



به نام خدا

محمد مهدی معتمدی

400213022 | m242mm242@gmail.com

- پروژه آز پایگاه داده با موضوع نرم افزار مدیریت املاک (کاربری بنگاه های املاک).
- لینک دریافت پروژه در انتهای این PDF قرار دارد.

هدف کلی پروژه:

این نرم افزار هدف تسهیل کردن کار دفاتر املاک در جست و جو افراد و املاک داراست. اما نکته این کار این است با توجه به این موضوع که هدف از انجام پروژه تحویل پروژه برای درس آز پایگاه است به انجام چند سناریو بسنده می کنیم.

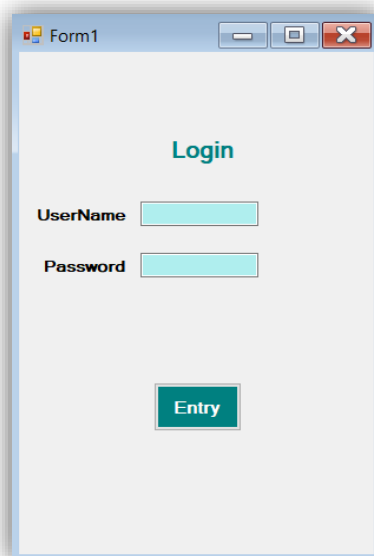
مزایا پروژه:

- صرفه جویی در زمان برای جستجو به دنبال شخص مورد نظر در لیست.
 - مشاهده توضیحات اضافی برای هر مورد خریدار، فروشنده و اجاره دهنده و اجاره کننده.
 - اعمال فیلتر هایی بر روی اشخاص ذخیره شده برای مثال با جست و جو در قسمت ذخیره شده اشخاص در قسمت خریدار همه خریدار را می توانیم ببینیم.
 - برنامه می تواند بین ملک و افراد تفاوت ایجاد کند به این معنی که در قسمتی که به ذخیره اطلاعات می پردازیم تفاوت بین افراد که تقاضای خرید و اجاره دارند و همینطور املاک مربوطه داشته باشیم.
- کاربر ما در این سناریو نه خریدار و نه فروشنده و نه رهن کننده است بلکه منظور از کاربر شخص املاکی است.

سناریوها:

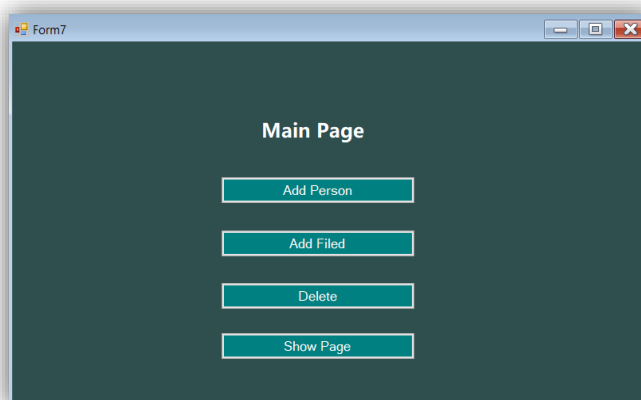
1. کاربر سیستم را بوت کرده و یوزر و پس خود را وارد کرده و دسترسی او به پایگاه داده اپلیکیشن فراهم می‌شود.

User Name: Diacko and The Password: Diacko242



2. سناریو ذخیره یک خریدار:

a. با مراجعه شخص خریدار به کاربر، کاربر گزینه ذخیره شخص را انتخاب می‌کند.



b. پنل مربوط به ذخیره شخص باز شود (Add Person) و پس از گرفتن اطلاعات شخص شامل شماره و نام و اطلاعاتی شامل

بودجه و extra information را کاربر وارد می‌کند.

c. کاربر نوع ذخیره سازی شخص را تعیین کند برای مثال گزینه خریدار را در انتهای پنل بفشارد (buyer).

d. در نهایت اطلاعات وارد شده را ذخیره کند و در پایگاه داده این اطلاعات ذخیره شود.

3. سناریو ذخیره یک اجاره کننده:

a. همچون سناریو قبلی اتفاق بیوفتد ولی در انتخاب نوع شخص گزینه اجاره کننده را بزند.

4. سناریو ذخیره یک ملک فروشی:

a. با مراجعه فروشنده به کاربر، کاربر در main page بر روی گزینه اضافه کردن یک ملک کلیک کرده

5. فرم مورد نظر به او نشان داده می شود و کاربر با دریافت اطلاعات فروشنده و تعیین نوع ملک فروشی یا اجاره ای (در واقع

اطلاعات شخص فروشنده به همراه اطلاعات ملک) و سپس ذخیره کردن اطلاعات در پایگاه داده کار به اتمام برسد.

6. سناریو حذف یک شخص یا ملک:

a. اگر به هر دلیلی کاربر تصمیم به حذف یک مورد در پایگاه داده چه ملک چه شخص داشته باشد کفایت با در اختیار داشتن

اطلاعات اصلی شخص از طریق main page یا پنل اصلی وارد قسمت حذف با دلالت بشود.

- b. پس از ان اطلاعات خواسته شده را وارد می کند.
- c. در نهایت نوع فیلد مربوطه خریدار یا فروشنده یا زمین فروشی و... را انتخاب می کند.
- d. با زدن دکمه سابمیت در صورت وجود داشتن اطلاعات ردیف مربوطه به اطلاعات موردنظر در دیتابیس حذف می شود.
- e. در صورت بروز خطا، اعلام خطا را چاپ کند.

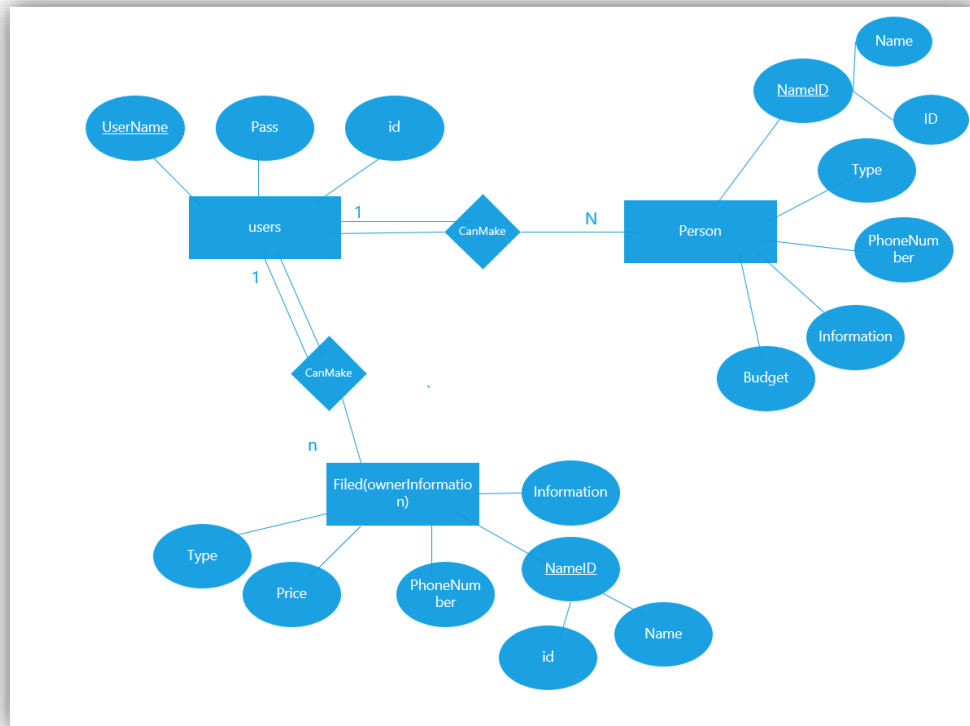
7. سناریو مشاهده اطلاعات:

- a. در صورت نیاز به مشاهده اطلاعات وارد قسمت show page می شویم که در آن برای مشاهده اطلاعات مورد نظر باید بر روی کومبو باکس آن کلیک کرد.

باید بتوان اطلاعات قابل نمایش را باهم مقایسه کرد برای مثال فروشنده ها را با خریدار ها با هم در اختیار داشته باشیم و در کنار یکدیگر بتوانیم به راحتی آنها را مقایسه کنیم.

طراحی منطقی یا ERD:

- مدل طراحی شده اولیه برای نرم افزار مورد نظر به این شکل بوده است:



- طراحی به این شکل است و مزایای خاص خود را دارد.

- در این طراحی تعداد جداول کمی داریم در صورت محدود بودن فضای ذخیره سازی این مورد مهمی است.
- دوم اینکه در این مدل به دلیل کم شدن جداول Data Redundancy کمتری خواهیم داشت.
- سوم اینکه در این مدل برای جست و جو در پایگاه داده سربار پردازشی کمتری نسبت به مدل بعدی خواهیم داشت.

- از معایب این مدل می توان به موارد زیر اشاره کرد:

- به دلیل کم بودن جداول امکان جست و جو کارآمد در جداول نیست مثلاً برای پیدا کردن فردی که هم خریدار است هم اجاره کننده جست و جو در این مدل کار سختی می باشد.
- برای قسمت Delete حتماً باید مقدار ای دی و نام فرد مورد نظر را برای حذف کردن بدانیم.
- پیاده سازی با پرفورمنس پایین تر.

طراحی مدل فیزیکی پایگاه داده (پیاده سازی شده در پروژه اصلی):

با بررسی های انجام شده پایگاه داده در حدی باشد که بتواند ذخیره کند که کدام کاربر کدامین فیلد ها را یا کدامین اشخاص را ذخیره کرده کافیسست از طرفی باید این اختیار وجود داشته باشد که یک شخص بتواند هم به عنوان خریدار هم به عنوان فروشنده هم به عنوان رهن کننده در نظر گرفته شود اما در مدل ER ذکر شده اگر کلید جدول شخص را نام در نظر بگیریم این اتفاق نمی تواند رخ دهد (به دلیل سختی مقدار دهی جداول پس از ساختن آنها جداول را ساده تر ساختیم).

✚ پس در نهایت به این مدل نهایی برای جداول رسیدیم:

User:

<u>username</u>	Password	id
-----------------	----------	----

Buyer:

<u>Name</u>	Phoner Number	budget	Creator ID	Information
-------------	---------------	--------	------------	-------------

Tenant:

<u>Name</u>	Phone Number	budget	Information	Creator ID	<u>id</u>
-------------	--------------	--------	-------------	------------	-----------

Seller:

<u>Name</u>	Phone Number	Price	Information	Creator ID	<u>id</u>
-------------	--------------	-------	-------------	------------	-----------

Land Lord:

<u>Name</u>	Phone Number	Price	Information	Creator ID	<u>id</u>
-------------	--------------	-------	-------------	------------	-----------

ایدی فقط در سناریو هایی استفاده می شود که نیاز باشد یک فرد را دوبار در یک جدول ذخیره کرد در پروژه اصلی من از این موضوع صرف نظر کردم و صرفا در این موارد اسم آنها را تغییر می دهم.
نحوه پیاده سازی آنها در پروژه:
جدول Users:

```

Design T-SQL
1 CREATE TABLE [dbo].[Users] (
2     [Id] INT NOT NULL,
3     [userName] VARCHAR (50) NOT NULL,
4     [password] VARCHAR (50) NOT NULL,
5     PRIMARY KEY CLUSTERED ([Id] ASC)
6 );
7
8

```

صاحب خانه یا Lord Land:

```
Design T-SQL
1 CREATE TABLE [dbo].[landLord] (
2     [Id] INT NULL,
3     [address] VARCHAR (50) NOT NULL,
4     [name] VARCHAR (50) NOT NULL,
5     [phoneNumber] INT NOT NULL,
6     [price] INT NOT NULL,
7     [information] VARCHAR (50) NULL,
8     [CreatorID] INT NULL,
9     PRIMARY KEY CLUSTERED ([name] ASC),
10    CONSTRAINT [FK_landLord_ToTable] FOREIGN KEY ([CreatorID]) REFERENCES [Users]([id])
11 );
```

100 % No issues found
Connection Ready

:Buyer

```
T-SQL
CREATE TABLE [dbo].[buyer] (
    [Id] INT NULL,
    [name] VARCHAR (50) NOT NULL,
    [priceLimit] VARCHAR (50) NOT NULL,
    [information] VARCHAR (50) NOT NULL,
    [phoneNumber] FLOAT (53) NOT NULL,
    [CreatorID] INT NOT NULL,
    PRIMARY KEY CLUSTERED ([name]),
    CONSTRAINT [FK_buyer_ToTable] FOREIGN KEY ([CreatorID]) REFERENCES [Users]([id])
);
```

:Seller

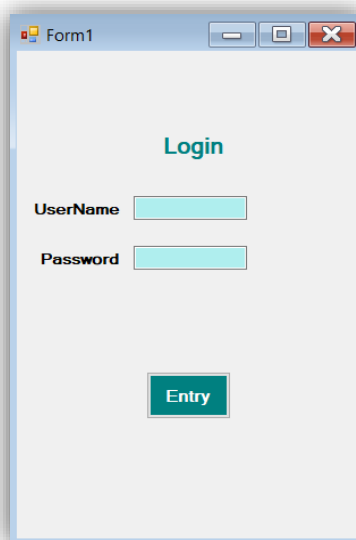
```
sign T-SQL
1 CREATE TABLE [dbo].[seller] (
2     [Id] INT NULL,
3     [address] NCHAR (10) NOT NULL,
4     [name] NCHAR (10) NOT NULL,
5     [phoneNumber] INT NOT NULL,
6     [price] INT NOT NULL,
7     [information] NCHAR (10) NOT NULL,
8     [CreatorID] INT NULL,
9     PRIMARY KEY CLUSTERED ([name] ASC),
10    CONSTRAINT [FK_seller_ToTable] FOREIGN KEY ([CreatorID]) REFERENCES [Users]([id])
11 );
```

No issues found

دسترسی به دیتابیس و نحوه کدزنی رابط کاربری:

```
1 reference
private void Button1_Click(object sender, EventArgs e)
{
    string userName = textBox1.Text;
    string password = textBox2.Text;
    string query = "SELECT " + "*" + " FROM Users WHERE userName='" + userName + "'" + " AND Password='" + password + "'";
    try
    {
        sql.Open();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
    SqlCommand command = new SqlCommand(query, sql);
    SqlDataReader reader = command.ExecuteReader(); //Update Write Delete
    while (reader.Read())
    {
        if (reader["Password"].ToString() == password)
        {
            Form7 fr72 = new Form7();
            fr72.Show();
            this.Hide();
        }
        else
        {
            textBox1.Clear();
            textBox2.Clear();
        }
    }
}
```

- ✚ در تصویر بالا نحوه بروز اکسپشن در صورت وارد کردن نام کاربری و رمز عبوری که در پایگاه داده نیست را نمایش می‌دهد.
- ✚ در نظر داشته باشیم که در ویندوز فرم قبل از انجام کواری ها باید پایگاه داده مورد نظر را به گونه ای به فرم مورد نظر وصل کنیم که به وسیله SqlConnection انجام می‌شد که باید شیئی از آن ایجاد کرد و مقدار Connection String را به کانستراکتور آن داد.
- ✚ روند نوشتن کواری ها به اینصورت است که همه کواری ها را به وسیله استرینگ های متناظر ذخیره کرده و آنها را اجرا می‌کنیم.
- ✚ تمامی خطوط کد بالا در قسمت دکمه ورود در فرم اول قرار دارند.



قسمت Main Page:

مورد بخصوصی در این قسمت وجود ندارد و صرفاً شامل صدا زدن ویندوز فرم ها در یکدیگر می باشد.

قسمت Add Person:

این قسمت و قسمت Add Filed هر دو روند ثابتی دارند چون این دو فرم برای عمل insertion هستند و در نتیجه تنها در فیلد های جدولشان تفاوت هایی دارند.

در این قسمت در قسمت Button1 که در واقع همان Submit هست با وجود یک if و چندین else تلاش می شود که اطلاعات را به درستی در پایگاه داده ذخیره سازی کنیم.

```
34 SqlConnection sql = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Diacko\source\repos\FF_Os_Lab_Motamed
35 string name = textBox1.Text;
36 string priceLimit = textBox2.Text;
37 string phoneNumber = textBox4.Text;
38 string information = textBox3.Text;
39 /* if baraaie type*/
40 sql.Open();
41
42 //SqlDataReader reader = command.ExecuteReader(); //Update Write Delete
43 if (checkedListBox1.SelectedIndex == 0)
44 {
45     string query = "INSERT INTO buyer (name,priceLimit,information,phoneNumber)" +
46         "VALUES ('" + name + "','" + priceLimit + "','" + information + "','" + phoneNumber + "')";
47     SqlCommand cmd = new SqlCommand(query, sql);
48
49     int i = cmd.ExecuteNonQuery();
50     if (i > 0)
51     {
52         Form7 fr = new Form7();
53         fr.Show();
54         this.Hide();
55     }
56     else
57     {
58         MessageBox.Show("Error");
59     }
60 }
61
62 if (checkedListBox1.SelectedIndex == 1)
63 {
64     string query = "INSERT INTO tenant (name,priceLimit,phoneNumber,information)" +
65         "VALUES ('" + name + "','" + priceLimit + "','" + phoneNumber + "','" + information + "')";
66     SqlCommand cmd = new SqlCommand(query, sql);
67 }
```

در واقع ExecuteNonQuery() در این متد اگر خروجی آن برابر صفر باشد متوجه خطا در روند اجرای insertion هستیم.

```

        if (checkedListBox1.SelectedIndex == 1) {
            //biad dorstesh koname
            string query = "INSERT INTO landLord (address,name,phoneNumber,price,information)" +
                "VALUES ('" + address + "','" + name + "','" + phoneNumber + "','" + price + "','" + information + "')";

            SqlCommand cmd = new SqlCommand(query, sql);


            int i = cmd.ExecuteNonQuery();
            if (i > 0)
            {
                Form7 fr = new Form7();
                fr.Show();
                this.Hide();
            }
            else
                MessageBox.Show("Error");
        }
        Form7 fr7 = new Form7();
        fr7.Show();
        this.Hide();
        sql.Close();
    }

```

تمامی کد های مربوط به ارتباط با DB معمولا در قسمت Button ها هستن به جز در موارد خاصی که در جلوتر به توضیح آن می پردازیم. 

دو if اصلی وجود دارد که برای دسترسی به تیبل های متفاوت می باشد برای مثال buyer و tenant. 


قسمت Delete:

در این قسمت با استفاده از نام شخص و مشخص کردن نوع دیتابیس ذخیره شده (نوع درخواست وی برای ثبت در حافظه) و همینطور id آن می توانیم آگهی مورد نظر خود را از DB و جدول مربوطه حذف کنیم. 

```

32  SqlConnection sql = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Diacko\source\repos\FF_Os_Lab_Motamedi\FF_Os_Lab
33  sql.Open();
34  SqlCommand cmd;
35  if (checkedListBox1.SelectedIndex == 0)
36  {
37      //string query = "INSERT INTO buyer (name,priceLimit,information,phoneNumber)" +
38      // "VALUES ('" + name + "','" + priceLimit + "','" + information + "','" + phoneNumber + "')";
39      //DELETE FROM dbo.PurchaseOrderDetail WHERE YourCondition;
40      string query1 = "delete FROM buyer WHERE buyer.name='"+name+"'";
41      cmd = new SqlCommand(query1, sql);
42      int i = cmd.ExecuteNonQuery();
43  }
44  if (checkedListBox1.SelectedIndex == 1)
45  {
46      string query2 = "delete FROM seller WHERE seller.name='" + name + "'";
47      cmd = new SqlCommand(query2, sql);
48      int i = cmd.ExecuteNonQuery();
49  }
50  if (checkedListBox1.SelectedIndex == 2)
51  {
52      string query3 = "delete FROM tenant WHERE tenant.name='" + name + "'";
53      cmd = new SqlCommand(query3, sql);
54      int i = cmd.ExecuteNonQuery();
55  }
56  if (checkedListBox1.SelectedIndex == 3)
57  {
58      string query4 = "delete FROM landLord WHERE LandLord.name='" + name + "'";
59      cmd = new SqlCommand(query4, sql);
60      int i = cmd.ExecuteNonQuery();
61  }
62
63  Form7 fr7 = new Form7();
64  fr7.Show();
65  this.Hide();

```

باتوجه به تصور بالا تمامی if های مورد استفاده برای پیدا کردن تیبل مورد نظر می باشد. 

همانند بالا در صورتی که بخواهیم از هر کدام از صفحه های برنامه به صفحه مین برویم و نخواهیم insert یا delete انجام دهیم کافیت دکمه
 سابمیت را بفشاریم چون null داخل جدول نمی تواند یک ردیف جدید اضافه کند.
 قسمت Show Page :
 در این قسمت نیز برای هر Combo Box داریم:

```
try
{
    comboBox1.Items.Clear();
    string query = "SELECT * FROM buyer";
    SqlConnection sql = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Diacko\source\repos\FF_Os_Lab_Motam\");
    sql.Open();
    SqlCommand command = new SqlCommand(query, sql);
    var dr = command.ExecuteReader();
    while (dr.Read())
    {
        comboBox1.Items.Add(dr["name"] + "," + dr["pricelimit"] + "," + dr["information"] + "," + dr["phoneNumber"]);
    }
    sql.Close();
}
catch (Exception ex)
{
    MessageBox.Show("Error: " + ex.Message);
}
```

با استفاده از یک Try Catch توانستیم اکسپشن های مورد نظر در هنگام خوانش از پایگاه داده و همینطور ریختن آن در Combo Box را هندل
 کنیم.
 برای هر کمبو باکس روند کلی همانند شکل بالاست.

لینک دریافت پروژه:

https://github.com/Diacko242/DB_Lab