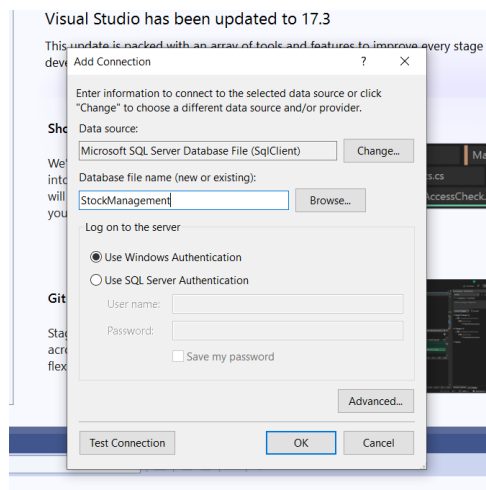


## DOT NET TECHNOLOGIES

### Stock Management Application

#### DataBase Part :

1. Open visual studio code and navigate to the “ View Section “ and then select the “ Server Explorer Option “
2. Then right click on the “ Data Connections “ and then click on “ Add Connection Option “
3. Now, select the “ Microsoft SQL Server Database File (sql Client) “ option
4. Then, create a Database with the name Stock management.



5. Now, create a 3 tables with the names Login, Stock, Products and their queries are below there :

#### Login Table Query

```
CREATE TABLE Login
(
    [UserName] VARCHAR(50) NOT NULL PRIMARY KEY,
    [Password] VARCHAR(50) NULL
)
```

#### Products Table Query

```
CREATE TABLE Products
(
    [ProductCode] INT NOT NULL PRIMARY KEY,
    [ProductName] VARCHAR(150) NULL,
    [ProductStatus] BIT NULL
)
```

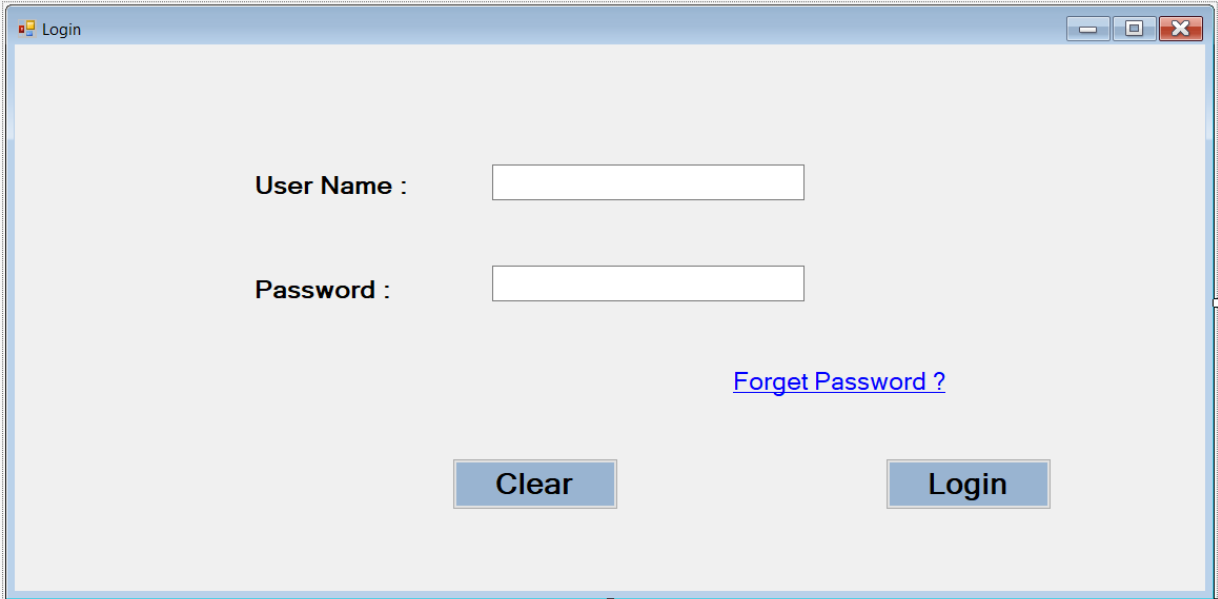
)

### Stocks Table Query

```
CREATE TABLE Stock
(  
    [ProductCode] INT NOT NULL PRIMARY KEY,  
    [ProductName] VARCHAR(150) NULL,  
    [TransDate] DATETIME NULL,  
    [Quantity] FLOAT NULL  
)
```

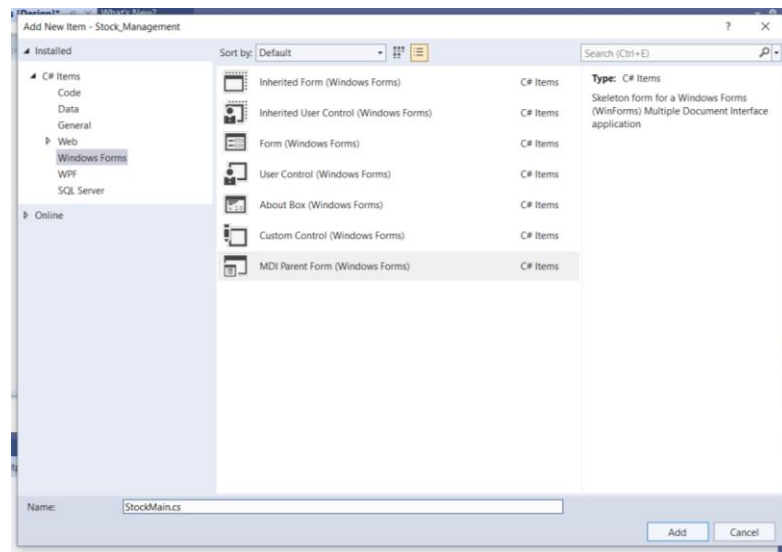
### Login Form Part :

1. Open the New Project then select the windows form application and save it to the local machine.
2. navigate to the “ View Section “ and then select the “ Solution Explorer Option “
3. now, right click on the project name then click on “ ADD Component “ and then select the windows form application and save it as “Login.cs”
4. Now, design the Login Form Just like below one .

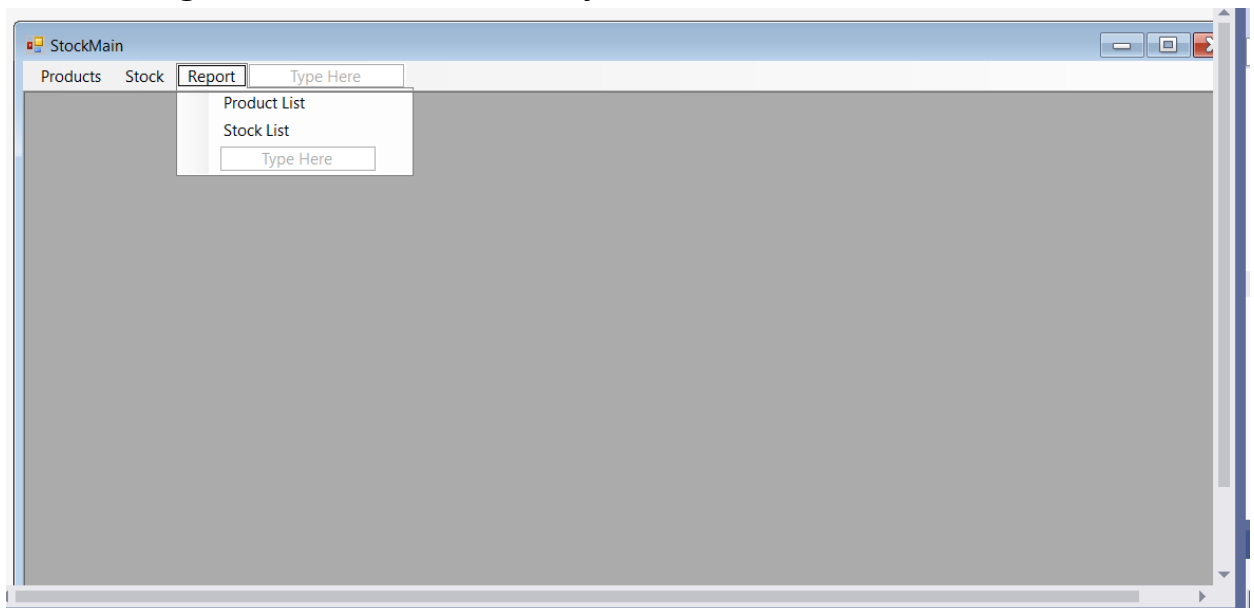


The screenshot shows a Windows application window titled "Login". Inside the window, there is a light gray background. On the left side, there are two labels: "User Name :" and "Password :". To the right of each label is a white text input box. Below the "Password :" input box, there is a blue hyperlink that says "Forget Password ?". At the bottom of the form, there are two blue buttons: "Clear" on the left and "Login" on the right. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

5. now, right click on the project name then click on “ ADD Component “ and then select the “ MDI Parent Form (Windows Forms) and save it as “StockMain.cs”



6. after adding this component you will see this is like and notepad .
7. now, remove all the items in the menu strip and the bar below the menu strip also.
8. Now, design this “StockMain.cs “ just like below one.



9. Now, we will write the C# Code for the “ StockMain.cs” and “ Login.cs”.
10. Then we will validate the Login Form details or credentials.
11. The codes for both “StockMain and login “ are there below one:

## Stock Main.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Stock_Management
{
    public partial class StockMain : Form
    {
        public StockMain()
        {
            InitializeComponent();
        }

        private void productsToolStripMenuItem_Click(object sender, EventArgs e)
        {
            // products option in Navbar
            Products pro = new Products();
            pro.MdiParent = this;
            pro.StartPosition = FormStartPosition.CenterScreen;
            pro.Show();
        }

        bool close = true;
        private void StockMain_FormClosing(object sender, FormClosingEventArgs e)
        {
            if(close)
            {
                DialogResult result = MessageBox.Show(" Are You Sure Want To Exit ",
"Exit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

                if (result == DialogResult.Yes)
                {
                    close = false;
                    Application.Exit();
                }
                else
                {
                    e.Cancel = true;
                }
            }
        }
    }
}
```

## Login Form Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

```

using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient; // add this line

namespace Stock_Management
{
    public partial class Login : Form
    {
        public Login()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // Clear Button
            textBox1.Text = "";
            textBox2.Clear();
            textBox1.Focus();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            // validating the Login Credentials
            SqlConnection con = new SqlConnection("Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=C:\\Users\\SHREEE\\OneDrive\\Documents\\St
ockManagement.mdf;Integrated Security=True;Connect Timeout=30");
            con.Open();

            SqlCommand cmd = new SqlCommand("select * from Login where UserName =
@UserName and Password = @Password", con);
            cmd.Parameters.AddWithValue("@UserName", textBox1.Text);
            cmd.Parameters.AddWithValue("@Password", textBox2.Text);
            cmd.ExecuteNonQuery();

            SqlDataAdapter sda = new SqlDataAdapter(cmd);
            DataTable dt = new DataTable();
            sda.Fill(dt);

            if(dt.Rows.Count == 1)
            {
                this.Hide();
                StockMain main = new StockMain();
                main.Show();
            }
            else
            {
                MessageBox.Show("Invalid Username and password ... !", "Error",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
                button1_Click(sender, e);
            }
            con.Close();
            // Login Button
        }
    }
}

```

### Note :

Use the Connection String of Your Own DataBase.

## Connection String Part :

1. When we are writing the coding part in back of the form we mostly use the database connection string many times and this string is very large.
2. So, to reduce the database connection string we can do some minor changes in the “ App.config ” and the code is below there

### App.Config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>

  <connectionStrings>
    <add name="StockConn" connectionString="Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\SHREEE\OneDrive\Documents\StockMan
agement.mdf;Integrated Security=True;Connect Timeout=30"
providerName="System.Data.SqlClient" />
  </connectionStrings>
</configuration>
```

3. After modifying the code in app.config we have to add a component.
4. now, right click on the project name then click on “ ADD Component ” and then select the “ Class ” and save it as “Connection.cs”
5. In this connection.cs file we will write the only single static function and it is invoked when we want to use the database connection string.

### Connection.cs

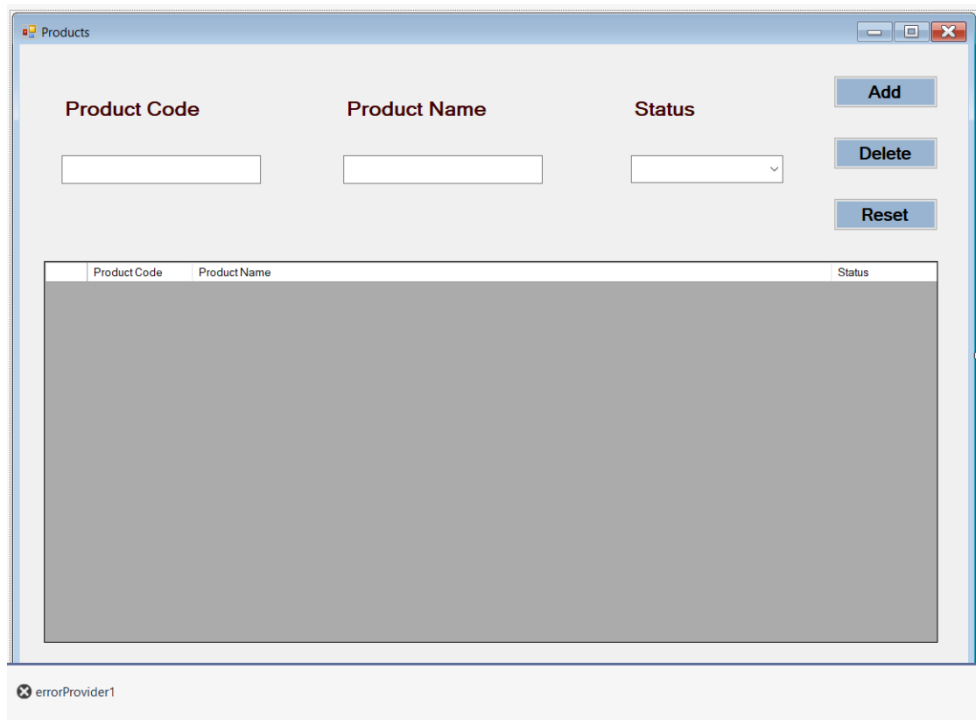
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace Stock_Management
{
    public static class Connection
    {
        public static SqlConnection getConnection()
        {
            SqlConnection con = new SqlConnection();
            con.ConnectionString =
System.Configuration.ConfigurationManager.ConnectionStrings["StockConn"].ConnectionString;

            return con;
        }
    }
}
```

## Products Form :

1. now, right click on the project name then click on “ ADD Component “ and then select the “ Windows Forms “ and save it as “products.cs”
2. navigate to the Form Tool Box and search for “ Error provider “ and double click on that then navigate to the Product.cs



3. When you add combo box to the status then in drop down add these 2 parameters also they are “ Active & Deactive “.
4. The code for products form will below there :

### Products Form Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Stock_Management
```

```

{
    public partial class Products : Form
    {
        public Products()
        {
            InitializeComponent();
        }

        private void Products_Load(object sender, EventArgs e)
        {
            comboBox1.SelectedIndex = 0;
            LoadData();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // Add button
            if(Validation())
            {
                SqlConnection con = Connection.getConnection();
                con.Open();

                bool status = false;
                if (comboBox1.SelectedIndex == 0)
                {
                    status = true;
                }
                else
                {
                    status = false;
                }

                var sqlquery = "";
                if (IfProductExists(con, textBox1.Text))
                {
                    sqlquery = "update Products set ProductName = @ProductName,
ProductStatus = @ProductStatus where ProductCode = @ProductCode";
                }
                else
                {
                    sqlquery = "insert into Products values(@ProductCode,
@ProductName, @ProductStatus)";
                }

                SqlCommand cmd = new SqlCommand(sqlquery, con);
                cmd.Parameters.AddWithValue("@ProductCode",
int.Parse(textBox1.Text));
                cmd.Parameters.AddWithValue("@ProductName", textBox2.Text);
                cmd.Parameters.AddWithValue("@ProductStatus", status);
                cmd.ExecuteNonQuery();

                con.Close();

                LoadData();

                ResetRecords(); //...
            }
        }

        private bool IfProductExists(SqlConnection con, string productCode)
        {
            SqlDataAdapter sda = new SqlDataAdapter("select 1 from Products where
ProductCode = '" + productCode + "'", con);
            DataTable dt = new DataTable();

```



```

        sda.Fill(dt);

        if(dt.Rows.Count > 0)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    public void LoadData()
    {
        SqlConnection con = Connection.getConnection();
        con.Open();

        SqlDataAdapter sda = new SqlDataAdapter("select * from Products",
con);

        DataTable dt = new DataTable();
        sda.Fill(dt);

        dataGridView1.Rows.Clear();
        foreach (DataRow item in dt.Rows)
        {
            int n = dataGridView1.Rows.Add();

            if ((bool)item["ProductStatus"])
            {
                dataGridView1.Rows[n].Cells[2].Value = "Active";
            }
            else
            {
                dataGridView1.Rows[n].Cells[2].Value = "Deactive";
            }

            dataGridView1.Rows[n].Cells[0].Value =
item["ProductCode"].ToString();
            dataGridView1.Rows[n].Cells[1].Value =
item["ProductName"].ToString();
            //dataGridView1.Rows[n].Cells[0].Value =
item["ProductStatus"].ToString();
        }
    }

    private void dataGridView1_MouseDoubleClick(object sender, MouseEventArgs
e)
    {
        button1.Text = "Update"; //..

        textBox1.Text =
dataGridView1.SelectedRows[0].Cells[0].Value.ToString();
        textBox2.Text =
dataGridView1.SelectedRows[0].Cells[1].Value.ToString();
        //comboBox1.SelectedText =
dataGridView1.SelectedRows[0].Cells[2].Value.ToString();

        if(dataGridView1.SelectedRows[0].Cells[2].Value.ToString() ==
"Active")
        {
            comboBox1.SelectedIndex = 0;
        }
        else
        {
            comboBox1.SelectedIndex = 1;
        }
    }
}

```

```

private void button2_Click(object sender, EventArgs e)
{
    // Delete button

    DialogResult dialogResult = MessageBox.Show(" Are You Sure Want To
Delete ", "Message", MessageBoxButtons.YesNo);
    if(dialogResult == DialogResult.Yes)
    {
        if (Validation())
        {
            SqlConnection con = Connection.getConnection();
            con.Open();

            var sqlquery = "";
            if (IfProductExists(con, textBox1.Text))
            {
                sqlquery = "delete from Products where ProductCode =
@ProductCode";

                SqlCommand cmd = new SqlCommand(sqlquery, con);
                cmd.Parameters.AddWithValue("@ProductCode",
int.Parse(textBox1.Text));
                cmd.ExecuteNonQuery();
            }
            else
            {
                MessageBox.Show("Record Doesn't Exists in the table .....
!");
            }

            con.Close();

            LoadData();
            ResetRecords();
        }
    }

}

private void ResetRecords()
{
    //..
    textBox1.Clear();
    textBox2.Clear();
    comboBox1.SelectedIndex = -1;

    button1.Text = "Add";
    textBox1.Focus();
}

private void button3_Click(object sender, EventArgs e)
{
    // ..
    ResetRecords();
}

private bool Validation()
{
    // ..
    bool result = false;

    if(string.IsNullOrEmpty(textBox1.Text))
    {
        errorProvider1.Clear();
    }
}

```

```

        errorProvider1.SetError(textBox1, " Product Code required ");
    }
    else if(string.IsNullOrEmpty(textBox2.Text))
    {
        errorProvider1.Clear();
        errorProvider1.SetError(textBox2, " Product Name Required ");
    }
    else if(comboBox1.SelectedIndex == -1)
    {
        errorProvider1.Clear();
        errorProvider1.SetError(comboBox1, " Select the status ");
    }
    else
    {
        return true;
    }
    return result;
}
}
}

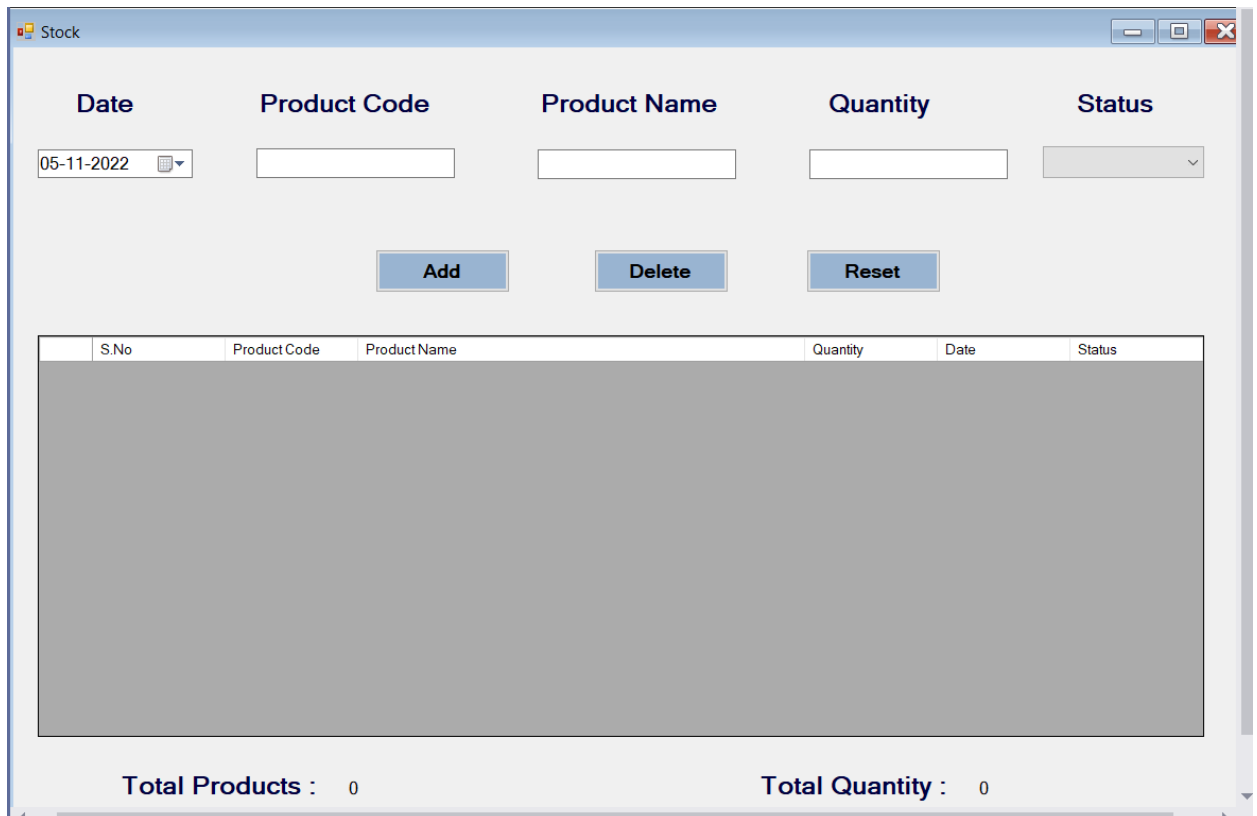
```

**Note :**

Use the Connection String of Your Own DataBase.

## Stock Form :

1. now, right click on the project name then click on “ ADD Component “ and then select the “ Windows Forms “ and save it as “stock.cs”
2. navigate to the Form Tool Box and search for “ Error provider “ and double click on that then navigate to the stock.cs



The screenshot shows a Windows Forms application titled "Stock". The form has a light gray background and a blue title bar. At the top, there are five labels: "Date", "Product Code", "Product Name", "Quantity", and "Status". Below these labels are five input fields. The "Date" field contains "05-11-2022" and has a calendar icon. The "Product Code", "Product Name", and "Quantity" fields are empty. The "Status" field is a dropdown menu with a downward arrow. Below the input fields are three buttons: "Add", "Delete", and "Reset". Below the buttons is a table with the following columns: "S.No", "Product Code", "Product Name", "Quantity", "Date", and "Status". The table is currently empty. At the bottom of the form, there are two labels: "Total Products : 0" and "Total Quantity : 0".

3. When you add combo box to the status then in drop down add these 2 parameters also they are “ Active & Deactive “.
4. The code for Stock form will below there :

## Stock.cs Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Stock_Management
{
    public partial class Stock : Form
    {
        public Stock()
        {
            InitializeComponent();
        }

        private void Stock_Load(object sender, EventArgs e)
        {
            this.ActiveControl = dateTimePicker1;
            comboBox1.SelectedIndex = 0;
            LoadData();

            Search();
        }

        private void dateTimePicker1_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                textBox1.Focus();
            }
        }
    }
}
```

```

    }
}

private void textBox1_KeyDown(object sender, KeyEventArgs e)
{
    if(e.KeyCode == Keys.Enter)
    {
        if(dgview.Rows.Count > 0)
        {
            textBox1.Text = dgview.SelectedRows[0].Cells[0].Value.ToString();
            textBox2.Text = dgview.SelectedRows[0].Cells[1].Value.ToString();
            this.dgview.Visible = false;
            textBox3.Focus();
        }
        else
        {
            this.dgview.Visible = false;
        }
    }
}

private void textBox2_KeyDown(object sender, KeyEventArgs e)
{
    if(e.KeyCode == Keys.Enter)
    {
        if(textBox2.Text.Length > 0)
        {
            textBox3.Focus();
        }
        else
        {
            textBox2.Focus();
        }
    }
}

bool change = true;

```

```
private void proCode_MouseDoubleClick(object sender,
MouseEventArgs e)
{
    if (change)
    {
        change = false;
        textBox1.Text = dgview.SelectedRows[0].Cells[0].Value.ToString();
        textBox2.Text = dgview.SelectedRows[0].Cells[1].Value.ToString();
        this.dgview.Visible = false;
        textBox3.Focus();
        change = true;
    }
}
```

```
private void textBox3_KeyDown(object sender, KeyEventArgs e)
{
    if(e.KeyCode == Keys.Enter)
    {
        if(textBox3.Text.Length > 0)
        {
            comboBox1.Focus();
        }
        else
        {
            textBox3.Focus();
        }
    }
}
```

```
private void comboBox1_KeyDown(object sender, KeyEventArgs e)
{
    if(e.KeyCode == Keys.Enter)
    {
        if(comboBox1.SelectedIndex != -1)
        {
            button1.Focus();
        }
    }
}
```

```

        else
        {
            comboBox1.Focus();
        }
    }
}

private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    if(!char.IsNumber(e.KeyChar) & (Keys)e.KeyChar != Keys.Back &
e.KeyChar != '.')
    {
        e.Handled = true;
    }
}

private void textBox3_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsNumber(e.KeyChar) & (Keys)e.KeyChar != Keys.Back &
e.KeyChar != '.')
    {
        e.Handled = true;
    }
}

private void ResetRecords()
{
    dateTimePicker1.Value = DateTime.Now;
    textBox1.Clear();
    textBox2.Clear();
    textBox3.Clear();
    comboBox1.SelectedIndex = -1;
    button1.Text = "Add";
    dateTimePicker1.Focus();
}

private void button3_Click(object sender, EventArgs e)

```



```

{
    ResetRecords();
}

private bool Validation()
{
    bool result = false;

    if(string.IsNullOrEmpty(textBox1.Text))
    {
        errorProvider1.Clear();
        errorProvider1.SetError(textBox1, " Product Code Required ");
    }
    else if(string.IsNullOrEmpty(textBox2.Text))
    {
        errorProvider1.Clear();
        errorProvider1.SetError(textBox2, " Product Name Required ");
    }
    else if(string.IsNullOrEmpty(textBox3.Text))
    {
        errorProvider1.Clear();
        errorProvider1.SetError(textBox3, " Quantity Required ");
    }
    else if(comboBox1.SelectedIndex == -1)
    {
        errorProvider1.Clear();
        errorProvider1.SetError(comboBox1, " Select Status ");
    }
    else
    {
        errorProvider1.Clear();
        result = true;
    }
    return result;
}

private bool IfProductExists(SqlConnection con, string productCode)

```

```
{

    SqlDataAdapter sda = new SqlDataAdapter("select 1 from Stock where
ProductCode = '" + productCode + "'", con);
    DataTable dt = new DataTable();
    sda.Fill(dt);

    if (dt.Rows.Count > 0)
    {
        return true;
    }
    else
    {
        return false;
    }
}

private void button1_Click(object sender, EventArgs e)
{
    // Add Button
    if(Validation())
    {
        SqlConnection con = Connection.getConnection();
        con.Open();
        bool status = false;

        if(comboBox1.SelectedIndex == 0)
        {
            status = true;
        }
        else
        {
            status = false;
        }

        var sqlquery = "";
        if (IfProductExists(con, textBox1.Text))
```

```

        {
            sqlquery = "update Stock set ProductName = @ProductName,
ProductStatus = @ProductStatus, Quantity = @Quantity where ProductCode
= @ProductCode";
        }
        else
        {
            sqlquery = "insert into Stock values(@ProductCode,
@ProductName, @TransDate, @Quantity, @ProductStatus)";
        }

        SqlCommand cmd = new SqlCommand(sqlquery, con);
        cmd.Parameters.AddWithValue("@ProductCode",
int.Parse(textBox1.Text));
        cmd.Parameters.AddWithValue("@ProductName", textBox2.Text);
        cmd.Parameters.AddWithValue("@TransDate",
dateTimePicker1.Value.ToString("MM/dd/yyyy"));
        cmd.Parameters.AddWithValue("@Quantity", textBox3.Text);
        cmd.Parameters.AddWithValue("@ProductStatus", status);
        cmd.ExecuteNonQuery();

        con.Close();
        MessageBox.Show(" Record Saved Successfully ");
        LoadData();

        ResetRecords(); //..
    }
}

public void LoadData()
{
    SqlConnection con = Connection.getConnection();
    SqlDataAdapter sda = new SqlDataAdapter("select * from Stock", con);
    DataTable dt = new DataTable();
    sda.Fill(dt);
    dataGridView1.Rows.Clear();
}

```

```

foreach(DataRow item in dt.Rows)
{
    int n = dataGridView1.Rows.Add();
    dataGridView1.Rows[n].Cells["dgSno"].Value = n + 1;
    dataGridView1.Rows[n].Cells["dgProCode"].Value =
item["ProductCode"].ToString();
    dataGridView1.Rows[n].Cells["dgProName"].Value =
item["ProductName"].ToString();
    dataGridView1.Rows[n].Cells["dgQuantity"].Value =
float.Parse(item["Quantity"].ToString());
    dataGridView1.Rows[n].Cells["dgDate"].Value =
Convert.ToDateTime(item["TransDate"].ToString()).ToString("dd/MM/yyyy");

    if ((bool)item["ProductStatus"])
    {
        dataGridView1.Rows[n].Cells["dgStatus"].Value = "Active";
    }
    else
    {
        dataGridView1.Rows[n].Cells["dgStatus"].Value = "Deactive";
    }
}

if(dataGridView1.Rows.Count > 0)
{
    label8.Text = dataGridView1.Rows.Count.ToString();
    float totQty = 0;
    for(int i = 0; i < dataGridView1.Rows.Count; ++i)
    {
        totQty +=
float.Parse(dataGridView1.Rows[i].Cells["dgQuantity"].Value.ToString());
        label9.Text = totQty.ToString();
    }
}
else
{
    label8.Text = "0";
}

```

```

        label9.Text = "0";
    }
}

private void dataGridView1_MouseDoubleClick(object sender,
MouseEventArgs e)
{
    button1.Text = "Update";
    textBox1.Text =
dataGridView1.SelectedRows[0].Cells["dgProCode"].Value.ToString();
    textBox2.Text =
dataGridView1.SelectedRows[0].Cells["dgProName"].Value.ToString();
    textBox3.Text =
dataGridView1.SelectedRows[0].Cells["dgQuantity"].Value.ToString();
    dateTimePicker1.Text =
DateTime.Parse(dataGridView1.SelectedRows[0].Cells["dgDate"].Value.ToStri
ng()).ToString("dd/MM/yyyy");

    if (dataGridView1.SelectedRows[0].Cells["dgStatus"].Value.ToString()
== "Active")
    {
        comboBox1.SelectedIndex = 0;
    }
    else
    {
        comboBox1.SelectedIndex = 1;
    }
}

private void button2_Click(object sender, EventArgs e)
{
    // Delete Button
    DialogResult dialogResult = MessageBox.Show(" Are You Sure Want To
Delete ", "Message", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        if (Validation())

```

```

{
    SqlConnection con = Connection.getConnection();
    con.Open();

    var sqlquery = "";
    if (IfProductExists(con, textBox1.Text))
    {
        sqlquery = "delete from Stock where ProductCode =
@ProductCode";

        SqlCommand cmd = new SqlCommand(sqlquery, con);
        cmd.Parameters.AddWithValue("@ProductCode",
int.Parse(textBox1.Text));
        cmd.ExecuteNonQuery();
    }
    else
    {
        MessageBox.Show("Record Doesn't Exists in the table ..... !");
    }

    con.Close();

    LoadData();
    ResetRecords();
}
}

private void textBox1_TextChanged(object sender, EventArgs e)
{
    // product Code textbox
    if(textBox1.Text.Length > 0)
    {
        this.dgview.Visible = true;
        dgview.BringToFront();
        Search(150, 105, 430, 200, "Pro Code, Pro Name", "100");
        // To do : mouse Double Click Event
    }
}

```

```
this.dgview.MouseDoubleClick += new  
System.Windows.Forms.MouseEventHandler(this.proCode_MouseDoubleClic  
k);
```

```
SqlConnection con = Connection.getConnection();  
con.Open();
```

```
SqlCommand cmd = new SqlCommand("select Top(10) ProductCode,  
ProductName from Products where ProductCode like @ProductCode", con);  
cmd.Parameters.AddWithValue("@ProductCode",  
int.Parse(textBox1.Text));
```

```
SqlDataAdapter sda = new SqlDataAdapter(cmd);  
DataTable dt = new DataTable();  
sda.Fill(dt);
```

```
dgview.Rows.Clear();  
foreach(DataRow row in dt.Rows)  
{  
    int n = dgview.Rows.Add();  
    dgview.Rows[n].Cells[0].Value = row["ProductCode"].ToString();  
    dgview.Rows[n].Cells[1].Value = row["ProductName"].ToString();  
}  
}  
else  
{  
    dgview.Visible = false;  
}  
}
```

```
private DataGridView dgview;  
private DataGridViewTextBoxColumn dgviewcol1;  
private DataGridViewTextBoxColumn dgviewcol2;
```

```
void Search()  
{  
    dgview = new DataGridView();
```

```

        dgviewcol1 = new DataGridViewTextBoxColumn();
        dgviewcol2 = new DataGridViewTextBoxColumn();
        this.dgview.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoS
ize;
        this.dgview.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] { this.dgviewcol1,
this.dgviewcol2 });
        this.dgview.Name = "dgview";
        dgview.Visible = false;
        this.dgviewcol2.AutoSizeMode =
DataGridViewAutoSizeColumnMode.Fill;
        this.dgviewcol1.Visible = false;
        this.dgviewcol2.Visible = false;
        this.dgview.AllowUserToAddRows = false;
        this.dgview.RowHeadersVisible = false;
        this.dgview.SelectionMode =
System.Windows.Forms.DataGridViewSelectionMode.FullRowSelect;
        //this.dgview.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.dgview_KeyDown);

        this.Controls.Add(dgview);
        this.dgview.ReadOnly = true;
        dgview.BringToFront();
    }

    //Two Column
    void Search(int LX, int LY, int DW, int DH, string ColName, String ColSize)
    {
        this.dgview.Location = new System.Drawing.Point(LX, LY);
        this.dgview.Size = new System.Drawing.Size(DW, DH);

        string[] ClSize = ColSize.Split(',');
        //Size
        for(int i = 0; i < ClSize.Length; i++)
        {
            if(int.Parse(ClSize[i]) != 0)

```



```

        {
            dgvview.Columns[i].Width = int.Parse(CISize[i]);
        }
        else
        {
            dgvview.Columns[i].AutoSizeMode =
System.Windows.Forms.DataGridViewAutoSizeColumnMode.Fill;
        }
    }
    //Name
    string[] CIName = ColName.Split(',');

    for (int i = 0; i < CIName.Length; i++)
    {
        this.dgvview.Columns[i].HeaderText = CIName[i];
        this.dgvview.Columns[i].Visible = true;
    }
}

}
}

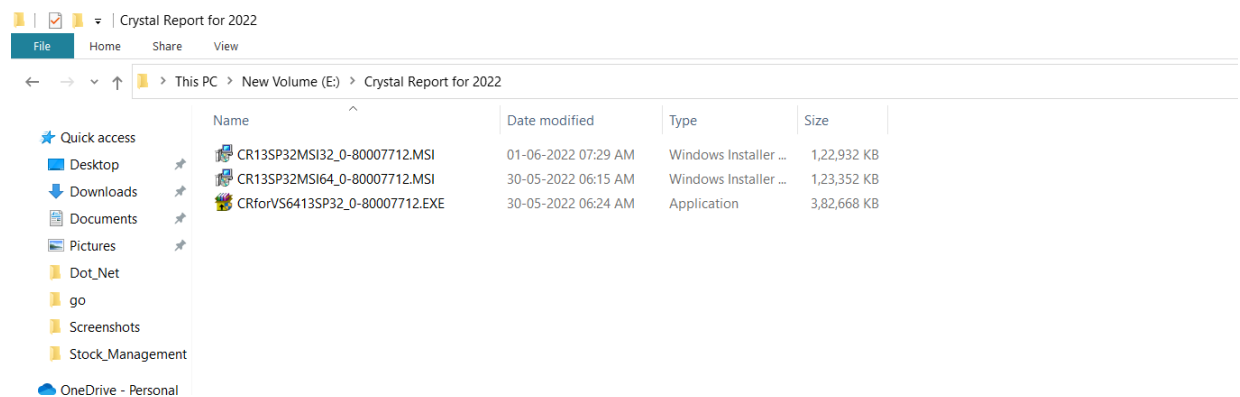
```

## Installation Of The Crystal Reporting Services Tool

1. Before we go for the generating the reports for the products and stock windows form.
2. First we have to install the crystal reporting services tool in your systems.
3. Download the crystal reporting services from the below link

<https://drive.google.com/file/d/1AEuZ2AM-5B1daK0BAQEhRQCNqHbe-dim/view?usp=sharing>

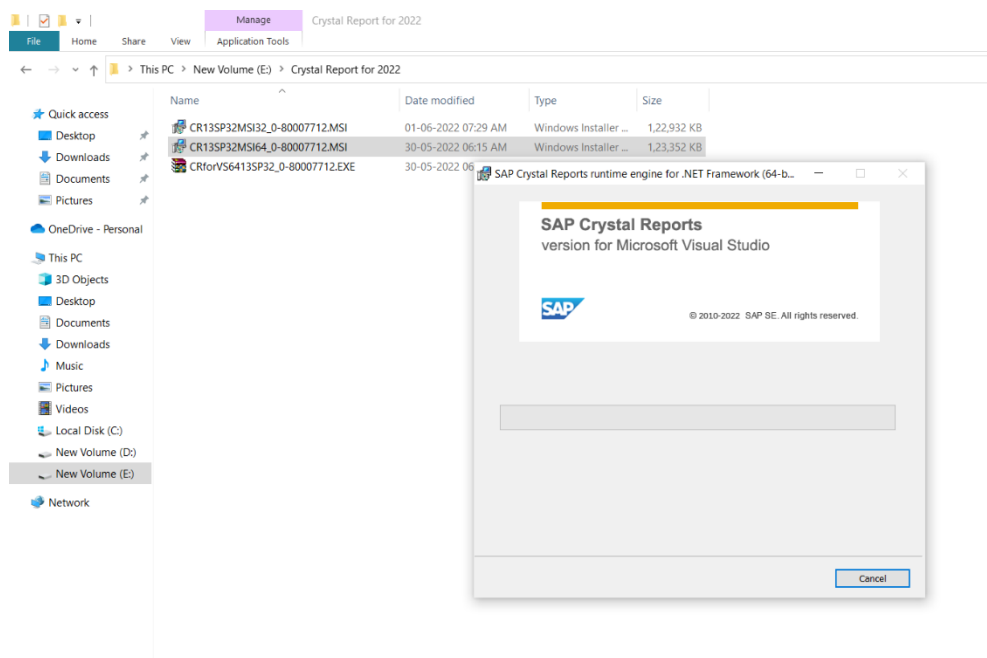
4. After downloading the file then unzip the rar file you will see there are 3 files in that folder.



5. First run the “ Second File ” and give all permissions and access to the tool.

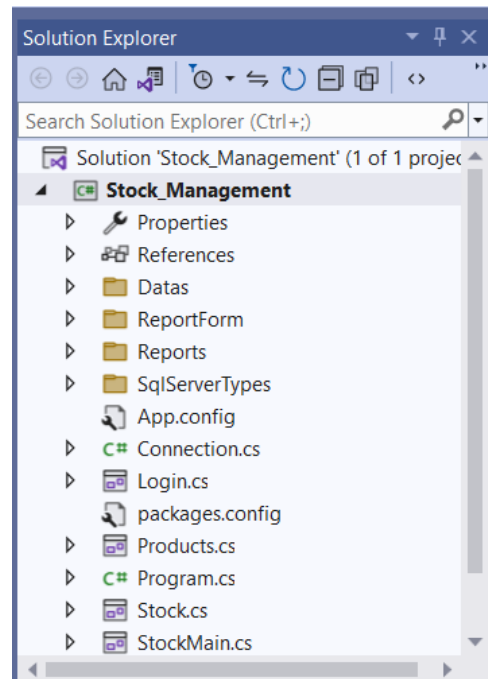
6. Next run the “ Last File ” and give all permissions and access to the tool.

### Sample Installation Pic :

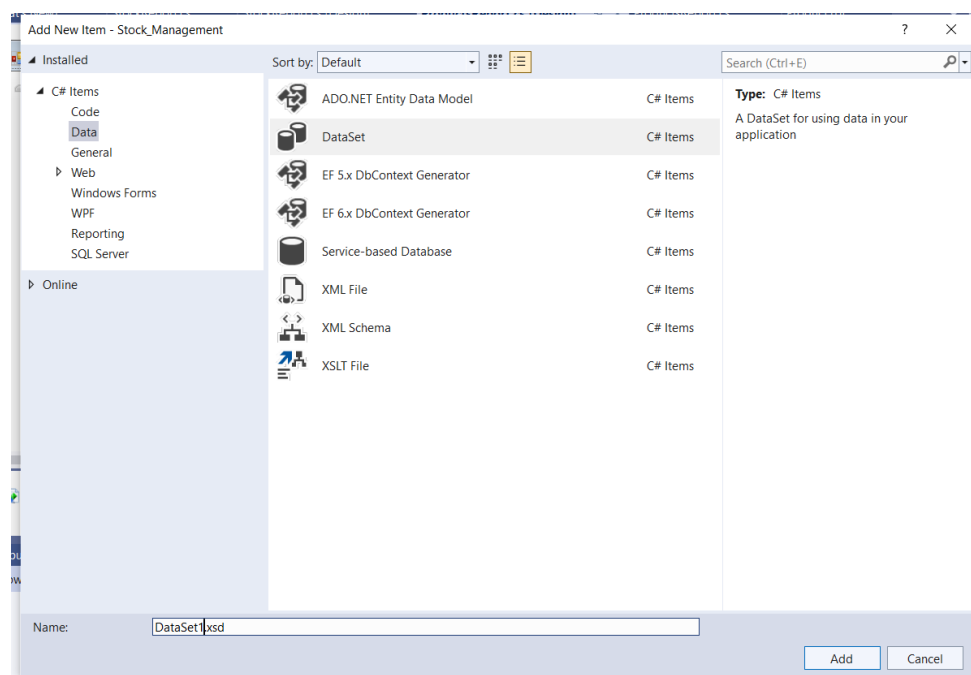


## Dataset Creation Part

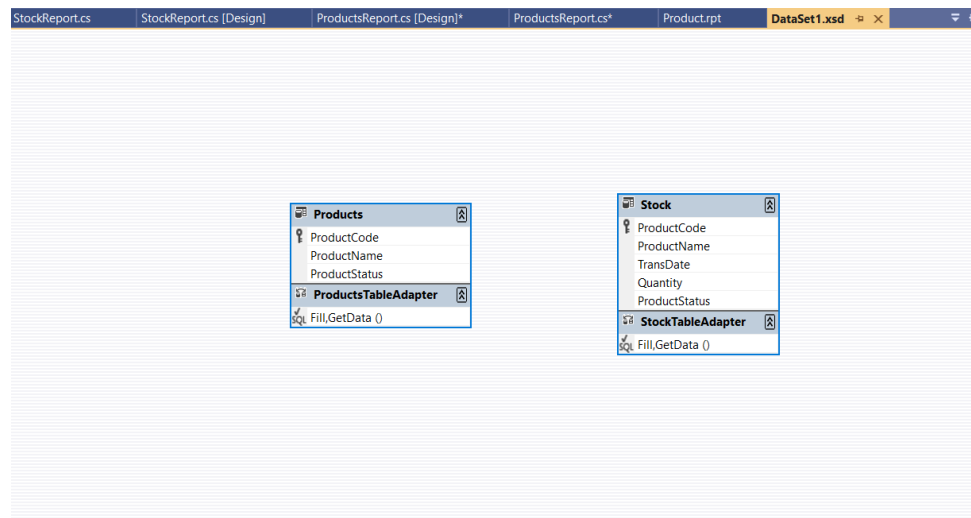
1. Open the solution explorer and right click on project name and create a 3 new folders with Names “ Datas “, “ ReportForm “ “ Reports “ just like below one.



2. Now right click on Project name and choose “ Add item option” And then select the Dataset option.

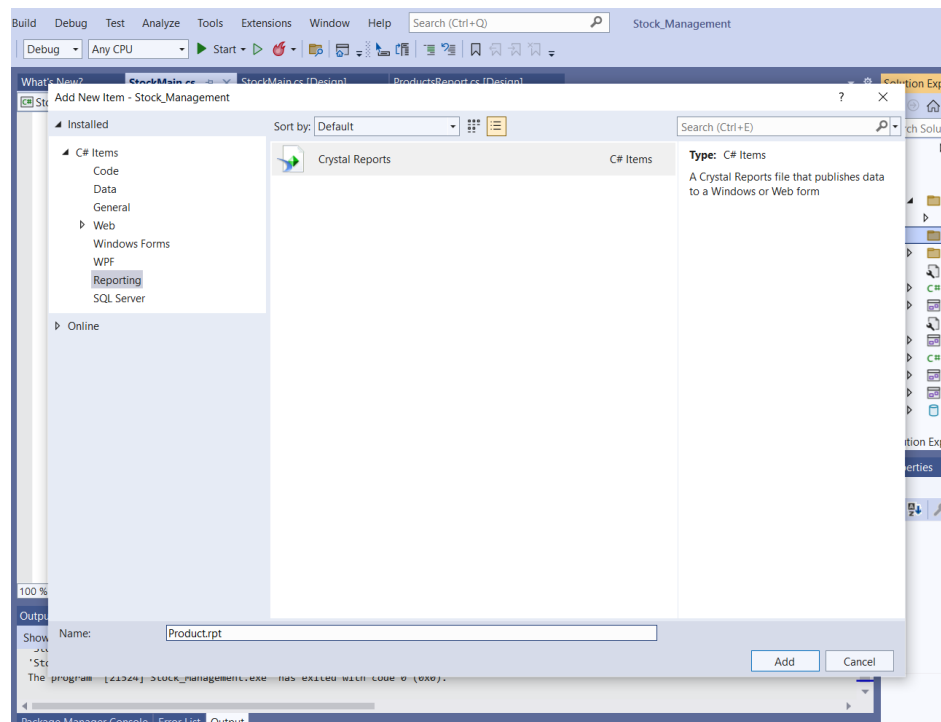


3. After creating it “ Double click on the DataSet1.xsd “ and just drag the Products and Stock tables from the server explorer into the DataSet1.xsd

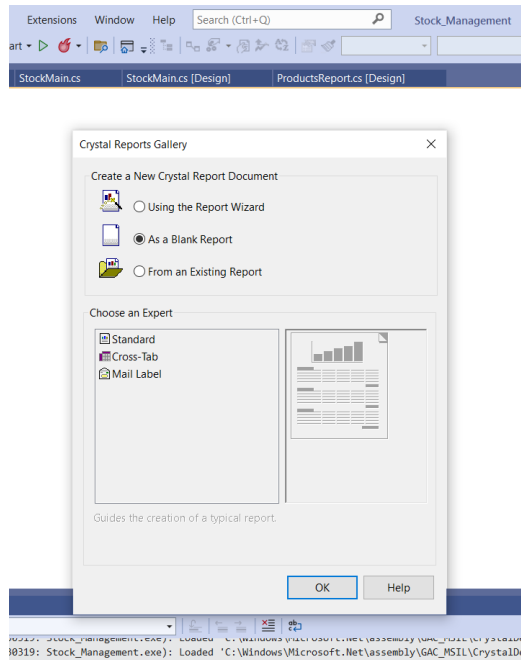


## Product Report Part

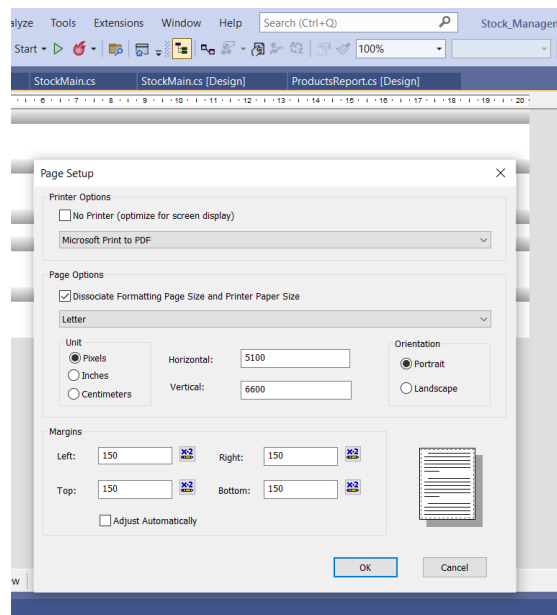
1. Right click on the reports folder and then add an item crystal reporting and save as Product.rpt



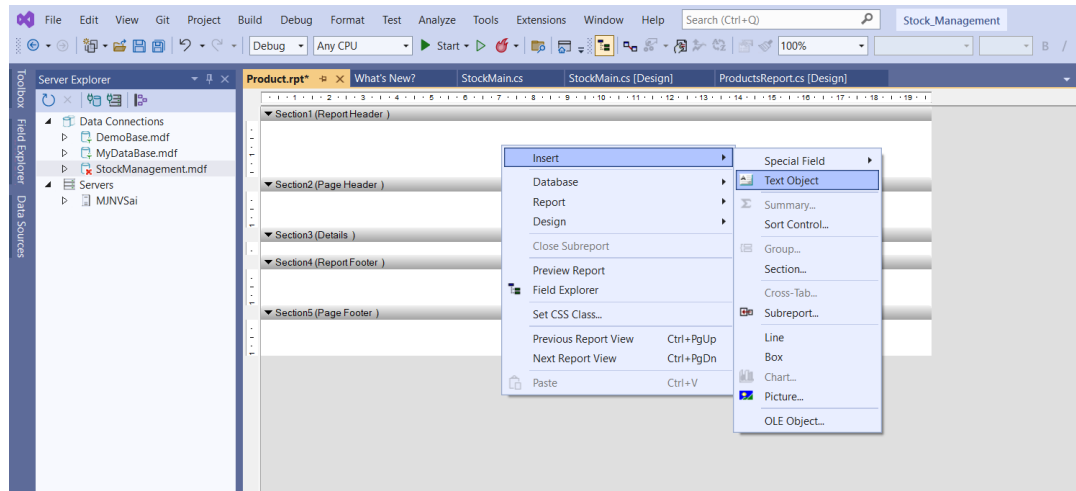
2. Select the Blank report option and click on ok.



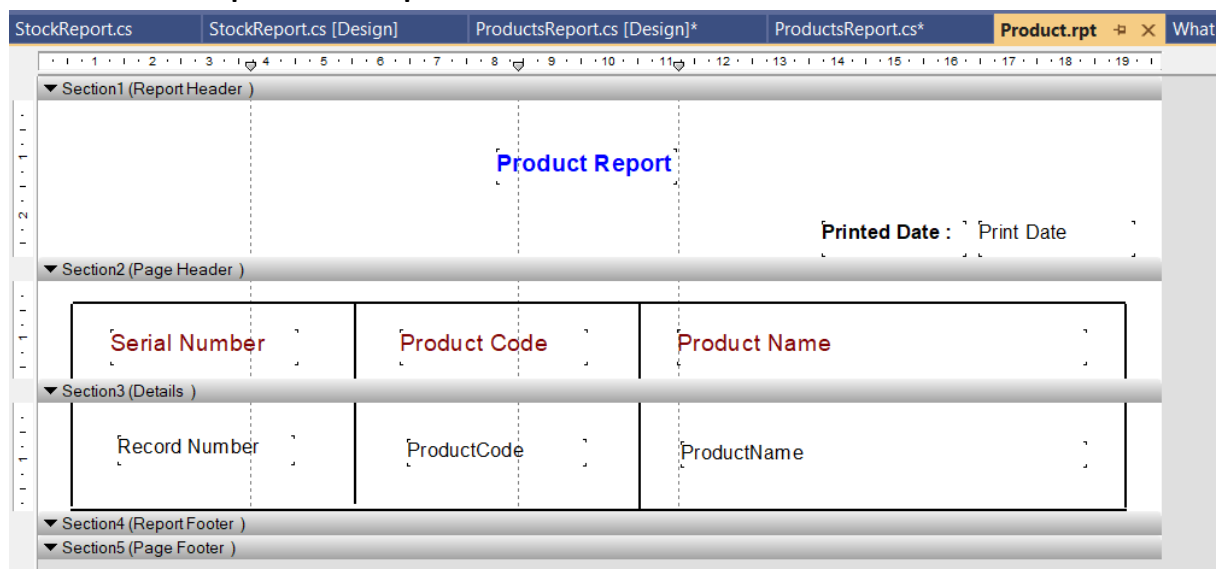
3. Now right click on the section 1 and select the page setup  
Change the Letter format to the A4 sheet format.



4. To write on the report , to print the date , row number in any section you just right click on that section you will get the options releated to that.  
For example you want textfield you can select or you date option then go to the special field.



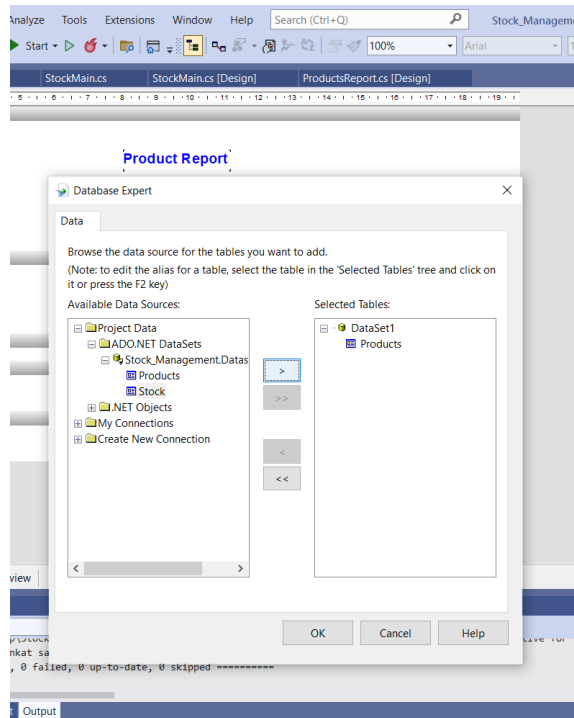
5. The final report template will be like below one.



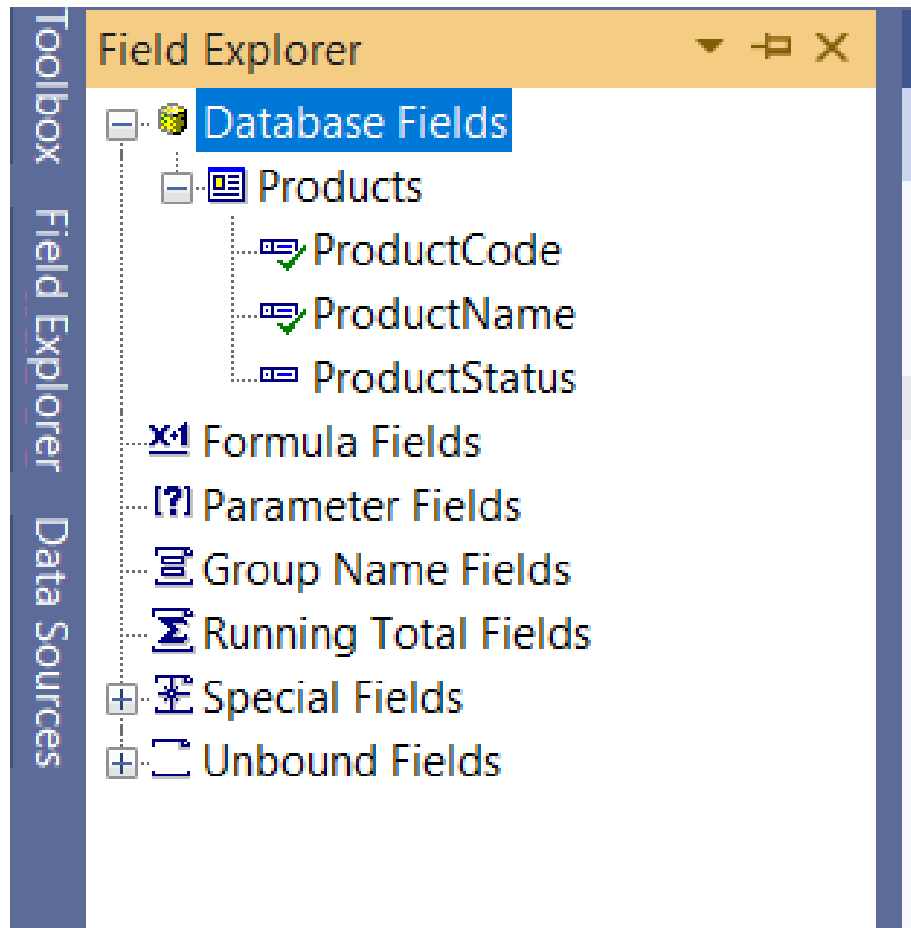
6. You can design the section 1, section 2, section 4, section 5 in the report in your way but the section 3 is connected to the dataset and database.

7. On left side panel click on the Field Explorer tab and then right click on the Database option then choose “ database expert “ Option.

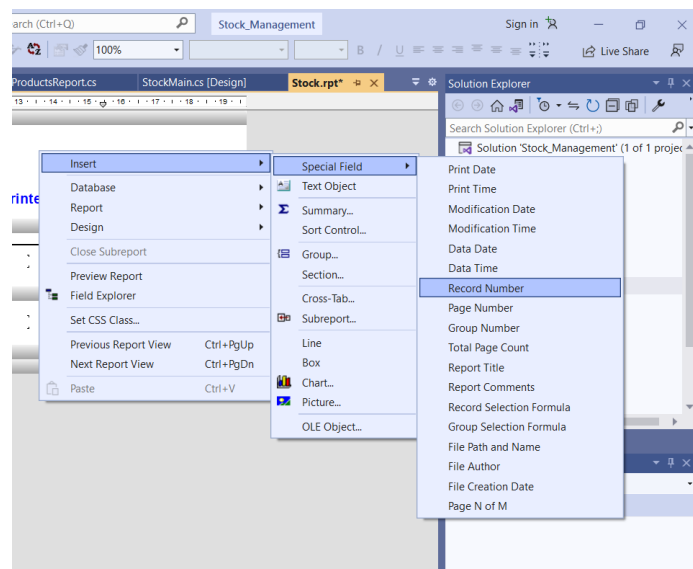
8. Now send the “ Products table “ from the available data sources to the selected tables list then click on ok.



9. Now drag the Product code , product name fields from the database fields to the “ Details Section “.



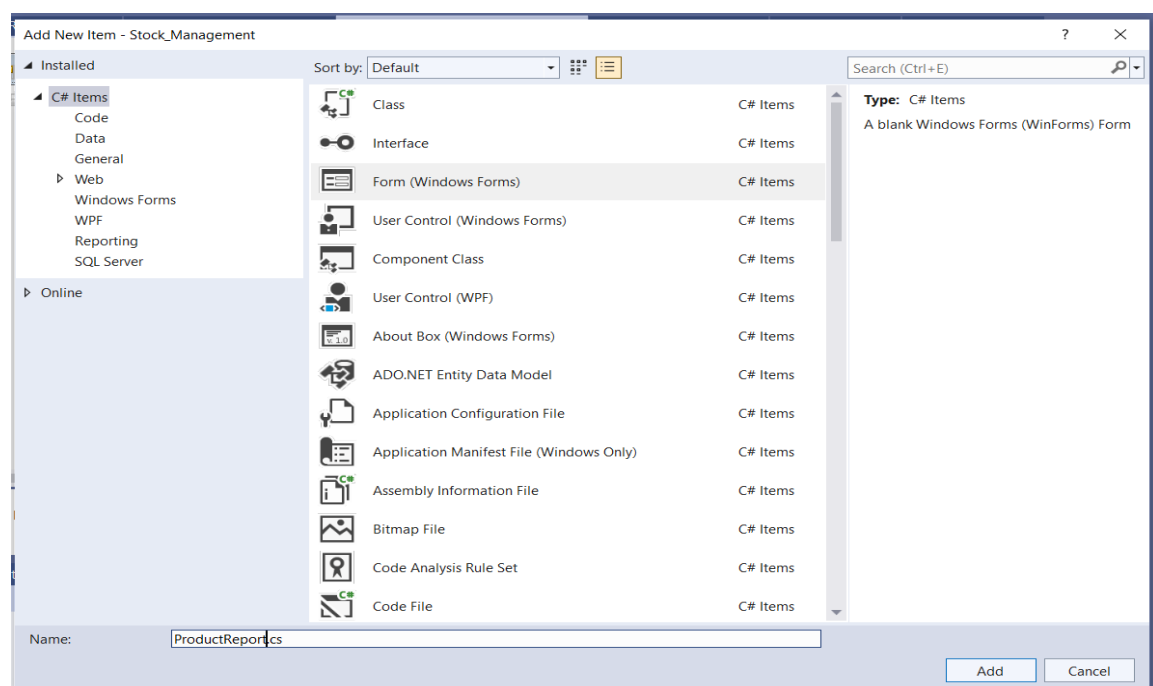
10. In details section you can record number box just like below one.



11. With this the Product report template is completed.

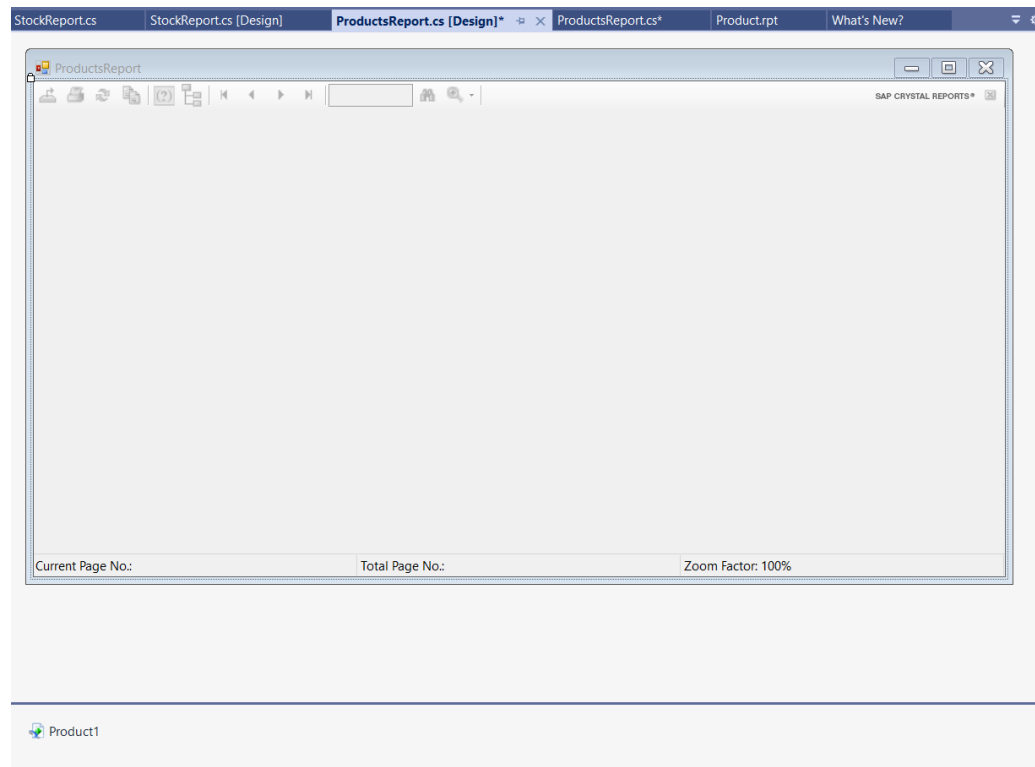
## Product Report Windows Form Part

1. Right click on the reportform folder and add a windows form and save it as a name ProductReport.cs





2. Now select the “ crystal report viwer tool “ from the tool box and drag into the windows form.



3. Double click on the top of the windows form then it will open the Form load section code.

### ProductReport.cs File Code

```
using CrystalDecisions.CrystalReports.Engine;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using Stock_Management.Datas;

namespace Stock_Management.ReportForm
{
    public partial class ProductsReport : Form
    {
        ReportDocument cryprt = new ReportDocument();
        public ProductsReport()
        {
            InitializeComponent();
        }
    }
}
```

```

private void ProductsReport_Load(object sender, EventArgs e)
{
    // Load Form

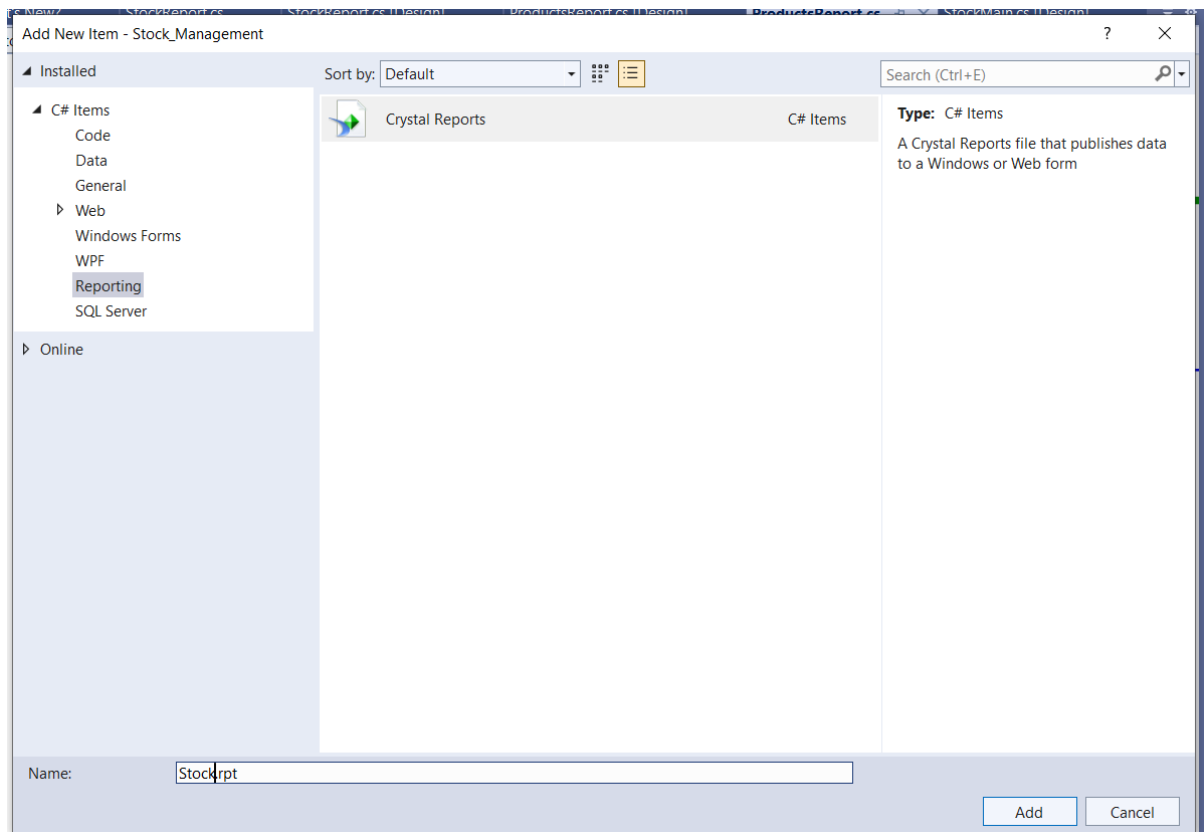
    cryprt.Load(@"E:\venkat
sai\Dot_Net\Csharp\Stock_Management\Stock_Management\Reports\Product.rpt");
    SqlConnection con = Connection.getConnection();
    con.Open();

    DataSet dst = new DataSet();
    SqlDataAdapter sda = new SqlDataAdapter("Select * From Products", con);
    DataTable dt = new DataTable();
    sda.Fill(dt);
    cryprt.SetDataSource(dt);
    crystalReportViewer1.ReportSource = cryprt;
}
}
}

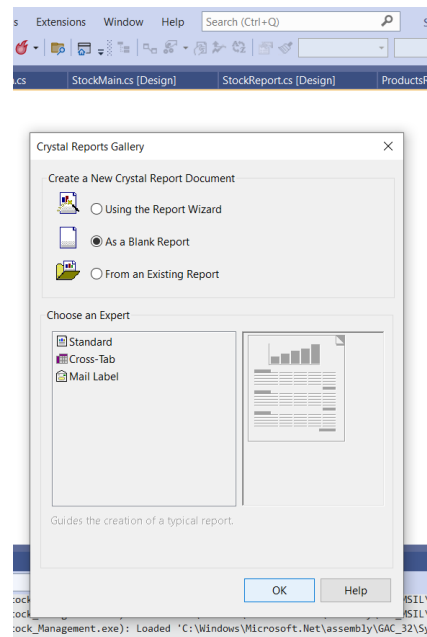
```

## Stock Report Part

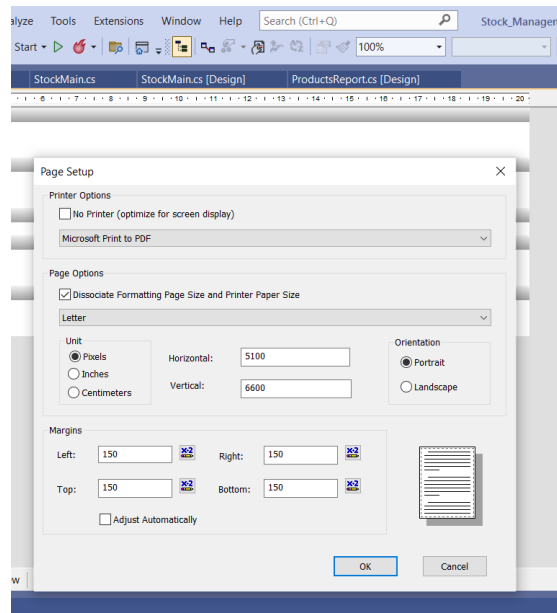
1. Right click on the reports folder and then add an item crystal reporting and save as Stock.rpt



2. Select the blank report as an option and click on ok.

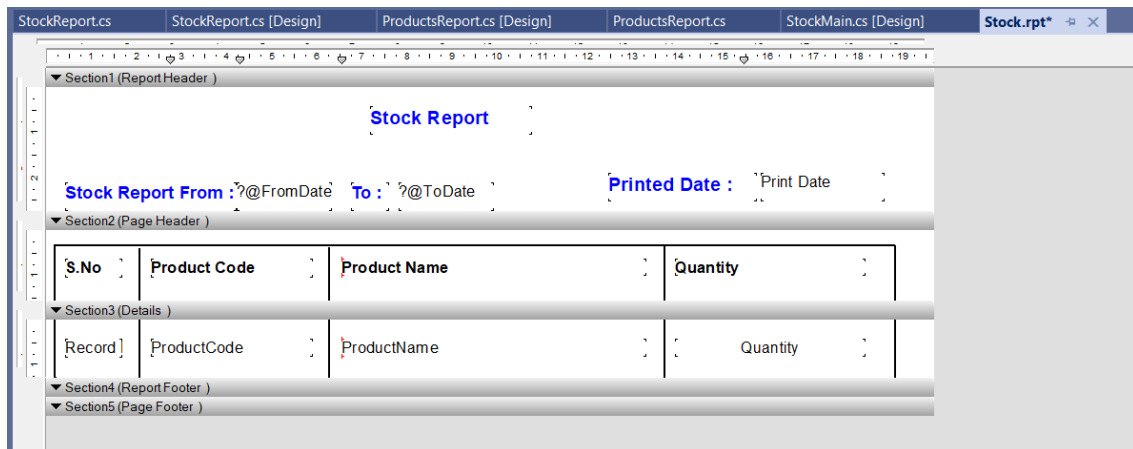


3. Now right click on the section 1 and select the page setup  
Change the Letter format to the A4 sheet format.



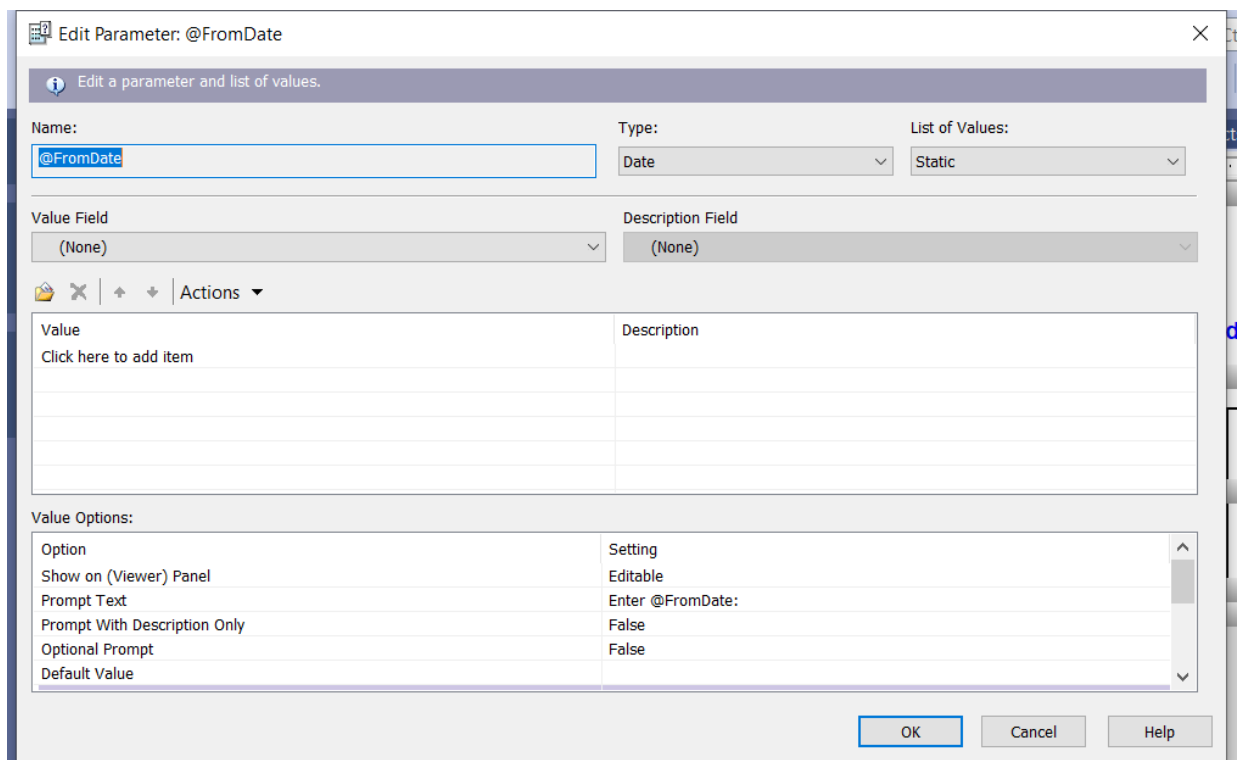
4. Now design the stock report template in your way and in header section you have print the from date to the to date date.

5. We can achieve that by using the parameter fields in the field explorer tab.



6. Click on field tools tab on the left side panel there right click on the parameter fields and click on NEW.

7. You have to 2 parameters fields one is FromDate field and ToDate Field.



**Edit Parameter: @ToDate**

Edit a parameter and list of values.

Name:  Type:  List of Values:

Value Field:  Description Field:

Value

Click here to add item

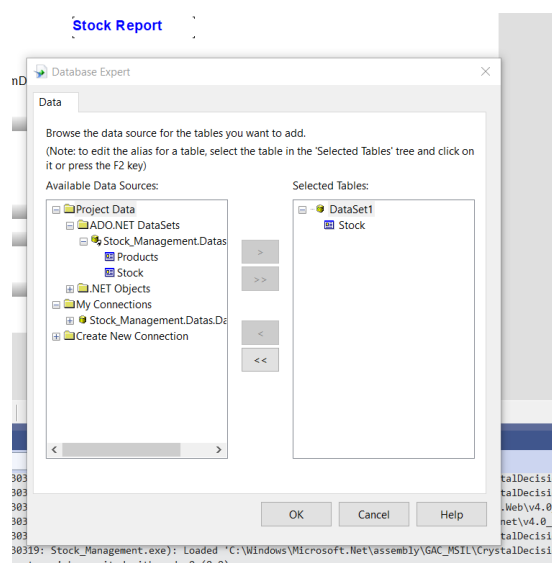
Description

Value Options:

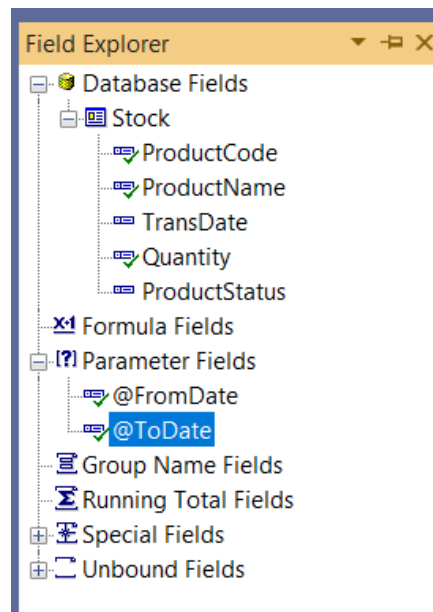
Option	Setting
Show on (Viewer) Panel	Editable
Prompt Text	Enter @ToDate:
Prompt With Description Only	False
Optional Prompt	False
Default Value	

OK Cancel Help

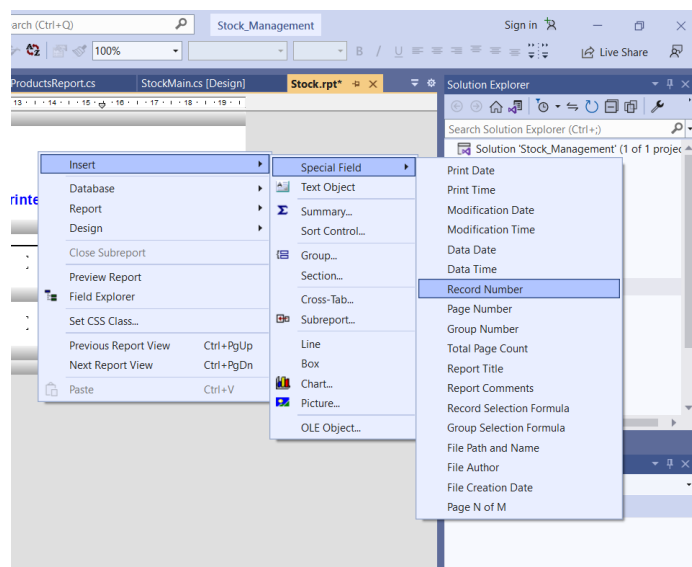
8. After adding both parameters into the parameter fields. Just drag the FromDate and ToDate fields from the parameter fields to the Section 1 (Report header ).
9. Now right click on the database fields and choose the database expert option.
10. Now send the “ Products table “ from the available data sources to the selected tables list then click on ok.



11. Now drag the Product code , product name, quantity fields from the database fields to the “ Details Section “.



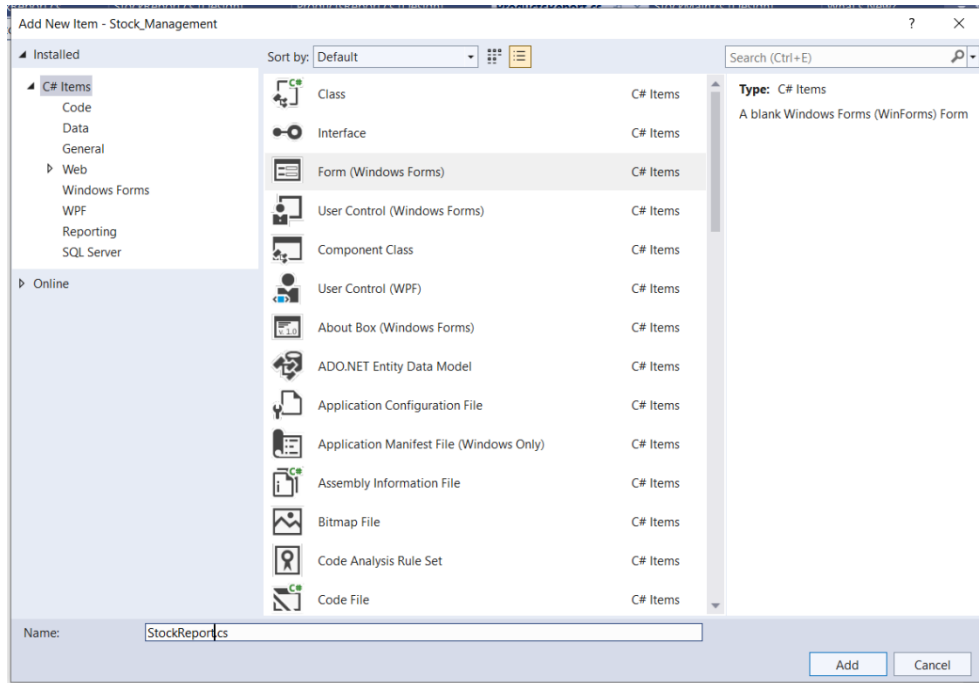
12. In details section you can record number box just like below one.



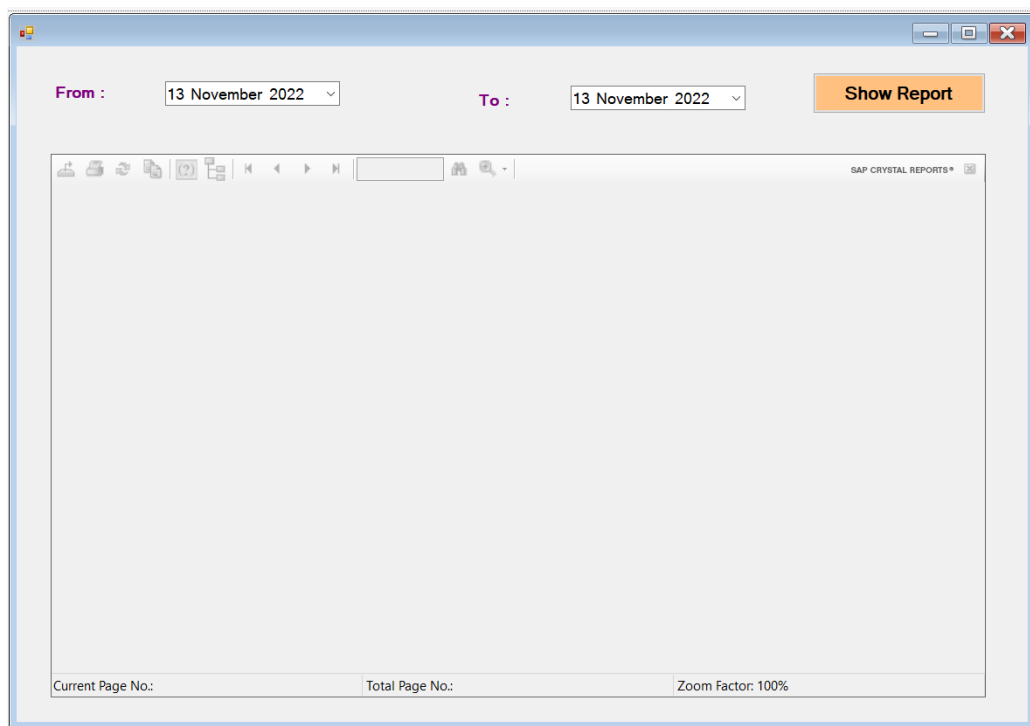
13. With these the Stock report template is completed.

## Stock Report Windows Form Part

1. Right click on the reportform folder and add a windows form and save it as a name StockReport.cs



2. Now select the “ crystal report viwer tool “ from the tool box and drag into the windows form and design the form just like below one..



3. Double click on the Show report button then it will open the Button section code.

### StockReport.cs Form Code

```
using CrystalDecisions.CrystalReports.Engine;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Stock_Management.ReportForm
{
    public partial class StockReport : Form
    {
        ReportDocument crystal = new ReportDocument();
        public StockReport()
        {
            InitializeComponent();
        }

        private void StockReport_Load(object sender, EventArgs e)
        {
            // Form load Section
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // Show Report
            crystal.Load(@"E:\venkat
sai\Dot_Net\Csharp\Stock_Management\Stock_Management\Reports\Stock.rpt");
            SqlConnection con = Connection.getConnection();
            con.Open();

            DataSet dst = new DataSet();
            SqlDataAdapter sda = new SqlDataAdapter("Select * From Stock where Cast(
TransDate as Date) between '" + dateTimePicker1.Value.ToString("MM/dd/yyyy") + "' and '" +
dateTimePicker2.Value.ToString("MM/dd/yyyy") + "'", con);

            sda.Fill(dst, "Stock");
            crystal.SetDataSource(dst);

            crystal.SetParameterValue("@FromDate",
dateTimePicker1.Value.ToString("dd/MM/yyyy"));
            crystal.SetParameterValue("@ToDate",
dateTimePicker2.Value.ToString("dd/MM/yyyy"));
            crystalReportViewer1.ReportSource = crystal;
        }
    }
}
```



## Update the StockMain Windows Form Part

### Updated StockMain.cs Form Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Stock_Management
{
    public partial class StockMain : Form
    {
        public StockMain()
        {
            InitializeComponent();
        }

        private void productsToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            // products option in Navbar
            Products pro = new Products();
            pro.MdiParent = this;
            pro.StartPosition = FormStartPosition.CenterScreen;
            pro.Show();
        }

        bool close = true;
```

```

private void StockMain_FormClosing(object sender,
FormClosingEventArgs e)
{
    if(close)
    {
        DialogResult result = MessageBox.Show(" Are You Sure Want To Exit
", "Exit", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

        if (result == DialogResult.Yes)
        {
            close = false;
            Application.Exit();
        }
        else
        {
            e.Cancel = true;
        }
    }
}

private void stockToolStripMenuItem_Click(object sender, EventArgs e)
{
    // Stock
    Stock stk = new Stock();
    stk.MdiParent = this;
    stk.StartPosition = FormStartPosition.CenterScreen;
    stk.Show();
}

private void productListToolStripMenuItem_Click(object sender,
EventArgs e)
{
    // Products Report
    ReportForm.ProductsReport prod = new
ReportForm.ProductsReport();
    prod.MdiParent = this;
    prod.StartPosition = FormStartPosition.CenterScreen;

```

```

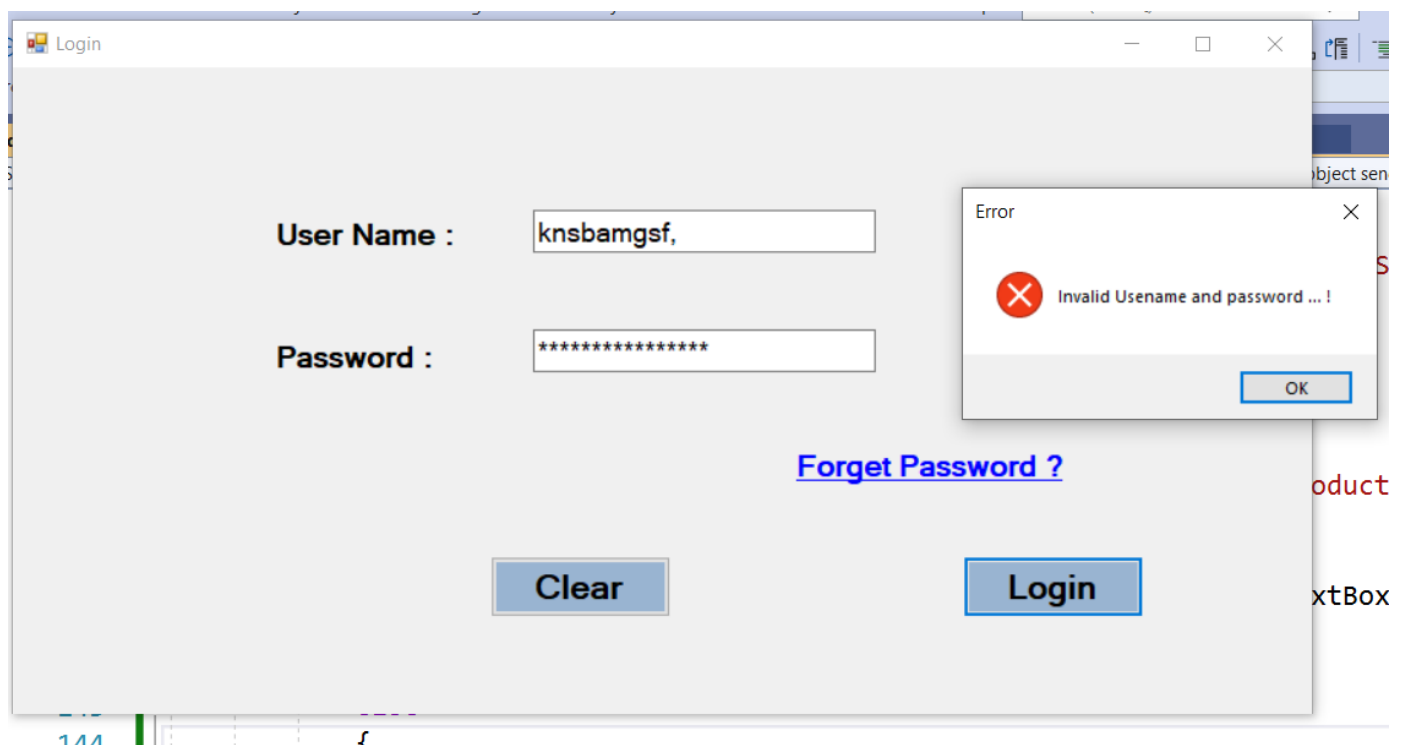
        prod.Show();
    }

    private void stockListToolStripMenuItem_Click(object sender, EventArgs
e)
    {
        // Stock Report
        ReportForm.StockReport prod = new ReportForm.StockReport();
        prod.MdiParent = this;
        prod.StartPosition = FormStartPosition.CenterScreen;
        prod.Show();
    }
}

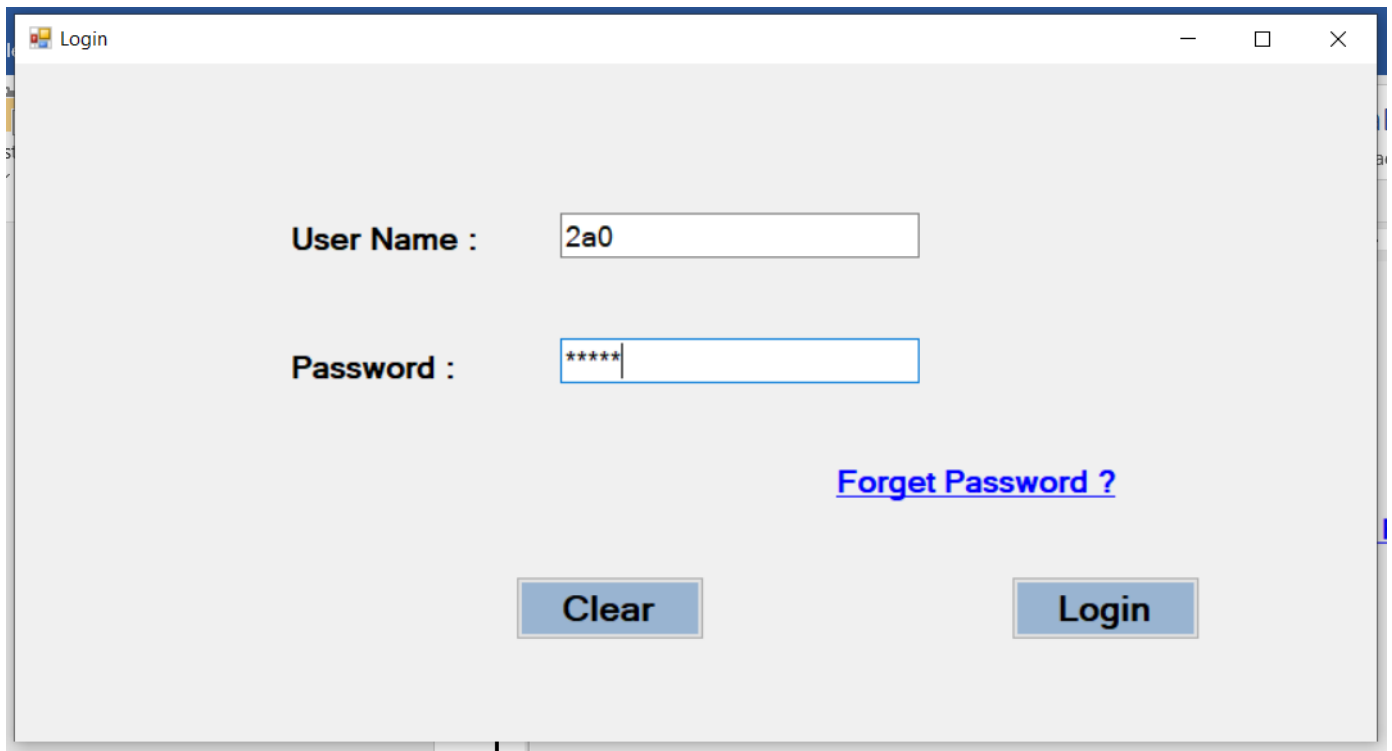
```

**Outputs :**

**Invalid Login :**



## Valid Login :



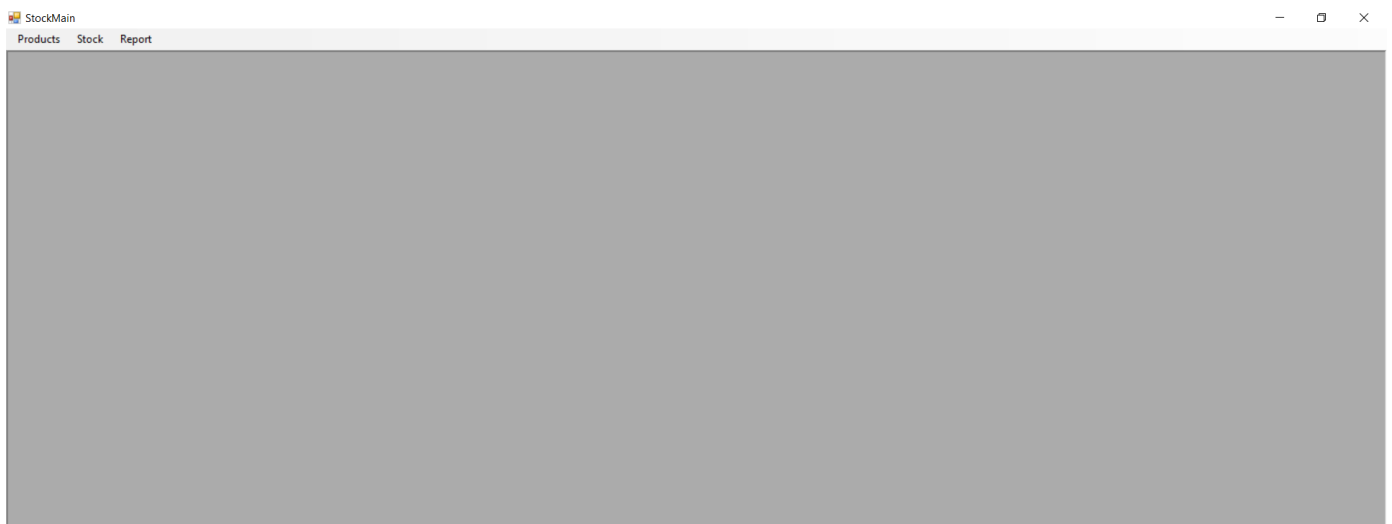
The screenshot shows a window titled "Login" with a light gray background. It contains two input fields: "User Name :" with the value "2a0" and "Password :" with masked characters "\*\*\*\*\*". Below the password field is a blue hyperlink "Forget Password ?". At the bottom are two buttons: "Clear" and "Login".

User Name : 2a0

Password : \*\*\*\*\*

[Forget Password ?](#)

Clear Login



## Products Form :

### Button : Add

StockMain  
Products Stock Report

Products

Product Code Product Name Status

Active

Product Code	Product Name	Status
67	of 1L	Deactive
99	Chicken Noodles	Deactive
235	Beans 1kg	Active

Products

Product Code Product Name Status

Product Code	Product Name	Status
56	Cricket	Deactive
66	Charger	Deactive
68	Asus laptop	Active
75	Kokine	Active
80	2 Shirts	Active
83	Iphone 14 pro	Active
99	Chicken Noodles	Deactive
100	Fanta 10L	Deactive
345	5 Shorts	Active
456	Drinks	Deactive
789	Apples	Active

## Button : Update

Products

Product Code

Product Name

Status

56

Cricket Bat

Active

Update

Delete

Reset

	Product Code	Product Name	Status
▶	56	Cricket	Deactive
	66	Charger	Deactive
	68	Asus laptop	Active
	75	Kokine	Active
	80	2 Shirts	Active
	83	Iphone 14 pro	Active
	99	Chicken Noodles	Deactive
	100	Fanta 10L	Deactive
	345	5 Shorts	Active
	456	Drinks	Deactive
	789	Apples	Active

Products

Product Code

Product Name

Status

Add

Delete

Reset

	Product Code	Product Name	Status
▶	56	Cricket Bat	Active
	66	Charger	Deactive
	68	Asus laptop	Active
	75	Kokine	Active
	80	2 Shirts	Active
	83	Iphone 14 pro	Active
	99	Chicken Noodles	Deactive
	100	Fanta 10L	Deactive
	345	5 Shorts	Active
	456	Drinks	Deactive
	789	Apples	Active

## Button : Delete

The 'Products' window displays a table with the following data:

Product Code	Product Name	Status
56	Cricket Bat	Active
68	Asus laptop	Active
75	Kokine	Active
80	2 Shirts	Active
83	Iphone 14 pro	Active
99	Chicken Noodles	Deactive
100	Fanta 10L	Deactive
345	5 Shorts	Active
456	Drinks	Deactive
789	Apples	Active

Buttons: Add, Delete, Reset

## Stock Form :

### Button : Add

The 'Stock' window displays a table with the following data:

S.No	Product Code	Product Name	Quantity	Date	Status
1	12	Oil 34L	2	05-11-2022	Active
2	66	Charger	3	05-11-2022	Active
3	67	oil 1L	45	05-11-2022	Deactive
4	68	Asus laptop	110	05-11-2022	Active
5	83	Iphone 14 pro	2	13-11-2022	Active
6	234	Dolo 64	20	10-11-2022	Deactive
7	420	Room Spray	520	10-11-2022	Active
8	569	lenovo i7	58	05-11-2022	Deactive
9	898	laptop	2	05-11-2022	Active

Buttons: Add, Delete, Reset

Total Products : 9      Total Quantity : 762

Stock

Date

15-11-2022

Product Code

Product Name

Quantity

Status

Add

Delete

Reset

S.No	Product Code	Product Name	Quantity	Date	Status
1	12	Oil 34L	2	05-11-2022	Active
2	56	Cricket Bat	10	15-11-2022	Active
3	66	Charger	3	05-11-2022	Active
4	67	oil 1L	45	05-11-2022	Deactive
5	68	Asus laptop	110	05-11-2022	Active
6	83	Iphone 14 pro	2	13-11-2022	Active
7	234	Dolo 64	20	10-11-2022	Deactive
8	420	Room Spray	520	10-11-2022	Active
9	569	lenovo i7	58	05-11-2022	Deactive
10	898	laptop	2	05-11-2022	Active

Total Products : 10

Total Quantity : 772

## Button : Update

Stock

Date

15-11-2022

Product Code

56

Product Name

Cricket Bat

Quantity

10

Status

Deactive

Update

Delete

Reset

S.No	Product Code	Product Name	Quantity	Date	Status
1	12	Oil 34L	2	05-11-2022	Active
2	56	Cricket Bat	10	15-11-2022	Active
3	66	Charger	3	05-11-2022	Active
4	67	oil 1L	45	05-11-2022	Deactive
5	68	Asus laptop	110	05-11-2022	Active
6	83	Iphone 14 pro	2	13-11-2022	Active
7	234	Dolo 64	20	10-11-2022	Deactive
8	420	Room Spray	520	10-11-2022	Active
9	569	lenovo i7	58	05-11-2022	Deactive
10	898	laptop	2	05-11-2022	Active

Total Products : 10

Total Quantity : 772



Stock

Date

Product Code

Product Name

Quantity

Status

13-11-2022

Add

Delete

Reset

S.No	Product Code	Product Name	Quantity	Date	Status
1	12	Oil 34L	2	05-11-2022	Active
2	56	Cricket Bat	10	15-11-2022	Deactive
3	66	Charger	3	05-11-2022	Active
4	67	oil 1L	45	05-11-2022	Deactive
5	68	Asus laptop	110	05-11-2022	Active
6	83	Iphone 14 pro	2	13-11-2022	Active
7	234	Dolo 64	20	10-11-2022	Deactive
8	420	Room Spray	520	10-11-2022	Active
9	569	lenovo i7	58	05-11-2022	Deactive
10	898	laptop	2	05-11-2022	Active

Total Products : 10

Total Quantity : 772

## Button : Delete

Stock

Date

Product Code

Product Name

Quantity

Status

13-11-2022

Add

Delete

Reset

S.No	Product Code	Product Name	Quantity	Date	Status
1	12	Oil 34L	2	05-11-2022	Active
2	56	Cricket Bat	10	15-11-2022	Deactive
3	66	Charger	3	05-11-2022	Active
4	67	oil 1L	45	05-11-2022	Deactive
5	68	Asus laptop	110	05-11-2022	Active
6	234	Dolo 64	20	10-11-2022	Deactive
7	420	Room Spray	520	10-11-2022	Active
8	569	lenovo i7	58	05-11-2022	Deactive
9	898	laptop	2	05-11-2022	Active

Total Products : 9

Total Quantity : 770

## Product List Report :

ProductsReport

SAP CRYSTAL REPORTS

Main Report

**Product Report**

Printed Date : 13-11-2022

Serial Number	Product Code	Product Name
1	56	Cricket Bat
2	68	Asus laptop
3	75	Kokine

Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

## Stock List Report :

From : 01 November 2022 To : 13 November 2022 Show Report

SAP CRYSTAL REPORTS

Main Report

**Stock Report**

Stock Report From : 01/11/2022 To : 13/11/2022 Printed Date : 13/11/2022

S.No	Product Code	Product Name	Quantity
1	12	Oil 34L	2.00
2	66	Charger	3.00
3	67	oil 1L	45.00
4	68	Asus laptop	110.00

Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

**END OF TASK**