องค์ความรู้ด้านการกำหนด API เพื่อส่งข้อมูลในรูปแบบของ JSON

โครงการแผนปฏิบัติการจัดการความรู้ประจำปีงบประมาณ พ.ศ. 2561

นายคณาพล อมรรัตนเกศ

สถานส่งเสริมและพัฒนาระบบสารสนเทศเพื่อการจัดการ มหาวิทยาลัยเทคโนโลยีสุรนารี | โทร 0-4422-4023

สารบัญ

องค์ความรู้ด้านการกำหนด API เพื่อส่งข้อมูลในรูปแบบของ JSON

พื้นฐานที่ควรรู้มาก่อน

1. HTML, CSS เบื้องต้น
2. มีความรู้พื้นฐานการเขียนโปรแกรมภาษาใดภาษาหนึ่ง เช่น JAVA, C, VB.NET, JavaScript
3. มีความรู้พื้นฐานภาษา SQL เช่น INSERT, UPDATE, DELETE, SELECT และชนิดข้อมูล

การพัฒนา JSON API

การพัฒนาเว็บแอพพลิเคชั่นในปัจจุบันเปลี่ยนแปลงไปจากเมื่อก่อนอย่างมาก เนื่องจากการเติบโตอย่างก้าวกระโดดของอุปกรณ์โทรศัพท์เครื่องที่ (Mobile device) โดยเฉพาะการการที่สามารถเข้าถึงอินเตอร์เน็ตได้ จึงมีการพัฒนาให้เว็บแอพพลิเคชั่นสามารถแสดงผลได้อย่างรวดเร็วและได้ทุกขนาดหน้าจอ ในปัจจุบันมีผู้คิดค้นเครื่องมือในการพัฒนาและออกแบบสถาปัตยกรรมที่เหมาะสมกับโมบายแอพพลิเคชั่นโดยเฉพาะ

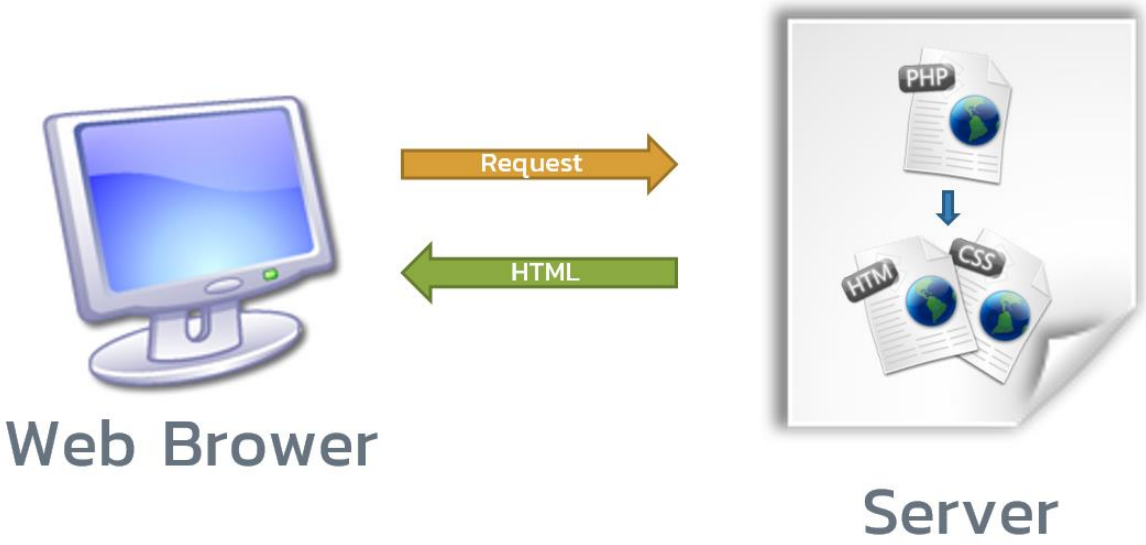
หลักของการพัฒนา API คือการใช้แนวคิดด้านสถาปัตยกรรมเชิงเซอร์วิส SOA (Service-Oriented Architecture) เพื่อประยุกต์ใช้ออกแบบที่จะดึงความสามารถศักยภาพให้เว็บแอพพลิเคชั่นสามารถตอบโจทย์ปัญหาในยุคที่ไคลเอนต์ (Client) มีจำนวนมหาศาลและหลากหลาย หากวันนี้ถ้ายังคงใช้แนวคิดและวิธีการแบบเดิม ๆ เชื่อว่าเว็บแอพพลิเคชั่นอย่าง Facebook คงยังไม่สามารถรองรับผู้ใช้งานมหาศาลจำนวนกว่าหนึ่งพันห้าร้อยล้านจากคนทั่วโลกได้

API ย่อมาจาก Application Programming Interface คือ ช่องทางการแลกเปลี่ยนข้อมูลระหว่างเว็บไซต์หนึ่งไปยังอีกเว็บไซต์หนึ่ง หรือเป็นการเชื่อมต่อระหว่างผู้ใช้งาน กับ Server หรือจาก Server เชื่อมต่อไปหา Server โดยข้อมูลที่ส่งหากันนั้นจะอยู่ในรูปแบบต่าง ๆ อาทิเช่น XML, JSON ซึ่งในกรณีศึกษานี้จะกล่าวถึง JSON API

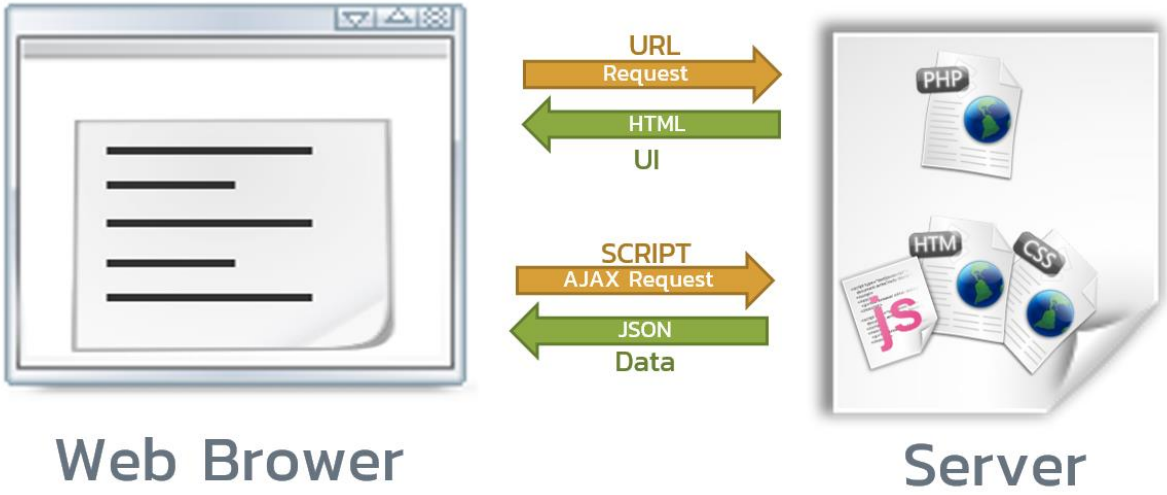
JSON หรือ Java Script Object Notation เป็นรูปแบบสำหรับแลกเปลี่ยนข้อมูลคอมพิวเตอร์ ที่ทำให้ JavaScript สามารแลกเปลี่ยนข้อมูลกับ Server ได้ ข้อมูลจะอยู่ในรูปแบบข้อความธรรมดาที่ทั้งมนุษย์และโปรแกรมคอมพิวเตอร์สามารถอ่านเข้าใจได้ ในปัจจุบัน JSON นิยมใช้กับทั้งเว็บแอพพลิเคชั่นลและแอพพลิเตชั่นบนอุปกรณ์เคลื่อนที่

การเขียน PHP ปกติกับการเขียน JSON API

PHP แบบเดิม ผู้ใช้จะเรียกเว็บไซต์ไปที่ Server แล้ว Server ทำการประมวนผลด้วย PHP Complier เป็น HTML แล้วส่งกลับไปที่หน้าจอผู้ใช้



PHP แบบ JSON API ผู้ใช้จะเรียกหน้าเว็บไซต์ไปยัง Server แล้ว Server ทำการส่งหน้า HTML กลับมาเมื่อแสดงผลหน้าเว็บไซต์เรียบร้อย เบราเซอร์จะทำการขอข้อมูลไปที่ Server อีกครับ Server จะส่งข้อมูลกลับมาในรูปแบบของ JSON



ฟังก์ชันที่จำเป็นในการเขียน API

ฟังก์ชันใน PHP ที่จำเป็นต่อการทำ JSON API ที่สำคัญมีอยู่ด้วยกัน 2 ฟังก์ชัน คือ

1. header("Content-Type: application/json; charset=UTF-8");
   - Content-Type: application/json เป็นตัวกำหนดให้ เบราเซอร์อ่านข้อมูลเป็นรูปแบบ JSON (หากไม่มี เบราเซอร์จะมองเป็น HTML ทำให้การเขียกใช้ข้อมูลเกิดข้อผิดพลาด)
   - charset=UTF-8 เป็นเข้ารหัสที่รองรับอักษรภาษาไทย (หรือเข้ารหัสตามข้อมูลที่เราต้องการส่ง)
2. json_encode({value}) เป็นการทำข้อมูล {value} มาทำให้อยู่ในรูปแบบ JSON

ขั้นตอนการเขียน API อยู่สามารถแบ่งเป็น 3 ขั้นตอนหลัก

ขั้นตอนแรกเป็นการกำหนดประเภทของข้อมูลที่แสดงผลให้อยู่ในรูปแบบของ JSON ด้วยฟังก์ชัน header("Content-Type: application/json; charset=UTF-8"); ขั้นตอนนี้จะเป็นการตั้งค่าต่าง ๆ ให้กับ API

ขั้นตอนที่สอง คือขั้นตอนของการทำข้อมูล อาจได้จากการดึกข้อมูลจากฐานข้อมูลหรือการคำนวณต่าง ๆ แล้วเก็บไว้ในตัวแปรใด ๆ 1 ตัว เพื่อเตรียมเข้ารหัส

ขั้นตอนที่สาม เป็นขั้นตอนการเข้ารหัสข้อมูลให้อยู่ในรูปแบบของ JSON ด้วยฟังก์ชัน json_encode แล้วทำการแสดงผล (echo)

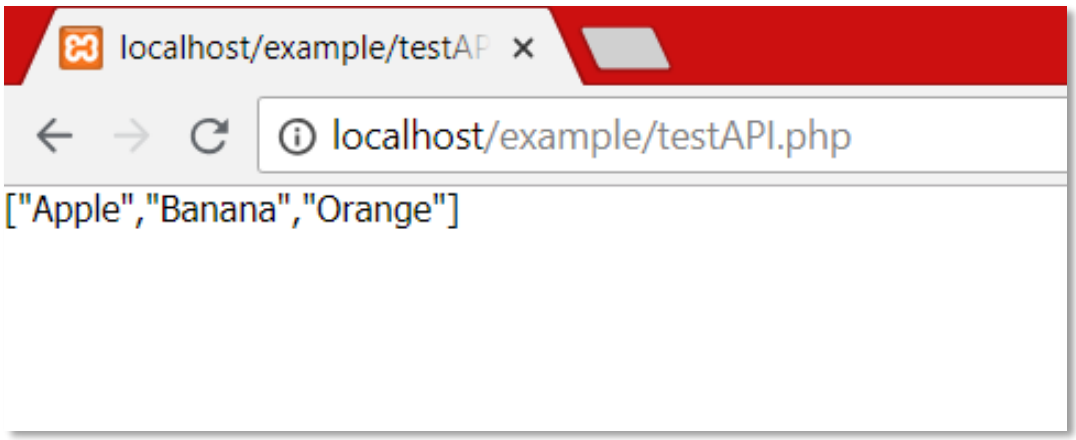
ตัวอย่างที่ 1 เป็นการส่งข้อมูลเป็น Array

testAPI.php

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

$data = array("Apple", "Banana", "Orange");

echo json_encode($data);
?>
```



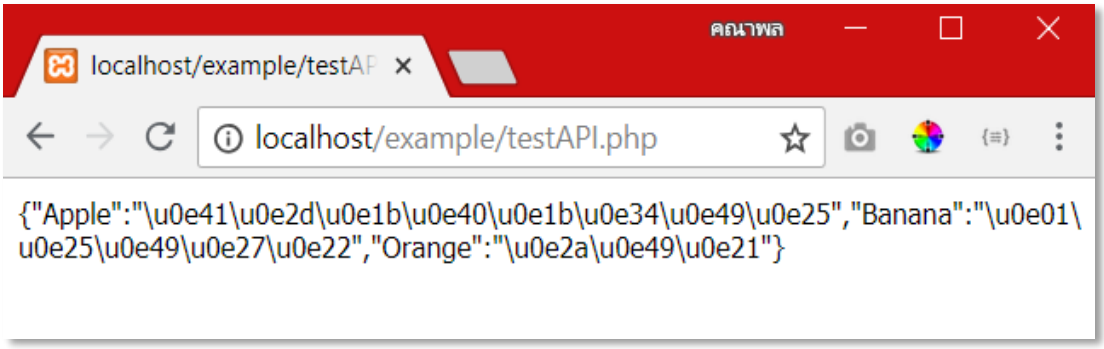
ตัวอย่างที่ 2 เป็นการส่งข้อมูลเป็น Object

testAPI.php

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

$data = array("Apple"=>"แอปเปิ้ล", "Banana"=>"กล้วย", "Orange"=>"ส้ม");

echo json_encode($data);
?>
```



กำหนดรูปแบบการส่งข้อมูล

ในกรณีที่มีผู้พัฒนาหลายคน เพื่อความเข้าใจที่ตรงกันเราจึงต้องมีการกำหนดรูปแบบของการส่งข้อมูล ซึ่งจะช่วยจัดระเบียบและลดความซับซ้อนของการเขียนโปรแกรมเพื่อดึงข้อมูลลงได้

ในหนึ่งชุดข้อมูลประกอบไปด้วย

1. instance คือ ส่วนที่เก็บข้อมูลที่นำไปใช้งาน กรณีที่มีข้อมูลเพียงรายการเดียว จะมีชนิดข้อมูลเป็น Object หากมีมากกว่า 1 รายการ จะมีชนิดข้อมูลเป็น Array
2. message หรือ alert คือ ข้อความที่ API ส่งไปพร้อมกับชุดข้อมูล เพื่อสื่อสารกับผู้รับข้อมูล อาจจะมีหรือไม่มีก็ได้ มีชนิดข้อมูลเป็น ข้อความ
3. pagination มีชนิดข้อมูลเป็น Object คือ ส่วนเก็บข้อมูลในกรณีที่ส่งข้อมูลมามากกว่า 1 รายการ ประกอบไปด้วย
    - total จำนวนทั้งหมดของชุดข้อมูล มีชนิดข้อมูลเป็นตัวเลข
    - page หน้าปัจจุบัน มีชนิดข้อมูลเป็นตัวเลข
    - pageStart ลำดับแรกของรายการในหน้าปัจจุบัน มีชนิดข้อมูลเป็นตัวเลข
    - perPage จำนวนรายการในหนึ่งหน้า มีชนิดข้อมูลเป็นตัวเลข
4. date_now คือ ข้อมูลเวลา ณ ขณะที่ส่งข้อมูล

ทั้งนี้รูปแบบการส่งข้อมูลไม่ได้กำหนดตายตัวว่าต้องเป็นแบบที่กล่าวมาทั้งหมด ซึ่งผู้พัฒนาสามารถเพิ่ม Field อื่น ๆ ที่จำเป็นในการพัฒนาโปรแกรม อาทิเช่น

5. last_id ในกรณีทำการบันทึกข้อมูลใหม่ หากบันทึกสำเร็จจะทำการส่ง Primary key กลับไป
6. status ในกรณีที่มีการ Update หรือ Delete ข้อมูล จะส่งสถานะกลับมา เป็น true เมื่อกระทำการสำเร็จ หรือ false หากกระทำการไม่สำเร็จ
7. sql ใช้ในการเช็คข้อผิดพลาดของคำสั่ง SQL

ตัวอย่างการแสดงผลข้อมูล

ตัวอย่างรูปแบบข้อมูลที่ได้ กรณีข้อมูลมีมากกว่าหนึ่งรายการ


```
{
  "instance": Array[5][
    {
      "id": 1,
      "name": "Apple",
      "phone": "0811111111"
    },
    {⇔},
    {⇔},
    {⇔},
    {⇔}
  ],
  "pagination": {
    "total": 22,
    "page": 1,
    "perPage": 5
  },
  "date_now": "2018-07-19 14:54:20"
}
```


ตัวอย่างรูปแบบข้อมูลที่ได้ กรณีข้อมูลมีหนึ่งรายการ


```
{
  "instance": {
    "id": 1,
    "name": "Apple",
    "phone": "0811111111"
  },
  "date_now": "2018-07-19 15:01:41"
}
```


ตัวอย่างรูปแบบข้อมูลที่ได้ กรณีเพิ่มข้อมูลใหม่


```
{
    "last_id": 23,
    "date_now": "2018-07-19 15:18:09"
}
```


ตัวอย่างรูปแบบข้อมูลที่ได้ กรณีเพิ่มทำรายกการ*ไม่สำเร็จ*

ในกรณีที่ไม่ได้ผลลัพธ์ตามต้องการ ที่ไม่ได้มาจากข้อผิดพลาดของการเขียนรูปแบบโปรแกรมผิด (Syntax ERROR) หรือ ข้อผิดพลาดจากการเขียนคำสั่ง SQL เราสามารถทำให้ API ส่งข้อผิดพลาดนั้นกลับไปตรวจสอบได้ อาทิเช่น

8. เรียกที่อยู่ API ผิด

9. ส่งข้อมูลที่จำเป็นต่อการดึงข้อมูลไม่ครบ เช่น ต้องการข้อมูลสมาชิกลำดับที่ 10 (id=10) แต่ไม่ได้ส่งค่าลำดับไป Query

หรือ

ในกรณีที่มีข้อผิดพลาดจากการเขียนโปรแกรม (Syntax ERROR) หรือ ข้อผิดพลาดจากการเขียนคำสั่ง SQL ผิด API จะหยุดทำงานทันที ส่งผลให้ข้อมูลที่ส่งกลับมาไม่ได้อยู่ในรูปแบบของ JSON ดังนั้นผู้พัฒนาจะต้องมีความรอบคอบในการเขียนโค้ด

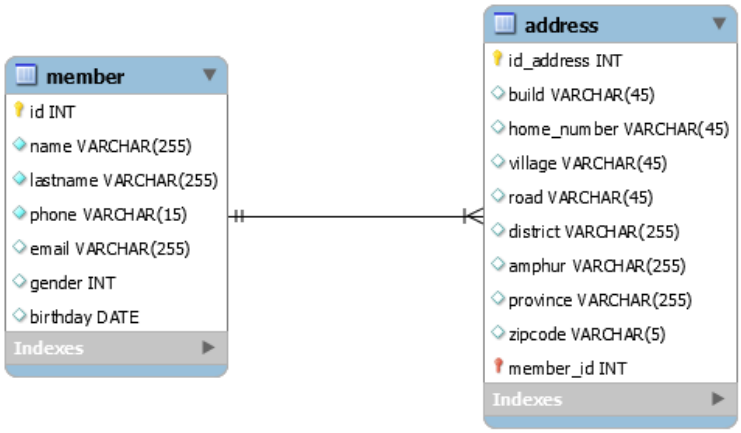
สร้างโฟลเดอร์สำหรับเก็บไฟล์ต่าง ๆ ดังนี้

1. สร้างโฟลเดอร์ชื่อ model สำหรับเก็บไฟล์ php ที่ใช้ในการคิวรี่ข้อมูล
    - สร้างโฟลเดอร์ชื่อ opdb สำหรับเก็บไฟล์ที่ใช้ติดต่อฐานข้อมูล
    - สร้างโฟลเดอร์ชื่อ main สำหรับเก็บฟังก์ชันกลางหรือไฟล์ใด ๆ ที่ถูกเรียกใช้ในหน้าอื่น ๆ ร่วมกัน
    - สร้างโฟลเดอร์ตามชื่อตาราง (Table) เพื่อเก็บไฟล์ php ที่ใช้ในการจัดการกับตารางนั้น ๆ
2. สร้างโฟลเดอร์ชื่อ controller
3. สร้างโฟลเดอร์ชื่อ view เก็บไฟล์ php ที่ใช้สำหรับการแสดงผล
    - สร้างโฟลเดอร์ ตามชื่อตาราง (Table) สำหรับเก็บไฟล์ php เพื่อใช้ในการแสดงผล
4. สร้างไฟล์ index.php

เริ่มต้นสร้าง API

ออกแบบโครงสร้างฐานข้อมูล

กำหนดกรณีศึกษาในการเขียน API คือ สร้างระบบฐานข้อมูลเพื่อเก็บข้อมูลสมาชิกสำหรับส่งสินค้า โดยสมาชิกหนึ่งคนมีที่อยู่ในการจัดส่งสินค้าได้มากกว่าหนึ่งที่ ดังภาพตัวอย่างความสัมพันธ์ของทั้งสองตาราง



สร้างฐานข้อมูลใน phpmyadmin

ทำการสร้างฐานข้อมูลชื่อ example_db

```
CREATE DATABASE example_db DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci
```

สร้างตาราง member เพื่อเก็บข้อมูลสมาชิก และตาราง address เก็บข้อมูลที่อยู่ของสมาชิก

```
CREATE TABLE IF NOT EXISTS `member` (
        `id` INT NOT NULL AUTO_INCREMENT,
        `name` VARCHAR(255) NOT NULL COMMENT 'ชื่อจริง',
        `lastname` VARCHAR(255) NOT NULL COMMENT 'นามสกุล',
        `phone` VARCHAR(15) NOT NULL COMMENT 'เบอร์โทรศัพท์',
        `email` VARCHAR(255) NULL COMMENT 'อีเมล',
        `gender` INT NULL COMMENT 'เพศ [1: ชาย, 2: หญิง]',
        `birthday` DATE NULL COMMENT 'วันเกิด',
        PRIMARY KEY (`id`))
ENGINE = InnoDB;
CREATE TABLE IF NOT EXISTS `address` (
        `id_address` INT NOT NULL AUTO_INCREMENT,
        `build` VARCHAR(255) NULL COMMENT 'อาคาร/ห้อง',
```

```
    `home_number` VARCHAR(45) NOT NULL COMMENT 'บ้านเลขที่',
    `village` VARCHAR(255) NULL COMMENT 'หมู่ที่',
    `road` VARCHAR(255) NULL COMMENT 'ถนน',
    `district` VARCHAR(255) NOT NULL COMMENT 'ตำบล',
    `amphur` VARCHAR(255) NOT NULL COMMENT 'อำเภอ',
    `province` VARCHAR(255) NOT NULL COMMENT 'จังหวัด',
    `zipcode` VARCHAR(5) NOT NULL COMMENT 'รหัสไปรษณีย์',
    `member_id` INT NOT NULL,
    PRIMARY KEY (`id_address`, `member_id`),
    INDEX `fk_address_member_idx` (`member_id` ASC),
    CONSTRAINT `fk_address_member`
        FOREIGN KEY (`member_id`)
        REFERENCES `member` (`id`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

ตาราง member

	#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
☐	1	**id** 🔑	int(11)			No	*None*	AUTO_INCREMENT	Change Drop More
☐	2	**name**	varchar(255)			No	*None*		Change Drop More
☐	3	**lastname**	varchar(255)			No	*None*		Change Drop More
☐	4	**phone**	varchar(15)			No	*None*		Change Drop More
☐	5	**email**	varchar(255)			Yes	*NULL*		Change Drop More
☐	6	**gender**	int(11)			Yes	*NULL*		Change Drop More
☐	7	**birthday**	date			Yes	*NULL*		Change Drop More

ตาราง address

	#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
☐	1	id_address	int(11)			No	None	AUTO_INCREMENT	Change Drop More
☐	2	build	varchar(45)			Yes	NULL		Change Drop More
☐	3	home_number	varchar(45)			No	None		Change Drop More
☐	4	village	varchar(45)			Yes	NULL		Change Drop More
☐	5	road	varchar(45)			Yes	NULL		Change Drop More
☐	6	district	varchar(255)			No	None		Change Drop More
☐	7	amphur	varchar(255)			No	None		Change Drop More
☐	8	province	varchar(255)			No	None		Change Drop More
☐	9	zipcode	varchar(5)			No	None		Change Drop More
☐	10	member_id	int(11)			No	None		Change Drop More

การเขียน API สำหรับตารางสมาชิก (member) ด้วยการเขียนแบบโครงสร้าง

เริ่มต้นสร้าง JSON API โดยยกกรณีศึกษา คือ การจัดการข้อมูลตารางสมาชิก (member) ทำการสร้างโฟลเดอร์ชื่อ member ไว้ในโฟลเดอร์ model โดยแบ่งไฟล์ตามลักษณะการทำงานของคำสั่ง SQL ดังนี้

5. memberInsert.php ทำหน้าที่เพิ่มข้อมูลใหม่ให้กับตารางสมาชิก
6. memberList.php ทำหน้าที่ดึงข้อมูลจากตารางสมาชิกมาครั้งละหลาย ๆ รายการ
7. memberShow.php ทำหน้าที่ดึงข้อมูลจากตารางสมาชิกมาครั้ง 1 รายการโดยการละบุคีย์หลักที่ใช้อ้างอิง
8. memberUpdate.php ทำหน้าที่ปรับปรุงข้อมูลสมาชิก โดยการละบุคีย์หลักที่ใช้อ้างอิงการ
9. memberDelete.php ทำหน้าที่ลบข้อมูลสมาชิก โดยการละบุคีย์หลักที่ใช้อ้างอิง

1. คำสั่ง INSERT ไฟล์ชื่อ memberInsert.php API สำหรับบันทึกข้อมูลลงฐานข้อมูล

```php
<?php
    error_reporting(E_ALL);
    ob_start();
    ini_set("display_errors", 1);
    date_default_timezone_set("Asia/Bangkok");
    header("Access-Control-Allow-Origin: *");
    header("Content-Type: application/json; charset=UTF-8");
    $servername = "localhost";
    $database = "example_db";
    $username = "root";
    $password = "";
    // Create connection
```

```
13. $conn = new mysqli($servername, $username, $password, $database);
14. $conn->set_charset("utf8");
15. $json = array();
16. // Check connection
17. if ($conn->connect_error) {
18.             $json['connect'] = ("Connection failed: " . $conn->connect_error);
19. }else{
20.             if (isset($_POST) && $_POST){
21.                         $field = array();
22.                         $sql = "SHOW FULL COLUMNS FROM member WHERE Extra!='auto_increment'
    ";
23.                         $excute = mysqli_query($conn, $sql);
24.                         while ($instanc = mysqli_fetch_object($excute)){
25.                                     $field[] = $instanc->Field;
26.                         }
27.                         $col = "";   $val = "";   $c="";
28.                         foreach ($_POST as $key=>$value) {
29.                                     if (in_array($key, $field)){
30.                                                 $col.=$c;   $val.=$c;
31.                                                 $col.=$key;
32.                                                 $val."".$value."";
33.                                                 $c=",";
34.                                     }
35.                         }
36.                         $val = str_replace("", "NULL", $val);
37.                         $insertSql = "INSERT INTO member (".$col.") VALUES (".$val.")";
38.                         $excute = mysqli_query($conn, $insertSql);
39.                         if ($excute){
40.                                     $last_id = $conn->insert_id;
41.                                     $json["last_id"] = $last_id;
42.                         }else{
43.                                     $json["alert"] = $excute;
44.                                     $json["sql"] = $insertSql;
45.                         }
46.                         $json["date_now"] = date("Y-m-d H:i:s");
47.             }else{
```

```
48.                         $json["alert"] = "No record information available!!";
49.             }
50.             mysqli_close($conn);
51. }
52. echo json_encode($json);
53. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 – 11 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 13 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 14 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 15 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 17 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 18 หากเชื่อมต่อฐานข้อมูลสำเร็จจะเค้าไปทำงานในบรรทัดที่ 20 – 51

บรรทัดที่ 18 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 19 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 20 ตรวจสอบว่ามีข้อมูล POST ส่งมาหรือไม่ ถ้ามีค่าส่งมาจะทำงานภายในบรรทัดที่ 21 - 47

บรรทัดที่ 21 – 26 ดึงข้อมูลชื่อคอลัมน์จากตาราง member โดยไม่สนใจคอลัมน์ที่มีคุณสมบัติเป็น auto_increment (ตัวเลขอัตโนมัติ) มาเก็บไว้ในตัวแปร $field สำหรับนำไปตรวจสอบข้อมูลที่ถูกส่งมา

บรรทัดที่ 27 – 35 เป็นการสร้างคำสั่ง SQL INSERT จากข้อมูล POST ที่ถูกส่งมา จะมีการตรวจสอบชื่อตัวแปรที่อยู่ใน POST ว่ามีชื่ออยู่ในคอลัมน์ของตาราง ก่อนทำการสร้างคำสั่ง SQL INSERT

บรรทัดที่ 36 ทำการเปลี่ยนแปลงข้อมูลที่จะ INSERT ลงตารางหาคอลัมน์ใดไม่มีค่าจะถูกเปลี่ยนเป็น NULL

บรรทัดที่ 37 ประกาศตัวแปร $insertSql เพื่อสร้างคำสั่ง SQL จากการทำงานในบรรทัดที่ 33 – 44

บรรทัดที่ 38 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 39 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 40 - 41 ทำการดึง Primary key ของข้อมูลที่ถูก INSERT ลงไป มาเก็บไว้ในตัวแปร $last_id แล้วเก็บข้อมูล $last_id ลงในตัวแปร $json ใน property ชื่อว่า last_id

บรรทัดที่ 52 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่าตัวแปร $excute

บรรทัดที่ 43 – 44 เก็บข้อผิดพลาดไว้ในตัวแปร $json ใน property ชื่อว่า alert และส่งคำสั่ง SQL กลับไปเพื่อตรวจสอบความถูกต้องของคำสั่ง ใน property ชื่อว่า sql

บรรทัดที่ 46 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 47 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า POST

บรรทัดที่ 48 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 50 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 52 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 53 จบการทำงานของไฟล์ memberInsert.php

2. คำสั่ง SELECT สามารถแบ่งได้ 2 ลักษณะการทำงาน คือ

1) ไฟล์ชื่อ memberList.php API สำหรับดึงข้อมูลจากฐานข้อมูลมาเป็นรายการ

```
1. <?php
2.    error_reporting(E_ALL);
3.    ob_start();
4.    ini_set("display_errors", 1);
5.    date_default_timezone_set("Asia/Bangkok");
6.    header("Access-Control-Allow-Origin: *");
7.    header("Content-Type: application/json; charset=UTF-8");
8.    $servername = "localhost";
9.    $database = "example_db";
10.   $username = "root";
11.   $password = "";
12.   // Create connection
13.   $conn = new mysqli($servername, $username, $password, $database);
14.   $conn->set_charset("utf8");
15.   $json = array();
16.   // Check connection
17.   if ($conn->connect_error) {
18.               $json['connect'] = ("Connection failed: " . $conn->connect_error);
19.   }else{
20.               $json['instance'] = array();
```

```
21.             $perPage = isset($_GET["perPage"]) ? $_GET["perPage"] : 5;
22.             $page = isset($_GET["page"]) ? $_GET["page"] : 1;
23.             $pageStart = ($page-1)*$perPage;
24.             $sql = "SELECT * FROM member LIMIT ".$pageStart.",".$perPage;
25.             $query = mysqli_query($conn, $sql);
26.             while ($instance=mysqli_fetch_assoc($query)) {
27.                             $json["instance"][] = $instance;
28.             }
29.             $sqlCount = "SELECT count(id) AS total FROM member";
30.             $query = mysqli_query($conn, $sqlCount);
31.             $total = mysqli_fetch_assoc($query);
32.             $json['pagination'] = array();
33.             $json['pagination']['total'] = intval($total['total']);
34.             $json['pagination']['page'] = intval($page);
35.             $json['pagination']['pageStart'] = intval($pageStart);
36.             $json['pagination']['perPage'] = intval($perPage);
37.             mysqli_close($conn);
38.             $json["date_now"] = date("Y-m-d H:i:s");
39. }
40. echo json_encode($json);
41. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 – 11 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 13 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 14 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 15 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 17 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 18 หากเชื่อมต่อฐานข้อมูลสำเร็จจะเค้าไปทำงานในบรรทัดที่ 20 – 38

บรรทัดที่ 18 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 19 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 20 ประกาศ property ชื่อว่า instance ในตัวแปร $json ให้เป็น array เพื่อเก็บข้อมูลที่ได้จากฐานข้อมูล

บรรทัดที่ 21 ประกาศตัวแปร $perPage เพื่อกำหนดจำนวนชุดข้อมูลที่มากที่สุดที่ได้จากฐานข้อมูล โดยมีค่าพื้นฐานเป็น 5 รายการต่อ 1 ครั้ง หรือเมื่อมีการส่งตัวแปร perPage ผ่าน method GET

บรรทัดที่ 22 ประกาศตัวแปรชื่อ $page เพื่อรับค่าข้อมูลเลขหน้าปัจจุบันที่ส่งมาในตัวแปร page ผ่าน method GET หากไม่มีค่าส่งมาจะมีค่าเป็น 1

บรรทัดที่ 23 ประกาศตัวแปรชื่อ $pageStart เพิ่มเก็บข้อมูลลำดับเริ่มแรกในการดึงข้อมูล โดยค่าเริ่มต้นจะเป็น 0

บรรทัดที่ 24 ประกาศตัวแปร $sql เพื่อเก็บคำสั่ง SQL

บรรทัดที่ 25 ประกาศตัวแปร $query เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 26 ทำการวนลูปรับค่าจากการดึงข้อมูลในแต่ละเรคคอร์ดมาเก็บไว้ใน $instance จนกว่าจะดึงข้อมูลจนครบ ตามคำสั่ง SQL

บรรทัดที่ 27 นำข้อมูลใน $instance มาเก็บไว้ในตัวแปร $json[‘instance’] ในแต่ละช่องของ array

บรรทัดที่ 29 – 31 นับจำนวนเรคคอร์ดทั้งหมดของข้อมูลโดยไม่จำกัดจำนวน แล้วเก็บไว้ในตัวแปร $total

บรรทัดที่ 32 ประกาศ property ชื่อว่า pagination ในตัวแปร $json ให้เป็น array เพื่อเก็บข้อมูลการแบ่งหน้า

บรรทัดที่ 33 เก็บข้อมูลจากตัวแปร $total[‘total’] ไว้ในตัวแปร $json['pagination'] property ชื่อว่า total พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 34 เก็บข้อมูลจากตัวแปร $page ไว้ในตัวแปร $json['pagination'] property ชื่อว่า page พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 35 เก็บข้อมูลจากตัวแปร $pageStart ไว้ในตัวแปร $json['pagination'] property ชื่อว่า pageStart พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 36 เก็บข้อมูลจากตัวแปร $perPage ไว้ในตัวแปร $json['pagination'] property ชื่อว่า perPage พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 37 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 38 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 40 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 41 จบการทำงานของไฟล์ memberList.php

2) ไฟล์ชื่อ memberShow.php API สำหรับดึงข้อมูลจากฐานข้อมูลมาเพียง 1 รายการ

```php
<?php
 error_reporting(E_ALL);
 ob_start();
 ini_set("display_errors", 1);
 date_default_timezone_set("Asia/Bangkok");
 header("Access-Control-Allow-Origin: *");
 header("Content-Type: application/json; charset=UTF-8");
 $servername = "localhost";
 $database = "example_db";
 $username = "root";
 $password = "";
 // Create connection
 $conn = new mysqli($servername, $username, $password, $database);
 $conn->set_charset("utf8");
 $json = array();
 // Check connection
 if ($conn->connect_error) {
        $json['connect'] = ("Connection failed: " . $conn->connect_error);
 }else{
        $json['instance'] = array();
        $id = isset($_GET['id']) ? $_GET['id'] : null;
        if ($id){
                $sql = "SELECT * FROM member WHERE id=".$id;
                $query = mysqli_query($conn, $sql);
                $json["instance"]=mysqli_fetch_assoc($query);
        }else{
                $json["alert"] = "No record information!!";
        }
        mysqli_close($conn);
        $json["date_now"] = date("Y-m-d H:i:s");
 }
 echo json_encode($json);
 ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 – 11 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 13 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 14 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 15 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 17 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 23 - 25 หากเชื่อมต่อฐานข้อมูลสำเร็จจะเข้าไปทำงานในบรรทัดที่ 20 – 30

บรรทัดที่ 18 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 19 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 20 ประกาศ property ชื่อว่า instance ในตัวแปร $json ให้เป็น array เพื่อเก็บข้อมูลที่ได้จากฐานข้อมูล

บรรทัดที่ 21 ประกาศตัวแปร $id เพื่อเก็บค่า id ที่ส่งมาผ่าน method GET พร้อมกับตรวจสอบว่า มีค่าส่งมาจริง หากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 22 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 23 - 25 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 27

บรรทัดที่ 23 ประกาศตัวแปร $sql เพื่อเก็บคำสั่ง SQL

บรรทัดที่ 24 ประกาศตัวแปร $query เพื่อรับค่าผลลัพธ์ที่ได้จากการนำคำสั่ง SQL ไปประมวลผลในฟังก์ชัน mysqli_query

บรรทัดที่ 25 เรียกใช้ฟังก์ชัน mysqli_fetch_assoc เพื่อดึงข้อมูลจากตัวแปร $query มาเก็บไว้ในตัวแปร $json[‘instance’]

บรรทัดที่ 26 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $id

บรรทัดที่ 27 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 29 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 30 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 32 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 33 จบการทำงานของไฟล์ memberShow.php

3. คำสั่ง UPDATE ไฟล์ชื่อ memberUpdate.php API สำหรับแก้ไขปรับปรุงข้อมูลในฐานข้อมูล

```
1. <?php
2.    error_reporting(E_ALL);
3.    ob_start();
4.    ini_set("display_errors", 1);
5.    date_default_timezone_set("Asia/Bangkok");
6.    header("Access-Control-Allow-Origin: *");
7.    header("Content-Type: application/json; charset=UTF-8");
8.    $servername = "localhost";
9.    $database = "example_db";
10.  $username = "root";
11.  $password = "";
12.  // Create connection
13.  $conn = new mysqli($servername, $username, $password, $database);
14.  $conn->set_charset("utf8");
15.  $json = array();
16.  // Check connection
17.  if ($conn->connect_error) {
18.              $json['connect'] = ("Connection failed: " . $conn->connect_error);
19.  }else{
20.              $id = isset($_GET["id"]) ? $_GET["id"] : (isset($_POST["id"]) ? $_POST["id"] : null);
21.              if ($id){
22.                              $field = array();
23.                              $sql = "SHOW FULL COLUMNS FROM member WHERE Extra!='auto_increment'
     ";
24.                              $excute = mysqli_query($conn, $sql);
25.                              while ($instanc = mysqli_fetch_object($excute)){
26.                                              $field[] = $instanc->Field;
27.                              }
28.                              $col = "";   $val = "";   $c="";
29.                              foreach ($_POST as $key=>$value) {
30.                                              if (in_array($key, $field)){
31.                                                              $col.=$c;
```

```
32.                                         $col.= $key."='".$value."'";
33.                                         $c=",";
34.                                 }
35.                         }
36.                         $col = str_replace("'", "NULL", $col);
37.                         $updateSql = "UPDATE member SET ".$col." WHERE id='".$id."'";
38.                         $excute = mysqli_query($conn, $updateSql);
39.                         if ($excute){
40.                                 $json["update_id"] = $id;
41.                                 $json["status"] = true;
42.                         }else{
43.                                 $json["sql"] = $updateSql;
44.                                 $json["status"] = false;
45.                         }
46.             }else{
47.                         $json["alert"] = "No record information available!!";
48.             }
49.             $json["date_now"] = date("Y-m-d H:i:s");
50.             mysqli_close($conn);
51. }
52. echo json_encode($json);
53. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 – 11 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 13 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 14 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 15 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 17 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 18 หากเชื่อมต่อฐานข้อมูลสำเร็จจะเข้าไปทำงานในบรรทัดที่ 20 – 50

บรรทัดที่ 18 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 19 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 20 ประกาศตัวแปร $id เพื่อเก็บค่า id ที่ส่งมาผ่าน method GET หรือ POST พร้อมกับตรวจสอบว่า มีค่าส่งมาจริง หากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 21 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 22 - 45 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 47

บรรทัดที่ 22 - 27 ดึงข้อมูลชื่อคอลัมน์จากตาราง member โดยไม่สนใจคอลัมน์ที่มีคุณสมบัติเป็น auto_increment (ตัวเลขอัตโนมัติ) มาเก็บไว้ในตัวแปร $field สำหรับนำไปตรวจสอบข้อมูลที่ถูกส่งมา

บรรทัดที่ 28 – 35 เป็นการสร้างคำสั่ง SQL UPDATE จากข้อมูล POST ที่ถูกส่งมา จะมีการตรวจสอบชื่อตัวแปรที่อยู่ใน POST ว่ามีชื่ออยู่ในคอลัมน์ของตารางก่อนทำการสร้างคำสั่ง SQL UPDATE

บรรทัดที่ 36 ทำการปรับปรุงทำสั่ง SQL UPDATE ด้วยฟังก์ชัน str_replace() โดยแทนที่ข้อความ ‘’ (ข้อมูลที่ไม่มีค่า) ให้เป็น NULL แล้วเก็บไว้ในตัวแปรเดิม ($col)

บรรทัดที่ 37 กำหนดตัวแปล $updateSql เพื่อเก็บข้อมูลคำสั่ง SQL UPDATE ที่สร้างจากตัวแปร $col ซึ่งผ่านการตรวจสอบความถูกต้องมาแล้ว พร้อมกับใส่เงื่อนไขการปรับปรุงข้อมูลโดยระบุ Primary key ที่มีค่าเท่ากับตัวแปร $id

บรรทัดที่ 38 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 39 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 40 เก็บค่า $id ของข้อมูลที่ทำการปรับปรุงลงในตัวแปร $json ใน property ชื่อว่า update_id

บรรทัดที่ 41 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น true

บรรทัดที่ 42 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $ excute

บรรทัดที่ 43 กำหนดตัวแปร $json ใน property ชื่อว่า sql เก็บค่าคำสั่ง SQL UPDATE ที่อยู่นตัวแปร $updateSql ส่งกลับไปเพื่อตรวจสอบความถูกต้อง

บรรทัดที่ 44 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น false

บรรทัดที่ 46 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $id

บรรทัดที่ 47 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 49 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 50 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 52 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 53 จบการทำงานของไฟล์ memberShow.php

4. คำสั่ง DELETE ไฟล์ชื่อ memberDelete.php API สำหรับลบข้อมูล

```php
<?php
error_reporting(E_ALL);
ob_start();
ini_set("display_errors", 1);
date_default_timezone_set("Asia/Bangkok");
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
$servername = "localhost";
$database = "example_db";
$username = "root";
$password = "";
// Create connection
$conn = new mysqli($servername, $username, $password, $database);
$conn->set_charset("utf8");
$json = array();
// Check connection
if ($conn->connect_error) {
        $json['connect'] = ("Connection failed: " . $conn->connect_error);
}else{
        $id = isset($_POST["id"]) ? $_POST["id"] : null;
        if ($id){
                $deleteSql = "DELETE FROM member WHERE id='".$id."'";
                $excute = mysqli_query($conn, $deleteSql);
                if ($excute){
                        $json['status'] = true;
                        $json["message"] = "Delete Success.";
                }else{
                        $json['status'] = false;
                        $json["message"] = "Delete Fail!!";
                        $json["alert"] = $excute;
                        $json["sql"] = $deleteSql;
                }
        }else{
                $json["alert"] = "No record information available!!";
        }
        $json["date_now"] = date("Y-m-d H:i:s");
        mysqli_close($conn);
}
```

```
39. echo json_encode($json);
40. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 – 11 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 13 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 14 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 15 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 17 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 18 หากเชื่อมต่อฐานข้อมูลสำเร็จจะเข้าไปทำงานในบรรทัดที่ 20 – 37

บรรทัดที่ 18 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 19 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 20 ประกาศตัวแปร $id เพื่อเก็บค่า id ที่ส่งมาผ่าน method GET พร้อมกับตรวจสอบว่า มีค่าส่งมาจริง หากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 21 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 22 - 32 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 34

บรรทัดที่ 22 กำหนดตัวแปล $deleteSql เพื่อเก็บข้อมูลคำสั่ง SQL DELETE พร้อมกับใส่เงื่อนไขการลบข้อมูลโดยระบุ Primary key ที่มีค่าเท่ากับตัวแปร $id

บรรทัดที่ 23 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 24 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 25 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น true

บรรทัดที่ 26 กำหนดตัวแปร $json ใน property ชื่อว่า massage เก็บข้อความว่า “Delete Success.”

บรรทัดที่ 27 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $ excute

บรรทัดที่ 28 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น false

บรรทัดที่ 29 กำหนดตัวแปร $json ใน property ชื่อว่า massage เก็บข้อความว่า “Delete Fail!!”

บรรทัดที่ 30 กำหนดตัวแปร $json ใน property ชื่อว่า alert เก็บค่าที่ถูกส่งมาจากการเรียกใช้งานฟังก์ชัน mysqli_query() ในบรรทัดที่ 23

บรรทัดที่ 31 กำหนดตัวแปร $json ใน property ชื่อว่า sql เก็บค่าคำสั่ง SQL DELETE ที่อยู่นตัวแปร $deleteSql ส่งกลับไปเพื่อตรวจสอบความถูกต้อง

บรรทัดที่ 33 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $id

บรรทัดที่ 34 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 36 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 37 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 39 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 40 จบการทำงานของไฟล์ memberDelete.php

วิธีการเรียกใช้งาน API ตารางสมาชิก (member)

มีรูปแบบดังนี้ http://{ชื่อเว็บไซต์}/model/{ชื่อตาราง}/{ชื่อไฟล์}.php เช่น ต้องการรายชื่อสมาชิกทั้งหมดจะสามารถเรียก API ผ่าน URL http://localhost/model/member/memberList.php

การเขียน API สำหรับตารางที่อยู่ของสมาชิก (address) ด้วยการเขียนแบบฟังก์ชัน

จะสังเกตว่ามีการเขียนโค้ดชุดเดียวกันในหลาย ๆ จุด ซึ่งเราสามารแยกโค้ดเหล่านั้นออกมาเป็นไฟล์ใหม่ได้ เพื่อความสะดวกในการแก้ไขและเรียกใช้งาน กรณีศึกษาดังนี้

สร้างไฟล์ Library สำหรับใช้งานร่วมกับฟังก์ชันอื่น ๆ

1. ไฟล์สำหรับการเชื่อมต่อฐานข้อมูล สร้างไฟล์ชื่อ connect.php ไว้ในโฟลเดอร์ model/opdb

```
1.  <?php
2.  $servername = "localhost";
3.  $database = "example_db";
4.  $username = "root";
5.  $password = "";
6.  // Create connection
7.  $conn = new mysqli($servername, $username, $password, $database);
8.  $conn->set_charset("utf8");
9.  if ($conn->connect_error) {
10.     $json['connect'] = ("Connection failed: " . $conn->connect_error);
11.     mysqli_close($conn);
12.     $json["date_now"] = date("Y-m-d H:i:s");
13.     echo json_encode($json);
```

```
14.         die();
15.  }
16.  ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 – 5 เป็นการประกาศตัวแปรพร้อมกับข้อมูลที่จำเป็นในการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 7 ประกาศตัวแปร $conn เก็บข้อมูลการเชื่อมต่อกับฐานข้อมูล

บรรทัดที่ 8 กำหนดให้ฐานข้อมูลเข้ารหัสเป็น UTF8

บรรทัดที่ 9 ตรวจสอบการเชื่อมต่อฐานข้อมูล ถ้าเชื่อมต่อฐานข้อมูลไม่สำเร็จจะเข้าไปทำงานในบรรทัดที่ 10 – 14

บรรทัดที่ 10 หากการเชื่อมต่อฐานข้อมูลเกิดข้อผิดพลาด ให้ $json เก็บค่าการแจ้งเตือนข้อผิดพลาดนั้นไว้ใน property ชื่อว่า connect

บรรทัดที่ 11 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 12 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 13 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 14 เรียกใช้ฟังก์ชัน die() เพื่อจบการทำงานของ php

บรรทัดที่ 16 จบการทำงานของไฟล์ connect.php

2. ไฟล์สำหรับการตั้งค่า header สร้างไฟล์ชื่อ herder.php ไว้ในโฟลเดอร์ model/main

```
1.  <?php
2.  error_reporting(E_ALL);
3.  ob_start();
4.  ini_set("display_errors", 1);
5.  date_default_timezone_set("Asia/Bangkok");
6.  header("Access-Control-Allow-Origin: *");
7.  header("Content-Type: application/json; charset=UTF-8");
8.  $json = array();
9.  ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 error_reporting(E_ALL); เป็นคำสั่งแสดงข้อความแจ้งข้อผิดพลาดของโปรแกรมที่จะแสดงออกมา โดยจะแสดงว่าเกิดข้อผิดพลาดอะไรขึ้นที่ไหน เพื่อให้นักพัฒนาสามารถ Debug ได้ง่ายและถูกจุด

บรรทัดที่ 3 ob_start(); เพื่อให้ฟังก์ชัน header() มาสารถใช้งานได้

บรรทัดที่ 4 ini_set("display_errors", 1); เพื่อให้ php แจ้งให้เราทราบว่ามีข้อผิดพลาดที่บรรทัดไหนอย่างไร

บรรทัดที่ 5 date_default_timezone_set("Asia/Bangkok"); ตั้งค่าเวลาพื้นฐานเป็นเวลาของประเทศไทย

บรรทัดที่ 6 header("Access-Control-Allow-Origin: *"); เป็นการกำหนดสิทธิ์การในการเข้าใช้งาน ในกรณีนี้มีค่าเป็นเครื่องหมายดอกจัน (*) จะเป็นการอนุญาตให้เข้าถึงได้ทุกกรณี

บรรทัดที่ 7 header("Content-Type: application/json; charset=UTF-8"); เป็นการกำหนดหน้าข้อมูลแสดงผลในรูปแบบ JSON (การแสดงผลปกติจะอยู่ในรูปแบบ HTML) และเข้ารหัสข้อความด้วยมาตรฐาน UTF-8

บรรทัดที่ 8 ประกาศตัวแปร $json สำหรับเป็นที่เก็บข้อมูลทั้งหมดที่จะทำการส่ง API

บรรทัดที่ 9 จบการทำงานของไฟล์ herder.php

3. ไฟล์สำหรับดึงข้อมูลคอลัมน์ในตารางต่าง ๆ สร้างไฟล์ชื่อ getColumname.php ไว้ในโฟลเดอร์ model/main

```
1.  <?php
2.  function getColumname($conn, $table=""){
3.      $field = array();
4.      if ($table){
5.                  $sql = "SHOW FULL COLUMNS FROM ".$table." WHERE Extra!='auto_increment' ";
6.                  $excute = mysqli_query($conn, $sql);
7.                  while ($instanc = mysqli_fetch_object($excute)){
8.                          $field[] = $instanc->Field;
9.                  }
10.     }
11.     return $field;
12. }
13. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ getColumname โดยรับค่ามาทำงานในฟังก์ชันสองค่า คือ ตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล และตัวแปล $table เก็บชื่อตาราง

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $field สำหรับเก็บข้อมูลคอลัมน์ในตารางที่ได้จากการประมวลผล

บรรทัดที่ 4 ตรวจสอบค่าของตัวแปร $table ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 4 - 10

บรรทัดที่ 5 ประกาศตัวแปร $sql เพื่อเก็บคำสั่ง SQL ที่ใช้ค้นหาคอลัมน์ที่อยู่ในตางรางโดยไม่สนใจคอลัมน์ประเภทตัวเลขที่เพิ่มค่าขึ้นโดยอัตโนมัติ

บรรทัดที่ 6 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 7 ทำการวนลูปรับค่าจากการดึงข้อมูลในแต่ละรายการมาเก็บไว้ใน $instance จนกว่าจะดึงข้อมูลจนครบ ตามคำสั่ง SQL

บรรทัดที่ 8 นำข้อมูลชื่อคอลัมน์ใน $instance->Field มาเก็บไว้ในตัวแปร $field ในแต่ละช่องของ array

บรรทัดที่ 11 ส่งค่าตัวแปร $field กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชัน

บรรทัดที่ 13 จบการทำงานของไฟล์ getColumname.php

สร้างไฟล์ API สำหรับจัดการตารางที่อยู่สมาชิก (address)

เริ่มต้นสร้าง JSON API สำหรับจัดการข้อมูลตารางที่อยู่ของสมาชิก (address) ทำการสร้างโฟลเดอร์ชื่อ address ไว้ในโฟลเดอร์ model ซึ่งสามารแบ่งไฟล์ตามลักษณะการทำงานของคำสั่ง SQL ในกรณีนี้จะเป็นการเขียนโดยใช้การทำงานแบบฟังก์ชัน โดยไฟล์ที่ใช้ในการประมวลผลมีข้อกำหนดว่า ชื่อไฟล์ต้องเป็นชื่อเดียวกับชื่อฟังก์ชัน

1. ไฟล์ชื่อ index.php ทำหน้าที่ในการเรียกใช้งาน service (function) ที่อยู่ในโฟลเดอร์

```
1. <?php
2. include("../main/herder.php");
3. include("../opdb/connect.php");
4. $action = isset($_GET["action"]) ? $_GET["action"] : null;
5. $actionFile = $action.".php";
6. if (file_exists($actionFile)){
7.                 include($actionFile);
8.                 if (function_exists($action)){
9.                                 $json = $action($conn);
10.                }else{
11.                                $json["alert"] = "Function not found!!!";
12.                }
13. }else{
14.                $json["alert"] = "File not found!!!";
15. }
16. $json["date_now"] = date("Y-m-d H:i:s");
17. mysqli_close($conn);
18. echo json_encode($json);
19. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 นำเข้าไฟล์ herder.php

บรรทัดที่ 3 นำเข้าไฟล์ connect.php สำหรับเชื่อมต่อฐานข้อมูล

บรรทัดที่ 4 ประกาศตัวแปล $action เพื่อเก็บค่าชื่อกระบวนการของ API (ชื่อฟังชันการทำงาน)

บรรทัดที่ 5 ประกาศตัวแปร $actionFile เพื่อเก็บชื่อไฟล์ที่ใช้ในการทำงานของ API โดยทำการเพิ่มนามสกุลไฟล์ .php ต่อท้าย

บรรทัดที่ 6 ตรวจสอบชื่อไฟล์ในตัวแปล $actionFile ว่ามีไฟล์ชื่อดังกล่าวอยู่จริงหรือไม่ โดยใช้ด้วยฟังก์ชัน file_exists หากไฟล์นั้นมีอยู่จริง จะเข้าไปทำงานในบรรทัดที่ 9 – 12 หากไม่พบไฟล์จะเข้าไปทำงานในบรรทัดที่ 14

บรรทัดที่ 7 นำเข้าไฟล์ที่เก็บในตัวแปล $actionFile โดยใช้ด้วยฟังก์ชัน function_exists หากฟังก์ชันนั้นมีอยู่จริง จะเข้าไปทำงานในบรรทัดที่ 9 หากไม่พบฟังก์ชันจะเข้าไปทำงานในบรรทัดที่ 11

บรรทัดที่ 8 ตรวจสอบชื่อฟังก์ชันในตัวแปร $action ว่ามีชื่อฟังก์ชันนั้นอยู่จริงหรือไม่

บรรทัดที่ 9 เรียกใช้งานฟังก์ชันในตัวแปร $action โดยส่งค่าตัวแปรในการเชื่อต่อฐานข้อมูลเข้าไปทำงานในฟังก์ชัน แล้วให้ตัวแปร $json เก็บค่าผลลัพธ์ที่ได้จากการประมวลผล

บรรทัดที่ 10 เริ่มการทำงานในส่วนของ else จากการตรวจสอบชื่อฟังก์ชัน

บรรทัดที่ 11 ทำการส่งข้อความ "Function not found!!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 13 เริ่มการทำงานในส่วนของ else จากการตรวจสอบชื่อไฟล์

บรรทัดที่ 14 ทำการส่งข้อความ "File not found!!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 16 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 17 คำสั่งปิดการเชื่อมต่อฐานข้อมูลที่ตัวแปร $conn เก็บค่าไว้

บรรทัดที่ 18 echo json_encode($json); นำตัวแปร $json เข้ารหัสให้อยู่ในรูปแบบของ JSON แล้วทำการแสดงออกที่หน้าจอ (echo)

บรรทัดที่ 19 จบการทำงานของไฟล์ index.php

2. ไฟล์ชื่อ addressInsert.php ทำหน้าที่เพิ่มข้อมูลใหม่ให้กับตารางที่อยู่ของสมาชิก

```
1. <?php
2. function addressInsert($conn){
3.              $json = array();
4.              if (isset($_POST) && $_POST){
5.                              include("../main/getColumname.php");
6.                              $field = getColumname($conn, "address");
7.                              $col = "";   $val = "";   $c="";
8.                              foreach ($_POST as $key=>$value) {
9.                              if (in_array($key, $field)){
10.                                                             $col.=$c;   $val.=$c;
11.                                                             $col.=$key;
12.                                                             $val."".$value."";
13.                                                             $c=",";
14.                                             }
15.                             }
16.                             $val = str_replace("""", "NULL", $val);
17.                             $insertSql = "INSERT INTO address (".$col.") VALUES (".$val.")";
18.                             $excute = mysqli_query($conn, $insertSql);
19.                             if ($excute){
20.                                             $last_id = $conn->insert_id;
21.                                             $json["last_id"] = $last_id;
22.                             }else{
23.                                             $json["alert"] = $excute;
```

```
24.                                 $json["sql"] = $insertSql;
25.                         }
26.             }else{
27.                         $json["alert"] = "No record information available!!";
28.             }
29.             $json["date_now"] = date("Y-m-d H:i:s");
30.             return $json;
31.  }
32.  ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ addressInsert โดยรับค่าตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $json สำหรับเก็บข้อมูลที่ได้จากการประมวลผล

บรรทัดที่ 4 ตรวจสอบว่ามีข้อมูล POST ส่งมาหรือไม่ ถ้ามีค่าส่งมาจะทำงานภายในบรรทัดที่ 5 – 26

บรรทัดที่ 5 นำเข้าไฟล์ getColumname.php สำหรับใช้งานฟังก์ชัน getColumname โดยส่งค่าชื่อตาราง “address”

บรรทัดที่ 6 ประกาศตัวแปร $field เพื่อเก็บข้อมูลชื่อคอลัมน์จากตาราง address ที่ถูกส่งมาจากฟังก์ชัน getColumname

บรรทัดที่ 7 – 15 เป็นการสร้างคำสั่ง SQL INSERT จากข้อมูล POST ที่ถูกส่งมา จะมีการตรวจสอบชื่อตัวแปรที่อยู่ใน POST ว่าต้องมีชื่ออยู่ในคอลัมน์ของตาราง address ก่อนทำการสร้างคำสั่ง SQL INSERT

บรรทัดที่ 16 ทำการเปลี่ยนแปลงข้อมูลที่จะ INSERT คอลัมน์ใดไม่มีค่าจะถูกเปลี่ยนเป็น NULL

บรรทัดที่ 17 ประกาศตัวแปร $insertSql เพื่อสร้างคำสั่ง SQL จากการทำงานในบรรทัดที่ 7 – 16

บรรทัดที่ 18 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 19 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 20 - 21 ทำการดึง Primary key ของข้อมูลที่ถูก INSERT ลงไป มาเก็บไว้ในตัวแปร $last_id แล้วเก็บข้อมูล $last_id ลงในตัวแปร $json ใน property ชื่อว่า last_id

บรรทัดที่ 22 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่าตัวแปร $excute

บรรทัดที่ 23 – 24 เก็บข้อผิดพลาดไว้ในตัวแปร $json ใน property ชื่อว่า alert และส่งคำสั่ง SQL กลับไปเพื่อตรวจสอบความถูกต้องของคำสั่ง ใน property ชื่อว่า sql

บรรทัดที่ 26 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า POST

บรรทัดที่ 27 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $json ใน property ชื่อว่า alert

บรรทัดที่ 29 เก็บค่าวันและเวลาที่ API ถูกเรียกใช้งานไว้ในตัวแปร $json ใน property ชื่อว่า date_now

บรรทัดที่ 30 ส่งค่าตัวแปร $json กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชันในหน้า index.php

บรรทัดที่ 31 จบการทำงานฟังก์ชัน addressInsert

บรรทัดที่ 32 จบการทำงานของไฟล์ addressInsert.php

3. ไฟล์ชื่อ addressList.php ทำหน้าที่ดึงข้อมูลจากตารางที่อยู่ของสมาชิกมาครั้งละหลาย ๆ รายการ

```
1. <?php
2. function addressList($conn){
3.              $jsonData = array();
4.              // default pagination
5.              $perPage = isset($_GET["perPage"]) ? $_GET["perPage"] : 5;
6.              $page = isset($_GET["page"]) ? $_GET["page"] : 1;
7.              $pageStart = ($page-1)*$perPage;
8.                              // query data
9.              $sql = "SELECT address.*, member.name, member.lastname  FROM address, member
    WHERE member.id=address.member_id LIMIT ".$pageStart.",".$perPage;
10.             $query = mysqli_query($conn, $sql);
11.             $jsonData["instance"] = array();
12.             while ($instance=mysqli_fetch_assoc($query)) {
13.                             $jsonData["instance"][] = $instance;
14.             }
15.             // set pagination
16.             $sqlCount = "SELECT count(id_address) AS total FROM  address, member WHERE
    member.id=address.member_id";
17.             $query = mysqli_query($conn, $sqlCount);
18.             $total = mysqli_fetch_assoc($query);
19.             $jsonData['pagination'] = array();
20.             $jsonData['pagination']['total'] = intval($total['total']);
21.             $jsonData['pagination']['page'] = intval($page);
22.             $jsonData['pagination']['pageStart'] = intval($pageStart);
23.             $jsonData['pagination']['perPage'] = intval($perPage);
24.             return $jsonData;
25. }
26. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ addressList โดยรับค่าตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $jsonData สำหรับเก็บข้อมูลที่ได้จากการประมวลผล

บรรทัดที่ 5 ประกาศตัวแปร $perPage เพื่อกำหนดจำนวนชุดข้อมูลที่มากที่สุดที่ได้จากฐานข้อมูล โดยมีค่าพื้นฐานเป็น 5 รายการต่อ 1 ครั้ง หรือเมื่อมีการส่งตัวแปร perPage ผ่าน method GET

บรรทัดที่ 6 ประกาศตัวแปรชื่อ $page เพื่อรับค่าข้อมูลเลขหน้าปัจจุบันที่ส่งมาในตัวแปร page ผ่าน method GET หากไม่มีค่าส่งมาจะมีค่าเป็น 1

บรรทัดที่ 7 ประกาศตัวแปรชื่อ $pageStart เพิ่มเก็บข้อมูลลำดับเริ่มแรกในการดึงข้อมูล โดยค่าเริ่มต้นจะเป็น 0

บรรทัดที่ 9 ประกาศตัวแปร $sql เพื่อเก็บคำสั่ง SQL

บรรทัดที่ 10 ประกาศตัวแปร $query เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 11 ประกาศ property ชื่อว่า instance ในตัวแปร $jsonData ให้เป็น array เพื่อเก็บข้อมูลที่ได้จากฐานข้อมูล

บรรทัดที่ 12 ทำการวนลูปรับค่าจากการดึงข้อมูลในแต่ละรายการมาเก็บไว้ใน $instance จนกว่าจะดึงข้อมูลจนครบ ตามคำสั่ง SQL

บรรทัดที่ 13 นำข้อมูลใน $instance มาเก็บไว้ในตัวแปร $jsonData[‘instance’] ในแต่ละช่องของ array

บรรทัดที่ 15 – 17 นับจำนวนรายการทั้งหมดของข้อมูล โดยไม่จำกัดจำนวน แล้วเก็บไว้ในตัวแปร $total

บรรทัดที่ 18 ประกาศ property ชื่อว่า pagination ในตัวแปร $jsonData ให้เป็น array เพื่อเก็บข้อมูลการแบ่งหน้า

บรรทัดที่ 19 เก็บข้อมูลจากตัวแปร $total[‘total’] ไว้ในตัวแปร $ jsonData ['pagination'] property ชื่อว่า total พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 20 เก็บข้อมูลจากตัวแปร $page ไว้ในตัวแปร $ jsonData ['pagination'] property ชื่อว่า page พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 21 เก็บข้อมูลจากตัวแปร $pageStart ไว้ในตัวแปร $ jsonData ['pagination'] property ชื่อว่า pageStart พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 22 เก็บข้อมูลจากตัวแปร $perPage ไว้ในตัวแปร $ jsonData ['pagination'] property ชื่อว่า perPage พร้อมทั้งแปลงข้อมูลเป็นตัวตัวเลขด้วยฟังก์ชัน intval

บรรทัดที่ 23 ส่งค่าตัวแปร $ jsonData กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชันในหน้า index.php

บรรทัดที่ 24 จบการทำงานฟังก์ชัน addressList

บรรทัดที่ 25 จบการทำงานของไฟล์ addressList.php

4. ไฟล์ชื่อ addressShow.php ทำหน้าที่ดึงข้อมูลจากตารางที่อยู่ของสมาชิกครั้ง 1 รายการ

```
1. <?php
2. function addressShow($conn){
3.              $jsonData = array();
4.              $id = isset($_GET['id']) ? $_GET['id'] : null;
5.              $jsonData["instance"] = array();
6.              if ($id){
7.                          $sql = "SELECT address.*, member.name, member.lastname FROM address,
    member WHERE member.id=address.member_id AND address.id_address=".$id;
```

```
8.                      $query = mysqli_query($conn, $sql);
9.                      $jsonData["instance"]=mysqli_fetch_assoc($query);
10.             }else{
11.                     $jsonData["alert"] = "No record information!!";
12.             }
13.             return $jsonData;
14.     }
15.     ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ addressShow โดยรับค่าตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $jsonData สำหรับเก็บข้อมูลที่ได้จากการประมวลผล

บรรทัดที่ 4 ประกาศตัวแปร $id เพื่อเก็บค่า id ที่ส่งมาผ่าน method GET พร้อมกับตรวจสอบว่า มีค่าส่งมาจริงหากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 5 ประกาศ property ชื่อว่า instance ในตัวแปร $ jsonData ให้เป็น array เพื่อเก็บข้อมูลที่ได้จากฐานข้อมูล

บรรทัดที่ 6 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 7 - 9 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 11

บรรทัดที่ 7 ประกาศตัวแปร $sql เพื่อเก็บคำสั่ง SQL

บรรทัดที่ 8 ประกาศตัวแปร $query เพื่อรับค่าผลลัพธ์ที่ได้จากการนำคำสั่ง SQL ไปประมวลผลในฟังก์ชัน mysqli_query

บรรทัดที่ 9 เรียกใช้ฟังก์ชัน mysqli_fetch_assoc เพื่อดึงข้อมูลจากตัวแปร $query มาเก็บไว้ในตัวแปร $ jsonData[‘instance’]

บรรทัดที่ 10 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $id

บรรทัดที่ 11 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $jsonData ใน property ชื่อว่า alert

บรรทัดที่ 13 ส่งค่าตัวแปร $jsonData กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชันในหน้า index.php

บรรทัดที่ 14 จบการทำงานฟังก์ชัน addressShow

บรรทัดที่ 15 จบการทำงานของไฟล์ addressShow.php

5. ไฟล์ชื่อ addressUpdate.php ทำหน้าที่ปรับปรุงข้อมูลที่อยู่ของสมาชิก โดยการละบุคีย์หลักที่ใช้อ้างอิงการ

```
1. <?php
2. function addressUpdate($conn){
3.              $jsonData = array();
4.              $id = isset($_GET["id"]) ? $_GET["id"] : (isset($_POST["id_address"]) ? $_POST["id_address"] :
    null);
```

```
5.          if ($id && isset($_POST) && $_POST){
6.                  include("../main/getColumname.php");
7.                  $field = getColumname($conn, "address");
8.                  if (isset($_POST["id_address"]))
9.                          unset($_POST["id_address"]);
10.                 $col = "";   $val = "";   $c="";
11.                 foreach ($_POST as $key=>$value) {
12.                         if (in_array($key, $field)){
13.                                 $col.=$c;
14.                                 $col.= $key."='".$value."'";
15.                                 $c=",";
16.                         }
17.                 }
18.                 $col = str_replace("'", "NULL", $col);
19.                 $updateSql = "UPDATE address SET ".$col." WHERE id_address='".$id."'";
20.                 $excute = mysqli_query($conn, $updateSql);
21.                 if ($excute){
22.                         $jsonData["update_id"] = $id;
23.                         $jsonData["status"] = true;
24.                 }else{
25.                         $jsonData["sql"] = $updateSql;
26.                         $jsonData["status"] = false;
27.                 }
28.         }else{
29.                 $jsonData["alert"] = "No record information available!!";
30.         }
31.         return $jsonData;
32. }
33. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ addressUpdate โดยรับค่าตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $jsonData สำหรับเก็บข้อมูลที่ได้จากการประมวลผล

บรรทัดที่ 4 ประกาศตัวแปร $id เพื่อเก็บค่า id_address ที่ส่งมาผ่าน method GET หรือ POST พร้อมกับตรวจสอบว่า มีค่าส่งมาจริง หากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 5 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 6 - 28 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 30

บรรทัดที่ 6 นำเข้าไฟล์ getColumname.php สำหรับใช้งานฟังก์ชัน getColumname โดยส่งค่าชื่อตาราง “address”

บรรทัดที่ 7 ประกาศตัวแปร $field เพื่อเก็บข้อมูลชื่อคอลัมน์จากตาราง address ที่ถูกส่งมาจากฟังก์ชัน getColumname

บรรทัดที่ 8 – 9 ตรวจสอบค่า POST ที่ถูกส่งมาว่ามีตัวแปรชื่อ id_address หรือไม่ ถ้าพบว่ามีจะทำการลบออกจาก POST ด้วยคำสั่ง unset เนื่องจาก id_address เป็น primary key จะไม่ถูกเปลี่ยนแปลงค่า

บรรทัดที่ 10 – 17 เป็นการสร้างคำสั่ง SQL UPDATE จากข้อมูล POST ที่ถูกส่งมา จะมีการตรวจสอบชื่อตัวแปรที่อยู่ใน POST ว่ามีชื่ออยู่ในคอลัมน์ของตารางก่อนทำการสร้างคำสั่ง SQL UPDATE

บรรทัดที่ 18 ทำการปรับปรุงทำสั่ง SQL UPDATE ด้วยฟังก์ชัน str_replace() โดยแทนที่ข้อความ ‘’ (ข้อมูลที่ไม่มีค่า) ให้เป็น NULL แล้วเก็บไว้ในตัวแปรเดิม ($col)

บรรทัดที่ 19 กำหนดตัวแปล $updateSql เพื่อเก็บข้อมูลคำสั่ง SQL UPDATE ที่สร้างจากตัวแปร $col ซึ่งผ่านการตรวจสอบความถูกต้องมาแล้ว พร้อมกับใส่เงื่อนไขการปรับปรุงข้อมูลโดยระบุ Primary key ที่มีค่าเท่ากับตัวแปร $id

บรรทัดที่ 20 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 21 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 22 เก็บค่า $id ของข้อมูลที่ทำการปรับปรุงลงในตัวแปร $jsonData ใน property ชื่อว่า update_id

บรรทัดที่ 23 กำหนดตัวแปร $jsonData ใน property ชื่อว่า status ให้มีค่าเป็น true

บรรทัดที่ 24 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $ excute

บรรทัดที่ 25 กำหนดตัวแปร $jsonData ใน property ชื่อว่า sql เก็บค่าคำสั่ง SQL UPDATE ที่อยู่นตัวแปร $updateSql ส่งกลับไปเพื่อตรวจสอบความถูกต้อง

บรรทัดที่ 26 กำหนดตัวแปร $jsonData ใน property ชื่อว่า status ให้มีค่าเป็น false

บรรทัดที่ 28 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า POST

บรรทัดที่ 29 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $jsonData ใน property ชื่อว่า alert

บรรทัดที่ 31 ส่งค่าตัวแปร $jsonData กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชันในหน้า index.php

บรรทัดที่ 32 จบการทำงานฟังก์ชัน addressUpdate

บรรทัดที่ 33 จบการทำงานของไฟล์ addressUpdate.php

6. ไฟล์ชื่อ addressDelete.php ทำหน้าที่ลบข้อมูลที่อยู่ของสมาชิก โดยการละบุคีย์หลักที่ใช้อ้างอิง

```
1. <?php
2. function addressDelete($conn){
3.              $json = array();
4.              $id = isset($_POST["id"]) ? $_POST["id"] : null;
```

```
5.          if ($id){
6.                  $deleteSql = "DELETE FROM address WHERE id_address='".$id."'";
7.                  $excute = mysqli_query($conn, $deleteSql);
8.                  if ($excute){
9.                          $json['status'] = true;
10.                         $json["message"] = "Delete Success.";
11.                 }else{
12.                         $json['status'] = false;
13.                         $json["message"] = "Delete Fail!!";
14.                         $json["alert"] = $excute;
15.                         $json["sql"] = $deleteSql;
16.                 }
17.         }else{
18.                 $json["alert"] = "No record information available!!";
19.         }
20.         return $json;
21. }
22. ?>
```

บรรทัดที่ 1 คำสั่งเปิดแท็ก php

บรรทัดที่ 2 จุดเริ่มต้นการทำงานของ function ชื่อ addressDelete โดยรับค่าตัวแปร $conn เก็บค่าการเชื่อมต่อฐานข้อมูล

บรรทัดที่ 3 ประกาศตัวแปรอาเรย์ชื่อ $json สำหรับเก็บข้อมูลที่ได้จากการประมวลผล

บรรทัดที่ 4 ประกาศตัวแปร $id เพื่อเก็บค่า id ที่ส่งมาผ่าน method POST พร้อมกับตรวจสอบว่า มีค่าส่งมาจริงหากไม่มีค่าส่งมาจะถูกกำหนดให้เป็น null

บรรทัดที่ 5 ตรวจสอบค่าของตัวแปร $id ว่ามีค่าหรือไม่ ถ้ามีค่าจะเข้าไปทำงานในบรรทัดที่ 6 - 17 หากไม่มีค่าจะเข้าไปทำงานในบรรทัดที่ 19

บรรทัดที่ 6 กำหนดตัวแปล $deleteSql เพื่อเก็บข้อมูลคำสั่ง SQL DELETE พร้อมกับใส่เงื่อนไขการลบข้อมูลโดยระบุ Primary key ที่มีค่าเท่ากับตัวแปร $id

บรรทัดที่ 7 ประกาศตัวแปร $excute เพื่อเก็บผลลัพธ์จากการประมวลผลคำสั่ง SQL ด้วยฟังก์ชัน mysqli_query()

บรรทัดที่ 24 ตรวจสอบค่าตัวแปร $excute ว่าทำงานสำเร็จหรือไม่

บรรทัดที่ 25 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น true

บรรทัดที่ 26 กำหนดตัวแปร $json ใน property ชื่อว่า massage เก็บข้อความว่า “Delete Success.”

บรรทัดที่ 27 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า $ excute

บรรทัดที่ 28 กำหนดตัวแปร $json ใน property ชื่อว่า status ให้มีค่าเป็น false

บรรทัดที่ 29 กำหนดตัวแปร $json ใน property ชื่อว่า massage เก็บข้อความว่า “Delete Fail!!”

บรรทัดที่ 30 กำหนดตัวแปร $json ใน property ชื่อว่า alert เก็บค่าที่ถูกส่งมาจากการเรียกใช้งานฟังก์ชัน mysqli_query() ในบรรทัดที่ 23

บรรทัดที่ 31 กำหนดตัวแปร $json ใน property ชื่อว่า sql เก็บค่าคำสั่ง SQL DELETE ที่อยู่นตัวแปร $deleteSql ส่งกลับไปเพื่อตรวจสอบความถูกต้อง

บรรทัดที่ 18 เริ่มเข้าสู่การทำงานในส่วนของ else จากการตรวจสอบค่า POST

บรรทัดที่ 19 ทำการส่งข้อความ "No record information available!!" ไว้ในตัวแปร $jsonData ใน property ชื่อว่า alert

บรรทัดที่ 20 ส่งค่าตัวแปร $json กลับไปยังจุดที่ถูกเรียกใช้งานฟังก์ชันในหน้า index.php

บรรทัดที่ 21 จบการทำงานฟังก์ชัน addressDelete

บรรทัดที่ 22 จบการทำงานของไฟล์ addressDelete.php

วิธีการเรียกใช้งาน API ตารางที่อยู่ของสมาชิก (address)

มีรูปแบบดังนี้ http://{ชื่อเว็บไซต์}/model/{ชื่อตาราง}/?action={ชื่อฟังก์ชันการทำงาน} เช่น ต้องการรายที่อยู่ของสมาชิกทั้งหมดจะสามารถเรียก API ผ่าน URL http://localhost/model/address/?action=addressList