

پروژه آز پایگاه داده ها

دانشجو: امید طرب آور 400213016

استاد: آقای دکتر سامانی پور

• لینک ریپازیتوری پروژه در گیت هاب:

[omidTarabavar/DBLab_Project \(github.com\)](https://github.com/omidTarabavar/DBLab_Project)

انتخاب و تعریف هدف سیستم

سیستم مدیریت آموزش (LMS) با هدف ارائه خدمات آموزشی بصورت فراگیر و قابل دسترس به کمک دانشجویان/دانش آموزان و کارکنان آموزشی آمده است. این سیستم تمامی آنچه که در قبل بصورت فیزیکی بر روی کاغذ انجام می شد، چنانچه قابلیت اتوماسیون یا انجام از راه دور را دارند را به فضا اینترنت منتقل می کند و به روند انجام کارها سرعت می بخشد چنانچه که دیگر حضور افراد برای انجام کار اجباری نیست. بدیهی است که یکی از نیازهای اصلی کاربران امنیت خواهد بود پس علاوه بر خدمات باید امنیت کاربران را نیز فراهم کنیم و در حفظ اطلاعات آنها کوشا باشیم.

تعریف سناریو ها و عملکردهای اصلی سیستم

سناریو 1: ثبت نام

- (a) کاربر بر روی ثبت نام کلیک می کند.
- (b) سیستم صفحه ی ثبت نام را به کاربر نمایش می دهد.
- (a) کاربر اطلاعات خود را (نام و نام خانوادگی، ایمیل، شماره تلفن و پست و نوع [استاد/دانشجو]) وارد کرده و گزینه اتمام ثبت نام را انتخاب می کند.
- (b) سیستم اطلاعات کاربر را بررسی می کند و در صورت صحت و عدم تشابه ایمیل کاربر با ایمیل کاربران قبلی، ثبت نام کاربر را تایید کرده و او را به صفحه شخصی خود هدایت می کند.

سناریو 2: ورود

- (a) کاربر بر روی ورود کلیک می کند.

- (b) سیستم صفحه ی ورود را به کاربر نمایش می دهد.
- (a) کاربر اطلاعات خود را (ایمیل و پسورد) وارد کرده و گزینه وارد شدن را انتخاب می کند.
- (b) سیستم اطلاعات کاربر را بررسی می کند و در صورت صحت، ورود کاربر را تایید کرده و او را به صفحه شخصی خود هدایت می کند.

سناریو 3: ایجاد درس

- (a) استاد گزینه "ایجاد درس" را انتخاب می کند.
- (b) سیستم صفحه ی ایجاد درس را به استاد نمایش می دهد.
- (a) استاد اطلاعات درس را (عنوان درس، سرترم درس، نام دانشکده و آیدی استاد) وارد کرده و گزینه تکمیل ایجاد درس را انتخاب می کند.
- (a) سیستم اطلاعات استاد را بررسی کرده و درس را در سیستم ذخیره می کند و آیدی منحصر به آن درس را به استاد نمایش می دهد.
- (b) استاد آیدی دریافت شده توسط سیستم را در اختیار دانشجویان قرار داده تا آن ها بتوانند در آن درس عضو شوند.

سناریو 4: عضویت در درس

- (a) دانشجو گزینه "عضویت در درس" را انتخاب می کند.
- (b) سیستم صفحه ی عضویت در درس را به دانشجو نمایش می دهد.
- (a) دانشجو اطلاعات درس را (آیدی درس) وارد کرده و گزینه درخواست عضویت را انتخاب می کند.
- (b) سیستم با بررسی آیدی وارد شده، درخواست عضویت دانشجو را به استاد آن درس میفرستد و در صورت تایید استاد، عضویت دانشجو در درس با موفقیت انجام می شود.

سناریو 5: ارسال محتوا توسط استاد

- (a) استاد در صفحه ی درس گزینه "ارسال محتوا" را انتخاب می کند.
- (b) سیستم صفحه ی ارسال محتوا را به استاد نمایش می دهد.
- (a) استاد اطلاعات محتوا را (نام محتوا، لینک فایل محتوا) وارد کرده و گزینه ارسال را انتخاب می کند.
- (b) سیستم فایل محتوا را دریافت کرده و آن را در درس صفحه درس در اختیار دانشجویان قرار می دهد.

سناریو 6: دریافت محتوا توسط دانشجویان

- (a) دانشجو از لیست دروس خود، درس مورد نظر را انتخاب می کند.
- (b) سیستم صفحه ی درس را به دانشجو نمایش می دهد.
- (a) دانشجو محتوای مورد نظر را انتخاب می کند.
- (b) سیستم لینک دانلود محتوا را در اختیار دانشجو قرار می دهد.

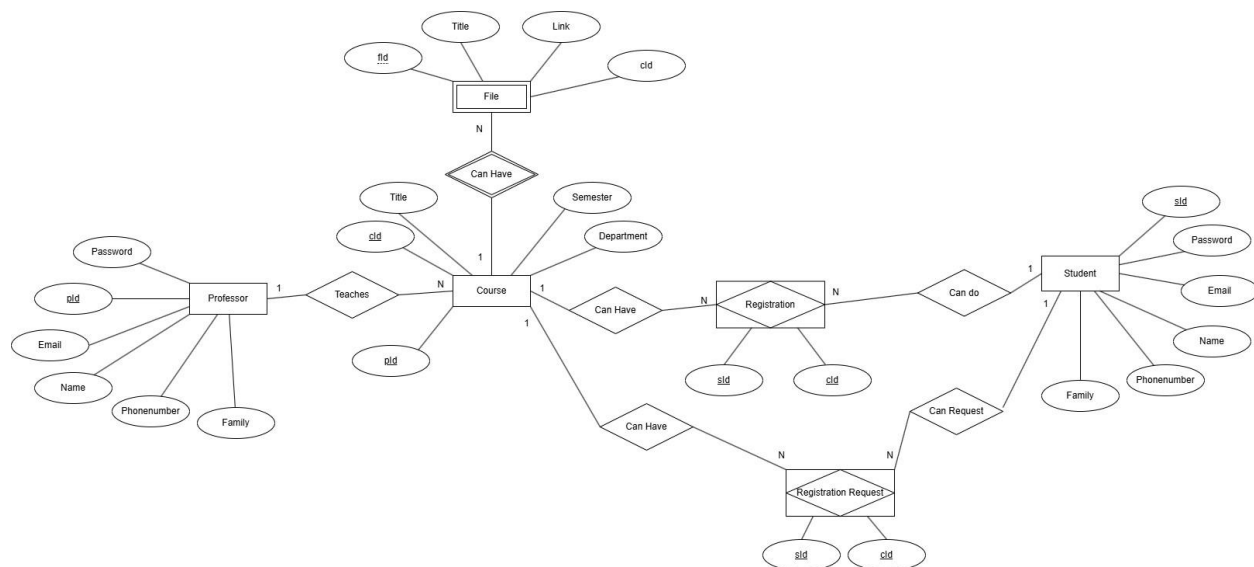
سناریو 7: ویرایش اطلاعات کاربری

- (a) کاربر وارد صفحه شخصی خود می شود و ویرایش اطلاعات را انتخاب می کند.
- (b) سیستم صفحه ویرایش اطلاعات را به کاربر نمایش می دهد. در این صفحه اطلاعاتی که کاربر در هنگام ثبت نام وارد کرده بود نمایش داده می شود و امکان تغییر آن ها وجود دارد.
- (a) کاربر تغییرات مورد نیاز خود را اعمال می کند و پس از آن بر روی بروزرسانی کلیک می کند.
- (b) سیستم تغییرات کاربر را بررسی کرده و آن را اعمال می کند.

سناریو 8: ویرایش اطلاعات درس توسط استاد

- (a) استاد وارد صفحه درس می شود و ویرایش اطلاعات را انتخاب می کند.
- (b) سیستم صفحه ویرایش اطلاعات درس را به استاد نمایش می دهد. در این صفحه اطلاعاتی که استاد در هنگام ایجاد درس وارد کرده بود نمایش داده می شود و امکان تغییر آن ها وجود دارد.
- (a) استاد تغییرات مورد نیاز خود را اعمال می کند و پس از آن بر روی بروزرسانی کلیک می کند.
- (b) سیستم تغییرات استاد را بررسی کرده و آن را اعمال می کند.

طراحی مدل منطقی پایگاه داده



پیاده سازی برنامه کاربردی

- نمای کلی از محیط برنامه:

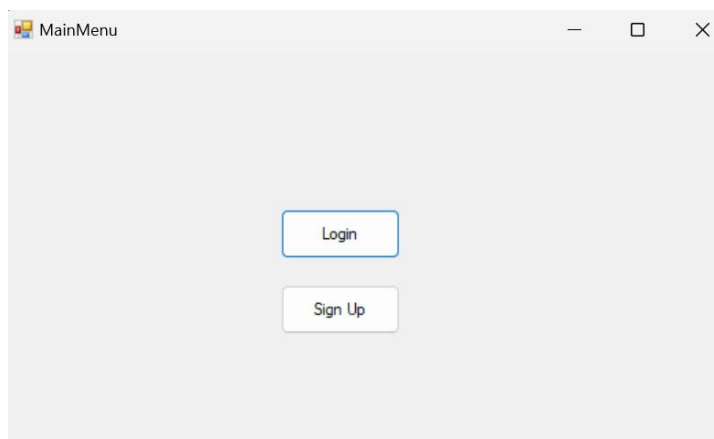
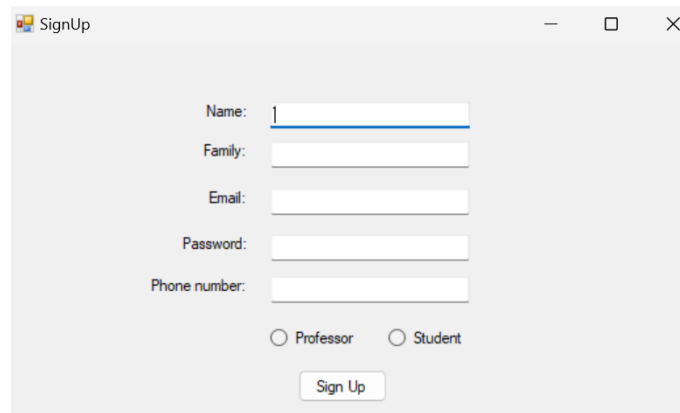


Figure1 -Main Menu



SignUp

Name:

Family:

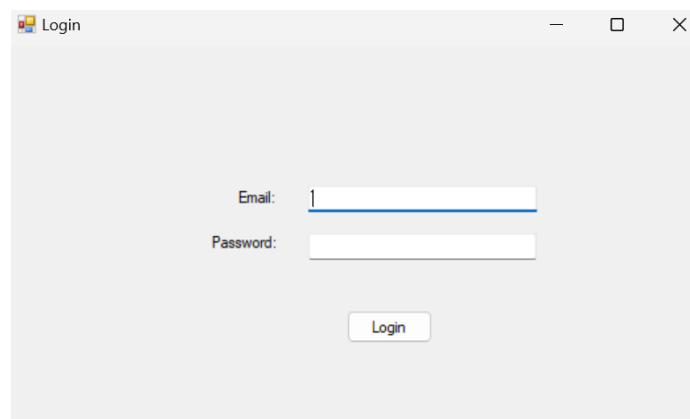
Email:

Password:

Phone number:

☐ Professor ☐ Student

Figure2 - Sign Up Menu

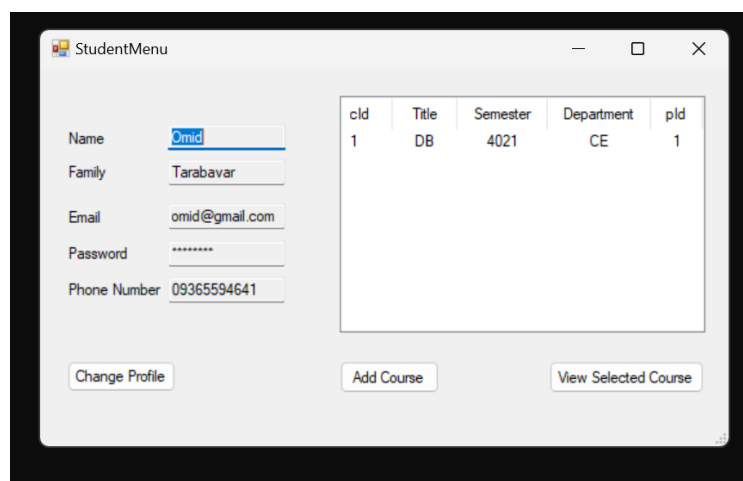


Login

Email:

Password:

Figure 3 - Login Menu



StudentMenu

Name:

Family:

Email:

Password:

Phone Number:

cld	Title	Semester	Department	pld
1	DB	4021	CE	1

Figure 4 - Student Menu

StudentMenu

Name: Omid

Family: Tarabavar

Email: omid@gmail.com

Password: 1234

Phone Number: 09365594641

cld	Title	Semester	Department	pld
1	DB	4021	CE	1

Save Add Course View Selected Course

Figure 5 - Student Menu (after clicking change profile)

AddCourseStd

Course Id:

Request

Figure 6 - Add Course Menu for student

CourseMenuStd

Title: DB

Semester: 4021

Department: CE

pld: 1

fld	Title	Link
1	Temp	google.com

View Selected File

Figure 7 - Course Menu for student (after clicking view selected course)

FileMenuStd

Title:

Link:

Figure 8 - File Menu for student (after clicking view selected file)

ProfessorMenu

Name:

Family:

Email:

Password:

Phone Number:

cld	Title	Semester	Department	pld
1	DB	4021	CE	1

Figure 9 - Professor Menu

AddCourse

Title:

Semester:

Department:

Professor Id:

Figure 10 - Add Course for professor

CourseMenuProf

Title: DB

Semester: 4021

Department: CE

pId: 1

Change Details

View Files

Remove Selected Student

View Requests

sId	Name	Family	Email
1	Omid	Tarabavar	omid@gmail.com

Figure 11 - Course Menu for professor (after clicking view selected course)

FileListMenu

fId	Title	Link	cId
1	Temp	google.com	1

Add File

Remove Selected File

Figure 12 - File Menu for professor (after clicking view files)

RequestMenu

sId	Name	Family	Email
1	Omid	Tarabavar	omid@gmail.com

Accept

Reject

- Request Menu for professor (after clicking view requests) 13 Figure

- تکه کد های مهم همراه با توضیحات لازم:

```
1 public static int ExecuteNonQuery(string query, SqlParameter[] parameters)
2 {
3     using (SqlConnection connection = new SqlConnection(cs))
4     {
5         connection.Open();
6
7         using (SqlCommand command = new SqlCommand(query, connection))
8         {
9             if (parameters != null)
10            {
11                command.Parameters.AddRange(parameters);
12            }
13            try
14            {
15                return command.ExecuteNonQuery();
16            }
17            catch (Exception ex)
18            {
19                return 0;
20            }
21        }
22    }
23 }
```

```
1 public static DataTable ExecuteQuery(string query, SqlParameter[] parameters)
2 {
3     using (SqlConnection connection = new SqlConnection(cs))
4     {
5         connection.Open();
6         using (SqlCommand command = new SqlCommand(query, connection))
7         {
8             if (parameters != null)
9             {
10                command.Parameters.AddRange(parameters);
11            }
12
13            using (SqlDataAdapter adapter = new SqlDataAdapter(command))
14            {
15                DataTable dataTable = new DataTable();
16                adapter.Fill(dataTable);
17                command.Parameters.Clear();
18                return dataTable;
19            }
20        }
21    }
22 }
```

برای راحتی در اجرای کوئری ها، آن ها را به دو نوع Query و Non-Query تقسیم کردیم. Query ها آن هایی هستند که داده ای بر می گردانند (مانند دستور Select) و Non-Query آن هایی هستند که فقط تغییراتی در محتویات جداول انجام می دهند و داده ای را بر نمی گردانند

با استفاده از کلاس DBHelper و این دو متد، ما این کوئری ها را اجرا می کنیم. پارامتر های هر کوئری، داده هایی هستند که ما آن ها را به کوئری می چسبانیم. در جلوتر مثالی از این نوع داده ها آورده خواهد شد.

```
1 public static int Login(string email, string password)
2 {
3     SqlParameter[] sqlParameters = {
4         new SqlParameter("@Email", SqlDbType.VarChar) { Value = email },
5         new SqlParameter("@Password", SqlDbType.VarChar) { Value = password }
6     };
7
8     string query = "SELECT pId FROM Professor WHERE Email = @Email AND Password = @Password";
9     DataTable res = DBHelper.ExecuteQuery(query, sqlParameters);
10    if (res.Rows.Count > 0)
11        return 0;
12    sqlParameters = new SqlParameter[] {
13        new SqlParameter("@Email", SqlDbType.VarChar) { Value = email },
14        new SqlParameter("@Password", SqlDbType.VarChar) { Value = password }
15    };
16    query = "SELECT sId FROM Student WHERE Email = @Email AND Password = @Password";
17    res = DBHelper.ExecuteQuery(query, sqlParameters);
18    if (res.Rows.Count > 0)
19        return 1;
20    return -1;
21 }
```

برای ورود، از ایمیل و پسورد استفاده شده است. از آنجایی که نمی دانیم در هنگام ورود، فردی که درخواست ورود دارد دانشجو است یا استاد، مجبور هستیم اول جدول استاد را چک کنیم و ببینیم آیا استادی با این ایمیل و پسورد موجود است و اگر نبود جدول دانشجو را چک کنیم.

از جمله داده هایی که به یک کوئری به عنوان پارامتر چسبیده شده اند در اینجا می توان به Email و Password اشاره کرد.

از آنجایی که ما کلاسی به نام User داریم و بعد دو کلاس به نام های Student و Professor از آن ارث بری کردند، ما Professor را User ای با تایپ 0 در نظر گرفتیم و Student را User ای با تایپ 1. دلیل بازگشت عدد 0 در صورتی پیدا کردن استاد و عدد 1 در صورت پیدا کردن دانشجو نیز همین است.

```

1 public static int AddCourse(string title, int semester, string department, int pId)
2 {
3     string query;
4     SqlParameter[] sqlParameters;
5     query = "INSERT INTO Course (Title, Semester, Department, pId) VALUES (@Title, @Semester, @Department, @pId)";
6     sqlParameters = new SqlParameter[] {
7         new SqlParameter("@Title", SqlDbType.VarChar) {Value = title},
8         new SqlParameter("@Semester", SqlDbType.Int) {Value = semester},
9         new SqlParameter("@Department", SqlDbType.VarChar) {Value = department},
10        new SqlParameter("@pId", SqlDbType.Int) {Value = pId},
11    };
12    int res = DBHelper.ExecuteNonQuery(query, sqlParameters);
13    if (res == 0)
14        return -1;
15    else
16        return 1;
17 }

```

برای اضافه کردن Course از متد AddCourse در کلاس Professor استفاده می شود. این متد با استفاده کوئری Insert ساده با استفاده از پارامتر های ورودی کلاس جدید را ایجاد می کند.

```

1 private void button1_Click(object sender, EventArgs e)
2 {
3     if (textBox1.Text.Equals(""))
4         MessageBox.Show("Please enter course id!");
5     else
6     {
7         int cId = int.Parse(textBox1.Text);
8         string query = "INSERT INTO Registration_Request (sId, cId) VALUES (@sId, @cId)";
9         SqlParameter[] sqlParameters = {
10            new SqlParameter("@sId", SqlDbType.VarChar) {Value = student.id},
11            new SqlParameter("@cId", SqlDbType.Int) {Value = cId}
12        };
13        int res = DBHelper.ExecuteNonQuery(query, sqlParameters);
14        if (res > 0)
15        {
16            this.Dispose();
17        }
18        else
19        {
20            MessageBox.Show("Can't send request. Maybe you already did!");
21        }
22    }
23 }

```

همانطور که در سناریو 4 گفته شد، دانشجو برای عضویت در کلاس باید درخواست دهد و این درخواست را استاد قبول یا در می کند. در این متد ما از جدول Registration_Request استفاده کردیم که درخواست عضویت خود را به ثبت برسانیم. دانشجو با

وارد کردن cId یا آیدی کلاس، یک Entry به این جدول اضافه می کند. در قسمت بعدی نحوه کنترل این درخواست ها توسط استاد را مشاهده می کنیم.

```
1 private void loadRequests()
2 {
3     reqLV.Items.Clear();
4     string query = "SELECT s.sId as sId, s.Name as Name, s.Family as Family, s.Email as
5     Email FROM Registration_Request as r JOIN Student as s ON r.sId = s.sId WHERE r.cId = @cId";
6     SqlParameter[] sqlParameters = {
7         new SqlParameter("@cId", SqlDbType.Int) {Value = cid}
8     };
9     DataTable dt = DBHelper.ExecuteQuery(query, sqlParameters);
10    foreach (DataRow dr in dt.Rows)
11    {
12        ListViewItem item = new ListViewItem(dr["sId"].ToString());
13        item.SubItems.Add(dr["Name"].ToString());
14        item.SubItems.Add(dr["Family"].ToString());
15        item.SubItems.Add(dr["Email"].ToString());
16        reqLV.Items.Add(item);
17    }
```

```
1 private void accBT_Click(object sender, EventArgs e)
2 {
3     if (reqLV.SelectedItems.Count > 0)
4     {
5         ListViewItem slcItem = reqLV.SelectedItems[0];
6         int sId = int.Parse(slcItem.SubItems[0].Text);
7         SqlParameter[] sqlParameters = {
8             new SqlParameter("@sId", SqlDbType.Int) {Value = sId},
9         };
10        string query = "DELETE FROM Registration_Request WHERE sId = @sId";
11        DBHelper.ExecuteNonQuery(query, sqlParameters);
12        sqlParameters = new SqlParameter[] {
13            new SqlParameter("@sId", SqlDbType.Int) {Value = sId},
14            new SqlParameter("@cId", SqlDbType.Int) {Value = cid}
15        };
16        query = "INSERT INTO Registration (sId, cId) VALUES (@sId, @cId)";
17        DBHelper.ExecuteNonQuery(query, sqlParameters);
18        RequestMenu_Load(sender, e);
19    }
20    else
21    {
22        MessageBox.Show("Please select a request!");
23    }
24 }
```

```

1 private void rejBT_Click(object sender, EventArgs e)
2 {
3     if (reqLV.SelectedItems.Count > 0)
4     {
5         ListViewItem slcItem = reqLV.SelectedItems[0];
6         int sId = int.Parse(slcItem.SubItems[0].Text);
7         SqlParameter[] sqlParameters = {
8             new SqlParameter("@sId", SqlDbType.Int) {Value = sId},
9         };
10        string query = "DELETE FROM Registration_Request WHERE sId = @sId";
11        DBHelper.ExecuteNonQuery(query, sqlParameters);
12        RequestMenu_Load(sender, e);
13    }
14    else
15    {
16        MessageBox.Show("Please select a request!");
17    }
18 }

```

اول با استفاده از متد loadRequests درخواست هایی که از طرف دانشجویان برای عضویت در یک کلاس خاص ارسال شده را بدست می آوریم. این درخواست ها در جدول Registration_Request ذخیره شده اند. از آنجایی که این جدول دو ستون sId و cId را بیشتر ندارد، برای بدست آوردن مشخصات دانشجو مجبور بودیم آن را با جدول Student جویین کنیم.

اگر استاد درخواست عضویت دانشجو را قبول کرد، درخواست او از جدول Registration_Request حذف و sId او (آیدی دانشجو) همراه با cId (آیدی کلاس) در جدول Registration ذخیره می شود. این جدول برای این است که هر دانشجو در چه کلاسی ثبت نام کرده است.

اگر استاد درخواست عضویت دانشجو را قبول نکرد، فقط کافی درخواست او را از جدول Registration_Request حذف کنیم.

```

1 public static int Upload_File(string title, string link, int cId)
2 {
3     string query;
4     SqlParameter[] sqlParameters;
5     query = "INSERT INTO Files (Title, Link, cId) VALUES (@Title, @Link, @cId);";
6     sqlParameters = new SqlParameter[] {
7         new SqlParameter("@Title", SqlDbType.VarChar) {Value = title},
8         new SqlParameter("@Link", SqlDbType.VarChar) {Value = link},
9         new SqlParameter("@cId", SqlDbType.Int) {Value = cId},
10    };
11    int res = DBHelper.ExecuteNonQuery(query, sqlParameters);
12    if (res == 0)
13        return -1;
14    else
15        return 1;
16 }

```

برای آپلود فایل استاد می تواند از متد Upload_Files کلاس Files استفاده کند. این متد طبق سناریو 5 عنوان و لینک را از استاد دریافت می کند و همراه با آیدی کلاس، Entry ای در جدول Files برای آن ایجاد می کند.

```

1 private void downBT_Click(object sender, EventArgs e)
2 {
3     string url = linkTB.Text;
4     if (!url.StartsWith("http://") && !url.StartsWith("https://"))
5     {
6         url = "http://" + url;
7     }
8
9     if (Uri.IsWellFormedUriString(url, UriKind.Absolute))
10    {
11        try
12        {
13            Process.Start(new ProcessStartInfo
14            {
15                FileName = url,
16                UseShellExecute = true
17            });
18        }
19        catch (Exception ex)
20        {
21            MessageBox.Show("Invalid Link!");
22        }
23    }
24 }

```


دانشجو طبق سناریو 6 می تواند محتوایی که استاد در کلاس منتشر می کند را با استفاده از متد `downBT_Click` دریافت کند. در این متد لینک فایل بررسی می شود و اگر با `http` یا `https` شروع نشده بود، آن را کامل می کند و بعد با استفاده از کلاس `System.Process`، آن را در مروجر باز می کند که طبیعتا اگر لینک درست باشد، دانلود فایل آغاز خواهد شد.

```
1 private void rmvStdBT_Click(object sender, EventArgs e)
2 {
3     if (stdLV.SelectedItems.Count > 0)
4     {
5         ListViewItem slcItem = stdLV.SelectedItems[0];
6         int sId = int.Parse(slcItem.SubItems[0].Text);
7         SqlParameter[] sqlParameters = {
8             new SqlParameter("@sId", SqlDbType.Int) {Value = sId},
9         };
10        string query = "DELETE FROM Registration WHERE sId = @sId";
11        DBHelper.ExecuteNonQuery(query, sqlParameters);
12        CourseMenuProf_Load(sender, e);
13    }
14    else
15    {
16        MessageBox.Show("Please select a student!");
17    }
18 }
```

استاد می تواند دانشجوی مورد نظر را از کلاس حذف کند! برای اینکار کافی است پس از اینکه لیستی از دانشجویان به اون نمایش داده می شود (مانند Figure 11) با انتخاب یک دانشجو از لیست کلیک بروی `Remove Selected Student` دانشجوی مورد نظر را از کلاس حذف کند. برای حذف دانشجو کافیست `Entry` مربوط به آن دانشجو را از جدول `Registration` حذف کرد.

```

1 private void chngProf_Click_1(object sender, EventArgs e)
2 {
3     nameTB.ReadOnly = false;
4     familyTB.ReadOnly = false;
5     emailTB.ReadOnly = false;
6     pwTB.ReadOnly = false;
7     pwTB.Text = student.password;
8     pnTB.ReadOnly = false;
9     saveBT.Visible = true;
10    chngProf.Visible = false;
11 }
12
13 private void saveBT_Click_1(object sender, EventArgs e)
14 {
15     int res = User.ChangeProfile(student.id, nameTB.Text,
16     familyTB.Text, emailTB.Text, pwTB.Text, pnTB.Text, 1);
17     if (res == 1)
18     {
19         student.name = nameTB.Text;
20         student.family = familyTB.Text;
21         student.email = emailTB.Text;
22         student.password = pwTB.Text;
23         student.phoneNumber = pnTB.Text;
24         StudentMenu_Load(sender, EventArgs.Empty);
25     }
26     else
27     {
28         MessageBox.Show("Couldn't save this new information!");
29     }
30 }

```

چه در منوی استاد چه در منوی دانشجو، امکان تغییر اطلاعات کاربری وجود دارد. اینکار با استفاده از دکمه Change Profile انجام می شود. با کلیک بر روی این دکمه، مشخصات قابل تغییر خواهند شد و از حالت Read Only خارج می شوند و بعد کاربر (استاد یا دانشجو) می تواند آن ها را تغییر دهد. البته نکته ای لازم به ذکر است که اگر بعد از ایجاد تغییر، ایمیل جدید کاربر با ایمیلی از کاربران قبلی یکسان باشد، به کاربر پیغام خطا نمایش داده می شود زیرا ایمیل کاربر باید منحصر به فرد باشد.

نکته: با کلیک بر روی Change Profile، دکمه Change Profile پنهان و دکمه Save نمایان می شود و با کلیک بر روی Save، دکمه Save پنهان و دکمه Change Profile نمایان می شود.


```

1 private void chngDetBT_Click(object sender, EventArgs e)
2 {
3     titleTB.ReadOnly = false;
4     depTB.ReadOnly = false;
5     semTB.ReadOnly = false;
6     pidTB.ReadOnly = false;
7     saveBT.Visible = true;
8     chngDetBT.Visible = false;
9 }
10
11 private void saveBT_Click(object sender, EventArgs e)
12 {
13     string title = titleTB.Text;
14     string dep = depTB.Text;
15     int sem = int.Parse(semTB.Text);
16     int pid = int.Parse(pidTB.Text);
17     int res = Course.ChangeCourseInfo(title, sem, dep, pid,
18     course.id);
19     if (res == 1)
20     {
21         course.title = title;
22         course.semester = sem;
23         course.pid = pid;
24         course.department = dep;
25         CourseMenuProf_Load(sender, e);
26     }
27 }

```

تغییر اطلاعات درس توسط استاد نیز همانند تغییر اطلاعات کاربری انجام می شود.

در این گزارش سعی شد قسمت های مهم کد بررسی شود. باقی کد ها تماما در گیت هاب موجود هستند.

راهنمای ریپازیتوری گیت هاب:

- فولدر ERD مربوط به طراحی مدل منطقی پایگاه داده می شود و تصویر آن در این فایل قرار دارد.

- فولدر Queries مربوط به کوئری ساخت جداول می باشد.
- فولدر LMS مربوط به کد های برنامه کاربردی می باشد.
- فولدر Database مربوط به فایل های دیتابیس می باشد.

پایان.