Lab: Building Your First ASP.NET MVC Core Project

Exercise 0: Setting up

1. Prepare Your Workspace
    - Create a folder for all coursework
    - e.g. C:\AMIS3610 (Windows) or ~/AMIS3610 (Mac / Linux)
2. Verify Node is installed
    - at a terminal window, run

```
node -v
```

    - if Node is not installed, install it: https://nodejs.org/en/download/
3. Install Yeoman

```
npm install –g yo
```

    - the '-g' flag tells the node package manager to install Yeoman in the global package system.
4. Install Bower

```
npm install –g bower
```

5. Install Bower

```
npm install –g generator-aspnet
```

6. Verify everything works

```
yo aspnet --help
```

Exercise 1: Your first ASP.NET MVC project

1. Create a new .NET Core project in your AMIS workspace:

```
mkdir helloweb

cd helloweb

dotnet new -t web
```

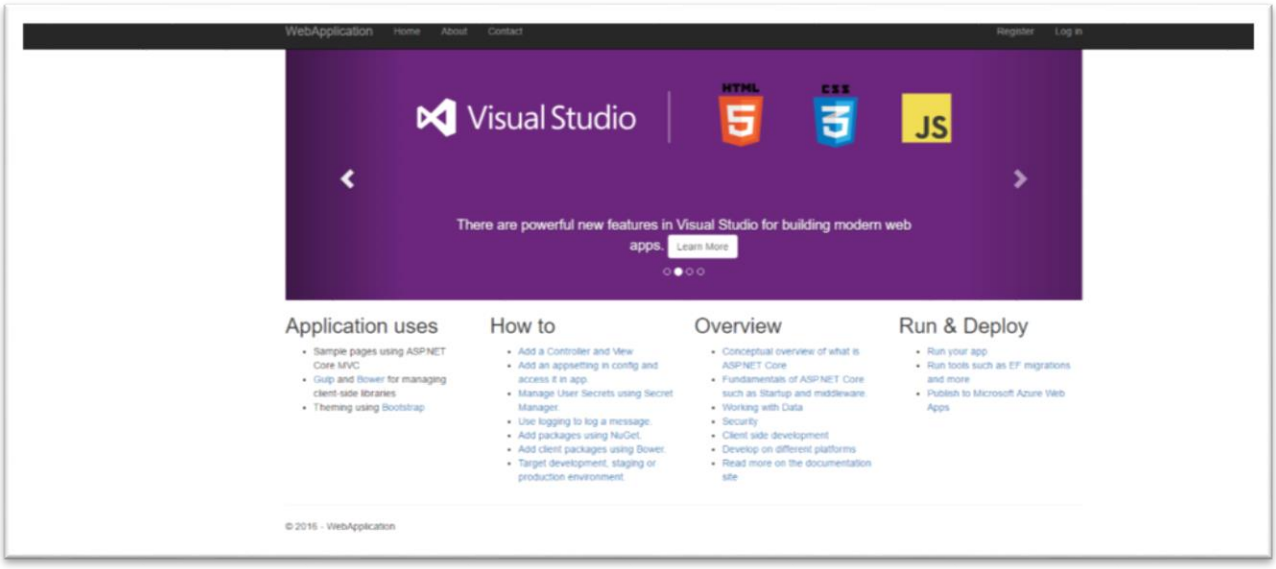
2. Restore the packages:

```
dotnet restore
```

3. Run the app (the `dotnet run` command will build the app when it's out of date):

```
dotnet run
```

4. Browse to `http://localhost:5000`



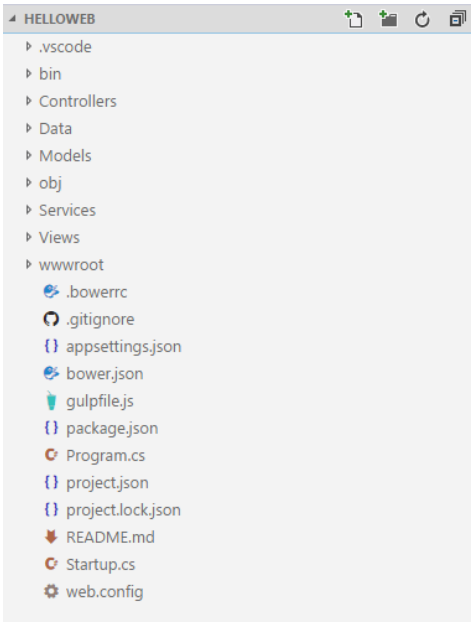
Exercise 2: Examine your project

1. Open your project in an editor

```
code .
```

   - If Visual Studio Code is not setup properly, or if you are using Visual Studio Community, you can open the project the old-fashioned way: File -> Open [helloweb directory]

2. Examine the top level artifacts in your project folder:



3. Examine project.json file

   - Note the dependencies section
   - Note the tools section
   - Note the frameworks section
   - Note the buildOptions section

4. Examine the program.cs file
5. Examine the startup.c file
6. Examine the Controllers folder

   - Note the AccountController

- Note the HomeController
- Note the ManageController

7. Examine the Models folder

    - Note the AccountViewModel folder and contents
    - Note the ApplicationUser.cs file

8. Examine the Views folder

    - Note the Account folder and contents

Exercise 3: Your first ASP.NET Web API project

1. Make sure you are in the root of your AMIS workspace (e.g. C:\AMIS or ~/amis) and run Yeoman

```
yo
```

2. Use the arrow keys to select ASPNET

```
> ASPNET
```

3. Use the arrow keys to select Web API Application

```
> Web API Application
```

4. Name the application “helloapi”
5. Change to the helloapi directory

```
cd helloapi
```

6. Open your project in an editor

```
code .
```

9. Examine project structure file
10. Open the integrated terminal

    - either hit CTRL + ` for Windows or Command + ` for Mac in Code
    - or File -> Integrated Terminal
    - Open your project in an editor

11. Restore your packages and build the solution then run it

```
dotnet restore
```

```
dotnet build

dotnet run
```

12. Use Postman to send an HTTP Request to helloapi

    - GET http://localhost:5000/api/values
    - What happens when you change the URI?
    - What happens when you POST?
    - What happens when you ask for the resource with id of 5?
        - http://localhost:5000/api/values/5

Exercise 4: Initialize your Git repository

1. In Code, select the Git icon
2. Press the “Initialize” button
3. Enter a check-in message and hit the Commit All check mark icon
4. Change the controller method in ValuesController to return “Value 5”

```
// GET api/values/5

 [HttpGet("{id}")]

  public string Get(int id)

  {

     return "value 5";

  }
```

5. What happened in the Git pane?
6. Enter a relevant change message and hit the Commit All check mark icon
7. Examine the history of your repo

```
git log
```

8. Create a repo on GitHub called ‘helloapi’
9. Set the remote origin

```
git remote add origin https://github.com/{yourgithub}/helloapi.git
```

10. Verify the remote origin

```
git push remote
```

11. Push your code to GitHub

You may need to consult some resources to get this to work (https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/)