



รายงาน

คู่มือการพัฒนาเว็บไซต์

ผู้จัดทำ

นางสาว สุตาร์ตน์ สุขอยู่ รหัสนิต 66163297

เสนอ

รศ.ดร. สิทธิชัย ชูสำโรง

รายงานประกอบวิชา 104325 – 3 การพัฒนาแผนที่บนเว็บ

(Web GIS Development) ภาคการศึกษาที่ 1 ปีการศึกษา 2568

สาขาภูมิศาสตร์ คณะเกษตรศาสตร์ ทรัพยากรธรรมชาติและสิ่งแวดล้อม

มหาวิทยาลัยนครสวรรค์

บทนำ

ในปัจจุบัน เทคโนโลยีสารสนเทศมีบทบาทสำคัญอย่างยิ่งต่อการจัดการข้อมูล โดยเฉพาะข้อมูลที่มีลักษณะเชิงพื้นที่ (Spatial Data) ซึ่งเกี่ยวข้องกับตำแหน่งทางภูมิศาสตร์ เช่น พื้นที่ใช้สอย ถนน อาคาร หรือขอบเขตพื้นที่ต่าง ๆ การจัดเก็บ วิเคราะห์ และนำเสนอข้อมูลลักษณะนี้จำเป็นต้องอาศัยระบบสารสนเทศภูมิศาสตร์ หรือที่รู้จักกันในชื่อ GIS (Geographic Information System) ซึ่งช่วยให้สามารถนำข้อมูลเชิงพื้นที่มาวิเคราะห์ร่วมกับข้อมูลเชิงบรรยาย (Attribute Data) เพื่อสนับสนุนการตัดสินใจและการวางแผนได้อย่างมีประสิทธิภาพ

ในอดีต การใช้งานระบบ GIS มักจำกัดอยู่บนเครื่องคอมพิวเตอร์เฉพาะทาง (Desktop GIS) ซึ่งผู้ใช้งานต้องติดตั้งซอฟต์แวร์เฉพาะ เช่น ArcGIS หรือ QGIS จึงจะสามารถเข้าถึงข้อมูลได้ แต่ด้วยการพัฒนาอย่างรวดเร็วของเทคโนโลยีเว็บ ทำให้เกิดแนวทางใหม่ในการนำเสนอข้อมูล GIS ผ่านเครือข่ายอินเทอร์เน็ต เรียกว่า Web GIS ซึ่งช่วยให้ผู้ใช้งานสามารถเข้าถึงและโต้ตอบกับข้อมูลเชิงพื้นที่ได้ทุกที่ ทุกเวลา ผ่านเว็บเบราว์เซอร์โดยไม่ต้องติดตั้งโปรแกรมเพิ่มเติม

การพัฒนา ระบบ Web GIS จำเป็นต้องมียุคประกอบทางเทคโนโลยีหลายส่วนที่ทำงานร่วมกันอย่างเป็นระบบ ได้แก่

- ส่วนติดต่อผู้ใช้ (Frontend) ใช้เทคโนโลยี เช่น HTML, CSS, Bootstrap และ Leaflet.js เพื่อออกแบบและสร้างหน้าเว็บสำหรับแสดงผลแผนที่และข้อมูลต่าง ๆ ให้ผู้ใช้สามารถโต้ตอบได้อย่างสะดวก
- ส่วนประมวลผลและฐานข้อมูล (Backend / Database) ใช้ PHP สำหรับประมวลผลคำสั่งและเชื่อมต่อกับฐานข้อมูล PostgreSQL ซึ่งติดตั้งส่วนขยาย PostGIS เพื่อรองรับการจัดเก็บและวิเคราะห์ข้อมูลเชิงพื้นที่
- GIS Server ใช้ GeoServer สำหรับเผยแพร่ข้อมูลแผนที่ในรูปแบบมาตรฐาน เช่น WMS, WFS และ WCS ให้สามารถเรียกใช้งานผ่านเว็บหรือแอปพลิเคชันอื่นได้
- เครื่องมือช่วยพัฒนา (Tools) เช่น QGIS ใช้ในการเตรียมและตรวจสอบข้อมูลเชิงพื้นที่ และ Visual Studio Code (VS Code) ใช้ในการพัฒนาและจัดการซอร์สโค้ดของระบบ

ระบบ Web GIS ที่พัฒนาขึ้นจึงเป็นการผสมผสานระหว่างเทคโนโลยีเว็บและเทคโนโลยีภูมิสารสนเทศ ทำให้สามารถนำเสนอข้อมูลเชิงพื้นที่ในรูปแบบที่เข้าใจง่าย มีความสวยงาม และสามารถโต้ตอบกับผู้ใช้งานได้อย่างมีประสิทธิภาพ อีกทั้งยังช่วยเพิ่มศักยภาพในการวิเคราะห์และบริหารจัดการข้อมูลเชิงพื้นที่ให้กับองค์กรหรือหน่วยงานต่าง ๆ ได้อย่างทันสมัยและเหมาะสมกับยุคดิจิทัล

วัตถุประสงค์ของรายงาน (Objectives)

โครงการนี้มีวัตถุประสงค์เพื่อพัฒนาและนำเสนอระบบ Web GIS (Web-based Geographic Information System) ที่สามารถแสดงผลและจัดการข้อมูลเชิงพื้นที่ผ่านเว็บเบราว์เซอร์ได้อย่างสะดวกและมีประสิทธิภาพ โดยมีวัตถุประสงค์หลักดังต่อไปนี้

1. เพื่อออกแบบและพัฒนา **ระบบ Web GIS** ที่สามารถแสดงข้อมูลแผนที่และข้อมูลเชิงบรรยาย (Attribute Data) ได้อย่างถูกต้องและเข้าใจง่าย
2. เพื่อศึกษาและประยุกต์ใช้เทคโนโลยี **Frontend (HTML, CSS, Bootstrap, Leaflet.js)** ในการพัฒนาอินเทอร์เฟซที่ใช้งานง่ายและตอบสนองต่อผู้ใช้
3. เพื่อพัฒนา **Backend** โดยใช้ **PHP** เชื่อมต่อกับฐานข้อมูล **PostgreSQL + PostGIS** สำหรับจัดเก็บและบริหารจัดการข้อมูลเชิงพื้นที่
4. เพื่อใช้งาน **GeoServer** ในการเผยแพร่ข้อมูลแผนที่ผ่านมาตรฐาน OGC เช่น WMS และ WFS ให้สามารถเชื่อมโยงกับระบบ Frontend ได้
5. เพื่อใช้ **QGIS** ในการเตรียม วิเคราะห์ และตรวจสอบข้อมูลเชิงพื้นที่ก่อนนำเข้าสู่ฐานข้อมูล
6. เพื่อส่งเสริมความเข้าใจในการบูรณาการเทคโนโลยีเว็บและระบบภูมิสารสนเทศ สำหรับการพัฒนาแอปพลิเคชันเชิงพื้นที่ในอนาคต

กลุ่มผู้ใช้งาน

1. อาจารย์ผู้ประเมิน สำหรับการตรวจประเมินผลสัมฤทธิ์ของโครงการ
2. นักศึกษาและผู้สนใจ ใช้เป็นแนวทางหรือตัวอย่างในการพัฒนาโปรเจกต์ลักษณะเดียวกัน
3. ผู้ใช้งานทั่วไป เข้ามาเพื่อดูข้อมูลส่วนตัว หรือข้อมูลแผนที่ที่น่าสนใจ

ภาพรวมระบบ

หน้าล็อกอิน (Login) หน้าเข้าสู่ระบบของผู้ใช้งานก่อนเข้าสู่หน้าหลัก มีระบบตรวจสอบชื่อผู้ใช้และรหัสผ่าน

ส่วนที่ 1 กำหนดโครงสร้างเอกสาร HTML

```
<!DOCTYPE html>
<html lang="th">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login - Rae's Space</title>
  <link rel="stylesheet" href="style.css">
</head>
```

<!DOCTYPE html> ระบุว่าเอกสารนี้เป็น HTML5

<html lang="th"> กำหนดภาษาเป็นภาษาไทย

<meta charset="UTF-8"> ตั้งค่าการเข้ารหัสตัวอักษรให้รองรับภาษาไทย

<meta name="viewport" ...> ทำให้หน้าเว็บแสดงผลได้เหมาะสมกับทุกขนาดหน้าจอ (Responsive)

<title> ชื่อของหน้าเว็บ จะแสดงบนแท็บเบราว์เซอร์

<link rel="stylesheet" href="style.css"> เชื่อมต่อไฟล์ CSS ภายนอกเพื่อจัดรูปแบบหน้าเว็บ

ส่วนที่ 2 ส่วนเนื้อหาหลักของหน้าเว็บ (Body)

```
<body>
  <div class="login-container">
    <h2>เข้าสู่ระบบ</h2>
    <form id="loginForm">
      <label for="username">ชื่อผู้ใช้:</label>
      <input type="text" id="username" required>

      <label for="password">รหัสผ่าน:</label>
      <input type="password" id="password" required>

      <button type="submit">เข้าสู่ระบบ</button>
    </form>
    <p id="login-message"></p>
  </div>
</body>
```

<div class="login-container"> กล่องหลักสำหรับจัดตำแหน่งฟอร์มเข้าสู่ระบบ

<form id="loginForm"> ฟอร์มเก็บข้อมูลชื่อผู้ใช้และรหัสผ่าน

<label> แสดงข้อความกำกับช่องกรอกข้อมูล

<input type="text"> และ <input type="password"> ใช้สำหรับรับชื่อผู้ใช้และรหัสผ่าน

required บังคับให้ต้องกรอกก่อนส่งฟอร์ม

<button type="submit"> ปุ่มสำหรับเข้าสู่ระบบ

<p id="login-message"></p> พื้นที่แสดงข้อความแจ้งเตือน เช่น “ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง”

ส่วนที่ 3 สคริปต์ควบคุมการทำงาน (JavaScript)

```
<script>
document.getElementById('loginForm').addEventListener('submit', function(e) {
    e.preventDefault();

    const user = document.getElementById('username').value;
    const pass = document.getElementById('password').value;

    // ตัวอย่างจำลอง (ในโปรเจกต์จริงควรใช้ backend)
    if (user === "rae" && pass === "1234") {
        localStorage.setItem("loggedIn", "true");
        window.location.href = "index.html";
    } else {
        document.getElementById('login-message').textContent = "ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง";
    }
});
</script>
```

addEventListener('submit', ...) ดักเหตุการณ์เมื่อผู้ใช้กดปุ่ม “เข้าสู่ระบบ”

e.preventDefault(); ป้องกันไม่ให้ฟอร์มรีเฟรชหน้าเว็บอัตโนมัติ

ดึงค่าที่กรอกจากช่อง <input> โดยใช้ document.getElementById().value

ตรวจสอบเงื่อนไขว่า

ถ้าชื่อผู้ใช้ = "rae" และรหัสผ่าน = "1234" → บันทึกสถานะการเข้าสู่ระบบใน Local Storage ด้วยคำสั่ง

localStorage.setItem("loggedIn", "true");

→ และเปลี่ยนหน้าไปที่ index.html

ถ้าไม่ตรง → แสดงข้อความแจ้งเตือนใน <p id="login-message">

สรุปการทำงานโดยรวม

หน้าเว็บนี้เป็น ฟรอมต์เอนด์ระบบอย่างง่าย (Frontend Only)

ใช้ HTML สร้างโครงสร้างฟรอมต์

ใช้ CSS (จากไฟล์ style.css) ตกแต่งหน้าตา

ใช้ JavaScript ตรวจสอบข้อมูลที่กรอก

ไม่มีการเชื่อมต่อกับ Backend จริง — เป็นการจำลองระบบเข้าสู่ระบบเบื้องต้น

เหมาะสำหรับการสาธิตหลักการทำงานของระบบ Login ในเว็บแอปพลิเคชัน

ส่วนที่ 1 กำหนดโครงสร้างเอกสาร HTML

```
<!DOCTYPE html>
<html lang="th">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Logout - Rae's Space</title>
  <link rel="stylesheet" href="style.css">
</head>
```

<!DOCTYPE html> ระบุว่าเอกสารนี้ใช้มาตรฐาน HTML5

<html lang="th"> กำหนดภาษาของเอกสารเป็นภาษาไทย

<meta charset="UTF-8"> ตั้งค่าระบบเข้ารหัสตัวอักษรให้รองรับภาษาไทย

<meta name="viewport" ...> ทำให้หน้าเว็บแสดงผลได้ดีทั้งบนคอมพิวเตอร์และมือถือ (Responsive Design)

<title> ตั้งชื่อแท็บของหน้าเว็บ (ในที่นี้คือ “Logout - Rae’s Space”)

<link rel="stylesheet" href="style.css"> เชื่อมต่อกับไฟล์ style.css เพื่อจัดรูปแบบหน้าตาเว็บ

ส่วนที่ 2 ส่วนเนื้อหาหลักของหน้าเว็บ (Body)

```
<body>
  <div class="logout-container">
    <h2>ออกจากระบบแล้ว</h2>
    <p>ขอบคุณที่เข้ามาเยี่ยมชม Rae's Space 🐼</p>
    <a href="login.html" class="btn">เข้าสู่ระบบอีกครั้ง</a>
  </div>
</body>
```

`<div class="logout-container">` เป็นกล่องหลักสำหรับแสดงข้อความหลังจากผู้ใช้กดออกจากระบบ

`<h2>` แสดงหัวข้อใหญ่ “ออกจากระบบแล้ว”

`<p>` แสดงข้อความขอบคุณผู้ใช้งาน

`` สร้างปุ่มลิงก์เพื่อให้ผู้ใช้กลับไปหน้าเข้าสู่ระบบอีกครั้ง (ใช้ CSS class "btn" ในไฟล์ style.css เพื่อจัดรูปแบบให้ดูเหมือนปุ่ม)

ส่วนที่ 3 สคริปต์ควบคุมการทำงาน (JavaScript)

```
<script>
  // เคลียร์สถานะล็อกอิน
  localStorage.removeItem("loggedIn");
</script>
```

ส่วนนี้คือคำสั่ง JavaScript สำหรับจัดการสถานะของผู้ใช้เมื่อออกจากระบบ

`localStorage.removeItem("loggedIn");`

เป็นคำสั่ง ลบข้อมูลที่บันทึกไว้ใน Local Storage ภายใต้อักขระ "loggedIn"

(ซึ่งถูกตั้งค่าไว้ตอนผู้ใช้เข้าสู่ระบบในหน้า login.html)

เมื่อเปิดหน้า logout ระบบจะลบสถานะการล็อกอินออกทันที เพื่อป้องกันไม่ให้อ่านหน้าหลักโดยไม่ได้ล็อกอินอีกต่อไป

สรุปการทำงานโดยรวม

เมื่อผู้ใช้กดปุ่ม “ออกจากระบบ” จากหน้าอื่น ระบบจะพาไปที่หน้า logout.html

โค้ด JavaScript จะทำการลบสถานะการล็อกอินออกจาก Local Storage

แสดงข้อความแจ้งว่า “ออกจากระบบแล้ว” พร้อมปุ่มให้กลับไปหน้า login.html เพื่อเข้าสู่ระบบใหม่อีกครั้ง

หน้าแนะนำตัว (About Me)

เขียนด้วย HTML พร้อมตกแต่งด้วย CSS, ใช้ Google Fonts และ Font Awesome Icons เพื่อเพิ่มความสวยงามและความน่าสนใจ แสดงข้อมูลส่วนตัวของผู้จัดทำ เช่น ชื่อ, รูปภาพ, ความสนใจ

ส่วนที่ 1 กำหนดโครงสร้างพื้นฐานของเอกสาร

```
<!DOCTYPE html>
<html lang="th">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Profile - Sudarat S.</title>
  <link rel="stylesheet" href="style.css">
```

<!DOCTYPE html> กำหนดให้ไฟล์นี้เป็นเอกสาร HTML5

<html lang="th"> ระบุว่าภาษาหลักของเว็บเพจเป็นภาษาไทย

<meta charset="UTF-8"> ตั้งค่าระบบอักขระให้รองรับภาษาไทย

<meta name="viewport"...> ปรับให้หน้าเว็บแสดงผลได้เหมาะสมกับทุกอุปกรณ์

<title> ตั้งชื่อหน้าเว็บที่จะแสดงบนแท็บเบราว์เซอร์

<link rel="stylesheet" href="style.css"> เชื่อมโยงไฟล์ CSS ภายนอกสำหรับจัดรูปแบบ

ส่วนที่ 2 การเชื่อมต่อฟอนต์และไอคอน

```
<!-- Font & Icon -->
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Kanit:wght@300;400;600&displ
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/font-awesome/
```

ใช้ Google Fonts ฟอนต์ Kanit เพื่อให้ข้อความดูสวยงามและอ่านง่าย

ใช้ Font Awesome เพื่อเพิ่มไอคอน เช่น (email), (Facebook), (Instagram)

preconnect ช่วยให้โหลดฟอนต์เร็วขึ้น

ส่วนที่ 3 เนื้อหาหลักของหน้าเว็บ (Body)

```
<body class="dusty-blue-theme">

  <!-- Back to home button -->
  <a href="index.html" class="back-link">🏠 Back to home</a>
  <h1>Introduce myself</h1>
  <p>Let's get to know me.</p>
```

<body class="dusty-blue-theme"> กำหนดคลาสสีของหน้า (อาจใช้ใน CSS เพื่อกำหนดสีพื้นหลังหรือโทนสี)

 ลิงก์ย้อนกลับไปหน้าแรกของเว็บไซต์

<h1> และ <p> เป็นหัวเรื่องและคำบรรยายสั้น ๆ แนะนำเนื้อหาหน้า

ส่วนที่ 4 ส่วนหลักของโปรไฟล์

```
<main class="wrapper">
  <div class="container">
```

<main> ส่วนหลักของเนื้อหาทั้งหมด

<div class="container"> ใช้จัดโครงสร้างหน้าแบบสองฝั่ง (ซ้าย-ขวา)

ส่วนที่ 5 ฝั่งซ้าย (Left Side)

```
<div class="left">
  
  <div class="name">สุรัตน์ สุขอยู่</div>
  <div class="position">นักศึกษาสาขาภูมิศาสตร์ชั้นปีที่ 3</div>
```

แสดงรูปภาพโปรไฟล์ ชื่อ และตำแหน่งของเจ้าของเว็บไซต์

ใช้ เพื่อแสดงภาพ (alt ใช้ระบุข้อความแทนในกรณีที่โหลดภาพไม่ได้)

ส่วนข้อมูลการติดต่อ (Contact)

```
<div class="contact">
  <p><i class="fas fa-envelope"></i>
    <a href="mailto:sudarats66@nu.ac.th">sudarats66@nu.ac.th</a>
  </p>
</div>
```

แสดงอีเมลติดต่อ โดยใช้ไอคอนของจดหมายจาก Font Awesome (fa-envelope)

ลิงก์ mailto: ช่วยให้คลิกแล้วเปิดโปรแกรมส่งอีเมลได้ทันที

ส่วนลิงก์โซเชียลมีเดีย

```
<div class="social-links">
  <a href="https://www.facebook.com/SudaratSukoIU" target="_blank"><i class="fab fa-fb">
  <a href="https://www.instagram.com/eraesas/" target="_blank"><i class="fab fa-ig">
</div>
```

เพิ่มไอคอน Facebook และ Instagram

target="_blank" ทำให้ลิงก์เปิดในแท็บใหม่

ส่วนที่ 6 ฝั่งขวา (Right Side)

```
<div class="right">
  <div class="section">
    <h3>แนะนำตัว</h3>
    <p><strong>รหัสติดต่อ</strong> - 66163297</p>
    ...
  </div>
```

ใช้แบ่งข้อมูลออกเป็นหัวข้อย่อยหลายส่วน (แนะนำตัว, ทักษะ, การศึกษา, งานอดิเรก)

<h3> ใช้เป็นหัวข้อของแต่ละหมวด

<p> ใช้แสดงรายละเอียดแต่ละบรรทัด

ส่วนที่ 7 เนื้อหาเพิ่มเติม

```
<div class="section skills">
  <h3>ทักษะ</h3>
  <p>สามารถใช้โปรแกรม Arcgis, Qgis, Visual Studio Code, PostgreSQL</p>
  ...
</div>
```

ระบุ ทักษะด้านโปรแกรมและเครื่องมือ GIS

และหมวดอื่น ๆ เช่น การศึกษา และ งานอดิเรก

สรุปภาพรวมการทำงาน

ใช้ HTML จัดโครงสร้างข้อมูลและ CSS (style.css) ในการตกแต่ง เช่น สีพื้นหลัง การจัดวางสองคอลัมน์ ฟอนต์ และปุ่ม

ใช้ Google Fonts (Kanit) เพื่อให้ข้อความอ่านง่ายและดูทันสมัย และFont Awesome สำหรับไอคอนตกแต่ง และสัญลักษณ์โซเชียล

โครงสร้างแบบนี้เหมาะกับการนำเสนอ “ข้อมูลส่วนตัวของผู้พัฒนา” หรือ “หน้าข้อมูลสมาชิกทีม” ภายในเว็บไซต์ Web GIS หรือโครงการอื่น ๆ

หน้ากิจกรรมในมหาวิทยาลัย

แสดงรายการกิจกรรมและภาพถ่ายของกิจกรรมต่าง ๆ

อธิบายทีละส่วน (detail)

<head>

ตั้งค่า charset และ viewport เพื่อรองรับภาษาไทยและการแสดงผลแบบ responsive

ตั้ง <title> และเชื่อม style.css เพื่อจัดรูปแบบแสดงผล

ส่วนบนของ <body>

 — ลิงก์กลับหน้าแรก (ควรมี aria-label ถ้าต้องการเพิ่ม accessibility)

<h1> / <p> — หัวเรื่องหลักและคำอธิบายย่อของเพจ

<div class="activities-grid">

เป็นคอนเทนเนอร์จัดเลย์เอาต์เป็นกริด (โดยปกติจะใช้ CSS Grid หรือ Flexbox ใน style.css)

แต่ละ .activity-card ประกอบด้วย

 — รูปกิจกรรม

<h3> — ชื่อกิจกรรม

<p> — คำอธิบายกิจกรรม

หน้าตารางเรียนของฉัน

แสดงตารางเรียนของผู้ใช้ในแต่ละวัน ออกแบบให้ตอบสนองต่ออุปกรณ์ (Responsive)

ส่วนที่ 1 กำหนดโครงสร้าง HTML

```
<!DOCTYPE html>
<html lang="th">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Profile - Sudarat S.</title>
  <link rel="stylesheet" href="style.css">
```

กำหนดเอกสารเป็น HTML5 (<!DOCTYPE html>)

ตั้งภาษาเป็นไทย (lang="th")

ใช้ meta viewport เพื่อให้หน้าเว็บ responsive

เชื่อม CSS ภายนอก (style_schedule.css) เพื่อจัดรูปแบบ

ส่วนที่ 2 สไตล์เพิ่มเติม

```
<style>
  .calendar-checker { ... }
  .calendar-checker input[type="date"] { ... }
  .calendar-checker button { ... }
  #daily-schedule-result { ... }
</style>
```

เพิ่มสไตล์เฉพาะสำหรับ ปฏิทินตรวจสอบตารางเรียนรายวัน

.calendar-checker — กล่องสำหรับเลือกวันที่

<input type="date"> — เลือกวันที่

<button> — ปุ่มตรวจสอบตารางเรียน

#daily-schedule-result — พื้นที่แสดงผลตารางเรียนรายวัน

ส่วนที่ 3 เนื้อหาใน <body>

```
<div class="schedule-container">
  <a href="index.html" class="back-link">🏠 Back to Home</a>
  <h1>My Class Schedule</h1>
  <p>ปีการศึกษา 2568 / เทอม 1</p>
```

ลิงก์ย้อนกลับหน้าแรก

หัวเรื่อง <h1> และปีการศึกษา

ส่วนปฏิทินตรวจสอบตารางเรียนรายวัน

```
<div class="calendar-checker">
  <h2>📅 ตรวจสอบตารางเรียนรายวัน</h2>
  <input type="date" id="date-input">
  <button onclick="checkSchedule()">ดูตารางเรียน</button>
  <div id="daily-schedule-result">
    <p>กรุณาเลือกวันที่เพื่อดูตารางเรียน</p>
  </div>
</div>
```

ผู้ใช้เลือกวันที่แล้วกดปุ่ม ดูตารางเรียน

พื้นที่ #daily-schedule-result จะแสดงตารางเรียนของวันที่เลือก

ใช้งานง่ายและโต้ตอบแบบ client-side โดยไม่ต้อง reload หน้า

ส่วนตารางเรียนเต็ม (Full Schedule)

```
<div class="schedule-grid" id="full-schedule">
  <div class="day-card" data-day="Monday"> ... </div>
  <div class="day-card" data-day="Tuesday"> ... </div>
  ...
</div>
```

แต่ละวันเก็บใน .day-card

data-day="Monday" ใช้สำหรับค้นหาตารางเรียนตามวัน

แต่ละ .class-item ภายในวันเก็บข้อมูล วิชา, รหัสวิชา, เวลา, ผู้สอน, ห้องเรียน

ส่วน JavaScript สำหรับตรวจสอบวันเรียน

```
document.getElementById('full-schedule').style.display = 'none';

function checkSchedule() {
  const dateInput = document.getElementById('date-input');
  const resultDiv = document.getElementById('daily-schedule-result');

  if (!dateInput.value) {
    resultDiv.innerHTML = '<p>⚠ กรุณาเลือกวันที่ยกก่อนครับ</p>';
    return;
  }

  const selectedDate = new Date(dateInput.value);
  const dayOfWeek = selectedDate.toLocaleString('en-US', { weekday: 'long' });

  const dayCard = document.querySelector(`.day-card[data-day="${dayOfWeek}"]`);

  if (dayCard) {
    resultDiv.innerHTML = dayCard.innerHTML;
  } else {
    resultDiv.innerHTML = '<h3>🚫 วันนี้ไม่มีเรียน!</h3>';
  }
}
```

ซ่อนตารางเรียนเต็ม (full-schedule) ตอนโหลดหน้า

ฟังก์ชัน checkSchedule() ทำงานเมื่อผู้ใช้กดปุ่ม

ตรวจสอบว่าผู้ใช้เลือกวันที่หรือไม่

แปลงวันที่ที่เลือกเป็น Date object และหาชื่อวันภาษาอังกฤษ (Monday, Tuesday...)

ค้นหา .day-card ที่ตรงกับวันนั้น (data-day="Monday")

ถ้ามีวันเรียน → นำเนื้อหาของวันนั้นมาแสดงใน #daily-schedule-result

ถ้าไม่มีวันเรียน → แสดงข้อความ “วันนี้ไม่มีเรียน”

สรุปภาพรวมการทำงาน

หน้าเว็บนี้ แสดงตารางเรียนเต็มและตรวจสอบตารางรายวัน

ใช้ HTML จัดโครงสร้างข้อมูล

ใช้ CSS ตกแต่ง (ไฟล์ style_schedule.css + inline style สำหรับปฏิทิน)

ใช้ JavaScript สำหรับเลือกวันและแสดงตารางเรียนรายวันแบบไดนามิก

ไม่มีการเชื่อมต่อ Backend — เป็นระบบ client-side ล้วน

หน้าข้อมูลนิสิตปี 64-67

แสดงรายชื่อและข้อมูลนิสิตแต่ละปีการศึกษาในรูปแบบตาราง

1. <head>

ตั้ง meta และ title ของเว็บ

โหลด Chart.js, Leaflet, Font Awesome, Google Fonts

กำหนด CSS Variables สำหรับธีมสี

สไตล์ทั่วไป เช่น layout, header, card, table, map, responsive

2. <body>

a) Back link

ปุ่ม “กลับไปหน้าหลัก”

b) Header (.glass-header)

โลโก้ + ชื่อระบบ + ผู้ใช้งาน + ปุ่มอัปเดตข้อมูล

c) Navigation Pills (.modern-nav)

ปุ่มเลือกปีการศึกษา (AGI64–AGI67)

แสดง badge จำนวนนิสิตในแต่ละปี

d) Stats Grid (.stats-grid)

การ์ดสรุปข้อมูล: จำนวนนิสิต, หลักสูตร, ภาควิชา, จังหวัด

e) Dashboard Layout (.dashboard-layout)

Left: Chart + Table

Chart: แสดงสถิติแยกตามหลักสูตร/จังหวัด/ภาควิชา/โรงเรียน

Table: แสดงรายละเอียดนิสิต พร้อม search, sort, pagination

Right: Map

แสดงตำแหน่งนิสิตบนแผนที่

เลือกมุมมอง standard/satellite

f) Floating Action Button

ปุ่มเลื่อนกลับด้านบน

3. <script>

ตัวแปร global เก็บข้อมูลนิสิต, chart, map, markers, pagination

ฟังก์ชันหลัก

selectTable() → เปลี่ยนปีการศึกษาและโหลดข้อมูลใหม่

loadStudents() → ดึงข้อมูลนิสิตจาก API

updateDisplay() → อัปเดต table, map, pagination

updateTable() → แสดงข้อมูลนิสิตในตาราง

updateMap() → แสดง markers บนแผนที่

updateMapView() → เปลี่ยนมุมมอง map

updateSummary() → อัปเดตสรุปข้อมูลสถิติ

updateChart() → สร้าง/อัปเดต chart

searchTable() → ค้นหา นิสิตในตาราง

sortTable() → เรียงข้อมูลในตาราง

updatePagination(), previousPage(), nextPage() → จัดการหน้าตาราง

initMap() → เริ่มต้น Leaflet map

showNotification() → แสดง toast message

scrollToTop() → เลื่อนกลับด้านบน

สรุป

HTML → โครงสร้างหน้า, CSS → สไตล์สวยงาม, responsive

JS → ดึงข้อมูล, แสดง table, chart, map, filter, pagination, toast

หน้าแผนที่ที่ปักจังหวัดลำปาง แสดงแผนที่ Google Map ของหอพัก เพื่อค้นหาที่พักได้ง่าย

1. <head>

ตั้ง title, meta charset

โหลด Leaflet, Leaflet MarkerCluster, Font Awesome

CSS จัด layout, sidebar, map, legend, responsive

2. <body>

a) หัวเรื่อง <h2>

แสดงชื่อหน้า “ที่พักในจังหวัดลำปาง”

b) Container (.container)

ย้าย <div id="map"> → แผนที่ Leaflet

ขวา .sidebar → ปรับขยาย/ย่อ, มี

ปุ่ม toggle sidebar

เลือก Base Map (OSM, Google, Esri)

เปิด/ปิด Overlay Layers (ที่พัก, ขอบเขตอำเภอ, ถนน, จุดท่องเที่ยว)

Legend แสดงประเภทที่พัก

3. <script>

a) Map setup

สร้าง Leaflet map พร้อม base layers (OSM, Google, Esri)

b) WMS Layers

โหลดข้อมูลจาก GeoServer:

ขอบเขตอำเภอ (lampang_amphoe)

ถนน (lampang_road, lampang_roads)

จุดท่องเที่ยว (lampang_point)

c) Overlay Marker Cluster

โหลด JSON ที่ปัก (json_sql.php)

วนสร้าง marker แต่ละที่ปัก

ตั้งสีตาม ประเภทที่ปัก

แสดง popup info

ใส่เข้า accommodationsLayer (Clustered markers)

d) Legend

แสดงสีของแต่ละประเภทที่ปัก

คลิก Legend → กรองแสดงเฉพาะประเภทที่เลือก

e) Overlay toggles

checkbox เปิด/ปิด แต่ละ overlay layer บน map

f) Base map toggle

radio switch เปลี่ยน base map

g) Sidebar toggle

ปุ่ม ≡ ย่อ/ขยาย sidebar

สรุป

แสดง แผนที่จังหวัดลำปาง มี layer หลัก (base map) + overlay (ที่ปัก, ถนน, เขต, จุดท่องเที่ยว)

Marker cluster สำหรับที่ปัก พร้อม popup

Legend แยกประเภทที่ปัก และสามารถ filter

Sidebar สำหรับควบคุม layers และปรับขยาย

Thailand Web GIS Dashboard

1. <head>

โหลด Font Poppins, Leaflet, jQuery, Leaflet AJAX, Font Awesome

CSS

ปรับ body และ header ให้สวยงาม

จัดขนาดแผนที่เต็มหน้าจอ

ปรับ popup, form, layer control, legend ให้ดูดี

2. <body>

header แสดงชื่อ “Thailand Web GIS Dashboard” พร้อมไอคอน

div#map พื้นที่สำหรับแผนที่

3. <script>

a) Map setup

สร้างแผนที่ Leaflet (map)

ตั้ง center และ zoom

marker_arr เก็บ marker ที่ผู้ใช้คลิก

b) Base Maps

โหลด tile layers หลายแบบ: Wikimedia, Google, OSM, Esri, Carto, OpenTopoMap

ตั้ง Wikimedia เป็นค่าเริ่มต้น

c) WMS Layers (จาก GeoServer)

plk_amphoe → ขอบเขตอำเภอ

utt_road → ถนน

utt_gcp → จุดสำคัญ (GCP)

d) Custom Markers

buildingIcon → ภาพ marker อาคาร

nakhonSawanIcon → DivIcon ไอคอนฝนสีน้ำเงิน

ตัวอย่าง Marker, Circle, Polyline, Polygon

popUp → ฟังก์ชันแสดง property ของ GeoJSON

e) GeoJSON Layer

โหลด thailand_province.geojson และแสดงเป็น LayerGroup

f) Search by Distance (Click Event)

คลิกบนแผนที่ → สร้าง marker พร้อม popup form

form ให้กรอก Lat, Lng, ระยะทาง แล้วกด Search

sendtodb() → แสดง alert ข้อมูลที่กรอก (สามารถส่งต่อ DB ได้)

g) API Data (Thaiwater)

โหลดข้อมูลฝน 24 ชม. จาก Thaiwater API

นครสวรรค์ → ใช้ icon สีฟ้า

ทั่วประเทศ → icon เปลี่ยนสีตามปริมาณฝน (getColor())

สร้าง marker พร้อม popup แสดงชื่อสถานี, จังหวัด, ปริมาณฝน

h) Layer Control

แยก baselayers กับ overlays

overlays จุด, เส้น, พื้นที่, GeoJSON, WMS, ฝน API

สามารถเปิด/ปิด layer ได้จาก control

i) Legend

แสดงสีตามปริมาณฝน 24 ชม.

ตำแหน่ง bottomright

ไล่สีตามระดับฝน (0, 10, 35, 90 mm) และ "ไม่มีฝน"

สรุป

แสดง Web GIS Dashboard แบบ interactive

มี base map หลายแบบและ WMS layers จาก GeoServer

สามารถ คลิกเพื่อวัดระยะทาง / ค้นหาตามจุด

โหลด ข้อมูลฝน API และแสดง สี marker ตามปริมาณฝน

มี Layer control + Legend ให้ผู้ใช้เปิด/ปิด layer และดูข้อมูลระดับฝน

1. <head>

โหลด Bootstrap และไฟล์ CSS ภายนอก (style.css)

ตั้ง meta viewport เพื่อรองรับมือถือ

CSS ภายในไฟล์

จัดสไตล์ body และพื้นหลัง

.back-link-wrapper / .back-link → ปุ่ม “Back to home” ขวบน

.dashboard-header → หัวข้อใหญ่ของ dashboard

.nav-buttons → ปุ่มเมนูสำหรับโหลดหน้า iframe

.iframe-search-wrapper → กรอบสี่ขาวสำหรับ iframe

ปรับ iframe ให้เต็มความกว้างและสูง 600px

2. <body>

2.1. Back to home

```
<div class="back-link-wrapper">
  <a href="index.html" class="back-link">🏠 Back to home</a>
</div>
```

ปุ่มย้อนกลับไปหน้า index

2.2. Dashboard header

```
<div class="dashboard-header">
  <h1>📖 ข้อมูลนักศึกษา 📊</h1>
</div>
```

แสดงชื่อ Dashboard แบบเด่น ๆ

2.3. Navigation Buttons

```
<div class="nav-buttons">
  <button onclick="loadPage('map-frame.html')" class="icon-link">📊 แผนที่</button>
  ...
</div>
```

ปุ่มแต่ละปุ่มโหลด iframe ต่าง ๆ

แผนที่, Pie chart, Bar chart, Line graph, ตารางข้อมูล, รายละเอียด

2.4. Iframe wrapper

```
<div class="iframe-search-wrapper">
  <iframe id="contentFrame" src="map-frame.html"></iframe>
</div>
```

พื้นที่สำหรับ โหลดหน้าเนื้อหา ที่เลือกจากปุ่ม nav

โหลดหน้าเริ่มต้นเป็น map-frame.html

3. <script>

3.1. ฟังก์ชัน loadPage(page)

```
function loadPage(page) {
  document.getElementById('contentFrame').src = page;
  document.getElementById('searchInput').value = '';
}
```

เปลี่ยน iframe src ตามปุ่มที่กด

ล้างค่า search input

3.2. ส่งคำค้นหาไปยัง iframe (mapdata.html)

```
document.getElementById('searchButton').addEventListener('click', function () {
  const term = document.getElementById('searchInput').value;
  const contentFrame = document.getElementById('contentFrame');
  if (contentFrame.src.includes('mapdata.html')) {
    contentFrame.contentWindow.postMessage({ type: 'search', term }, '*');
  } else {
    alert('ฟังก์ชันค้นหาใช้ได้เฉพาะหน้า "ข้อมูล" เท่านั้น');
  }
});
```

ใช้ postMessage ส่งคำค้นหาไปยัง iframe เฉพาะหน้า mapdata.html

3.3. Reset search

```
document.getElementById('resetButton').addEventListener('click', function () {
  document.getElementById('searchInput').value = '';
  const contentFrame = document.getElementById('contentFrame');
  if (contentFrame.src.includes('mapdata.html')) {
    contentFrame.contentWindow.postMessage({ type: 'reset' }, '*');
  }
});
```

ล้างค่า input และส่งคำสั่ง reset ไป iframe

3.4. โหลดหน้าเริ่มต้นเมื่อเปิดเว็บ

```
document.addEventListener('DOMContentLoaded', () => {  
  loadPage('map-frame.html'); // หน้าแรกคือแผนที่  
});
```

สรุป

1. หน้า Dashboard หลักมี:
 - ปุ่ม “กลับบ้าน home”
 - Header ของ dashboard
 - ปุ่มเมนูเปลี่ยน iframe content
 - iframe สำหรับแสดงหน้าต่าง ๆ เช่น แผนที่, chart, ตาราง, รายละเอียด
2. รองรับ search และ reset เฉพาะหน้า mapdata.html
3. โหลดหน้าเริ่มต้นเป็น แผนที่

เทคโนโลยีที่ใช้ในการพัฒนา Frontend (HTML5, CSS3, JavaScript)

1. HTML5 ใช้สำหรับสร้างโครงสร้างหน้าเว็บ เช่น header, ปุ่ม, ตาราง, และ iframe
 - CSS3 ใช้สำหรับตกแต่งหน้าเว็บ เช่น สีพื้นหลัง, ขอบ, ขนาดตัวอักษร, การจัดวาง layout ให้สวยงามและตอบสนองต่อหน้าจอหลายขนาด
 - JavaScript ใช้สำหรับเพิ่มความโต้ตอบ เช่น การเปลี่ยนหน้า iframe ตามปุ่มที่กด, การค้นหาข้อมูล, การสื่อสารกับ iframe, การโหลดแผนที่และแสดงข้อมูลแบบ dynamic
2. Design Tools (Canva, Figma)
 - ใช้สำหรับออกแบบ UI/UX ของ Dashboard และส่วนต่าง ๆ ของหน้าเว็บ
 - ช่วยให้การจัดวางปุ่ม, สี, ไอคอน และองค์ประกอบของหน้าเว็บมีความสวยงามและใช้งานง่าย
3. Localhost สำหรับทดสอบ
 - ใช้เพื่อจำลองการทำงานของเว็บบนเครื่องคอมพิวเตอร์ก่อนนำไปใช้งานจริง
 - ช่วยให้สามารถเชื่อมต่อกับไฟล์ GeoJSON, WMS, หรือฐานข้อมูลที่อยู่บนเครื่องได้อย่างปลอดภัย
4. Google Maps Embed API / Leaflet.js
 - ใช้ในการแสดง แผนที่แบบ interactive บนเว็บ
 - สามารถวาง marker, วงกลม, เส้น, หรือ polygon บนแผนที่ได้
 - สามารถเพิ่มข้อมูล dynamic เช่น ที่ปัก, สถานีฝน หรือเส้นทาง
 - ทำให้ผู้ใช้สามารถ ชุม, เลื่อนแผนที่ และดูข้อมูลเพิ่มเติมจาก popup

ขั้นตอนการพัฒนาเว็บไซต์ (รายละเอียด)

1. ศึกษาความต้องการของระบบ

วัตถุประสงค์ ทำความเข้าใจว่าเว็บไซต์ต้องการอะไร และใครคือผู้ใช้งานเป้าหมาย

รายละเอียดการทำงาน

รวบรวมข้อมูลที่ต้องแสดงบนเว็บไซต์ เช่น ข้อมูลนักศึกษา, แผนที่, ตารางสถิติ, กราฟ

ระบุฟังก์ชันหลักที่ต้องมี เช่น การค้นหาข้อมูล, การแสดงผลแบบ Interactive, การสลับหน้า dashboard

สรุปความต้องการในรูปแบบข้อกำหนดของระบบ (Requirement Specification)

เครื่องมือ/เทคโนโลยี เอกสาร, ตาราง, การสัมภาษณ์ผู้ใช้งาน

2. ออกแบบโครงสร้างและหน้าตาเว็บไซต์

วัตถุประสงค์ วางโครงสร้างและออกแบบ UI/UX เพื่อให้ผู้ใช้งานใช้งานง่ายและสวยงาม

รายละเอียดการทำงาน

ออกแบบ Layout ของหน้าเว็บ เช่น Header, Navigation bar, Content, Footer

วางโครงสร้างการนำทาง (Navigation Flow) เช่น หน้าแผนที่, กราฟวงกลม, ตารางข้อมูล

ออกแบบ Mockup หรือ Prototype ของแต่ละหน้า

เครื่องมือ/เทคโนโลยี

Canva / Figma สำหรับออกแบบ UI/UX

การใช้สี, ไอคอน, และ typography ให้เหมาะสม

3. พัฒนาโค้ดในแต่ละหน้า

วัตถุประสงค์ สร้างเว็บไซต์ให้ทำงานตามที่ออกแบบ

รายละเอียดการทำงาน

เขียน Frontend ด้วย HTML5, CSS3, JavaScript

สร้างหน้า Dashboard ที่มีปุ่ม Navigation และ iframe สำหรับสลับหน้า

แสดงข้อมูลบนแผนที่ Interactive โดยใช้ Google Maps Embed API หรือ Leaflet.js

แสดงกราฟและตารางข้อมูล เช่น แผนภูมิวงกลม, แผนภูมิแท่ง, กราฟเส้น

ทำระบบค้นหาข้อมูล และการส่งค่าคืนไปยัง iframe

เครื่องมือ/เทคโนโลยี

HTML5 / CSS3 / JavaScript

Google Maps Embed API, Leaflet.js

Bootstrap สำหรับการจัด Layout และ Responsive

4. ทดสอบการใช้งานและความถูกต้องของข้อมูล

วัตถุประสงค์ ตรวจสอบว่าเว็บไซต์ทำงานได้ถูกต้องและข้อมูลถูกต้องครบถ้วน

รายละเอียดการทำงาน

ทดสอบการใช้งานฟังก์ชันทั้งหมด เช่น การสลับหน้า Dashboard, การค้นหาข้อมูล, การแสดง Marker และ Popup บนแผนที่, การโหลดกราฟและตารางข้อมูล

ตรวจสอบความถูกต้องของข้อมูลที่แสดงบนแผนที่และกราฟ

ทดสอบ Responsive เพื่อให้เว็บไซต์ใช้งานได้ดีบนอุปกรณ์ทุกขนาด

เครื่องมือ/เทคโนโลยี

Localhost สำหรับทดสอบ

Browser Developer Tools สำหรับตรวจสอบโค้ดและ Debug

5. ปรับปรุงแก้ไขและเผยแพร่เว็บไซต์

วัตถุประสงค์ ทำให้เว็บไซต์สมบูรณ์และพร้อมใช้งานจริง

รายละเอียดการทำงาน

แก้ไขข้อบกพร่องและปรับปรุง UI/UX

ปรับแต่งให้โหลดเร็วขึ้นและรองรับอุปกรณ์ต่าง ๆ (Responsive)

ตรวจสอบความเข้ากันได้กับ Browser ต่าง ๆ

เผยแพร่เว็บไซต์บน Server หรือ Hosting เพื่อให้ผู้ใช้งานสามารถเข้าถึงได้

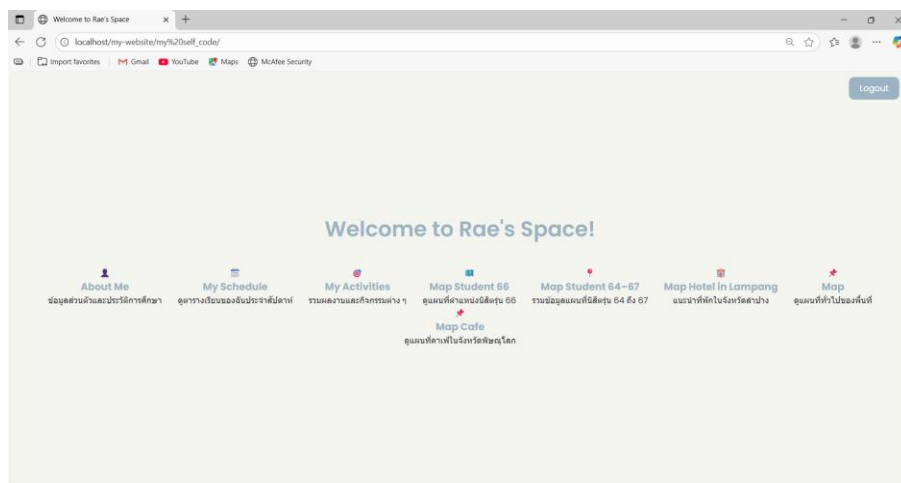
เครื่องมือ/เทคโนโลยี

Web Hosting หรือ Server เช่น XAMPP, Localhost, Netlify, GitHub Pages

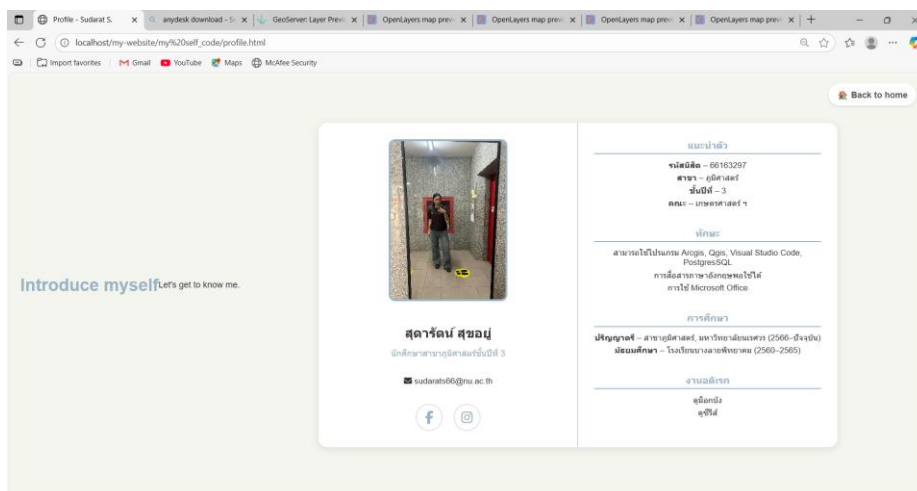
ผลลัพธ์การพัฒนา

เว็บไซต์สามารถใช้งานได้จริงตามวัตถุประสงค์ โดยมีหน้าแสดงข้อมูลครบถ้วน ได้แก่

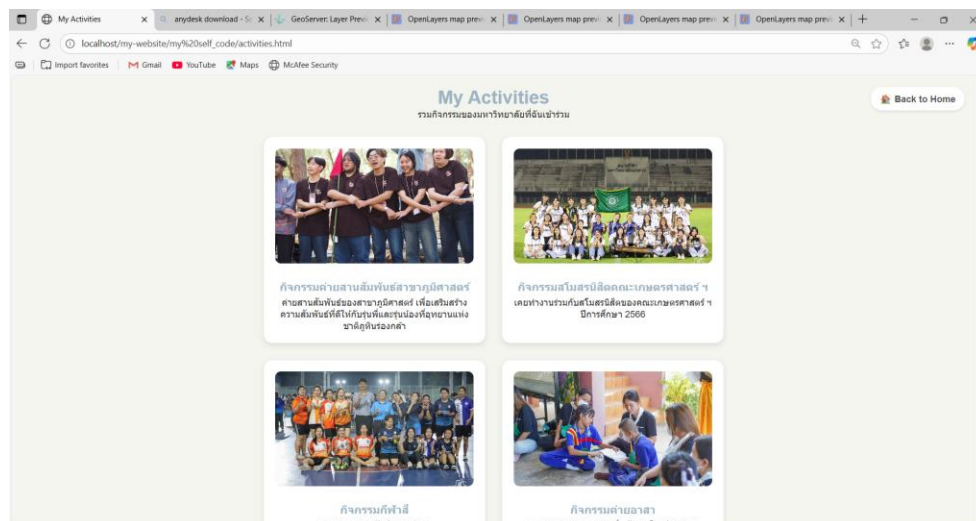
หน้าหลัก



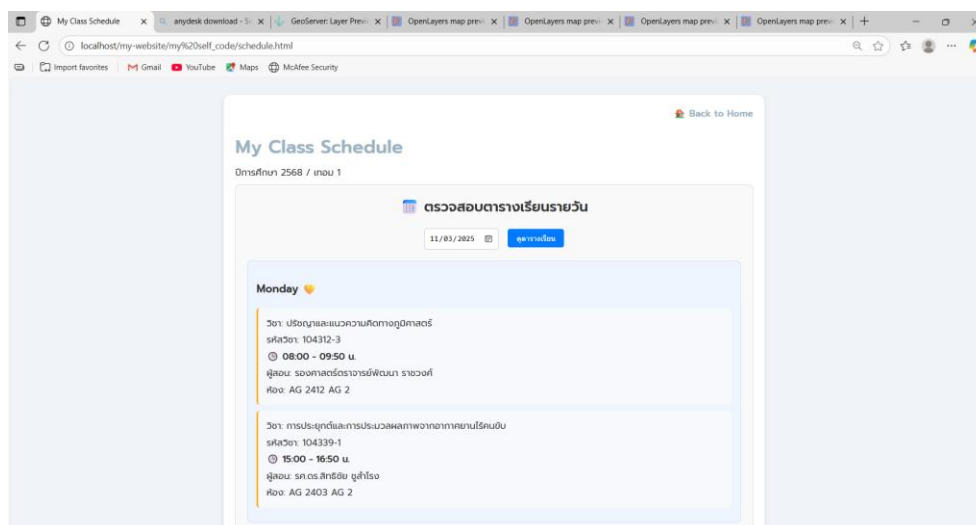
หน้าแนะนำตัว



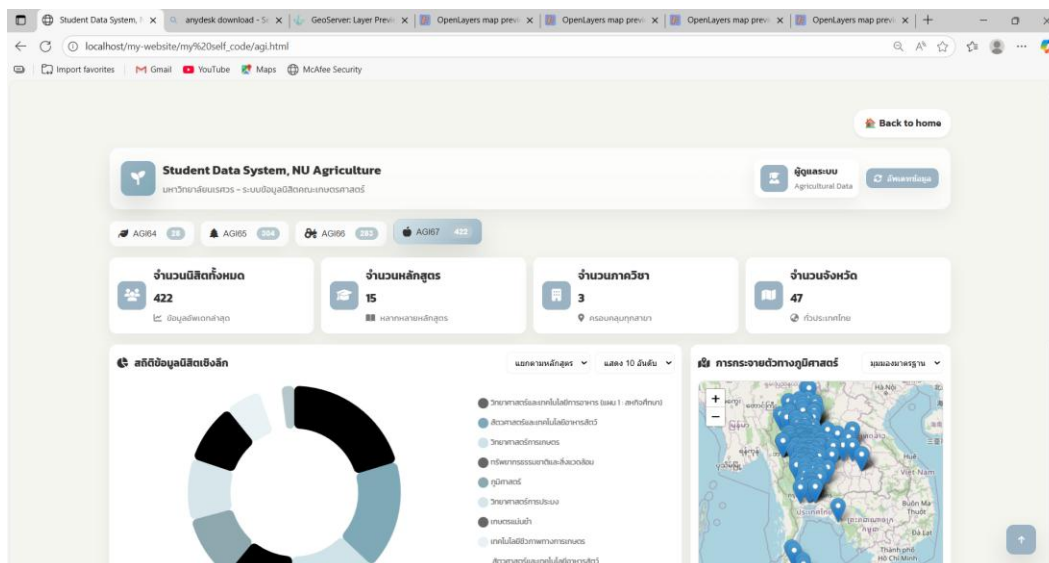
หน้ากิจกรรม



หน้าตารางเรียน



หน้าินิต แผนที่ ที่ปัก



ระบบล็อกอินก่อนเข้าสู่ระบบหลัก

เข้าสู่ระบบ

ชื่อผู้ใช้:

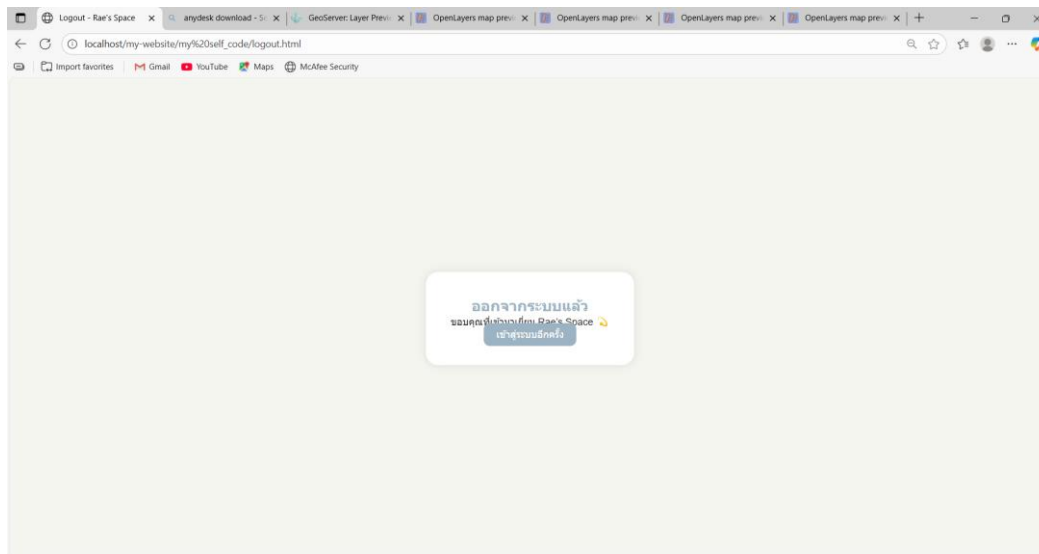
rae

รหัสผ่าน:

1234

เข้าสู่ระบบ

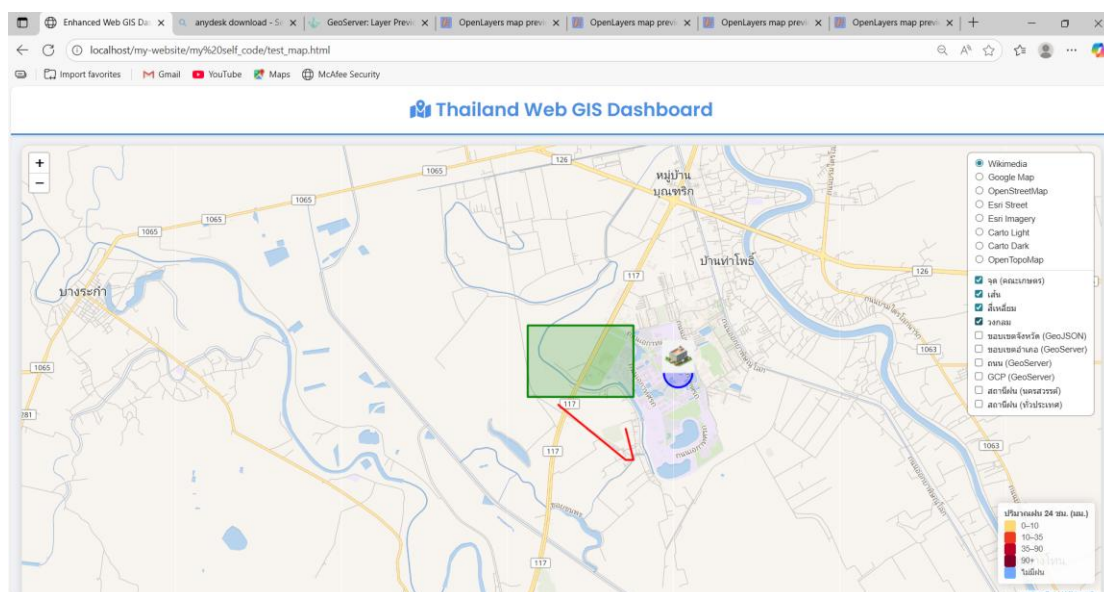
ระบบล็อกเข้าตัวออกจากระบบหลัก



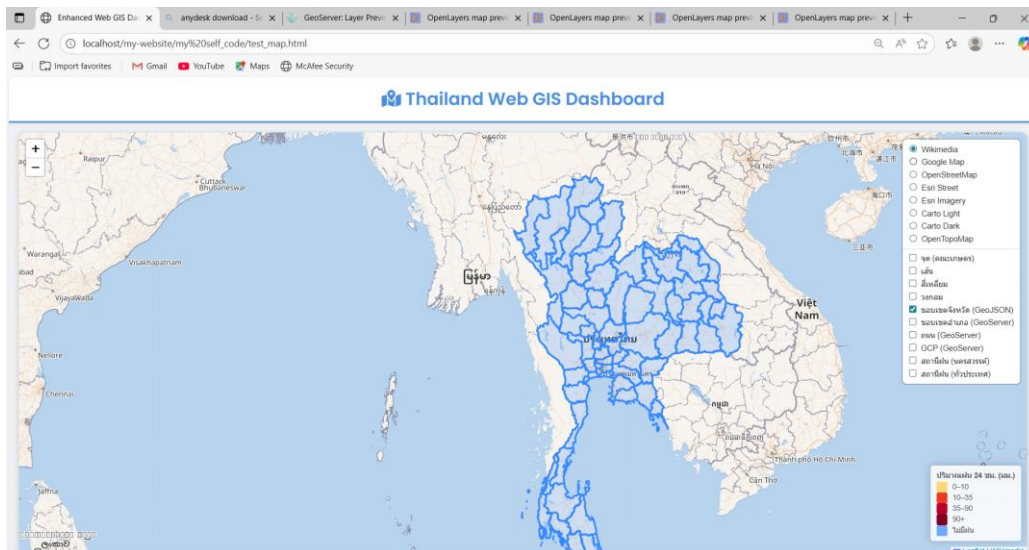
หน้า Thailand Web GIS Dashboard

แผนที่ GIS ของประเทศไทย พร้อมข้อมูลฝน, WMS Layer, Marker, Popup และ Legend

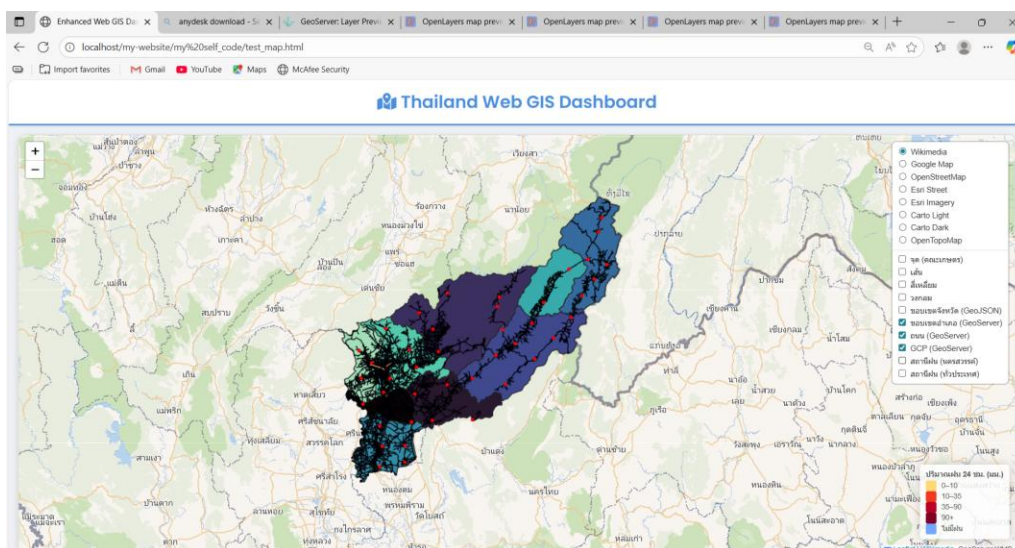
แสดงจุด เส้น สีเหลี่ยม และวงกลม



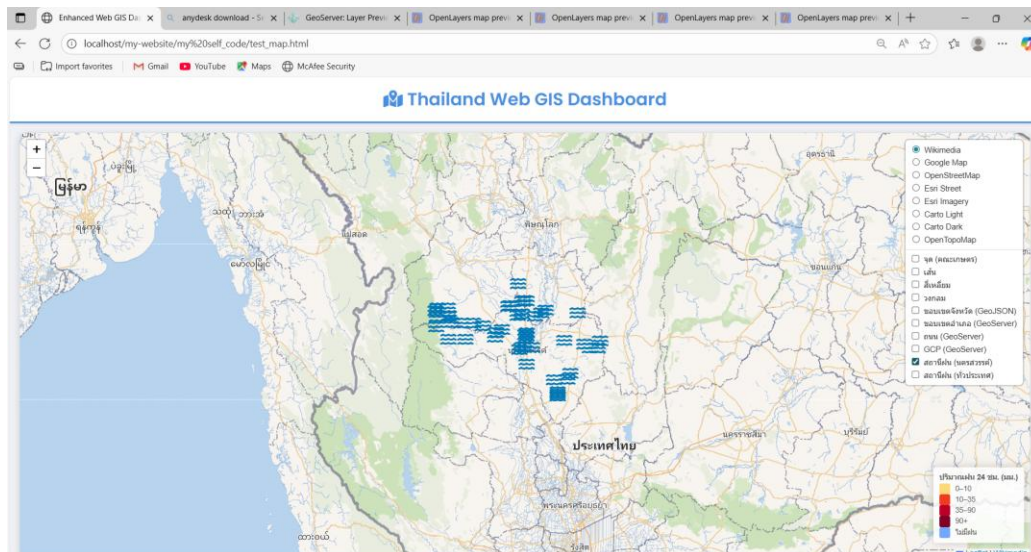
แสดงขอบเขตจังหวัด GeoJson



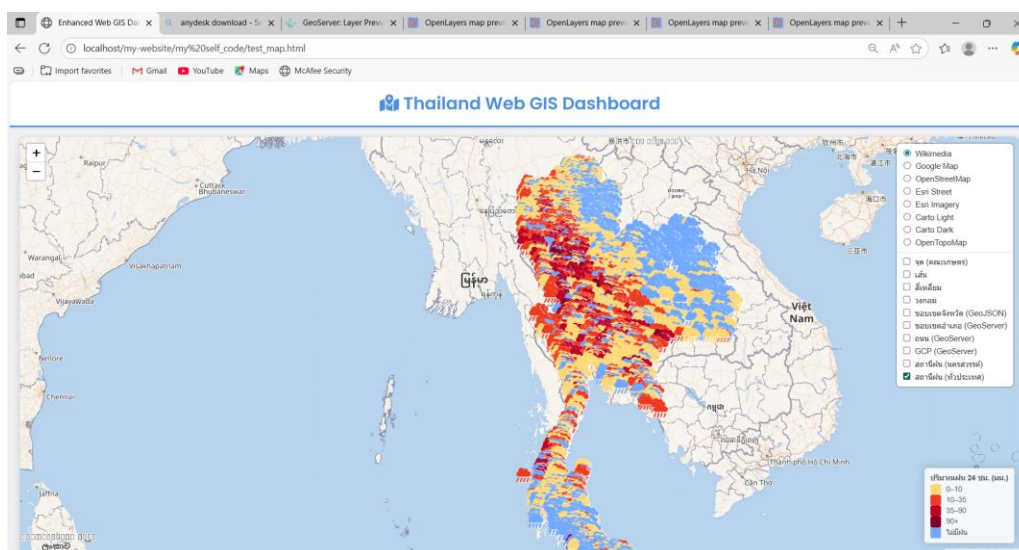
แสดงขอบเขตอำเภอ ถนน และจุด GCP ของจังหวัดอุดรธานี



แสดงสถานีฝนในจังหวัดนครสวรรค์



แสดงสถานีฝนทั่วประเทศไทย พร้อมมี Legend บอกปริมาณน้ำฝน 24 ชม.



บรรณานุกรม

W3Schools. (n.d.). W3Schools Online Web Tutorials. สืบค้น 2 พฤศจิกายน 2025, จาก <https://www.w3schools.com/>

Canva. (n.d.). Canva – Visual Suite for Everyone. สืบค้น 2 พฤศจิกายน 2025, จาก <https://www.canva.com/>

Google Developers. (n.d.). Google Maps Platform. สืบค้น 2 พฤศจิกายน 2025, จาก <https://developers.google.com/maps>