ASP.NET Web APIs

Notes by rajeeshvengalapurath@gmail.com
20 Jul 2019

Introduction

ASP.NET Web API is a framework for building Web APIs, ie. HTTP based services on top of the .NET Framework.

RESTful services

REST(Representational State Transfer). First introduced in 2000 by Roy Fielding. REST is an architectural pattern for creating services. REST architectural pattern specifies a set of constraints that a system should adhere to

REST constraints

(constraint = a limitation or restriction.)

- Client Server
- Stateless: communication between client and server must be stateless between requests. Means we should not be storing anything related to the client on server. Each request is treated independently by the server.
- Cacheable
- Uniform Interface

/Employee	GET	Gets list of employees
/Employee/1	GET	Gets employee with id=1
/Employee	POST	Creates a new employee
/Employee/1	PUT	Updates employee with Id=1
/Employee/1	DELETe	Deletes employee with Id=1

- HATEOS (Hypermedia as the Engine of Application State)
- Layered System
- Code on Demand (optional)

When to use WCF over ASP.NET Web API

- Creating services that are transport/protocol independent. Single service with multiple end point
- You have an existing SOAP service you must support but want to add REST to reach more clients
- .NET 3.5 limitation

Controllers

`Controller` class is present in `System.Web.Mvc`
`ApiController` is present in `System.Web.Http`

HTTP - Terms & Concepts

Request Verbs (GET, POST, PUT & DELETE)

These verbs desribe what should be done with the resource. For the complete list of the HTTP verbs http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

Request Haader

Contains additional information about the request. Example - what type of response is required (XML, JSON)

Request Body

Contains the data to send to the server. The format of data may be XML or JASON

Response Body

Contains the data sent as a response from the server. Example - product details in XML or JSON format

Response Status Codes

Provide the client, the status of the request. Example 200 for OK, 404 for not found, 204 no content. For the complete list of HTTP status codes
http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

Content Negotiation

Using "Accept" header the client can specify the format of the response they want

- Accept:application/xml
- Accept:application/json
- Accept:application/xml,application/json (multiple values)

You can also specify "quality" factor

- application/xml;q=0.8,application/json;q=0.5 (priority to the high quality)

If “accept” header is not specified, by default Web API returns JSON data

MediaTypeFormatter

MediaTypeFormatter is an abstract class from which JsonMediaTypeFormatter and XmlMediaTypeFormatter classes inherit from. Used to specify response type as JSON or XML.

To return only JSON from ASP.NET Web API Service

config.Formatters.Remove(config.Formatters.XmlFormatters);

To return only XML from ASP.NET Web API Service

config.Formatters.Remove(config.Formatters.JsonFormatters);

FromBody

```
public void Post([FromBody] Product product) {}
```

Meaning that product object will be in request body.

Get Sample

```
public HttpResponseMessage Get(int id)
{
	using (SampleDBContext db = new SampleDBContext())
	{
		var entity = db.Products.Where(p=>p.Id==id).FirstOrDefault();
		if (entity != null)
			return Request.CreateResponse(HttpStatusCode.OK, entity);
		else
			return Request.CreateErrorResponse(HttpStatusCode.NotFound,
				"Product with is " + id + " not found" );
	}
}
```

Post Sample

```
public HttpResponseMessage Post([FromBody] Product product)
{
	try
	{
		using(SampleDBContext db = new SampleDBContext())
		{
			db.Products.Add(product);
			db.SaveChanges();

			var message = Request.CreateResponse(HttpStatusCode.Created,
				product);
			message.Headers.Location = new Uri(Request.RequestUri + "/" +
				product.Id.ToString());
			return message;
		}
	}
	catch (Exception ex)
	{
		return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
	}
}
```

Delete Sample

```
public HttpResponseMessage Delete(int id)
```

```
{
	try
	{
		using (SampleDBContext db = new SampleDBContext())
		{
			var entity = db.Products.FirstOrDefault(p => p.Id == id);
			if (entity == null)
			{
				return Request.CreateErrorResponse(
					HttpStatusCode.NotFound
					, "Product with id " + id + " not found");
			}
			else
			{
				db.Products.Remove(entity);
				db.SaveChanges();
				return Request.CreateResponse(HttpStatusCode.OK);
			}
		}
	}
	catch (Exception ex)
	{
		return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
	}
}
```

Put Sample

```
public HttpResponseMessage Put(int id, [FromBody]Product product)
{
	try
	{
		using (SampleDBContext db = new SampleDBContext())
		{
			var entity = db.Products.FirstOrDefault(p => p.Id == id);
			if (entity == null)
			{
				return Request.CreateErrorResponse(
					HttpStatusCode.NotFound,
					"Product with id not " + id + " found");
			}
			else
			{
				entity.Name = product.Name;
				entity.Rate = product.Rate;
				entity.CategoryID = product.CategoryID;
				db.SaveChanges();
				return Request.CreateResponse(HttpStatusCode.OK);
			}
		}
	}
	catch (Exception ex)
	{
		return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);
```

```
	}
}
```

Get by name sample

https://localhost:44310/api/Product/?name=p
https://localhost:44310/api/Product?name=p

```
public HttpResponseMessage Get(string name)
{
	using (SampleDBContext db = new SampleDBContext())
	{
		var entity = db.Products.Where(p =>
					p.Name.Contains(name)).FirstOrDefault();
		if (entity != null)
			return Request.CreateResponse(HttpStatusCode.OK, entity);
		else
			return Request.CreateErrorResponse(HttpStatusCode.NotFound,
				"Product with name " + name + " not found");
	}
}
```

Method Mapping

```
[HttpGet][HttpPost][HttpPut][HttpDelete]
```

Attributes

Web API default convention for binding parameters

- If the parameter is a simple type like int, bool, bouble etec., Web API tries to get the value from the URI (either from route data or query string)
- If the parameter is a complex type like Customer, Employee etc., Web API tries to get the value from the request body
- Use `[FromBody]` attribute to force Web API to get simple types from the request body
- Use `[FromUri]` attribute to force Web API to get complex type data from the URI (i.e Route data or query string)

Microsoft.AspNet.WebApi.Cors

This package contains the components to enable Cross-Origin Resource Sharing (CORS) in ASP.NET Web API (Must be enabled if api is accessed from a different domain)

WebApiConfig.cs

```
public static class WebApiConfig
{
	public static void Register(HttpConfiguration config)
	{
		config.MapHttpAttributeRoutes();
```

```
            config.Routes.MapHttpRoute(
               name: "DefaultApi",
               routeTemplate: "api/{controller}/{id}",
               defaults: new { id = RouteParameter.Optional }
            );

            EnableCorsAttribute cors = new EnableCorsAttribute("*", "*", "*");
            config.EnableCors(cors);
        }
}
```

ProductController.cs

```
using System.Web.Http.Cors;

[EnableCorsAttribute("*","*","*")] //Enabled cors for all methods of the class
public class ProductController : ApiController
{
        [DisableCors] //disable cors for this particular method
        public IEnumerable<Product> Get()
        {
                ...
        }
        public HttpResponseMessage Get(int id)
        {
                ...
        }
}
```

SSL

To enable SSL

1. Click on API project
2. Press F4
3. SSL Enabled = True

Use/Redirect to HTTPS when an HTTP request comes in

```
// Manually created file. After creating this class and content, register it in
WebApiConfig.cs

using System.Web.Http.Filters;
using System.Net.Http

public class RequireHttpsAttribute : AuthorizationFilterAttribute
{
        public override void OnAuthorization(HttpActionContext actionContext)
        {
                if (actionContext.Request.RequestUri.Scheme != Uri.UriSchemeHttps)
                {
```

```
                        actionContext.Response =
                        actionContext.Request.CreateResponse(
                                                System.Net.HttpStatusCode.Found);
                        actionContext.Request.Content = new StringContent(
                                                "<p>Use HTTPS instead of HTTP</p>");

                        UriBuilder uriBuilder = new UriBuilder(
                                                actionContext.Request.RequestUri);
                        uriBuilder.Scheme = Uri.UriSchemeHttps;
                        uriBuilder.Port = 44311;

                        actionContext.Response.Headers.Location = uriBuilder.Uri;
                }
                else
                {
                        base.OnAuthorization(actionContext);
                }
        }
}

public static class WebApiConfig
{
        public static void Register(HttpConfiguration config)
        {
                // Web API configuration and services

                // Web API routes
                config.MapHttpAttributeRoutes();

                config.Routes.MapHttpRoute(
                        name: "DefaultApi",
                        routeTemplate: "api/{controller}/{id}",
                        defaults: new { id = RouteParameter.Optional }
                );

                EnableCorsAttribute cors = new EnableCorsAttribute("*", "*", "*");
                config.EnableCors(cors);

                config.Filters.Add(new RequireHttpsAttribute());
        }
}
```

Basic Authentication

Create BasicAuthenticationAttribute.cs with the following content. authenticationToken will be username:password in base 64

```
using System.Web.Http.Filters; //AuthorizationFilterAttribute
using System.Net.Http; //CreateResponse
using System.Net; //HttpStatusCode
using System.Text; //Encoding
using System.Threading;
using System.Security.Principal; //GenericPrincipal, GenericIdentity
```

```
public override void OnAuthorization(HttpActionContext actionContext)
{
	if(actionContext.Request.Headers.Authorization == null)
	{
		actionContext.Response = actionContext.Request.CreateResponse(
							HttpStatusCode.Unauthorized);
	}
	else
	{
		string authenticationToken = actionContext.Request.Headers.
							Authorization.Parameter;
		string decodedAuthenticationToken = Encoding.UTF8.
					GetString(Convert.FromBase64String(
						authenticationToken));
		string[] usernamePasswordArray = decodedAuthenticationToken.
								Split(':');
		string username = usernamePasswordArray[0];
		string password = usernamePasswordArray[1];

		using (SampleDBContext db = new SampleDBContext())
		{
			if (db.Users.Any(user=>user.Username.ToLower() ==
					username.ToLower() && user.Password == password))
			{
				Thread.CurrentPrincipal = new GenericPrincipal(new
					GenericIdentity(username), null);
			}
			else
			{
				actionContext.Response = actionContext.Request.
					CreateResponse(HttpStatusCode.Unauthorized;
			}
		}
	}
	base.OnAuthorization(actionContext);
}
```

Register that in WebApiConfig.cs

```
public static class WebApiConfig
{
	public static void Register(HttpConfiguration config)
	{
		// Web API configuration and services

		// Web API routes
		config.MapHttpAttributeRoutes();

		config.Routes.MapHttpRoute(
			name: "DefaultApi",
			routeTemplate: "api/{controller}/{id}",
			defaults: new { id = RouteParameter.Optional }
			);
```

```
            EnableCorsAttribute cors = new EnableCorsAttribute("*", "*", "*");
            config.EnableCors(cors);

            config.Filters.Add(new RequireHttpsAttribute());
            config.Filters.Add(new BasicAuthenticationAttribute());
        }
    }
```

Set basic authentication for a method alone

Remove `config.Filters.Add(new BasicAuthenticationAttribute());` from WebApiConfig

```
public class ProductController : ApiController
{
       [BasicAuthentication]
       public IEnumerable<Product> Get()
       {
             ...
       }
       ...
}
```

Sample Webpage to access API using JQuery Ajax

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <script src="Scripts/jquery-3.3.1.js"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            var ulProducts = $('#ulProducts');
            $('#btn').click(function () {
                var username = $('#txtUsername').val();
                var password = $('#txtPassword').val();
                $.ajax({
                    type: 'GET',
                    url: 'api/Product',
                    dataType: 'json',
                    headers: {
                        'Authorization': 'Basic ' + btoa(username + ":" + password)
                                                        //btoa = to base 64
                    },
                    success: function (data) {
                        ulProducts.empty();
                        $.each(data, function (index, val) {
                            var name = val.Name;
                            ulProducts.append('<li>' + name + '</li>')
                        });
                    },
                    complete: function (jqXHR) { //JQery XML HTTP Request Object
                        if (jqXHR.status == '401') {
                            ulProducts.empty();
```

```
                            ulProducts.append('<li style="color:red">' +
                           jqXHR.status + ' : ' + jqXHR.statusText + '</li>');
                        }
                    }
                });
            });

            $('#btnClear').click(function () {
                ulProducts.empty();
            });
        })
    </script>
    <title></title>
</head>
<body>
    Username: <input type="text" id="txtUsername" />
    Password: <input type="password" id="txtPassword" />
    <br /><br />
    <input type="button" id="btn" value="Authenticate and Get Products" />
    <input type="button" id="btnClear" value="Clear" />
    <ul id="ulProducts"></ul>
</body>
</html>
```

ASP NET Web API token authentication

Select "Individual User Accounts" as the authenticion when creating the ASP.NET Web API application

Sample User Registration Page in bootstrap for ASP NET Web API token authentication

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <link href="Content/bootstrap.min.css" rel="stylesheet" />
    <title></title>
</head>
<body style="padding-top:20px">
    <div class="col-md-10 col-md-offset-1">
        <div class="well">
            <table class="table-bordered">
                <thead>
                    <tr class="success">
                        <td colspan="2">
                            New User Registration
                        </td>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>Email</td>
```

```
                        <td><input type="text" id="txtEmail" placeholder="Email"
/></td>
                    </tr>
                    <tr>
                        <td>Password</td>
                        <td><input type="password" id="txtPassword"
placeholder="Password" /></td>
                    </tr>
                    <tr>
                        <td>Confirm Password</td>
                        <td><input type="password" id="txtConfirmPassword"
placeholder="ConfirmPassword" /></td>
                    </tr>
                    <tr class="success">
                        <td colspan="2">
                            <input type="button" id="btnRegister" class="btn
btn-success" value="Register"/>
                        </td>

                    </tr>
                </tbody>
            </table>
            <div class="modal fade" tabindex="-1" id="successModal"
data-keyboard="false" data-backdrop="static">
                <div class="modal-dialog modal-sm">
                    <div class="modal-content">
                        <div class="modal-header">
                            <button type="button" class="close"
data-dismiss="modal">
                                &times;
                            </button>
                            <h4>Success</h4>
                        </div>
                        <div class="modal-body">
                            <h2>Registration Successful!</h2>
                        </div>
                        <div class="modal-footer">
                            <button type="button" data-dismiss="modal" class="btn
btn-success">Close</button>
                        </div>
                    </div>
                </div>
            </div>
            <div id="divError" class="alert alert-danger collapse">
                <a id="linkClose" class="close" href="#">&times;</a>
                <div id="divErrorText"></div>
            </div>
        </div>
    </div>

    <script src="Scripts/jquery-3.3.1.min.js"></script>
    <script src="Scripts/bootstrap.min.js"></script>
    <script type="text/javascript">
        $(document).ready(function () {
```

```
            $('#btnRegister').click(function () {
                $.ajax({
                    url: '/api/account/register',
                    method: 'POST',
                    data: {
                        email: $('#txtEmail').val(),
                        password: $('#txtPassword').val(),
                        confirmPassword: $('#txtConfirmPassword').val()
                    },
                    success: function () {
                        $('#successModal').modal('show');
                    },
                    error: function (jqXHR) {
                        $('#divErrorText').text(jqXHR.responseText);
                        $('#divError').show('fade');
                    }
                });
            });
            $('#linkClose').click(function () {
                $('#divError').hide('fade');
            });
        });
    </script>
</body>
</html>
```

sessionStorage (js)

sessionStorage data is lost when the browser window is closed

```
sessionStorage.setItem("access-token", response.access-token);
sessionStorage.getItem("access-token");
sessionStorage.removeItem("access-token");
```

Sample Login Webpage in bootstrap

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <link href="Content/bootstrap.min.css" rel="stylesheet" />
    <title></title>
</head>
<body style="padding-top:20px">
    <div class="col-md-10 col-md-offset-1">
        <div class="well">
            <table class="table-bordered">
                <thead>
                    <tr class="success">
                        <td colspan="2">
                            Existing user login
                            <a class="btn btn-success pull-right"
href="Registration.html">Register</a>
```

```
                        </td>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>Username</td>
                        <td><input type="text" id="txtUsername"
placeholder="Username" /></td>
                    </tr>
                    <tr>
                        <td>Password</td>
                        <td><input type="password" id="txtPassword"
placeholder="Password" /></td>
                    </tr>
                    <tr class="success">
                        <td colspan="2">
                            <input type="button" id="btnLogin" class="btn
btn-success" value="Login"/>
                        </td>

                    </tr>
                </tbody>
            </table>

            <div id="divError" class="alert alert-danger collapse">
                <a id="linkClose" class="close" href="#">&times;</a>
                <div id="divErrorText"></div>
            </div>
        </div>
    </div>

    <script src="Scripts/jquery-3.3.1.min.js"></script>
    <script src="Scripts/bootstrap.min.js"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            $('#btnLogin').click(function () {
                $.ajax({
                    url: '/token',
                    method: 'POST',
                    contentType: "application/json",
                    data: {
                        username: $('#txtUsername').val(),
                        password: $('#txtPassword').val(),
                        grant_type: 'password'
                    },
                    success: function (response) {
                        sessionStorage.setItem('accessToken',
response.access_token);
                        window.location.href = 'Data.html';
                    },
                    error: function (jqXHR) {
                        $('#divErrorText').text(jqXHR.responseText);
                        $('#divError').show('fade');
                    }
```

```
            });
        });
        $('#linkClose').click(function () {
            $('#divError').hide('fade');
        });
    });
</script>
</body>
</html>
```

Sample Response JSON Object with Token

```
{
       "access_token":"MXHMbZfyTNdlAHEjhi8FnSv_4QdprsFHDCpCeLhZdXVfhcAHShdAwohhdopw
LGHVAtigZ43yaYDlRVLvKpo319hwBb5ZUBs2PToyA3OhPGK5jOSRz-XdkSddhrwwJ1RWqZLszQ42L70s3gr
gLeIRDpfWElMebX8ehl4o-bU6oYO6eQNTFMY_HDSQY4jUrANByS1zAHVIlhznIwtPfLILcLw6mR14epKYud
lZXe5fm_97uC2DTTs8CZaLZWWCc5GVFmmATouho4vettPElcbI4otfOvxPwKXsIVXYUpGEBoZdYJrJEob6V
Tf2IWop1TEhb6BcqBB3XzQQ6OJFZL7TTM86m4GYIjV3B_Nisio7ZzHT6F4d0SHibsryZNW7aUV53xqn5eLH
144tVkZ-J9_zJKnep3HUd1YPqgnolv0gax0SudbPwVYY79OgLRKW5kF9RBzGkJq2XfVvZVRhaMaY8MT4DaE
PT7_UlBKnBRU3j4VUXBTiEEWyvz2NsOCwKo7-",
       "token_type":"bearer",
       "expires_in":1209599,"
       userName":"rajeeshvp@gmail.com",
       ".issued":"Fri, 26 Jul 2019 09:06:20 GMT",
       ".expires":"Fri, 09 Aug 2019 09:06:20 GMT"
}
```

Sample Data Display Webpage in bootstrap Using Token, Session, Login if not Logged in, Logoff

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <link href="Content/bootstrap.min.css" rel="stylesheet" />
    <title></title>
</head>
<body style="padding-top:20px">
    <div class="col-md-10 col-md-offset-1">
        <div class="well">
            <input type="button" id="btnLoadProducts" class="btn btn-success"
value="Load Products" />
            <input type="button" id="btnLogoff" class="btn btn-success pull-right"
value="Logoff"/>
        </div>
        <div class="well hidden" id="divData">
            <table class="table table-bordered" id="tblData">
                <thead>
                    <tr>
                        <th>
                            Id
                        </th>
                        <th>
```

```
                            Name
                        </th>
                    </tr>
                </thead>
                <tbody id="tblBody"></tbody>
            </table>
            <div class="modal fade" tabindex="-1" id="errorModal"
data-keyboard="false" data-backdrop="static">
                <div class="modal-dialog modal-sm">
                    <div class="modal-content">
                        <div class="modal-header">
                            <button type="button" class="close"
data-dismiss="modal">
                                &times;
                            </button>
                            <h4>Error</h4>
                        </div>
                        <div class="modal-body">
                            <h2>Session expired. Close this message to login</h2>
                        </div>
                        <div class="modal-footer">
                            <button type="button" data-dismiss="modal" class="btn
btn-success">Close</button>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <div id="divError" class="alert alert-danger collapse">
            <a id="linkClose" class="close" href="#">&times;</a>
            <div id="divErrorText"></div>
        </div>
    </div>

    <script src="Scripts/jquery-3.3.1.min.js"></script>
    <script src="Scripts/bootstrap.min.js"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            if (sessionStorage.getItem('accessToken') == null)
                window.location.href = 'Login.html';
            $('#btnLoadProducts').click(function () {
                $.ajax({
                    url: '/api/Product',
                    method: 'GET',
                    headers: {
                        'Authorization': 'Bearer ' +
sessionStorage.getItem('accessToken')
                    },
                    success: function (data) {
                        $('#divData').removeClass('hidden');
                        $('#tblBody').empty();
                        $.each(data, function (index, item) {
                            var row = $('<tr><td>'
                                + item.Id + '</td><td>'
```

```
                            + item.Name + '</td></tr>');
                        $('#tblData').append(row);
                    })
                },
                error: function (jqXHR) {
                    if (jqXHR.state == '401') { //session expired
                        $('#errorModal').modal('show');
                    }
                    else {
                        $('#divErrorText').text(jqXHR.responseText);
                        $('#divError').show('fade');
                    }
                }
            });
        });
        $('#linkClose').click(function () {
            $('#divError').hide('fade');
        });
        $('errorModal').on('hidden.bs.modal'), function () { //when session
expired message is closed
            window.location.href = "Login.html";
        };
        $('#btnLogoff').click(function () {
            sessionStorage.removeItem('accessToken');
            window.location.href = "Login.html";
        });
    });
</script>
</body>
</html>
```

localStorage

localStorage data is not lost when browser window is closed

Register Application with Google

To register the application with Google

1. navigate to https://console.developers.google/
2. Login
3. Create new project
4. Input project name and save
5. Click on credentials tab
6. Select OAuth consent screen
7. Provide a product name
8. Select “create credential” drop down menu
9. Select “OAuth client id”
10. Select “Web application”
11. Input Authorised JavaScript origins: ex. https://localhost:44310
12. Input Authorised redirect URIs: https://localhost:44310/signin-google
13. Save to get

    a. Clinet id:
       `740207003977-g7glc784l2gcv8b6pcq7f3p8lt4nbg6e.apps.googleusercontent.com`
    b. Client secret:
       `a22er_FkzKA40ITUFpGrF8BX`

14. Enable google plus api service
    a. Click on library
    b. User Social APIs, click on Google+ API
    c. Enable
15. After enabling google authentication on Startup.Auth.cs, goto https://localhost:44363/api/Account/ExternalLogins?returnUrl=%2F&generateState=true

    Result:

```
<ArrayOfExternalLoginViewModel
xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.datacontract.org/2004/07/WebApiTokenAuthenticationSample.Models">
<ExternalLoginViewModel>
<Name>Google</Name>
<State>RjNNR52FoSMixelgB4QPxSk0QqeHITapjqnFcqb4a-g1</State>
<Url>
/api/Account/ExternalLogin?provider=Google&response_type=token&client_id=self&redirect_uri=https%3A%2F%2Flocalhost%3A44363%2F&state=RjNNR52FoSMixelgB4QPxSk0QqeHITapjqnFcqb4a-g1
</Url>
</ExternalLoginViewModel>
</ArrayOfExternalLoginViewModel>
```

16. Use the <Url> link in Login.html
17. Encode the url https://localhost:44363/Login.html to https%3A%2F%2Flocalhost%3A44363%2FLogin.html
18. Change the redirect_uri
19. In ApplicationOAuthProvider.cs change Uri expectedRootUri = new Uri(context.Request.Uri, "/"); to Uri expectedRootUri = new Uri(context.Request.Uri, "/Login.html");
20.

Enable google authentication

App_Start\Startup.Auth.cs

```
public partial class Startup
{
	...
	public void ConfigureAuth(IAppBuilder app)
	{
		...
```

```
            app.UseGoogleAuthentication(new GoogleOAuth2AuthenticationOptions()
            {
                  ClientId =
            "740207003977-g7glc784l2gcv8b6pcq7f3p8lt4nbg6e.apps.googleusercontent.
            com",
                  ClientSecret = "a22er_FkzKA40ITUFpGrF8BX"
            });
      }
}
```

Attribute Routing

Without a route attribute, multiple get methods with same parameters do not work.

```
public class ValuesController : ApiController
{
      // GET api/values/5
      public MyValue Get(int id)
      {
            ...
      }

      // GET api/values/category/4
      [Route("api/values/category/{id}")]
      public MyValue GetCategory(int id)
      {
            ...
      }
}
```

RoutePrefix

By using `RoutePrefix` attribute, we can use `[Route("category/{id}")]` instead of `[Route("api/values/category/{id}")]`

```
[RoutePrefix("api/values")]
public class ValuesController : ApiController
{
      [Route("category/{id}")]
      public MyValue GetCategory(int id)
      {
            ...
      }
}
```

HTTPResponseMessage, HTTPStatusCode Sample

```
public HTTPResponseMessage Get(int id)
{
```

```
        var student = student.FirstOrDefault(s => s.Id == id);
        if(student == null)
        {
                return Request.CreateErrorResponse(HTTPStatusCode.NotFound,
                                                                "Student not found");
        }
        return Request.CreateResponse(HTTPStatusCode.OK, student);
}
```

IHTTPActionResult Sample

```
public IHTTPActionResult Get(int id)
{
        var student = student.FirstOrDefault(s => s.Id == id);
        if(student == null)
        {
                return NotFound();
                -OR-
                return Content(HTTPStatusCode.NotFound, "Student not found");
        }
        return OK(student);
}
```

Versioning

Why versioning required in Web API

- Once a Web API service is made public, different clinet applications start using and relying on it
- As the business grows and requirements change, the services need to change without breaking existing clients and at the same time satisfying new client requirements

Versioning can be implemented using

- URIs
- QueryString
- Version Header
- Accept Header
- Media Type

URIs Sample

Separate controllers required for separate versions

```
public static class WebApiConfig
{
        public static void Register(HttpConfiguration config)
        {
                ...
```

```
            config.Routes.MapHttpRoute(
                  name: "Version1", //Any name
                  routeTemplate: "api/v1/students/{id}",
                  defaults: new { id = RouteParameter.Optional, "StudentsV1" }

            config.Routes.MapHttpRoute(
                  name: "Version2", //Any name
                  routeTemplate: "api/v2/students/{id}",
                  defaults: new { id = RouteParameter.Optional, "StudentsV1" }
            );
      }
}
```
EoF