**Steps of Creating ContactsManager Using WindowsForms:**

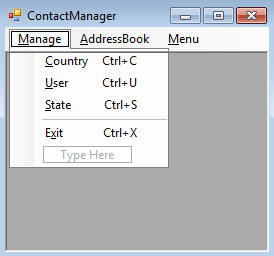
1. Manage Project
   1. MDI Form
   2. ConnectionString in Config File.
   3. Helper Class
   4. Enum
2. Manage Country
   1. Create Tables
   2. Create Stored Procedures
   3. Data
   4. DB
   5. BO
   6. Dialog
   7. Form
   8. Link to MDI
3. Manage State
   1. Create Tables
   2. Create Stored Procedures
   3. Data
   4. DB
   5. BO
   6. Dialog
   7. Form
   8. Link to MDI
4. Manage User
   1. Create Tables
   2. Create Stored Procedures
   3. Data
   4. DB
   5. BO
   6. Dialog
   7. Form
   8. Link to MDI
5. Address Book Form
   1. Create Tables
   2. Create Stored Procedures
   3. Data
   4. DB
   5. BO
   6. Dialog
   7. Form
   8. Link to MDI
6. Login Form / Authentication
   1. Create SP
   2. Form
   3. Edit Program.CS

**Setting up the Project**

Should be a MDI Based Application.

Through Menu Admin should be able to manage Users, Countries and States.

1. Go to Visual Studio  File  New project Select Windows Forms Application Name = ContactManager  OK
2. Right click on project  Add Windows FormsName: MDIParentForm OK
3. Go to MDIParent Form Properties and Change the Text as ContactManager
4. Select the MDI Form and go to Properties and change **IsMdiContainer** = True (In property window)
5. Go to →Tool Box → Add MenuStrip Control and Create Menu as shown below



1. Name of Menu Items, mnuManage, mnuManageUser, mnuManageCountry, mnuManageState, mnuExit, mnuAddressBook
2. click Menu → Add Menu items, Title Horx, Title Vertex, Cascading

double click on Title horx, Title vertex, cascading and write the following code

|  |
| --- |
| private void titleHorxToolStripMenuItem\_Click(object sender, EventArgs e)  {  this.LayoutMdi(MdiLayout.TileHorizontal);  }  private void titleVertexToolStripMenuItem\_Click(object sender, EventArgs e)  {  this.LayoutMdi(MdiLayout.TileVertical);  }  private void cascadeToolStripMenuItem\_Click(object sender, EventArgs e)  {  this.LayoutMdi(MdiLayout.Cascade);  } |

1. For ShortCutKeys on Country Menu, Properties, shortcutKeys→ Modifiers→Check-->Ctr and Key-->C
2. On User Menu, Properties, shortcutkeys→ Modifiers→ Check→Ctr and Key-->U
3. On State Menu, Properties, shortcutKeys→ Modifiers→Check→Ctr and Key-->S
4. On Exit Menu, Properties, ShortcutKeys→Modifiers→Check→Ctr and Key-->X
5. Edit Main method in Program.cs as below

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

MDIParentForm parent = newMDIParentForm();

Application.Run(parent);

}

1. Creation of Database:
   1. Server Explorer  Data Connections  Right Click  Create New SQL Server Database…
   2. Server Name = .\SqlExpress, Select Use Windows Authentication, New database name: AddressBook
2. Add the following to App.Config, under <configuration>

<connectionStrings>

<add name="ContactManagerDB" connectionString="Data Source=LAKSHMI-

PC\SQLEXPRESS;Initial Catalog=Contactsmanager;Integrated Security=True" />

</connectionStrings>

1. Go to Project →Right click→Add→Add a New class, Name = Helper.cs

Public class Helper

{

public static string ConnectionString

{

get

{

return System.Configuration.ConfigurationManager.ConnectionStrings["ContactManagerDB"].ConnectionString;

}

}

}

1. To easily track, Add/Edit/Delete Actions add the enum to Helper.cs

publicenumActionType

{

Add = 1,

Update,

Delete

}

public static string Role;

public static int UserId;

public enum ActiveStatus

{

All = -1,

Active = -2,

Inactive = -3

}

**Manage Country**

Manage Country represents All the details of Countries, Admin can see all the countries and Insert, Update, Delete Countries.

User will get All the Country Details in Dropdownlist while adding Addresses.

Zipcode should be 6 digits, Zipcode Start should not greater than Zipcode End.

1. **Create Country Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data type** | **Constraints** |
| PKCountryId | Int | PK, Not Null |
| CountryName | varchar(50) | Not null |
| ZipCodeStart | bigint | Not null |
| ZipCodeEnd | Bigint | Not null |
| IsActive | bit | Not null |

1. Create the following Stored Procedure to retrieve one or all countries based on @CountryId

CREATE PROCEDURE [dbo].[spGetAllCountries](@CountryId int) as

BEGIN

if @CountryId=-1

Select \* from Country

else if @CountryId=-2

Select \* from Country where IsActive=1

else if @CountryId=-2

Select \* from Country where IsActive=0

else

select \* from Country where PKCountryId = @CountryId

END

1. Create the following Stored Procedure to Add/Edit/Delete Country

CREATE PROCEDURE [dbo].[spManageCountry](@ActionType tinyint,@PKCountryId int output, @CountryName varchar(50), @ZipCodeStart int, @ZipCodeEnd int, @IsActive bit) as

BEGIN

if(@ActionType =1)

begin

insert into Country(CountryName,ZipCodeStart,ZipCodeEnd,IsActive)values(@CountryName,@ZipCodeStart,@ZipCodeEnd,@IsActive)

select @PKCountryId=@@Identity

End

else if(@ActionType =2)

update Country set

CountryName = @CountryName, ZipCodeStart=@ZipCodeStart,ZipCodeEnd=@ZipCodeEnd,IsActive=@IsActive where PKCountryId=@PKCountryId

else if(@ActionType=3)

delete from Country where PKCountryId=@PKCountryId

END

Go to Project→ Right Click→Click on Add →Add New Folder→Name: Data

Go to→ Data Folder →Right click→ Add→Select Class→ Give Like Name: Country→ Ok

1. Write the following code in Country Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ContactsManager.Data

{

public class Country

{

private string \_Validation = "";

public void Validate()

{

if (!String.IsNullOrEmpty(\_Validation))

{

throw new ApplicationException(\_Validation);

}

}

public Country()

{

}

public Country(int countryId, string countryName, long zipCodeStart, long zipcodeEnd, bool isActive)

{

\_CountryId = countryId;

this.CountryName = countryName;

this.ZipCodeStart = zipCodeStart;

this.ZipCodeEnd = zipcodeEnd;

this.IsActive = isActive;

}

private int \_CountryId;

public int CountryId

{

get

{

return \_CountryId;

}

}

private string \_CountryName;

public string CountryName

{

get

{

return \_CountryName;

}

set

{

if (value == "")

\_Validation += "Please provide a Value for CountryName" + "\t";

\_CountryName = value;

}

}

private long \_ZipCodeStart;

public long ZipCodeStart

{

get

{

return \_ZipCodeStart;

}

set

{

if (value <= 0)

\_Validation += "Please provide a valid ZipCode" + "\t";

\_ZipCodeStart = value;

}

}

private long \_ZipCodeEnd;

public long ZipCodeEnd

{

get

{

return \_ZipCodeEnd;

}

set

{

if (value <= 0)

\_Validation += "Please provide a valid ZipCode" + "\t";

\_ZipCodeEnd = value;

}

}

private bool \_IsActive;

public bool IsActive

{

get

{

return \_IsActive;

}

set

{

\_IsActive = value;

}

}

}

}

Go to Solution Explorer→Project→ RightClick on Project→Click on Add →Add New Folder→Name : DB

Go to→ DB Folder →Right click→ Add→Select Class→ Give Like Name: CountryDB → Ok

1. Write the following code in CountryDB class

using ContactsManager.Data;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ContactsManager.DB

{

public class CountryDB

{

public static void ManageCountry(Country objCountry, ActionType action)

{

string spName = "spManageCountry";

SqlParameter pActionType = new SqlParameter("@ActionType", SqlDbType.TinyInt);

SqlParameter pCountryId = new SqlParameter("@PKCountryId", SqlDbType.Int);

SqlParameter pCountryName = new SqlParameter("@CountryName", SqlDbType.VarChar, 50);

SqlParameter pZipCodeStart = new SqlParameter("@ZipCodeStart", SqlDbType.BigInt);

SqlParameter pZipCodeEnd = new SqlParameter("@ZipCodeEnd", SqlDbType.BigInt);

SqlParameter pIsActive = new SqlParameter("@IsActive", SqlDbType.Bit);

pActionType.Value = action;

if (action == ActionType.Add)

pCountryId.Direction = ParameterDirection.Input;

else

pCountryId.Value = objCountry.CountryId;

if (action == ActionType.Add || action == ActionType.Update)

{

pCountryName.Value = objCountry.CountryName;

pZipCodeStart.Value = objCountry.ZipCodeStart;

pZipCodeEnd.Value = objCountry.ZipCodeEnd;

pIsActive.Value = objCountry.IsActive;

}

SqlHelper.ExecuteNonQuery(Helper.ConnectionString, CommandType.StoredProcedure, spName, pActionType, pCountryId, pCountryName, pZipCodeStart, pZipCodeEnd, pIsActive);

}

public static DataSet GetCountries(int countryId)

{

string spCountryName = "spGetAllCountries";

SqlParameter pCountryId = new SqlParameter("@CountryId", SqlDbType.Int);

pCountryId.Value = countryId;

DataSet ds = SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spCountryName, pCountryId);

ds.Tables[0].TableName = "Country";

return ds;

}

public static Country GetCountry(int countryId)

{

string spName = "spGetAllCountries";

SqlParameter pCountryId = new SqlParameter("@CountryId", SqlDbType.Int);

pCountryId.Value = countryId;

SqlDataReader drRow = SqlHelper.ExecuteReader(Helper.ConnectionString, CommandType.StoredProcedure, spName, pCountryId);

Country objCountry = null;

if (drRow.HasRows)

{

drRow.Read();

objCountry = new Country(drRow.GetInt32(drRow.GetOrdinal("PKCountryId")), drRow["CountryName"].ToString(), Convert.ToInt32(drRow["ZipCodeStart"]), Convert.ToInt32(drRow["ZipCodeEnd"]), (bool)drRow["IsActive"]);

}

drRow.Close();

return objCountry;

}

}

}

Go to Solution Explorer→Project→ RightClick→Click on Add →Add New Folder→Name: BO

Go to→ BO Folder →Right click→ Click on Add→Select Class→ Give Like Name: CountryBO → Ok

1. Write the following code in CountryBO Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Data;

using System.Data.SqlClient;

using ContactsManager.Data;

public class CountryBO

{

public void InsertCountry(Country objCountry)

{

objCountry.Validate();

CountryDB.ManageCountry(objCountry, ActionType.Add);

}

public void InsertCountry(string CountryName, long ZipCodeStart, long ZipCodeEnd, bool IsActive)

{

try

{

Country objCountry = new Country(-1, CountryName, ZipCodeStart, ZipCodeEnd, IsActive);

objCountry.Validate();

CountryDB.ManageCountry(objCountry, ActionType.Add);

}

catch(System.Data.SqlClient.SqlException)

{

throw;

}

}

public void UpdateCountry(Country objCountry)

{

objCountry.Validate();

CountryDB.ManageCountry(objCountry, ActionType.Update);

}

public void UpdateCountry(int countyId, string countryName, int zipCodeStart, int zipCodeEnd, bool isActive)

{

try

{

Country objCountry = new Country(countyId, countryName, zipCodeStart, zipCodeEnd, isActive);

objCountry.Validate();

CountryDB.ManageCountry(objCountry, ActionType.Update);

}

catch(System.Data.SqlClient.SqlException)

{

throw;

}

}

public void DeletCountry(int countryId)

{

Country objCountry = new Country(countryId, "", 0, 0, false);

CountryDB.ManageCountry(objCountry, ActionType.Delete);

}

public DataSet GetCountries()

{

return CountryDB.GetCountries((int)ActiveStatus.ALL);

}

public DataSet GetActiveCountries()

{

return CountryDB.GelCountries((int)ActiveStatus.Active);

}

public Country GetCountry(int countryId)

{

Country objCountry = new Country();

objCountry = CountryDB.GetCoundtry(countryId);

return objCountry;

}

}

1. Go to Solution Explorer→Project→ Right click→ click on Add→click on New Item→Select a New Windows Forms, give Name = CountryDialog
2. Right click on CountryDialog form → Select properties → Set the following properties

FormBorderStyle = FixedDialog, MaximizeBox = false, MinimizeBox = false, Show InTaskBar = False,

1. Design the dialog as below

a)Go to->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties- >Text=&Country Name

b)Go to->ToolBox->Select TextBox->Drag and Drop on the Form-> Select TextBox on the Form->Right Click- >Properties->Name=txtCountryName

c)Go to->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties- >Text=Zipcode &Start.

d)Go to->ToolBox->Select TextBox->Drag and Drop on the Form-> Select TextBox on the Form->Right Click- >Properties->Name=txtZipCodeStart.

e)Go to->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties- >Text=Zipcode &End.

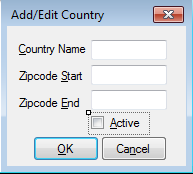
f)Go to->ToolBox->Select TextBox->Drag and Drop on the Form-> Select TextBox on the Form->Right Click- >Properties->Name=txtZipCodeEnd

g)Goto->ToolBox->Select CheckBox->Drag and Drop on the Form->Select CheckBox on the Form->Right Click- >Properties->Name=chkIsActive; Text=&Active

h)Goto->ToolBox->Select Button->Drag and Drop on the Form->Select the Button on the Form->Right Click- >Properties->Name=btnOK; Text=&OK->Double click on button to generate btnOK\_Click Event

i)Goto->ToolBox->Select Button->Drag and Drop on the Form->Select the Button on the Form->Right Click- >Properties->Name=btnCancel; Text=&Cancel->Double click on button to generate btnCancel\_Click Event j)Goto->ToolBox->Select ErrorProvider->Drag and Drop on the Form-> Select the ErrorProvider->Right Click- >Properties->Name=epCountry

k) RightClick on Form->Properties->AcceptButton=btnOK; CancelButton=btnCancel





1. Add the following corresponding class(CountryDialog .cs)

using ContactsManager.BO;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace ContactsManager

{

public partial class CountryDialog : Form

{

CountryBO objCountryBO;

public CountryDialog()

{

InitializeComponent();

}

private int \_CountryId = 0;

public int CountryId

{

get

{

return \_CountryId;

}

set

{

if (!string.IsNullOrEmpty(value.ToString()))

{

\_CountryId = value;

}

else

{

\_CountryId = 0;

}

}

}

public String CountryName

{

get

{

return txtCountryName.Text;

}

set

{

txtCountryName.Text = value;

}

}

public long ZipCodeStart

{

get

{

return Convert.ToInt32(txtZipCodeStart.Text);

}

set

{

txtZipCodeStart.Text = value.ToString();

}

}

public long ZipCodeEnd

{

get

{

return Convert.ToInt32(txtZipCodeEnd.Text);

}

set

{

txtZipCodeEnd.Text = value.ToString();

}

}

public bool IsActive

{

get

{

return chkIsActive.Checked;

}

set

{

chkIsActive.Checked = value;

}

}

private void btnOK\_Click(object sender, EventArgs e)

{

//Checking validations

bool errorflag = false;

epCountry.Clear();

if (txtCountryName.Text.Trim() == "")

{

errorflag = true;

epCountry.SetError(txtCountryName, "Please provide Country Name");

}

if (txtZipCodeStart.Text.Trim() == "")

{

errorflag = true;

epCountry.SetError(txtZipCodeStart, "Please provide Zip code Start Value");

}

if (txtZipCodeEnd.Text.Trim() == "")

{

errorflag = true;

epCountry.SetError(txtZipCodeEnd, "Please provide Zipcode End Value");

}

if (txtZipCodeEnd.Text.Trim() != "" && txtZipCodeStart.Text.Trim() != "")

{

if (Convert.ToInt64(txtZipCodeEnd.Text.Trim()) < Convert.ToInt64(txtZipCodeStart.Text.Trim()))

{

errorflag = true;

MessageBox.Show("Please enter a valid Zipcode range");

}

}

if (errorflag == false)

{

this.DialogResult = System.Windows.Forms.DialogResult.OK

}

}

private void txtZipCodeStart\_KeyPress(object sender, KeyPressEventArgs e)

{

int ascii = (int)e.KeyChar;

//For BackSpace

if (ascii == 8)

return;

//To Enter Only Digits

if (ascii < 48 || ascii > 57)

{

e.Handled = true;

}

}

private void txtZipCodeEnd\_KeyPress(object sender, KeyPressEventArgs e)

{

//For BackSpace

int ascii = (int)e.KeyChar;

if (ascii == 8)

return;

//To Enter Only Digits

if (ascii < 48 || ascii > 57)

{

e.Handled = true;

}

}

private void btnCancel\_Click\_1(object sender, EventArgs e)

{

this.DialogResult = System.Windows.Forms.DialogResult.Cancel;

}

}

}

Go to Solution Explorer→Project→Right click on project → click on Add → click on New Item → Select a New Windows Forms, Give Name: CountryForm

 Go to Toolbox →Drag and drop The Data Grid To the Form, and give the Name: dgvCountry

Edit the dgvCountry as below

dgvCountry→Add Columns→

1. Name:CountryID,Type:DataGridViewTextBoxColumn,HeaderText:CountryId,ReadOnly:checked,

Visible:Unchecked

1. Name:CountryName,Type:DataGridViewTextBoxColumn,HeaderText:CountryName,Visible:checked
2. Name:ZipcodeStart,Type:DataGridViewTextBoxColumn,HeaderText:ZipcodeStart,Visible:checked
3. Name:ZipcodeEnd,Type:DataGridViewTextBoxColumn,HeaderText:ZipcodeEnd,Visible:checked
4. Name:IsActive,Type:DataGridViewCheckBoxColumn,HeaderText:IsActive,Visible:checked
5. Right click on dgvCountry → click on Properties → give Readonly=True, multyRow=False, Text=Manage Country

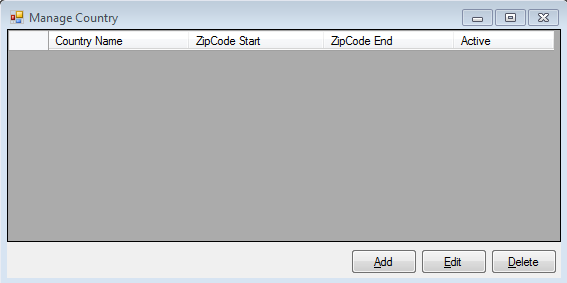
1. Design the Form as below

→ Go to Tool Box drag and drop button to CountryForm → Right click on Button → click on properties

give Text=Add, Name=btnOk

→ Go to Tool Box drag and drop button to CountryForm → Right click on Button → click on properties give Text=Edit, Name=btnEdit

→ Go to Tool Box drag and drop button to CountryForm → Right click on Button → click on properties give Text=Delete, Name=btnDelete



1. Add the following code to the Form class(CountryForm.cs)

public CountryForm()

{

InitializeComponent();

}

private void BindDataToGrid()

{

try{

//Bind all records into gridview

DataSet ds;

CountryBO objCountryBO = new CountryBO();

ds = objCountryBO.GetCountries();

dgvCountry.DataSource = ds;

dgvCountry.DataMember = "Country";

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("1.Check the Connection string ","Unable to connect database ");

}

}

private void ManageCountry\_Load(object sender, EventArgs e)

{

dgvCountry.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

dgvCountry.AutoGenerateColumns = false;

dgvCountry.AllowUserToAddRows = false;

dgvCountry.AllowUserToDeleteRows = false;

dgvCountry.ReadOnly = true;

dgvCountry.MultiSelect = false;

BindDataToGrid();

}

private void btnAdd\_Click(object sender, EventArgs e)

{

//Creating an object of CountryDialog

CountryDialog dlgCountry = new CountryDialog();

if (dlgCountry.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

try

{

Country objCountry = new Country();

CountryBO objCountryBO = new CountryBO();

objCountryBO.InsertCountry(dlgCountry.CountryName, dlgCountry.ZipCodeStart, dlgCountry.ZipCodeEnd, dlgCountry.IsActive);

BindDataToGrid();

}

catch (Exception ex)

{

MessageBox.Show("CountryName AlredyEntered");

}

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

//If user click on Delete button

if (dgvCountry.RowCount != 0)

{

DialogResult dr;

dr = MessageBox.Show("Are you sure?", "Delete", MessageBoxButtons.YesNo);

if (dr == System.Windows.Forms.DialogResult.No)

{

return;

}

//Create an object of Delete Dialog

int countryId = (int)dgvCountry.SelectedRows[0].Cells[0].Value;

CountryBO objCountryBO = new CountryBO();

objCountryBO.DeletCountry(countryId);

BindDataToGrid();

}

}

private void btnUpdateCountry\_Click(object sender, EventArgs e)

{

if (dgvCountry.RowCount != 0)

{

int countryId;

countryId = (int)(dgvCountry.SelectedRows[0].Cells[0].Value);

CountryDialog dlgCountry = new CountryDialog();

CountryBO objCountryBO = new CountryBO();

Country objCountry = objCountryBO.GetCountry(countryId);

dlgCountry.CountryId = objCountry.CountryId;

dlgCountry.CountryName = objCountry.CountryName;

dlgCountry.ZipCodeStart = Convert.ToInt64(objCountry.ZipCodeStart);

dlgCountry.ZipCodeEnd = Convert.ToInt64(objCountry.ZipCodeEnd);

dlgCountry.IsActive = objCountry.IsActive;

string country = objCountry.CountryName;

if (dlgCountry.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

objCountry.CountryName = dlgCountry.CountryName;

objCountry.ZipCodeStart = dlgCountry.ZipCodeStart;

objCountry.ZipCodeEnd = dlgCountry.ZipCodeEnd;

objCountry.IsActive = dlgCountry.IsActive;

objCountryBO.UpdateCountry(objCountry);

}

BindDataToGrid();

}

}

}

}

}

1. In MDIMainForm, Handle Click event of mnuCountry and write the following code.

CountryForm objCountry;

private void mnuManageCountry\_Click(object sender, EventArgs e)

{

if (objCountry == null)

{

objCountry = new CountryForm();

objCountry.Show();

objCountry.MdiParent = this;

objCountry.FormClosing += objCountry\_FormClosing;

}

else

{

objCountry.Activate();

}

void objCountry\_FormClosing(object sender, FormClosingEventArgs e)

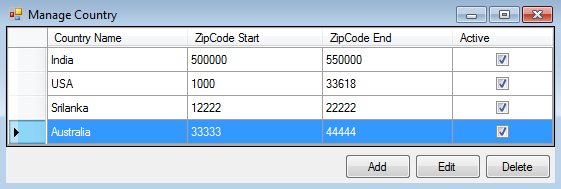
{

objCountry = null;

}  
 }

1. Run and perform Add/Edit/Delete Operations on Country.

=> show given below



Manage State

Admin can Manage States and he can perform Insert, Update, Delete operations.

Admin can Get All States and also based upon country name.

1. **Create State Table**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data type** | **Constraints** |
| PKStateId | **Int** | PK, Not null |
| FKCountryId | **Int** | FK, Not null |
| StateName | **Varchar(50)** | Varchar(50),Not null |
| IsActive | **Bit** | Not null |

1. Create the following Stored Procedure to retrieve one or all States based on @StateId

CREATE PROCEDURE [dbo].[spGetAllStates](@StateId int, @CountryId int = NULL)

AS

BEGIN

IF @StateId=-1

Select s.PKStateId as PKStateId, s.FKCountryId as FKCountryId, s.StateName as StateName, s.IsActive as IsActive, c.CountryName as CountryName

FROM State s JOIN Country c ON s.FKCountryId = c.PKCountryId where FKCountryID=ISNULL(@CountryId,FKCountryID)

ELSE IF @StateId=-2

SELECT s.PKStateId as PKStateId, s.FKCountryId as FKCountryId, s.StateName as StateName, s.IsActive as IsActive, c.CountryName as CountryName

FROM State s JOIN Country c

ON s.FKCountryId = c.PKCountryId where s.IsActive=1 and FKCountryID=ISNULL(@CountryId,FKCountryID)

ELSE IF @StateId=-3

SELECT s.PKStateId as PKStateId, s.FKCountryId as FKCountryId , s.StateName as StateName, s.IsActive as IsActive , c.CountryName as CountryName

FROM State s JOIN Country c

ON s.FKCountryId = c.PKCountryId where s.IsActive=0 and FKCountryID=ISNULL(@CountryId,FKCountryID)

ELSE

SELECT s.PKStateId as PKStateId, s.FKCountryId as FKCountryId , s.StateName as StateName, s.IsActive as IsActive , c.CountryName as CountryName

FROM State s JOIN Country c

ON s.FKCountryId = c.PKCountryId

WHERE PKStateId =@StateId

END

1. Create the following Stored Procedure to Add/Edit/Delete State

CREATE PROCEDURE [dbo].[spManageState](@ActionType tinyint, @PKStateId int output,

@FKCountryId int,@StateName varchar(50), @IsActive bit) as

if(@ActionType=1)

begin

insert into State(FKCountryId,StateName, IsActive)values(@FKCountryId,@StateName,

@IsActive)

select @PKStateId= @@Identity

End

else if(@ActionType=2)

Update State set

StateName=@StateName, IsActive=@IsActive,FKCountryId=@FKCountryId where

PKStateId=@PKStateId

else if(@ActionType =3)

delete from State where PKStateId = @PKStateId

1. Create the following Stored Procedure to Check if the Country is already existing or not

CREATEPROCEDURE [dbo].[spIsStateExist](@StateName asvarchar(50))

AS

BEGIN

Select count(\*) from Country c join State s on c.PKCountryId=s.FKCountryId where s.StateName=@StateName;

END

=>Go to Data folder → Right click → click on Add → select class → Give the name= State

1. Write the following code in State class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ContactsManager.Data

{

public class State

{

private string \_Validation = "";

public void Validate()

{

if (!string.IsNullOrEmpty(\_Validation))

{

throw new ApplicationException(\_Validation);

}

}

public State()

{ }

public State(int stateId, int countryId, string stateName, bool isActive)

{

//Assign all parameters for checking properties

\_StateId = stateId;

this.CountryId = countryId;

this.StateName = stateName;

this.IsActive = isActive;

}

private int \_StateId;

public int StateId

{

get

{

return \_StateId;

}

}

private int \_CountryId;

public int CountryId

{

get

{

return \_CountryId;

}

set

{

\_CountryId = value;

}

}

private string \_StateName;

public string StateName

{

get

{

return \_StateName;

}

set

{

if (value == "")

\_Validation += "Please provide a value for StateName" + "\t";

\_StateName = value;

}

}

private bool \_IsActive;

public bool IsActive

{

get

{

return \_IsActive;

}

set

{

\_IsActive = value;

}

}

}

}

=> Go to DB Folder → Right click → click on Add → click on class → Give Name:StateDB

1. Write the following code in StateDB class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using ContactsManager.Data;

using System.Data.SqlClient;

using System.Data;

public class StateDB

{

public StateDB()

{

}

/// To perform CRUD operations on State

public static void ManageState(State objState, ActionType action)

{

try

{

string spName = "spManageState";

SqlParameter pActionType = new SqlParameter("@ActionType", SqlDbType.TinyInt);

SqlParameter pStateId = new SqlParameter("@PKStateId", SqlDbType.Int);

SqlParameter pCountryId = new SqlParameter("@FKCountryId", SqlDbType.Int);

SqlParameter pStateName = new SqlParameter("@StateName", SqlDbType.VarChar, 50);

SqlParameter pIsActive = new SqlParameter("@IsActive", SqlDbType.Bit);

pActionType.Value = action;

if (action == ActionType.Add)

pStateId.Direction = ParameterDirection.Input;

else

pStateId.Value = objState.StateId;

if (action == ActionType.Add || action == ActionType.Update)

{

pCountryId.Value = objState.CountryId;

pStateName.Value = objState.StateName;

pIsActive.Value = objState.IsActive;

}

SqlHelper.ExecuteNonQuery(Helper.ConnectionString, CommandType.StoredProcedure, spName, pActionType, pStateId, pCountryId, pStateName, pIsActive);

}

catch (Exception ex)

{

throw ex;

}

}

/// To fetch all the states

public static DataSet GetStates(int stateId, int countryId=0 )

{

try

{

string spName = "spGetAllStates";

SqlParameter pStateId = new SqlParameter("@StateId", SqlDbType.Int);

SqlParameter pCountryId = new SqlParameter("@CountryId", SqlDbType.Int);

pStateId.Value = stateId;

if (countryId>0)

pCountryId.Value = countryId;

else

pCountryId.Value = DBNull.Value;

DataSet ds = SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spName, pStateId, pCountryId);

return ds;

}

catch (Exception ex)

{

throw ex;

}

}

public static State GetState(int stateId)

{

try

{

string spName = "spGetAllStates";

SqlParameter pStateId = new SqlParameter("@StateId", SqlDbType.Int);

pStateId.Value = stateId;

SqlDataReader drRow = SqlHelper.ExecuteReader(Helper.ConnectionString, CommandType.StoredProcedure, spName, pStateId);

State objState = null;

if (drRow.HasRows)

{

drRow.Read();

objState = new State(drRow.GetInt32(drRow.GetOrdinal("PKStateId")), drRow.GetInt32(drRow.GetOrdinal("FKCountryId")), drRow["StateName"].ToString(), Convert.ToBoolean(drRow["IsActive"]));

}

drRow.Close();

return objState;

}

catch (Exception ex)

{

throw ex;

}

}

/// To verify whether state exist in country

public static bool IsStateExist(string stateName)

{

try

{

string spName = "spIsStateExist";

SqlParameter pStateName = new SqlParameter("@StateName", SqlDbType.VarChar, 50);

SqlParameter pCountryName = new SqlParameter("@CountryName", SqlDbType.VarChar, 20);

pStateName.Value = StateName;

int count = Convert.ToInt32(SqlHelper.ExecuteScalar(Helper.ConnectionString, CommandType.StoredProcedure, spName, pStateName));

return count != 0;

}

catch (Exception ex)

{

throw ex;

}

}

}

=> Go to BO Folder → Right click → click on Add → click on class → Give Name: StateBO

1. Write the following code in StateBO class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using ContactsManager.Data;

using System.Data;

namespace ContactsManager.BO

{

public class StateBO

{

public StateBO()

{ }

/// Add a Single record to State Table

public void InsertState(State objState)

{

objState.Validate();

StateDB.ManageState(objState, ActionType.Add);

}

/// Add record to state table

public void InsertState(int countryId, String stateName, bool isActive)

{

try

{

State objState = new State(-1, countryId, stateName, isActive);

objState.Validate();

StateDB.ManageState(objState, ActionType.Add);

}

catch(System.Data.SqlClient.SqlException)

{

throw;

}

}

/// Fetch the records into gridview takes from State table

public DataSet GetStates(int countryId=0 )

{

return StateDB.GetStates((int)ActiveStatus.All, countryId);

}

public void UpdateState(int stateId, int countryId, String stateName, bool isActive)

{

State objState = new State(stateId, countryId, stateName, isActive);

objState.Validate();

StateDB.ManageState(objState, ActionType.Update);

}

/// Fetch the records from State table

public State GetState(int stateId)

{

return StateDB.GetState(stateId);

}

/// Delete records to State table

public void DeleteState(int stateId)

{

State objState = new State(stateId, 0, "", false);

StateDB.ManageState(objState, ActionType.Delete);

}

/// Bind the Countryname using countryid

public DataSet GetActiveStates(int countryId = 0)

{

return StateDB.GetStates((int)ActiveStatus.Active, countryId);

}

public bool IsStateExist(string stateName)

{

if (StateDB.IsStateExist( stateName))

throw new ApplicationException("state name already exists");

else

return StateDB.IsStateExist(stateName);

}

}

}

=> Go to → Project →Right click → Click on Add → Click on New Item → Select a New Windows Forms, Name = StateDialog

1. Set the following properties
2. FormBorderStyle = FixedDialog, MaximizeBox = false, MinimizeBox = false, ShowInTaskBar = False
3. Design the dialog as below

=> Go to tool Box → Drag and drop Label → give Text: Country Name

=> Go to tool Box → Drag and drop Combobox → Give Name: cmbCountryName

=> Go to tool Box → Drag and drop Label → give Text: State Name

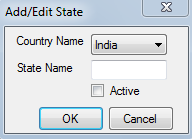
=> Go to tool Box → Drag and drop textBox → give Name: txtStateName

=> Go to tool Box → Drag and Drop CheckBox → give Name: chkIsActive, Text:Active

=> Go to tool Box → Drag and Drop Button → Give Text:Ok, Name:btnOk

=> Go to tool Box → Drag and Drop Button → Give Text:Cancel, Name:btnCancel

=> Go to Toll Box drag and drop Error Provider, Name=epstateDialog





1. Add the following corresponding StateDialog.cs Class

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.BO;

namespace ContactsManager

{

public partial class StateDialog : Form

{

public StateDialog()

{

InitializeComponent();

BindCountries();

}

public int CountryId

{

get

{

return (int)cmbCountryName.SelectedValue;

}

set

{

cmbCountryName.SelectedValue = value;

}

}

public string CountryName

{

get

{

return cmbCountryName.SelectedItem.ToString();

}

set

{

cmbCountryName.SelectedItem = value;

}

}

public string StateName

{

get

{

return txtStateName.Text;

}

set

{

txtStateName.Text = value;

}

}

public bool IsActive

{

get

{

return chkIsActive.Checked;

}

set

{

chkIsActive.Checked = value;

}

}

private void btnOK\_Click(object sender, EventArgs e)

{

//Checking Validations

bool errorflag = false;

epstateDialog.Clear();

if (txtStateName.Text.Trim() == "")

{

errorflag = true;

epstateDialog.SetError(txtStateName, "Please provide StateName");

}

if (errorflag == false)

{

this.DialogResult = System.Windows.Forms.DialogResult.OK;

}

}

CountryId =(int)cmbCountryName.SelectedValue;

StateName = txtStateName.Text;

CountryName = cmbCountryName.Text;

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.DialogResult = System.Windows.Forms.DialogResult.Cancel;

}

private void StateDialogForm\_Load(object sender, EventArgs e)

{

cmbCountryName.SelectedValue = CountryId;

txtStateName.Text = StateName;

chkIsActive.Checked = IsActive;

}

private void BindCountries()

{

CountryBO objCountryBO=new CountryBO();

//Bind Country names using object

cmbCountryName.DisplayMember = "CountryName";

cmbCountryName.ValueMember = "PKCountryId";

DataSet ds =objCountryBO.GetCountries();

cmbCountryName.DataSource = ds.Tables[0];

}

}

}

=> Go to Project → Right click → Click on Add → Click on New Item → Select a New Windows Form to the Project Name = StateForm

1. Design the Form as below

dgvState-->Add Columns-->

1. Name:StateID,Type:DataGridViewTextBoxColumn,HeaderText:CountryId,ReadOnly:checked,

Visible:Unchecked.

1. Name:CountryName,Type:DataGridViewTextBoxColumn,HeaderText:CountryName,Visible:checked
2. Name:StateName,Type:DataGridViewTextBoxColumn,HeaderText:StateName,Visible:checked
3. Name:IsActive,Type:DataGridViewCheckBoxColumn,HeaderText:IsActive,Visible:checked
4. Go to dgvState Properties give Readonly=True,multyRow=False

→ Go to Tool Box drag and drop button to StateForm → Right click on Button → click on properties

give Text=Add, Name=btnOk

→ Go to Tool Box drag and drop button to StateForm → Right click on Button → click on properties give Text=Edit, Name=btnEdit

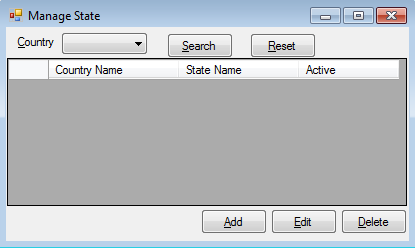
→ Go to Tool Box drag and drop button to StateForm → Right click on Button → click on properties give Text=Delete, Name=btnDelete

→ Go to Tool Box drag and drop button to StateForm → Right click on Button → click on properties give Text=Search, Name=btnSearch

→ Go to Tool Box drag and drop button to StateForm → Right click on Button → click on properties give Text=Reset, Name=btnReset

→ Go to tool Box → Drag and drop Combobox → Give Name: cmbCountryName

→ Go to tool Box → Drag and drop Label → give Text: Country



1. Add the following code to the Form class(StateForm.cs)

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.BO;

using ContactsManager.Data;

namespace ContactsManager

{

public partial class StateForm : Form

{

public ManageStateForm()

{

InitializeComponent();

}

private void btnAdd\_Click(object sender, EventArgs e)

{

try

{

//Creating an object of State Dialog

StateDialog dlgState = new StateDialog();

if (dlgState.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

//if User click on OK button of dialog

State objState = new State();

StateBO objStateBO = new StateBO();

//IChecking the state is Exist or not

objStateBO.IsStateExist(dlgState.StateName)

objStateBO.InsertState(dlgState.CountryId, dlgState.StateName, dlgState.IsActive);

//Bind the records into grid

BindDataToGrid();

}

}

catch (ApplicationException ex)

{

MessageBox.Show(ex.Message);

}

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

//If user click on Edit button

if (dgvState.RowCount != 0)

{

int stateId;

stateId = (int)(dgState.SelectedRows[0].Cells[0].Value);

StateDialog dlgState = new StateDialog();

StateBO objStateBO = new StateBO();

//update the records

State objState = objStateBO.GetState(stateId);

dlgState.CountryName = objState.FKCountryId.ToString();

dlgState.StateName = objState.StateName;

dlgState.IsActive = objState.IsActive;

string state = objState.StateName;

if (dlgState.ShowDialog() == DialogResult.OK)

{

try{

if(!(objState.StateName==dlgState.StateName))

objStateBo.IsStateExist(dlgState.StateName);

objStateBO.UpdateState(objState.StateId,objState.CountryId, dlgState.StateName, dlgState.IsActive);

}

catch(Exception ex)

{

MessageBox.Show(EX.Message);

}

BindGridView();

}

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

//If Userclick on Delete button

if (dgvState.RowCount != 0)

{

DialogResult dr;

dr = MessageBox.Show("Are you sure?If you delete state corresponding addresses will get deleted", "Delete", MessageBoxButtons.YesNo);

if (dr == System.Windows.Forms.DialogResult.No)

{

return;

}

//Create an object of Delete dialog

int StateId = (int)dgState.SelectedRows[0].Cells[0].Value;

StateBO objStateBO = new StateBO();

objStateBO.DeleteState(StateId);

//Delete records from grid

BindDataToGrid();

}

}

private void ManageState\_Load(object sender, EventArgs e)

{

dgvState.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

dgvState.AutoGenerateColumns = false;

dgvState.AllowUserToAddRows = false;

dgvState.AllowUserToDeleteRows = false;

dgvState.ReadOnly = true;

dgvState.MultiSelect = false;

BindDataToGrid();

BindCountries();

}

private void BindCountries()

{

CountryBO objCountryBO=new CountryBO();

//Bind Country names using object

cmbCountryName.DisplayMember = "CountryName";

cmbCountryName.ValueMember = "PKCountryId";

DataSet ds =objCountryBO.GetCountries();

cmbCountryName.DataSource = ds.Tables[0];

**}**

private void BindDataToGrid()

{

try

{

//Bind all records into gridview

DataSet ds;

StateBO objStateBO = new StateBO();

ds = objStateBO.GetStates();

dgState.DataSource = ds.Tables[0];

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("1.Check the Connection string ","Unable to connect database ");

}

}

public void BindFilteredData()

{

DataSet ds = objStateBo.GetActiveStates(Convert.ToInt32(cmbCountry.SelectedValue.ToString()));

dgvState.DataSource = ds.Tables[0];

}

private void btnSearch\_Click(object sender, EventArgs e)

{

BindFilteredData();

}

}

}

1. In MDIMainForm, Handle Click event of mnuState Write the following code

StateForm objState;

private void mnuManageState\_Click(object sender, EventArgs e)

{

if (objState == null)

{

objState = new ManageStateForm();

objState.Show();

objState.MdiParent = this;

objState.FormClosing += objState\_FormClosing;

}

else

{

objState.Activate();

}

}

void objState\_FormClosing(object sender, FormClosingEventArgs e)

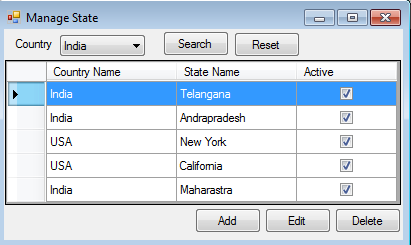
{

objState = null;

}

1. Run and perform Add/Edit/Delete Operations on State.

=> show the give below



**Manage User**

Admin can Manage UserDetails and can perform Update, Delete Operations.

Admin can also Insert the user Details.

1. **Create UserDetails Table**

|  |  |  |
| --- | --- | --- |
| Column Name | Data type | Constraints |
| PKUserId | Int | PK, Not null |
| UserName | Varchar(50) | Not null |
| Password | Varchar(50) | Not null |
| FirstName | Varchar(50) | Not null |
| LastName | Varchar(50) | Allow nulls |
| EmailId | Varchar(50) | Not null |
| PhoneNo | Varchar(50) | Not null |
| IsActive | Bit | Allow null |

1. Create the following Stored Procedure to retrieve one or all Users based on @UserId

CREATE PROCEDURE [dbo].[spGetAllUsers](@UserId int)as

if @UserId=-1

Select \*from UserDetails

else if @UserId = -2

Select \* from UserDetails where IsActive=1

else if @UserId = -3

Select \* from UserDetails where IsActive=0

else

Select \*from UserDetails where PKUserId = @UserId

1. Create the following Stored Procedure to Add/Edit/Delete User

ALTER PROCEDURE [dbo].[spManageUserDetails](@ActionType tinyint, @PKUserId int output, @UserName varchar(50), @Password varchar(50), @FirstName varchar(50),@LastName varchar(50), @EmailId varchar(50),@PhoneNo varchar(50), @IsActive bit)as

if(@ActionType=1)

begin

insert into UserDetails(UserName,Password,FirstName,LastName,EmailId,PhoneNo,IsActive)

values(@UserName,@Password,@FirstName,@LastName,@EmailId,@PhoneNo,@IsActive)

select @PKUserId = @@Identity

End

else if(@ActionType=2)

update UserDetails set UserName=@UserName,Password=@Password,FirstName=@FirstName,LastName=@LastName,EmailId=@EmailId,PhoneNo=@PhoneNo,IsActive=@IsActive

where PKUserId=@PKUserId

else if(@ActionType=3)

delete from UserDetails where PKUserId=@PKUserId

1. Create the following Stored Procedure to AuthenticateUser

CREATE PROCEDURE [dbo].[spAuthenticateUser](@UserName varchar(50),@Password varchar(50),@PKUserId int output)

AS

Begin

Select @PKUserId=u.PKUserId from UserDetails as u where u.UserName=@UserName and u.Password=@Password

End

Go to--> Data Folder -->Right click--> Add-->Select Class--> Give Like Name: User-->Ok

1. Add the following code in User class

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

namespace ContactsManager.Data

{

public class User

{

private string \_Validation = "";

public void Validate()

{

if (!String.IsNullOrEmpty(\_Validation))

{

throw new ApplicationException(\_Validation);

}

}

public User()

{ }

public User(int userId, string userName, string password, string firstName, string lastName, string emailId, string phoneNo, bool isActive)

{

//Assign all properties for checking properties

\_UserId = userId;

this.UserName = userName;

this.Password = password;

this.FirstName = firstName;

this.LastName = lastName;

this.EmailId = emailId;

this.PhoneNo = phoneNo;

this.IsActive = isActive;

}

private int \_UserId;

public int UserId

{

get

{

return \_UserId;

}

}

private string \_UserName;

public string UserName

{

get

{

return \_UserName;

}

set

{

if (value == "")

\_Validation += "Please provide a value for UserName" + "\t";

\_UserName = value;

}

}

private string \_Password;

public string Password

{

get

{

return \_Password;

}

set

{

if (value == "")

\_Validation += "Please provide a value for Password" + "\t";

\_Password = value;

}

}

private string \_FirstName;

public string FirstName

{

get

{

return \_FirstName;

}

set

{

if (value == "")

\_Validation += "Please provide a value for FirstName" + "\t";

\_FirstName = value;

}

}

private string \_LastName;

public string LastName

{

get

{

return \_LastName;

}

set

{

\_LastName = value;

}

}

private string \_EmailId;

public string EmailId

{

get

{

return \_EmailId;

}

set

{

if (value == "")

\_Validation += "Please provide a value for EmailId" + "\t";

\_EmailId = value;

}

}

private string \_PhoneNo;

public string PhoneNo

{

get

{

return \_PhoneNo;

}

set

{

if (value == "")

\_Validation += "Please provide a value for Phone" + "\t";

\_PhoneNo = value;

}

}

private bool \_IsActive;

public bool IsActive

{

get

{

return \_IsActive;

}

set

{

\_IsActive = value;

}

}

}

}

Go to--> DB Folder -->Right click--> Add-->Select Class--> Give Like Name: UserDB-->Ok

1. Write the following code in the UserDB class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using ContactsManager.Data;

using System.Data.SqlClient;

using System.Data;using System.Threading.Tasks;

namespace ContactsManager.DB

{

public class UserDB

{

public static void ManageUser(User objUser, ActionType action)

{

try

{

string spName = "spManageUser";

SqlParameter pActionType = new SqlParameter("@ActionType", SqlDbType.TinyInt);

SqlParameter pPKUserId = new SqlParameter("@PKUserId", SqlDbType.Int);

SqlParameter pUserName = new SqlParameter("@UserName", SqlDbType.VarChar, 50);

SqlParameter pPassword = new SqlParameter("@Password", SqlDbType.VarChar, 50);

SqlParameter pFirstName = new SqlParameter("@FirstName", SqlDbType.VarChar, 50);

SqlParameter pLastName = new SqlParameter("@LastName", SqlDbType.VarChar, 50);

SqlParameter pEmailId = new SqlParameter("@EmailId", SqlDbType.VarChar, 50);

SqlParameter pPhoneNo = new SqlParameter("@PhoneNo", SqlDbType.VarChar, 50);

SqlParameter pIsActive = new SqlParameter("@IsActive", SqlDbType.Bit);

pActionType.Value = (int)action;

if (action == ActionType.Add)

pPKUserId.Direction = ParameterDirection.Input;

else

pPKUserId.Value = objUser.UserId;

if (action == ActionType.Add || action == ActionType.Update)

{

pUserName.Value = objUser.UserName;

pPassword.Value = objUser.Password;

pFirstName.Value = objUser.FirstName;

pLastName.Value = objUser.LastName;

pEmailId.Value = objUser.EmailId;

pPhoneNo.Value = objUser.PhoneNo;

pIsActive.Value = objUser.IsActive;

}

SqlHelper.ExecuteNonQuery(Helper.ConnectionString, CommandType.StoredProcedure, spName, pActionType, pPKUserId, pUserName, pPassword, pFirstName, pLastName, pEmailId, pPhoneNo, pIsActive);

}

catch (Exception ex)

{

throw ex;

}

}

public static DataSet GetUsers(int userId)

{

try

{

string spUserName = "spGetAllUsers";

SqlParameter pUserId = new SqlParameter("@UserId", SqlDbType.Int);

pUserId.Value = userId;

return SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spUserName, pUserId);

}

catch (Exception ex)

{

throw ex;

}

} /// <summary>

public static User GetUserId(int userId)

{

try

{

string spName = "spGetAllUsers";

SqlParameter pUserId = new SqlParameter("@UserId", SqlDbType.Int);

pUserId.Value = userId;

SqlDataReader drRow = SqlHelper.ExecuteReader(Helper.ConnectionString, CommandType.StoredProcedure, spName, pUserId);

User objUser = null;

if (drRow.HasRows)

{

drRow.Read();

objUser = new User(drRow.GetInt32(drRow.GetOrdinal("PKUserId")), drRow["UserName"].ToString(), drRow["Password"].ToString(), drRow["FirstName"].ToString(), drRow["LastName"].ToString(), drRow["EmailId"].ToString(), drRow["PhoneNo"].ToString(), (bool)drRow["IsActive"]);

}

drRow.Close();

return objUser;

}

catch (Exception ex)

{

throw ex;

}

}

/// Authenticate username and password to the user table

public static int AuthenticateUser(string userName, string password)

{

try

{

string spName = "spAuthenticateUser";

SqlParameter pPKUserId = new SqlParameter("@PKUserId", SqlDbType.Int);

SqlParameter pUserName = new SqlParameter("@UserName", SqlDbType.VarChar, 50);

SqlParameter pPassword = new SqlParameter("@Password", SqlDbType.VarChar, 50);

pPKUserId.Direction = ParameterDirection.Output;

pUserName.Value = userName;

pPassword.Value = password;

SqlHelper.ExecuteNonQuery(Helper.ConnectionString, CommandType.StoredProcedure, spName, pUserName, pPassword, pPKUserId);

if (pPKUserId.Value == DBNull.Value)

return 0

return Convert.ToInt32(pPKUserId.Value);

}

catch (Exception ex)

{

throw ex;

}

}

}

}

Go to-->BO Folder -->Right click--> Add-->Select Class--> Give Like Name: UserBO-->Ok

1. Write the Following code in the UserBo Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data.SqlClient;

using ContactsManager.DB;

using ContactsManager.Data;

using System.Data;

namespace ContactsManager.BO

{

public class UserBO

{

UserDB objUserDB = new UserDB();

public UserBO()

{ }

public void InsertUser(User objUser)

{

objUser.Validate();

UserDB.ManageUser(objUser, ActionType.Add);

}

public void InsertUser(string userName, string password, string firstName, string lastName, string emailId, string phoneNo, bool isActive)

{

User objUser = new User(-1, userName, password, firstName, lastName, emailId, phoneNo, isActive);

objUser.Validate();

if (IsUserExist(userName))

{

throw new ApplicationException("User already exists");

}

else

{

UserDB.ManageUser(objUser, ActionType.Add);

}

}

public DataSet GetUsers()

{

return UserDB.GetUsers((int)ActiveStatus.All);

}

public DataSet GetActiveUsers()

{

return UserDB.GetlUsers((int)ActiveStatus.Active);

}

public void UpdateUser(User objUser)

{

objUser.Validate();

UserDB.ManageUser(objUser, ActionType.Update);

}

public void UpdateUser(int PKUserId, string UserName, string Password, string FirstName, string LastName, string EmailId, string PhoneNo, bool IsActive)

{

User objUser = new User(PKUserId, UserName, Password, FirstName, LastName, EmailId, PhoneNo, IsActive);

objUser.Validate();

UserDB.ManageUser(objUser, ActionType.Update);

}

public void DeleteUser(int PKUserId)

{

User objUser = new User(PKUserId, "", "", "", "", "", "", false);

UserDB.ManageUser(objUser, ActionType.Delete);

}

public User GetUser(int UserId)

{

User objUser = new User();

objUser = UserDB.GetUserById(UserId);

return objUser;

}

public int AuthenticateUser(string userName, string password)

{

return UserDB.AuthenticateUser(userName, password);

}

}

}

=> Go to project → Right click → click on Add → click on New item → Select a New Windows Forms, Name = UserDialog

1. Set the following properties

FormBorderStyle = FixedDialog, MaximizeBox = false, MinimizeBox = false, ShowInTaskBar = False,

1. Design the dialog as below

a)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties- >Text=&User Name

b)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtUserName

c)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Password

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtPassword

Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties->Text=&First Name

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click->Properties->Name=txtFirstName

Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties->Text=&Last Name

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click->Properties->Name=txtLastName

Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties->Text=&Email Id

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click->Properties->Name=txtEmailId

Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click->Properties->Text=&Phone No

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click->Properties->Name=txtPhoneNo

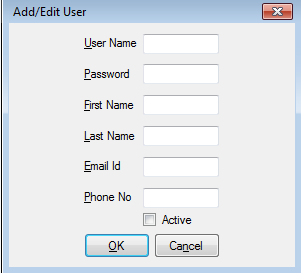
e)Goto->ToolBox->Select CheckBox-> Drag and Drop on the Form->Select CheckBox on the Form->Right Click->Properties->Name=chkIsActive; Text=&Active

f)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click->Properties->Name=btnOK; Text=&OK->Double click on button to generate btnOK\_Click Event

g)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click->Properties->Name=btnCancel; Text=&Cancel->Double click on button to generate btnCancel\_Click Event

h) RightClick on Form->Properties->AcceptButton=btnOK; CancelButton=btnCancel

i)Goto->ToolBox->Select ErrorProvider->Drag and Drop on the Form-> Select the ErrorProvider->Right Click->Properties->Name=epUserDialog





1. Add the following corresponding class(UserDailog.cs)

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.Data;

using ContactsManager.BO;

using System.Text.RegularExpressions;

namespace ContactsManager

{

public partial class UserDialog : Form

{

public UserDialog()

{

InitializeComponent();

}

private int \_UserId = 0;

public int UserId

{

get

{

return \_UserId;

}

set

{

if (!string.IsNullOrEmpty(value.ToString()))

{

\_UserId = value;

}

else

{

\_UserId = 0;

}

}

}

public string UserName

{

get

{

return txtUserName.Text;

}

set

{

txtUserName.Text = value;

}

}

public string Password

{

get

{

return txtPassword.Text;

}

set

{

txtPassword.Text = value;

}

}

public string FirstName

{

get

{

return txtFirstName.Text;

}

set

{

txtFirstName.Text = value;

}

}

public string Lastname

{

get

{

return txtLastName.Text;

}

set

{

txtLastName.Text = value;

}

}

public string EmailId

{

get

{

return txtEmailId.Text;

}

set

{

txtEmailId.Text = value;

}

}

public string PhoneNo

{

get

{

return txtPhoneNo.Text;

}

set

{

txtPhoneNo.Text = value;

}

}

public bool IsActive

{

get

{

return chkIsActive.Checked;

}

set

{

chkIsActive.Checked = value;

}

}

private void btnOK\_Click(object sender, EventArgs e)

{

//Checking Validations

epUserDialog.Clear();

bool errorflag = false;

if (txtUserName.Text.Trim() == "")

{

epUserDialog.SetError(txtUserName, "Please provide UserName");

errorflag = true;

}

if (txtFirstName.Text.Trim() == "")

{

epUserDialog.SetError(txtFirstName, "Please provide FirstName");

errorflag = true;

}

if (txtPhoneNo.Text.Trim() == "")

{

epUserDialog.SetError(txtPhoneNo, "Please provide PhoneNo like 9845362548");

errorflag = true;

}

if (txtPassword.Text.Trim() == "")

{

epUserDialog.SetError(txtPassword, "Please provide Password");

errorflag = true;

}

if (txtEmailId.Text.Trim() == "")

{

epUserDialog.SetError(txtEmailId, "Please provide EmailId");

errorflag = true;

}

else if (!Regex.IsMatch(txtEmailId.Text.Trim(), "[\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*](file:///Z:/\\w+(%5B-+.'%5D\\w+)*@\\w+(%5B-.%5D\\w+)*\\.\\w+(%5B-.%5D\\w+)*)"))

{

epUserDialog.SetError(txtEmailId, "Enter correct Email Id ex:info@deccansoft.com");

errorflag = true;

}

if (errorflag == false)

{

//creating an object of User

User objUser = new User();

UserBO objUserBO = new UserBO();

try

{

this.DialogResult = System.Windows.Forms.DialogResult.OK;

}

catch

{

MessageBox.Show("user Name is already Exists");

}

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

1. Go to → Project → Right click → clivk on Add → click on New Item → Select a New Windows Form to the Project Name = UserForm
2. Design the Form as below

dgvUser-Add Columns 

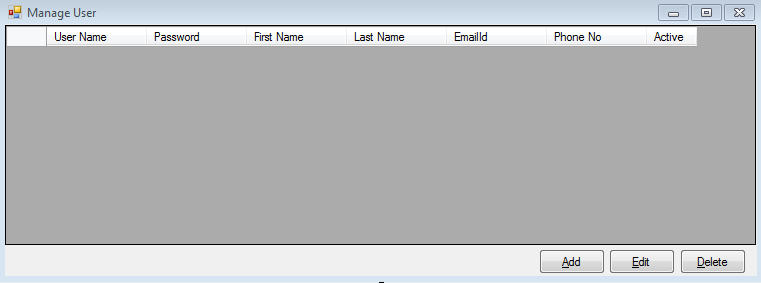
1. Name:UserID,Type:DataGridViewTextBoxColumn,HeaderText:UserId,ReadOnly:checked
2. Name:UserName,Type:DataGridViewTextBoxColumn,HeaderText: UserName,Visible:checked
3. Name:Password,Type:DataGridViewTextBoxColumn,HeaderText: Password,Visible:checked
4. Name:FirstName,Type:DataGridViewTextBoxColumn,HeaderText: FirstName,Visible:checked
5. Name:LastName,Type:DataGridViewTextBoxColumn,HeaderText: LastName,Visible:checked
6. Name:EmailID,Type:DataGridViewTextBoxColumn,HeaderText: EmailID,Visible:checked
7. Name:PhoneNo,Type:DataGridViewTextBoxColumn,HeaderText:PhoneNo,Visible:checked
8. Name:IsActive,Type:DataGridViewCheckBoxColumn,HeaderText:IsActive,Visible:checked
9. Go to dgvUser Property give Readonly=True,multyRow=False

→ Go to Tool Box Add button to UserForm → Right click on Button → click on properties

give Text=Add, Name=btnOk

→ Go to Tool Box Add button to UserForm → Right click on Button → click on properties give Text=Edit, Name=btnEdit

→ Go to Tool Box Add button to UserForm → Right click on Button → click on properties give Text=Delete, Name=btnDelete



1. Add the following code to the Form class(UserForm .cs)

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.Data;

using ContactsManager.BO;

namespace ContactsManager

{

public partial class UserForm : Form

{

public ManagerUserForm()

{

InitializeComponent();

}

private void BindDataToGrid()

{

try

{

//Bind all records into gridview

DataSet ds;

UserBO objUserBO = new UserBO();

ds = objUserBO.GetAllUsers();

dgvUser.DataSource = ds.Tables[0];

}

catch (System.Data.SqlClient.SqlException)

{

MessageBox.Show("1.Check the Connection string ","Unable to connect database ");

}

}

private void btnAdd\_Click(object sender, EventArgs e)

{

//Creating an object of Dialog

UserDialog dlgUser = new UserDialog();

if (dlgUser.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

//if users click on ok button dialog

User objUser = new User();

//Insert the record

UserBO objUserBO = new UserBO();

objUserBO.InsertUser(dlgUser.UserName, dlgUser.Password, dlgUser.FirstName, dlgUser.Lastname, dlgUser.EmailId, dlgUser.PhoneNo, dlgUser.IsActive);

//Insert into grid

BindDataToGrid();

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

//If user click on Delete button

if (dgvUser.RowCount != 0)

{

DialogResult dr;

dr = MessageBox.Show("Are you sure?", "Delete", MessageBoxButtons.YesNo);

if (dr == System.Windows.Forms.DialogResult.No)

{

return;

}

//Create an object of Delete Dialog

int userId = (int)dgvUser.SelectedRows[0].Cells[0].Value;

UserBO objUserBO = new UserBO();

bool hasRecords = objAddressBO.HasCountryRecords(userId.ToString());

if (!hasRecords)

objUserBO.DeleteUser(userId);

else

MessageBox.Show("Addresses exist that refer to this Country!", "Delete aborted", MessageBoxButtons.OK);

//Delete records from gridview

BindDataToGrid();

}

}

private void btnEdit\_Click(object sender, EventArgs e)

{

string user = "";

if (dgvUser.RowCount != 0)

{

int userId;

userId = Convert.ToInt32(dgvUser.SelectedRows[0].Cells[0].Value);

UserDialog dlgUser = new UserDialog();

UserBO objUserBO = new UserBO();

User objUser = objUserBO.GetUser(userId);

dlgUser.UserName = objUser.UserName;

dlgUser.Password = objUser.Password;

dlgUser.FirstName = objUser.FirstName;

dlgUser.Lastname = objUser.LastName;

dlgUser.EmailId = objUser.EmailId;

dlgUser.PhoneNo = objUser.Password;

dlgUser.IsActive = objUser.IsActive;

user = objUser.UserName;

if (dlgUser.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

objUser.UserName = dlgUser.UserName;

objUser.Password = dlgUser.Password;

objUser.FirstName = dlgUser.FirstName;

objUser.LastName = dlgUser.Lastname;

objUser.EmailId = dlgUser.EmailId;

objUser.PhoneNo = dlgUser.PhoneNo;

objUser.IsActive = dlgUser.IsActive;

if (user == dlgUser.UserName)

{

objUserBO.UpdateUser(objUser);

}

else

{

if (objUserBO.IsUserExist(objUser.UserName))

MessageBox.Show("UserName is Already Exist");

else

objUserBO.UpdateUser(objUser);

}

BindDataToGrid();

}

}

}

private void ManageUserForm\_Load(object sender, EventArgs e)

{

dgvUser.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

dgvUser.AutoGenerateColumns = false;

dgvUser.AllowUserToAddRows = false;

dgvUser.AllowUserToDeleteRows = false;

dgvUser.ReadOnly = true;

dgvUser.MultiSelect = false;

BindDataToGrid();

}

}

1. In MDIMainForm, Handle Click event of mnuUser and write the following code

UserForm objUser;

private void mnuManageUser\_Click(object sender, EventArgs e)

{

if (objUser == null)

{

objUser = new UserForm();

objUser.Show();

objUser.MdiParent = this;

objUser.FormClosing += objUser\_FormClosing;

}

else

{

objUser.Activate();

}

}

void objUser\_FormClosing(object sender, FormClosingEventArgs e)

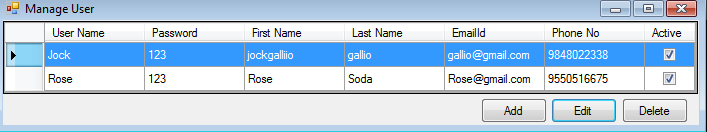
{

objUser = null;

}

1. Run and perform Add/Edit/Delete Operations on User.

=> show as below



**Manage Addressbook Form**

1. **Create Addressbook Table**

User can Manage Addresses and can perform Insert, Update, Delete Operations.

Both Admin and User can filter all the addresses based upon CountryName and StateName.

Admin can Update and Delete Addresses but Admin Can not Add the Address.

|  |  |  |
| --- | --- | --- |
| Column Name | Data type | Constraints |
| PKAddressId | Int | PK, Not null |
| FKStateId | Int | FK, Not null |
| FKUserId | Int | FK, Not null |
| FirstName | Varchar(50) | Not null |
| LastName | Varchar(50) | Allow nulls |
| EmailId | Varchar(50) | Not null |
| PhoneNo | Varchar(50) | Not null |
| Address1 | Varchar(50) | Not null |
| Address2 | Varchar(50) | Allow nulls |
| Street | Varchar(50) | Not null |
| City | Varchar(50) | Not null |
| ZipCode | Bigint | Not null |
| IsActive | Bit | Not null |

1. Create the following Stored Procedure to retrieve one or all Addresses based on @AddressId

CREATE PROCEDURE [dbo].[spGetAllAddresses](@PKAddressId int) as

Begin

if(@PKAddressId=-1)

Select ab.PKAddressId as PKAddressId, c.CountryName as CountryName,c.pkCountryid as FKCountryId ,ab.FKStateId as FKStateId, s.StateName as StateName, ab.FKUserId as FKUserId,ab.FirstName as FirstName,ab.LastName as LastName,ab.EmailId as EmailId,ab.PhoneNo as PhoneNo,ab.Address1 as Address1,ab.Address2 as Address2,ab.Street as Street,ab.City as City,ab.ZipCode as ZipCode,ab.IsActive as IsActive

from Addressbook as ab

join State as s on s.PKStateId = ab.FKStateId

join Country as c on c.PKCountryId =s.FKCountryId

else

Select ab.PKAddressId as PKAddressId, c.CountryName as CountryName,c.pkCountryid as FKCountryId ,ab.FKStateId as FKStateId, s.StateName as StateName, ab.FKUserId as FKUserId,ab.FirstName as FirstName,ab.LastName as LastName,ab.EmailId as EmailId,ab.PhoneNo as PhoneNo,ab.Address1 as Address1,ab.Address2 as Address2,ab.Street as Street,ab.City as City,ab.ZipCode as ZipCode,ab.IsActive as IsActive

from Addressbook as ab

join State as s on s.PKStateId = ab.FKStateId

join Country as c on c.PKCountryId =s.FKCountryId

where ab.PKAddressId=@PKAddressId

End

Store Procedure for GetAllAddressByUserId

CREATE Procedure [dbo].[spGetAllUserAddresses](@PKUserId int)

As

Begin

if @PKUserId=-1

select a.PKAddressId as PKAddressId, a.FKStateId as FKStateId, c. CountryName as CountryName,s.statename as StateName,a.FKUserId as FKUserId,a.FirstName as Firstname,a.LastName as LastName,a.EmailId as EmailId,a.PhoneNo as PhoneNo,a.Address1 as Address1,a.Street as Street,a.City as City,a.ZipCode as ZipCode,a.IsActive as IsActive

from AddressBook as a

join State as s

on s.pkstateid=a.fkstateid

join country as c

on c.pkcountryid=s.fkcountryid

else

select a.PKAddressId as PKAddressId, a.FKStateId as FKStateId,c.CountryName as CountryName,s.StateName as StateName,a.FKUserId as FKUserId,a.firstname as FirstName,a.LastName as LastName,a.EmailId as EmailId,a.PhoneNo as PhoneNo,a.Address1 as Address1,a.Street as Street,a.City as City,a.ZipCode as ZipCode,a.IsActive as IsActive

from AddressBook as a

join State as s

on s.pkstateid=s.fkstateid

join country as c

on c.pkcountryid=s.fkcountryid

Where a.fkuserid=@PKUserId

End

1. Create the following Stored Procedure to Add/Edit/Delete Address

CREATE PROCEDURE [dbo].[spManageAddress](@ActionType tinyint,@PKAddressId int output, @FKStateId int, @FKUserId int, @FirstName varchar(50),@LastName varchar(50),@EmailId varchar(50),@PhoneNo varchar(50),@Address1 varchar(50),@Address2 varchar(50),@Street varchar(50), @City varchar(50), @ZipCode bigint,@IsActive bit )as

if(@ActionType=1)

begin

insert into Addressbook(FKUserId,FKStateId,FirstName,LastName,EmailId,PhoneNo,Address1,Address2,Street,City,ZipCode,IsActive)values(@FKUserId,@FKStateId,@FirstName,@LastName,@EmailId,@PhoneNo,@Address1,@Address2,@Street,@City,@ZipCode,@IsActive)

select @PKAddressId = @@Identity

End

else if(@ActionType = 2)

update Addressbook set FKStateId=@FKStateId,FKUserId=@FKUserId,FirstName=@FirstName,LastName=@LastName,EmailId=@EmailId,PhoneNo=@PhoneNo,Address1=@Address1,Address2=@Address2,Street=@Street,City=@City,ZipCode=@ZipCode,IsActive=@IsActive where PKAddressId=@PKAddressId

else if(@ActionType=3)

delete from Addressbook where PKAddressId=@PKAddressId

1. Create the following Stored Procedure to Filter the AddressBook Based on CountryId,StateId,IsActive

ALTER PROCEDURE [dbo].[spGetFilteredAddressbook]

(@PKCountryID int, @PKStateID int, @IsActive bit)

AS

BEGIN

SELECT FirstName, a.City AS City, FKStateId, FKCountryId, EmailId, PhoneNO, Street, Address1, StateName, co.CountryName AS CountryName, a.IsActive AS IsActive, a.PKAddressId AS PKAddressId FROM dbo.Addressbook a JOIN dbo.State s ON s.PKStateId = a.FKStateId JOIN dbo.Country co ON s.FKCountryId = co.PKCountryId

**EXCEPT**

(SELECT FirstName, a.City AS City, FKStateId, FKCountryId, EmailId, PhoneNO, Street, Address1, StateName, co.CountryName AS CountryName, a.IsActive AS IsActive, a.PKAddressId AS PKAddressId FROM dbo.Addressbook a JOIN dbo.State s ON s.PKStateId = a.FKStateId JOIN dbo.Country co ON s.FKCountryId = co.PKCountryId

WHERE s.FKCountryId != @PKCountryID

UNION

SELECT FirstName, a.City AS City, FKStateId, FKCountryId, EmailId, PhoneNO, Street, Address1, StateName, co.CountryName AS CountryName, a.IsActive AS IsActive, a.PKAddressId AS PKAddressId FROM dbo.Addressbook a JOIN dbo.State s ON s.PKStateId = a.FKStateId JOIN dbo.Country co ON s.FKCountryId = co.PKCountryId

WHERE a.FKStateId != @PKStateID

UNION

SELECT FirstName, a.City AS City, FKStateId, FKCountryId, EmailId, PhoneNO, Street, Address1, StateName, co.CountryName AS CountryName, a.IsActive AS IsActive, a.PKAddressId AS PKAddressId FROM dbo.Addressbook a JOIN dbo.State s ON s.PKStateId = a.FKStateId JOIN dbo.Country co ON s.FKCountryId = co.PKCountryId

WHERE a.IsActive != @IsActive)

END

=> Go to → Data folder → Right click → click on Add → click on class → give the name as AddressBook

1. Write the following code in AddressBook Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ContactsManager.Data

{

public class AddressBook

{

private string \_Validation = "";

public void Validate()

{

if (!String.IsNullOrEmpty(\_Validation))

{

throw new ApplicationException(\_Validation);

}

}

public Addressbook()

{

}

public Addressbook(int addressId, string stateName, int stateId, int userId, string firstName, string lastName, string emailId, string phoneNo, string address1, string address2, string street, string city, long zipCode, bool isActive)

{

//Assign all parameters for checking properties

\_AddressId = addressId;

this.StateId = stateId;

this.StateName = stateName;

this.UserId = userId;

this.FirstName = firstName;

this.LastName = lastName;

this.EmailId = emailId;

this.PhoneNo = phoneNo;

this.Address1 = address1;

this.Address2 = address2;

this.Street = street;

this.City = city;

this.ZipCode = zipCode;

this.IsActive = isActive;

}

public Addressbook(int addressId, string stateName, int stateId, int countryId, int userId, string firstName, string lastName, string emailId, string phoneNo, string address1, string address2, string street, string city, long zipCode, bool isActive)

{

\_AddressId = addressId;

this.StateId = stateId;

this.StateName = stateName;

this.UserId = userId;

this.FirstName = firstName;

this.LastName = lastName;

this.EmailId = emailId;

this.PhoneNo = phoneNo;

this.Address1 = address1;

this.Address2 = address2;

this.Street = street;

this.City = city;

this.ZipCode = zipCode;

this.IsActive = isActive;

this.CountryID = countryId;

}

public Addressbook(int addressId, int stateId, int userId, string firstName, string lastName, string emailId, string phoneNo, string address1, string address2, string street, string city, long zipCode, bool isActive)

{

\_AddressId = addressId;

this.StateId = stateId;

this.UserId = userId;

this.FirstName = firstName;

this.LastName = lastName;

this.EmailId = emailId;

this.PhoneNo = phoneNo;

this.Address1 = address1;

this.Address2 = address2;

this.Street = street;

this.City = city;

this.ZipCode = zipCode;

this.IsActive = isActive;

}

private int \_AddressId;

public int AddressId

{

get

{

return \_AddressId;

}

set

{

\_AddressId = value;

}

}

private int \_StateId;

public int StateId

{

get

{

return \_StateId;

}

set

{

\_StateId = value;

}

}

public int \_CountryID;

public int CountryID

{

get

{

return \_CountryID;

}

set

{

\_CountryID = value;

}

}

private int \_UserId;

public int UserId

{

get

{

return \_UserId;

}

set

{

\_UserId = value;

}

}

private string \_StateName;

public string StateName

{

get

{

return \_StateName;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for StateName" + "\t";

\_StateName = value;

}

else

\_StateName = value;

}

}

private string \_FirstName;

public string FirstName

{

get

{

return \_FirstName;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for FirstName" + "\t";

\_FirstName = value;

}

else

\_FirstName = value;

}

}

private string \_LastName;

public string LastName

{

get

{

return \_LastName;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for LastName" + "\t";

\_LastName = value;

}

else

{

\_LastName = value;

}

}

}

private string \_EmailId;

public string EmailId

{

get

{

return \_EmailId;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for EmailId" + "\t";

\_EmailId = value;

}

else

\_EmailId = value;

}

}

private string \_PhoneNo;

public string PhoneNo

{

get

{

return \_PhoneNo;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for PhoneNo" + "\t";

\_PhoneNo = value;

}

else

\_PhoneNo = value;

}

}

private string \_Address1;

public string Address1

{

get

{

return \_Address1;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for Address1" + "\t";

\_Address1 = value;

}

else

\_Address1 = value;

}

}

private string \_Address2;

public string Address2

{

get

{

return \_Address2;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for Address2" + "\t";

\_Address2 = value;

}

else

\_Address2 = value;

}

}

private string \_Street;

public string Street

{

get

{

return \_Street;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for Street" + "\t";

\_Address1 = value;

}

else

\_Street = value;

}

}

private string \_City;

public string City

{

get

{

return \_City;

}

set

{

if (value == "")

{

\_Validation += "Please provide a value for City" + "\t";

\_City = value;

}

else

\_City = value;

}

}

private long \_ZipCode;

public long ZipCode

{

get

{

return \_ZipCode;

}

set

{

if (value <= 0)

{

\_Validation += "Please provide a value for ZipCode" + "\t";

\_ZipCode = value;

}

else

\_ZipCode = value;

}

}

private bool \_IsActive;

public bool IsActive

{

get

{

return \_IsActive;

}

set

{

\_IsActive = value;

}

}

}

}

=> Go to → DB Folder → Right click → Click on Add → Click on Class → give the name:AddressBookDB

1. Write the following code in AddressBookDB class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using ContactsManager.Data;

using System.Data;

using System.Data.SqlClient;

namespace ContactsManager.DB

{

public class AddressBookDB

{

public AddressbookDB()

{ }

/// To perform the CRUD operations on Address

public static void ManageAddress(AddressBook objAddressbook, ActionType action)

{

try

{

string spName = "spManageAddress";

SqlParameter pActionType = new SqlParameter("@ActionType", SqlDbType.TinyInt);

SqlParameter pAddressId = new SqlParameter("@PKAddressId", SqlDbType.Int);

SqlParameter pStateId = new SqlParameter("@FKStateId", SqlDbType.Int);

SqlParameter pUserId = new SqlParameter("@FKUserId", SqlDbType.Int);

SqlParameter pFirstName = new SqlParameter("@FirstName", SqlDbType.VarChar, 50);

SqlParameter pLastName = new SqlParameter("@LastName", SqlDbType.VarChar, 50);

SqlParameter pEmailId = new SqlParameter("@EmailId", SqlDbType.VarChar, 50);

SqlParameter pPhoneNo = new SqlParameter("@PhoneNo", SqlDbType.VarChar, 50);

SqlParameter pAddress1 = new SqlParameter("@Address1", SqlDbType.VarChar, 50);

SqlParameter pAddress2 = new SqlParameter("@Address2", SqlDbType.VarChar, 50);

SqlParameter pStreet = new SqlParameter("@Street", SqlDbType.VarChar, 50);

SqlParameter pCity = new SqlParameter("@City", SqlDbType.VarChar, 50);

SqlParameter pZipCode = new SqlParameter("@ZipCode", SqlDbType.BigInt, 50);

SqlParameter pIsActive = new SqlParameter("@IsActive", SqlDbType.Bit);

pActionType.Value = action;

if (action == ActionType.Add)

pAddressId.Direction = ParameterDirection.Input;

else

pAddressId.Value = objAddressbook.AddressId;

if (action == ActionType.Add || action == ActionType.Update)

{

pStateId.Value = objAddressbook.StateId;

pUserId.Value = objAddressbook.UserId;

pFirstName.Value = objAddressbook.FirstName;

pLastName.Value = objAddressbook.LastName;

pEmailId.Value = objAddressbook.EmailId;

pPhoneNo.Value = objAddressbook.PhoneNo;

pAddress1.Value = objAddressbook.Address1;

pAddress2.Value = objAddressbook.Address2;

pStreet.Value = objAddressbook.Street;

pCity.Value = objAddressbook.City;

pZipCode.Value = objAddressbook.ZipCode;

pIsActive.Value = objAddressbook.IsActive;

}

SqlHelper.ExecuteNonQuery(Helper.ConnectionString, CommandType.StoredProcedure, spName, pActionType, pAddressId, pStateId, pUserId, pFirstName, pLastName, pEmailId, pPhoneNo, pAddress1, pAddress2, pStreet, pCity, pZipCode, pIsActive);

}

catch (Exception ex)

{

throw ex;

}

}

/// To fetch the all addresses

public static DataSet GetUserAddresses()

{

try

{

string spName = "spGetAllUserAddresses";

SqlParameter pPKUserId = new SqlParameter("@PKUserId", SqlDbType.Int);

pPKUserId.Value = Helper.UserId;

DataSet ds = SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spName, pPKUserId);

return ds;

}

catch (Exception ex)

{

throw ex;

}

}

/// To fetch the Address by AddressId

public static AddressBook GetAddress(int addressId)

{

try

{

string spName = "spGetAllAddresses";

SqlParameter pAddressId = new SqlParameter("@PKAddressId", SqlDbType.Int);

pAddressId.Value = addressId;

SqlDataReader drRow = SqlHelper.ExecuteReader(Helper.ConnectionString, CommandType.StoredProcedure, spName, pAddressId);

AddressBook objAddbook = null;

if (drRow.HasRows)

{

drRow.Read();

int addressid = drRow.GetInt32(drRow.GetOrdinal("PKAddressId"));

string stateName = drRow.GetString(drRow.GetOrdinal("StateName"));

int stateId = drRow.GetInt32(drRow.GetOrdinal("FKStateId"));

int userId = drRow.GetInt32(drRow.GetOrdinal("FKUserId"));

string firstName = drRow.GetString(drRow.GetOrdinal("FirstName"));

string lastName = drRow.GetString(drRow.GetOrdinal("LastName"));

string emailId = drRow.GetString(drRow.GetOrdinal("EmailId"));

string phoneNo = drRow.GetString(drRow.GetOrdinal("PhoneNo"));

string address1 = drRow.GetString(drRow.GetOrdinal("Address1"));

string address2 = drRow.GetString(drRow.GetOrdinal("Address2"));

string street = drRow.GetString(drRow.GetOrdinal("Street"));

string city = drRow.GetString(drRow.GetOrdinal("City"));

long zipCode = drRow.GetInt64(drRow.GetOrdinal("ZipCode"));

bool isActive = Convert.ToBoolean(drRow["IsActive"]);

int countryId = drRow.GetInt32(drRow.GetOrdinal("FKCountryId"));

objAddbook = new AddressBook(addressid, stateName, stateId, countryId, userId, firstName, lastName, emailId, phoneNo, address1, address2, street, city, zipCode, isActive);

}

drRow.Close();

return objAddbook;

}

catch (Exception ex)

{

throw ex;

}

}

public static DataSet GetAddresses()

{

try

{

string spName = "spGetAllAddresses";

SqlParameter pPKAddressId = new SqlParameter("@PKAddressId", SqlDbType.Int);

pPKAddressId.Value = -1;

DataSet ds = SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spName, pPKAddressId);

return ds;

}

catch (Exception ex)

{

throw ex;

}

}

/// To fetch specific records from DB

public static DataSet GetFilteredAddressbook(string countryId, string stateId, bool isActive)

{

try

{

string spName = "spGetFilteredAddressbook";

SqlParameter pCountryID = new SqlParameter("@PKCountryID", SqlDbType.Int);

pCountryID.Value = countryId == "-99" ? null : countryId;

SqlParameter pStateID = new SqlParameter("@PKStateID", SqlDbType.Int);

pStateID.Value = stateId == "-99" ? null : stateId;

SqlParameter pIsActive = new SqlParameter("@IsActive", SqlDbType.Bit);

pIsActive.Value = isActive;

DataSet ds = SqlHelper.ExecuteDataset(Helper.ConnectionString, CommandType.StoredProcedure, spName, pCountryID, pStateID, pIsActive);

ds.Tables[0].TableName = "Addressbook";

return ds;

}

catch (Exception ex)

{

throw ex;

}

}

}

1. Add the AddressBookBO class Under BO folder

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

using ContactsManager.DB;

using ContactsManager.Data;

namespace ContactsManager.BO

{

public class AddressBookBO

{

public AddressbookBO()

{ }

public DataSet GetAddress()

{

return AddressBookDB.GetAddresses();

}

/// Adds a single record to Addressbook table

public void InsertAddressbook(AddressBook objAddbook)

{

objAddbook.Validate();

AddressBookDB.ManageAddress(objAddbook, ActionType.Add);

}

/// To insert the Address

public void InsertAddressbook(int stateId, int userId, string firstName, string lastName, string emailId, string phoneNo, string address1, string address2, string city, string street, long zipCode, bool isActive)

{

AddressBook objAddbook = new AddressBook(-1, stateId, userId, firstName, lastName, emailId, phoneNo, address1, address2, street, city, zipCode, isActive);

objAddbook.Validate();

AddressBookDB.ManageAddress(objAddbook, ActionType.Add);

}

public void UpdateAddressbook(int addressId, int stateId, int userId, string firstName, string lastName, string emailId, string phoneNo, string address1, string address2, string city, string street, long zipCode, bool isActive)

{

AddressBook objAddbook = new AddressBook(addressId, stateId, userId, firstName, lastName, emailId, phoneNo, address1, address2, street, city, zipCode, isActive);

objAddbook.Validate();

AddressBookDB.ManageAddress(objAddbook, ActionType.Update);

}

/// To fetch the Address record by AddressId

public AddressBook GetAddress(int addressId)

{

return AddressBookDB.GetAddress(addressId);

}

/// To delete the Addressbook

public void DeleteAddressbook(int adressId)

{

AddressBook objAddbook = new AddressBook(adressId, 0, 0, "", "", "", "", "", "", "", "", 0, false);

AddressBookDB.ManageAddress(objAddbook, ActionType.Delete);

}

/// This is for fetching filtered information

public DataSet GetFilteredAddressbook(string countryId, string stateId, string isActive)

{

bool ia;

ia = isActive == "1" ? true : false;

return AddressBookDB.GetFilteredAddressbook(countryId, stateId, ia);

}

public DataSet GetUserAddresses()

{

return AddressBookDB.GetUserAddresses ();

}

}

}

1. Go to → Project → Right click ->click on Add → Click on New Item → Select a New Windows Forms, Name = AddressBookDialog
2. Right click on AddressBookDialog → go to properties → Set the following properties

FormBorderStyle = FixedDialog, MaximizeBox = false, MinimizeBox = false, ShowInTaskBar = False,

1. Design the dialog as below

a)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&First Name

b)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtFirstName

c)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Last Name

d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtLastName

e)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Email Id

f)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtEmailId

g)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Phone No

h)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtPhoneNo

i)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Country

j)Goto->ToolBox->Select ComboBox-> Drag and Drop on the Form->Select Combo Box on the Form->Right Click->Properties->Name=cmbCountry; DropDownStyle=DropDownList.

k)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&State

l)Goto->ToolBox->Select ComboBox-> Drag and Drop on the Form->Select Combo Box on the Form->Right Click->Properties->Name=cmbState; DropDownStyle=DropDownList.

m)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=Address&1

n)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtAddress1

o)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=Address&2

p)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtAddress2

q)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=Ci&ty

r)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtCity

s)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=St&reet

t)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtStreet

u)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&Zipcode

v)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtZipcode

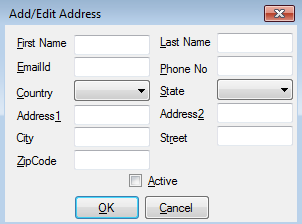
w)Goto->ToolBox->Select CheckBox-> Drag and Drop on the Form->Select CheckBox on the Form->Right Click- >Properties->Name=chkIsActive; Text=&Active

x)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click- >Properties->Name=btnOK; Text=&OK->Double click on button to generate btnOK\_Click Event

y)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click- >Properties->Name=btnCancel; Text=&Cancel->Double click on button to generate btnCancel\_Click Event

z) RightClick on Form->Properties->AcceptButton=btnOK; CancelButton=btnCancel

* + - 1. .Goto->ToolBox->Select ErrorProvider->Drag and Drop on the Form-> Select the ErrorProvider->Right Click->Properties->Name=epAddressDialog





1. Add the following code Inside AddressBookDialog.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.BO;

using ContactsManager.Data;

using System.Text.RegularExpressions;

namespace ContactsManager

{

public partial class AddressBookDialog : Form

{

public AddressDialog()

{

InitializeComponent();

BindCoutries();

BindStates();

}

public string CountryName

{

get

{

return cmbCountryName.SelectedValue.ToString();

}

set

{

cmbCountryName.SelectedValue = value;

}

}

public string StateName

{

get

{

return cmbStateName.SelectedValue.ToString();

}

set

{

cmbStateName.SelectedValue = value;

}

}

public string EmailId

{

get

{

return txtEmailId.Text;

}

set

{

txtEmailId.Text = value;

}

}

public string phoneNo

{

get

{

return txtPhoneNo.Text;

}

set

{

txtPhoneNo.Text = value;

}

}

public string Address1

{

get

{

return txtAddress1.Text;

}

set

{

txtAddress1.Text = value;

}

}

public string Address2

{

get

{

return txtAddress2.Text;

}

set

{

txtAddress2.Text = value;

}

}

public string City

{

get

{

return txtCity.Text;

}

set

{

txtCity.Text = value;

}

}

public long ZipCode

{

get

{

return Convert.ToInt64(txtZipCode.Text);

}

set

{

txtZipCode.Text = value.ToString();

}

}

public string FirstName

{

get

{

return txtFirstName.Text;

}

set

{

txtFirstName.Text = value;

}

}

public string LastName

{

get

{

return txtLastName.Text;

}

set

{

txtLastName.Text = value;

}

}

public string Street

{

get

{

return txtStreet.Text;

}

set

{

txtStreet.Text = value;

}

}

public bool IsActive

{

get

{

return chkIsActive.Checked;

}

set

{

chkIsActive.Checked = value;

}

}

private void btnOK\_Click(object sender, EventArgs e)

{

int ZipCode = 0;

bool errorflag = false;

epAddressDailog.Clear();

Country obj = new CountryBO().GetCountryById(Convert.ToInt32(cmbCountryName.SelectedValue));

if (txtZipCode.Text.Trim() != "")

{

ZipCode = Convert.ToInt32(txtZipCode.Text.Trim());

}

if (txtFirstName.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtFirstName, "Please provide FirstName");

}

if (!Regex.IsMatch(txtEmailId.Text.Trim(), "\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*"))

{

epAddressDailog.SetError(txtEmailId, "Enter correct Email Id ex:info@deccansoft.com");

errorflag = true;

}

if (txtEmailId.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtEmailId, "Please provide EmailId");

}

if (txtPhoneNo.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtPhoneNo, "Please provide PhoneNo like 9965883566");

}

if (txtStreet.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtStreet, "Please provide StreetName");

}

if (txtCityName.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtCityName, "Please provide CityName");

}

if (txtZipCode.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtZipCode, "Please provide ZipCode ex:500045");

}

if (txtAddress1.Text.Trim() == "")

{

errorflag = true;

epAddressDailog.SetError(txtAddress1, "Please provide Address1");

}

if (obj.ZipCodeStart > ZipCode || obj.ZipCodeEnd < ZipCode)

{

errorflag = true;

epAddressDailog.SetError(txtZipCode, "Please provide valid zipcode");

}

if (btnOK.Text != "Update" && errorflag == false)

{

try

{

this.DialogResult = System.Windows.Forms.DialogResult.OK;

}

catch (ApplicationException ex)

{

MessageBox.Show(ex.Message);

}

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.DialogResult = System.Windows.Forms.DialogResult.Cancel;

}

private void BindCoutries()

{

cmbCountryName.DisplayMember = "CountryName";

cmbCountryName.ValueMember = "PKCountryId";

DataSet ds = new CountryBO().**GetCountries**();

cmbCountryName.DataSource = ds.Tables[0];

}

private void BindStates()

{

cmbStateName.DisplayMember = "StateName";

cmbStateName.ValueMember = "PKStateId";

DataSet ds = new StateBO().GetAllActiveStates(Convert.ToInt32(cmbCountryName.SelectedValue));

cmbStateName.DataSource = ds.Tables[0];

}

private void cmbCountryName\_SelectedIndexChanged(object sender, EventArgs e)

{

int countryID = Convert.ToInt32(cmbCountryName.SelectedValue);

BindStates();

}

private void txtZipCode\_KeyPress(object sender, KeyPressEventArgs e)

{

int ascii = (int)e.KeyChar;

//For BackSpace

if (ascii < 8)

return;

//To Enter only Digits

if (ascii < 48 || ascii > 57)

{

e.Handled = true;

}

}

}

}

1. Go to → Project → Right click ->click on Add → Click on New Item → Select a New Windows Forms, Name = AddressBookForm

dgvAddressbookAdd Columns

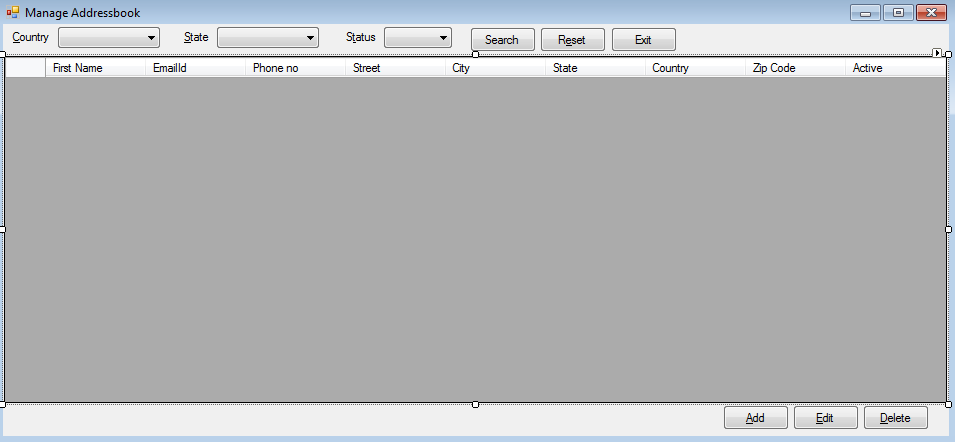
* Name:AddressID,Type:DataGridViewTextBoxColumn,HeaderText:AddressId,ReadOnly:checked
* Name:StateId,Type:DataGridViewTextBoxColumn,HeaderText:StateId,ReadOnly:checked
* Name:UserId,Type:DataGridViewTextBoxColumn,HeaderText:UserId,ReadOnly:checked
* Name:FirstNameType:DataGridViewTextBoxColumn,HeaderText:FirstName,Visible:checked
* Name:LastName,Type:DataGridViewTextBoxColumn,HeaderText:LastName, ReadOnly:checked
* Name:EmailID,Type:DataGridViewTextBoxColumn,HeaderText: EmailID,Visible:checked
* Name:PhoneNo,Type:DataGridViewTextBoxColumn,HeaderText:PhoneNo,Visible:checked
* Name:Street,Type:DataGridViewTextBoxColumn,HeaderText:Street,Visible:checked
* Name:City,Type:DataGridViewTextBoxColumn,HeaderText:City,Visible:checked
* Name:State,Type:DataGridViewTextBoxColumn,HeaderText:StateVisible:checked
* Name:Country,Type:DataGridViewTextBoxColumn,HeaderText:Country,Visible:checked
* Name:IsActive,Type:DataGridViewCheckBoxColumn,HeaderText:IsActive,Visible:checked
* Go to dgvAddressbook Properties give Readonly=True,multyRow=False

1. Design the Form as below

a. Goto->ToolBox->Select->Button->Drag and Drop Button-> Right click on the button->Properties->Name=btnAdd; Anchor=Bottom,Right; Text=&Add; ->Double click on button to generate btnAdd\_Click Event

b. Goto->ToolBox->Select->Button->Drag and Drop Button-> Right click on the button->Properties->Name=btnEdit; Anchor=Bottom,Right; Text=&Edit; ->Double click on button to generate btnEdit\_Click Event

c. Goto->ToolBox->Select->Button->Drag and Drop Button-> Right click on the button->Properties->Name=btnDelete; Anchor=Bottom,Right; Text=&Delete; ->Double click on button to generate btnDelete\_Click Event



1. Add the following code to the Form AddressBookForm.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using ContactsManager.BO;

using ContactsManager.Data;

using ContactsManager;

namespace ContactsManager

{

public partial class AddressBookForm : Form

{

public ManageAddressbookForm()

{

InitializeComponent();

}

private void AddressBook\_Load(object sender, EventArgs e)

{

if (Helper.Role == "Admin")

{

BindDataToGrid();

btnAdd.Enabled = false;

}

else

{

BindUserDataToGrid();

btnAdd.Enabled = true;

}

cmbCountry.DisplayMember = "CountryName";

cmbCountry.ValueMember = "PKCountryId";

DataSet ds =objCountryBO.GetActiveCountries();

cmbCountry.DataSource = ds.Tables[0];

cmbIsActive.Items.Insert(0, "InActive");

cmbIsActive.Items.Insert(1, "Active");

cmbIsActive.SelectedIndex = 1;

dgvManageAddressbook.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

dgvManageAddressbook.AutoGenerateColumns = false;

dgvManageAddressbook.AllowUserToAddRows = false;

dgvManageAddressbook.AllowUserToDeleteRows = false;

dgvManageAddressbook.ReadOnly = true;

dgvManageAddressbook.MultiSelect = false;

}

private void BindDataToGrid()

{

//Bind all records into Datagridview

DataSet ds;

AddressBookBO objAddressBO = new AddressBookBO();

ds = objAddressBO.GetAddresses();

dgvAddress.DataSource = ds.Tables[0];

}

private void btnAdd\_Click(object sender, EventArgs e)

{

try

{

//Creating an object of Addressbook Dialog

AddressBookDialog dlgAddressDialog = new AddressBookDialog();

if (dlgAddressDialog.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

//if user clicks on ok button of Dialog

AddressBook objAddbook = new AddressBook();

AddressBookBO objAddbookBO = new AddressBookBO();

//Insert the address

objAddbookBO.InsertAddressbook(Convert.ToInt32(dlgAddressDialog.StateName), Helper.PKUserId, dlgAddressDialog.FirstName, dlgAddressDialog.LastName, dlgAddressDialog.EmailId, dlgAddressDialog.phoneNo, dlgAddressDialog.Address1, dlgAddressDialog.Address2, dlgAddressDialog.Street, dlgAddressDialog.City, dlgAddressDialog.ZipCode, dlgAddressDialog.IsActive);

//Updating the grid based on Role

if (Helper.Role == "Admin")

{

BindDataToGrid();

btnAdd.Enabled = false;

}

else

{

BindUserDataToGrid();

btnAdd.Enabled = true;

}

}

}

catch (ApplicationException ex)

{

MessageBox.Show(ex.Message);

}

}

private void btnEdit\_Click(object sender, EventArgs e)

{

//If user click on edit butoon

if (dgManageAddressbook.RowCount != 0)

{

int PKAddressId;

PKAddressId = (int)(dgManageAddressbook.SelectedRows[0].Cells[0].Value);

AddressBookDialog dlgAddressDialog = new AddressBookDialog();

AddressBookBO objAddbookBO = new AddressBookBO();

//Update the records

AddressBook objAddbook = objAddbookBO.GetAddressByAddressId(PKAddressId);

dlgAddressDialog.CountryName = objAddbook.CountryID.ToString();

dlgAddressDialog.StateName = objAddbook.StateId.ToString();

dlgAddressDialog.FirstName = objAddbook.FirstName.ToString();

dlgAddressDialog.LastName = objAddbook.LastName.ToString();

dlgAddressDialog.EmailId = objAddbook.EmailId.ToString();

dlgAddressDialog.phoneNo = objAddbook.PhoneNo.ToString();

dlgAddressDialog.Address1 = objAddbook.Address1.ToString();

dlgAddressDialog.Address2 = objAddbook.Address2.ToString();

dlgAddressDialog.Street = objAddbook.Street.ToString();

dlgAddressDialog.City = objAddbook.City.ToString();

dlgAddressDialog.ZipCode = objAddbook.ZipCode;

dlgAddressDialog.IsActive = objAddbook.IsActive;

if (dlgAddressDialog.ShowDialog() == DialogResult.OK)

{

objAddbookBO.UpdateAddressbook(objAddbook.PKAddressId, Convert.ToInt32(dlgAddressDialog.StateName), objAddbook.FKUserId, dlgAddressDialog.FirstName, dlgAddressDialog.LastName, dlgAddressDialog.EmailId, dlgAddressDialog.phoneNo, dlgAddressDialog.Address1, dlgAddressDialog.Address2, dlgAddressDialog.City, dlgAddressDialog.Street, dlgAddressDialog.ZipCode, Convert.ToBoolean(dlgAddressDialog.IsActive));

//Update the grid based on Role

if (Helper.Role == "Admin")

{

BindDataToGrid();

btnAdd.Enabled = false;

}

else

{

BindUserDataToGrid();

btnAdd.Enabled = true;

}

}

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

//If user click on Delete button

if (dgManageAddressbook.RowCount != 0)

{

DialogResult dr;

dr = MessageBox.Show("Are you sure? Delete Address", "Delete", MessageBoxButtons.YesNo);

if (dr == System.Windows.Forms.DialogResult.No)

{

return;

}

//creating an object of Delete Dialog

int PKAddressId = (int)dgManageAddressbook.SelectedRows[0].Cells[0].Value;

AddressBookBO objAddbookBO = new AddressBookBO();

objAddbookBO.DeleteAddressbook(PKAddressId);

//Delete the records from gridview

if (Helper.Role == "Admin")

{

BindDataToGrid();

btnAdd.Enabled = false;

}

else

{

BindUserDataToGrid();

btnAdd.Enabled = true;

}

}

}

private void btnSearch\_Click(object sender, EventArgs e)

{

BindFilteredDataToGrid();

}

private void BindFilteredDataToGrid()

{

string pkStateID = cmbState.SelectedValue == null ? "-99" : cmbState.SelectedValue.ToString();

DataSet ds = objAddressBO

.GetFilteredAddressbook(cmbCountry.SelectedValue.ToString(), pkStateID, cmbIsActive.SelectedIndex.ToString());

dgvAddress.DataSource = ds.Tables[0];

dgvAddress.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

}

private void cbxCountry\_SelectedIndexChanged(object sender, EventArgs e)

{

cmbState.DisplayMember = "StateName";

cmbState.ValueMember = "PKStateId";

DataSet ds = new StateBO().GetAllActiveStates(Convert.ToInt32(cmbCountry.SelectedValue));

int rowCount = ds.Tables[0].Rows.Count;

if (rowCount != 0)

{

cmbState.DataSource = ds.Tables[0];

}

else

{

cmbState.DataSource = null;

}

}

private void btnReset\_Click(object sender, EventArgs e)

{

if (Helper.Role == "Admin")

{

BindDataToGrid();

btnAdd.Enabled = false;

}

else

{

BindUserDataToGrid();

btnAdd.Enabled = true;

}

}

private void BindUserDataToGrid()

{

DataSet ds = objAddressBO.GetUserAddresses();

dgvAddressBook.DataSource = ds.Tables[0];

dgvAddressBook.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

}

private void btnExitClick(object sender, EventArgs e)

{

this.Close();

}

}

}

1. In MDIMainForm, Handle Click event of mnuAddressbook and write the following code

AddressBookForm objAddressbook;

private void mnuAddressBook\_Click(object sender, EventArgs e)

{

if (objAddressbook == null)

{

objAddressbook = new AddressBookForm();

objAddressbook.Show();

objAddressbook.MdiParent = this;

objAddress.FormClosing += objAddress\_FormClosing;

}

else

{

objAddress.Activate();

}

}

void objAddress\_FormClosing(object sender, FormClosingEventArgs e)

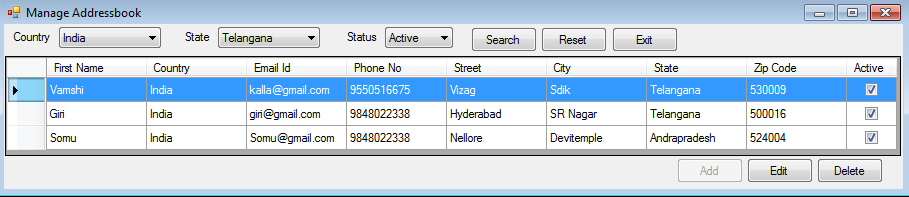
{

objAddress = null;

}

=> Run the application then check operations Add/Edit/Delete

=>Show given below



**Authentication / Login Form**

1. In App. Config,

|  |
| --- |
| <appSettings>  <add key="AdminUserName" value="Admin"></add>  <add key="AdminPassword" value="Admin"></add>  </appSettings> |

1. Add a New Windows Form to the Project. Name = LoginForm
2. Design the form as below

a)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&UserName

b)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtUserName

c)Goto->ToolBox->Select Label->Drag and Drop on the Form->Select Label on the Form->Right Click- >Properties->Text=&PassWord

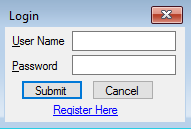
d)Goto->ToolBox->Select TextBox-> Drag and Drop on the Form->Select TextBox on the Form->Right Click- >Properties->Name=txtPassword

e)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click- >Properties->Name=btnSubmit; Text=&Submit ->Double click on button to generate btnSubmit\_Click Event

f)Goto->ToolBox->Select Button-> Drag and Drop on the Form->Select Button on the Form->Right Click- >Properties->Name=btnCancel; Text=&Cancel ->Double click on button to generate btnCancel\_Click Event

g) RightClick on Form->Properties->AcceptButton=btnSubmit; CancelButton=btnCancel

h)Goto->ToolBox->select LinkLabel->Drag and Drop on th Form->Select the LinkLabel on The Form->Right Click->Properties->Name=lbRegister; Text=Register Here->Double Click on LinkLabel(Register Here) to generate lbRegister\_LinkClicked event.



1. Add the following code to LoginForm.cs

private string \_role;

public string role

{

get

{

return \_role;

}

set

{

\_role = value;

}

}

MDIMainForm parent = new MDIMainForm();

private void btnSubmit\_Click(object sender, EventArgs e)

{

if (txtUserName.Text.Trim() != "" && txtPassword.Text.Trim() != "")

{

//Checking the Admin and username

string adminUserName = System.Configuration.ConfigurationManager.AppSettings["AdminUserName"];

string adminPassword = System.Configuration.ConfigurationManager.AppSettings["AdminPassword"];

if (txtUserName.Text.ToLower() == adminUserName.ToLower() && txtPassword.Text.Trim() == adminPassword)

{

Helper.Role = "Admin";

Helper.UserId = -1;

this.DialogResult = DialogResult.OK;

}

else

{

int pkuserid = objUserBO.AuthenticateUser(txtUserName.Text.Trim(), txtPassword.Text.Trim());

if (pkuserid !=0 )

{

Helper.Role = "user";

Helper.UserId = Convert.ToInt32(pkuserid);

this.DialogResult = DialogResult.OK;

}

else

{

MessageBox.Show("Please provide the valid credentials");

}

}

}

else

{

MessageBox.Show("Please provide username and password");

}

}

private void LoginForm\_Load(object sender, EventArgs e)

{

}

private void lnkRegister\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

UserDialog dlguser = new UserDialog();

if (objUserdlg.ShowDialog() == System.Windows.Forms.DialogResult.OK)

{

UserBO objUserBO = new UserBO();

if (objUserBO.IsUserExist(objUserdlg.UserName)

MessageBox.Show("UserName is Already Exist");

Else

ObjUserBO.InsertUser(objUserdlg.UserName, objUserdlg.Password, objUserdlg.FirstName, objUserdlg.Lastname, objUserdlg.EmailId, objUserdlg.PhoneNo, objUserdlg.IsActive);

MessageBox.Show("Registred Sucessfully");

Helper.Role = "Visitor";

}

private void btnCancel\_Click(object sender, EventArgs e)

{

txtUserName.Text = "";

txtPassword.Text = "";

this.DialogResult = DialogResult.Cancel;

}

}

1. Edit Main method of Program.cs as give below:

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

LoginForm obj = new LoginForm();

Form frm;

if (obj.ShowDialog() == DialogResult.OK)

{

if (Helper.Role == "Admin")

frm = new MDIMainForm();

else

frm = new ManageAddressbookForm();

Application.Run(frm);

}

}

1. Run the application and for different Login users and manage Contacts…