



Industrial Informatics Project  
Bodea Ciprian, Daczo-Cadar Amalia, Moldovan Ovidiu  
Automation and Applied Informatics  
2018-2019

# HelpStud

Course materials sharing between students

# Table of Contents

1. Introduction .....	2
• The problem	
• The objective	
• The specifications	
2. Application architecture .....	3
• UML Diagram	
• Use-Case Diagram	
3. Application implementation .....	5
• Database implementation	
• WebMethods implementation	
• Windows Forms implementation	
4. Application functionality .....	7
• Sign in form	
• Admin panel	
• User panel	
• User account panel	
5. Application testing .....	9
6. Appendix .....	10

# Introduction

**Our team** members are:

- Moldovan Ovidiu - Team-leader, Developer
- Daczo-Cadar Amalia - Developer
- Bodea Ciprian - Tester

**The problem** our application is focusing on is the dilemma of the students' archives of materials for different courses taken by undergraduates during their scholarship. Using HelpStud, students could have a place allotted for engaging in course-related conversations and other study related useful information, such as tips for the upcoming exams, YouTube videos that were found to be helpful with clarifying different concepts encountered during one's studies or notes which were taken during the year.

**Our objective** is to give the students such a place so that they may freely share thoughts and feelings with each other, along with different materials which could aid not only them, but also the future generations of scholars.

**As specifications**, it should be mentioned that the project is a *multi-level application*: the first level is a *database* made in *Microsoft SQL Server*; the second level is an *ASP. NET* web server with the functionality of the application represented by various *WebMethods* and the third level is a *Windows Forms* user interface based on the *.NET Framework*.

## 2. Application architecture

The default role in the application is that of a *user*. The user can see other posts, can update his/her profile, can post and can comment already uploaded content.

The *admin* role can add/remove/delete faculties, courses and posts.

## 2.1. UML Diagram

The UML diagram is presented in *Figure 2.1*, showing the main classes of our application.

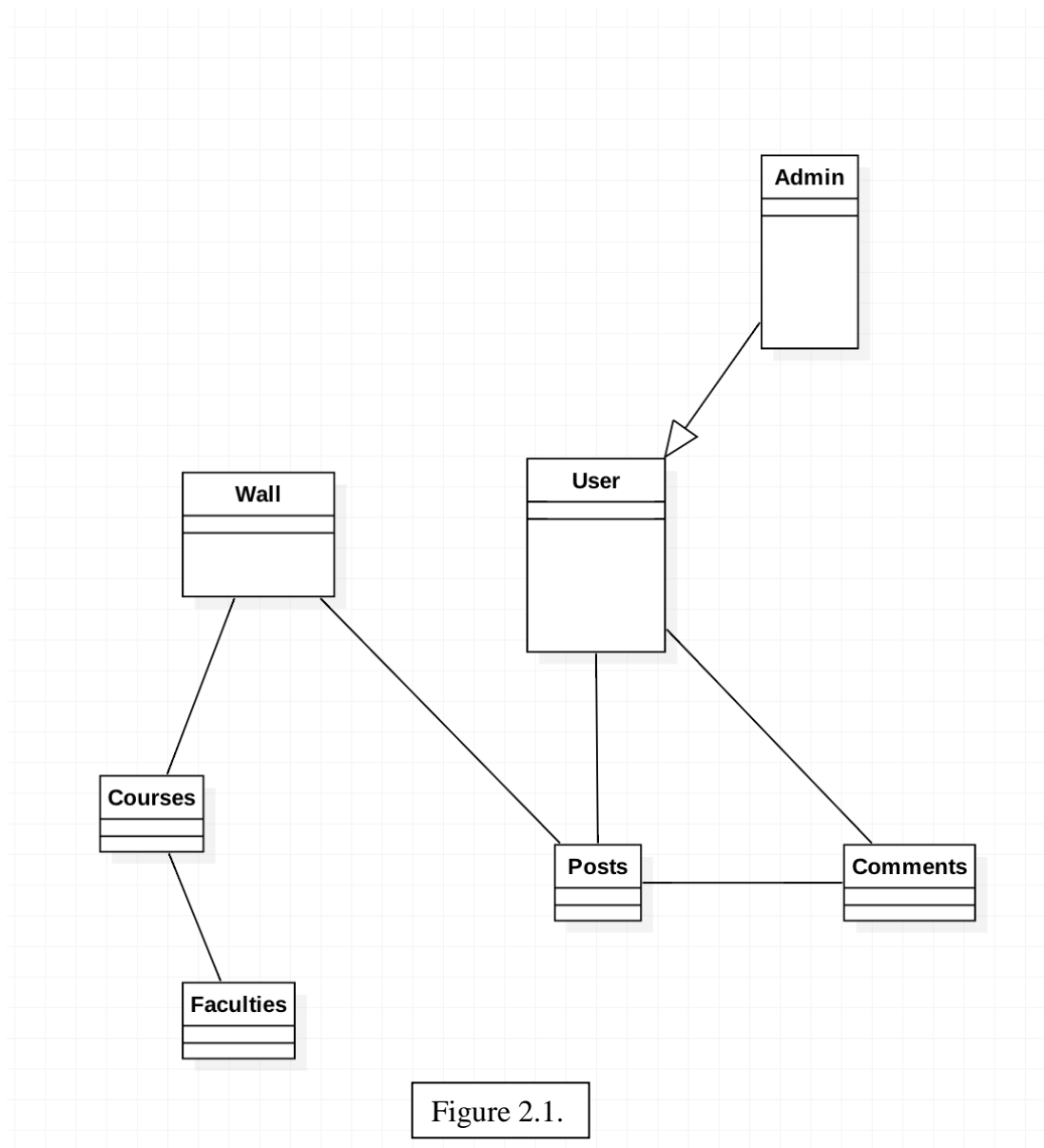


Figure 2.1.

## 2.2. Use-Case Diagram

The Use-Case diagram is presented in *Figure 2.2*, showing the roles of each type of user, as well as their rights.

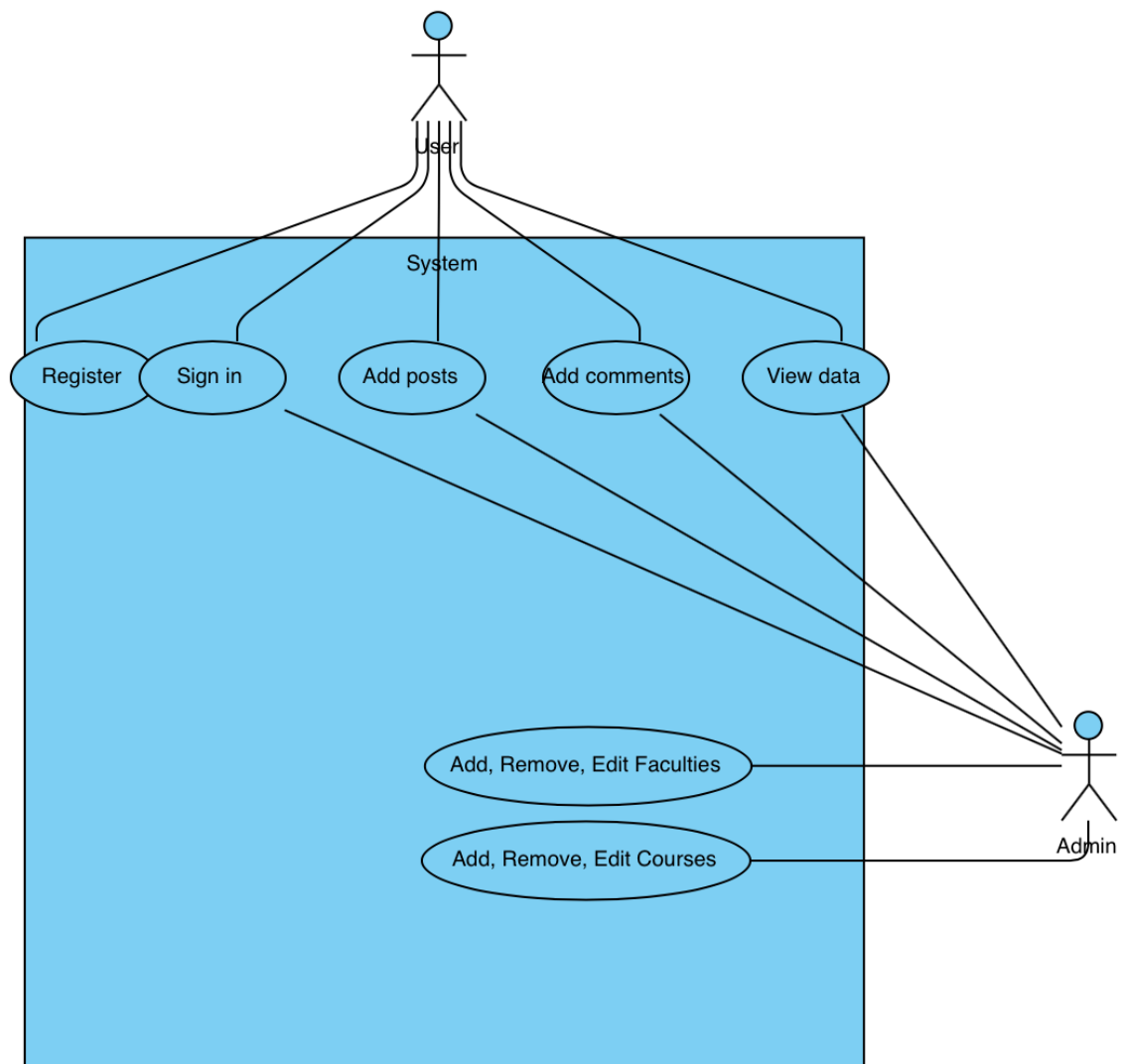


Figure 2.2.

## 3. Application implementation

### 3.1. Database implementation

The database has been implemented in the server part of the application, with a representation of the tables presented in *Figure 3.1*.

### 3.2. WebMethods implementation

Most of the WebMethods implemented execute SQL Queries in such a manner that the client can access the data it needs from the database.

Example:

```
1. [WebMethod]
2. public DataSet getAllPosts()
3. {
4.     myCon.ConnectionString = conn;
5.     myCon.Open();
6.
7.     SqlCommand cmd = new SqlCommand("select title, content
from Posts", myCon);
8.     SqlDataAdapter da = new SqlDataAdapter(cmd);
9.     DataSet ds = new DataSet();
10.    da.Fill(ds, "Posts");
11.    return ds;
12. }
```

\*full code can be found in Appendix.

### 3.3. Windows Forms implementation

The client part of the application is a GUI with several forms that transmit the data through, towards and from the WebMethods.

Example:

```
1. public AppUser(int idUser)
2. {
3.     this.idUser = idUser;
4.     InitializeComponent();
5.     LoadYearSemester();
6.     LoadTitles();
7.     comboBox1.Text = "Faculty";
8.     textBoxNewPassword.PasswordChar = '*';
9.     textBoxOldPassword.PasswordChar = '*';
10.
11.     DataSet dsFaculties = new DataSet();
12.
13.     dsFaculties = service.getAllFaculties();
14.
15.     try
16.     {
17.
18.         foreach (DataRow dr in dsFaculties.Tables[0].Rows)
```

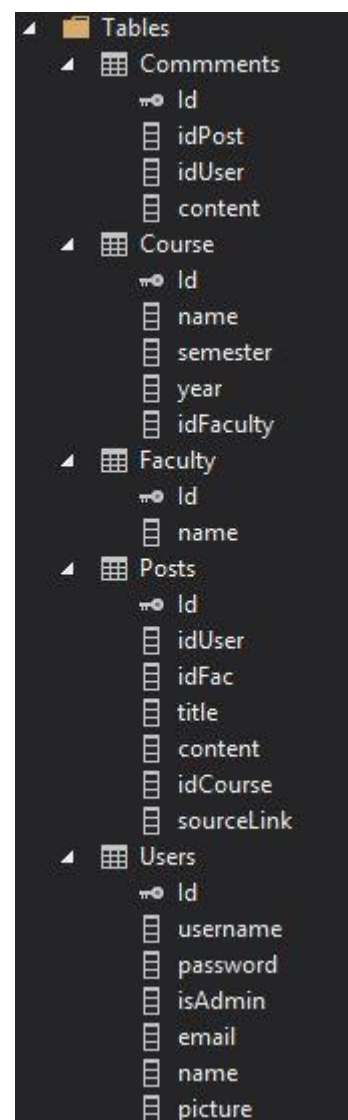


Figure 3.1.

```

19.
20.         {
21.             String name = dr.ItemArray.GetValue(1).ToString();
22.             comboBox1.Items.Add(name);
23.         }
24.     }
25.
26.     catch (Exception xe)
27.     {
28.         MessageBox.Show("No faculties");
29.     }
30. }

```

\*full code can be found in Appendix.

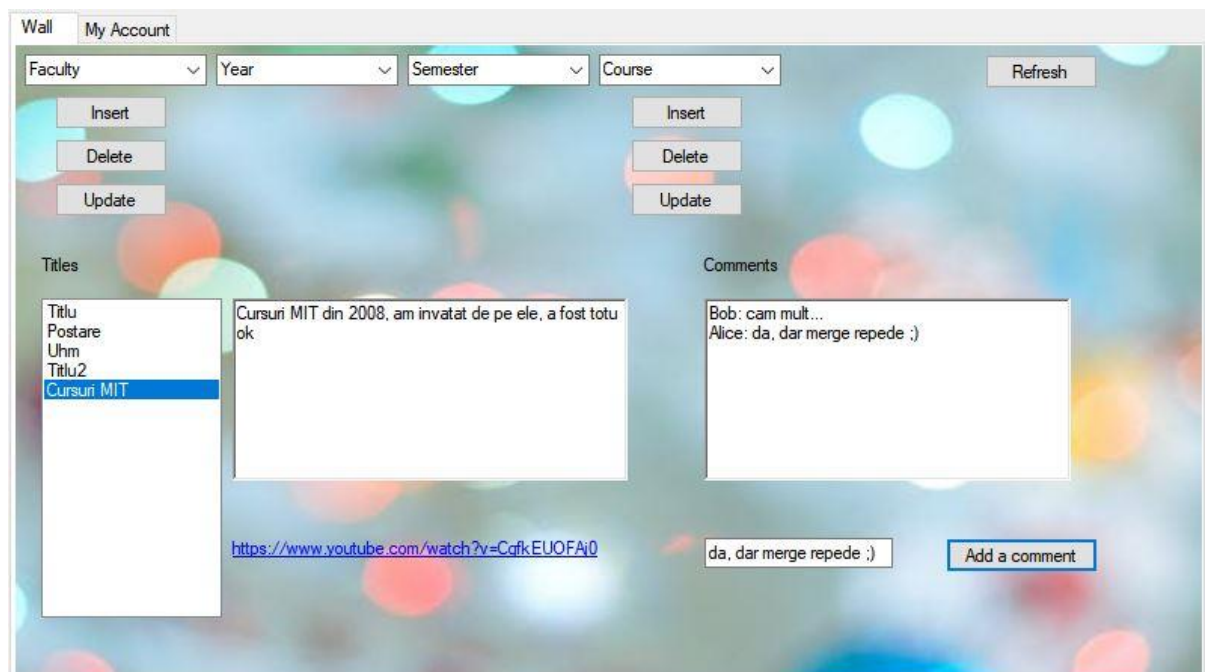
## 4. Application functionality

### 4.1. Sign in form



A screenshot of a sign-in form. It features two input fields: 'Username' and 'Password'. Below the 'Password' field is a 'Sign in' button. At the bottom, there is a 'Register' link followed by the text 'here' in blue, underlined.

### 4.2. Admin panel



A screenshot of an admin panel interface. At the top, there are tabs for 'Wall' and 'My Account'. Below the tabs are four dropdown menus: 'Faculty', 'Year', 'Semester', and 'Course', followed by a 'Refresh' button. There are two sets of buttons: 'Insert', 'Delete', and 'Update' on the left, and another set of 'Insert', 'Delete', and 'Update' on the right. The main area is divided into two sections: 'Titles' and 'Comments'. The 'Titles' section has a list of titles: 'Titlu', 'Postare', 'Uhm', 'Titlu2', and 'Cursuri MIT' (which is highlighted in blue). Below the list is a text area containing the text 'Cursuri MIT din 2008, am invatat de pe ele, a fost totu ok' and a URL 'https://www.youtube.com/watch?v=CgfkEUQFAi0'. The 'Comments' section has a text area containing the text 'Bob: cam mult...' and 'Alice: da, dar merge repede ;)'. Below the text area is a button labeled 'da, dar merge repede :)' and an 'Add a comment' button.

### 4.3. User panel

The screenshot shows a web interface for a user panel. At the top, there are tabs for 'Wall' and 'My Account'. Below the tabs, there are four dropdown menus: 'Facultatea de Autom', '3', '2', and 'II'. A 'Refresh' button is to the right of these menus. The main content area is divided into two columns. The left column is titled 'Titles' and contains a list of links: 'Link download VS' and 'Cursuri'. The right column is titled 'Comments' and contains a text area with the following text: 'Alice: Care e parola???' and 'Bob: Nu o stiu, scuze :D'. Below the comments, there is a text input field and an 'Add a comment' button. At the bottom of the page, there is a link: [http://users.utcluj.ro/~valean/industrial\\_informatics.html](http://users.utcluj.ro/~valean/industrial_informatics.html).

### 4.4. User account panel

The screenshot shows a web interface for a user account panel. On the left, there is a user profile section with a photo of a man, the text 'Hello, user3', a 'Sign out' link, and buttons for 'Change photo' and 'Add a post!'. The main content area is divided into four sections, each with a 'Modify' button: 'Change Name' (current name: Bob), 'Change Username' (current username: user3), 'Change Email' (current email: user3@yahoo.com), and 'Change Password' (current password: Please introduce your old password!).

## 5. Application testing

The initial testing of the application was done in two different stages. The first part of the testing was done while developing different methods for the implementation, while the second part of the testing was fulfilled by introducing invalid data, on purpose, in different forms in order to test for errors and bugs.

The main cause of errors proved to be the appearance of null values in the database, a situation remedied through the employment of try and catch blocks, which are used to surround the



problematic lines of code. This solution proved not only to be an elegant one but also one which does not present any impediment on the application's functionality.

Another testing is done while registering a new account, moment in which the application verifies if the username is already taken. It also verifies that the password is at least 6 characters, the username is at least 4 characters and the validity of the email address of the new user (must contain the '@' character).

## 7. Appendix – Code

Note: Because the code is very large, the full code can be accessed at <https://github.com/ovi1602/HelpStud>

```

1.     public class WebService1 : System.Web.Services.WebService
2.     {
3.         static string conn = @"Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Ovi\Desktop\proiect\WebApp\
WebApp\App_Data\Database1.mdf;Integrated Security=True";
4.         SqlConnection myCon = new SqlConnection(conn);
5.         [WebMethod]
6.         public int logIn(string username, string password, ref int id, ref int isAd
min) {
7.             myCon.ConnectionString = conn;
8.             myCon.Open();
9.             string valString = "SELECT * FROM Users WHERE username = @val0 AND
password = @val1";
10.            SqlCommand comm = new SqlCommand();
11.            comm.Connection = myCon;
12.            comm.CommandText = valString;
13.            comm.Parameters.AddWithValue("@val0", username);
14.            comm.Parameters.AddWithValue("@val1", password);
15.            SqlDataReader sdr = comm.ExecuteReader();
16.
17.
18.            if (sdr.Read())
19.            {
20.                sdr.Close();
21.                SqlCommand IdCommand = new SqlCommand("SELECT id FROM Users WHERE
username = '" + username + "'", myCon);
22.                int lastId = (int)IdCommand.ExecuteScalar();
23.                SqlCommand Admin = new SqlCommand("SELECT isAdmin FROM Users WHERE
username = '" + username + "'", myCon);
24.                int IsAdmin = (int)Admin.ExecuteScalar();
25.                myCon.Close();
26.                isAdmin = IsAdmin;
27.                id = lastId;
28.                return lastId;
29.            }
30.            else
31.                return 0;
32.        }
33.        [WebMethod]
34.        public string addUser(string username, string password, string email, string
name)
35.        {
36.            try
37.            {
38.                myCon.ConnectionString = conn;
39.                myCon.Open();
40.
41.                //Checking if there is already an user with the same username

```

```

42.         string valString = "SELECT * FROM Users WHERE username = @val0";
43.         SqlCommand comm1 = new SqlCommand();
44.         comm1.Connection = myCon;
45.         comm1.CommandText = valString;
46.         comm1.Parameters.AddWithValue("@val0", username);
47.         SqlDataReader sdr = comm1.ExecuteReader();
48.         if (sdr.Read())
49.         {
50.             return "Username already exists";
51.         }
52.         sdr.Close();
53.         string valString2 = "SELECT * FROM Users WHERE email = @val1";
54.         SqlCommand comm2 = new SqlCommand();
55.         comm2.Connection = myCon;
56.         comm2.CommandText = valString2;
57.         comm2.Parameters.AddWithValue("@val1", email);
58.         SqlDataReader sdr2 = comm2.ExecuteReader();
59.         if (sdr2.Read())
60.         {
61.             return "Email already exists!";
62.         }
63.         sdr2.Close();
64.
65.         string commandText = "INSERT INTO Users (Id, username, password,
isAdmin, email,name) VALUES (@val0, @val1, @val2, @val3,@val4,@val5)";
66.
67.         //Next id
68.         int lastId;
69.         SqlCommand IdCommand = new SqlCommand("SELECT MAX(id) FROM Users",
myCon);
70.         lastId = (int)IdCommand.ExecuteScalar();
71.         SqlCommand comm = new SqlCommand();
72.         comm.Connection = myCon;
73.         comm.CommandText = commandText;
74.         comm.Parameters.AddWithValue("@val0", lastId + 1);
75.         comm.Parameters.AddWithValue("@val1", username);
76.         comm.Parameters.AddWithValue("@val2", password);
77.         comm.Parameters.AddWithValue("@val3", 0);
78.         comm.Parameters.AddWithValue("@val4", email);
79.         comm.Parameters.AddWithValue("@val5", name);
80.
81.
82.         comm.ExecuteNonQuery();
83.         myCon.Close();
84.         return "Succesfully registered";
85.     }
86.     catch (SqlException xe)
87.     {
88.         myCon.Close();
89.         return xe.ToString();
90.     }
91. }
92.
93. [WebMethod]
94. public bool addFaculty(string name)
95. {
96.     string commandText = "INSERT INTO Faculty (Id, name) VALUES (@val0,
@val1)";
97.     try
98.     {
99.         myCon.ConnectionString = conn;
100.        myCon.Open();
101.
102.        string valString = "SELECT * FROM Faculty WHERE name =
@val0";
103.        SqlCommand comm1 = new SqlCommand();

```

```

104.         comm1.Connection = myCon;
105.         comm1.CommandText = valString;
106.         comm1.Parameters.AddWithValue("@val0", name);
107.         SqlDataReader sdr = comm1.ExecuteReader();
108.         if (sdr.Read())
109.         {
110.             return false;
111.         }
112.         sdr.Close();
113.
114.         int lastId;
115.         SqlCommand IdCommand = new SqlCommand("SELECT MAX(id) FROM
Faculty", myCon);
116.         lastId = (int)IdCommand.ExecuteScalar();
117.         SqlCommand comm = new SqlCommand();
118.         comm.Connection = myCon;
119.         comm.CommandText = commandText;
120.         comm.Parameters.AddWithValue("@val0", lastId + 1);
121.         comm.Parameters.AddWithValue("@val1", name);
122.
123.         comm.ExecuteNonQuery();
124.         myCon.Close();
125.         return true;
126.     }
127.     catch (SqlException xe)
128.     {
129.         myCon.Close();
130.         return false;
131.     }
132. }
133. [WebMethod]
134. public bool deleteFaculty(string name)
135. {
136.     myCon.ConnectionString = conn;
137.     myCon.Open();
138.     try
139.     {
140.         string commandText = "DELETE FROM Faculty WHERE name =
@val0";
141.
142.         SqlCommand comm = new SqlCommand();
143.         comm.Connection = myCon;
144.         comm.CommandText = commandText;
145.         comm.Parameters.AddWithValue("@val0", name);
146.         comm.ExecuteNonQuery();
147.         myCon.Close();
148.         return true;
149.     }
150.     catch (Exception e)
151.     {
152.         return false;
153.     }
154. }
155. [WebMethod]
156. public bool updateFaculty(string newName, string oldName)
157. {
158.     myCon.ConnectionString = conn;
159.     myCon.Open();
160.     try
161.     {
162.         SqlCommand IdCommand = new SqlCommand("SELECT id FROM
Faculty WHERE name = '" + oldName + "'", myCon);
163.         int idFaculty = (int)IdCommand.ExecuteScalar();
164.         string commandText = "Update Faculty set name = @val0 WHERE
id = @val1";
165.         SqlCommand comm = new SqlCommand();

```

```

166.         comm.Connection = myCon;
167.         comm.CommandText = commandText;
168.         comm.Parameters.AddWithValue("@val0", newName);
169.         comm.Parameters.AddWithValue("@val1", idFaculty);
170.         comm.ExecuteNonQuery();
171.         myCon.Close();
172.         return true;
173.     }
174.     catch (Exception e)
175.     {
176.         return false;
177.     }
178. }
179. [WebMethod]
180. public string addPost(string username
181. , string title, string content, string course, string link)
182. {
183.     try
184.     {
185.         myCon.ConnectionString = conn;
186.         myCon.Open();
187.         SqlCommand IdCommand1 = new SqlCommand("SELECT id FROM Users
188. WHERE username = '" + username + "'", myCon);
189.         int idUser= (int)IdCommand1.ExecuteScalar();
190.         SqlCommand IdCommand2 = new SqlCommand("SELECT id FROM
191. Course WHERE name = '" + course + "'", myCon);
192.         int idCourse = (int)IdCommand2.ExecuteScalar();
193.         SqlCommand IdCommand3 = new SqlCommand("SELECT idFaculty
194. FROM Course WHERE name = '" + course + "'", myCon);
195.         int idFac = (int)IdCommand3.ExecuteScalar();
196.         string commandText = "INSERT INTO Posts (Id, Title, Content,
197. idUser, idFac, idCourse, sourceLink) VALUES (@val0, @val1, @val2, @val3, @val4,
198. @val5, @val6)";
199.
200.         int lastId;
201.         SqlCommand IdCommand = new SqlCommand("SELECT MAX(id) FROM
202. Posts", myCon);
203.         lastId = (int)IdCommand.ExecuteScalar();
204.         SqlCommand comm = new SqlCommand();
205.         comm.Connection = myCon;
206.         comm.CommandText = commandText;
207.         comm.Parameters.AddWithValue("@val0", lastId + 1);
208.         comm.Parameters.AddWithValue("@val1", title);
209.         comm.Parameters.AddWithValue("@val2", content);
210.         comm.Parameters.AddWithValue("@val3", idUser);
211.         comm.Parameters.AddWithValue("@val4", idFac);
212.         comm.Parameters.AddWithValue("@val5", idCourse);
213.         comm.Parameters.AddWithValue("@val6", link);
214.
215.         comm.ExecuteNonQuery();
216.         myCon.Close();
217.         return "Post added!";
218.     }
219.     catch (SqlException xe)
220.     {
221.         myCon.Close();
222.         return xe.ToString();
223.     }
224. }
225. public partial class AppUser : Form

```

```

225.         {
226.             WindowsFormsAppProject.ServiceReference1.WebService1SoapClient servi
ce = new WindowsFormsAppProject.ServiceReference1.WebService1SoapClient();
227.             static string fac;
228.             static int an;
229.             static int sem;
230.             protected int idUser;
231.             static string user;
232.             public AppUser(int idUser)
233.             {
234.                 this.idUser = idUser;
235.                 InitializeComponent();
236.                 LoadYearSemester();
237.                 LoadTitles();
238.                 comboBox1.Text = "Faculty";
239.                 textBoxNewPassword.PasswordChar = '*';
240.                 textBoxOldPassword.PasswordChar = '*';
241.
242.                 DataSet dsFaculties = new DataSet();
243.
244.                 dsFaculties = service.GetAllFaculties();
245.
246.                 try
247.                 {
248.
249.                     foreach (DataRow dr in dsFaculties.Tables[0].Rows)
250.
251.                     {
252.                         String name = dr.ItemArray.GetValue(1).ToString();
253.                         comboBox1.Items.Add(name);
254.                     }
255.                 }
256.
257.                 catch (Exception xe)
258.                 {
259.                     MessageBox.Show("No faculties");
260.                 }
261.             }
262.
263.             private void Form2_Load(object sender, EventArgs e)
264.             {
265.
266.             }
267.
268.             public void setUsername(string usern)
269.             {
270.                 user = usern;
271.                 fillInfo();
272.             }
273.             public void fillInfo()
274.             {
275.
276.                 string str = service.getUserData(user);
277.                 if (str.Length > 1)
278.                 {
279.                     string[] str1 = str.Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries);
280.                     labelName.Text = str1[0].ToString();
281.                     labelEmail.Text = str1[1].ToString();
282.                 }
283.                 labelUsername.Text = user;
284.                 labelUser.Text = user;
285.                 try
286.                 {
287.                     if (service.getImage(user) != null)
288.                     {

```

```

289.         byte[] img = service.getImage(user);
290.         MemoryStream ms = new MemoryStream(img);
291.         pictureBox1.Image = Image.FromStream(ms);
292.         pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
293.     }
294. }
295. catch
296. {
297.
298. }
299. }
300. public static string MD5Hash(string input)
301. {
302.     StringBuilder hash = new StringBuilder();
303.     MD5CryptoServiceProvider
md5provider = new MD5CryptoServiceProvider();
304.     byte[] bytes = md5provider.ComputeHash(new UTF8Encoding().GetByt
es(input));
305.
306.     for (int i = 0; i < bytes.Length; i++)
307.     {
308.         hash.Append(bytes[i].ToString("x2"));
309.     }
310.     return hash.ToString();
311. }
312. private void comboBox1_SelectedIndexChanged(object sender, EventArgs
e)
313. {
314.     fac = comboBox1.Text;
315.     comboBox4.Items.Clear();
316.     comboBox4.Text = "Course";
317. }
318.
319. private void comboBox2_SelectedIndexChanged(object sender, EventArgs
e)
320. {
321.     an = Convert.ToInt16(comboBox2.Text);
322.     comboBox4.Items.Clear();
323.     comboBox4.Text = "Course";
324. }
325.
326. private void comboBox3_SelectedIndexChanged(object sender, EventArgs
e)
327. {
328.     comboBox4.Items.Clear();
329.     sem = Convert.ToInt16(comboBox3.Text);
330.     DataSet dsCourses = new DataSet();
331.
332.     try
333.     {
334.         dsCourses = service.getCourse(fac, an, sem);
335.         foreach (DataRow dr in dsCourses.Tables[0].Rows)
336.         {
337.             String name = dr.ItemArray.GetValue(0).ToString();
338.             comboBox4.Items.Add(name);
339.         }
340.     }
341.     catch
342.     {
343.         //do nothing
344.     }
345. }
346.

```