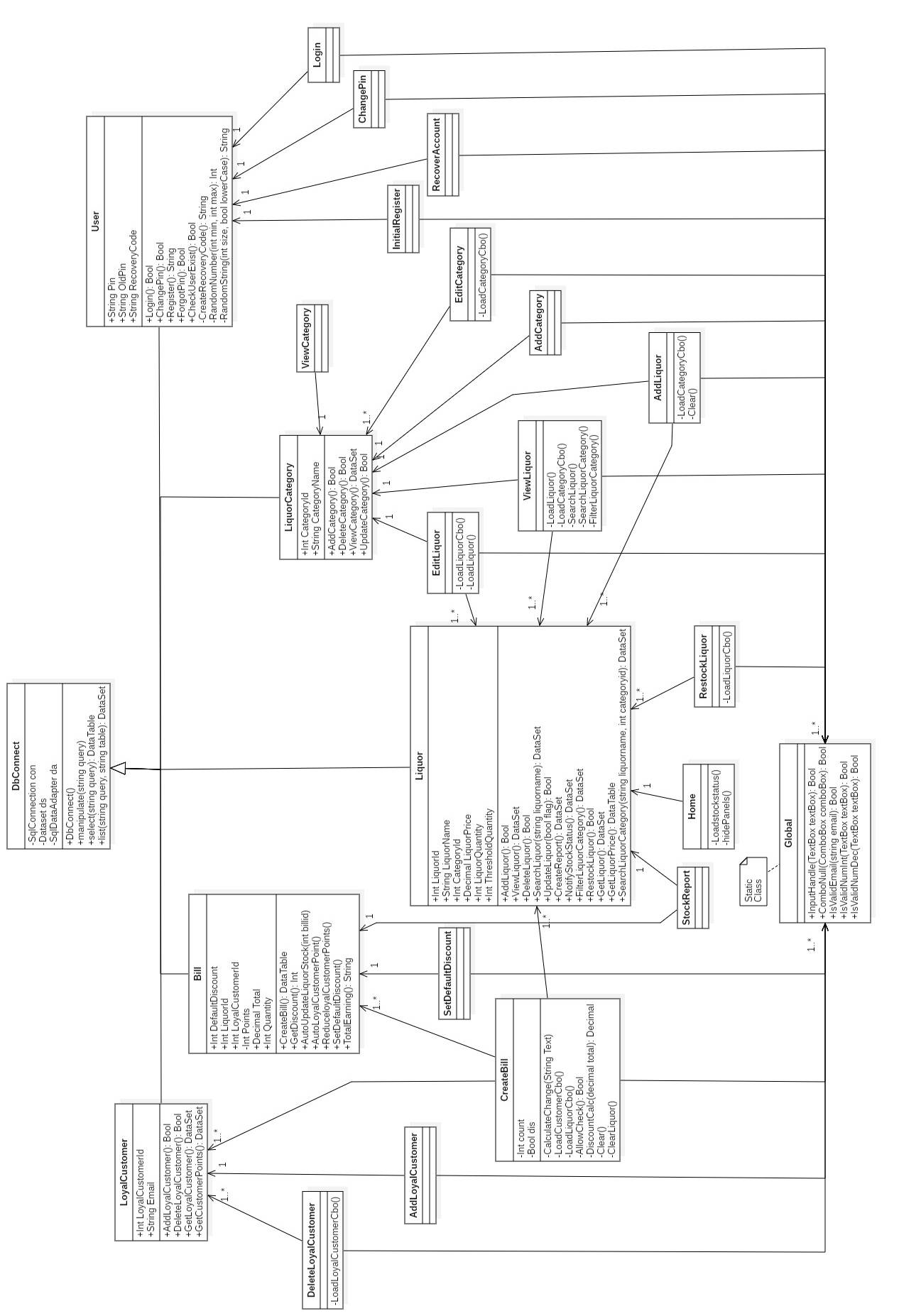


Figure 6: Final Class Diagram

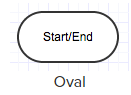
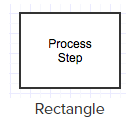
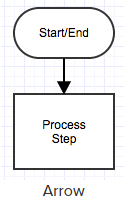
The above class diagram is for this project. The project will be developed using C# .net framework. The class diagram follows the Model View Controller design pattern. The diagram includes 5 controllers and 15 views.

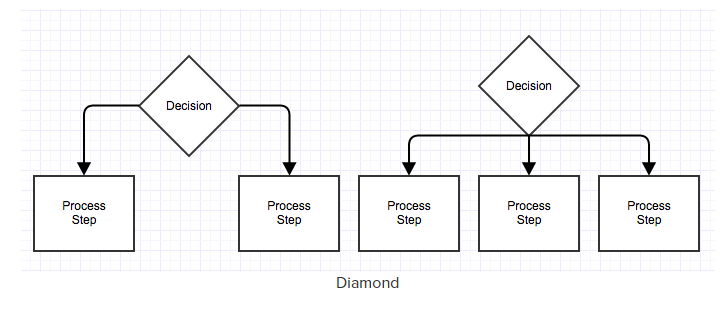
DbConnect acts as the model. It performs connection of database and all the query execution for the system. LoyalCustomer, Bill, User, LiquorCategory, Liquor classes act as the controller handling the business logics and other are views. The User classes perform all user verifications and registration tasks and other classes perform various functions according to the criteria. The windows forms act as the view for the system.

## 3.1.2 Flowchart

A flowchart is a graphical representation of the computer algorithm. It is a step by step approach to solve a task or show the process or workflow.

**Notation used:**



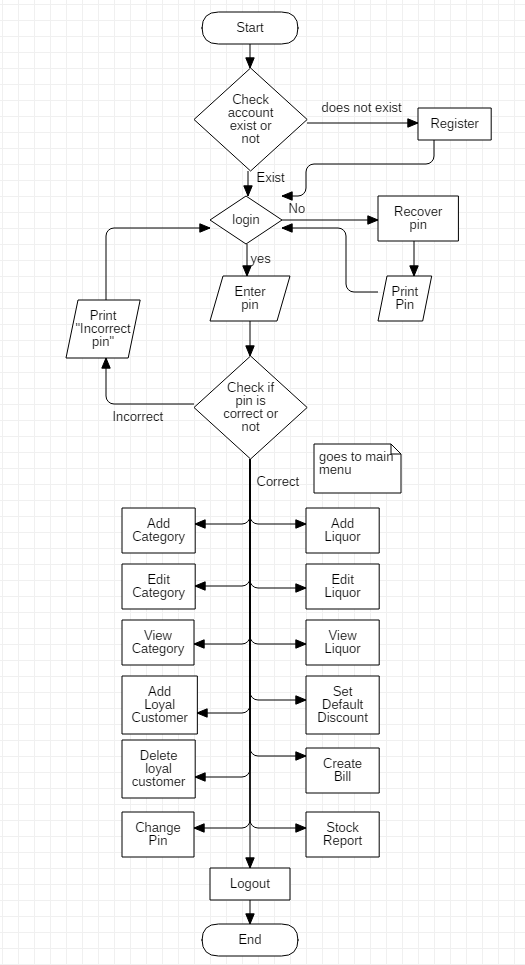


Figure 7: Flowchart

The flowchart shows the application’s initial registration and login process. If an account exists then it will straight proceed to open Login form. User can either login using an authenticated registered pin or recover pin using recovery code that is provided after registration. After logging in user can perform all the tasks as listed in the rectangle box. The flowchart ends when user logs out from the system.

## 3.2 Behavior Modeling

Behavior model represents the interaction among the structural diagrams in the system. It shows the dynamic sequence flow of the system.

## 3.2.1 Activity Diagram

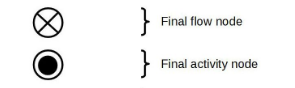
An activity diagram represents the flow or sequences of activities in a process. It includes sequential, parallel activities and decision that are made in the process. It represents the dynamic workflow behavior of a system.

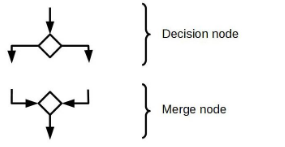
**Notation Used:**

****

****

****

****

****

****

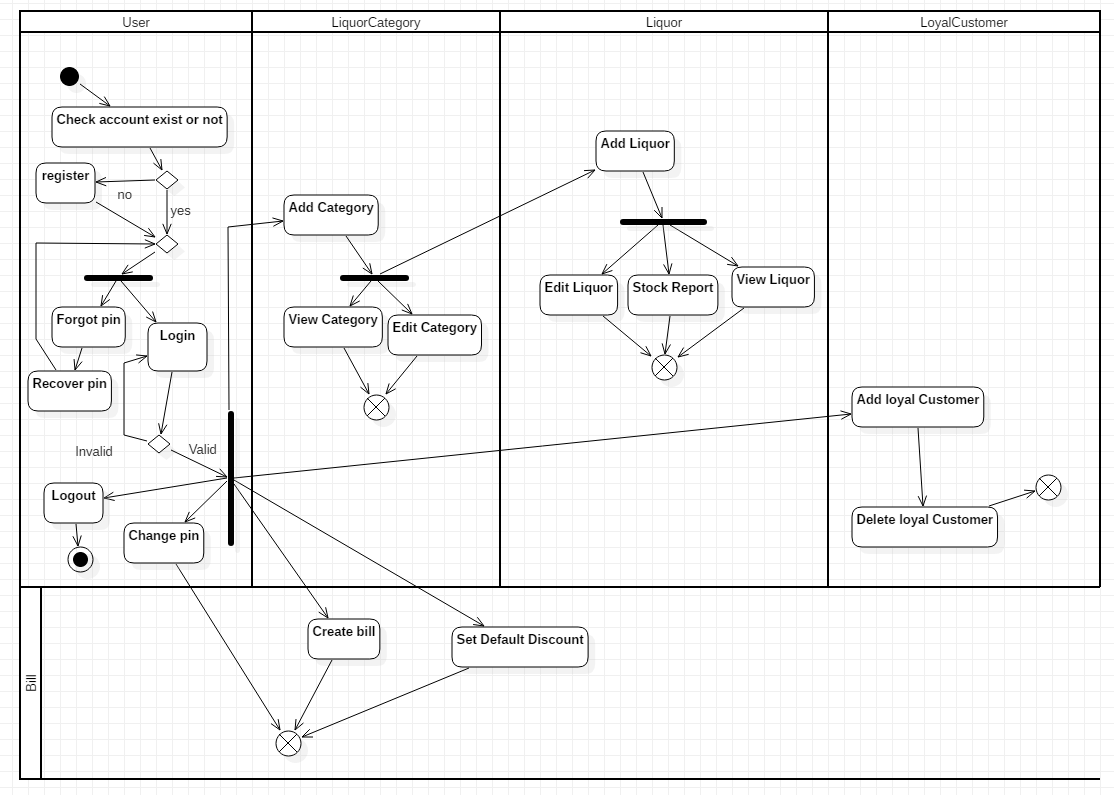


Figure 8: System Overview Activity Diagram

Above activity diagram is the full system overview of how system registration, login and its function occurs. The system checks if there is an account or not. If an account exists it will not allow to register again. The user is able to login after registration of the system. The user gets recovery code after registration which can be used to recover pin when the user forgets the pin code. After logging in, the user can add liquor to stock if the liquor category exists. Liquor category is essential for adding liquor. User can create a bill, set default discount percent per 100 points, add and delete loyal customers, change the pin and log out to end the flow.

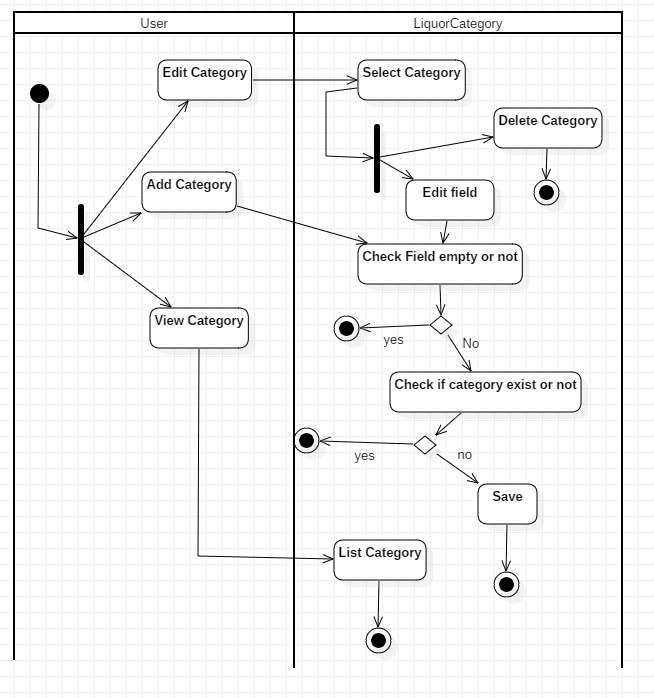


Figure 9: Liquor Category Activity Diagram

This activity diagram shows how the user add, edit, delete and view the category. While adding new category or editing existing category it checks if null values are entered or not. It also does not allow to enter the same category twice.

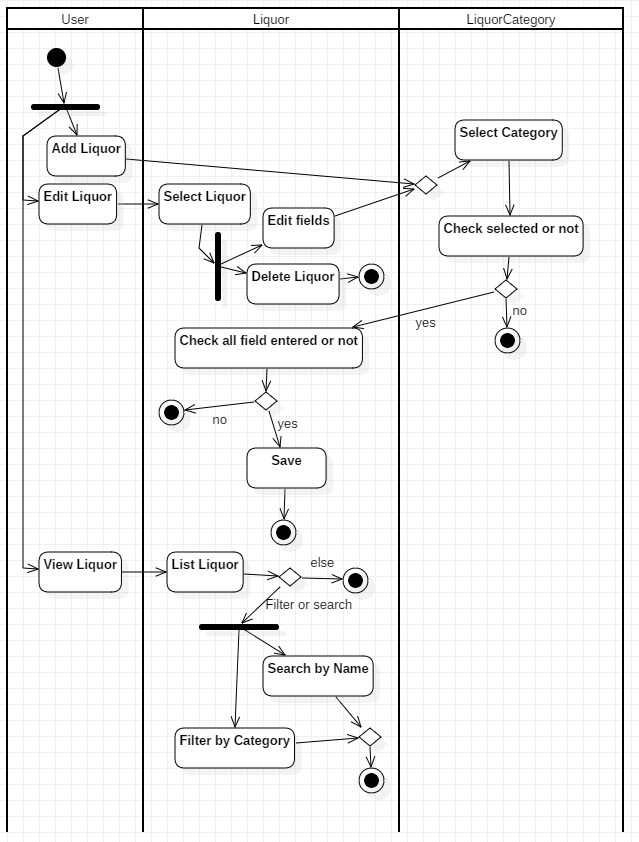


Figure 10: Liquor Activity Diagram

This activity diagram shows how the user add, edit, delete and view the liquors. Similar to the liquor category it does not allow any null value while adding or editing the liquors. User can search the liquor by its name and filter the liquor according to its category while viewing the liquors.

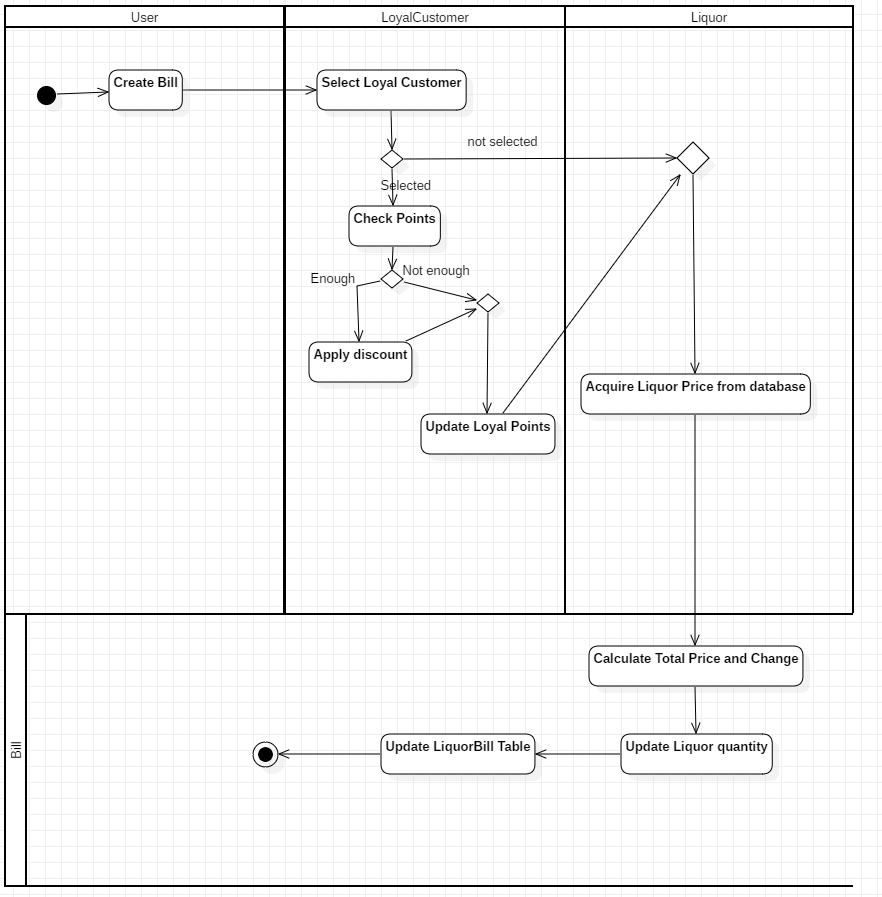


Figure 11: Bill Activity Diagram

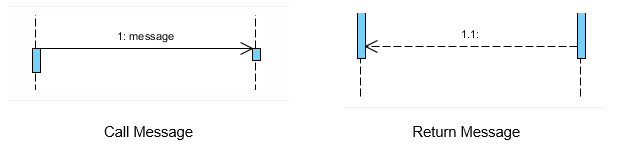
This activity diagram shows how the discount is applied for a loyal customer when they have enough points while creating the bill. After calculating the total price the bill updates the liquor stock quantity and stores bill information separately.

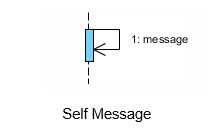
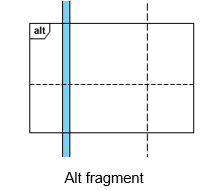
## 3.2.2 Sequence Diagram

A sequence diagram is used to show the interaction between objects in a sequential order in which it takes place. It describes how and in what order the objects in a system functions. It allows to show the logic of relationships between the objects.

**Notation Used:**

****

****

**** ****

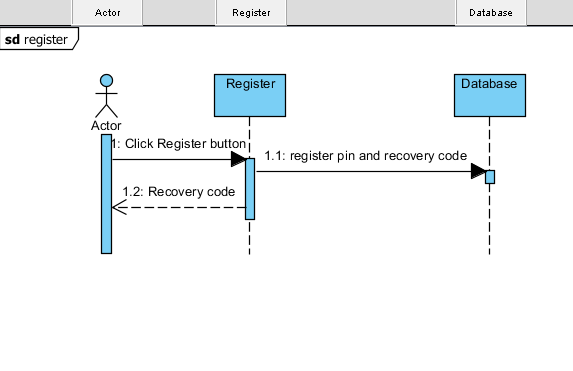


Figure 12: Initial Register Sequence Diagram

This Sequence diagram shows the initial registration process of the application. After entering the pin the register sends to store in database as well as provides recovery code to the actor.

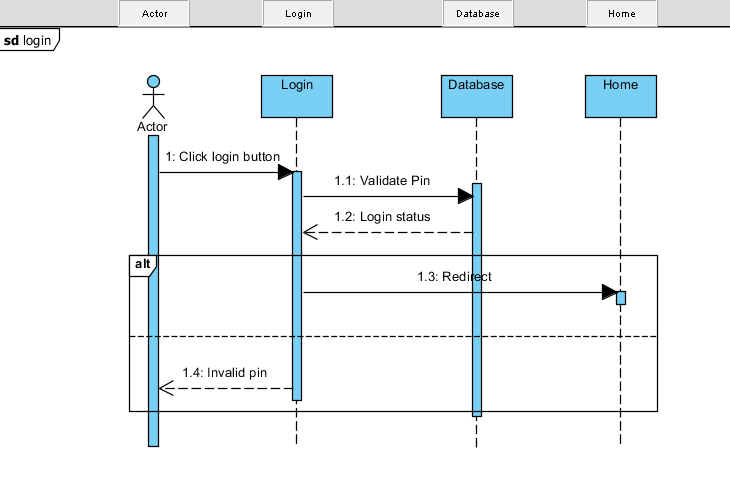


Figure 13: Login Sequence Diagram

This sequence diagram shows the login process takes place in sequential order. Validation takes place in the database and if it is authentic then the actor is redirected to home else shows actor the pin is invalid.

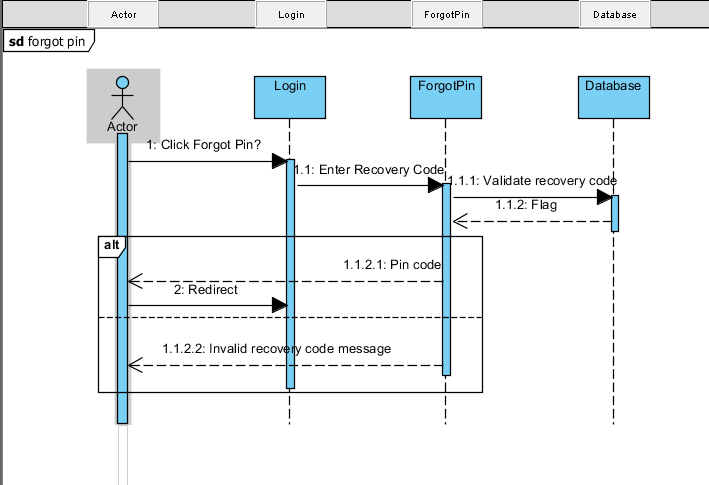


Figure 14: Forgot Pin Sequence Diagram

This sequence diagram shows the pin recovery process. After selecting forgot pin and entering recovery code through login, ForgotPin validates the recovery code in the database. If the code is valid then it returns pin code else it throws invalid recovery code message.

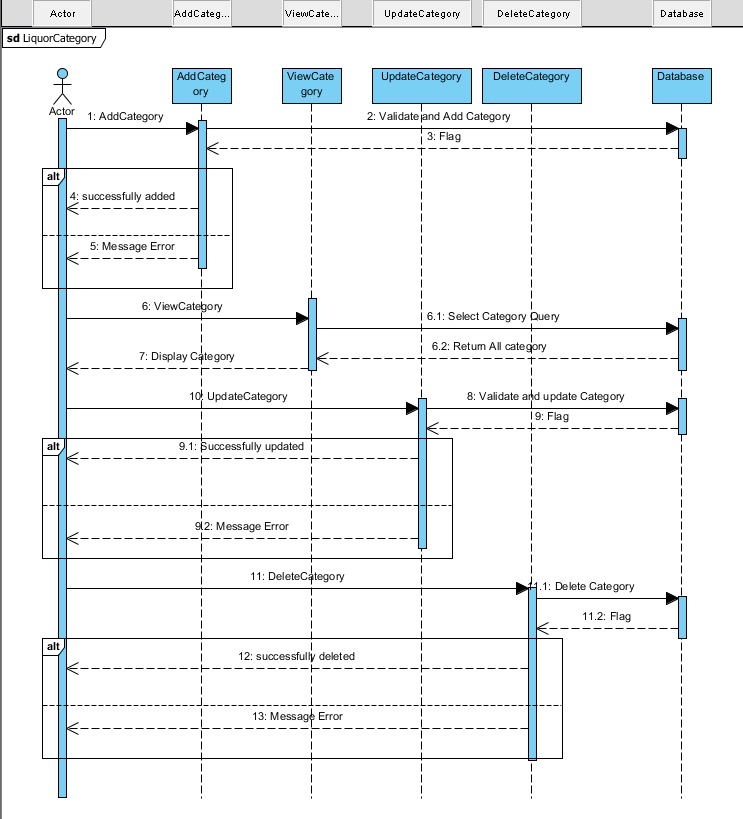


Figure 15: Liquor Category Sequence Diagram

This sequence diagram shows add, update, delete, and view process of liquor category. While adding, updating the category, the database validates and creates a flag telling whether the process is successful or unsuccessful. While viewing category the query is executed in DataBase and returns a list of the liquor category to display. While deleting if there are any exception it throws an error message.

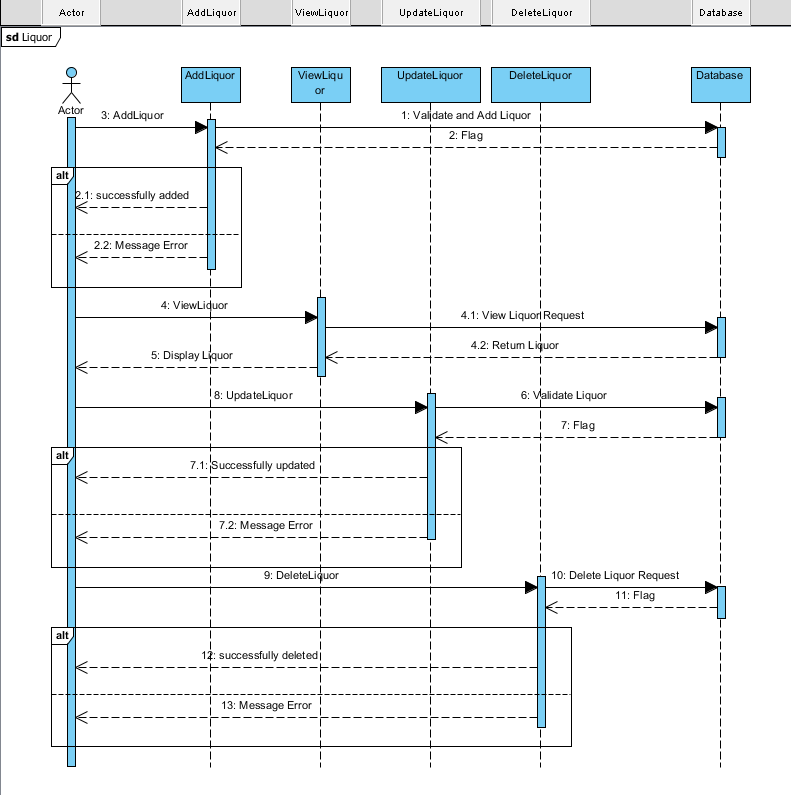


Figure 16: Liquor Sequence Diagram

This sequence diagram shows add, update, delete, and view process of the liquor. While adding, updating the liquor, the database validates and creates a flag telling whether the process is successful or unsuccessful. While viewing liquor the query is executed in DataBase and returns a list of the liquor to display. While deleting if there are any exception it throws an error message.

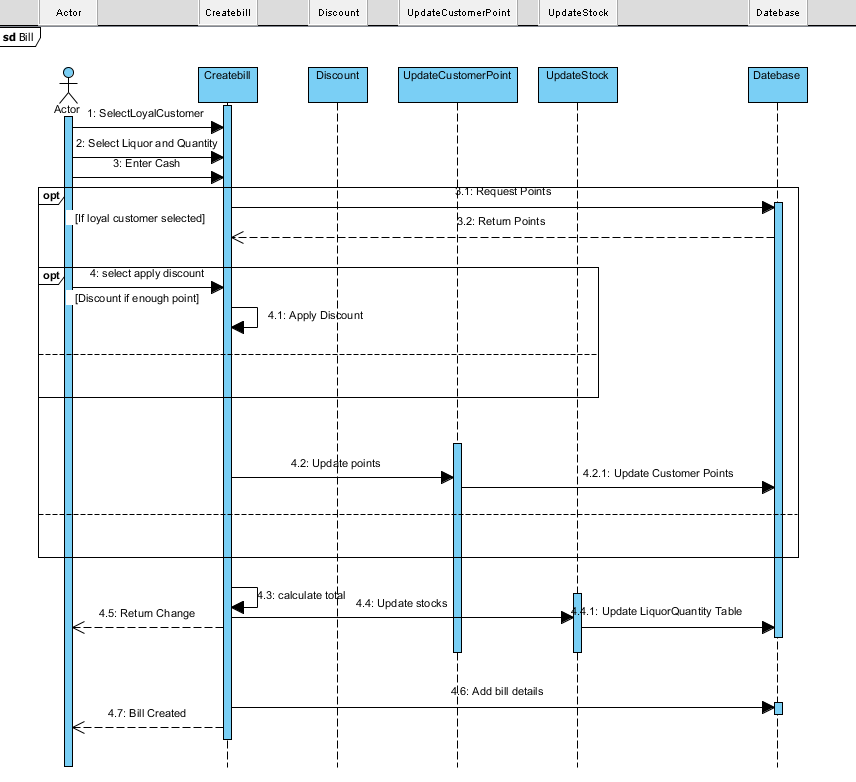


Figure 17: Create Bill Sequence Diagram

This sequence diagram shows the process of creating a bill. It allows to enter liquor, its quantity and paid cash. It shows if the loyal customer is selected then their points are updated. If the loyal customer has enough points they can apply the discount. The total amount is calculated and the liquor quantity stocks are updated. It returns change cash and adds bill details in the database.

## 3.3 Database Modeling

Database modeling is the process of designing a database model that helps while developing and implementing the system. It shows the relationship between entities, normalized table, metadata, and fields.

## 3.3.1 Data Dictionary

The Data dictionary is a set of files that contains metadata of a database. In the following data dictionary data type, length, null or not, key and constraints are included.

Table: User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| Pin | varchar | 25 | No | Primary Key | Pk\_Pin |
| RecoveryCode | varchar | 25 | No | - | - |

Table: LiquorCategory

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| CategoryId | int | - | No | Primary Key | Pk\_CategoryId |
| CategoryName | varchar | 255 | No | - | - |

Table: Liquor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| LiquorId | int | - | No | Primary Key | Pk\_LiquorId |
| LiquorName | varchar | 255 | No | - | - |
| LiquorPrice | decimal | - | No | - | - |
| CategoryId | int | - | No | Foreign Key | Fk\_CategoryId |

Table: LiquorQuantity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| LiquorId | int | - | No | Foreign Key | Fk\_LiquorId |
| Quantity | int | - | No | - | - |
| Threshold | int | - | No | - | - |

Table: Bill

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| BillId | int | - | No | Primary Key | Pk\_BillId |
| TotalAmount | decimal | - | No | - | - |
| CustomerId | int | - | Yes | - | Fk\_CustomerId |

Table: LoyalCustomer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Null** | **Key** | **Constraints** |
| LoyalCustomerId | int | - | No | Primary Key | Pk\_LoyalCustomerId |
| Email | varchar | 255 | No | - | Uk\_Email |
| Points | int | - | No | - | - |

Table: LiquorBill

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Type** | **Length** | **Null** | **Key** | **Constraints** |
| BillId | int | - | No | Foreign Key | Fk\_BillId |
| LiquorId | int | - | No | Foreign Key | Fk\_LiquorId2 |
| Quantity | int | - | No | - | - |

Table 7: Data dictionary

## 3.3.2 ER diagram

An ER diagram is the structural diagram for use in database design. It shows the major entities, attributes and their relationship among them.

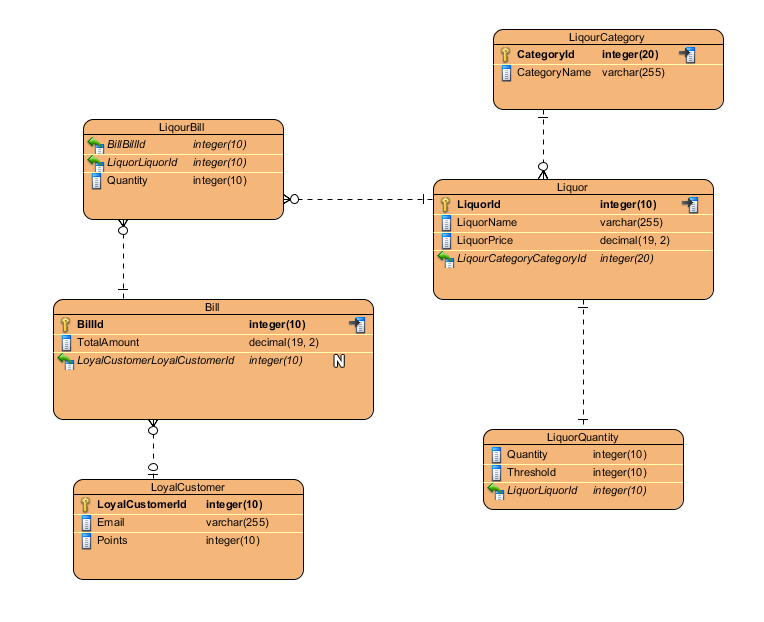


Figure 18: Entity-Relationship Diagram

The Above ER diagram is built based upon the data dictionary in 3.3.1. In this diagram, the entities are shown with their attributes and their respected relationships with each other.

There are total 8 tables for this project. The User and discount entity is not included in the above ER diagram because it is an independent entity as the system is standalone application thus does not requires any relationship with other tables. There is discount table with single discount column to store the default discount to calculate bill.

## 3.4: Architectural Model

The system is being built for the standalone computer. Basically, it is a computer system that can run a local application on its own without the need of LAN or WAN connection.

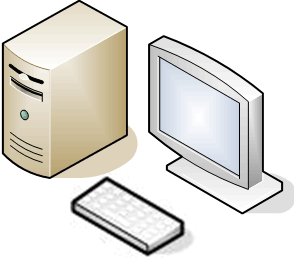


Figure 19: Standalone computer

The system that is being built is for a small/ medium size liquor store where there traditional hardcopy way of storing stock information and billing system. The main objective is to make an automated stock and billing system for a single store. The standalone architecture main benefit is it does not require any external connection for the application to run.

The database is store in the singular computer where the application is installed. This allows access of database virtually faster than any other architecture i.e. client-server, peer to peer. The standalone application is never kept online, and remote access of data is practically impossible. This allows for high security of the stored data.

The computer resources are fully available for the application whereas one user might waste resources in the client-server architecture. It allows monitoring and accessing application to be more controlled. Whenever the application crashes it can be reinstalled without any problem of accessing the network.

The standalone computer is the best to use for having the support of more hardware such as barcode reader, printer, biometric devices, etc. The stationary printer is still relevant and can be better according to preference. The feature for printing bills, using the barcode to scan liquor can be implemented.

## 3.5: Prototype UI Design

The graphical representation of an application is called the user interface. The prototype UI design is the early model of the application’s user interface. Here is the prototype The UI design of this project.

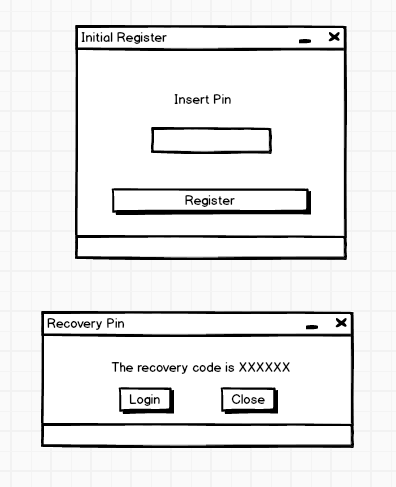


Figure 20: initial register

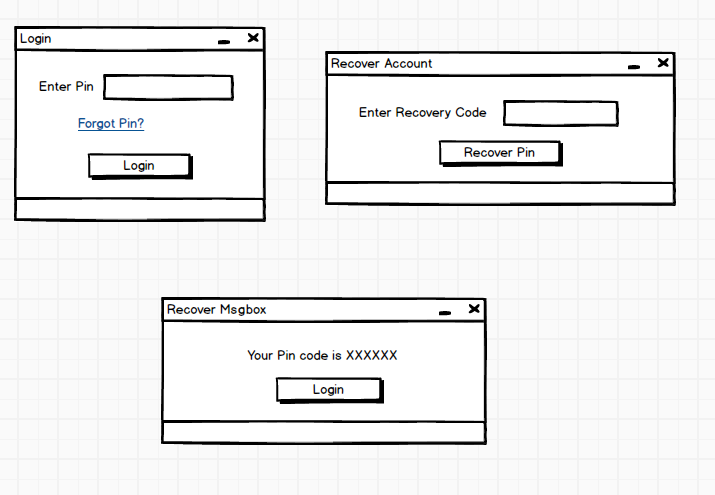


Figure 21: Login and recover account form

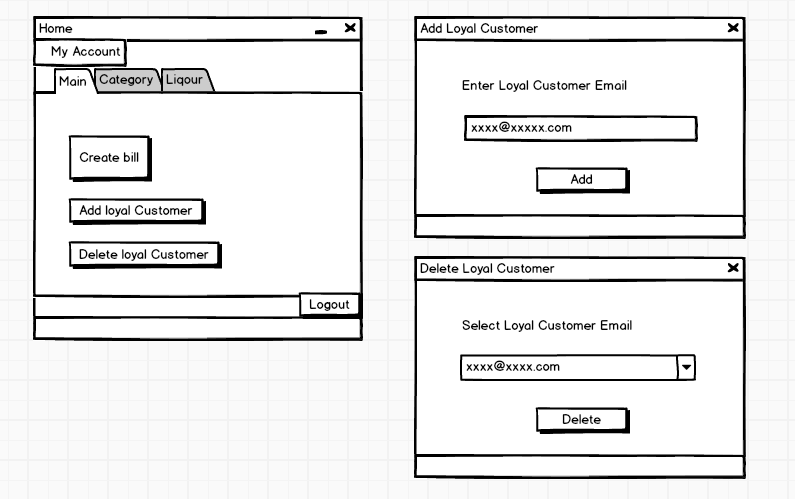


Figure 22: Login and recover account form

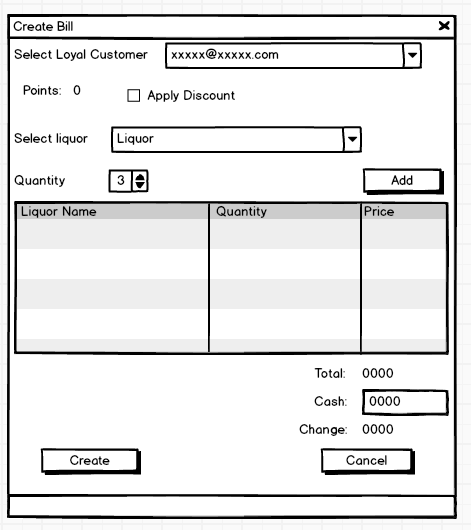


Figure 23: Create bill

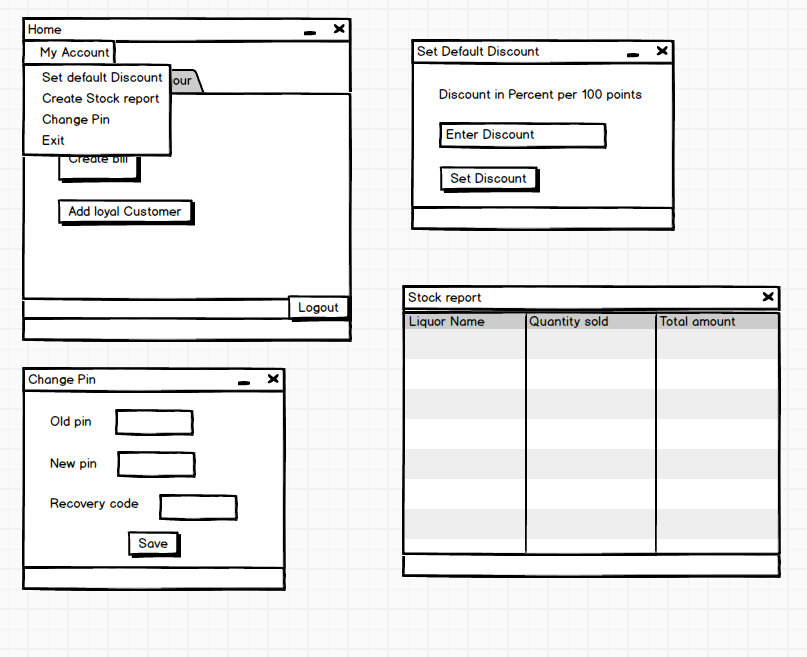


Figure 24: Menu bar and its forms

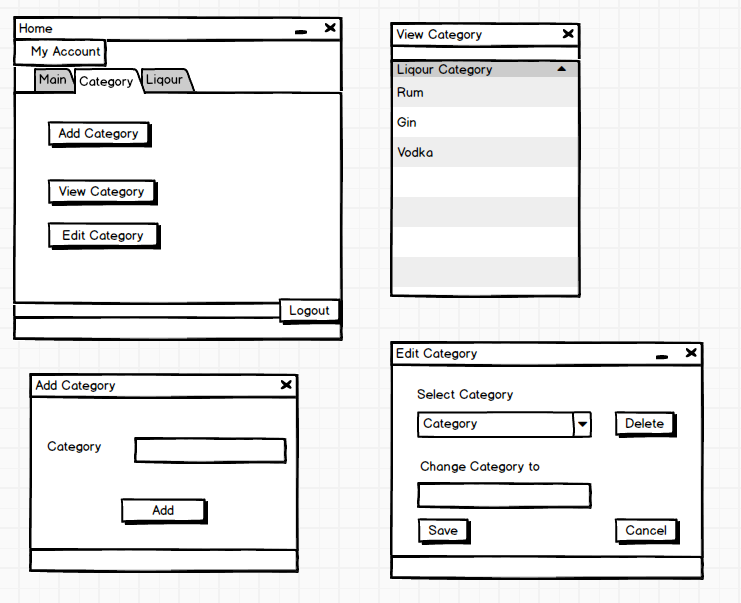


Figure 25: Liquor Category forms

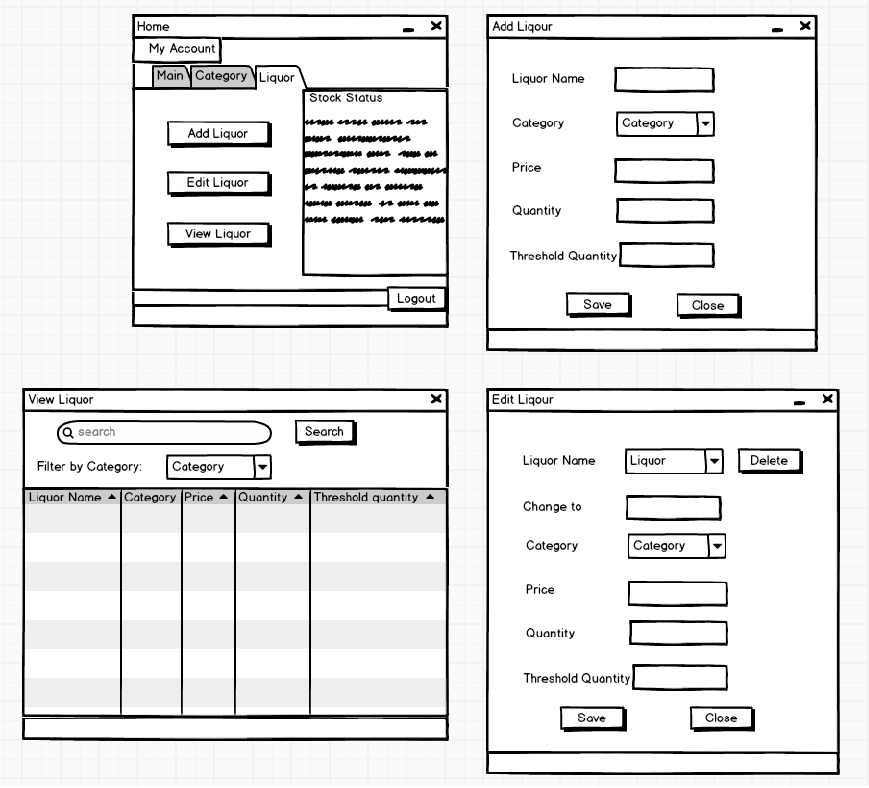


Figure 26: Liquor forms

# Chapter 4: Implementation

The implementation is the third phase of the software development life cycle. In this stage, the team builds the actual product using the base created through analysis and design. The team decides appropriate the programming tools and starts to write the source code for the project. All components of software are implemented in this phase.

## 4.1 Programming Language

For this project, I have selected C# as the programming language .Net framework. C# is a type-safe object-oriented language that allows developers to build a variety of robust and secure application that runs on the .NET framework. It allows to create Windows client applications, client-server applications, XML web services, distributed components, database applications and many other types of applications. Visual C# provides an advanced code editor, integrated debugger, suitable user interface designers and many other tools to develop an application on C# .Net easier.

This project is windows form .Net application.

Microsoft SQL server has been used for the database. SQL server is a relational database management system that supports a wide variety of transaction processing, analytics applications and business intelligence in corporate IT environment.

## 4.2 Design pattern and framework

I have used ADO.NET model of .NET framework for this application. This model is mainly used for establishing a connection between application and data sources i.e. SQL server. It acts as the bridge between the backend controls and the frontend database.

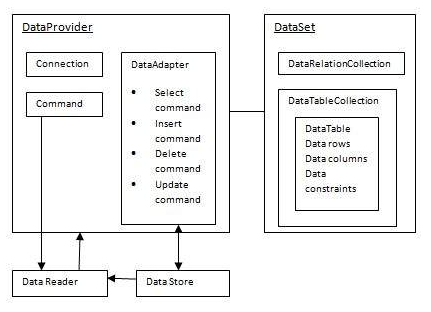


Figure 27: ADO.NET architecture

I have used ADO.NET to implement (Model View Controller) MVC design pattern. I have used database connection layer class as Model for manipulating data in the database. The business logic layer class as Controller where all business logic and SQL script are coded. The windows forms act as View. The input data are validated and sent to the controller.

## 4.3 Tool Used

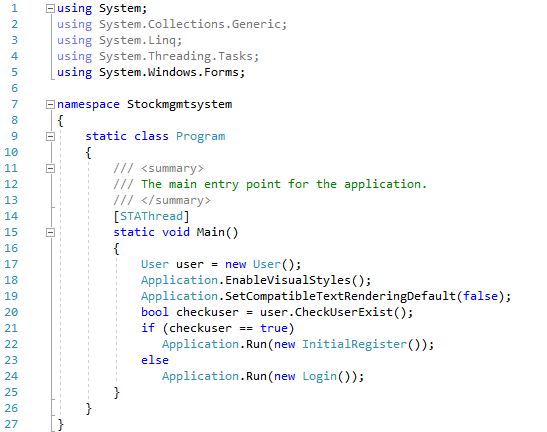
Operating System used: **Windows 10 v1903**

Tools used for implementing the system:

* **Microsoft Visual studio 2017** as a source code editor, debugging and unit testing tool.
* **Microsoft Server SQL management Studio 17** to create and manipulate the database.

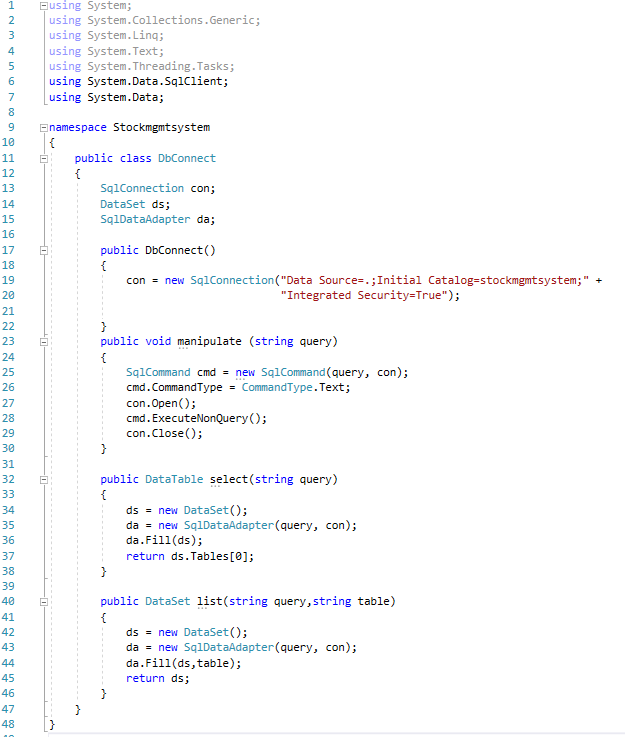
## 4.4 Coding Screenshots

**Main Class Program**

****

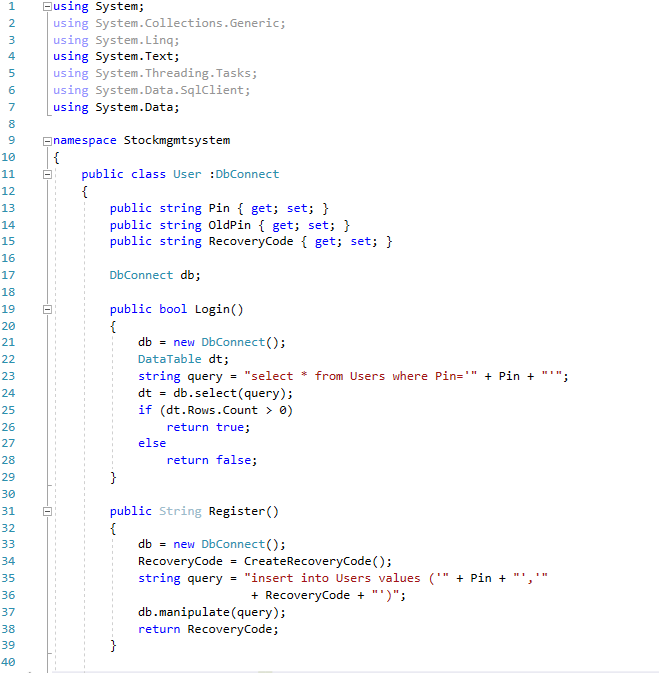
**Model**

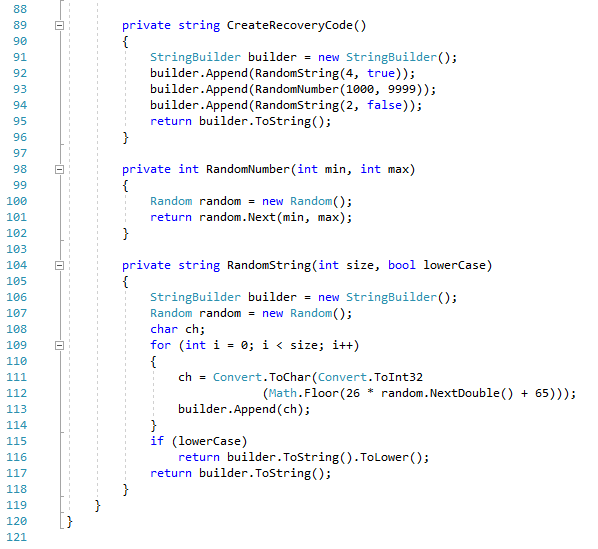
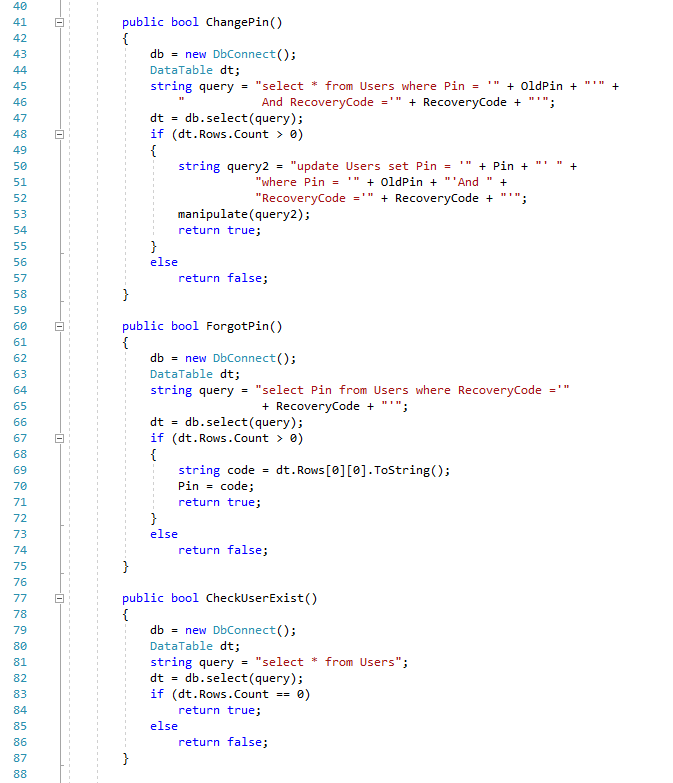
**Class DbConnect**



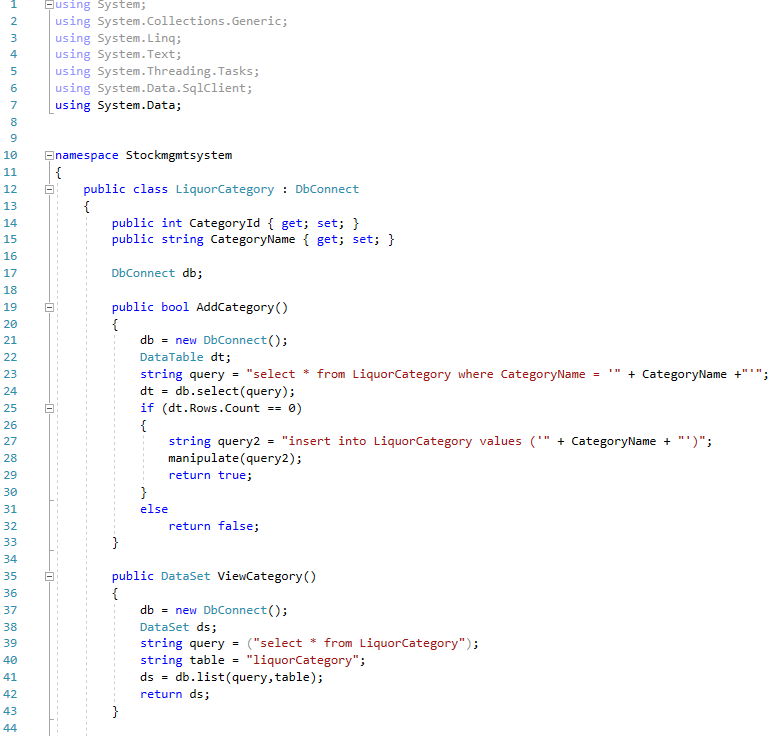
**Controller**

**Class User**

****

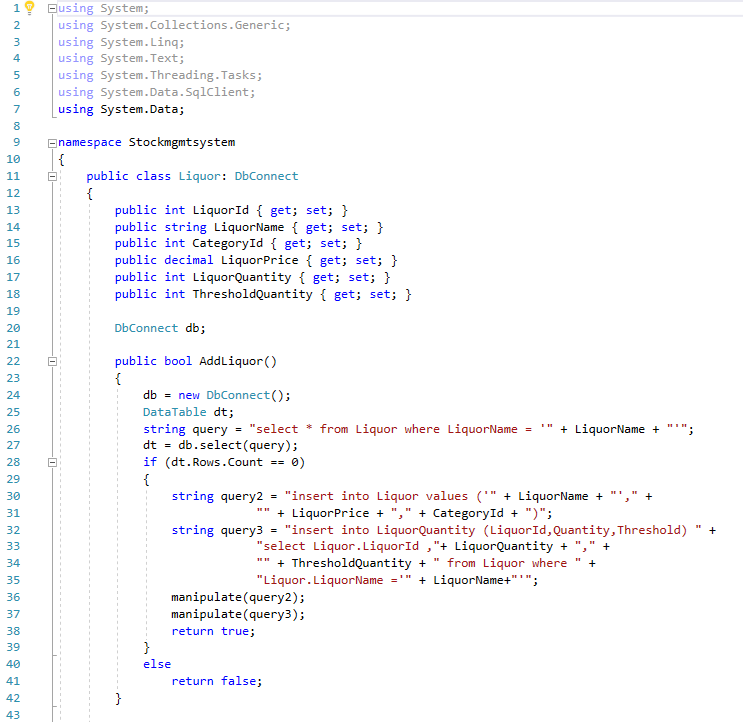
****

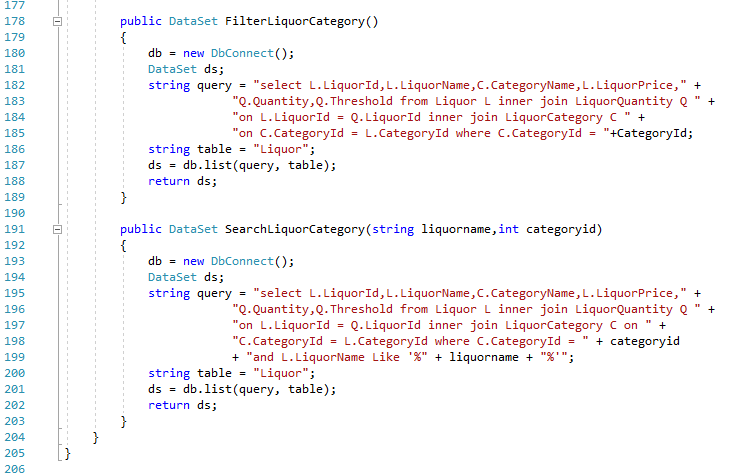
**Class LiquorCategory**

****

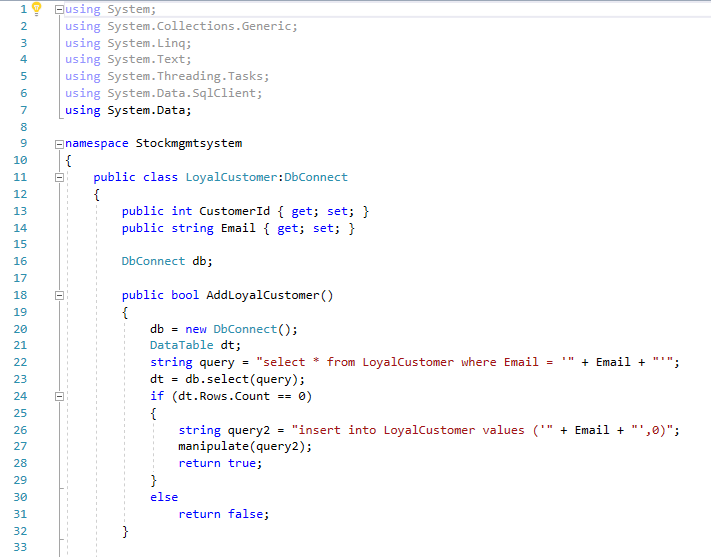
****

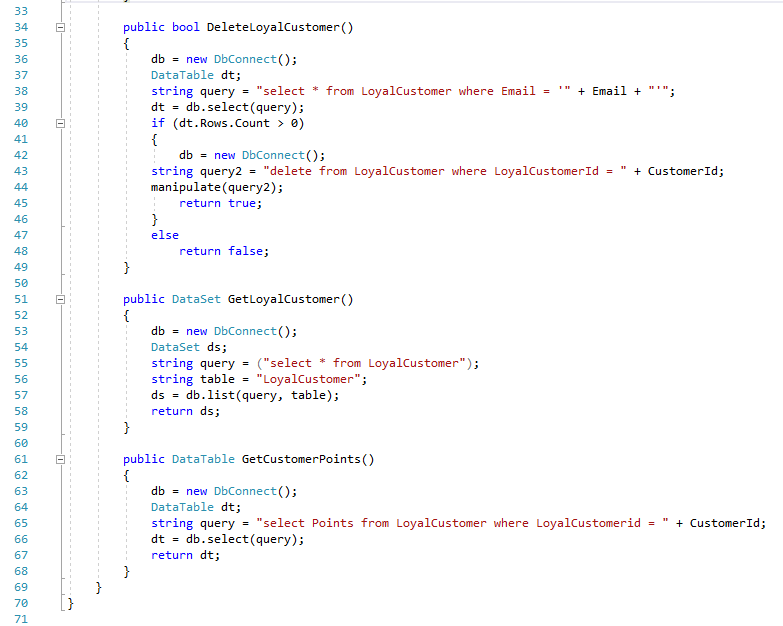
**Class Liquor**

****

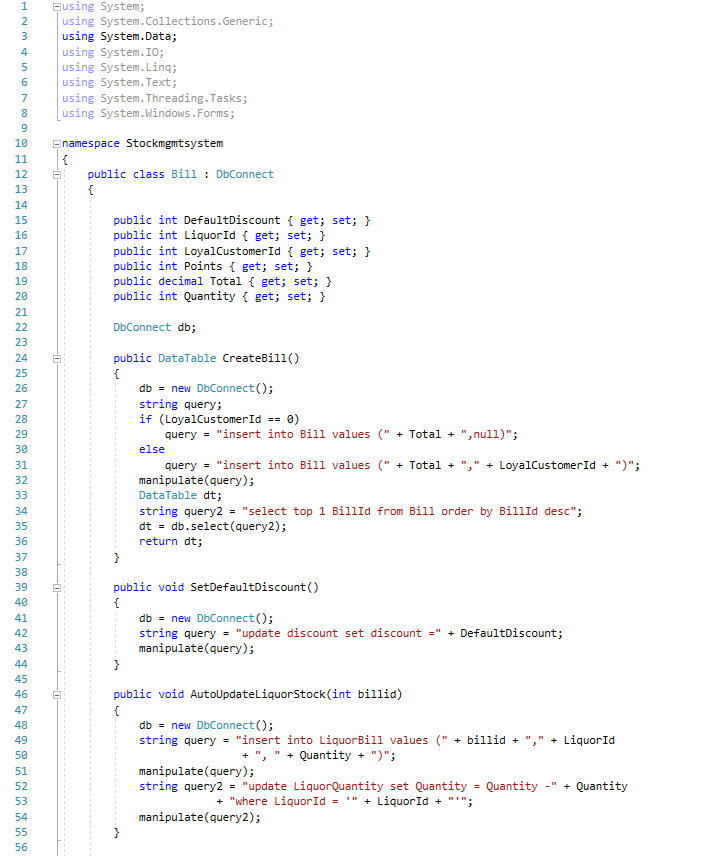
****

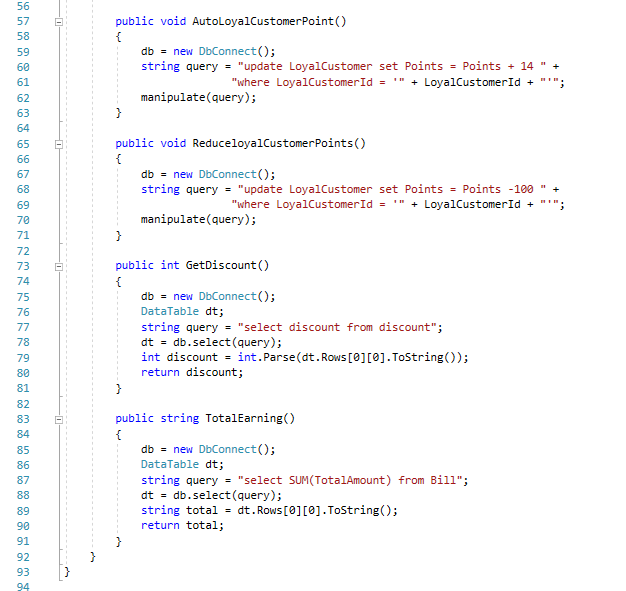
**Class LoyalCustomer**

****

****

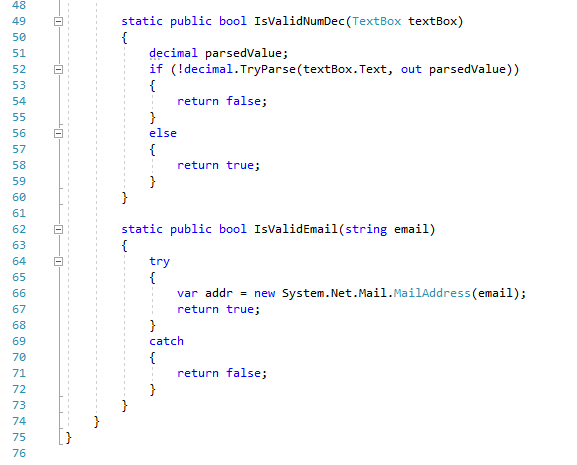
**Class Bill**

****

****

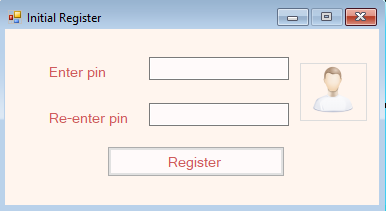
**Static Class Global**

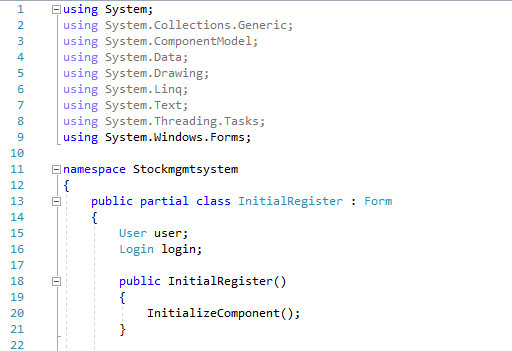
****

****

**View**

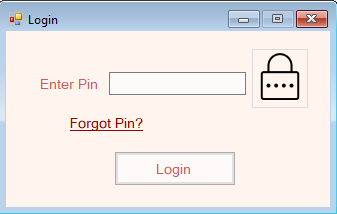
**Class InitialRegister**

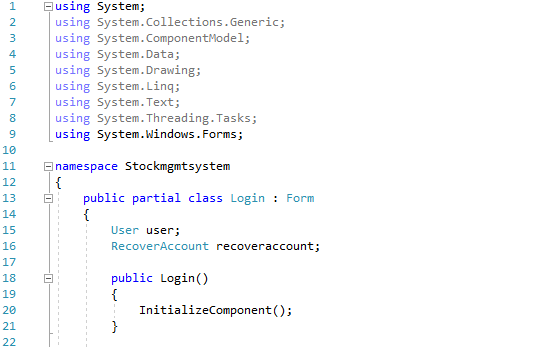
****

****

****

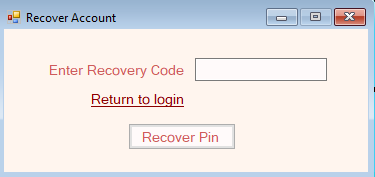
**Class Login**

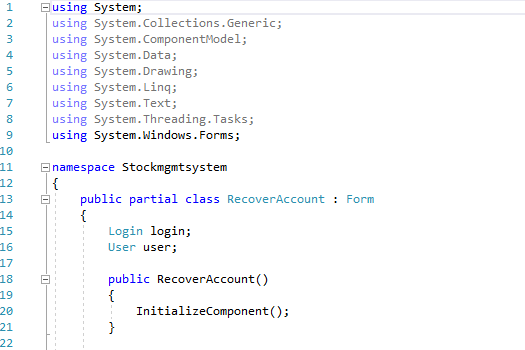
****

****

****

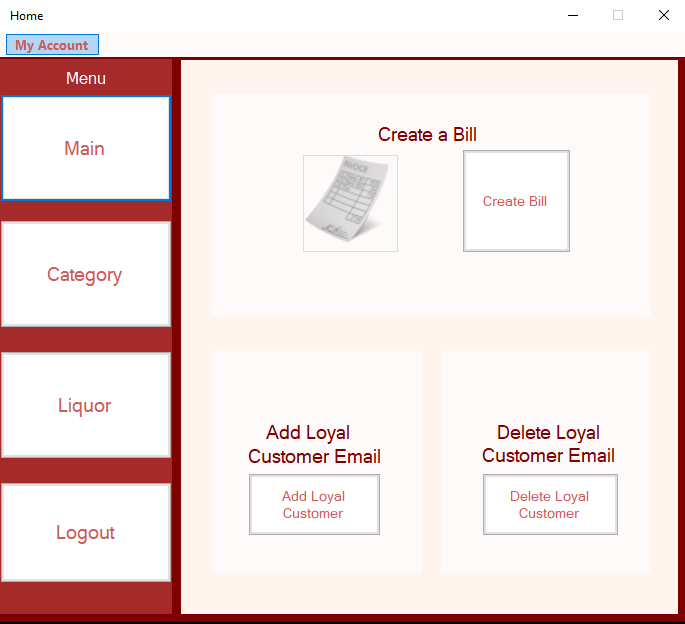
**Class RecoverAccount**

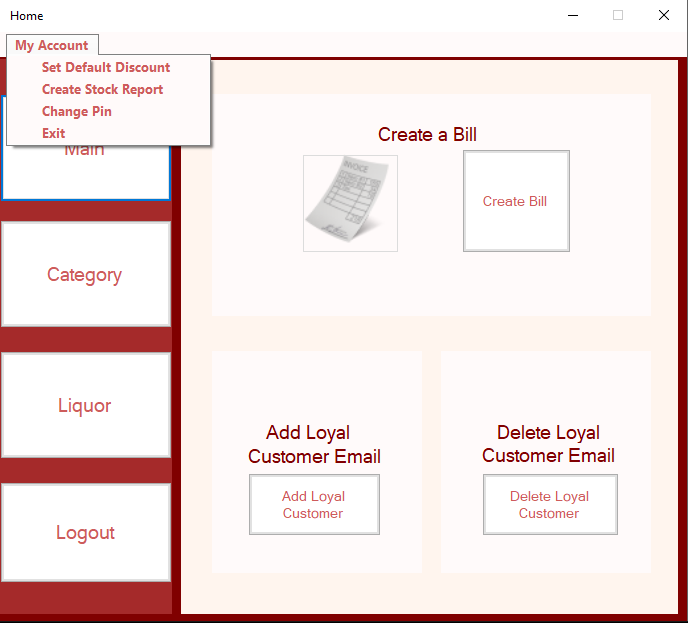
****

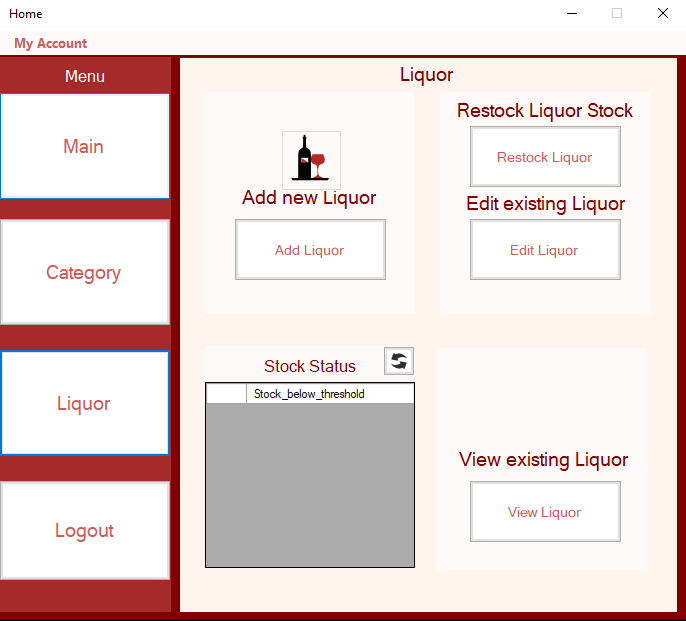
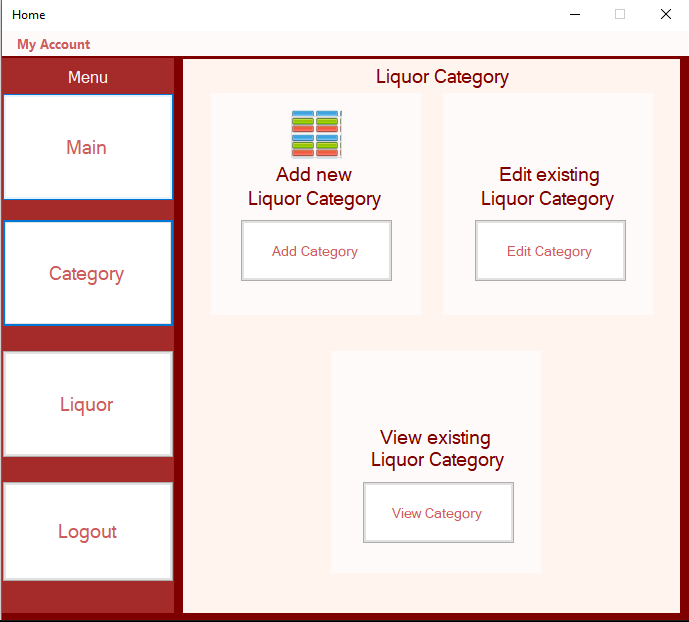
****

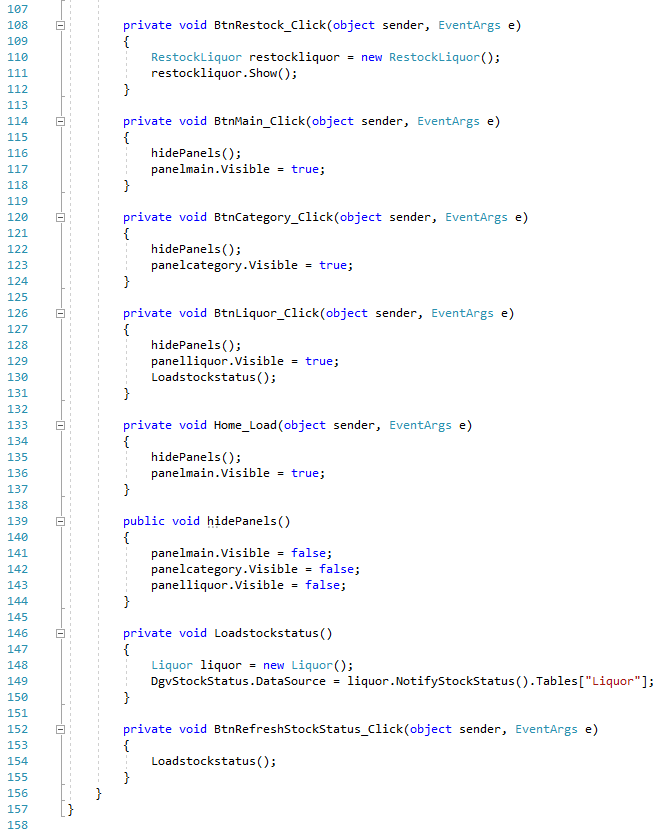
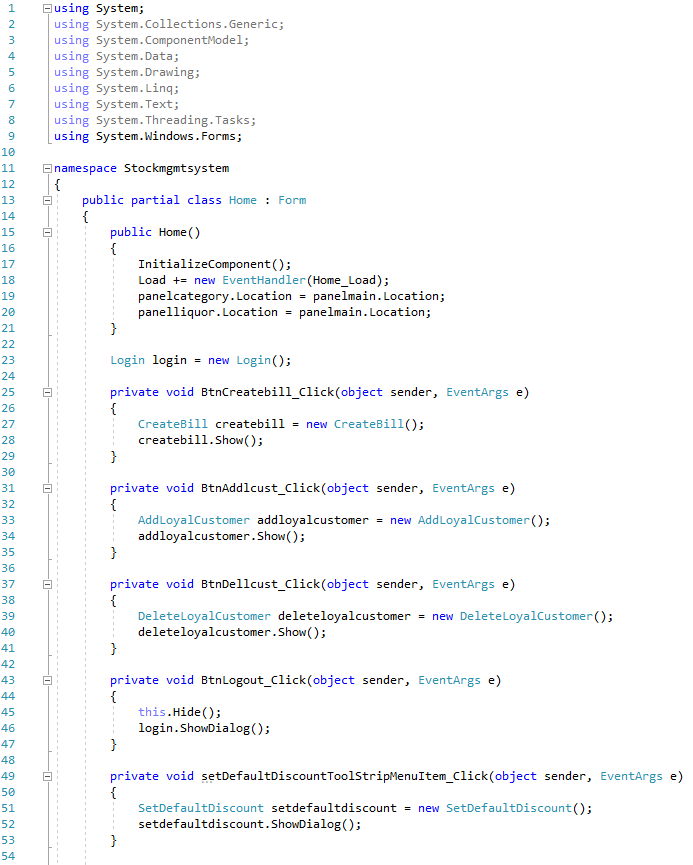
****

**Class Home**

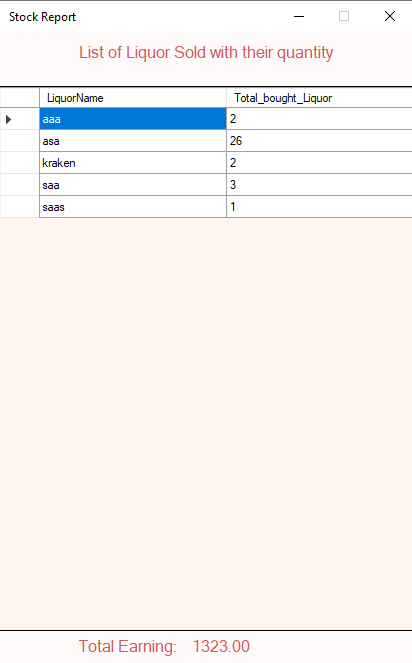






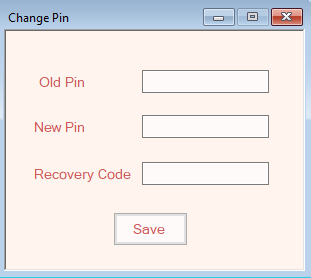
****

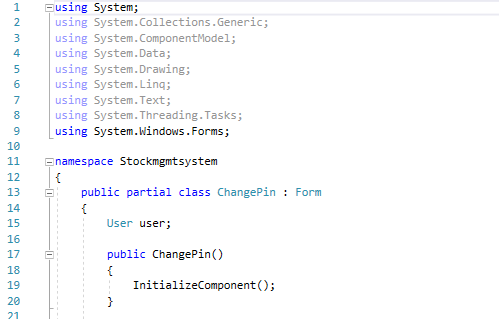
**Class StockReport**

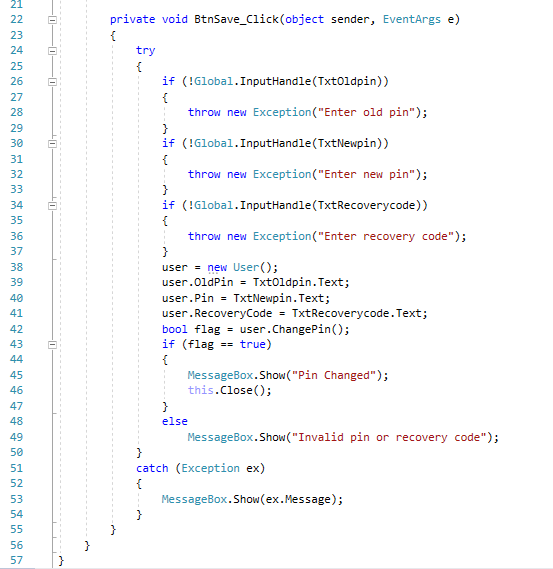
****

****

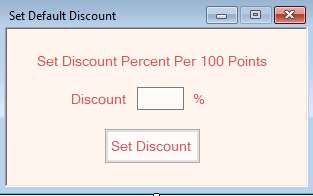
**Class ChangePin**

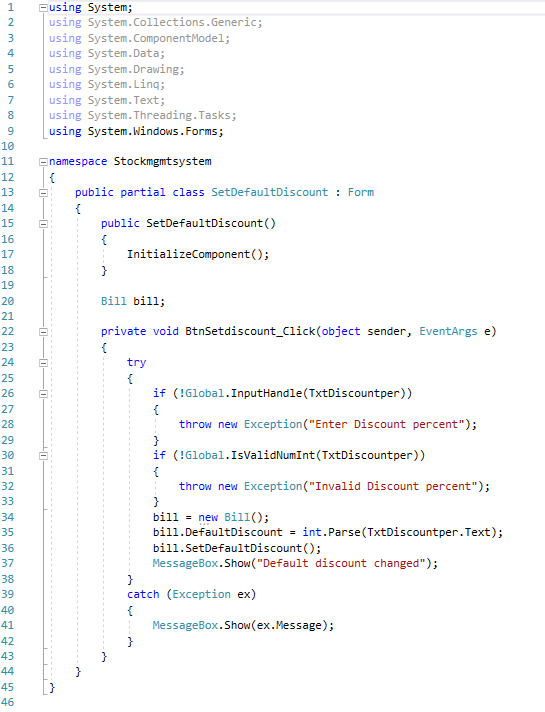
****

****

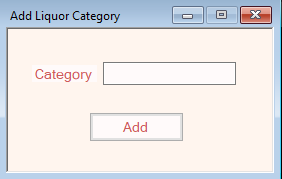
****

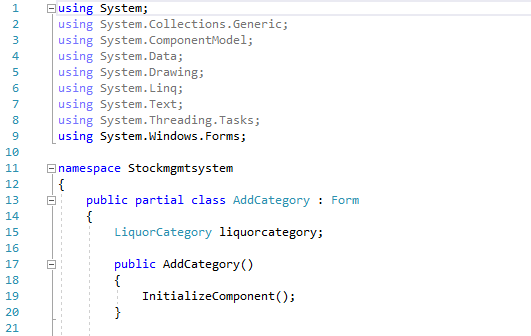
**Class SetDefaultDiscount**

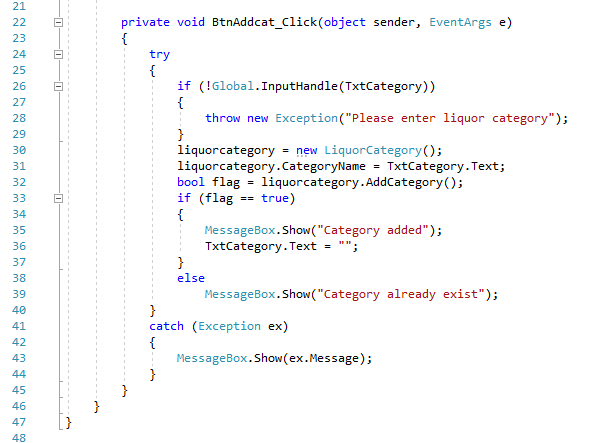
****

****

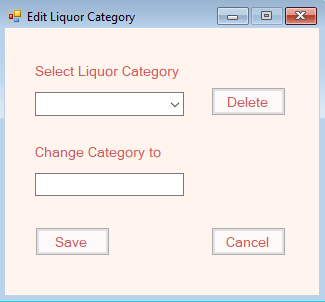
**Class AddCategory**

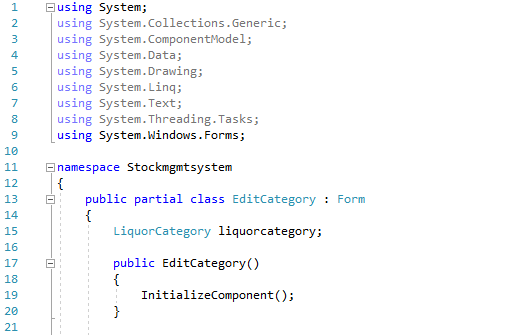
****

****

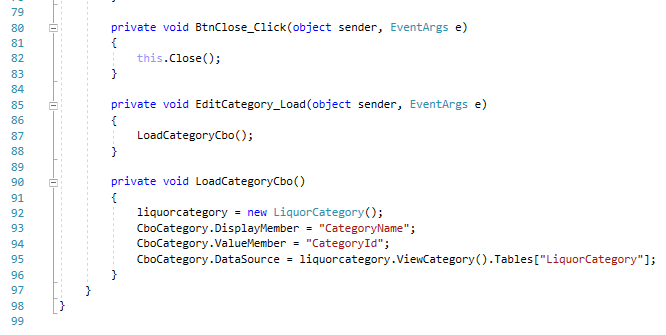
****

**Class EditCategory**

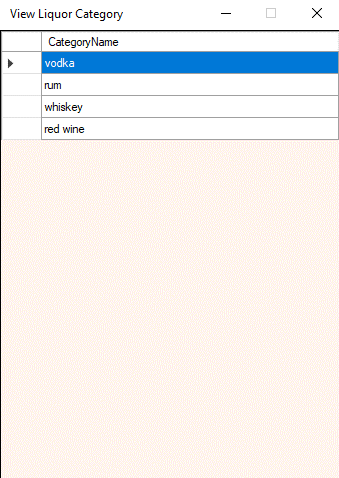
****

****

****

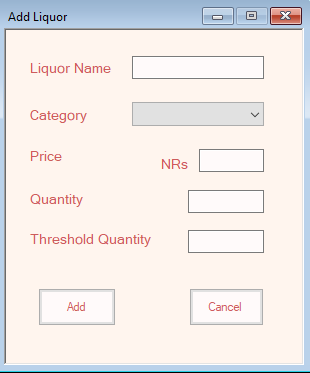


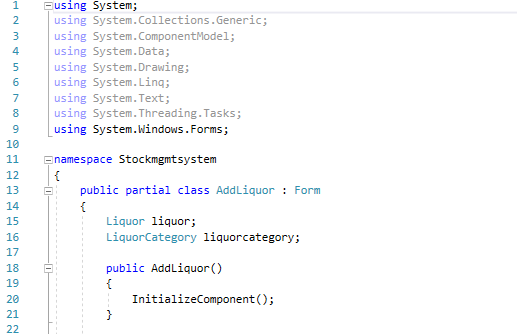
**Class ViewCategory**



****

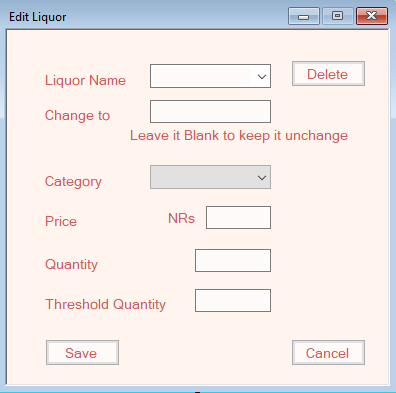
**Class AddLiquor**

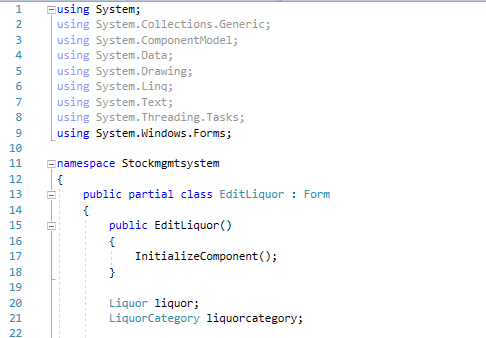
****

****

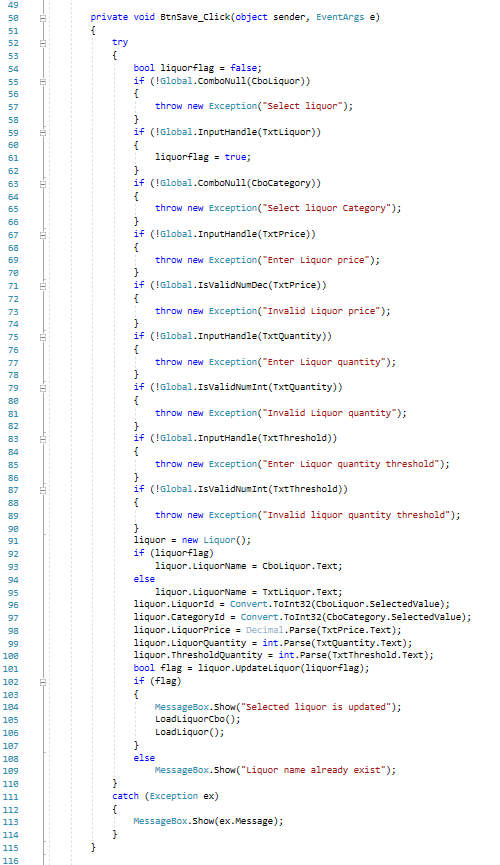
** **

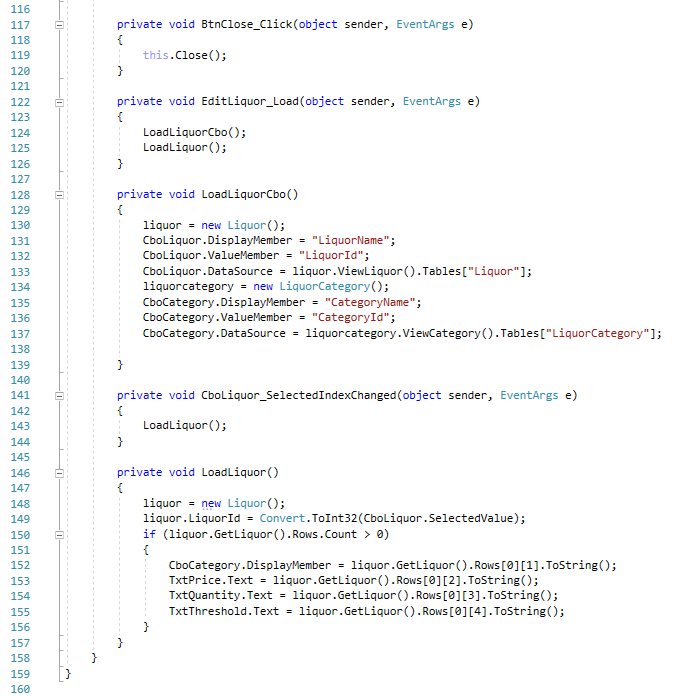
**Class EditLiquor**

****

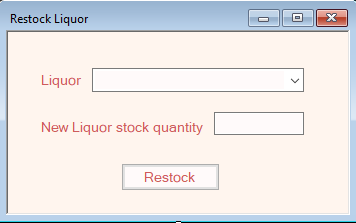
****

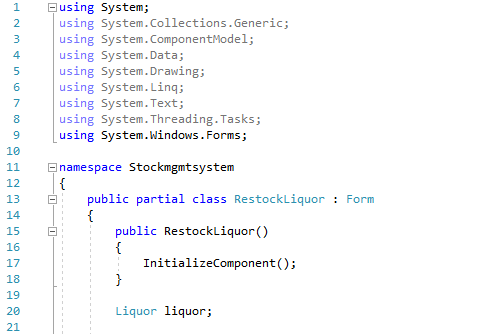
****

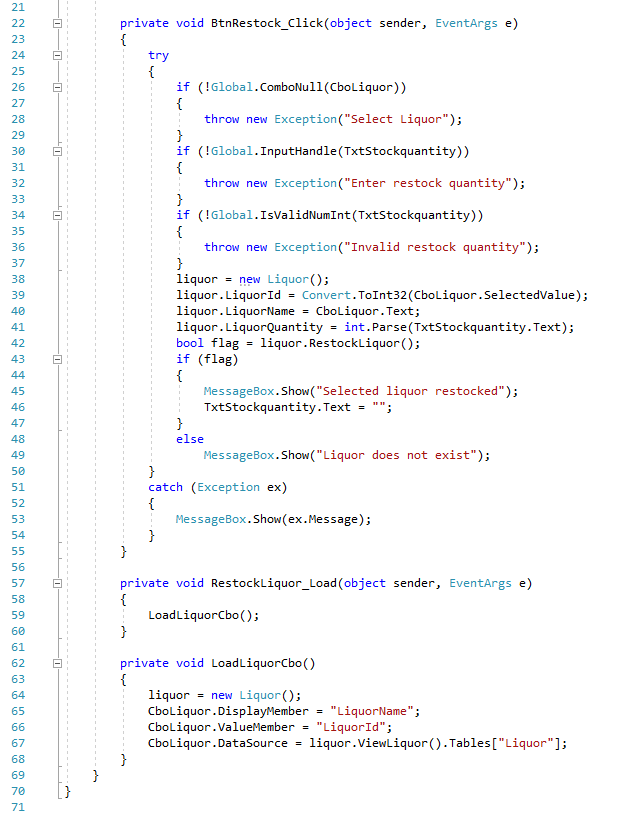
****

****

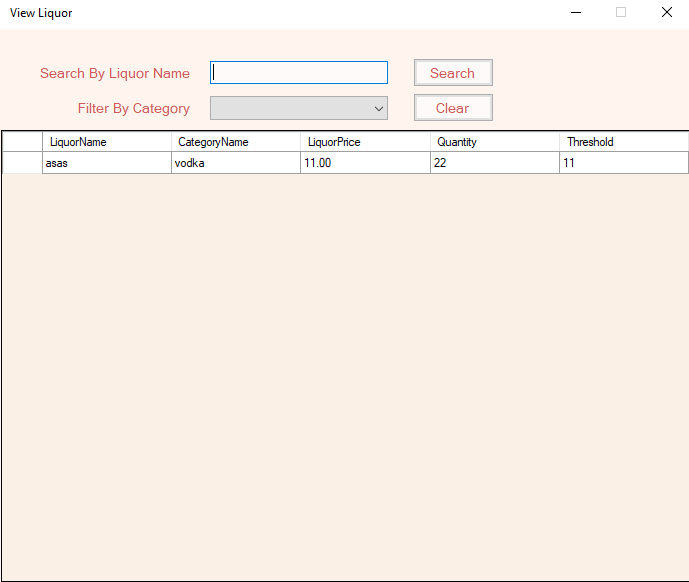
**Class RestockLiquor**

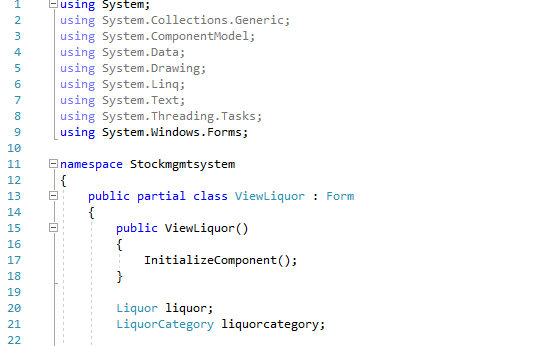






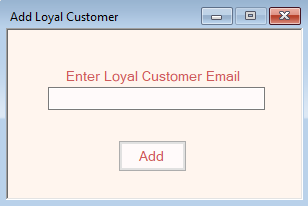
**Class ViewLiquor**

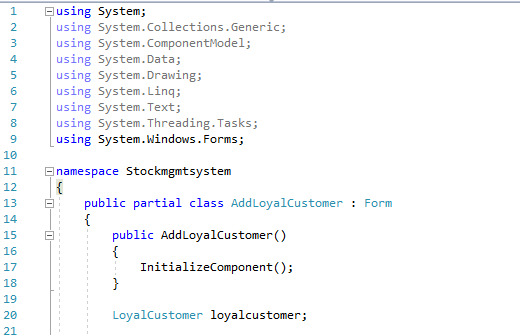
****

****

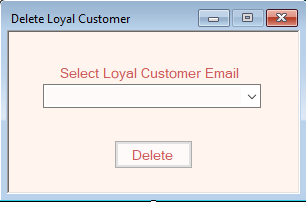
**Class AddLoyalCustomer**

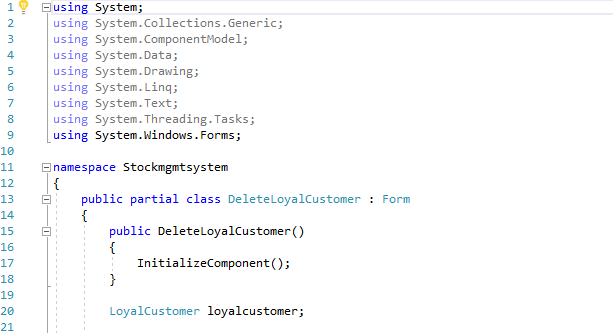
****

****

****

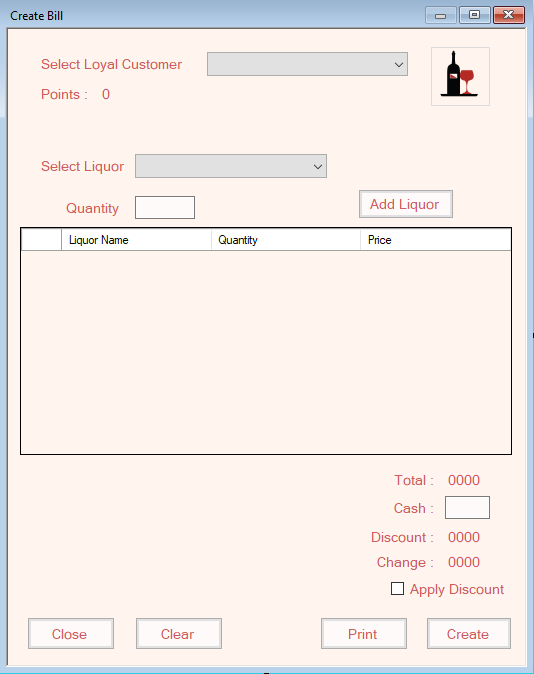
**Class DeleteLoyalCustomer**

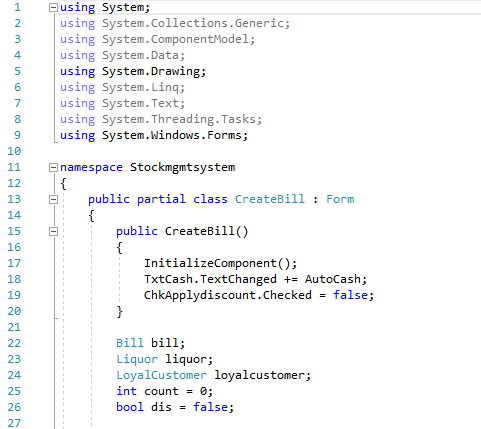
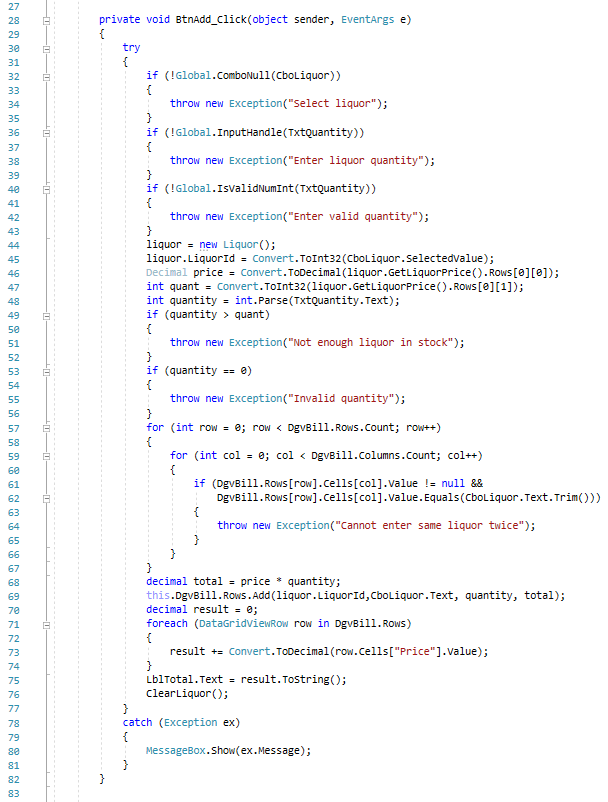
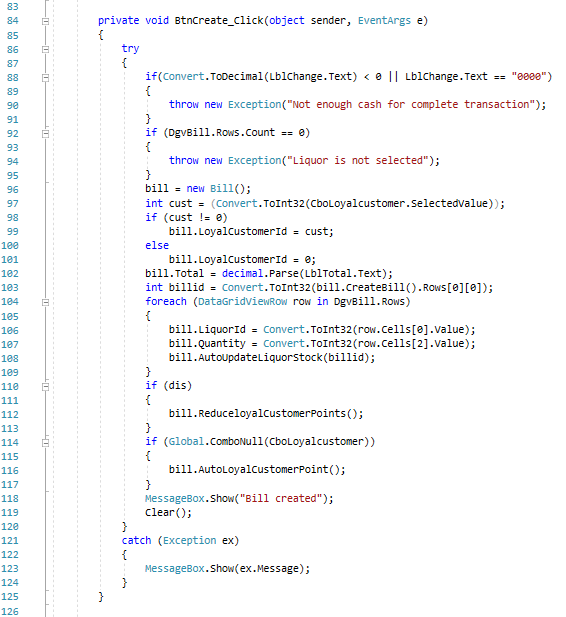
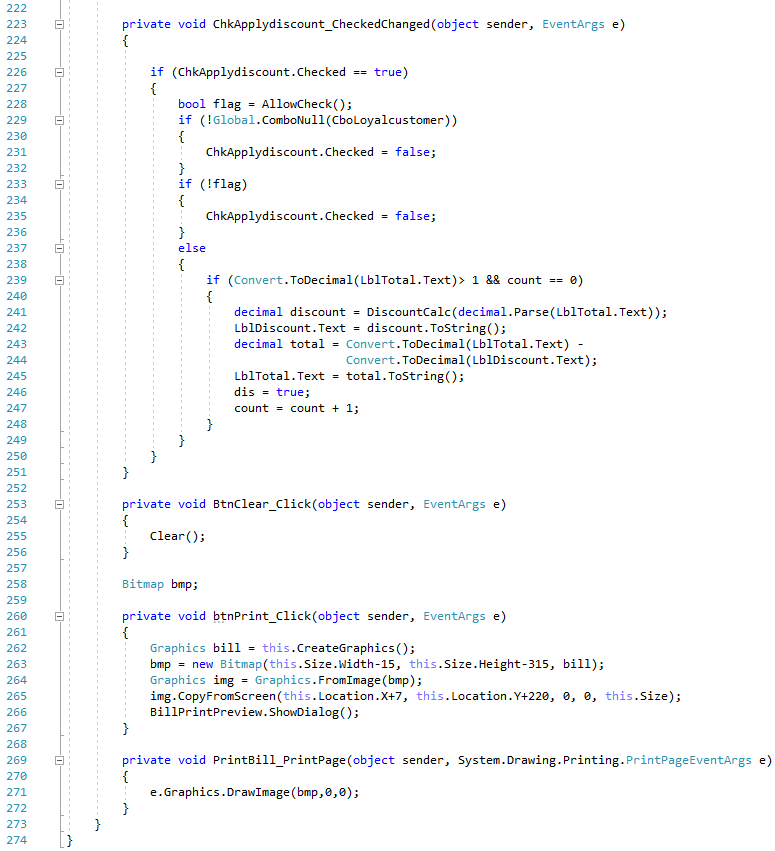
****

****

****

**Class CreateBill**

****

**** ****    

# Chapter 5: Testing

After completion of the software development, it is deployed in the testing environment. The testing team starts the testing of the entire system. All their functionality are tested and documented whether they pass the test or not. The team may find bugs and defects which are later communicated with the developer. The developer then fixes the bugs and are re-send for testing. The cycle continues and after completion of the testing phase, the software is ready to deploy. For this project I have selected unit testing and black box testing.

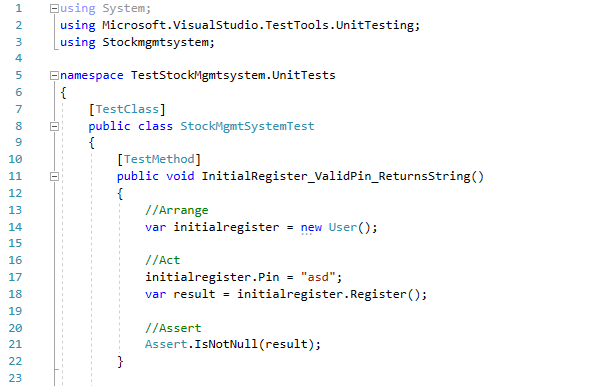
## 5.1 Unit testing

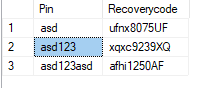
It is a type of testing where individual units of software are tested. Here is some unit test of the major function of the project. Visual studio has its own integrated unit testing framework. So, the unit testing tool of visual studio is used for this unit testing.

Features to be unit tested:

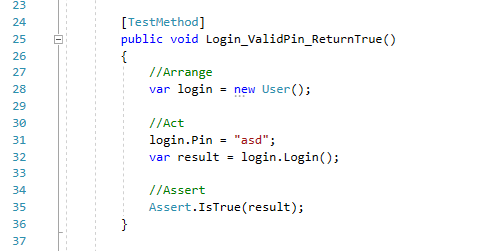
* Initial register
* Login
* Forgot pin
* Change pin
* Add category
* Add liquor
* Add loyal customer
* Restock liquor
* Update liquor
* Delete liquor

**Initial register Unit test:**

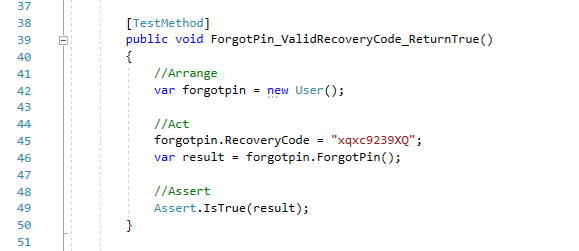


Result in database: 

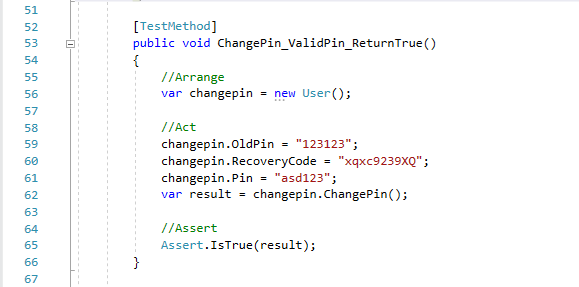
**Login Unit test:**

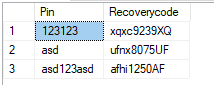
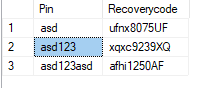


**Forgot Pin Unit test:**

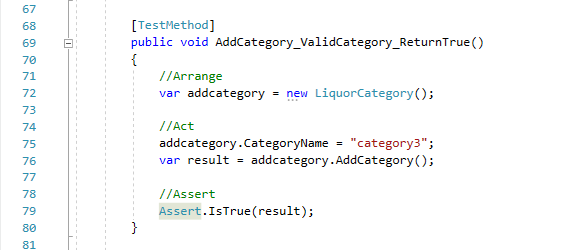


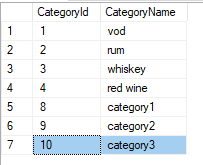
**Change Pin Unit test:**



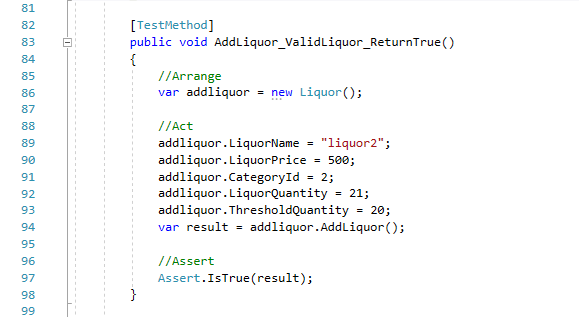
Result in database:  

**Add Category Unit test:**

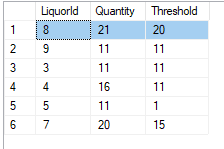


Result in database: 

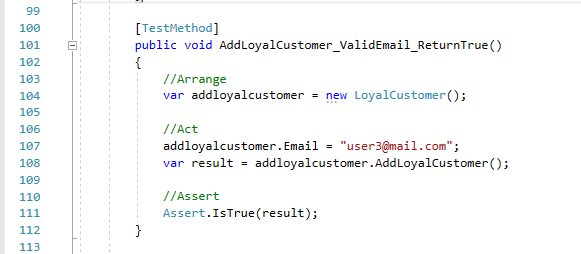
**Add Liquor Unit test:**

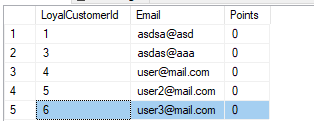


Result in database:

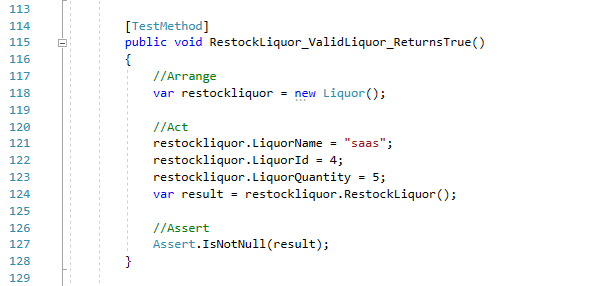
 

**Add Loyal Customer Unit test:**

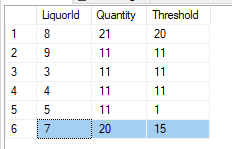
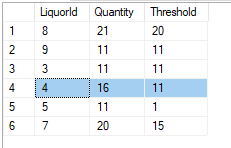


Result in database: 

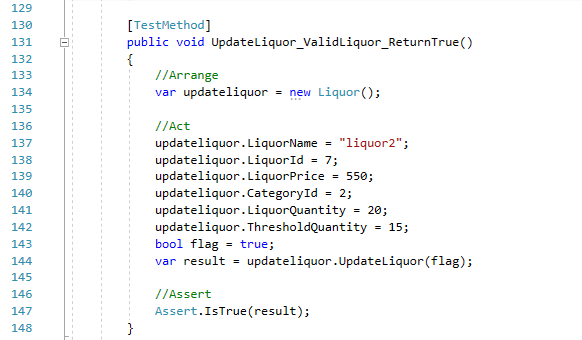
**Restock Liquor Unit test:**



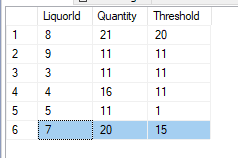
Result in database:

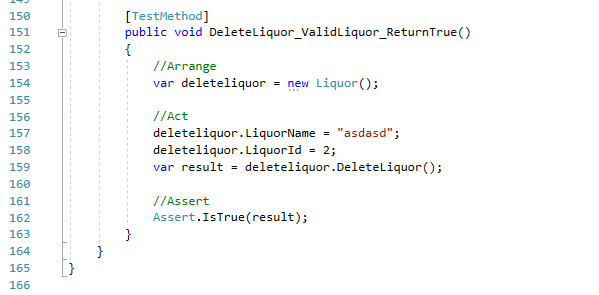
**Update Liquor Unit test:**

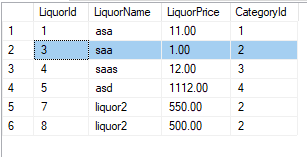


Result in database:

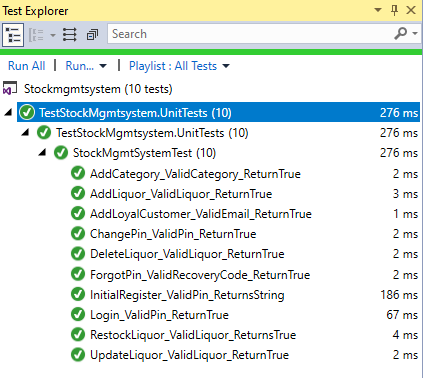
 

**Delete Liquor Unit test:**



Result in database: 

Whole unit test result:



The selected data used for testing are according to the data present in the database. The test result might vary according to the data in the database. To perform the test some data are manually entered through application for manipulation through the test.

## 5.2 Black box testing

It is a type of testing technique in which functionality of application are testing without checking the source code. Test can be both functional and non-functional.

**Test plan**

**Introduction:**

The test plan is created to test the features of the application. Different features and validation are tested such as login, register, adding category and liquor and their manipulation.

**Scope:**

The scope of this test plan is to check whether the application and its features work as intended. The UI events work as intended and do not go to break codes during runtime.

**Test items:**

Liquor Stock Management System

**Feature to be tested:**

* Initial Registration
* Login
* Forgot Pin
* Change Pin
* Adding Category
* Adding Liquor
* Restocking Liquor
* Creating Bill
* Search function of Liquor
* Category filter function of Liquor

**Item pass/fail criteria:**

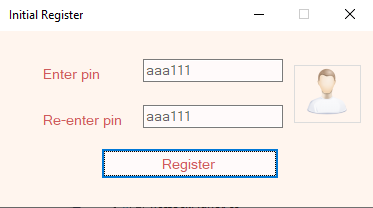
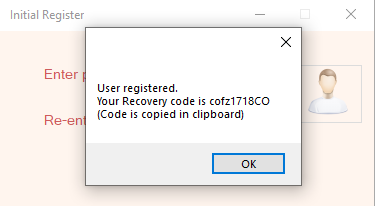
Whenever the actual outcome is the same as the expected outcome, the test items will be declared pass else it will be declared fail.

**Test log:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T01 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** To Initial register | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing registration of user. | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User should not be registered. | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open application |  | User should get recovery code and redirected to login window | | User got recovery code copied in clipboard and redirected to login window | Pass | Completed |
| 2 | Enter Pin and valid re-enter pin | aaa111 |
| 3 | Click on Register |  |

Table 8: Initial register black box test

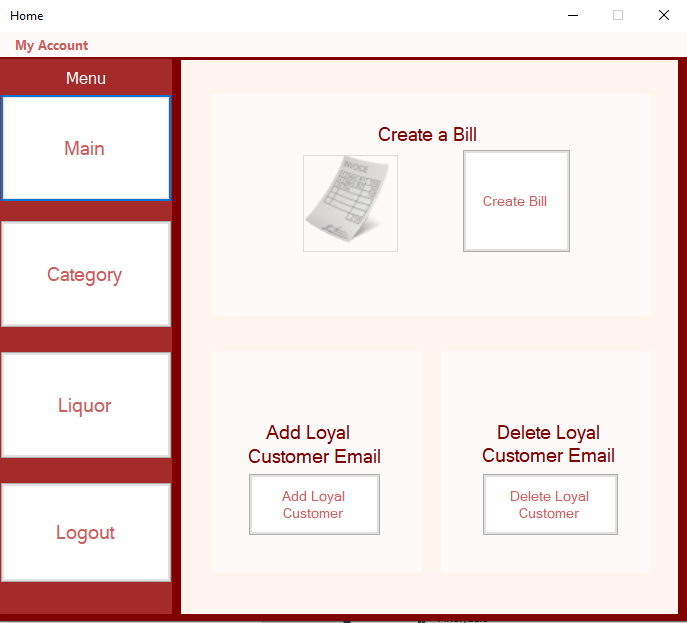
Screenshot of test:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T02 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** To verify user login | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing the login of a user | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must register. | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open application |  | User should be redirected to home window | | User was redirected to home window | Pass | Completed |
| 2 | Enter Pin | aaa111 |
| 3 | Click on Login |  |

Table 9: User login black box test

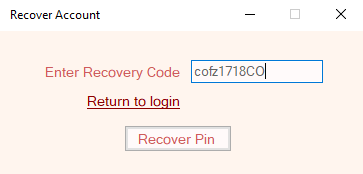
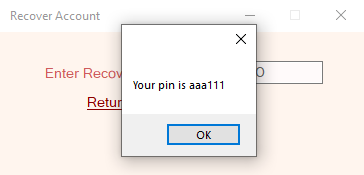
Screenshot of test:

** **

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T03 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Forgot pin | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to retrieve pin using recovery code. | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open application and click on Forgot Pin? |  | User must get login pin | | User got message box with login pin | Pass | Completed |
| 2 | Enter recovery code | cofz1718CO |
| 3 | Click Recover Pin |  |

Table 10: Forgot pin black box test

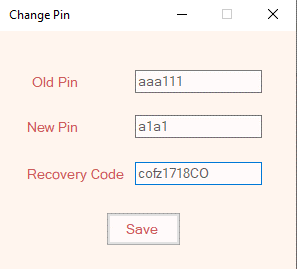
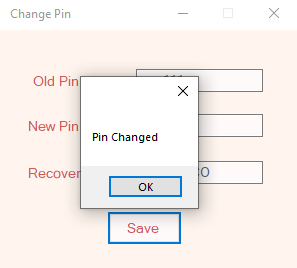
Screenshot of test:

** **

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T04 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** To Change pin | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing the display of all products | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open menu strip and click on Change Pin |  | Pin should be changed | | Pin was changed | Pass | Completed |
| 2 | Enter Old Pin | aaa111 |
| 3 | Enter New Pin | a1a1 |
| 4 | Enter Recovery Code | cofz1718CO |
| 5 | Click on Save |  |

Table 11: Change pin black box test

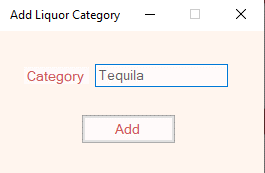
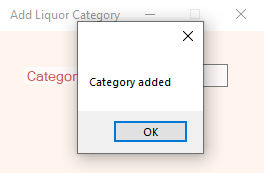
Screenshot of test:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T05 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Adding new liquor category | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to add liquor category | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open Add Category from Category tab |  | New category should be added | | New category was added | Pass | Completed |
| 2 | Enter Category | Tequila |
| 3 | Click on Add |  |

Table 12: Add liquor category black box test

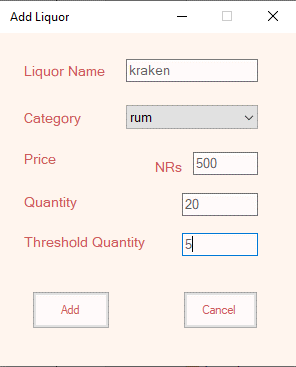
Screenshot of test:

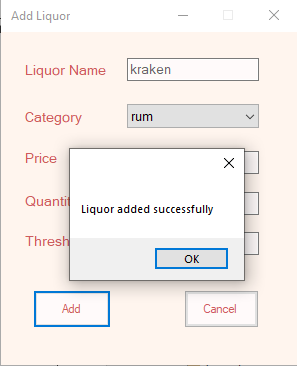
 

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T06 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Adding new liquor | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to add liquor | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open Add Liquor from Liquor tab |  | New liquor should be added | | New Liquor was added | Pass | Completed |
| 2 | Enter Liquor Name | kraken |
| 3 | Select Category | rum |
| 4 | Enter Price | 500 |
| 5 | Enter Quantity | 20 |
| 6 | Enter Threshold Quantity | 5 |
| 7 | Click on Add |  |

Table 13: Add liquor black box test

Screenshot of test:

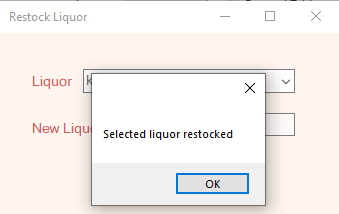
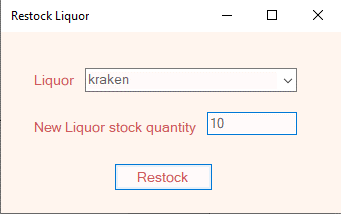




|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T07 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Restocking liquor quantity | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to restock liquor quantity | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open Restock Liquor from Liquor tab |  | Liquor quantity should be restocked | | Liquor quantity was restocked | Pass | Completed |
| 2 | Select Liquor | kraken |
| 3 | Enter New Liquor stock quantity | 10 |
| 4 | Click on Restock |  |

Table 14: Restock liquor quantity black box test

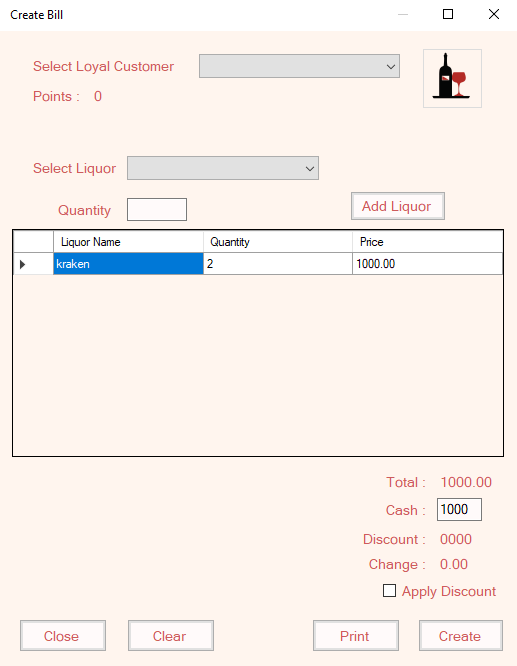
Screenshot of test:

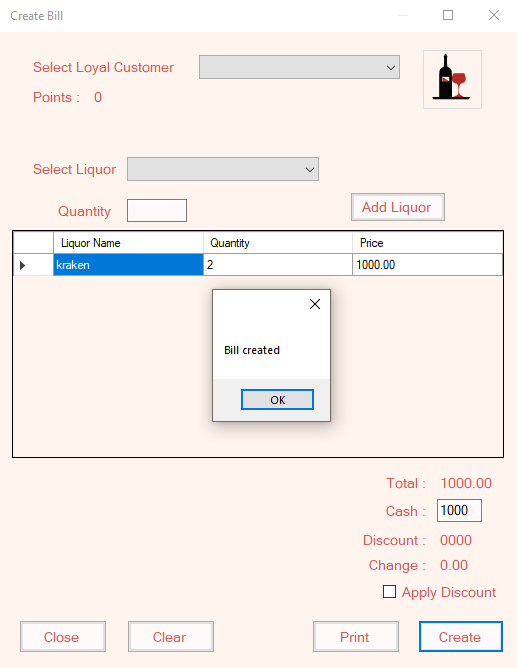


|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T08 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Creating a bill | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to create bill | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open Create Bill from Main tab |  | Bill should be created | | Bill was created | Pass | Completed |
| 2 | Select Liquor | kraken |
| 3 | Enter Quantity | 2 |
| 4 | Enter Cash | 1000 |
| 5 | Click on Create |  |

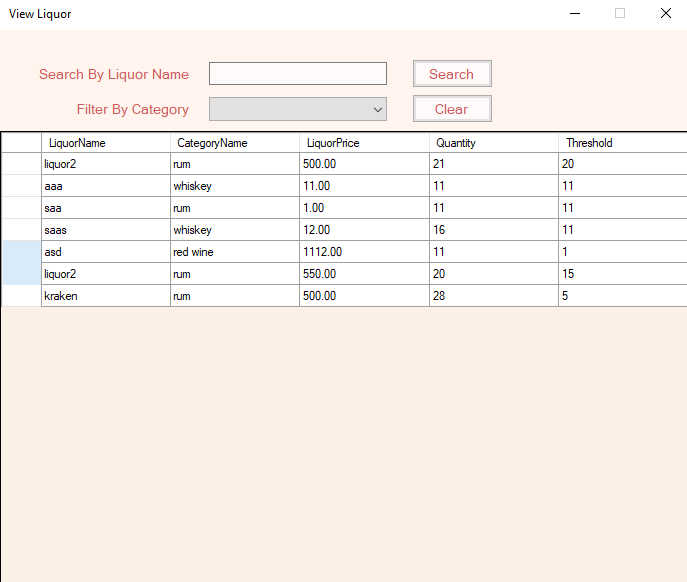
Table 15: Create bill black box test

Screenshot of test:





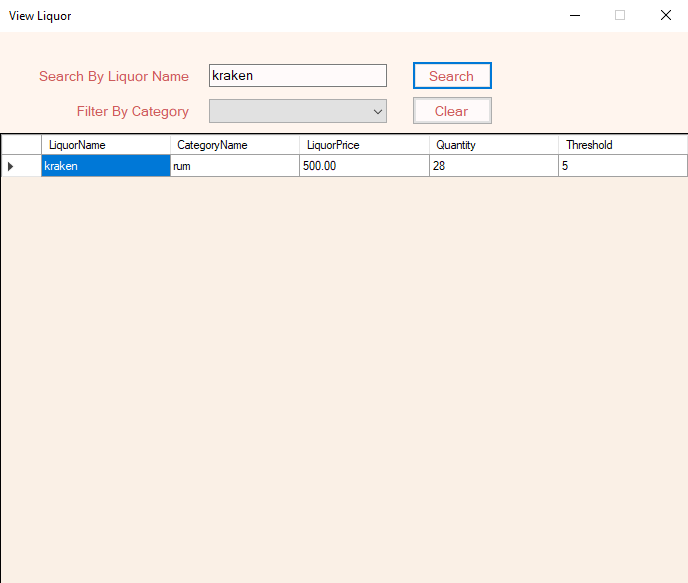
**List of all Liquors for next test**

****

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T09 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Searching liquor | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to search liquor by its name | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open View Liquor from Liquor tab |  | Display kraken liquor and its information | | kraken liquor and its information was displayed | Pass | Completed |
| 2 | Enter Liquor Name | kraken |
| 3 | Click on Search |  |

Table 16: Search liquor black box test

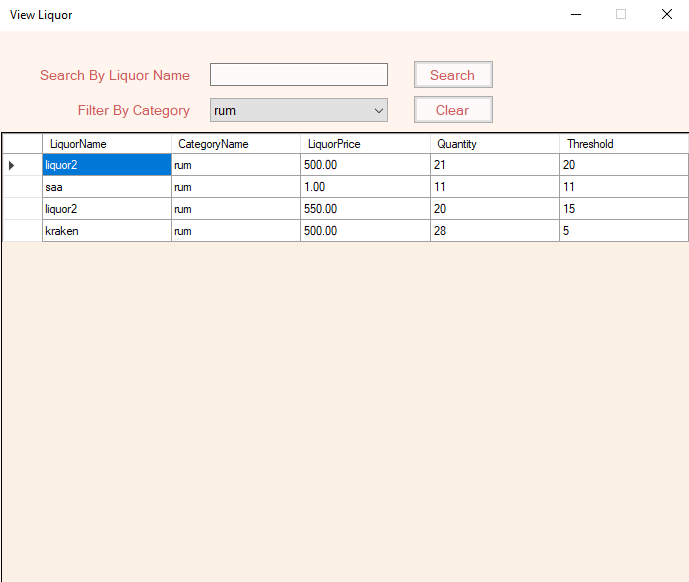
Screenshot of test:

****

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Title:** Liquor Stock Management System | | | | | | | |
| **Test Case ID:** T10 | | | | **Designed By:** Ajar Maharjan | | | |
| **Test Title:** Filtering liquor through category | | | | **Date of Design:** June 22th, 2019 | | | |
| **Test Description:** Testing to filter liquor through category | | | | **Executed By:** Ajar Maharjan | | | |
| **Date of Execution:** June 29th, 2019 | | | |
| **Pre-Condition:** User must enter valid data | | | | | | | |
|  | | | | | | | |
| **Step S.N.** | **Test Steps** | **Test Data** | **Expected Result** | | **Actual Result** | **Status** | **Remarks** |
| 1 | Open View Liquor from Liquor tab |  | Display list of all liquor having rum category | | List of liquor having rum category was displayed | Pass | Completed |
| 2 | Select Liquor Category | rum |

Table 17: Filter liquor through category black box test

Screenshot of test:



# Chapter 5: Other Project Issues

## 5.1 Project Limitation

There is some limitation to the project. They are:

* **Application based for a stand-alone computer**

The application does store its data online, therefore, user can only access the application and its data only at the computer where it is installed

* **Compatibility**

The application only compatible on Windows operating system. Does not support Linux and Mac.

* **Does not cover damaged, stolen or scrapped stocks.**

The application does not cover the problem raised from the stocks that are damaged or stolen.

## 5.2 Future work

The project was developed according to the gathered requirements but there are lots of features that can be introduced and developed in future. Some of the features can be:

* Function to print bill for invoice printer.
* Function to add stocks with the help of barcode scanner.
* Function to keep user kept logged if not logged out from the application.
* Function to have multi-user stock manipulation.
* Function to create monthly sales and stock report.
* Function to show windows notification when stock level is below the threshold quantity.

## 5.3 Other project issues

During the development of the system, here are some issues that has occurred:

* **Time estimation problem.**

The time estimated for designing the class, sequence and activity diagram was enough.

The backend development for the application was way shorter than expected.

* **Debugging.**

The bug on login function when the application closes automatically when going from the login menu to forgot pin after logging out from the application.

Some queries were not working as intended (incorrect queries undiscovered).

## 5.4 Risk management

The process to identify and control the threats and risks that can harm the project with the appropriate solution is called risk management.

**Value of Consequences**

|  |  |
| --- | --- |
| **Consequences** | **Value** |
| Very Low | 1 |
| Low | 2 |
| Medium | 3 |
| High | 4 |
| Very High | 5 |

**Value of Likelihood**

|  |  |
| --- | --- |
| **Likelihood** | **Value** |
| Low | 1 |
| Medium | 2 |
| High | 3 |

Some of the risks while doing the project were:

**Impact**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.N.** | **Risk** | **Consequences** | **Likelihood** | **Impact** | **Remarks** |
| 1 | Improper time/resource management. | 3 | 2 | 6 | The designing phase was giving more time while cutting the time for implementation. |
| 2 | Incorrect database queries. | 4 | 2 | 8 | All queries were testing on MS-SQL management studio. |
| 3 | Requirement inflation. | 3 | 1 | 3 | Requirements were analyzed again. |
| 4 | Bugs or errors. | 4 | 3 | 12 | Various bugs might occur during development of the application which are later addressed after testing. |
| 5 | Missed bugs or incomplete features. | 5 | 1 | 5 | Due to time boundary, Some feature might not work as intended. |
| 6 | Incompatibility while switching devices to develop application. | 5 | 2 | 10 | The application is built on the latest version of MS-SQL server and management studio for backend and visual studio 17 for frontend. |

Table 18: Risk management table

## 5.5 Configuration management

Software Configuration management is the discipline which is consist of processes and techniques to manage the changes made to the software project. It helps to identify the individual elements and configuration, tracks the changes and version selection and control. It is an efficient method to track all the folders and files to reduce redundant work and prevent any future anomalies and manage effectively.

The configuration management of the system that is being developed is as show in the screenshot below:

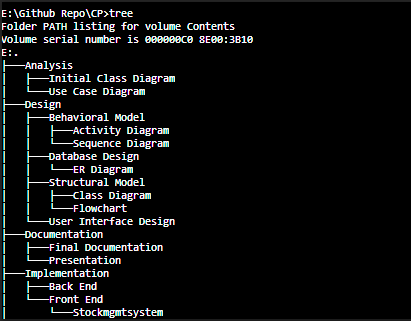


Figure 28: Configuration management 1.1

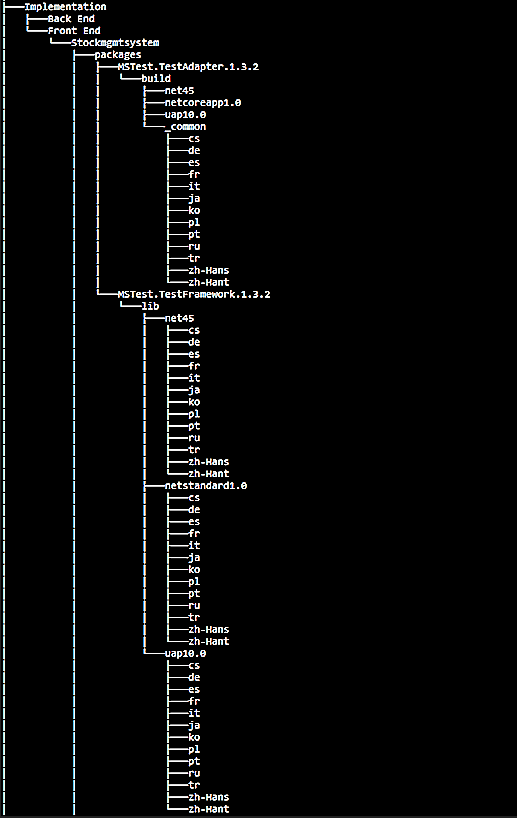


Figure 29: Configuration management 1.2

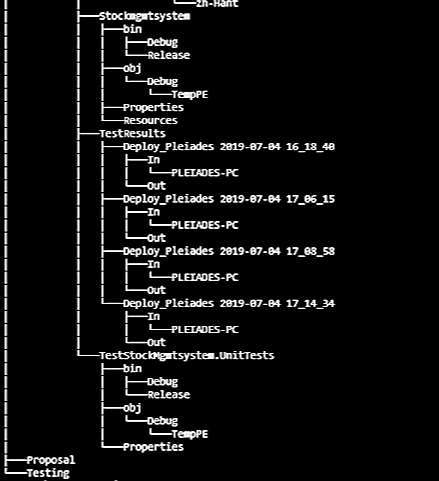
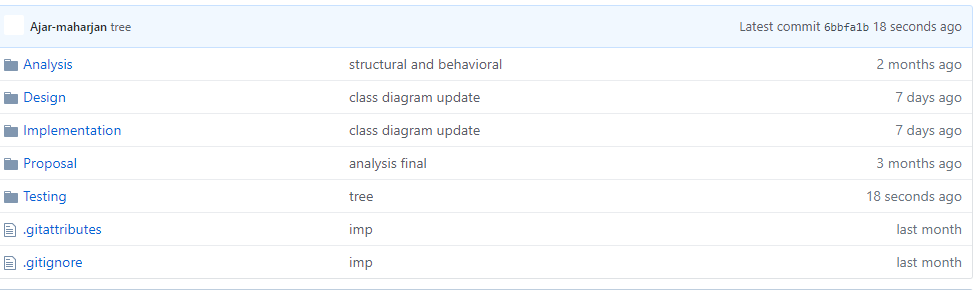


Figure 30: Configuration management 1.3\

Backup for the project is done using GitHub.

GitHub clone link: <https://github.com/Ajar-maharjan/CP.git>

GitHub ID: Ajar-maharjan



**Figure 31: GitHub backup configuration management**

## 5.6 User manual

**If the application is not registered then it shows register menu.**

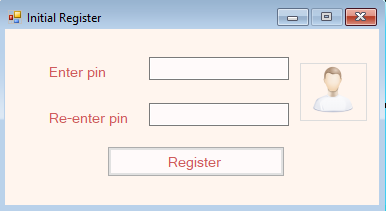
****

Figure 32: Initial register menu

* Enter correct pin at both text area then click on register to register user.

After registering first user the application never allows to create another user. After registering the user the application will give recovery code for the user. This recovery code is used to recover account when user forget the password.

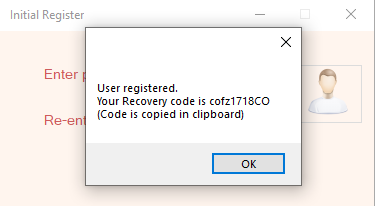


Figure 33: Recovery code after registering.

**After registering the user can login the system.**

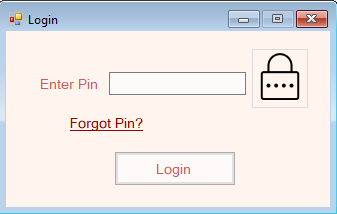
****

Figure 34: login menu

* Type correct pin then press on Login to login the system.
* **Click Forgot Pin? To go to forgot pin window.**

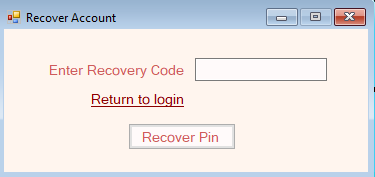
****

Figure 35: Forgot Pin window

* Enter the recovery code and press Recover Pin.
* The system will give the pin.

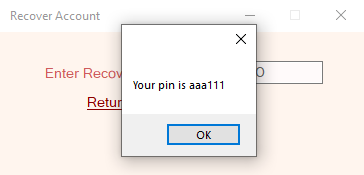
****

Figure 36: after recovering pin

**After logging in the application redirects user to home menu.**

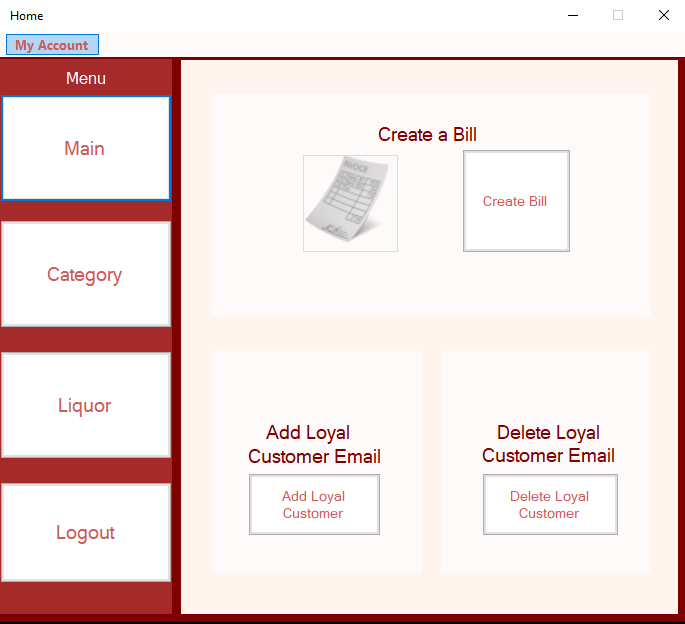


Figure 37: Home menu

* **Click on Create Bill to create Bill**

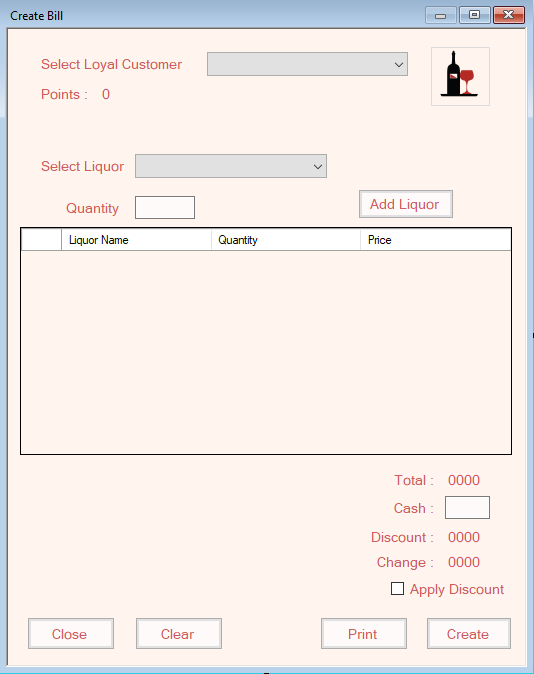
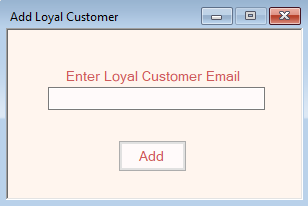
****

Figure 38: Create Bill window

* + Select loyal customer according to need.
  + Select liquor and enter liquor quantity.
  + Press Add liquor after selecting to add on bill.
  + Enter cash provided by customer on cash field.
  + Select Apply Discount if points are enough and customer wants to apply discount.
  + Press Print to print the bill.
  + Press Create to enter bill on database.
  + Press Close to cancel the bill and close the window.
* **Click on Add Loyal Customer to add customer email**

****

* Figure 39: Add Loyal Customer window
  + Enter the customer Email address.
  + Click on Add to add new customer.
* **Click on Delete Loyal Customer to remove customer email.**

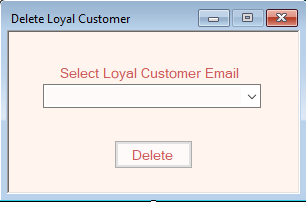
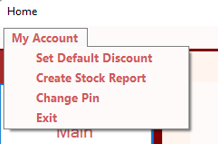
****

Figure 40: Delete Loyal Customer window

* + Select Customer Email.
  + Click on Delete to remove customer email.
* **My account menu bar tab to access application basic function I.e.** 
* **Category tab to access liquor category functions.**

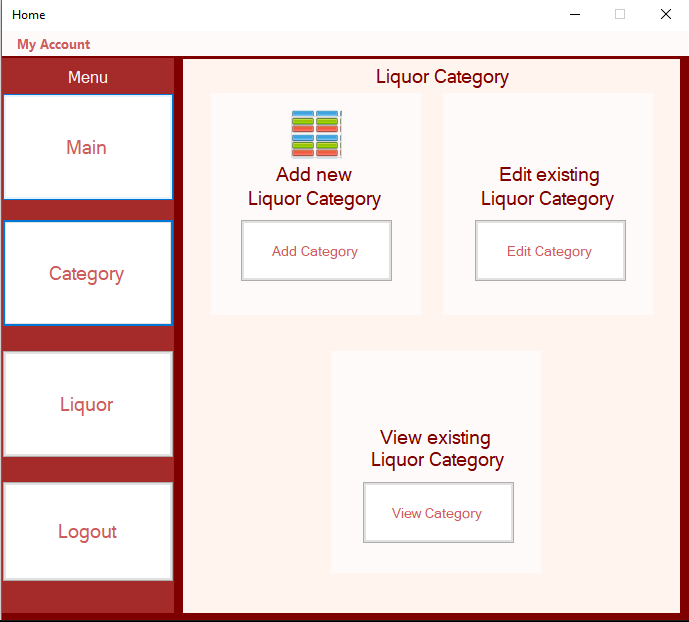


Figure 41: Liquor category tab

* + **Click on Add Category to add category**

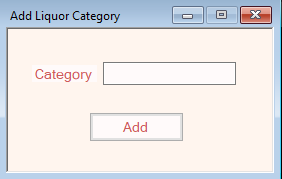


Figure 42: Add category Window

* + - Enter category name and click Add to add category.
  + **Click on Edit Category to edit existing category name.**

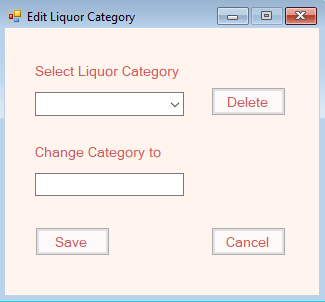
****

Figure 43: Edit liquor category window

* + - Select existing liquor category.
    - Click on Delete to delete the category.
    - Enter new category name and click on save to change category name.
  + **Click on View Category to show category list.**

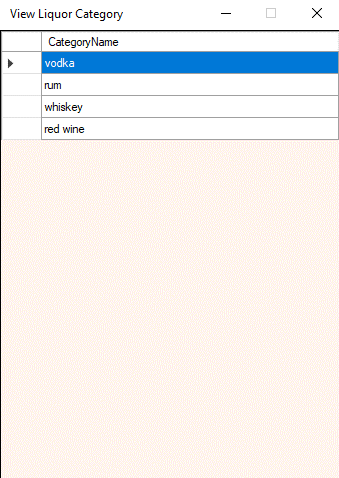


Figure 44: View Category window

* **Click on Liquor tab to access liquor functions.**

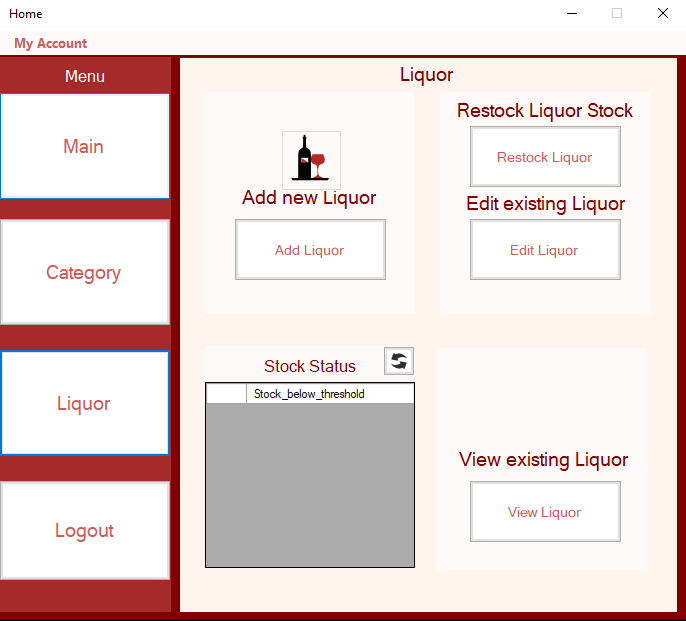


Figure 45: Liquor Tab

* + Click on  to refresh stock status.
  + **Click on Add Liquor to add new liquor**

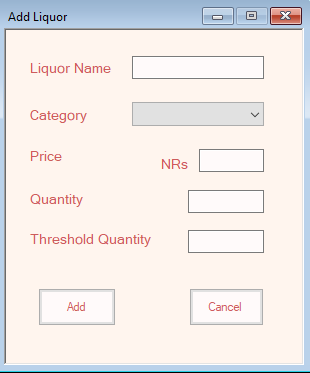
****

Figure 46: Add liquor Window

* + - Enter new liquor name.
    - Select category for the liquor.
    - Enter price of the liquor.
    - Enter quantity and its threshold quantity.
    - Click on Add to add the liquor.
    - Click on Cancel to close the window.
  + **Click on Restock Liquor to restock liquor quantity**

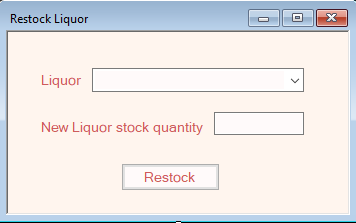


Figure 47: Restock Liquor Window

* + - Select Liquor Name Enter new total quantity to be added.
    - Click restock.
  + **Click on Edit Liquor to Edit existing liquor**

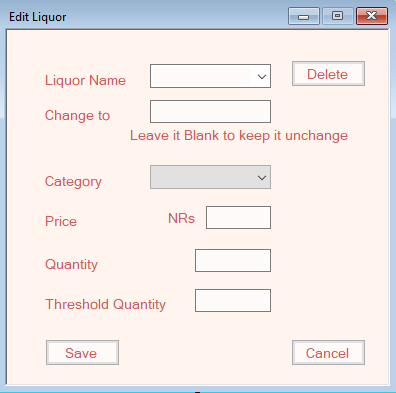
****

Figure 48: Edit liquor window

* Select Liquor Name
* Click on Delete to delete the liquor.
* Enter on other fields to which you want to change.
* Click on save to make changes.
  + **Click on View Liquor to Show liquor list.**

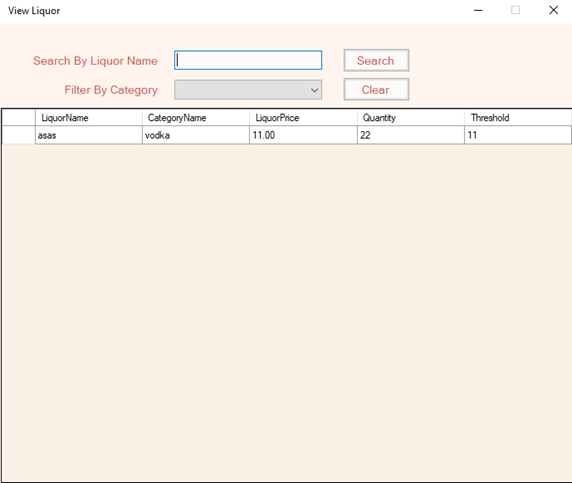


Figure 49: View Liquor window

* Enter Liquor Name and click search to search by liquor name.
* Select category to filter liquor through category.
* Enter liquor name and select category to search filtering the category.
* Click clear to clear liquor search field and category field
* Press Logout to logout from the system.

**Menu bar**

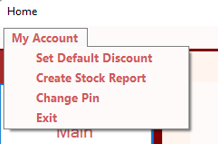


Figure 50: Menu bar options

* **To set default discount click on Set Default Discount option.**

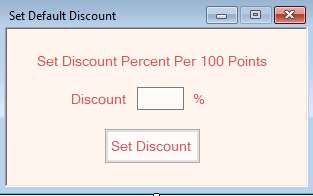
****

Figure 51: Set default discount window.

* + Enter discount percent then click on Set Discount.
* **To Create Stock report click on Create Stock Report**

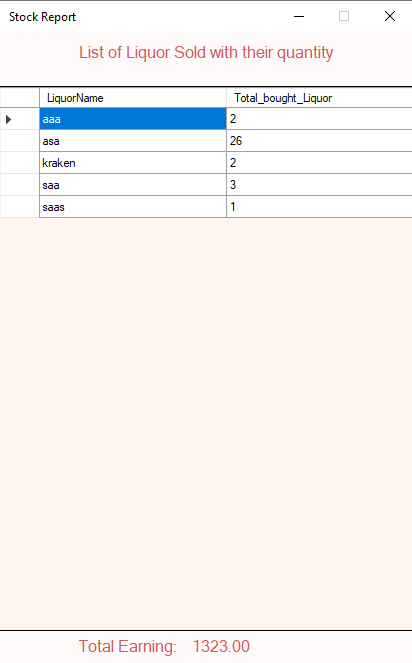
****

Figure 52: Stock report window

* **To Change Pin click On Change Pin**

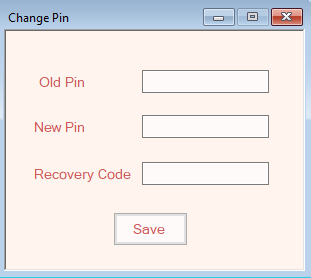
****

Figure 53: Change Pin

* + Enter old Pin, New Pin and Recovery Code provided when registering the user.
  + Click on Save to change pin.

# Chapter 6: Conclusion

By following the waterfall SDLC, the project began with the requirements gathering and analysis of the requirements. Techniques such as documents reviews, brainstorming, focus-group discussions, and observation were utilized for gathering the requirements. After this feasibility study was done to check whether the project is feasible or not. Through various analysis, OOAD methodology was chosen for this project. The functional and non-functional were listed and Moscow prioritized. After Prioritization, Use case diagram of the proposed system was developed.

After analysis, Designing phase was done. In this phase all required UML diagrams were made i.e. Entity-relationship diagram and its data dictionary, Class diagram, flowchart, activity diagram and sequence diagram for the proposed system. The UI prototypes were made to visualize how the user the interface of an actual application.

In the Implementation phase, the backend database was made according to ER-diagram and data dictionary from the design phase in MS-SQL Server. Later, the frontend application was made using visual studio 17, C#.net framework. The application function was implemented according to activity and sequence diagram made at the design phase and some functions were modified according to need.

During the implementation phase, the bugs and errors were resolved in this testing phase. Various error queries were tested multiple times in MS-SQL server studio management and later debugged on the main application. Following the testing, the final documentation is made to show how the project was done and for future reference.

# References