



รายงาน การพัฒนาแผนที่บนเว็บ

จัดทำโดย

นางสาว สุลิสรา จันทรวีไล

ชั้นปีการศึกษาที่ 3

รหัสนิสิต 66163501

นำเสนอ

อาจารย์ผู้สอน

รศ.ดร.สิทธิชัย ชูสำโรง

รายงานนี้เป็นส่วนหนึ่งของวิชา WEB GIS DEVELOPMENT

การพัฒนาแผนที่บนเว็บ

ภาคเรียนที่ 1 ปีการศึกษา 2568

มหาวิทยาลัยนเรศวร

คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของรายวิชา **Web GIS Development: การพัฒนาแผนที่บนเว็บ** โดยมีวัตถุประสงค์เพื่อศึกษาและฝึกปฏิบัติการพัฒนาเว็บแอปพลิเคชันสำหรับระบบสารสนเทศภูมิศาสตร์ (Web GIS) ซึ่งเป็นการประยุกต์ใช้เทคโนโลยีในการแสดงผลและจัดการข้อมูลเชิงพื้นที่ผ่านระบบออนไลน์

ในการจัดทำรายงานนี้ ผู้จัดทำได้ศึกษาและทดลองใช้เทคโนโลยีทั้งในส่วน Frontend และ Backend ตลอดจนระบบฐานข้อมูลเชิงพื้นที่ เพื่อให้เข้าใจโครงสร้างและกระบวนการทำงานของระบบ Web GIS อย่างครบวงจร นอกจากนี้ยังได้มีการประยุกต์ใช้ เครื่องมือและ API ต่าง ๆ เช่น Leaflet.js, GeoServer, PostgreSQL/PostGIS, และ Thaiwater API เพื่อพัฒนาและเชื่อมโยงข้อมูลให้สามารถแสดงผลได้อย่างมีประสิทธิภาพ

รายงานฉบับนี้ครอบคลุมตั้งแต่ขั้นตอนการออกแบบระบบ การจัดการฐานข้อมูล การแสดงผลข้อมูลเชิงพื้นที่ ไปจนถึงการวิเคราะห์และนำเสนอผลลัพธ์เชิงภูมิสารสนเทศ เพื่อให้ผู้อ่านสามารถเข้าใจแนวทางการพัฒนา Web GIS และนำความรู้ที่ได้รับไปประยุกต์ใช้ในการศึกษา งานวิจัย หรือโครงการจริงในอนาคตได้อย่างเหมาะสม

จัดทำโดย

สุลิสรา จันทรวีไล

สารบัญ

คำนำ	หน้า
การล็อกอินและเข้าสู่ระบบ	1
การเพิ่มหรือแก้ไขข้อมูลจุดพิกัด (Location)	2
การดึงข้อมูลจากฐานข้อมูล locations	4
ฟอร์มล็อกอิน	9
ฟอร์มเข้าสู่ระบบ	10
การออกจากระบบ (logout.php)	12
ลบข้อมูลออกจากฐานข้อมูล	13
บันทึกข้อมูลลงฐานข้อมูล	14
เชื่อมต่อฐานข้อมูล PostgreSQL	16
ข้อมูลสำหรับเชื่อมฐานข้อมูลไว้ในเวลาระบบ	17
API	18
หน้า admin/index.php	21
JavaScript	25

การล็อกอินและเข้าสู่ระบบ

ตรวจสอบว่าผู้ใช้ได้ล็อกอินเข้าสู่ระบบแล้วหรือยัง

```
auth.php
1 <?php
2 if (session_status() === PHP_SESSION_NONE) { session_start(); }
3 if (!isset($_SESSION['user'])) {
4     header("Location: login.php?next=" . urlencode($_SERVER['REQUEST_URI']));
5     exit;
6 }
7 ?>
```

```
if (session_status() === PHP_SESSION_NONE) { session_start(); }
```

บรรทัดนี้จะเป็นการ ตรวจสอบว่าถ้ายังไม่มี การเริ่มต้น session อยู่ (PHP_SESSION_NONE)

ถ้ายังไม่มีก็จะสั่ง session_start() เพื่อเริ่ม session ใหม่ (จำเป็นเพราะข้อมูลผู้ใช้จะถูกเก็บไว้ใน \$_S)

if (!isset(\$_SESSION['user'])) { ตรวจสอบว่ามีค่าตัวแปร \$_SESSION['user'] หรือไม่ ตัวแปรนี้มักถูกกำหนด
ตอนผู้ใช้ล็อกอินสำเร็จ เช่น \$_SESSION['user'] = \$username; ถ้าไม่มี แปลว่ายังไม่ได้ล็อกอิน

header("Location: login.php?next=" . urlencode(\$_SERVER['REQUEST_URI']));exit; ใช้ header()
เพื่อสั่งให้ browser redirect ไปที่หน้า login.php พร้อมแนบพารามิเตอร์ next = URL ของหน้าปัจจุบัน
(\$_SERVER['REQUEST_URI'])→ เพื่อให้หลังจากล็อกอินสำเร็จสามารถกลับมาหน้าเดิมได้

ตัวอย่างเช่น login.php?next=/dashboard.php urlencode() ใช้เข้ารหัส URL เพื่อป้องกันอักขระพิเศษทำให้
ลิงก์เสีย exit; หยุดการทำงานของสคริปต์ทันทีหลัง redirect

การเพิ่มหรือแก้ไขข้อมูลจุดพิกัด (Location) เช่น พิกัดร้าน 7-Eleven หรือจุดนักเรียน

```
form.php
1 <?php
2 require_once __DIR__ . '/auth.php';
3 require_once __DIR__ . '/../api/db.php';
4 $id = isset($_GET['id']) ? intval($_GET['id']) : 0;
5 $item = ['category'=>'student_location','code'=>'', 'name'=>'', 'tambon'=>'', 'amphoe'=>'', 'province'=>'', 'lat'=>'', 'lon'=>''];
6 if ($id>0){
7     $stmt = $pdo->prepare("SELECT * FROM locations WHERE id=:id");
8     $stmt->execute([':id'=>$id]);
9     $item = $stmt->fetch();
10    if (!$item) die('ไม่พบข้อมูล');
11 }
12 ?>
13 <!doctype html>
14 <html lang="th">
15 <head>
16 <meta charset="utf-8">
17 <meta name="viewport" content="width=device-width, initial-scale=1">
18 <title><?=$id>?แก้ไข:เพิ่ม?> : Location</title>
19 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
20 <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@300;400;600;700&display=swap" rel="stylesheet">
21 <style>body{font-family:'Prompt',system-ui,Arial,sans-serif;background:#f7f8fb}.card{border:none;border-radius:1rem;box-shadow:0 4px
22 </head>
```

```
<div class="col-md-4">
  <label class="form-label">หมวด</label>
  <select class="form-select" name="category" required>
    <option value="seven" <?=$item['category']==='seven'? 'selected':'' >>>seven (7-Eleven)</option>
    <option value="student_location" <?=$item['category']==='student_location'? 'selected':'' >>>student_location</option>
  </select>
</div>
<div class="col-md-4">
  <label class="form-label">รหัส (ไม่จำเป็นหากเลือก)</label>
  <input class="form-control" name="code" value="<?=$item['code']>" >>
</div>
<div class="col-md-4">
  <label class="form-label">ชื่อ</label>
  <input class="form-control" name="name" required value="<?=$item['name']>" >>
</div>
<div class="col-md-4">
  <label class="form-label">ตำบล</label>
  <input class="form-control" name="tambon" value="<?=$item['tambon']>" >>
</div>
<div class="col-md-4">
  <label class="form-label">อำเภอ</label>
  <input class="form-control" name="amphoe" value="<?=$item['amphoe']>" >>
</div>
<div class="col-md-4">
  <label class="form-label">จังหวัด</label>
  <input class="form-control" name="province" value="<?=$item['province']>" >>
</div>
<div class="col-md-3">
```

บรรทัดนี้คือการ เรียกไฟล์อื่นเข้ามาใช้งาน

`require_once __DIR__ . '/auth.php';` auth.php เอาไว้ตรวจสอบว่าผู้ใช้ล็อกอินแล้วหรือยัง

`require_once __DIR__ . '/../api/db.php';` db.php คือไฟล์เชื่อมต่อฐานข้อมูล (มีตัวแปร `$pdo` สำหรับใช้รันคำสั่ง SQL)

`$id = isset($_GET['id']) ? intval($_GET['id']) : 0;` ตรวจสอบว่ามีพารามิเตอร์ `id` ถูกส่งมาทาง URL หรือไม่

ถ้ามี → แปลงเป็นตัวเลขเก็บไว้ใน `$id` ถ้าไม่มี → ตั้งค่า `$id = 0` หมายถึง “เพิ่มข้อมูลใหม่”

`$item =`

`['category'=>'student_location','code'=>'', 'name'=>'', 'tambon'=>'', 'amphoe'=>'', 'province'=>'', 'lat'=>'', 'lon'=>''];` สร้างตัวแปร `$item` เก็บข้อมูลเริ่มต้นของฟอร์มไว้ (ค่าจะว่างหมด ยกเว้น `category` มีค่าเริ่มต้นเป็น `student_location`)

```

- if ($id>0){

    $stmt = $pdo->prepare("SELECT * FROM locations WHERE id=:id");
    $stmt->execute([':id'=>$id]);

    $item = $stmt->fetch();

    if (!$item){ die('ไม่พบข้อมูล'); }

}

```

ถ้า \$id มากกว่า 0 (แสดงว่ามี id ส่งมา → เป็นโหมดแก้ไข)

ดึงข้อมูลจากตาราง locations โดยใช้คำสั่ง SQL

ถ้าหาข้อมูลไม่เจอ → แสดงข้อความ “ไม่พบข้อมูล” แล้วหยุดทำงานทันที

ถ้าเจอ → เก็บข้อมูลนั้นไว้ใน \$item เพื่อเอาไปแสดงในฟอร์มด้านล่าง

ตรงนี้เป็นหน้าตาของแบบฟอร์มกรอกข้อมูล

<title><?= \$id>0?'แก้ไข':'เพิ่ม' ?> · Location</title>

ถ้ามี id → ขึ้นว่า “แก้ไข” ถ้าไม่มี → ขึ้นว่า “เพิ่ม”

<form method="post" action="save.php">

เมื่อกดปุ่ม “บันทึก” ข้อมูลจะถูกส่งแบบ POST ไปที่ไฟล์ save.php เพื่อจัดการบันทึกลงฐานข้อมูล

ในฟอร์มจะมีช่องกรอกหลายช่อง เช่น

- หมวด (category) → ให้เลือกเป็น seven หรือ student_location
- รหัส (code)
- ชื่อ (name)
- ตำบล (tambon), อำเภอ (amphoe), จังหวัด (province)
- พิกัดละติจูด (lat) และลองจิจูด (lon)

โดยแต่ละช่องจะใส่ค่าจาก `$item[...]` ที่โหลดมาจากฐานข้อมูล เพื่อให้เวลาจะแก้ไข ข้อมูลเดิมจะถูกกรอกให้พร้อมแล้ว

ส่วนท้ายของฟอร์มมีปุ่ม

`<button class="btn btn-primary" type="submit">บันทึก</button>`

`ยกเลิก`

ปุ่ม “บันทึก” → ส่งข้อมูลไป `save.php`

ปุ่ม “ยกเลิก” → กลับไปหน้า `index.php` (น่าจะเป็นหน้ารายการทั้งหมด)

การดึงข้อมูลจากฐานข้อมูล **locations**

แล้วแสดงในตาราง โดยสามารถ “ค้นหา” และ “กรองตามหมวด” ได้

และมีปุ่มให้ “เพิ่ม”, “แก้ไข”, “ลบ” แต่ละรายการได้

```
index.php
1 <?php
2 require_once __DIR__ . '/auth.php';
3 require_once __DIR__ . '/../api/db.php';
4 $category = $_GET['category'] ?? '';
5 $q = trim($_GET['q'] ?? '');
6
7 $sql = "SELECT * FROM locations";
8 $conds = [];
9 $params = [];
10 if ($category !== '') { $conds[] = "category = :category"; $params[':category'] = $category; }
11 if ($q !== '') { $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))"; $params[':q'] = '%'.$q.'%'; }
12 if ($conds) { $sql .= " WHERE ".implode(" AND ", $conds); }
13 $sql .= " ORDER BY id DESC";
14 $stmt = $pdo->prepare($sql);
15 $stmt->execute($params);
16 $rows = $stmt->fetchAll();
17 >
18 <!doctype html>
19 <html lang="th">
20 <head>
21 <meta charset="utf-8">
22 <meta name="viewport" content="width=device-width, initial-scale=1">
23 <title>Admin · Locations</title>
24 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
25 <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@300;400;600;700&display=swap" rel="stylesheet">
```

เรียกไฟล์อื่นเข้ามา

`require_once __DIR__ . '/auth.php';` `auth.php` ตรวจสอบว่าผู้ใช้ล็อกอินอยู่หรือยัง (กันคนแอบเข้า)

`require_once __DIR__ . '/../api/db.php'; db.php` ไฟล์เชื่อมต่อฐานข้อมูล (มีตัวแปร `$pdo` ใช้ติดต่อฐานข้อมูล)

รับค่าค้นหาจาก URL

```
$category = $_GET['category'] ?? '';
```

```
$q = trim($_GET['q'] ?? '');
```

ถ้ามี `?category=...` หรือ `?q=...` ใน URL ก็เอาค่านั้นมาใช้ `$category` คือหมวด (เช่น `seven` หรือ `student_location`) `$q` คือคำค้น (เช่น ชื่อหรือรหัส)

เตรียม SQL สำหรับดึงข้อมูล

```
$sql = "SELECT * FROM locations";
```

```
$conds = [];
```

```
$params = [];
```

เริ่มต้นด้วย SQL พื้นฐาน แล้วค่อย ๆ เพิ่มเงื่อนไขถ้ามีการค้นหา

กำหนดเงื่อนไขตามค่าที่รับมา

```
if ($category !== '') {
```

```
    $conds[] = "category = :category";
```

```
    $params[':category'] = $category;
```

```
}
```

```
if ($q !== '') {
```

```
    $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))";
```

```
    $params[':q'] = '%'.$q.'%';
```

```
}
```

ถ้ามีเลือกหมวด → กรองข้อมูลเฉพาะหมวดนั้น

ถ้ามีคำค้น → ค้นทั้งในช่องชื่อ (name) และรหัส (code)

ใช้ LIKE และ LOWER() เพื่อค้นหาแบบไม่สนตัวพิมพ์ใหญ่-เล็ก

รวมเงื่อนไขเข้ากับ SQL

```
if ($conds) {  
    $sql .= " WHERE ".implode(" AND ", $conds);  
}
```

```
$sql .= " ORDER BY id DESC";
```

ถ้ามีเงื่อนไขมากกว่า 1 อัน จะเชื่อมด้วย AND เรียงผลลัพธ์จากรายการล่าสุดก่อน (id จากมากไปน้อย)

รันคำสั่ง SQL

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->execute($params);
```

```
$rows = $stmt->fetchAll();
```

เตรียมคำสั่ง SQL และค่าพารามิเตอร์ให้ปลอดภัยจาก SQL Injection

ดึงผลลัพธ์ทั้งหมดเก็บไว้ใน \$rows (เป็น array ของรายการทั้งหมด)

ส่วน HTML (หน้าเว็บจริง)

แถบเมนูด้านบน

```
<nav class="navbar navbar-expand-lg bg-white">
```

...

```
<span>ผู้ใช้: <b><?=$_SESSION['user']['username']?></b></span>
```

```
<a href="logout.php">ออกจากระบบ</a>
```

```
</nav>
```

แสดงชื่อผู้ใช้ที่ล็อกอินอยู่

มีปุ่ม “ออกจากระบบ”

มีปุ่ม “เปิดหน้าแผนที่” (ไปหน้า index.html ภายนอก)

ฟอร์มค้นหา

```
<form class="row g-2 align-items-end">

  <select name="category">...</select>

  <input name="q" placeholder="ชื่อ / รหัส">

  <button>ค้นหา</button>

  <a href="index.php">ล้าง</a>

</form>
```

ใช้กรองข้อมูลตามหมวดหรือคำค้น ปุ่ม “ล้าง” จะรีเฟรชหน้ากลับมาเหมือนตอนยังไม่กรอง

ส่วนแสดงรายการในตาราง

```
<h4>รายการจุดทั้งหมด (<?= count($rows) ?>)</h4>

<a class="btn btn-success" href="form.php">+ เพิ่มข้อมูล</a>
```

บอกจำนวนทั้งหมดที่ค้นเจอ ปุ่ม “+ เพิ่มข้อมูล” → ไปหน้า form.php เพื่อเพิ่มข้อมูลใหม่

ตารางข้อมูล

```
<table>

  <tr>

    <th>#</th><th>หมวด</th><th>รหัส</th><th>ชื่อ</th>...

  </tr>

  <?php foreach ($rows as $r): ?>

    <tr>

      <td><?= $r['id'] ?></td>

      <td><span class="badge ..."><?= htmlspecialchars($r['category']) ?></span></td>
```

```

<td><?= htmlspecialchars($r['code']) ?></td>

...

<td class="text-end">

    <a href="form.php?id=<?= $r['id'] ?>">แก้ไข</a>

    <a href="remove.php?id=<?= $r['id'] ?>" onclick="return confirm('ลบข้อมูลรายการนี้?')">ลบ
</a>

</td>

</tr>

<?php endforeach; ?>

</table>

```

วนลูปแสดงแต่ละรายการจาก \$rows

มีปุ่ม “แก้ไข” → ไปหน้า form.php?id=...

มีปุ่ม “ลบ” → ไปหน้า remove.php?id=... และจะถามยืนยันก่อนลบจริง

ใช้ htmlspecialchars() เพื่อกันโค้ดอันตราย (XSS)

ฟอร์มล็อกอิน

```
login_check.php
1  <?php
2  if (session_status() === PHP_SESSION_NONE) { session_start(); }
3  $user = $_POST['username'] ?? '';
4  $pass = $_POST['password'] ?? '';
5  $next = $_POST['next'] ?? 'index.php';
6
7  // Hardcoded credentials
8  if ($user === 'admin' && $pass === 'admin1234') {
9      $_SESSION['user'] = ['username' => 'admin'];
10     header("Location: " . $next);
11     exit;
12 }
13 header("Location: login.php?err=1&next=" . urlencode($next));
14 exit;
15 ?>
```

โค้ดนี้คือสคริปต์ที่รับฟอร์มล็อกอิน (ส่งมาด้วย POST) แล้วตรวจว่าชื่อผู้ใช้งานที่ส่งมาตรงกับข้อมูลที่ฝังไว้ในโค้ดหรือไม่ ถ้าตรงก็เก็บข้อมูลผู้ใช้งานไว้ใน `$_SESSION` แล้วพาไปที่หน้า `$next` ถ้าไม่ตรงก็ส่งกลับไปหน้า `login.php` พร้อมพารามิเตอร์ `err=1` และ `next` เพื่อบอกว่าล็อกอินไม่ผ่าน

```
if (session_status() === PHP_SESSION_NONE) { session_start(); }
```

— ถ้ายังไม่เริ่ม session ให้เริ่ม (เก็บสถานะผู้ใช้)

```
$user = $_POST['username'] ?? ''; / $pass = $_POST['password'] ?? '';
```

— รับค่าจากฟอร์ม ถ้าไม่มีให้เป็นสตริงว่าง

```
$next = $_POST['next'] ?? 'index.php';
```

— หน้าเป้าหมายที่จะไปหลังล็อกอิน (ถ้าไม่ได้ระบุไว้จะไป `index.php`)

```
if ($user === 'admin' && $pass === 'admin1234') { ... }
```

— ตรวจเงื่อนไข: ถ้าชื่อผู้ใช้งาน = admin และรหัสผ่าน = admin1234 → เก็บ `$_SESSION['user']` แล้ว

```
header("Location: " . $next); exit;
```

ถ้าไม่ผ่าน → รีไดเร็กกลับไป `login.php?err=1&next=...`

ฟอร์มเข้าสู่ระบบ (login.php)

```
login.php
1  <?php
2  if (session_status() === PHP_SESSION_NONE) { session_start(); }
3  $err = $_GET['err'] ?? '';
4  $next = $_GET['next'] ?? 'index.php';
5  ?>
6  <!doctype html>
7  <html lang="th">
8  <head>
9      <meta charset="utf-8">
10     <meta name="viewport" content="width=device-width, initial-scale=1">
11     <title>เข้าสู่ระบบ - Admin</title>
12     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
13     <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@300;400;600;700&display=swap" rel="stylesheet">
14     <style>
15         body{font-family:'Prompt',system-ui,Arial,sans-serif;background:linear-gradient(135deg,#eef4ff,#ffffff)}
16         .card{border:none;border-radius:1rem;box-shadow:0 15px 40px rgba(13,110,253,.12)}
17     </style>
18 </head>
19 <body class="d-flex align-items-center" style="min-height:100vh">
20     <div class="container">
21         <div class="row justify-content-center">
```

โค้ดนี้คือหน้า ฟอร์มเข้าสู่ระบบ (login.php) ที่ทำงานร่วมกับสคริปต์ login_check.php

ส่วนที่ 1 — เริ่มต้นและรับค่า err / next

```
if (session_status() === PHP_SESSION_NONE) { session_start(); }
```

```
$err = $_GET['err'] ?? '';
```

```
$next = $_GET['next'] ?? 'index.php';
```

เริ่ม session (กรณีที่ยังไม่มี)

\$err ใช้ตรวจสอบว่ามี error (ล็อกอินไม่ถูกต้อง) หรือไม่

\$next ใช้เก็บ URL ที่จะกลับไปหลังล็อกอินสำเร็จ (ถ้าไม่มีให้กลับหน้า index.php)

ส่วนที่ 2 — HTML และ Bootstrap UI

```
<!doctype html>
```

```
<html lang="th">
```

```
...
```

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
```

```
rel="stylesheet">
```

```
<link
href="https://fonts.googleapis.com/css2?family=Prompt:wght@300;400;600;700&display=swap"
rel="stylesheet">
```

ใช้ Bootstrap 5.3 จัด layout และ style

ใช้ฟอนต์ Prompt เพื่อให้เข้ากับภาษาไทย

พื้นหลังเป็น ไกล่สีอ่อน (gradient) ดูสะอาดตา

กล่อง login (.card) มีเงาและขอบโค้งมน

ส่วนที่ 3 — โครงสร้างฟอร์ม login

```
<form method="post" action="login_check.php">

  <input type="hidden" name="next" value="<?= htmlspecialchars($next) ?>">

  <div class="mb-3">

    <label class="form-label">Username</label>

    <input class="form-control" name="username" required autofocus>

  </div>

  <div class="mb-3">

    <label class="form-label">Password</label>

    <input class="form-control" name="password" type="password" required>

  </div>

  <button class="btn btn-primary w-100" type="submit">เข้าสู่ระบบ</button>

</form>
```

เมื่อผู้ใช้กรอกข้อมูลแล้วกด submit → ฟอร์มจะส่งไปยัง login_check.php ด้วยวิธี POST

มี input ซ่อน (hidden) สำหรับส่งค่า \$next กลับไปด้วย

ช่อง username และ password ถูกบังคับ (required) เพื่อป้องกันการส่งค่าว่าง

ส่วนที่ 4 — แสดงข้อความ **error**

```
<?php if($err): ?>
```

```
<div class="alert alert-danger py-2">ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง</div>
```

```
<?php endif; ?>
```

ถ้ามีพารามิเตอร์ `?err=1` มาจาก `login_check.php` → จะแสดงกล่องข้อความสีแดงแจ้งเตือน

ส่วนที่ 5 — ตัวอย่างชื่อผู้ใช้

```
<div class="small text-muted text-center mt-3">
```

```
user: <code>admin</code> · pass: <code>admin1234</code>
```

```
</div>
```

แสดงตัวอย่าง `user/password` ที่ถูกต้อง (ตามที่ `hardcode` ไว้ใน `login_check.php`)

ช่วยให้ทดสอบง่ายตอนพัฒนา

ออกจากระบบ (`logout.php`)

```
logout.php
1  <?php
2  if (session_status() === PHP_SESSION_NONE) { session_start(); }
3  session_destroy();
4  header("Location: login.php");
5  exit;
6  ?>
```

เริ่มต้น **session** (ถ้ายังไม่มี)

```
if (session_status() === PHP_SESSION_NONE) { session_start(); }
```

ก่อนจะทำอะไรกับ `session` ต้องมั่นใจว่า `session` ถูกเริ่มไว้แล้ว

บางครั้งหน้า `logout` อาจถูกเรียกโดยตรง (เช่น คลิกลิงก์ `logout.php`) → จึงต้องเช็คก่อน

ลบข้อมูล **session** ทั้งหมด

```
session_destroy();
```

ฟังก์ชันนี้จะ ทำลาย **session** ทั้งหมดของผู้ใช้ในฝั่งเซิร์ฟเวอร์

หมายความว่า ข้อมูลเช่น \$_SESSION['user'] จะถูกลบ

ผู้ใช้งานถือว่า “ออกจากระบบ” ทันที

เปลี่ยนเส้นทางกลับหน้า **login**

```
header("Location: login.php");
```

```
exit;
```

หลังจากทำลาย session เสร็จ → ระบบจะ **redirect** กลับไปที่หน้า **login.php**

exit; ใช้เพื่อหยุดการทำงานของสคริปต์ทันทีหลัง **redirect** (เป็นมาตรฐานความปลอดภัยที่ดี)

ลบข้อมูลออกจากฐานข้อมูล

โค้ดนี้คือสคริปต์ **remove.php** ที่ใช้สำหรับ

```
remove.php
1  <?php
2  require_once __DIR__ . '/auth.php';
3  require_once __DIR__ . '/../api/db.php';
4  $id = intval($_GET['id'] ?? 0);
5  if ($id>0){
6      $stmt = $pdo->prepare("DELETE FROM locations WHERE id=:id");
7      $stmt->execute([':id'=>$id]);
8  }
9  header("Location: index.php"); exit;
10 ?>
```

ตรวจสอบสิทธิ์และเชื่อมต่อกับฐานข้อมูล

```
require_once __DIR__ . '/auth.php';
```

```
require_once __DIR__ . '/../api/db.php';
```


auth.php : ตรวจสอบว่าผู้ใช้ล็อกอินแล้วหรือยัง→ ถ้ายังไม่ล็อกอิน จะ redirect ไปหน้า login

db.php : ไฟล์นี้เชื่อมต่อฐานข้อมูลผ่าน \$pd

ดึงค่า id ที่ต้องการลบจาก URL

```
$id = intval($_GET['id'] ?? 0);
```

ใช้ \$_GET['id'] รับค่ามาจากพารามิเตอร์ใน URL เช่น remove.php?id=12

remove.php?id=12

ใช้ intval() แปลงให้เป็นตัวเลข เพื่อป้องกันการแทรกโค้ดอันตราย (SQL Injection)

ถ้ามีค่า id มากกว่า 0 → สั่งลบข้อมูล

```
if ($id>0){
```

```
    $stmt = $pdo->prepare("DELETE FROM locations WHERE id=:id");
```

```
    $stmt->execute([':id'=>$id]);
```

```
}
```

เงื่อนไขนี้ช่วยกันความผิดพลาด เช่น ถ้าไม่มี id จะไม่ทำงาน

ใช้ Prepared Statement (:id) ป้องกัน SQL Injection ได้อย่างปลอดภัย

คำสั่ง DELETE จะลบข้อมูลจากตาราง locations ที่มี id ตรงกับค่าที่รับมา

กลับไปหน้า index หลังลบเสร็จ

```
header("Location: index.php");
```

```
exit;
```

หลังจากลบเสร็จ ระบบจะ redirect กลับหน้า index.php (หน้ารายการข้อมูล)

```
exit;
```

หยุดการทำงานของสคริปต์หลัง redirect เพื่อความปลอดภัย

บันทึกข้อมูลลงฐานข้อมูล

โค้ดนี้คือไฟล์ save.php — หน้าที่ของมันคือ “บันทึกข้อมูลลงฐานข้อมูล” จะใช้ทั้งตอน เพิ่มข้อมูลใหม่ และ แก้ไขข้อมูลเดิม ในตาราง locations

```
<?php
require_once __DIR__ . '/auth.php';
require_once __DIR__ . '/../api/db.php';
$id = intval($_POST['id'] ?? 0);
$category = $_POST['category'] ?? 'student_location';
$code = $_POST['code'] ?? '';
$name = $_POST['name'] ?? '';
$tambon = $_POST['tambon'] ?? '';
$amphoe = $_POST['amphoe'] ?? '';
$province = $_POST['province'] ?? '';
$lat = ($_POST['lat'] != '' ? floatval($_POST['lat']) : null);
$lon = ($_POST['lon'] != '' ? floatval($_POST['lon']) : null);

if ($id>0){
    $sql = "UPDATE locations SET category=:category, code=:code, name=:name, tambon=:tambon, amphoe=:amphoe, province=:province, lat=:lat";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([':category'=>$category, ':code'=>$code, ':name'=>$name, ':tambon'=>$tambon, ':amphoe'=>$amphoe, ':province'=>$province, ':lat'=>$lat, ':lon'=>$lon]);
} else {
    $sql = "INSERT INTO locations(category, code, name, tambon, amphoe, province, lat, lon) VALUES(:category,:code,:name,:tambon,:amphoe,:province,:lat,:lon)";
    $stmt = $pdo->prepare($sql);
    $stmt->execute([':category'=>$category, ':code'=>$code, ':name'=>$name, ':tambon'=>$tambon, ':amphoe'=>$amphoe, ':province'=>$province, ':lat'=>$lat, ':lon'=>$lon]);
}
header("Location: index.php"); exit;
?>
```

เช็กลิสต์และเชื่อมต่อฐานข้อมูล

```
require_once __DIR__ . '/auth.php';
```

```
require_once __DIR__ . '/../api/db.php';
```

ตรวจสอบก่อนว่าผู้ใช้ล็อกอินแล้วหรือยัง (auth.php) ถ้ายังไม่ได้ล็อกอินจะถูกส่งกลับไปหน้า login.php จากนั้นเชื่อมต่อฐานข้อมูล (db.php) เพื่อใช้ตัวแปร \$pdo

ดึงค่าที่ส่งมาจากฟอร์ม

```
$id = intval($_POST['id'] ?? 0);
```

```
$category = $_POST['category'] ?? 'student_location';
```

```
$code = $_POST['code'] ?? '';
```

```
$name = $_POST['name'] ?? '';
```

```
$tambon = $_POST['tambon'] ?? '';
```

```
$amphoe = $_POST['amphoe'] ?? '';
```

```
$province = $_POST['province'] ?? '';
```

```
$lat = ($_POST['lat'] != "" ? floatval($_POST['lat']) : null);
```

```
$lon = ($_POST['lon'] != "" ? floatval($_POST['lon']) : null);
```

ดึงค่าจากฟอร์ม เช่น ชื่อ, ตำบล, ละติจูด, ลองจิจูด ถ้ามีค่า id → แสดงว่าเป็นการ “แก้ไขข้อมูล” ถ้าไม่มี id → แสดงว่าเป็น “การเพิ่มข้อมูลใหม่” floatval() แปลงตัวเลขเป็นค่าทศนิยม เช่น 16.12345 ถ้าช่องละติจูด/ลองจิจูดว่าง → ให้เป็น null (ไม่ต้องบังคับกรอก)

ถ้ามี id แสดงว่าเป็น “การแก้ไขข้อมูล”

```
if ($id>0){
```

```
    $sql = "UPDATE locations
```

```
        SET category=:category, code=:code, name=:name,
```

```
        tambon=:tambon, amphoe=:amphoe, province=:province,
```

```
        lat=:lat, lon=:lon
```

```
        WHERE id=:id";
```

```
    $stmt = $pdo->prepare($sql);
```

```
    $stmt->execute([...]);
```

```
}
```

ใช้คำสั่ง UPDATE เพื่อแก้ไขข้อมูลแถวเดิมในตาราง locations

ใช้ prepared statement เพื่อป้องกัน SQL Injection

ตรง WHERE id=:id คือกำหนดว่าจะอัปเดตแถวไหน

ถ้าไม่มี id → เป็น “การเพิ่มข้อมูลใหม่”

```
else {
```

```
    $sql = "INSERT INTO locations(category, code, name, tambon, amphoe, province, lat, lon)
```

```
        VALUES(:category,:code,:name,:tambon,:amphoe,:province,:lat,:lon)";
```

```
    $stmt = $pdo->prepare($sql);
```

```
$stmt->execute([...]);  
}
```

ใช้คำสั่ง INSERT INTO เพื่อเพิ่มแถวใหม่ในตาราง เก็บข้อมูลใหม่ทั้งหมดจากฟอร์ม

เมื่อบันทึกเสร็จ → กลับไปหน้า index.php

```
header("Location: index.php");
```

```
exit;
```

redirect ผู้ใช้กลับไปหน้ารายการ (index.php) เพื่อดูผลการบันทึก exit; ป้องกันไม่ให้สคริปต์ทำงานต่อ

เชื่อมต่อฐานข้อมูล PostgreSQL

โค้ดนี้คือไฟล์ db.php — ใช้สำหรับ “เชื่อมต่อฐานข้อมูล PostgreSQL” ผ่าน PHP (PDO)

```
db.php  
1 <?php  
2 require_once __DIR__ . '/env.php';  
3 try {  
4     $dsn = "pgsql:host=$DB_HOST;port=$DB_PORT;dbname=$DB_NAME;";  
5     $pdo = new PDO($dsn, $DB_USER, $DB_PASS, [  
6         PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,  
7         PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC  
8     ]);  
9 } catch (Throwable $e) {  
10     header("Content-Type: application/json; charset=UTF-8");  
11     http_response_code(500);  
12     echo json_encode(["error" => "DB connect failed", "detail" => $e->getMessage()], JSON_UNESCAPED_UNICODE);  
13     exit;  
14 }  
15 ?>
```

ตั้งค่าตัวแปรจากไฟล์ env.php

```
require_once __DIR__ . '/env.php';
```

ไฟล์ env.php จะเก็บค่าพวกนี้ไว้ เช่น

```
<?php
```

```
$DB_HOST = "localhost";
```

```
$DB_PORT = "5432";
```

```
$DB_NAME = "gis_db";
```

```
$DB_USER = "postgres";
```

```
$DB_PASS = "1234";
```

```
?>
```

พยายามเชื่อมต่อฐานข้อมูล (try)

```
$dsn = "pgsql:host=$DB_HOST;port=$DB_PORT;dbname=$DB_NAME;";
```

```
$pdo = new PDO($dsn, $DB_USER, $DB_PASS, [  
    PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,  
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC  
]);
```

ใช้ PDO (PHP Data Object) ซึ่งเป็นวิธีเชื่อมต่อฐานข้อมูลที่ปลอดภัยและยืดหยุ่น

pgsql: คือ driver สำหรับ PostgreSQL

ตั้งค่าเพิ่มเติม:

```
PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
```

→ ถ้ามี error จะโยน exception ออกมาให้จับได้

```
PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
```

→ เวลา fetch ข้อมูล จะได้ array แบบชื่อคอลัมน์ (ไม่ใช่ตัวเลข)

ถ้าเชื่อมต่อไม่ได้ → แสดง error แบบ JSON

```
} catch (Throwable $e) {
```

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

```
    http_response_code(500);
```

```
    echo json_encode([
```

```
        "error" => "DB connect failed",
```

```
        "detail" => $e->getMessage()
```

```
], JSON_UNESCAPED_UNICODE);
```

```
exit;
```

```
}
```

ถ้าโค้ดมีปัญหา เช่น ใส่รหัสผ่านผิด หรือฐานข้อมูลล่ม โค้ดใน `catch` จะทำงานแทน ส่งกลับข้อความ `error` เป็น `JSON` (เหมาะกับเว็บ API มาก) `http_response_code(500)` คือรหัสสถานะ “เซิร์ฟเวอร์ผิดพลาด”

เก็บข้อมูลสำหรับเชื่อมฐานข้อมูลไว้ในเวลาระบบ

โค้ดนี้เป็นที่เก็บข้อมูลสำหรับเชื่อมฐานข้อมูลไว้ในเวลาระบบ (เช่น `index.php`, `save.php`, `remove.php`) ต้องต่อฐานข้อมูล มันจะเรียกใช้ไฟล์ `db.php` ซึ่งจะมาอ่านค่าจากไฟล์นี้อีกที

```
env.php
1  <?php
2  // === DB CONFIG (แก้ค่าให้ตรงกับเครื่องคุณ) ===
3  $DB_HOST = 'localhost';
4  $DB_PORT = '5433';
5  $DB_NAME = 'webgis_db';
6  $DB_USER = 'postgres';
7  $DB_PASS = '1234';
8  ?>
```

`$DB_HOST = 'localhost';` คือชื่อ **โฮสต์ (Host)** ของฐานข้อมูล ถ้าใช้ฐานข้อมูลในเครื่องเดียวกันกับ PHP → ใช้ `localhost` ได้เลยถ้าอยู่บนเซิร์ฟเวอร์อื่น อาจเป็นชื่อโดเมนหรือ IP เช่น

`192.168.1.10` หรือ `db.myserver.com`

`$DB_PORT = '5433';`

คือ **พอร์ต (Port)** ที่ PostgreSQL ใช้งาน ปกติค่าเริ่มต้นคือ `5432` แต่ของคุณตั้งไว้เป็น `5433` → ต้องให้ตรงกับค่าที่ตั้งในฐานข้อมูลของเครื่องคุณ

`$DB_NAME = 'webgis_db';`

ชื่อของฐานข้อมูลที่คุณสร้างไว้ใน PostgreSQL เช่นใน pgAdmin จะเห็นชื่อ `database` ตรงนี้

`$DB_USER = 'postgres';`

```
$DB_PASS = '1234';
```

คือชื่อผู้ใช้และรหัสผ่านที่ใช้เข้าถึงฐานข้อมูล ปกติผู้ใช้หลักของ PostgreSQL คือ postgres ส่วนรหัสผ่าน (1234) ต้องตรงกับที่ตั้งไว้ตอนติดตั้ง

API

ไฟล์นี้คือ API endpoint ที่ใช้ดึงข้อมูลจากฐานข้อมูล PostgreSQL ส่งผลลัพธ์ออกมาในรูปแบบ JSON

สามารถใช้ได้กับ JavaScript (fetch, axios, Leaflet

```
locations.php
1  <?php
2  header("Content-Type: application/json; charset=UTF-8");
3  header("Access-Control-Allow-Origin: *");
4  require_once __DIR__ . '/db.php';
5
6  $category = $_GET['category'] ?? null;
7  $q = trim($_GET['q'] ?? '');
8
9  $sql = "SELECT id, category, code, name, tambon, amphoe, province, lat, lon FROM locations";
10 $conds = [];
11 $params = [];
12 if ($category) { $conds[] = "category = :category"; $params[':category'] = $category; }
13 if ($q !== '') {
14     $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))";
15     $params[':q'] = '%'.$q.'%';
16 }
17 if ($conds) { $sql .= " WHERE ".implode(" AND ", $conds); }
18 $sql .= " ORDER BY name ASC";
19
20 $stmt = $pdo->prepare($sql);
21 $stmt->execute($params);
22 echo json_encode($stmt->fetchAll(), JSON_UNESCAPED_UNICODE | JSON_PRETTY_PRINT);
23 ?>
```

ตั้งค่าหัวของ Response

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

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

บอก browser ว่าข้อมูลที่ส่งกลับเป็น **JSON UTF-8** เปิดให้เว็บภายนอกเรียกใช้ API ได้ (CORS)

เรียกไฟล์เชื่อมต่อฐานข้อมูล

```
require_once __DIR__ . '/db.php';
```

โหลดการเชื่อมต่อ \$pdo ที่เตรียมไว้ใน db.php ใช้เชื่อมต่อกับฐานข้อมูล PostgreSQL

อ่านค่าพารามิเตอร์จาก URL

```
$category = $_GET['category'] ?? null;
```

```
$q = trim($_GET['q'] ?? '');
```

ดึงค่าจาก URL เช่น /api/list.php?category=school&q= ถ้าไม่มีค่าก็จะใช้ค่าเริ่มต้นเป็น null หรือ string ว่าง

สร้างคำสั่ง SQL ตามเงื่อนไข

```
$sql = "SELECT id, category, code, name, tambon, amphoe, province, lat, lon FROM locations";
```

```
$conds = [];
```

```
$params = [];
```

```
if ($category) {
```

```
    $conds[] = "category = :category";
```

```
    $params[':category'] = $category;
```

```
}
```

```
if ($q !== "") {
```

```
    $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))";
```

```
    $params[':q'] = '%'.$q.'%';
```

```
}
```

```
if ($conds) {
```

```
    $sql .= " WHERE ".implode(" AND ", $conds);
```

```
}
```

```
$sql .= " ORDER BY name ASC";
```

สร้าง SQL แบบ dynamic ป้องกัน SQL Injection ด้วย prepared statements เงื่อนไขจะถูกเพิ่มเฉพาะเมื่อมีค่าพารามิเตอร์

ดึงข้อมูลและส่งออก

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->execute($params);
```

```
echo json_encode($stmt->fetchAll(), JSON_UNESCAPED_UNICODE | JSON_PRETTY_PRINT);
```

เตรียมและรันคำสั่ง SQL

แปลงผลลัพธ์เป็น JSON

JSON_UNESCAPED_UNICODE → แสดงภาษาไทยได้ตรง

JSON_PRETTY_PRINT → ทำให้อ่านง่ายเวลาทดสอบใน browser

หน้า admin/index.php

ใช้สำหรับให้ผู้ดูแลระบบเข้ามาค้นหา / กรองข้อมูลตำแหน่งดูรายการทั้งหมดในฐานข้อมูลแก้ไข หรือลบจุดได้เพิ่มข้อมูลใหม่

```
admin > index.php
1  <?php
2  require_once __DIR__ . '/auth.php';
3  require_once __DIR__ . '/../api/db.php';
4  $category = $_GET['category'] ?? '';
5  $q = trim($_GET['q'] ?? '');
6
7  $sql = "SELECT * FROM locations";
8  $conds = [];
9  $params = [];
10 if ($category != '') { $conds[] = "category = :category"; $params[':category'] = $category; }
11 if ($q != '') { $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))"; $params[':q'] = '%'.$q.'%'; }
12 if ($conds) { $sql .= " WHERE ".implode(" AND ", $conds); }
13 $sql .= " ORDER BY id DESC";
14 $stmt = $pdo->prepare($sql);
15 $stmt->execute($params);
16 $rows = $stmt->fetchAll();
17 ?>
18 <!doctype html>
19 <html lang="th">
20 <head>
21   <meta charset="utf-8">
22   <meta name="viewport" content="width=device-width, initial-scale=1">
23   <title>Admin - Locations</title>
24   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
25   <link href="https://fonts.googleapis.com/css2?family=Prompt:wght@300;400;600;700&display=swap" rel="stylesheet">
26   <style>
27     body{font-family:'Prompt',system-ui,Arial,sans-serif;background:#f7f8fb;}
28     .card{border:none;border-radius:1rem;box-shadow:0 4px 20px rgba(0,0,0,.06)}
29     .navbar{border-bottom:1px solid #eaecef}
30     .badge-cat{font-weight:600}
31   </style>
32 </head>
33 <body>
34 <nav class="navbar navbar-expand-lg bg-white">
```

ตรวจสอบสิทธิ์การเข้าถึง

```
require_once __DIR__ . '/auth.php';
```

```
require_once __DIR__ . '/../api/db.php';
```

auth.php → ตรวจสอบว่าเข้าสู่ระบบแล้วหรือไม่ ถ้ายังไม่ได้ login จะ redirect ไปหน้า login.php

db.php → เชื่อมต่อกับฐานข้อมูล PostgreSQL ผ่าน PDO

ดึงค่าจาก URL และสร้าง SQL

```
$category = $_GET['category'] ?? '';
```

```
$q = trim($_GET['q'] ?? '');
```

`$category` หมวดข้อมูล (เช่น `seven`, `student_location`) `$q` คือคำค้น (ชื่อหรือรหัส)

จากนั้นสร้าง SQL:

```
$sql = "SELECT * FROM locations";
```

```
$conds = [];
```

```
$params = [];
```

```
if ($category !== "") { $conds[] = "category = :category"; $params[':category'] = $category; }
```

```
if ($q !== "") { $conds[] = "(LOWER(name) LIKE LOWER(:q) OR LOWER(code) LIKE LOWER(:q))";  
$params[':q'] = '%'.$q.'%'; }
```

```
if ($conds) { $sql .= " WHERE ".implode(" AND ", $conds); }
```

```
$sql .= " ORDER BY id DESC";
```

หมายถึง — ถ้าเลือก `category` หรือพิมพ์คำค้น ก็จะมีการกรองผลลัพธ์ ใช้ `prepared statement` เพื่อป้องกัน SQL Injection

ดึงข้อมูลจากฐานข้อมูล

```
$stmt = $pdo->prepare($sql);
```

```
$stmt->execute($params);
```

```
$rows = $stmt->fetchAll();
```

ดึงผลลัพธ์ทั้งหมดจากตาราง `locations` เก็บไว้ในตัวแปร `$rows` เพื่อแสดงในตาราง

ส่วน **HTML (Bootstrap UI)**

```
<nav class="navbar navbar-expand-lg bg-white">
```

...

```
<a class="btn btn-outline-danger btn-sm" href="logout.php">ออกจากระบบ</a>
```

```
</nav>
```

แสดงชื่อระบบ “Admin · จัดการจุด”

ปุ่ม “↗ เปิดหน้าแผนที่” → เปิด `index.html` (หน้าแสดงแผนที่หลัก)

แสดงชื่อผู้ใช้ที่ล็อกอิน (\$_SESSION['user']['username'])

ปุ่ม “ออกจากระบบ” → logout.php

ฟอร์มค้นหา

```
<form class="row g-2 align-items-end">

  <select class="form-select" name="category">...</select>

  <input class="form-control" name="q" placeholder="ชื่อ / รหัส">

  <button class="btn btn-dark" type="submit">ค้นหา</button>

  <a class="btn btn-outline-secondary" href="index.php">ล้าง</a>

</form>
```

กรองตามหมวด (seven หรือ student_location) ค้นหาจากชื่อหรือรหัส (q) ปุ่ม “ล้าง” รีเซ็ตฟอร์มกลับหน้า index

ตารางแสดงข้อมูล

```
<table class="table table-hover align-middle m-0">

  <thead class="table-light">...</thead>

  <tbody>

    <?php foreach ($rows as $r): ?>

      <tr>

        <td><?= $r['id'] ?></td>

        <td><span class="badge ..."><?= htmlspecialchars($r['category']) ?></span></td>

        <td><?= htmlspecialchars($r['code']) ?></td>

        <td><?= htmlspecialchars($r['name']) ?></td>

        <td><?= htmlspecialchars($r['tambon']) ?></td>

        <td><?= htmlspecialchars($r['amphoe']) ?></td>
```

```

<td><?= htmlspecialchars($r['province']) ?></td>
<td><?= htmlspecialchars($r['lat']) ?></td>
<td><?= htmlspecialchars($r['lon']) ?></td>
<td class="text-end">
    <a href="form.php?id=<?= $r['id'] ?>">แก้ไข</a>
    <a href="remove.php?id=<?= $r['id'] ?>" onclick="return confirm('ลบข้อมูลรายการนี้?')">ลบ
</a>
</td>
</tr>
<?php endforeach; ?>
</tbody>
</table>

```

แสดงข้อมูลจากฐานข้อมูล ปุ่ม “แก้ไข” → เปิดหน้า form.php พร้อม id ปุ่ม “ลบ” → เรียก remove.php?id=...
ลบข้อมูล (มี confirm popup)

JavaScript

โค้ดนี้คือ สคริปต์ JavaScript หลักของหน้าเว็บแผนที่ (Web GIS)

โดยใช้ Leaflet.js สำหรับแสดงแผนที่ และดึงข้อมูลจาก api/locations.php มาแสดงเป็น marker

“สคริปต์ที่ทำให้แผนที่มีชีวิต” — โหลดข้อมูลจริง, ปักหมุด, ค้นหา และซูมให้อัตโนมัติ

```

js > JS app.js > loadData
1  let map, markers, sevIcon;
2
3  function initMap(){
4      map = L.map('map').setView([16.748, 100.199], 13);
5      L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {maxZoom:19}).addTo(map);
6      markers = L.layerGroup().addTo(map);
7
8      // Custom 7-11 icon
9      sevIcon = L.icon({
10         iconUrl: 'img/seven.png',
11         iconSize: [36, 36],
12         iconAnchor: [18, 34],
13         popupAnchor: [0, -28]
14     });
15
16     sevIcon2 = L.icon({
17         iconUrl: 'img/student.png',
18         iconSize: [36, 36],
19         iconAnchor: [18, 34],
20         popupAnchor: [0, -28]
21     });
22     loadData();
23 }
24

```

ประกาศตัวแปรหลัก

let map, markers, sevIcon;

map → ตัวแผนที่ Leaflet

markers → กลุ่มจุด (LayerGroup) ที่จะใช้เก็บ marker ทั้งหมด

sevIcon → ไอคอน 7-Eleven (กำหนดไว้ใช้ตอนสร้าง marker)

ฟังก์ชันเริ่มต้น initMap()

function initMap(){

map = L.map('map').setView([16.748, 100.199], 13);

L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {maxZoom:19}).addTo(map);

markers = L.layerGroup().addTo(map);

สร้างแผนที่ Leaflet ตรงจุดเริ่มต้น (พิกัดพิษณุโลกประมาณนี้) แล้วดึงพื้นหลังแผนที่จาก OpenStreetMap

ตั้งค่า “ไอคอนรูปภาพ” สำหรับแต่ละหมวด

```
sevlcon = L.icon({  
  imageUrl: 'img/seven.png',  
  iconSize: [36, 36],  
  iconAnchor: [18, 34],  
  popupAnchor: [0, -28]  
});
```

ใช้ภาพ `img/seven.png` สำหรับจุดร้าน 7-Eleven ขนาด 36x36 พิกเซลกำหนดจุด `anchor` เพื่อให้ “ปลายหมุด” อยู่ตรงพิกัดจริง

```
sevlcon2 = L.icon({  
  imageUrl: 'img/student.png',  
  ...  
});
```

ใช้ภาพ `img/student.png` สำหรับจุด `student_location` (เช่นจุดนักเรียน)

โหลดข้อมูลจากฐานข้อมูล

```
loadData();
```

เรียกฟังก์ชัน `loadData()` เพื่อไปดึงข้อมูล `marker` จาก API

ฟังก์ชัน `loadData()`

“ดึงข้อมูลจาก API แล้วแสดงบนแผนที่”

```
async function loadData(){  
  markers.clearLayers(); // ล้างหมุดเก่าทั้งหมดก่อน
```

สร้าง URL สำหรับเรียก API

```
const cat = document.getElementById('category').value;
```

```
const q = document.getElementById('search').value.trim();
const url = new URL('api/locations.php', window.location.href);
url.searchParams.set('category', cat);
if (q) url.searchParams.set('q', q);
```

ดึงค่าหมวด (category) และคำค้น (search) จาก input บนหน้าเว็บ สร้าง URL เช่น
api/locations.php?category=seven&q=phitsanulok

ดึงข้อมูลจาก API

```
const resp = await fetch(url.toString());
const rows = await resp.json();
```

เรียก API ผ่าน fetch แล้วแปลงผลลัพธ์เป็น JSON แต่ละแถว (row) คือข้อมูล 1 จุดในฐานข้อมูล

สร้าง marker ที่ละจุด

```
rows.forEach(r => {
  if (r.lat && r.lon){
    const icon = (r.category === 'seven') ? sevIcon : sevIcon2;

    const m = L.marker([r.lat, r.lon], icon?{icon}:undefined).addTo(markers);
    m.bindPopup(`
      <b>${r.name} || '(!ไม่มีชื่อ)')</b><br/>
      รหัส: ${r.code} || '-'<br/>
      ตำบล: ${r.tambon} || '-' | อำเภอ: ${r.amphoe} || '-'<br/>
      จังหวัด: ${r.province} || '-'
    `);
  }
});
```


ถ้ามีพิกัด lat/lon → สร้าง marker ถ้า category เป็น seven → ใช้ icon 7-Eleven แต่ถ้าเป็น student_location → ใช้ icon นักเรียน คลิก marker แล้วจะมี popup แสดงรายละเอียด

ซูมแผนที่ให้พอดี

```
if (markers.getLayers().length){  
    const g = L.featureGroup(markers.getLayers());  
    map.fitBounds(g.getBounds().pad(0.2));  
}
```

หลังจากปัดหมดแล้วระบบจะ “ซูมแผนที่ให้เห็นทุกจุดพอดีในจอ” โดยอัตโนมัติ

ฟังก์ชัน **clearSearch()**

```
function clearSearch(){  
    document.getElementById('search').value = "";  
    loadData();  
}
```

เคลียร์ช่องค้นหา แล้วโหลดข้อมูลใหม่ (แสดงทุกจุดทั้งหมด)

ฟังก์ชัน **exportCSV()**

```
function exportCSV(){  
    const cat = document.getElementById('category').value;  
    const q = document.getElementById('search').value.trim();  
    const url = new URL('api/locations.php', window.location.href);  
    url.searchParams.set('category', cat);  
    if (q) url.searchParams.set('q', q);  
    window.open(url.toString(), '_blank');  
}
```

ใช้ API เดิม (locations.php)เปิดข้อมูล JSON ในแท็บใหม่ผู้ใช้สามารถ “ก๊อปปี้ข้อมูลออกมาเก็บเป็น CSV/JSON” ได้เอง

เริ่มทำงานเมื่อน้ำโหลดเสร็จ

```
window.addEventListener('DOMContentLoaded', initMap);
```

หมายถึงเมื่อน้ำเว็บโหลดเสร็จ → ให้เรียก `initMap()` เพื่อเริ่มแสดงแผนที่และโหลดข้อมูลทันที