**C# Veri Tabanı Kullanarak Login**

C# ile OOP üzerine belli bir seviyeye geldik ve artık bir adım ileri giderek veri tabanı kullanarak uygulamalar geliştirmeye başlıyoruz. Bu yazıyı okumadan önce lütfen veri tabanı nedir, CRUD denilen temel SQL komutlarını ve veri tabanı ile tablo oluşturma, veri türleri üzerine çalışmalarınızı tamamlayınız. Biz bu yazımızda bunları biliyorsunuz kabul ederek C# ile veri tabanı bağlantısı nasıl yapılır, SQL komutları nasıl yazılır, yazılan SQL komutları nasıl işlenir ve veri tabanından aldığımız veri nasıl kullanılır bunları bir login senaryosu üzerinden anlatacağız.

Senaryomuz:

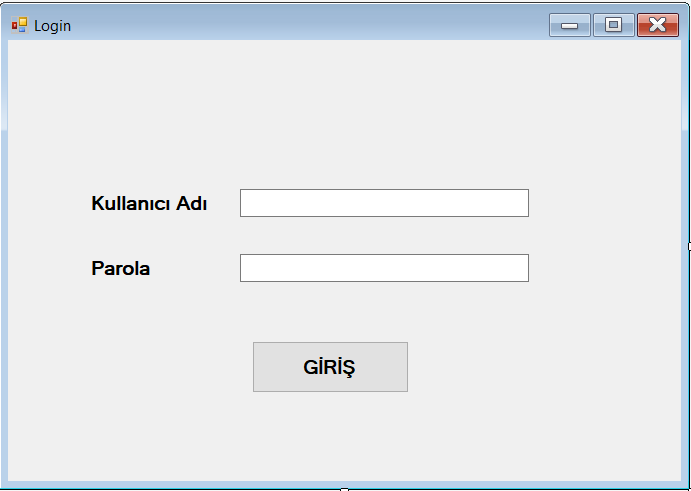
- Veri tabanında kullanıcıya ait kullanıcı adı, parola, yetkisi, durumu ve giriş hakkı olarak bilgilerini tutacağız.

- Kullanıcı giriş yapmaya çalıştığında veri tabanından bilgileri alarak kontrol edeceğiz ve ikisi de doğru ve aynı zamanda durumu aktif ise sisteme giriş yapacak, durumu kilitli ise giriş yapamayacaktır.

- Eğer kullanıcı adı doğru ancak parola yanlış ise giriş hakkı bir azaltılacak, giriş sağlanmayacak ve kullanıcı uyarılacaktır. 3 kere hatalı giriş olursa durumu kilitli yapılacak ve girişi engellenecektir. Eğer Hata yaptıktan sonra doğru giriş sağlanırsa kullanıcının giriş hakkı yeniden 3 yapılacaktır.

1. Form’un Tasarlanması

Uygulamamızda iki adet form kullanacağız. Form1’i giriş ekranımız olarak tasarlayacağız; Form2’yi ise login işlemi başarılı olduktan sonra yönlendireceğimiz form olarak tasarlayacağız. Form1’i tasarlarken basit bir düzen olarak iki adet label, iki adet textbox ve bir adet button kullanacağız. Genel olarak tasarımımız şu şekilde olacaktır:

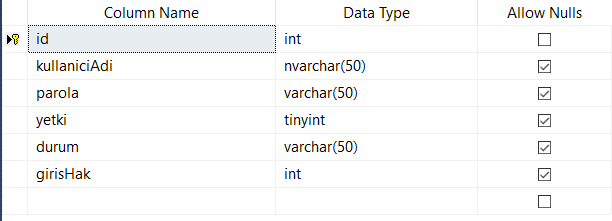


Dilerseniz gerçekçi olması açısından parolayı yazacağımız textbox’ın UseSystemPassword özelliğini Properties kısmından true yapabilirsiniz.

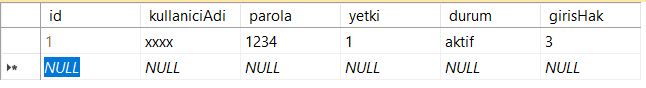
Form2 ile bir işimiz olmadığı için bir tasarım planlamadık. Login işlemi başarılı olduğunda Form2 açılacak ve işlemimiz bitmiş olacak.

2. Veri Tabanının Tasarlanması

Biz uygulamamızda Microsoft SQL Server 2012 kullanacağız. Eğer farklı bir veri tabanıyla çalışıyorsanız buradaki yazıyı okumaya devam edin. Burada temel mantığı öğreneceksiniz. Ardından kısa bir araştırmayla kullanacağınız veri tabanına entegre olabilirsiniz.

 Uygulamamızda tek bir tablo oluşturacağız. Tablomuzun design kısmı aşağıdaki gibi olacaktır:

‘id’ sütununu primary key ve column properties kısmından sütunun Identity Specification kısmını true yapmayı unutmayın. Bizim uygulamamızda yapmasak da bir şey olmaz ancak ileride ilişkili tablolar üzerine çalışırken işimize yarayacak. Temel olarak id sütununu birincil anahtar olarak belirleyip otomatik artan olarak seçtiğimizi bilsek yeterli.

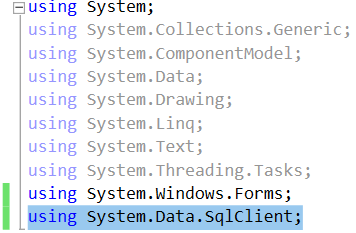
 Daha sonra tablomuzun üzerinde sağ tıklayarak Edit Top 200 Rows diyoruz ve kontrol amaçlı kullanacağımız değerleri tablomuza giriyoruz. Daha sonra kendinizce senaryoyu genişleterek yetkiye göre kullanıcı ekleme işlemi de yapabilirsiniz. Ancak biz bu uygulamamızda yapmayacak, temel veri tabanı işlemleri mantığını çalışacağız. Aşağıdakine benzer bir tablonuzun olması yeterlidir.

3. Veri Tabanının Tanıtılması ve Bağlanması

Bu bölümde tasarladığımız veri tabanını programımıza tanıtacak ve bağlantımızı gerçekleştireceğiz.

İlk olarak kısaca temel mantığından bahsedecek olursak;

* Veri tabanı için gerekli kütüphanelerin yüklenmesi,
* Connection string’i ile veri tabanımızın yolunu tanımlayıp bağlantının gerçekleştirilmesi.

 Form1.cs’yi açarak veri tabanımız için gerekli olan kütüphanemizi ekliyoruz:

Şimdi sıra connection string’den bahsedebiliriz. Connecion string’i veri tabanımıza bağlanmak için programa veri tabanımızın yolunun bildirilmesidir diyebiliriz.

 Genel olarak bir connection string’in yapısı aşağıdaki gibidir:

Buradaki @ işareti bir dosya yolu bildireceğimiz anlamını taşır. Data Source kısmı veri tabanımızın nerede olduğunun bilgisini taşır. Initial Catalog kısmı veri tabanımızın adını belirtir. Integrated Security kısmını true yaparak User Id ve password girmeden veri tabanına bağlanmayı sağlıyoruz.

Daha sonra SqlConnection sınıfından bir nesne oluşturarak bağlantımızı sağlıyoruz:

SqlConnection baglanti = new SqlConnection(veriYolu);

Artık baglanti.Open() dediğimiz an veri tabanına bağlanmış olacağız. Bir sonraki adımda ise veri tabanına bağlandıktan sonra sorgu yazmayı ve bu sorgu neticesinde elde ettiğimiz veriyi kullanmayı göreceğiz.

4. SqlCommand Sınıfı

Veri tabanına bağlandıktan sonra sıradaki işlemlerimiz veriyi alma ve veriyi manipüle etmek olacaktır. Bunun için SqlCommand sınıfından yararlanarak sorgularımızı oluşturacak ve ilgili metotlarla işlemimizi gerçekleştireceğiz. Temel mantığımızı şu iki başlık altında oturtacağız: a. insert, update, delete işlemleri b. select işlemleri

4.a. Insert,update ve delete işlemleri:

Bu işlemler için bir sorgu string’i tanımlayacağız, SqlCommand sınıfından bir komut nesnesi türetecek ve yazdığımız sorgu ve ilk baştaki bağlantımızı parametre olarak vereceğiz. Ardından komut nesnemizin ExecuteNonQuery() metodunu kullanarak sorgumuzu çalıştırıp veri tabanımızı manipüle edeceğiz. Örnek kodlarımızı açıklama satırlarıyla beraber paylaşacağız ve oradan incelediğinizde anlamanız kolay olacaktır.

4.b. Select işlemi:

Malumunuz ver tabanımızdan veri alıp bu veriyi kullanmak için select komutundan yararlanırız. Veriye ulaşmak için yine bir sorgu string’i tanımlayıp yukarıdaki gibi SqlCommand sınıfından nesne oluşturarak veri tabanında bir işlem gerçekleştireceğimizi bildiriyoruz. Şimdi burada ulaştığımız verileri geçici olarak saklamak için bir yardımcıya ihtiyacımız var. Biz bu yardımı DataReader sınıfından alacağız.

baglanti.Open();

string sorgu = "SELECT kullaniciAdi,parola,durum,girisHak FROM T\_Kullanicilar";

SqlCommand komut = new SqlCommand(sorgu,baglanti);//Baglantimizi kurduk.Simdi sorguyu calistirip veriyi almamiz lazim.

//Komutu gerçekleştirdik.

SqlDataReader dr = komut.ExecuteReader();

Yukarıdaki komutlarla tablomuzdan aldığımız veriyi bir ‘dr’ nesnesinin içine atadık. Bundan sonra bir while döngüsüyle aldığımız veriyi istediğimiz gibi kullanabiliriz.

Genel mantığımız bu şekilde olacaktır. Kaynak kodları detaylı bir şekilde inceleyerek bu mantığı oturtup daha rahat çalışmalar yapabilirsiniz. Tavsiyem kafanızın takıldığı yere breakpoint koyarak programı takip etmeniz yararınıza olacaktır.

Kaynak Kodlar:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*

\*\* YAZAN.............: Emre Harman

\*\* TARİH.............: 12.05.2020

\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace \_02

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

//Connection stringimiz ve bağlantımız

const string veriYolu = @"Data Source=DESKTOP-QH1AV1B; Initial Catalog=dboCalisma2; Integrated Security=True";

SqlConnection baglanti = new SqlConnection(veriYolu);

private void Form1\_Load(object sender, EventArgs e)

{

}

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

{

//Kullanıcının girdiği değerler

string kulAd = txtKullaniciAdi.Text;

string parola = txtParola.Text;

//Bağlantımızı açtık

baglanti.Open();

string sorgu = "SELECT kullaniciAdi,parola,durum,girisHak FROM T\_Kullanicilar";//Sordu cümlemizi yazdık.

SqlCommand komut = new SqlCommand(sorgu,baglanti);//Baglantimizi kurduk.Simdi sorguyu calistirip veriyi almamiz lazim.

//Komutu gerçekleştirdik.

SqlDataReader dr = komut.ExecuteReader();

//Tablomuzu satır satır okuyup verilemizi kontrol edeceğiz.

while (dr.Read())

{

//Eğer alanlardan biri ya da ikisi birden boş ise

if(kulAd=="" || parola == "")

{

MessageBox.Show("Kullanıcı adı ya da Parola boş bırakılamaz.");

baglanti.Close();

break;

}

//Kullanıcı adı ve parola doğru girilmiş ve aynı zamanda kullanıcının durumu aktif ise form2'ye yönlendir.

//Kullanıcının başarılı giriş yaptığı için her ihtimale karşı giriş hakkını yeniden 3 yap.

if (dr["kullaniciAdi"].ToString() == kulAd && dr["parola"].ToString()==parola && dr["durum"].ToString()=="aktif")

{

Form2 form2 = new Form2();

form2.Show();

this.Close();

dr.Close();

komut.CommandText = "UPDATE T\_Kullanicilar SET girisHak=3 WHERE kullaniciAdi='"+kulAd+"' ";

komut.ExecuteNonQuery();

baglanti.Close();

break;

}

//Kullanıcı adı ve parola doğru ancak durumu kilitli ise hata ver, tekrar girişi engelle.

if(dr["kullaniciAdi"].ToString() == kulAd && dr["parola"].ToString() == parola && dr["durum"].ToString() == "kilitli")

{

MessageBox.Show("Hesabınız kilitlidir. Lütfen sistem yöneticinizle iletişime geçin.");

btnGiris.Enabled = false;

baglanti.Close();

break;

}

//Kullanıcı adı ya da parola yanlışsa

if(dr["kullaniciAdi"].ToString()!=kulAd || dr["parola"].ToString() != parola)

{

//KUllanıcının durumu kilitli ise girişini engelle.

if (dr["durum"].ToString() == "kilitli")

{

MessageBox.Show("Hesabınız kilitlidir. Lütfen sistem yöneticinizle iletişime geçin.");

btnGiris.Enabled = false;

baglanti.Close();

break;

}

dr.Close();//Yeni komut yazmak için açık olan DataReader'ı kapatmamız lazım.

//Kullanıcının giriş hakkını 1 azalt, veri tabanında güncelle

komut.CommandText = "UPDATE T\_Kullanicilar SET girisHak=girisHak-1";

komut.ExecuteNonQuery();

komut.CommandText = "SELECT \* FROM T\_Kullanicilar WHERE kullaniciAdi='"+kulAd+"'";

dr = komut.ExecuteReader();//Güncel bilgiyi tekrar oku.

//Güncel bilgiyi okuyoruz.

while (dr.Read())

{

//Kullanıcının kalan hakkını bildirdik.

MessageBox.Show("Kullanıcı adınız ya da Parolanız yanlış. Kalan hak "+dr["girisHak"].ToString());

//giriş hakkı bittiyse durumunu kilitli yaparak girişini engelliyoruz.

if (dr["girisHak"].ToString() == "0")

{

dr.Close();

komut.CommandText = "UPDATE T\_Kullanicilar SET durum='kilitli'";

komut.ExecuteNonQuery();

MessageBox.Show("Hesabınız kilitlenmiştir. Lütfen sistem yöneticinizle iletişime geçin.");

btnGiris.Enabled = false;

txtKullaniciAdi.Clear();

txtParola.Clear();

baglanti.Close();

break;

}

}

txtKullaniciAdi.Clear();

txtParola.Clear();

baglanti.Close();

break;

}

}

}

}

}