(T2)討論Api的Post、Get、Put、Delete對應到資料庫的Create、Read、Update、Delete(CRUD)  
CourseGUID 4c5822ff-7111-4e25-a336-ef18d48d54bd  
=======================================================================  
(T2)討論Api的Post、Get、Put、Delete對應到資料庫的Create、Read、Update、Delete(CRUD)

(T2-1)討論Api的Get (1. to 4.)

(T2-2)用Fiddler測試Api的Get的Json、Xml格式 (5.)

(T2-3)用FPostman測試Api的Get的Json、Xml格式 (6.)

(T2-4)討論Api的Get對應到資料庫的Read (7.)

(T2-5)討論Api的Post對應到資料庫的Create (7.)

(T2-6)討論Api的Delete對應到資料庫的Delete (8.)

(T2-7)討論Api的Put對應到資料庫的Update (9.)  
=======================================================================  
1. OnlineGame DB

1.0. Some points

1.1. TSQL

1.2. Security login

-----------

2. OnlineGame Solution

2.1. OnlineGame Solution

2.2. OnlineGame.WebApi

2.3. OnlineGame.Data

-----------

3. OnlineGame.Data

3.1. Install Entity Framework

3.2. ADO.Net Entity Data Model - Entity Framework

-----------

4. OnlineGame.WebApi

4.1. Install Entity Framework

4.2. Web.config : Add Connection String

4.3. Add Reference

4.4. Controllers/GamerController.cs

4.5. Controllers/GamerController.cs

-----------

5. Fiddler - GET

5.1. Fiddler - Get Response by XML/JSON format

5.2. OnlineGame.WebApi/App\_Start/WebApiConfig.cs - serialization from pascal case to camel case

-----------

6. Postman - GET

6.1. Postman - Get Response by XML/JSON format

-----------

7. Controllers/GamerController.cs - GET, POST

7.1. Controllers/GamerController.cs - GET, POST

7.2. Controllers/GamerController.cs - GET

7.2.1. Controllers/GamerController.cs - GET

7.2.2. Controllers/GamerController.cs - GET, Better version

7.3. Controllers/GamerController.cs - POST

7.3.1. Controllers/GamerController.cs - POST

7.3.2. Controllers/GamerController.cs - POST, Better version

-----------

8. Controllers/GamerController.cs - Delete

8.1. Controllers/GamerController.cs

8.2. Controllers/GamerController.cs -Delete

8.2.1. Controllers/GamerController.cs -Delete

8.2.2. Controllers/GamerController.cs - GET, Better version

-----------

9. Controllers/GamerController.cs - Put

9.1. Controllers/GamerController.cs - Put

9.2. Controllers/GamerController.cs - Put

9.2.1. Controllers/GamerController.cs - Put

9.2.2. Controllers/GamerController.cs - PUT, Better version  
=======================================================================

1. OnlineGame DB

The tutorial will discuss

What is RESTful ASP.NET Web API

Create the API with Get、Post、Put、Delete

and then Read, Insert, Update, Delete data from the database

-------------------------------------

本堂課討論

建立一個API with Get、Post、Put、Delete

並且Read, Insert, Update, Delete data from the database

1.0. Some points

1.

Regular expression

<https://regexr.com/>

2.

Calling Stored Procedure from Entity Framework 6 Code First

<http://www.dotnetodyssey.com/2015/03/12/calling-stored-procedure-from-entity-framework-6-code-first/>

1.1. TSQL

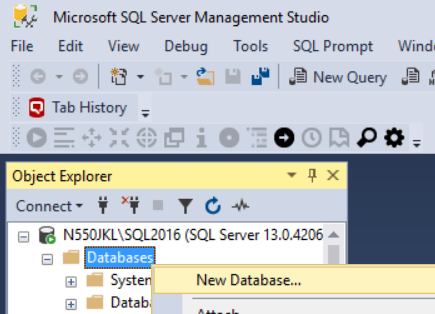
In SQL server Management Studio (SSMS)

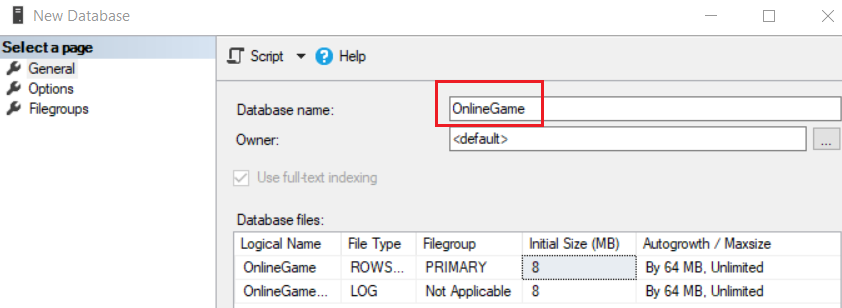
Database --> Right Click --> New Database -->

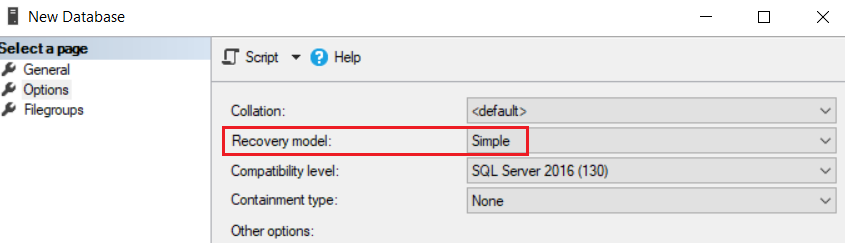
In General Tab -->

Name: **OnlineGame**

In options Tab --> Recovery model : **Simple**







--1 ----------------------------------------------------------

--Drop Table if it exists.

IF ( EXISTS ( SELECT    \*

              FROM      INFORMATION\_SCHEMA.TABLES

              WHERE     TABLE\_NAME = 'Gamer' ) )

    BEGIN

        TRUNCATE TABLE Gamer;

        DROP TABLE Gamer;

    END;

GO -- Run the previous command and begins new batch

--2 ----------------------------------------------------------

CREATE TABLE Gamer

    (

      Id INT PRIMARY KEY

             IDENTITY(1, 1)

             NOT NULL ,

      Name NVARCHAR(50) NOT NULL ,

      Gender NVARCHAR(50) NOT NULL ,

      Score INT NOT NULL ,

      GameMoney INT NOT NULL

    );

GO -- Run the previous command and begins new batch

--3 ----------------------------------------------------------

INSERT  INTO Gamer

VALUES  ( 'NameOne ABC', 'Male', 5000, 550 );

INSERT  INTO Gamer

VALUES  ( 'NameTwo ABCDE', 'Female', 4500, 1200 );

INSERT  INTO Gamer

VALUES  ( 'NameThree EFGH', 'Male', 6500, 3050 );

INSERT  INTO Gamer

VALUES  ( 'NameFour HIJKLMN', 'Female', 45000, 450 );

INSERT  INTO Gamer

VALUES  ( 'NameFive NOP', 'Male', 3000, 200 );

INSERT  INTO Gamer

VALUES  ( 'NameSix PQRSTUVW', 'Male', 4000, 700 );

INSERT  INTO Gamer

VALUES  ( 'NameSeven XYZ', 'Male', 450, 1500 );

GO -- Run the previous command and begins new batch

1.2. Security login

In SQL server

Object Explorer --> Security --> Logins --> New Logins

-->

General Tab

Login Name :

**Tester2**

Password:

**1234**

Default Database:

**OnlineGame**

-->

Server Roles Tab

Select

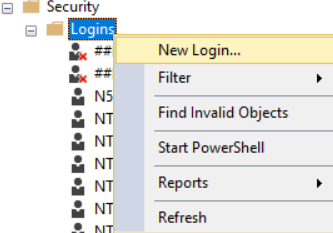
**sysadmin**

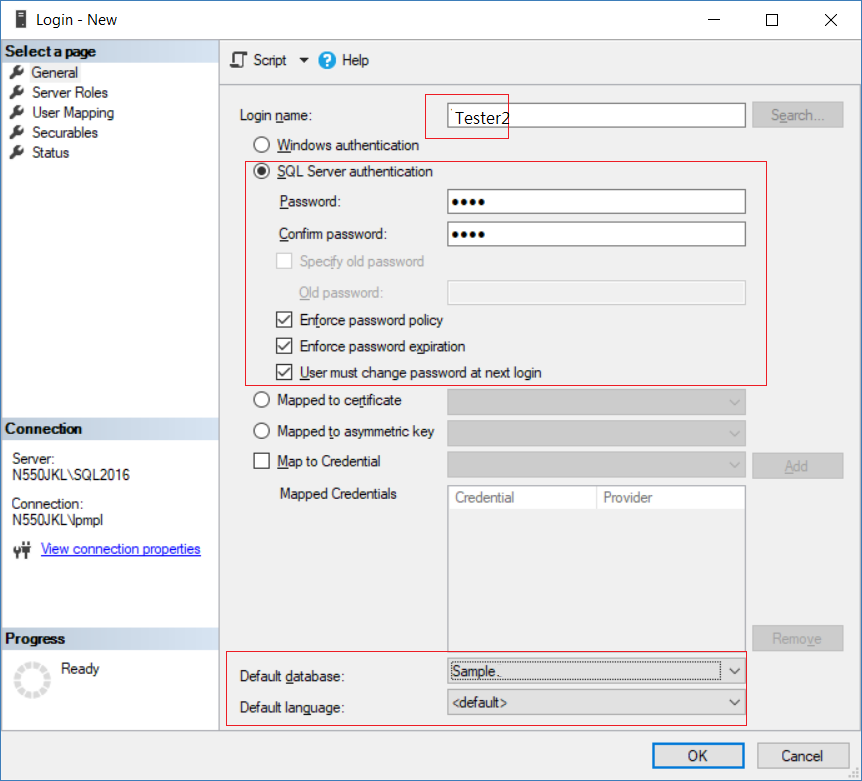
-->

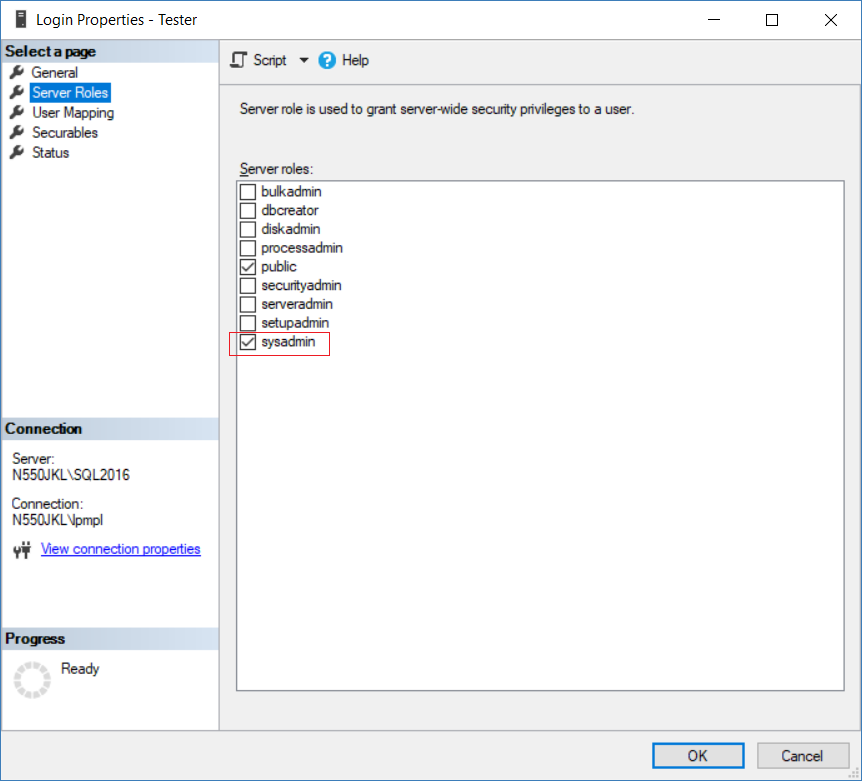
User Mapping Tab

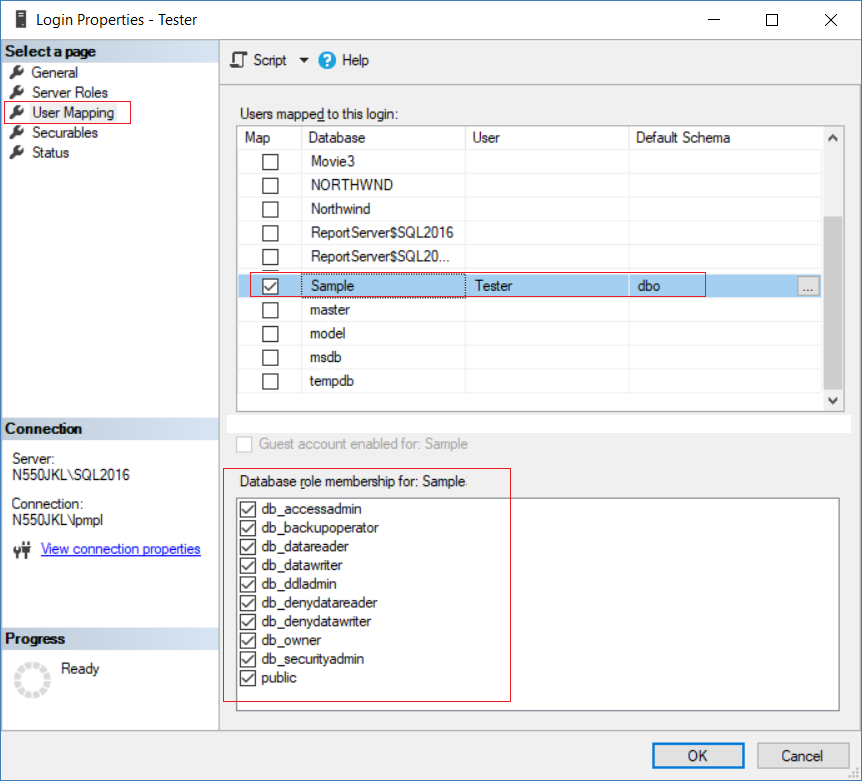
Select **OnlineGame**

Select every single role.









2. OnlineGame Solution

2.1. OnlineGame Solution

File --> New --> Project... -->

Other Project Types --> Visual Studio Solutions -->  Blank Solution

-->

Name: **OnlineGame**

2.2. OnlineGame.WebApi

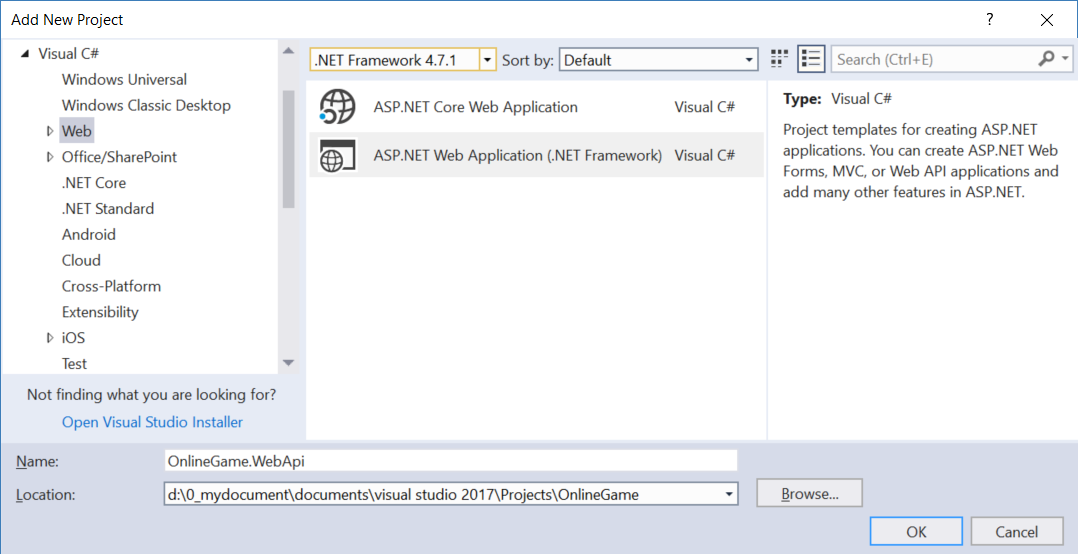
Solutions Name --> Add --> New Project -->

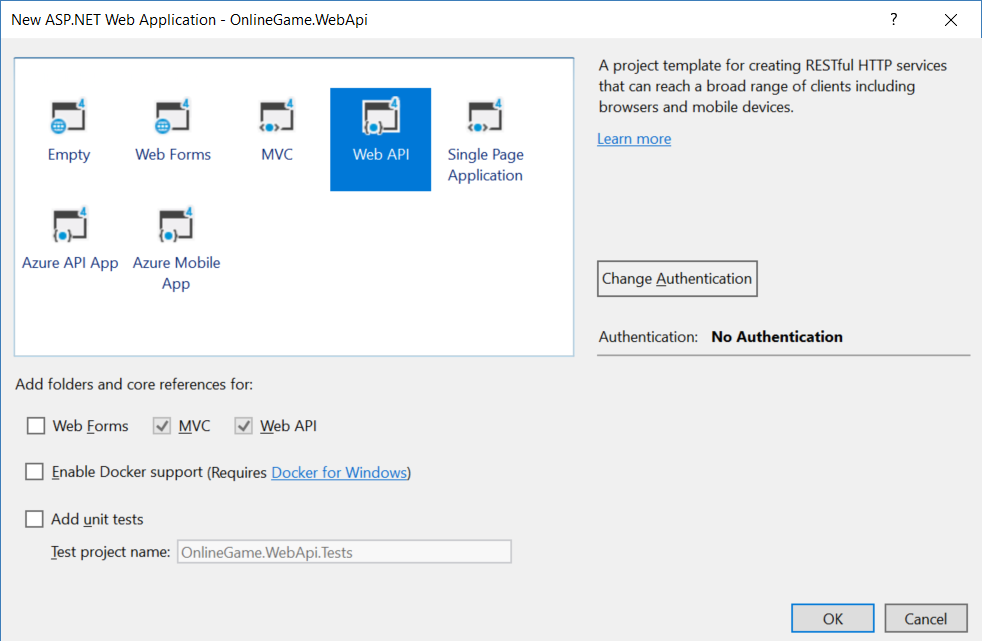
Visual C# --> Web --> [ASP.NET](http://asp.net/)Web Application (.Net Framework)

-->

Name: **OnlineGame.WebApi**

--> Select "**Web API**" --> OK





2.3. OnlineGame.Data

Solutions Name --> Add --> New Project -->

Visual C# --> **Class Library (.NET Framework)**

-->

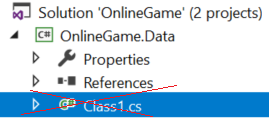
Name:

**OnlineGame.Data**

-->

Delete Class1.cs





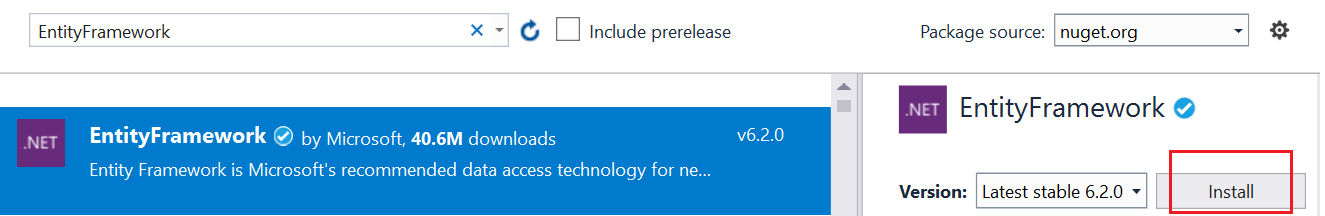
3. OnlineGame.Data

3.1. Install Entity Framework

Tools --> NuGet Package Manager --> Manage NuGet Packages for Solutions...

--> Browse tab --> Search  :  **EntityFramework**

--> Install it



3.2. ADO.Net Entity Data Model - Entity Framework

In Visual Studio 2017

**Project Name** --> Right Click --> Add --> New Item

--> Visual C# --> Data  -->  ADO.Net Entity Data Model

Name:

**OnlineGameDataModel**

-->

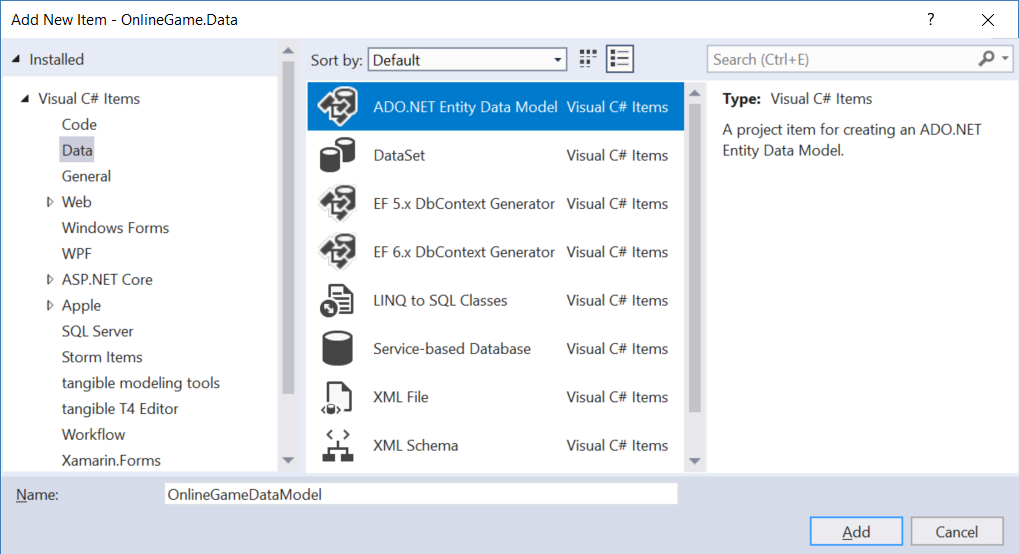
EF Designer from database

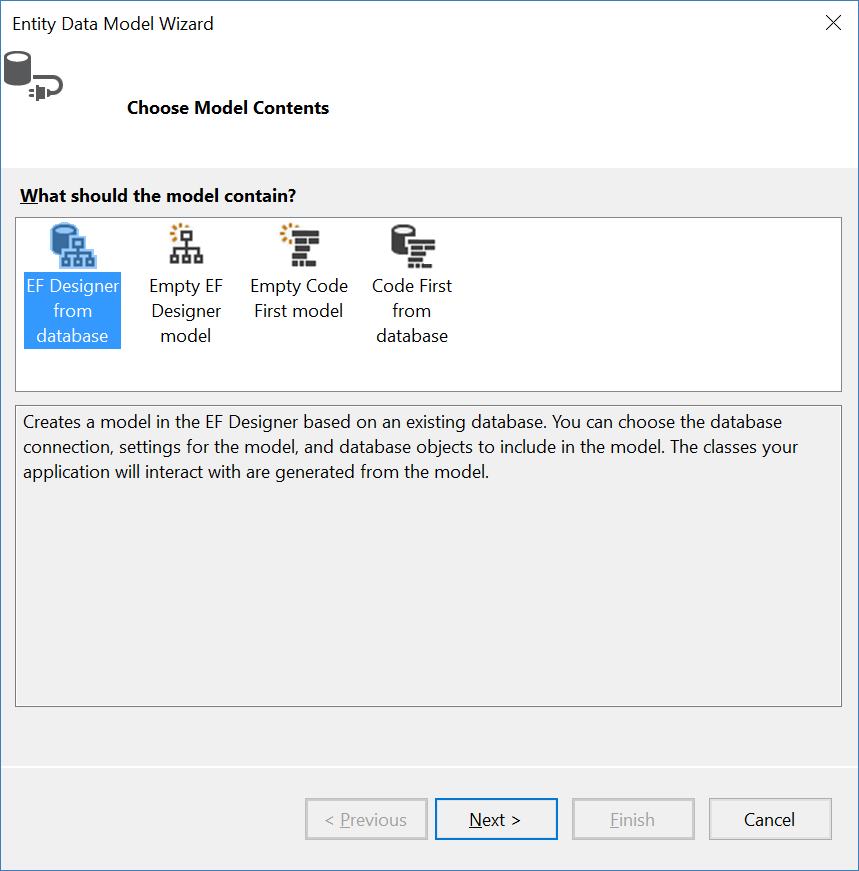
....

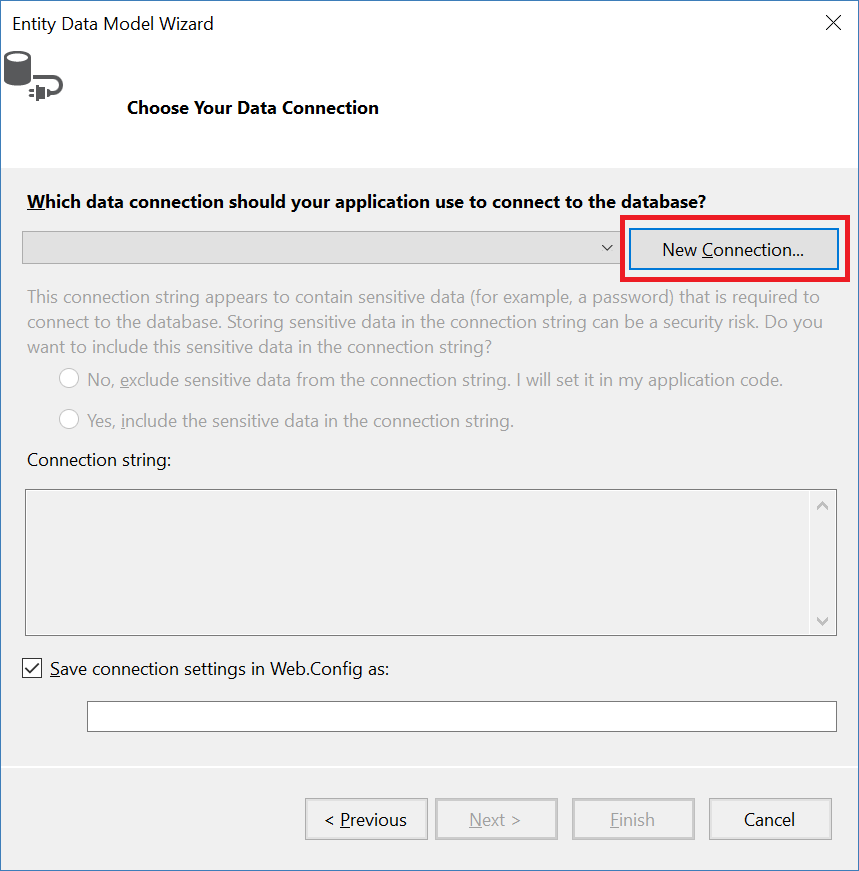
-->

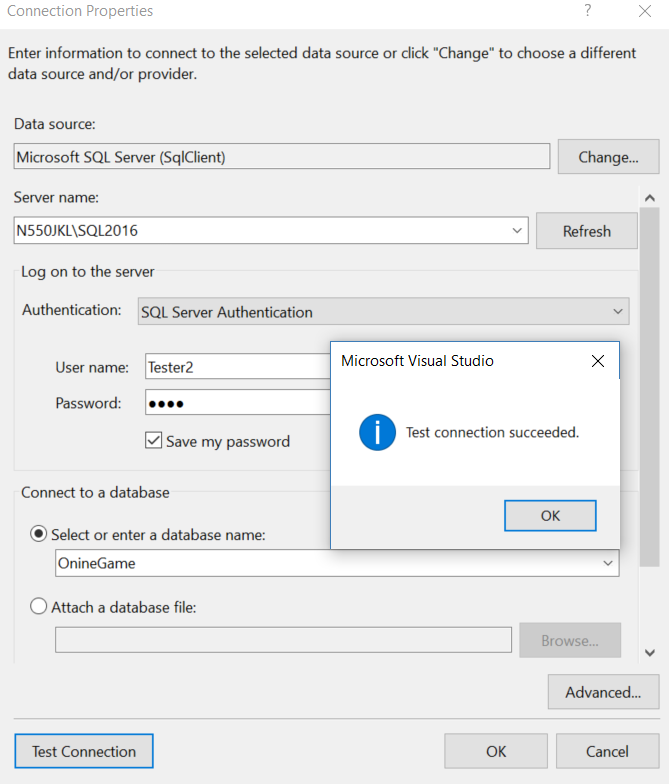
Save Connection settings in Web.Config as:

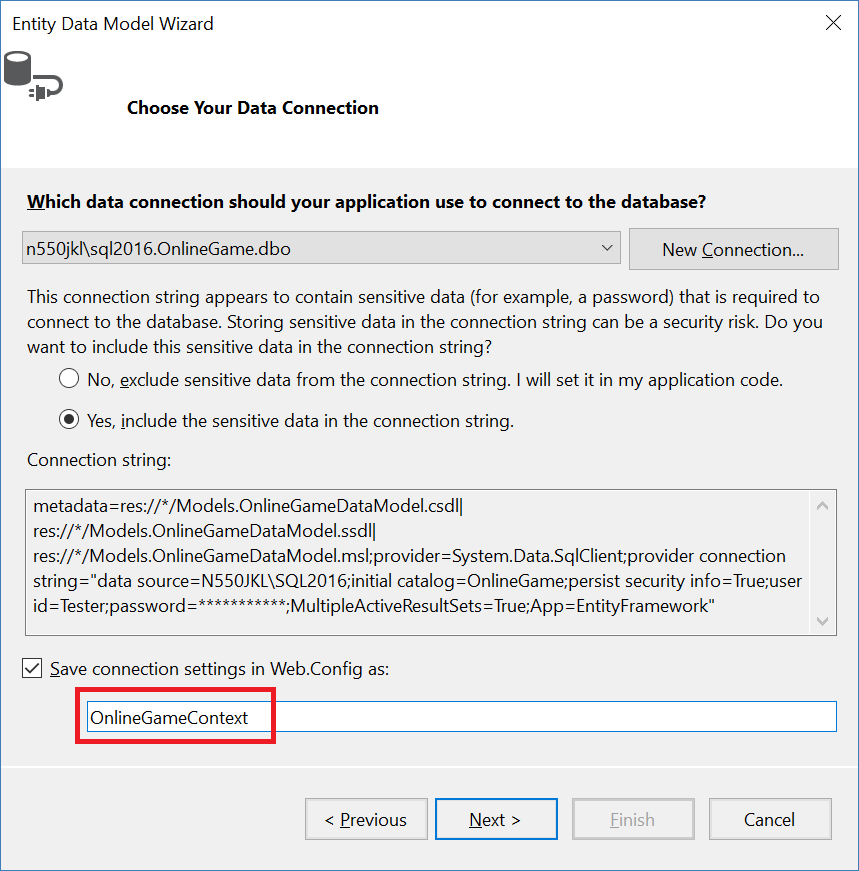
**OnlineGameContext**

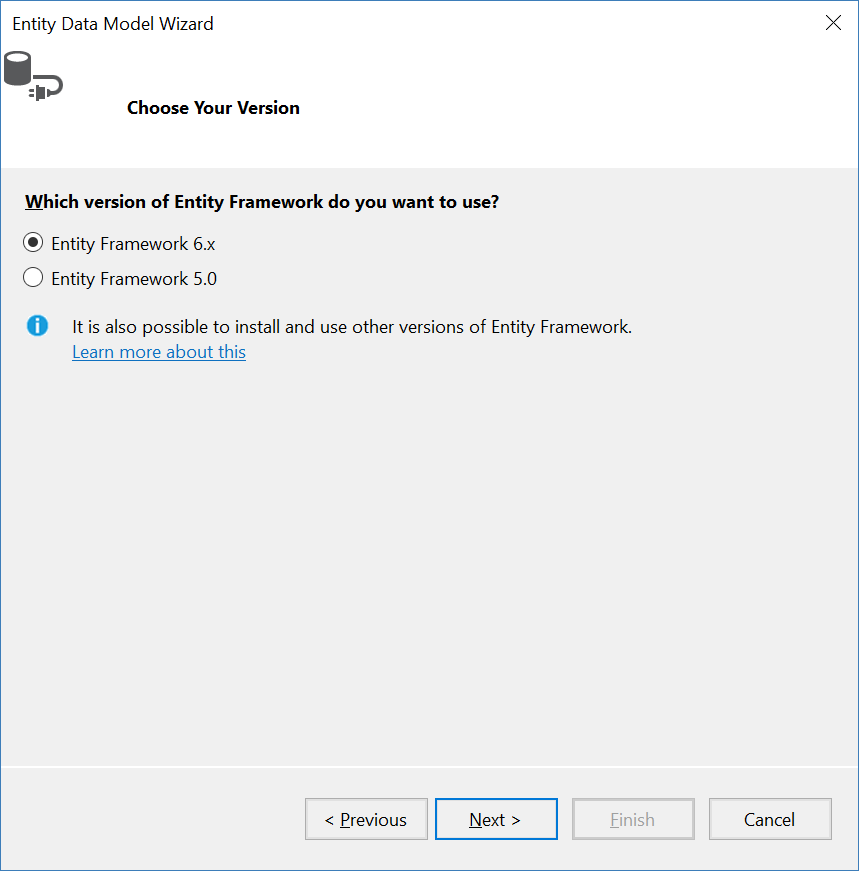


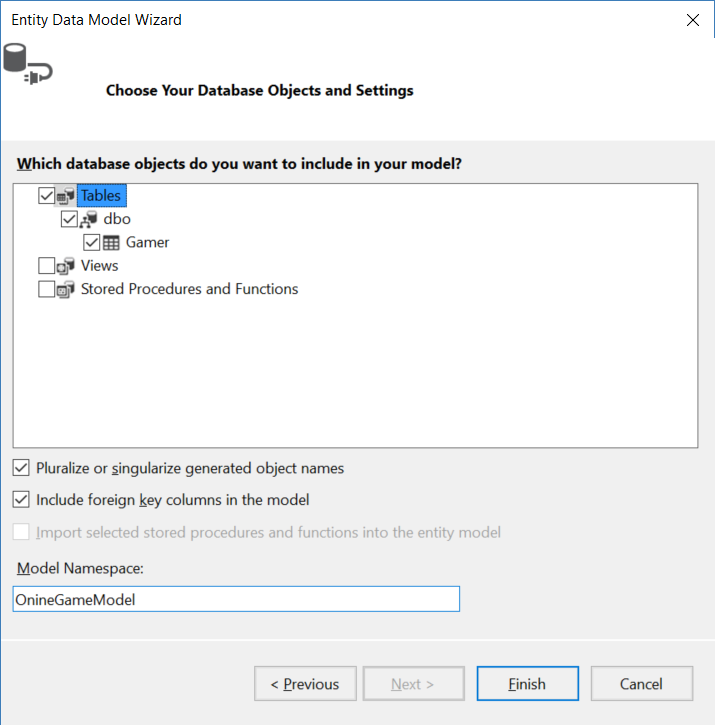


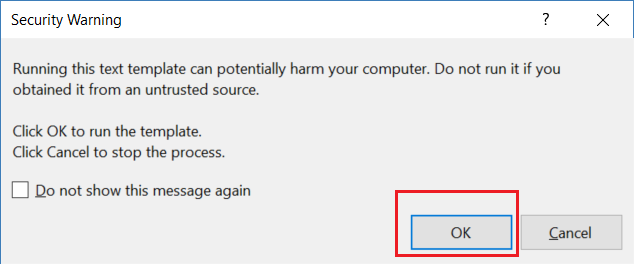


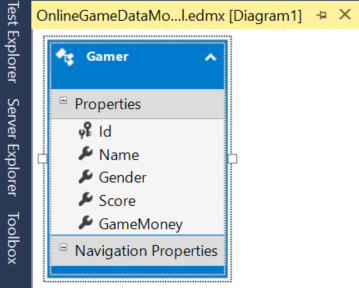












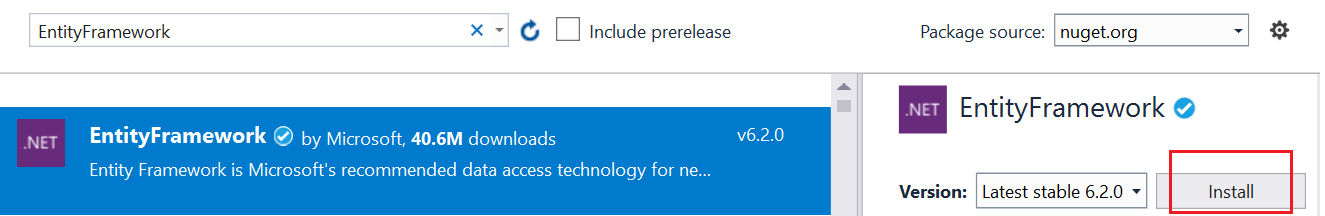
4. OnlineGame.WebApi

4.1. Install Entity Framework

Tools --> NuGet Package Manager --> Manage NuGet Packages for Solutions...

--> Browse tab --> Search  :  **EntityFramework**

--> Install it



4.2. Web.config : Add Connection String

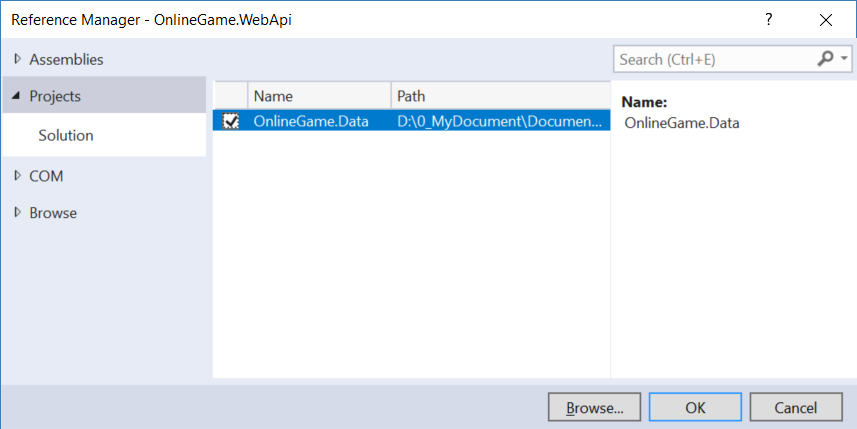


<connectionStrings>

    <add name="OnlineGameContext" connectionString="metadata=res://\*/OnlineGameDataModel.csdl|res://\*/OnlineGameDataModel.ssdl|res://\*/OnlineGameDataModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=N550JKL\SQL2016;initial catalog=OnineGame;persist security info=True;user id=Tester2;password=1234;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

  </connectionStrings>

4.3. Add Reference



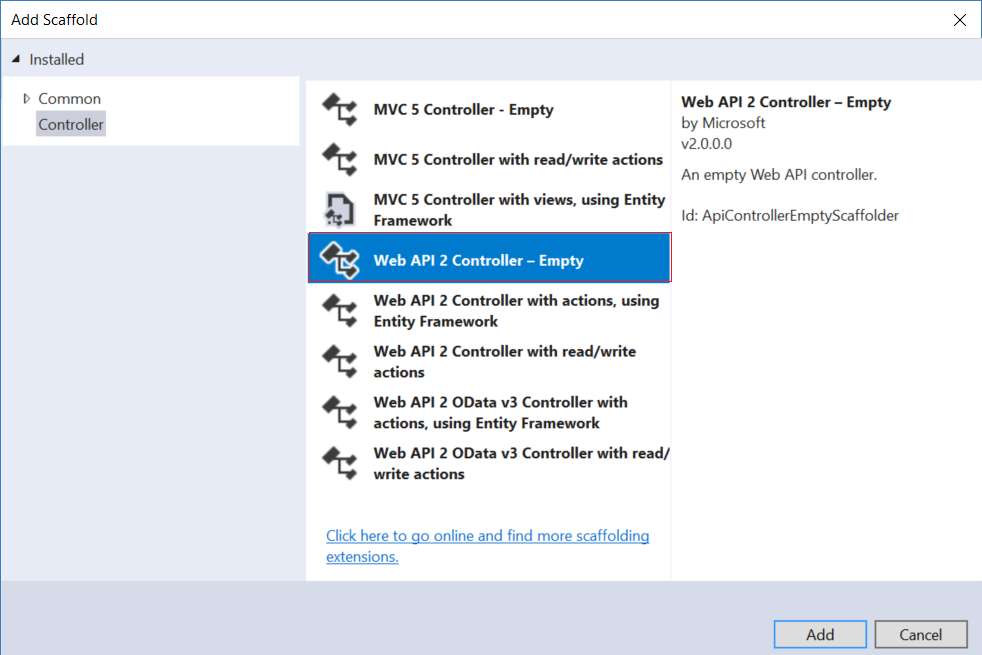
4.4. Controllers/GamerController.cs

Controllers  folder --> Right Click --> Add --> Controller

--> Web API 2 Controller - Empty

--> **GamerController**

if you have any error message, please ensure re-build whole solutions.



Graphical user interface, application, Word

Description automatically generated

4.5. Controllers/GamerController.cs

using System.Collections.Generic;

using System.Data.Entity;

using System.Threading.Tasks;

using System.Web.Http;

using OnlineGame.Data;

namespace OnlineGame.WebApi.Controllers

{

    public class GamerController : ApiController

    {

        public async Task<IEnumerable<Gamer>> Get()

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                return await db.Gamers.ToListAsync();

            }

        }

        public async Task<Gamer> Get(int id)

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                return await db.Gamers.FirstOrDefaultAsync(g => g.Id == id);

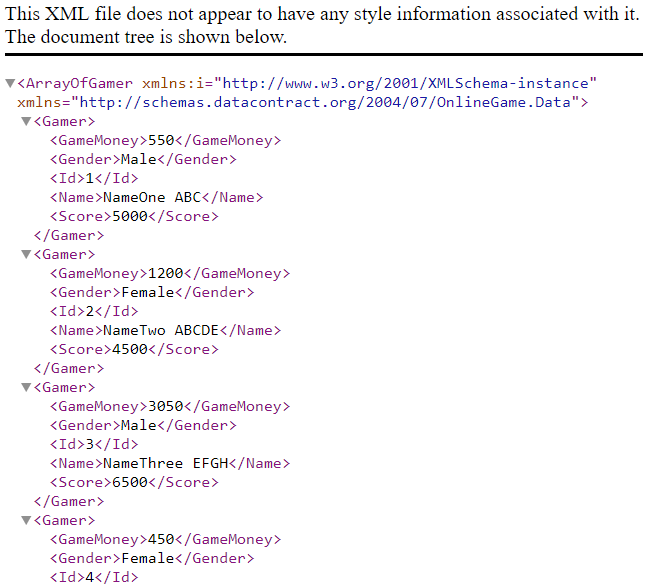
            }

        }

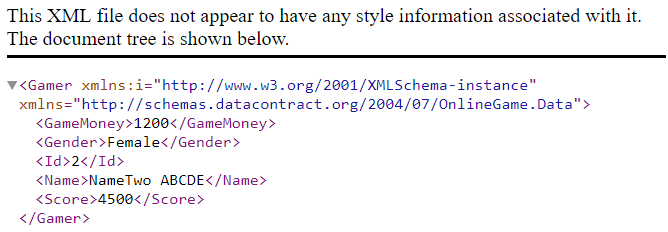
    }

}

<http://localhost:52578/api/gamer>



<http://localhost:52578/api/gamer/2>



5. Fiddler - GET

5.1. Fiddler - Get Response by XML/JSON format

Graphical user interface, text, application, email

Description automatically generated

1.

The Client can use Accept in Request header to specify the format for the response.

Content Negotiation means the format of the response from the server depends on the **Accept in client Request** header.

E.g.1.

<http://localhost:52578/api/gamer>

E.g. 1.1.

**Host: localhost:52578**

**Accept: application/json**

Return Response in JSON format

E.g. 1.2.

**Host: localhost:52578**

**Accept: application/xml**

Return Response in XML format

2.

Accept in client Request can take multiple values.

Theoretically, the server will pick the first format.

However, when I test the theory, for some reason, the theory is wrong.

When I specify JSON and XML together, no matter which order,

the response from server will be in JSON format.

I will say please always have a real test before you give the conclusion.

E.g. 2.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

**Accept: application/json, application/xml**

OR

**Host: localhost:52578**

**Accept: application/xml, application/json**

Both will return the response in JSON format

3.

**q means quality factor.**

When Accept in client Request has multiple values,

theoretically, it will see which format has a higher quality factor,

then the response will be in that format.

E.g.3.

<http://localhost:52578/api/gamer>

E.g. 3.1.

**Host: localhost:52578**

**Accept: application/xml;q=0.2,application/json;q=0.1**

q of XML is higher, so it will return the response in XML format

E.g. 3.2.

**Host: localhost:52578**

**Accept: application/xml;q=0.1,application/json;q=0.2**

q of JSON is higher, it will still return the response in JSON format

4.

If there is no Accept in client Request, by default, it will return **JSON** format.

E.g.4.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

------------------------------------------------

E.g.1.

<http://localhost:52578/api/gamer>

E.g. 1.1.

**Host: localhost:52578**

**Accept: application/json**

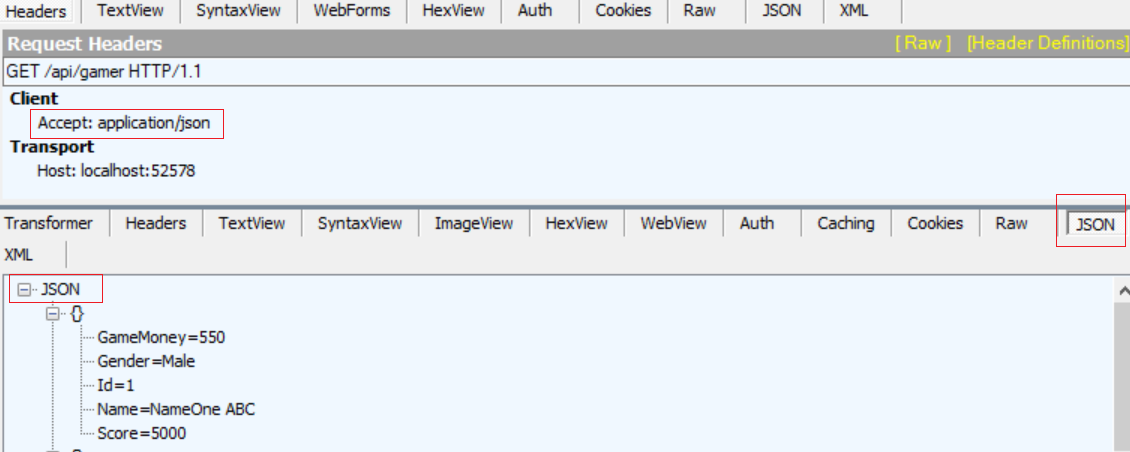
Return Response in JSON format

E.g. 1.2.

**Host: localhost:52578**

**Accept: application/xml**

Return Response in XML format





----------------------------------------------

E.g. 2.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

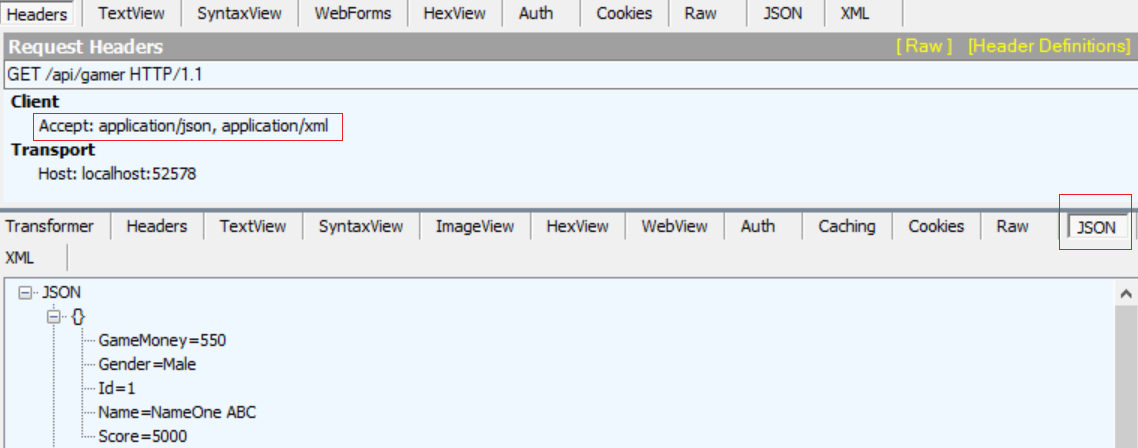
**Accept: application/json, application/xml**

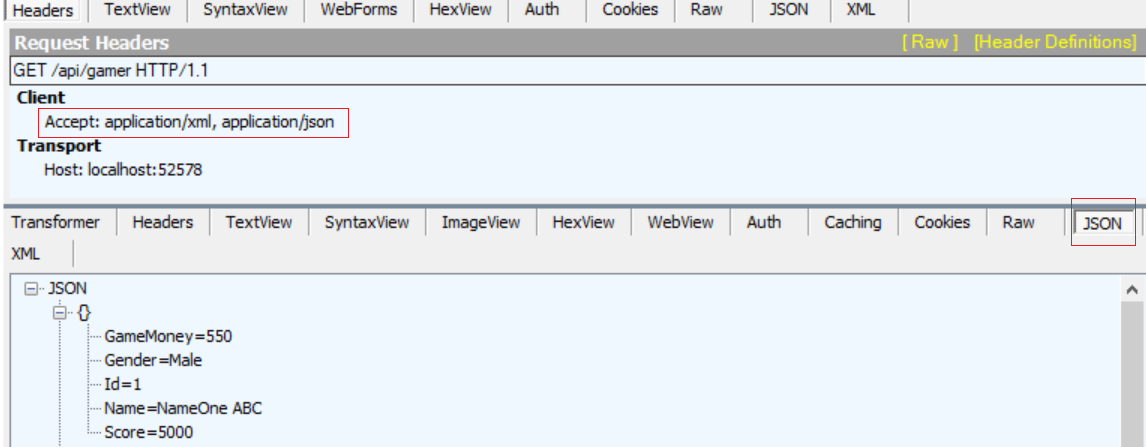
OR

**Host: localhost:52578**

**Accept: application/xml, application/json**

Both will return the response in JSON format





----------------------------------------------

E.g.3.

<http://localhost:52578/api/gamer>

E.g. 3.1.

**Host: localhost:52578**

**Accept: application/xml;q=0.2,application/json;q=0.1**

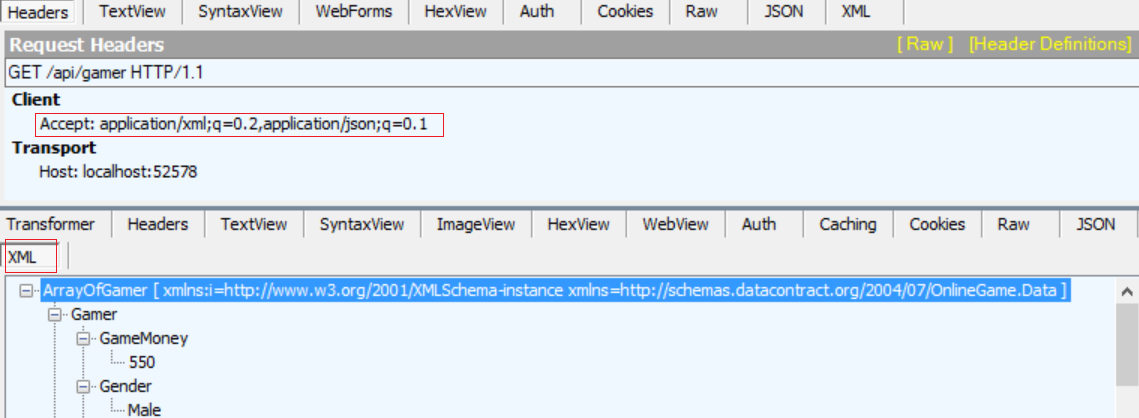
q of XML is higher, so it will return the response in XML format

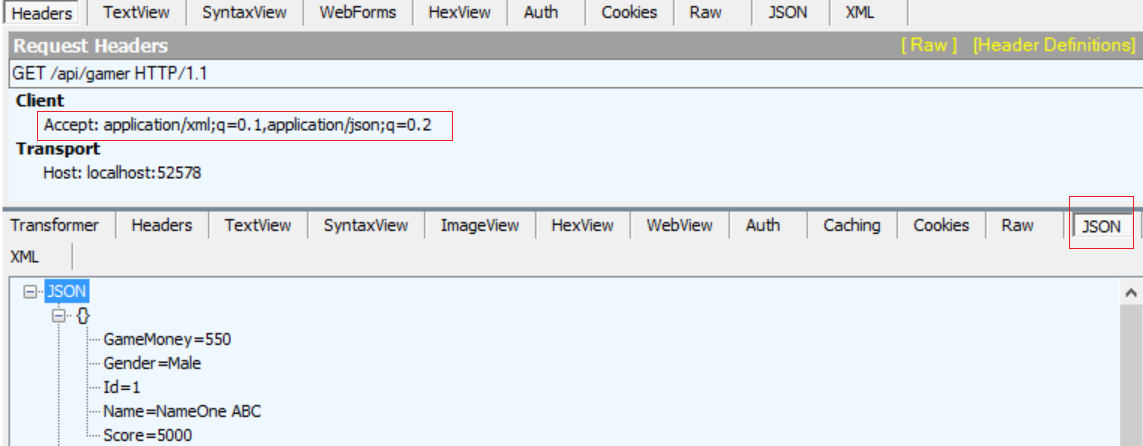
E.g. 3.2.

**Host: localhost:52578**

**Accept: application/xml;q=0.1,application/json;q=0.2**

q of JSON is higher, it will still return the response in JSON format





----------------------------------------------

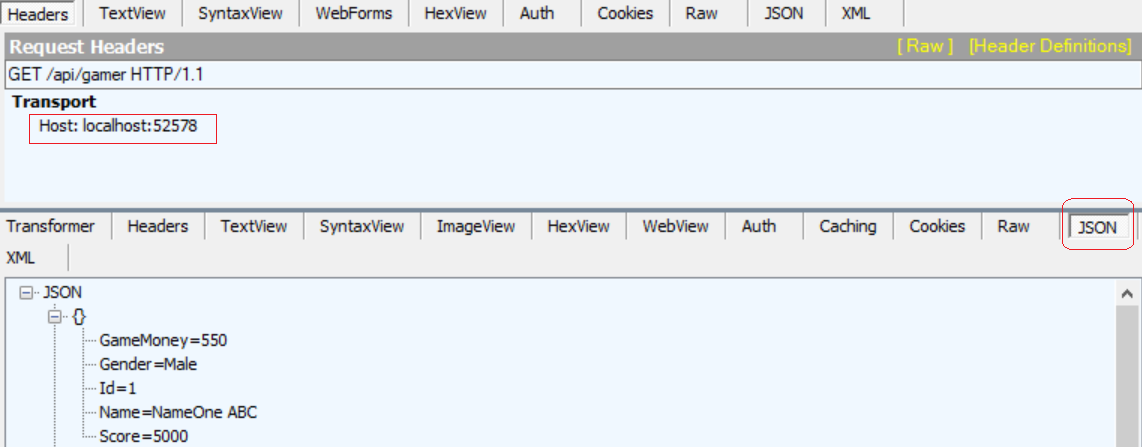
4.

If there is no Accept in client Request, by default, it will return **JSON** format.

E.g.4.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**



5.2. OnlineGame.WebApi/App\_Start/WebApiConfig.cs - serialization from pascal case to camel case

using System.Web.Http;

using Newtonsoft.Json.Serialization;

namespace OnlineGame.WebApi

{

    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 }

            );

            //by default, the server will return the response in Json format with pascal case.

            //If the server want to return the camel case Json data,

            //you have to modify the serialization settings as shown below

            config.Formatters.JsonFormatter.SerializerSettings.Formatting =

                            Newtonsoft.Json.Formatting.Indented;

            config.Formatters.JsonFormatter.SerializerSettings.ContractResolver =

                new CamelCasePropertyNamesContractResolver();

        }

    }

}

/\*

1.

Content Negotiation means the format of the response from server

is depending on the Accept in client Request header.

When Accept in client Request is "application/json",

by default, the server will return the response in Json format with pascal case.

E.g. pascal case --> "GamerMoney"

E.g. camel case --> "gamerMoney"

If the server want to return the camel case Json data,

you have to modify the serialization settings as shown below

//config.Formatters.JsonFormatter.SerializerSettings.Formatting =

//Newtonsoft.Json.Formatting.Indented;

//config.Formatters.JsonFormatter.SerializerSettings.ContractResolver =

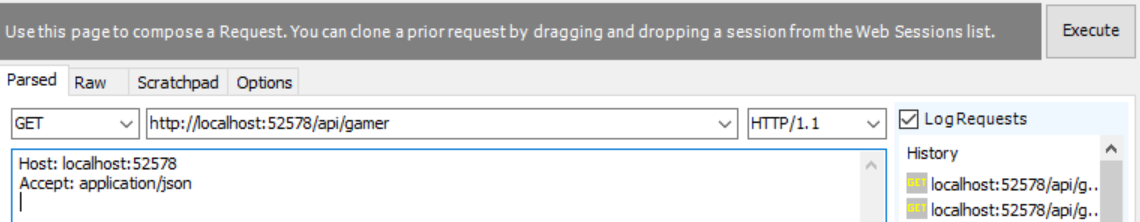
//new CamelCasePropertyNamesContractResolver();

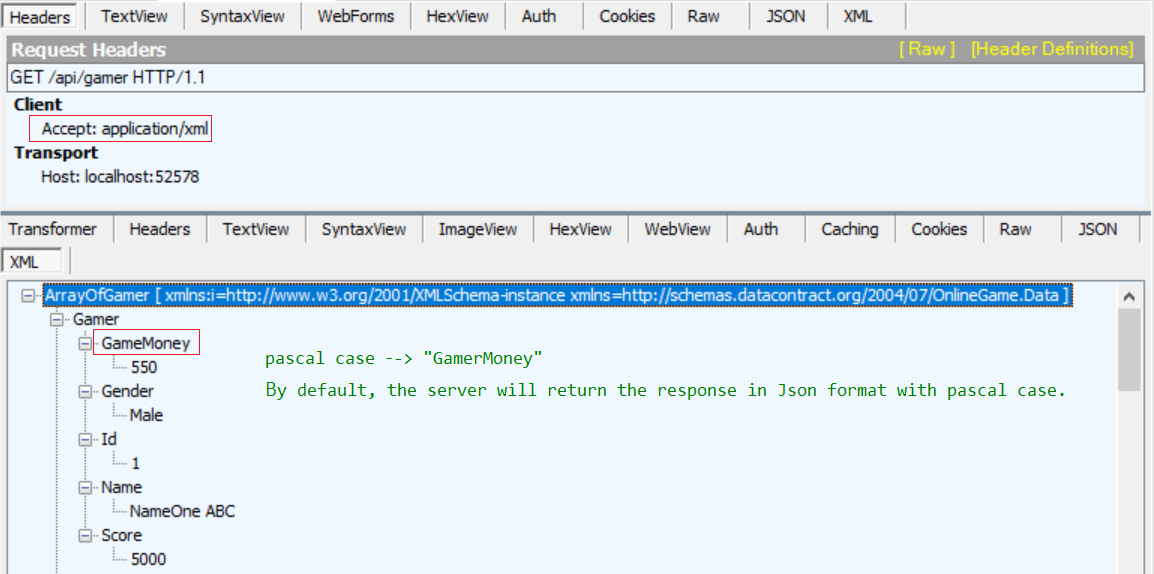
\*/

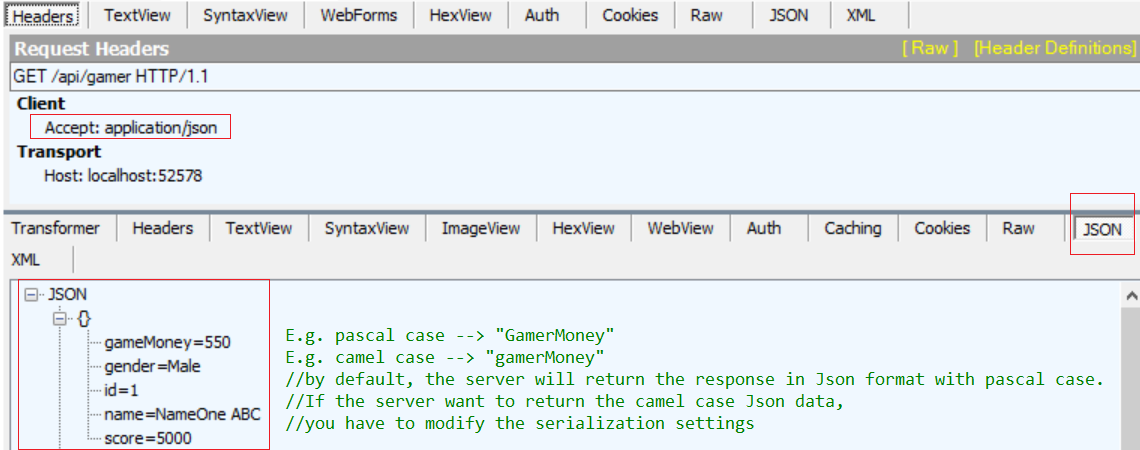
<http://localhost:52578/api/gamer>

**Host: localhost:52578**

**Accept: application/json**







6. Postman - GET

6.1. Postman - Get Response by XML/JSON format

1.

The Client can use Accept in Request header to specify the format for the response.

Content Negotiation means the format of the response from the server depends on the **Accept in client Request** header.

E.g.1.

<http://localhost:52578/api/gamer>

E.g. 1.1.

**Host: localhost:52578**

**Accept: application/json**

Return Response in JSON format

E.g. 1.2.

**Host: localhost:52578**

**Accept: application/xml**

Return Response in XML format

2.

Accept in client Request can take multiple values.

Theoretically, the server will pick the first format.

However, when I test the theory, for some reason, the theory is wrong.

When I specify JSON and XML together, no matter which order,

the response from sever will be in JSON format.

I will say please always have a real test before you give the conclusion.

E.g. 2.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

**Accept: application/json, application/xml**

OR

**Host: localhost:52578**

**Accept: application/xml, application/json**

Both will return the response in JSON format

3.

**q means quality factor.**

When Accept in client Request has multiple values,

theoretically, it will see which format has a higher quality factor,

then the response will be in that format.

E.g.3.

<http://localhost:52578/api/gamer>

E.g. 3.1.

**Host: localhost:52578**

**Accept: application/xml;q=0.2,application/json;q=0.1**

q of XML is higher, so it will return the response in XML format

E.g. 3.2.

**Host: localhost:52578**

**Accept: application/xml;q=0.1,application/json;q=0.2**

q of JSON is higher, it will still return the response in JSON format

4.

If there is no Accept in client Request, by default, it will return **JSON** format.

E.g.4.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

----------------------------------------------------------------------------------

E.g.1.

<http://localhost:52578/api/gamer>

E.g. 1.1.

**Host: localhost:52578**

**Accept: application/json**

Return Response in JSON format

E.g. 1.2.

**Host: localhost:52578**

**Accept: application/xml**

Return Response in XML format

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

----------------------------------------------------------------------------------

E.g. 2.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**

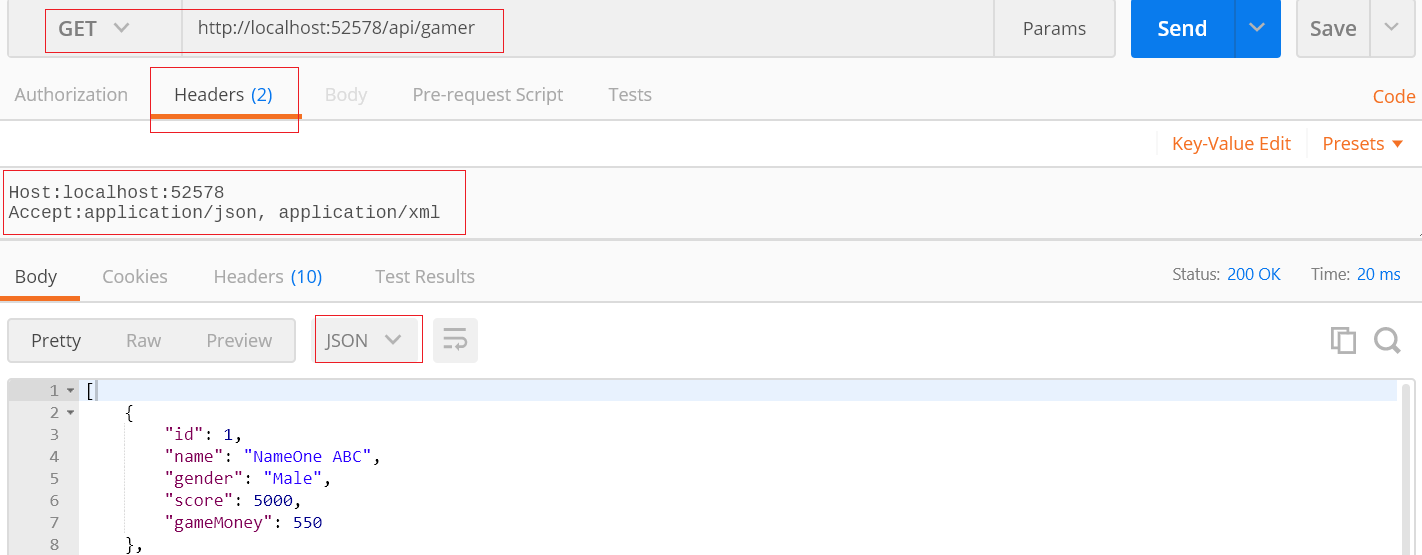
**Accept: application/json, application/xml**

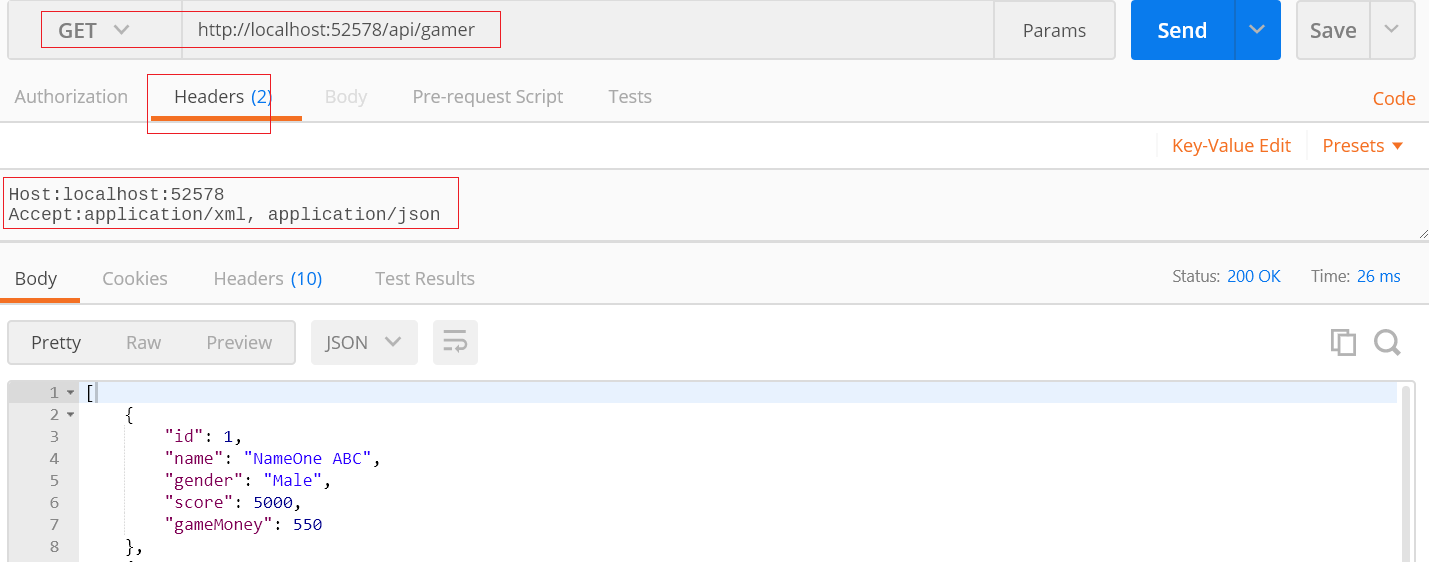
OR

**Host: localhost:52578**

**Accept: application/xml, application/json**

Both will return the response in JSON format





----------------------------------------------------------------------------------

E.g.3.

<http://localhost:52578/api/gamer>

E.g. 3.1.

**Host: localhost:52578**

**Accept: application/xml;q=0.2,application/json;q=0.1**

q of XML is higher, so it will return the response in XML format

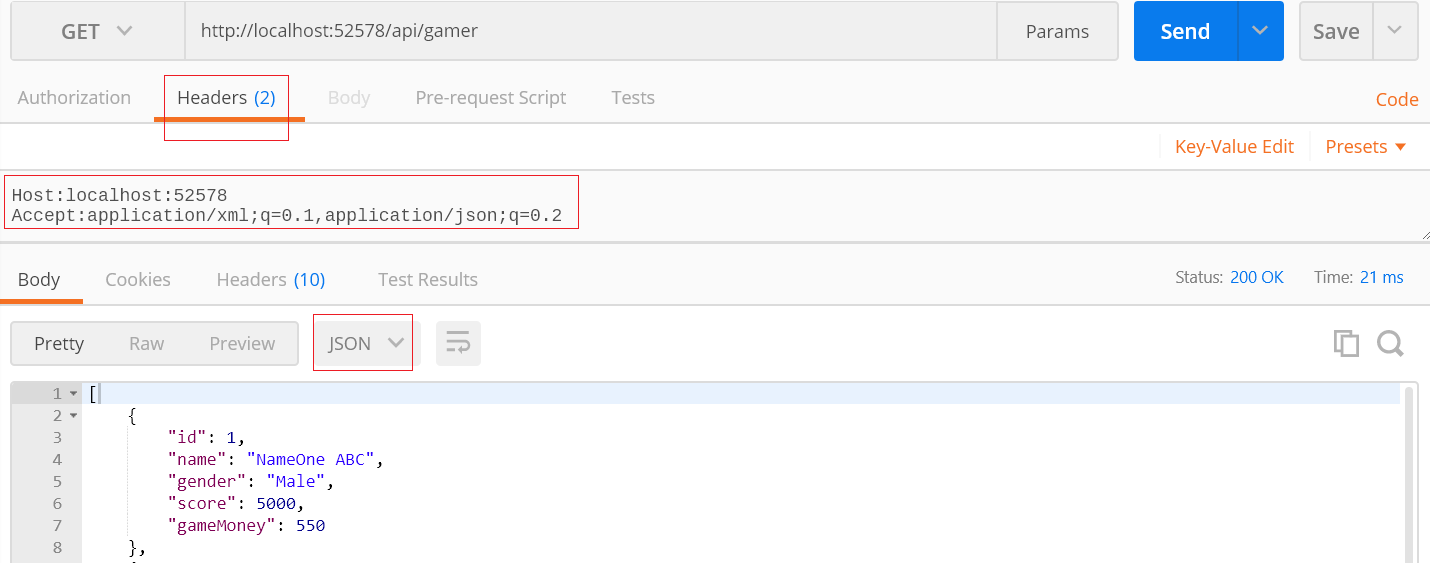
E.g. 3.2.

**Host: localhost:52578**

**Accept: application/xml;q=0.1,application/json;q=0.2**

q of JSON is higher, it will still return the response in JSON format



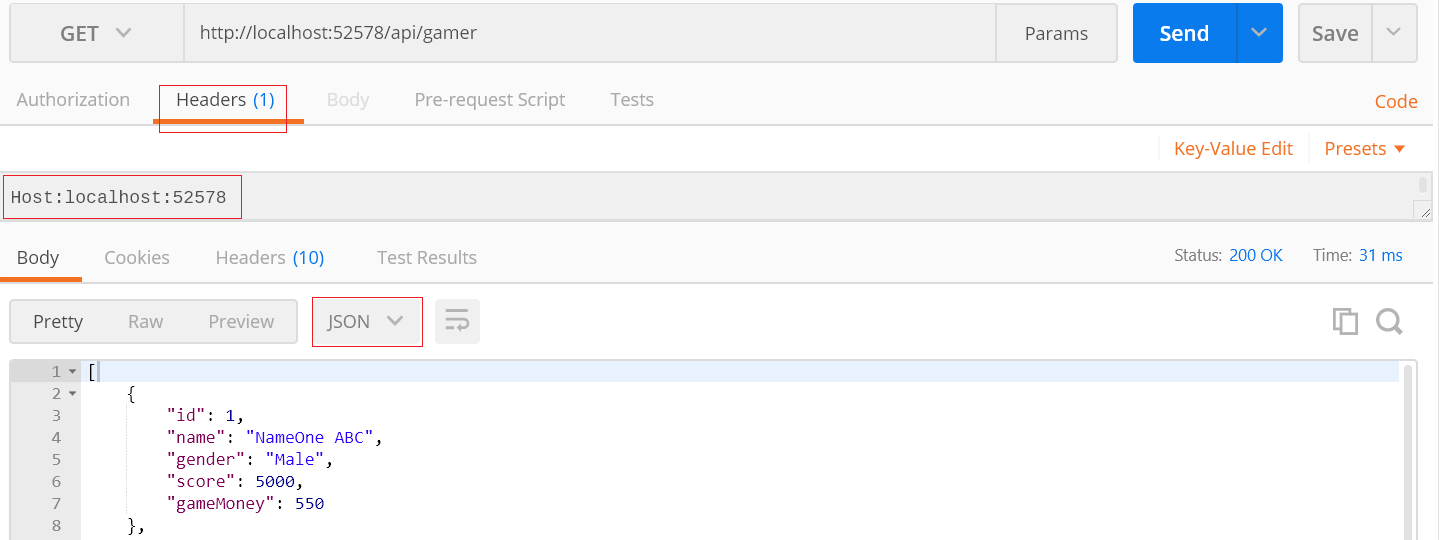


----------------------------------------------------------------------------------

E.g.4.

<http://localhost:52578/api/gamer>

**Host: localhost:52578**



7. Controllers/GamerController.cs - GET, POST

7.1. Controllers/GamerController.cs - GET, POST

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Net;

using System.Net.Http;

using System.Threading.Tasks;

using System.Web.Http;

using OnlineGame.Data;

namespace OnlineGame.WebApi.Controllers

{

    public class GamerController : ApiController

    {

        public async Task<IEnumerable<Gamer>> Get()

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                return await db.Gamers.ToListAsync();

            }

        }

        ////GET 1.

        ////It will return 200 OK even no matter the gamer is found or not.

        ////It should return 404 Not found when the gamer is not found.

        //public async Task<Gamer> Get(int id)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        return await db.Gamers.FirstOrDefaultAsync(g => g.Id == id);

        //    }

        //}

        //GET 2.

        //It will return 200 OK even no matter the gamer is found or not.

        //It should return 404 Not found when the gamer is not found.

        public async Task<HttpResponseMessage> Get(int id)

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                Gamer gamer = await db.Gamers.FirstOrDefaultAsync(g => g.Id == id);

                if (gamer != null)

                {

                    return Request.CreateResponse(HttpStatusCode.OK, gamer);

                }

                return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                    $"Gamer with Id {id} not found");

            }

        }

        ////POST 1.

        ////"void" will return 204 No Content.

        ////It should return status code 201 Item Created.

        ////If there is an exception, return status code 500 internal server error.

        //public void Post([FromBody] Gamer gamer)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        db.Gamers.Add(gamer);

        //        db.SaveChanges();

        //    }

        //}

        //POST 2.

        //"void" will return 204 No Content.

        //It should return status code 201 Item Created.

        //If there is an exception, return status code 500 internal server error.

        //[FromBody] means the gamer data comes from the client request body.

        public HttpResponseMessage Post([FromBody] Gamer gamer)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    db.Gamers.Add(gamer);

                    db.SaveChanges();

                    //Create a HttpResponseMessage with status code 201 Item Created.

                    //Pass the gamer into 2nd parameter as the created value.

                    HttpResponseMessage message =

                        Request.CreateResponse(HttpStatusCode.Created, gamer);

                    //The Headers.Location should know the URI of the created item.

                    message.Headers.Location = new Uri(Request.RequestUri +

                        gamer.Id.ToString());

                    return message;

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

    }

}

7.2. Controllers/GamerController.cs - GET

7.2.1. Controllers/GamerController.cs - GET

//GET 1.

//It will return 200 OK even no matter the gamer is found or not.

//It should return 404 Not found when the gamer is not found.

public async Task<Gamer> Get(int id)

{

    using (OnlineGameContext db = new OnlineGameContext())

    {

        return await db.Gamers.FirstOrDefaultAsync(g => g.Id == id);

    }

}

A.

GET

<http://localhost:52578/api/gamer/1>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

B.1.

Accept: application/json

means we request JSON format response.

-->

It will return 200 OK even no matter the gamer is found or not.

E.g. 1.

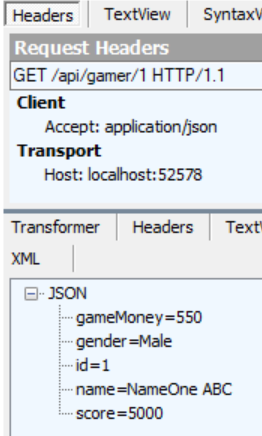
The Gamer is found by the id.



-->



-->



---------------------------------------------------------------------------

A.

GET

<http://localhost:52578/api/gamer/100>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

B.1.

Accept: application/json

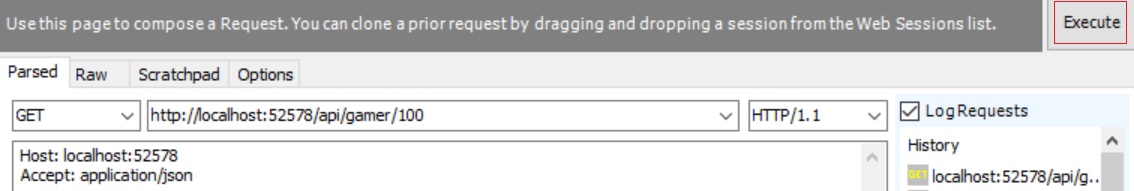
means we request JSON format response.

-->

It will return 200 OK even no matter the gamer is found or not.

E.g. 2.

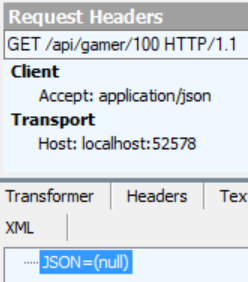
The Gamer is NOT found by the id.



-->



-->



7.2.2. Controllers/GamerController.cs - GET, Better version

//GET 2.

//It will return 200 OK even no matter the gamer is found or not.

//It should return 404 Not found when the gamer is not found.

public async Task<HttpResponseMessage> Get(int id)

{

    using (OnlineGameContext db = new OnlineGameContext())

    {

        Gamer gamer = await db.Gamers.FirstOrDefaultAsync(g => g.Id == id);

        if (gamer != null)

        {

            return Request.CreateResponse(HttpStatusCode.OK, gamer);

        }

        return Request.CreateErrorResponse(HttpStatusCode.NotFound,

            $"Gamer with Id {id} not found");

    }

}

A.

GET

<http://localhost:52578/api/gamer/1>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

B.1.

Accept: application/json

means we request JSON format response.

-->

It will return 200 OK even no matter the gamer is found or not.

E.g. 1.

The Gamer is found by the id.

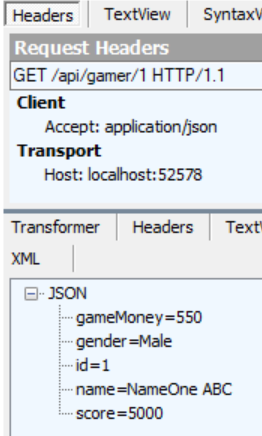
Graphical user interface, text, application

Description automatically generated

-->



-->



-------------------------------------------------------------

A.

GET

<http://localhost:52578/api/gamer/100>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

B.1.

Accept: application/json

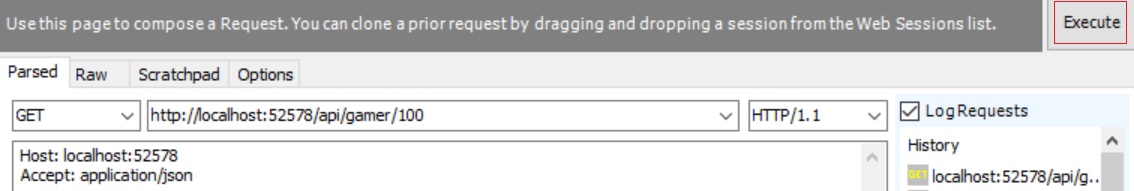
means we request JSON format response.

-->

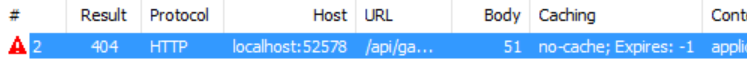
It will return 200 OK even no matter the gamer is found or not.

E.g. 2.

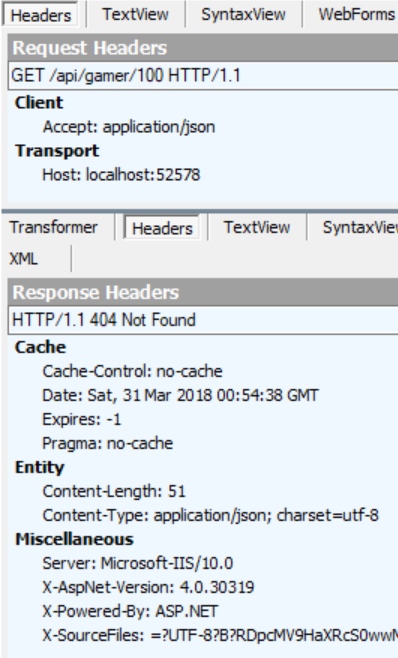
The Gamer is NOT found by the id.



-->



-->



7.3. Controllers/GamerController.cs - POST

7.3.1. Controllers/GamerController.cs - POST

//1.

//"void" will return 204 No Content.

//It should return status code 201 Item Created.

//If there is any exception, return status code 500 internal server error.

public void Post([FromBody] Gamer gamer)

{

    using (OnlineGameContext db = new OnlineGameContext())

    {

        db.Gamers.Add(gamer);

        db.SaveChanges();

    }

}

A.

Post

<http://localhost:52578/api/gamer>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

**Content-Type: application/json**

**Content-Length: 80**

B.1.

Accept: application/json

means we request JSON format response.

B.2.

Content-Type: application/json

Content-Length: 80

The client will post a data to the server, the data format is JSON and length is 80.

C.

Request Body

**{**

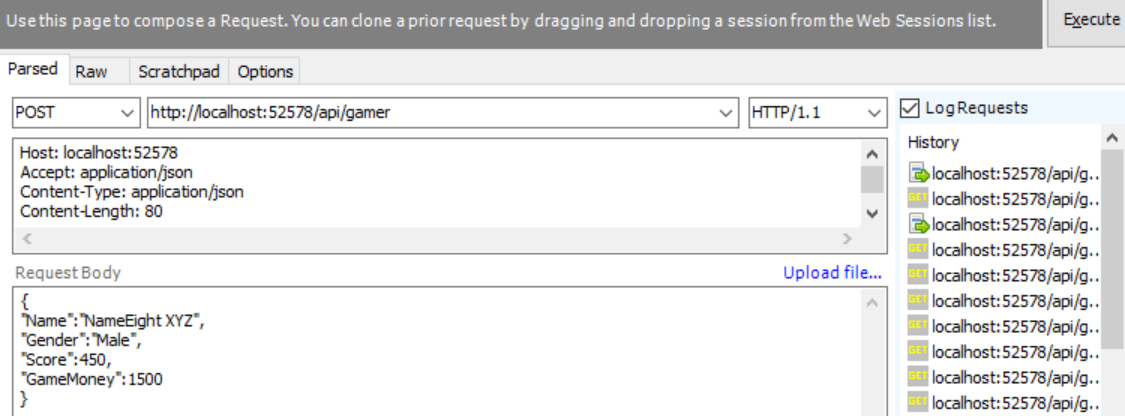
**"Name":"NameEight XYZ",**

**"Gender":"Male",**

**"Score":450,**

**"GameMoney":1500**

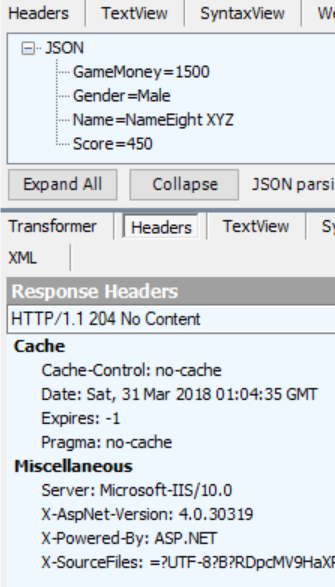
**}**



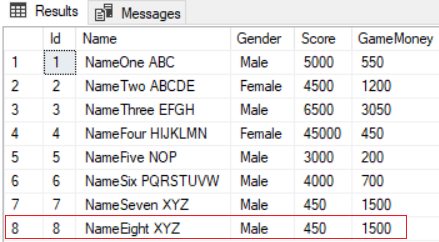
-->



-->



-->



7.3.2. Controllers/GamerController.cs - POST, Better version

//2.

//"void" will return 204 No Content.

//It should return status code 201 Item Created.

//If there is any exception, return status code 500 internal server error.

public HttpResponseMessage Post([FromBody] Gamer gamer)

{

    try

    {

        using (OnlineGameContext db = new OnlineGameContext())

        {

            db.Gamers.Add(gamer);

            db.SaveChanges();

            //Create a HttpResponseMessage with status code 201 Item Created.

            //Pass the gamer into 2nd parameter as the created value.

            HttpResponseMessage message =

                Request.CreateResponse(HttpStatusCode.Created, gamer);

            //The Headers.Location should know the URI of the created item.

            message.Headers.Location = new Uri(Request.RequestUri +

                gamer.Id.ToString());

            return message;

        }

    }

    catch (Exception ex)

    {

        return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

    }

}

A.

Post

<http://localhost:52578/api/gamer>

B.

Request Header

**Host: localhost:52578**

**Accept: application/json**

**Content-Type: application/json**

**Content-Length: 80**

B.1.

Accept: application/json

means we request JSON format response.

B.2.

Content-Type: application/json

Content-Length: 80

The client will post a data to server, the data format is JSON and length is 80.

C.

Request Body

**{**

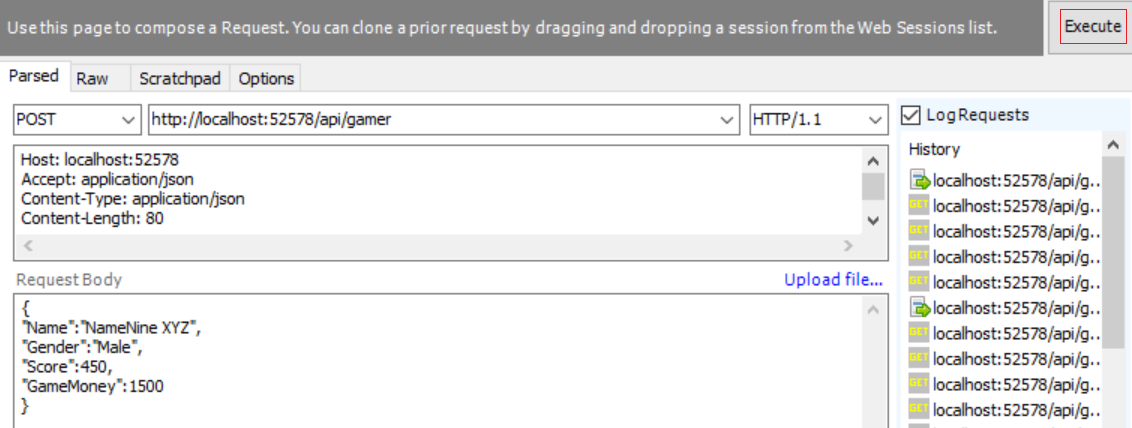
**"Name":"NameNine XYZ",**

**"Gender":"Male",**

**"Score":450,**

**"GameMoney":1500**

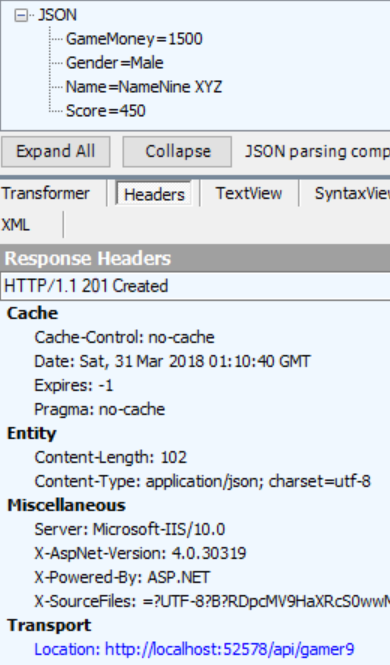
**}**



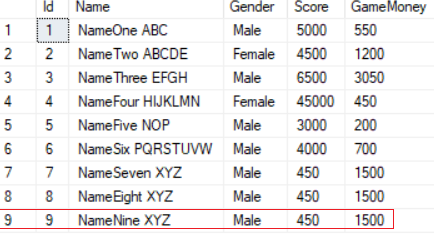
-->



-->



-->



8. Controllers/GamerController.cs - Delete

8.1. Controllers/GamerController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using OnlineGame.Data;

namespace OnlineGame.WebApi.Controllers

{

    public class GamerController : ApiController

    {

        public IEnumerable<Gamer> Get()

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                return db.Gamers.ToList();

            }

        }

        ////GET 1.

        ////It will return 200 OK even no matter the gamer is found or not.

        ////It should return 404 Not found when the gamer is not found.

        //public Gamer Get(int id)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        return db.Gamers.FirstOrDefault(g => g.Id == id);

        //    }

        //}

        //GET 2.

        //It will return 200 OK even no matter the gamer is found or not.

        //It should return 404 Not found when the gamer is not found.

        public HttpResponseMessage Get(int id)

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                Gamer gamer = db.Gamers.FirstOrDefault(g => g.Id == id);

                if (gamer != null)

                {

                    return Request.CreateResponse(HttpStatusCode.OK, gamer);

                }

                return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                    $"Gamer with Id {id} not found");

            }

        }

        ////POST 1.

        ////"void" will return 204 No Content.

        ////It should return status code 201 Item Created.

        ////If there is any exception, return status code 500 internal server error.

        //public void Post([FromBody] Gamer gamer)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        db.Gamers.Add(gamer);

        //        db.SaveChanges();

        //    }

        //}

        //POST 2.

        //"void" will return 204 No Content.

        //It should return status code 201 Item Created.

        //If there is any exception, return status code 500 internal server error.

        public HttpResponseMessage Post([FromBody] Gamer gamer)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    db.Gamers.Add(gamer);

                    db.SaveChanges();

                    //Create a HttpResponseMessage with status code 201 Item Created.

                    //Pass the gamer into 2nd parameter as the created value.

                    HttpResponseMessage message =

                        Request.CreateResponse(HttpStatusCode.Created, gamer);

                    //The Headers.Location should know the URI of the created item.

                    message.Headers.Location = new Uri(Request.RequestUri +

                        gamer.Id.ToString());

                    return message;

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

        ////DELETE 1.

        ////"void" will return 204 No Content.

        ////It should return status code 200 OK when deleting the Item successfully.

        ////If the gamer id is not found, then return 404 Not found.

        ////If there is any exception, return status code 500 internal server error.

        //public void Delete(int id)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        Gamer gamer = db.Gamers.FirstOrDefault(g => g.Id == id);

        //        if (gamer != null) db.Gamers.Remove(gamer);

        //        db.SaveChanges();

        //    }

        //}

        //DELETE 2.

        //"void" will return 204 No Content.

        //It should return status code 200 OK when deleting the Item successfully.

        //If the gamer id is not found, then return 404 Not found.

        //If there is any exception, return status code 500 internal server error.

        public HttpResponseMessage Delete(int id)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    Gamer entity = db.Gamers.FirstOrDefault(g => g.Id == id);

                    if (entity == null)

                    {

                        return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                            $"Gamer with Id = {id} not found to delete");

                    }

                    db.Gamers.Remove(entity);

                    db.SaveChanges();

                    return Request.CreateResponse(HttpStatusCode.OK);

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

    }

}

8.2. Controllers/GamerController.cs -Delete

8.2.1. Controllers/GamerController.cs -Delete

//3.

//"void" will return 204 No Content.

//It should return status code 200 OK when deleting the Item successfully.

//If the gamer id is not found, then return 404 Not found.

//If there is any exception, return status code 500 internal server error.

public void Delete(int id)

{

    using (OnlineGameContext db = new OnlineGameContext())

    {

        Gamer gamer = db.Gamers.FirstOrDefault(g => g.Id == id);

        if (gamer != null) db.Gamers.Remove(gamer);

        db.SaveChanges();

    }

}

A.

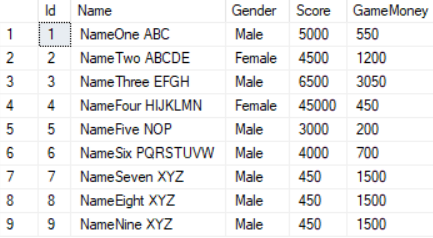
Delete

<http://localhost:52578/api/gamer/9>

B.

Request Header

**Host: localhost:52578**



-->

Background pattern

Description automatically generated with medium confidence

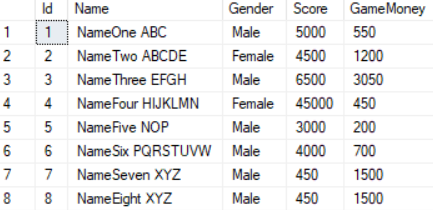
-->



-->



-->



8.2.2. Controllers/GamerController.cs - GET, Better version

//DELETE 2.

//"void" will return 204 No Content.

//It should return status code 200 OK when deleting the Item successfully.

//If the gamer id is not found, then return 404 Not found.

//If there is any exception, return status code 500 internal server error.

public HttpResponseMessage Delete(int id)

{

    try

    {

        using (OnlineGameContext db = new OnlineGameContext())

        {

            Gamer entity = db.Gamers.FirstOrDefault(g => g.Id == id);

            if (entity == null)

            {

                return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                    $"Gamer with Id = {id} not found to delete");

            }

            db.Gamers.Remove(entity);

            db.SaveChanges();

            return Request.CreateResponse(HttpStatusCode.OK);

        }

    }

    catch (Exception ex)

    {

        return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

    }

}

A.

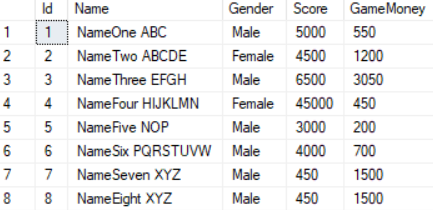
Post

<http://localhost:52578/api/gamer/9>

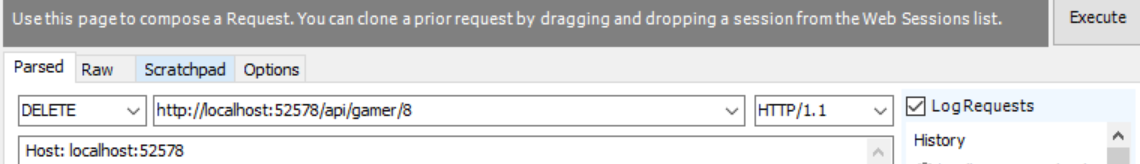
B.

Request Header

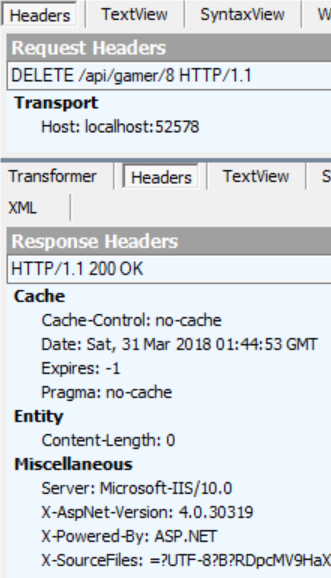
**Host: localhost:52578**



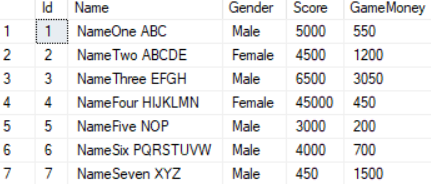
-->



-->



-->



-----------------------------------------------------

A.

Post

<http://localhost:52578/api/gamer/100>

B.

Request Header

**Host: localhost:52578**

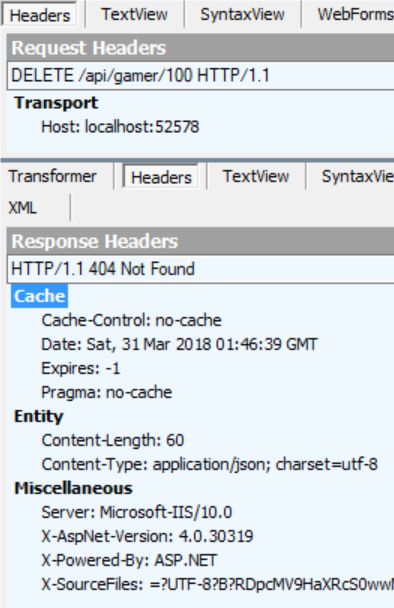
Text

Description automatically generated with medium confidence

-->



-->



9. Controllers/GamerController.cs - Put

9.1. Controllers/GamerController.cs - Put

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using  System.Net.Http;

using System.Web.Http;

using OnlineGame.Data;

namespace OnlineGame.WebApi.Controllers

{

    public class GamerController : ApiController

    {

        public IEnumerable<Gamer> Get()

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                return db.Gamers.ToList();

            }

        }

        ////GET 1.

        ////It will return 200 OK even no matter the gamer is found or not.

        ////It should return 404 Not found when the gamer is not found.

        //public Gamer Get(int id)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        return db.Gamers.FirstOrDefault(g => g.Id == id);

        //    }

        //}

        //GET 2.

        //It will return 200 OK even no matter the gamer is found or not.

        //It should return 404 Not found when the gamer is not found.

        public HttpResponseMessage Get(int id)

        {

            using (OnlineGameContext db = new OnlineGameContext())

            {

                Gamer gamer = db.Gamers.FirstOrDefault(g => g.Id == id);

                if (gamer != null)

                {

                    return Request.CreateResponse(HttpStatusCode.OK, gamer);

                }

                return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                    $"Gamer with Id {id} not found");

            }

        }

        ////POST 1.

        ////"void" will return 204 No Content.

        ////It should return status code 201 Item Created.

        ////If there is an exception, return status code 500 internal server error.

        //public void Post([FromBody] Gamer gamer)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        db.Gamers.Add(gamer);

        //        db.SaveChanges();

        //    }

        //}

        //POST 2.

        //"void" will return 204 No Content.

        //It should return status code 201 Item Created.

        //If there is an exception, return status code 500 internal server error.

        //[FromBody] means the gamer data comes from the client request body.

        public HttpResponseMessage Post([FromBody] Gamer gamer)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    db.Gamers.Add(gamer);

                    db.SaveChanges();

                    //Create a HttpResponseMessage with status code 201 Item Created.

                    //Pass the gamer into 2nd parameter as the created value.

                    HttpResponseMessage message =

                        Request.CreateResponse(HttpStatusCode.Created, gamer);

                    //The Headers.Location should know the URI of the created item.

                    message.Headers.Location = new Uri(Request.RequestUri +

                        gamer.Id.ToString());

                    return message;

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

        ////DELETE 1.

        ////"void" will return 204 No Content.

        ////It should return status code 200 OK when deleting the Item successfully.

        ////If the gamer id is not found, then return 404 Not found.

        ////If there is an exception, return status code 500 internal server error.

        //public void Delete(int id)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        Gamer gamer = db.Gamers.FirstOrDefault(g => g.Id == id);

        //        if (gamer != null) db.Gamers.Remove(gamer);

        //        db.SaveChanges();

        //    }

        //}

        //DELETE 2.

        //"void" will return 204 No Content.

        //It should return status code 200 OK when deleting the Item successfully.

        //If the gamer id is not found, then return 404 Not found.

        //If there is an exception, return status code 500 internal server error.

        public HttpResponseMessage Delete(int id)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    Gamer entity = db.Gamers.FirstOrDefault(g => g.Id == id);

                    if (entity == null)

                    {

                        return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                            $"Gamer with Id = {id} not found to delete");

                    }

                    db.Gamers.Remove(entity);

                    db.SaveChanges();

                    return Request.CreateResponse(HttpStatusCode.OK);

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

        ////PUT 1.

        ////"void" will return 204 No Content.

        ////It should return status code 200 OK when delete the Item successfully.

        ////If the gamer by id is not found, then return 404 Not found.

        ////If there is an exception, return status code 500 internal server error.

        ////[FromBody] means the gamer data comes from the client request body.

        //public void Put(int id, [FromBody]Gamer gamer)

        //{

        //    using (OnlineGameContext db = new OnlineGameContext())

        //    {

        //        Gamer currentGamer = db.Gamers.FirstOrDefault(g => g.Id == id);

        //        if (currentGamer != null)

        //        {

        //            currentGamer.Name = gamer.Name;

        //            currentGamer.Gender = gamer.Gender;

        //            currentGamer.Score = gamer.Score;

        //            currentGamer.GameMoney = gamer.GameMoney;

        //        }

        //        db.SaveChanges();

        //    }

        //}

        //PUT 2.

        //"void" will return 204 No Content.

        //It should return status code 200 OK when delete the Item successfully.

        //If the gamer by id is not found, then return 404 Not found.

        //If there is an exception, return status code 500 internal server error.

        //[FromBody] means the gamer data comes from the client request body.

        public HttpResponseMessage Put(int id, [FromBody]Gamer gamer)

        {

            try

            {

                using (OnlineGameContext db = new OnlineGameContext())

                {

                    Gamer currentGamer = db.Gamers.FirstOrDefault(g => g.Id == id);

                    if (currentGamer == null)

                    {

                        return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                            $"Gamer with Id {id} not found to update");

                    }

                    currentGamer.Name = gamer.Name;

                    currentGamer.Gender = gamer.Gender;

                    currentGamer.Score = gamer.Score;

                    currentGamer.GameMoney = gamer.GameMoney;

                    db.SaveChanges();

                    return Request.CreateResponse(HttpStatusCode.OK, currentGamer);

                }

            }

            catch (Exception ex)

            {

                return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

            }

        }

    }

}

9.2. Controllers/GamerController.cs - Put

9.2.1. Controllers/GamerController.cs - Put

//PUT 1.

//"void" will return 204 No Content.

//It should return status code 200 OK when deleting the Item successfully.

//If the gamer id is not found, then return 404 Not found.

//If there is any exception, return status code 500 internal server error.

//[FromBody] means the gamer data comes from the client request body.

public void Put(int id, [FromBody]Gamer gamer)

{

    using (OnlineGameContext db = new OnlineGameContext())

    {

        Gamer currentGamer = db.Gamers.FirstOrDefault(g => g.Id == id);

        if (currentGamer != null)

        {

            currentGamer.Name = gamer.Name;

            currentGamer.Gender = gamer.Gender;

            currentGamer.Score = gamer.Score;

            currentGamer.GameMoney = gamer.GameMoney;

        }

        db.SaveChanges();

    }

}

A.

PUT

<http://localhost:52578/api/gamer/1>

B.

Request Header

**Host: localhost:52578**

**Content-Type: application/json**

B.1.

Content-Type: application/json

Content-Length: 80

The client will post a data to server, the data format is JSON and length is 80.

C.

Request Body

**{**

**"Name":"NameOne ABC222",**

**"Gender":"Male",**

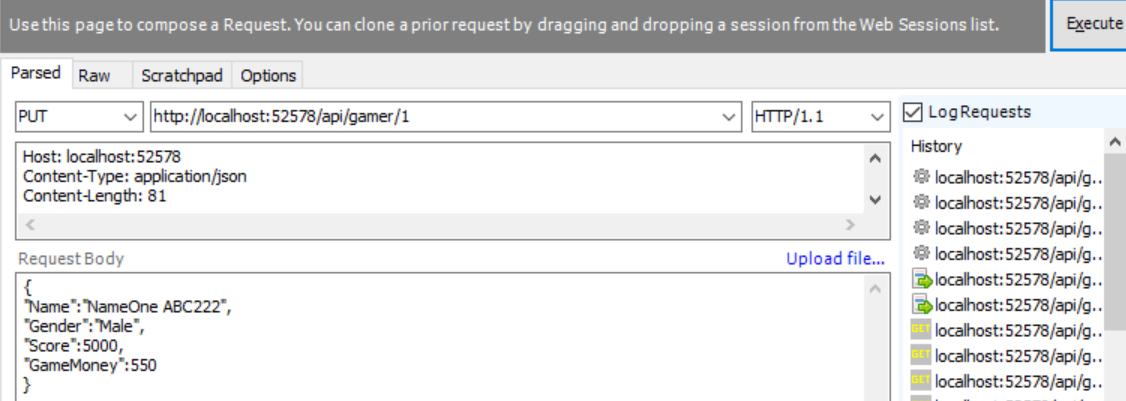
**"Score":5000,**

**"GameMoney":550**

**}**



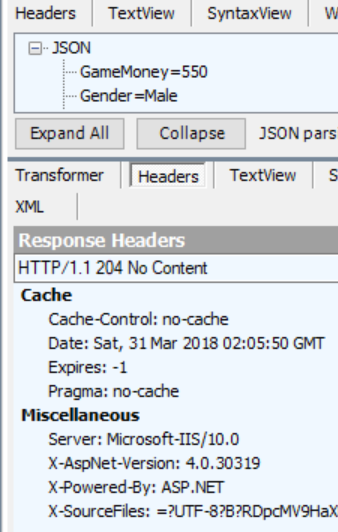
-->



-->



-->



-->



9.2.2. Controllers/GamerController.cs - PUT, Better version

//PUT 2.

//"void" will return 204 No Content.

//It should return status code 200 OK when deleting the Item successfully.

//If the gamer id is not found, then return 404 Not found.

//If there is any exception, return status code 500 internal server error.

//[FromBody] means the gamer data comes from the client request body.

public HttpResponseMessage Put(int id, [FromBody]Gamer gamer)

{

    try

    {

        using (OnlineGameContext db = new OnlineGameContext())

        {

            Gamer currentGamer = db.Gamers.FirstOrDefault(g => g.Id == id);

            if (currentGamer == null)

            {

                return Request.CreateErrorResponse(HttpStatusCode.NotFound,

                    $"Gamer with Id {id} not found to update");

            }

            currentGamer.Name = gamer.Name;

            currentGamer.Gender = gamer.Gender;

            currentGamer.Score = gamer.Score;

            currentGamer.GameMoney = gamer.GameMoney;

            db.SaveChanges();

            return Request.CreateResponse(HttpStatusCode.OK, currentGamer);

        }

    }

    catch (Exception ex)

    {

        return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex);

    }

}

A.

PUT

<http://localhost:52578/api/gamer/1>

B.

Request Header

**Host: localhost:52578**

**Content-Type: application/json**

B.1.

Content-Type: application/json

Content-Length: 80

The client will post a data to server, the data format is JSON and length is 80.

C.

Request Body

**{**

**"Name":"NameOne ABC",**

**"Gender":"Male",**

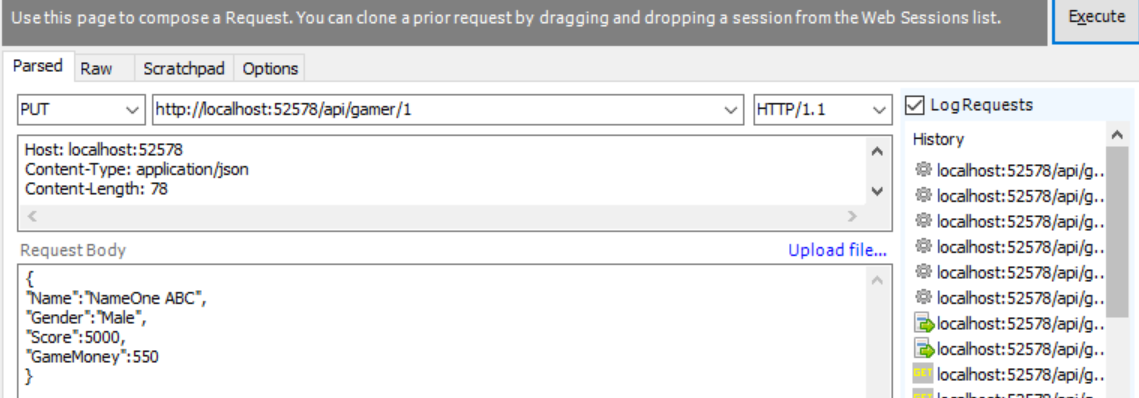
**"Score":5000,**

**"GameMoney":550**

**}**



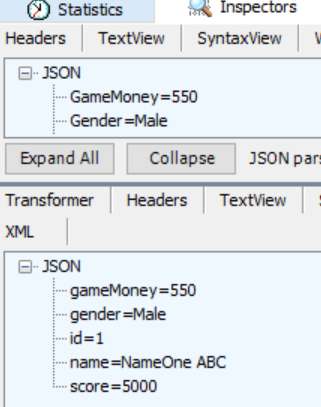
-->



-->



-->



-->



-----------------------------------------------------------------------

A.

PUT

<http://localhost:52578/api/gamer/100>

B.

Request Header

**Host: localhost:52578**

**Content-Type: application/json**

B.1.

Content-Type: application/json

Content-Length: 80

The client will post a data to server, the data format is JSON and length is 80.

C.

Request Body

**{**

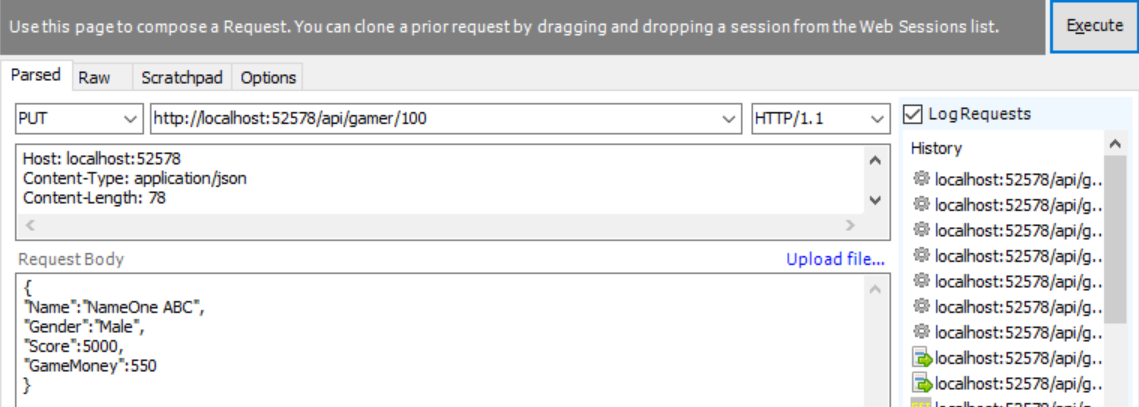
**"Name":"NameOne ABC222",**

**"Gender":"Male",**

**"Score":5000,**

**"GameMoney":550**

**}**



-->



-->

Graphical user interface, text, application, email

Description automatically generated