

# Industrial Informatics 2023

## -Homework 2 –

The second Homework consists of a Web Application that has access to a database. The Web Service of this WebApp is then consumed by a Windows Form App.

The theme of the following project is a video game where players can create, view, and accept quests. Said quests grant

**Tema:** Creati un serviciu Web care ofera acces la o baza de date care stocheaza informatii. Apoi, creati un client de tip Windows Form App care utilizeaza serviciul. Serviciul va oferi metode de adaugarea si modificare a informatiilor. Tema aplicatiei este la alegere.

them prices which determine the player's rank in a leaderboard. Certain aspects of the application are simplified as they were unnecessary for presenting a general understanding of functionality of the project. Project will be nicknamed "QuestBringer".

### 1. QuestWebApp

First, the QuestWebApp is a ASP.NET Web Application which will provide a number of webservices, which also access the database: "QuestBringer". The QuestWebService has the following functions:

The screenshot displays the development environment for the QuestWebApp. On the left, the Visual Studio code editor shows the `QuestWebService.asmx.cs` file. The code defines a `QuestWebService` class that inherits from `System.Web.Services.WebService`. It includes a `SqlConnection` object `myCon` and a `WebMethod` `AddQuest` that inserts data into a database table named `Quests`. The code is as follows:

```

10 using System.Text;
11 using System.Threading.Tasks;
12 using System.Data;
13 using System.CodeDom.Compiler;
14
15 namespace QuestWebApp
16 {
17     /// <summary>
18     /// Summary description for QuestWebService
19     /// </summary>
20     [WebService(Namespace = "http://tempuri.org/")]
21     [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
22     [System.ComponentModel.ToolboxItem(false)]
23     // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the fo
24     // [System.Web.Script.Services.ScriptService]
25     0 references
26     public class QuestWebService : System.Web.Services.WebService
27     {
28         SqlConnection myCon = new SqlConnection();
29
30         [WebMethod]
31         0 references
32         public void AddQuest(string Title, string Description, int Tokens, int Badges, int Crea
33         {
34
35             myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog='D:\\SCHOOL
36             var procedure = string.Format("Insert Into Quests Values( '{0}', '{1}', {2}, {3}, {4
37
38             using (myCon)
39             {
40                 try

```

On the right, a web browser window shows the `QuestWebService` interface at `localhost:44304/QuestWebS`. The page lists the following operations supported by the service:

- [AcceptQuest](#)
- [AddQuest](#)
- [AddUser](#)
- [CheckUser](#)
- [Reward](#)
- [ShowLeaderboard](#)
- [ShowQuest](#)
- [ShowQuestBoard](#)
- [ShowUser](#)

Below the list, a note states: "This web service is using http://tempuri.org". A recommendation is provided: "Recommendation: Change the default name". The text explains that each XML Web service needs a unique namespace in development, but published XML Web services should have namespaces that look like URLs. It also mentions that for XML Web services created using ASP.NET, the default namespace is `http://microsoft.com/web`.

- AddQuest receives parameters with which it creates a new entry in the “Quests” Table
- AddUser receives parameters with which it creates a new entry in the “Users” Table
- CheckUser is part of the login process and identifies the user corresponding to the given parameters (username and password). It will either return the id of the identified user or ‘0’ which represents that no user was found with the given credentials
- ShowUser returns data of the user with a given id
- ShowQuest returns data of a quest with a given id
- ShowLeaderboard returns a list of all registered users ranked by their game currency
- ShowQuestBoard returns a list of all quest entries
- AcceptQuest will delete a quest of a given id, as a player/user “claims it”
- Reward adds the in-game currency related to the quest they claimed (through an update query)

```
[WebMethod]
0 references
public void AddQuest(string Title, string Description, int Tokens, int Badges, int CreatorId)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCE...\\QUESTWEBAPP\\QUESTWEBAPP\\APP_DATA\\QUESTS";
    var procedure = string.Format("Insert Into Quests Values( '{0}','{1}',{2},{3},{4} )", Title,Description,Tokens,Badges,CreatorId);

    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
            Console.WriteLine("Added Quest!");
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}

[WebMethod]
0 references
public void AddUser(string Username,string Password, int Tokens, int Badges )
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCE...\\QUESTWEBAPP\\QUESTWEBAPP\\APP_DATA\\QUESTS";
    var procedure = string.Format("Insert Into Users Values( '{0}','{1}',{2},{3} )", Username, Password, Tokens, Badges);
    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
            Console.WriteLine("Added User!");
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}
```

```
    Console.WriteLine("Added User!");
}
catch (Exception ex)
{
    Console.WriteLine(ex.Message);
}
myCon.Close();
}

[WebMethod]
0 references
public string[] ShowQuest(int id)
{
    string[] quest;//title,desc,tokens,badges, creator
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCE...\\QUESTWEBAPP\\QUESTWEBAPP\\APP_DATA\\QUESTS";
    var procedure = string.Format("Select Quests.Id, Title,Description,Quests.Tokens,Quests.Badges, Users.Username From Quests Join Users ON Quests.CreatorId = Users.Id WHERE Quests.Id = {0}", id);
    DataSet ds = new DataSet();
    myCon.Open();
    using (myCon)
    {
        SqlDataAdapter da = new SqlDataAdapter(procedure, myCon);
        da.Fill(ds,"Quests");
        DataTable dt = ds.Tables[0];
        DataRow row = dt.Rows[0];
        quest= new string[row.ItemArray.Length];
        for (int i= 0; i < row.ItemArray.Length; i++)
        {
            quest[i] = row[i].ToString();
            Console.WriteLine(quest[i] + row[i].ToString());
        }
    }
    myCon.Close();
    return quest;
}
```

```

0 references
public int CheckUser(string username, string password)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog='D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCESA\\
    int userId; //username,tokens,badges
    var procedure = string.Format("Select * From Users Where Username = '{0}' And Password = '{1}'", username,password);
    DataSet ds = new DataSet();
    myCon.Open();
    using (myCon)
    {
        SqlDataAdapter da = new SqlDataAdapter(procedure, myCon);
        da.Fill(ds, "Users");

        DataTable dt = ds.Tables[0];
        if (dt.Rows.Count > 0)
        {
            DataRow row = dt.Rows[0];
            userId = int.Parse(row[0].ToString());
            myCon.Close();
            return userId;
        }
        else
            return 0;
    }
}

[WebMethod]
0 references
public string[] ShowUser(int id)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog='D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCESA\\
    string[] user; //username,tokens,badges
    bool find = true; //change find after successful select
    var procedure = string.Format("Select *,RANK() OVER(ORDER BY Badges * 20 + Tokens DESC) 'Player_Rank ' From Users Where Id = {0}", id);
    DataSet ds = new DataSet();
    myCon.Open();
    using (myCon)
    {
        SqlDataAdapter da = new SqlDataAdapter(procedure, myCon);
        da.Fill(ds, "Users");
        DataTable dt = ds.Tables[0];
        DataRow row = dt.Rows[0];
        user = new string[row.ItemArray.Length];
        for (int i = 0; i < row.ItemArray.Length ; i++)
        {
            user[i] = row[i].ToString();
        }
    }
    myCon.Close();
    if (find)
    {
        return user;
    }
    else return null;
}

[WebMethod]
0 references
public string[] ShowLeaderboard()
{
    string[] rankings;
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog='D:\\SCHOOL BOOKS\\AN III SEMESTRU II\\INDUSTRIAL INFORMATICS\\LAB\\PROJECTACCESA\\
    var procedure = string.Format("Select Username, Badges, Tokens, RANK() OVER(ORDER BY Badges * 20 + Tokens DESC) 'Player_Rank ' from Users ");
    DataSet ds = new DataSet();
    myCon.Open();
    using (myCon)
    {
        SqlDataAdapter da = new SqlDataAdapter(procedure, myCon);
        da.Fill(ds, "Users");
        DataTable dt = ds.Tables[0];

        rankings = new string[dt.Rows.Count]; int i = 0;
        string[] temp = new string[dt.Rows[0].ItemArray.Length];
        foreach (DataRow row in dt.Rows)
        {
            for (int j = 0; j < row.ItemArray.Length ; j++)
            {
                temp[j] = row[j].ToString();
            }
            rankings[i] = string.Join(" ", temp);
            i++;
        }
    }
    return rankings ;
}

```

```
[WebMethod]
0 references
public string[] ShowQuestBoard()
{
    string[] quests;
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\VAN III SEMESTRU II\\INDUSTRIAL INFOR...";
    var procedure = string.Format("Select * From Quests");
    DataSet ds = new DataSet();
    // string questList = "";
    myCon.Open();
    using (myCon)
    {
        SqlDataAdapter da = new SqlDataAdapter(procedure, myCon);
        da.Fill(ds, "Quests");
        DataTable dt = ds.Tables[0];

        quests = new string[dt.Rows.Count]; int i=0;
        string[] temp =new string[dt.Rows[0].ItemArray.Length];
        foreach (DataRow row in dt.Rows)
        {
            for (int j = 0; j < row.ItemArray.Length - 1; j++)
            {
                temp[j] = row[j].ToString();
            }
            quests[i] = string.Join(" ", temp);
            i++;
        }
    }
    return quests;
}

[WebMethod]
0 references
public void AcceptQuest(int id)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\VAN III SEMESTRU II\\INDUSTRIAL INFOR...";
    var procedure = string.Format("DELETE FROM QUESTS WHERE Id = {0}",id);

    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}

[WebMethod] //Both AcceptQuest and Reward are called one after the other
0 references
public void Reward(int id, int tokens,int badges)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\VAN III SEMESTRU II\\INDUSTRIAL INFOR...";
    var procedure = string.Format("Update Users SET Tokens=(Tokens+{1}), Badges=(Badges+{2}) WHERE Id = {0}", id,tokens,badges);

    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}
```

```
[WebMethod]
0 references
public void AcceptQuest(int id)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\VAN III SEMESTRU II\\INDUSTRIAL INFOR...";
    var procedure = string.Format("DELETE FROM QUESTS WHERE Id = {0}",id);

    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}


[WebMethod] //Both AcceptQuest and Reward are called one after the other
0 references
public void Reward(int id, int tokens,int badges)
{
    myCon.ConnectionString = "Data Source=DESKTOP-C5G2Q55;Initial Catalog=\\D:\\SCHOOL BOOKS\\VAN III SEMESTRU II\\INDUSTRIAL INFOR...";
    var procedure = string.Format("Update Users SET Tokens=(Tokens+{1}), Badges=(Badges+{2}) WHERE Id = {0}", id,tokens,badges);

    using (myCon)
    {
        try
        {
            SqlCommand command = new SqlCommand(procedure, myCon);
            command.Connection.Open();
            command.ExecuteNonQuery();
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
    myCon.Close();
}
```

The Database consists of two tables, Users, and Quests, with the following fields:


QuestWebApp - dbo.Users [Data]

Update Script File: dbo.Users.sql

	Name	Data Type	Allow Nulls	Default
	 Id	int	<input type="checkbox"/>	
	Username	nchar(50)	<input type="checkbox"/>	
	Password	nchar(50)	<input type="checkbox"/>	
	Badges	int	<input checked="" type="checkbox"/>	
	Tokens	int	<input checked="" type="checkbox"/>	
			<input type="checkbox"/>	

QuestWebApp - dbo.Users [Design]

Update Script File: dbo.Users.sql

	Name	Data Type	Allow Nulls	Default
	 Id	int	<input type="checkbox"/>	
	Title	nchar(50)	<input type="checkbox"/>	
	Description	nvarchar(MAX)	<input type="checkbox"/>	
	Tokens	int	<input type="checkbox"/>	
	Badges	int	<input type="checkbox"/>	
	CreatorId	int	<input type="checkbox"/>	
			<input type="checkbox"/>	

We also start with the following data already added to the table:

dbo.Users [Data]

QuestWebService.asmx.cs

dbo.Users [Design]

Max Rows: 1000

	Id	Username	Password	Badges	Tokens
▶	1	Alex_The_Great ...	password ...	3	120
	2	Brave_Player ...	password1 ...	2	20
	3	Challenger ...	password2 ...	1	200
	4	QuestLord ...	password3 ...	4	0
	5	Alin ...	password4 ...	20	2
	6	Alex ...	password5 ...	1	10
	NULL	NULL	NULL	NULL	NULL

QuestWebApp - dbo.Quests [Data]

dbo.Quests [Data]

dbo.Quests [Design]

Max Rows: 1000

	Id	Title	Description	Tokens	Badges	CreatorId
▶	2	Escort ...	Protect the King...	40	1	2
	3	Missing person ...	Help find a mis...	20	0	1
	4	Slay the Dragon...	Go to the Moun...	80	0	1
	5	Explore Dunge...	The Dungeon t...	100	0	3
	6	Mine Pests ...	The mine in the...	20	2	3

As for an example, here we test the ShowQuest(id) function:

The screenshot shows the QuestWebService interface on the left and its XML response in a browser on the right.

**QuestWebService Interface:**

- Header: QuestWebService
- Link: Click [here](#) for a complete list of operations.
- Section: ShowQuest
- Test: To test the operation using the HTTP POST protocol, click the 'Invoke' button.
- Parameter Table:
 

Parameter	Value
id:	2
- Invoke button
- SOAP 1.1: The following is a sample SOAP 1.1 request and response. The placeholders show...
- POST /QuestWebService.asmx HTTP/1.1

**XML Response:**

```
<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://schemas.microsoft.com/2003/10/Serializer/Model" >
  <string>2</string>
  <string>Escort </string>
  <string>Protect the King on an escort mission</string>
  <string>40</string>
  <string>1</string>
  <string>Brave_Player </string>
</ArrayOfString>
```

## 2. QuestWinForm

QuestWinForm is a ASP.NET Windows Form that will connect to the QuestWebService and be the interface of the project. It consists of 4 forms: Login (sign in for the user or creating a new account ) , Form2 (Home/ User Page), Leaderboard (showing the users' rankings) and Questboard (showing the available quests and letting the user pick and accept a quest).

```
5 references
public partial class Login : Form
{
    QuestWinForm.ServiceReference1.QuestWebServiceSoapClient service = new QuestWinForm.ServiceReference1.QuestWebServiceSoapClient();
    2 references
    public Login()
    {
        InitializeComponent();
        pictureBoxTitle.Image = Image.FromFile("quest.png");
        pictureBoxUser.Image = Image.FromFile("adventurer.png");
    }

    1 reference
    private void loginButton_Click(object sender, EventArgs e)
    {
        if (!string.IsNullOrEmpty(usernameBox.Text))
        {
            if (!string.IsNullOrEmpty(passwordBox.Text)) {
                int id = service.CheckUser(usernameBox.Text.ToString(), passwordBox.Text.ToString());
                if (id != 0) {
                    Hide();
                    Form2 next = new Form2(id);
                    next.Show();
                }
                else MessageBox.Show("Incorrect Login Information!", "Warning");
            }
            else MessageBox.Show("Please enter your password to log in!", "Warning");
        }
        else MessageBox.Show("Please enter your username to log in!", "Warning");
    }

    1 reference
    private void registerButton_Click(object sender, EventArgs e)
    {
        if (!string.IsNullOrEmpty(usernameBox.Text))
        {
            if (!string.IsNullOrEmpty(passwordBox.Text))
            {
                int id;
                service.AddUser(usernameBox.Text.ToString(), passwordBox.Text.ToString(), 0, 0);
                id = service.CheckUser(usernameBox.Text.ToString(), passwordBox.Text.ToString());
                Hide();
                Form2 next = new Form2(id);
                next.Show();
            }
            else MessageBox.Show("Please create a password to register!.", "Warning");
        }
        else MessageBox.Show("Please create a username to register!.", "Warning");
    }
}
```

```

m2.cs [Design]  Questboard.cs  Questboard.cs [Design]  Leaderboard.cs  Form2.cs  Lo
QuestWinForm
14 namespace QuestWinForm
15 {
16     11 references
17     public partial class Form2 : Form
18     {
19         QuestWinForm.ServiceReference1.QuestWebServiceSoapClient service = new QuestWinForm.ServiceReference1.QuestWebServiceSoapClient();
20         int id;
21         0 references
22         public Form2()
23         {
24             InitializeComponent();
25         }
26         4 references
27         public Form2(int id)
28         {
29             this.id = id;
30             InitializeComponent();
31             pictureBoxUser.Image = Image.FromFile("adventurer.png");
32             pictureBoxTitle.Image = Image.FromFile("quest.png");
33             // pictureBoxRank.Image = Image.FromFile("leaderboard.png");
34             pictureBoxBadge.Image = Image.FromFile("badge.png");
35             pictureBoxBadge1.Image = Image.FromFile("badge.png");
36             pictureBoxBadge2.Image = Image.FromFile("badge.png");
37             pictureBoxToken.Image = Image.FromFile("token.png");
38             pictureBoxToken1.Image = Image.FromFile("token.png");
39             pictureBoxToken2.Image = Image.FromFile("token.png");
40             tabControl.SelectedTab = Home;
41             ArrayOfString userInfo;
42             userInfo = service.ShowUser(id);
43             usernameBox.Text = userInfo[1];
44             badgeCount.Text = userInfo[3];
45             tokenCount.Text = userInfo[4];
46         }
47         1 reference
48         private void Form2_Load(object sender, EventArgs e)
49         { }
50         1 reference
51         private void logoutButton_Click(object sender, EventArgs e)
52         {
53             Hide();
54             Login next = new Login();
55             next.Show();
56         }
57         1 reference
58         private void quest_Click(object sender, EventArgs e)
59         {
60             tabControl.SelectedTab = Quest_Maker;
61             badgeBox.Text = "0";
62             tokenBox.Text = "0";
63             titleBox.Text = "0";
64         }
65         1 reference
66         private void postQuestButton_Click(object sender, EventArgs e)
67         {
68             if (int.Parse(badgeBox.Text) <= int.Parse(badgeCount.Text) && int.Parse(tokenBox.Text) <= int.Parse(tokenCount.Text)) {
69                 var result = MessageBox.Show("Are you sure you want to post this Quest? It cannot be deleted afterwards!", "Confirm",
70                     MessageBoxButtons.YesNo,
71                     MessageBoxIcon.Question);
72                 if (result == DialogResult.Yes)
73                 {
74                     service.AddQuest(titleBox.Text.ToString(), taskBox.Text.ToString(), int.Parse(tokenBox.Text), int.Parse(badgeBox.Text), id);
75                 }
76                 else MessageBox.Show("Insufficient resources for reward.", "Warning");
77             }
78         }
79         1 reference
80         private void leaderboardButton_Click(object sender, EventArgs e)
81         {
82             Hide();
83             Leaderboard next = new Leaderboard(id);
84             next.Show();
85         }
86         1 reference
87         private void questboardButton_Click(object sender, EventArgs e)
88         {
89             Hide();
90             Questboard next = new Questboard(id);
91             next.Show();
92         }
93     }
94 }

```

```

namespace QuestWinForm
{
    5 references
    public partial class Leaderboard : Form
    {
        QuestWinForm.ServiceReference1.QuestionServiceSoapClient service = new QuestWinForm.ServiceReference1.QuestionServiceSoapClient();
        int id;
        0 references
        public Leaderboard()
        {
        }
        1 reference
        public Leaderboard(int id)
        {
            this.id = id;
            InitializeComponent();

            pictureBoxCrown.Image = Image.FromFile("first.png");
            pictureBoxLeaderboard.Image = Image.FromFile("leaderboard.png");

            //add user rankings
            ArrayOfString leaderboard;
            leaderboard = service.ShowLeaderboard();
            string space;
            foreach (string row in leaderboard) {
                string[] splitrow = row.Split(';');
                space = new String(' ', 32 - splitrow[0].Trim().Length);
                listBoxRank.Items.Add(splitrow[0].Trim() + space + splitrow[2].Trim() + new String(' ', 16 - splitrow[2].Trim().Length) + splitrow[1].Trim() + new String(' ', 16 - splitrow[1].Trim().Length) + "*" + splitrow[3].Trim());
            }
        }

        1 reference
        private void homeButton_Click(object sender, EventArgs e)
        {
            Hide();
            Form2 next = new Form2(id);
            next.Show();
        }
    }
}

```

```

5 references
public partial class Questboard : Form
{
    QuestWinForm.ServiceReference1.QuestionServiceSoapClient service = new QuestWinForm.ServiceReference1.QuestionServiceSoapClient();
    int id;
    ArrayOfString questboard;
    int questId;
    0 references
    public Questboard()
    {
        InitializeComponent();
    }

    1 reference
    public Questboard(int id)
    {
        this.id = id;
        InitializeComponent();
        pictureBoxQuest.Image = Image.FromFile("questboard.png");

        questboard = service.ShowQuestBoard();
        string space;
        foreach (string row in questboard)
        {
            string[] splitrow = row.Split(';');
            space = new String(' ', 60 - splitrow[1].Trim().Length);
            listBoxQuests.Items.Add(splitrow[1].Trim() + space + splitrow[3].Trim() + " Tokens " + splitrow[4].Trim() + " Badges");
        }

        pictureBoxTitle.Image = Image.FromFile("quest.png");
        pictureBoxBadge2.Image = Image.FromFile("badge.png");
        pictureBoxToken2.Image = Image.FromFile("token.png");
    }

    1 reference
    private void pickQuestButton_Click(object sender, EventArgs e)
    {
        string[] splitrow = questboard[listBoxQuests.SelectedIndex].Split(';');
        questId = int.Parse(splitrow[0]);
        tabControl1.SelectedTab = QuestPage;

        ArrayOfString questInfo;
        questInfo = service.ShowQuest(questId);
        titleBox.Text = questInfo[1];
        taskBox.Text = questInfo[2];
        tokenBox.Text = questInfo[3];
        badgeBox.Text = questInfo[4];
        creatorBox.Text = questInfo[5];
    }
}

```

```

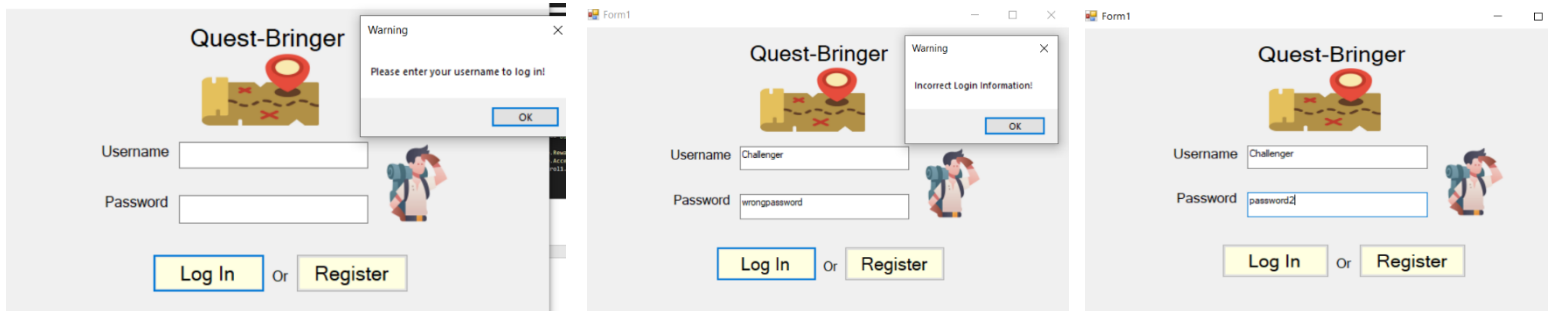
1 reference
private void homeButton_Click(object sender, EventArgs e)
{
    Hide();
    Form2 next = new Form2(id);
    next.Show();
}

1 reference
private void takeQuestButton_Click(object sender, EventArgs e)
{
    var result = MessageBox.Show("Are you sure you want to accept this Quest?", "Confirm",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Question);

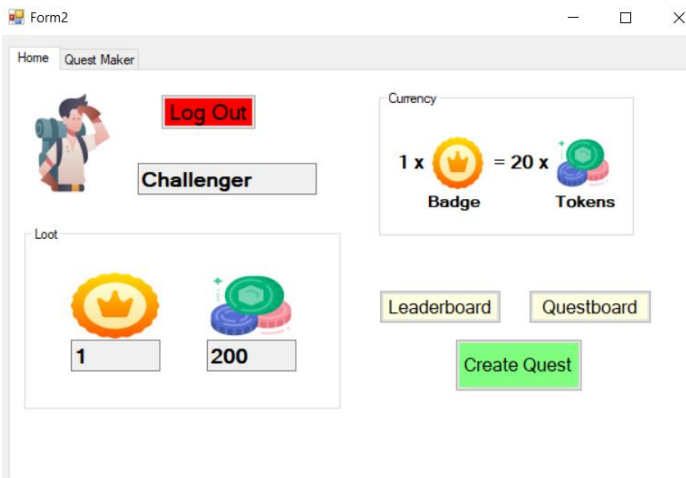
    if (result == DialogResult.Yes)
    {
        service.Reward(id, int.Parse(tokenBox.Text), int.Parse(badgeBox.Text));
        service.AcceptQuest(questId);
        tabControl1.SelectedTab = BoardPage;
    }
}
}

```

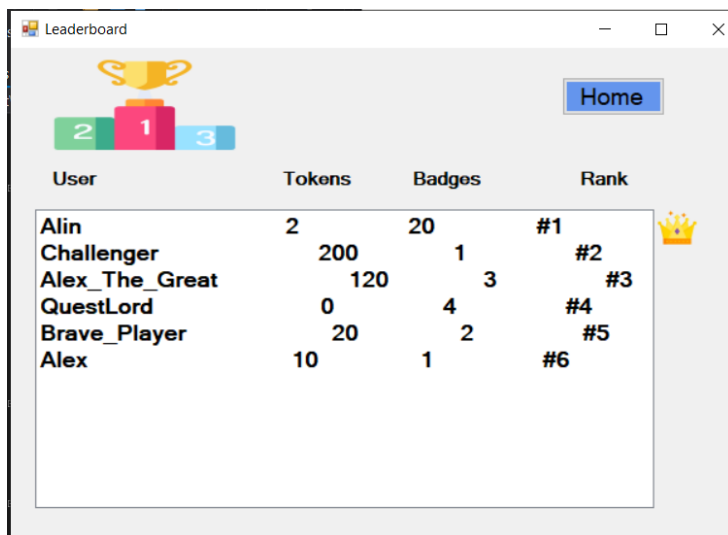
The Login will ask for a user's username and password. Failing to fill one of the fields prompts a warning message. If the user tries to log in with wrong information, a warning prompt will appear as well.



Once we are logged in, we have the options to logout, see the leaderboard, see the questboard or create a quest.



The leaderboard will be displayed with a listBox in descending order. The "Home" button brings us back to the user menu.





Questboard

Quest

Title:

Lost Treasure


Description

Find the lost treasure of the queen near the Southern beach.

Rewards

Tokens: 10

Badge: 1



Made by

Challenger

Accept

Questboard
Quest

Title:
Slay the Dragon

Description
Go to the Mountain Village and kill the dragon

Rewards

Tokens:
80

Badge:
0


Made by
Alex\_The\_Great

Accept

Confirm
Are you sure you want to accept this Quest?
Yes No

Before accepting a quest, we confirm it. Then after exiting the form and reentering, the quest will no longer be present on the QuestBoard. We can also see the rewards added to the user's Loot section:



Home
Quest Maker



Log Out


Challenger


Currency

1 x  = 20 x 

Badge
Tokens

Loot


1


280

Leaderboard
Questboard

Create Quest

Questboard
Quest

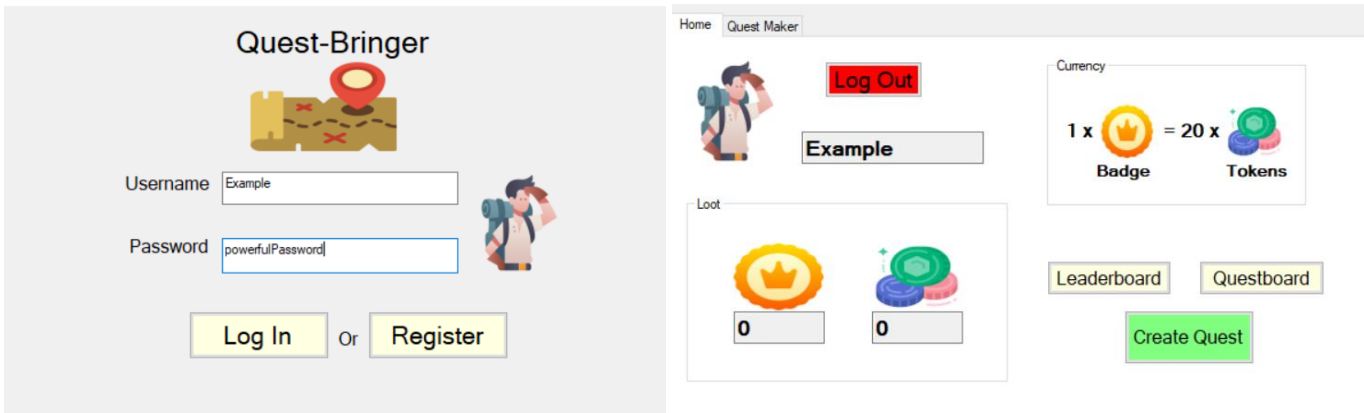


### Available Quests

Title	Reward
Escort	40 Tokens 1 Badges
Missing person	20 Tokens 0 Badges
Explore Dungeon	100 Tokens 0 Badges
Mine Pests	20 Tokens 2 Badges
Lost Treasure	10 Tokens 1 Badges

Home
Pick Quest

Finally, we log out, enter new user data and register, seeing that a new user starts with no loot:



Here are the Tables at the end of all the previous operations:

Id	Username	Password	Badges	Tokens
1	Alex_The_Great ...	password ...	3	120
2	Brave_Player ...	password1 ...	2	20
3	Challenger ...	password2 ...	1	280
4	QuestLord ...	password3 ...	4	0
5	Alin ...	password4 ...	20	2
6	Alex ...	password5 ...	1	10
7	Example ...	powerfulPasswo...	0	0
8	NULL	NULL	NULL	NULL

Id	Title	Description	Tokens	Badges	CreatorId
2	Escort ...	Protect the King...	40	1	2
3	Missing person ...	Help find a mis...	20	0	1
5	Explore Dunge...	The Dungeon t...	100	0	3
6	Mine Pests ...	The mine in the...	20	2	3
9	Lost Treasure ...	Find the lost tre...	10	1	3
10	NULL	NULL	NULL	NULL	NULL

### 3. Conclusions

As presented, the application allows a user to log in and interact with the quest system through the webservice functions. The WebApp offers the ability to insert into, update, select and delete from the given tables, using primarily the id of the Users entry and of the Quests entry. Besides improvements to the visual aspect of the interface, there are several features that could be implemented in the future:

- Users able to modify their username or password
- Creating a quest consumes your in-game currency
- Admin user-type
- User not being allowed or accept quests made by themselves