AngularJS – Part 2

Agenda

- Using AngularJS with ASP.NET MVC (Web API)

Tools

- Visual Studio 2013

Pre-requisite

- AngularJS – Part 1
- Understanding of ASP.NET MVC
- Understanding of ASP.NET Web API

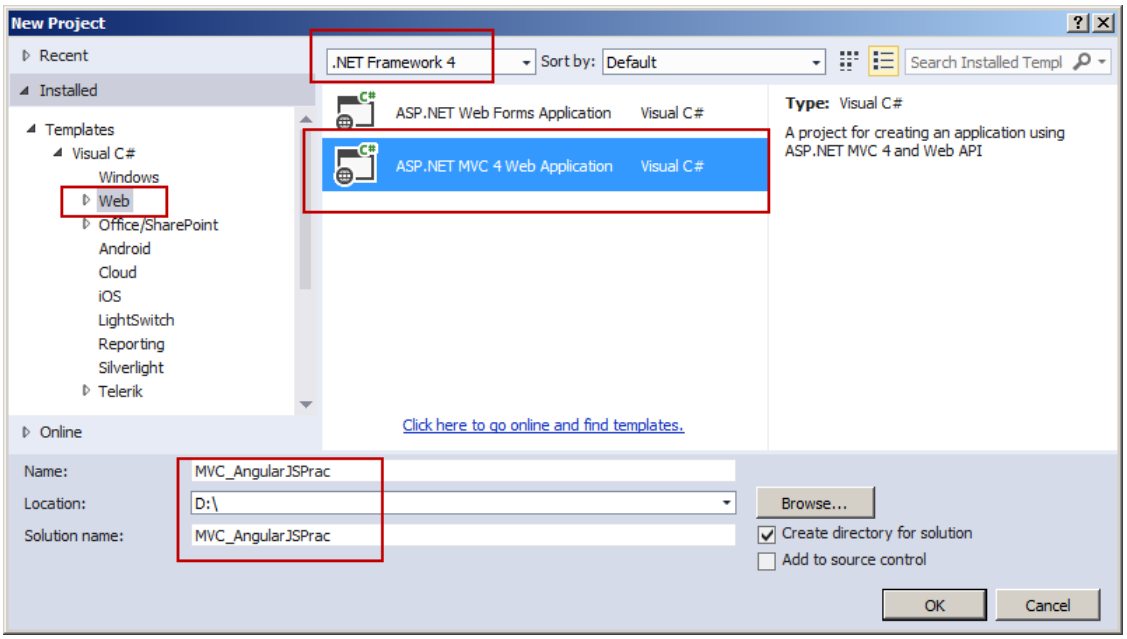
Version	Last updated	Comments	Modified By
V1.0	25-05-2016		Bilal Shahzad

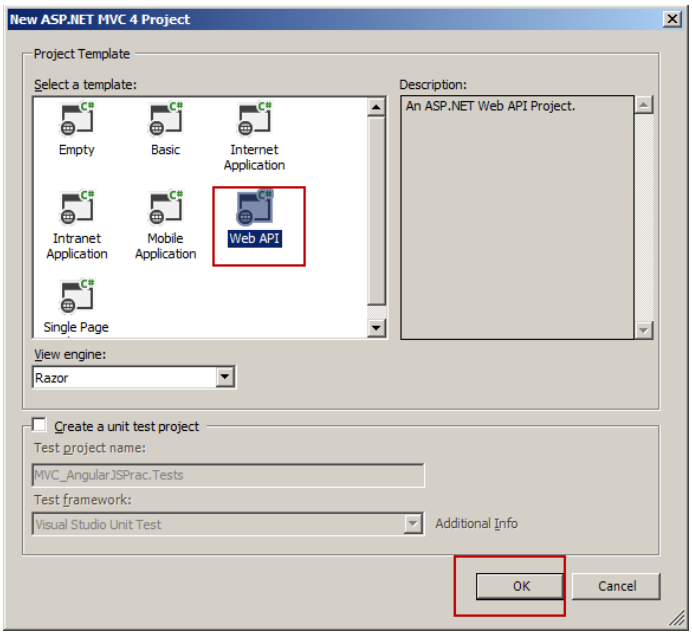
Brief Introduction

1- ASP.NET MVC (Web API) application creation.
2- Creating a view which will make AJAX call when loaded and get posts from server and display.
3- Login screen which will pass login data to Web API controller through AJAX

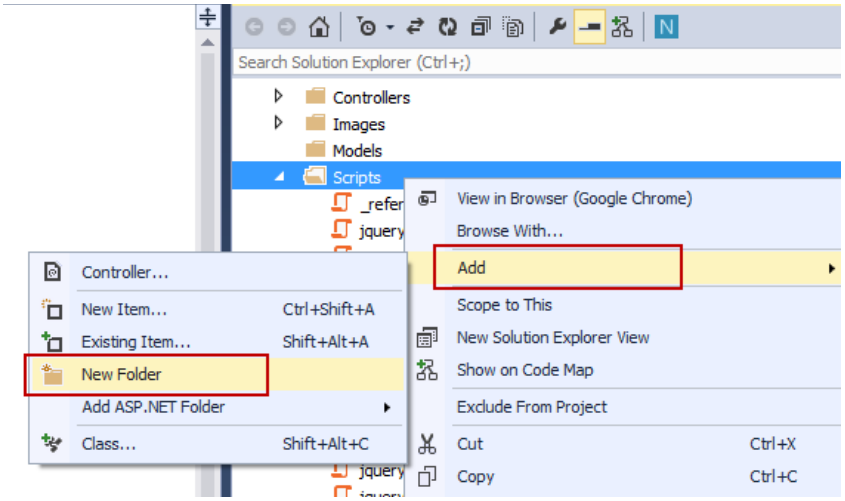
Step by Step Walkthrough

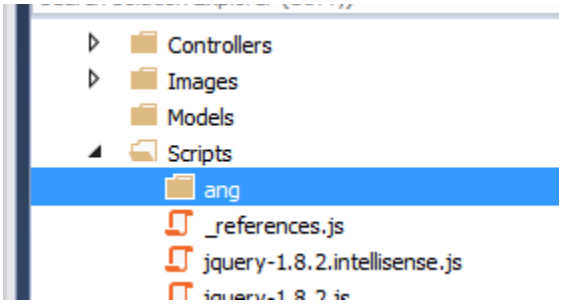
1- Create a new "Web" project in VS 2013.
   a. Go to File => New => Project
   b. Choose options as shown below and give some name to the project (e.g. MVC_AngularJSPrac). Click OK.



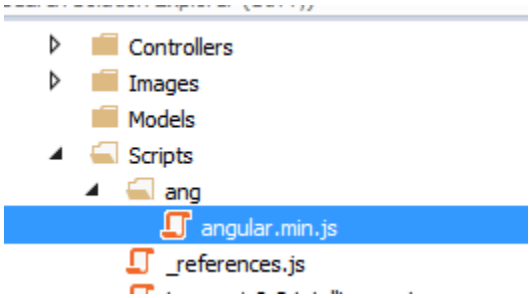


2- Right click on “scripts” => Add => “New Folder”. Name it something (e.g. ang)

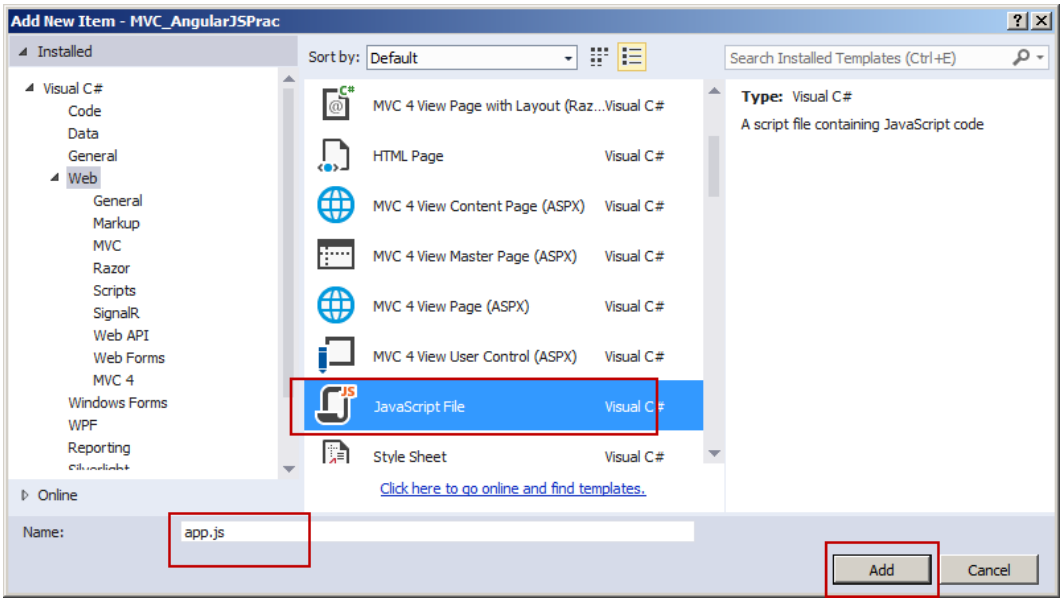




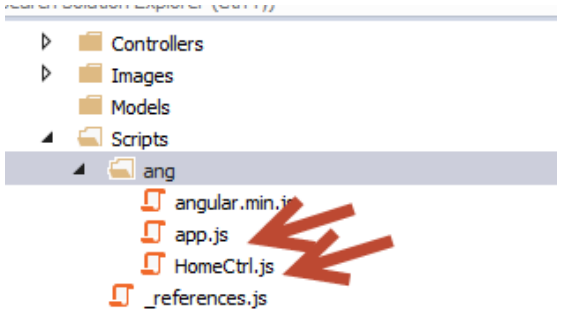
3- Copy paste “angular.min.js” file in “ang” folder.



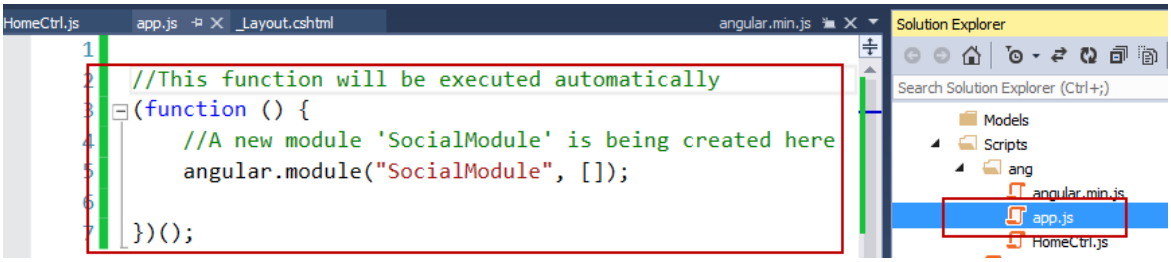
4- Right click on “ang” folder => Add => New Item.



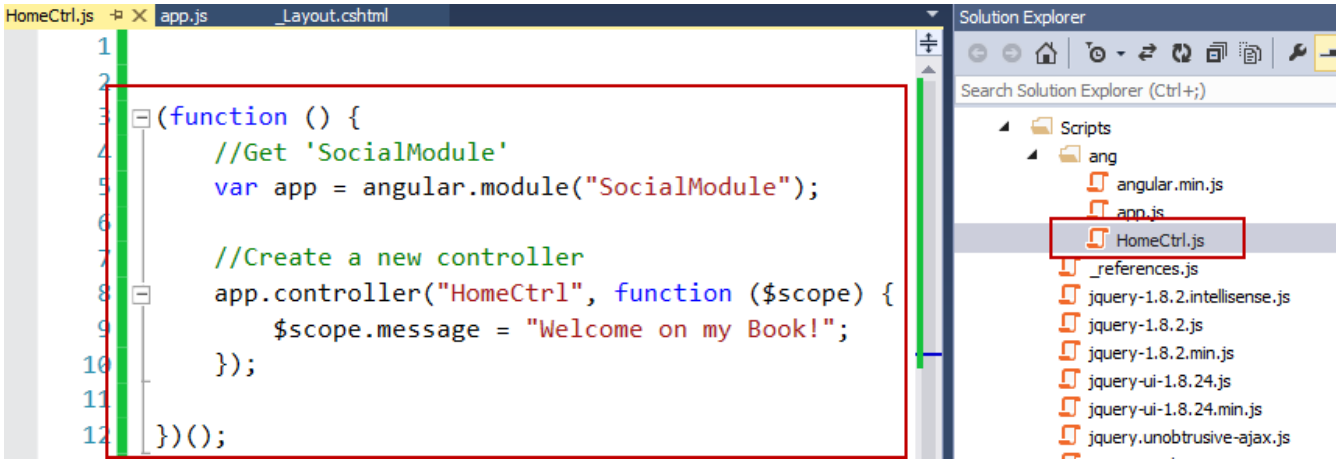
5- Add a new JS file (HomeCtrl.js) in “ang” folder.



6- Now open “app.js” file by double clicking on it and add following code in it. Here we are creating a new module. Please note that we are passing“[]” as second parameter. In second parameter we pass the modules details on which our module depends on.

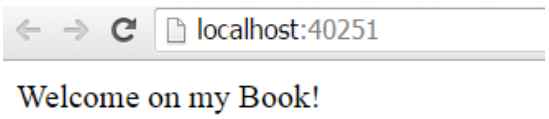


7- Now open “HomeCtrl.js” file by double clicking on it and add following code in it. Here we are accessing already created module object and then adding a controller inside it. In controller, we are adding a simple property “message” to our “$scope” object.



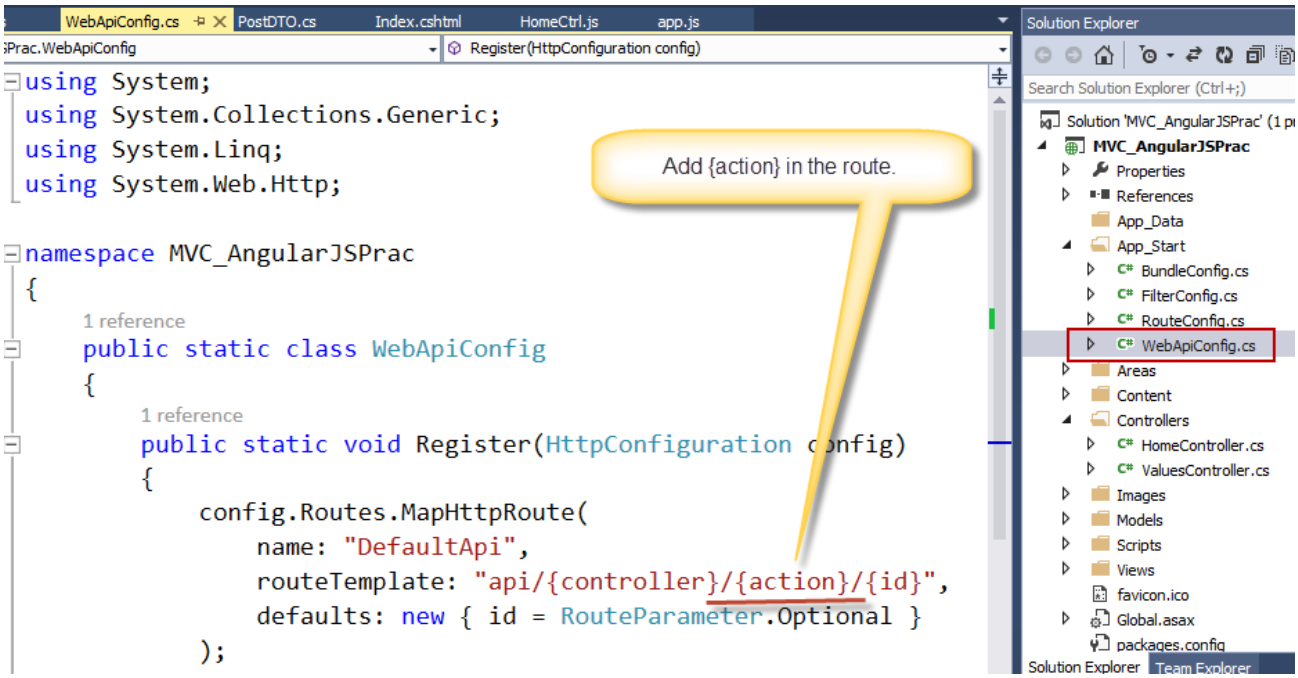
8- Now open (Views/Home/Index.cshtml) file and remove existing content. Add content as highlighted below. We are currently doing to do it without (Layout) page to make things simpler.

9- Run the application and check if you see following result or not.



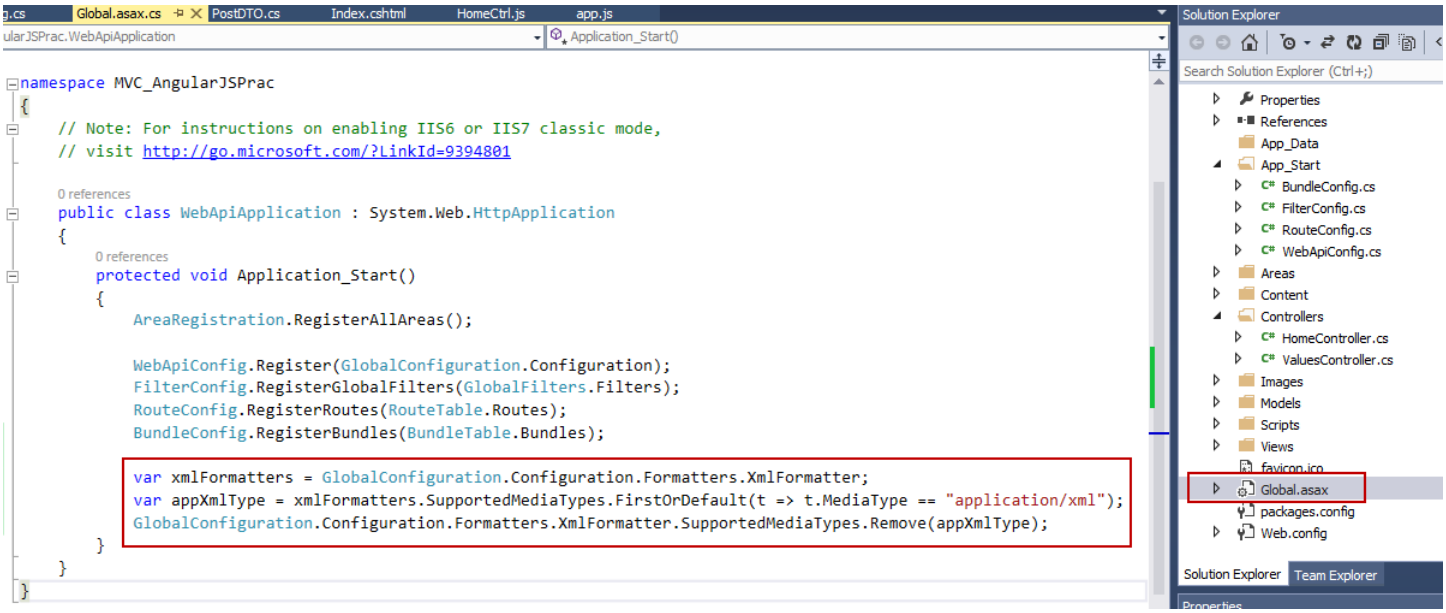
10- Let's make some changes for WEB API.

11- Now open "App_Start/WebApiConfig.cs" file and make one change as highlighted below.

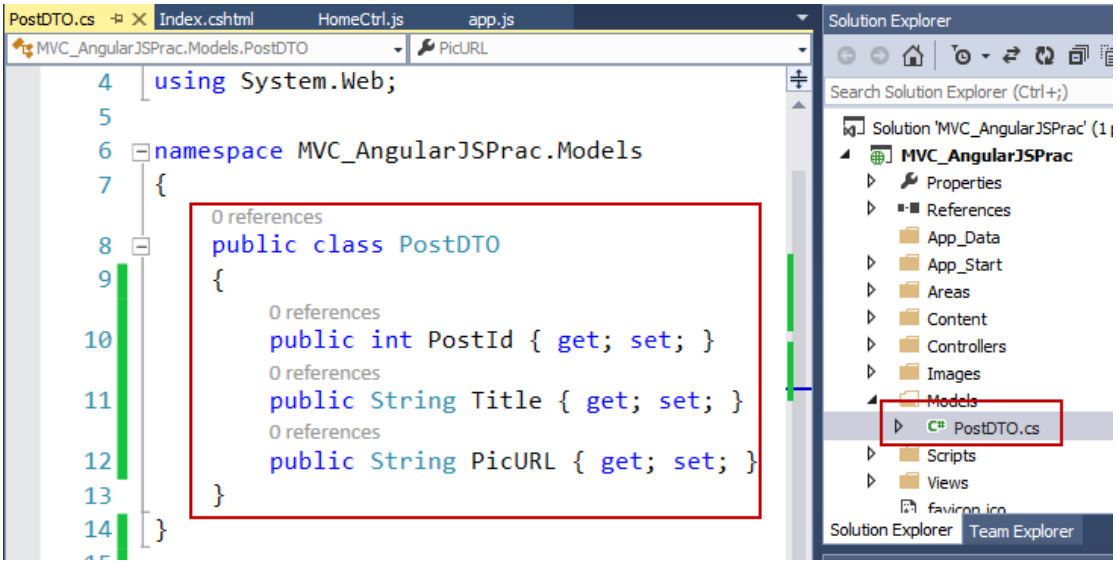


12- Open "Global.asax" file and add following three lines (to remove XML formatter and get data in JSON by default) as shown in the screenshot below.

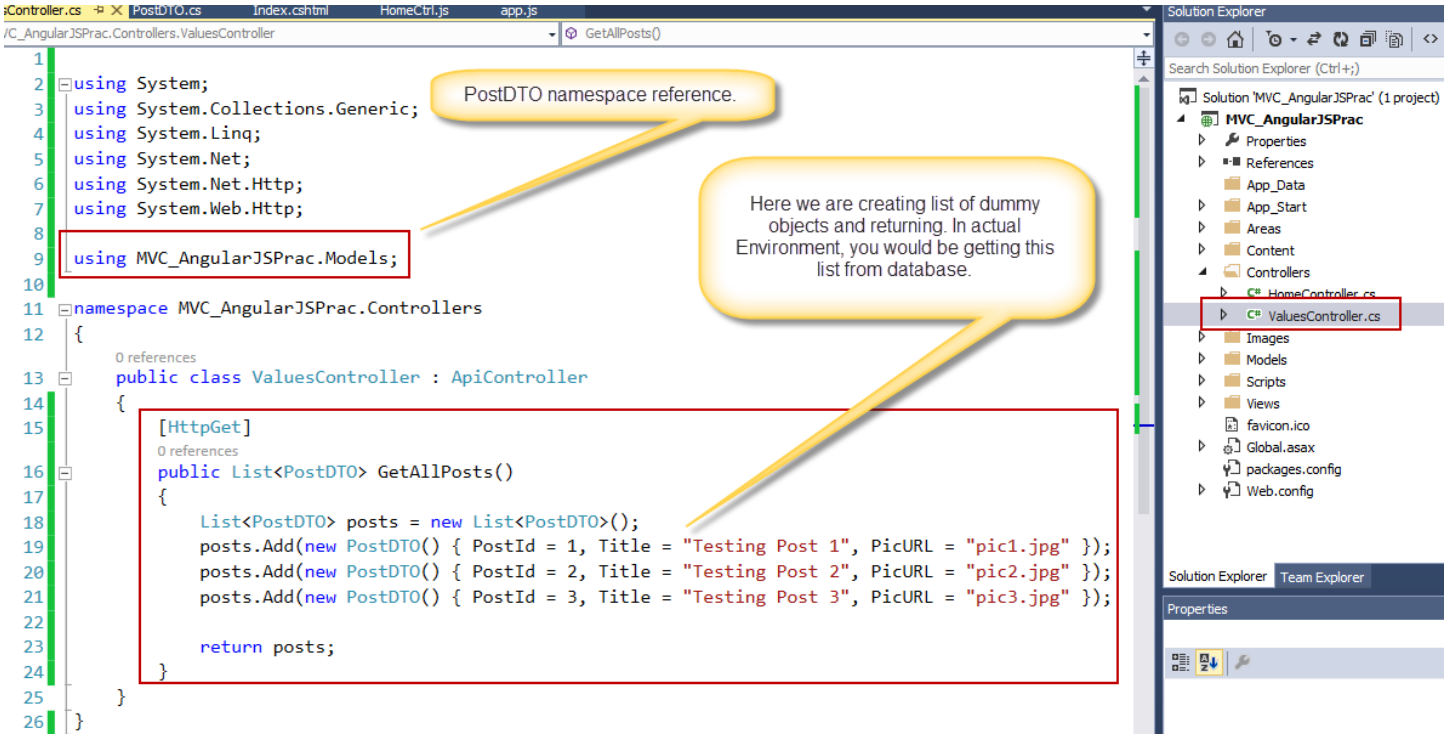
```
var xmlFormatters = GlobalConfiguration.Configuration.Formatters.XmlFormatter;
var appXmlType = xmlFormatters.SupportedMediaTypes.FirstOrDefault(t => t.MediaType == "application/xml");
GlobalConfiguration.Configuration.Formatters.XmlFormatter.SupportedMediaTypes.Remove(appXmlType);
```



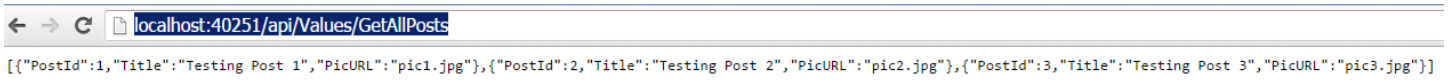
13- Now create a new class file in “Models” folder. Name it “PostDTO.cs”. Add following content in the class.



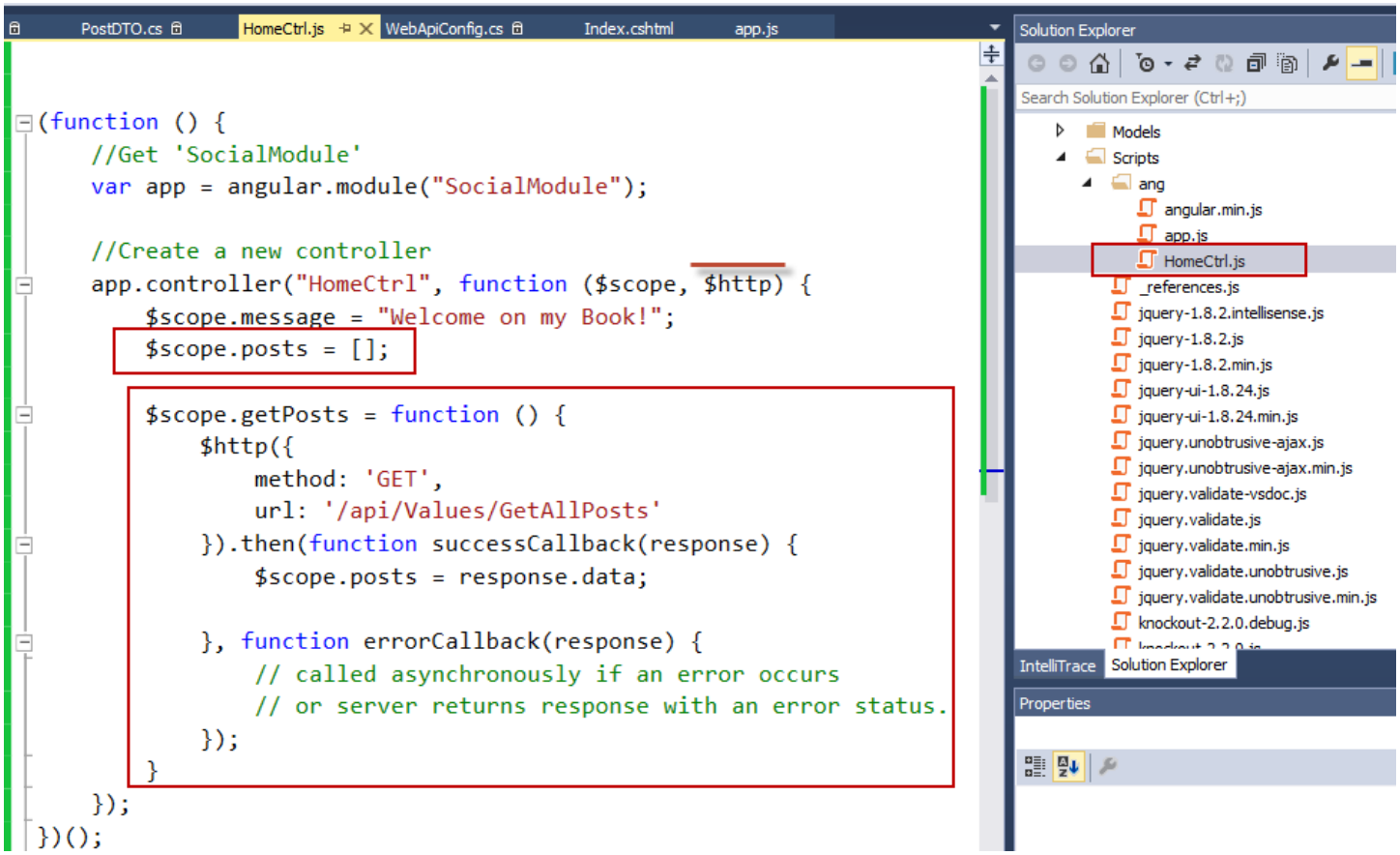
14- Now open “Controllers/ValuesController.cs” file. Remove all functions from class and add a new function as shown in following screenshot. Here we are creating a dummy list of posts and returning it to request.



15- Run the project and try to access "/api/Values/GetAllPosts". Please verify if result is coming as expected (as shown in following screenshot)



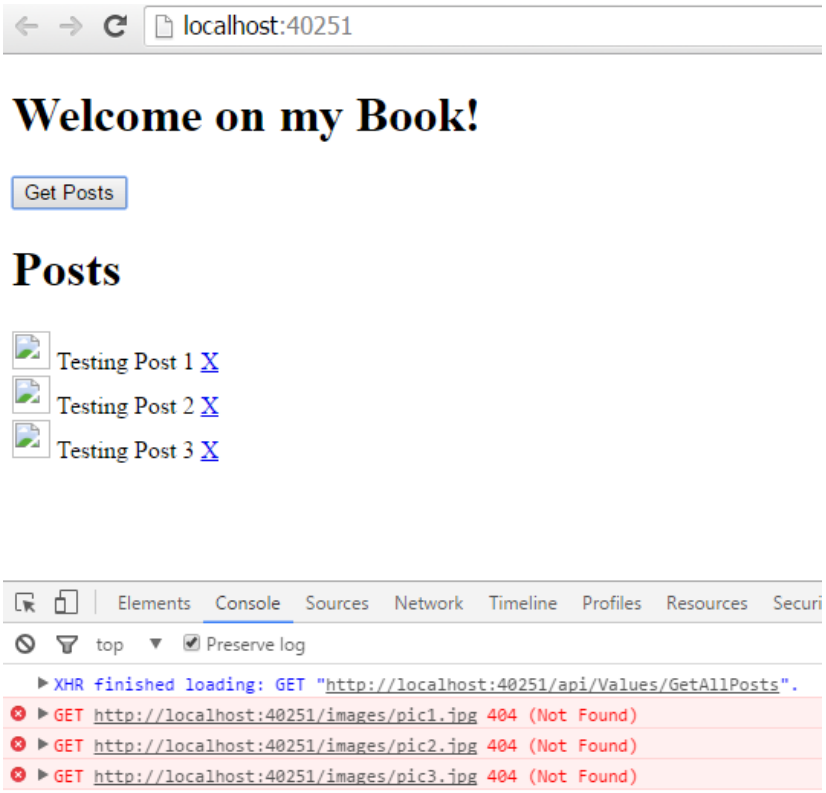
16- Now let's write code in our "HomeCtrl.js" file to make AJAX call to this Web API and get posts. Here we've added one more parameter to our controller function i.e. $http. "$http" is required to make AJAX call. We've also created a "posts" variable in our $scope. Then we've added a new function "getPosts" in $scope. In this function, we are making an HTTP hit to our Web API URL to get posts. Once response is available, it can be accessed using "data" property. Please check following screenshot for these changes.



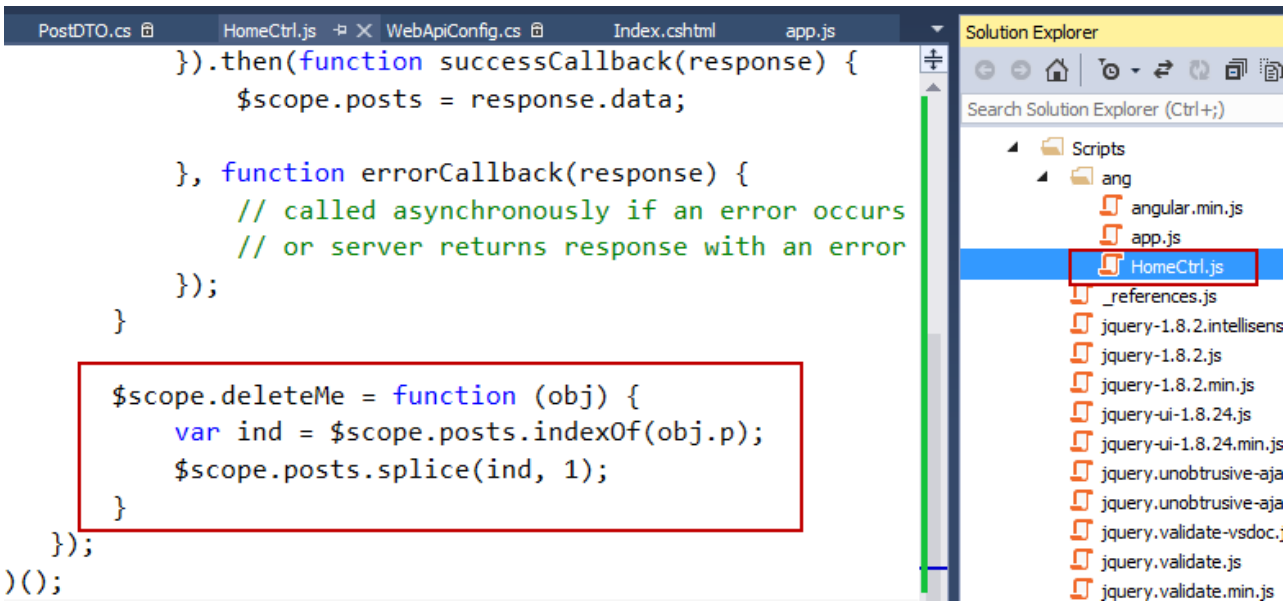
17- Now make following changes in "Views/Home/Index.cshtml" file. Here we are calling "getPosts" method on button click. And then we are displaying the data of "posts" object. We are assuming that images are in "images" folder. We are also showing a hyper link (X) which is calling a function "deleteMe(this). We are passing current attached object (with which this record has been created) to "deleteMe" function. **Note: We hadn't defined "deleteMe" function in our controller.**

```
    <script src='@Url.Content("~/Scripts/ang/angular.min.js")' type="text/javascript"></script>
    <script src='@Url.Content("~/Scripts/ang/app.js")' type="text/javascript"></script>
    <script src='@Url.Content("~/Scripts/ang/HomeCtrl.js")' type="text/javascript"></script>

</head>
<body ng-controller="HomeCtrl">
    <h1>{{message}}</h1>

    <input type="submit" ng-click="getPosts()" value="Get Posts" />
    <div>
        <h1>Posts</h1>
        <div ng-repeat="p in posts">
            <img ng-src='@Url.Content("~/images/{{p.PicURL}}")' height="25px" width="25px" />
            <span>{{p.Title}}</span>
            <a href="#" ng-click="deleteMe(this);">X</a>
        </div>
    </div>
</body>
</html>
```

18- Run the project, click on "Get Posts" button and see if posts are displayed or not. Also inspect the html of page to see if any error is appearing in console. As we don't have these images in our project so errors are appearing.

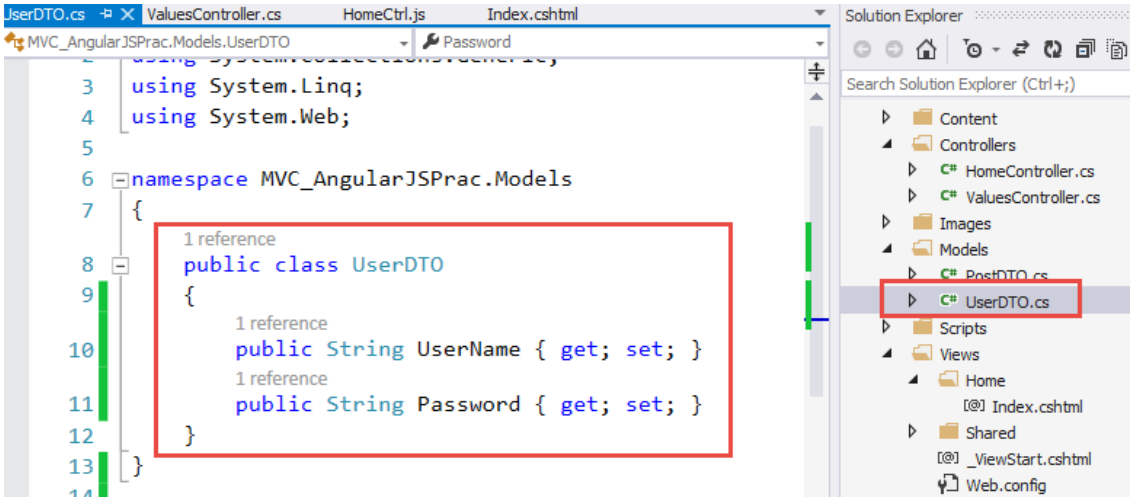


19- Now add following code in “HomeCtrl.js” file to define “deleteMe” function. “obj” will contain the object in “p” property as we’ve used “p” alias during repeat. So here are finding the post in our array and then removing it (using splice function). Note: We are going to delete post from local array (not from server) for now.

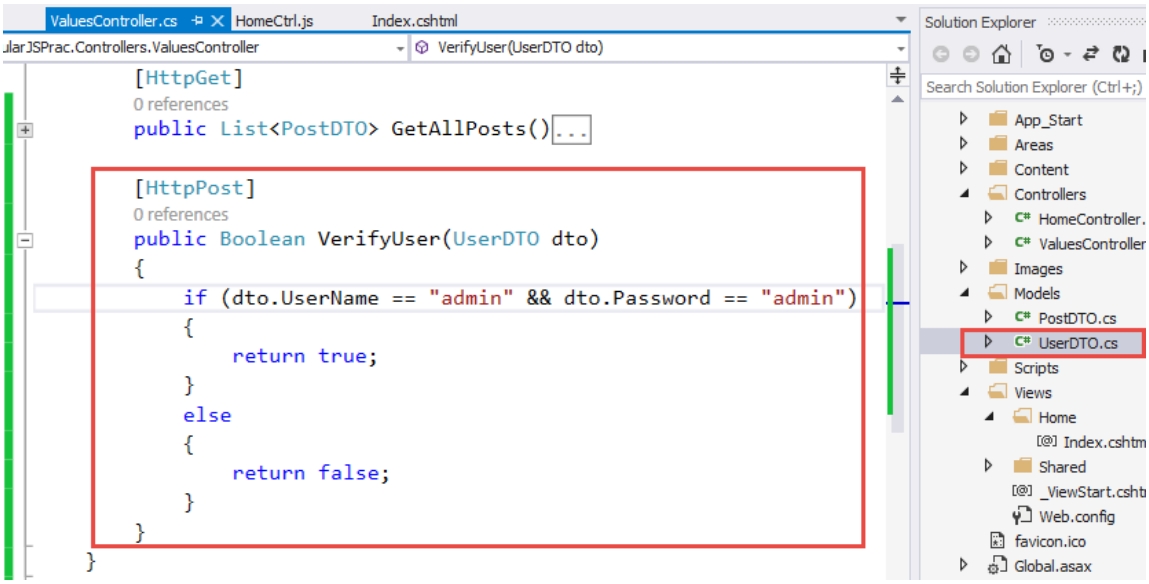


20- Let’s check how to do AJAX Post hit. For this, let’s work on login screen.

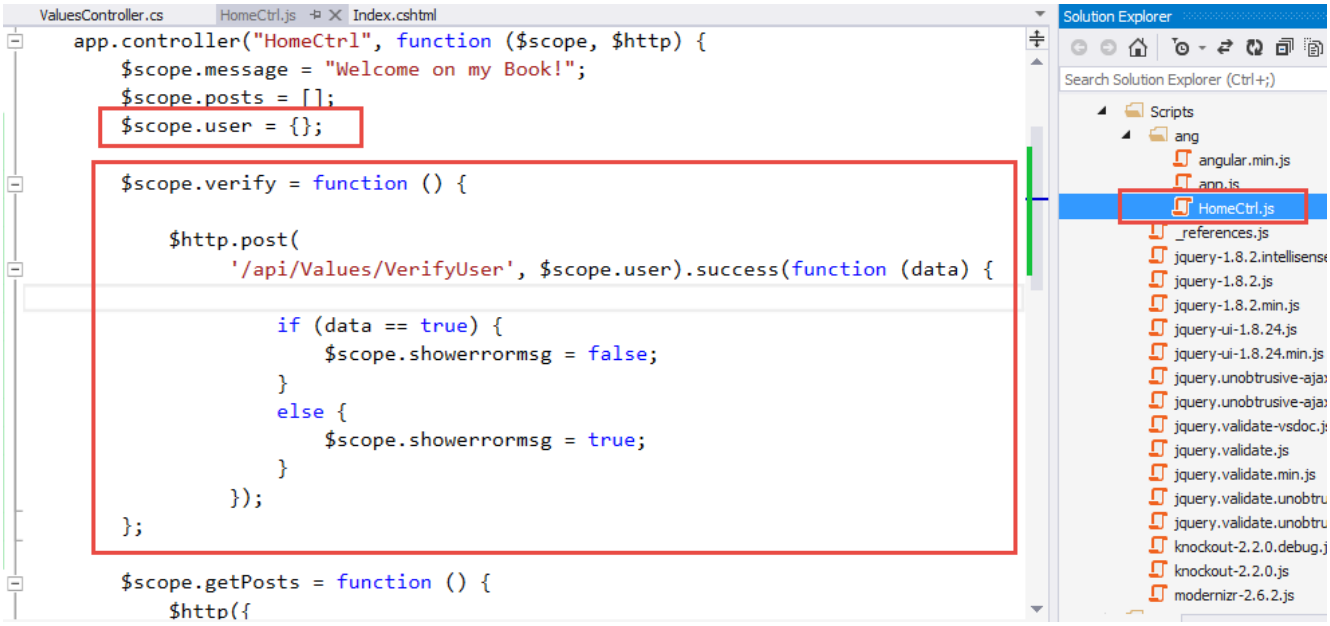
21- Create a new class in Model (i.e. UserDTO) as shown in following screenshot.



22- Add following method in Web API controller. It will receive object of type UserDTO in method.



23- Add following code in “HomeCtrl.js” file which will make hit to Web API for user verification. Hect re we’ve a “user” object which will be bound with form elements.



24- Add following code in “Views/Home/Index.cshtml” file. It is to show login controls. Note that “ng-show” is another directive which hides or show the control based on the bound property.

25- Run the application and check if everything is working fine as expected. First provide correct credentials (admin & admin) and then provide wrong credentials.

Tasks for Practice

Once you are done with above tutorial. Try to work on these tasks.

1- https://material.angularjs.org/latest/
2-

Useful Links

https://angularjs.org/

http://www.w3schools.com/angular/

http://www.tutorialspoint.com/angularjs/