COMPUTING SUBJECT:	Restful ASP.Net Core-services
TYPE:	Assignment
IDENTIFICATION:	RestService#1
COPYRIGHT:	Peter Levinsky & Michael Claudius
LEVEL:	Medium
TIME CONSUMPTION:	1½-2 hours
EXTENT:	120 lines
OBJECTIVE:	Restful services based on ASP.Net Core
PRECONDITIONS:	Rest service theory. Http-concepts Computer Networks Ch. 2.2

COMMANDS:

IDENTIFICATION: RestService#1 / PELE with kindly respect and inspiration from MICL

Overall Purpose

The overall purpose for the group of 'RestService' assignments is to be able to provide and consume restful ASP.Net Core web services, to prepare the 'RestService' to be published in Azure, including testing the service and finally to setup the 'RestService' to be consumed from a browser (e.g. using Typescript) i.e. support CORS.

The whole group of assignments consist of 7 steps:

1. A simple **REST** Service with **CRUD**. (this assignment)

- 2. More advanced and complex URI's.
- 3. Adding help-pages to the REST Service (Swagger)
- 4. Testing a REST Service and publish in Azure.
- 5. Consuming a REST service from a C# Console application.
- 6. Adding Support for CORS to the REST Service
- 7. A REST Service using a database

Background Material:

The HTTP protocol: See Computer Network chap 2 pp. 111-136

Note of REST (Peter Levinsky): See NetHttpNote.pdf

Oswago Universitet: RESTful Service Best Practices: Recommendations for Creating Web Services: See <u>http://cs.oswego.edu/~alex/teaching/csc435/RESTful.pdf</u>

Usefull tools (Postman & Fiddler): See <u>Tools.htm</u> (tool #3 & tool #4)

This Assignment: RestService#1

Purpose

The purpose of this assignment is to be able to provide a simple Restful ASP.Net Core web services with CRUD.

Mission

You are to make a restful web services based on the ASP.Net Core services by setting up a service (provider). The service supports simple CRUD by implementing the classic GET, POST, PUT and DELETE requests. This we shall do in 2 major steps:

- 1. Investigate a simple default REST Service
 - a. Create the REST Service
 - b. Run the REST Service
 - c. Try the REST Service (Postman or Fiddler)
 - d. Explore the REST Service
- 2. Create your own REST simple Service
 - a. Create a model class
 - b. Create a controller
 - c. Run and try your REST Service (Postman or Fiddler)

Domain description

First you shall just utilize the simple auto generated web service defined by ValuesContolller. Next you are to create your own REST Service (Provider), which can manipulate (make CRUD) on model elements 'Item's.

The Visual Studio 2019 version screen dumps are placed at the end. In the exercise text you see the 2017 version.

Assignment 1.a: Create The RESTful ASP.Net Core-service provider

You are to make a Rest Service provider 'RestItemService'.

Start Visual Studio:File -> New -> Project.

Choose: Web -> ASP.NET Core Web Application (not .Net Framework).

Browse to a convenient location and give the name 'RestItemService'.

New Project						? ×
Recent	1 5	ort by: Default	- # E		Search (Ctrl+E)	ρ
Installed	1	Console App (N	ET Core)	Visual Cit	Type: Visual C#	
 Visual CV Get Started Windows Ur 		Class Library (NE		Visual C#	Project templates for creating Core applications for Window macOS using .NET Core or .N	ws. Linux and VET
Windows De		MSTest Test Proj	ect (NET Core)	Visual Cil	Framework: Create Razor Pag Web API, and Single Page (SF Applications.	
NET Core		NUnit Test Projec	ct (.NET Core)	Visual C#		
NET Standa Cloud		×Unit Test Projec	t (NET Core)	Visual C/		
Extensibility Test	0	ASP.NET Core W	eb Application	Visual C#	`	
WCF						
Visual Basic						
Visual C++						
Visual FV SQL Server						
Azure Data Lake						
Parale Data Lake Parale Data Lake						
Azure Stream A	nalytics 🚽					
	you are looking for? Studio Installer					
Name	RestitemService					
Location	C\Users\EASJ\source\re	epos			Browse_	
Solution name	RestitemService				Create girectory for solution Create new Git repository	
					OK	Cencel

Click OK. (For 2019 version see picture at the end)

v ASP.NET Core W	ASP.NET Core 2.1 Learn more	7
Empty		te for creating an ASP.NET Core an example Controller for a RESTfu is template can also be used for VC Views and Controllers.
Angular		Microsoft IDK 2.1.505
Enable Docker	Support (Requires Docker for Windows)	No Authentication
/		OK Cancel

Now a project template with options are given.

(For 2019 version see picture at the end)

Choose the API.

DON'T tick Docker support. But Untick the HTTPS

Now you have to wait a while...

Then we are ready to investigate the created project.

Assignment 1.b: Run the REST Service

Execute the web application (click the green arrow as usually) and it will open a local Browser.



As you can see the application is running on port 44343 on my computer (it will be different on yours).

Try to give the url:

http://localhost:44343/api/values/5 and http://localhost:44343/api/values/17

What do you get? (Remember to use your own port number)

<u>Assignment 1.c: Fiddler/Postman</u> Have the Postman or Fiddler installed – see <u>tools.htm</u>

Try to invoke the methods from Fiddler/Postman using the same url's as before. If you are not familiar with fiddler (or postman) for details see 2.c below.

Assignment 1.d: Exploring the REST Service i.e. the whole set up Some questions arise:

Why does the project start with api/values? Where are the Http-methods defined? Why is it port 44343 (or more correct your port-number)? How can it be executed in console mode on a different port?

To find the answers take a look in Solution Explorer:

Controller -> ValuesController Propeties -> LaunchSettings.json

Modify the Get method body to

```
[HttpGet("{id}")]
public ActionResult<int> Get(int id)
{return id;}
```

Execute again. Any difference? What happens?

Finally, at the 'green arrow' change 'IISExpress' to 'RestItemService' and run the application again.

What could be the purpose of doing this?

This ends the investigation of the auto generated service. In the following assignments you shall create your own service.

Assignment 2.a: Create a model class Item

We need a model class 'Item' before you can make your own REST service.

Therefore to the Solution (right click -> add new project) create a new project (Class Library .Net Core) and named it '**ModelLib**'. Like this:

NOU New Project					? ×
Record	1	Sort by	Default • 📰 🖭		Search (Chi+D)
# instalfed		50	Console App-(INET Core)	Visual CP	Type: Visual C#
 Yinual C# Get Started 	A CONTRACTOR OF A CONTRACTOR OFTA CONTRACTOR O	1	Gass (Joney (NET Com)	Moster	A project for creating a class library that targets. NET Core.
Windows D Windows D		-	MSTeat Test Project (NET Core)	Visual C4	
P Web NET Core	•	-	NUnit Test Project (.NET.Cont)	Visual Ca	
NET Shinds Cloud	arti	-	eUnit Test Project (NET Close)	Wittel CA	
Extensibility Test	·	Ð	ASPINET Core Web Application	Visual C#	
WCF F Vinual Basic					
Visual C++					
 Vitual ## SQL Server 					
Azurw Data Lok	- C				
 JavaScript Ature Stream J 	knałytice				
Critice					
	ryou are looking for? I Studio Installar				
Name	Castilibrary1			11	
			withemService		Browns

(For 2019 version see picture at the end)

In this new project create a folder '**model**' (right click project, choose Add -> Folder) and in this folder add a class, "**Item**", with the properties:

- int Id;
- string Name;
- string Quality;
- double Quantity;

Remember to two constructors:

- Data() { ... }
 //empty constructor needed for JSON transfer. Serializable objects.
- Item(int id, string name, string quality, double quantity) { ... } Intializing all the data fields

And make a ToString – method

Before you are finished, first make the class public and second remember to compile the library i.e. build the project. (If you are really nice delete the 'Class1' class in the library)

Now you are ready to the next step.

Assignment 2.b: Creating a controller (making the REST API operation)

In the solution in the controller folder, add (i.e. Right click) a new controller named '**ItemsController**'. Choose 'API Controller **with read/write actions**'.

Inde Starbald		
# instante		
F Conner Seeling	Mot Controller - Impry Improvement Improvement Improvement Improvement Improvement Improvement Improvement Improvement Improvement Improvement	APC controller with reading the states by the second vision and APC control as well RECT actions to means and upping bidding and lot antition. It ApD control are With Action Confliction
	API Controller with residents unlines	
	Out have to go onlive and free may advecting exercises	
		Ald Cared

Now you have a template controller (the url is: <u>http://localhost:44343/api/Items</u> with your port number).

To let your controller using your model class (Item) from 2.b, you have to add a reference to refer to your library i.e. right-click at the Dependencies -> add reference choose from your solution 'ModelLib'.

Modify the five signatures in the 'ItemsController' to: (Modification marked with bold)

- public IEnumerable<**Item**> Get()
- public **Item** Get(int id)
- public void Post([FromBody] Item value)
- public void Put(int id, [FromBody] **Item** value)
- public void Delete(int id)

In this simple CRUD REST Service you are <u>not</u> using a database, but instead a static list of Item's as an instance field.

private static List<Item> items = new List<Item>()

You can fill in some default values e.g. like:

{

private static readonly List<Item> items = new List<Item>()

```
new Item(1,"Bread","Low",33),
new Item(2,"Bread","Middle",21),
new Item(3,"Beer","low",70.5),
new Item(4,"Soda","High",21.4),
new Item(5,"Milk","Low",55.8)
};
```

Now you can modify the body of your five controller method using this list.

- public IEnumerable<Item> Get() return items;
- public Item Get(int id) return items.Find(i => i.Id == id);
- public void Post([FromBody] Item value) items.Add(value);

```
    public void Put(int id, [FromBody] Item value)
Item item = Get(id);
if (item != null)
        {
            item.Id = value.Id;
            item.Name= value.Name;
            item.Quality = value.Quality;
            item.Quantity = value.Quantity;
        }
        }
```

 public void Delete(int id) Item item = Get(id); items.Remove(item);

Now you can compile (build) and run your REST Service.

Try from your bowser

http://localhost:44343/api/Items or http://localhost:44343/api/Items/3

Can you call post from your browser? What about put or delete?

Assignment 2.c: Run and try your REST Service (Postman or Fiddler)

Have Postman or Fiddler installed. (See previous exercises). To be able to try all methods in your REST Service you need tools like Postman or Fiddler.

Open e.g. Fiddler and in the file-tab untick capture traffic, so you do not get all traffic in and out of your computer. (Then you can clear window by under the X click remove all).

Click on the **composer tab** and write the URL to your REST Service e.g. <u>http://localhost:44343/api/Items</u> click execute. To see the result click on the latest result in the window to the left.

Now try with <u>http://localhost:44343/api/Items/3</u> again look at the result.

All this is performing the GET method, next step is to try the other methods:

POST:

- First pick the POST method
- Set the Content-Type: application/json (just below the 'Host'-header)
- Request body must hold the data as a Json-string e.g. {"id":30,"name":"Coke","quality":"Middle","quantity":13.6}

It should look like:

s] 4	Result 200	Protocol HTTP	Host localhost:50327	/api/items	Fiddler Orchestra Beta Grader Scriptore Statistics Inspectors	pt 📄 Log 🔲 Filte ∮ AutoResponder	ers 🔄 Timeline 🖉 Composer
5	404 200	HTTP	localhost:50327 localhost:50327	/api/items /api/items	Use this page to compose a Request. You can do dropping a session from the Web Sessions list.	one a prior request by dragging (Egecute
					User-Agent: Fiddler Host: localhost: 50327 Content-Engel: explication/pont Content-Length: 60 RequestBody {"Id": 30, "name": "Coke", "quality": "Middle", "quanti	Upload file ty*:13.6)	History
_		_		2	c		

- Execute the REST Call
- Lookup the result what do you see?
- Try to get/Items/30 what do you get?

PUT:

- First pick the PUT method
- Change the URL to/api/Items/30
- Set the Content-Type: application/json (just below the 'Host'-header)
- Request body must hold the data as a Json-string e.g. {"id":30,"name":"Coca Coke","quality":"Middle","quantity":13.6}

I.e. change the name

- Execute the REST call
- Try to get/Items/30 what do you get?

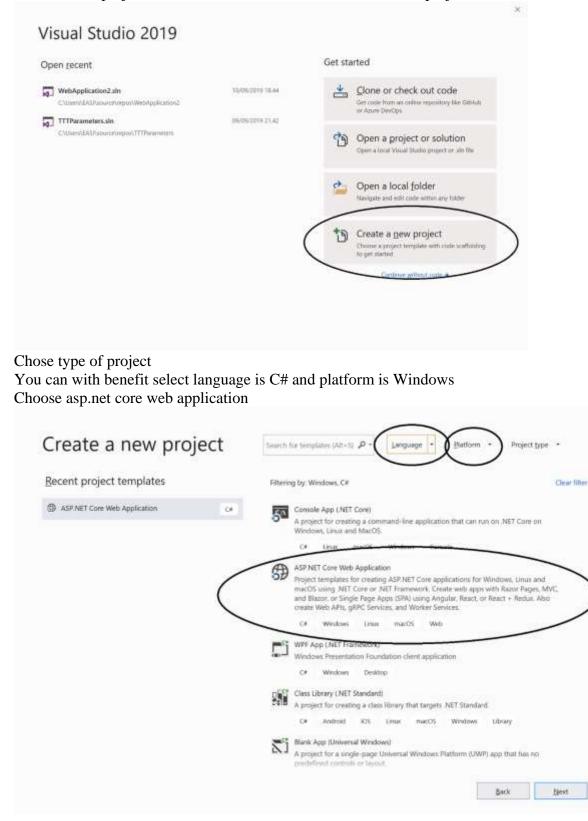
DELETE:

- First pick the PUT method
- Change the URL to/api/Items/30
- Execute the REST call
- Try to get/Items/30 what do you get?

That was all – You have nicely finished a simple crud REST-Service and you are ready to do more advanced REST services.

The Visual Studio 2019 version

To create the project, start Visual Studio and choose create new project



Give the project And solution a name (here the same)

SPINET Core Web Application C+ Wentows Linux	194005 (Allen		
roject game			
Iten2019RestService			
scation			
ChUserNEASI/source/vepas	•		
olution name 🕕			
Iten2019RestService	1		
Race solution and project in the same gliectory			

Configure type of web application NB untick HTTPS – and choose API

ASP NET Com 2.1		
npty	Auth	nentication
이 같은 것 같은 것 같은 것 같은 것 같은 것 같은 것 같이 있는 것 같이 같이 같이 같이 같이 같이 많이		athentication
	Chang	E.
	1	
	Adva	anced
eb Application	(🗆 0	onligure for HTTPS
project template for creating an ASP.NET Core application with example ASP.NET Core Razor Pages		able Docker Support
ntent,	(R	equires Docker Desktop)
eb Application (Model-View-Controller)	1 mile	
eb Application (Model-View-Controller) project template for creating an ASP.NET Core application with example ASP.NET Core MNC Views and introllers. This template can also be used for IESTful HTTP services.	1 mile	
project template for creating an ASP.NET Core application with example ASP.NET Core MNC Views and	Linne	
	empty project template for creating an ASP NET Core application. This template does not have any nempty project template for creating an ASP NET Core application with an example Controller for a RESTINI HTTP vice. This template for creating an ASP NET Core application with an example Controller for a RESTINI HTTP vice. This template can also be traced for ASP NET Core MVC. Views and Controller.	empty project template for creating an ASP.NET Core application. This template does not have any neutrinin a. No Au Chursh and the second seco

Create a Library To filter the project write .net core, whereby only these project types are available Chose class library (.net core)

Add a new projec		net core ו Unguage • Platform • Project type •
Recent project templates		Filtering by: Windows, C# Clear filte
ASP NET Core Web Application	ø	Console App (NET Core) A project for creating a command-line application that can run on NET Core on Windows, Linux and MacOS.
		C# Linux macOS Windows Console
		ASP.NET Core Web Application Project templates for creating ASP.NET Core applications for Windows, Linux and macOS using .NET Core or .NET Framework, Create web apps with Rizzor Pages, MVC, and Blazor, or Single Page App (SPA) using Angular, React, or React + Redux. Also create Web APIs, gRPC Services, and Worker Services.
		C# Windows Linux marOS Web
	6	Class Library (NET Core) A project for creating a class library that targets. RET Core.
		C4 Windows Dinus macOS Library
		MSTest Test Project (AFE Care) A project that contains MSTest unit tests that can run on: NET Core on Windows, Smar and MacOS
		G# Lisux macO5 Windows liest
		A project that containe admittent tests that can non any NET Cont in Windows, Unite