**[Week 1 Lab](https://online.oit.edu/webapps/assignment/uploadAssignment?content_id=_1598834_1&course_id=_18384_1&group_id=&mode=view)**

**Put the answers to the questions below in a text file that you add to your GitHub repository.**

**Exercise 1**

1. Using Visual Studio, create the sample ASP.NET MVC Web application that uses C#.  Leave "No Authentication" selected and "Add unit tests" unchecked.
2. Start the application without debugging from Visual Studio.
3. Click on the "About" link in the navigation bar.  What code in the application just got executed?  Hint: Look at the address bar in the browser.

[*http://localhost:51252/Home/About*](http://localhost:51252/Home/About)*. The code in the application is ASP.NET MVC.*

1. Click on the "Application name" in the navigation bar to return to the home page.
2. In the address bar of the browser, add /home/about to the end of the URL (after the port number) and hit enter.  Explain the process the MVC framework takes to result in the page displayed.  How does the MVC framework know what to display based upon the URL it receives?

*When making a web request Home/About, the model (being Home in this case) has a data (being called About). Which also acts as a method. Once we make the request to go to the About page, we have the Controller send request to the Model to display its data in View.*

**Exercise 2**

1. What part of the Home controller generates the "About" page?

*The method part.*

1. What data is being generated by the controller to be displayed in the page?

*The Home data called About.*

1. Where is the view that the controller invokes to display the page?

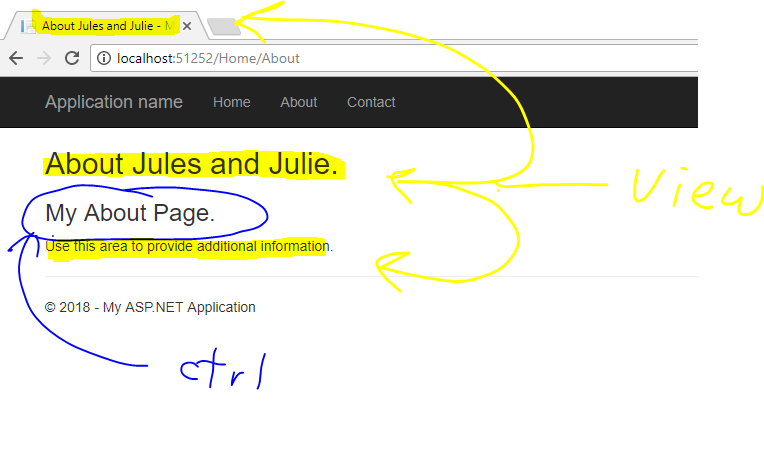
*The Internet browser.*

1. Describe the folder hierarchy convention that allows the MVC framework to find the correct view.

*Within the View folder, there lies the Model Home and it contains About, Contact, and Index.*

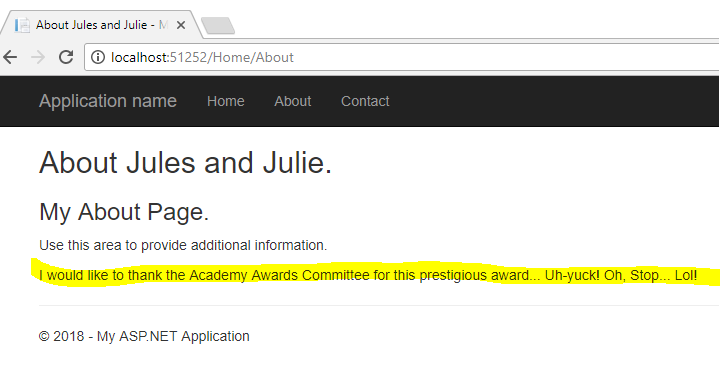
1. Find the place in the view where the data generated by the controller is displayed.  This is a simple example of the Razor syntax for adding C# logic to a view. (*see combined answer with question 6*)
2. Change the data generated by the controller, rebuild and restart the application.  Redisplay the "About" page to see the new data displayed. (*see combined answer with question 5*)

Answer 5 & 6:



*The data generated by the controller is under the C# Razor syntax, in the Home folder of the View folder in the program, which I modified to get a better picture on exactly which data is being displayed in the view, and which data is not necessarily in the View folder. As you can see, the phrase “My About Page” is only data not in the View folder but spotted in the About method of the Controller class code.*

1. Add another piece of data to be displayed in the "About" page.  Redisplay the "About" page to see the new data.



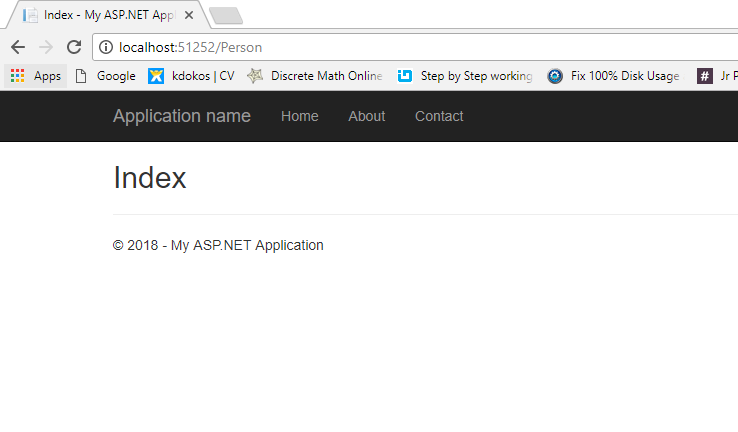
**Exercise 3**

1. Create a new empty controller for the "Person" entity in your application with an "Index" method for displaying information for a person.  
   Right click on the "Controllers" folder, select "Add" and then "Controller".
2. Start the application in Visual Studio.
3. Go to the address bar and enter Person after the port number in the URL.  Hit enter.  What do you see?  Why?

*A server error is displayed. The reason being is that the controller for the Person was created but neither the View nor the Model were created.*

1. Create a view for displaying a person.  Right click on the "Person" controller under "Views",  
   select "Add" and then "View".  Keep all of the default settings in the displayed dialog.
2. Repeat steps 2 and 3.  What do you see now?

An actual blank web page with the word *Index* in it.



1. Add "person" data to the ViewBag in your new controller.
2. Add the display of the new "person" data to the view that you created.
3. Repeat steps 2 and 3.  What do you see now?

**Exercise 4**

1. Add a "Person" model to your project.  Right click on "Models", select "Add" and then "Class".
2. Add data to the "Person" model that you think is relevant to your application.

**Complete**

1. Put the code you created for this lab up on a GitHub repository.  Don't forget to include a text file in the repo that contains the answers to the question above.
2. Send me an email indicating that you finished the lab and include the URL to your GitHub repository for this lab.