# 

T.C.

CUMHURİYET ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

# FULL-STACK WEB DEVELOPER/UYGULAMA GELİŞTİRME

## Fahrettin raşit kılıç

## 2015123008

### *Özet*

Bu çalışmada bir adres defteri projesi hazırlanmıştır. Adres defteri projesinde kullanıcı programı çalıştırarak kayıt etmek istediği kişinin adı, soyadı, telefon numarası ve email verilerini girerek kendisine bir adres defteri oluşturmuş olur.

Proje hazırlanırken front- end ve back-en süreçleri dikkate alındı.

Front- end geliştirme süresinde ASP.NET kullanıldı. Proje geliştirilirken ASP.NET ‘in içinde bulunan toolbox(Araç Kutusu) ve XAML dili kullanılarak kullanıcının işlem yapacağı görsel kısım hazırlandı.

Back-End kısmı hazırlanırken CRUD işlemlerini gerçekleştirme için ADO.NET kullanıldı.

Kullanıcı verileri girdikten sonra veri tabanı işlemlerini gerçekleştirebilmek için MSSQL SERVER kullanıldı.

***Anahtar Kelimeler:*** ASP.NET, ADO.NET, CRUD,MSSQL

*Abstract*

In this study, is prepared an address book. The address book project creates an address book by entering the name, surname, phone number and e-mail address of the person who wants to register by running the user program.

We carried front-end and back-end processes.

In front end development in ASP.NET was used in the development. As the project was being developed, a visual section was created to handle ASP.NET's toolbox (Toolbox) and the XAML language.

ADO.NET was used when the back-end was prepared before performing CRUD operations.

MSSQL SERVER was used to store database transactions after user entered data.

***Keywords:*** ASP.NET, ADO.NET, CRUD,MSSQL

## Giriş

Günümüzde arayüz odaklı geliştirilen yazılımlar iki temel kısımdan oluşmaktadır. Bunlar front-end ve back-end kısımlarıdır. Bu iki kısım birbirinden gerek teknoloji gerekse kullanılan araçlar olsun neredeyse birbirinden bağımsız ve izole olarak bir araya gelip bir yazılım ürünü meydana getirmektedirler.

Front-end dediğimiz kısım aslında bir web uygulaması, mobil uygulama ve masaüstü uygulaması yani son kullanıcının gördüğü ve müdahale edebildiği programın önyüzüdür.

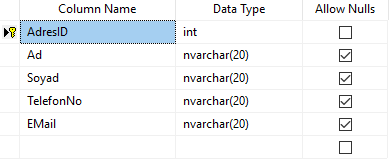
Back-end ise müşterinin göremediği sadece yazılımcının ön yüzden aldığı verilerle uygulamayı geliştirdiği kısımdır.

Bu projede front- end geliştirme aracı olarak ASP.NET kullanılmıştır. Müşterinin göreceği kısımlar XAML dilinden yararlanılmıştır. Back-end kısmında ise ADO.NET kullanarak veritabanı ile iletişim sağlanmıştır. Veritabanı olarak ise MSSQL SERVER kullanılmıştır.

.

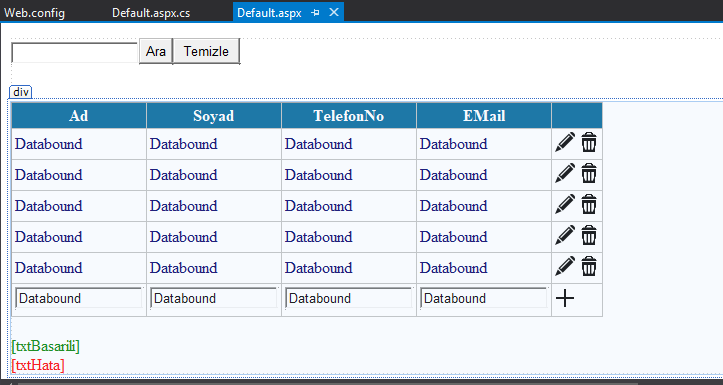
## UYGULAMA

Uygulamanın ilk bölümünde bir MSSQL kullanılarak bir veritabanı oluşturulmuştur. dbAdresDefteri isimli veritabanına Adres adında içerisinde AdresID, Ad, Soyad, TelefonNo, Email kolonları olan bir tablo açılmıştır. Şekil-1 de tablo tasarımı verilmiştir.



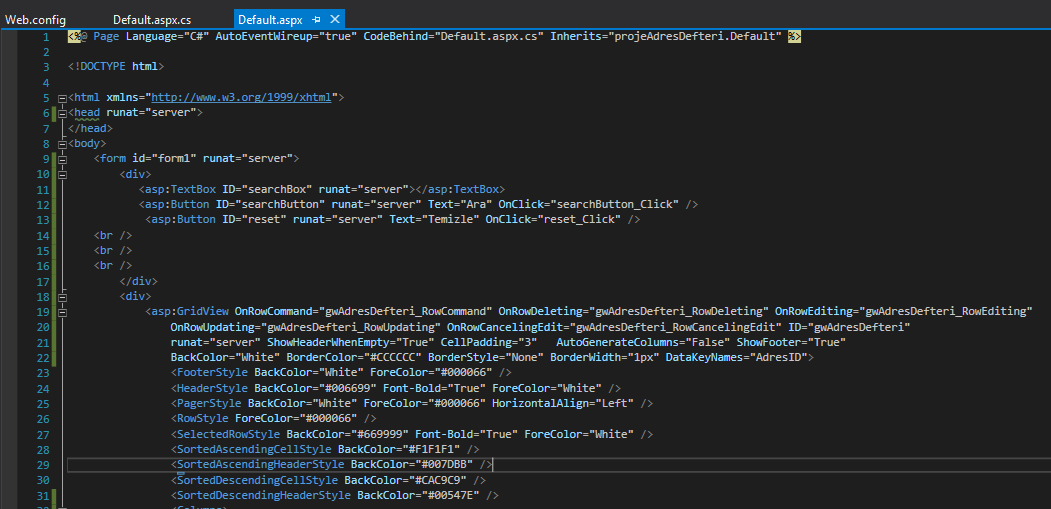
Şekil-1 Veri tabanı Tasarımı

Uygulamanın ikinci bölümünde müşterinin görebileceği kısımlar hazırlanmıştır. Projede verileri göstermek için bir adet Gridview kullanılmıştır. Arama yapmak için ise bir adet Textbox ve 2 adet Button kullanılmıştır. Güncelleme işlemleri sonrasında müşterileri bilgilendirmek için ise 2 adet Label kullanılmıştır. Gridview’ in son kolonunda 5 tane ImageButton CRUD işlemlerini gerçekleştirmek için kullanılmıştır. Şekil-2 de ki görselde front-end tasarımı gösterilmiştir.



Şekil-2 Front-end Tasarımı

Uygulamanın kaynak kod kısmı, ön yüze tasarım aracı eklendikçe kendiliğinden XAML olarak oluşabiliyor. Bu kodlara ek olarak kodla Gridview in üzerine CRUD işlemlerini gerçekleştirilmesi için Click Eventleri atanmıştır. Ayrıca gridview üzerindeki kolonlar bu bölümde hazırlanmıştır. <Column> </Column> etiketleri arasında Şekil-3 de verilmiştir. Projenin XAML kısmı aşağıdaki gibidir.



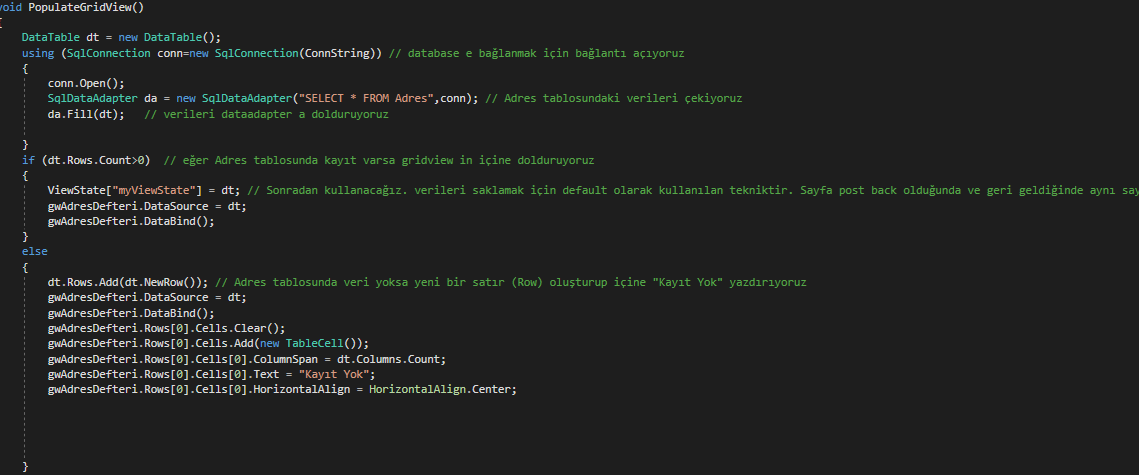


Şekil-3 Front-end kaynak kodları

Front-end tasarımı bittikten sonra Back-end kodları hazırlandı. Veri tabanına bağlanmak için ConnectionString ayarlandı ve ADO.NET kullanımı için gerekli kütüphane eklendi(System.Data.SqlClient).

string ConnString = @"Data Source= DESKTOP-J7VCPEP; Integrated Security=True; Initial Catalog=dbAdresDefteri"

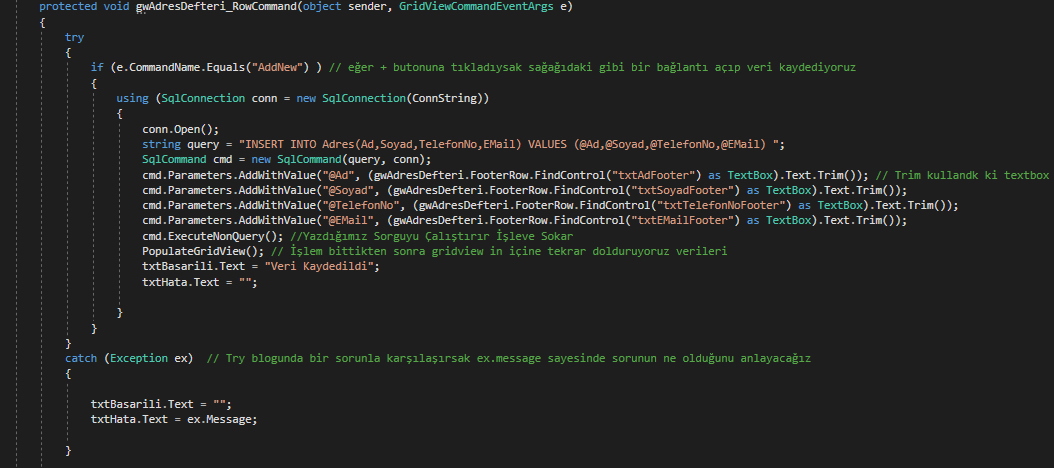
Proje açıldığında veri tabanından kayıtları Gridview getirmesi için Page\_Load içerisinde bir fonksiyon yazıldı. Şekil-4 de verilmiştir



Şekil-4 Veri tabanındaki verilerin ekranda gösterilmesi

Veri getirirken ADO.NET Kütüphanesi kullanılmıştır. SqlDataAdapter kullanarak veriyi veri tabanından alınmıştır ve DataTable içerisine atılmıştır. Veri tabanında kayıt varsa Gridview içine veriler dolacaktır yoksa “Kayıt Yok ” mesajı ile yeni bir kayıt ekleyebilmesi için bir satır oluşturulmuştur.

Veri eklenebilmesi için aşağıdaki kod parçacığı hazırlanmıştır. Şekil-5 de gösterilmektedir. Gridview içine eklediğimiz ImageButtonlara verdiğimiz isimler sayesinde hangi Buttona tıkladığımızı yakalayıp, uygun kodları hazırlayabiliyoruz. Ekleme ImageButtonuna “AddNew” ismi verilmiştir.



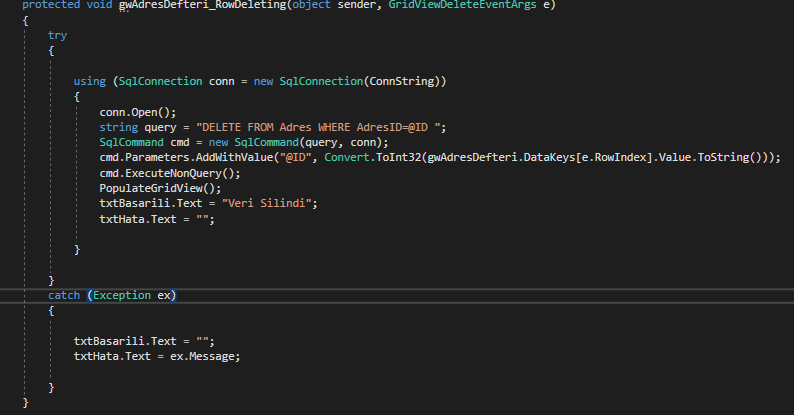
Şekil-5 Verilerin Kaydedilmesi

Seçilen satırın güncellenmesi aşağıdaki Şekil-6 kod bloğunda verilmiştir. Seçilen satırı bulmak için satırın RowIndex özelliğinden faydalanılmıştır. Her veri tabanı işleminde olduğu gibi veri tabanına bağlanılmıştır. Gridview içinde güncelleme yapacağımız satırın verileri parametre olarak veri tabanına gönderilmiştir ve güncelleme olduktan sonra Gridview i doldurmak için oluşturduğumuz PopulateGridView() methodu tekrar çağırılmıştır. İşlemin sonucu Label içerisinde müşteriye gösterilmektedir.

## 

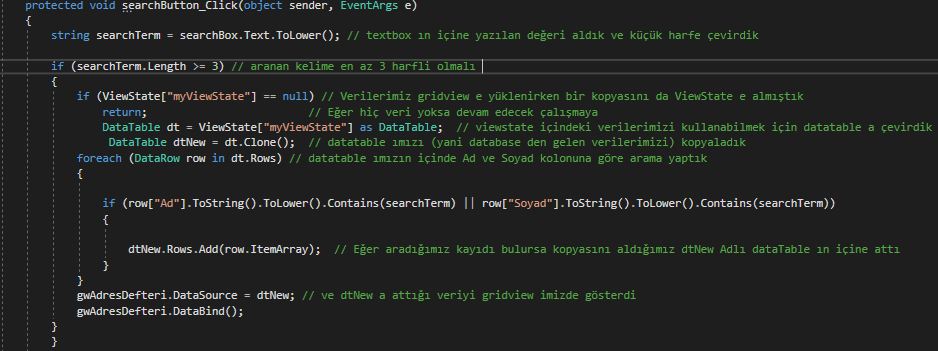
Şekil-6 Verilerin Güncelleme İşlemi

Seçilen satırın silinme işlemi Şekil-7 de verilmiştir. Seçilen satırın bulunmasında yine satırın RowIndex özelliğinden yararlanılmıştır.

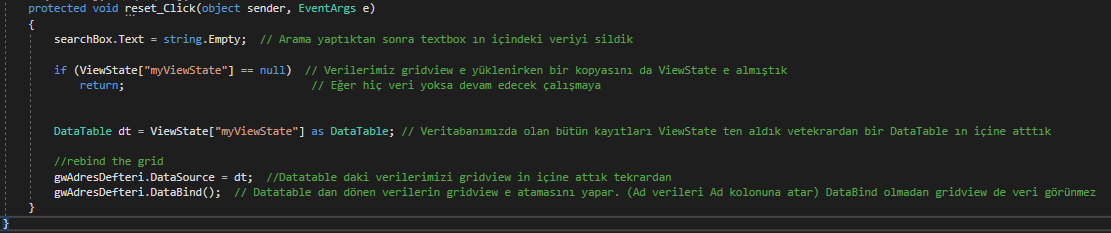


Şekil-7 Verilerin Silinmesi

Verilerin arama işlemi için bir adet Textbox ve 2 adet Button kullanılmıştır. Butonlardan biri Textbox a yazılan verinin aramasını sağlamaktadır. Diğer Button ise Textbox ın içini temizlemek ve Gridview i eski haline çevirmekle görevlidir. Kod parçacığı Şekil-8 ve Şekil-9 da verilmiştir.



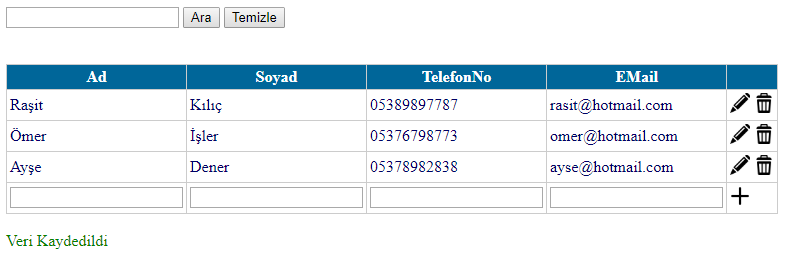
Şekil- 8 Verilerin aranma işlemi



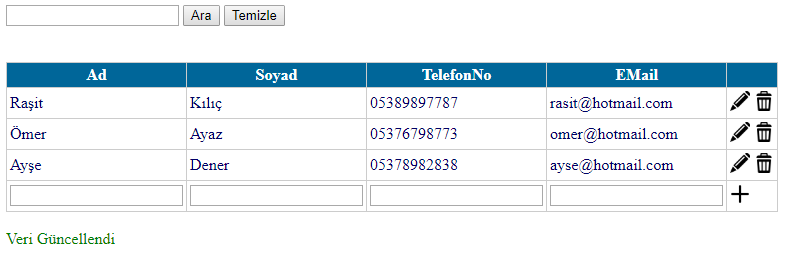
Şekil- 9 Arama işleminin temizlenmesi işlemi

## SONUÇLAR

Ana ekran görüntüsü Şekil- 10 da verilmiştir. 2 adet veri ekledikten sonra görüntü aşağıdaki gibidir.

Şekil-10 Veri ekleme ve gösterme

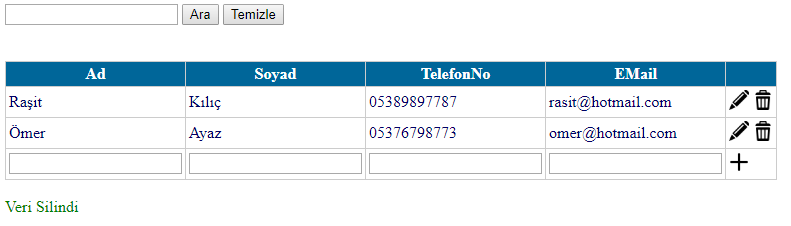
Veri düzenleme işleminden sonraki görüntü Şekil-11 de verilmiştir.

Şekil-11 Veri Güncelleme

Veri arama işlemi Şekil-12 de verilmiştir.



Veri silindikten sonraki ekran görüntüsü Şekil-13 de verilmiştir.



Şekil-13 Veri silme işlemi

## KAYNAK KODLAR

FRONT\_END

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="projeAdresDefteri.Default" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

</head>

<body>

<form id="form1" runat="server">

<div>

<asp:TextBox ID="searchBox" runat="server"></asp:TextBox>

<asp:Button ID="searchButton" runat="server" Text="Ara" OnClick="searchButton\_Click" />

<asp:Button ID="reset" runat="server" Text="Temizle" OnClick="reset\_Click" />

<br />

<br />

<br />

</div>

<div>

<asp:GridView OnRowCommand="gwAdresDefteri\_RowCommand" OnRowDeleting="gwAdresDefteri\_RowDeleting" OnRowEditing="gwAdresDefteri\_RowEditing"

OnRowUpdating="gwAdresDefteri\_RowUpdating" OnRowCancelingEdit="gwAdresDefteri\_RowCancelingEdit" ID="gwAdresDefteri"

runat="server" ShowHeaderWhenEmpty="True" CellPadding="3" AutoGenerateColumns="False" ShowFooter="True"

BackColor="White" BorderColor="#CCCCCC" BorderStyle="None" BorderWidth="1px" DataKeyNames="AdresID">

<FooterStyle BackColor="White" ForeColor="#000066" />

<HeaderStyle BackColor="#006699" Font-Bold="True" ForeColor="White" />

<PagerStyle BackColor="White" ForeColor="#000066" HorizontalAlign="Left" />

<RowStyle ForeColor="#000066" />

<SelectedRowStyle BackColor="#669999" Font-Bold="True" ForeColor="White" />

<SortedAscendingCellStyle BackColor="#F1F1F1" />

<SortedAscendingHeaderStyle BackColor="#007DBB" />

<SortedDescendingCellStyle BackColor="#CAC9C9" />

<SortedDescendingHeaderStyle BackColor="#00547E" />

<Columns>

<asp:TemplateField HeaderText="Ad">

<ItemTemplate>

<asp:Label Text='<%# Eval("Ad") %>' runat="server"></asp:Label>

</ItemTemplate>

<EditItemTemplate>

<asp:TextBox ID="txtAd" Text='<%# Eval("Ad") %>' runat="server" />

</EditItemTemplate>

<FooterTemplate>

<asp:TextBox ID="txtAdFooter" Text='<%# Eval("Ad") %>' runat="server"> </asp:TextBox>

</FooterTemplate>

</asp:TemplateField>

<asp:TemplateField HeaderText="Soyad">

<ItemTemplate>

<asp:Label Text='<%# Eval("Soyad") %>' runat="server"></asp:Label>

</ItemTemplate>

<EditItemTemplate>

<asp:TextBox ID="txtSoyad" Text='<%# Eval("Soyad") %>' runat="server"> </asp:TextBox>

</EditItemTemplate>

<FooterTemplate>

<asp:TextBox ID="txtSoyadFooter" Text='<%# Eval("Soyad") %>' runat="server"> </asp:TextBox>

</FooterTemplate>

</asp:TemplateField>

<asp:TemplateField HeaderText="TelefonNo">

<ItemTemplate>

<asp:Label Text='<%# Eval("TelefonNo") %>' runat="server"></asp:Label>

</ItemTemplate>

<EditItemTemplate>

<asp:TextBox ID="txtTelefonNo" Text='<%# Eval("TelefonNo") %>' runat="server"> </asp:TextBox>

</EditItemTemplate>

<FooterTemplate>

<asp:TextBox ID="txtTelefonNoFooter" Text='<%# Eval("TelefonNo") %>' runat="server"> </asp:TextBox>

</FooterTemplate>

</asp:TemplateField>

<asp:TemplateField HeaderText="EMail">

<ItemTemplate>

<asp:Label Text='<%# Eval("EMail") %>' runat="server"></asp:Label>

</ItemTemplate>

<EditItemTemplate>

<asp:TextBox ID="txtEMail" Text='<%# Eval("EMail") %>' runat="server"> </asp:TextBox>

</EditItemTemplate>

<FooterTemplate>

<asp:TextBox ID="txtEMailFooter" Text='<%# Eval("EMail") %>' runat="server"> </asp:TextBox>

</FooterTemplate>

</asp:TemplateField>

<asp:TemplateField>

<ItemTemplate>

<asp:ImageButton ImageUrl="~/Images/edit.png" runat="server" CommandName="Edit" ToolTip="Edit" Width="20px" Height="20px"/>

<asp:ImageButton ImageUrl="~/Images/delete.png" runat="server" CommandName="Delete" ToolTip="Delete" Width="20px" Height="20px"/>

</ItemTemplate>

<EditItemTemplate>

<asp:ImageButton ImageUrl="~/Images/save.png" runat="server" CommandName="Update" ToolTip="Update" Width="20px" Height="20px"/>

<asp:ImageButton ImageUrl="~/Images/cancel.png" runat="server" CommandName="Cancel" ToolTip="Cancel" Width="20px" Height="20px"/>

</EditItemTemplate>

<FooterTemplate>

<asp:ImageButton ImageUrl="~/Images/add.png" runat="server" CommandName="AddNew" ToolTip="Add New" Width="20px" Height="20px"/>

</FooterTemplate>

</asp:TemplateField>

</Columns>

</asp:GridView>

<br />

<asp:Label runat="server" Text="" ForeColor="Green" ID="txtBasarili"></asp:Label>

<br />

<asp:Label runat="server" Text="" ForeColor="Red" ID="txtHata"></asp:Label>

</div>

</form>

</body>

</html>

BACK-END

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace projeAdresDefteri

{

public partial class Default : System.Web.UI.Page

{

string ConnString = @"Data Source= DESKTOP-J7VCPEP; Integrated Security=True; Initial Catalog=dbAdresDefteri" ; //database bağlantılarını ayarladık

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack) // eğer sayfa ilk kez açılıyorsa gridview i doldur. Eğer ilk defa sayfa yüklenmiyorsa devam et. Gridview in içindeki elemanların çoklanmasını önler.

{

PopulateGridView();

}

}

void PopulateGridView()

{

DataTable dt = new DataTable();

using (SqlConnection conn=new SqlConnection(ConnString)) // database e bağlanmak için bağlantı açıyoruz

{

conn.Open();

SqlDataAdapter da = new SqlDataAdapter("SELECT \* FROM Adres",conn); // Adres tablosundaki verileri çekiyoruz

da.Fill(dt); // verileri dataadapter a dolduruyoruz

}

if (dt.Rows.Count>0) // eğer Adres tablosunda kayıt varsa gridview in içine dolduruyoruz

{

ViewState["myViewState"] = dt; // Sonradan kullanacağız. verileri saklamak için default olarak kullanılan tekniktir. Sayfa post back olduğunda ve geri geldiğinde aynı sayfadaki değerlerin saklanması işlemini ViewState yapar.

gwAdresDefteri.DataSource = dt;

gwAdresDefteri.DataBind();

}

else

{

dt.Rows.Add(dt.NewRow()); // Adres tablosunda veri yoksa yeni bir satır (Row) oluşturup içine "Kayıt Yok" yazdırıyoruz

gwAdresDefteri.DataSource = dt;

gwAdresDefteri.DataBind();

gwAdresDefteri.Rows[0].Cells.Clear();

gwAdresDefteri.Rows[0].Cells.Add(new TableCell());

gwAdresDefteri.Rows[0].Cells[0].ColumnSpan = dt.Columns.Count;

gwAdresDefteri.Rows[0].Cells[0].Text = "Kayıt Yok";

gwAdresDefteri.Rows[0].Cells[0].HorizontalAlign = HorizontalAlign.Center;

}

}

protected void gwAdresDefteri\_RowCommand(object sender, GridViewCommandEventArgs e)

{

try

{

if (e.CommandName.Equals("AddNew") ) // eğer + butonuna tıkladıysak sağağıdaki gibi bir bağlantı açıp veri kaydediyoruz

{

using (SqlConnection conn = new SqlConnection(ConnString))

{

conn.Open();

string query = "INSERT INTO Adres(Ad,Soyad,TelefonNo,EMail) VALUES (@Ad,@Soyad,@TelefonNo,@EMail) ";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@Ad", (gwAdresDefteri.FooterRow.FindControl("txtAdFooter") as TextBox).Text.Trim()); // Trim kullandk ki textbox in içinde boş karakter girdiysek temizlesin

cmd.Parameters.AddWithValue("@Soyad", (gwAdresDefteri.FooterRow.FindControl("txtSoyadFooter") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@TelefonNo", (gwAdresDefteri.FooterRow.FindControl("txtTelefonNoFooter") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@EMail", (gwAdresDefteri.FooterRow.FindControl("txtEMailFooter") as TextBox).Text.Trim());

cmd.ExecuteNonQuery(); //Yazdığımız Sorguyu Çalıştırır İşleve Sokar

PopulateGridView(); // İşlem bittikten sonra gridview in içine tekrar dolduruyoruz verileri

txtBasarili.Text = "Veri Kaydedildi";

txtHata.Text = "";

}

}

}

catch (Exception ex) // Try blogunda bir sorunla karşılaşırsak ex.message sayesinde sorunun ne olduğunu anlayacağız

{

txtBasarili.Text = "";

txtHata.Text = ex.Message;

}

}

protected void gwAdresDefteri\_RowEditing(object sender, GridViewEditEventArgs e)

{

gwAdresDefteri.EditIndex = e.NewEditIndex; // se.tiğimiz satır üzerinden update yada cancel gibi işlemlerimize izin verecek. Satırı aktif hala getirecek

PopulateGridView();

}

protected void gwAdresDefteri\_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)

{

gwAdresDefteri.EditIndex = -1; // gridview üzerinde değişiklik yapmaktan vazgeçersek gridview i eski haline geri dönderiyoruz.

PopulateGridView(); // gridview i tekrar yükledik

}

protected void gwAdresDefteri\_RowUpdating(object sender, GridViewUpdateEventArgs e)

{

try

{

using (SqlConnection conn = new SqlConnection(ConnString)) // database bağlantı ayarlarını yaptık

{

conn.Open();

string query = "UPDATE Adres SET Ad=@Ad,Soyad=@Soyad,TelefonNo=@TelefonNo,EMail=@EMail WHERE AdresID=@ID ";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@Ad", (gwAdresDefteri.Rows[e.RowIndex].FindControl("txtAd") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@Soyad", (gwAdresDefteri.Rows[e.RowIndex].FindControl("txtSoyad") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@TelefonNo", (gwAdresDefteri.Rows[e.RowIndex].FindControl("txtTelefonNo") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@EMail", (gwAdresDefteri.Rows[e.RowIndex].FindControl("txtEMail") as TextBox).Text.Trim());

cmd.Parameters.AddWithValue("@ID",Convert.ToInt32(gwAdresDefteri.DataKeys[e.RowIndex].Value.ToString()));

cmd.ExecuteNonQuery(); // Update sorgusunu çalıştırdık ve işlettik

gwAdresDefteri.EditIndex = -1; // İşlem yaptığımız satırdan çıkabilmek için edit index i -1 yaptık

PopulateGridView();

txtBasarili.Text = "Veri Güncellendi";

txtHata.Text = "";

}

}

catch (Exception ex)

{

txtBasarili.Text = "";

txtHata.Text = ex.Message;

}

}

protected void gwAdresDefteri\_RowDeleting(object sender, GridViewDeleteEventArgs e)

{

try

{

using (SqlConnection conn = new SqlConnection(ConnString))

{

conn.Open();

string query = "DELETE FROM Adres WHERE AdresID=@ID ";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@ID", Convert.ToInt32(gwAdresDefteri.DataKeys[e.RowIndex].Value.ToString()));

cmd.ExecuteNonQuery();

PopulateGridView();

txtBasarili.Text = "Veri Silindi";

txtHata.Text = "";

}

}

catch (Exception ex)

{

txtBasarili.Text = "";

txtHata.Text = ex.Message;

}

}

protected void searchButton\_Click(object sender, EventArgs e)

{

string searchTerm = searchBox.Text.ToLower(); // textbox ın içine yazılan değeri aldık ve küçük harfe çevirdik

if (searchTerm.Length >= 3) // aranan kelime en az 3 harfli olmalı

{

if (ViewState["myViewState"] == null) // Verilerimiz gridview e yüklenirken bir kopyasını da ViewState e almıştık

return; // Eğer hiç veri yoksa devam edecek çalışmaya

DataTable dt = ViewState["myViewState"] as DataTable; // viewstate içindeki verilerimizi kullanabilmek için datatable a çevirdik

DataTable dtNew = dt.Clone(); // datatable ımızı (yani database den gelen verilerimizi) kopyaladık

foreach (DataRow row in dt.Rows) // datatable ımızın içinde Ad ve Soyad kolonuna göre arama yaptık

{

if (row["Ad"].ToString().ToLower().Contains(searchTerm) || row["Soyad"].ToString().ToLower().Contains(searchTerm))

{

dtNew.Rows.Add(row.ItemArray); // Eğer aradığımız kayıdı bulursa kopyasını aldığımız dtNew Adlı dataTable ın içine attı

}

}

gwAdresDefteri.DataSource = dtNew; // ve dtNew a attığı veriyi gridview imizde gösterdi

gwAdresDefteri.DataBind();

txtBasarili.Text = "Veri Bulundu";

}

}

protected void reset\_Click(object sender, EventArgs e)

{

searchBox.Text = string.Empty; // Arama yaptıktan sonra textbox ın içindeki veriyi sildik

if (ViewState["myViewState"] == null) // Verilerimiz gridview e yüklenirken bir kopyasını da ViewState e almıştık

return; // Eğer hiç veri yoksa devam edecek çalışmaya

DataTable dt = ViewState["myViewState"] as DataTable; // Veritabanımızda olan bütün kayıtları ViewState ten aldık vetekrardan bir DataTable ın içine atttık

//rebind the grid

gwAdresDefteri.DataSource = dt; //Datatable daki verilerimizi gridview in içine attık tekrardan

gwAdresDefteri.DataBind(); // Datatable dan dönen verilerin gridview e atamasını yapar. (Ad verileri Ad kolonuna atar) DataBind olmadan gridview de veri görünmez

}

}

}

VERİTABANI

USE [dbAdresDefteri]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Adres](

[AdresID] [int] IDENTITY(1,1) NOT NULL,

[Ad] [nvarchar](20) NULL,

[Soyad] [nvarchar](20) NULL,

[TelefonNo] [nvarchar](20) NULL,

[EMail] [nvarchar](20) NULL,

CONSTRAINT [PK\_Adres] PRIMARY KEY CLUSTERED

(

[AdresID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO