

Acknowledgement

The success and final outcome of this assignment what required a lot of guidance and assistance from many people and I extremely fortunate to have got this all along the completion of my assignment work whatever we have done is only due to search guidance and assistant, and I will not forget to thank them I respect and appreciate our lecturer Mrs.Upeka for giving me an opportunity to do this assignment work and providing me with all support and guidance which made me complete the assignment on time and I extremely grateful to him for providing such an excellent support and guidance.

Contents

Acknowledgement	4
Summery	6
Task 01	7
Common Type System & Common Language Specification	7
Task 02	8
Visual studio	8
C# Programming language	10
Task 03	11
System Architecture diagram	11
Sequence diagram	11
Use case diagram	18
Class diagram	19
ER diagram	20
UI design	21
Task 04	29
Source codes	29
Task 05	36
Test plan	36
Test case	38
Task 06	58
Deploy User Application or Component Using Windows Installer	58
Conclusion	60
Strength	60
Weakness	60
Future recommendation	60
Referencing	61

Summery

Black Lotus is a famous flower supplier in Sri Lanka. BL owns the company's flower sales outlets, and many affiliate sellers buy and resell BL flowers. BL Main Office maintains Excel spreadsheets to keep track of orders and flower inventory. Current connections and recordkeeping methods have caused many problems with the BL, such as misplaced orders, repeated reservations, supply delays, etc. The company's management decided to implement a sale and inventory management system to overcome these issues and improve service and profit.

In the first task has been explained about what is standard language specification and common type system as well.

In the second task has been explained about the visual studio and the importance of the visual studio for the users and how easy to use the tool without having much work, as well as the user-friendliness about the tool and furthermore in document, has explained about the c# language and importance about the language and features of the system as well.

In the third task have provided the system architecture diagram and UML diagrams and Data model design (ER diagram) and finally UI designs of the system as well.

In the fourth task have Documented the main functionalities with source code, interface screenshots with proper explanation.

In the fifth task have provided a proper test plan as well as test cases with the test result. In the sixth task Lastly, I have provided references according to Harvard referencing.

Task 01

Common Type System & Common Language Specification

CLS is the variety of the principles and requirements that each language must pursue. The CLS characterizes the sorts permitted in external commands, however the standards for utilizing them, depending upon the objective of the client. Casually CLS is an essentially an agreement between programming language designers and class library creators. The CLS is essentially only a subset of the whole arrangement of feature supported by the CLR.

The CLS incorporates things, for example, calling virtual techniques, overloading strategies and does exclude things, for example, unsigned varieties. CLS is weighted vigorously for the library fashioners. CLS determines the language compiler must firmly grasp the many decisions to make.

Compliant compiler needs essentially to cling to these standards and Despite the fact that the CLI doesn't expect compilers to pursue CLS, code that follows the CLS rules is perfect with every all other language that keep the CLS rules.

Common type system

NET implementation is language independent. This doesn't just mean that programmers can write code in any language that can be compiled to IL. This also means that designer needs to be able to interact with code written in other languages that can be used in a .NET implementation.

In order to do this transparently, there has to be a common way to describe all supported types. This is what the Common Type System (CTS) is in charge of doing. It was made to do several things:

- Set up a system for cross-language execution.
- Give an object-oriented model to help actualizing different languages on a .NET execution.
- Characterize a lot of rules that all languages must pursue with regards to working with types.
- Give a library that contains the fundamental primitive types that are utilized in application advancement, (for example, Boolean, Byte, Char and so forth).

CTS specifies two main types that should be supported: reference and value types and their names indicate their definitions.

Reference types' objects are spoken to by a reference to the objects real worth and It basically relates to a memory space where the objects' values are. This profoundly affects how these types are utilized. In the event that you allocate a reference type to a variable for example, any progressions to the object will be considered the principle object there is no duplicating.

CTS define verities of type such as

- Classes
- Structures
- Interfaces
- Delegates

Task 02

Visual studio

The Visual Studio Integrated Development Environment is a creative launchpad that can use to edit, debug, and generate code before publishing the application. The integrated advanced development environment is an element-rich program that can be used for some parts of programming improvements.

Visual Studio with an open project and various key tool windows Users likely use such as:

Solution Explorer, you can view, navigate, and manage code files. Solution Explorer can help organize code by grouping files into solutions and projects.

The user may spend most of the time in the editor window displaying the contents of the file. Users can edit code or design user interfaces, such as windows with buttons and text boxes.

Team Explorer, users can use version control technologies such as Git and Team Foundation version control to track work items and share code with others.

Visual Studio is accessible on both Windows and Mac. At the same time, Visual Studio for Mac has many similar highlights provided by Visual Studio 2019 and enhances the capabilities of both mobile and mobile applications. This article is organized around the Windows form of Visual Studio 2019.

features

quick action and squiggles

Styles are wavy underlines that alert you to errors or possible code problems as you type. These intuitive clues let you solve problems right away without having to wait for mistakes to be found during the build process or while running the program. If you hover over the squiggle and user will see additional information about the error. The lamp may also appear in the left margin with a procedure called "Quick Action" to correct the error.

code clean up

With a click of a button, format your code and apply any proposed software fixes through code style settings, editorconfig agreements, and Roslyn parsers. Code cleaning helps you solve problems in the code before moving to code review and its currently only available for C #.

Refactoring

refactoring includes operations such as the smart renaming of variables, extracting one or more lines of code in a new method, changing the order of method parameters, and more.

IntelliSense

IntelliSense is a number of highlighted terms that can legally display data about your code in editing and sometimes write a limited amount of code for the user. Much like the inline base archive in the user, this saves the user from having to search for type data elsewhere. The usefulness of IntelliSense varies by language. For more data, see C # IntelliSense, Visual C ++ IntelliSense, JavaScript IntelliSense, and Visual Basic IntelliSense.

live share

No matter what your application type or programming language is, you can collaborate and edit and debug with others in real-time. You can immediately and securely share your project and debug sessions, terminal instances, localhost web applications, voice calls, etc. as needed.

Call Hierarchy

The Call Hierarchy window shows the techniques that call a chose strategy. This can be helpful data when user considering changing or removing the technique, or when you're attempting to find a bug.

C# Programming language

C # is a programming application generated by Microsoft that uses C # in most of its products and its Object-oriented programming language, whenever the user uses C # application, user implementing the .NET Framework as well.

C # is a programming language that was created by Microsoft in 2000 to assemble applications on their .NET stages. Even though C # hasn't been around that long, it has just earned the honour of being named the programming language of the year. Because of C # 's versatility and ability to serve small and multinational companies as well.

C# can create an important project, such as:

- Web applications
- Windows applications
- Mobile apps
- Cloud-based services
- Desktop-based applications for Windows
- Any games using Unity
- Windows services
-

C # is particularly similar to other C programming languages, such as C, C ++, and Java. If User is not familiar with these projects, then it is not difficult for user to increase user's understanding of C #. Generally, C # is an easy-to-use program that provides an unusual programming foundation. The simplicity of your learning will depend on your experience. In any case, the countdown is not really effective. The goal is to treat C # as a high-level language. And it operates similar to English. It was then planned for ease of use. C # handles it for you. For different projects, such as C ++, engineers need to focus on monitoring them. Working with C# is very easy for developers aswell.

Features

Easy to maintain

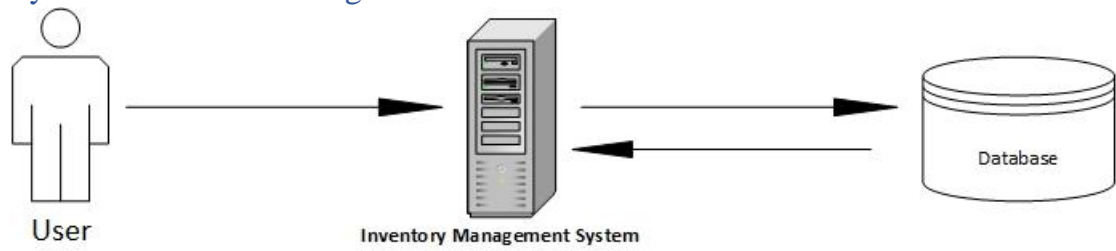
C # is a static language, which means that your code will be checked for errors before being integrated into an application.

Fast

C # is faster than a dynamically typed language because things are more clearly defined. So when the application is running, checking the definition of something in your code doesn't waste Users machine resources.

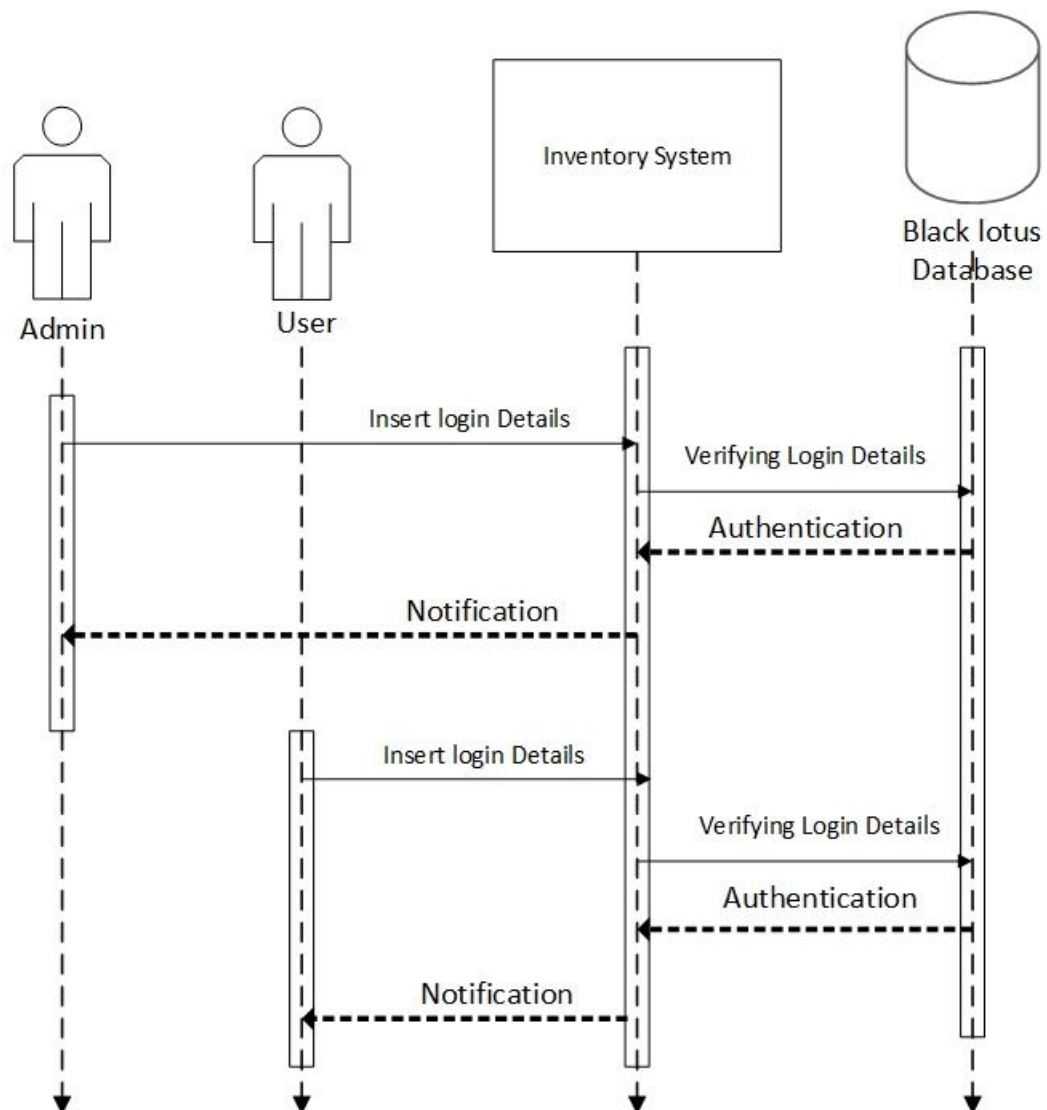
Task 03

System Architecture diagram

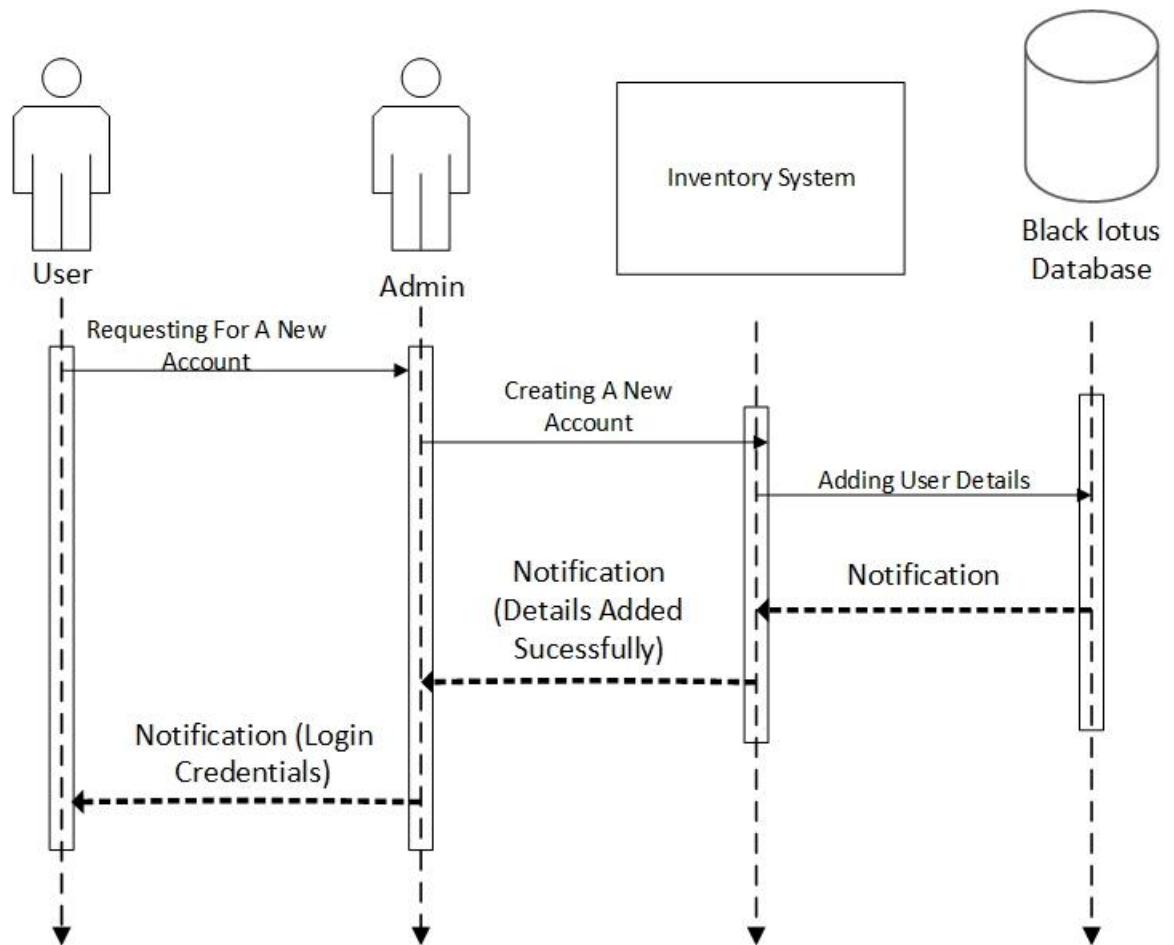


Sequence diagram

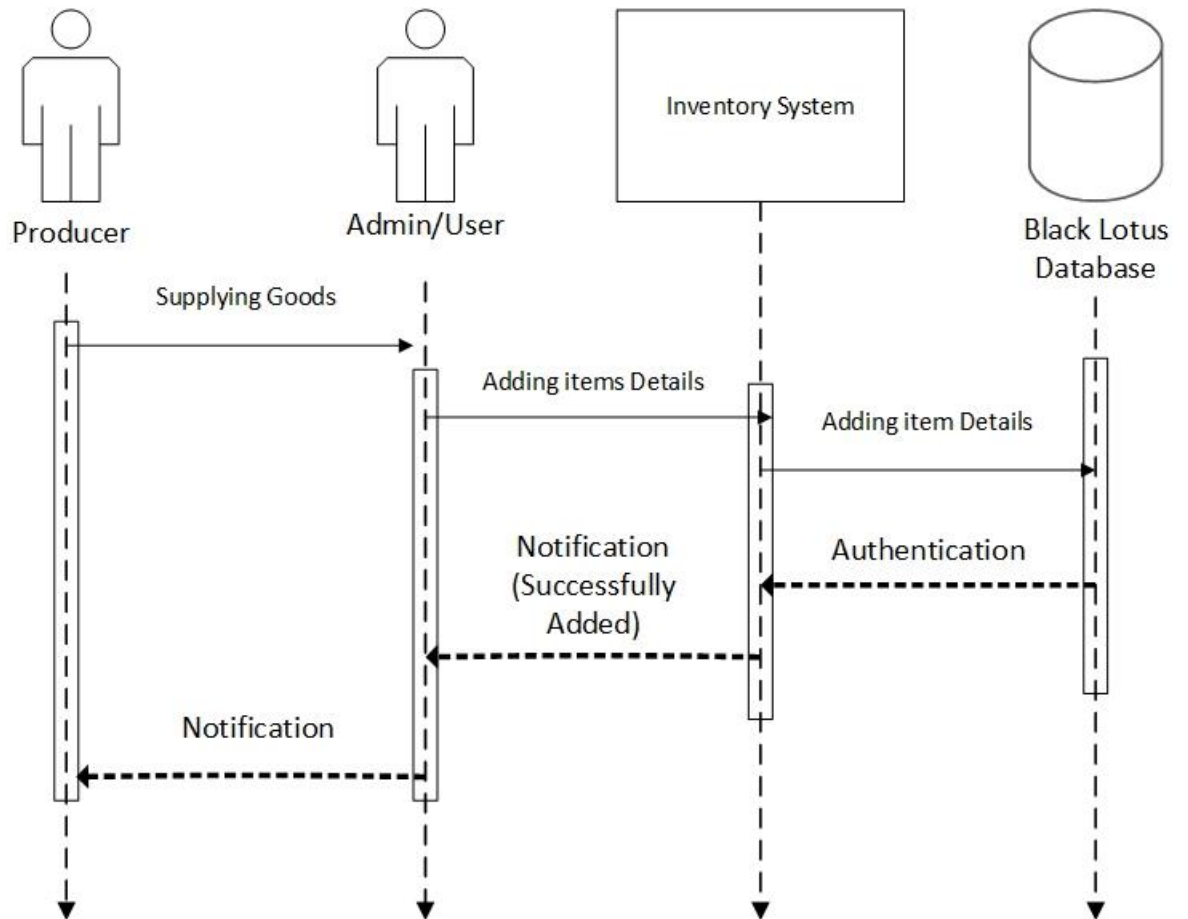
Login



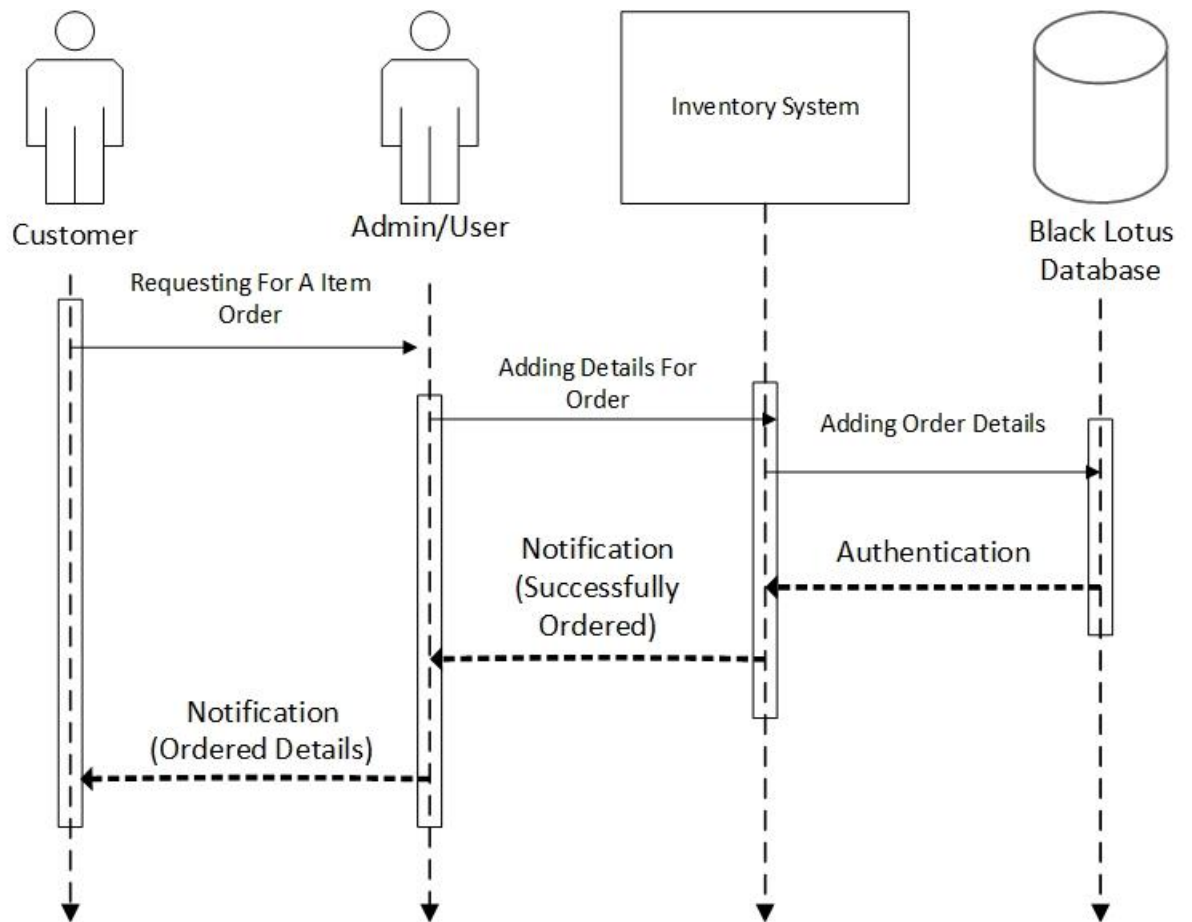
Creating account employee



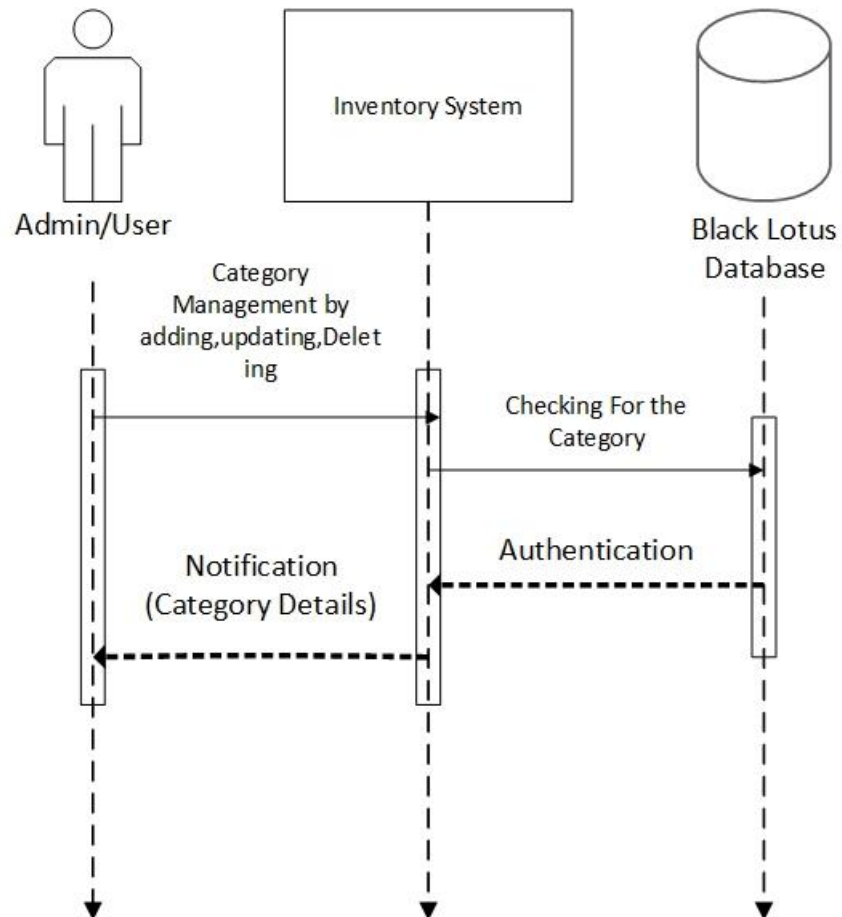
Inventory management



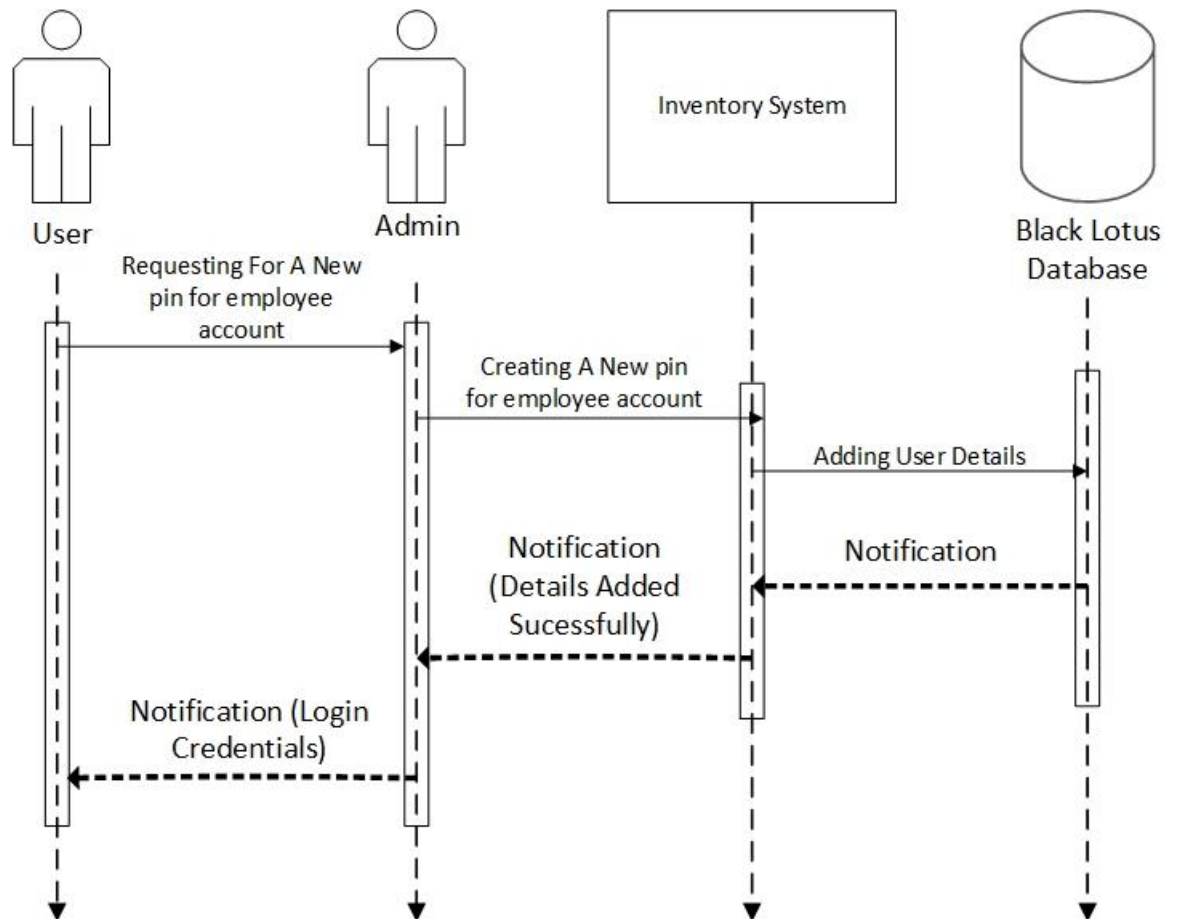
Order management



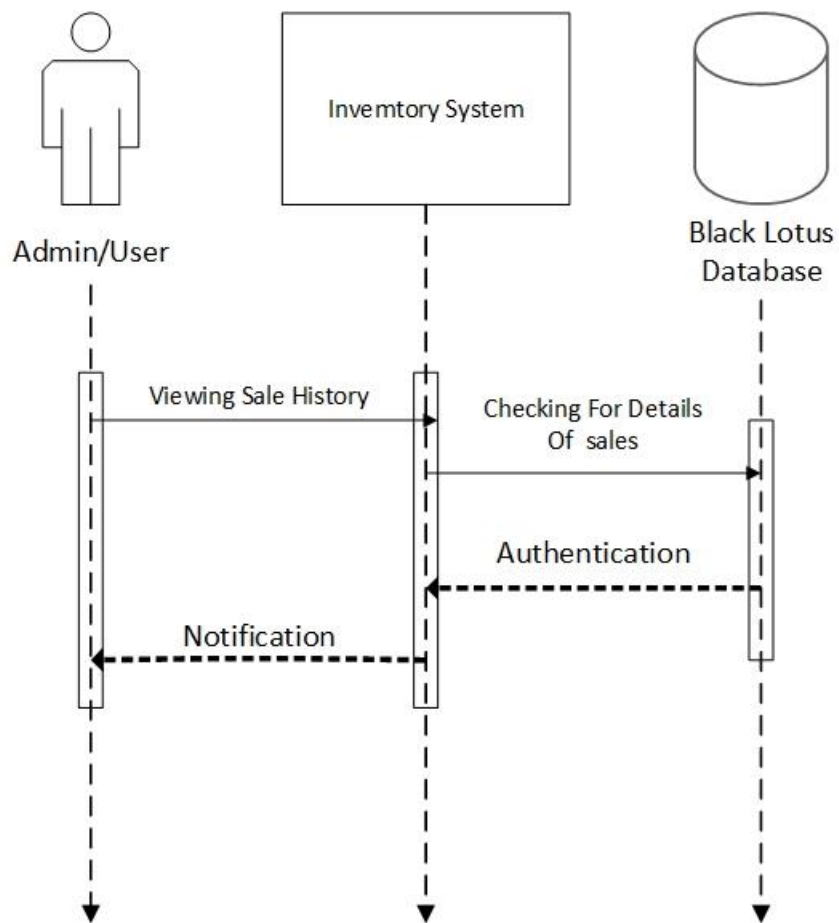
Category management



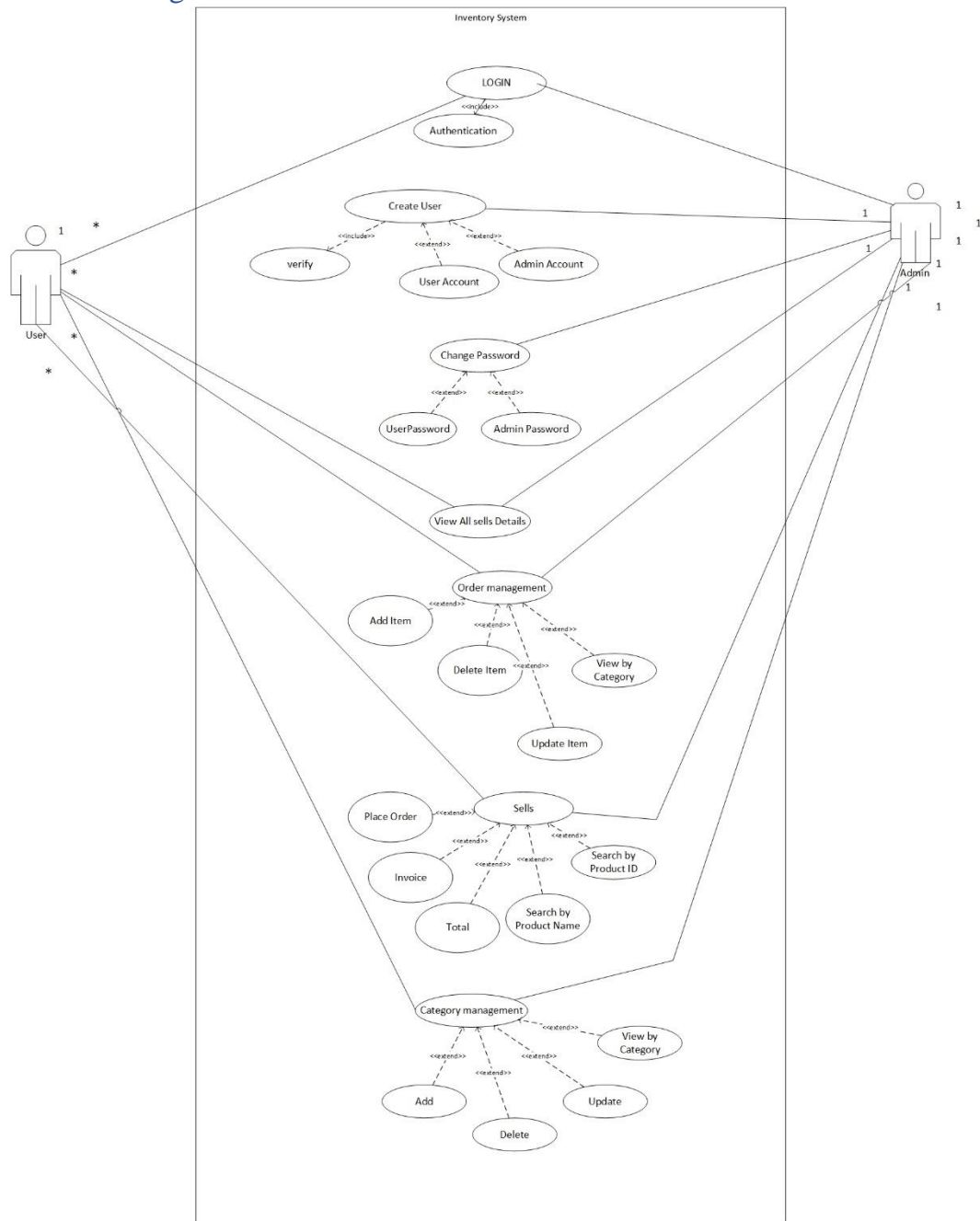
User management



Sale history

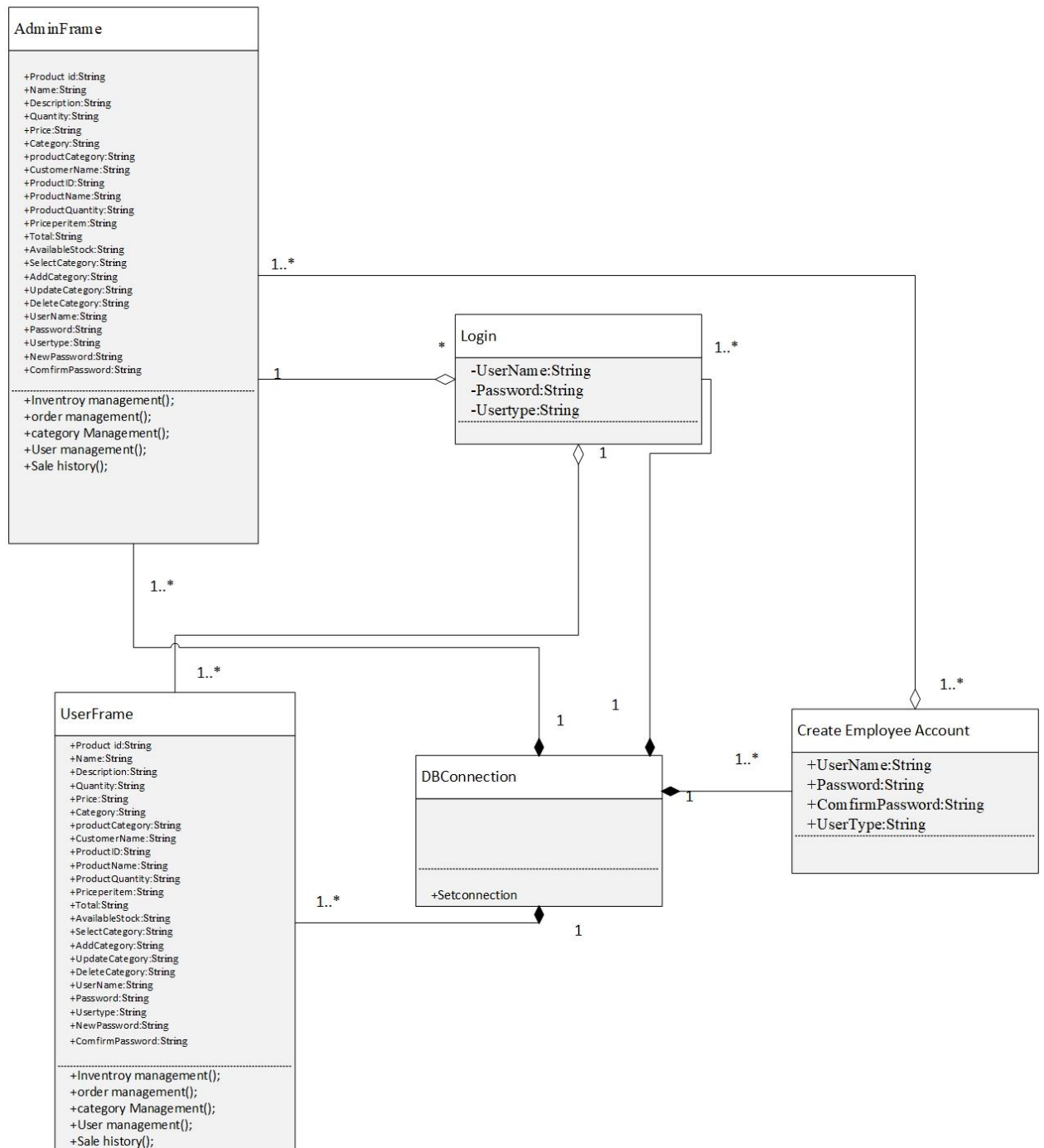


Use case diagram

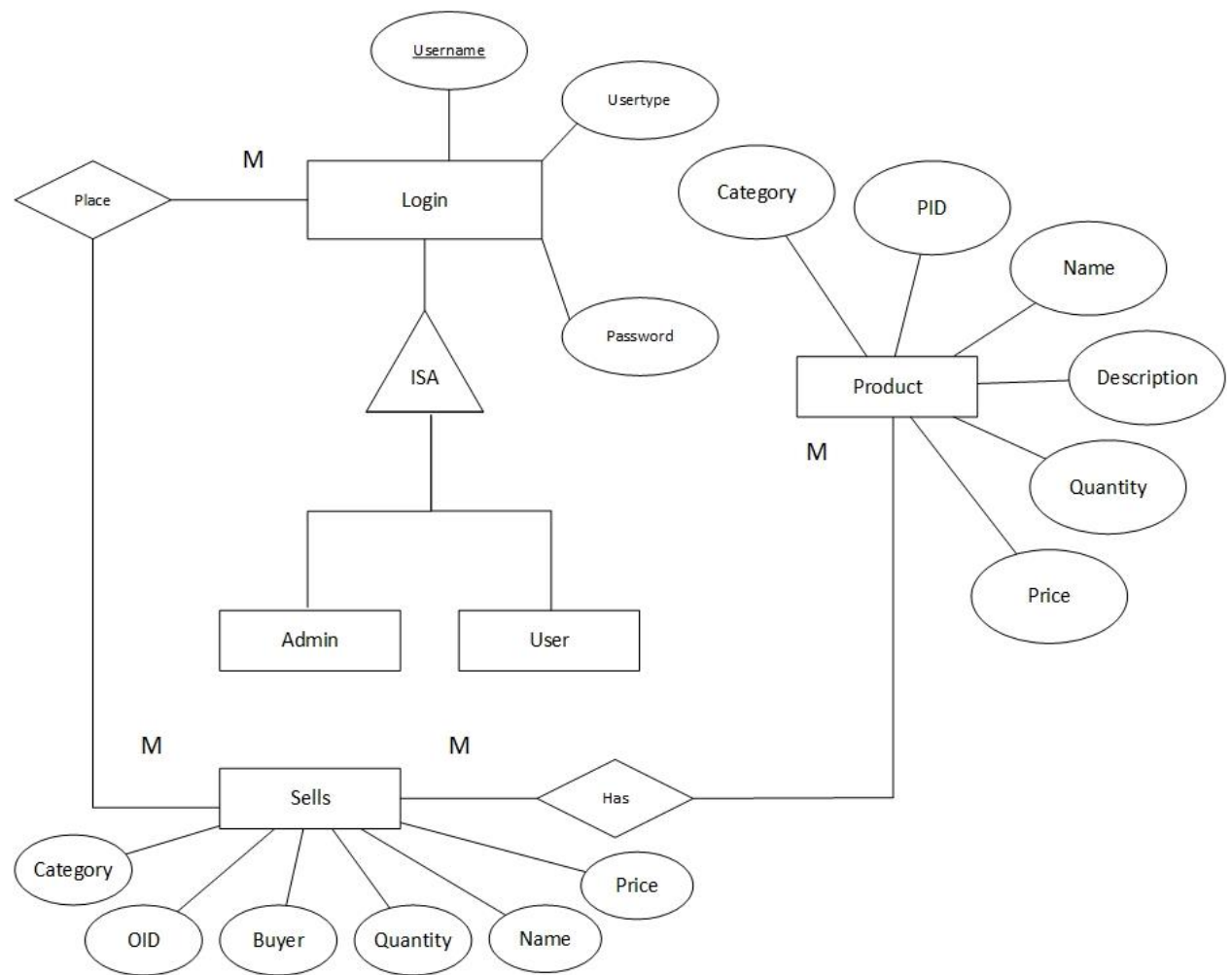


There are two actors such as User and Admin. Admin will be able to access all of the functions such as login, Inventory management, User management, create employee accounts, sale history, etc.. but there are limited functionalities in the User frame such as User cannot make any changes to any account as well as User will not be able to create any account as well because Admin have more power than User in every business.

Class diagram

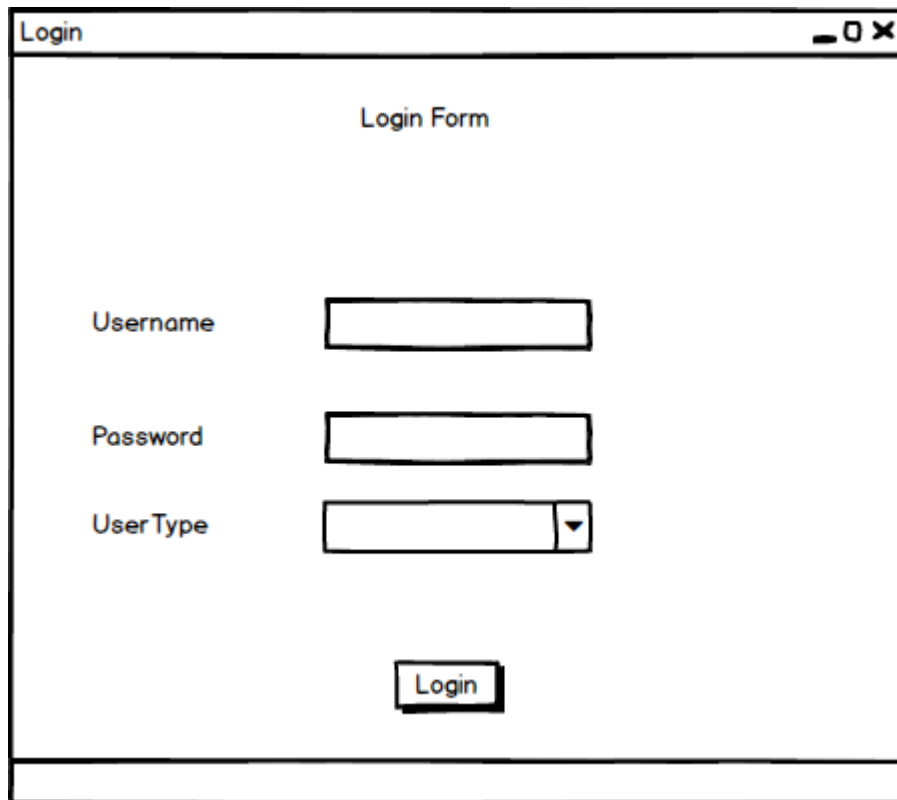


ER diagram



UI design

Login



A wireframe of a login form within a window titled "Login". The window has standard OS controls (minimize, maximize, close) in the top right corner. The form is titled "Login Form" and contains three input fields: "Username", "Password", and "UserType". The "UserType" field is a dropdown menu. Below the input fields is a "Login" button.

Login

Login Form

Username

Password

UserType

Login

Employee registration

Registration Form

Register

Username

Password

Comfirm Password

UserType

Clear

Register

Back

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

Inventory management

Order management

Category management

User management

Sell history

User from

User Form

Black Lotus

Logout

Inventory management

Order management

Category management

Sell history

Inventory management

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

Inventory management

Product ID

Name

Description

Quantity

Price

Category

Product Category

ComboBox

Refresh

Clear

Add Item

Delete Item

Update Item

User form

User Form

Black Lotus

Logout

Inventory management

Product ID

Name

Description

Quantity

Price

Category

Product Category

ComboBox

Refresh

Clear

Add Item

Delete Item

Update Item

Order management

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

Order management

Customer Name

Product ID

Search by ID

Product Name

Search by Name

Product Category

Product Quantity

Price Per Item

Enter The Quantity

Total

Total

Available Stock

Invoice

Place Order

Clear

User form

User Form

Black Lotus

Logout

Order management

Customer Name

Product ID

Search by ID

Product Name

Search by Name

Product Category

Product Quantity

Price Per Item

Enter The Quantity

Total

Total

Available Stock

Invoice

Place Order

Clear

Category management

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

Category management

Refresh

Select Category

ComboBox

Update Category

Update Item

Delete Category

Delete Item

Clear

User form

Admin Form

Black Lotus

Logout

Category management

Refresh

Select Category

ComboBox

Update Category

Update Item

Delete Category

Delete Item

Clear

User management

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

User management

Username

Password

Usertype

ComboBox

Check

New Password

Confirm Password

Change

Clear

Sell history

Admin form

Admin Form

Black Lotus

Logout

Create Employee Account

Create

Sale History

Refresh

User form

User Form

Black Lotus

Logout

Sale History

Refresh

Task 04

Source codes

Login



```
bool valid = true;
if (string.IsNullOrEmpty(password.Text) || string.IsNullOrEmpty(username.Text))
{
    MessageBox.Show("Need to fill all the values", "Login Form", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);
    valid = false;
}

if (valid)
{
    string userType = userType.Text.Trim();
    SqlCommand cmd = new SqlCommand("select userType from login where username = '" + username.Text.Trim() + "' and password = '" + password.Text.Trim() + "'", sqlConn);

    SqlDataAdapter dr = cmd.ExecuteReader();

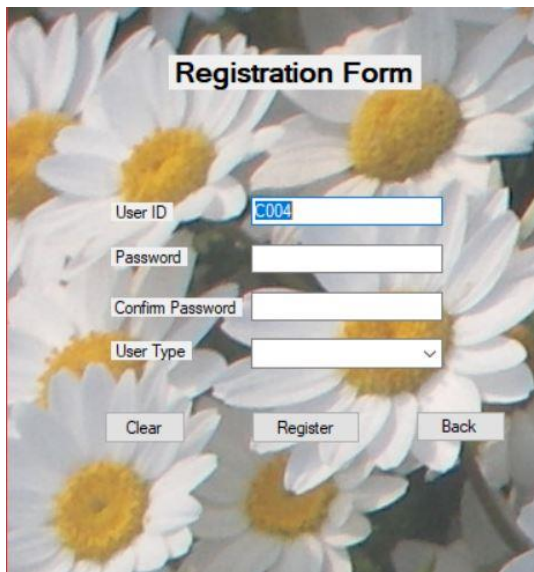
    if (dr.HasRows)
    {
        while (dr.Read())
        {
            userType = dr[0].ToString().Trim();

            if (userType.Equals("Admin"))
            {
                admin obj = new admin();
                obj.Show();
                obj.Hide();
                dr.Close();
            }

            else if (userType.Equals("User"))
            {
                user obj = new user();
                obj.Show();
                obj.Hide();
                dr.Close();
            }
        }
    }
    else
    {
        MessageBox.Show("Invalid user", "Login Form", MessageBoxButtons.OK, MessageBoxIcon.Error);
        dr.Close();
    }
}
```

First of all in the login frame it will export the SQL query where username is and if it is wrong it will show a “invalid” notice and if it is correct it will check given password is correct by the user and verify whether it is cashier or manager and after login successful it will show notice as “login as “ with the user type and pass the frame.

Create Employee Account

A registration form titled "Registration Form" is displayed over a background of white daisies. The form contains four input fields: "User ID" with the value "C004", "Password", "Confirm Password", and "User Type" which is a dropdown menu. Below the input fields are three buttons: "Clear", "Register", and "Back".

```
private void btnRegister_Click(object sender, EventArgs e)
{
    if (usrIdb.Text == "" || Confpassword.Text == "" || Password.Text == "")
    {
        MessageBox.Show("All fields are compulsory", "Register form", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    if (Confpassword.Text != Password.Text)
    {
        MessageBox.Show("Password or Confirm Password is not matched", "Register form", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        SqlCommand cmdInsert = new SqlCommand("Insert into login values( '" + usrIdb.Text.ToString().Trim() + "', '" + Confpassword.Text.ToString().Trim() + "', '" + Password.Text.ToString().Trim() + "')", sqlConn);
        cmdInsert.CommandType = CommandType.Text;
        cmdInsert.ExecuteNonQuery();
        MessageBox.Show("User Added", "Register form", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}

//Insertion ex)
{
    MessageBox.Show(ex.Message);
}

//Display
{
    sqlConn.Close();
}
}
```

In this function, I have used “Auto Account Number” method which has assigned into the User ID and admin will not be able to create an account if any text field is blank and I will pop up notice as well. if the user mistakenly wrote the wrong password user can easily click clear button to erase the previous texts as well.

Inventory management

Inventory management Order management Category management User Management Sale history

Product Category:

Product ID:

Name:

Description:

Quantity:

Price:

Category:

PID	Name	Description	Quantity	Price	Catago
P001	s	zx	zc	zcx	CXZ

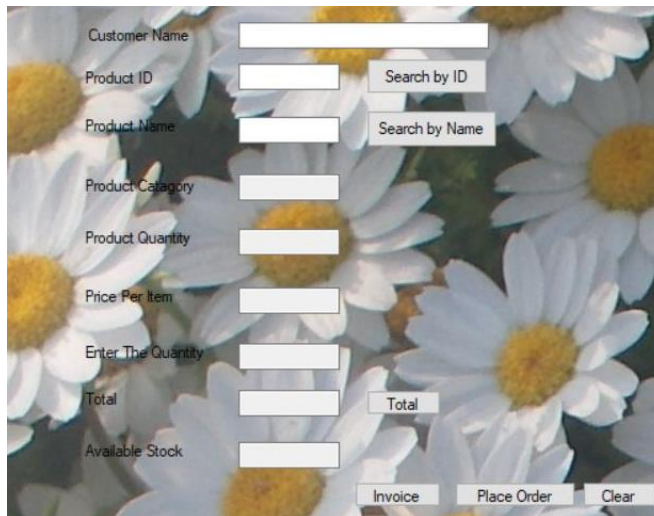
```
private void Add_Click(object sender, EventArgs e)
{
    try
    {
        SqlCommand cmd = new SqlCommand("insert into product values( '"+PID.Text.Trim()+"','"+Name.Text.Trim()+"','"+Description.Text.Trim()+"','"+Quantity.Text.Trim()+"','"+Price.Text.Trim()+"','"+Category.Text.Trim()+"' );", sqlcon);
        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();
        MessageBox.Show("Item added", "Admin form", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlcon.State == ConnectionState.Open)
        {
            sqlcon.Close();
        }
    }
}
```

```
try
{
    SqlCommand cmd = new SqlCommand("delete from product where PID = '" + PID.Text + "'", sqlcon);
    int recordnum = cmd.ExecuteNonQuery();
    if (recordnum > 0)
    {
        MessageBox.Show("Item Deleted", "Admin form", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        MessageBox.Show("Invalid Item", "Admin form", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
catch (Exception ex)
{
    MessageBox.Show("Error deleting data " + ex);
}
```

```
private void Update_Click(object sender, EventArgs e)
{
    try
    {
        SqlCommand cmd = new SqlCommand("update product set Name = '"+Name.Text.Trim()+"',Description = '"+Description.Text.Trim()+"',Quantity = '"+Quantity.Text.Trim()+"',Price = '"+Price.Text.Trim()+"',Category = '"+Category.Text.Trim()+"' where PID = '" + PID.Text.Trim()+"' ;", sqlcon);
        int recordnum = cmd.ExecuteNonQuery();
        if (recordnum > 0)
        {
            MessageBox.Show("Item updated", "Admin form", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else
        {
            MessageBox.Show("Invalid Item", "Admin form", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error updating data " + ex);
    }
}
```

In this function, I have used the “Auto Product Number” method which has assigned into the Product ID. When the admin/user double-clicking the table that the admin/user needs to get details, it will automatically assign into the text fields which given in the form. It will allow the admin/user to update delete and by using auto-id method user can easily add items and refresh button will update the table, and the clear button will erase the text fields as well.

Order management



```
SqlCommand cmd = new SqlCommand("select Name,Quantity,Price,Category from product where PID = '" + PID2.Text.Trim() + "'", sqlCon);

SqlDataAdapter dr = cmd.ExecuteReader();

if (dr.HasRows)
{
    if (dr.Read())
    {
        pname2.Text = dr["Name"].ToString().Trim();
        quat2.Text = dr["Quantity"].ToString().Trim();
        price2.Text = dr["Price"].ToString().Trim();
        cat2.Text = dr["Category"].ToString().Trim();
        MessageBox.Show("Product Successfully Detected", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
        ex.ReadOnly = false;
        total9.ReadOnly = false;
    }
    else {
        MessageBox.Show("Invalid ID", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    dr.Close();
}
}
catch (Exception ex)
{
    MessageBox.Show("Error in adding data" + ex.Message, "D Form", MessageBoxButtons.OKCancel, MessageBoxIcon.Error);
}
finally
{
    if (sqlCon.State == ConnectionState.Open)
    {
        sqlCon.Close();
    }
}
```

```
private void invoice_Click(object sender, EventArgs e)
{
    try
    {
        sqlCon.Open();
        SqlCommand cmd = new SqlCommand("Insert into sells values( '" + PID2.Text.Trim() + "','" + pname2.Text.Trim() + "','" + eq.Text.Trim() + "','" + total9.Text.Trim() + "','" + cat2.Text.Trim() + "','" + cname.Text.Trim() + "'", sqlCon);

        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();

        MessageBox.Show("Item Added", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlCon.State == ConnectionState.Open)
        {
            sqlCon.Close();
        }
    }
}
```

```
private void order_Click(object sender, EventArgs e)
{
    try
    {
        sqlCon.Open();
        SqlCommand cmd = new SqlCommand("update product set Quantity='" + astock.Text.Trim() + "' where PID=''" + PID2.Text.Trim() + "'", sqlCon);

        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();

        MessageBox.Show("Order Has been Placed", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlCon.State == ConnectionState.Open)
        {
            sqlCon.Close();
        }
    }
}
```

In this function, the admin/user can search the product by product id and product name and also when the user gives the quantity to the system by clicking total use can get the full amount of the quantity and by clicking invoice, user can transfer the data into the sale history database, and by placing order, admin/user can place the order as well.

Category management

Inventory management Order management **Category management** User Management Sale history

Select category:

Update category:

Delete category:

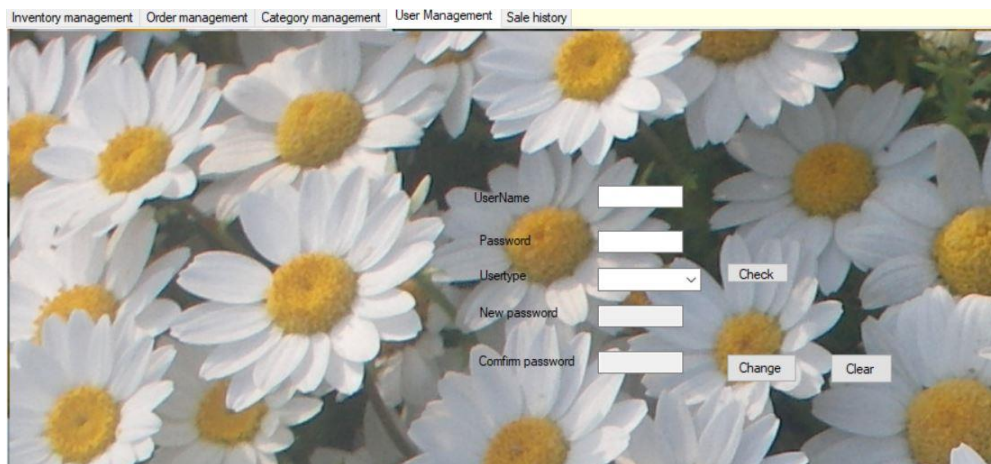
	PID	Name	Description	Quantity	Price	Category
▶	P002					lotus
*						

```
private void update_Click(object sender, EventArgs e)
{
    try
    {
        if (updateText.Text == "")
        {
            MessageBox.Show("Enter Employee id to update");
        }
        else
        {
            SqlCommand cmd = new SqlCommand("update product set Category='" + updateText.Text.Trim() + "' where Category='" + comboBox1.Text.Trim() + "'", sqlCon);
            cmd.CommandType = CommandType.Text;
            cmd.ExecuteNonQuery();
            MessageBox.Show("Category updated", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlCon.State == ConnectionState.Open)
        {
            sqlCon.Close();
        }
    }
}
```

```
try
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Employee id to update");
    }
    else
    {
        SqlCommand cmd = new SqlCommand("delete from product where Category='" + textBox2.Text.Trim() + "'", sqlCon);
        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();
        MessageBox.Show("Category deleted", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
        sqlCon.Close();
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (sqlCon.State == ConnectionState.Open)
    {
        sqlCon.Close();
    }
}
```

In this function, the admin/user can see what the category has now in the system, and the user will be able to edit category and updating the category as well

User management



Inventory management | Order management | Category management | **User Management** | Sale history

Username

Password

User type

New password

Confirm password

```
private void update_catg_Click(object sender, EventArgs e)
{
    try
    {
        if (updatecat.Text == "")
        {
            MessageBox.Show("Enter employee id to update");
        }
        else
        {
            SqlCommand cmd = new SqlCommand("update product set Category='" + updatecat.Text.Trim() + "' where Category='" + catgbox.Text.Trim() + "'", sqlCon);

            cmd.CommandType = CommandType.Text;
            cmd.ExecuteNonQuery();

            MessageBox.Show("Category updated", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlCon.State == ConnectionState.Open)
        {
            sqlCon.Close();
        }
    }
}
```

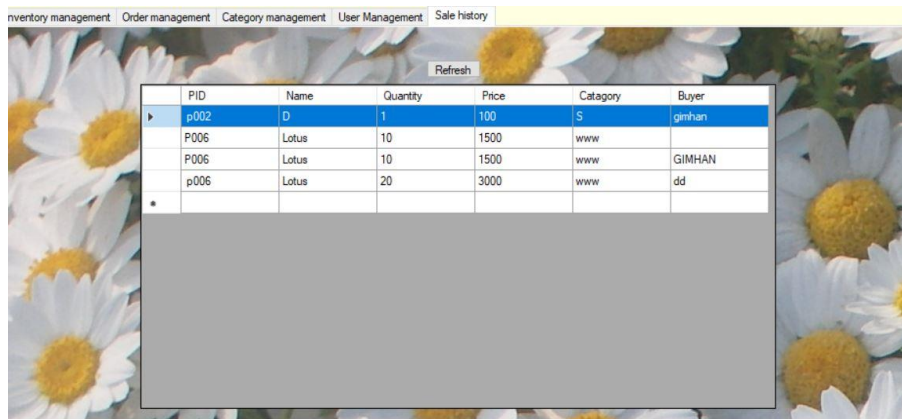
```
private void del_Click(object sender, EventArgs e)
{
    try
    {
        if (textBox2.Text == "")
        {
            MessageBox.Show("Enter Employee id to update");
        }
        else
        {
            SqlCommand cmd = new SqlCommand("delete from product where Category='" + textBox2.Text.Trim() + "'", sqlCon);

            cmd.CommandType = CommandType.Text;
            cmd.ExecuteNonQuery();

            MessageBox.Show("Category deleted", "Admin Form", MessageBoxButtons.OK, MessageBoxIcon.Information);
            sqlCon.Close();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (sqlCon.State == ConnectionState.Open)
        {
            sqlCon.Close();
        }
    }
}
```

In this function, the admin can change the password for security measures in the form has imbedded the previous password if it correct admin can change the password aswell.

Sell history



	PID	Name	Quantity	Price	Category	Buyer
▶	p002	D	1	100	S	gimhan
	P006	Lotus	10	1500	www	
	P006	Lotus	10	1500	www	GIMHAN
	p006	Lotus	20	3000	www	dd
•						

```
try
{
    SqlCommand SearchCustomerNameCommand = new SqlCommand
        ("select * from sells", sqlCon);

    SqlDataAdapter da = new SqlDataAdapter(SearchCustomerNameCommand);
    DataSet ds = new DataSet();
    da.Fill(ds, "sells");

    dataGridView2.DataSource = ds;
    dataGridView2.DataMember = "sells";
}
catch (Exception ex)
{
    MessageBox.Show("Error loading the product " + ex);
}
```

In this function, the admin/user can sell that have done and by clicking refresh can refresh the table aswell.

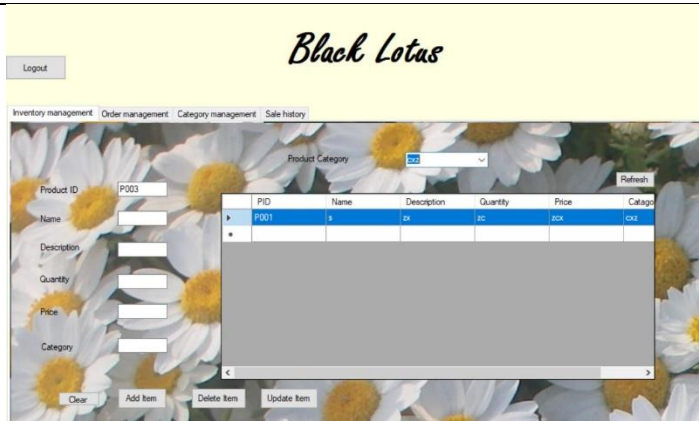
Task 05


Test plan

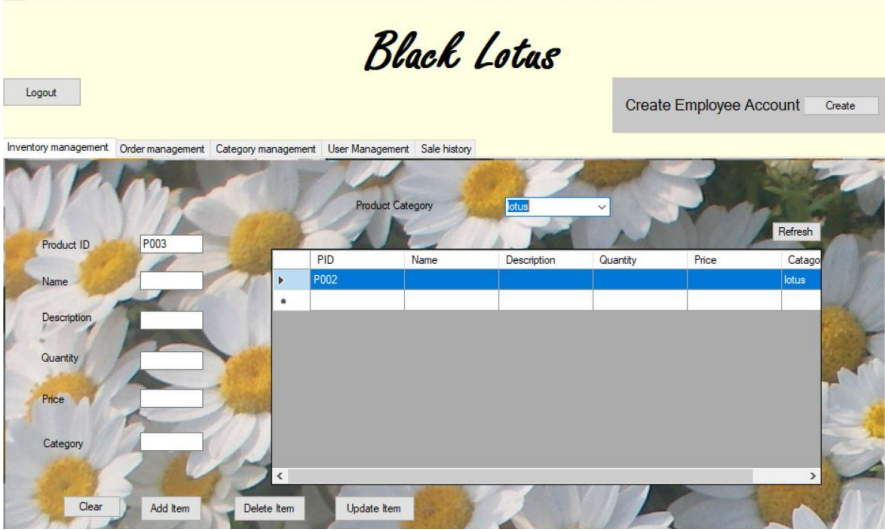
TEST CASE ID	CASE NAME	SENARIO	EXPECTED RESULT
TEST 1.0	Checking User and Admin login username and password working	Login credentials should work as planned	<ul style="list-style-type: none"> • Login successfully if its right. • Invalid note. If it's wrong.
TEST 2.0	Checking "create Employee account" function is working	create Employee account function should display as planned and back, clear, create buttons should work as planned on Admin frame	<ul style="list-style-type: none"> • Account should be created • Clear button worked • back button worked create button create the account.
TEST 3.0	Checking Inventory management add button is working as well as the clear, update, Delete buttons on admin frame	Details should be Added and clear, update, delete buttons should work as planned	<ul style="list-style-type: none"> • Add button worked • Clear button worked • update button worked • delete button updated the account details.
TEST 4.0	Checking Inventory management add button is working as well as the clear, update, Delete buttons on User frame	Details should be Added and clear, update, delete buttons should work as planned	<ul style="list-style-type: none"> • Add button worked • Clear button worked • update button worked • delete button updated the account details.
TEST 5.0	Checking "Order management" function is working	Order management function should display as planned and Search by ID, Search by name, total, invoice,placeorder,clear buttons should work as planned on Admin frame.	<ul style="list-style-type: none"> • Order should be Placed • total button worked • Search by name and ID button worked • Clear button should work
TEST 6.0	Order management function should display as planned and Search by ID, Search by name, total, invoice,placeorder,clear buttons should work as planned on User frame.	Order management function should display as planned and Search by ID, Search by name, total, invoice, placeorder,clear buttons should work as planned on User frame.	<ul style="list-style-type: none"> • Order should be Placed • total button worked • Search by name and ID button worked • Clear button should work

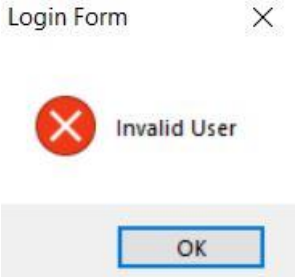
TEST 7.0	Checking “Category management” function is working	Category management function should display as planned and update, Delete, clear buttons should work as planned on Admin frame.	<ul style="list-style-type: none"> • Update button worked • Delete button worked • Clear button worked
TEST 8.0	Checking “Category management” function is working	Category management function should display as planned and update, Delete, clear buttons should work as planned on User frame.	<ul style="list-style-type: none"> • Update button worked • Delete button worked • Clear button worked
TEST 9.0	Checking “User management” function is working on admin	User management function should display as planned and check button, Change buttons, clear button should work as planned on manager and cashier frame.	<ul style="list-style-type: none"> • Check button worked • Change button worked • Clear button worked • Checking confirm password invalid notice
TEST 10.0	Checking “Sale history” function is working on manager and Admin frame	Sale history list should display as planned on manager and Admin frame.	<ul style="list-style-type: none"> • Sale history list should display
TEST 11.0	Checking “Sale history” function is working on manager and User frame	Sale history list should display as planned on manager and User frame.	<ul style="list-style-type: none"> • Sale history list should display

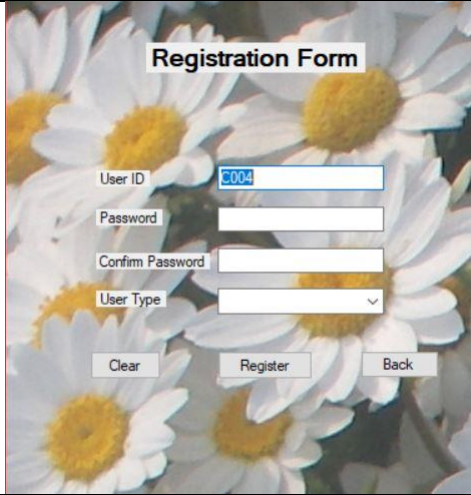
Test case

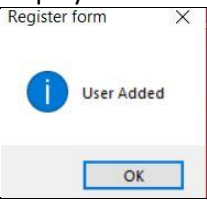
Test case	Test 1.0
Test objective	Checking User login is correct
Test data	Username: c007 Password: 2245510
Expected result	Display User menu
Actual result	
conclusion	Working correctly

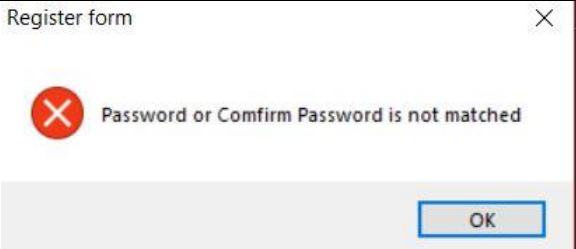
Test case	Test 1.1
Test objective	Checking User login is correct
Test data	Username: U008 Password: pass123
Expected result	Display invalid note
Actual result	
conclusion	Working correctly

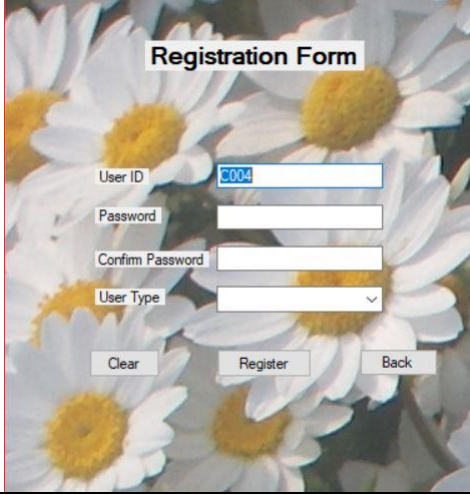
Test case	Test 1.3
Test objective	Checking Admin login is correct
Test data	Username: C005 Password: 2245510
Expected result	Display Admin menu
	
conclusion	Working correctly

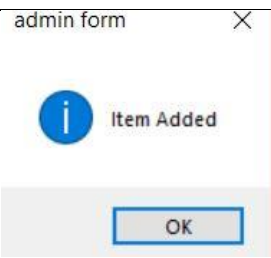
Test case	Test 1.4
Test objective	Checking Manager login is correct
Test data	Username: U008 Password: pass123
Expected result	Display invalid note
Actual result	
conclusion	Working correctly


Test case	Test 2.0
Test objective	Checking “Create Employee Account” function is working on Admin Frame
Test data	
Expected result	“Employee Registration” form should be displayed
Actual result	
conclusion	Working correctly

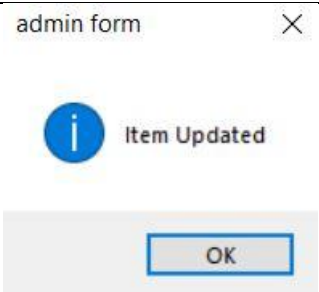
Test case	Test 2.1
Test objective	Checking Create Customer Account is correct
Test data	Filling the Application
Expected result	“New Employee Registered” notice should be displayed
Actual result	
conclusion	Working correctly

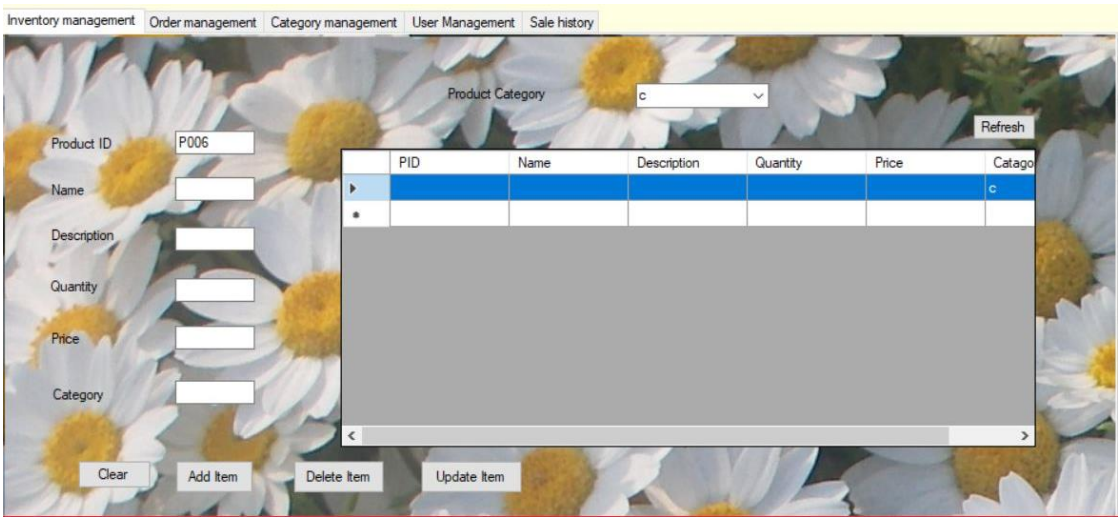
Test case	Test 2.2
Test objective	Checking Create Customer Account is correct
Test data	New Password: 2242510 Confirm Password:224251
Expected result	"New Employee Registered" notice should be displayed
Actual result	
conclusion	Working correctly

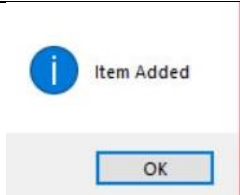
Test case	Test 2.3
Test objective	Checking "Create Employee Account" function clear button is working on Admin Frame
Expected result	Text fields should be erased
Test data	
conclusion	Working correctly

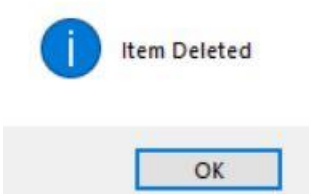
Test case	Test 3.0
Test objective	Checking inventory management ADD function is correct in admin form
Test data	Product ID:P005
Expected result	"Item Added" notice should be displayed
Actual result	
conclusion	Working correctly


Test case	Test 3.1
Test objective	Checking inventory management Delete function is correct in admin form
Test data	Product ID:P005
Expected result	"Item Deleted" notice should be displayed
Actual result	
conclusion	Working correctly

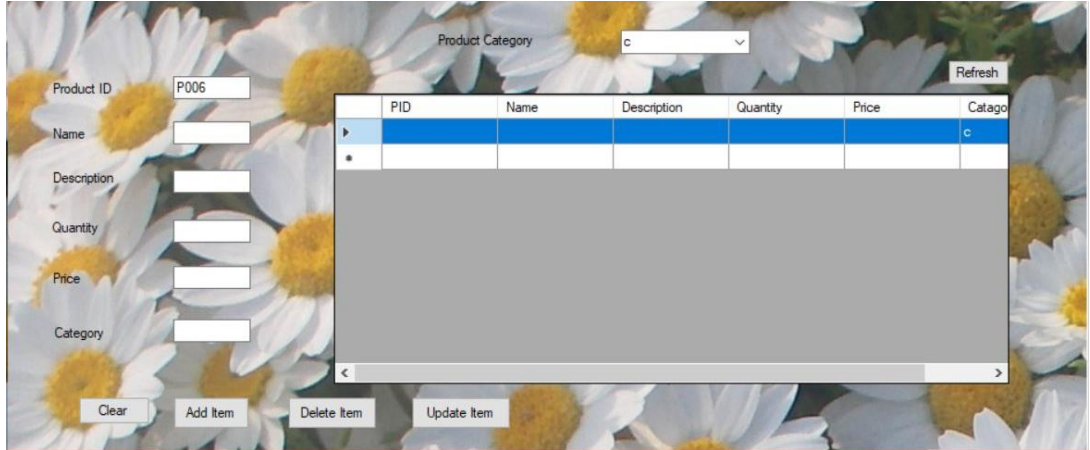
Test case	Test 3.2
Test objective	Checking inventory management Update function is correct in admin form
Test data	Product ID:P005
Expected result	"Item Updated" notice should be displayed
Actual result	
conclusion	Working correctly

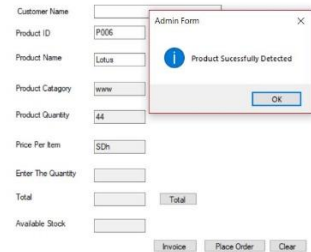
Test case	Test 3.3
Test objective	Checking "Inventory management" function clear button is working on Admin Frame
Expected result	Text fields should be erased
Test data	
Conclusion	Working correctly

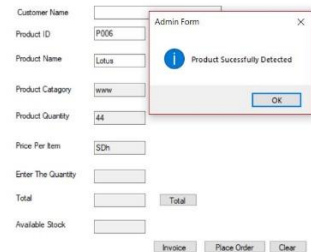
Test case	Test 4.0
Test objective	Checking inventory management ADD function is correct in User form
Test data	Product ID:P006
Expected result	"Item Added" notice should be displayed
Actual result	 <p>A screenshot of a notification box with a blue circular icon containing a white 'i'. To the right of the icon is the text 'Item Added'. Below the icon and text is a grey rectangular button with the text 'OK' in blue.</p>
conclusion	Working correctly

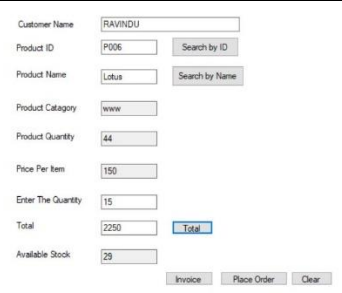
Test case	Test 4.1
Test objective	Checking inventory management Delete function is correct in User form
Test data	Product ID:P006
Expected result	"Item Deleted" notice should be displayed
Actual result	 <p>A screenshot of a notification box with a blue circular icon containing a white 'i'. To the right of the icon is the text 'Item Deleted'. Below the icon and text is a grey rectangular button with the text 'OK' in blue.</p>
conclusion	Working correctly


Test case	Test 4.2
Test objective	Checking inventory management Update function is correct in User form
Test data	Product ID:P006
Expected result	"Item Updated" notice should be displayed
Actual result	
conclusion	Working correctly

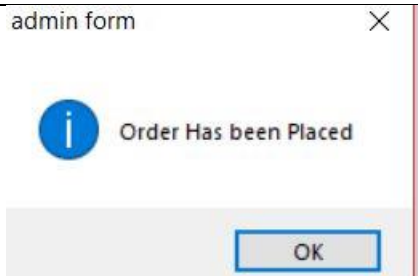
Test case	Test 4.3
Test objective	Checking "Inventory management" function clear button is working on User Frame
Test data	
Expected result	Text fields should be erased

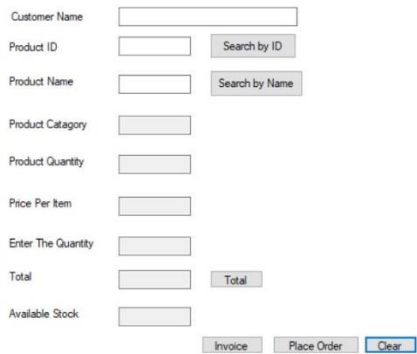
Test case	Test 5.0
Test objective	Checking Order management Search by product id function is correct in admin form
Test data	Product ID:P006
Expected result	“Product successfully detected “ notice and Product Details Should be displayed
Actual result	 <p>The screenshot shows the Admin Form with the following fields: Customer Name, Product ID (P006), Product Name (Lotus), Product Category (www), Product Quantity (44), Price Per Item (SDh), Enter The Quantity, Total, and Available Stock. A modal dialog box titled 'Admin Form' is displayed, showing a blue information icon and the message 'Product Successfully Detected' with an 'OK' button. At the bottom of the form are buttons for 'Invoice', 'Place Order', and 'Clear'.</p>
Conclusion	Working Correctly

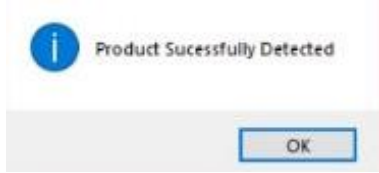
Test case	Test 5.1
Test objective	Checking Order management Search by product Name function is correct in admin form
Test data	Product Name: Lotus
Expected result	“Product successfully detected “notice and Product Details Should be displayed
Actual result	 <p>The screenshot shows the Admin Form with the following fields: Customer Name, Product ID (P006), Product Name (Lotus), Product Category (www), Product Quantity (44), Price Per Item (SDh), Enter The Quantity, Total, and Available Stock. A modal dialog box titled 'Admin Form' is displayed, showing a blue information icon and the message 'Product Successfully Detected' with an 'OK' button. At the bottom of the form are buttons for 'Invoice', 'Place Order', and 'Clear'.</p>
Conclusion	Working Correctly

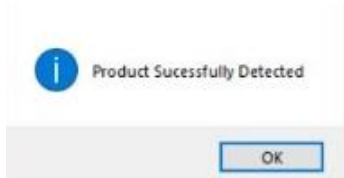
Test case	Test 5.2
Test objective	Checking Order management Total function is correct in admin form
Test data	Enter the Quantity : 15
Expected result	Total Should be given
Actual result	 <p>The screenshot shows the 'admin form' for order management. It includes input fields for Customer Name (RAVIN DU), Product ID (P006), Product Name (Lotus), Product Category (www), Product Quantity (44), Price Per Item (150), Enter The Quantity (15), Total (2250), and Available Stock (29). A 'Total' button is highlighted, and the calculated total is displayed as 2250. At the bottom, there are buttons for 'Invoice', 'Place Order', and 'Clear'.</p>
Conclusion	Working Correctly

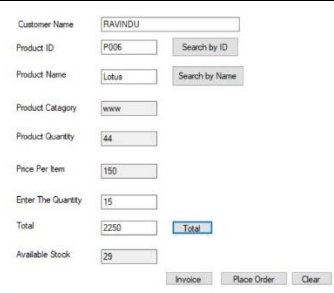
Test case	Test 5.3
Test objective	Checking Order management invoice function is correct in admin form
Test data	n/a
Expected result	"Item Added" notice should be displayed
Actual result	 <p>The screenshot shows a notification dialog box titled 'admin form' with a close button (X). It contains an information icon (i) and the text 'Item Added'. An 'OK' button is at the bottom.</p>
Conclusion	Working Correctly


Test case	Test 5.4
Test objective	Checking Order management Place Order function is correct in admin form
Test data	n/a
Expected result	"The order has been placed" notice should be displayed
Actual result	
Conclusion	Working Correctly

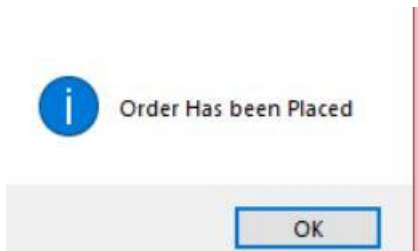
Test case	Test 5.5
Test objective	Checking "order management" function clear button is working on Admin Frame
Test data	
Expected result	Text fields should be erased


Test case	Test 6.0
Test objective	Checking Order management Search by product id function is correct in User form
Test data	Product ID:P006
Expected result	"Product successfully detected " notice and Product Details Should be displayed
Actual result	
Conclusion	Working Correctly

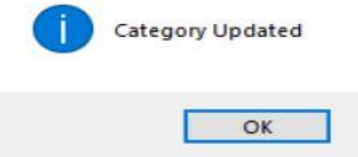
Test case	Test 6.1
Test objective	Checking Order management Search by product Name function is correct in User form
Test data	Product Name: Lotus
Expected result	"Product successfully detected "notice and Product Details Should be displayed
Actual result	
Conclusion	Working Correctly

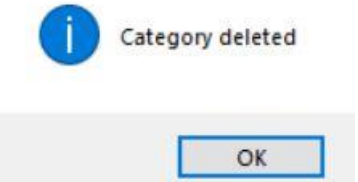
Test case	Test 6.2
Test objective	Checking Order management Total function is correct in User form
Test data	Enter the Quantity : 15
Expected result	Total Should be given
Actual result	 <p>The screenshot shows a web form for order management. Fields include: Customer Name (RAVIN DU), Product ID (P006), Product Name (Lotus), Product Category (www), Product Quantity (44), Price Per Item (150), Enter The Quantity (15), Total (2250), and Available Stock (29). A 'Total' button is highlighted in blue. At the bottom are 'Invoice', 'Place Order', and 'Clear' buttons.</p>
Conclusion	Working Correctly

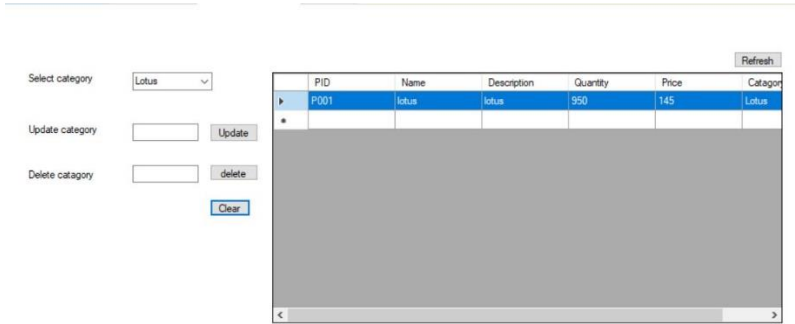
Test case	Test 6.3
Test objective	Checking Order management invoice function is correct in User form
Test data	n/a
Expected result	"Item Added" notice should be displayed
Actual result	 <p>The screenshot shows a notification dialog box with a blue information icon, the text 'Item Added', and an 'OK' button.</p>
Conclusion	Working Correctly

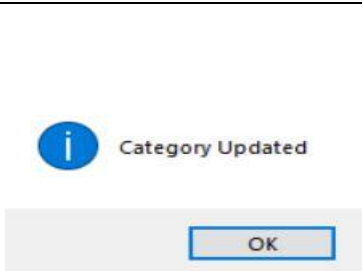
Test case	Test 6.4
Test objective	Checking Order management Place Order function is correct in admin form
Test data	n/a
Expected result	"The order has been placed" notice should be displayed
Actual result	
Conclusion	Working Correctly

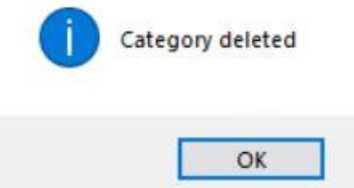
Test case	Test 6.5
Test objective	Checking "order management" function clear button is working on User Frame
Test data	
Expected result	Text fields should be erased

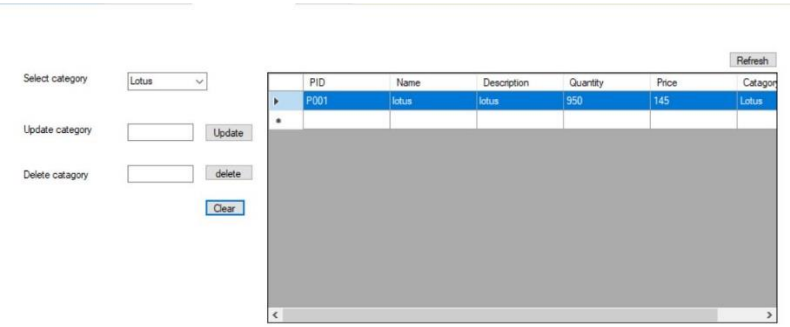
Test case	Test 7.0
Test objective	Checking “Category management” function Update button is working on Admin Frame
Test Data	Update Category :lotus
Expected result	“Category Updated” notice should be displayed
Test data	
Expected result	Working Correctly

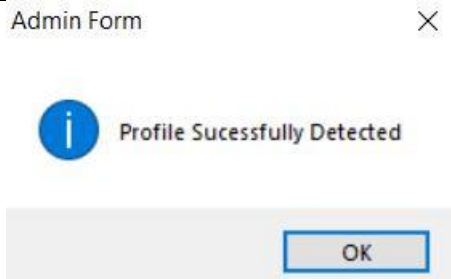
Test case	Test 7.1
Test objective	Checking “Category management” function Delete button is working on Admin Frame
Test Data	Update Category: lotus
Expected result	“Category deleted” notice should be displayed
Test data	
Expected result	Working Correctly


Test case	Test 7.2
Test objective	Checking “Category management” function clear button is working on Admin Frame
Test Data	N/a
Expected result	Text fields should be erased
Test data	 <p>The screenshot shows a web application interface for category management. On the left, there are three input fields: 'Select category' (a dropdown menu with 'Lotus' selected), 'Update category' (an empty text box), and 'Delete category' (an empty text box). To the right of these fields are three buttons: 'Update', 'delete', and 'Clear'. The 'Clear' button is highlighted with a blue border. On the right side of the interface, there is a table with columns: PID, Name, Description, Quantity, Price, and Category. The table contains one row with data: P001, Lotus, Lotus, 950, 145, Lotus. Above the table is a 'Refresh' button. Below the table is a scroll bar.</p>
Expected result	Working Correctly


Test case	Test 8.0
Test objective	Checking “Category management” function Update button is working on User Frame
Test Data	Update Category: lotus
Expected result	“Category Updated” notice should be displayed
Test data	 <p>The screenshot shows a notification message box. It has a blue circular icon with a white 'i' on the left. To the right of the icon, the text 'Category Updated' is displayed. Below the text is an 'OK' button with a blue border.</p>
Expected result	Working Correctly

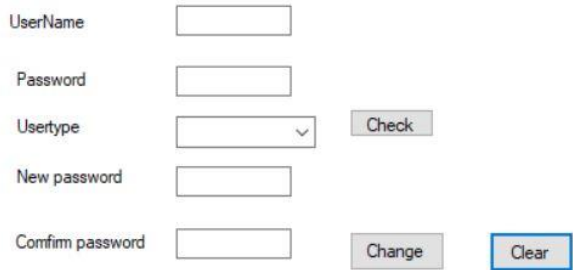
Test case	Test 8.1
Test objective	Checking “Category management” function Delete button is working on User Frame
Test Data	Update Category: lotus
Expected result	“Category Deleted” notice should be displayed
Test data	
Expected result	Working Correctly

Test case	Test 8.2
Test objective	Checking “Category management” function clear button is working on User Frame
Test Data	N/a
Expected result	Text fields should be erased
Test data	
Expected result	Working Correctly

Test case	Test 9.0
Test objective	Checking “User management” function Check button is working on Admin Frame
Test Data	Username:C001 Password:2242510 Usertype:admin
Expected result	“Account has been detected”notice should be displayed
Test data	
Expected result	Working Correctly

Test case	Test 9.1
Test objective	Checking “User management” function Change button is working on Admin Frame
Test Data	Comfirm Password:123456 New Password:2242510
Expected result	“Password Updated” notice should be displayed
Test data	
Expected result	Working Correctly

Test case	Test 9.2
Test objective	Checking User management is correct
Test data	New Password: 2242510 Confirm Password:224251
Expected result	"New Employee Registered" notice should be displayed
Actual result	
conclusion	Working correctly

Test case	Test 9.3
Test objective	Checking "User management" function clear button is working on Admin Frame
Test data	
Expected result	Working Correctly

Test case	Test 10.0																																										
Test objective	Checking “Sale history” function is working on Admin Frame																																										
Test data	<div><div>Refresh</div><table><tr><th></th><th>PID</th><th>Name</th><th>Quantity</th><th>Price</th><th>Catagony</th><th>Buyer</th></tr><tr><td>▶</td><td>p002</td><td>D</td><td>1</td><td>100</td><td>S</td><td>gmhan</td></tr><tr><td></td><td>P006</td><td>Lotus</td><td>10</td><td>1500</td><td>www</td><td></td></tr><tr><td></td><td>P006</td><td>Lotus</td><td>10</td><td>1500</td><td>www</td><td>GIMHAN</td></tr><tr><td></td><td>p006</td><td>Lotus</td><td>20</td><td>3000</td><td>www</td><td>dd</td></tr><tr><td>*</td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table></div>		PID	Name	Quantity	Price	Catagony	Buyer	▶	p002	D	1	100	S	gmhan		P006	Lotus	10	1500	www			P006	Lotus	10	1500	www	GIMHAN		p006	Lotus	20	3000	www	dd	*						
	PID	Name	Quantity	Price	Catagony	Buyer																																					
▶	p002	D	1	100	S	gmhan																																					
	P006	Lotus	10	1500	www																																						
	P006	Lotus	10	1500	www	GIMHAN																																					
	p006	Lotus	20	3000	www	dd																																					
*																																											
Expected result	Working Correctly																																										

Test case	Test 11.0																																										
Test objective	Checking “Sale history” function is working on User Frame																																										
Test data	<div><div>Refresh</div><table><tr><th></th><th>PID</th><th>Name</th><th>Quantity</th><th>Price</th><th>Category</th><th>Buyer</th></tr><tr><td>▶</td><td>p002</td><td>D</td><td>1</td><td>100</td><td>S</td><td>gmhan</td></tr><tr><td></td><td>P006</td><td>Lotus</td><td>10</td><td>1500</td><td>www</td><td></td></tr><tr><td></td><td>P006</td><td>Lotus</td><td>10</td><td>1500</td><td>www</td><td>GIMHAN</td></tr><tr><td></td><td>p006</td><td>Lotus</td><td>20</td><td>3000</td><td>www</td><td>dd</td></tr><tr><td>*</td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table></div>		PID	Name	Quantity	Price	Category	Buyer	▶	p002	D	1	100	S	gmhan		P006	Lotus	10	1500	www			P006	Lotus	10	1500	www	GIMHAN		p006	Lotus	20	3000	www	dd	*						
	PID	Name	Quantity	Price	Category	Buyer																																					
▶	p002	D	1	100	S	gmhan																																					
	P006	Lotus	10	1500	www																																						
	P006	Lotus	10	1500	www	GIMHAN																																					
	p006	Lotus	20	3000	www	dd																																					
*																																											
Expected result	Working Correctly																																										

Task 06

Deploy User Application or Component Using Windows Installer

Introduction

Whenever we are dissatisfied with the layout of an application or segment, we are afraid of the idiomatic scripted layout program. Microsoft .NET installation and deployment will be arranged for you in a simple and clear way. Many programmers using Microsoft Visual Studio .NET do not fully pay attention to the Windows installers that come with Visual Studio. They are looking for other setup programs that require hard script knowledge.

About Microsoft Installer

Windows Installer enables the user to send applications and components proficiently. The installer provides new features that can be highlighted without installing them, introducing items as required, and including customer customization and it reduces the total cost of purchasing.

Five steps

Step 01

Open the Visual Studio IDE, and select File->New ->Setup and Deployment Projects. Select the Setup Wizard. Give the File name and location where to store that project.

Step 02

Welcome to Setup Project Wizard will show up. It is a straightforward Four-Step Wizard. Click on the Next Button. In the second step on the wizard, it will request the sort from the setup and click Next.

Step 03

In this step, it will request to include your created application or segment and different files that you have to put in the customer's machine, for instance, assume documentation documents afterwards user will redirect to another frame.

Step 04

In the application folder is the folder where the application and supporting files will be installed. The user's desktop and program menus are the client's desktop and program menus, respectively. If you place short files and help files for the main application on the desktop and user program menus, during the installation process, by placing the files provided in the application, the application, supporting files and Shortcuts, user desktop directories and program menus.

User can use the following editors during the deployment of application such as:

File system editor

It enables the user to add project output, files and other projects to the deployment project and specify wherever they are installed on the target computer.

Registry settings management editor

It enables the user to define which registry keys as well as the values to add to the target computer's registry.

File types management editor

It is applied to set file associations on the target computer.

Custom action management editor

It enables the user to define extra actions to be performed on the target computer at the end of the installation.

Launch condition management editor

It lets the user define the requirements that must be fitted in order for the installation to run successfully.

User interface management editor

User Interface Management Editor is utilized to determine and set the properties of predefined dialogue boxes that are displayed through installation on the selected system.

Step 05

After building the application, the user will obtain the install program for using our software or component in the projects debug directory.

Conclusion

In this black lotus, the system has four frames, such as the login menu, register menu, admin menu, and user menu as well. In these two menus (admin menu and user menu) without having many frames I have used tab method in order to make it more user-friendly to the user, and it will make time-consuming rather than having many frames in the system. Admin frame and user frame has the same function such as inventory management, order management, category management and sale history. Still, in the admin, the frame has more extra two functionality that only admin can access such as create employee account and user management. In the login part user needs to choose their user type to make it more secure rather than implementing only the user name and password. In the registration form and inventory management have used auto-id generating method to make easier to register to the system and avoid the duplication issue as well.

Strength

- User levels have used to the system
- Software is compatible with every device
- Not many tables in the database and easier to retrieve data because it's in one place.
- All the function works in the one framework using tabs to make it more user friendly except create employee account.
- When creating accounts username and product ID will automatically generate for the user
- User can order product by searching product ID and product name

Weakness

- System is not colorful to attract the user.
- If any database attack into the database can make occur huge data loss because of having few tables which all details have stored.
- If mistakenly user add the same username or product ID for any other employee or product, it will may occur duplicate ID's.

Future recommendation

- Implementing cloud storage to the system.
- Enhancing the system with an online website-based method
- Implementing AI and machine learning technology to the system.

Referencing

- CodeProject. 2019. Deploy your Application or Component Using Windows Installer - CodeProject. [ONLINE] Available at: <https://www.codeproject.com/Articles/12903/Deploy-your-Application-or-Component-Using-Windows>. [Accessed 29 December 2019].
- Why Learn C# - Best Programming Language. 2019. Why Learn C# - Best Programming Language. [ONLINE] Available at: <http://www.bestprogramminglanguagefor.me/why-learn-c-sharp>. [Accessed 30 December 2019].
- TerryGLee. 2019. Overview of Visual Studio | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019>. [Accessed 27 December 2019].
- gewarren. 2019. Common Type System & Common Language Specification | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/dotnet/standard/common-type-system>. [Accessed 28 December 2019].
- What is Common Language Specification. 2019. What is Common Language Specification. [ONLINE] Available at: http://vb.net-informations.com/framework/common_language_specification.htm. [Accessed 27 December 2019].