# Teamwork Project Assignment for the [ASP.NET MVC Course @ SoftUni](https://softuni.bg/courses/asp-net-mvc/)

This teamwork project assignment is designed to develop skills for creating dynamic data-driven Web applications using ASP.NET MVC and deployment in cloud environment like Azure and AppHarbor.

## Project Description

Design and implement ASP.NET MVC Web application. Deploy the Web application in a cloud environment. The application can be a social network, online gaming site or any other Web application by your choice.

The application should have **public part** (accessible without authentication), **private part** (available for registered users) and **administrative part** (available for administrators only).

## General Requirements

Your Web application should use the following technologies, frameworks and development techniques:

* The application must be implemented using **ASP.NET MVC** framework.
* Use **Visual Studio 2013** (update 4 is recommended).
* Use **Razor** template engine for generating the UI.
  + Rendering with ASP.NET Web Forms is not allowed.
  + Use **sections** and **partial views**.
  + Use **editor** and **display templates**.
* Use **Microsoft SQL Server** as database back-end.
* Use **Entity Framework 6** to access you database.
  + Obligatory use **Repository** and **Unit of Work** patterns.
* Use **MVC Areas** to separate different parts of your application (e.g. area for administration).
* Adapt the default **ASP.NET MVC site template** from Visual Studio 2013 or get another free theme.
  + Use responsive design based on **Twitter Bootstrap**.
* Use the standard **ASP.NET Identity System** for managing **users** and **roles**.
  + Your registered users should have at least one of these roles: **user** and **administrator**.
* Use **AJAX** request to asynchronously load and display datasomewhere in your application.
* Use **SignalR** communication somewhere in your application.
* Write **unit tests** for your logic, controllers, actions, helpers, etc.
* Implement **error handling** and **data validation** to avoid crashes when invalid data is entered (both **client-side** and **server-side**).
* Handle correctly the special HTML characters and tags like **<br />** and **<script> (escape special characters).**
* **Use Ninject (or any other dependency injection container).**
* **Use AutoМapper.**
* **Prevent from security vulnerabilities like SQL Injection, XSS, XSRF, parameter tampering, etc.**
* Host the application in a **cloud environment**, e.g. in **AppHarbor** or **Azure**.
* Use a **file storage cloud API**, e.g. **Dropbox**, **Google Drive** or other for storing the files.

### Public Part

The **public part** of your application should be **visible** **without authentication**. This public part could be the application start page, the user login and user registration forms, as well as the public data of the users, e.g. the public offers in a bid system, the product list in an e-commerce system, etc.

### Private Part (User Part)

**Registered users** should have personal area in the Web application accessible after **successful login**. This area could hold for example the user's **profiles** management functionality, the user's offers in a bid system, the user's posts in a blog system, the user's photos in a photo sharing system, the user's contacts in a social network, etc.

### Administration Part

**System administrators** should have administrative access to the system. They should be able to administer all major information objects in the system, e.g. create / edit / delete users and other administrators, edit / delete offers in a bid system, edit / delete photos and album in a photo sharing system, edit / delete posts in a blogging system, edit / delete products and categories in an e-commerce system, etc.

## Additional Requirements

* Follow the best practices for OO design and **high-quality code** for the Web application:
  + Use data encapsulation.
  + Use exception handling properly.
  + Use inheritance, abstraction and polymorphism properly.
  + Follow the principles of strong cohesion and loose coupling.
  + Correctly format and structure your code, name your identifiers and make the code readable.
* Well looking user interface (UI).
* Good usability (easy to use UI).
* Supporting of all modern Web browsers.
* Use caching where appropriate.
* Use a **source control system** by choice, e.g. Git, SVN, GitHub, CodePlex.
  + Submit a link to your public source code repository.

## Public Project Defense

Each team will have to deliver a **public defense** of its work in front of the other students, trainers and assistants. Teams will have **only 15 minutes** for the following:

* **Demonstrate** how the application works (very shortly).
* Show the **source code** and explain how it works.
* Explain how each team member has **contributed**: display the commit logs in the Source Control system you are using.
* Optionally you might prepare a **presentation** (3-4 slides).

Please be **strict in timing**! On the 15th minute you **will be interrupted**! It is good idea to leave **the last 2-3 minutes for questions** from the other students, trainers and assistants.

Be **well prepared** for presenting maximum of your work for minimum time. Bring your own laptop. Test it preliminary with the multimedia projector. Open the project assets beforehand to save time.

## Assessment Criteria

* **Functionality** – **0…20**
* **Implementing controllers correctly** (controllers should do only their work) **– 0...5**
* **Implementing views correctly** (using display and editor templates) **– 0…5**
* **Unit tests** (unit test for some of the controllers using mocking) **– 0…10**
* **Security** (prevent SQL injection, XSS, CSRF, parameter tampering, etc.) **– 0…5**
* **File storage** (cloud file storage) **– 0…5**
* **Data validation** (validation in the models and input models) **– 0…10**
* **Hosting the application in the cloud – 0…5**
* **Using auto mapper and inversion of control – 0…5**
* **Using areas with multiple layouts – 0…10**
* **Code quality** (well-structured code, following the MVC pattern, following SOLID principles, etc.) – **0…10**
* **Teamwork**\* (source control; each team member contributed in 5 different days; distribution of tasks) – **0…5**
* **Bonus** (bonus points are given for exceptional project) – **0..5**

\* If not all team members have contributed to the project, this **does not affect** the teamwork points.

## Give Feedback about Your Teammates

You will be invited to **provide feedback** about all your teammates, their attitude to this project, their technical skills, their team working skills, their contribution to the project, etc. The feedback is important part of the project evaluation so **take it seriously** and be honest.