This document describes the **project assignment** for the **ASP.NET MVC** course at Telerik Academy.

# Project Description

Design and implement an **ASP.NET MVC application**. It can be a discussion forum, blog system, e-commerce site, online gaming site, social network, 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:

* Use **ASP.NET MVC 5** and **Visual Studio 2013**
* Have at least **15 controllers**
* Have at least **35 actions**
* You should use **Razor** template engine for generating theUI
  + You may use **Kendo UI widgets** (with the ASP.NET MVC Wrappers)
  + ASP.NET WebForms is **not** allowed
  + Use at least **3 sections** and at least **10 partial views**
  + Use at least **15 editor or display templates**
* Use **MS SQL Server** as database back-end
* Use **Entity Framework 6** to access your database
  + Using **Unit of Work** and **Repository pattern** is a must
* Use at least **two MVC Area** in your project (e.g. for administration)
* Create **tables with data** with **server-side paging** and **sorting for every model entity**
  + You can use Kendo UI Grid or generate your own HTML tables
* Adapt the **default ASP.NET MVC site template** from Visual Studio 2013 or get another free theme
  + Use responsive design based on **Twitter Bootstrap**
  + You may change the standard theme and modify it to apply own web design and visual styles
* Use the standard **ASP.NET Identity System** for managing **users** and **roles**
  + Your registered users should have at least one of the two roles: **user** and **administrator**
* Use **AJAX form and/or SignalR communication in some parts of your application**
* Write **unit tests** for your logic, controllers, actions, helpers, etc.
* Apply **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 />**
* Use **Ninject (or any other dependency container)** and **Automapper**
* Use **proper architecture** for your application
* **Prevent yourself** from security holes (XSS, XSRF, Parameter Tampering, etc.)
* Use **GitHub** for source control system

## Public Part

The **public part** of your projects 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 blog posts in a blog system, the public offers in a bid system, the products in an e-commerce system, etc.

## Private Part (User Area)

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

## Other Requirements

* Nice looking UI supporting of all modern Web browsers
* Good usability (easy to use UI)
* Originality of the idea
* Use caching where appropriate

## Deliverables

Put the following in a **ZIP archive** and submit it:

* The **source code** (Controllers, Views, Models, C# files, images, scripts, styles, etc.)
* **Don't submit the** **NuGet packages**! They are not needed and take too much disk space.
* Optional: brief documentation (few sentence **readme** file).

## Public Project Defense

Each student will have to make a **public defense** of its work to the trainers (in 15 minutes). It includes:

* Live **demonstration** of the developed Web application (please prepare your project on your computer and feed it with sample data).
* Explain application structure and its **source code**: Controllers, Views, Data Models, C# code, etc.
* Show the **commit logs** in the source control repository to prove your contribution from the beginning to the end.