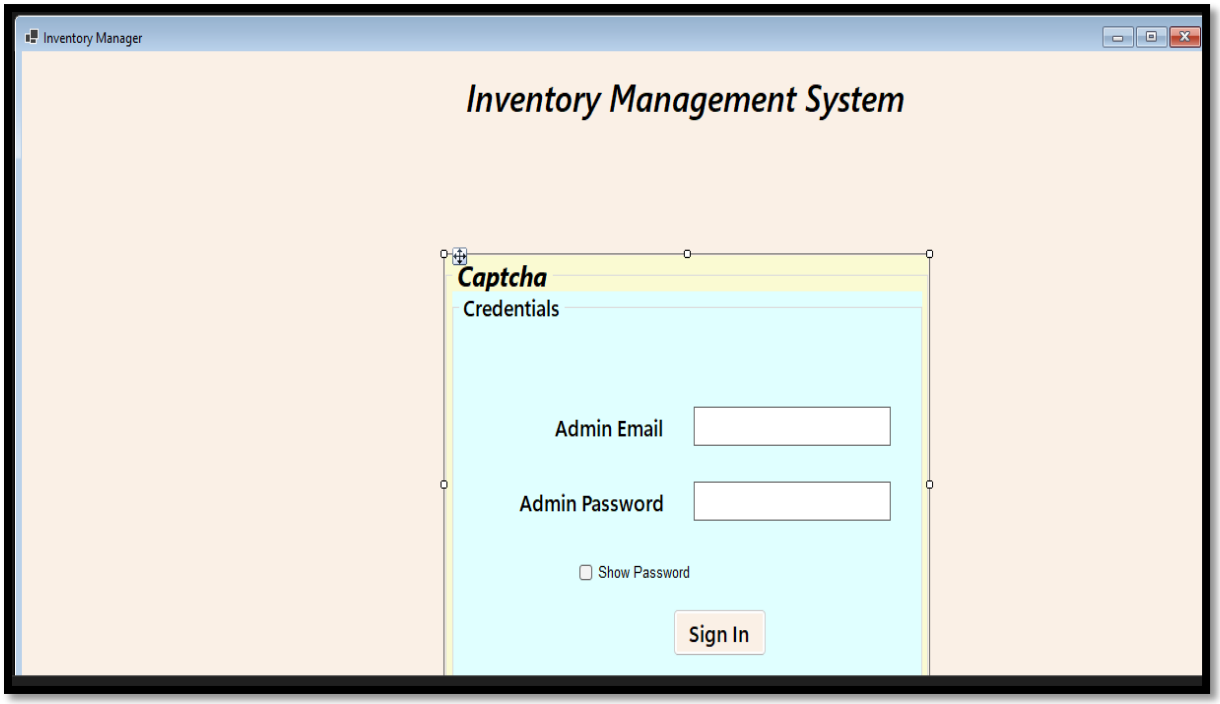
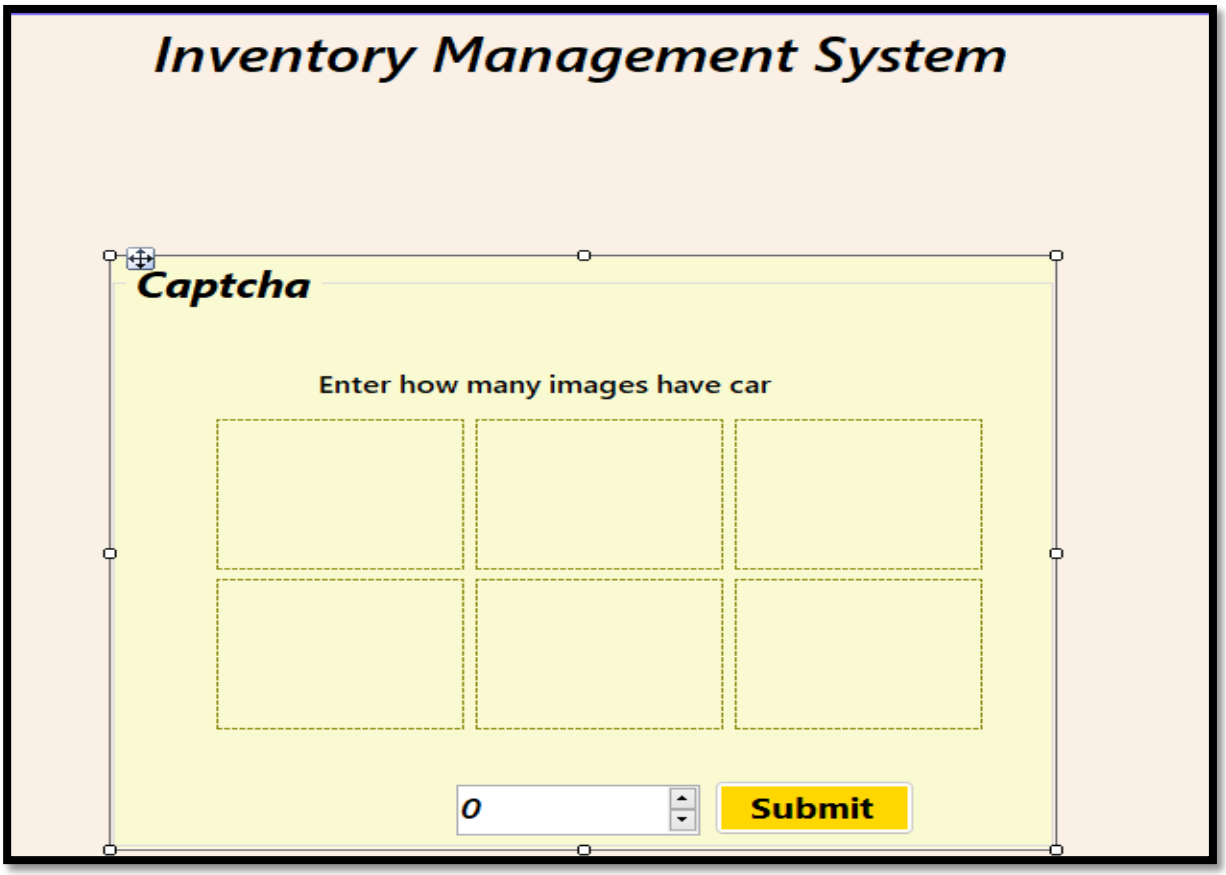
INVENTORY MANAGEMENT SYSTEM

Form1: Admin Login Page





Code:

```
using System;
using System.Data;
using System.Data.SqlClient;
namespace WinFormsApp1
{
    public partial class Form1 : Form   {
        SqlConnection conn;
        SqlCommand cmd;
        SqlDataReader dr;
        string str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;"; // srj pc
        int carCount = 0, roadCount = 0;
        public Form1()    {
            InitializeComponent();
        }
        private void LoadCaptcha()    {
            Random r = new Random();
            int[] randomIds = new int[6];
            HashSet<int> generatedIds = new HashSet<int>(); // for random Ids
            for (int i = 0; i < 6; i++)    {
                int newId;
                do    {
                    newId = r.Next(1, 21); // Generates a number between 1 and 20
                } while (generatedIds.Contains(newId)); // Ensure no duplicates
                generatedIds.Add(newId);
                randomIds[i] = newId;
            }
            conn = new SqlConnection(str);
            string query = $"SELECT ImgName , ImgPath FROM ImageCaptcha WHERE ImgId in
({randomIds[0]},{randomIds[1]}, {randomIds[2]},{randomIds[3]},{randomIds[4]},{randomIds[5]})";
            cmd = new SqlCommand(query, conn);
            conn.Open();
            dr = cmd.ExecuteReader();
            List<string> captchaImages = new List<string>();
```

```
            while (dr.Read())       {
                string path = dr["ImgPath"].ToString();
                string name = dr["ImgName"].ToString();
                if (name == "Car")        {
                    carCount++;
                    captchaImages.Add(path);
                }
                else if (name == "Road")       {
                    roadCount++;
                    captchaImages.Add(path);
                }
                if (carCount + roadCount == 6)         {
                    break;
                }      }
            dr.Close();      conn.Close();       cmd.Dispose();
            pictureBox1.Image = Image.FromFile(captchaImages.ElementAt(0));
            pictureBox2.Image = Image.FromFile(captchaImages.ElementAt(1));
            pictureBox3.Image = Image.FromFile(captchaImages.ElementAt(2));
            pictureBox4.Image = Image.FromFile(captchaImages.ElementAt(3));
            pictureBox5.Image = Image.FromFile(captchaImages.ElementAt(4));
            pictureBox6.Image = Image.FromFile(captchaImages.ElementAt(5));
        }
        private void button1_Click(object sender, EventArgs e)        {
            DBOperations dbobj = new DBOperations();
            //str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;";
            conn = new SqlConnection(str);
            string query = $"SELECT * FROM Admin where admin_email = '{usernameBox.Text}' and
admin_password = '{passwdBox.Text}' ";
            string adminEmail = "", adminPassword = "";
            SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            SqlDataReader dr = cmd.ExecuteReader();
            try      {
                if (dr.HasRows)     {
```

```
                dr.Read(); // only one record to read
                adminEmail = dr["admin_email"].ToString();
                adminPassword = dr["admin_password"].ToString();
                if (usernameBox.Text == adminEmail && passwdBox.Text == adminPassword)                    {
                    dbobj.setAdminLogs(dr["admin_id"].ToString(), dr["admin_name"].ToString(),
dr["admin_email"].ToString(), dr["admin_phone"].ToString(), DateTime.Now);
                    LoadCaptcha();
                    captchaBox1.Visible = true;
                }
                else        {
                    MessageBox.Show("Wrong Credentials!! Please try again", "Warning", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
                }      }
            else              {
                MessageBox.Show("No Admin Record Found", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }        }
        catch (Exception ex)           {
            MessageBox.Show($"{ex}");
        }
        finally           {
            conn.Close();    dr.Close();     cmd.Dispose();
        }       }
    private void checkBox1_CheckedChanged(object sender, EventArgs e)        {
        if (checkBox1.Checked)           {
            passwdBox.PasswordChar = '\0'; // null character
        }
        else if (!checkBox1.Checked)           {
            passwdBox.PasswordChar = '*';
        }       }
    private void button2_Click(object sender, EventArgs e)        {
        int user_ans = Convert.ToInt32(captchaAnswer.Value);
        if (user_ans == carCount)        {
            MessageBox.Show("Right Answer");
            carCount = 0;
```

```
            roadCount = 0;
            Form2 fm2 = new Form2();
            fm2.ShowDialog();
            captchaBox1.Visible = false;
            usernameBox.Clear();
            passwdBox.Clear();
        }
        else          {
            MessageBox.Show("Wrong Answer");
        }       }   }
// ------------------------------ IMS Entities -------------------------------------------
public class Customer    {
    public string cutomer_name { get; set; }
    public string customer_email { get; set; }
    public string customer_password { get; set; }
    public decimal customer_phone { get; set; }  // change in SQL
    public string customer_address { get; set; }
    public Customer(string name, string email, string password, decimal phone, string address)       {
        this.cutomer_name = name;
        this.customer_email = email;
        this.customer_password = password;
        this.customer_phone = phone;
        this.customer_address = address;
    }    } // Customer Class ends
class Admin    {
    int admin_id { get; }
    string admin_name { get; set; }
    string admin_email { get; set; }
    string admin_password { get; set; }
    string admin_phone { get; set; }
    string admin_address { get; set; }
    public Admin(string name, string email, string password, string phone, string address)       {
        this.admin_name = name;
        this.admin_email = email;
```

```
        this.admin_password = password;
        this.admin_phone = phone;
        this.admin_address = address;
    }   }
class Supplier  {
    string supplier_name { get; set; }
    string product_name { get; set; }
    string product_quantity { get; set; }
    string product_price { get; set; }
    string product_description { get; set; }
    DateTime dateOfSupply { get; set; }   // think about this
    int total_payment { get; set; }

} // Supplier class ends
public class Orders    {
    public int order_number { get; set; }
    public string customer_name { get; set; }
    public decimal order_amount { get; set; }
    public DateTime order_date { get; set; }
    public Orders(int orderNum, string custName, decimal orderAmt, DateTime orderDate)       {
        this.order_number = orderNum;
        this.customer_name = custName;
        this.order_amount = orderAmt;
        this.order_date = orderDate;
    }   } // Order class ends
class Stock    {
    int product_id { get; set; }    // automatically generated
    string product_name { get; set; }
    int product_quantity { get; set; }
    decimal product_price { get; set; }
    string product_description { get; set; }
    public Stock(int id, string name, int quantity, decimal price, string desc)       {
        this.product_id = id;
        this.product_name = name;
```

```
        this.product_price = price;
        this.product_quantity = quantity;
        this.product_description = desc;
    }    } // Stock class ends
public class Purchase   {
    public int purchase_id { get; }
    public string supplier_name { get; set; }
    public DateTime dateOfSupply { get; set; }
    public decimal totalPayment { get; set; }
    public Purchase(string supplierName, DateTime supplyDate, decimal payment)        {
        this.supplier_name = supplierName;
        this.dateOfSupply = supplyDate;
        this.totalPayment = payment;
    }    }
public class PurchaseDetails   {
    public int purchase_id { get; set; } // you need to set purchase id here
    public string product_name { get; set; }
    public int product_quantity { get; set; }
    public decimal product_price { get; set; }
    public string product_description { get; set; }
    public PurchaseDetails(int purId, string prodName, int prodQuantity, decimal prodPrice, string desc)      {
        this.purchase_id = purId;
        this.product_name = prodName;
        this.product_quantity = prodQuantity;
        this.product_price = prodPrice;
        this.product_description = desc;
    }   }
public class DBOperations   {
    private string str = "";
    public SqlConnection conn;
    public DBOperations()   {
        //str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;"; //srj pc
        conn = new SqlConnection(str);
    }
```

```
    public void DecreaseStock(Dictionary<string, int> dict)   {
        int affect = 0;
        try  {
            conn.Open();
            foreach (var item in dict)   {
                string productName = item.Key;
                int quantityToDecrease = item.Value;
                string query = $"UPDATE STOCK SET product_quantity = product_quantity - {quantityToDecrease}
WHERE product_name = '{productName}'";
                SqlCommand cmd = new SqlCommand(query, conn);
                affect += cmd.ExecuteNonQuery();
            }
            MessageBox.Show($"Rows Affected : {affect}");
            MessageBox.Show("Stock quantities have been updated successfully.");
        }
        catch (Exception ex)       {
            MessageBox.Show($"An error occurred while updating stock: {ex.Message}");
        }
        finally     {
            conn.Close();
        }       }
    public void IncreaseStock(string productName, int quantityToAdd, decimal productPrice, string
productDescription)
    {
        try       {
            conn.Open();
            string checkQuery = $"SELECT COUNT(*) FROM STOCK WHERE LOWER(product_name) =
'{productName.ToLower()}'";
            SqlCommand checkCmd = new SqlCommand(checkQuery, conn);
            int productExists = Convert.ToInt32(checkCmd.ExecuteScalar());
            if (productExists > 0)        {
                string updateQuery = $"UPDATE STOCK SET product_quantity = product_quantity + {quantityToAdd}
WHERE LOWER(product_name) = '{productName.ToLower()}'";
                SqlCommand updateCmd = new SqlCommand(updateQuery, conn);
                int rowsUpdated = updateCmd.ExecuteNonQuery();
```

```
                MessageBox.Show($"Product '{productName}' quantity increased. Rows Affected: {rowsUpdated}");
            }
            else        {
                string insertQuery = $"INSERT INTO STOCK (product_name, product_quantity, product_price, product_description) VALUES ('{productName}', {quantityToAdd}, {productPrice}, '{productDescription}')";
                SqlCommand insertCmd = new SqlCommand(insertQuery, conn);
                int rowsInserted = insertCmd.ExecuteNonQuery();
                MessageBox.Show($"New product '{productName}' added to stock. Rows Affected: {rowsInserted}");
            }         }
        catch (Exception ex)       {
            MessageBox.Show($"An error occurred while updating stock: {ex}");
        }
        finally       {
            conn.Close();
        }     }
    public void NewOrder(Orders obj)        {
        string query = $"INSERT INTO ORDERS  (customer_name, order_amount, order_date) VALUES ('{obj.customer_name}', {obj.order_amount}, '{obj.order_date.ToString("yyyy-MM-dd")}')";
        SqlCommand cmd = new SqlCommand(query, conn);
        conn.Open();
        cmd.ExecuteScalar();
        conn.Close();
        MessageBox.Show("Order Commited Successfully");
    }
    public void NewCustomer(Customer obj)      {
        try         {
            string query = $"INSERT INTO CUSTOMER VALUES ('{obj.cutomer_name}', '{obj.customer_email}', '{obj.customer_password}', {obj.customer_phone}, '{obj.customer_address}')";
            SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            cmd.ExecuteScalar();
            MessageBox.Show("New Customer Created Successfully");
        }
        catch (Exception ex)     {
```

```
            MessageBox.Show($"An Exception Occured : {ex}");
        }
        finally     {
            conn.Close();
        }       }
    public int NewPurchase(Purchase obj)       {
        int id = -1; // initially
        try        {
            string query = $"INSERT INTO Purchase (supplier_name, date_of_supply, total_payment) VALUES
('{obj.supplier_name}', '{obj.dateOfSupply.ToString("yyyy-MM-dd")}' , {obj.totalPayment}); " + "SELECT
SCOPE_IDENTITY();";
            SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            id = Convert.ToInt32(cmd.ExecuteScalar());
        }
        catch (Exception ex)       {
            MessageBox.Show($"{ex}");
        }
        finally        {
            conn.Close();
        }
        return id; // latest purchase id
    }
    public void NewPurchaseDetails(PurchaseDetails obj)       {
        try        {
            string query = $"INSERT INTO PurchaseDetails (purchase_id, product_name, product_quantity,
product_price, product_description) VALUES ({obj.purchase_id}, '{obj.product_name}', {obj.product_quantity},
{obj.product_price}, '{obj.product_description}');";
            SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            cmd.ExecuteScalar();
        }
        catch (Exception ex)       {
            MessageBox.Show($"{ex}");
        }
```

```
    finally     {
        conn.Close();
    }
    IncreaseStock(obj.product_name, obj.product_quantity, obj.product_price, obj.product_description)
}
public DataTable ShowStockData(string paraType, string paraEntry)     {
    DataTable dt = new DataTable();  // Create a DataTable to hold the data
    try         {
        string query = "";
        if (paraType == "Complete Stock")        {
            query = "SELECT * FROM STOCK";
        }
        else if (paraType == "Product Name")          {
            query = $"SELECT * FROM STOCK WHERE product_name LIKE '%{paraEntry}%'";
        }
        else if (paraType == "Product Quantity")          {
            query = $"SELECT * FROM STOCK WHERE product_quantity = '{paraEntry}'";
        }
        conn.Open();
        SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
        sqlDa.Fill(dt);
    }
    catch (Exception ex)   {
        MessageBox.Show($"An exception occurred: {ex.Message}");
    }
    finally  {
        conn.Close();
    }
    return dt;
}
public DataTable ShowOrderData(string paraType, string paraEntry)     {
    DataTable dt = new DataTable();
    try    {
        string query = "";
```

```
        if (paraType == "All")    {
            query = $"SELECT * FROM Orders";
        }
        else if (paraType == "Order Number")          {
            query = $"SELECT * FROM Orders WHERE order_number = {Convert.ToInt32(paraEntry)} ";
        }
        else if (paraType == "Customer Name")      {
            query = $"SELECT * FROM Orders WHERE customer_name LIKE '%{paraEntry}%' ";
        }
        else if (paraType == "Order Amount")    {
            query = $"SELECT * FROM Orders WHERE order_amount = {Convert.ToInt32(paraEntry)} ";
        }
        else if (paraType == "Date")       {
            query = $"SELECT * FROM Orders WHERE order_date = '{paraEntry}' ";
        }
        conn.Open();
        if(string.IsNullOrEmpty(query))      {
            MessageBox.Show("query empty");
        }
        SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
        sqlDa.Fill(dt);
    }
    catch (Exception ex)       {
        MessageBox.Show($"An exception occured\n : {ex}");
    }
    finally       {
        conn.Close();
    }
    return dt;
}
public DataTable ShowOrderData(string paraType, string paraEntry1, string paraEntry2)      {
    DataTable dt = new DataTable();
    try       {
        string query = "";
```

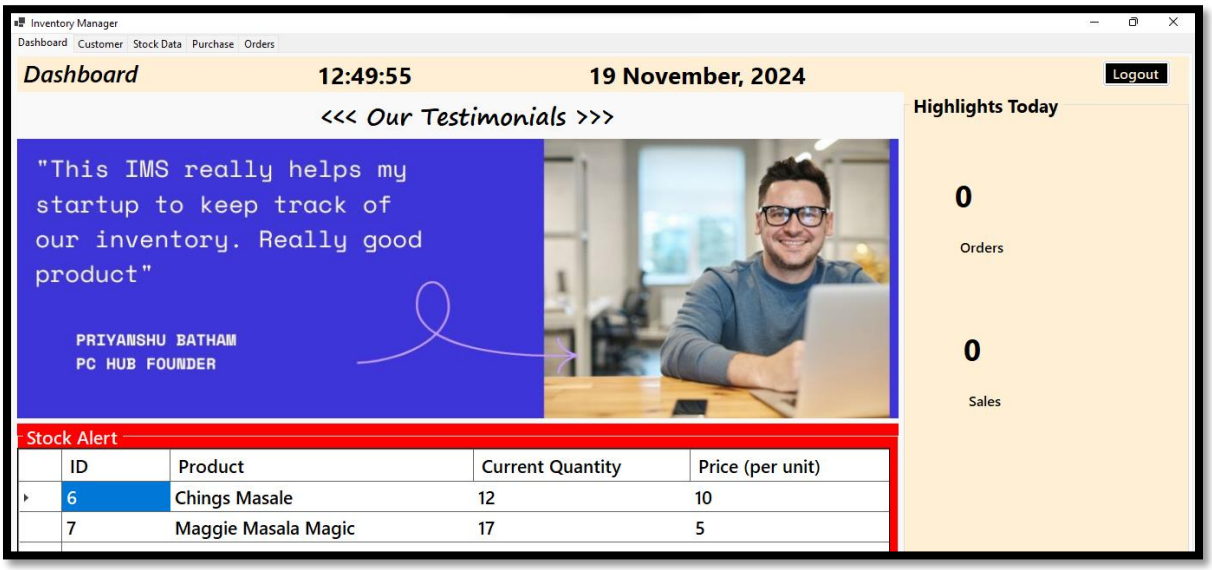
```
            if (paraType == "Date Range")        {
                query = $"SELECT * FROM Orders WHERE order_date between '{paraEntry1}' AND '{paraEntry2}' ";
            }
            else if (paraType == "Price Range")         {
                query = $"SELECT * FROM Orders WHERE order_amount between {Convert.ToInt32(paraEntry1)}
AND {Convert.ToInt32(paraEntry2)} ";
            }
            conn.Open();
            if (string.IsNullOrEmpty(query))            {
                MessageBox.Show("query empty");
            }
            SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
            sqlDa.Fill(dt);
        }
        catch (Exception ex)         {
            MessageBox.Show($"{ex}");
        }
        finally      {
            conn.Close();
        }
        return dt;
    }
    public DataTable ShowPurchaseData(string paraType, string paraEntry) // single paraEntry
    {
        DataTable dt = new DataTable();
        try     {
            string query = "";
            if (paraType == "All")               {
                query = $"SELECT * FROM Purchase";
            }
            else if (paraType == "Date")               {
                query = $"SELECT * FROM Purchase WHERE date_of_supply = '{paraEntry}'";
            }
            else if (paraType == "Supplier")              {
```

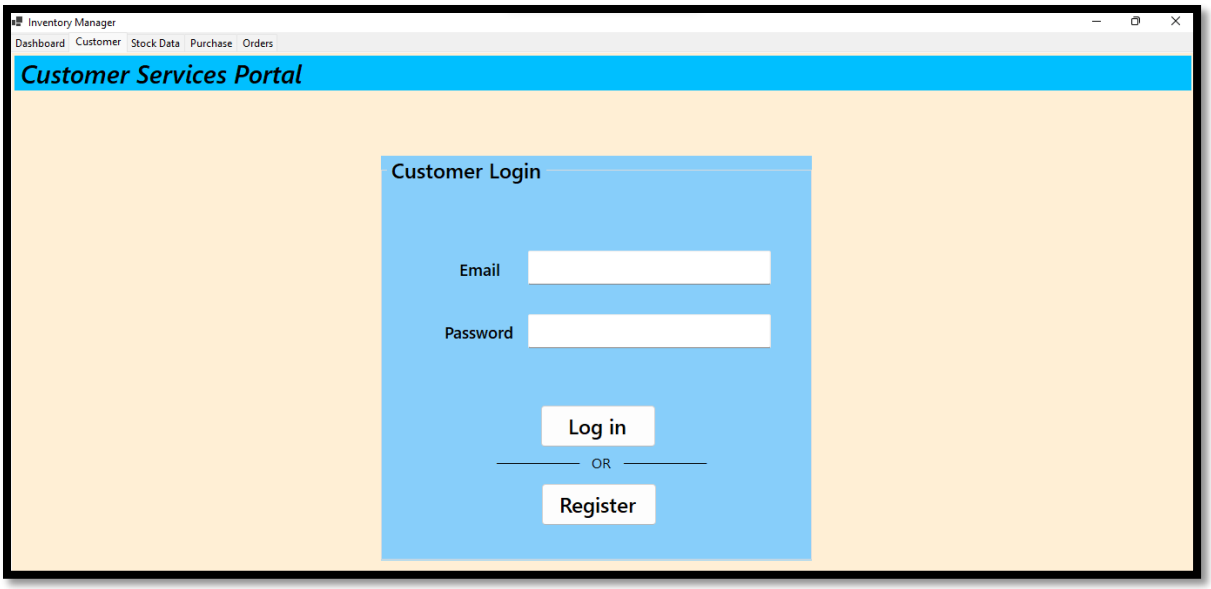
```
                query = $"SELECT * FROM Purchase WHERE supplier_name LIKE '%{paraEntry}%' ";
            }
            else if (paraType == "Minimum Price")            {
                query = $"SELECT * FROM Purchase WHERE total_payment > {Convert.ToDecimal(paraEntry)} ";
            }
            else if (paraType == "Maximum Price")              {
                query = $"SELECT * FROM Purchase WHERE total_payment < {Convert.ToDecimal(paraEntry)} ";
            }
            conn.Open();
            SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
            sqlDa.Fill(dt);
        }
        catch (Exception ex)     {
            MessageBox.Show($"{ex}");
        }
        finally    {
            conn.Close();
        }
        return dt;
    }
    public DataTable ShowPurchaseData(string paraType, string paraEntry1, string paraEntry2)
    {
        DataTable dt = new DataTable();
        try       {
            string query = "";
            if (paraType == "Date Range")          {
                query = $"SELECT * FROM Purchase WHERE date_of_supply between '{paraEntry1}' AND
'{paraEntry2}' ";
            }
            else if (paraType == "Price Range")         {
                query = $"SELECT * FROM Purchase WHERE total_payment between
'{Convert.ToDecimal(paraEntry1)}' AND '{Convert.ToDecimal(paraEntry2)}' ";
            }
            conn.Open();
            SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
```

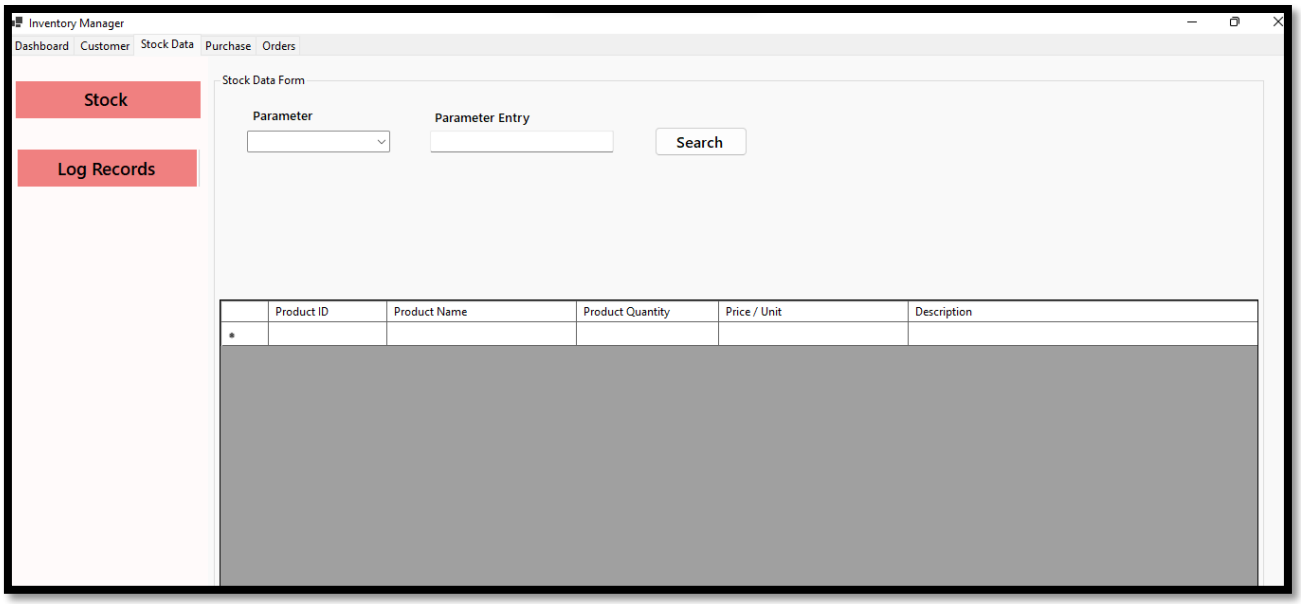
```
            sqlDa.Fill(dt);
        }
        catch (Exception ex)        {
            MessageBox.Show($"{ex}");
        }
        finally     {
            conn.Close();
        }
        return dt;
    }
    public DataTable StockAlert()     {
        DataTable dt = new DataTable();  // Create a DataTable to hold the data
        string query = "";
        try         {
            query = "SELECT product_id , product_name , product_quantity, product_price FROM STOCK WHERE
product_quantity < 30 ";
            conn.Open();
            SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
            sqlDa.Fill(dt);
        }
        catch (Exception ex)         {
            MessageBox.Show($"{ex}");
        }
        finally      {
            conn.Close();
        }
        return dt;
    }
    public void setAdminLogs(string id, string name, string email, string phone, DateTime loginTime)       {
        string query = "";
        try           {
            string formattedDate = loginTime.ToString("yyyy-MM-dd HH:mm:ss");
            query = $"INSERT INTO AdminLogs VALUES ({Convert.ToInt32(id)}, '{name}', '{email}',
{Convert.ToDecimal(phone)}, '{formattedDate}')";
            conn.Open();
```

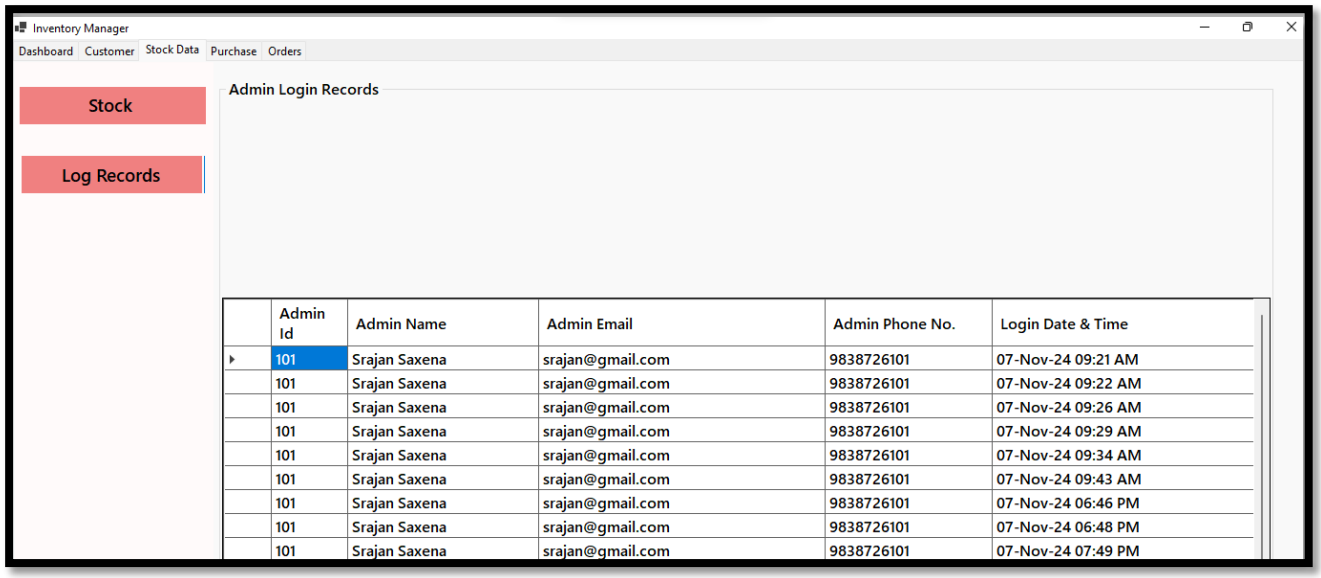
```
            SqlCommand cmd = new SqlCommand(query, conn);
            cmd.ExecuteNonQuery();
        }
        catch (Exception ex)       {
            MessageBox.Show($"{ex}");
        }
        finally       {
            conn.Close();
        }       }
    public DataTable getAdminLogs()      {
        DataTable dt = new DataTable();
        string query = "";
        try    {
            query = "SELECT * FROM AdminLogs";
            conn.Open();
            SqlDataAdapter sqlDa = new SqlDataAdapter(query, conn);
            sqlDa.Fill(dt);
        }
        catch (Exception ex)     {
            MessageBox.Show($"{ex}");
        }
        finally   {
            conn.Close();
        }
        return dt;
    }
  }
}
```

Form 2: Dashboard, Customer Tab, Stock Data Tab, Purchase Tab, Orders Tab

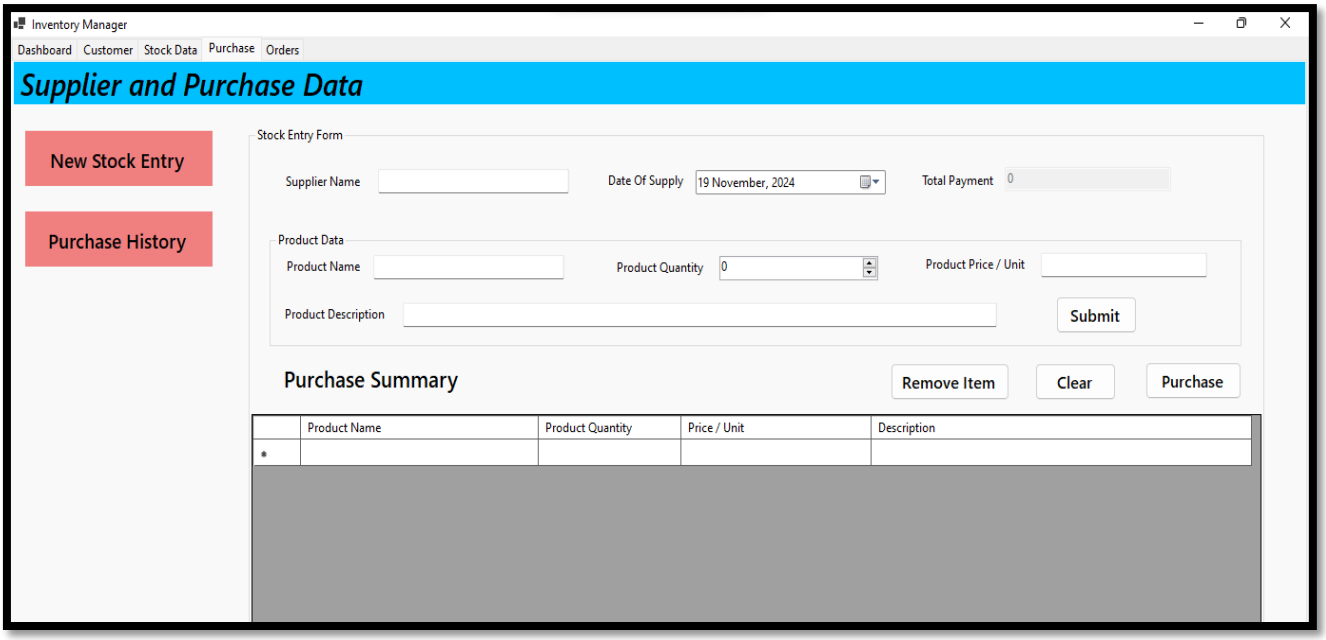


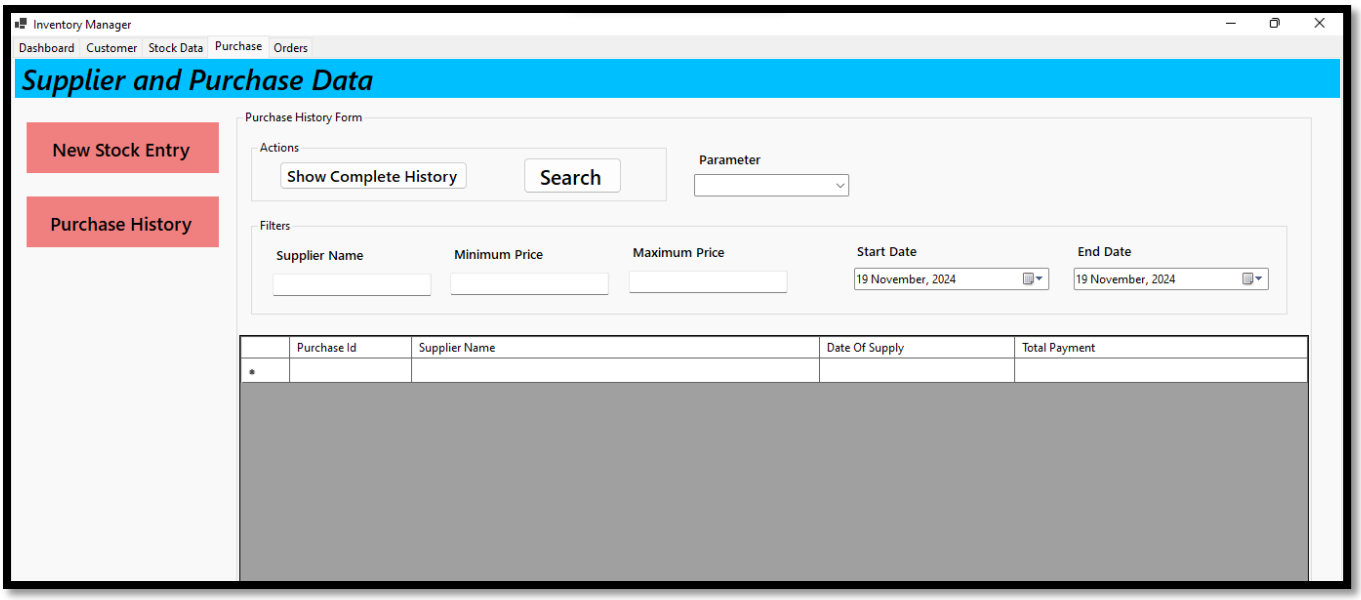


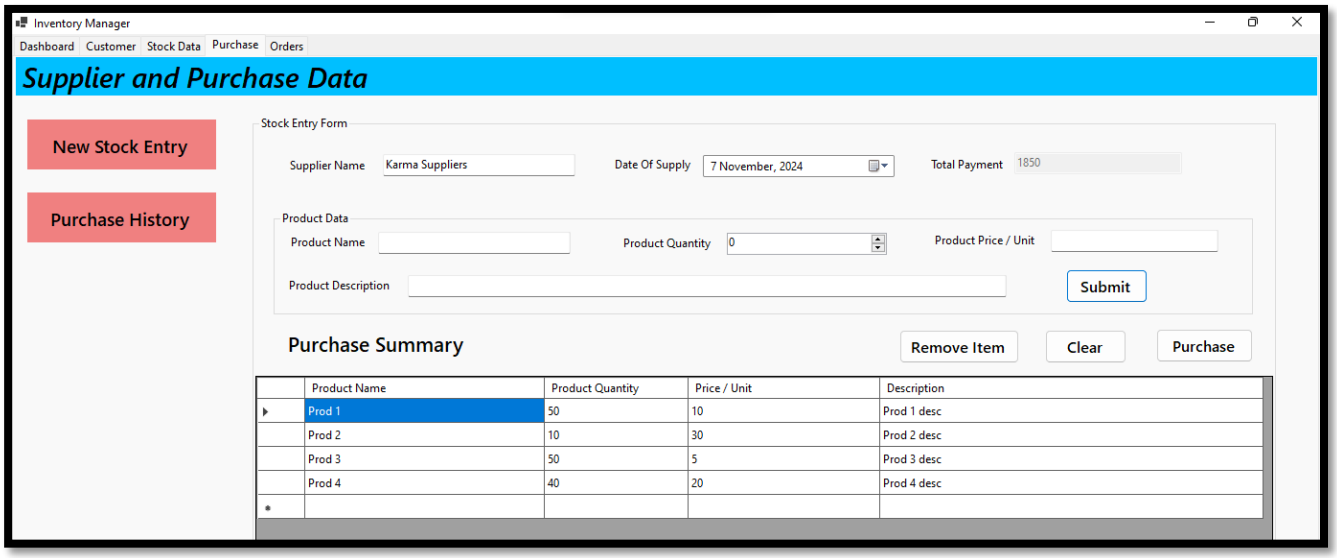


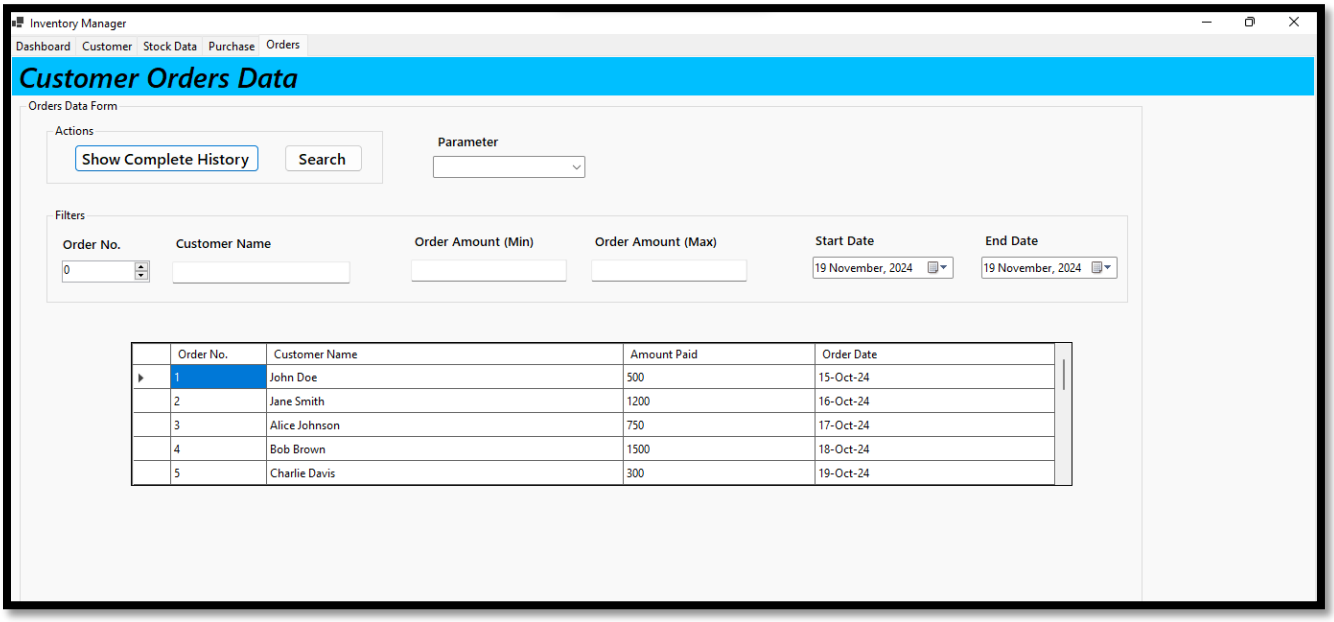


Admin Id	Admin Name	Admin Email	Admin Phone No.	Login Date & Time
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:21 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:22 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:26 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:29 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:34 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 09:43 AM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 06:46 PM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 06:48 PM
101	Srajan Saxena	srajan@gmail.com	9838726101	07-Nov-24 07:49 PM




Inventory Manager
Dashboard
Customer
Stock Data
Purchase
Orders
Supplier and Purchase Data
New Stock Entry
Purchase History
Purchase History Form
Actions
Show Complete History
Search
Parameter
Filters
Supplier Name
Minimum Price
Maximum Price
Start Date
19 November, 2024
End Date
19 November, 2024
Purchase Id
Supplier Name
Date Of Supply
Total Payment


Inventory Manager
Dashboard
Customer
Stock Data
Purchase
Orders
Supplier and Purchase Data
New Stock Entry
Purchase History
Stock Entry Form
Supplier Name
Karma Suppliers
Date Of Supply
7 November, 2024
Total Payment
1850
Product Data
Product Name
Product Quantity
0
Product Price / Unit
Product Description
Submit
Purchase Summary
Remove Item
Clear
Purchase
Product Name
Product Quantity
Price / Unit
Description
Prod 1
50
10
Prod 1 desc
Prod 2
10
30
Prod 2 desc
Prod 3
50
5
Prod 3 desc
Prod 4
40
20
Prod 4 desc


Inventory Manager
Dashboard
Customer
Stock Data
Purchase
Orders
Customer Orders Data
Orders Data Form
Actions
Show Complete History
Search
Parameter
Filters
Order No.
0
Customer Name
Order Amount (Min)
Order Amount (Max)
Start Date
19 November, 2024
End Date
19 November, 2024
Order No.
Customer Name
Amount Paid
Order Date
1
John Doe
500
15-Oct-24
2
Jane Smith
1200
16-Oct-24
3
Alice Johnson
750
17-Oct-24
4
Bob Brown
1500
18-Oct-24
5
Charlie Davis
300
19-Oct-24

Code:

```
using System.Collections.Generic;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Formats.Tar;
using System.Text.RegularExpressions;
namespace WinFormsApp1 {
    public partial class Form2 : Form    {
        SqlConnection conn;
        SqlCommand cmd;
        SqlDataReader dr;
        string query;
        string str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;"; // srj pc
        string customer_Name = "";
        DBOperations ops = new DBOperations();
        List<Image> images = new List<Image>();
        int imageIndex;
        public Form2()     {
            InitializeComponent();
        }
        private void Form2_Load(object sender, EventArgs e)       {
            timer1.Start();
            setDateDay();
            setHighlights();
            setStockAlert();
            addImagesCarousel();
            setVisibilityStatus();
            LoadProductListBox();
        }
        public void setHighlights()     {
            string str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;"; //srj
```

```
        SqlConnection conn = new SqlConnection(str);
        int totalOrders = 0;
        decimal totalSales = 0;
        string today = DateTime.Now.ToString("yyyy-MM-dd");
        MessageBox.Show(today);
        string query = $"SELECT COUNT(*) AS TotalOrders, SUM(order_amount) AS TotalSales FROM Orders
WHERE order_date = '{today}' ";
        try        {
            SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            SqlDataReader reader = cmd.ExecuteReader();
            if (reader.Read())               {
                totalOrders = reader.GetInt32(0); // The first column is TotalOrders
                if (!reader.IsDBNull(1))        {
                    totalSales = reader.GetInt32(1); // The second column is TotalSales
                }
                else            {
                    totalSales = 0; // Set to 0 if it's null
                }    }
            reader.Close();
            label42.Text = totalOrders.ToString();
            label45.Text = totalSales.ToString();
        }
        catch (Exception ex)       {
            MessageBox.Show($"Error retrieving order highlights: {ex.Message}");
        }
        finally  {
            conn.Close();
        }   }
      public void addImagesCarousel()      {
        images.Add(Image.FromFile("F:\\C# SEM 5\\Inventory Management
System\\WinFormsApp1\\WinFormsApp1\\bin\\Debug\\net8.0-windows\\feedback3.jpg"));
```

```
        images.Add(Image.FromFile("F:\\C# SEM 5\\Inventory Management
System\\WinFormsApp1\\WinFormsApp1\\bin\\Debug\\net8.0-windows\\feedback1.jpg"));

        images.Add(Image.FromFile("F:\\C# SEM 5\\Inventory Management
System\\WinFormsApp1\\WinFormsApp1\\bin\\Debug\\net8.0-windows\\feedback4.jpg"));

        imageIndex = 0;

        pictureBox1.Image = images[imageIndex];

    }

    private void timer2_Tick(object sender, EventArgs e)     {

        imageIndex++;

        imageIndex %= 3; // returns len of list<>

        pictureBox1.Image = images[imageIndex];

    }

    private void timer1_Tick(object sender, EventArgs e)    {

        setDateDay();

        timer1.Start();

    }

    private void setDateDay()    {

        timeLbl.Text = DateTime.Now.ToLongTimeString();

        dayLbl.Text = DateTime.Now.ToLongDateString();

    }

    private void setVisibilityStatus()      {

        customerLoginBox.Visible = true;

        customerLoginBox.Enabled = true;

        // disable these :

        orderDetailsBox.Visible = false;

        orderDetailsBox.Enabled = false;

        productSearchBox.Visible = false;

        productSearchBox.Enabled = false;

    }

    private void LoadProductListBox()     {

        str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;";

        conn = new SqlConnection(str);

        query = "SELECT product_name, product_quantity FROM STOCK WHERE product_quantity > 0";
```

```
        cmd = new SqlCommand(query, conn);


        try      {
            conn.Open();
            dr = cmd.ExecuteReader(); // Initialize data reader
            while (dr.Read())              {
                productListBox.Items.Add(dr["product_name"].ToString());
            }         }
        catch (Exception ex)        {
            MessageBox.Show("Error: " + ex.Message);
        }
        finally        {
            dr.Close();    conn.Close();    cmd.Dispose();
        }
    }
    private void button3_Click(object sender, EventArgs e)       {
        str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;";
        conn = new SqlConnection(str);
        string custEmail = cust_email.Text;
        string custPasswd = cust_password.Text;
        query = $"SELECT customer_name , customer_email , customer_password from CUSTOMER where
customer_email = '{custEmail}' AND customer_password = '{custPasswd}' ";
        cmd = new SqlCommand(query, conn);
        conn.Open();
        dr = cmd.ExecuteReader();
        try       {
            if (dr.HasRows && dr.Read())              {
                customer_Name = dr["customer_name"].ToString();
                MessageBox.Show("Logged In Successfully");
                customerLoginBox.Visible = false;
                customerLoginBox.Enabled = false;
                productSearchBox.Visible = true;
                productSearchBox.Enabled = true;
```

```
            orderDetailsBox.Visible = true;
            orderDetailsBox.Enabled = true;


        }
        else          {
            MessageBox.Show("Wrong Credentials");
        }     }
    catch (Exception ex)       {
        MessageBox.Show($"Error Occured : {ex}");
    }
    finally       {
        dr.Close();     conn.Close();    cmd.Dispose();
        button19.Visible = true; // show logout button
    }      }
private void button1_Click(object sender, EventArgs e)   {    this.Close();    }
private void button12_Click(object sender, EventArgs e)        {
    refreshList();
    string searchProd = searchProdBox.Text;
    List<string> matchedItems = new List<string>();
    foreach (var item in productListBox.Items)           {
        if (item.ToString().IndexOf(searchProd, StringComparison.OrdinalIgnoreCase) >= 0)                {
            matchedItems.Add(item.ToString());
        }     }
    if (matchedItems.Count > 0)           {
        productListBox.Items.Clear();
        foreach (string matchedItem in matchedItems)          {
            productListBox.Items.Add(matchedItem);
        }    }
    else     {
        MessageBox.Show("Product not found.");
    }     }
private void button13_Click(object sender, EventArgs e)   {
```

```
        if (productListBox.SelectedItem == null)        {
            MessageBox.Show("Please select a product.");
            return;
        }
        string prodName = productListBox.SelectedItem.ToString();
        str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;";
        conn = new SqlConnection(str);
        query = $"SELECT product_quantity, product_price FROM STOCK WHERE product_name =
'{prodName}'";
        cmd = new SqlCommand(query, conn);
        conn.Open();
        dr = cmd.ExecuteReader();
        try  {
            if (dr.HasRows && dr.Read())   {
                int prodQuantity = Convert.ToInt32(dr["product_quantity"]);
                decimal prodPrice = Convert.ToDecimal(dr["product_price"]);
                bool itemExists = false;
                for (int i = 0; i < receiptBox.Items.Count; i++)    {
                    string item = receiptBox.Items[i].ToString();
                    if (item.IndexOf(prodName, StringComparison.OrdinalIgnoreCase) >= 0)   {
                        itemExists = true;
                        string[] details = item.Split('\t', StringSplitOptions.RemoveEmptyEntries);
                        if (details.Length < 3 || !int.TryParse(details[1].Trim(), out int existingQuantity))     {
                            MessageBox.Show("Invalid item format.");
                            return;
                        }
                        int updatedQuantity = existingQuantity + 1;
                        if (updatedQuantity <= prodQuantity)   {
                            decimal updatedPrice = prodPrice * updatedQuantity;
                            receiptBox.Items[i] = $"{prodName}\t{updatedQuantity}\t{updatedPrice:F2}";
                        }
                        else   {
```

```
                    MessageBox.Show("Insufficient stock available.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Warning);
                }
                break;
            }  }
        if (!itemExists)  {
            receiptBox.Items.Add($"{prodName}\t1\t{prodPrice:F2}");
        }    }    }
    catch (Exception ex)   {
        MessageBox.Show($"Error: {ex.Message}");
    }
    finally     {
        dr.Close();     conn.Close();      cmd.Dispose();
    }     }
  private void button14_Click(object sender, EventArgs e)    {
    if (receiptBox.SelectedItem == null)   {
        MessageBox.Show("Please select an item to remove.");
        return;
    }
    string selectedItem = receiptBox.SelectedItem.ToString();
    string[] details = selectedItem.Split('\t', StringSplitOptions.RemoveEmptyEntries);
    if (details.Length < 3)     {
        MessageBox.Show("Invalid item format.");
        return;
    }
    string prodName = details[0].Trim();
    if (!int.TryParse(details[1].Trim(), out int currentQuantity) || currentQuantity <= 0 ||
        !decimal.TryParse(details[2].Trim(), out decimal totalPrice))      {
        MessageBox.Show("Invalid quantity or price format.");
        return;
    }
    decimal unitPrice = totalPrice / currentQuantity;
    if (currentQuantity > 1)    {
```

```
            int updatedQuantity = currentQuantity - 1;
            decimal updatedPrice = unitPrice * updatedQuantity;
            receiptBox.Items[receiptBox.SelectedIndex] = $"{prodName}\t{updatedQuantity}\t{updatedPrice:F2}";
        }
        else    {
            receiptBox.Items.RemoveAt(receiptBox.SelectedIndex);
        }     }
    private void button15_Click(object sender, EventArgs e)   {
        if (receiptBox.SelectedItem == null)    {
            MessageBox.Show("Please select an item to remove.");
            return;
        }
        string selectedItem = receiptBox.SelectedItem.ToString();
        string[] details = selectedItem.Split('\t', StringSplitOptions.RemoveEmptyEntries);
        if (details.Length < 3)    {
            MessageBox.Show("Invalid item format.");
            return;
        }
        if (int.TryParse(details[1].Trim(), out int quantity) && decimal.TryParse(details[2].Trim(), out decimal
totalItemPrice))   {
            receiptBox.Items.RemoveAt(receiptBox.SelectedIndex);
        }    }
    private void button16_Click(object sender, EventArgs e)    {
        receiptBox.Items.Clear();
    }
    private void button18_Click(object sender, EventArgs e)    {
        refreshList();
    }
    private void refreshList()  {
        productListBox.Items.Clear();
        LoadProductListBox();
    }
    private decimal getTotalAmount()     {
```

```
        decimal totalAmount = 0;


        foreach (var item in receiptBox.Items)     {
            string itemText = item.ToString();
            string[] details = itemText.Split('\t', StringSplitOptions.RemoveEmptyEntries);
            if (decimal.TryParse(details[^1], out decimal price))    {
                totalAmount += price;
            }    }
        return totalAmount;
    }
    private void button17_Click(object sender, EventArgs e)     {
        DialogResult result = MessageBox.Show("Are you sure you want to place the order?", "Confirm Order",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)     {
            decimal totalAmount = getTotalAmount();
            MessageBox.Show("Order placed successfully!"); // show Payment Form
            PaymentForm pfm = new PaymentForm(customer_Name, totalAmount);
            pfm.ShowDialog();
            Dictionary<string, int> ItemsBought = new Dictionary<string, int>();
            foreach (var item in receiptBox.Items)     {
                string[] parts = item.ToString().Split('\t');
                if (parts.Length >= 2)    {
                    string prodName = parts[0];
                    int quantity = int.Parse(parts[1]); // Directly parse the quantity as an integer
                    ItemsBought[prodName] = quantity;
                }     }
            ops.DecreaseStock(ItemsBought);
            setHighlights();
        }
        else    {
            MessageBox.Show("Order canceled.");
        }  }
    private void button19_Click(object sender, EventArgs e)     {
```

```
    orderDetailsBox.Visible = false;
    orderDetailsBox.Enabled = false;
    productSearchBox.Visible = false;
    productSearchBox.Enabled = false;
    cust_email.Text = "";
    cust_password.Text = "";
    customerLoginBox.Visible = true;
    customerLoginBox.Enabled = true;
    button19.Visible = false; // hide this button as well
}
private bool validPhoneNumber(string phn)   {
    if (phn.Length != 10) { return false; }
    foreach (char s in phn) {
        if (!char.IsDigit(s))    {
            return false;
        }   }
    return true;
}
public bool validEmail(string emailText)   {
    string emailPattern = @"^[^@\s]+@[^@\s]+\.[^@\s]+$";
    return Regex.IsMatch(emailText, emailPattern);
}
private void button21_Click(object sender, EventArgs e)    {
    if (string.IsNullOrWhiteSpace(newCustName.Text) ||
        string.IsNullOrWhiteSpace(newCustAddress.Text) ||
        string.IsNullOrWhiteSpace(newCustEmail.Text) ||
        string.IsNullOrWhiteSpace(newCustPassword.Text) ||
        !validPhoneNumber(newCustPhone.Text.ToString()))
    {
        MessageBox.Show("Please enter all details properly");
        return;
    }
```

```
    decimal newPhone;
    try     {
        newPhone = Convert.ToDecimal(newCustPhone.Text);
    }
    catch (FormatException)    {
        MessageBox.Show("Please enter a valid phone number.");
        newCustPhone.Focus(); // Set focus back to the masked text box
        label14.ForeColor = Color.Red;
        return;
    }
    if (!validEmail(newCustEmail.Text)) {
        MessageBox.Show("Please enter a valid Email Id.");
        newCustEmail.Focus(); // Set focus back to the masked text box
        label15.ForeColor = Color.Red;
        return;
    }
    string newName = newCustName.Text;
    string newEmail = newCustEmail.Text;
    string newPasswd = newCustPassword.Text;
    string newAddress = newCustAddress.Text;
    ops.NewCustomer(new Customer(newName, newEmail, newPasswd, newPhone, newAddress));
    custRegisterBox.Visible = false;
    customerLoginBox.Visible = true;
}
private void button20_Click(object sender, EventArgs e)      {
    custRegisterBox.Visible = true;
    customerLoginBox.Visible = false;
}
private void panel5_Paint(object sender, PaintEventArgs e)       {
    stockEntryForm.Visible = true;
    stockEntryForm.Enabled = true;
    purchaseHistoryForm.Visible = false;
```

```
        purchaseHistoryForm.Enabled = false;
    }
    private void label20_Click(object sender, EventArgs e)        {
        stockEntryForm.Visible = true;
        stockEntryForm.Enabled = true;
        purchaseHistoryForm.Visible = false;
        purchaseHistoryForm.Enabled = false;
    }
    private void panel7_Paint(object sender, PaintEventArgs e)        {
        stockEntryForm.Visible = false;
        stockEntryForm.Enabled = false;
        purchaseHistoryForm.Visible = true;
        purchaseHistoryForm.Enabled = true;
    }
    private void label22_Click(object sender, EventArgs e)        {
        stockEntryForm.Visible = false;
        stockEntryForm.Enabled = false;
        purchaseHistoryForm.Visible = true;
        purchaseHistoryForm.Enabled = true;
    }
    private void button23_Click(object sender, EventArgs e)       {
        string supplierName = textBox3.Text;
        DateTime supplyDate = supDate.Value;
        decimal totalAmt = Convert.ToDecimal(textBox5.Text);
        int latestPurchaseId = ops.NewPurchase(new Purchase(supplierName, supplyDate, totalAmt));
        foreach (DataGridViewRow row in stockDataGrid.Rows)            {
            if (!row.IsNewRow)   {
                string productName = row.Cells["ProductName"].Value?.ToString();
                int productQuantity = Convert.ToInt32(row.Cells["ProductQuantity"].Value);
                decimal productPrice = Convert.ToDecimal(row.Cells["ProductPrice"].Value);
                string productDescription = row.Cells["ProductDescription"].Value?.ToString();
                ops.NewPurchaseDetails(new PurchaseDetails(latestPurchaseId, productName, productQuantity,
productPrice, productDescription));
```

```
            }   }

        MessageBox.Show("Purchase and product details added successfully.");
    }
    private void stockSubmitBtn_Click(object sender, EventArgs e)     {
        if (string.IsNullOrWhiteSpace(textBox8.Text) ||
            numericUpDown1.Value <= 0 ||
            string.IsNullOrWhiteSpace(textBox6.Text) ||
            string.IsNullOrWhiteSpace(textBox9.Text))
        {
            MessageBox.Show("Please enter all product details.");
            return;
        }
        string productName = textBox8.Text;
        int productQuantity = (int)numericUpDown1.Value;
        decimal productPrice = Convert.ToDecimal(textBox6.Text);
        string productDescription = textBox9.Text;
        stockDataGrid.Rows.Add(productName, productQuantity, productPrice, productDescription);
        textBox5.Text = (Convert.ToInt32(textBox5.Text) + (productQuantity * productPrice)).ToString();
        textBox8.Clear();
        numericUpDown1.Value = 0;
        textBox6.Clear();
        textBox9.Clear();
    }
    private void button28_Click(object sender, EventArgs e)     {
        if (string.IsNullOrWhiteSpace(stockParameterBox.SelectedItem?.ToString()) ||
string.IsNullOrWhiteSpace(stockParameter.Text))
        {
            MessageBox.Show("Please select a parameter and enter the values");
            return;
        }
        string paraType = stockParameterBox.Text;
        string paraEntry = stockParameter.Text;
```

```
    DataTable stockData = ops.ShowStockData(paraType, paraEntry);


    stockSearchGrid.DefaultCellStyle.ForeColor = Color.Black;
    stockSearchGrid.DefaultCellStyle.BackColor = Color.White;
    if (stockData.Rows.Count > 0)          {
        stockSearchGrid.DataSource = stockData;
        MessageBox.Show("Data loaded successfully.");
    }
    else    {
        MessageBox.Show("No data found.");
    }    }
private void stockParameterBox_SelectedIndexChanged(object sender, EventArgs e)       {
    if (stockParameterBox.SelectedItem.ToString() == "Complete Stock")    {
        stockParameter.Text = "All";
        stockParameter.Enabled = false;
    }
    else   {
        stockParameter.Text = "";
        stockParameter.Enabled = true;
    }    }
private void button5_Click(object sender, EventArgs e)    {
    stockDataForm.Visible = true;
    stockDataForm.Enabled = true;
    adminLogBox.Visible = false;
    adminLogBox.Enabled = false;
}
private void button7_Click(object sender, EventArgs e)        {
    stockDataForm.Visible = false;
    stockDataForm.Enabled = true;
    adminLogBox.Visible = true;
    adminLogBox.Enabled = true;
    DataTable logsData = ops.getAdminLogs();
```

```
    adminLogsGrid.DataSource = logsData;
}
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)        {
    if (comboBox1.SelectedItem.ToString() == "Supplier")            {
        foreach (Control ctrl in purchaseFilters.Controls)              {
            if (ctrl is TextBox || ctrl is DateTimePicker)                  {
                ctrl.Enabled = false;
            }    }
        textBox4.Enabled = true;
    }
    else if (comboBox1.SelectedItem.ToString() == "Date")       {
        foreach (Control ctrl in purchaseFilters.Controls)        {
            if (ctrl is TextBox || ctrl is DateTimePicker)        {
                ctrl.Enabled = false;
            }        }
        dateTimePicker1.Enabled = true;
        label34.Text = "Pick Date : ";
    }
    else if (comboBox1.SelectedItem.ToString() == "Minimum Price")      {
        foreach (Control ctrl in purchaseFilters.Controls)        {
            if (ctrl is TextBox || ctrl is DateTimePicker)        {
                ctrl.Enabled = false;
            }       }
        textBox7.Enabled = true;
    }
    else if (comboBox1.SelectedItem.ToString() == "Maximum Price")       {
        foreach (Control ctrl in purchaseFilters.Controls)        {
            if (ctrl is TextBox || ctrl is DateTimePicker)        {
                ctrl.Enabled = false;
            } }
        textBox10.Enabled = true;
    }
```

```
    else if (comboBox1.SelectedItem.ToString() == "Date Range")  {
        foreach (Control ctrl in purchaseFilters.Controls)     {
            if (ctrl is TextBox || ctrl is DateTimePicker)        {
                ctrl.Enabled = false;
            }           }
        dateTimePicker1.Enabled = true;
        dateTimePicker2.Enabled = true;
        label34.Text = "Start Date";
        label35.Text = "End Date";
    }
    else if (comboBox1.SelectedItem.ToString() == "Price Range")    {
        foreach (Control ctrl in purchaseFilters.Controls)   {
            if (ctrl is TextBox || ctrl is DateTimePicker)   {
                ctrl.Enabled = false;
            }  }
        textBox7.Enabled = true;
        textBox10.Enabled = true;
    }  }
private void button27_Click(object sender, EventArgs e)  {
    if (string.IsNullOrWhiteSpace(comboBox1.SelectedItem?.ToString()))  {
        MessageBox.Show("Please select a parameter and enter the values");
        return;
    }
    string paraType = comboBox1.Text;
    string paraEntry = "";
    if (paraType == "Supplier")   {
        if (string.IsNullOrWhiteSpace(textBox4.Text))   {
            MessageBox.Show("Please enter the Supplier Name");
            return;
        }
        paraEntry = textBox4.Text;
    }
```

```
        else if (paraType == "Date")    {
            DateTime startDate = dateTimePicker1.Value;
            paraEntry = startDate.ToString("yyyy-MM-dd");
        }
        else if (paraType == "Minimum Price")    {
            if (string.IsNullOrWhiteSpace(textBox7.Text))  {
                MessageBox.Show("Please enter the Min Price");
                return;
            }
            paraEntry = textBox7.Text;
        }
        else if (paraType == "Maximum Price")     {
            if (string.IsNullOrWhiteSpace(textBox10.Text))     {
                MessageBox.Show("Please enter the Max Price");
                return;
            }
            paraEntry = textBox10.Text;  // later convert to decimal
        }
        else if (paraType == "Date Range")   {
            DateTime startDate = dateTimePicker1.Value;
            DateTime endDate = dateTimePicker2.Value;
            DataTable ans = ops.ShowPurchaseData(paraType, startDate.ToString("yyyy-MM-dd"),
endDate.ToString("yyyy-MM-dd"));
            if (ans.Rows.Count > 0)   {
                dataGridView2.DataSource = ans;
                return;
            }
            else  {
                MessageBox.Show("No data found.");
                return;
            }   }
        else if (paraType == "Price Range")  {
            string minPrice = textBox7.Text;
```

```
        string maxPrice = textBox10.Text;   // later convert to decimal
        DataTable ans = ops.ShowPurchaseData(paraType, minPrice, maxPrice);
        if (ans.Rows.Count > 0)  {
            dataGridView2.DataSource = ans;
            return;
        }
        else  {
            MessageBox.Show("No data found.");
            return;
        }  }
    DataTable purchaseData = ops.ShowPurchaseData(paraType, paraEntry);
    if (purchaseData.Rows.Count > 0)  {
        dataGridView2.DataSource = purchaseData;
        MessageBox.Show("Data Loaded Successfully");
    }
    else   {
        MessageBox.Show("No data found.");
    }  }
private void button25_Click(object sender, EventArgs e)    {
    DataTable purchaseData = ops.ShowPurchaseData("All", "*");
    if (purchaseData.Rows.Count > 0)      {
        dataGridView2.DataSource = purchaseData;
        MessageBox.Show("Data Loaded Successfully");
    }
    else     {
        MessageBox.Show("No data found.");
    }     }
private void button22_Click(object sender, EventArgs e)        {
    foreach (Control ctrl in stockEntryForm.Controls)            {
        if (ctrl is TextBox textBox)       {
            textBox.Clear();
            textBox5.Text = "0"; // to avoid exception after clear
```

```
        }

        else if (ctrl is DateTimePicker dateTimePicker)     {
            dateTimePicker.Value = DateTime.Now;
        }
        else if (ctrl is NumericUpDown numericUpDown)    {
            numericUpDown.Value = numericUpDown.Minimum;
        }
        else if (ctrl is DataGridView dataGridView)     {
            dataGridView.Rows.Clear();
        } } }
    private void tabPage1_Click(object sender, EventArgs e)   {
        setStockAlert();
        setHighlights();
}
    private void setStockAlert()    {
        DataTable stockAlertData = ops.StockAlert();
        stockAlertGrid.DataSource = stockAlertData;
    }
    private void button6_Click_1(object sender, EventArgs e)      {
        custRegisterBox.Visible = false;
        customerLoginBox.Visible = true;
    }
    private void button10_Click(object sender, EventArgs e)       {
        DataTable orderHistory = ops.ShowOrderData("All", "*");
        if (orderHistory.Rows.Count > 0)     {
            ordersDataGrid.DataSource = orderHistory;
            MessageBox.Show("Data Loaded Successfully");
        }
        else   {
            MessageBox.Show("No data found.");
        } }
```

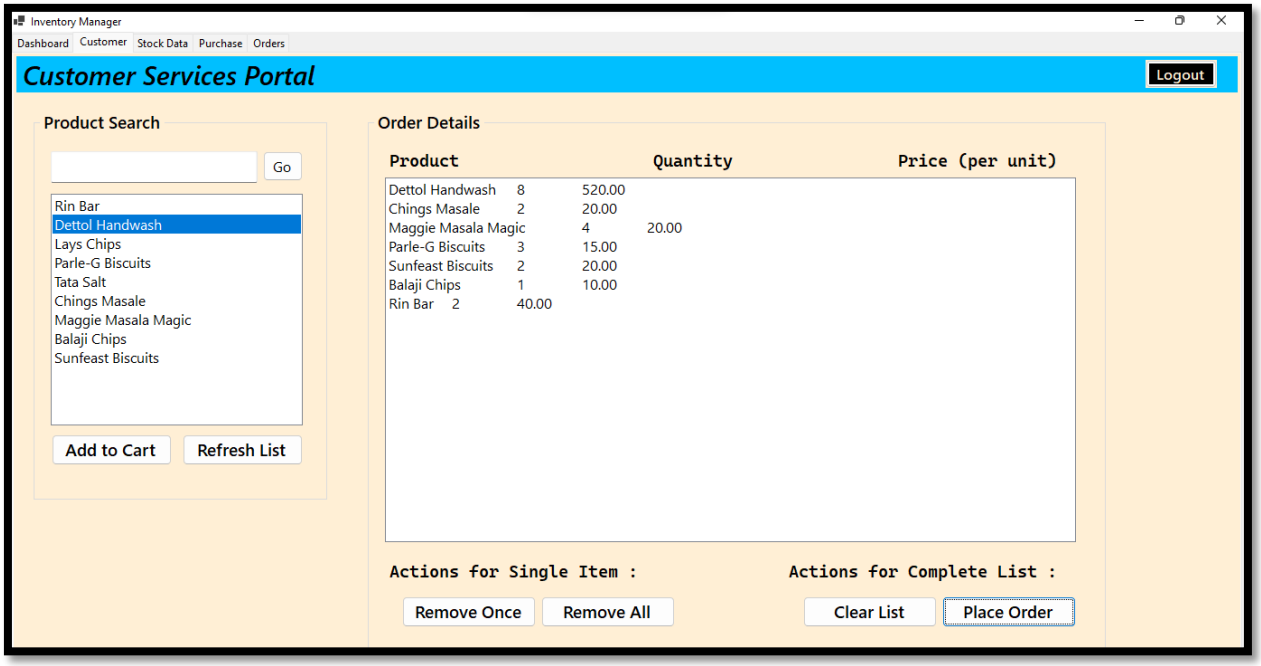
```
private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)    {
    if (comboBox2.SelectedItem.ToString() == "Customer Name")     {
        foreach (Control ctrl in orderFilters.Controls)      {
            if (ctrl is TextBox || ctrl is DateTimePicker || ctrl is NumericUpDown)     {
                ctrl.Enabled = false;
            }   }
        textBox12.Enabled = true;
    }
    else if (comboBox2.SelectedItem.ToString() == "Date")    {
        foreach (Control ctrl in orderFilters.Controls)     {
            if (ctrl is TextBox || ctrl is DateTimePicker || ctrl is NumericUpDown)    {
                ctrl.Enabled = false;
            }     }
        dateTimePicker4.Enabled = true;
        label50.Text = "Pick Date : ";
    }
    else if (comboBox2.SelectedItem.ToString() == "Order Amount")     {
        foreach (Control ctrl in orderFilters.Controls)     {
            if (ctrl is TextBox || ctrl is DateTimePicker || ctrl is NumericUpDown)    {
                ctrl.Enabled = false;
            }      }
        textBox11.Enabled = true;
        label51.Text = "Order Amount : ";
    }
    else if (comboBox2.SelectedItem.ToString() == "Order Number")       {
        foreach (Control ctrl in orderFilters.Controls)     {
            if (ctrl is TextBox || ctrl is DateTimePicker)   {
                ctrl.Enabled = false;
            }   }
        numericUpDown2.Enabled = true;
    }
    else if (comboBox2.SelectedItem.ToString() == "Price Range")    {
```

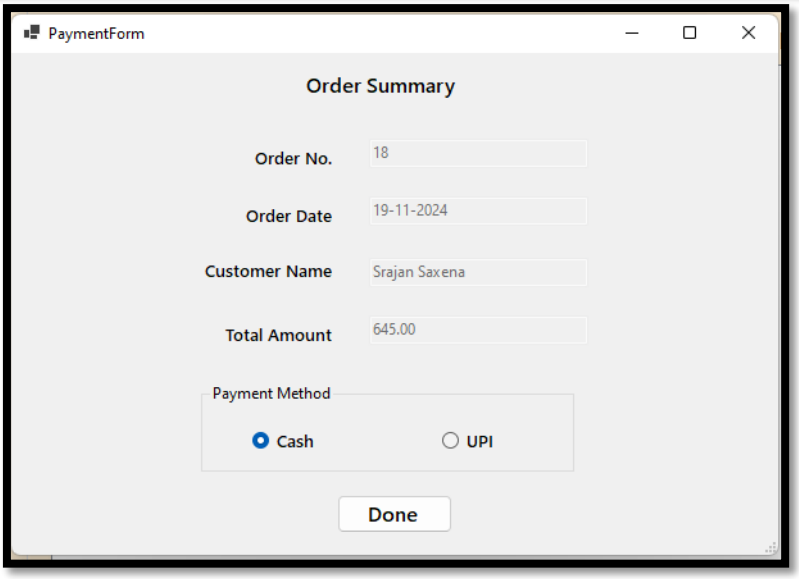
```
            foreach (Control ctrl in orderFilters.Controls)   {
                if (ctrl is TextBox || ctrl is DateTimePicker || ctrl is NumericUpDown)   {
                    ctrl.Enabled = false;
                }     }
            label51.Text = "Order Amount (Min)";
            textBox11.Enabled = true;
            textBox13.Enabled = true;       }
        else if (comboBox2.SelectedItem.ToString() == "Date Range")         {
            foreach (Control ctrl in orderFilters.Controls)              {
                if (ctrl is TextBox || ctrl is NumericUpDown)                 {
                    ctrl.Enabled = false;                 }
                else if (ctrl is DateTimePicker)                 {
                    ctrl.Enabled = true;      }   }     }     }
    private void button9_Click(object sender, EventArgs e)      {
        if (string.IsNullOrWhiteSpace(comboBox2.SelectedItem?.ToString()))     {
            MessageBox.Show("Please select a parameter and enter the values");
            return;
        }
        string paraType = comboBox2.Text;
        string paraEntry = "";
        if (paraType == "Date")     {
            DateTime startDate = dateTimePicker4.Value;
            paraEntry = startDate.ToString("yyyy-MM-dd");
        }
        else if (paraType == "Date Range")     {
            DateTime startDate = dateTimePicker4.Value;
            DateTime endDate = dateTimePicker3.Value;
            DataTable ans = ops.ShowOrderData(paraType, startDate.ToString("yyyy-MM-dd"),
endDate.ToString("yyyy-MM-dd"));
            if (ans.Rows.Count > 0)  {
                ordersDataGrid.DataSource = ans;
                return;
            }
```

```
    else   {
        MessageBox.Show("No data found.");
        return;
    }   }
else if(paraType == "Price Range")   {
    if (string.IsNullOrWhiteSpace(textBox11.Text) || string.IsNullOrWhiteSpace(textBox13.Text)) {
        MessageBox.Show("Please enter the Price Range");
        return;
    }
    string minPrice = textBox11.Text;
    string maxPrice = textBox13.Text;   // later convert to int
    DataTable ans = ops.ShowOrderData(paraType, minPrice, maxPrice);
    if (ans.Rows.Count > 0)  {
        ordersDataGrid.DataSource = ans;
        return;
    }
    else   {
        MessageBox.Show("No data found.");
        return;
    }  }
else if(paraType == "Order Number")      {
    paraEntry = numericUpDown2.Text.ToString();
}
else if(paraType == "Customer Name")    {
    if (string.IsNullOrWhiteSpace(textBox12.Text))  {
        MessageBox.Show("Please enter the Customer Name");
        return;
    }
    paraEntry = textBox12.Text;
}
else if(paraType == "Order Amount")   {
    if (string.IsNullOrWhiteSpace(textBox11.Text))      {
```

```
            MessageBox.Show("Please enter the Order Amount");
            return;
        }
        paraEntry = textBox11.Text; // later convert to int
    }
    DataTable ordersData = ops.ShowOrderData(paraType, paraEntry);
    if (ordersData.Rows.Count > 0)    {
        ordersDataGrid.DataSource = ordersData;
        MessageBox.Show("Data Loaded Successfully");
    }
    else      {
        MessageBox.Show("No data found");
    }   }
  } // Form 2 class ends
}
```

Form 3: Payment Form and Order Booking in Customer Tab of Form 2





Code:

```
using System.Collections.Generic;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace WinFormsApp1  {
   public partial class PaymentForm : Form    {
      DBOperations ops = new DBOperations();
      public string customerName;
      public decimal totalAmount;
      int nextOrderNumber = 1;  // Default to 1 in case there are no previous orders
      DateTime currentDate = DateTime.Now;
      SqlConnection conn;
      SqlCommand cmd;
      SqlDataReader dr;
      string query, str;
      public PaymentForm(string customerName, decimal totalAmount)       {
         InitializeComponent();
         this.customerName = customerName;
         this.totalAmount = totalAmount;
         custNameBox.Text = customerName.ToString();
         totalAmtBox.Text = totalAmount.ToString();     }
      private void PaymentForm_Load(object sender, EventArgs e)       {
         getDetails();   // orderNo. + orderDate     }
      private void getDetails()    {
         orderDateBox.Text = currentDate.ToString("dd-MM-yyyy");
         str = "Server=localhost;Database=SAMPLE;Trusted_Connection=True;";   // srj pc
         conn = new SqlConnection(str);
         try    {
            conn.Open();
            query = "SELECT ISNULL(MAX(order_number), 0) FROM orders"; // ?
            SqlCommand cmd = new SqlCommand(query, conn);
            int maxOrderNumber = Convert.ToInt32(cmd.ExecuteScalar());  // returns 1 value
```

```
        nextOrderNumber = maxOrderNumber + 1;
        orderNumBox.Text = nextOrderNumber.ToString();
    }
    catch (Exception ex)     {
        MessageBox.Show("Error: " + ex.Message);
    }
    finally      {
        conn.Close();
    }    } // get details function over
  private void radioButton2_CheckedChanged(object sender, EventArgs e)      {
    if (radioButton2.Checked)       {
        crossBtn.Visible = true;       qrBox.Visible = true;
    }   }
  private void button1_Click(object sender, EventArgs e)     {
    if (string.IsNullOrWhiteSpace(orderNumBox.Text) ||
        string.IsNullOrWhiteSpace(orderDateBox.Text) ||
        string.IsNullOrWhiteSpace(custNameBox.Text) ||
        string.IsNullOrWhiteSpace(totalAmtBox.Text) ||
        (!radioButton1.Checked && !radioButton2.Checked))            {
        MessageBox.Show("Order Summary Incomplete! Please select a Payment Method.");
        return;
    }
    ops.NewOrder(new Orders(nextOrderNumber, customerName, totalAmount, currentDate));
    this.Close(); // close the payment form
  }
  private void button2_Click(object sender, EventArgs e)     {
    qrBox.Visible = false; // close
    crossBtn.Visible=false;
    radioButton2.Checked=false;
  } }}
```