PROJECT REPORT

ON

General Store Management System

B.Tech (CE) Sem-VI

In the Subject of

Service Oriented Computing (CE-619)

Zarmy Patel (CE - 097) (17CEUON055)

Yug Rajani (CE - 109) (17ITUOS059)

Sani Ranpariya (CE - 112) (17CEUOF017)

Under the Guidance of

Prof. Ankit P. Vaishnav

Department Of Computer Engineering

Faculty of Technology,

Dharmsinh Desai University, Nadiad.

DHARMSINH DESAI UNIVERSITY

College Road, NADIAD-387001(Gujarat)

CERTIFICATE

This is to certify that the term work carried out in the subject of **Service Oriented Computing** and recorded in this report is bonafide work of **Ms. Zarmy Patel (Roll No.: 097, ID: 17CEUON055), Mr. Yug Rajani (Roll No.: 109, ID: 17ITUOS059) and Mr. Sani Ranpariya (Roll No.: 112, ID: 17CEUOF017)** of **B.Tech Semester VI** in the branch of Computer Engineering during the academic year 2019-20.

Prof. Ankit P. Vaishnav	Dr. C.K Bhensdadia
(Project Guide and Assistant Professor)	Head of CE Dept.,
Faculty of Technology,	Faculty of Technology,
Dharmsinh Desai University,	Dharmsinh Desai University,
Nadiad.	Nadiad.

Table of Contents

Abstract

In an effort to embrace automation, various stores, both big and small start out with a system to manage their business processes to scale up their productivity and to accomplish their business in the fast growing technological world. General Store Management System is Client-Server Desktop Application build for managing and handling the products in any store. This system is developed using Windows Communication Foundation (WCF) and Windows Forms Application, WCF used for server side to provide services and clients are made by using windows form which can be easily integrated with the services.

General Store Management system has client-side application developed using Windows Form maintaining the records of various products in a store which has a wider scope and can be used even in a supermarket for easy maintenance of their products and the whole system revolves around the entity product.

1. Introduction to the Project

1.1 Brief Introduction

Project Name :	General Store Management System
Project Definition/Aim :	The main aim of project is to provide better maintainability of products record for various general stores.
Developed for :	Whole sellers, Super Market, Small/Big Stores
Front End :	Windows Forms Application (with C#)
Back End :	WCF (Windows Communication Foundation)
Other Technologies :	Microsoft SQL Server 2017
Documentation Tool :	MS Word
Submitted To :	Any General Store
IDE :	Microsoft Visual Studio (2019-Community)

1.2 Tools and Technology Used for Development of the Project

Technology Used:

- C# Language
- Windows Forms
- SQL

Platform Used:

- Windows

Tools Used:

- Microsoft Visual Studio Community 2019
- ERD plus
- UMLet
- Microsoft SQL Server 2017

2. Software Requirements Specification

2.1 Scope

The scope of the system is very large so it is used in any large scale shop which has many branches and also retailer, whole seller etc can used this system.

2.2 System Functional Requirements

<u>Users:</u>

1. Manager
2. Cashier

1. Manager

R1: Login/Logout

R.1.1: Enter credentials

Input: Manager’s username and password.

Output: "Successfully logged in/signed up" message.

R.1.2: Logout

Description: When admin wants to leave the site, then he may log out of his account.

Input: User selection.

Output: "Successfully logged out" message.

R2: Manage Users

R.2.1: Add User

Input: User information.

Output: "Successfully added" message along with ID and Password.

R.2.2: Change User Profile

Description: Admin is allowed to change the profile details of the users along with their password upon request by the user.

Input: Details to be updated along with new values.

Output: "Successfully changed" message.

R.2.3: Delete User

Description: Admin may delete the account of any user as and when required.

Input: ID of the user to be deleted.

Output: "Successfully deleted" message.

R3: Manage Products

R.3.1: Add Product

Description: Manager can add products in to the store’s stock

Input: Enter Product Details

Output: Product Added Successfully

R.3.2: Remove Product

Description: Manager can remove products in to the store's stock

Input: Enter Product Details

Output: Product Removed Successfully

R.3.3: Update Product

Description: Manager can update products in to the store's stock

Input: Enter Product Details

Output: Product Updated Successfully

R.3.4: View all Products

Description: Manager can view all products available in to the store's stock

Input: User selection

Output: Statistical Information of Total Products available

R.3.5: Search Product

Description: Manager can search products in to the store stock using any detail of the product.

Input: Enter Product's anyone Detail.

Output: Filtered Search results.

R.3.6: Generate Barcode for any new Product

Description: Manager can make a barcode for the products in to the store's stock.

Input: Enter Product Details.

Output: Barcode generated successfully.

R4: Manage stock of Products

R.4.1: Add Stock

Description: Manager can add stock in to the store

Input: Enter Stock Details

Output: Stock Added Successfully

R.4.2: Delete stock

Description: Manager can delete stock which are expired.

Input: Enter Stock Details

Output: Stock deleted Successfully

2. Cashier

R1: Login/Logout

R.1.1: Enter credentials

Input: Cashier information.

Output: "Successfully logged in/signed up" message.

R.1.2: Logout

Description: When cashier wants to leave the site, then he may log out of his account.

Input: User selection.

Output: "Successfully logged out" message.

R2: Manage Orders

R.2.1: Add Order

Description: Cashier can add order in to the system

Input: Enter Order Details

Output: Order Added Successfully

R.2.2: Update Order

Description: Cashier can update order in to the system

Input: Enter Order Details

Output: Order Updated Successfully

R3: Manage Bills

Description: It can manage the bills which are being generated and can do some statistical analysis.

R.3.1: Generate Bill

Description: Cashier can generate in to the system

Input: Enter Bill Details

Output: Bill generated successfully

R.3.2: Print Bill

Description: Cashier can print bill in to the system

Input: Enter Bill Details

Output: Bill Printed Successfully

2.3 Other Non-Functional Requirements

1. **Performance**
   The system must be fast and accurate as the main aim is automation and efficiency, it must provide accurate results in less time and using less energy.

2. **Safety**
   As all the data of the products would be stored in the system, safety is one of the main concerns of the system and hence only authorised users should be allowed to login to the system.

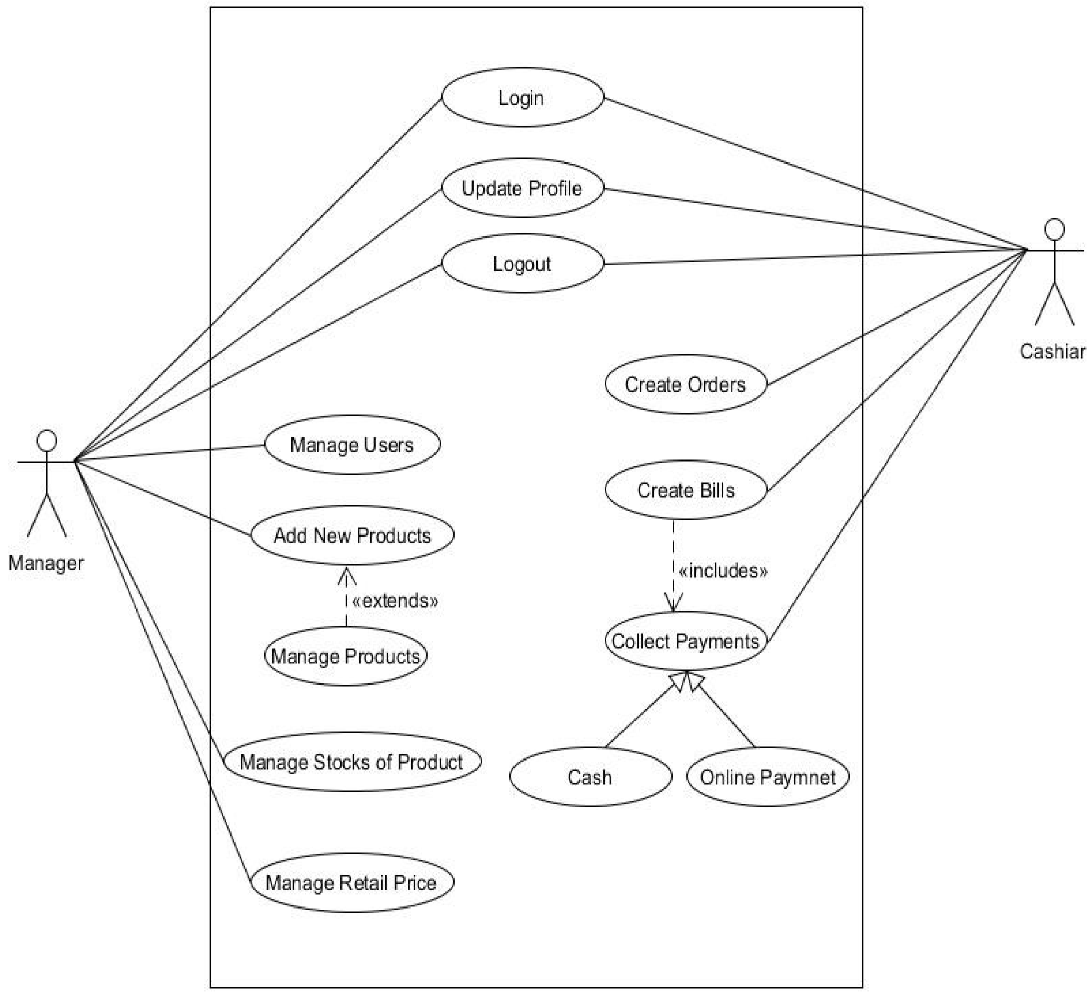
3. **Reliability**
   The data and personal information of any product(s) should not be leaked and must be in safe hands so the system must be reliable.

4. **Database**

   The database of the system should function perfectly, only then the data would be stored into the system correctly thus improving the performance of the system.

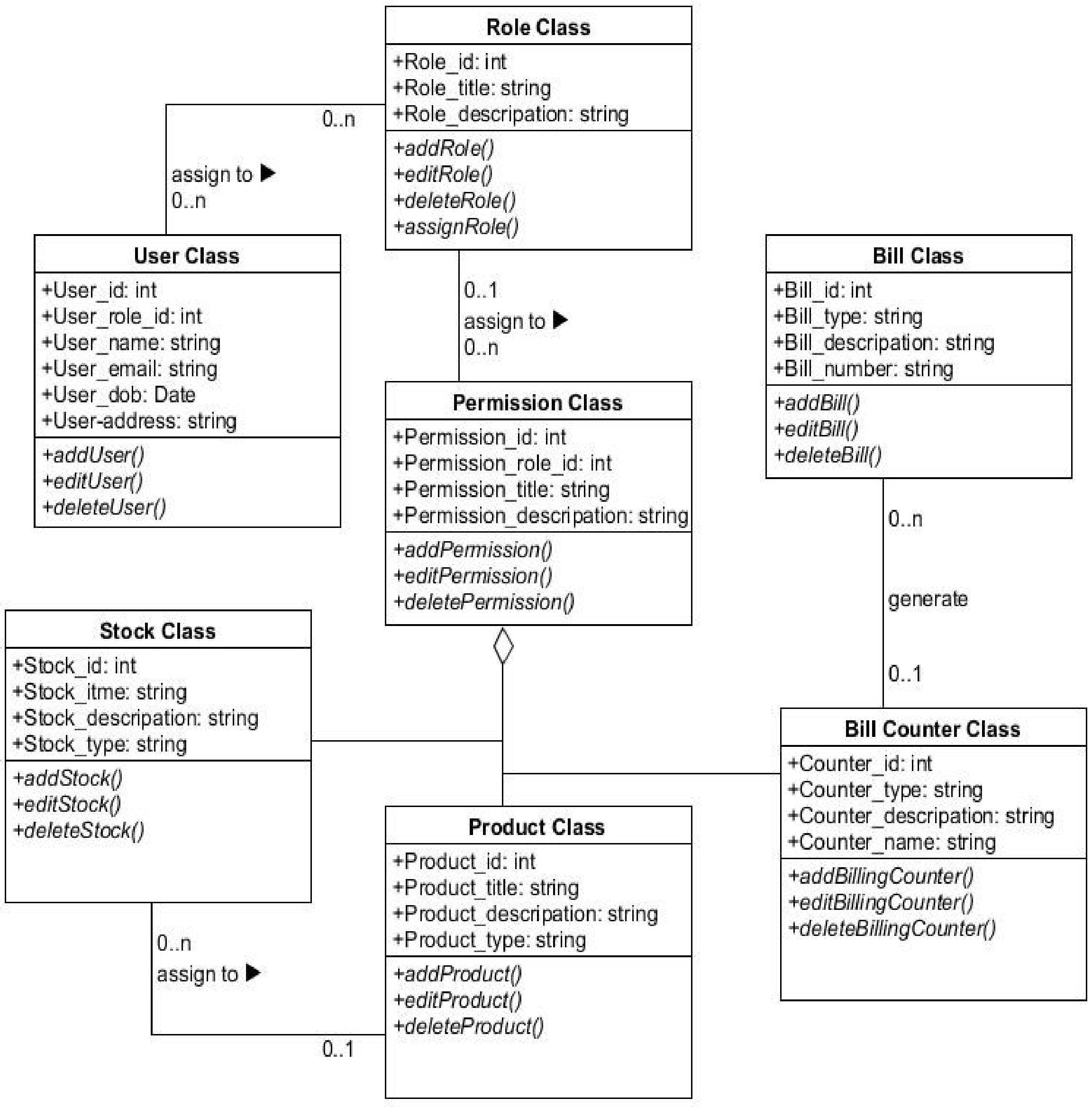
3. Design

3.1 Use Case Diagram



Usecase Diagram of General Store Management System

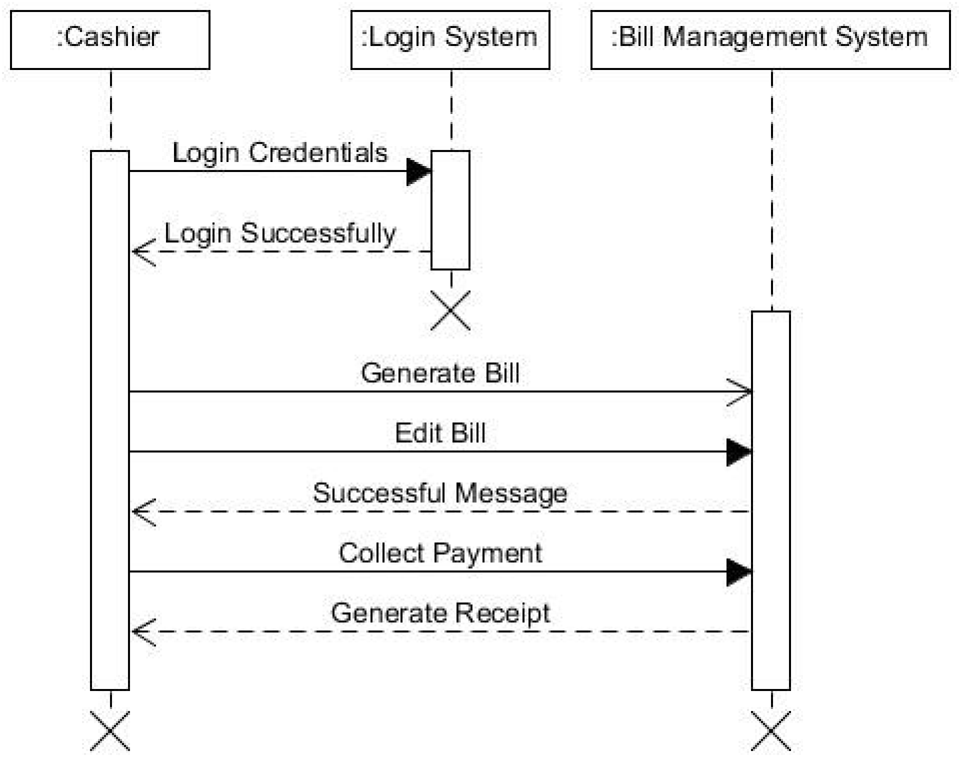
3.2 Class Diagram



Class Diagram of General Store Management System

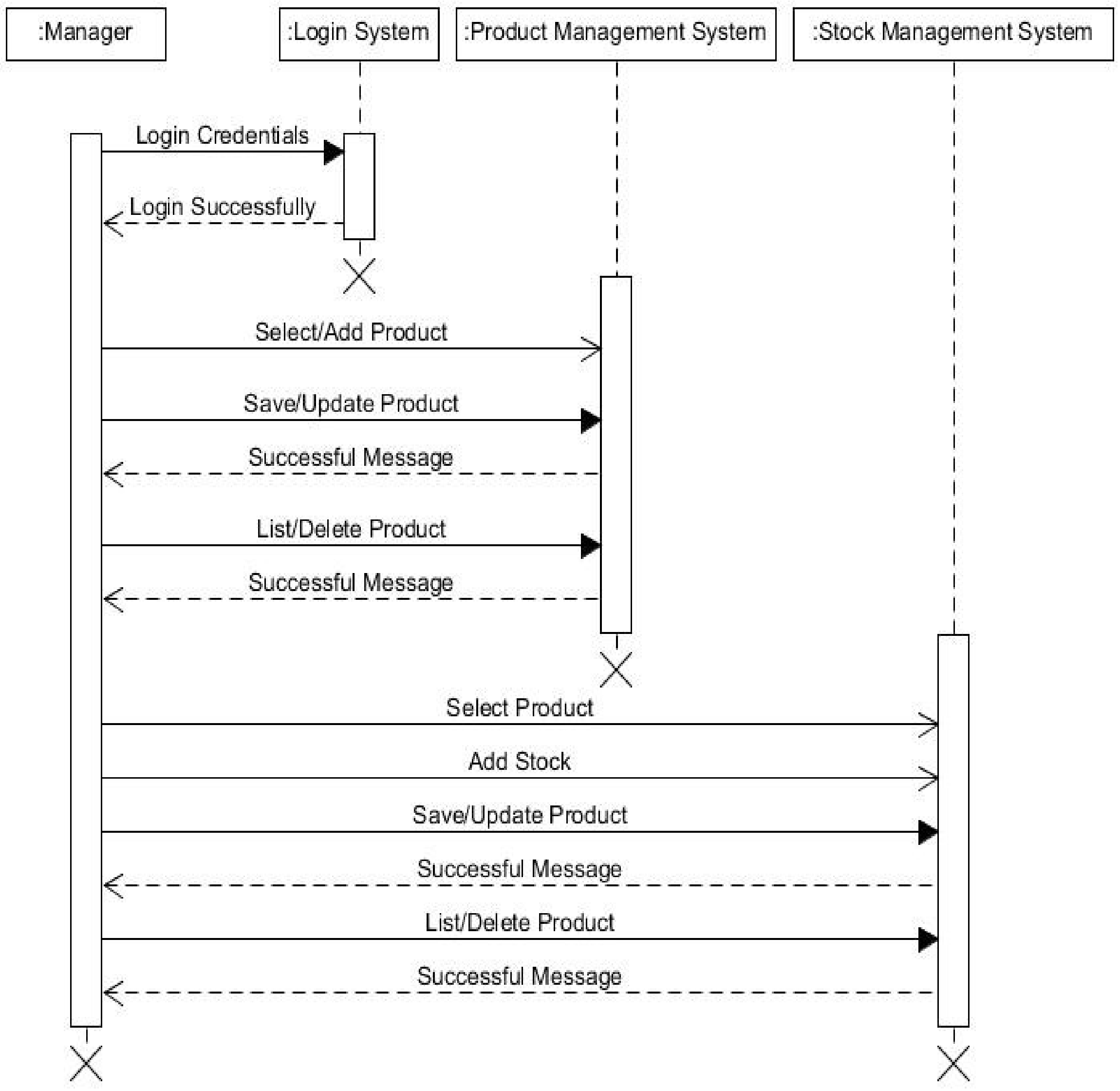
3.3 Sequence Diagram

1. Cashier Sequence Diagram



Cashier Sequence Diagram of General Store Management System

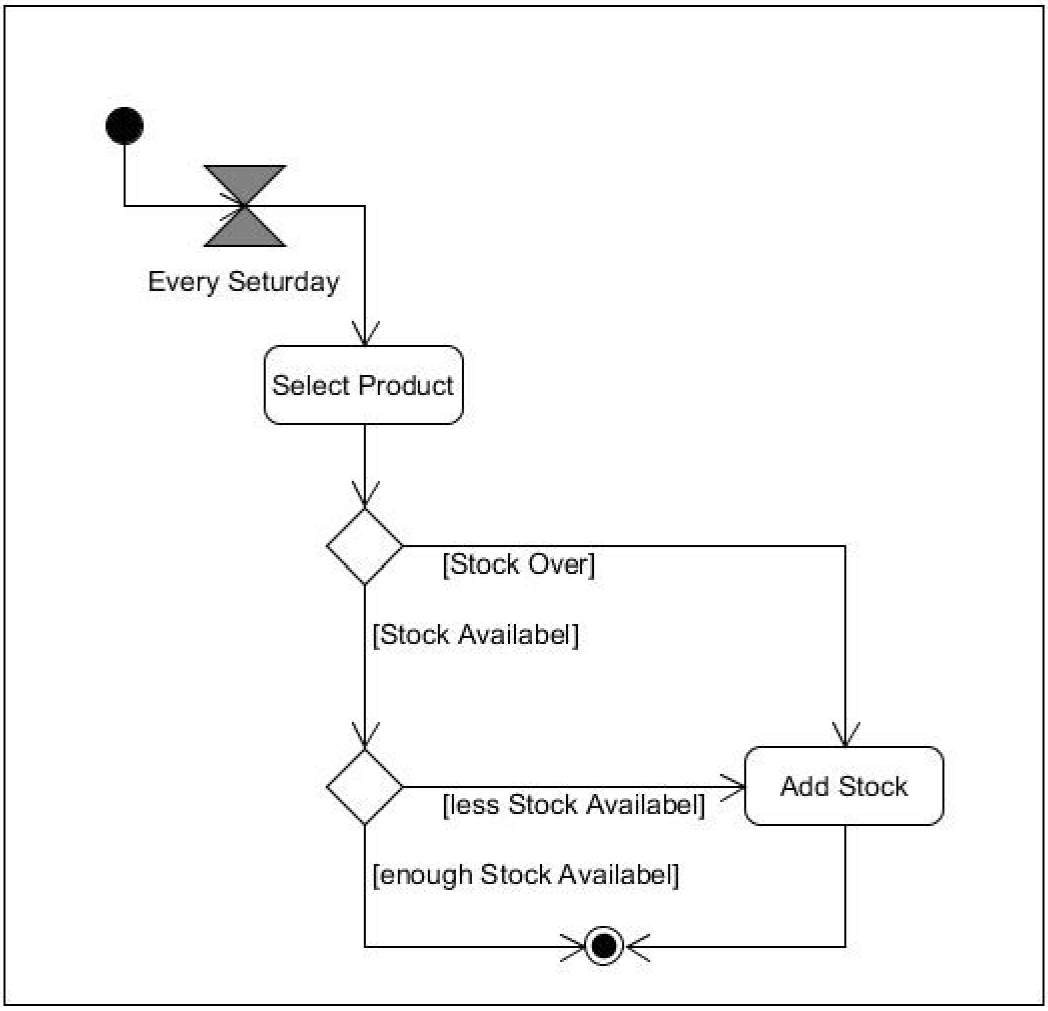
2. Manager Sequence Diagram



Manager Sequence Diagram of General Store Management System

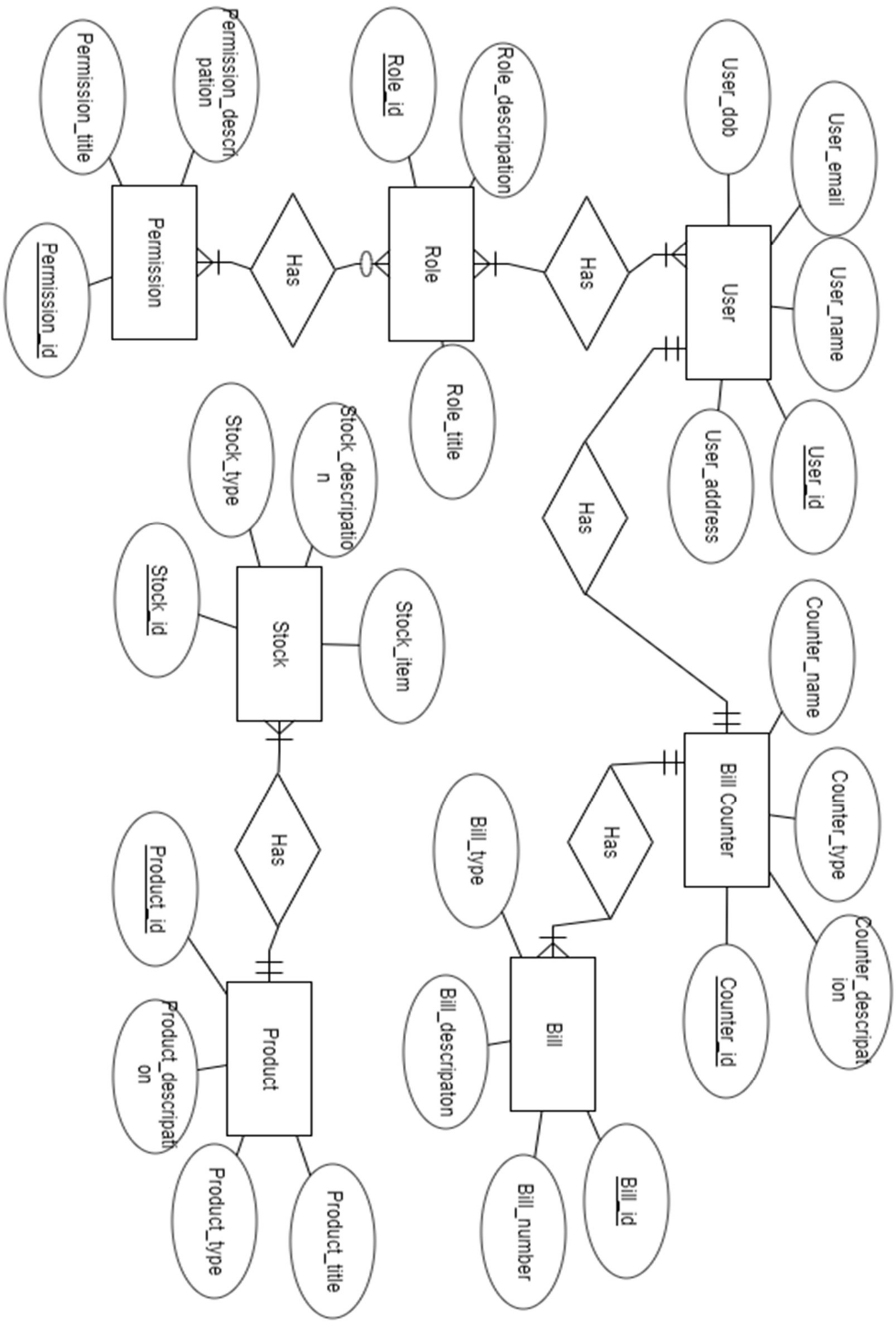
3.4 Activity Diagram

1. Stock Management Activity Diagram



Stock Management Activity Diagram of General Store Management System

3.5 ER Diagram



4. Implementation Details

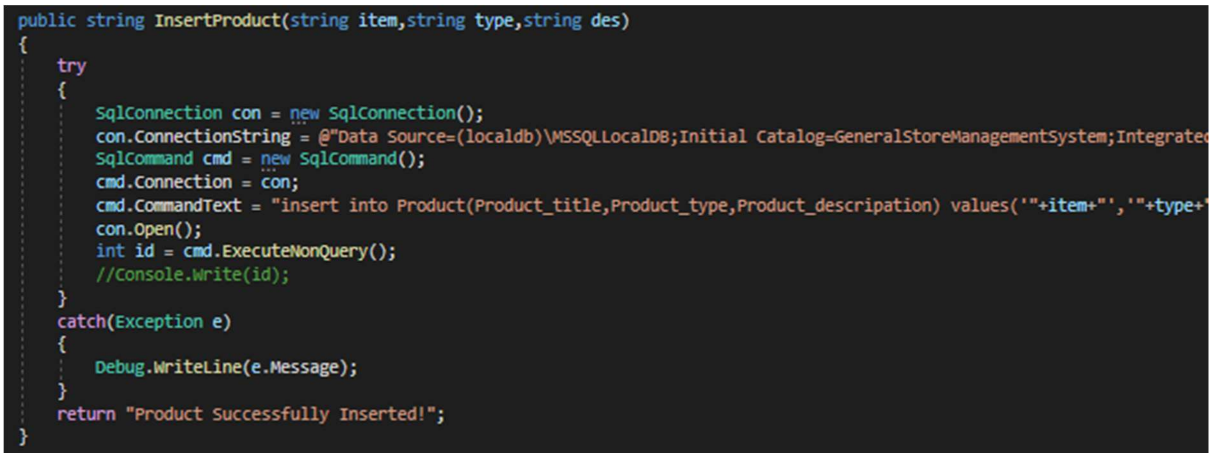
4.1 Description of Modules

All the modules described here require appropriate credentials to login and use the mentioned features which leads to access of system by authorized and authenticated users only.

1. **Manager**
   In this module, Managers can add new products, update the Products and remove the products also generate barcode for the new product and Check the status of stock in the store.

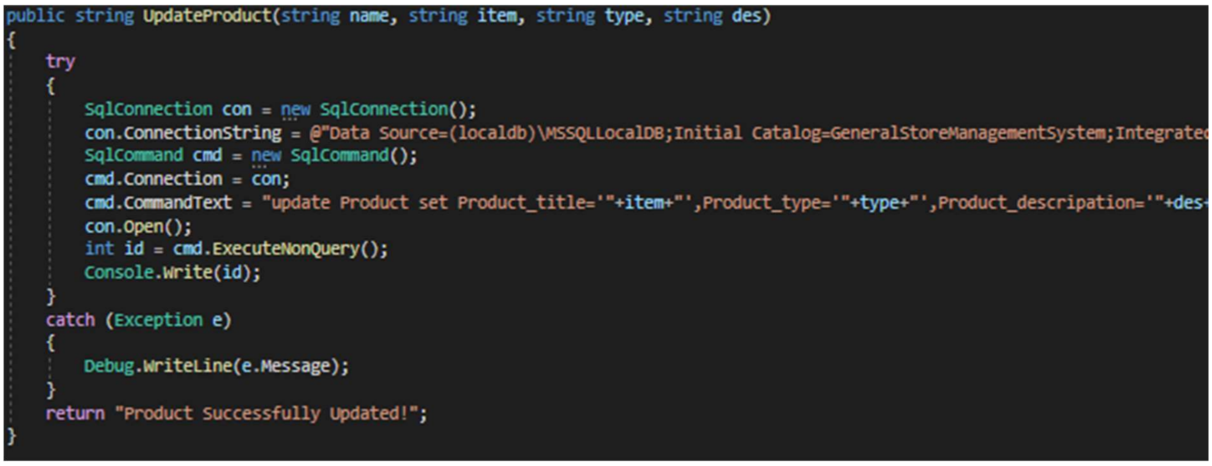
2. **Cashier**
   In this module, the cashier can manage the orders and make bills for the products.

4.2 Function Prototypes



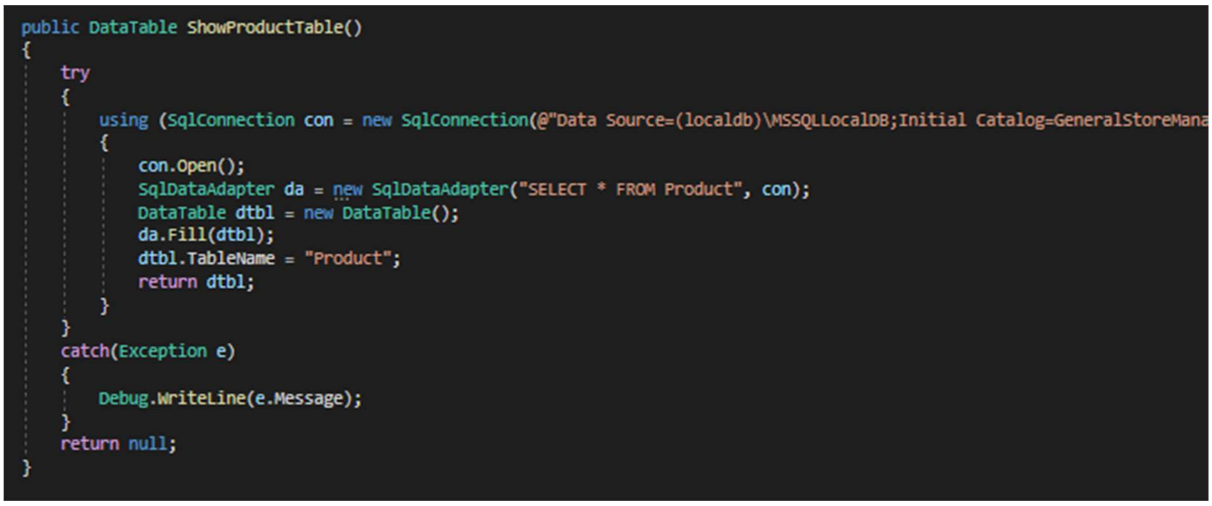
```
public string InsertProduct(string item,string type,string des)
{
    try
    {
        SqlConnection con = new SqlConnection();
        con.ConnectionString = @"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=GeneralStoreManagementSystem;Integrated
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = con;
        cmd.CommandText = "insert into Product(Product_title,Product_type,Product_descripation) values('"+item+"','"+type+"
        con.Open();
        int id = cmd.ExecuteNonQuery();
        //Console.Write(id);
    }
    catch(Exception e)
    {
        Debug.WriteLine(e.Message);
    }
    return "Product Successfully Inserted!";
}
```

Function Prototype 1: Insertion of Product by Manager



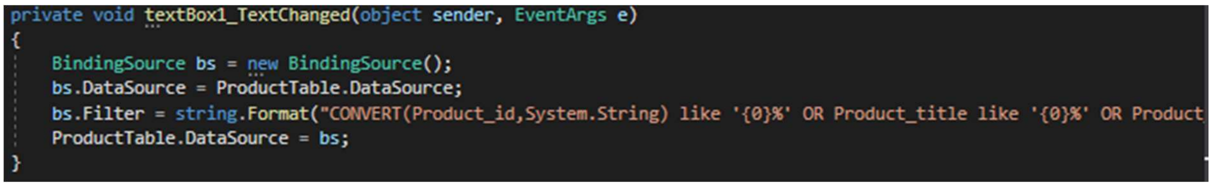
```
public string UpdateProduct(string name, string item, string type, string des)
{
    try
    {
        SqlConnection con = new SqlConnection();
        con.ConnectionString = @"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=GeneralStoreManagementSystem;Integrated
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = con;
        cmd.CommandText = "update Product set Product_title='"+item+"',Product_type='"+type+"',Product_descripation='"+des+
        con.Open();
        int id = cmd.ExecuteNonQuery();
        Console.Write(id);
    }
    catch (Exception e)
    {
        Debug.WriteLine(e.Message);
    }
    return "Product Successfully Updated!";
}
```

Function Prototype 2: Update of Product by Manager



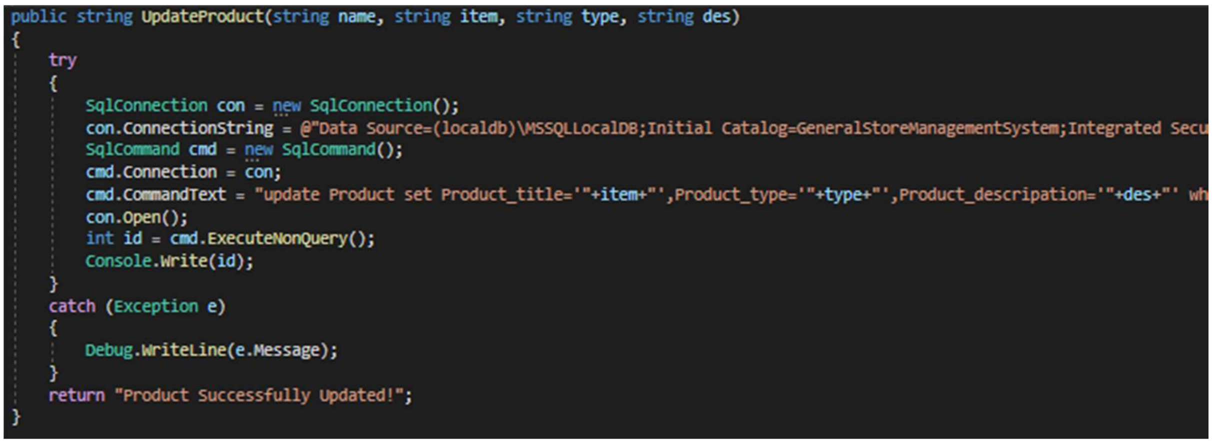
```
public DataTable ShowProductTable()
{
    try
    {
        using (SqlConnection con = new SqlConnection(@"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=GeneralStoreMana
        {
            con.Open();
            SqlDataAdapter da = new SqlDataAdapter("SELECT * FROM Product", con);
            DataTable dtbl = new DataTable();
            da.Fill(dtbl);
            dtbl.TableName = "Product";
            return dtbl;
        }
    }
    catch(Exception e)
    {
        Debug.WriteLine(e.Message);
    }
    return null;
}
```

Function Prototype 3: Display of Product Table using SqlDataAdapter



```
private void textBox1_TextChanged(object sender, EventArgs e)
{
    BindingSource bs = new BindingSource();
    bs.DataSource = ProductTable.DataSource;
    bs.Filter = string.Format("CONVERT(Product_id,System.String) like '{0}%' OR Product_title like '{0}%' OR Product
    ProductTable.DataSource = bs;
}
```

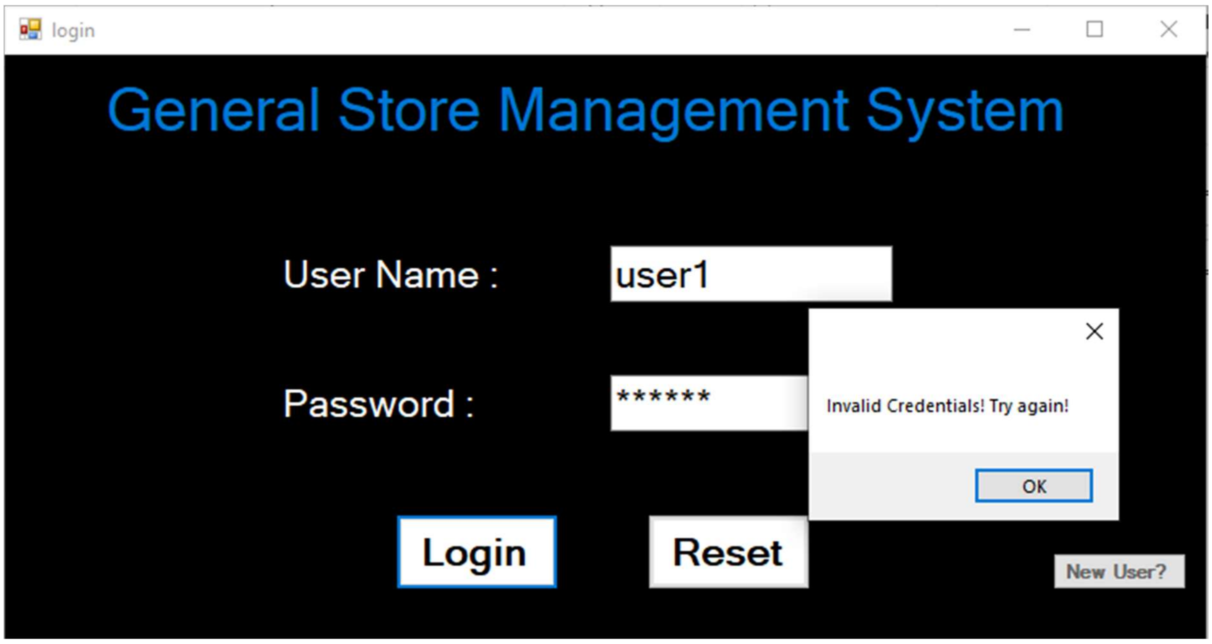
Function Prototype 4: Live search based on any column entry



```
public string UpdateProduct(string name, string item, string type, string des)
{
    try
    {
        SqlConnection con = new SqlConnection();
        con.ConnectionString = @"Data Source=(localdb)\MSSQLLocalDB;Initial Catalog=GeneralStoreManagementSystem;Integrated Secu
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = con;
        cmd.CommandText = "update Product set Product_title='"+item+"',Product_type='"+type+"',Product_descripation='"+des+"' wh
        con.Open();
        int id = cmd.ExecuteNonQuery();
        Console.Write(id);
    }
    catch (Exception e)
    {
        Debug.WriteLine(e.Message);
    }
    return "Product Successfully Updated!";
}
```

Function Prototype 5: Deletion of Product Table by Manager

5. Testing


login

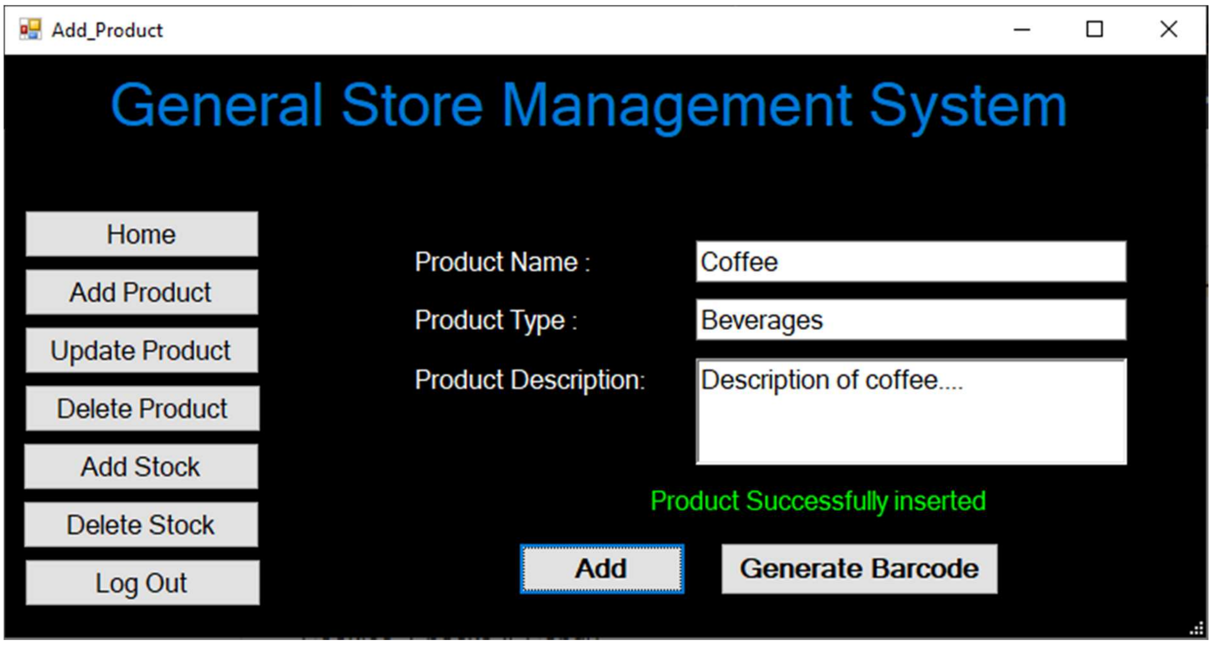
General Store Management System

User Name : user1

Password : ******

Invalid Credentials! Try again!

Testing 1: Using incorrect credentials to login


Add_Product

General Store Management System

Product Name : Coffee

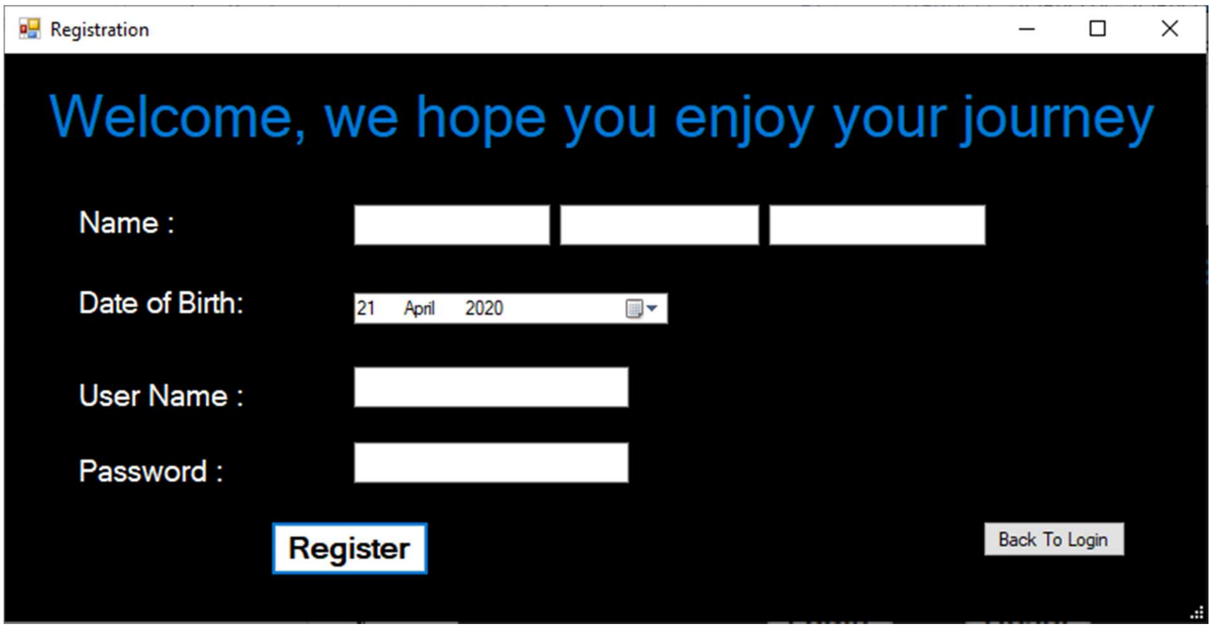
Product Type : Beverages

Product Description: Description of coffee....

Product Successfully inserted

Testing 2: Successful addition of a product

6. Screenshots


Registration

Welcome, we hope you enjoy your journey

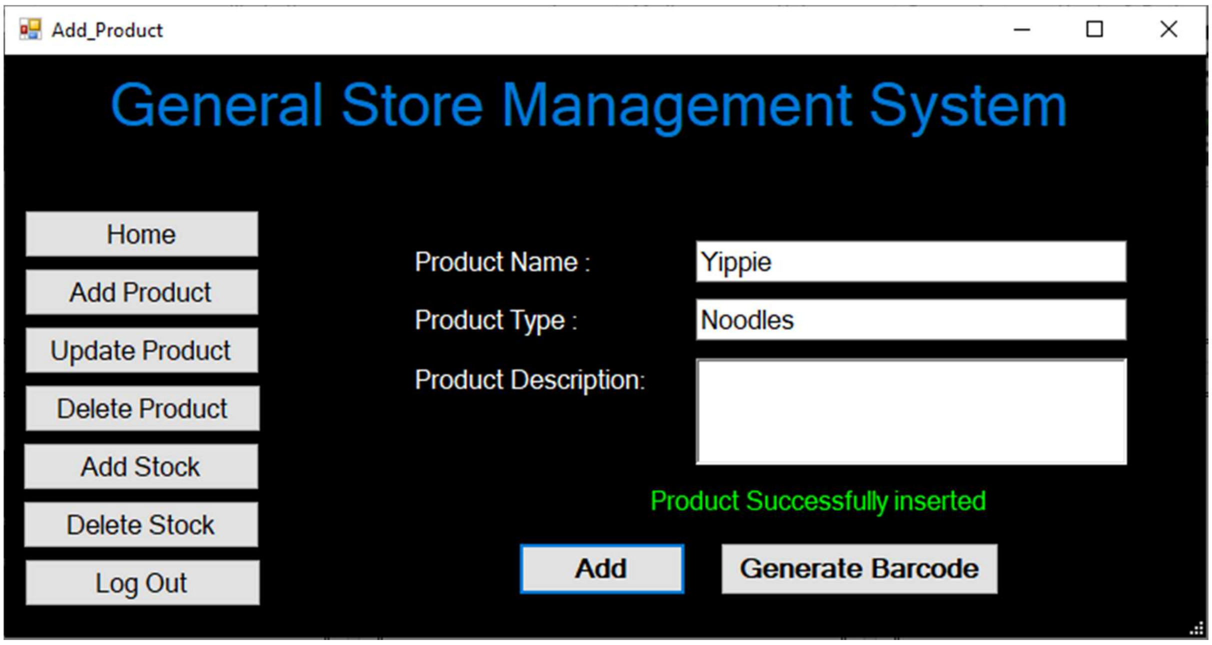
Name :

Date of Birth: 21 April 2020

User Name :

Password :

Screenshot 1: Registration Page for Manager


Add_Product

General Store Management System

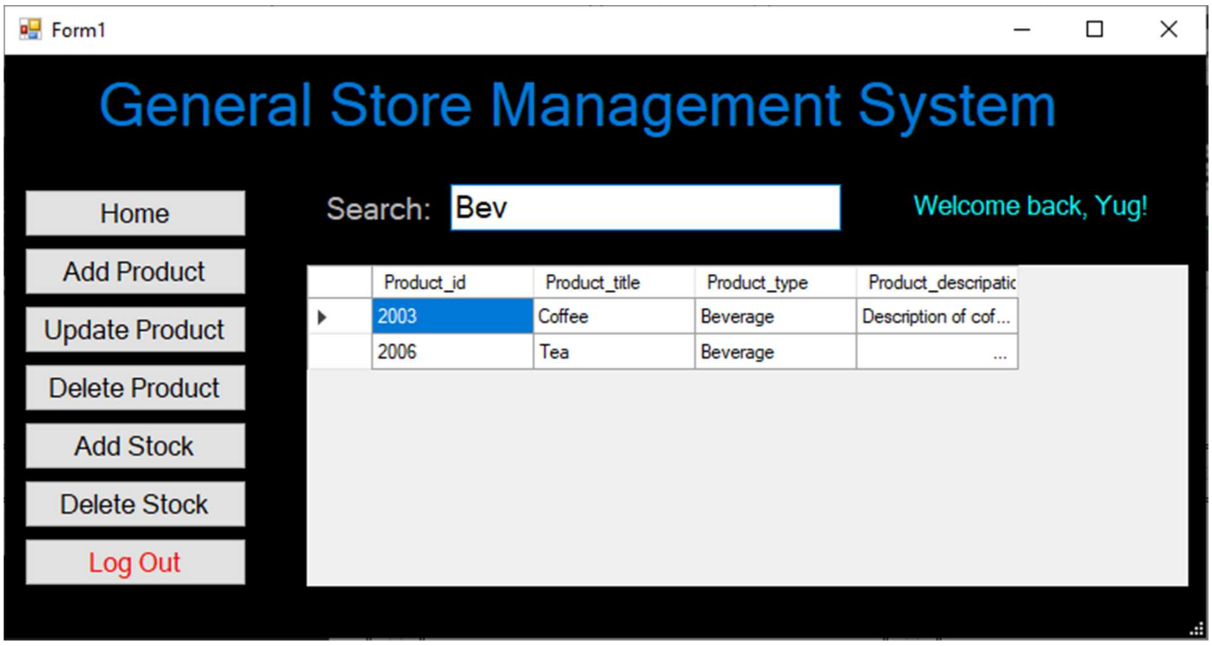
Product Name : Yippie

Product Type : Noodles

Product Description:

Product Successfully inserted

Screenshot 2: Addition of New Product by Manager


Form1

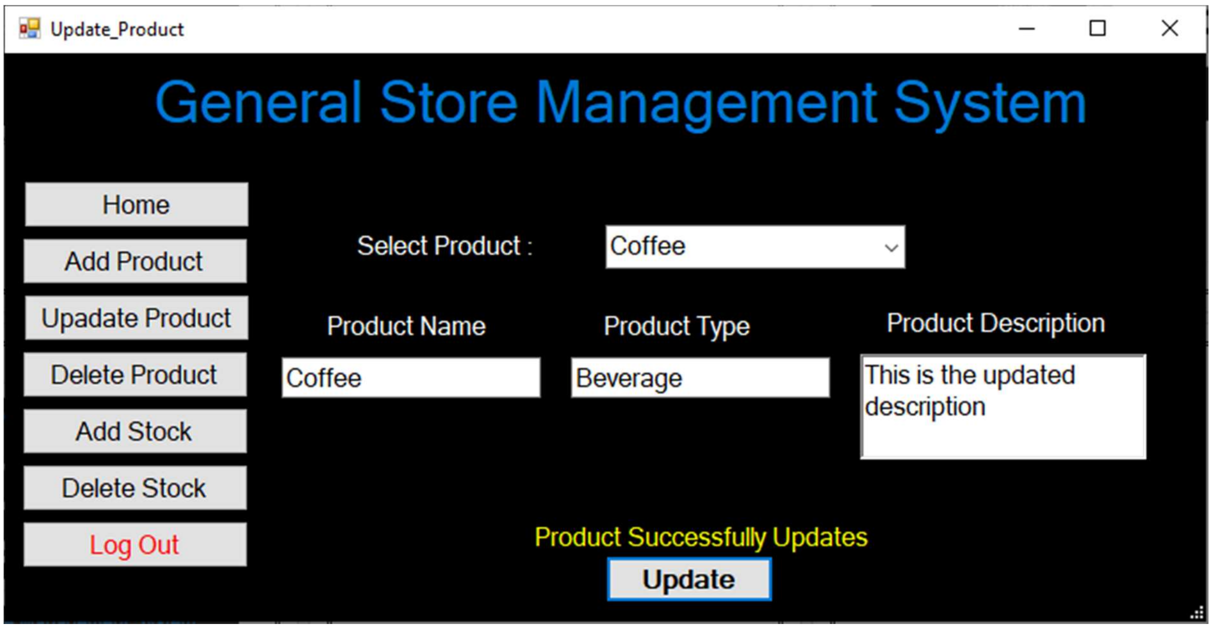
General Store Management System

Search: Bev

Welcome back, Yug!

Product_id	Product_title	Product_type	Product_descripatic
2003	Coffee	Beverage	Description of cof...
2006	Tea	Beverage	...

Screenshot 3: Search Results Live for keyword “bev” based on Type


Update_Product

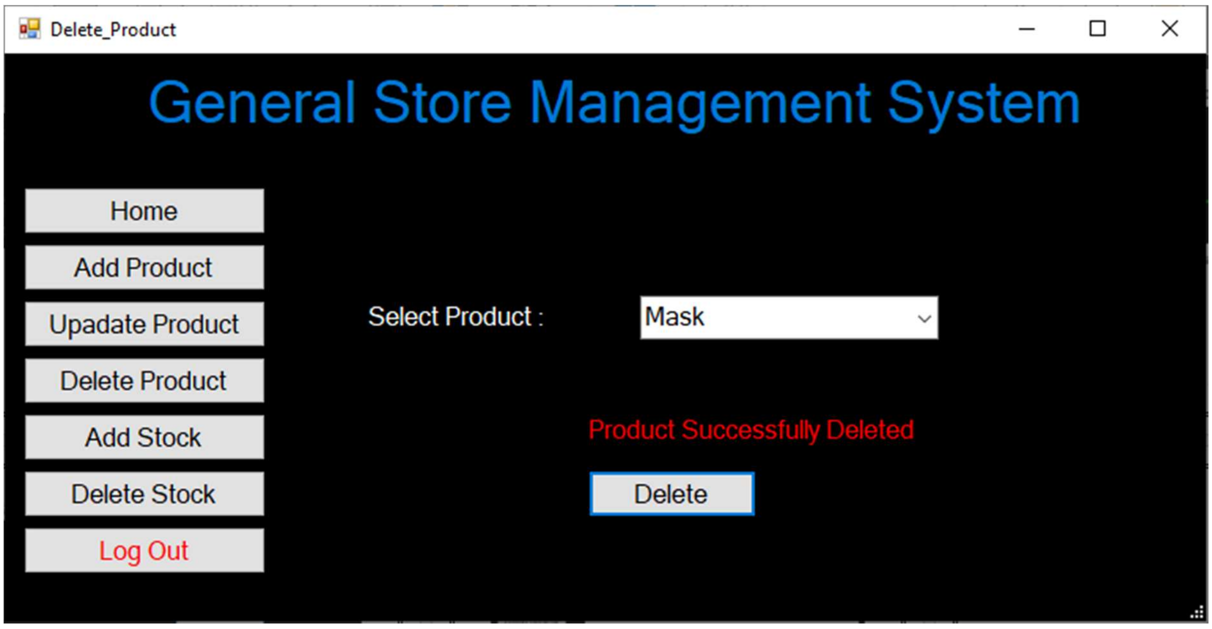
General Store Management System

Select Product : Coffee

Product Name	Product Type	Product Description
Coffee	Beverage	This is the updated description

Product Successfully Updates

Screenshot 4: Update in description of a Product

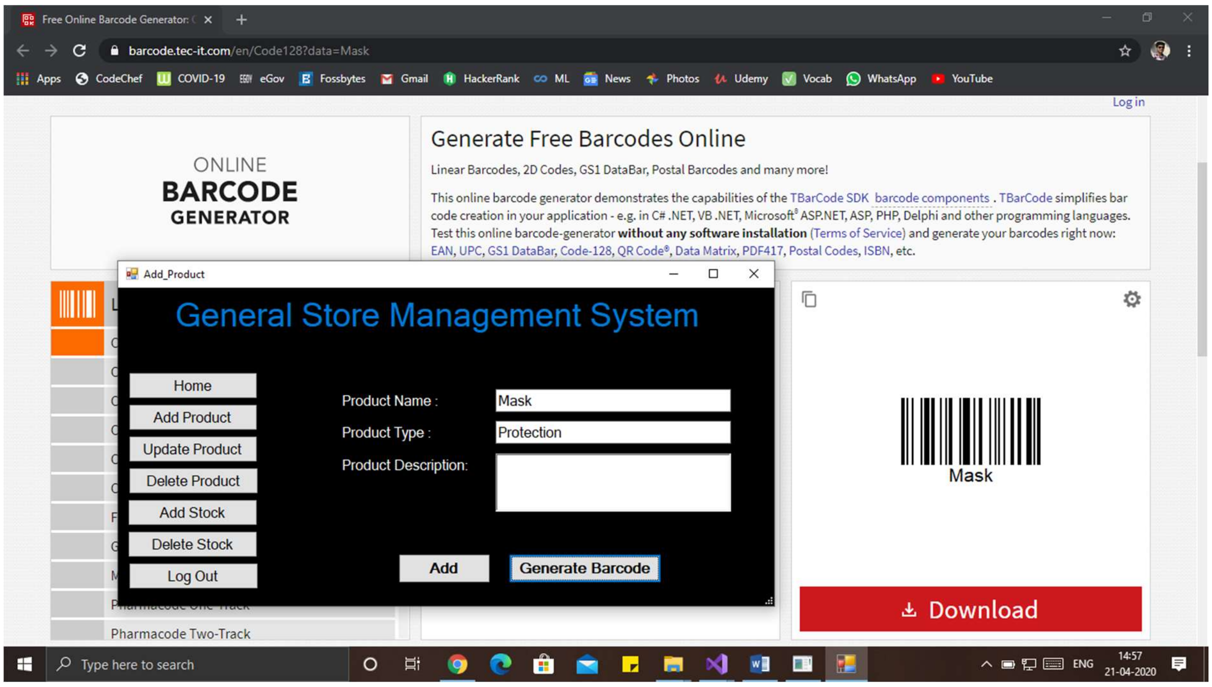


General Store Management System

Select Product : Mask

Product Successfully Deleted

Screenshot 5: Deletion of Product by Manager



Generate Free Barcodes Online

Linear Barcodes, 2D Codes, GS1 DataBar, Postal Barcodes and many more!

This online barcode generator demonstrates the capabilities of the TBarCode SDK barcode components. TBarCode simplifies bar code creation in your application - e.g. in C# .NET, VB .NET, Microsoft® ASP.NET, ASP, PHP, Delphi and other programming languages. Test this online barcode-generator **without any software installation** (Terms of Service) and generate your barcodes right now: EAN, UPC, GS1 DataBar, Code-128, QR Code®, Data Matrix, PDF417, Postal Codes, ISBN, etc.

General Store Management System

Product Name : Mask

Product Type : Protection

Product Description:

Screenshot 6: Generation of Barcode for a Product

7. Conclusion

The functionality implementation in the system was done after understanding all the system modules according to their particular requirements.

Functionalities that are successfully implemented are:

- Login
- Registration
- Manage Products
- View Products
- Live Search Product based on any column
- Generate Barcode for a product

After the successful implementation of the system along with all the modules and functionalities, comprehensive testing was performed to determine the loopholes and possible flaws/errors in the system.

8. Limitations and Future Extension

8.1 Limitations

- Manager is required to handle the app.
- Manager should have basic knowledge to handle the app.
- Product Types are not further divided into sub-types.
- It is Desktop Based Hardware Dependent.

8.2 Future Extension

- Mobile based notification alerts on low stocks.
- Email and SMS based OTP verification for login and registration.
- Generation of PDF Bill with digital signature.

9. Bibliography

- Stack Overflow:

  https://stackoverflow.com/

- Microsoft Docs:

  https://docs.microsoft.com/en-us/dotnet/framework/winforms/

- C Sharp Corner:

  https://www.c-sharpcorner.com/technologies/windows-forms

- Wikipedia:

  https://en.wikipedia.org/wiki/Windows_Forms