AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)

Dept. of Computer Science
Faculty of Science and Technology

CSC2210: OBJECT ORIENTED PROGRAMMING 2

Summer 2023-2024

Section: [M]

Project Report On

Online Shop app

Supervised By

TALHA BIN SARWAR

Submitted By:

Name	ID
1. Miraj Hossain	**22-49183-3**
2. Tonmoy Chandra Kar	**23-51480-1**
3. Md Shafayet Hossain Rizon	**20-43599-1**
4. Md Arif Billah	**23-50470-3**

Obtained Marks for CO2 and CO3 (Description given in the following page)

Assessment Criteria	Not Attended/ Incorrect (0)	Inadequate (1-2)	Average (3)	Good (4)	Excellent (5)

Evaluation Criteria (CO2)	Total =	Evaluation Criteria (CO3)	Total =
Requirement fulfillment		Organization of the application	
Validation		Representation and Integration of Database	
Verification		Graphical User Interface	

CO2: Display and verify the mean of a real-life Project using the concepts of C# Graphical User Interface based environment with database integration to depict a desktop-based application.					
Assessment Criteria	**Not Attended/ Incorrect (0)**	**Inadequate (1-2)**	**Average (3)**	**Good (4)**	**Excellent (5)**
Evaluation Criteria	**Evaluation Definition**				
Requirement fulfillment	Fails to demonstrate any understanding of real-life scenario-based project development or functional requirement identification. There is no attempt to depict a project or identify functional requirements accurately.	Demonstrates limited understanding of real-life scenario-based project development and functional requirement identification. The project depicted lacks coherence or relevance to real-life scenarios, and functional requirements are inaccurately identified or insufficiently described.	Presents a basic depiction of a real-life scenario-based project and identifies some functional requirements. However, the project lacks depth or complexity, and some functional requirements may be vaguely defined or missing key details.	Effectively demonstrates a realistic scenario-based project and accurately identifies most functional requirements. The project is well-developed with appropriate complexity, and functional requirements are clearly articulated with relevant details.	Exhibits an exceptional understanding of real-life scenario-based project development and accurately identifies all functional requirements. The project is meticulously developed with thorough attention to detail, reflecting a comprehensive understanding of Object-Oriented Programming project development activities.
Validation	Fails to demonstrate any understanding or implementation of validation forms in their system. There is no attempt to deal with data	Demonstrates limited understanding of validation forms and data validation techniques. While some attempt may be made to implement	Shows a basic understanding of validation forms and data validation techniques. They attempt to implement validation, but some aspects may be	Effectively demonstrates the use of validation forms and implements data validation techniques. Validation is mostly accurate and	Exhibits an exceptional understanding and implementation of validation forms and data validation techniques. Validation is meticulously

	validation, and validation requirements are completely ignored or incorrectly applied.	validation, it is incomplete or poorly executed, leading to inadequate handling of data validation.	missing or incorrectly implemented, resulting in partial or inconsistent handling of data validation.	comprehensive, ensuring the proper handling of data input and verification in the system.	implemented with thorough attention to detail, ensuring robust data validation procedures and contributing to the overall reliability and integrity of the system.
Verification	Fails to demonstrate any attempt to verify the system data or functional requirements. There is no evidence of understanding or implementation of verification processes, and data flow is not considered.	Demonstrates limited understanding of verification processes and data flow in the system. Verification attempts are incomplete or inaccurate, and there is insufficient consideration given to ensuring data integrity and functionality.	Shows a basic understanding of verification processes and attempts to verify system data. However, verification efforts may be inconsistent or lack thoroughness, and there may be gaps in ensuring proper functional requirements and data flow.	Identifies and verifies system data, ensuring proper functional requirements are met. Verification efforts are mostly accurate and thorough, with attention to ensuring data integrity and appropriate data flow within the system.	Exhibits an exceptional understanding of verification processes and meticulously verifies system data. Verification efforts are comprehensive and precise, with a keen focus on ensuring all functional requirements are met and maintaining proper data flow throughout the system.

CO3: Prepare and Explain a real life desktop based application synthesizing several component of C# along with development tools to adhere the given requirements.

Assessment Criteria	Not Attended/ Incorrect (0)	Inadequate (1-2)	Average (3)	Good (4)	Excellent (5)
Evaluation Criteria	**Evaluation Definition**				
Organization of the application	Fails to identify any suitable real time application or requirements for project development	Limited understanding about the project scopes and scenarios or identification of functional requirements.	Lacks depth or relevance to OOP project development activities and may contain inaccuracies. Real-life scenarios are	Consider and integrate the idea of several core aspects of the project along with relevance to real-life scenarios.	Generalize and exhibits an exceptional understanding of project preparation according to a to real-life scenarios. Also

	activities related to OOP.		mentioned, but the discussion lacks depth or clarity.	Demonstrating a solid understanding of the application presentation.	contains proper and insightful identification of the system which is comprehensive and precise.
Representation and Integration of Database	Fails to identify and present any understanding or implementation of database. Also failed to integrate the data with the project itself.	Limited understanding of the database concepts or their proper way of using in a real time project. While some attempt may be made to implement but it is incomplete or poorly executed, leading to inadequate design.	Lacks depth or relevance to database integration with the application. Shows a basic understanding but some aspects may be missing or incorrectly implemented, resulting in partial or inconsistency. May lack proper normalization.	Integrate the database with the forms properly and implements it with proper validation which is mostly accurate and comprehensive , ensuring the proper handling of data input and verification along with general normalization.	Exhibits an exceptional understanding and implementation of database ensuring attention to detail, and robust data manipulation procedures and contributing to the overall clarity.
Graphical User Interface	Fails to present or prepare GUI based application interfaces. There is no evidence of creating or integrating such things according to their usefulness.	Limited understanding of graphical user interfaces. Lack of design knowledge. Very poor attempt to make such things which are currently obsolete or can't be identified as coherent.	Shows a basic understanding of creating user interfaces. Most of them are interconnected but maybe some of them lack it. However, most of it can be described as user friendly.	Effectively identifies and meet the consider the simplicity. Design related works are mostly accurate and taken proper attention to ensuring a user-friendly coherent system.	Exhibits an exceptional work design following a high standard of simple and elegant work. Several controls and mechanism has been organized in a preferred way according to the coherent usage .

Table of Contents: Page no.

1. Chapter :01 (Introduction)-- 03
2. Chapter :02(User Story)--- 03
3. Chapter :03 (ER Diagram)--- 05
4. Chapter :03 (SQL Queries)-- 06
5. Chapter:04 (Screenshots) -- 09

Chapter-1: Introduction

This report details the design and development of a desktop-based e-commerce management system created using Windows Forms and C#. The application offers a user-friendly interface for both customers and administrators to manage products, process payments, and handle essential e-commerce tasks.

The primary objective of the project was to develop a platform that replicates real-world online shopping experiences by incorporating core features such as product browsing, cart management, user registration, and payment processing. SQL Server (.mdf database) is utilized for secure and efficient data storage and retrieval. The application is divided into two main sections: the UserPanel, where customers can explore products, manage their carts, and complete payments, and the AdminPanel, where administrators can manage users, products, and payment data.

The project was designed with scalability and ease of use in mind. Key features include basic CRUD (Create, Read, Update, Delete) operations for users and products, as well as efficient cart management. Additionally, the system provides a secure payment form for processing credit/debit card details, supporting multiple payment options. Administrative features include tracking user activity and payment history, giving administrators full control over the system.

Chapter-2: User Story

The online shop revolves around two main types of users: regular users (customers) and administrators (store managers). Each user type has distinct needs and functionalities, as described below:

User Story: Regular Users (Customers)

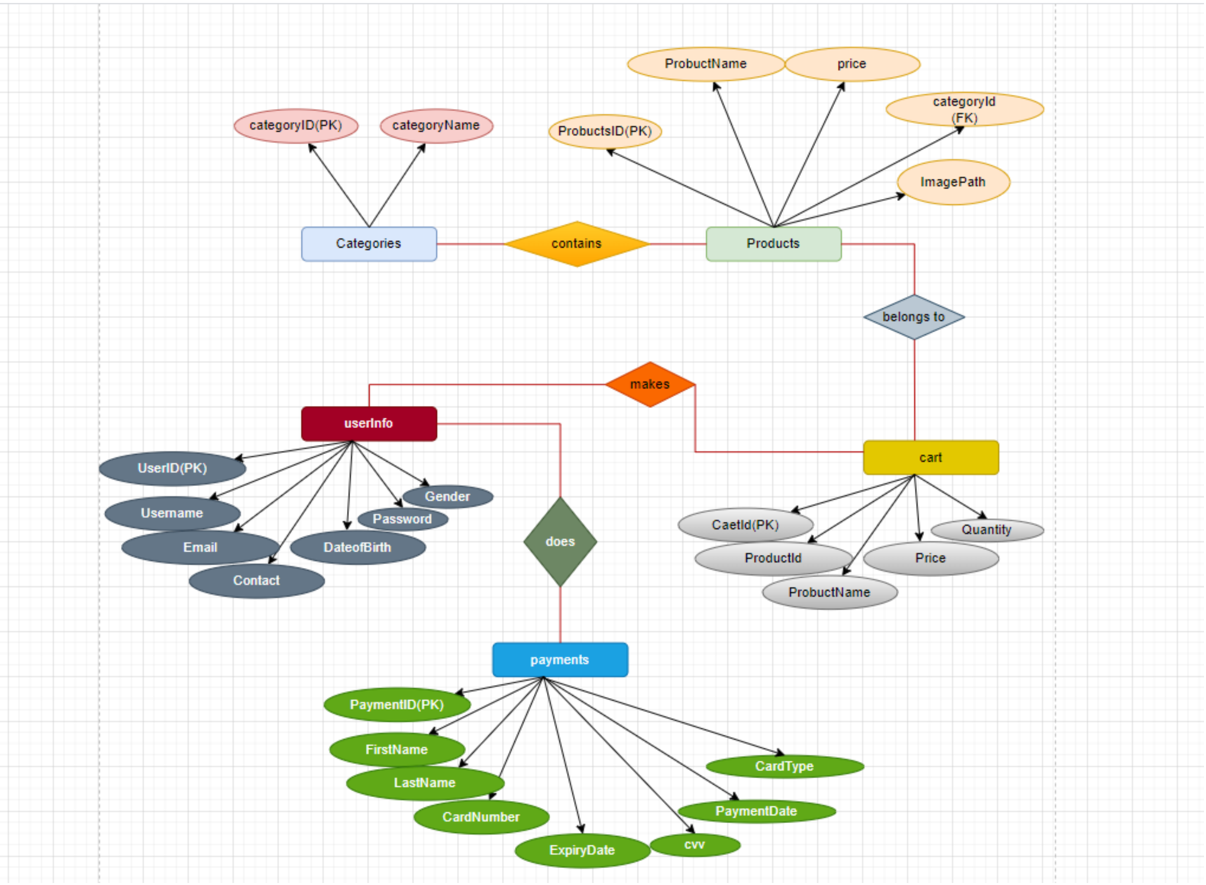
1. **Login and Registration**: Users can either log in to the system using their existing credentials or register for a new account. The login and sign-up processes are seamlessly integrated with the system's database for user authentication and storage.
2. **Product Browsing**: Once logged in, users are presented with a shop interface where they can browse available products. Each product displays relevant details such as name, price, and image.
3. **Add to Cart**: Users can add any selected product to their cart by clicking the "Add to Cart" button. The system ensures that if a product is added more than once, the quantity is correctly updated rather than duplicating the entry.
4. **Cart Management**: Users can review the items in their cart at any time. They can increase or decrease the quantity of items or remove unwanted items from the cart using the delete functionality. The cart data is reflected in a DataGridView for easy access and review.
5. **Checkout and Payment**: After reviewing their cart, users can proceed to checkout. The checkout process requires entering payment information. The system supports payment via **Cash** or **Card** (Visa, MasterCard, etc.), validated through secure forms.
6. **Payment Confirmation**: Once the payment is successful, users are provided with confirmation, and the cart is reset, allowing them to start a new shopping session.

User Story: Admin

1. **Admin Login**: The administrator has a separate login interface, ensuring secure access to the admin functionalities. Once logged in, the admin has access to all system data.
2. **Manage Users**: The admin can view all registered users and perform operations such as editing user information or deleting accounts. All user information is stored in the database and retrieved using SQL queries.
3. **Manage Products**: The admin can add new products to the system, specifying details such as product name, price, and image. Products can also be edited or removed if needed.
4. **View and Manage Payment Information**: The admin has access to a dedicated section where all payment data is displayed in a DataGridView. This information is fetched from the payment table in the database and provides insights into transaction history.
5. **Maintain Data Integrity**: The admin ensures the integrity of user and product data by deleting obsolete or unused entries. In the event of deletions, the system ensures that unique IDs (e.g., product or user IDs) are not reused, preventing conflicts or errors in the system.

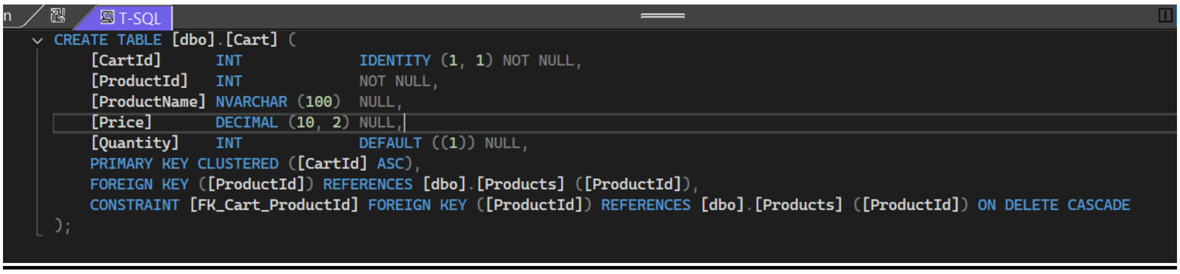
By combining the functionality of both the UserPanel and AdminPanel, the system successfully creates a seamless user experience for online shopping while ensuring efficient administration of the online shop operations.

Chapter:3- ER_Diagram



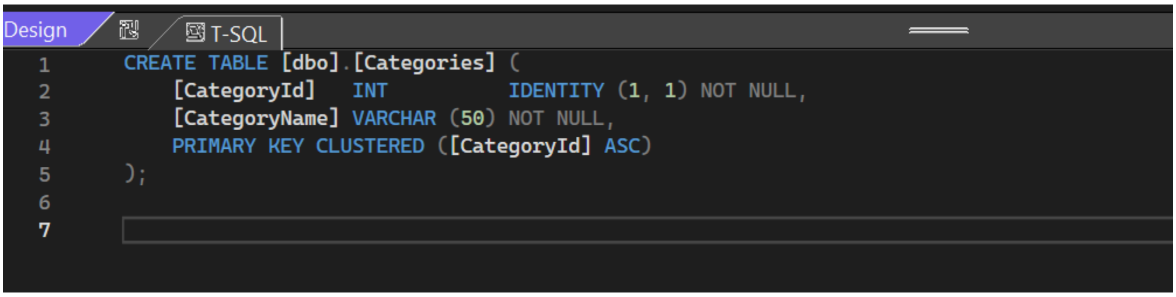
Chapter-4: SQL Queries

a) Table Defination Queries



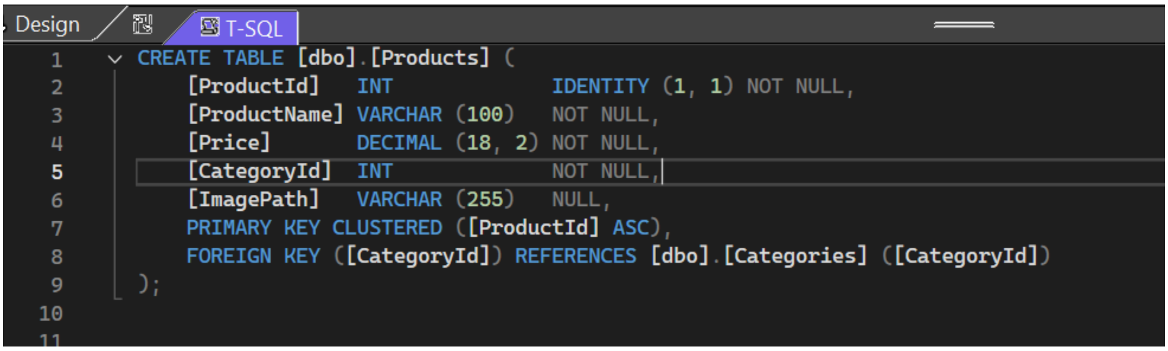
```sql
CREATE TABLE [dbo].[Cart] (
    [CartId]      INT             IDENTITY (1, 1) NOT NULL,
    [ProductId]   INT             NOT NULL,
    [ProductName] NVARCHAR (100)  NULL,
    [Price]       DECIMAL (10, 2) NULL,
    [Quantity]    INT             DEFAULT ((1)) NULL,
    PRIMARY KEY CLUSTERED ([CartId] ASC),
    FOREIGN KEY ([ProductId]) REFERENCES [dbo].[Products] ([ProductId]),
    CONSTRAINT [FK_Cart_ProductId] FOREIGN KEY ([ProductId]) REFERENCES [dbo].[Products] ([ProductId]) ON DELETE CASCADE
);
```

Fig: Cart Table Creation



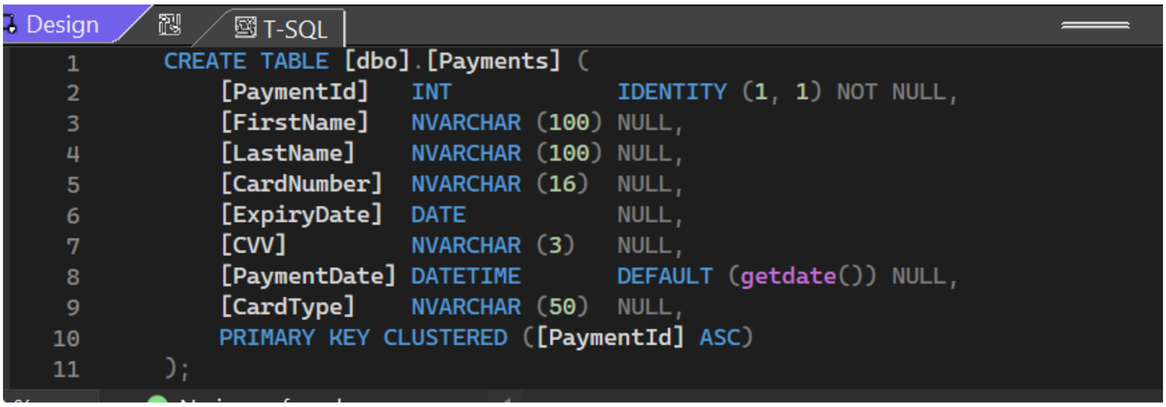
```sql
CREATE TABLE [dbo].[Categories] (
    [CategoryId]   INT          IDENTITY (1, 1) NOT NULL,
    [CategoryName] VARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([CategoryId] ASC)
);
```

Fig: Category Table Creation



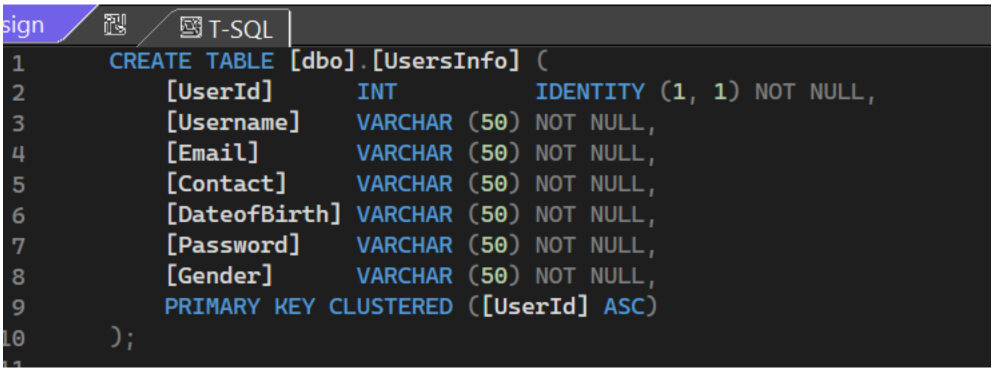
```sql
CREATE TABLE [dbo].[Products] (
    [ProductId]   INT             IDENTITY (1, 1) NOT NULL,
    [ProductName] VARCHAR (100)   NOT NULL,
    [Price]       DECIMAL (18, 2) NOT NULL,
    [CategoryId]  INT             NOT NULL,
    [ImagePath]   VARCHAR (255)   NULL,
    PRIMARY KEY CLUSTERED ([ProductId] ASC),
    FOREIGN KEY ([CategoryId]) REFERENCES [dbo].[Categories] ([CategoryId])
);
```

Fig: Product Table Creation



```sql
CREATE TABLE [dbo].[Payments] (
    [PaymentId]   INT            IDENTITY (1, 1) NOT NULL,
    [FirstName]   NVARCHAR (100) NULL,
    [LastName]    NVARCHAR (100) NULL,
    [CardNumber]  NVARCHAR (16)  NULL,
    [ExpiryDate]  DATE           NULL,
    [CVV]         NVARCHAR (3)   NULL,
    [PaymentDate] DATETIME       DEFAULT (getdate()) NULL,
    [CardType]    NVARCHAR (50)  NULL,
    PRIMARY KEY CLUSTERED ([PaymentId] ASC)
);
```

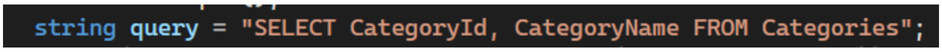
Fig: Payment Table Creation



```sql
CREATE TABLE [dbo].[UsersInfo] (
    [UserId]      INT          IDENTITY (1, 1) NOT NULL,
    [Username]    VARCHAR (50) NOT NULL,
    [Email]       VARCHAR (50) NOT NULL,
    [Contact]     VARCHAR (50) NOT NULL,
    [DateofBirth] VARCHAR (50) NOT NULL,
    [Password]    VARCHAR (50) NOT NULL,
    [Gender]      VARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([UserId] ASC)
);
```

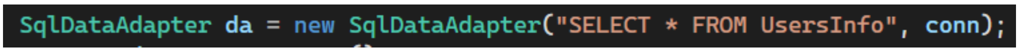
Fig: User Info Table

<u>b)Query in .cs file</u>



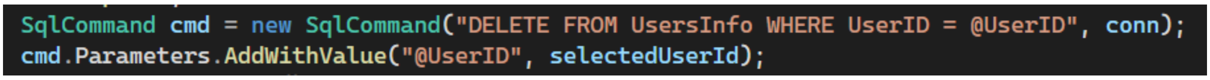
```csharp
string query = "SELECT CategoryId, CategoryName FROM Categories";
```

Selection from category table



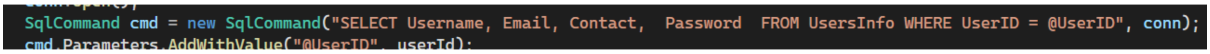
```csharp
SqlDataAdapter da = new SqlDataAdapter("SELECT * FROM UsersInfo", conn);
```

Selection all from UsersInfo table



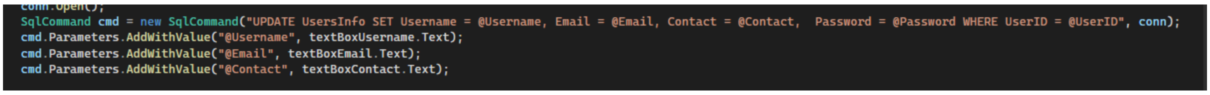
```csharp
SqlCommand cmd = new SqlCommand("DELETE FROM UsersInfo WHERE UserID = @UserID", conn);
cmd.Parameters.AddWithValue("@UserID", selectedUserId);
```

Delete from usersInfo table



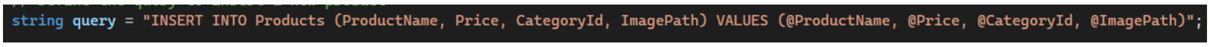
```csharp
SqlCommand cmd = new SqlCommand("SELECT Username, Email, Contact,  Password  FROM UsersInfo WHERE UserID = @UserID", conn);
cmd.Parameters.AddWithValue("@UserID", userId);
```

Select only selected items from usersinfo



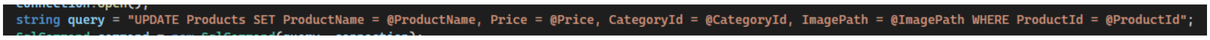
```csharp
conn.Open();
SqlCommand cmd = new SqlCommand("UPDATE UsersInfo SET Username = @Username, Email = @Email, Contact = @Contact,  Password = @Password WHERE UserID = @UserID", conn);
cmd.Parameters.AddWithValue("@Username", textBoxUsername.Text);
cmd.Parameters.AddWithValue("@Email", textBoxEmail.Text);
cmd.Parameters.AddWithValue("@Contact", textBoxContact.Text);
```

Update data in Usersinfo table



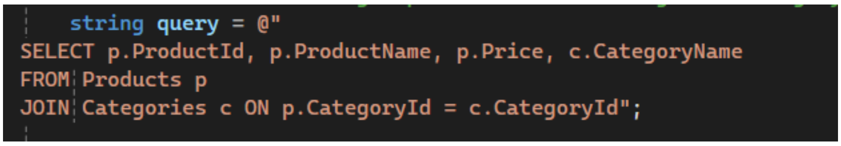
```csharp
string query = "INSERT INTO Products (ProductName, Price, CategoryId, ImagePath) VALUES (@ProductName, @Price, @CategoryId, @ImagePath)";
```

Insertion for Products Table



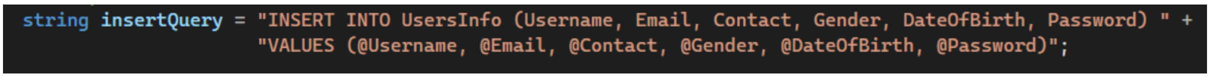
```csharp
string query = "UPDATE Products SET ProductName = @ProductName, Price = @Price, CategoryId = @CategoryId, ImagePath = @ImagePath WHERE ProductId = @ProductId";
```

Update and set data for Products Table



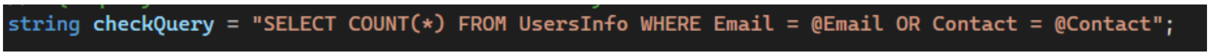
```csharp
    string query = @"
SELECT p.ProductId, p.ProductName, p.Price, c.CategoryName
FROM Products p
JOIN Categories c ON p.CategoryId = c.CategoryId";
```

Join Products and Category Table



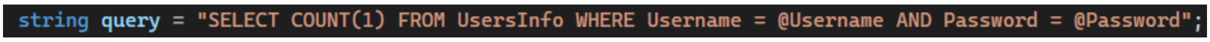
```csharp
string insertQuery = "INSERT INTO UsersInfo (Username, Email, Contact, Gender, DateOfBirth, Password) " +
                     "VALUES (@Username, @Email, @Contact, @Gender, @DateOfBirth, @Password)";
```

Insertion of users in UsersInfo Table



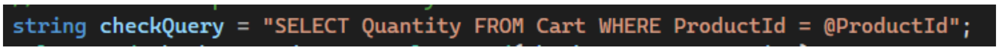
```
string checkQuery = "SELECT COUNT(*) FROM UsersInfo WHERE Email = @Email OR Contact = @Contact";
```

SQL query to check if email or contact already exists



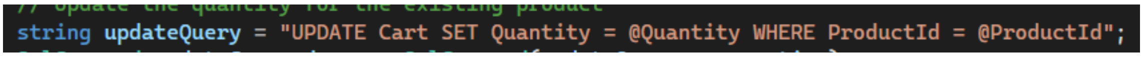
```
string query = "SELECT COUNT(1) FROM UsersInfo WHERE Username = @Username AND Password = @Password";
```

Username and password match



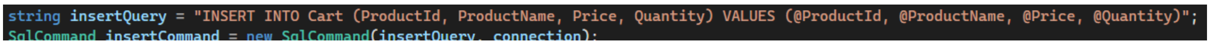
```
string checkQuery = "SELECT Quantity FROM Cart WHERE ProductId = @ProductId";
```

Selection quantity column from Cart table with condition



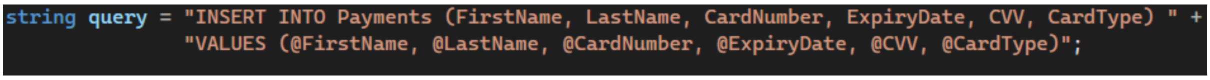
```
// Update the quantity for the existing product
string updateQuery = "UPDATE Cart SET Quantity = @Quantity WHERE ProductId = @ProductId";
```

Update Cart table with conditions



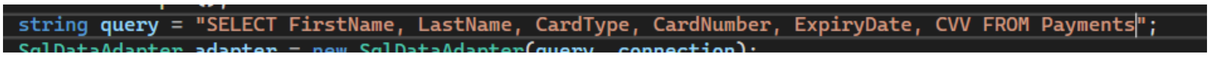
```
string insertQuery = "INSERT INTO Cart (ProductId, ProductName, Price, Quantity) VALUES (@ProductId, @ProductName, @Price, @Quantity)";
SqlCommand insertCommand = new SqlCommand(insertQuery, connection);
```

Insertion into Cart Table



```
string query = "INSERT INTO Payments (FirstName, LastName, CardNumber, ExpiryDate, CVV, CardType) " +
               "VALUES (@FirstName, @LastName, @CardNumber, @ExpiryDate, @CVV, @CardType)";
```

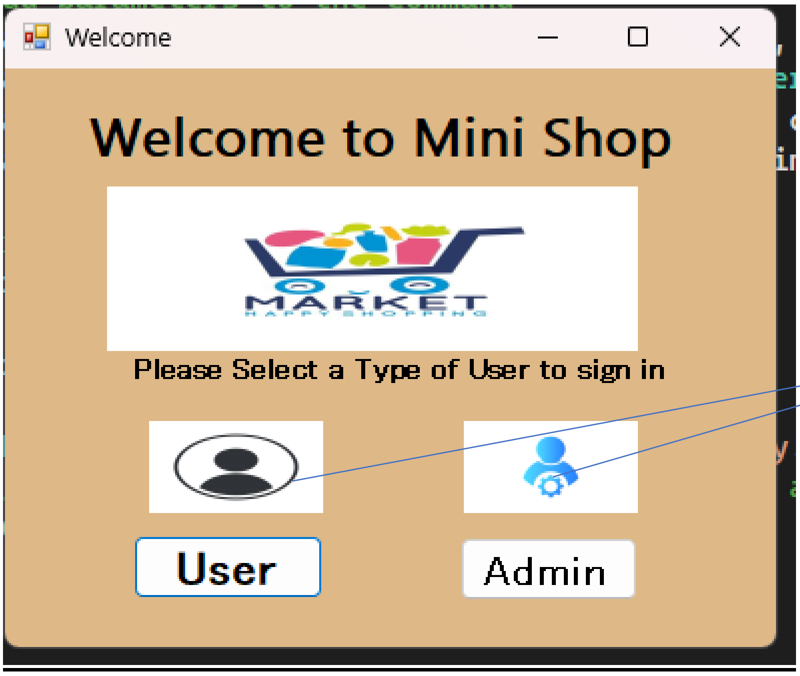
Insertion into Payment



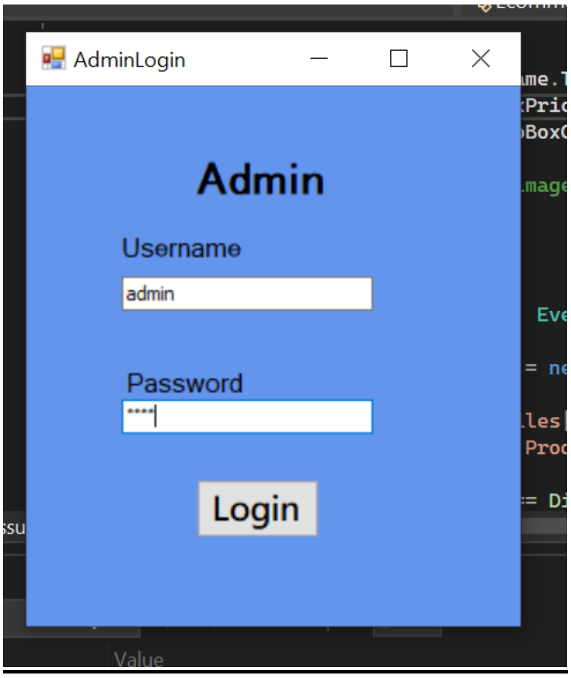
```
string query = "SELECT FirstName, LastName, CardType, CardNumber, ExpiryDate, CVV FROM Payments";
SqlDataAdapter adapter = new SqlDataAdapter(query, connection);
```

Selection from payments table

Chapter:5- Screenshots

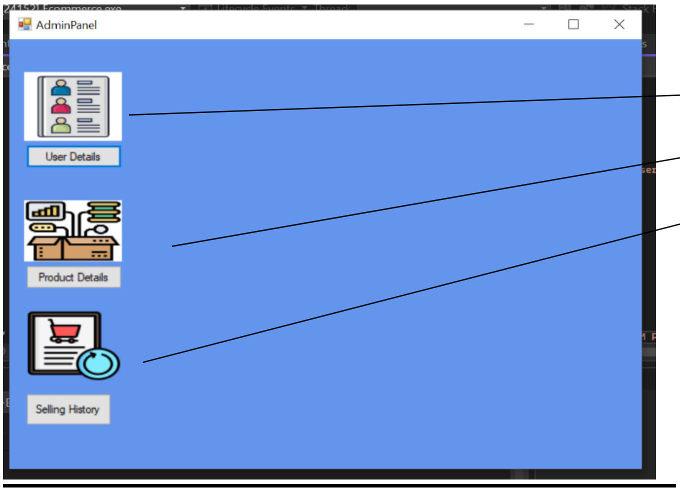


Two Types of User: Admin and User

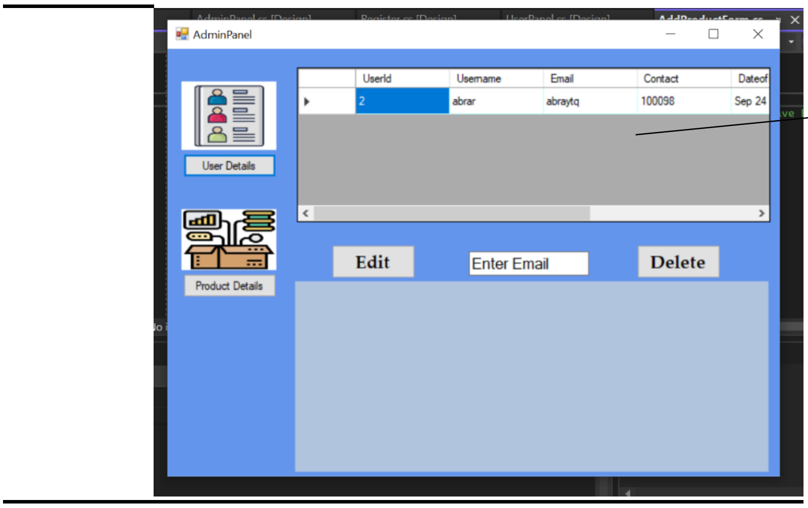


Username: admin

Password: ****

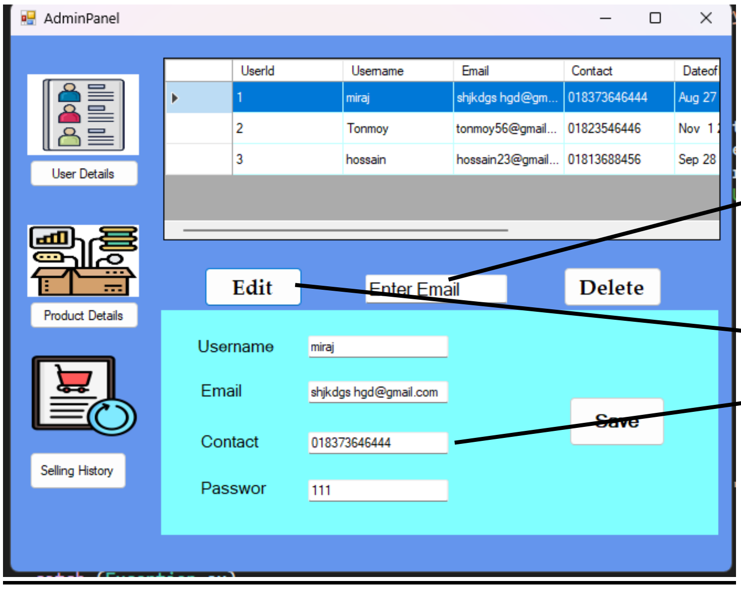


AdminPanel contains 3 options



UserId	Username	Email	Contact	Dateof
2	abrar	abraytq	100098	Sep 24

Datagrid for showing userinfo



UserId	Username	Email	Contact	Dateof
1	miraj	shjkdgs hgd@gm...	018373646444	Aug 27
2	Tonmoy	tonmoy56@gmail...	01823546446	Nov 1
3	hossain	hossain23@gmail...	01813688456	Sep 28

Username miraj

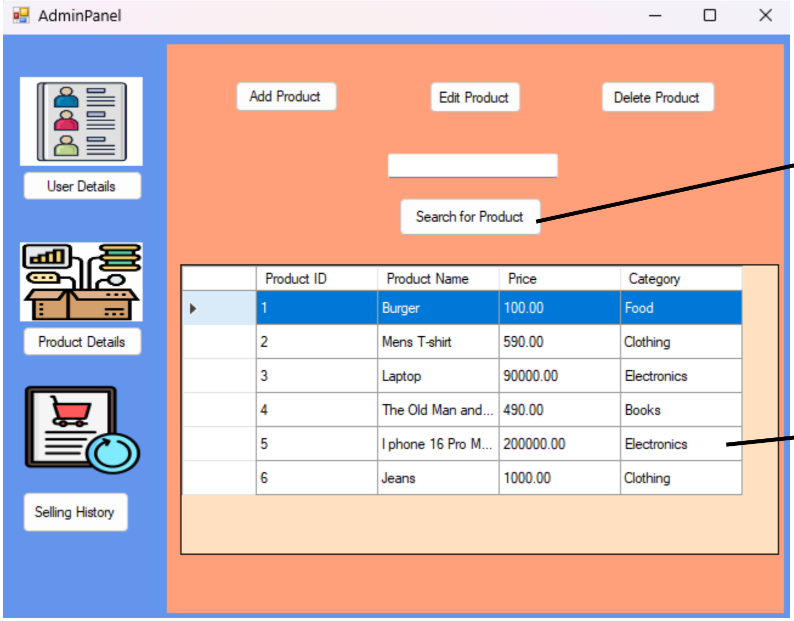
Email shjkdgs hgd@gmail.com

Contact 018373646444

Passwor 111

Search by dynamic

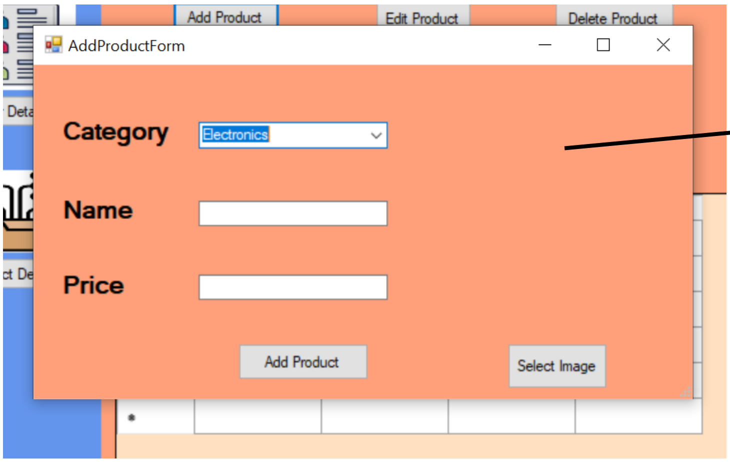
Edit button to enable this



	Product ID	Product Name	Price	Category
▸	1	Burger	100.00	Food
	2	Mens T-shirt	590.00	Clothing
	3	Laptop	90000.00	Electronics
	4	The Old Man and...	490.00	Books
	5	I phone 16 Pro M...	200000.00	Electronics
	6	Jeans	1000.00	Clothing

Usercontrol for Product Details

Datagrid for showing the product available

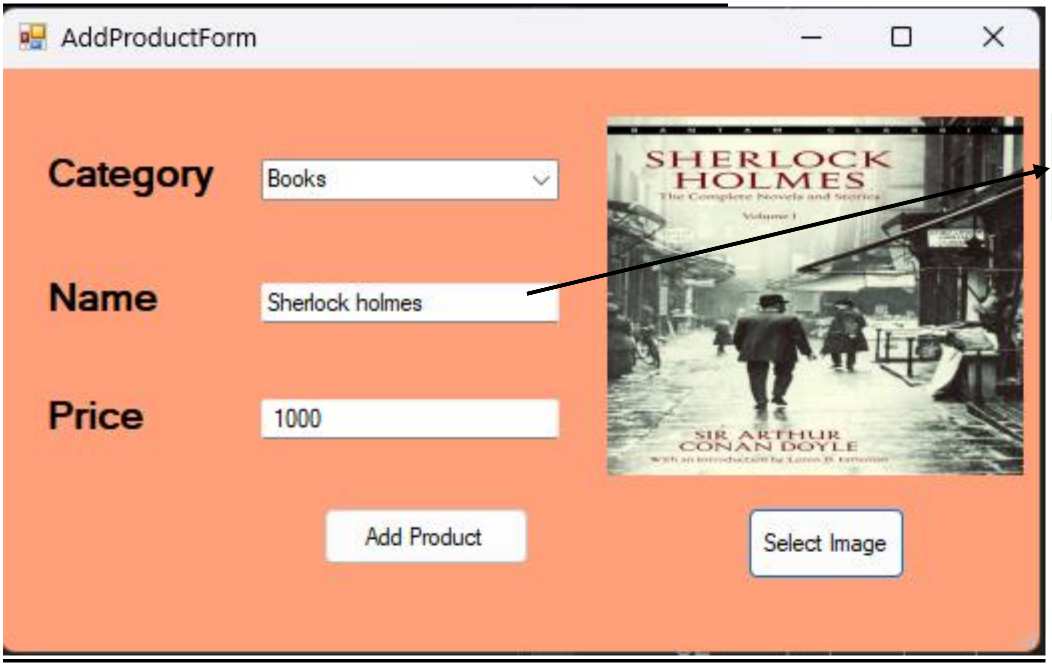


Category: Electronics

Name:

Price:

Add new product form

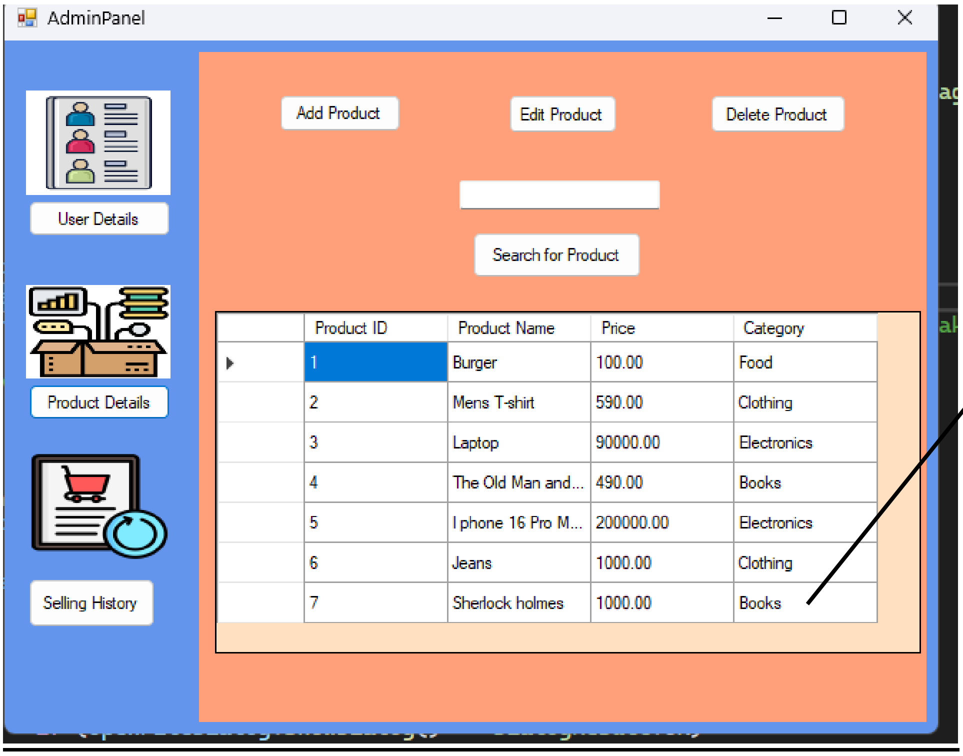


Category: Books

Name: Sherlock holmes

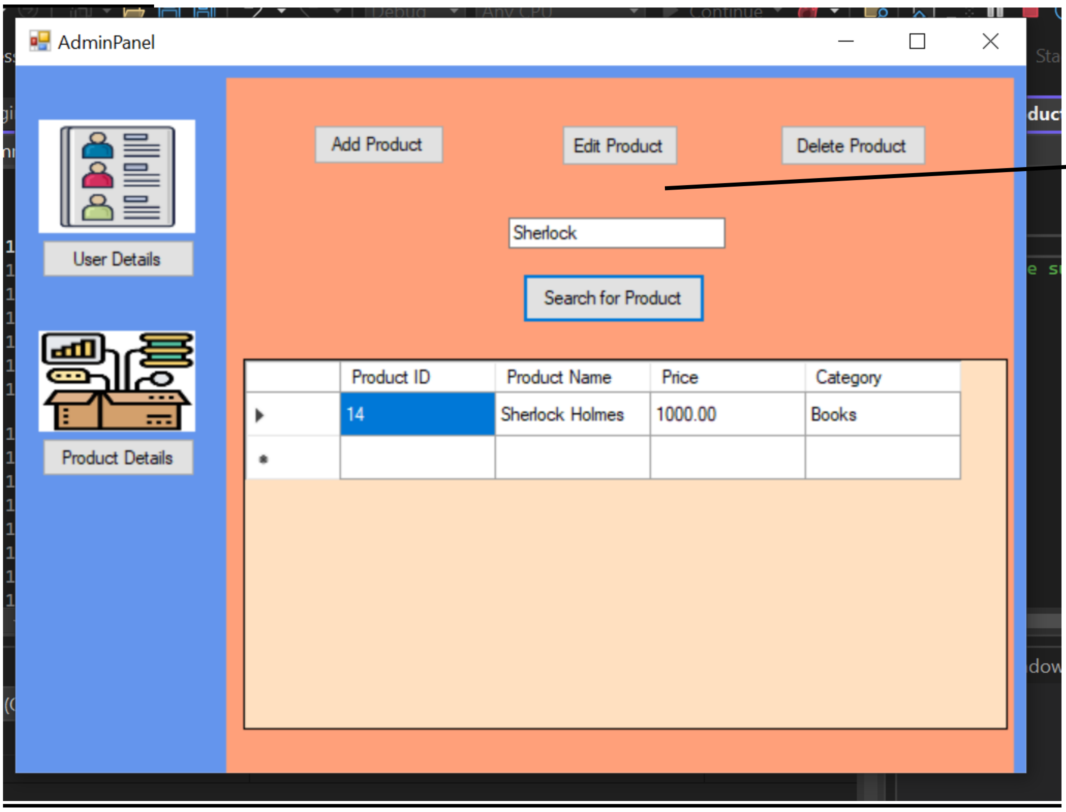
Price: 1000

New info and image added



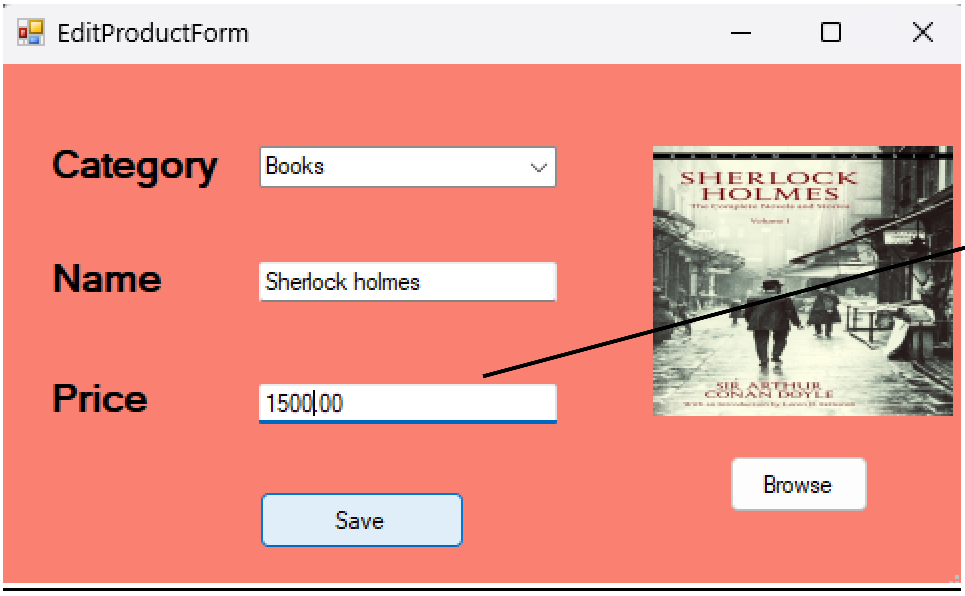
	Product ID	Product Name	Price	Category
▶	1	Burger	100.00	Food
	2	Mens T-shirt	590.00	Clothing
	3	Laptop	90000.00	Electronics
	4	The Old Man and...	490.00	Books
	5	I phone 16 Pro M...	200000.00	Electronics
	6	Jeans	1000.00	Clothing
	7	Sherlock holmes	1000.00	Books

New product books is added



	Product ID	Product Name	Price	Category
▶	14	Sherlock Holmes	1000.00	Books
*				

Search by name working

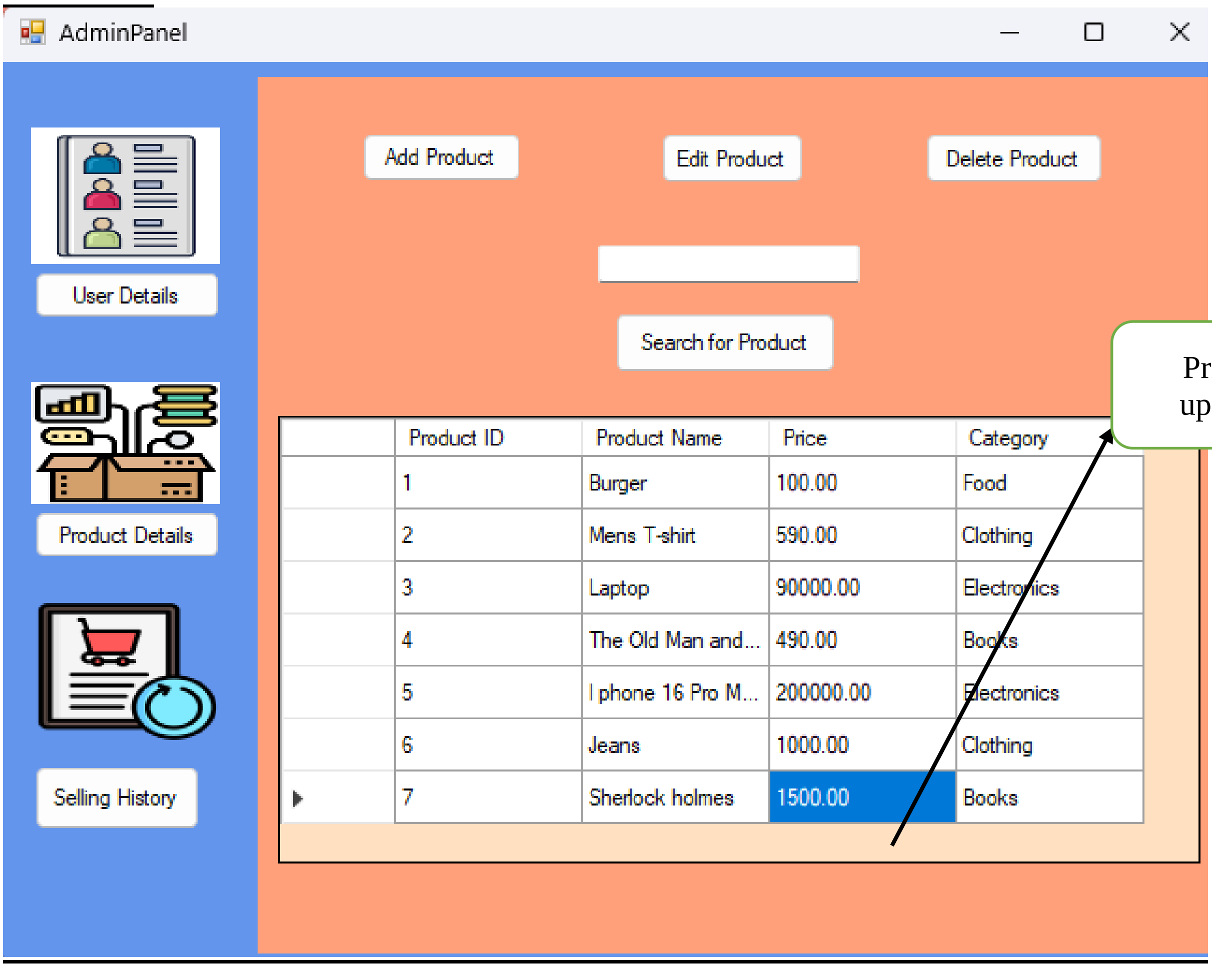


Category: Books

Name: Sherlock holmes

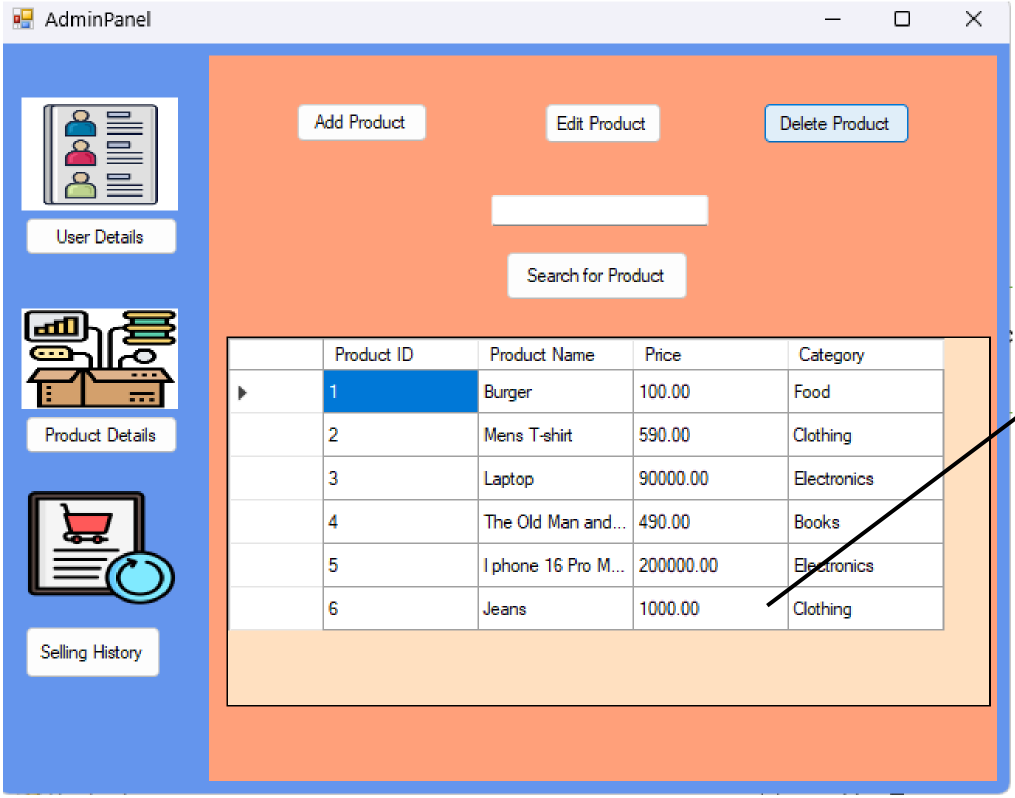
Price: 1500.00

Editing Price to 1500 instead of 1000



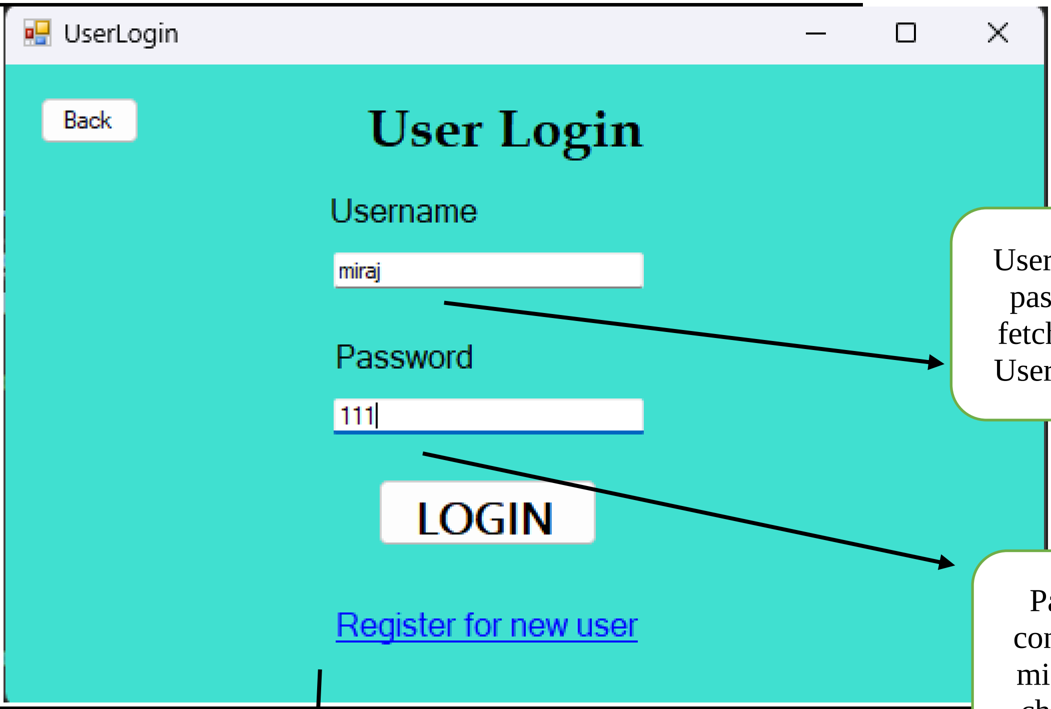
	Product ID	Product Name	Price	Category
	1	Burger	100.00	Food
	2	Mens T-shirt	590.00	Clothing
	3	Laptop	90000.00	Electronics
	4	The Old Man and...	490.00	Books
	5	I phone 16 Pro M...	200000.00	Electronics
	6	Jeans	1000.00	Clothing
▸	7	Sherlock holmes	1500.00	Books

Price is updated



	Product ID	Product Name	Price	Category
▸	1	Burger	100.00	Food
	2	Mens T-shirt	590.00	Clothing
	3	Laptop	90000.00	Electronics
	4	The Old Man and...	490.00	Books
	5	I phone 16 Pro M...	200000.00	Electronics
	6	Jeans	1000.00	Clothing

Sherlock Holmes product is Deleted



User Login

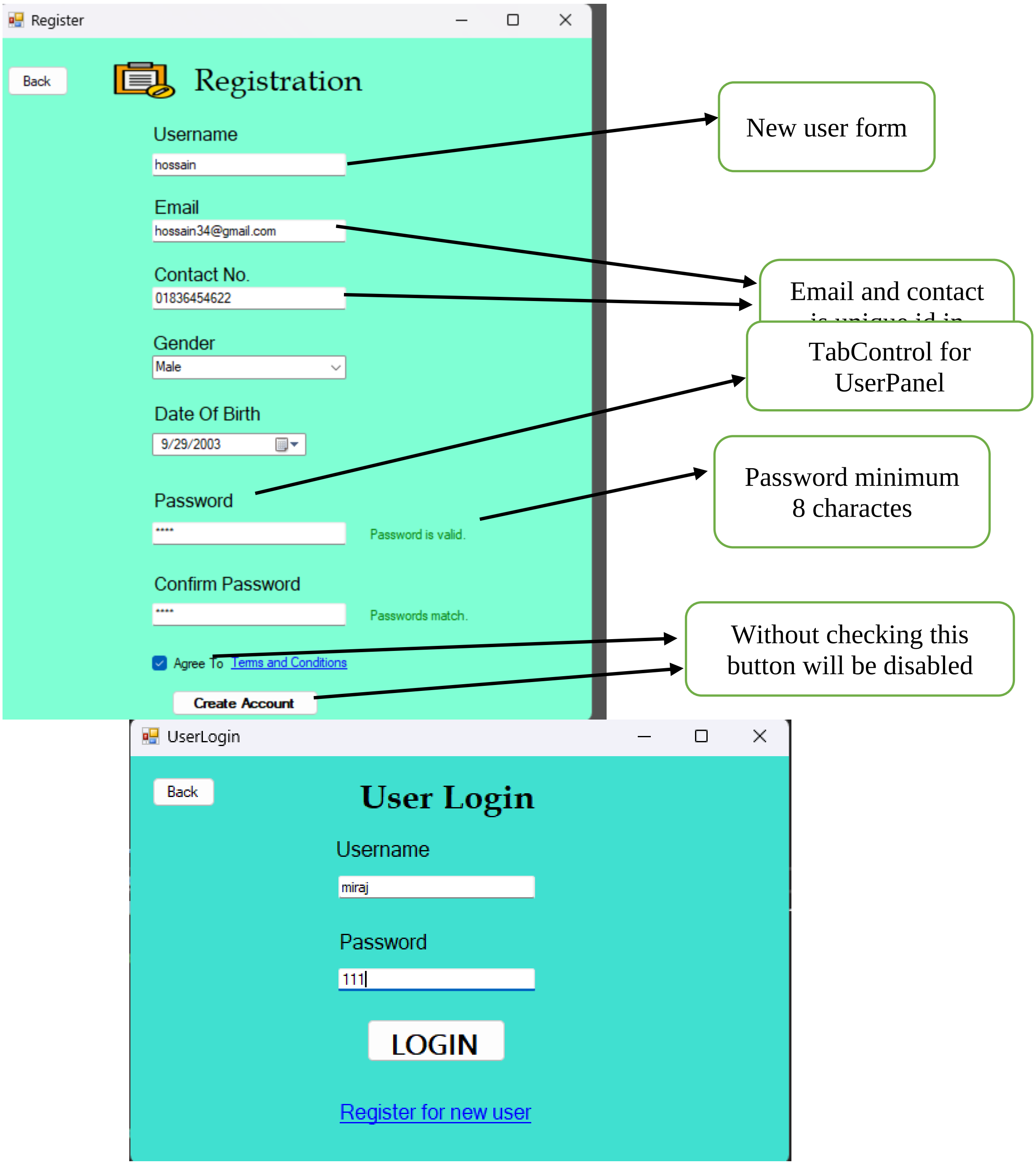
Username: miraj

Password: 111

Username and password is fetching form Userinfo table

Password constraint is minimum 3 chararcters

Create new user



Username: hossain

Email: hossain34@gmail.com

Contact No.: 01836454622

Gender: Male

Date Of Birth: 9/29/2003

Password: **** Password is valid.

Confirm Password: **** Passwords match.

☑ Agree To Terms and Conditions

New user form

Email and contact is unique id in

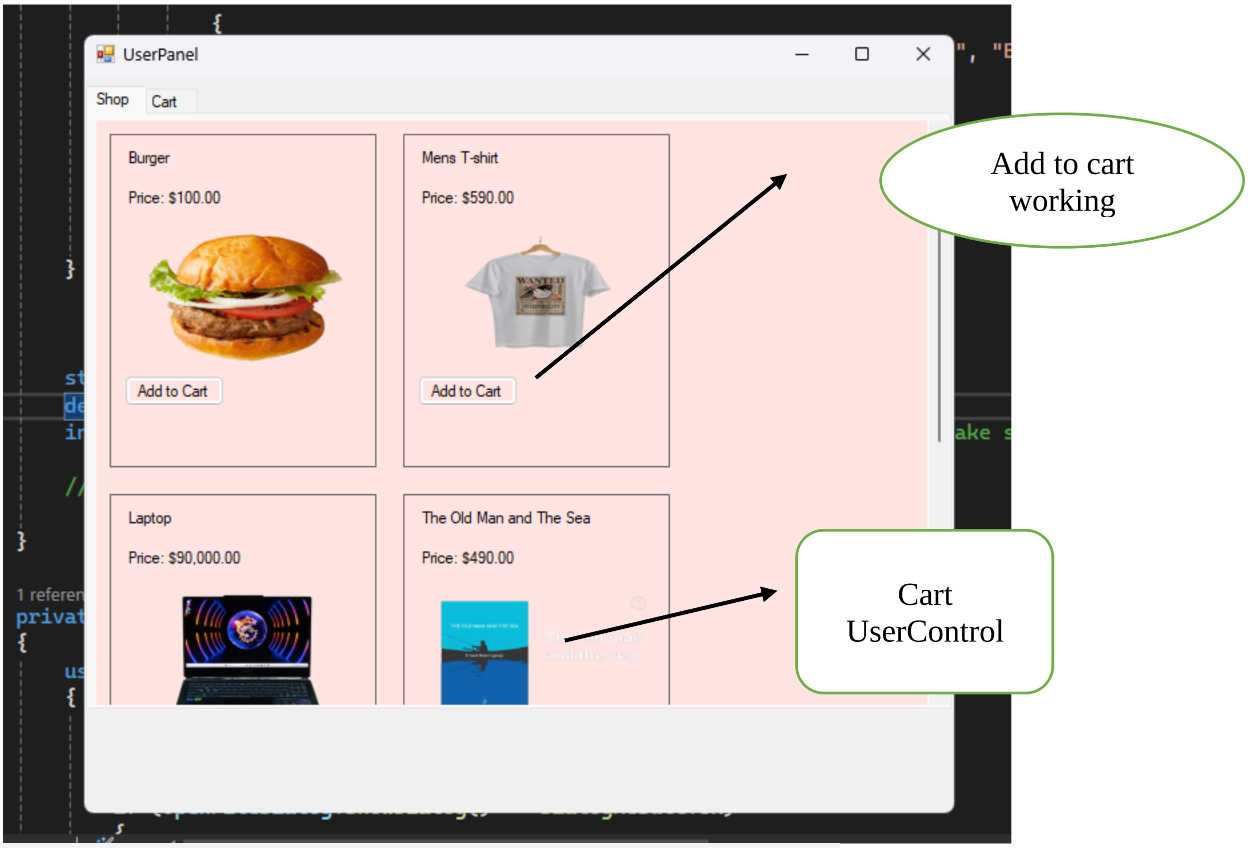
TabControl for UserPanel

Password minimum 8 charactes

Without checking this button will be disabled

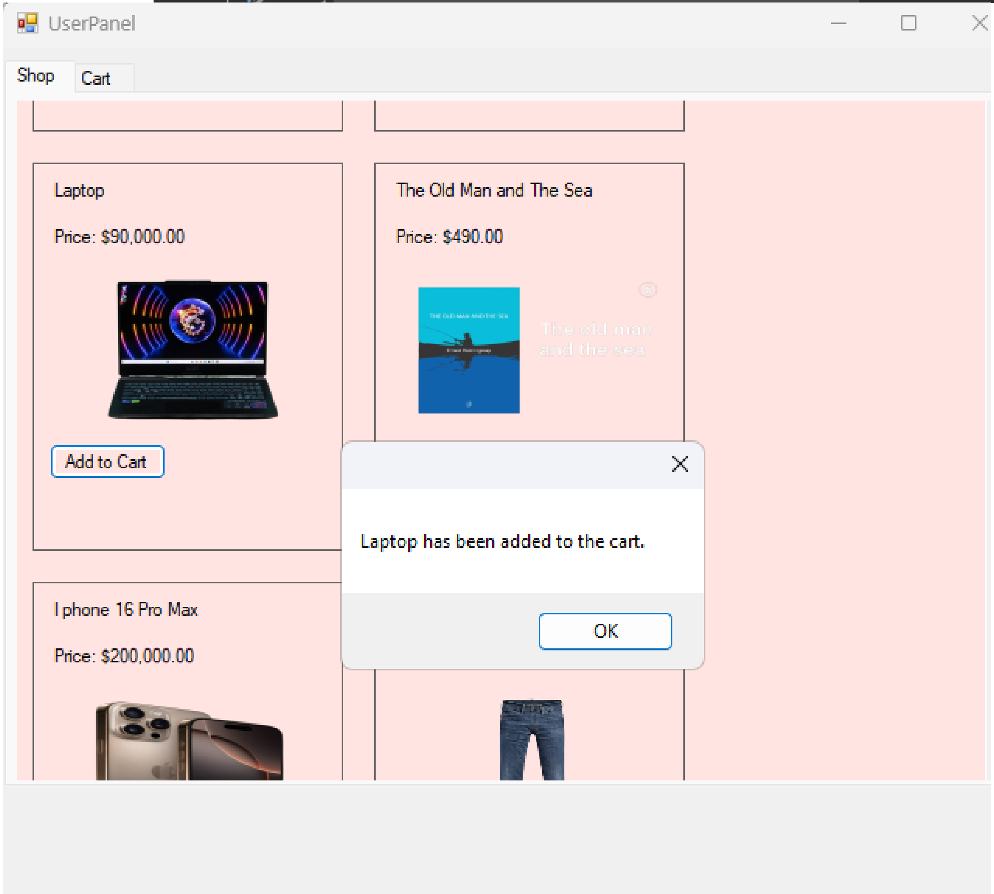
Username: miraj

Password: 111

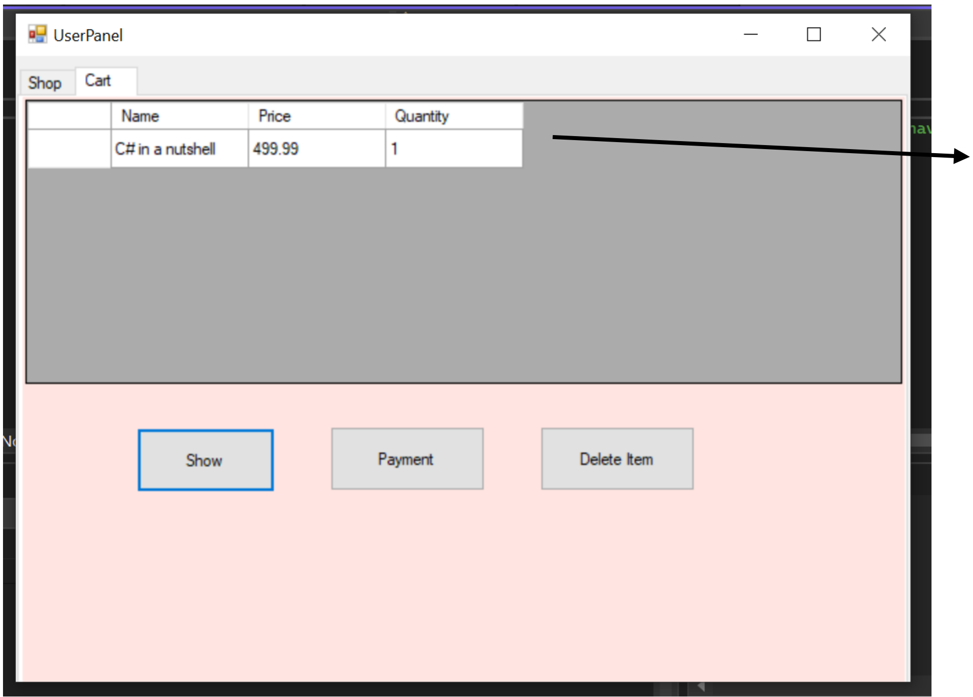


Add to cart working

Cart UserControl

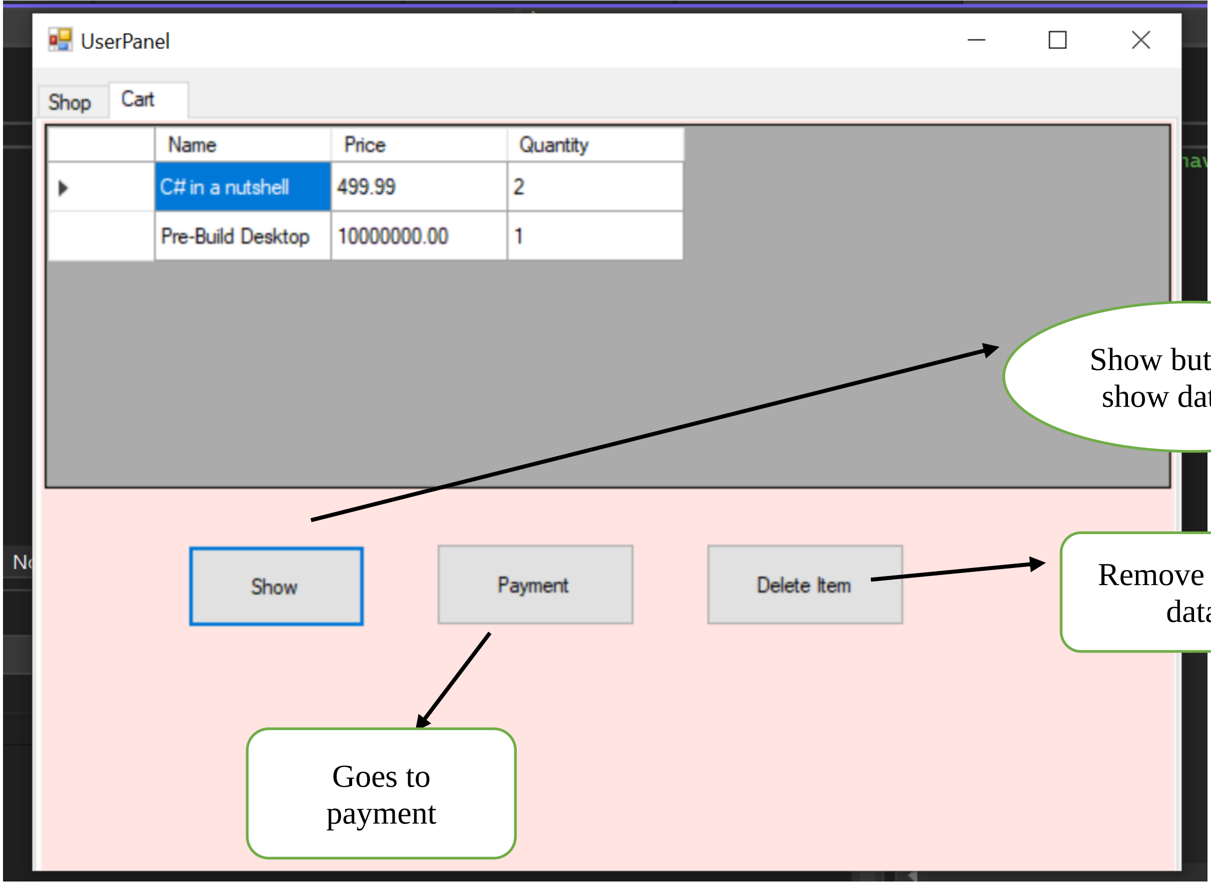


Laptop has been added to the cart.



Name	Price	Quantity
C# in a nutshell	499.99	1

Product added

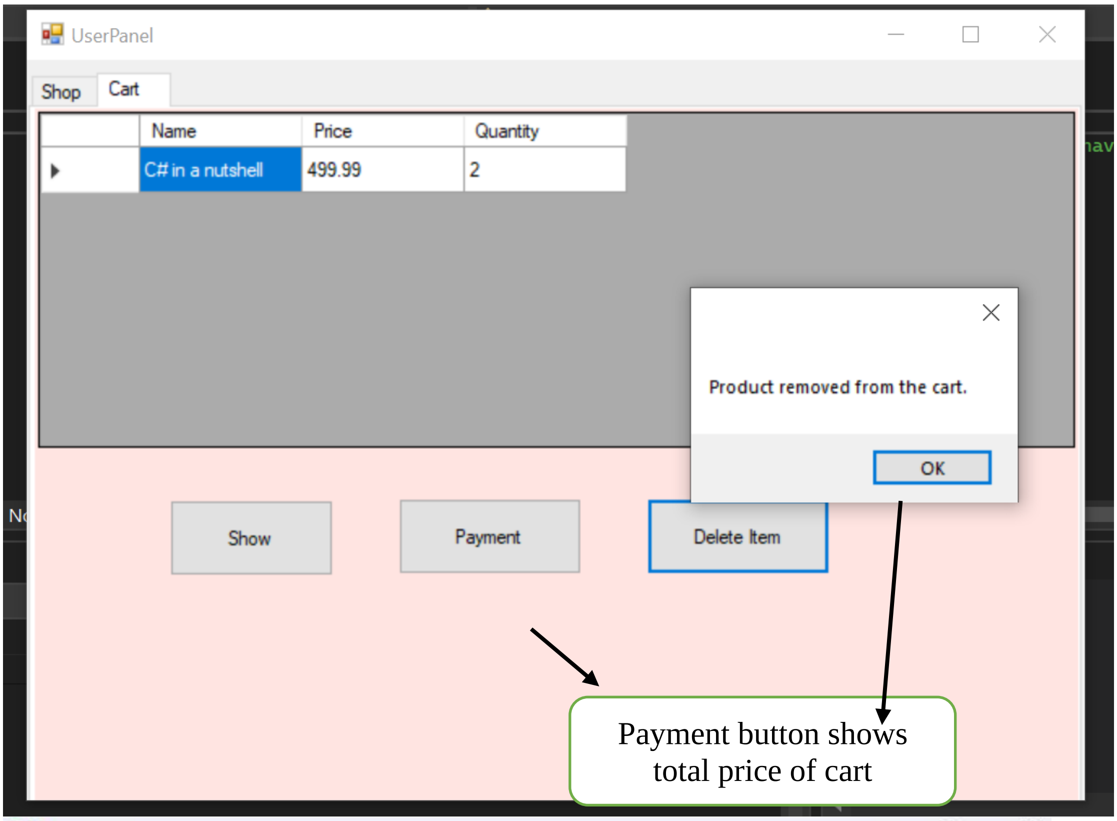


Name	Price	Quantity
C# in a nutshell	499.99	2
Pre-Build Desktop	10000000.00	1

Show button tot show datagrid

Remove item from datagrid

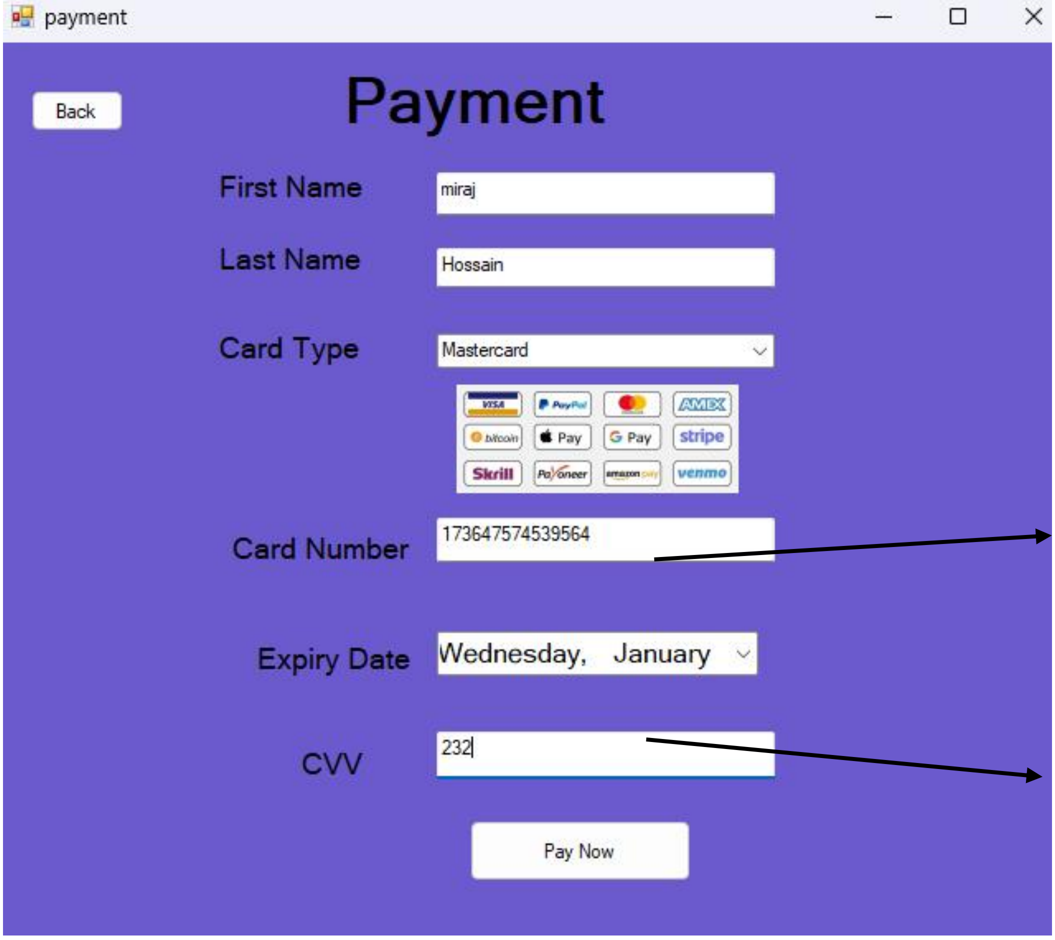
Goes to payment



	Name	Price	Quantity
▶	C# in a nutshell	499.99	2

Product removed from the cart.

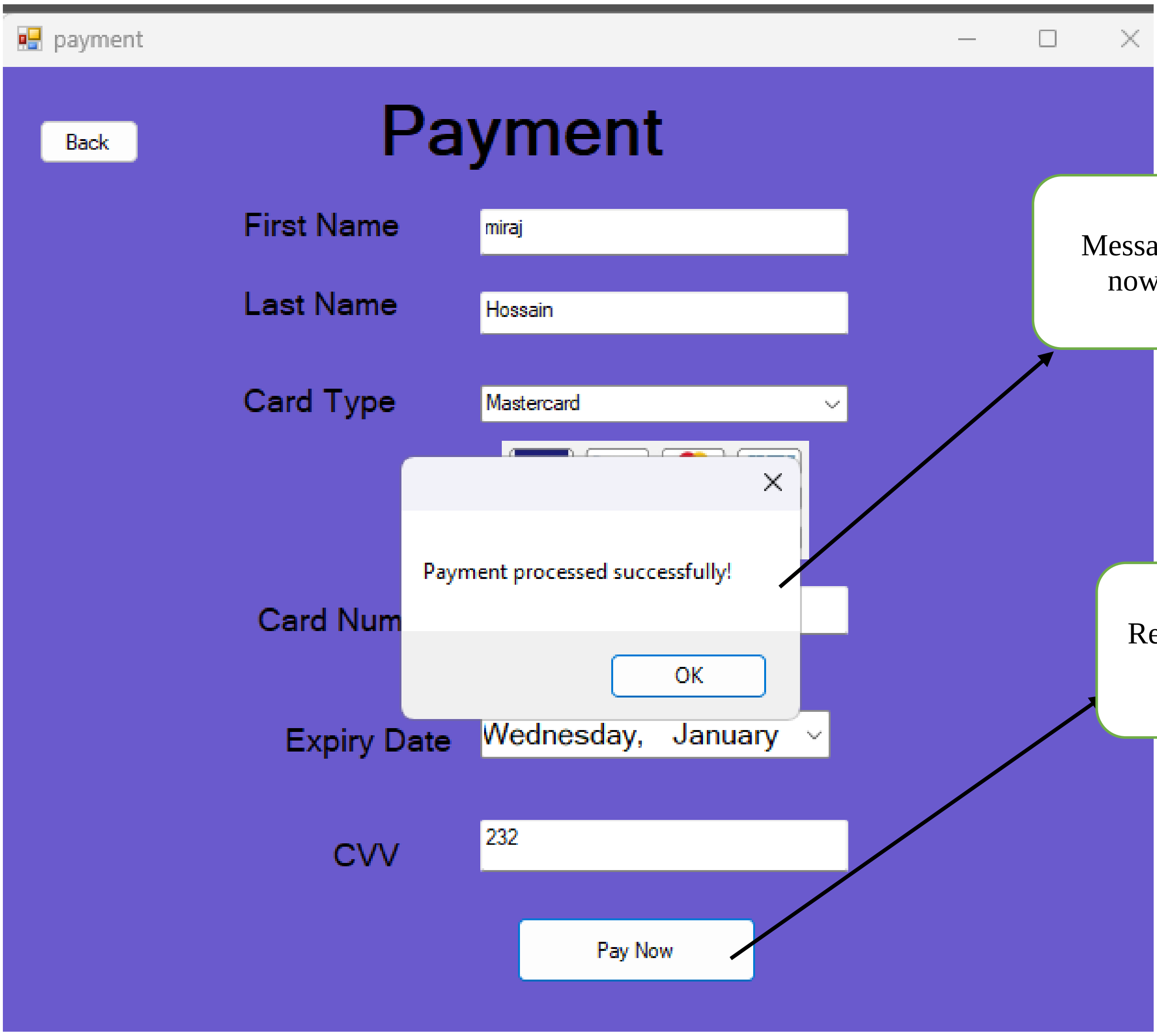
Payment button shows total price of cart



First Name	miraj
Last Name	Hossain
Card Type	Mastercard
Card Number	173647574539564
Expiry Date	Wednesday, January
CVV	232

number is 16 for card number

Cvv number is 3 (fixed)



payment

Payment

First Name: miraj

Last Name: Hossain

Card Type: Mastercard

Card Num

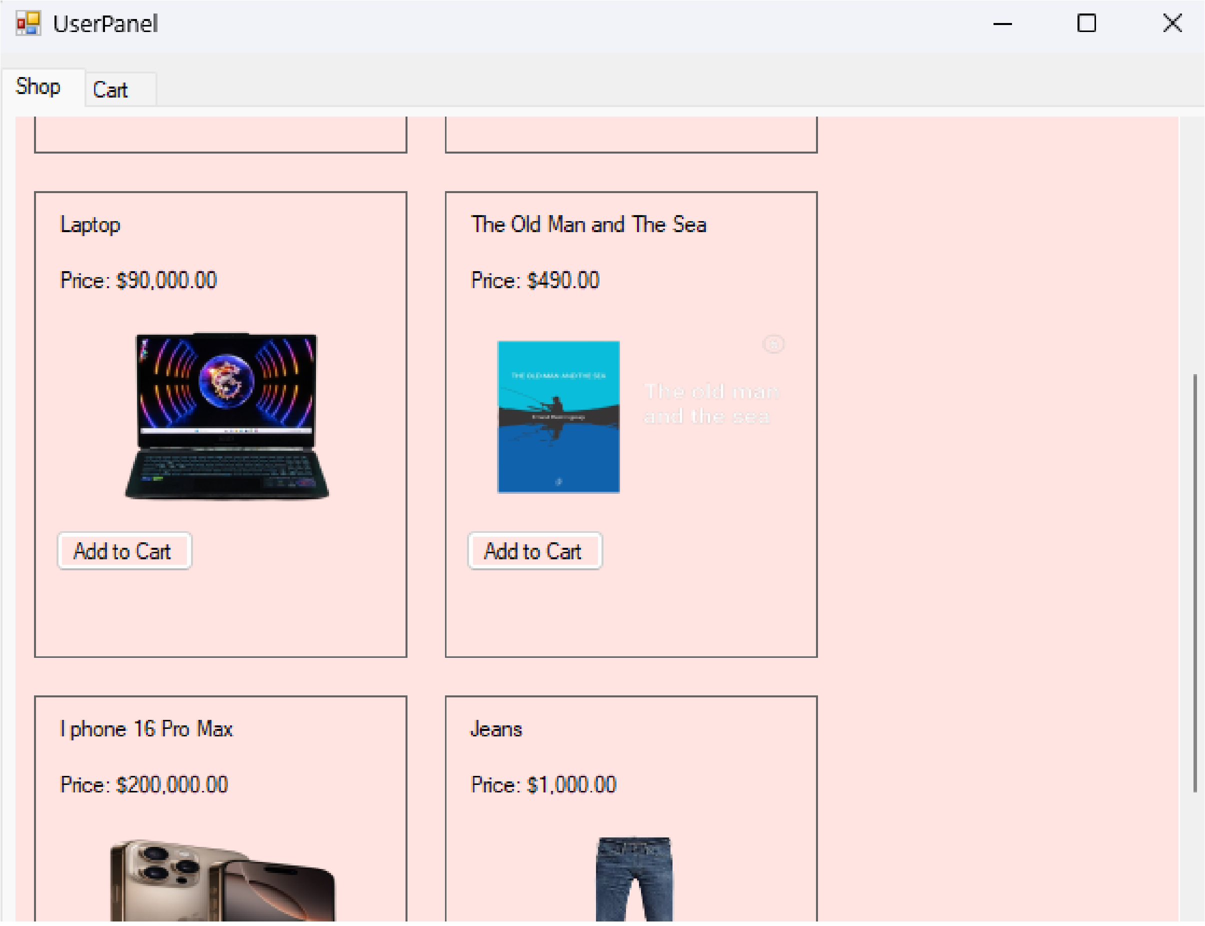
Payment processed successfully!

Expiry Date: Wednesday, January

CVV: 232

Message of pay now button

Returns back to shop



UserPanel

Laptop

Price: $90,000.00

The Old Man and The Sea

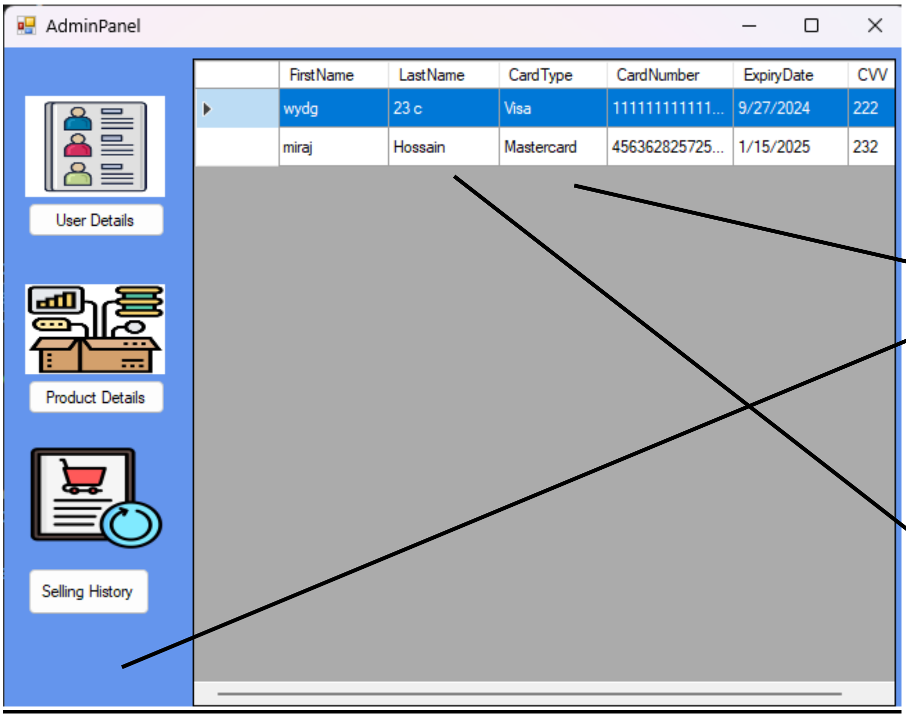
Price: $490.00

I phone 16 Pro Max

Price: $200,000.00

Jeans

Price: $1,000.00



AdminPanel

	FirstName	LastName	CardType	CardNumber	ExpiryDate	CVV
▸	wydg	23 c	Visa	111111111111...	9/27/2024	222
	miraj	Hossain	Mastercard	456362825725...	1/15/2025	232

Opens datagrid after hitting seller payment info

Fetches data from Payments