CSE446 Software Integration and Engineering
Spring 2021

Instructor: Dr. Yinong Chen

Project 1 (100 Points)

Introduction

The aim of this project is to make sure that you have read the lecture slides and text, and you have understood the concepts covered in the lectures and in the text, including basic WCF services, advanced WCF services, and RESTful services. By the end of this project, you should have a good understanding of these concepts and have applied these concepts in developing operational programs.

Preparation and Practice Exercises (No submission required)

1. Read Textbook Chapter 3 if you have not taken the course CSE445 Distributed Software Development.

2. Read Textbook Chapter 7 to prepare for this project.

3. Download and install Visual Studio 2019 Professional/Enterprise Edition. It is fine if you have VS 2017. You may also use the computer in CIDSE lab in BYENG 114, where the required software is installed. Note, when you install VS, choose all options related to C# web services and web application to install.

4. Get familiar with the service hosting. Follow textbook chapter 3 to develop the basic-three service. You can choose to use IIS Express to host the service, which will provide a localhost port number. To test if your project works on a different computer, send the zipped project file to another computer, unzip the file and test the project on that machine. Service oriented software is distributed. It may have saved different parts in different locations. You need to make sure that you submit all the parts required.

5. Practice exercises: Questions in Textbook Chapter 7, Section 7.6 multiple choice questions. You can find the solution key in the course web page. These questions help you to prepare the quizzes and the multiple choice questions in the exam.

6. Follow Text Section 7.1.1 and repeat the development process of the self-hosting service given in the section. If the Visual Studio Version used in the text is different from the version you use, try to find the differences and complete the process. You may find the updated instruction from MSDN library online.

7. Follow Text Section 7.1.2 and repeat the development process of the console application that invokes the service developed in Section 7.1.1.

8. Follow Text Sections 7.3.3 and 7.3.4, and repeat the development process of the RESTful services

9. Follow Text Section 7.3.5 and repeat the development process of the clients that consume the RESTful services.

Assignment 1   WCF Services and Application (Submission Required)

In this part of the project, you will implement the number guessing services and number guessing games using different techniques. You are given the class definition below as the basic functions required in the number guessing services.

```
public class NumberGuess {
  public int SecretNumber(int lower, int upper) {
        DateTime currentDate = DateTime.Now;
        int seed = (int)currentDate.Ticks;
        Random random = new Random(seed);
        int sNumber = random.Next(lower, upper);
        return sNumber;
  }
   public string checkNumber(int userNum, int SecretNum) {
          if (userNum == SecretNum)
             return "correct";
          else
             if (userNum > SecretNum)
                 return "too big";
             else return "too small";
   }
}
```

1. Based on the above given class, write a WCF .svc service with two operations on localhost of .Net IIS Express server. You may want to review Text Section 3.2 before completing this assignment.   [10 points]

2. Create an ASP .Net Windows Forms Application to consume the .svc service in question 1. Visual Studio 2019 and 2017 project options are shown in Figure 1(a) and (b), respectively.

   Choose Windows Forms App (.Net Framework). Do not choose Windows Forms App (.Net Core).

(a) Search for Windows Forms App (.Net Framework) template in Visual Studio 2019

(b) Choose Windows Forms App (.Net Framework) template in Visual Studio 2017

Figure 1. Creating a Windows Forms App project in (a) VS 2019 and (b) VS 201

Then, you can use the Toolbox controls to create the GUI design, as shown in Figure 2.
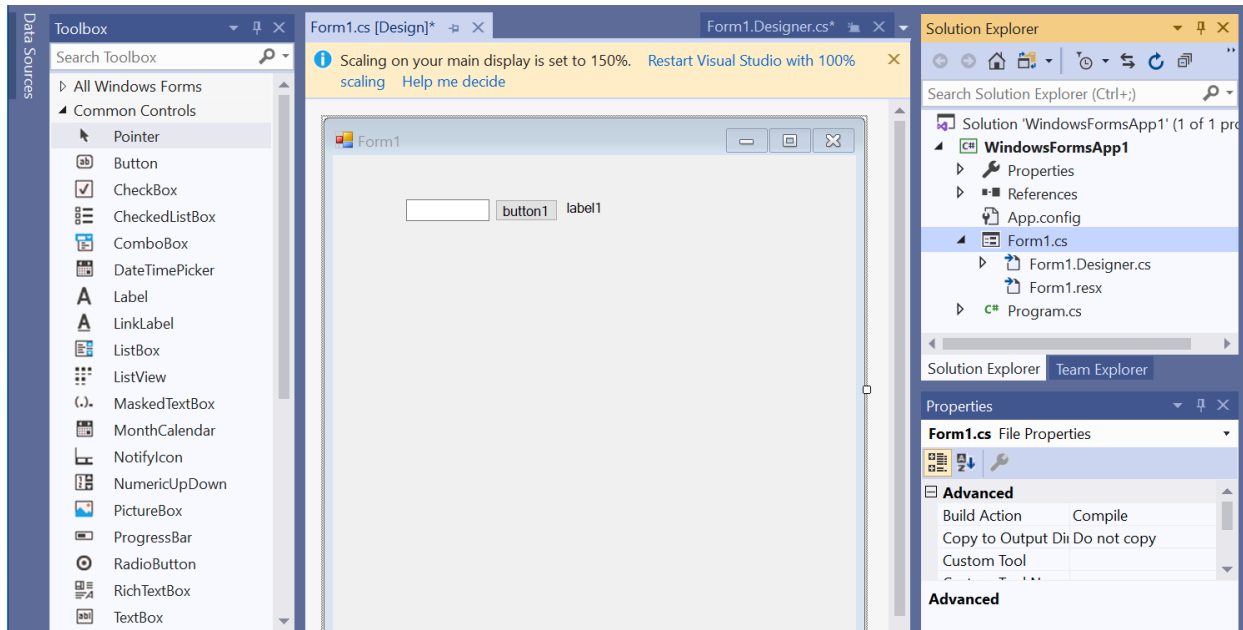
Figure 2. Design View in Visual Studio 2019

The application GUI must contain at least the following items, as shown in Figure 3. You can add additional components in your design.   [15 points]

Figure 3. Number Guessing Game Basic GUI Design

- The first two textboxes are used for a player to enter the lower and upper limits (integers) of the random number to be generated;
- The code behind the button "Generate a Secret Number" will generate a secret number and save it as a state;
- The last textbox allows the player to enter an integer to make a guess of the secret number.
- A label (attempts) that displays how many attempts have been made after a secret number is generated.
- A label (The number is) that displays if the given number is too small, too big, or a correct guess.

3. Create a self-hosing service to implement the NumberGuess class, with the SecretNumber and CheckNumber as operations. The program must include the service and the hosting code.   [20 points]

4. Write a console application to call the self-hosting service and to display the output of the service and to play the game: You must enter the guess number from the console to make the guess. The application must use svcutil.exe to generate the proxy.   [5 points]

Assignment 1 submission requirement:

Questions 1 and 2 must be in one solution, and questions 3 and 4 must be in another solution. Each solution folder must contain all the files. Put the two solution folders into another folder. Zip the folder (a single file) for submission. Testing results (with screenshots) must be submitted in a Word document or in PDF. You must indicate in the document the Visual Studio version (2019 or 2017) and the template that you use, so that the TA can use the same tools to test your program.

Canvas submission notice: The assignment consists of multiple distributed projects and components. They may be stored in different locations on your computer when you create them. You must copy these projects into the submission folder for Canvas submission. To make sure that you have all the files included in the zip file and they work together, you must test them on a different computer before submission. You must also download your own submission from the Canvas. Unzip the file on a different location or computer, and test your assignment and see if you can run the solution in a different location, because the TA will test your application on a different machine.

Assignment 2   Advanced Service and Application Architecture (Submission Required)

In this assignment, you will continue to work on the project that you created in assignment 1.

5. Re-implement the NumberGuess service as a RESTful service. Host the service on IIS Express server. For each service operation in the service developed in the question above, add UriTemplate in the attribute [WebGet], so that the service operation names are not exposed to the callers of the service. You can start from the following template to create a RESTful service.   [20 points]
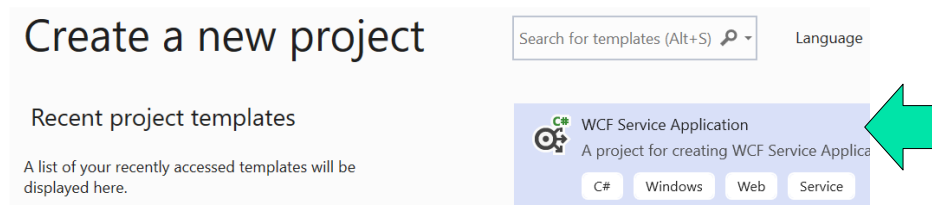
6. Choose **one** of the following assignment options to implement your application architecture.   [30 points]

6.1 Option 1: Create an MVC Web Application. You can choose ASU .Net Core Web Application or ASP .Net Web Application (.Net Framework). Please read Text Section 5.8.2 and Lecture 1-6 slides) to consume the RESTful service in question 6. You must create at least one asynchronous call to access the service using BeginGetResponse and a callback function. Please read text section 7.3.7 for an example. The following diagrams in Figure 4 (a) and (b) show some of the steps of creating an MVC Web application in VS 2019 and in VS 2017. If you use a different version, you may need to use different variation of the steps following MSDM documents.
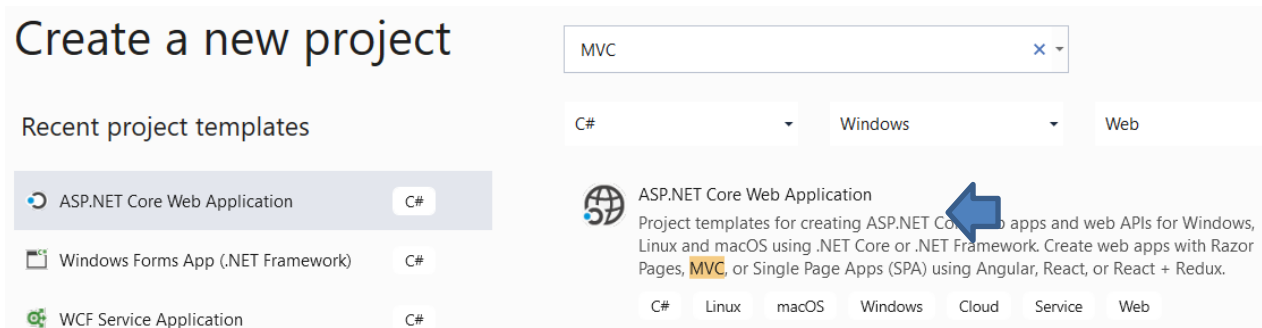
(a) Use Visual Studio 2019
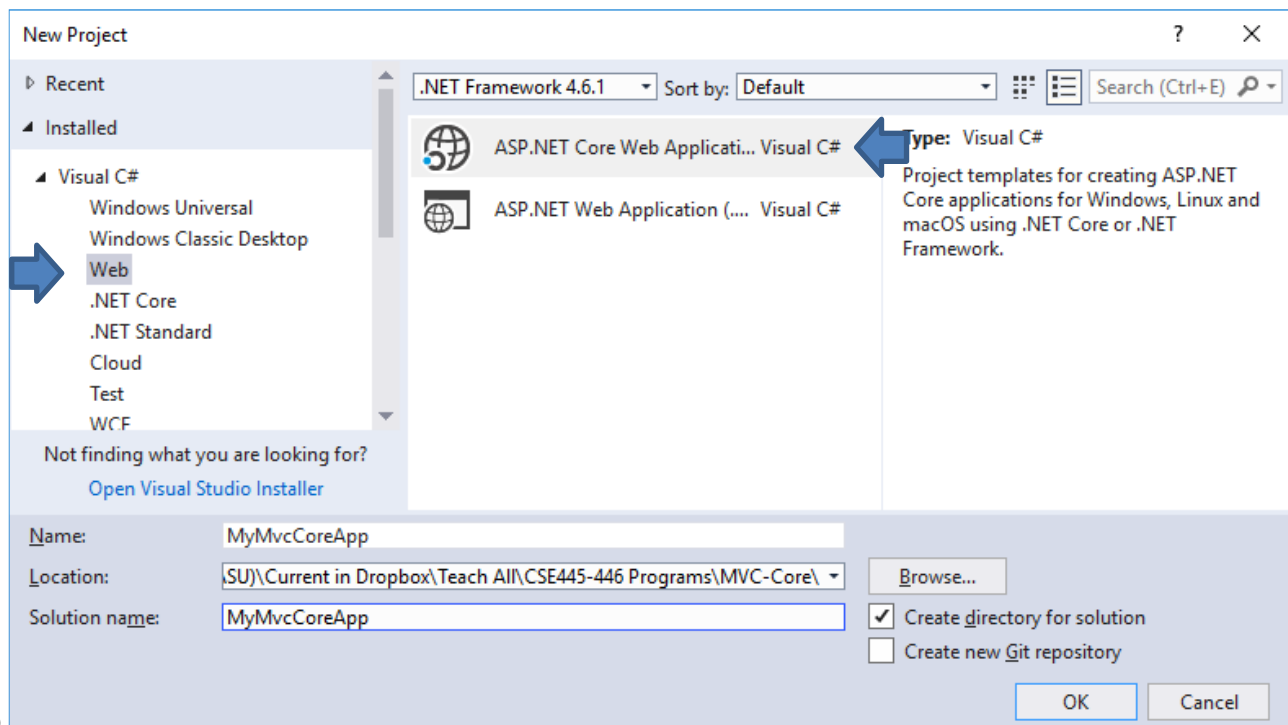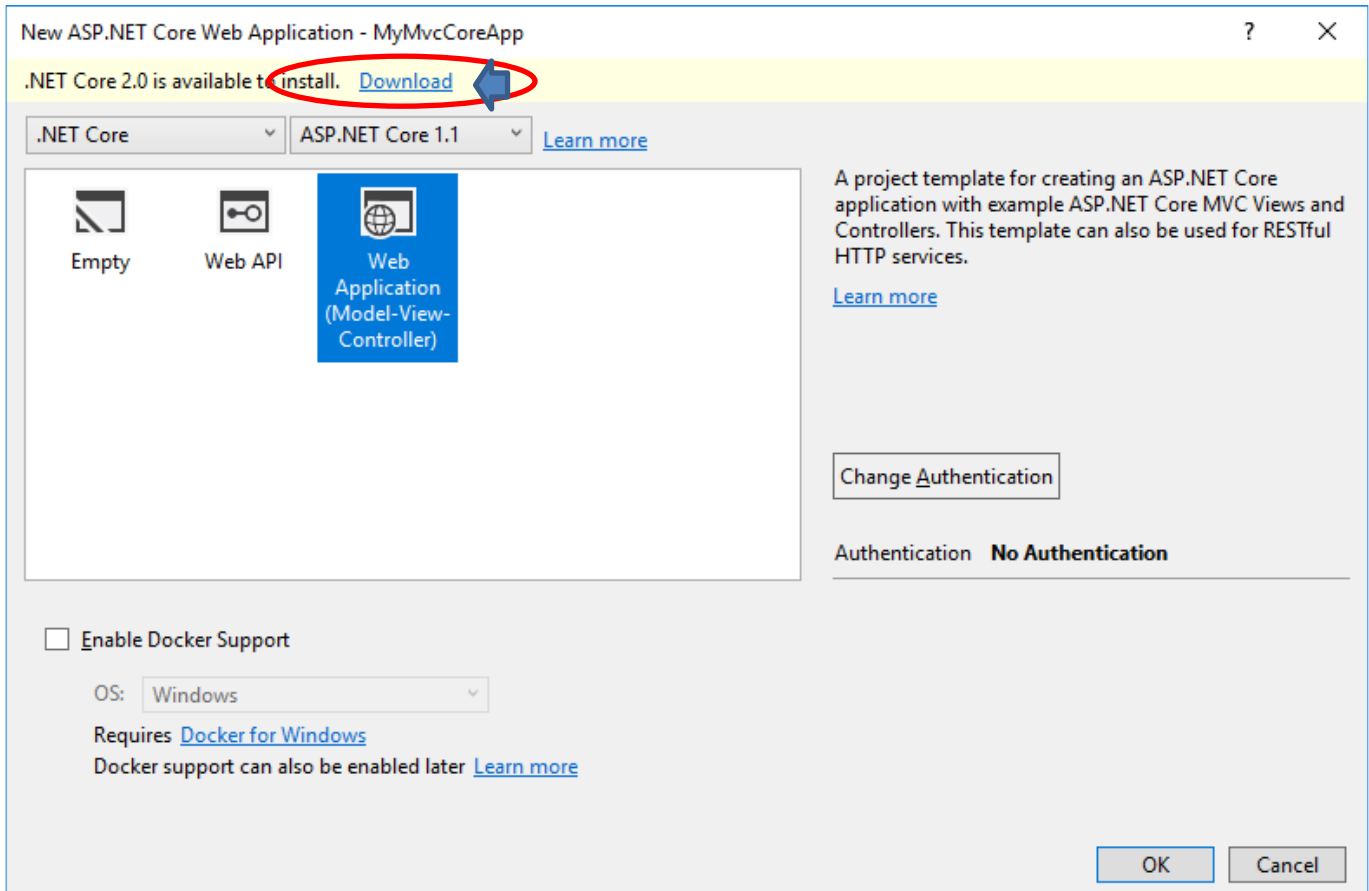

(a)

(b) Use Visual Studio 2017

Figure 4. Creating a Web App project in VS 2019(a) and in VS 2017(b)

If you use ASP .Net Core Web Application in VS 2017, you may need to download .Net Core, as shown in the following diagram.

To get started with .NET Core 2.0 in Visual Studio 2017 follow the steps

- Install .NET Core 2.0 - https://www.microsoft.com/net/core
- Restart Visual Studio 2017

You can host the application on IIS Express. The application GUI must contain at least the items that are shown in Figure 3.

6.2 Option 2: Create an HTML5 Web Application to consume the RESTful service in question 6. You are required to create graphic user interface. Creating animation is optional.

In this option, you will read additional reference materials to learn HTML5 programming model and HTML5 graphics libraries. There is a short introduction in Text Section 4.6.5. A sample code is given that shows the creation of graphic user interface and animation in HTML5.

You can use different platforms to create HTML Web application. You can use Visual Studio or use a Java environment. For example, you can follow the tutorial given at the link:
https://netbeans.org/kb/docs/webclient/html5-gettingstarted.html
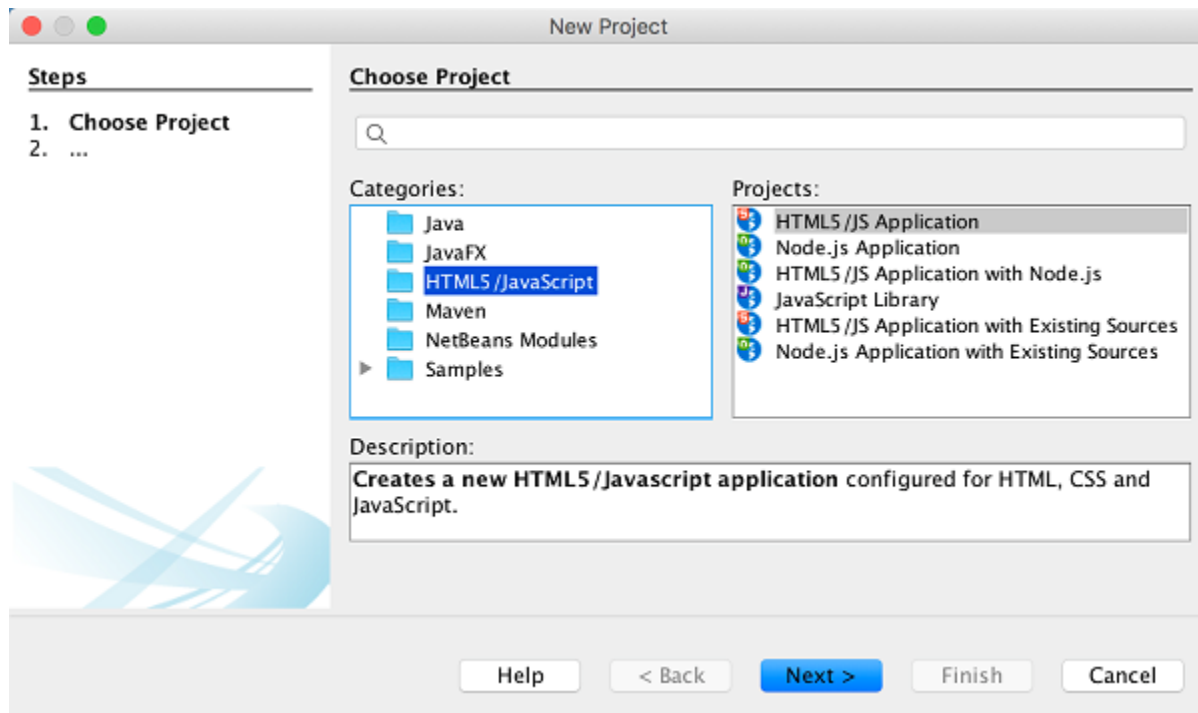
You can also use Visual Studio to implement this question. Choose New Project…, Web, and then Single Page Application template:
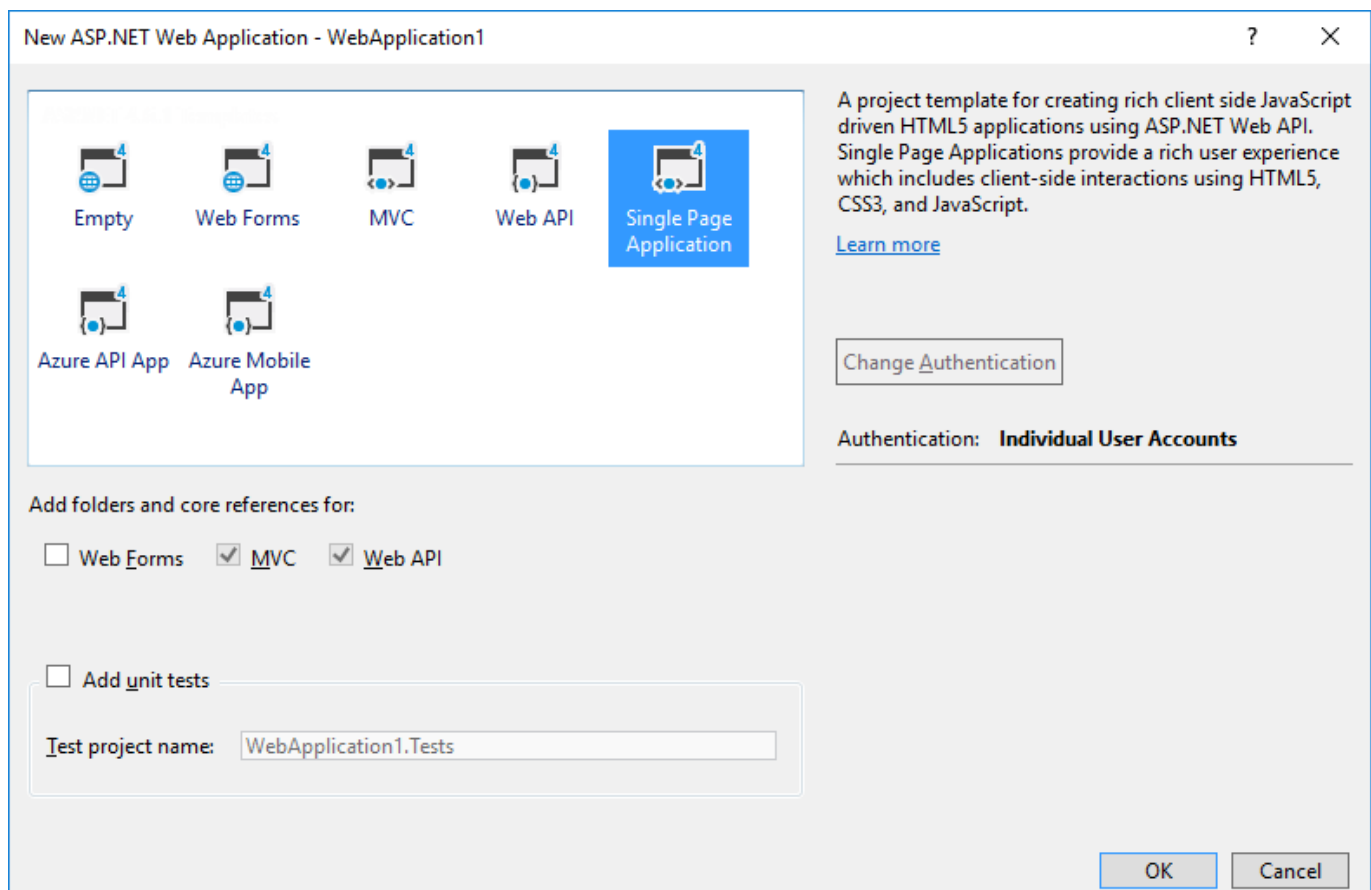
6.3 Option 3: Create an HTML5 Web Application with **animation** to implement the number guessing game. You must extend the Basic GUI Design in Figure 3 to include the animation in either input and output. Note, animation of making the images move or change dynamically is required.

In this option, you will not need to call the service in the application. You will need to read additional reference materials on HTML5 animation libraries. You can use any animation library supported by HTML5.

Assignment 2 submission requirement:

Questions 5 and 6 should be in one solution. The solution and project must contain all the files. Put your solution and project into a folder. Zip the folder (a single file) for submission. Testing results (with screenshots) must be submitted in a Word document or in a PDF document. You must indicate in the document the Visual Studio version and the template that you use, so that the TA can use the same tools to test your program.

General Submission Requirement

Submit the solution folder containing all the files so that the solution can be tested on the TA's machine. If multiple solutions must be submitted, all the solutions must put into another "root" folder. Zip the root folder to submit a single zip file.

Submission notice: The assignment consists of multiple distributed projects. They may be stored in different locations when you create them. You must copy these projects into a single folder for Canvas submission. To make sure that you have all the files included in the zip file and they work together, download your own submission from the Canvas. Unzip the file on a different machine and test it and see if you can run the solution in a different location because the TA will test your application on a different machine.

Grading and Rubrics

Each sub-question (programming tasks) has been assigned certain points. We will grade your programs following these steps:

(1) Compile the code. If it does not compile, 50% of the points given for the code under compilation will be deducted. Then, we will read the code and give points between 50% and 0, as shown in right part of the rubric table.

(2) If the code passes the compilation, we will execute and test the code using test cases. We will assign points based on the left part of the rubric table.

In both cases (passing compilation and failed compilation), we will read your program and give points based on the points allocated to each sub-question, the readability of your code (organization of the code and comments), logic, inclusion of the required functions, and correctness of the implementations of each function.

Please notice that we will not debug your program to figure out how big or how small the error is. You may lose 50% of your points for a small error such missing a comma or a space!

We will apply the following rubrics to **each sub-question** listed in the assignment. Assume that points assigned to a sub-question is *pts*:

Major	Code passed compilation				Code failed compilation		
Points	pts * 100%	pts * 90%	pts * 80%	pts * 70% - 60%	pts * 50% - 40%	pts * 30% - 10%	0
Each sub-question	Meeting all requirements, well commented, and working correctly in all test cases	Working correctly in all test cases. Comments not provided to explain what each part of code does.	Working with minor problem, such as not writing comments, code not working in certain uncommon boundary conditions.	Working in most test cases, but with major problem, such as the code fail a common test case	Failed compilation or not working correctly, but showing serious effort in addressing the problem.	Failed compilation, showing some effort, but the code does not implement the required work.	No attempt

Late submission deduction policy:

- Grace period (Sunday): No penalty for late submissions that are received within 24 hours of the given due date;
- 1% grade deduction for every hour after the first 24 hours of the grace period!
- No submission will be allowed after Tuesday midnight. The submission link will be disabled.