

ระบบการเชื่อมโยงข้อมูลภายใน มหาวิทยาลัยราชภัฏกำแพงเพชร

การพัฒนา API

การพัฒนาระบบเชื่อมโยงข้อมูลผ่านเว็บเซอร์วิส API



งานเทคโนโลยีสารสนเทศ สำนักส่งเสริมวิชาการและงานทะเบียน
มหาวิทยาลัยราชภัฏกำแพงเพชร

สารบัญ

การพัฒนา API.....	1
API คืออะไร.....	1
ประโยชน์ของ API.....	1
ภาษาที่ใช้ในการพัฒนา	1
เริ่มต้นการพัฒนา API	2
ตัวอย่างที่ 1 การสร้างข้อมูลเพื่อแสดงผลในรูปแบบ JSON ไฟล์	2
ตัวอย่างที่ 2 การดึงข้อมูลจากฐานข้อมูลเพื่อแสดงผลในรูปแบบ JSON ไฟล์.....	4
ตัวอย่างที่ 3 การเพิ่ม ลบ แก้ไข ข้อมูลผ่าน API	7
การติดตั้งโปรแกรม Postman	10
การใช้งาน Slim Framework ในการสร้าง API	10
เริ่มต้นติดตั้ง Slim Framework	14
เริ่มสร้าง Function เพื่อทดสอบการทำงาน.....	15
เริ่มต้นติดต่อฐานข้อมูล.....	19
การพัฒนาระบบเชื่อมโยงข้อมูลผ่านเว็บเซอร์วิส	21
การเชื่อมโยงระบบด้วยเว็บเซอร์วิส	21
การสร้างเว็บไซต์เพื่อให้เชื่อมต่อกับระบบ API.....	21
ภาษาที่ใช้ในการพัฒนา	21
เงื่อนไขการเขียนโปรแกรมใช้งาน API ผ่าน AngularJS.....	21
เริ่มต้นการพัฒนาเว็บเพจด้วยภาษา HTML และ AngularJS.....	22
การใช้ AngularJS เชื่อมโยง API เพื่อนำข้อมูลมาแสดงผล.....	24
การเชื่อมโยง API โดยการส่งข้อมูลในรูปแบบ GET	25
การประยุกต์ใช้ API เพื่อเชื่อมโยงข้อมูลในรูปแบบ GET	28
การเชื่อมต่อ API ในรูปแบบ POST	30



การพัฒนา API

API คืออะไร

API ย่อมาจาก Application Programming Interface คือ ช่องทางการเชื่อมต่อช่องทางหนึ่งที่จะเชื่อมต่อกับเว็บไซต์ผู้ให้บริการ API จากที่อื่น เป็นตัวกลางที่ทำให้โปรแกรมประยุกต์เชื่อมต่อกับโปรแกรมประยุกต์อื่น หรือเชื่อมการทำงานเข้ากับระบบปฏิบัติการ ตัวอย่าง เช่น Twitter มีหลายเว็บ ที่มีการเชื่อมโยงข้อมูลกับ twitter ทั้งเป็นการอ่านข้อมูลจาก twitter หรือ ส่งข้อมูลเข้า twitter เองก็ตาม ซึ่งล้วนอาศัยการเชื่อมต่อแลกเปลี่ยนข้อมูลกัน ด้วย API นั่นเอง และอีกหนึ่งตัวอย่าง เช่น Google Maps API คือบริการของ Google อีกรูปแบบหนึ่งที่เราสามารถนำข้อมูลของ Google Maps ที่ทาง Google ให้บริการโดยส่วนมากจะนำมาใช้กับเว็บไซต์ ของบริษัทหรือ เว็บไซต์ห้างร้านต่างๆ เพื่อเป็นอีกช่องทางที่ให้ลูกค้ารู้ว่าบริษัทฯ หรือห้างร้านนั้น

ประโยชน์ของ API

1. สามารถรับส่งข้อมูลข้าม Server ได้
2. ไม่จำเป็นต้องเข้าหน้าเว็บหลัก ก็มีข้อมูลของเว็บหลัก จากเว็บที่ดึง API เอพีไอ แบ่งเป็น
 - 2.1. เอพีไอที่ขึ้นกับภาษา (language-dependent API) คือ เอพีไอ ที่สามารถการเรียกใช้จากโปรแกรมที่เขียนขึ้นด้วยภาษาเพียงภาษาใดภาษาหนึ่ง
 - 2.2. เอพีไอไม่ขึ้นกับภาษา (language-independent API) คือ เอพีไอ ที่สามารถเรียกได้จากโปรแกรมหลายๆภาษา

API ถือเป็นกลุ่มของฟังก์ชัน ขั้นตอน หรือคลาส (Class) ที่ระบบปฏิบัติการ (OS) หรือผู้ให้บริการสร้างขึ้นมา เพื่อรองรับการเรียกขอข้อมูล จากโปรแกรมอื่น ๆ ทั้งนี้ API สามารถใช้งานได้กับภาษาในการเขียนโปรแกรมที่รองรับเท่านั้น ซึ่งมันจะถูกจัดทำให้อยู่ในรูปแบบ Syntax หรือ element ที่สามารถนำไปใช้ได้อย่างสะดวกสบาย

ภาษาที่ใช้ในการพัฒนา

1. PHP
2. ASP, ASP.NET
3. JAVASCRIPT
4. Node JS
5. ฯลฯ

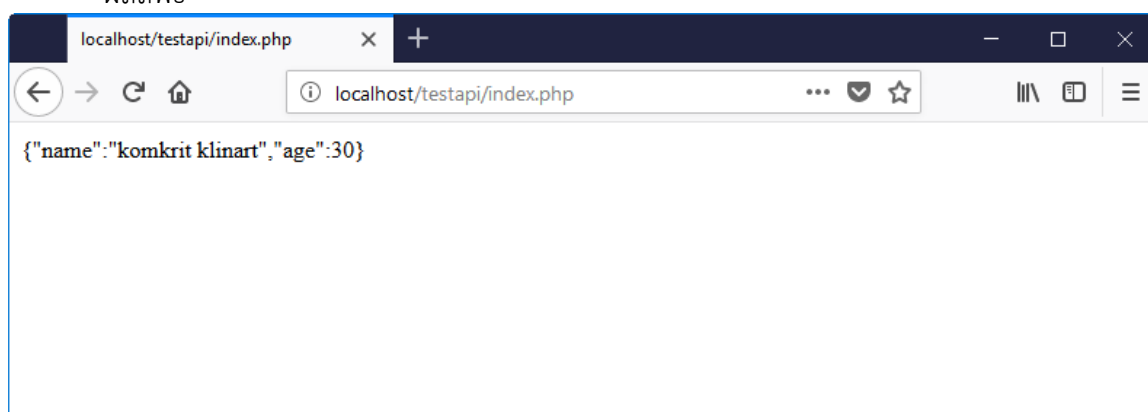
เริ่มต้นการพัฒนา API

ตัวอย่างที่ 1 การสร้างข้อมูลเพื่อแสดงผลในรูปแบบ JSON ไฟล์

index.php

```
<?php
    $arr = array(
        'name' => 'komkrit klinart',
        'age' => 30
    );
    echo json_encode($arr);
?>
```

ผลลัพธ์



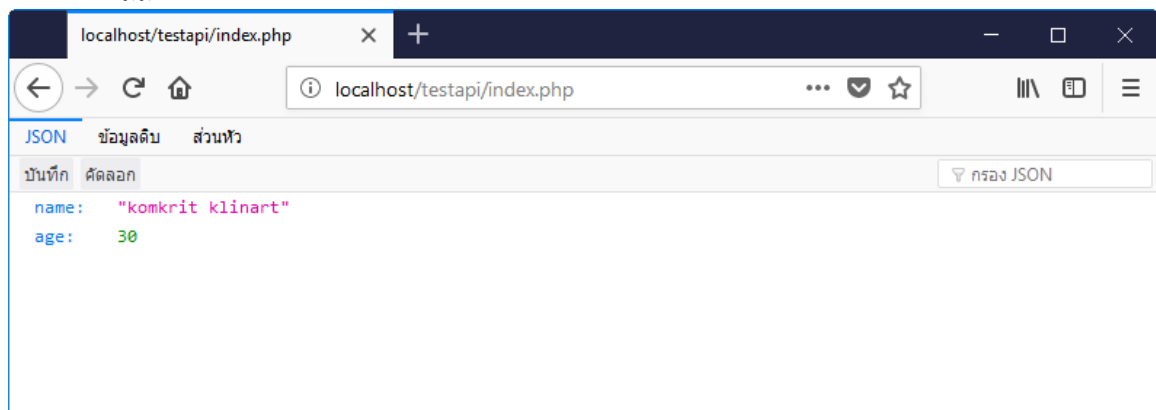
ตัวอย่างที่ 1 ยังคงเป็นเพียงการแสดงผลข้อมูลในรูปแบบ Array อยู่ ยังไม่สามารถนำไปใช้ในการแสดงผลในรูปแบบ JSON ที่ภาษาทุกภาษาสามารถนำไปใช้ได้ จะต้องเพิ่มเติมชุดคำสั่งอีกเล็กน้อยเพื่อให้สามารถใช้งานได้ โดยนำชุดคำสั่ง 2 บรรทัดด้านล่างไปวางไว้ด้านบน

```
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
```

```
<?php
    header("Access-Control-Allow-Origin: *"); //CORS อนุญาตให้ใช้ข้อมูลได้
    header("Content-Type: application/json; charset=UTF-8"); //ประกาศให้ข้อมูลต่างๆ
    เป็นข้อมูลประเภท JSON รองรับภาษา

    $arr = array(
        'name' => 'komkrit klinart',
        'age' => 30
    );
    echo json_encode($arr);
?>
```

ผลลัพธ์

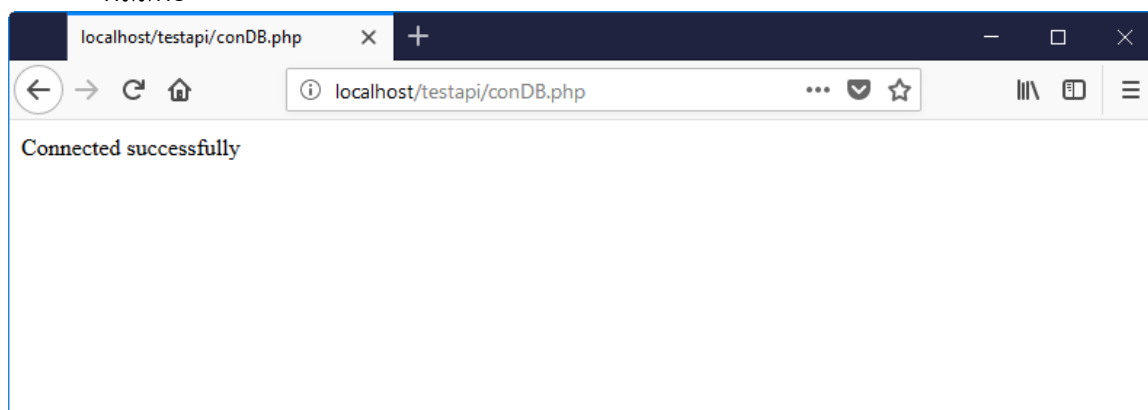


ตัวอย่างที่ 2 การดึงข้อมูลจากฐานข้อมูลเพื่อแสดงผลในรูปแบบ JSON ไฟล์
conDB.php

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username,
$password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
}
catch(PDOException $e)
{
    echo "Connection failed: " . $e->getMessage();
}
?>
```

ผลลัพธ์



เริ่มต้นดึงข้อมูลมาแสดงผล
index.php

```
<?php

    header("Access-Control-Allow-Origin: *");

    header("Content-Type: application/json; charset=UTF-8");

    include "conDB.php";

    $sql = "SELECT * FROM MyGuests";

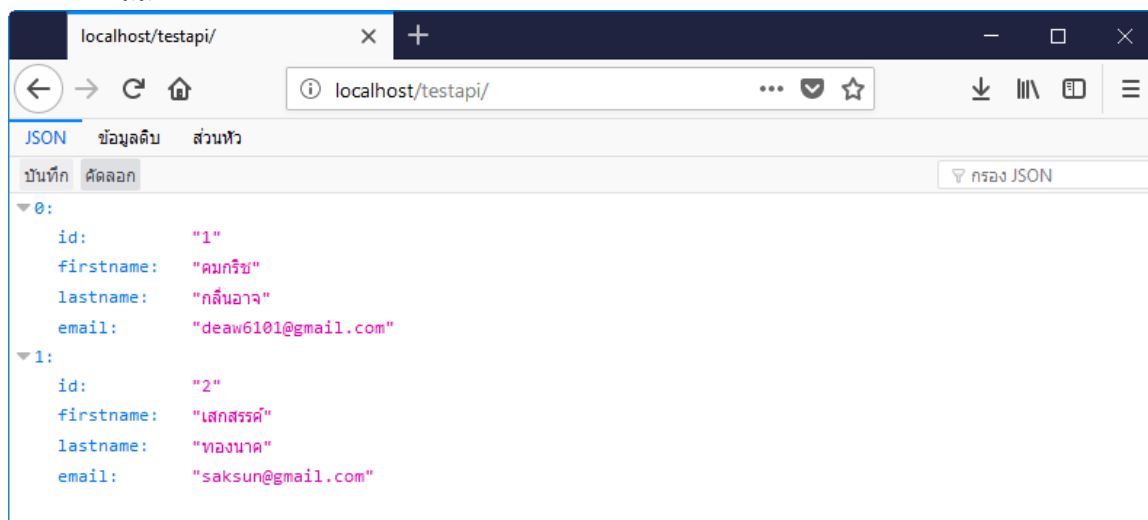
    $result = $conn->query($sql);

    $row = $result->fetchAll(PDO::FETCH_ASSOC);

    echo json_encode($row);

?>
```

ผลลัพธ์

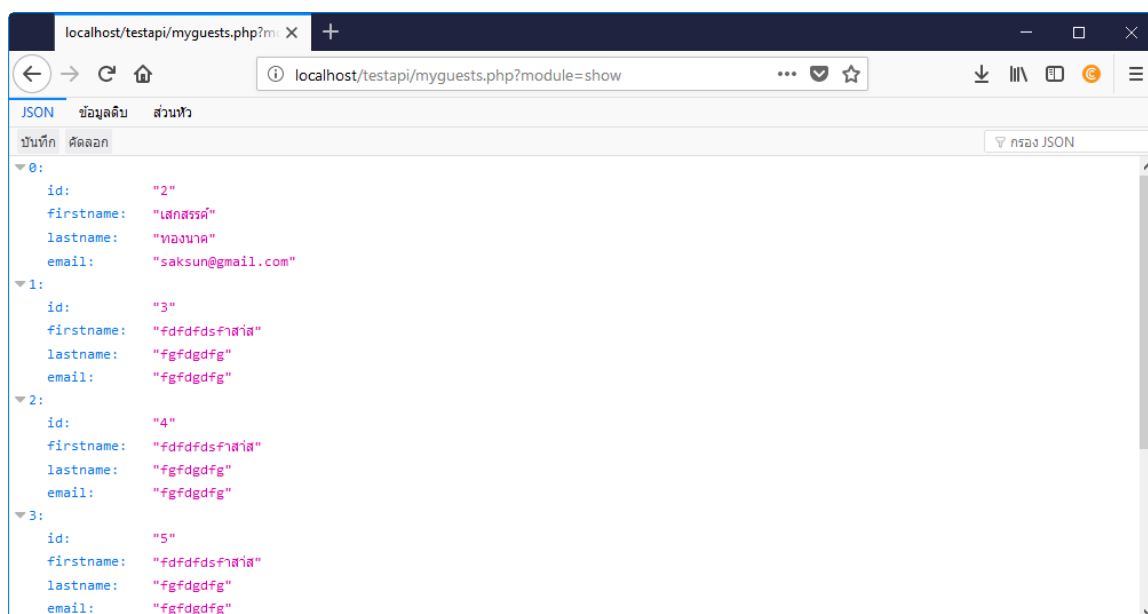


ตัวอย่างที่ 3 การเพิ่ม ลบ แก้ไข ข้อมูลผ่าน API

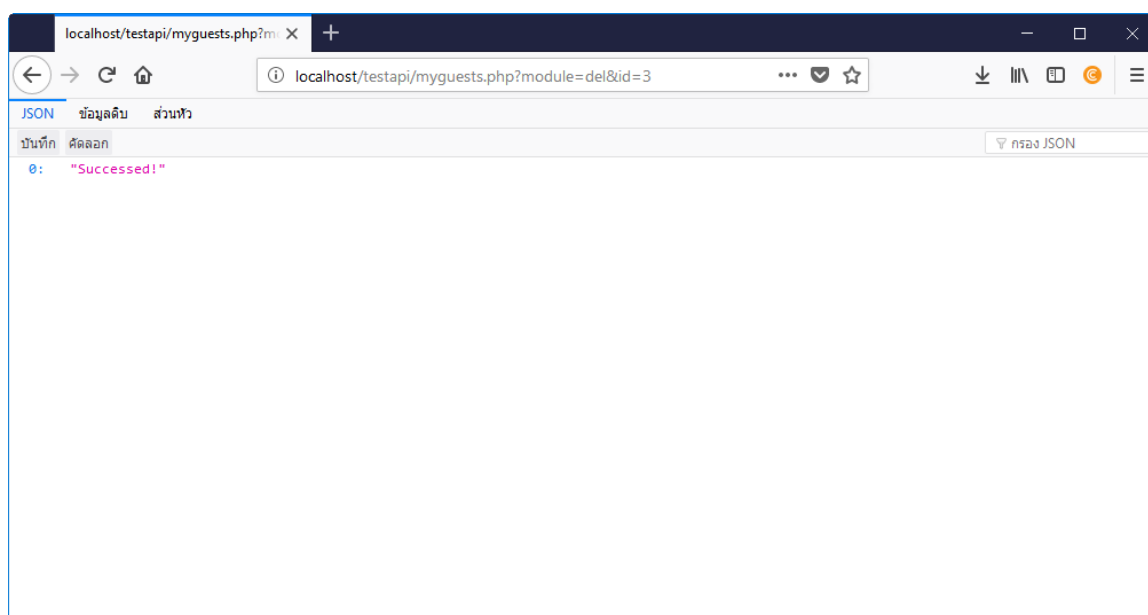
```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
include "conDB.php";
if($_GET['module'] == "show"){
    $sql = "SELECT * FROM MyGuests";
    $result = $conn->query($sql);
    $row = $result->fetchAll(PDO::FETCH_ASSOC);
    echo json_encode($row);
}
if($_GET['module'] == "del"){
    $sql = "delete FROM MyGuests where id = '".$_GET['id']."'";
    $result = $conn->query($sql);
    $rr = array("Succesed!");
    echo json_encode($rr);
}
if($_GET['module'] == "insert"){

    $sql = "insert into MyGuests (firstname,lastname,email) values
('".$_POST['firstname']."','".$_POST['lastname']."','".$_POST['email']."'";
    $result = $conn->query($sql);
    $rr = array("Succesed!");
    echo json_encode($rr);
}
?>
```

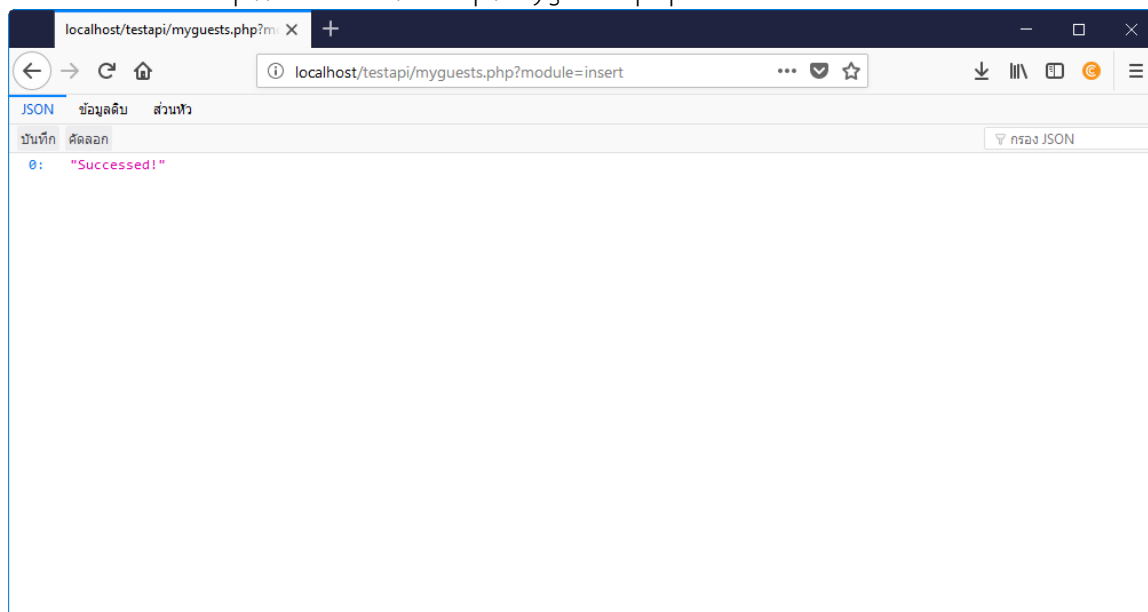
ผลลัพธ์ <http://localhost/testapi/myguests.php?module=show>



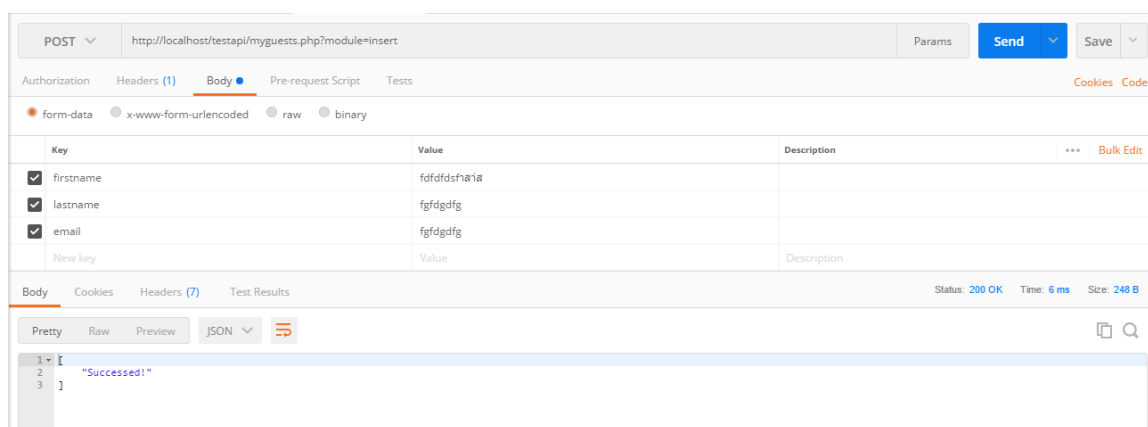
ผลลัพธ์ <http://localhost/testapi/myguests.php?module=del&id=3>



ผลลัพธ์ <http://localhost/testapi/myguests.php?module=insert>

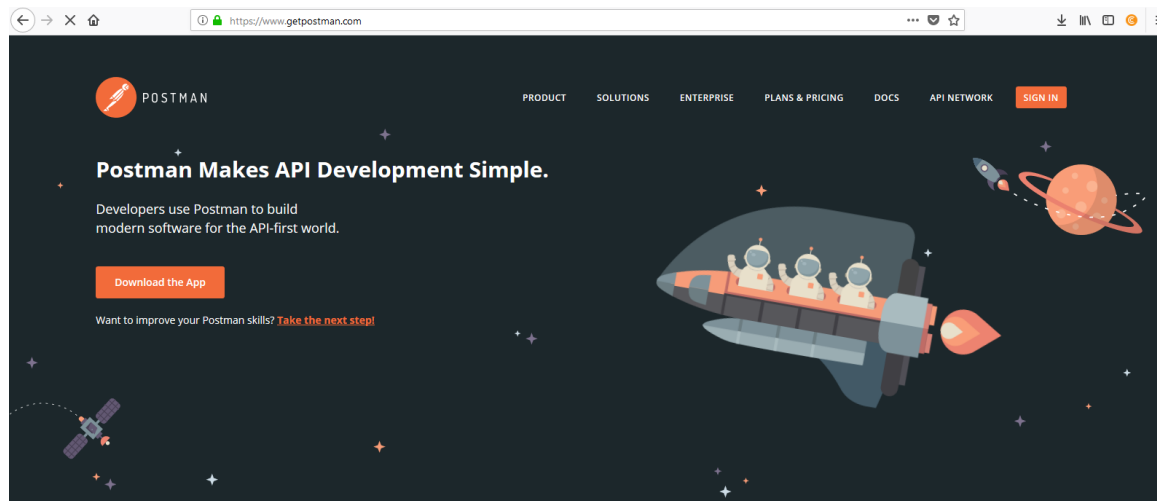


เนื่องจากการเพิ่มข้อมูลไม่สามารถใช้ Web Browser ได้ จึงแนะนำให้ใช้โปรแกรม Postman ในการเพิ่มข้อมูล



การติดตั้งโปรแกรม Postman

ดาวน์โหลดได้จาก <https://www.getpostman.com/>



การใช้งาน Slim Framework ในการสร้าง API

การใช้ Slim Framework ในการสร้าง API เพื่อให้การทำงานของ API มีประสิทธิภาพมากยิ่งขึ้น โดยสามารถศึกษาวิธีการใช้งานได้ที่ <https://www.slimframework.com/>

ความต้องการของ Slim Framework

1. Web server with URL rewriting
2. PHP 5.5 or newer

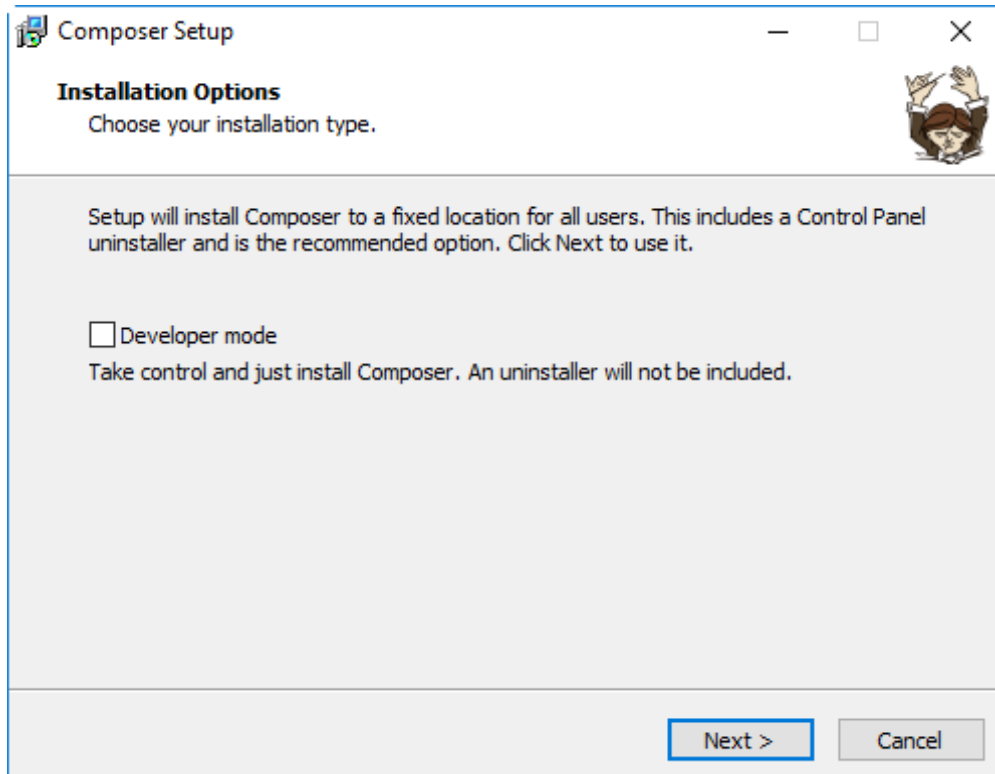
การติดตั้งต้องใช้ composer ในการติดตั้ง Slim Framework โดยสามารถดาวน์โหลดได้ที่ <https://getcomposer.org/>



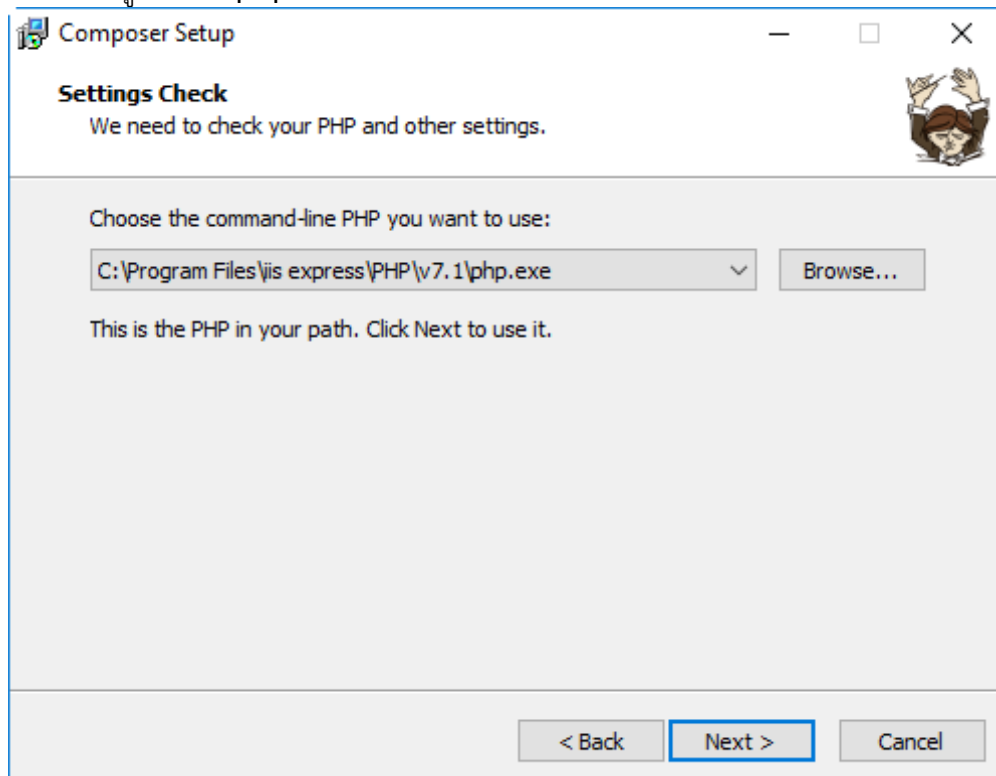
Dependency Manager for PHP

[Getting Started](#)[Download](#)[Documentation](#)[Browse Packages](#)[Issues](#)[GitHub](#)

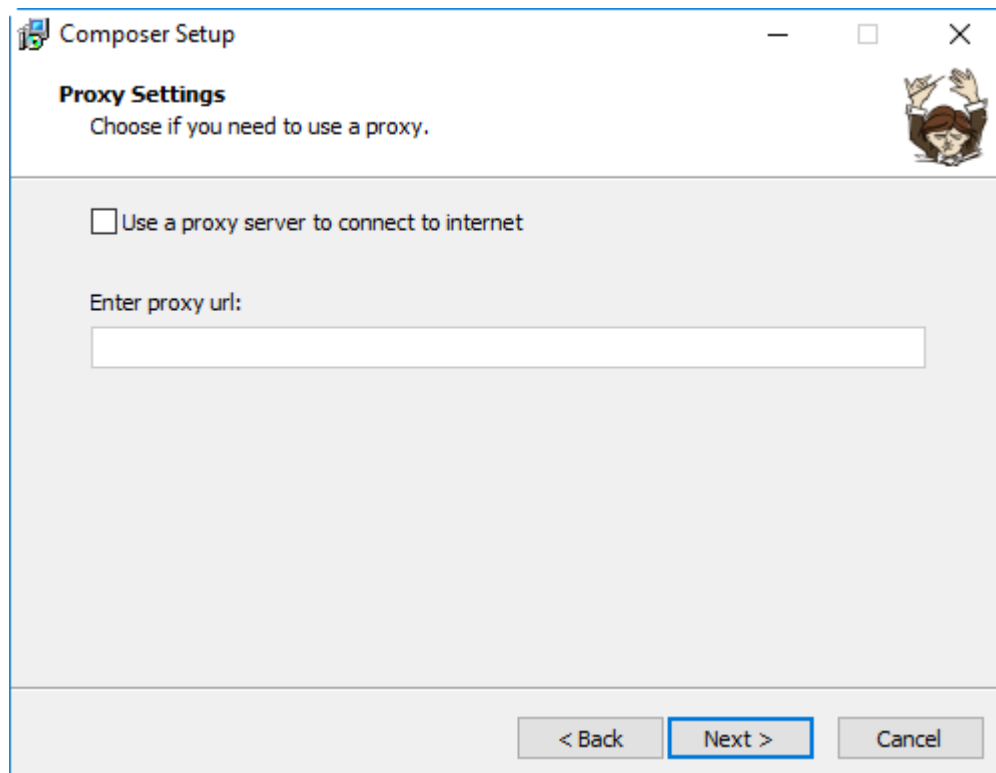
คลิก Next



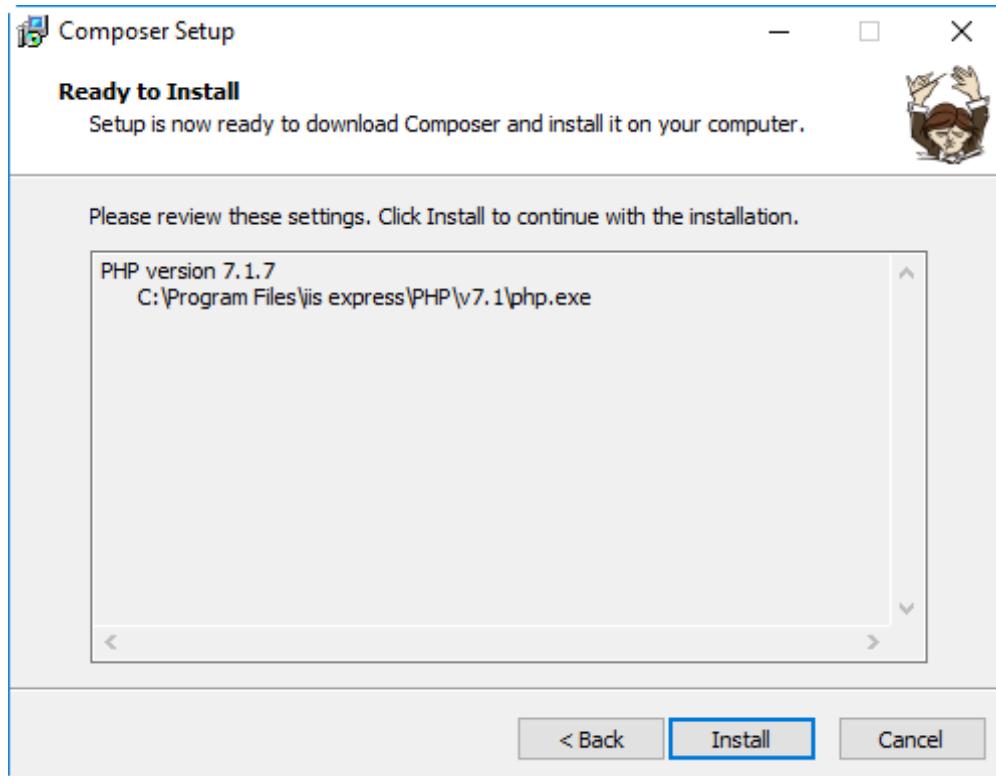
เลือกที่อยู่ของไฟล์ php.exe คลิก Next



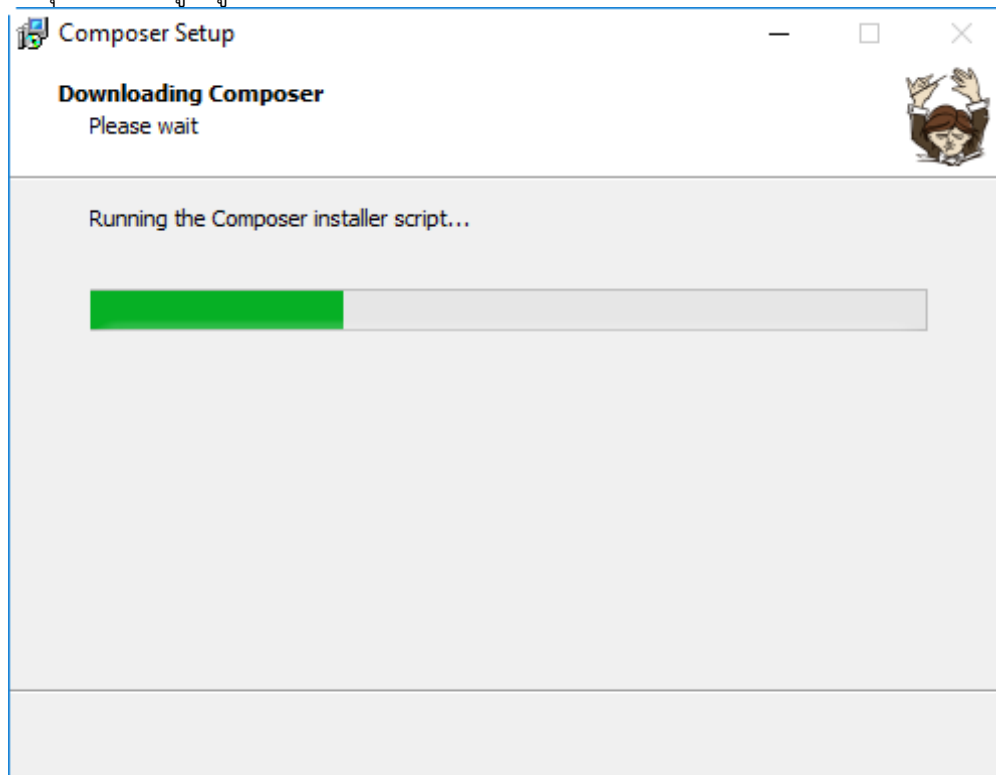
คลิก Next



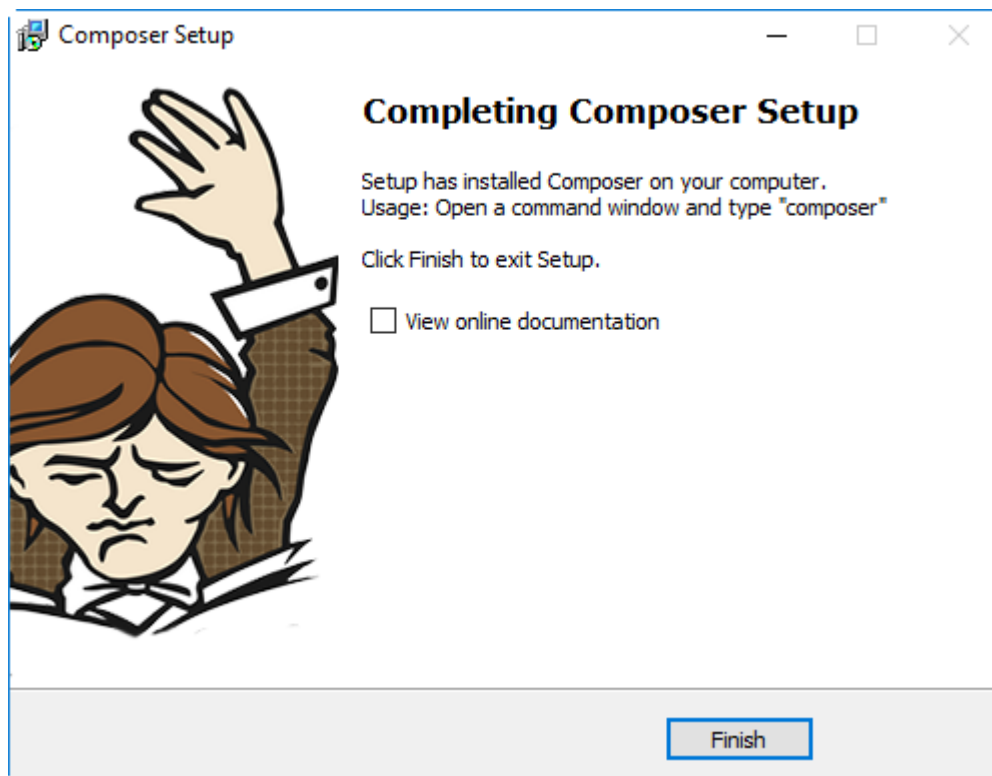
คลิก Next



กรุณารอสักครู่ อยู่ระหว่างการดำเนินการติดตั้ง



คลิก Finish



เริ่มต้นติดตั้ง Slim Framework

ให้ดำเนินการสร้างโฟลเดอร์สำหรับสร้างโปรเจกขึ้นมา แล้วใช้

```
composer require slim/slim "^3.0"
```

หลังจากติดตั้งแล้วให้เข้าไปยังโฟลเดอร์โปรเจก แล้วสร้างไฟล์ชื่อ index.php และแทรกชุดคำสั่งด้านล่างในไฟล์ index.php

```
<?php
require 'vendor/autoload.php';
```

หลังจากนั้นพิมพ์คำสั่งด้านล่างเพื่อเริ่มการใช้งาน Slim Framework

```
$app = new \Slim\App();
```


เริ่มสร้าง Function เพื่อทดสอบการทำงาน

```
$app->get('/', function ($req, $res) {  
    $res->write('Hi, I am Web API');  
    return $res;  
});
```

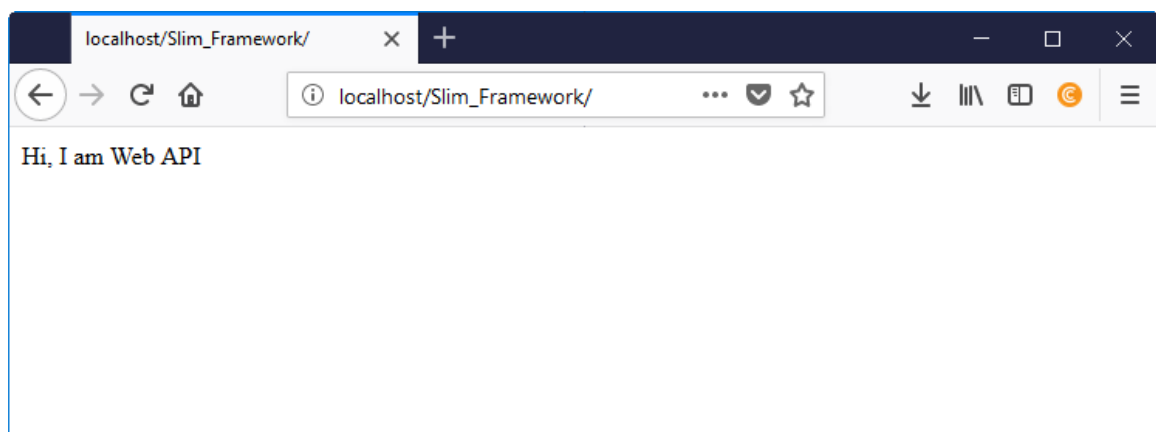
แล้วเพิ่มคำสั่ง `$app->run();` เพื่อสั่งให้ Slim Framework ทำงาน

```
$app->run();
```

แต่เนื่องจาก Slim Framework ต้องทำงานในรูปแบบ URL rewriting จึงต้องมีไฟล์ `.htaccess` เพื่อใช้งาน URL rewriting

```
RewriteEngine On  
RewriteCond %{REQUEST_FILENAME} !-f  
RewriteCond %{REQUEST_FILENAME} !-d  
RewriteRule ^ index.php [QSA,L]
```

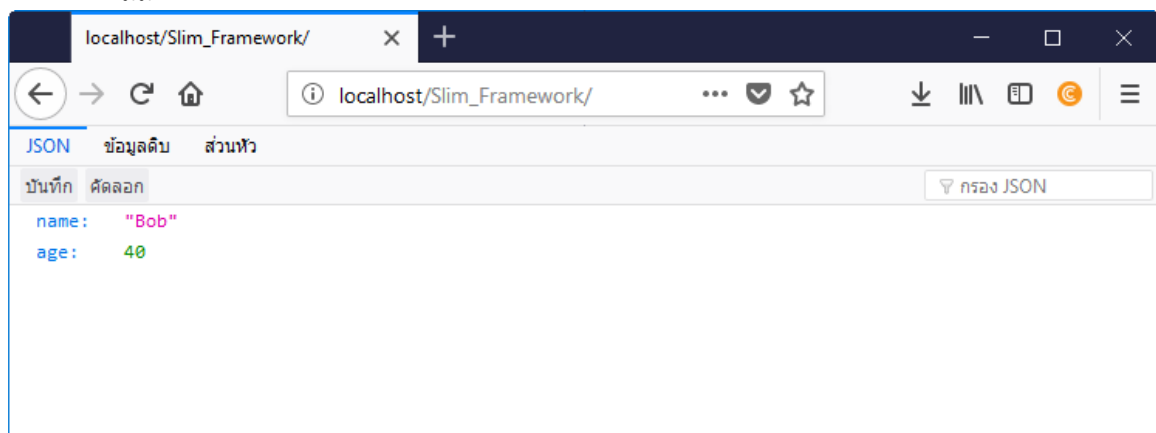
หลังจากเตรียมทุกอย่างเรียบร้อยแล้ว ให้ลองรันเพื่อทดสอบว่าการแสดงผลถูกต้องหรือไม่ หากสำเร็จจะได้ผลดังรูป



การแสดงผลในรูปแบบ JSON โดยพิมพ์ชุดคำสั่งด้านล่าง

```
$app->get('/', function ($req, $res) {
    $data = array('name' => 'Bob', 'age' => 40);
    $newResponse = $res->withJson($data);
    return $newResponse;
});
```

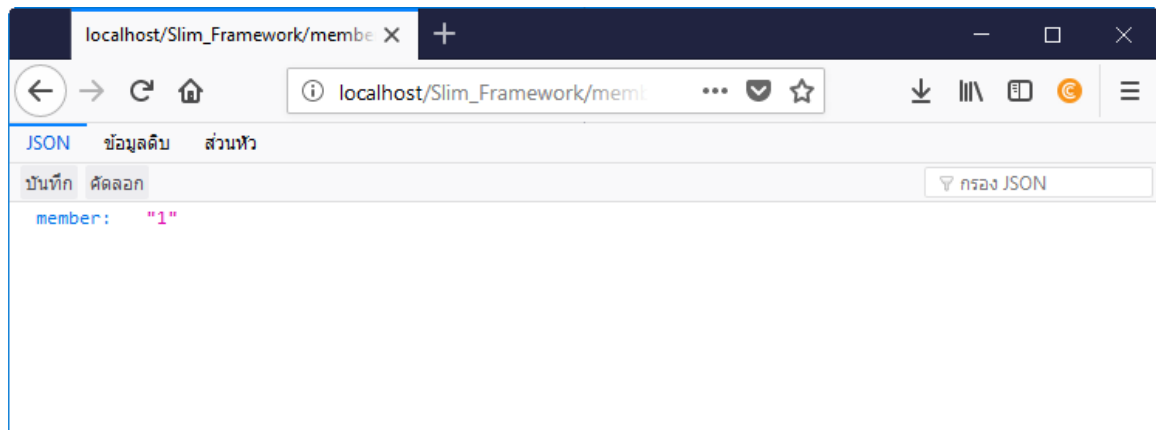
ผลลัพธ์



การส่งข้อมูลมาพร้อม URL แบบ GET

```
$app->get('/members/{id}', function ($req, $res) {
    $Members = $req->getAttribute('id');
    $data = array('member' => $Members);
    $newResponse = $res->withJson($data);
    return $newResponse;
});
```

ผลลัพธ์



การส่งข้อมูลมาพร้อม URL แบบ POST

```
$app->post('/members/', function ($req, $res) {
    $Members = $req->getParam('id');
    $data = array('member' => $Members);
    $newResponse = $res->withJson($data);
    return $newResponse;
});
```

แต่ยังไม่สามารถใช้งานได้เนื่องจาก Slim Framework ไม่อนุญาตให้ใช้งานได้เพราะติด CORS จึงต้องติดตั้ง CORS ใน Slim Framework เพื่อให้สามารถใช้งาน POST ได้ โดยทำตามขั้นตอนดังนี้

1. ให้เปิดไฟล์ composer.json และเพิ่มคำสั่งดังภาพ

```
{
  "require": {
    "slim/slim": "^3.0",
    "palanik/cors Slim": "dev-slim3"
  }
}
```

2. เปิด cmd แล้วรันคำสั่ง

Composer update

3. แทรกชุดคำสั่งต่อไปนี้เพื่อให้สามารถใช้งานได้

```
$corsOptions = array(
    "origin" => "*",
    "exposeHeader" => array("Content-Type", "X-Requested-With", "X-
authentication", "X-client"),
    "allowMethod" => array('GET', 'POST', 'PUT', 'DELETE', 'OPTIONS')
);
$cors = new \CorsSlim\CorsSlim($corsOptions);
$app->add($cors);
```

ผลลัพธ์

POST http://localhost/Slim_Framework/members/ Params Send

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

Body Cookies Headers (6) Test Results Status: 200 OK Time: 30 ms

Pretty Raw Preview JSON

```
1 {
2   "member": "1"
3 }
```

เริ่มต้นติดต่อฐานข้อมูล

```
<?php
require 'vendor/autoload.php';

$config['db']['host'] = "localhost";
$config['db']['user'] = "root";
$config['db']['pass'] = "51640826@tb";
$config['db']['dbname'] = "mydb";

$app = new \Slim\App(["settings" => $config]);

$corsOptions = array(
    "origin" => "*",
    "exposeHeader" => array("Content-Type", "X-Requested-With", "X-
authentication", "X-client"),
    "allowMethod" => array('GET','POST','PUT','DELETE','OPTIONS')
);
$cors = new \CorsSlim\CorsSlim($corsOptions);
$app->add($cors);

$container = $app->getContainer();

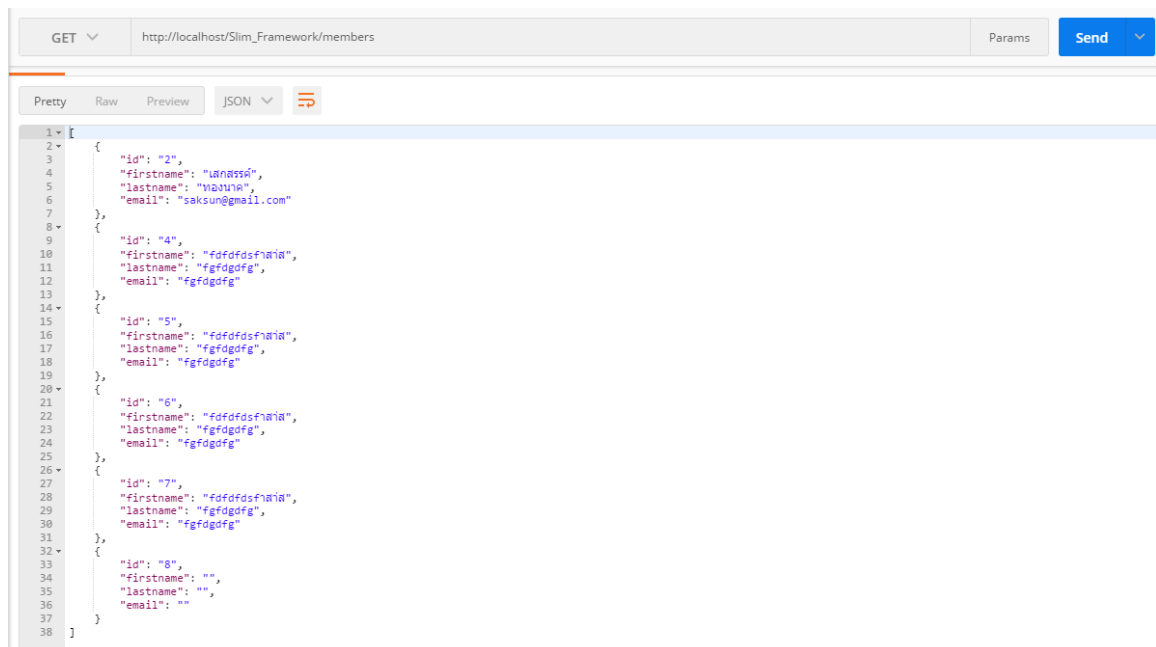
$container['db'] = function($c){
    $db = $c['settings']['db'];
    $pdo = new PDO("mysql:host=".
$db['host'].";dbname=".$db['dbname'],$db['user'], $db['pass']);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
    return $pdo;
};

$app->run();
?>
```

ทดสอบการดึงข้อมูลมาแสดงผล

```
$app->get('/members', function ($req, $res) {
    $result = array();
    $sql = "select * from myguests";
    $data = $this->db->query($sql);
    foreach($data as $rows){
        array_push($result,$rows);
    }
})
```

ผลลัพธ์



GET Params

Pretty Raw Preview JSON

```
[
  {
    "id": "2",
    "firstname": "เสกสรรค์",
    "lastname": "ทองขาว",
    "email": "saksun@gmail.com"
  },
  {
    "id": "4",
    "firstname": "สงกรานต์",
    "lastname": "สงกรานต์",
    "email": "fgfdgdfg"
  },
  {
    "id": "5",
    "firstname": "สงกรานต์",
    "lastname": "fgfdgdfg",
    "email": "fgfdgdfg"
  },
  {
    "id": "6",
    "firstname": "สงกรานต์",
    "lastname": "fgfdgdfg",
    "email": "fgfdgdfg"
  },
  {
    "id": "7",
    "firstname": "สงกรานต์",
    "lastname": "fgfdgdfg",
    "email": "fgfdgdfg"
  },
  {
    "id": "8",
    "firstname": "",
    "lastname": "",
    "email": ""
  }
]
```

การพัฒนาระบบเชื่อมโยงข้อมูลผ่านเว็บเซอร์วิส

การเชื่อมโยงระบบด้วยเว็บเซอร์วิส

ระบบสารสนเทศในปัจจุบันมีความจำเป็นต้องใช้ข้อมูลในด้านต่างๆ เพื่อจัดเก็บ นำเสนอ และสรุปรายงานผล และตามนโยบายของมหาวิทยาลัยฯ มีความต้องการให้หน่วยงานภายในมหาวิทยาลัยฯ สามารถเชื่อมโยงข้อมูลต่างๆ เข้าด้วยกัน เพื่อให้ข้อมูลเป็นปัจจุบัน ถูกต้อง และไม่ซ้ำซ้อน ซึ่งการเชื่อมโยงข้อมูลโดยใช้เพียงการเชื่อมต่อระบบฐานข้อมูลนั้นอาจมีข้อจำกัดหลายอย่าง เช่น ไม่สามารถเชื่อมฐานข้อมูลของต่างหน่วยงานได้เนื่องจากข้อจำกัดด้านความปลอดภัยของข้อมูล และหากแต่ละหน่วยงานสร้างฐานข้อมูลที่เก็บข้อมูลในรูปแบบคล้ายกันเพื่อใช้ในหน่วยงาน อาจทำให้เกิดการซ้ำซ้อนของข้อมูล ไม่สามารถยืนยันได้ว่าข้อมูลนี้ถูกต้องหรือเป็นปัจจุบันหรือไม่ การเขียนเว็บไซต์ให้สามารถเรียกใช้งานเว็บเซอร์วิสได้จึงเป็นคำตอบของปัญหาเหล่านี้ เพื่อให้ข้อมูลเป็นไปในแนวทางเดียวกัน ลดความซ้ำซ้อนของข้อมูล เจ้าหน้าที่ดูแลระบบจัดการเพียงข้อมูลของตนเองรับผิดชอบ และข้อมูลเป็นปัจจุบันสามารถยืนยันความถูกต้องได้

การสร้างเว็บไซต์เพื่อให้เชื่อมต่อกับระบบ API

การทำให้เว็บไซต์เชื่อมต่อกับ API นั้นผู้พัฒนาสามารถพัฒนาระบบได้หลากหลายวิธีตามที่คุณพัฒนานัดเพราะ API ที่มหาวิทยาลัยใช้นั้นอยู่ในรูปแบบที่เป็นมาตรฐานสากล โดยใช้ภาษา JSON ในการเชื่อมโยง ซึ่งผู้พัฒนาระบบสามารถเลือกใช้ภาษาใดก็ได้ที่สามารถเชื่อมโยงกับ JSON ได้ และสามารถเชื่อมโยงผ่าน HTTP (Hypertext Transfer Protocol) ได้ ซึ่งในคู่มือนี้จะขอใช้ AngularJS ในการเชื่อมโยงเนื่องจากเป็นภาษา JavaScript ที่สามารถแทรกเข้ากับโค้ด HTML ได้ง่าย และไม่ยุ่งยากในการใช้งานนัก

ภาษาที่ใช้ในการพัฒนา

1. HTML
2. JavaScript
3. ฯลฯ

เงื่อนไขการเขียนโปรแกรมใช้งาน API ผ่าน AngularJS

AngularJS เป็น JavaScript Framework ที่ทำงานในฝั่ง Client การเขียน AngularJS นั้นสามารถเขียนในไฟล์ HTML ได้โดยท่านจะใช้หรือไม่ใช้ภาษา PHP ในการเขียนร่วมก็ได้ ซึ่งโดยปกติไฟล์ HTML ที่เขียนได้ ท่านสามารถเปิดจากเครื่องที่ท่านพัฒนาได้ทันที แต่เมื่อท่านต้องการเรียก API อาจมีเงื่อนไขเพิ่มเติมคือ ฝั่ง API Server นั้นจะต้องการข้อมูล Header จากท่าน ซึ่งหากท่านเปิดหน้าเพจโดยการเปิดจากไฟล์โดยตรงจะทำให้เพจนั้นไม่มีข้อมูล Header ส่งผลให้ไม่สามารถเรียก API ได้ หากต้องการเรียก API ท่านจำเป็นต้องเปิดไฟล์ผ่าน Webserver เพื่อให้ไฟล์นั้นมี Header ส่งไปให้ API Server ได้ ซึ่ง Webserver ท่านจะใช้จากการจำลองขึ้นในเครื่องผู้พัฒนา หรือจากเครื่อง Server ก็ได้

เริ่มต้นการพัฒนาเว็บเพจด้วยภาษา HTML และ AngularJS

การเริ่มใช้งาน AngularJS นั้น จำเป็นต้องใช้ชุดโค้ดเพื่อเริ่มทำงาน โดยมีโครงสร้างดังภาพ

```
<html ng-app="kpru">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body ng-controller="mainController">

<table border="1">
  <tr>
    <th>รหัส</th>
    <th>ชื่อ</th>
    <th>สกุล</th>
    <th>อายุ</th>
  </tr>
  <tr ng-repeat="row in rows">
    <td>{{row.id}}</td>
    <td>{{row.firstname}}</td>
    <td>{{row.lastname}}</td>
    <td>{{row.age}}</td>
  </tr>
</table>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
<script>
  var app = angular.module("kpru", []);
  app.controller("mainController", function($scope, $http) {
    var url = "localhost/api/index.php";
    $http.get(url).then(function(respond){
      $scope.rows = respond;
    });
  });
</script>
</body>
</html>
```

1. การตั้งชื่อ App โดยใช้คำสั่ง ng-app ในโค้ด HTML

```
<html ng-app="kpru">
```

2. การตั้งชื่อ Controller โดยการใช้คำสั่ง ng-controller ในโค้ด HTML

```
<body ng-controller="mainController">
```

3. การวนลูปเพื่อสร้างชุดโค้ดจากจำนวน Array ของข้อมูล ใช้คำสั่ง ng-repeat ในโค้ด HTML โดยรูปแบบการเรียกใช้คำสั่งคือ “ตัวแปรรับค่า in ชื่อตัวแปรที่เก็บข้อมูล Array” โดยโปรแกรมจะนำข้อมูลใน Array มาใส่ในตัวแปรรับค่าทีละ 1 ค่า และจะลูปจนกว่าข้อมูลใน Array จะหมด

```
<tr ng-repeat="row in rows">
```


4. การเรียกใช้ข้อมูลเพื่อให้แสดงค่าในหน้าเพจ การเรียกข้อมูลของ AngularJS ในส่วนโค้ด HTML นั้น จะอยู่ในรูปแบบวงเล็บปีกกาเปิดและปิดอย่างละ 2 อัน ({{ และ }}) โดยชื่อตัวแปรที่ใส่ด้านในจะไม่มี \$scope ซึ่งต่างจากการเรียกตัวแปรในส่วนของ JavaScript

```
{{row.id}}
```

5. การเรียกใช้งานชุดโค้ด AngularJS เพื่อทำงานในเพจ ผู้พัฒนาสามารถโหลดชุดโค้ดไว้ใน Server ของท่านแล้วเรียกผ่านโดเมนของท่านโดยตรงได้เช่นกัน

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
```

6. การกำหนดการทำงานของโปรแกรมในฝั่ง JavaScript จะมีรูปแบบ ดังนี้ การประกาศเรียกใช้ App และการเรียกใช้ Controller ซึ่งเราจะสามารถเขียนเงื่อนไขการทำงานของเพจนั่นๆ ภายใต Controller นี้

```
<script>
  var app = angular.module("kpru", []);
  app.controller("mainController", function($scope, $http) {

    // ส่วนโค้ดเพื่อกำหนดการทำงานของโปรแกรม

  });
}</script>
```

การใช้ AngularJS เชื่อมโยง API เพื่อนำข้อมูลมาแสดงผล

คู่มือนี้ขอยกตัวอย่างการเชื่อมโยงข้อมูลสังกัดหน่วยงานภายในมหาวิทยาลัยราชภัฏกำแพงเพชร จาก API ระบบ MIS เพื่อนำมาแสดงผล โดยมี URL สำหรับเชื่อมโยงข้อมูลสังกัดคือ <https://mis.kpru.ac.th/api/OrganizationList> ท่านสามารถพิมพ์ตามได้ ดังภาพ

```
<html ng-app="KPRU">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body ng-controller="mainController">

<table border="1" >
  <tr>
    <th>รหัส</th>
    <th>ชื่อสังกัด</th>
  </tr>
  <tr ng-repeat="dataRow in dataArray">
    <td>{{dataRow.organization_id}}</td>
    <td>{{dataRow.organization_name_tha}}</td>
  </tr>
</table>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
<script>
  var app = angular.module("KPRU", []); //กำหนด App ที่ทำงานด้วย
  app.controller("mainController", function($scope, $http) { //กำหนด Controller และเรียกฟังก์ชันที่ต้องการ
    ใช้
    var url = "https://mis.kpru.ac.th/api/OrganizationList"; //กำหนด API URL ที่ต้องการเรียกข้อมูล
    $http.get(url).then(function(respond){
      $scope.dataArray = respond.data; //เมื่อรับค่าจาก API แล้วให้นำไปใส่ในตัวแปร dataArray
    });
  });
</script>
</body>
</html>
```

เมื่อทดลองเข้าเพจ จะแสดงข้อมูลสังกัดตามที่ API ส่งมา ดังภาพ

รหัส	ชื่อสังกัด
0000000001	คณะครุศาสตร์
0000000002	คณะมนุษยศาสตร์และสังคมศาสตร์
0000000003	คณะวิทยาศาสตร์และเทคโนโลยี
0000000004	คณะวิทยาการจัดการ
0000000005	คณะเทคโนโลยีอุตสาหกรรม
0000000006	สำนักงานอธิการบดี
0000000007	สำนักส่งเสริมวิชาการและงานทะเบียน
0000000008	สำนักวิทยบริการและเทคโนโลยีสารสนเทศ

การเชื่อมโยง API โดยการส่งข้อมูลในรูปแบบ GET

จากหัวข้อที่แล้ว ได้ทำการเชื่อมข้อมูลโดยการเรียกเพียงชื่อ API โดยไม่มีการส่งค่าใดๆเพื่อนำไปสร้างเงื่อนไข แต่ด้วยบางเหตุการณ์มีความจำเป็นต้องส่งค่าบางอย่างเพื่อให้ API นำไปสร้างเงื่อนไขเพื่อส่งข้อมูลกลับได้ตรงตามความต้องการของผู้ใช้งานระบบ สำหรับหัวข้อนี้จะพูดถึงการส่งค่าไปกับ URL ของ API เพื่อนำไปสร้างเงื่อนไข หรือเรียกว่าการส่งค่าในรูปแบบ GET การส่งค่าในรูปแบบ GET นี้ เหมาะสำหรับการส่งค่าที่ไม่เป็นความลับ เป็นค่าทั่วไปที่ไม่ต้องการความปลอดภัยมากนัก เนื่องจากการส่งค่าในรูปแบบ GET สามารถถูกดักจับข้อมูลได้ง่าย จึงไม่เหมาะกับการส่งข้อมูลที่เป๊ะความลับ หรือข้อมูลส่วนตัว รวมถึงการส่งข้อมูลที่มีปริมาณไม่มากนัก เนื่องจากการส่งข้อมูลไปบน URL หากยาวเกินไปอาจทำให้ข้อมูลที่ API Server ได้รับนั้นอาจไม่ครบถ้วน

คู่มือนี้จึงขอยกตัวอย่าง API ของระบบ KPRU MIS ซึ่งเป็น API แสดงข้อมูลบุคลากรโดยต้องส่ง ID ของสังกัดหน่วยงานไป จะได้รายชื่อบุคลากรภายในหน่วยงานนั้นออกมา ซึ่ง API นี้ จะตอบค่ากลับมาโดยแยกรายชื่อบุคลากรสายสอนและสายสนับสนุน โดยมี URL ของ API คือ <https://mis.kpru.ac.th/api/EmployeeInOrgClassify/> (ID ของสังกัด) เช่นหากต้องการข้อมูลรายชื่อบุคลากรจากหน่วยงานรหัส “0000000007” จะได้ URL ดังนี้ “<https://mis.kpru.ac.th/api/EmployeeInOrgClassify/0000000007>” ซึ่งเมื่อทดลองเชื่อมต่อ API ผ่านเว็บเบราว์เซอร์โดยตรงแล้วจะได้ข้อมูลดังภาพ

```
{
  Teacher: [ ],
  Staff: [
    - {
      employee_id: "000474",
      prename_full_tha: "นางสาว",
      first_name_tha: "กนกกร",
      last_name_tha: "ทองคำ",
      employee_name: "นางสาวกนกกร ทองคำ",
      office_phone: "055706547",
      work_status_name: "ทำงานปกติ",
      employee_type_name: "พนักงานราชการ",
      sunit_name: "สำนักส่งเสริมวิชาการและงานทะเบียน",
      position_rank_name: "พนักงานราชการ",
      picture: "https://mis.kpru.ac.th/images/pic_emp_50/000474.jpg"
    },
    - {
      employee_id: "000335",
      prename_full_tha: "นาย",
      first_name_tha: "คมกริช",
      last_name_tha: "กลิ่นอาจ",
      employee_name: "นายคมกริช กลิ่นอาจ",
      office_phone: "055706555-1479",
      work_status_name: "ทำงานปกติ",
      employee_type_name: "พนักงานมหาวิทยาลัย",
    }
  ]
}
```










จากข้อมูลที่ API ส่งกลับมา จะเห็นว่า Array อยู่ 2 ชุดคือ Teacher และ Staff ซึ่งท่านสามารถเขียนเพจเพื่อรับข้อมูลได้ โดยพิมพ์ตาม ดังภาพ

```
<html ng-app="KPRU">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body ng-controller="mainController">

<table border="1" >
  <tr>
    <th>ภาพบุคลากร</th>
    <th>ชื่อ-สกุล</th>
    <th>ชื่อสังกัด</th>
    <th>ตำแหน่ง</th>
  </tr>
  <tr>
    <td colspan="4" ng-if="dataArrayTeacher.length > 0">รายชื่ออาจารย์</td>
  </tr>
  <tr ng-repeat="dataRow in dataArrayTeacher">
    <td>
      
    </td>
    <td>{{dataRow.employee_name}}</td>
    <td>{{dataRow.sunit_name}}</td>
    <td>{{dataRow.position_rank_name}}</td>
  </tr>
  <tr>
    <td colspan="4" ng-if="dataArrayStaff.length > 0">รายชื่อเจ้าหน้าที่</td>
  </tr>
  <tr ng-repeat="dataRow in dataArrayStaff">
    <td>
      
    </td>
    <td>{{dataRow.employee_name}}</td>
    <td>{{dataRow.sunit_name}}</td>
    <td>{{dataRow.position_rank_name}}</td>
  </tr>
</table>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
<script>
  var app = angular.module("KPRU", []);
  app.controller("mainController", function($scope, $http) {
    var url = "https://mis.kpru.ac.th/api/EmployeeInOrgClassify/0000000005";
    $http.get(url).then(function(respond){
      $scope.dataArrayTeacher = respond.data.Teacher; //รับค่ากลุ่ม Teacher จาก API
      $scope.dataArrayStaff = respond.data.Staff; //รับค่ากลุ่ม Staff จาก API
    });
  });
</script>
</body>
</html>
```

จากภาพจะเห็นว่าได้มีการส่งข้อมูล ID ของหน่วยงานต่อท้าย URL ไปด้วย ซึ่งอยู่ในรูปแบบ GET และในโค้ดมีส่วนที่เพิ่มเติมขึ้นมาคือส่วนของ respond.data ได้เพิ่มชื่อชุดข้อมูล Array ต่อท้ายด้วย เพื่อระบุว่าให้ตัวแปรนั้นๆ รับค่าจาก Array ชุดใด รวมถึงสิ่งที่เพิ่มขึ้นมาอีกหนึ่งส่วนคือ ng-if ซึ่งเป็นคำสั่งของ AngularJS เพื่อใช้สร้างเงื่อนไขในการแสดงข้อมูลในโค้ด HTML ดังโค้ดตัวอย่าง ได้ใช้คำสั่ง ng-if="dataArrayTeacher.length > 0" เพื่อเช็คค่าหากตัวแปรชื่อ dataArrayTeacher มีจำนวนอาเรย์มากกว่าศูนย์ ให้แสดงชุดคำสั่ง HTML นั้นๆ เป็นต้น และเมื่อทดลองเข้าเพจ ได้จะดังภาพ

ภาพบุคลากร	ชื่อ-สกุล	ชื่อสังกัด	ตำแหน่ง
รายชื่ออาจารย์			
	นางสาวกนกวรรณ เขียววัน	สาขาวิชาเทคโนโลยีคอมพิวเตอร์	อาจารย์
	นายจตุรงค์ ชงชัย	สาขาวิชาเทคโนโลยีคอมพิวเตอร์	อาจารย์
	นายจักรพันธ์ ชงทอง	สาขาวิชาเทคโนโลยีก่อสร้าง	อาจารย์
	นายจารุกิตต์ พิบูลนถดม	สาขาวิชาเทคโนโลยีพลังงาน	อาจารย์
	ว่าที่ร้อยตรีชุดิเดช หันจันทร์	สาขาวิชาเทคโนโลยีคอมพิวเตอร์	อาจารย์ประจำตามสัญญาจ้าง
รายชื่อเจ้าหน้าที่			
	นางสาวศัลลียา ปัญญาอด	สำนักงานคณบดีคณะเทคโนโลยีอุตสาหกรรม	เจ้าหน้าที่บริหารงานทั่วไป
	นายเจด็จ สิ้นพากร	สาขาวิชาเทคโนโลยีโยธา	พนักงานปฏิบัติการ
	นางสาวขมภู สร้อยเกลียว	สำนักงานคณบดีคณะเทคโนโลยีอุตสาหกรรม	นักการภารโรง
	นายชาญชัย กาญจนจันทร์	สำนักงานคณบดีคณะเทคโนโลยีอุตสาหกรรม	นักการภารโรง
	นายชุติพงศ์ วันเสาร์	สำนักงานคณบดีคณะเทคโนโลยีอุตสาหกรรม	นักวิชาการคอมพิวเตอร์
	ว่าที่ร้อยตรีธิพน บุญมา วงษ์กันหา	สำนักงานคณบดีคณะเทคโนโลยีอุตสาหกรรม	นักวิชาการคอมพิวเตอร์

การประยุกต์ใช้ API เพื่อเชื่อมโยงข้อมูลในรูปแบบ GET

การรับข้อมูลจาก API สามารถนำมาประยุกต์ใช้ในรูปแบบต่างๆ ได้หลากหลายไม่แตกต่างจากการเชื่อมต่อกับฐานข้อมูลโดยตรง ในคู่มือนี้จะขอยกตัวอย่างการประยุกต์ใช้โดยการนำการเรียกข้อมูลหน่วยงานจาก API ของ <https://mis.kpru.ac.th/api/OrganizationList> มาแสดงเพื่อให้ผู้ใช้งานเลือกหน่วยงาน จากนั้นนำ ID ของหน่วยงานไปเรียกข้อมูลบุคลากรจาก API ของ <https://mis.kpru.ac.th/api/EmployeeInOrgClassify/...???...> เพื่อนำข้อมูลบุคลากรมาแสดง

```
<html ng-app="KPRU">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body ng-controller="mainController">
<select ng-model="orgModel" ng-change="selectOrg()">
  <option value="" hidden selected>--กรุณาเลือกหน่วยงาน--</option>
  <option ng-repeat="dataRow in getOrg" value=
"{{dataRow.organization_id}}">{{dataRow.organization_name_tha}}</option>
</select>
<table border="1" >
  <tr>
    <th>ภาพบุคลากร</th>
    <th>ชื่อ-สกุล</th>
    <th>ชื่อสังกัด</th>
    <th>ตำแหน่ง</th>
  </tr>
  <tr>
    <td colspan="4" style="background: green; color: white;" ng-if="dataArrayTeacher.length > 0">
รายชื่ออาจารย์</td>
  </tr>
  <tr ng-repeat="dataRow in dataArrayTeacher">
    <td>
      
    </td>
    <td>{{dataRow.employee_name}}</td>
    <td>{{dataRow.sunit_name}}</td>
    <td>{{dataRow.position_rank_name}}</td>
  </tr>
  <tr>
    <td colspan="4" style="background: blue; color: white;" ng-if="dataArrayStaff.length > 0">รายชื่อ
เจ้าหน้าที่</td>
  </tr>
  <tr ng-repeat="dataRow in dataArrayStaff">
    <td>
      
    </td>
    <td>{{dataRow.employee_name}}</td>
    <td>{{dataRow.sunit_name}}</td>
    <td>{{dataRow.position_rank_name}}</td>
  </tr>
</table>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
```

```

<script>
var app = angular.module("KPRU", []);
app.controller("mainController", function($scope, $http) {
var urlOrg = "https://mis.kpru.ac.th/api/OrganizationList";
$http.get(urlOrg).then(function(response){
$scope.getOrg = response.data;
});
$scope.selectOrg = function(){
var url = "https://mis.kpru.ac.th/api/EmployeeInOrgClassify/" + $scope.orgModel;
$http.get(url).then(function(response){
$scope.dataArrayTeacher = response.data.Teacher;
$scope.dataArrayStaff = response.data.Staff;
});
}
});
</script>
</body>
</html>

```

จากภาพ จะเห็นว่าได้เพิ่มในส่วนของ select tag ขึ้นมา เพื่อใช้แสดงรายชื่อหน่วยงาน และจะเห็นคำสั่งเพิ่มเติมคือ ng-change คำสั่งนี้จะทำงานเมื่อข้อมูลใน tag นั้นมีการเปลี่ยนแปลง เช่น เมื่อผู้ใช้เปลี่ยนหน่วยงาน 1 ครั้ง คำสั่งนี้จะถูกเรียกใช้งาน 1 ครั้ง ในส่วนของโค้ด AngularJS จะมีกระบวนการทำงานคือ เมื่อเพจเริ่มทำงานจะทำการเรียกข้อมูลหน่วยงานจาก API มาแสดงใน select tag และเมื่อผู้ใช้งานเลือกหน่วยงานแล้ว ng-change จะเรียกใช้ function selectOrg เพื่อให้เรียกข้อมูลบุคลากรจาก API โดยใช้ค่า ID ที่ผู้ใช้เลือกจาก select tag และนำไปแสดงในตาราง เมื่อทดลองเข้าเพจและเลือกหน่วยงาน จะแสดงผลดังภาพ

คณะวิทยาการจัดการ			
ภาพบุคลากร	ชื่อ-สกุล	ชื่อสังกัด	ตำแหน่ง
รายชื่ออาจารย์			
	นางสาวกนิษฐา ศรีภิรมย์	สาขาวิชาการจัดการทั่วไป	อาจารย์
	นายกรินทร์ เจริญสุวรรณ	สาขาวิชาการท่องเที่ยวและการโรงแรม	อาจารย์
	นางสาวคุณัญญา เบญจวรรณ	สาขาวิชาการเงินและการธนาคาร	อาจารย์
	นายจักรพันธ์ หว้าอ้อย	สาขาวิชาคอมพิวเตอร์ธุรกิจ	อาจารย์
	นายจิระ ประสพธรรม	สาขาวิชาการตลาด	อาจารย์
	นายจักรชัย อินทรประพันธ์	สาขาวิชาคอมพิวเตอร์ธุรกิจ	อาจารย์
	นางสาวชยานันท์ ศิริกิจเสถียร	สาขาวิชาเศรษฐศาสตร์การเงินการคลัง	อาจารย์
	นางสาวชลธิชา แสงงาม	สาขาวิชาคอมพิวเตอร์ธุรกิจ	อาจารย์
	นางชาลี ตระกูล	สาขาวิชาเศรษฐศาสตร์การเงินการคลัง	ผู้ช่วยศาสตราจารย์

การเชื่อมต่อ API ในรูปแบบ POST

การเชื่อมต่อ API ในรูปแบบ POST นั้นจะมีลักษณะการเชื่อมต่อที่คล้ายกับรูปแบบ GET แต่การส่งข้อมูลในรูปแบบ POST นั้นเหมาะกับการส่งข้อมูลจำนวนหลายค่า หรือเป็นข้อมูลที่มีความสำคัญสูง โดยการส่งข้อมูลในรูปแบบ POST นั้นเป็นการรวมค่าที่ต้องการส่งทั้งหมดให้อยู่ในรูปแบบ Array 1 ชุด แล้วจึงทำการส่งไป ซึ่งมีรูปแบบการส่งข้อมูลดังตัวอย่างโค้ด ดังภาพ

```
<html ng-app="KPRU">
<head>
  <meta charset="UTF-8">
</head>
<body ng-controller="mainController">
<table border="1" >
  <tr>
    <th>id</th>
    <th>ชื่อ-สกุล</th>
    <th>ชื่อสังกัด</th>
    <th>จัดการ</th>
  </tr>
  <tr ng-repeat="dataRow in dataArray">
    <td>{{dataRow.id}}</td>
    <td>{{dataRow.name}}</td>
    <td>{{dataRow.org_name}}</td>
    <td><a href="" ng-click="deleteData(dataRow)">ลบ</a></td>
  </tr>
</table>
<form ng-submit="insertData()">
  <p><strong>เพิ่มข้อมูล</strong></p>
  <p>ชื่อ - สกุล : <input type="text" placeholder="กรณารอก ชื่อ - สกุล" ng-model="dataForm.name"></p>
  <p>หน่วยงาน : <select ng-model="dataForm.org_name" >
    <option value="-" hidden selected>--กรุณาเลือกหน่วยงาน--</option>
    <option ng-repeat="dataRow in getOrg"
value="{{dataRow.organization_name_tha}}">{{dataRow.organization_name_tha}}</option>
  </select></p>
  <button type="submit">บันทึก</button>
</form>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
<script>
  var app = angular.module("KPRU", []);
  app.controller("mainController", function($scope, $http) {
    $scope.loadPage = function(){
      var url = "https://mua.kpru.ac.th/FrontEnd_MIS/testAPI/ShowAllData";
      $http.get(url).then(function(respond){ $scope.dataArray = respond.data; });
    }
    $scope.insertData = function(){
      var url = "https://mua.kpru.ac.th/FrontEnd_MIS/testAPI/PostInsert";
      $http.post(url,$scope.dataForm).then(function(respond){ $scope.loadPage(); });
    }
    $scope.deleteData = function(data) {
      var url = "https://mua.kpru.ac.th/FrontEnd_MIS/testAPI/GetDelete/" + data.id;
      $http.get(url).then(function(respond){ $scope.loadPage(); });
    }
    var urlOrg = "https://mis.kpru.ac.th/api/OrganizationList";
    $http.get(urlOrg).then(function(respond){
      $scope.getOrg = respond.data;
    });
    $scope.loadPage();
  });
</script>
</body>
</html>
```


จากภาพ จะเห็นได้ว่าสามารถประยุกต์ใช้ API ได้หลากหลายรูปแบบในหน้าเพจเดียวกัน เช่น การเรียกแบบ GET โดยไม่ส่งค่าอะไรไป เรียกแบบ GET พร้อมแนบค่าไป และเรียกแบบ POST พร้อมส่งค่าไป หรือแม้แต่จะเรียกแบบ POST พร้อมส่งค่าไปและแนบค่า GET ตามท้าย URL ไปด้วยก็สามารถทำได้เช่นเดียวกัน

จากโค้ดข้างต้น จะเห็นในว่ามีคำสั่ง AngularJS เพิ่มขึ้นมา คือ ng-click และ ng-submit จะทำงานต่อเมื่อกดลิงก์ หรือกดปุ่ม Submit โดยในตัวอย่างจะให้เรียกฟังก์ชันที่เขียนไว้ และคำสั่งที่เพิ่มขึ้นมาสำหรับการเรียก API รูปแบบ POST คือ \$http.post เป็นการเรียกใช้ API โดยการส่ง Method POST แล้วตามด้วย URL และค่า Array ที่ต้องการส่งไป เป็นต้น เมื่อเปิดหน้าเพจ จะแสดงดังภาพ

id	ชื่อ-สกุล	ชื่อสังกัด	จัดการ
1	นายทดสอบ คีย์ข้อมูล	คณะเทคโนโลยีอุตสาหกรรม	ลบ
3	น.ส.เทส คีย์	คณะวิทยาการจัดการ	ลบ

เพิ่มข้อมูล

ชื่อ - สกุล :

หน่วยงาน :