

# .NET CORE REST API



.NET  
Core

Presented By



# ภาพรวม

- การพัฒนาซอฟต์แวร์ในปัจจุบัน
- Web Service คืออะไร
- .Net Core คืออะไร
- Install Tools for Dev
- Workshop

# การพัฒนาซอฟต์แวร์ในปัจจุบัน



# Web Service คืออะไร

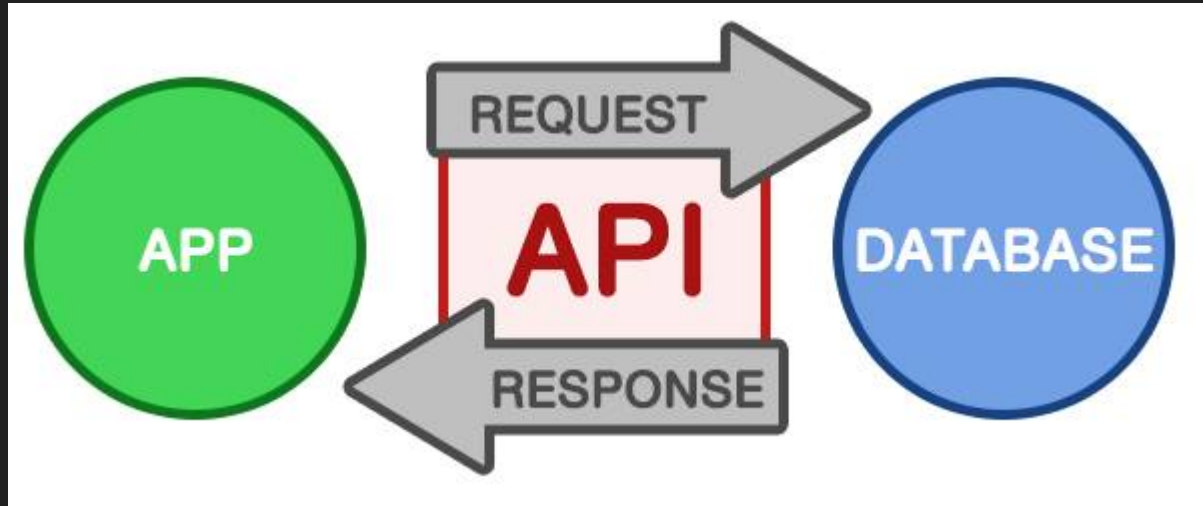
คือการแลกเปลี่ยนข้อมูลกันที่อยู่ในรูปแบบของ XML หรือ JSON สามารถแบ่งออกเป็น 2  
รูป SOAP กับ REST API



# REST Api คืออะไร

คือการแลกเปลี่ยนข้อมูลกันที่อยู่ในหลายรูปแบบที่นิยมที่สุดคือ JSON โดยใช้ http protocol ผ่าน http method ซึ่งประกอบไปด้วย

- GET
- POST
- PUT
- DELETE



# Http Status Code

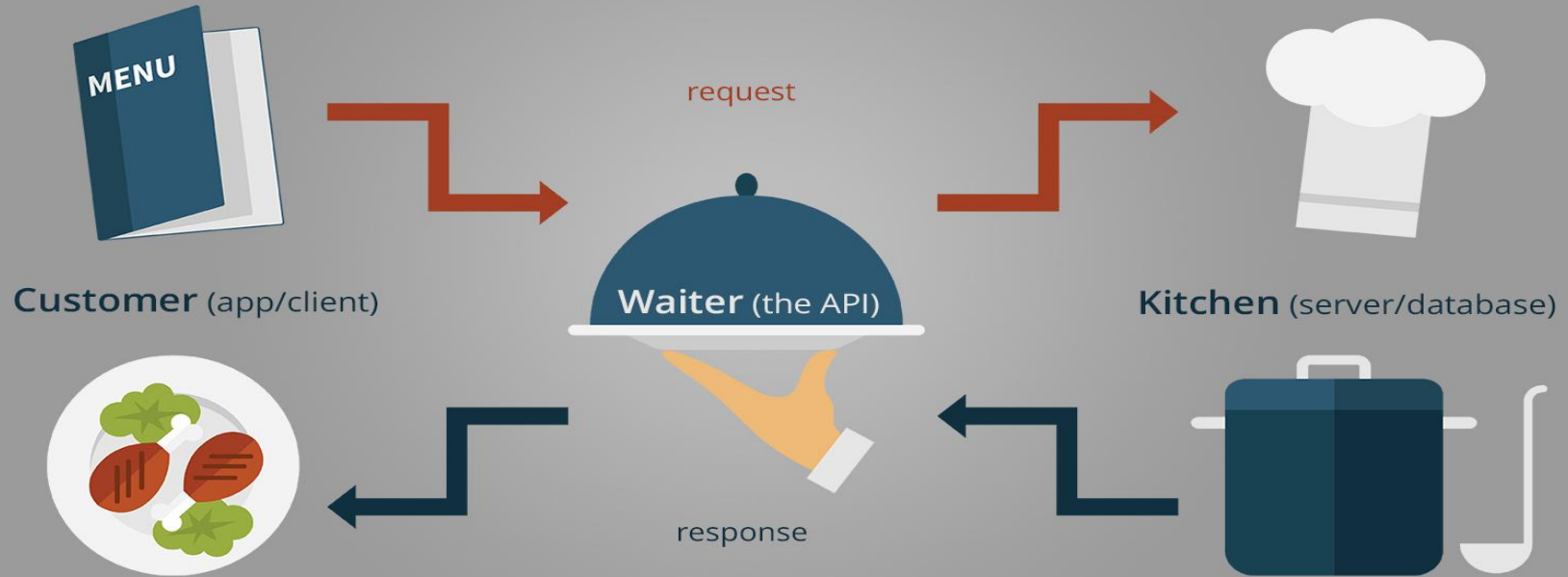
-200 OK

-500 Internal Server Error

-404 Not Found

<https://www.restapitutorial.com/httpstatuscodes.html>

# THE API RESTAURANT ANALOGY



10,000 ft

some elements provided by Vecteezy

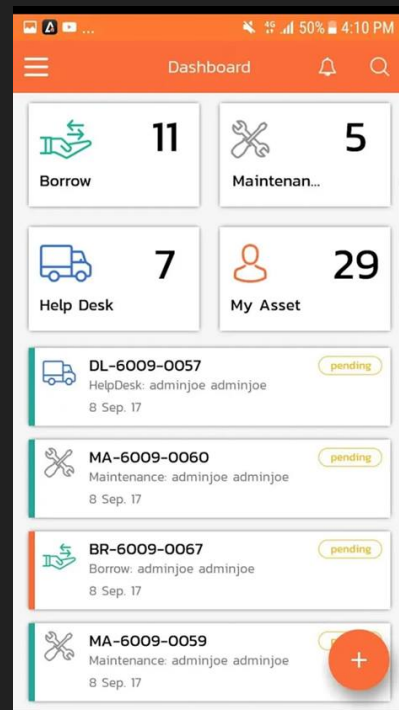
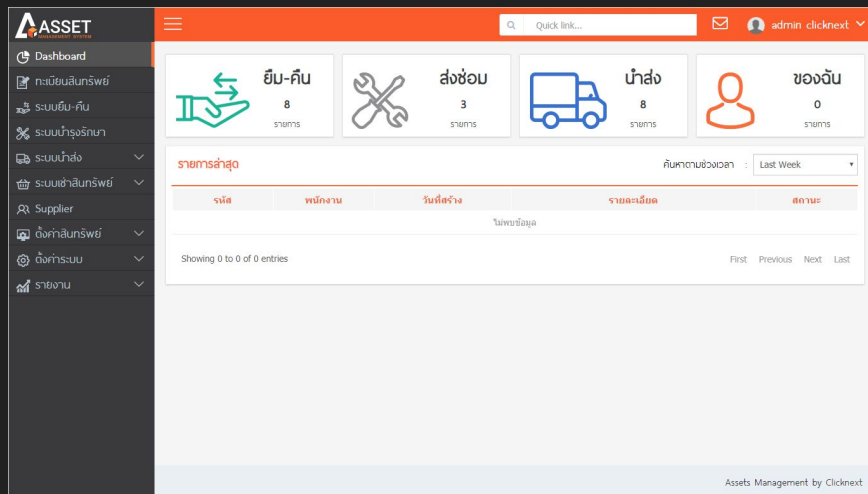
# .Net Core คืออะไร

เป็น framework ที่พัฒนาโดย Microsoft ซึ่งมีข้อดี คือ

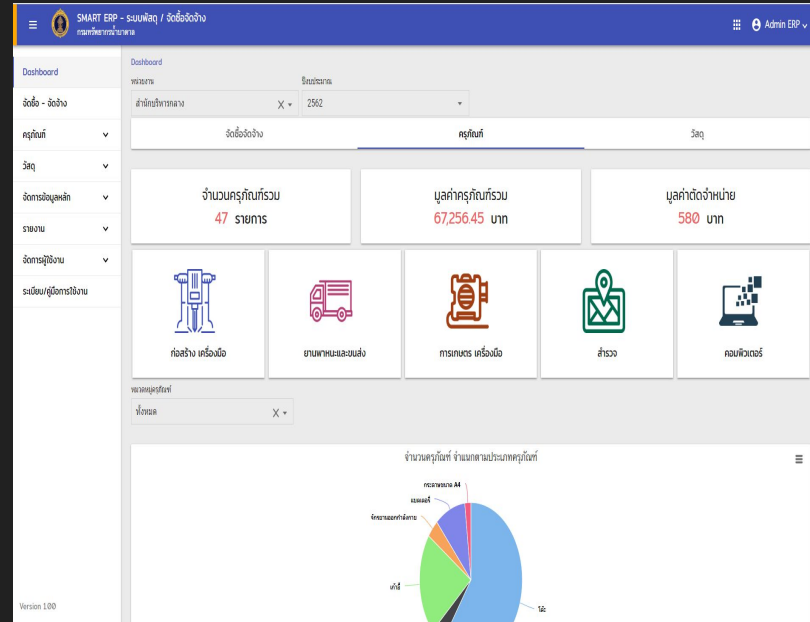
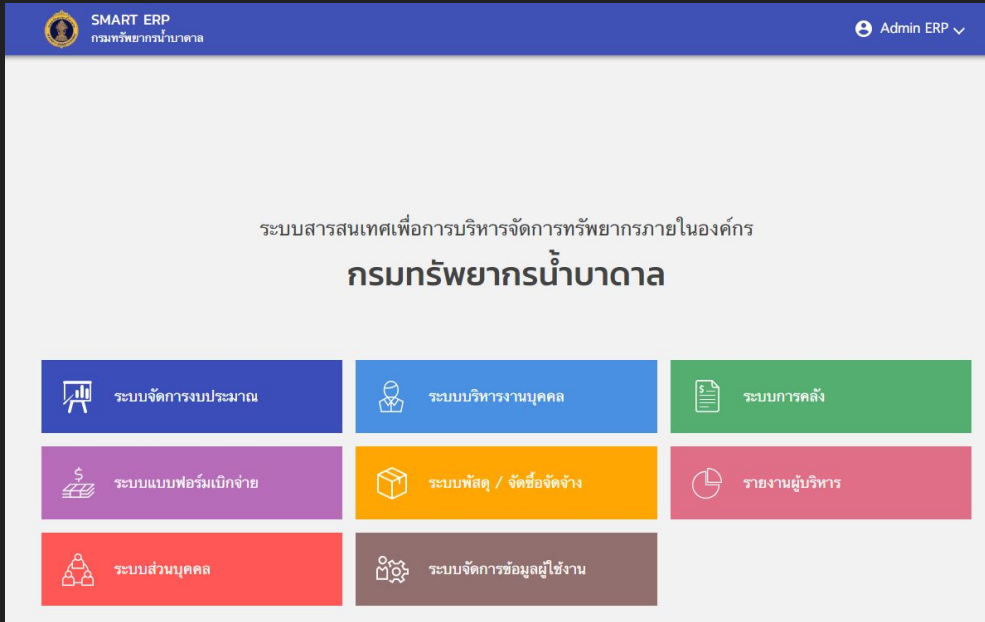
- cross-platform
- high-performance
- open-source



# Asset Management




# ERP กรมน้ำบาดาล



# CMS - PDMO

โทรศัพท์ : 02-265-8050 | 02-271-7999

หน้าหลัก | เกี่ยวกับเรา |ร่วมงานกับเรา | ติดต่อเรา | ลงทะเบียนรับข่าวสาร | ทำความคุ้นเคย | KM | PDMO Mail | Intranet | Smart Office | EN




สำนักงานบริหารหนี้สาธารณะ  
PUBLIC DEBT MANAGEMENT OFFICE

เข้าสู่ระบบ

Facebook Twitter Line

เกี่ยวกับเรา แผนการบริหารหนี้สาธารณะและรายงาน หนี้สาธารณะ ตราสารหนี้/สัญญากู้ยืม โครงการลงทุนภาครัฐ กฎหมาย ข่าวประชาสัมพันธ์ คลังเนื้อหา แอปพลิเคชัน



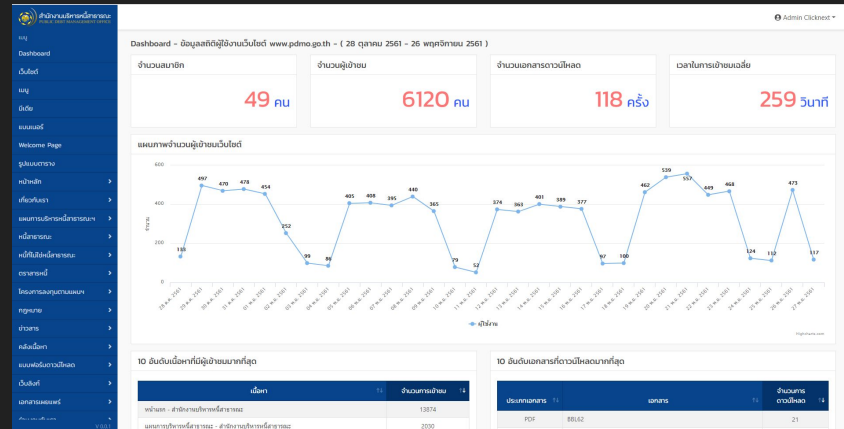
แผนการบริหารหนี้สาธารณะ  
ประจำปีงบประมาณ 2565

ดูข้อมูลเพิ่มเติม

สำนักงานบริหารการทะเบียนกำลังดำเนินการปรับปรุงเว็บไซต์ใหม่ ซึ่งจะมิใช่เอกสารบางส่วนอยู่ระหว่างถ่ายโอนข้อมูลเข้าสู่เว็บไซต์ใหม่ โดยสามารถรับชมได้เมื่อเปิดเว็บเบราว์เซอร์ Chrome เวอร์ชัน 59 ขึ้นไป ซึ่งหากต้องการเข้า  
ชมเว็บไซต์ใหม่ กรุณาคลิกที่นี่

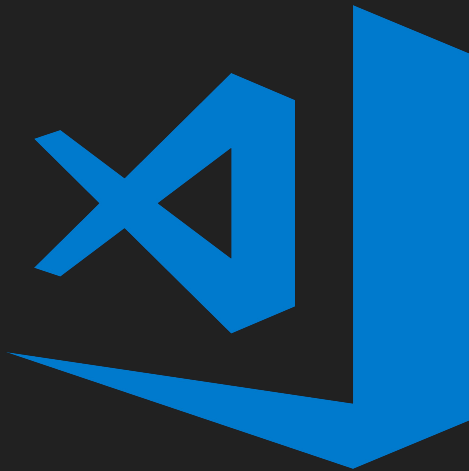
 ประเมินการสัดส่วนหนี้สาธารณะคงค้างต่อผลิตภัณฑ์มวลรวมในประเทศ และการหนี้ต้องประมาณ

สัดส่วน (%)	ปีงบประมาณ				
	2561	2562	2563	2564	2565
1. หนี้สาธารณะคงค้าง	41.70	42.73	43.59	44.67	46.01

[illegible]


# Install Tools for Dev

# Visual Studio Code





ดาวน์โหลด <https://code.visualstudio.com/>

# Extention in VS Code






**C#** `ms-vscode.csharp`

Microsoft |  8,968,124 |  | [Repository](#) | [License](#)

C# for Visual Studio Code (powered by OmniSharp).

[Install](#)



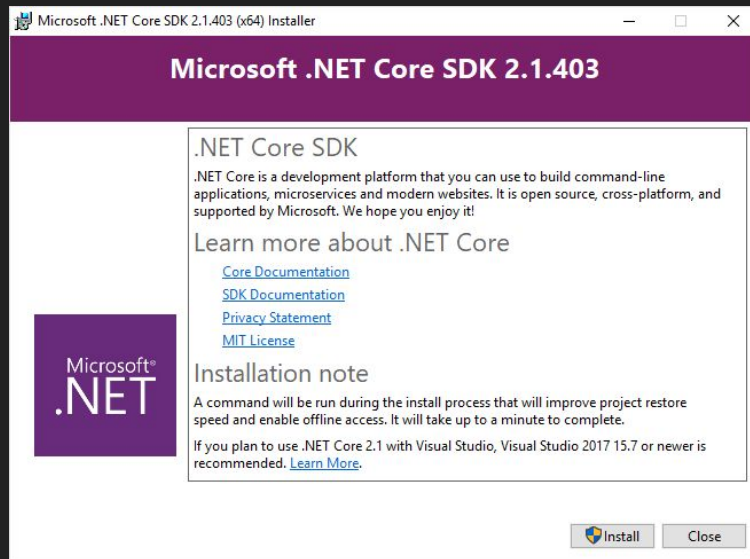
**C# Extensions** 1.3.0  302K  5

C# IDE Extensions for VSCode

jchannon

[Install](#)

# .Net Core SDK



ดาวน์โหลด <https://www.microsoft.com/net/download>

# CommandLine for .Net Core

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Sorakrai.k@BMSDev-08 MINGW64 ~

\$ dotnet

Usage: dotnet [options]

Usage: dotnet [path-to-application]

Options:

-h --help	Display help.
--info	Display .NET Core information.
--list-sdks	Display the installed SDKs.
--list-runtimes	Display the installed runtimes.

path-to-application:

The path to an application .dll file to execute.



# CommandLine for .Net Core(ต่อ)

คำสั่ง	รายละเอียด
new	Create new Project
build	Build Code
run	Build & Run Code
add	Add Package to Project
restore	Reinstall Package

# Create Web API

# Create Project Web API

- dotnet new webapi --name WebAPI --no-https

```
Sorakrai.k@BMSDev-08 MINGW64 ~
```

```
$ dotnet new webapi --name WebAPI --no-https
```

```
The template "ASP.NET Core Web API" was created successfully.
```

```
Processing post-creation actions...
```

```
Running 'dotnet restore' on WebAPI\WebAPI.csproj...
```

```
Restoring packages for C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj...
```

```
Generating MSBuild file C:\Users\Sorakrai.k\WebAPI\obj\WebAPI.csproj.nuget.g.props.
```

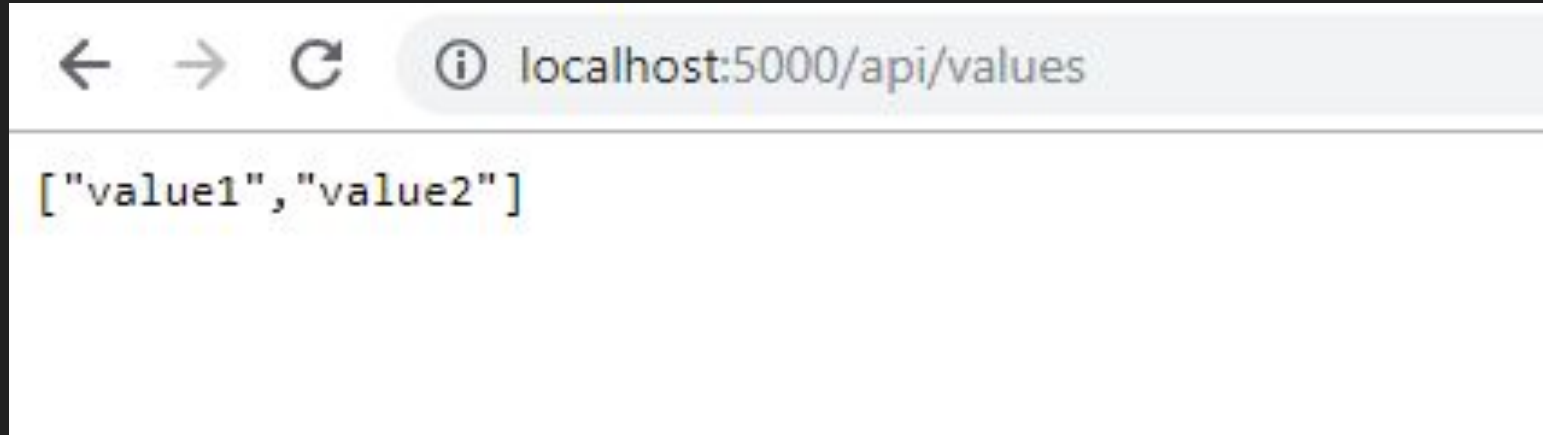
```
Generating MSBuild file C:\Users\Sorakrai.k\WebAPI\obj\WebAPI.csproj.nuget.g.targets.
```

```
Restore completed in 958.26 ms for C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj.
```

```
Restore succeeded.
```

# Run Project Web API

- dotnet run



# Code Generation tool

command used for generating controllers and views

# Code Generation tool(ต่อ)

-dotnet tool install --global dotnet-aspnet-codegenerator  
--version 2.1.5

```
Sorakrai.k@BMSDev-08 MINGW64 ~  
$ dotnet tool install --global dotnet-aspnet-codegenerator --version 2.1.5  
You can invoke the tool using the following command: dotnet-aspnet-codegenerator  
Tool 'dotnet-aspnet-codegenerator' (version '2.1.5') was successfully installed.
```

# Code Generation tool(ต่อ)

## - dotnet-aspnet-codegenerator

```
Sorakrai.k@BMSDev-08 MINGW64 ~
```

```
$ dotnet-aspnet-codegenerator
```

```
Scaffolding failed.
```

```
Could not find project file in C:\Users\Sorakrai.k
```

```
Usage: aspnet-codegenerator [arguments] [options]
```

Arguments:

generator Name of the generator. Check available generators below.

Options:

-p|--project Path to .csproj file in the project.  
-n|--nuget-package-dir  
-c|--configuration Configuration for the project (Possible values: Debug/ Release)  
-tfm|--target-framework Target Framework to use. (Short folder name of the tfm. eg. net46)  
-b|--build-base-path  
--no-build

To see more information, enable tracing by setting environment variable 'codegen\_trace' = 1.

RunTime 00:00:00.08

# Code Generation tool(ต่อ)

-dotnet add package

Microsoft.VisualStudio.Web.CodeGeneration.Design

```
Sorakrai.k@BMSDev-08 MINGW64 ~/WebAPI
$ dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design
Writing C:\Users\Sorakrai.k\AppData\Local\Temp\tmpB01C.tmp
info : Adding PackageReference for package 'Microsoft.VisualStudio.Web.CodeGeneration.Design' into pr
object 'C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj'.
log : Restoring packages for C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj...
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration.design/
index.json
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration.design/i
ndex.json 1365ms
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration.design/
2.1.6/microsoft.visualstudio.web.codegeneration.design.2.1.6.nupkg
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration.design/2
.1.6/microsoft.visualstudio.web.codegeneration.design.2.1.6.nupkg 970ms
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegenerators.mvc/in
dex.json
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegenerators.mvc/inde
x.json 270ms
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegenerators.mvc/2.1
.6/microsoft.visualstudio.web.codegenerators.mvc.2.1.6.nupkg
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegenerators.mvc/2.1
.6/microsoft.visualstudio.web.codegenerators.mvc.2.1.6.nupkg 270ms
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration/index.j
son
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration/index.js
on 580ms
info : GET https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration/2.1.6/m
icrosoft.visualstudio.web.codegeneration.2.1.6.nupkg
info : OK https://api.nuget.org/v3-flatcontainer/microsoft.visualstudio.web.codegeneration/2.1.6/mi
```



# Code Generation tool(ต่อ)

-dotnet restore

```
Sorakrai.k@BMSDev-08 MINGW64 ~/WebAPI
```

```
$ dotnet restore
```

```
Restoring packages for C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj...
```

```
Restore completed in 732.21 ms for C:\Users\Sorakrai.k\WebAPI\WebAPI.csproj.
```

# Code Generation tool(ต่อ)

## -dotnet-aspnet-codegenerator

```
Sorakrai.k@BMSDev-08 MINGW64 ~/WebAPI  
$ dotnet-aspnet-codegenerator
```

Usage: aspnet-codegenerator [arguments] [options]

Arguments:

generator Name of the generator. Check available generators below.

Options:

-p|--project Path to .csproj file in the project.  
-n|--nuget-package-dir  
-c|--configuration Configuration for the project (Possible values: Debug/ Release)  
-tfm|--target-framework Target Framework to use. (Short folder name of the tfm. eg. net46)  
-b|--build-base-path  
--no-build

Available generators:

area : Generates an MVC Area.  
controller: Generates a controller.  
identity : Generates an MVC Area with controllers and  
razorpage : Generates RazorPage(s).  
view : Generates a view.

RunTime 00:00:01.25

# Create Controller

-dotnet-aspnet-codegenerator controller -name FirstController -api -outDir Controllers

```
Sorakrai.k@BMSDev-08 MTINGW64 ~/WebAPT
```

```
$ dotnet-aspnet-codegenerator controller -name FirstController -api -outDir Controllers
```

```
Building project ...
```

```
Finding the generator 'controller'...
```

```
Running the generator 'controller'...
```

```
Added Controller : '\Controllers\FirstController.cs'.
```

```
RunTime 00:00:04.64
```

# Create First Action

- Http Method “GET”
- Return Type “string”
- Method Name “HelloWorld”

```
[HttpGet]  
0 references  
public string HelloWorld(){  
    |    return "Hello World";  
}
```

# Postman



ดาวน์โหลด <https://www.getpostman.com/apps>

# WorkShop

# User Management

## Requirement

1. สามารถแสดงรายการผู้ใช้งานทั้งหมดได้
2. สามารถเพิ่มข้อมูลผู้ใช้งานได้
3. สามารถแก้ไขข้อมูลผู้ใช้งานได้
4. สามารถลบข้อมูลผู้ใช้งานได้

# User Data

## ข้อมูลผู้ใช้งาน

1. ชื่อ(FirstName)
2. นามสกุล(LastName)
3. ชื่อเข้าใช้งาน(UserName)
4. รหัสผ่าน(Password)
5. อีเมล(Email)
6. เบอร์โทร(Tel)



## Step 1. Create Model Class

1. สร้าง folder สำหรับเก็บไฟล์ Model ชื่อ “Models”
2. สร้างไฟล์ class ชื่อ “UserModel” ใน folder Model

▲ WEBAPI

▸ .vscode

▸ bin

▸ Controllers

▲ Models

• UserModel.cs

▸ obj

▸ Properties

▸ wwwroot

{ } appsettings.Development.json

{ } appsettings.json

• Program.cs

• Startup.cs

• WebAPI.csproj

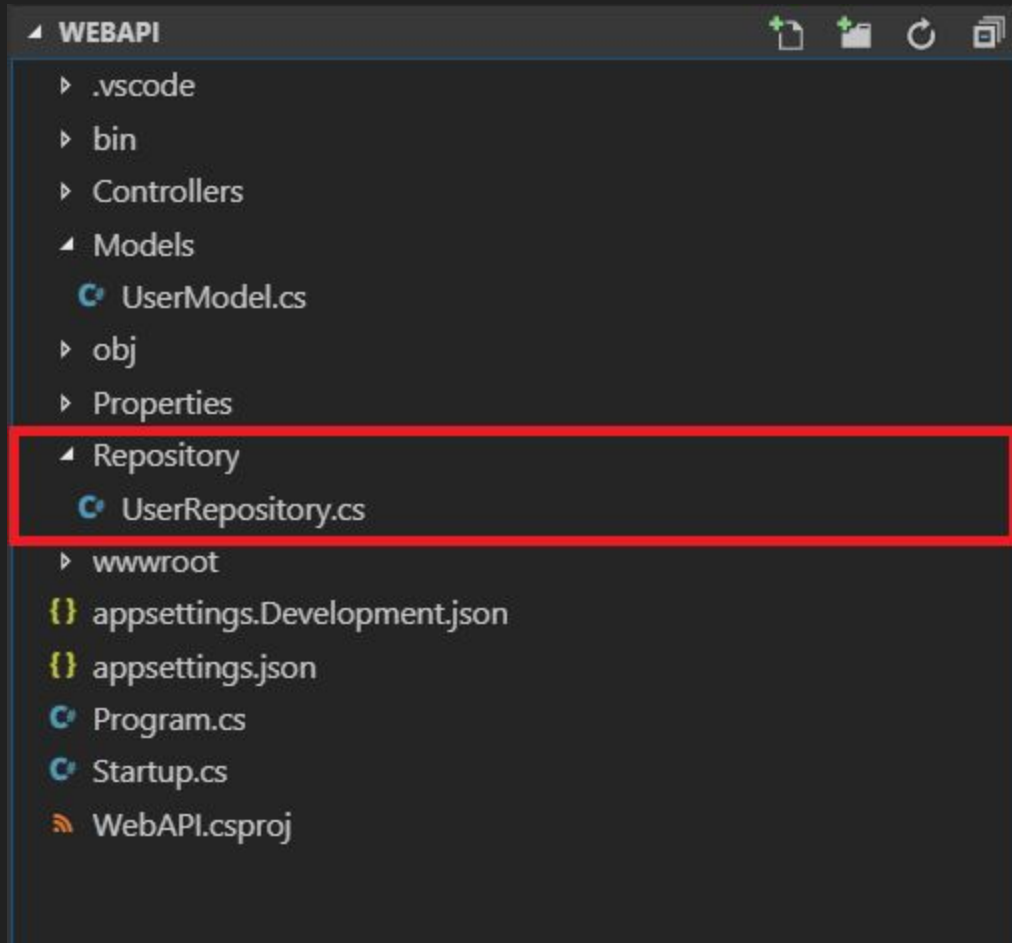
## Step 2. Define Property in UserModel

1. UserId:int
2. FirstName:string
3. LastName:string
4. UserName:string
5. Password:string
6. Email:string
7. Tel:string

```
1 namespace WebAPI.Models
2 {
3     0 references
4     public class UserModel
5     {
6         0 references
7         public int UserId { get; set; }
8         0 references
9         public string FirstName { get; set; }
10        0 references
11        public string LastName { get; set; }
12        0 references
13        public string UserName { get; set; }
14        0 references
15        public string Password { get; set; }
16        0 references
17        public string Email { get; set; }
18        0 references
19        public string Tel { get; set; }
20    }
21 }
```

## Step 3. Add Repository

1. สร้าง folder สำหรับเก็บไฟล์ Repository ชื่อ “Repository”
2. สร้างไฟล์ class ชื่อ “UserRepository” ใน folder Repository



## Step 5. Constructor in Repository

1. สร้าง constructor ใน class UserRepository
2. สร้างตัวแปร user เป็น List ข้อมูลของ UserModel เพื่อจัดการข้อมูล User

```
2 references
public List<UserModel> user;
0 references
public UserRepository(){
    this.user=new List<UserModel>();
}
```

## Step 6. Create Action Get All User

```
public List<UserModel> GetAllUser(){  
    return user.ToList();  
}
```

## Step 7. Register Service

- ต้อง register repository ก่อน controller จึงจะสามารถเรียกใช้ได้ วิธีการนี้ เรียกว่า “Dependency Injection”

```
25 // This method gets called by the runtime. Use this method to add services to the container.  
    0 references  
26 public void ConfigureServices(IServiceCollection services)  
27 {  
28     services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_2_2);  
29  
30     services.AddSingleton<UserRepository>();  
31 }
```



## Step 8. Add User Controller

- สร้าง controller ชื่อ UserController

```
dotnet-aspnet-codegenerator controller  
-name UserController -api -outDir  
Controllers
```

## Step 9. Constructor UserController

- สร้างตัวแปรจาก class UserRepository ชื่อ “userRepo”
- สร้าง constructor โดยมีการรับ paramer UserRepository ที่มาจากการ register ก่อนหน้านี้

```
1 reference
public UserRepository userRepo;
0 references
public UserController(UserRepository userRepo){
    this.userRepo=userRepo;
}
```

## Step 10. จัดการแสดงผู้ใช้งานทั้งหมด

- Http Method “GET”
- Return Type “List<UserModel>”
- Method Name “GetUser”

## Step 11. จัดการเพิ่มผู้ใช้งาน

### 1. เพิ่ม Method ใน Repository

- Return Type “void”
- Method Name “AddUser”
- Parameter UserModel
- Add User To List User

## Step 11. จัดการเพิ่มผู้ใช้งาน(ต่อ)

### 2. เพิ่ม Method ที่ controller

- Http Method “**POST**”
- Return Type “**void**”
- Method Name “**AddUser**”
- Parameter UserModel
- Call Method AddUser in UserRepository

## Step 12. จัดการแก้ไขผู้ใช้งาน

### 1. เพิ่ม Method ใน Repository

- Return Type “void”
- Method Name “EditUser”
- Parameter UserModel
- Edit User To List User By UserId

## Step 12. จัดการแก้ไขผู้ใช้งาน(ต่อ)

### 2. เพิ่ม Method ที่ controller

- Http Method “**POST**”
- Return Type “**void**”
- Method Name “**EditUser**”
- Parameter UserModel
- Call Method EditUser in UserRepository

## Step 13. จัดการลบผู้ใช้งาน

1. เพิ่ม Method ใน Repository
  - Return Type “void”
  - Method Name “DeleteUser”
  - Parameter UserModel
  - Delete User From List User By UserId



## Step 13. จัดการลบผู้ใช้งาน(ต่อ)

### 2. เพิ่ม Method ที่ controller

- Http Method “**POST**”
- Return Type “**void**”
- Method Name “**DeleteUser**”
- Parameter UserModel
- Call Method DeleteUser in UserRepository

## Example Use in FrontEnd

- สร้างไฟล์ **User.html** ใน folder wwwroot
- Copy Code From Url
- Set UseStaticFiles ในส่วน Configure  
`app.UseStaticFiles();`
- Open Browser  
<http://localhost:5000/User.html>

# User Management

เพิ่ม

FirstName	LastName	Email	Tel	
Sorakrai	Kasemsuk	test@mail.com	123456789	<a href="#">แก้ไข</a> <a href="#">ลบ</a>
test	test	test@mail.com	123456789	<a href="#">แก้ไข</a> <a href="#">ลบ</a>

Q&A

# แบบประเมิน



<https://drive.google.com/open?id=1fjh9frR2ydsFo-tAGVy-7yH6hLLmaj2Bg1SHjtrtKrw>

Editor Online : <http://collabedit.com/>

Code Training:

<https://github.com/sorakraikasemsuk/TrainingWebAPI>

FrontEnd Example:

<https://github.com/sorakraikasemsuk/TrainingWebAPI/blob/master/wwwroot/User.html>