SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร

SMART NU: Development of a Geospatial Information System for Transportation and Weather Awareness within Naresuan University

กัณฐมณี วรอินทร์

วิทยานิพนธ์ระดับปริญญาตรี เสนอภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม

คณะเกษตรศาสตร์ ทรัพยากรธรรมชาติและสิ่งแวดล้อม มหาวิทยาลัยนเรศวร

เพื่อเป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิทยาศาสตรบัณฑิต

สาขาวิชาภูมิศาสตร์

ตุลาคม 2568

ลิขสิทธิ์เป็นของมหาวิทยาลัยนเรศวร

อาจารย์ที่ปรึกษา ประธานบริหารหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาภูมิศาสตร์และหัวหน้าภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อมได้พิจารณา วิทยานิพนธ์ระดับปริญญาตรีเรื่อง "SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร" (SMART NU: Development of a Geospatial Information System for Transportation and Weather Awareness within Naresuan University) ของ กัณฐมณี วรอินทร์ เห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาภูมิศาสตร์มหาวิทยาลัยนเรศวร

.....

(รองศาสตราจารย์ ดร.สิทธิชัย ชูสำโรง)

อาจารย์ที่ปรึกษาวิทยานิพนธ์

.....

(อาจารย์ ธัญญาลัคน์ จันทร์สมบัติ)

ประธานบริหารหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาภูมิศาสตร์

.....

(ผู้ช่วยศาสตราจารย์ ว่าที่ ร.ต.ดร. รังสรรค์ เกตุอ๊อด)

หัวหน้าภาควิชาทรัพยากรธรรมชาติและสิ่งแวดล้อม

กิตติกรรมประกาศ

วิทยานิพนธ์ระดับปริญญาตรี เรื่อง SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร (SMART NU: Development of a Geospatial Information System for Transportation and Weather Awareness within Naresuan University) ฉบับนี้สำเร็จลุล่วงได้ด้วยดี เนื่องจากได้รับความช่วยเหลือและการสนับสนุนจากหลายฝ่าย

ขอขอบพระคุณ รองศาสตราจารย์ ดร.สิทธิชัย ชูสำโรง อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ได้เสียสละเวลาอันมีค่าในการให้คำปรึกษา แนะนำ และช่วยแก้ไขปัญหาพร้อมกับชี้แนวทางการดำเนินงานตลอดระยะเวลาในการทำวิทยานิพนธ์ฉบับนี้ ทั้งนี้ยังคอยตรวจสอบแก้ไขข้อบกพร่อง และติดตามผลการศึกษาอยู่อย่างสม่ำเสมอ ตลอดจนช่วยแก้ไขปัญหาต่าง ๆ ที่เกิดขึ้นระหว่างการดำเนินงานอันเป็นประโยชน์อย่างยิ่ง จนทำให้วิทยานิพนธ์ฉบับนี้เสร็จลุล่วงได้อย่างสมบูรณ์ และขอขอบพระคุณคณาจารย์สาขาภูมิศาสตร์ทุกท่านที่ได้ถ่ายทอดความรู้ต่าง ๆ กับผู้วิจัยให้สามารถนำความรู้ที่เรียนมาใช้ให้เกิดประโยชน์สูงสุด และให้คำแนะนำเพิ่มเติมจนทำให้วิทยานิพนธ์ฉบับนี้สมบูรณ์ยิ่งขึ้น

สุดท้ายนี้ขอขอบพระคุณบิดา มารดา และครอบครัว ที่คอยเป็นกำลังใจและคอยช่วยเหลือมาโดยตลอด เกี่ยวกับกำลังทรัพย์ตลอดจนสำเร็จการศึกษา รวมถึงเพื่อน ๆ พี่ ๆ และน้อง ๆ ทุกท่านที่เป็นผู้สนับสนุนให้คำปรึกษาอย่างสม่ำเสมอและคอยให้กำลังใจตลอดจนงานวิจัยฉบับนี้เสร็จลุล่วงไปด้วยดี ด้วยความเคารพและขอบพระคุณอย่างสูง

กัณฐมณี วรอินทร์

ชื่อเรื่องภาษาไทย	SMART NU: ระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร
ผู้วิจัย	นางสาวกัณฐมณี วรอินทร์
ที่ปรึกษาวิทยานิพนธ์	รศ.ดร.สิทธิชัย ชูสำโรง
ประเภทสารนิพนธ์	วิทยานิพนธ์ วท.บ. สาขาภูมิศาสตร์ มหาวิทยาลัยนเรศวร, 2568
คำสำคัญ	แผนที่ออนไลน์,ระบบค้นหาเส้นทาง, Web Application, PostGIS, pgRouting, OpenStreetMap, Apache

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนา ระบบภูมิสารสนเทศเชิงพื้นที่ (Geospatial Information System) ในรูปแบบ เว็บแอปพลิเคชัน เพื่อเพิ่มประสิทธิภาพในการเดินทางและการรับรู้ข้อมูลสภาพอากาศภายในมหาวิทยาลัยนเรศวร ระบบที่พัฒนาขึ้นแบ่งออกเป็น 2 ส่วนหลัก ได้แก่ 1. ระบบนำทางอัจฉริยะ (Smart Routing System) โดยใช้ฐานข้อมูล OpenStreetMap (OSM) ร่วมกับ PostgreSQL/PostGIS และส่วนขยาย pgRouting ซึ่งประยุกต์ใช้อัลกอริทึม Dijkstra สำหรับคำนวณเส้นทางที่สั้นที่สุดจากจุดเริ่มต้นไปยังจุดหมายภายในมหาวิทยาลัย และ 2. ระบบรับรู้สภาพอากาศและสิ่งแวดล้อมแบบเรียลไทม์ (Real-time Environmental Awareness System) ที่รวบรวมข้อมูลจากสถานีตรวจวัด 10 แห่งภายในมหาวิทยาลัย ประกอบด้วยข้อมูลค่าฝุ่นละอองขนาดเล็ก (PM2.5) อุณหภูมิ ความชื้นสัมพัทธ์ ปริมาณน้ำฝน และความเร็วลม โดยใช้ Node-RED เป็นตัวกลางในการประมวลผลและส่งข้อมูลเข้าสู่ฐานข้อมูล PostgreSQL พร้อมทั้งเชื่อมต่อกับ API เพื่อแสดงตำแหน่งรถโดยสารแบบเรียลไทม์ระบบ SMART NU ถูกพัฒนาโดยใช้ภาษา HTML, JavaScript, CSS, และ PHP ทำงานผ่านเว็บเซิร์ฟเวอร์ Apache และใช้ Leaflet.js สำหรับการแสดงผลข้อมูลเชิงพื้นที่ ผู้ใช้งานสามารถเข้าถึงระบบผ่านเว็บเบราว์เซอร์เพื่อค้นหาเส้นทางและดูข้อมูลสภาพอากาศแบบเรียลไทม์ได้อย่างสะดวก ผลการพัฒนาพบว่าระบบสามารถคำนวณเส้นทางได้ถูกต้องและรวดเร็ว แสดงข้อมูลสภาพอากาศจากทุกสถานีได้แบบทันเวลา และสามารถนำไปประยุกต์ใช้ในการบริหารจัดการพื้นที่และการเดินทางภายในมหาวิทยาลัยได้อย่างมีประสิทธิภาพ

Title	SMART NU: Development of a Geospatial Information System for Transportation and Weather Awareness within Naresuan University
Author	Kantamanee Worein
Advisor	Associate Professor Dr. Sittichai Choosumrong
Academic Paper	Thesis B.S. in Geography,Naresuan University, 2025
Keywords	Online Map, Routing System, Web Application, PostGIS, pgRouting, OpenStreetMap, Apache

ABSTRACT

This research aims to develop a Geospatial Information System (GIS) as a Web Application to enhance travel efficiency and real-time environmental awareness within Naresuan University. The system features two core components 1.Smart Routing System: It uses an OpenStreetMap (OSM) database with PostgreSQL/PostGIS and pgRouting. The Dijkstra algorithm is applied to calculate the shortest path between points on campus and 2.Real-time Environmental Awareness System: This component collects PM2.5, temperature, humidity, rainfall, and wind speed data from 10 monitoring stations. Node-RED processes this data into the PostgreSQL database, and an API displays the real-time location of the shuttle bus.Developed using HTML, JavaScript, CSS, and PHP on an Apache server with Leaflet.js for visualization, the SMART NU system allows users convenient web browser access for routing and real-time data viewing. Results confirm the system accurately and quickly calculates routes and displays instant weather data, proving its effectiveness for managing space and transportation within the university.

สารบัญ

บทที่ 1 1

บทนำ 1

1.1 ที่มาและความสำคัญ 1

1.2 วัตถุประสงค์ของงานวิจัย 2

1.3 ประโยชน์ที่คาดว่าจะได้รับ 2

1.4 ขอบเขตพื้นที่การศึกษา 2

1.4.1 ขอบเขตพื้นที่ศึกษา 2

1.5 นิยามศัพท์เฉพาะ 3

1.6 สมมติฐานของการวิจัย 3

1.7 กรอบแนวคิด 4

บทที่ 2 5

เอกสารและงานวิจัยที่เกี่ยวข้อง 5

2.1 เครื่องมือที่ช่วยในการพัฒนาระบบเว็บแอปพลิเคชัน 6

2.1.1 โปรแกรมที่ใช้ในการเก็บจุดพิกัด: GeoJSON 6

2.1.2 โปรแกรมที่ใช้ในการรับส่งข้อมูล: Node-RED, MQTT 6

2.1.3 เว็บไซต์เครือข่ายถนน: OpenStreetMap 10

2.1.4 ภาษาที่ใช้ในการเขียนโปรแกรม: HTML, JavaScript, PHP, SQL, CSS 11

2.1.5 โปรแกรมที่ใช้สร้างหน้าเว็บ: Visual Studio Code 15

2.1.6 โปรแกรมจัดการฐานข้อมูล: PostgreSQL, PostGIS 16

2.1.7 โปรแกรมที่ใช้ในการคำนวณเส้นทาง pgRouting 18

2.1.8 ซอฟต์แวร์อื่นๆ: Web GIS / Web Map Application, Apache 20

สารบัญ (ต่อ)

2.2 เอกสารงานวิจัยที่เกี่ยวข้อง 22

บทที่ 3 26

วิธีดำเนินงานวิจัย 26

3.1 พื้นที่ศึกษา 26

3.2 วิธีการดำเนินงาน 26

3.2.1 เก็บและรวบรวมข้อมูล 26

3.2.2 จัดการฐานข้อมูลสภาพอากาศ 27

3.2.3 เตรียมฐานข้อมูลและนำเข้าข้อมูลเครือข่ายถนน 28

3.2.4 การแยกประเภทเส้นทาง 32

3.2.5 ใช้คำสั่ง SQL ในการค้นหาเส้นทางด้วย pgRouting ฟังก์ชัน pgr_dijkstra 33

3.2.6 ออกแบบหน้าเว็บแอปพลิเคชัน 36

3.2.7 การพัฒนาสร้างโค้ดฐานข้อมูล 37

3.2.8 การพัฒนาเว็บแอปพลิเคชัน 39

บทที่ 4 44

ผลการวิจัย 44

4.1 หน้าเว็บแอปพลิเคชันหลัก 44

4.2 หมวดหมู่ระบบนำทางภายใน ม.นเรศวร 45

4.3 หมวดหมู่สถานีตรวจสอบสภาพอากาศ 49

4.4 หมวดหมู่ระบบรถโดยสารในมหาวิทยาลัย 53

บทที่ 5 54

สรุปผลการวิจัยและอภิปราย 54

สารบัญ (ต่อ)

5.1 สรุปผลการวิจัย 54

5.2 อภิปรายผลการวิจัย 55

5.3 ข้อเสนอแนะ 56

บรรณานุกรม 57

ภาคผนวก 59

ประวัติผู้วิจัย 60

สารบัญรูปภาพ

ภาพที่ 1 พื้นที่ศึกษา 2

ภาพที่ 2 กรอบแนวคิด 4

ภาพที่ 3 การใช้งาน Node-RED 7

ภาพที่ 4 ตัวอย่าง MQTT 8

ภาพที่ 5 การทำงานของ MQTT 9

ภาพที่ 6 โปรแกรม PostgreSQL 17

ภาพที่ 7 โปรแกรม PostGIS 18

ภาพที่ 8 ตัวอย่างการค้นหาแส้นทาง 19

ภาพที่ 9 API ตำแหน่งรถโดยสาร 26

ภาพที่ 10 ข้อมูลสภาพอากาศใน Node-RED 27

ภาพที่ 11 ข้อมูลสภาพอากาศที่ถูกจัดเก็บในฐานข้อมูล PostgreSQL 27

ภาพที่ 12 การใช้คำสั่ง CREATE EXTENSION 28

ภาพที่ 13 รันคำสั่ง Command Prompt 29

ภาพที่ 14 ways ในฐานข้อมูล PostgreSQL 31

ภาพที่ 15 ways_vertices_pgr ในฐานข้อมูล PostgreSQL 32

ภาพที่ 16 ประเภทเส้นทางภายในมหาวิทยาลัยนเรศวร 33

ภาพที่ 17 การคำนวณเส้นทาง pgRouting 34

ภาพที่ 18 ผลลัพธ์การคำนวณเส้นทางรถยนต์ 35

ภาพที่ 19 ผลลัพธ์การคำนวณเส้นทางคนเดิน 35

ภาพที่ 20 การออกแบบหน้าเว็บแอปพลิเคชัน 36

ภาพที่ 21 ตัวอย่างโค้ด pgrouting.php การคำนวณเส้นทาง 37

ภาพที่ 22 ตัวอย่างโค้ด weather.php ดึงข้อมูลสภาพอากาศจากฐานข้อมูล 38

สารบัญรูปภาพ (ต่อ)

ภาพที่ 23 ตัวอย่างโค้ด weather_history.php ข้อมูลย้อนหลัง....38
ภาพที่ 24 ตัวอย่างโค้ด carapi.php ตำแหน่งรถโดยสาร....39
ภาพที่ 25 ตัวอย่างโค้ดหน้าเว็บหลัก....40
ภาพที่ 26 ตัวอย่างโค้ดการคำนวณเส้นทาง....40
ภาพที่ 27 ตัวอย่างโค้ดเรียกข้อมูล weather.php....41
ภาพที่ 28 ตัวอย่างโค้ดการสร้างแผนที่ประมาณค่าฝุ่น PM2.5 (IDW)....42
ภาพที่ 29 ตัวอย่างโค้ดตำแหน่งรถแบบเรียลไทม์....43
ภาพที่ 30 ตัวอย่างหน้าเว็บบนแอปพลิเคชัน....45
ภาพที่ 31 ตัวอย่างหน้าระบบนำทางเว็บแอปพลิเคชัน....45
ภาพที่ 32 การคำนวณเส้นทางคนเดินจากคณะไปยังจุดหมายปลายทางที่เป็นคณะ....46
ภาพที่ 33 การคำนวณเส้นทางรถยนต์จากคณะไปยังจุดหมายปลายทางที่เป็นคณะ....46
ภาพที่ 34 การคำนวณเส้นทางคนเดิน....47
ภาพที่ 35 การคำนวณเส้นทางรถยนต์....47
ภาพที่ 36 การคำนวณเส้นทางมอเตอร์ไซค์....48
ภาพที่ 37 การคำนวณเส้นทางจักรยาน....48
ภาพที่ 38 ตำแหน่งสถานีตรวจสอบสภาพอากาศ....49
ภาพที่ 39 การแสดงผลค่าฝุ่น PM2.5....50
ภาพที่ 40 การแสดงผลค่าอุณหภูมิ....50
ภาพที่ 41 การแสดงผลค่าความชื้น....51
ภาพที่ 42 การแสดงผลค่าความเร็วลม....51

สารบัญรูปภาพ (ต่อ)

ภาพที่ 43 การแสดงผลค่าปริมาณฝน....52

ภาพที่ 44 กราฟแสดงข้อมูลทั้ง10 สถานี....52

ภาพที่ 45 แผนที่ประมาณค่าฝุ่น PM2.5 IDW....53

ภาพที่ 46 ตำแหน่งรถโดยสารในมหาวิทยาลัย....53

ภาพที่ 47 โค้ดที่ใช้ในการพัฒนาระบบ....59

สารบัญตาราง

ตารางที่ 1 Node-RED 7

ตารางที่ 2 หน้าจอการตั้งค่าการเชื่อมต่อ 9

ตารางที่ 3 หน้าจอหลัก Subscribe 10

ตารางที่ 4 คำสั่งที่ใช้ osm2pgrouting 30

ตารางที่ 5 ข้อมูลในตาราง ways 31

ตารางที่ 6 ข้อมูลในตาราง ways_vertices_pgr 32

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญ

ในยุคปัจจุบัน เทคโนโลยีสารสนเทศภูมิศาสตร์ (Geographic Information Systems: GIS) ได้เข้ามามีบทบาทสำคัญในการพัฒนาแนวคิด “เมืองอัจฉริยะ” (Smart City) ซึ่งเน้นการยกระดับประสิทธิภาพในการบริหารจัดการพื้นที่ การอำนวยความสะดวกในการเดินทาง และการเข้าถึงข้อมูลสภาพแวดล้อมในพื้นที่อย่างแม่นยำและทันสถานการณ์ การนำ GIS มาประยุกต์ใช้สามารถช่วยเพิ่มประสิทธิภาพในการจัดการเมืองและเสริมสร้างความสะดวกสบายให้กับผู้ใช้บริการต่าง ๆ

มหาวิทยาลัยนเรศวรถือเป็นสถาบันอุดมศึกษาขนาดใหญ่ที่มีพื้นที่ใช้งานหลากหลายประเภท เช่น คณะ สำนักงาน ห้องสมุด โรงอาหารและศูนย์บริการต่าง ๆ รวมถึงระบบขนส่งภายในมหาวิทยาลัย เช่น รถโดยสารรับส่งนิสิต ที่มีเส้นทางครอบคลุมภายในวิทยาเขตการบริหารจัดการพื้นที่และการให้บริการข้อมูลด้านการเดินทางในปัจจุบันยังขาดระบบนำทางและการวิเคราะห์เส้นทางที่สามารถตอบสนองต่อความต้องการของผู้ใช้งานได้อย่างแม่นยำ โดยเฉพาะกลุ่มนิสิตใหม่ บุคลากร หรือผู้มาเยือนที่ไม่คุ้นเคยกับผังบริเวณของมหาวิทยาลัย ซึ่งยังจำเป็นต้องพึ่งพาการสอบถามเส้นทางจากบุคคลทั่วไป ซึ่งอาจทำให้เกิดความไม่สะดวกในการเดินทาง และอาจไม่ได้รับข้อมูลที่ครบถ้วนหรือแม่นยำเกี่ยวกับตำแหน่งอาคารหรือจุดบริการต่าง ๆ ภายในมหาวิทยาลัย นอกจากนี้ มหาวิทยาลัยยังขาดระบบที่สามารถแสดงข้อมูลสภาพแวดล้อมภายในพื้นที่ได้แบบเวลาจริง เช่น ค่าฝุ่นละอองขนาดเล็ก (PM2.5) ข้อมูลอุณหภูมิ และปริมาณน้ำฝน ซึ่งข้อมูลเหล่านี้มีความสำคัญในการวางแผนการเดินทาง โดยเฉพาะในกรณีที่มีมลพิษทางอากาศสูงหรือสภาพอากาศที่ไม่เหมาะสม ซึ่งอาจมีผลกระทบต่อสุขภาพและความปลอดภัยของผู้ใช้งาน

จากปัจจัยดังกล่าว จึงเกิดความจำเป็นในการพัฒนา "SMART NU" ซึ่งเป็นระบบภูมิสารสนเทศเชิงพื้นที่และสิ่งแวดล้อม ที่สามารถรองรับการนำทางและการรับรู้ข้อมูลสภาพแวดล้อมในมหาวิทยาลัยนเรศวร โดยระบบจะพัฒนาเป็นแอปพลิเคชันบนเว็บ (Web-based Application) ซึ่งช่วยให้ผู้ใช้งานสามารถเข้าถึงข้อมูลเชิงพื้นที่ ที่ถูกต้อง คำนวณเส้นทางการเดินทางที่เหมาะสม และรับรู้ข้อมูลสภาพแวดล้อมในเวลาจริง การพัฒนาระบบนี้จะช่วยเพิ่มความสะดวกสบายในการใช้ชีวิตและการทำกิจกรรมภายในมหาวิทยาลัย โดยมุ่งเน้นการสร้างความสะดวกสบายและความปลอดภัยในการเดินทางทั้งในแง่ของการลดเวลาในการค้นหาทางและการปรับตัวให้เข้ากับสภาพแวดล้อมได้ดีขึ้น

1.2 วัตถุประสงค์ของงานวิจัย

เพื่อพัฒนาเว็บไซต์ระบบภูมิสารสนเทศที่สามารถค้นหาเส้นทางและคำนวณเส้นทางภายในมหาวิทยาลัยนเรศวรด้วย pgRouting และสามารถแสดงข้อมูลสภาพอากาศภายในมหาวิทยาลัยนเรศวร

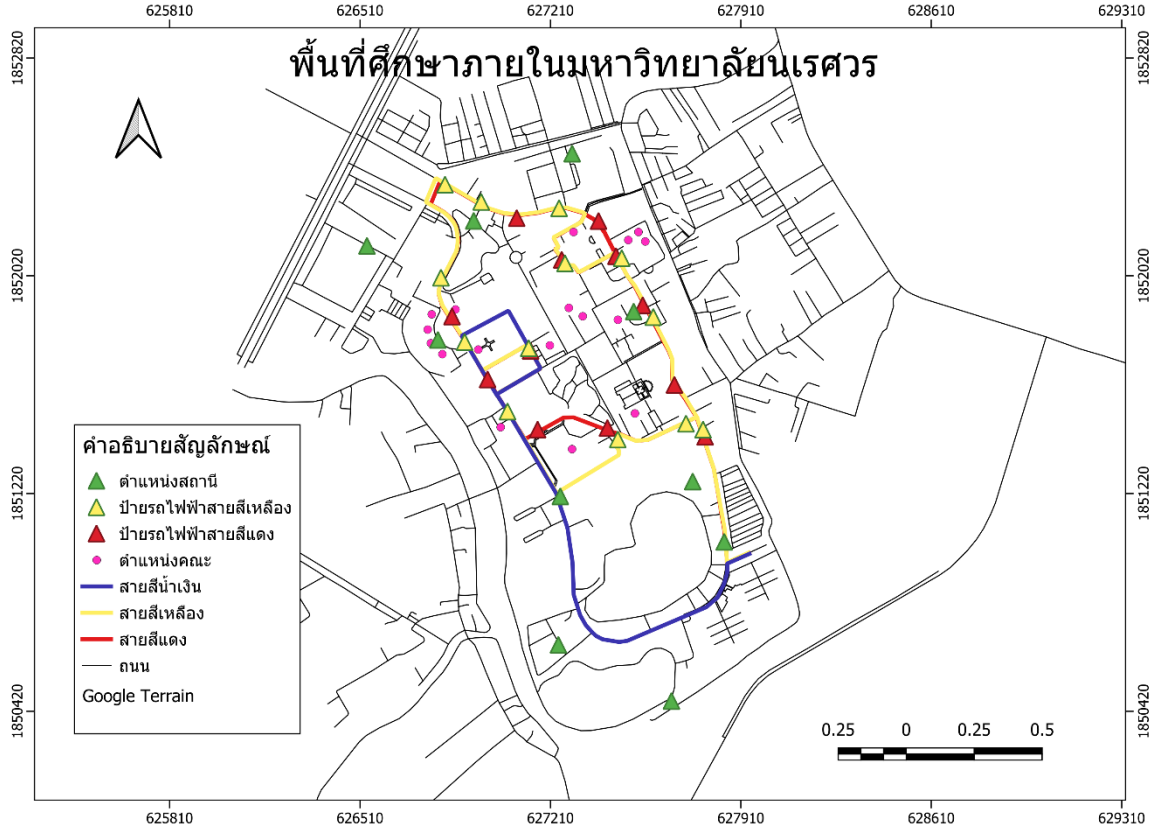
1.3 ประโยชน์ที่คาดว่าจะได้รับ

การพัฒนาระบบ SMART NU จะช่วยให้ได้ระบบภูมิสารสนเทศเชิงพื้นที่ในรูปแบบเว็บแอปพลิเคชันที่สามารถคำนวณและแสดงเส้นทางการเดินทางภายในมหาวิทยาลัยนเรศวรได้อย่างถูกต้องและรวดเร็ว พร้อมทั้งแสดงข้อมูลสภาพอากาศแบบเรียลไทม์เพื่อเพิ่มความสะดวก ปลอดภัย และประสิทธิภาพในการเดินทาง รวมถึงสนับสนุนการบริหารจัดการพื้นที่และการพัฒนาแนวคิดมหาวิทยาลัยอัจฉริยะได้

1.4 ขอบเขตพื้นที่การศึกษา

1.4.1 ขอบเขตพื้นที่ศึกษา

การศึกษาในครั้งนี้ได้กำหนดขอบเขตพื้นที่การศึกษา คือ ภายในมหาวิทยาลัยนเรศวร อำเภอเมือง จังหวัดพิษณุโลก



ภาพที่1 พื้นที่ศึกษา

1.5 นิยามศัพท์เฉพาะ

แผนที่ออนไลน์ การแสดงผลเชิงกราฟิกของพื้นที่ทางภูมิศาสตร์ไม่ว่าจะเป็น ถนน, อาคาร, ภูมิประเทศ ในรูปแบบดิจิทัล ซึ่งสามารถเข้าถึงและใช้งานได้ผ่านทางอินเทอร์เน็ตบนอุปกรณ์ต่างๆ

ระบบค้นหาเส้นทาง ฟังก์ชันหรืออัลกอริทึมหลัก ที่ทำหน้าที่ในการคำนวณวิเคราะห์ และกำหนดเส้นทางที่เหมาะสมที่สุดจากจุดเริ่มต้น ไปยังจุดหมายปลายทาง

Web Application โปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นเพื่อใช้งานผ่านเว็บเบราว์เซอร์ โดยผู้ใช้ไม่จำเป็นต้องติดตั้งโปรแกรมลงในอุปกรณ์และสามารถเข้าถึงการทำงานของระบบได้จากทุกที่ที่มีการเชื่อมต่ออินเทอร์เน็ต

PostGIS เป็นส่วนขยาย (Extension) ที่ติดตั้งบน PostgreSQL เพื่อเพิ่มขีดความสามารถในการจัดเก็บและจัดการข้อมูลเชิงพื้นที่ในรูปแบบของวัตถุทางภูมิศาสตร์ เช่น จุด (Point), เส้น (Line), และรูปหลายเหลี่ยม (Polygon)

pgRouting เครื่องมือที่เป็นส่วนขยาย (Extension) ที่ทำงานร่วมกับฐานข้อมูลเชิงพื้นที่ PostgreSQL/PostGIS โดยเพิ่มฟังก์ชันการคำนวณระยะทาง

OpenStreetMap ฐานข้อมูลแผนที่โลกแบบเปิด (Open Geospatial Database) ที่อนุญาตให้ผู้ใช้สามารถสร้าง แก้ไข และแบ่งปันข้อมูลภูมิสารสนเทศได้อย่างเสรี

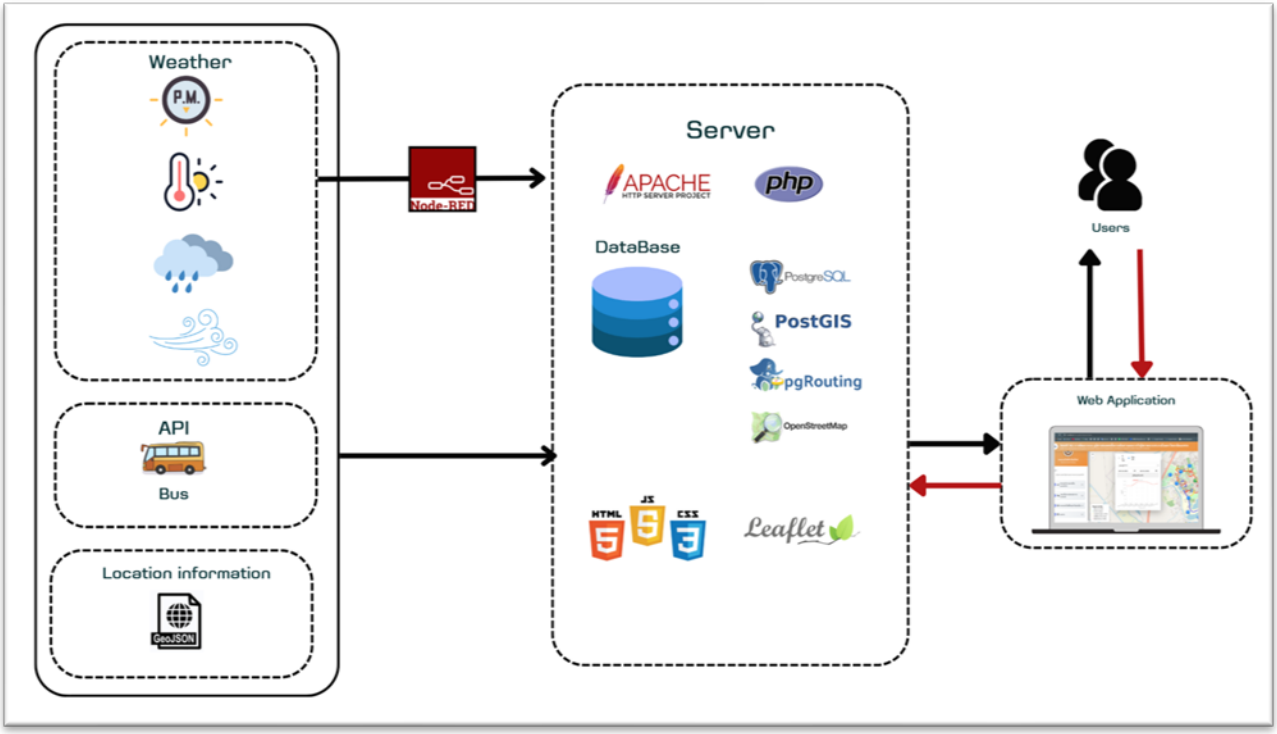
Apache คือ เว็บเซิร์ฟเวอร์แบบโอเพนซอร์ส (Open Source Web Server) ที่ได้รับความนิยมสูงในการให้บริการเว็บไซต์และเว็บแอปพลิเคชัน โดยมีหน้าที่รับคำขอ (HTTP requests) จากผู้ใช้ผ่านเว็บเบราว์เซอร์และตอบสนองด้วยการส่งข้อมูลกลับมายังผู้ใช้ Apache รองรับการทำงานร่วมกับภาษาโปรแกรมต่างๆ จึงเหมาะสำหรับใช้งานเป็นเซิร์ฟเวอร์สำหรับระบบ Web GIS เพื่อให้บริการข้อมูลและแผนที่ผ่านเว็บ

1.6 สมมติฐานของการวิจัย

ระบบที่พัฒนาขึ้นสามารถคำนวณและแสดงเส้นทางภายในมหาวิทยาลัยนเรศวรและสามารถดูข้อมูลสภาพอากาศได้

1.7 กรอบแนวคิด

ระบบจะรับข้อมูลสภาพอากาศแบบเรียลไทม์ (ค่าฝุ่น PM, อุณหภูมิ, ความชื้น, ปริมาณฝน, และความเร็วลม) จากเซนเซอร์ ซึ่งถูกส่งผ่าน Node-RED เพื่อประมวลผลและจัดเก็บในฐานข้อมูล PostgreSQL ร่วมกับการดึง API ตำแหน่งรถโดยสารแบบเรียลไทม์และข้อมูลพิกัดสถานที่สำคัญ โดยมีเซิร์ฟเวอร์ที่ใช้ Apache และ PHP ในการประมวลผลคำขอ จัดการข้อมูลเครือข่ายถนนจาก OpenStreetMap ในฐานข้อมูล PostgreSQL/PostGIS และใช้ pgRouting ที่มีอัลกอริทึม Dijkstra ในการคำนวณหาเส้นทางที่สั้นที่สุด ก่อนจะแสดงผลลัพธ์บนเว็บแอปพลิเคชันผ่าน HTML, JavaScript, CSS และไลบรารี Leaflet.js เพื่อให้ผู้ใช้งานสามารถเข้าถึงบริการการคำนวณเส้นทางและการรับรู้สภาพอากาศแบบเรียลไทม์ได้



ภาพที่ 2 กรอบแนวคิด

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

ในการวิจัยและพัฒนาระบบเว็บแอปพลิเคชันเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร ผู้วิจัยได้ทำการศึกษาเอกสารทางวิชาการ ทฤษฎีเชิงเทคนิค ตลอดจนงานวิจัยที่เกี่ยวข้องในหลายด้าน เพื่อใช้เป็นแนวทางในการออกแบบและพัฒนาระบบให้ทำงานได้อย่างถูกต้องและมีประสิทธิภาพ โดยเนื้อหาที่ศึกษาแบ่งออกเป็นหัวข้อต่าง ๆ มีรายละเอียดดังต่อไปนี้

2.1 เครื่องมือที่ช่วยในการพัฒนาระบบเว็บแอปพลิเคชัน

2.1.1. โปรแกรมที่ใช้ในการเก็บจุดพิกัด: GeoJSON

2.1.2. โปรแกรมที่ใช้ในการรับส่งข้อมูล: MQTT, Node-RED

2.1.3. เว็บไซต์เครือข่ายถนน: OpenStreetMap

2.1.4. ภาษาที่ใช้ในการเขียนโปรแกรม: HTML, JavaScript, PHP, SQL, CSS

2.1.5. โปรแกรมที่ใช้สร้างหน้าเว็บ: Visual Studio Code

2.1.6. โปรแกรมจัดการฐานข้อมูล: PostgreSQL, PostGIS

2.1.7. โปรแกรมที่ใช้ในการคำนวณเส้นทาง: pgRouting

2.1.8. ซอฟต์แวร์อื่นๆ: Web GIS / Web Map Application, Apache

2.2 เอกสารงานวิจัยที่เกี่ยวข้อง

2.1 เครื่องมือที่ช่วยในการพัฒนาระบบเว็บแอปพลิเคชัน

2.1.1 โปรแกรมที่ใช้ในการเก็บจุดข้อมูล: GeoJSON

GeoJSON (Geographic JavaScript Object Notation) คือ รูปแบบข้อมูลมาตรฐานเปิด (Open Standard Format) ที่ใช้สำหรับการเข้ารหัสและแลกเปลี่ยนข้อมูลเชิงพื้นที่ (Geospatial Data) โดยอิงตามโครงสร้างของ JSON (JavaScript Object Notation) ซึ่งเป็นรูปแบบข้อมูลที่อ่านและเขียนได้ง่ายสำหรับมนุษย์ รวมถึงประมวลผลได้สะดวกโดยภาษา JavaScript และภาษาคอมพิวเตอร์อื่นๆ GeoJSON สามารถใช้แทนข้อมูลทางภูมิศาสตร์ในรูปแบบต่าง ๆ เช่น จุด (Point), เส้น (LineString), รูปหลายเหลี่ยม (Polygon), รวมถึงกลุ่มของวัตถุ (FeatureCollection) ซึ่งสามารถรวมคุณลักษณะ (Attributes) และค่าพิกัดทางภูมิศาสตร์ (Latitude, Longitude) เข้าด้วยกัน เพื่อใช้ในการแสดงผลบนแผนที่หรือการวิเคราะห์ข้อมูลเชิงพื้นที่

GeoJSON ประกอบไปด้วย 2 ส่วนหลัก

1. คุณสมบัติเชิงเรขาคณิต (Geometry) กำหนดรูปร่างทางภูมิศาสตร์ของวัตถุ เช่น จุด เส้น หรือพื้นที่ปิด โดยระบุเป็นชุดของพิกัดทางภูมิศาสตร์ (ละติจูดและลองจิจูด)
2. ข้อมูลคุณสมบัติ (Properties) ข้อมูลเพิ่มเติมเกี่ยวกับวัตถุทางภูมิศาสตร์ที่ไม่ใช่เชิงพื้นที่ เช่น ชื่อ, ประเภท, คำอธิบาย, หรือข้อมูลอื่น ๆ ที่เกี่ยวข้อง

2.1.2 โปรแกรมที่ใช้ในการรับส่งข้อมูล: Node-RED, MQTT

Node-RED คือเครื่องมือสำหรับนักพัฒนาโปรแกรมที่ออกแบบมาเพื่อเชื่อมต่ออุปกรณ์ฮาร์ดแวร์เข้ากับ API (Application Programming Interface) และช่วยในการเขียนโปรแกรมแบบกำหนดขั้นตอน (Flow-Based Programming) ลักษณะและการทำงานของ Node-RED ใช้แนวคิดการเขียนโปรแกรมแบบเชื่อมโยง (Flow-Based) ซึ่งช่วยให้ผู้พัฒนาสามารถทำงานได้โดยไม่ต้องเขียนโค้ดจำนวนมาก เพียงแค่ลากและวาง (Drag and Drop) โหนด (Node) ลงในผังงาน (Flow) แล้วเชื่อมต่อกัน ก็สามารถสร้างกระบวนการทำงานได้

Node-RED ถูกพัฒนาโดย Nick O'Leary และ Dave Conway-Jones ในปี ค.ศ. 2013 ภายใต้โครงการของ IBM Emerging Technology เพื่อเชื่อมต่อข้อมูลจากอุปกรณ์ Internet of Things (IoT) ผ่านโปรโตคอล MQTT และแสดงผลในรูปแบบ Visualization ผ่านเว็บเบราว์เซอร์ ต่อมาในปี 2016 ได้เข้าร่วมกับ JS Foundation และในปี 2019 รวมเป็นส่วนหนึ่งของ OpenJS Foundation ซึ่งดูแลโดย FlowFuse Inc เพื่อพัฒนาให้รองรับการใช้งานในระดับองค์กร จุดเด่นคือการทำงานผ่านเว็บเบราว์เซอร์โดยใช้

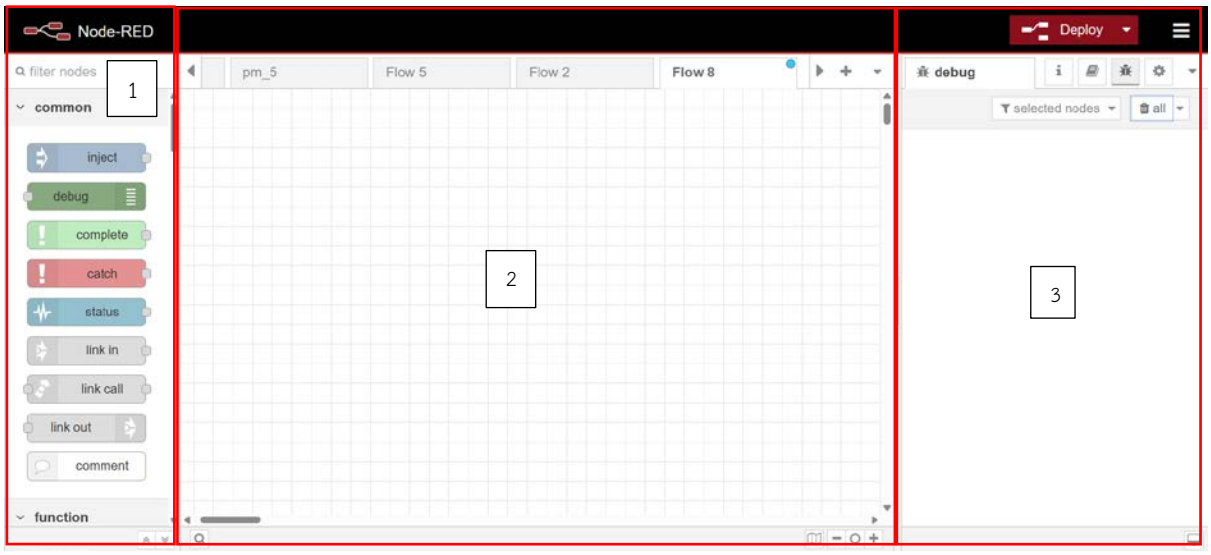
สถาปัตยกรรมของ Node.js ผู้ใช้สามารถเข้าถึงและจัดการข้อมูลได้จากทุกที่ มี Node สำเร็จรูปมากมายให้เลือกใช้งาน เช่น Node สำหรับเชื่อมต่อฐานข้อมูล, Node สำหรับประมวลผลข้อมูล, และ Node สำหรับส่งข้อมูลออกไปยัง API หรืออุปกรณ์อื่น ๆ นอกจากนี้ยังมี Node-RED Dashboard ที่ช่วยสร้างอินเทอร์เฟซผู้ใช้ (User Interface) สำหรับแสดงผลและควบคุมอุปกรณ์ IoT แบบเรียลไทม์ผ่านเว็บเบราว์เซอร์ได้โดยไม่ต้องเขียนโค้ดเพิ่มเติม

การใช้งาน Node-RED

1. Sidebar ด้านซ้าย แสดงกลุ่มโหนด (Node) เช่น input, output, functio

2. Workspace กลางสำหรับลากและเชื่อมต่อโหนดเพื่อสร้าง flow

3. Info & Debug ด้านขวา แสดงข้อมูลและผลลัพธ์ที่ได้จากโหนด

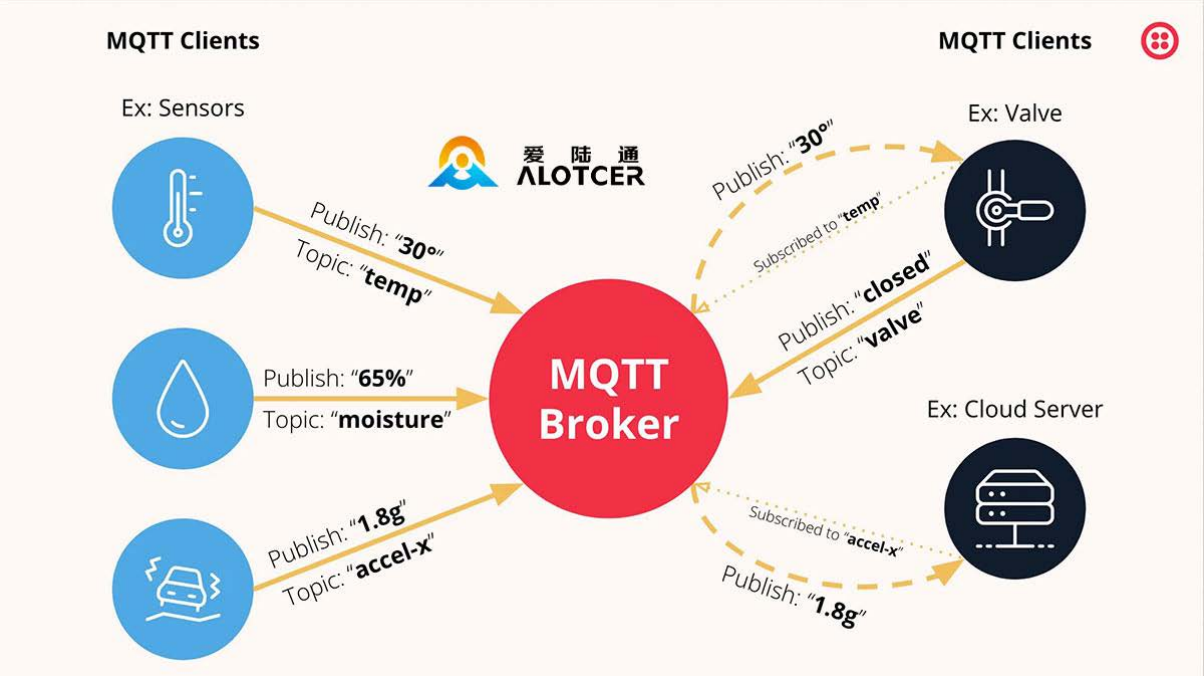


ภาพที่ 3 การใช้งาน Node-RED

ตารางที่ 1 Node-RED

ประเภท	ชื่อโหนด	คำอธิบาย
Input	Inject	ส่งค่าตามเวลาหรือกดเอง
Output	Debug	แสดงผลลัพธ์ในหน้าจอ
Function	Function	เขียน JavaScript เพื่อประมวลผลข้อมูล
Network	MQTT	รับหรือส่งข้อมูลผ่านโปรโตคอล MQTT
Storage	File	อ่านและเขียนไฟล์ในระบบ
UI	Dashboard	สร้างหน้าจอควบคุมสำหรับผู้ใช้

MQTT (Message Queuing Telemetry Transport) เป็นโปรโตคอลสำหรับการสื่อสารข้อมูลแบบเบา (Lightweight Messaging Protocol) ที่ออกแบบมาเพื่อใช้ในระบบที่มีข้อจำกัดด้านพลังงาน แบนด์วิธ หรือการเชื่อมต่อ เช่น ระบบ Internet of Things (IoT) โดยโปรโตคอลนี้ถูกพัฒนาโดย Andy Stanford-Clark จากบริษัท IBM และ Arlen Nipper จากบริษัท Arcom ในปี ค.ศ. 1999 เพื่อใช้ในการส่งข้อมูลจากอุปกรณ์ระยะไกลที่มีทรัพยากรจำกัด เช่น เซนเซอร์หรือไมโครคอนโทรลเลอร์ MQTT ใช้รูปแบบการสื่อสารแบบ Publish/Subscribe Model ซึ่งช่วยให้การแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์เป็นไปอย่างมีประสิทธิภาพโดยมีองค์ประกอบหลัก ได้แก่ ส่วนที่หนึ่ง Publisher อุปกรณ์หรือโปรแกรมที่ทำหน้าที่ส่งข้อมูล (เช่น เซนเซอร์อุณหภูมิส่งค่ามายัง Broker) ส่วนที่สอง Broker เซิร์ฟเวอร์กลางที่ทำหน้าที่จัดการและส่งต่อข้อมูลระหว่าง Publisher, Subscriber และส่วนที่สาม Subscriber – อุปกรณ์หรือโปรแกรมที่สมัครรับข้อมูลจาก Broker



ภาพที่ 4 ตัวอย่างMQTT

ที่มา: https://www.alotceriot.com/what-is-mqtt-messaging-protocol-iot/

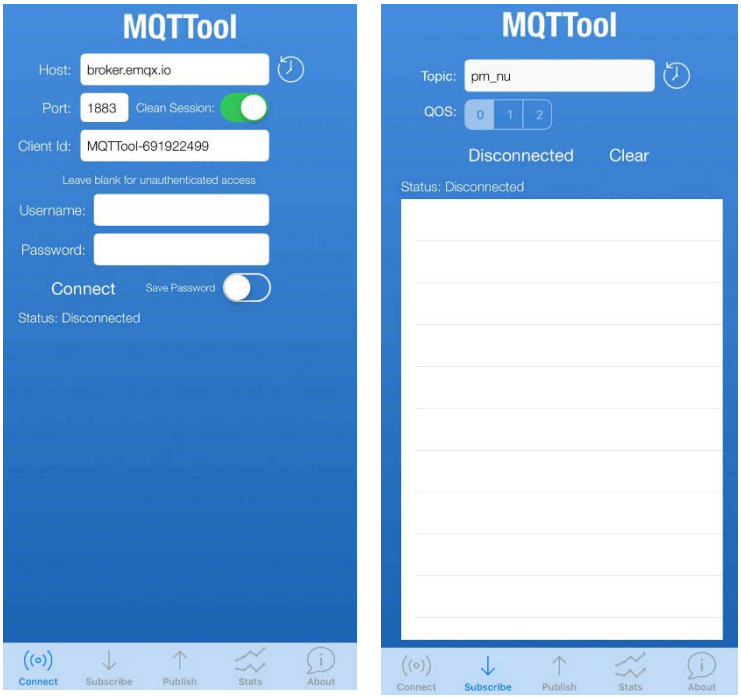
1. Publisher (ฝ่ายส่งข้อมูล) ซึ่งในภาพแสดงด้วยไอคอนของอุปกรณ์เซ็นเซอร์และอุปกรณ์ไฟฟ้า Publisher จะสร้างและส่งข้อความ (Message) ซึ่งโดยทั่วไปคือข้อมูลที่ได้จากเซ็นเซอร์ เช่น อุณหภูมิ หรือสถานะอุปกรณ์ ส่งข้อความเหล่านี้ไปยัง MQTT Broker ผ่านช่องทางที่เรียกว่า "Topic"

2. MQTT Broker (ตัวกลาง) เป็นหัวใจของระบบ ทำหน้าที่รับข้อความจาก Publisher แล้วจัดเก็บและกระจายข้อความไปยัง Subscriber ที่สมัครรับข้อมูลของ Topic นั้น โดย Broker ทำหน้าที่เป็นตัวกลางควบคุมการแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์ต่าง ๆ ในระบบ

3. Subscriber (ฝ่ายรับข้อมูล) ผู้รับข้อมูล ซึ่งในภาพแสดงเป็นโทรศัพท์มือถือและคอมพิวเตอร์ Subscriber จะสมัครรับข้อมูล (Subscribe) ใน Topic ที่สนใจ เมื่อ Broker ได้รับข้อมูลใหม่จาก Publisher ก็จะส่งข้อความเหล่านั้นไปให้ Subscriber ที่สมัครไว้โดยอัตโนมัติ

การทำงานของ MQT

ประกอบไปด้วย หน้าจอการตั้งค่าการเชื่อมต่อ (Connection Settings) และ หน้าจอหลัก Subscribe



ภาพที่ 5 การทำงานของ MQT

ตารางที่ 2 หน้าจอการตั้งค่าการเชื่อมต่อ

พารามิเตอร์	ความหมายและการใช้งาน
Host	ที่อยู่ (Address) ของ MQTT Broker (เซิร์ฟเวอร์กลาง) ที่ใช้รับและส่งข้อมูล
Port	พอร์ตมาตรฐาน ที่ใช้สำหรับการเชื่อมต่อ MQTT (ปกติคือ 1883 สำหรับ unencrypted หรือ 8883 สำหรับ SSL/TLS)
Client Id	รหัสเฉพาะ ที่ใช้ระบุตัวตนของอุปกรณ์หรือไคลเอนต์ที่เชื่อมต่อกับ Broker ต้องไม่ซ้ำกัน
Username / Password	ข้อมูลรับรองความถูกต้อง (Credentials) ใช้สำหรับเข้าสู่ระบบ หาก Broker มีการกำหนดให้ต้องล็อกอิน

ตารางที่ 3 หน้าจอหลัก Subscribe

พารามิเตอร์	ความหมายและการใช้งาน
Topic	หัวข้อ หรือ ช่องทาง ที่ต้องการรับส่งข้อความ เช่น อุณหภูมิหรือความชื้น
Status	สถานะปัจจุบัน ของการเชื่อมต่อแสดงสถานะ Disconnected หมายถึงอุปกรณ์ยังไม่ได้เชื่อมต่อกับ Broker

2.1.3 เว็บไซต์เครือข่ายถนน: OpenStreetMap

OpenStreetMap (OSM) เป็นฐานข้อมูลแผนที่โลกแบบเปิด (Open Geospatial Database) ที่อนุญาตให้ผู้ใช้สามารถสร้าง แก้ไข และแบ่งปันข้อมูลภูมิสารสนเทศได้อย่างเสรี โดยก่อตั้งขึ้นในปี ค.ศ. 2004 โดย Steve Coast ที่ประเทศอังกฤษ มีแนวคิดมาจากการสร้าง “แผนที่ของโลกโดยประชาชน” คล้ายกับแนวคิดของวิกิพีเดีย (The Free Wiki World Map) (OpenStreetMap Foundation, 2024) ต่อมาในปี ค.ศ. 2006 ได้มีการจัดตั้ง OpenStreetMap Foundation (OSMF) เพื่อดูแลโครงการและชุมชนผู้ใช้ทั่วโลกและในปี ค.ศ. 2012 OSM ได้ปรับใช้สัญญาอนุญาต Open Database License (ODbL) เพื่อเปิดโอกาสให้บุคคลและองค์กรสามารถนำข้อมูลไปใช้งานได้ฟรี โดยมีเงื่อนไขเพียงให้เครดิตแก่ OSM และผู้สร้างข้อมูล (ODbL, 2012) ปัจจุบัน OSM มีฐานข้อมูลแผนที่ที่ละเอียดและครอบคลุมทั่วโลก มีการอัปเดตโดยอาสาสมัครนับล้านคน ข้อมูลถูกนำไปใช้ในระบบสารสนเทศภูมิศาสตร์ (GIS) และผลิตภัณฑ์ของบริษัทเทคโนโลยีรายใหญ่ เช่น Apple Maps, Microsoft Bing Maps, และ Mapbox เนื่องจากความยืดหยุ่น ความเป็นอิสระ และคุณภาพของข้อมูลที่ได้รับการตรวจสอบจากชุมชนอย่างต่อเนื่อง

2.1.4 ภาษาที่ใช้ในการเขียนโปรแกรม: HTML, JavaScript, PHP, SQL, CSS

ภาษา HTML (Hyper Text Markup Language) คือภาษาคอมพิวเตอร์ที่ใช้สำหรับกำหนดโครงสร้างและเนื้อหาของเอกสารบนเว็บไซต์ หรือที่เรียกว่า “เว็บเพจ” โดยใช้หลักการของ Markup Language ซึ่งหมายถึงภาษาที่ใช้ “แท็ก (Tag)” ในการระบุว่าเนื้อหาส่วนใดมีความหมายหรือทำหน้าที่อะไร เช่น <p> สำหรับย่อหน้า, <img> สำหรับแทรกรูปภาพ, และ <a> สำหรับลิงก์เชื่อมโยง ภาษา HTML ถูกพัฒนาและกำหนดมาตรฐานโดยองค์กร World Wide Web Consortium (W3C) เพื่อให้เว็บเพจที่สร้างขึ้นสามารถแสดงผลได้อย่างถูกต้องและเป็นมาตรฐานเดียวกันในทุกเว็บเบราว์เซอร์ เช่น Microsoft Edge, Google Chrome, Mozilla Firefox, Opera, และ Safari นอกจากนี้ จากการพัฒนาของบริษัท Microsoft ทำให้ HTML สามารถใช้สร้างโปรแกรมประยุกต์ในลักษณะของ HTML Application (HTA) ได้อีกด้วย ซึ่งช่วยให้การเขียนเว็บไซต์มีความยืดหยุ่นและสามารถเชื่อมโยงกับเทคโนโลยีอื่นๆได้อย่างหลากหลาย

ลักษณะเด่นของภาษา HTML

1. เป็นภาษามาตรฐานกลางของเว็บ (Web Standard Language) HTML เป็นภาษาหลักที่ทุกเว็บไซต์บนโลกใช้ร่วมกัน ไม่ว่าจะเป็นเว็บไซต์ขนาดเล็กหรือระบบขนาดใหญ่ ทำให้สามารถแสดงผลได้ในทุกเบราว์เซอร์โดยไม่ต้องใช้ปลั๊กอินเพิ่มเติม

2. โครงสร้างแบบแท็ก (Tag-based Structure) HTML ใช้แท็กในการระบุส่วนประกอบของเนื้อหา เช่น <head>, <body>, <table>, <form> ทำให้โครงสร้างเว็บเข้าใจง่ายและสามารถปรับแต่งได้ตามต้องการ

3. สามารถผสานการทำงานกับภาษาอื่น HTML ทำงานร่วมกับ CSS สำหรับตกแต่งรูปแบบ และ JavaScript สำหรับการโต้ตอบ (interactivity) ได้อย่างสมบูรณ์แบบ ซึ่งเป็นพื้นฐานของการพัฒนาเว็บยุคใหม่

4. รองรับมัลติมีเดียและเทคโนโลยีใหม่ HTML5 รองรับการฝังวิดีโอ เสียง และกราฟิกแบบอินเทอร์แอคทีฟ เช่น Canvas และ SVG โดยไม่ต้องพึ่งโปรแกรมเสริมอย่าง Flash

5. ใช้งานง่ายและเรียนรู้ไม่ซับซ้อน HTML เป็นภาษาที่เหมาะกับผู้เริ่มต้น เพราะสามารถสร้างเว็บเพจได้ด้วยโปรแกรมพื้นฐานอย่าง Notepad, EditPlus หรือโปรแกรมช่วยเขียนเว็บ เช่น Dreamweaver, Visual Studio Code เป็นต้น

ความสำคัญของภาษา HTML ในการพัฒนาเว็บ HTML ถือเป็นรากฐานสำคัญของการพัฒนาเว็บทุกประเภท ไม่ว่าจะเป็นเว็บไซต์ทั่วไป เว็บแอปพลิเคชัน หรือระบบข้อมูลออนไลน์ การเข้าใจโครงสร้างของ HTML จะช่วยให้นักพัฒนาสามารถเชื่อมโยงกับเทคโนโลยีอื่น เช่น JavaScript, PHP และฐานข้อมูลได้อย่างมีประสิทธิภาพ นอกจากนี้ HTML ยังมีบทบาทสำคัญในการออกแบบเว็บเชิงภูมิสารสนเทศ (Web GIS) เช่น การแสดงแผนที่เชิงโต้ตอบ (Interactive Map) โดยใช้ไลบรารี Leaflet.js หรือ OpenLayers

ภาษา JavaScript คือ ภาษาคอมพิวเตอร์สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ตที่ได้รับความนิยมสูง เนื่องจากสามารถทำให้เว็บไซต์มีการโต้ตอบ (Interactive) และตอบสนองต่อการใช้งานของผู้ใช้ได้แบบเรียลไทม์ เป็นภาษาสคริปต์เชิงวัตถุ (Object-Oriented Scripting Language) ที่ออกแบบมาให้ทำงานร่วมกับภาษา HTML และ CSS เพื่อสร้างเว็บเพจแบบไดนามิกและอินเทอร์แอคทีฟ JavaScript ถูกพัฒนาโดยบริษัท Netscape Communications Corporation โดย Brendan Eich ในปี ค.ศ. 1995 เดิมใช้ชื่อว่า LiveScript และเปิดตัวครั้งแรกพร้อมกับเบราว์เซอร์ Netscape Navigator 2.0 ต่อมาได้เปลี่ยนชื่อเป็น JavaScript เพื่อให้สอดคล้องกับกระแสความนิยมของภาษา Java ในขณะนั้น ภายหลัง JavaScript ได้รับการรับรองให้เป็นมาตรฐานโดยองค์กร ECMA International ในชื่อ ECMAScript (ES) ซึ่งเป็นมาตรฐานกลางของภาษา JavaScript ทั่วโลก

ลักษณะเด่นของภาษา JavaScript

1. เป็นภาษาสคริปต์ที่ทำงานฝั่งผู้ใช้ (Client-side) JavaScript สามารถทำงานบนเว็บเบราว์เซอร์โดยตรง โดยไม่ต้องประมวลผลผ่านเซิร์ฟเวอร์ ทำให้สามารถโต้ตอบกับผู้ใช้งานได้ทันที เช่น การตรวจสอบข้อมูลในฟอร์ม หรือการเปลี่ยนแปลงเนื้อหาหน้าเว็บแบบไดนามิก

2. รองรับการทำงานแบบวัตถุ (Object-Oriented) JavaScript มีโครงสร้างแบบวัตถุ ซึ่งช่วยให้สามารถสร้างออบเจกต์ (Objects) และคลาส (Classes) สำหรับจัดการข้อมูลและพฤติกรรมขององค์ประกอบบนเว็บได้อย่างมีระบบ

3. มีความยืดหยุ่นสูง (Flexible Typing) JavaScript ไม่จำเป็นต้องกำหนดชนิดข้อมูลล่วงหน้า (Dynamic Typing) ทำให้เขียนโปรแกรมได้สะดวกและลดความซับซ้อนในการพัฒนา

4. ทำงานได้บนทุกแพลตฟอร์ม (Cross-platform) เนื่องจาก JavaScript ทำงานภายในเบราว์เซอร์ ทำให้สามารถใช้ได้บนทุกระบบปฏิบัติการ เช่น Windows, macOS, Linux, Android, และ iOS

5. รองรับการทำงานแบบอะซิงโครนัส (Asynchronous) ด้วยเทคนิคอย่าง Callback, Promise, และ Async/Await JavaScript สามารถประมวลผลหลายงานพร้อมกันได้โดยไม่ต้องรอคำสั่งก่อนหน้าเสร็จสิ้น เหมาะสำหรับเว็บที่ต้องติดต่อกับ API หรือฐานข้อมูลแบบเรียลไทม์

ภาษาPHP (Hypertext Preprocessor) มีประโยชน์หลักในด้านการสร้างเว็บไซต์แบบไดนามิกที่สามารถแสดงเนื้อหาหรือข้อมูลที่เปลี่ยนแปลงตามผู้ใช้และฐานข้อมูลได้ ทำให้เว็บไซต์มีความสามารถในการโต้ตอบและปรับตัวตามความต้องการ นอกจากนี้ PHP ยังรองรับการเชื่อมต่อกับฐานข้อมูลหลากหลายชนิด เช่น MySQL, PostgreSQL, MongoDB และอื่น ๆ ทำให้การจัดการและแสดงผลข้อมูลเป็นไปอย่างมีประสิทธิภาพและยืดหยุ่นสูง อีกทั้ง PHP สามารถทำงานร่วมกับภาษาของเว็บอย่าง HTML, CSS, JavaScript ได้อย่างราบรื่น ส่งผลให้เป็นภาษาที่เหมาะสมในการพัฒนาเว็บแอปพลิเคชันขนาดใหญ่ เช่น ระบบสมาชิก ระบบอีคอมเมิร์ซ และระบบจัดการข้อมูลต่าง ๆ

โดยเริ่มต้นพัฒนาในปี 1994 โดย Rasmus Lerdorf โปรแกรมเมอร์ชาวแคนาดา-เดนมาร์ก เขาได้สร้างชุดเครื่องมือชื่อ Personal Home Page (PHP) เพื่อใช้ติดตามผู้ที่เข้าชมเว็บไซต์ส่วนตัวของเขา PHP มีรากฐานมาจากภาษา C และ Perl ซึ่งในช่วงแรกเรียกว่า PHP/FI (Personal Home Page / Form Interpreter) โดยมีการพัฒนาคุณสมบัติการติดต่อกับฐานข้อมูลและการจัดการฟอร์มข้อมูลภายในเว็บไซต์ จากนั้นในช่วงปี 1995-1997 มีนักพัฒนาชาวอิสราเอลสองคนคือ Zeev Suraski และ Andi Gutmans มาร่วมพัฒนา PHP โดยปรับปรุงพื้นฐานของภาษาใหม่ในเวอร์ชัน PHP 3 ที่รองรับการเขียนโปรแกรมในรูปแบบ OOP (Object-Oriented Programming) และเพิ่มการติดต่อฐานข้อมูลได้หลากหลายมากขึ้น PHP 4 เปิดตัวในปี 2000 ด้วย Zend Engine ตัวแกนหลักที่ช่วยเพิ่มประสิทธิภาพและความเสถียรให้กับภาษา

ประโยชน์หลักของ PHP

1. การสร้างเว็บไซต์แบบไดนามิก PHP ช่วยให้เว็บไซต์สามารถสร้างหน้าเว็บที่มีเนื้อหาหรือข้อมูลเปลี่ยนแปลงตามผู้ใช้หรือข้อมูลฐานข้อมูลได้อย่างอัตโนมัติ ต่างจากเว็บไซต์แบบสถิติเช่น HTML ธรรมดา

2. ความยืดหยุ่นสูง รองรับการใช้งานร่วมกับเทคโนโลยีเว็บต่าง ๆ เช่น JavaScript, HTML, CSS ได้อย่างราบรื่น

3. เหมาะสำหรับการพัฒนาเว็บแอปพลิเคชัน: รองรับระบบขนาดใหญ่ที่มีการจัดการข้อมูลซับซ้อน เช่น ระบบสมาชิก ระบบอีคอมเมิร์ซ

4. ประสิทธิภาพและการขยายตัว: PHP ได้รับการพัฒนาให้ปรับปรุงประสิทธิภาพอย่างต่อเนื่อง และรองรับการใช้งานบนระบบคลาวด์ ทำให้ขยายตัวได้ง่าย

5. รองรับการเชื่อมต่อฐานข้อมูลหลายชนิด: ทำให้การจัดการข้อมูลและแสดงผลทำได้อย่างหลากหลายและสะดวก

6. มีฟังก์ชันในตัวจำนวนมาก: รองรับงานทั่วไปในเว็บ เช่น การจัดการไฟล์, เซสชัน, คุกกี้, ระบบอีเมล, และประมวลผลภาพ

เหตุผลที่ PHP ได้รับความนิยม

1. เป็นภาษาโอเพนซอร์สและฟรี จึงลดต้นทุนการพัฒนา

2. เรียนรู้และใช้งานง่าย ด้วยไวยากรณ์ที่ไม่ซับซ้อนและเอกสารประกอบที่มากมาย

3. มีความเร็วและประสิทธิภาพสูงโดยเฉพาะเมื่อนำไปรันบนเซิร์ฟเวอร์อย่าง Apache

4. รองรับการใช้งานกับระบบปฏิบัติการและเซิร์ฟเวอร์หลายแพลตฟอร์ม เช่น Windows, Linux, Unix

5. มีฟังก์ชันในตัวและไลบรารีให้เลือกใช้งานมากมาย ช่วยลดเวลาการพัฒนา

6. ง่ายในการจัดการไฟล์, การส่งอีเมล, การประมวลผลภาพ และอื่น ๆ ที่เกี่ยวข้องกับเว็บ

ข้อดีของ PHP

1. การพัฒนาที่รวดเร็วและต่อเนื่อง เหมาะกับโปรเจกต์ที่ต้องการเปิดตัวเร็ว

2. รองรับการทำงานข้ามแพลตฟอร์ม (Windows, Linux, macOS)

3. สนับสนุนการจัดการข้อมูลจำนวนมาก เช่น ในเว็บอีคอมเมิร์ซ หรือเว็บเครือข่ายสังคม

4. การติดตั้งและใช้งานที่ง่ายโดยไม่ยุ่งยาก

5. มีระบบรักษาความปลอดภัยและเครื่องมือช่วยป้องกันการโจมตี

6. สามารถบูรณาการกับบริการและ API ต่าง ๆ ได้รวดเร็ว

7. การเขียนโปรแกรมเชิงวัตถุ (OOP) ช่วยจัดระเบียบและลดความซ้ำซ้อนของโค้ด

ภาษา SQL (Structured Query Language) เป็นภาษามาตรฐานสำหรับการสอบถามและจัดการข้อมูลในระบบฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System: RDBMS). SQL ถูกคิดค้นขึ้นโดย IBM ในปี ค.ศ. 1970 จากแนวคิดของ Dr. Edgar Frank Codd ซึ่งวางรากฐานแบบจำลองฐานข้อมูลเชิงสัมพันธ์ในวารสารของ ACM. พัฒนาโดย Donald D. Chamberlin และ Raymond F. Boyce ภายใต้ชื่อแรก SEQUEL ก่อนจะเปลี่ยนชื่อเป็น SQL ในปี 1976 เนื่องจากชื่อ SEQUEL ไปซ้ำกับเครื่องหมายการค้าของบริษัทอื่น

ประเภทของคำสั่งใน SQL

CREATE DATABASEสร้างฐานข้อมูลใหม่

CREATE TABLEนิยามโครงสร้างข้อมูลในตารางที่สร้างขึ้น

DROP TABLE ลบโครงสร้างตารางข้อมูลออกจากระบบ

ALTER TABLEแก้ไขปรับปรุงโครงสร้างตาราง

SELECTเรียกค้นข้อมูลในตาราง

INSERTเพิ่มแถวข้อมูลลงไปในตาราง

UPDATEปรับปรุงแถวข้อมูลในตาราง

DELETEลบแถวข้อมูลในตาราง

ประโยชน์หลักของ SQL

1. บริหารจัดการข้อมูลอย่างมีประสิทธิภาพ: ช่วยให้การสร้างฐานข้อมูลใหม่ การสืบค้นข้อมูล การปรับปรุงและการลบข้อมูลเกิดขึ้นอย่างสมบูรณ์

2. รองรับฐานข้อมูลขนาดใหญ่และซับซ้อน: SQL สามารถจัดการข้อมูลจำนวนมากในองค์กรและระบบธุรกิจขนาดใหญ่ได้อย่างรวดเร็ว

3. สนับสนุนการพัฒนาระบบค้นหาข้อมูล: SQL เป็นหัวใจสำคัญของระบบค้นหาและการวิเคราะห์ข้อมูลในองค์กรยุคใหม่ ทำให้สามารถดึงและประมวลผลข้อมูลได้ทันที

4. ติดตั้งและใช้งานง่าย: ระบบฐานข้อมูลที่ใช้ SQL มักสามารถติดตั้งและเชื่อมต่อกับระบบงานต่าง ๆ ได้อย่างรวดเร็ว

5. เชื่อมต่อกับภาษาโปรแกรมอื่น: นักพัฒนาสามารถรวม SQL เข้ากับภาษาอย่าง PHP, Java, Python เพื่อพัฒนาเว็บแอปพลิเคชันที่มีระบบจัดการข้อมูลที่มีประสิทธิภาพสูง

ภาษา CSS (Cascading Style Sheet) หรือที่มักเรียกสั้น ๆ ว่า “สไตล์ชีต” คือภาษามาตรฐานทางเว็บที่ใช้ควบคุมและกำหนดรูปแบบการแสดงผลของเอกสาร HTML หรือ XML ทำให้เว็บไซต์มีความสวยงาม สามารถควบคุมสี ขนาด ระยะห่าง ตัวอักษร โครงร่าง และเลย์เอาต์ โดยแยกส่วนรูปแบบ (style) ออกจากเนื้อหา (content) ช่วยให้การจัดการเว็บไซต์สะดวกและมีความสม่ำเสมอ ประวัติและการพัฒนา CSS ถูกสร้างขึ้นเพื่อตอบโจทย์ปัญหาในยุคแรกของเว็บ เมื่อ HTML ยังไม่สามารถกำหนดรูปแบบการแสดงผลได้โดยตรง แนวคิดของ CSS เสนอโดย Håkon Wium Lie ในปี 1994. ภาษา CSS รุ่นแรกเผยแพร่โดย W3C ในปี 1996 และได้รับการกำหนดเป็นมาตรฐานตั้งแต่ HTML 4.0 เป็นต้นมาปัจจุบัน CSS มีหลายเวอร์ชัน (CSS1, CSS2, CSS3+) รองรับความสามารถที่ซับซ้อนมากขึ้น เช่น การแสดงผลแบบ responsive, animationและการออกแบบสำหรับอุปกรณ์หลากหลาย

หลักการทำงาน CSS ใช้ระบบ “Cascade” หรือกระบวนการให้ลำดับความสำคัญกับกฎสไตล์ที่เขียนไว้หลายระดับ เช่น กฎจากภายนอก ไฟล์ (external), ใน <style> ภายในเอกสาร (internal), หรือในบรรทัด (inline) โดยกฎที่มีลำดับความสำคัญมากกว่าจะถูกใช้งาน ส่งผลให้สามารถควบคุมรูปแบบได้อย่างมีประสิทธิภาพและปรับเปลี่ยนง่าย รวมถึงช่วยให้เว็บไซต์ทำงานรวดเร็วขึ้น เพราะลดซ้ำซ้อนของโค้ด HTML

ประโยชน์ของ CSS

1.ยกโครงสร้างเนื้อหาออกจากรูปแบบง่ายต่อการจัดการและอัปเดตทั้งเว็บไซต์

2.ควบคุมรูปแบบเอกสารทุกหน้าได้จาก style sheet ชุดเดียวเพิ่มความสม่ำเสมอและลดข้อผิดพลาดในการออกแบบ

3.ทำให้เว็บดาวน์โหลดเร็วขึ้นเนื่องจากลดขนาดโค้ด HTML

4.รองรับ responsive design ปรับหน้าตาเว็บให้เหมาะสมกับอุปกรณ์ที่แตกต่างกัน เช่น มือถือ แท็บเล็ต คอมพิวเตอร์

5.ตกแต่งเว็บไซต์ได้หลากหลาย เช่น กำหนดสี, ฟอนต์, เงา, เส้นขอบ, ระยะห่าง, การจัดวางองค์ประกอบ

2.1.5 โปรแกรมที่ใช้สร้างหน้าเว็บ Visual Studio Code

Visual Studio Code (VS Code) เป็นแอปพลิเคชันแก้ไขโค้ดแบบโอเพนซอร์สที่พัฒนาโดยไมโครซอฟท์ โดยมีเป้าหมายเพื่อให้เป็นเครื่องมือที่มีประสิทธิภาพสำหรับนักพัฒนาซอฟต์แวร์ ใช้งานได้ง่ายและหลากหลาย VS Code มีคุณสมบัติและฟีเจอร์ที่ทันสมัยและมีประสิทธิภาพสูง โดยเน้นการพัฒนาและการทดสอบซอฟต์แวร์อย่างรวดเร็วและสะดวกสบาย ซึ่งทำให้มีความนิยมในวงกว้างระหว่างนักพัฒนาทั้งมือใหม่และมืออาชีพรองรับระบบปฏิบัติการ macOS, Linux และ Windows สามารถเขียนโค้ดด้วยภาษาต่างๆ ได้

หลากหลาย เช่น JavaScript, Python, C++, HTML, CSS และอีกมากมาย นอกจากนี้ยังมีฟีเจอร์ IntelliSense ที่ช่วยเติมโค้ดอัตโนมัติและแนะนำโค้ด ช่วยให้เขียนโค้ดได้รวดเร็วและถูกต้องมากขึ้น

ลักษณะเด่นของ Visual Studio Code

1.ครอส-แพลตฟอร์ม (Cross-platform) VS Code รองรับทั้ง Windows, macOS และ Linux รวมถึงเวอร์ชันบนเว็บเบราว์เซอร์ด้วย ทำให้นักพัฒนาสามารถเลือกใช้ตามระบบปฏิบัติการที่ตนมีได้อย่างยืดหยุ่น

2.รองรับหลายภาษาโปรแกรมและไฟล์เว็บ (Multi-language & Web support): มีการไฮไลท์ซินแท็กซ์ (syntax highlighting), การเติมโค้ดอัตโนมัติ (IntelliSense) สำหรับ HTML, CSS, JavaScript ฯลฯ โดยตรง

3.เครื่องมือแก้ไขให้ทันสมัย (Modern editing features): เช่น การค้นหา/แทนที่, การทำ refactoring, Snippets, การแก้ไขหลายตำแหน่งพร้อมกัน (multi-cursor)

4.ระบบ Debug และ Integrations ที่แข็งแกร่ง (Built-in Debugging & Integrations): มีเทอร์มินอลในโปรแกรม, รองรับ Git ในตัว และสามารถติดตั้งส่วนขยาย (Extensions) เพิ่มเติมตามความต้องการ

5.ความสามารถในการปรับแต่งสูง (Highly customizable) ผู้ใช้สามารถเปลี่ยนธีม, ปุ่มลัด, ติดตั้งส่วนขยายได้ง่ายจาก VS Code Marketplace เพื่อให้เหมาะกับการพัฒนาของตนเอง

2.1.6 โปรแกรมจัดการฐานข้อมูล PostgreSQL, PostGIS

PostgreSQL เป็นระบบจัดการฐานข้อมูลเชิงวัตถุสัมพันธ์ (Object-Relational Database Management System - ORDBMS) แบบโอเพนซอร์ส ที่มีคุณสมบัติในการจัดการข้อมูลที่มีประสิทธิภาพสูง มีความน่าเชื่อถือในระดับองค์กร (Enterprise-ready) รองรับการทำงานทั้งแบบ SQL มาตรฐาน และการจัดการข้อมูลเชิงสัมพันธ์ (Relational Data) ควบคู่ไปกับการรองรับข้อมูลที่หลากหลาย (เช่น JSONB สำหรับ NoSQL และ PostGIS สำหรับข้อมูลเชิงพื้นที่) ซึ่งเป็นฐานข้อมูลที่ใช้งานในอุตสาหกรรมหลายประเภทตั้งแต่ขนาดเล็กไปจนถึงขนาดใหญ่ PostgreSQL ถูกออกแบบให้เป็นระบบที่มีความเสถียรสูงด้วยการรับประกันความถูกต้องของธุรกรรมตามหลัก ACID และการใช้ Multi-Version Concurrency Control (MVCC) เพื่อเพิ่มประสิทธิภาพในการทำงานพร้อมกันสามารถรองรับการทำงานที่ซับซ้อนและการประมวลผลที่มีประสิทธิภาพ และมีความยืดหยุ่นสูงในการขยายระบบ (Extensibility) ผ่านส่วนขยาย (Extensions) ต่างๆ ประวัติและความเป็นมา โครงการ PostgreSQL เริ่มต้นในปี พ.ศ. 2529 ภายใต้การดูแลของศาสตราจารย์ ไมเคิล สโตนเบรกเกอร์ (Michael Stonebraker) แห่งมหาวิทยาลัยแคลิฟอร์เนีย เบิร์กลีย์ เดิมทีโครงการนี้มีชื่อว่า POSTGRES โดยมุ่งหวังที่จะเพิ่มฟีเจอร์ที่จำเป็นให้น้อยที่สุดเพื่อรองรับประเภทข้อมูลที่หลากหลายได้อย่างสมบูรณ์ ในปี พ.ศ. 2539 โครงการนี้ได้เปลี่ยนชื่อเป็น PostgreSQL เพื่อแสดงให้เห็นถึงการสนับสนุนภาษาคิว

รี SQL (แม้ว่าโดยทั่วไปแล้ว PostgreSQL จะยังคงใช้ชื่อย่อว่า Postgres) ชุมชนผู้มีส่วนร่วมที่ทุ่มเทและหลากหลายอย่าง PostgreSQL Global Development Group ยังคงเผยแพร่โครงการฐานข้อมูลโอเพนซอร์สฟรีนี้อย่างต่อเนื่องทั้งเวอร์ชันหลักและเวอร์ชันรอง

คุณสมบัติหลักของ PostgreSQL

1.เป็นโอเพนซอร์สที่ใช้งานได้ฟรีและทรงพลัง รองรับการใช้ข้อมูลและแบบสอบถามทั้งเชิงสัมพันธ์และ JSON ที่ไม่ใช่เชิงสัมพันธ์

2.มีความน่าเชื่อถือสูง ปฏิบัติตามมาตรฐาน ACID เพื่อให้มั่นใจในความสมบูรณ์และความปลอดภัยของข้อมูล

3.ใช้กันอย่างแพร่หลาย ทั้งแอปพลิเคชันมือถือเว็บ และโซลูชันเชิงพื้นที่หรือวิเคราะห์ข้อมูลเชิงลึก

4.รองรับประเภทข้อมูลซับซ้อนและมีประสิทธิภาพใกล้เคียงกับฐานข้อมูลเชิงพาณิชย์ เช่น Oracle และ SQL Server

5.รองรับฟังก์ชันและกระบวนการจัดเก็บที่เขียนด้วยภาษาต่างๆ เพื่อขยายฟังก์ชันของฐานข้อมูล

ภาพที่ 6 โปรแกรม PostgreSQL

PostGIS คือส่วนขยายฐานข้อมูลเชิงพื้นที่แบบโอเพ่นซอร์สสำหรับระบบฐานข้อมูล PostgreSQL ซึ่งช่วยให้ PostgreSQL สามารถจัดเก็บ สืบค้น และวิเคราะห์ข้อมูลทางภูมิศาสตร์ (Geospatial) ได้ในฐานข้อมูลเชิงสัมพันธ์เดียวกัน โดย PostGIS เพิ่มชนิดข้อมูลเรขาคณิต (geometry) เช่น จุด (point), เส้น (line), รูปหลายเหลี่ยม (polygon) รวมถึงฟังก์ชันสำหรับการวิเคราะห์เชิงพื้นที่ เช่น การสร้างโซนรอบจุด (buffer), การหาจุดตัดกัน (intersection), และการรวมรูปทรง (union) PostGIS เป็นเครื่องมือหลักสำหรับนักพัฒนาแอปพลิเคชันที่จัดการข้อมูลแผนที่หรือระบบตำแหน่งต่างๆ เช่น แอปแสดงแผนที่, บริการระบบระบุตำแหน่ง, และการวิเคราะห์ข้อมูลภูมิศาสตร์เชิงลึกที่ต้องการจัดเก็บและจัดการข้อมูลภูมิสารสนเทศในฐานข้อมูลที่มีประสิทธิภาพสูงนอกจากนี้ PostGIS ยังช่วยให้ PostgreSQL รองรับการทำงานแบบ Spatial

Database พร้อมระบบดัชนีเชิงพื้นที่ที่ช่วยเพิ่มประสิทธิภาพการค้นหาข้อมูลตำแหน่งและความสัมพันธ์เชิงพื้นที่ได้ดีขึ้นรวมถึงสนับสนุนระบบพิกัดอ้างอิง (Coordinate Reference Systems – CRS) เพื่อให้แน่ใจว่าข้อมูลตำแหน่งถูกต้องแม่นยำ

คุณสมบัติหลักของ PostGIS

1. ส่วนขยายสำหรับ PostgreSQL เพื่อรองรับข้อมูลเชิงพื้นที่

2. รองรับชนิดข้อมูลเรขาคณิต (จุด, เส้น, รูปหลายเหลี่ยม)

3. ฟังก์ชันวิเคราะห์เชิงพื้นที่ เช่น buffer, intersection, union

4. การสนับสนุนระบบพิกัดอ้างอิง

ภาพที่ 7 โปรแกรม PostGIS

2.1.7 โปรแกรมที่ใช้ในการคำนวณเส้นทาง pgRouting

pgRouting เป็นเครื่องมือที่เป็นส่วนขยาย (Extension) ที่ทำงานร่วมกับฐานข้อมูลเชิงพื้นที่ PostgreSQL/PostGIS โดยเพิ่มฟังก์ชันการคำนวณระยะทางและการวิเคราะห์โครงข่ายอื่น ๆ เช่น การหาระยะทางที่สั้นที่สุด (Shortest Path) pgRouting ได้พัฒนามาจาก pgDijkstra เขียนโดย Sylvain Pasche จาก camptocamp ต่อมาได้นำไปพัฒนาต่อโดยบริษัท Orkney ประเทศญี่ปุ่น และเปลี่ยนชื่อใหม่เป็น pgRouting อย่างเป็นทางการวัตถุประสงค์หลักของ pgRouting คือจัดหาฟังก์ชันสำหรับการใช้งานใน PostgreSQL/PostGIS เพื่อสร้างเครื่องมือในการคำนวณหาระยะ โดยใช้หลักการของทฤษฎีกราฟ (Graph Theory) ซึ่งมีความคล้ายคลึงกับชุดคำสั่งในบางโปรแกรม เช่น คำสั่งหาระยะทางที่ใกล้ที่สุดในโปรแกรม

ArcGIS Desktop หรือการขอเส้นทางใน Google Maps แต่มีข้อได้เปรียบในการประมวลผลบนเซิร์ฟเวอร์ฐานข้อมูลโดยตรง (Server-side processing) ไม่แค่เฉพาะในเรื่องของระยะทางบนถนนเท่านั้น แต่สามารถใช้ได้กับข้อมูลอะไรก็ได้ที่เกี่ยวข้องกับระยะทาง การสิ้นเปลืองเวลา น้ำมัน เงิน เช่น เส้นทางเกี่ยวกับการเดินเรือ และระบบเน็ตเวิร์คแม่ข่ายอินเทอร์เน็ต รวมถึงการคำนวณพื้นที่บริการ (Service Area) หรือพื้นที่ที่เข้าถึงได้ซึ่งจะช่วยให้สามารถวางแผนการเดินทางส่งสินค้าได้อย่างสะดวกขึ้นอีกด้วย

pgr_dijkstra เป็นฟังก์ชันที่อยู่ใน pgRouting ซึ่งสามารถวิเคราะห์หาเส้นทางที่สั้นที่สุด (Shortest Path) มักจะถูกนำมาใช้ในการคำนวณหาเส้นทาง หลักการคำนวณใช้การคำนวณทางคณิตศาสตร์ของ Dijkstra ซึ่งเป็นการนำเอาทฤษฎีกราฟมาใช้ในการวิเคราะห์โครงข่ายที่จะถูกจำลองด้วยกราฟโดยใช้ จุดต่อ (Nodes) และ เส้น (Edges) แทนถนนที่เชื่อมต่อกัน กราฟถ่วงน้ำหนัก (Weighted Graph) มีการกำหนดระยะทางหรือค่าน้ำหนักระหว่างจุดเป็นตัวเลขเข้าไปในกราฟ โดยเรียกกราฟนั้นว่า กราฟถ่วงน้ำหนัก ซึ่งคือกราฟที่เส้นเชื่อมทุกเส้นมีค่าน้ำหนักที่ค่าเป็นจำนวนจริงที่ไม่ติดลบ

รูปแบบขของโครงสร้างการคำนวณเส้นทางด้วย pgRouting ประกอบไปด้วยคอลัมน์เหล่านี้

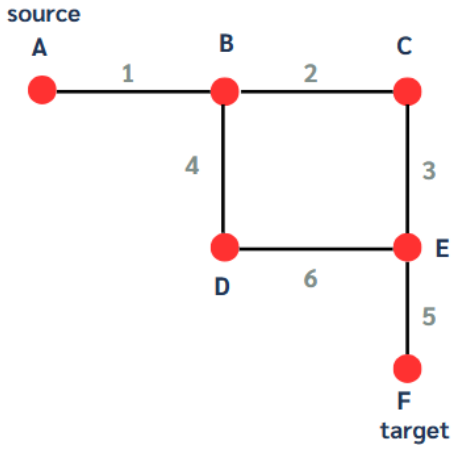
- ID (gid)
- Source (จุดเริ่มต้น)
- Target (จุดสิ้นสุดหรือปลายทาง)
- Length as Cost (ค่าระยะทาง ความเร็ว)

การคำนวณของ อัลกอริทึม Dijkstra จะทำงานเพื่อค้นหาเส้นทางที่มีค่าน้ำหนักรวมน้อยที่สุด (สั้นที่สุด) จากจุดเริ่มต้น (Source) ไปยังจุดหมายปลายทาง (Target)

จากตัวอย่างในภาพ จุดเริ่มต้น(Source) A, จุดหมายปลายทาง (Target) F โดยจะแบ่งเป็น 2 เส้นทาง

เส้นทางที่1 A⟶B⟶C⟶E⟶F เท่ากับ 1+2+3+5 ผลรวม 11 km

เส้นทางที่2 A⟶B⟶D⟶E⟶F เท่ากับ 1+4+6+5 ผลรวม 16 km



ภาพที่ 8 ตัวอย่างการค้นหาแส้นทาง

จากนั้นผลลัพธ์ เส้นทางที่สั้นที่สุด (Shortest Path) คือ เส้นทางที่1 1 (A⟶B⟶C⟶E⟶F) โดยมีค่าน้ำหนักรวมเท่ากับ 11 km

2.1.8. ซอฟต์แวร์ Web GIS / Web Map Application, Apache

Web GIS / Web Map Application Web GIS หมายถึงระบบสารสนเทศภูมิศาสตร์ (Geographic Information System: GIS) ที่ใช้เทคโนโลยีเว็บในการติดต่อสื่อสารระหว่างองค์ประกอบต่าง ๆ ในระบบ เช่น ระบบคลังข้อมูลเชิงพื้นที่, เซิร์ฟเวอร์ GIS, และผู้ใช้ปลายทางผ่านเว็บเบราว์เซอร์หรือแอปพลิเคชัน โดยทั่วไประบบ Web GIS อย่างน้อยต้องประกอบด้วย ลูกข่าย (client) และ เซิร์ฟเวอร์ (server) ที่ทำงานบนระบบปฏิบัติการที่เป็นแอปพลิเคชันเดสก์ท็อปหรือเว็บเบราว์เซอร์ โดยผู้ใช้สามารถสื่อสารผ่านเซิร์ฟเวอร์และเซิร์ฟเวอร์ทำหน้าที่เป็นเว็บเซิร์ฟเวอร์พร้อมกับ Map Application ให้ผู้ใช้สามารถเข้าถึงผ่านเว็บเบราว์เซอร์ได้โดยไม่จำเป็นต้องติดตั้งซอฟต์แวร์เฉพาะลงบนเครื่องของตนเองตัวอย่างการประยุกต์ใช้ Web Map Application ได้แก่ Google Maps API, MapServer

ลักษณะเด่น

1. ผู้ใช้สามารถเข้าถึงระบบผ่านเว็บเบราว์เซอร์ได้โดยตรง (เช่น Internet Explorer, Mozilla Firefox) โดยไม่จำเป็นต้องติดตั้งซอฟต์แวร์เฉพาะบนเครื่องผู้ใช้

2. รองรับแพลตฟอร์มหลากหลาย (Cross-platform) เพราะเว็บเบราว์เซอร์ทำงานบนระบบปฏิบัติการหลายแบบ

3. สามารถเชื่อมต่อกับเครือข่ายอินเทอร์เน็ต ทำให้ผู้ใช้ไม่จำเป็นต้องอยู่ในสถานที่เดียวกับเซิร์ฟเวอร์

4. ลดต้นทุนด้านซอฟต์แวร์และฮาร์ดแวร์ เนื่องจากผู้ใช้ไม่จำเป็นต้องซื้อโปรแกรม GIS เชิงเดสก์ท็อปจำนวนมาก

คุณสมบัติหลัก

1. รองรับการจัดเก็บการเรียกค้นจัดการและวิเคราะห์ข้อมูลเชิงพื้นที่ผ่านเว็บ

2. ระบบมีการแบ่งส่วนงานระหว่าง client และ server โดย client ส่งคำขอ (request) ไปยัง server ผ่าน HTTP และ server แสดงผล (response) กลับมายัง client ในรูปแบบ HTML, JSON, หรือไฟล์ภาพแผนที่ขึ้นอยู่กับคำขอ

3. รองรับการแสดงผลแบบแผนที่แบบโต้ตอบ (interactive map) ที่ผู้ใช้สามารถซูม เลื่อน เลือกชั้นข้อมูล (layers) และแสดงข้อมูลแบบไดนามิก

Apache คือ HTTP Server เป็นเว็บเซิร์ฟเวอร์แบบ Open Source ที่พัฒนามาจาก NCSA HTTPd และได้รับความนิยมอย่างกว้างขวางทั่วโลก ด้วยสถาปัตยกรรมแบบโมดูล (Modular Architecture) และการรองรับการทำงานบนหลากหลายระบบปฏิบัติการ ทำให้มีความยืดหยุ่นสูงในการให้บริการทั้งเนื้อหาแบบคงที่ (Static) และแบบไดนามิก (Dynamic) ผ่านการติดตั้งโมดูลต่าง ๆ (เช่น mod_php, mod_ssl, mod_rewrite, mod_vhost) และสามารถปรับแต่งวิธีการจัดการคำขอพร้อมกัน (Concurrency) ได้อย่างมีประสิทธิภาพผ่าน Multi-Processing Modules (MPMs) เช่น Prefork, Worker และ Event รวมถึงการรองรับการตั้งค่าระดับไดเรกทอรีด้วยไฟล์ .htaccess ซึ่งช่วยเพิ่มความยืดหยุ่นในการจัดการเว็บไซต์ต่าง ๆ แม้จะอยู่บนโฮสติงเดียวกันนอกจากนี้ Apache ยังรองรับฟีเจอร์สำหรับ Virtual Hosts ช่วยให้สามารถโฮสต์หลายเว็บไซต์ในเซิร์ฟเวอร์เดียวได้อย่างสะดวก และมีความปลอดภัยสูงทั้งในเรื่องการรองรับ SSL/TLS การปรับแต่งการเข้าถึงแต่ละโฟลเดอร์หรือซับโดเมนตลอดจนความสามารถในการเชื่อมต่อและทำงานร่วมกับซอฟต์แวร์และภาษาโปรแกรมมิ่งยอดนิยม เช่น PHP, Python, Perl และ Ruby ทำให้เหมาะกับทั้งเว็บไซต์ขนาดเล็กจนถึงเว็บไซต์ระดับองค์กร รวมถึงรองรับการขยายความสามารถได้ผ่านปลั๊กอินและโมดูลเพิ่มเติมที่ชุมชนพัฒนาอย่างต่อเนื่อง

ลักษณะเด่น

1. ฟรีและเปิดซอร์สรหัสต้นฉบับเผยแพร่แก่สาธารณะทำให้ชุมชนสามารถมีส่วนร่วมในการพัฒนาและตรวจสอบความปลอดภัยได้อย่างต่อเนื่อง

2. รองรับหลากหลายแพลตฟอร์ม (cross-platform) ทั้ง Linux, Windows, macOS และระบบยูนิกซ์อื่น ๆ

3. ยืดหยุ่นและขยายได้ด้วยโมดูล (modular architecture) สามารถติดตั้งโมดูลเพิ่มเติมเพื่อรองรับฟังก์ชันต่าง ๆ เช่น PHP, SSL/TLS, การควบคุมสิทธิ์

4. มีประวัติความนิยมสูงและมีเอกสารสนับสนุนจำนวนมากทำให้การตั้งค่าและดูแลระบบเป็นไปได้ง่ายสำหรับทั้งเว็บขนาดเล็กและขนาดใหญ่

คุณสมบัติหลัก ทำหน้าที่รับคำขอ HTTP/HTTPS จากผู้ใช้ (Client) และส่งคืนไฟล์เว็บ (HTML, CSS, JavaScript) หรือแหล่งข้อมูลอื่น ๆ ตามคำขอ รองรับการทำงานร่วมกับฐานข้อมูลและภาษาสคริปต์ฝั่งเซิร์ฟเวอร์ เช่น PHP ผ่านโมดูล mod_php, หรือ Python ผ่าน mod_wsgi และมีระบบความปลอดภัย เช่น การจัดการสิทธิ์เข้าถึง (Access control), การรองรับ TLS/SSL, การใช้ไฟล์ .htaccess/ .htpasswd เพื่อควบคุมการเข้าถึง

2.2 เอกสารงานวิจัยที่เกี่ยวข้อง

อรัญญา จันแดง (2023) **"OnTimeBus: แอปพลิเคชันตรวจสอบตำแหน่งรถบัสตามเวลาจริงและเวลาที่มาถึง"** มีวัตถุประสงค์เพื่อพัฒนาแอปพลิเคชันมือถือ (Mobile Application) โดยใช้ Flutter และภาษา Dart เพื่อระบุตำแหน่งรถไฟฟ้า EV และแสดงปริมาณผู้โดยสารพร้อมทั้ง คำนวณเวลาและระยะทางจากตำแหน่งปัจจุบันของผู้โดยสาร (ใช้ GPS) ไปยังตำแหน่งรถไฟฟ้าที่ใกล้ที่สุด ณ เวลาจริง โดยมีการใช้ซอฟต์แวร์โอเพนซอร์ส เช่น QGIS ในการแปลงข้อมูล (Digitize) , PostgreSQL/PostGIS เป็นฐานข้อมูลเชิงพื้นที่ , และ pgRouting ในการคำนวณระยะทางและเวลา พร้อมทั้งใช้ Visual Studio Code เพื่อสร้าง API ที่ดึงข้อมูลตำแหน่งรถไฟฟ้าปัจจุบันจาก API ภายนอก จากนั้นแอปพลิเคชันจะแสดงผลลัพธ์การคำนวณระยะทางและเวลาที่เหลือในการรอรถ โดยผลการทดลองยืนยันว่าระบบสามารถคำนวณและแสดงข้อมูลเหล่านี้ได้อย่างถูกต้อง เพื่อช่วยให้นักศึกษาและบุคลากรประหยัดเวลาในการรอรถ และสามารถกะเวลาได้อย่างถูกต้อง

ธัญพร ศรีดอกไม้(2010) **“ระบบค้นหาอาคารสถานที่และบุคลากรภายในมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือโดยใช้พื้นฐานทาง GIS”** มีวัตถุประสงค์เพื่อพัฒนากระบวนการเรียกค้นและแสดงผลข้อมูลบุคลากรและสถานที่ผ่านระบบเครือข่าย โดยใช้พื้นฐานของซอฟต์แวร์เปิดรหัส (Open Source Software) โดยเฉพาะการแปลงข้อมูลระบบสารสนเทศภูมิศาสตร์ (GIS) ไปสู่ระบบฐานข้อมูลภายใต้เทคโนโลยี SVG, MySQL, XML, และ PHP ซึ่งมีการแปลงแผนที่ต้นฉบับให้เป็นไฟล์ SVG (Scalable Vector Graphics) โดยใช้ Adobe Illustrator CS เพื่อให้แผนที่สามารถ โต้ตอบ (Interactive) กับผู้ใช้ได้ และใช้ภาษา PHP ในการเขียนคำสั่งเชื่อมโยงกับฐานข้อมูลทำให้ผู้ใช้สามารถคลิกเมาส์จากแผนที่อาคารเพื่อสืบค้นข้อมูล โดยผลการประเมินพบว่าระบบที่พัฒนาขึ้นนี้มีประสิทธิภาพในระดับ ดี และมีความยืดหยุ่นต่อการปรับแก้และใช้งานได้กับเว็บทุก Platform

เนื่องวงศ์ ทวยเจริญ (2018) **“การพัฒนาโปรแกรมประยุกต์บนเว็บและเว็บเซอร์วิสเพื่อบันทึกและค้นหาเส้นทางการขนส่งสาธารณะในกรุงเทพมหานคร” ”** มีวัตถุประสงค์เพื่อพัฒนาโปรแกรมประยุกต์บนเว็บ (Web-based Application) ที่ใช้สำหรับผู้ดูแลระบบ (Admin) ในการตรวจสอบความถูกต้องและแก้ไขหรือเพิ่มเติมข้อมูลเส้นทางและป้ายรถขนส่งสาธารณะ (เช่น รถโดยสารประจำทาง, รถไฟฟ้า) ในกรุงเทพฯ และปริมณฑลได้อย่างสะดวก โดยระบบนี้ถูกพัฒนาขึ้นเพื่อแก้ไขปัญหาจากงานวิจัยก่อนหน้าที่มีข้อจำกัดด้านความถูกต้องของข้อมูลเส้นทางที่อยู่ต่างระนาบ (เช่น ทางด่วน) และไม่สามารถรองรับการเปลี่ยนแปลงข้อมูลได้ ระบบนี้พัฒนาด้วยเทคโนโลยี Open Source ได้แก่ ภาษา PHP, ติดต่อกับฐานข้อมูล MySQL, และใช้ Google Maps API สำหรับแสดงผลแผนที่และให้ผู้ดูแลระบบสามารถลากเส้นเชื่อมระหว่างป้ายรถ เพื่อบันทึกเส้นทางใหม่หรือแก้ไขข้อมูลเส้นทางได้ นอกจากนี้ผู้วิจัยยังได้จัดทำเว็บแอปพลิเคชันสำหรับ

ผู้ใช้ เพื่อให้สามารถค้นหาเส้นทางการเดินทางด้วยระบบขนส่งสาธารณะที่สั้นที่สุดหรือมีการเปลี่ยนรถน้อยที่สุด โดยใช้ขั้นตอนวิธีของ Dijkstra ซึ่งทำงานบนเครื่องแม่ข่าย (Server) และส่งผลลัพธ์ในรูปแบบ JSON จากการทดสอบระบบบนเครื่องแม่ข่ายระบบกลุ่มเมฆสาธารณะ (Public Cloud) พบว่าระบบสามารถแสดงผลเส้นทาง (ร้องขอข้อมูล) ของสายรถ 84 (101 ป้าย) โดยใช้เวลาเฉลี่ย 1,236.08 มิลลิวินาที (ประมาณ 1.2 วินาที) และการบันทึกเส้นทางใหม่หรือแก้ไขข้อมูลใช้เวลาเฉลี่ยเพียง 7.48 มิลลิวินาที (0.00748 วินาที) โดยเวลาตอบกลับเฉลี่ยทั้งระบบน้อยกว่า 4 วินาที ซึ่งเป็นเวลาที่ผู้ใช้เว็บยอมรับได้

สุภัสสรา เมืองนันต์ (2022) **“การพัฒนาระบบฐานข้อมูลและระบบค้นหาเส้นทางในการเดินทางภายใน มหาวิทยาลัยนเรศวร”** มีวัตถุประสงค์เพื่อพัฒนาระบบฐานข้อมูลและระบบค้นหาเส้นทาง (Navigation System) บน Web GIS/Web Map Application เพื่อช่วยให้นิสิตชั้นปีที่ 1 และ 2 ซึ่งไม่คุ้นเคยกับพื้นที่มหาวิทยาลัยสามารถเดินทางไปยังอาคารเรียนและห้องเรียนที่ต้องการได้อย่างสะดวกและรวดเร็ว โดยระบบนี้ใช้ซอฟต์แวร์รหัสเปิด (FOSS4G) เป็นหลักโดยใช้ QGIS ในการนำเข้าและแปลงข้อมูลเชิงพื้นที่ (Digitize) เส้นทางเดินรถ ทางจักรยาน และทางเดินเท้า, ใช้ PostgreSQL/PostGIS ในการจัดเก็บและจัดการข้อมูลเชิงพื้นที่และใช้ pgRouting (ฟังก์ชันเสริม Dijkstra's Algorithm) ในการคำนวณเส้นทางที่สั้นที่สุดและเร็วที่สุดจากจุดเริ่มต้นไปยังจุดหมายปลายทาง (กำหนดเงื่อนไขการเดินทางได้) พร้อมทั้งออกแบบและพัฒนาเว็บไซต์แสดงผลแผนที่ออนไลน์ด้วยภาษา HTML และ JavaScript ซึ่งสามารถเข้าถึงฐานข้อมูลเพื่อค้นหาเส้นทางและแสดงตำแหน่งตึกเรียนได้ผ่าน Browser บนคอมพิวเตอร์และสมาร์ทโฟน

Abdullah Alattas(2022) **“3D pgRouting and Visualization in Cesium JS Using the Integrated Model of LADM and IndoorGML”** มีวัตถุประสงค์เพื่อพัฒนาเว็บแอปพลิเคชัน 3 มิติสำหรับการนำทางภายในอาคาร (Indoor Navigation) โดยอิงตามสิทธิ์การเข้าถึง (Access Rights) ของผู้ใช้ (เช่น นักเรียน, ครู, ผู้มาเยือน) และเวลาที่กำหนดซึ่งระบบนี้สร้างขึ้นเพื่อแก้ไขปัญหาที่แอปพลิเคชันนำทางในอาคารทั่วไปไม่ได้พิจารณาสิทธิ์การเข้าถึงของผู้ใช้การพัฒนาเริ่มต้นจากการนำไฟล์ 3D BIM/IFC ของอาคารเรียน (Faculty of Architecture and the Built Environment, TU Delft) มาจัดทำเป็นฐานข้อมูลที่สอดคล้องกับโมเดล LADM (Land Administration Domain Model) และ IndoorGML ในระบบ PostgreSQL/PostGIS โดยฐานข้อมูลมีการจัดเก็บโหนด (Nodes) และเส้นเชื่อม (Edges) ที่แสดงถึงพื้นที่และทางเชื่อมภายในอาคาร พร้อมทั้งเพิ่มข้อมูลสิทธิ์การเข้าถึงตามความสัมพันธ์ระหว่างผู้ใช้เวลาและพื้นที่การคำนวณเส้นทางที่สั้นที่สุด (Shortest Path) ใช้ส่วนขยาย pgRouting และอัลกอริทึม Dijkstra โดยใช้ Database Views เพื่อเลือกเฉพาะโหนดและเส้นเชื่อมที่ผู้ใช้คนปัจจุบัน (และเวลาที่กำหนด) มีสิทธิ์เข้าถึงได้แบบไดนามิก สุดท้ายใช้แพลตฟอร์มเว็บ 3 มิติ Cesium JS (WebGL) เพื่อสร้าง GUI ฝั่ง Client ที่สามารถให้ผู้ใช้ระบุจุดเริ่มต้นและจุดหมายปลายทาง ประเภทผู้ใช้ และเวลาเพื่อแสดงผลเส้นทางนำทางที่ถูกต้องตามสิทธิ์เป็นกราฟิก 3 มิติที่สามารถแสดงบนเดสก์ท็อป, แล็ปท็อป, แท็บเล็ต และโทรศัพท์มือถือได้

Taranjot Singh Bhatia, Harpinder Singh, P.K Litoria, Brijendra Pateriya(2018) **“Web GIS Development using Open Source Leaflet and Geoserver Toolkit”** มีวัตถุประสงค์เพื่อนำเสนอระเบียบวิธีที่ง่ายและรวดเร็วในการสร้างแอปพลิเคชัน Web GIS โดยใช้ชุดเครื่องมือ Open Source (FOSS) เพื่อให้ผู้คนจำนวนมากสามารถเข้าถึงข้อมูลเชิงพื้นที่ได้อย่างง่ายดาย โดยใช้สถาปัตยกรรม 3 ชั้น PostgreSQL/PostGIS เป็นฐานข้อมูลแบ็กเอนด์สำหรับจัดเก็บข้อมูลภูมิสารสนเทศ , GeoServer เป็นซอฟต์แวร์กลาง (Middleware) ทำหน้าที่เป็นเซิร์ฟเวอร์ในการสร้างบริการข้อมูล (Services) และแสดงสัญลักษณ์ตามมาตรฐานเปิด (Open Standards) เช่น WMS และ WFS , และ Leaflet ซึ่งเป็นไลบรารี JavaScript ที่มีน้ำหนักเบาและเป็นมิตรกับมือถือ (Mobile-friendly) ใช้ในการสร้าง GUI (Graphical User Interface) และแผนที่แบบโต้ตอบ งานวิจัยนี้เริ่มต้นด้วยการสร้างแอปพลิเคชันพื้นฐานโดยใช้ปลั๊กอิน qgis2web ของ QGIS และสาธิตการปรับแต่ง (Customization) แอปพลิเคชันด้วยปลั๊กอินของ Leaflet เพิ่มเติม (เช่น Mouse Position, Zoom/Home, Minimap, Panning, Attribution และ Info tool) โดยสรุปว่าสามารถสร้างแอปพลิเคชัน Web GIS ที่มีประสิทธิภาพสูงได้ภายในเวลาอันสั้นแม้สำหรับผู้ที่มีทักษะการเขียนโปรแกรมน้อยหรือไม่มีเลย และแนะนำว่าหากปริมาณข้อมูลน้อยสามารถใช้ไฟล์ JSON แทนฐานข้อมูล PostgreSQL เพื่อให้ตั้งค่าแอปพลิเคชันได้เร็วและง่ายยิ่งขึ้น

สรุปเอกสารงานวิจัยที่เกี่ยวข้อง

งานวิจัยที่ 1 พัฒนาแอปพลิเคชันสำหรับติดตามตำแหน่งและเวลาที่มาถึงของรถไฟฟ้าในมหาวิทยาลัยนเรศวร โดยใช้เทคโนโลยี GIS (QGIS) ระบบฐานข้อมูล PostgreSQL/PostGIS, และเครื่องมือพัฒนาแอป Flutter แอปพลิเคชันช่วยให้ผู้โดยสารทราบตำแหน่งปัจจุบันของรถไฟฟ้าและเวลาที่จะมาถึงสถานีที่รอจึงช่วยลดเวลาการรอและเพิ่มประสิทธิภาพในการเดินทางในมหาวิทยาลัย

งานวิจัยที่ 2 พัฒนาระบบสืบค้นข้อมูลบุคลากรและสถานที่ภายในมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ โดยใช้พื้นฐานของ GIS และซอฟต์แวร์โอเพนซอร์ส ระบบสามารถสนับสนุนการสืบค้นและแสดงผลข้อมูลแผนที่ได้อย่างมีประสิทธิภาพแผนที่ที่พัฒนาขึ้นมีความยืดหยุ่นในการปรับแก้และสามารถแสดงรายละเอียดของสถานที่เมื่อนำเมาส์ไปวางได้

งานวิจัยที่ 3 พัฒนาโปรแกรมประยุกต์บนเว็บสำหรับผู้ดูแลระบบ เพื่อให้สามารถเพิ่ม แก้ไข และปรับปรุงข้อมูลเส้นทางการขนส่งสาธารณะในกรุงเทพฯ และปริมณฑลได้อย่างสะดวกรวมถึงการพัฒนาระบบที่สามารถค้นหาเส้นทางที่สั้นที่สุดหรือเปลี่ยนรถน้อยที่สุด โดยใช้ข้อมูลจาก Google Map และขั้นตอนวิธีของ Dijkstra ในการคำนวณระบบที่พัฒนาขึ้นสามารถจัดการรูปภาพแผนที่และค้นหาเส้นทางที่สั้นที่สุดได้โดยใช้เวลาเฉลี่ยน้อยกว่า 4 วินาที

งานวิจัยที่ 4 พัฒนาระบบฐานข้อมูลและระบบค้นหาเส้นทางภายในมหาวิทยาลัยนเรศวร เพื่อแก้ไขปัญหานิสิตขับรถหลงทางและเข้าเรียนสาย ระบบนี้ใช้ซอฟต์แวร์รหัสเปิด (Open Source) เช่น PostgreSQL/PostGIS, QGIS และ pgRouting ในการพัฒนา ผลการวิจัยได้ฐานข้อมูลและระบบค้นหาเส้นทางที่ช่วยให้นิสิตเดินทางไปยังอาคารเรียนต่าง ๆ ได้สะดวกและรวดเร็วขึ้น

งานวิจัยที่ 5 งานวิจัยนี้ได้พัฒนาแอปพลิเคชันเว็บ 3 มิติสำหรับนำทางภายในอาคาร โดยรวมสิทธิ์การเข้าถึงของผู้ใช้เข้าไว้ด้วย ทำให้เส้นทางการนำทางเป็นไปตามสิทธิ์การเข้าถึงเฉพาะของแต่ละบุคคล ซึ่งเป็นการแก้ปัญหาที่แอปพลิเคชันนำทางในอาคารที่มีอยู่ไม่ได้พิจารณาสิทธิ์การเข้าถึงของผู้ใช้

งานวิจัยที่ 6 นำเสนอวิธีที่ง่ายและรวดเร็วในการสร้างแอปพลิเคชัน Web GIS โดยใช้สถาปัตยกรรม 3 ชั้นของเครื่องมือ Open Source (FOSS) ได้แก่ PostgreSQL/PostGIS (ฐานข้อมูล), GeoServer (ซอฟต์แวร์กลางสร้างบริการข้อมูล WMS/WFS) และ Leaflet (ไลบรารี JavaScript น้ำหนักเบาสำหรับสร้าง GUI แผนที่โต้ตอบ) โดยสรุปว่าสามารถสร้างแอปพลิเคชัน Web GIS ที่มีประสิทธิภาพสูงได้ภายในเวลาอันสั้นแม้สำหรับผู้มีทักษะการเขียนโปรแกรมน้อย

บทที่ 3

วิธีดำเนินงานวิจัย

ในการศึกษาเรื่อง SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวรด้วย pgRouting บน Web Application โดยมีรายละเอียด 2 ขั้นตอนดังนี้

1.พื้นที่ศึกษา

2.วิธีการดำเนินงาน

3.1 พื้นที่ศึกษา

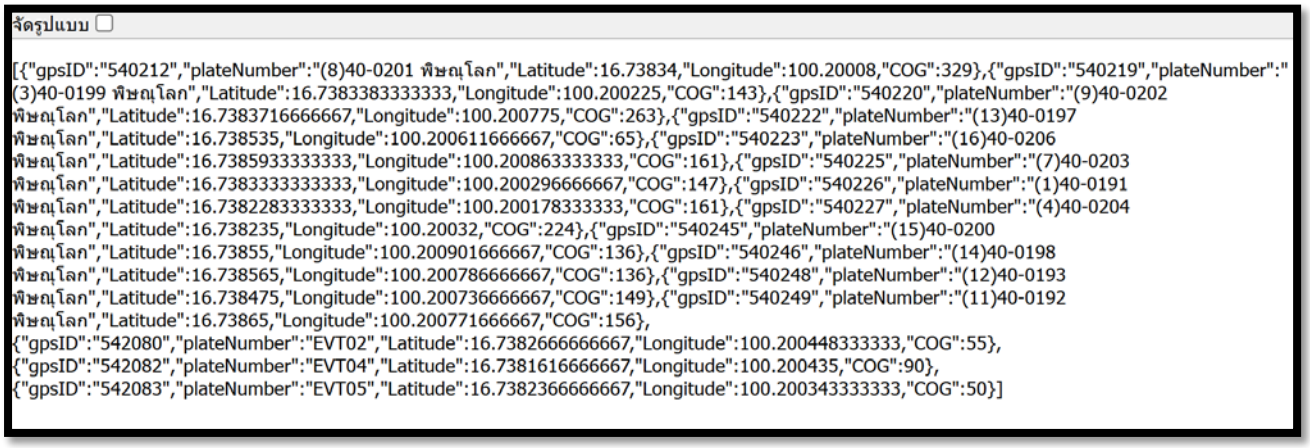
3.1.1 พื้นที่ศึกษา

มหาวิทยาลัยนเรศวร ตำบลท่าโพธิ์ อำเภอเมืองพิษณุโลก จังหวัดพิษณุโลก

3.2 วิธีการดำเนินงาน

3.2.1 เก็บและรวบรวมข้อมูล

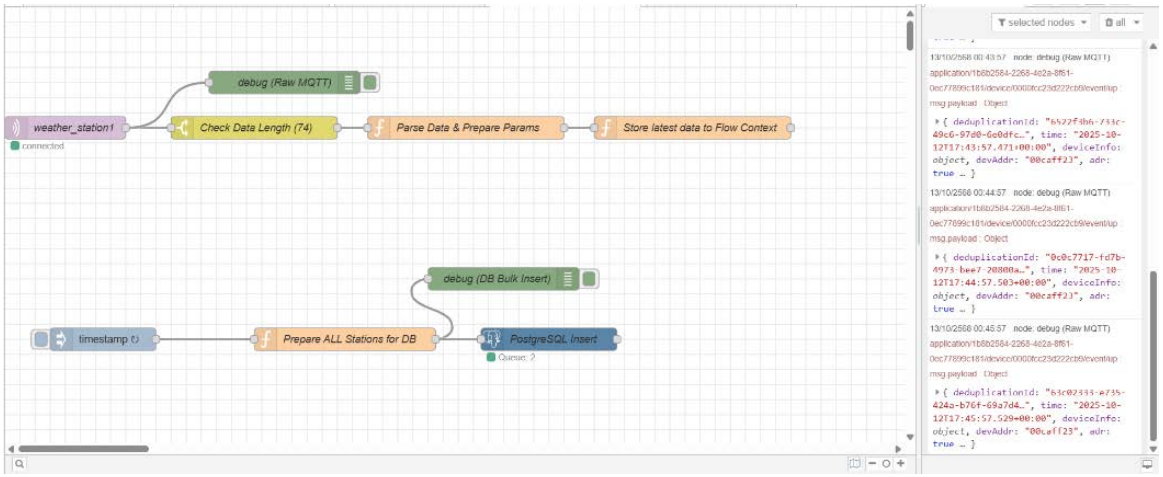
เก็บข้อมูลพิกัดสถานที่จาก GeoJSON เช่น คณะ, สำนักงาน, ป้ายรถเมล์สายสีเหลืองและป้ายรถเมล์สายสีแดงภายในมหาวิทยาลัยนเรศวรและข้อมูลตำแหน่งรถโดยสาร ใช้ API เพื่อดึงตำแหน่งรถโดยสาร(รถส้ม)แบบเรียลไทม์



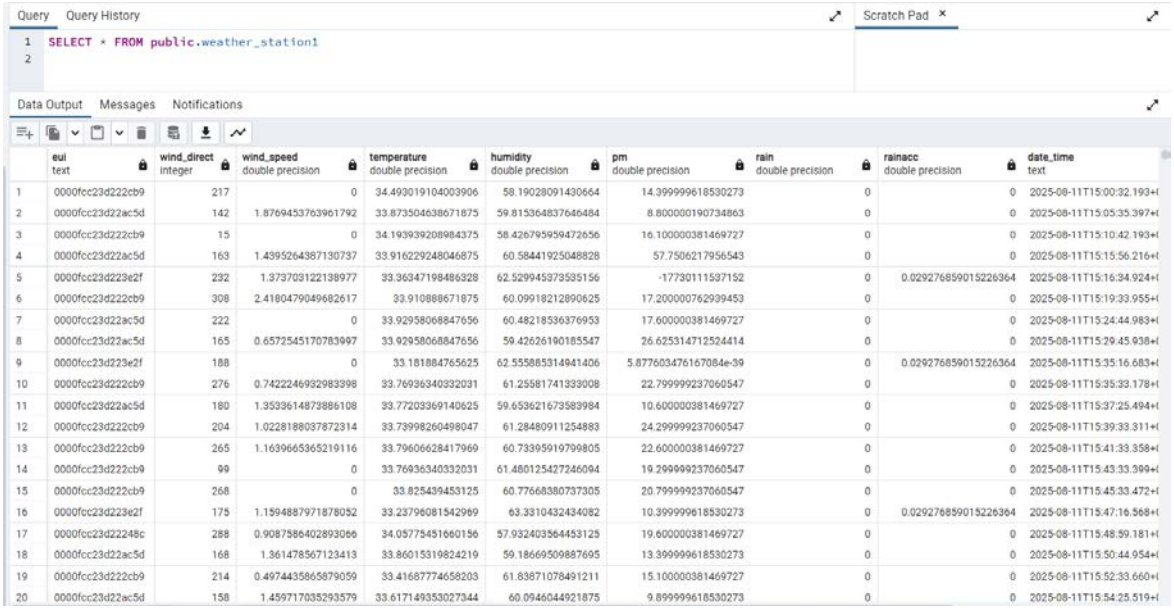
ภาพที่ 9 API ตำแหน่งรถโดยสาร

3.2.2 จัดการฐานข้อมูลสภาพอากาศ

ดึงข้อมูลจากสถานีตรวจวัดอากาศจากเซนเซอร์ทั้ง10สถานีภายในมหาวิทยาลัยนเรศวรและทำการนำเข้า Node-RED ในข้อมูลจากสถานีตรวจวัดอากาศจะมี ค่าฝุ่น (PM2.5), อุณหภูมิ, ความชื้น, ความเร็วลม, ปริมาณฝน, วันและเวลาแบบเรียลไทม์ รายชื่อสถานีตรวจวัดอากาศได้แก่ คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อม, แปลงเกษตร, อาคาร KNECC, สนามฟุตบอล, สนามฟุตซอล, คณะวิทยาศาสตร์, อ่างเก็บน้ำหลังหอใน, โรงเรียนอนุบาลและประถมสาธิต, คณะพยาบาลศาสตร์, ลานสมเด็จพระนเรศวรมหาราช จากนั้นนำข้อมูลสภาพอากาศแบบเรียลไทม์จัดเก็บไว้ในฐานข้อมูล PostgreSQL ทำการเชื่อม Node-RED กับ PostgreSQL



ภาพที่ 10 ข้อมูลสภาพอากาศใน Node-RED



	eui text	wind_direct integer	wind_speed double precision	temperature double precision	humidity double precision	pm double precision	rain double precision	rainacc double precision	date_time text
1	0000fcc23d222cb9	217	0	34.493019104003906	58.19028091430664	14.399999618530273	0	0	2025-08-11T15:00:32.193+(
2	0000fcc23d22ac5d	142	1.8769453763961792	33.873504638671875	59.815364837646484	8.800000190734863	0	0	2025-08-11T15:05:35.397+(
3	0000fcc23d222cb9	15	0	34.193939208984375	58.426795959472656	16.100000381469727	0	0	2025-08-11T15:10:42.193+(
4	0000fcc23d22ac5d	163	1.4895264387130737	33.916229248046875	60.58441925048828	57.7506217956543	0	0	2025-08-11T15:15:56.216+(
5	0000fcc23d223e2f	232	1.373703122138977	33.36347198486328	62.529945373535156	-177301111537152	0	0.029276859015226364	2025-08-11T15:16:34.924+(
6	0000fcc23d222cb9	308	2.4180479049682617	33.910888671875	60.09918212890625	17.200000762939453	0	0	2025-08-11T15:19:33.955+(
7	0000fcc23d22ac5d	222	0	33.92958068847656	60.48218536376953	17.600000381469727	0	0	2025-08-11T15:24:44.983+(
8	0000fcc23d22ac5d	165	0.6572545170783997	33.92958068847656	59.42626190185547	26.625314712524414	0	0	2025-08-11T15:29:45.938+(
9	0000fcc23d223e2f	188	0	33.181884765625	62.555885314941406	5.877603476167084e-39	0	0.029276859015226364	2025-08-11T15:35:16.683+(
10	0000fcc23d222cb9	276	0.7422246932983398	33.76936340332031	61.25581741333008	22.799999237060547	0	0	2025-08-11T15:35:33.178+(
11	0000fcc23d22ac5d	180	1.3539614873886108	33.77203369140625	59.653621673583984	10.600000381469727	0	0	2025-08-11T15:37:25.494+(
12	0000fcc23d222cb9	204	1.0228188037872314	33.73998260498047	61.28480911254883	24.299999237060547	0	0	2025-08-11T15:39:33.311+(
13	0000fcc23d222cb9	265	1.1639665365219116	33.79606628417969	60.73395919799805	22.600000381469727	0	0	2025-08-11T15:41:33.358+(
14	0000fcc23d222cb9	99	0	33.76936340332031	61.480125427246094	19.299999237060547	0	0	2025-08-11T15:43:33.399+(
15	0000fcc23d222cb9	268	0	33.825439453125	60.77668380737305	20.799999237060547	0	0	2025-08-11T15:45:33.472+(
16	0000fcc23d223e2f	175	1.1594887971878052	33.23796081542969	63.3310432434082	10.399999618530273	0	0.029276859015226364	2025-08-11T15:47:16.568+(
17	0000fcc23d222148c	288	0.9087586402893066	34.05775451660156	57.932403564453125	19.600000381469727	0	0	2025-08-11T15:48:59.181+(
18	0000fcc23d22ac5d	168	1.361478567123413	33.86015319824219	59.18669509887695	13.399999618530273	0	0	2025-08-11T15:50:44.954+(
19	0000fcc23d222cb9	214	0.4974435865879059	33.41687774658203	61.63871078491211	15.100000381469727	0	0	2025-08-11T15:52:33.660+(
20	0000fcc23d22ac5d	158	1.4597170352935791	33.617149353027344	60.0946044921875	9.899999618530273	0	0	2025-08-11T15:54:25.519+(

ภาพที่ 11 ข้อมูลสภาพอากาศที่ถูกจัดเก็บในฐานข้อมูล PostgreSQL

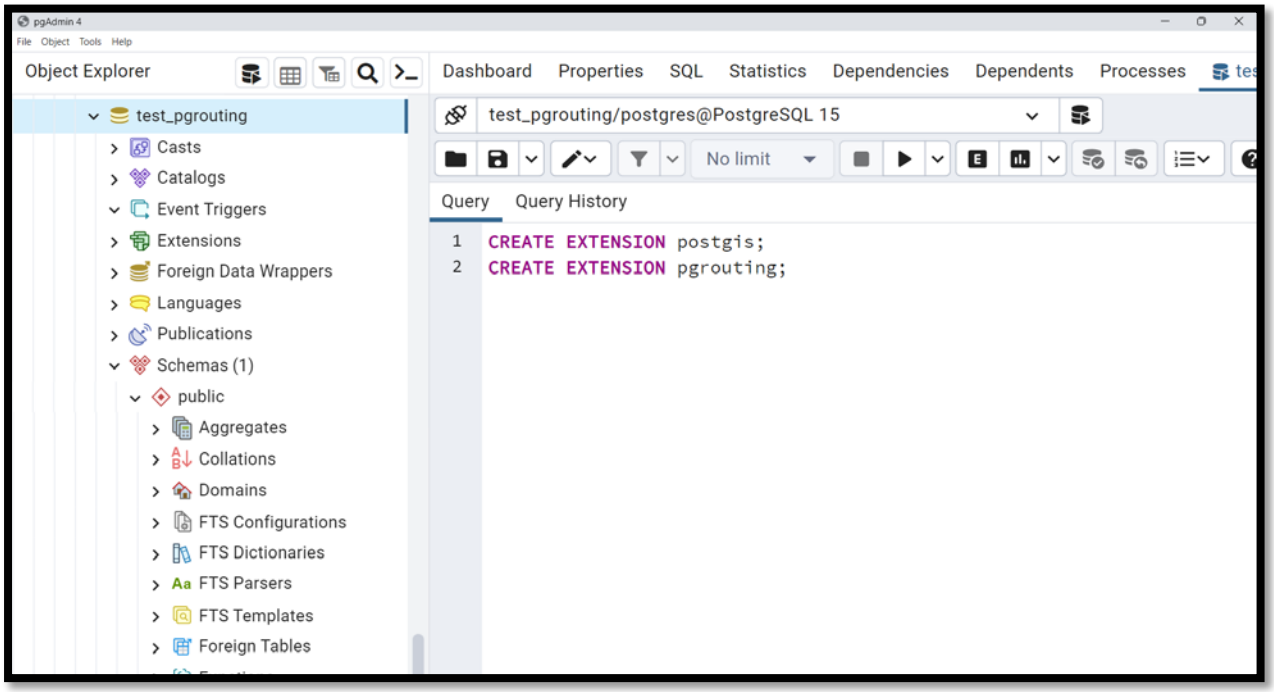
3.2.3 เตรียมฐานข้อมูลและนำเข้าข้อมูลเครือข่ายถนน

การดาวน์โหลดข้อมูลเครือข่ายถนนจาก OpenStreetMap (OSM) ซึ่งเป็นฐานข้อมูลแผนที่โลกแบบเปิด (Open Geospatial Database) ที่อนุญาตให้ผู้ใช้สามารถสร้าง แก้ไข และแบ่งปันข้อมูลภูมิสารสนเทศได้อย่างเสรี ข้อมูลเครือข่ายถนนที่ใช้สำหรับการคำนวณเส้นทางภายในมหาวิทยาลัยนเรศวรถูกดาวน์โหลดจากฐานข้อมูล และจัดเก็บในรูปแบบไฟล์ .osm

หลังจากได้ไฟล์ข้อมูลเครือข่ายถนนแล้ว ขั้นตอนต่อไปคือการสร้างฐานข้อมูลใน PostgreSQL ตั้งชื่อสร้างข้อมูลในPostgreSQ ด้วยคำสั่ง CREATE DATABASE test_pgrouting; เพื่อรองรับข้อมูลเชิงพื้นที่และฟังก์ชันการคำนวณเส้นทาง จากนั้นจึงทำการโหลดส่วนขยาย (Extension) ที่จำเป็นเข้าสู่ฐานข้อมูลด้วยคำสั่ง CREATE EXTENSION เพื่อเพิ่มขีดความสามารถในการจัดการข้อมูลเชิงพื้นที่

1. CREATE EXTENSION postgis; โหลดส่วนขยาย PostGIS เข้าสู่ฐานข้อมูล ซึ่งเพิ่มความสามารถให้ PostgreSQL ในการจัดเก็บ, สืบค้น และจัดการ ข้อมูลเชิงพื้นที่ในรูปแบบของวัตถุทางภูมิศาสตร์ เช่น จุด (Point), เส้น (Line) และรูปหลายเหลี่ยม (Polygon)

2. CREATE EXTENSION pgrouting; โหลดส่วนขยาย pgRouting เข้าสู่ฐานข้อมูลเป็นเครื่องมือที่ทำงานร่วมกับ PostGIS โดยเฉพาะเพื่อเพิ่มฟังก์ชันการคำนวณระยะทางและการวิเคราะห์โครงข่ายถนน ซึ่งใช้หลักการของทฤษฎีกราฟ (Graph Theory) เพื่อหาระยะทางที่สั้นที่สุด (Shortest Path)



ภาพที่ 12 การใช้คำสั่ง CREATE EXTENSION

การนำเข้าข้อมูลเครือข่ายถนนจาก OpenStreetMap ระบบใช้ข้อมูลเครือข่ายถนนจาก OpenStreetMap (OSM) ซึ่งเป็นฐานข้อมูลแผนที่โลกแบบเปิด เพื่อใช้เป็นฐานข้อมูลเส้นทางภายในมหาวิทยาลัยนเรศวร ใช้โปรแกรม osm2pgrouting.exe ในการแปลงและนำเข้าไฟล์ข้อมูลเครือข่ายถนน (.osm)

ขั้นตอนการรันคำสั่ง ผู้วิจัยได้นำไฟล์เครือข่ายถนน (nu.osm) ไปไว้ในไดเรกทอรี Program Files\PostgreSQL\15\bin บนเครื่องเซิร์ฟเวอร์ หลังจากนั้นพิมพ์คำว่า cmd ในช่องของ Program Files\PostgreSQL\15\bin Command Prompt ของ PostgresSQL ก็จะปรากฏขึ้น จากนั้นรัน osm2pgrouting คำสั่งที่ใช้ osm2pgrouting.exe -f nu.osm --conf mapconfig_for_cars.xml -d test_pgrouting -U postgres -h localhost -p 5432 -W postgres –clean

ภาพที่ 13 รันคำสั่ง Command Prompt

ตารางที่ 4 คำสั่งที่ใช้ osm2pgrouting

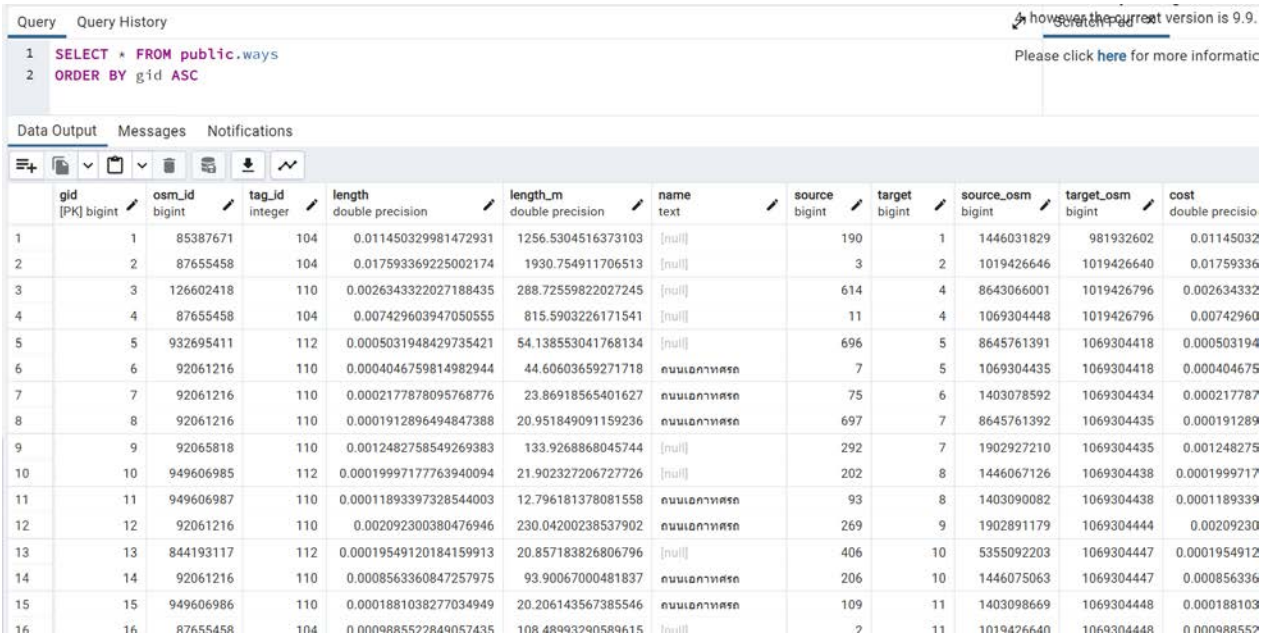
ส่วนของคำสั่ง	ความหมาย	บทบาทในระบบ SMART NU
osm2pgrouting.exe	สำหรับแปลงข้อมูล OpenStreetMap	เป็นเครื่องมือหลักในการนำเข้าไฟล์ OSM เข้าสู่ฐานข้อมูล PostgreSQL/PostGIS
-f nu.osm	ไฟล์ข้อมูลนำเข้า	ระบุว่าใช้ไฟล์ nu.osm เป็นข้อมูลเครือข่ายถนนภายในมหาวิทยาลัยนเรศวร
--conf mapconfig_for_cars.xml	ไฟล์การตั้งค่า	กำหนดกฎเกณฑ์การนำเข้า, สร้างตาราง,ประเภทเส้นทาง
-d test_pgrouting	ชื่อฐานข้อมูล	เชื่อมต่อและทำงานกับฐานข้อมูลชื่อ test_pgrouting
-U postgres	ชื่อผู้ใช้ฐานข้อมูล	ใช้ postgres ในการล็อกอิน
-h localhost	โฮสต์ของฐานข้อมูล	ฐานข้อมูลรันอยู่บนเครื่องโลคอลโฮสต์
-p 5432	พอร์ตของฐานข้อมูล	ใช้พอร์ตมาตรฐานของ PostgreSQL
-W postgres	รหัสผ่านผู้ใช้	ใช้รหัสผ่าน postgres
--clean	ล้างข้อมูลเดิม	คำสั่งนี้จะลบตารางและโครงสร้างข้อมูลที่เกี่ยวข้องกับ pgRouting ออกจากฐานข้อมูล test_pgrouting ก่อนที่จะสร้างใหม่

ผลลัพธ์การสร้างตารางโครงข่ายถนนภายในมหาวิทยาลัยนเรศวร หลังจากรันคำสั่ง osm2pgrouting ข้อมูลเครือข่ายถนนจะถูกจัดเก็บในฐานข้อมูล PostgreSQL โดยโปรแกรมจะสร้างตารางสำคัญ 2 ตาราง สำหรับการคำนวณเส้นทางด้วย pgRouting

1. ตาราง ways: ตารางนี้เก็บส่วนของถนนแต่ละส่วน (Edges) ซึ่งเป็นข้อมูลเส้นทาง โดยมีคอลัมน์สำคัญ เช่น gid (รหัสเฉพาะ), source (โหนดเริ่มต้น), target (โหนดปลายทาง) และ cost (ต้นทุนในการเดินทาง ซึ่งคือความยาวของส่วนถนนนั้น)

2. ตาราง ways_vertices_pgr: ตารางนี้ทำหน้าที่เก็บ จุดเชื่อมต่อ (Nodes) ของถนน ซึ่งเป็นจุดที่ถนนตั้งแต่สองสายขึ้นไปมาบรรจบกันตารางนี้ถูกใช้เพื่อระบุจุดเริ่มต้น และ จุดสิ้นสุดของการคำนวณเส้นทางด้วย pgRouting

ตาราง ways



Query Query History

```
SELECT * FROM public.ways
ORDER BY gid ASC
```

Data Output Messages Notifications

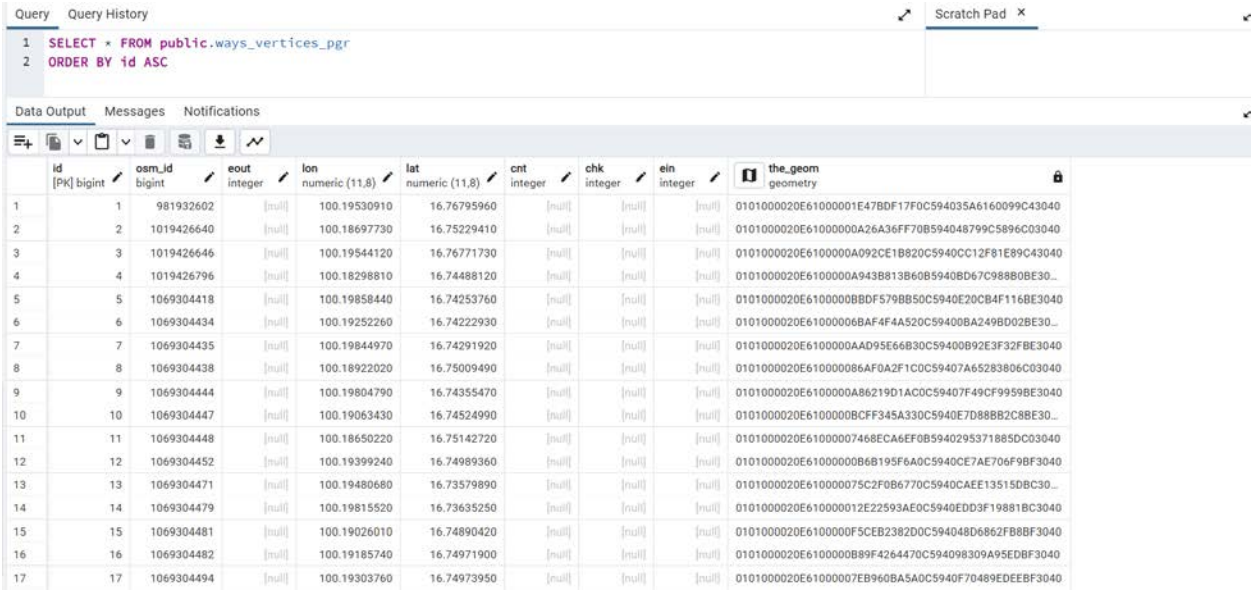
	gid [PK] bigint	osm_id bigint	tag_id integer	length double precision	length_m double precision	name text	source bigint	target bigint	source_osm bigint	target_osm bigint	cost double precisio
1	1	85387671	104	0.011450329981472931	1256.5304516373103	[null]	190	1	1446031829	981932602	0.01145032
2	2	87655458	104	0.017593369225002174	1930.754911706513	[null]	3	2	1019426646	1019426640	0.01759336
3	3	126602418	110	0.0026343322027188435	288.72559822027245	[null]	614	4	8643066001	1019426796	0.002634332
4	4	87655458	104	0.007429603947050555	815.5903226171541	[null]	11	4	1069304448	1019426796	0.00742960
5	5	932695411	112	0.0005031948429735421	54.138553041768134	[null]	696	5	8645761391	1069304418	0.000503194
6	6	92061216	110	0.0004046759814982944	44.60603659271718	ถนนเอกาทศรถ	7	5	1069304435	1069304418	0.000404675
7	7	92061216	110	0.00021778780957687776	23.86918565401627	ถนนเอกาทศรถ	75	6	1403078592	1069304434	0.000217787
8	8	92061216	110	0.0001912896494847388	20.951849091159236	ถนนเอกาทศรถ	697	7	8645761392	1069304435	0.000191289
9	9	92065818	110	0.0012482758549269383	133.9268868045744	[null]	292	7	1902927210	1069304435	0.001248275
10	10	949606985	112	0.0001999717763940094	21.902327206727726	[null]	202	8	1446067126	1069304438	0.0001999717
11	11	949606987	110	0.0001189397328544003	12.796181378081558	ถนนเอกาทศรถ	93	8	1403090082	1069304438	0.0001189339
12	12	92061216	110	0.002092300380476946	230.04200238537902	ถนนเอกาทศรถ	269	9	1902891179	1069304444	0.00209230
13	13	844193117	112	0.0001954912018415991 3	20.857183826806796	[null]	406	10	5355092203	1069304447	0.0001954912
14	14	92061216	110	0.0008563360847257975	93.90067000481837	ถนนเอกาทศรถ	206	10	1446075063	1069304447	0.000856336
15	15	949606986	110	0.0001881038277034949	20.206143567385546	ถนนเอกาทศรถ	109	11	1403098669	1069304448	0.000188103
16	16	87655458	104	0.0009885522849057435	108.489932905896 15	[null]	2	11	1019426640	1069304448	0.000988552

ภาพที่ 14 ways ในฐานข้อมูล PostgreSQL

ตารางที่ 5 ข้อมูลในตาราง ways

คอลัมน์ (Column)	ประเภทข้อมูล (Type)	คำอธิบาย (Description)
gid	bigint	รหัสเฉพาะของส่วนถนน (Primary Key)
osm_id	bigint	รหัสอ้างอิงจาก OpenStreetMap (OSM)
tag_id	integer	รหัสสำหรับเชื่อมโยงกับตาราง configuration เพื่อดึงค่าคุณสมบัติ (เช่น ความเร็วสูงสุด)
the_geom	Geometry (LineString)	ข้อมูลรูปทรงเรขาคณิต (Geometry) ของส่วนถนนนั้น
length	double precision	ความยาวของส่วนถนนนั้น
Source	bigint	โหนดเริ่มต้น ของส่วนถนนนี้ (เชื่อมโยงกับตาราง ways_vertices_pgr.id)
target	bigint	โหนดปลายทาง ของส่วนถนนนี้ (เชื่อมโยงกับตาราง ways_vertices_pgr.id)
cost	double precision	ต้นทุน ในการเดินทางจาก source ไป target

ตาราง ways_vertices_pgr



```
SELECT * FROM public.ways_vertices_pgr
ORDER BY id ASC
```

id [PK] bigint	osm_id bigint	eout integer	lon numeric (11,8)	lat numeric (11,8)	cnt integer	chk integer	ein integer	the_geom geometry
1	981932602	[null]	100.19530910	16.76795960	[null]	[null]	[null]	0101000020E61000001E47BDF17F0C594035A6160099C43040
2	1019426640	[null]	100.18697730	16.75229410	[null]	[null]	[null]	0101000020E61000000A26A36FF70B594048799C5896C03040
3	1019426646	[null]	100.19544120	16.76771730	[null]	[null]	[null]	0101000020E6100000A092CE1B820C5940CC12F81E89C43040
4	1019426796	[null]	100.18298810	16.74488120	[null]	[null]	[null]	0101000020E6100000A943B813B60B5940BD67C988B0BE30...
5	1069304418	[null]	100.19858440	16.74253760	[null]	[null]	[null]	0101000020E6100000BBDF579BB50C5940E20CB4F116BE3040
6	1069304434	[null]	100.19252260	16.74222930	[null]	[null]	[null]	0101000020E61000006BAF4F4A520C59400BA249BD02BE30...
7	1069304435	[null]	100.19844970	16.74291920	[null]	[null]	[null]	0101000020E6100000AAD95E66B30C59400B92E3F32FBE3040
8	1069304438	[null]	100.18922020	16.75009490	[null]	[null]	[null]	0101000020E61000086AF0A2F1C0C59407A65283806C03040
9	1069304444	[null]	100.19804790	16.74355470	[null]	[null]	[null]	0101000020E6100000AB6219D1AC0C59407F49CF9959BE3040
10	1069304447	[null]	100.19063430	16.74524990	[null]	[null]	[null]	0101000020E6100000BCFF345A330C5940E7D88BB2C8BE30...
11	1069304448	[null]	100.18650220	16.75142720	[null]	[null]	[null]	0101000020E61000007468ECA6EF0B5940295371885DC03040
12	1069304452	[null]	100.19399240	16.74989360	[null]	[null]	[null]	0101000020E61000000B6B195F6A0C5940CE7AE706F9BF3040
13	1069304471	[null]	100.19480680	16.73579890	[null]	[null]	[null]	0101000020E61000075C2F0B6770C5940CAEE13515DBC30...
14	1069304479	[null]	100.19815520	16.73635250	[null]	[null]	[null]	0101000020E61000012E22593AE0C5940EDD3F19881BC3040
15	1069304481	[null]	100.19026010	16.74890420	[null]	[null]	[null]	0101000020E6100000F5CEB2382D0C594048D6862FB8BF3040
16	1069304482	[null]	100.19185740	16.74971900	[null]	[null]	[null]	0101000020E6100000B89F4264470C594098309A95EDBF3040
17	1069304494	[null]	100.19303760	16.74973950	[null]	[null]	[null]	0101000020E61000007EB960BA5A0C5940F70489EDEEBF3040

ภาพที่ 15 ways_vertices_pgr ในฐานข้อมูล PostgreSQL

ตารางที่ 6 ข้อมูลในตาราง ways_vertices_pgr

คอลัมน์ (Column)	**ประเภทข้อมูล (Type)**	**คำอธิบาย (Description)**
id	bigint	รหัสเฉพาะของโหนด (Primary Key) ใช้ในการอ้างอิงจากตาราง ways
osm_id	bigint	รหัสโหนดดั้งเดิมจาก OpenStreetMap (OSM)
the_geom	Geometry (Point)	ข้อมูลรูปทรงเรขาคณิต (Geometry) ที่เป็น จุด ของโหนดนั้น
lon	numeric	ค่าลองจิจูด (Longitude) ของโหนด
lat	numeric	ค่าละติจูด (Latitude) ของโหนด

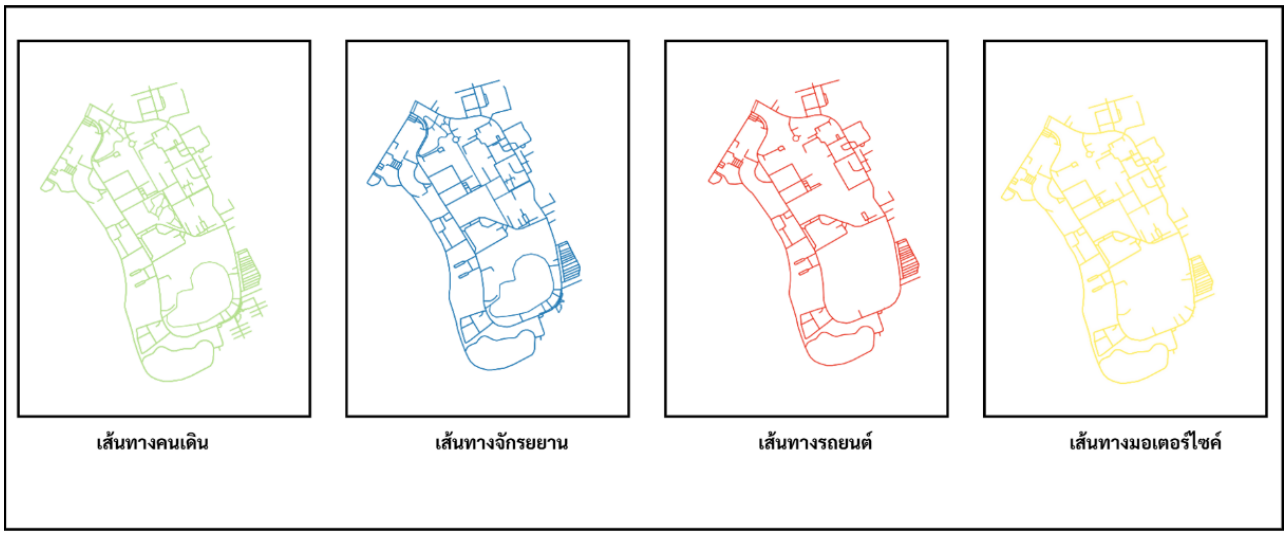
3.2.4 การแยกประเภทเส้นทาง

หลังจากนำเข้าข้อมูลเครือข่ายถนนจาก OpenStreetMap (OSM) และจัดเก็บในตาราง ways ของฐานข้อมูล PostgreSQL/PostGIS แล้ว ขั้นตอนต่อไปคือการกำหนดเงื่อนไขเพื่อแยกประเภทเส้นทางสำหรับการคำนวณที่แม่นยำยิ่งขึ้น

การกำหนดเงื่อนไขเส้นทาง เพื่อรองรับการใช้งานที่แตกต่างกันของนิสิตและบุคลากรภายในมหาวิทยาลัย ผู้วิจัยได้ทำการสร้างคอลัมน์เพิ่มเติมในตาราง ways เพื่อใช้เป็นเงื่อนไขในการคำนวณเส้นทาง โดยคอลัมน์เหล่านี้จะทำหน้าที่เป็นตัวกรอง (Filter) ในฟังก์ชัน pgr_dijkstra

ประเภทเส้นทางที่แยกออกเป็น 4 รูปแบบหลัก ได้แก่ เส้นทางคนเดิน เส้นทางจักรยาน เส้นทางรถยนต์และเส้นทางมอเตอร์ไซค์

เงื่อนไขการกำหนดเส้นทางให้เหมาะสมกับเส้นทางจริงภายในมหาวิทยาลัยนเรศวร TRUE (ผ่านได้) กำหนดให้ส่วนถนนนั้นอนุญาตให้พาหนะประเภทนั้นใช้เส้นทางได้ FALSE (ห้ามผ่าน) กำหนดให้ส่วนถนนนั้นไม่อนุญาตให้พาหนะประเภทนั้นใช้เส้นทางได้



ภาพที่ 16 ประเภทเส้นทางภายในมหาวิทยาลัยนเรศวร

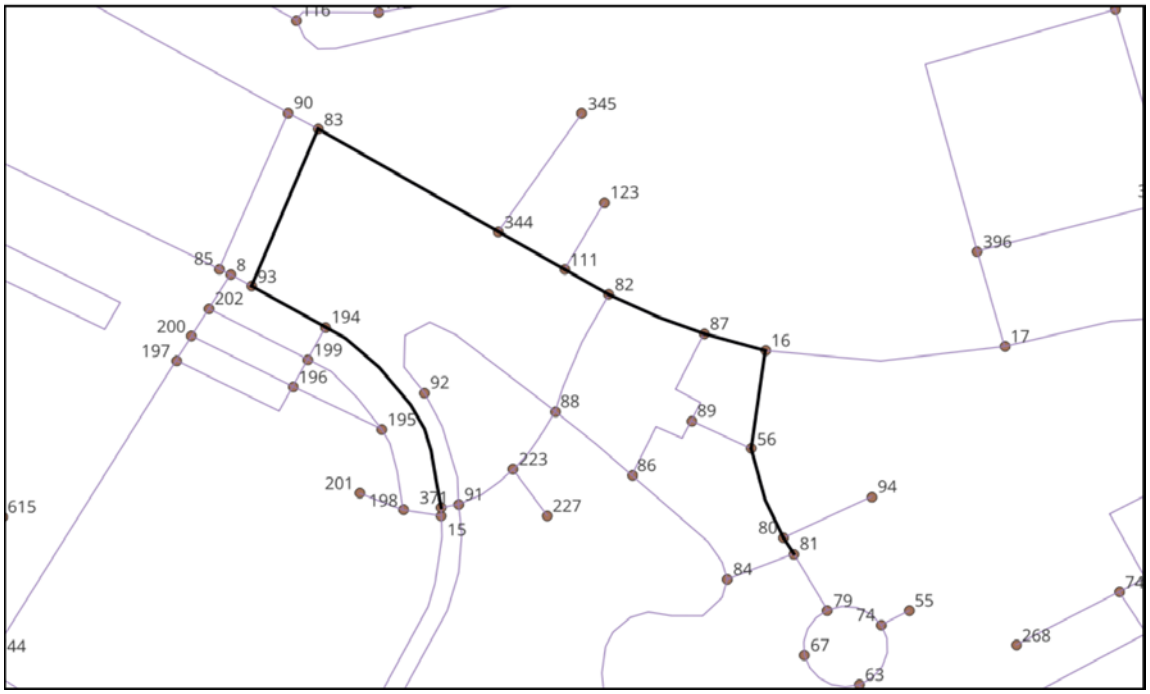
3.2.5 ใช้คำสั่ง SQL ในการค้นหาเส้นทางด้วย pgRouting ฟังก์ชัน pgr_dijkstra

ขั้นตอนนี้เป็นการสาธิตการใช้ส่วนขยาย pgRouting และฟังก์ชัน pgr_dijkstra ซึ่งเป็นหัวใจหลักของระบบนำทางอัจฉริยะเพื่อคำนวณหาเส้นทางที่สั้นที่สุดภายในมหาวิทยาลัยนเรศวร โดยใช้ข้อมูลโครงข่ายถนนจากตาราง ways หลักการทำงานของ pgr_dijkstra ฟังก์ชัน pgr_dijkstra ใช้หลักการคำนวณทางคณิตศาสตร์ของ อัลกอริทึม Dijkstra ซึ่งเป็นการนำเอาทฤษฎีกราฟมาใช้ในการวิเคราะห์โครงข่ายถนนที่จำลองด้วย จุดต่อ (Nodes) และ เส้น (Edges) แทนถนนที่เชื่อมต่อกัน โดยจะทำงานเพื่อค้นหาเส้นทางที่มี ค่าน้ำหนักรวม (Cost) น้อยที่สุด (สั้นที่สุด) จากจุดเริ่มต้นไปยังจุดหมายปลายทางได้สำเร็จ

โค้ด SQL สำหรับการคำนวณเส้นทางที่สั้นที่สุดสำหรับรถยนต์ (โดยมีเงื่อนไขการผ่านได้) จากโหนดเริ่มต้น (Source Node ID: 81) ไปยังโหนดปลายทาง (Target Node ID: 371) และสร้างผลลัพธ์เป็นตารางใหม่ชื่อ route_test

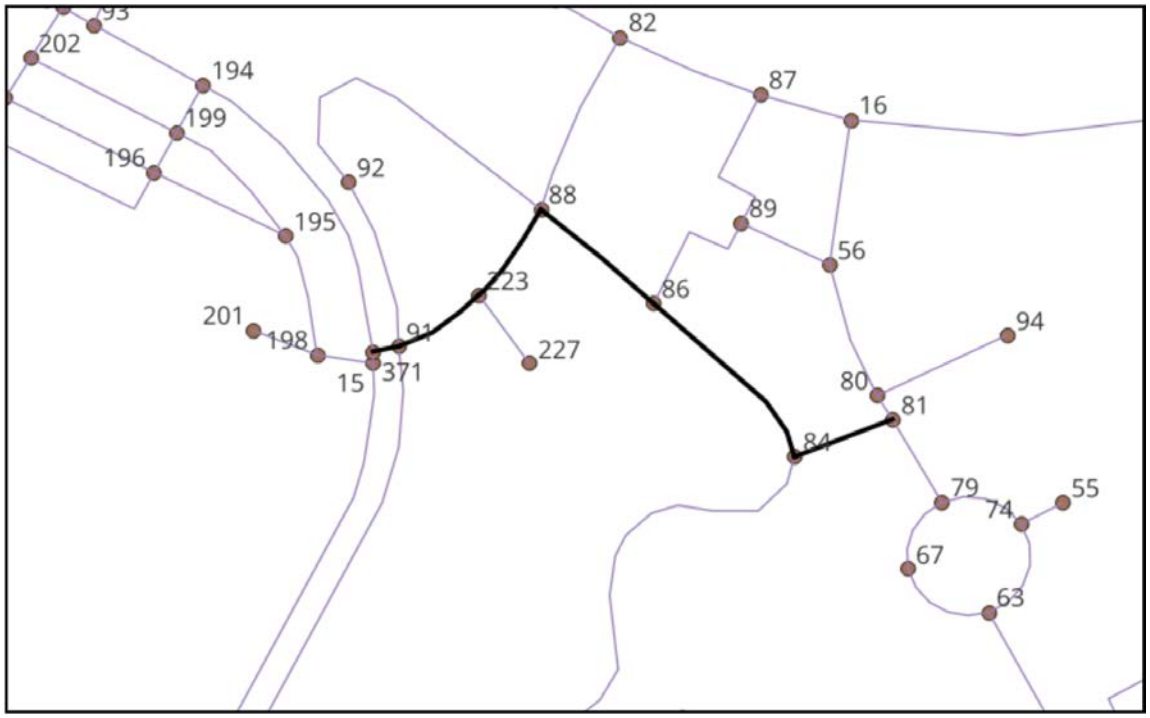
```
create table route_test as select the_geom from ways w,(SELECT * FROM pgr_dijkstra(
'SELECT gid AS id,
        source,
        target,
        length AS cost
  FROM ways
  WHERE  is_car_allowed = true ',
81,
371,
directed := false
)) as rt where w.gid=rt.edge;
```

ภาพที่ 17 การคำนวณเส้นทาง pgRouting



ภาพที่ 18 ผลลัพธ์การคำนวณเส้นทางรถยนต์

จากนั้นลองคำนวณเส้นทางคนเดิน โดยที่กำหนดโหนดเหมือนกับเส้นทางรถยนต์ โหนดเริ่มต้น (Source Node ID: 81) ไปยังโหนดปลายทาง (Target Node ID: 371)

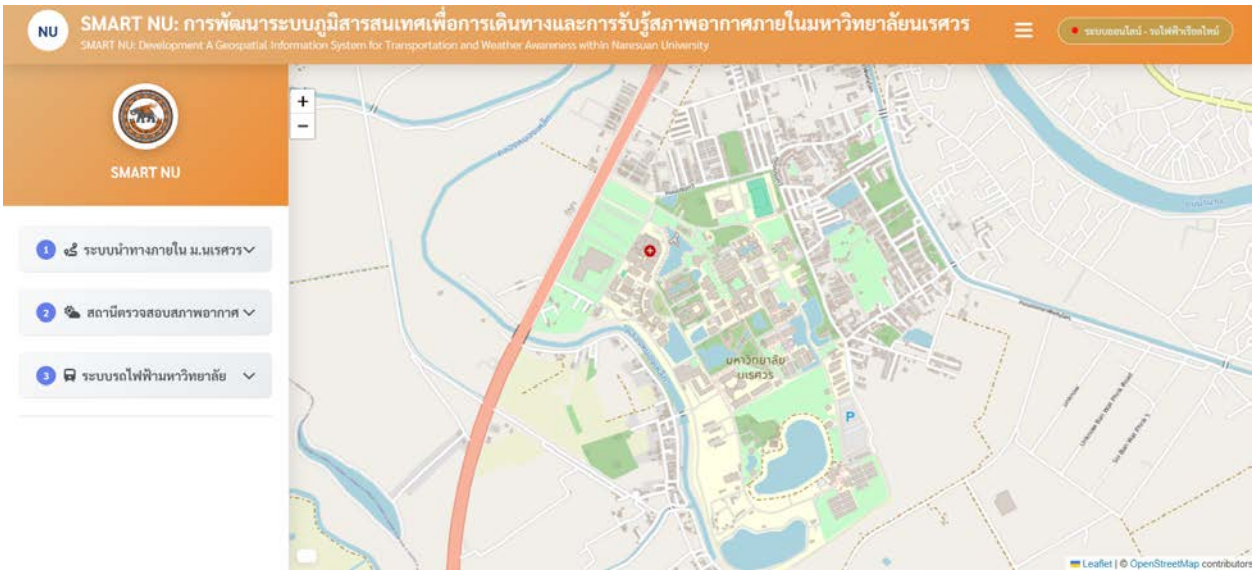


ภาพที่ 19 ผลลัพธ์การคำนวณเส้นทางคนเดิน

pgr_dijkstra จะทำการคำนวณโดยจะไล่สำรวจโหนดที่อยู่ใกล้ที่สุดทีละโหนด บันทึกค่าน้ำหนักรวมที่น้อยที่สุดที่สามารถเดินทางมาถึงโหนดนั้นได้จนกระทั่งหาเส้นทางที่รวมค่าน้ำหนักน้อยที่สุดไปยังจุดหมายปลายทางได้สำเร็จ ข้อมูลเส้นทางที่สั้นที่สุดสำหรับรถยนต์ระหว่างโหนด 81 และ 371 โดยเส้นทางที่ได้จะแตกต่างจากเส้นทางคนเดิน เนื่องจากเงื่อนไขที่พิจารณาเฉพาะถนนที่อนุญาตให้รถยนต์เข้าถึงเท่านั้น

3.2.6 ออกแบบหน้าเว็บแอปพลิเคชัน

หลังจากที่รวบรวมข้อมูลแล้ว ในขั้นตอนต่อไปคือการออกแบบการทำงานของระบบการค้นหาและคำนวณเส้นทางด้วย pgRouting พร้อมการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวรอีกทั้งยังสามารถดูตำแหน่งรถโดยสารได้แบบเรียลไทม์ จากนั้นจึงได้พัฒนาโค้ดสำหรับการเชื่อมฐานข้อมูลและหน้าเว็บแอปพิเคชัน หน้าเว็บถูกออกแบบด้วยโครงสร้างที่แบ่งออกเป็นสองส่วนหลัก คือ ส่วนแสดงผลแผนที่ทางด้านขวาและแถบควบคุมหลักทางด้านซ้าย ซึ่งแถบควบคุมนี้เป็นจุดรวมของฟังก์ชันการค้นหาเส้นทางและการควบคุม Layer ข้อมูลต่าง ๆ โดยแบ่งออกเป็นสามหมวดหมู่ได้แก่ ระบบนำทางภายใน ม.นเรศวร สถานีตรวจสอบสภาพอากาศและระบบรถโดยสารมหาวิทยาลัย



ภาพที่ 20 การออกแบบหน้าเว็บแอปพลิเคชัน

3.2.7 การพัฒนาสร้างโค้ดฐานข้อมูล

เมื่อได้ทำการสร้างฐานข้อมูลและนำเข้าข้อมูลเก็บไว้ใน PostgreSQL และ PostGIS ซึ่งมีทั้งข้อมูลสภาพอากาศข้อมูลเส้นทางภายในมหาวิทยาลัยนเรศวรในขั้นตอนนี้จะสร้างโค้ดสำหรับการเชื่อมต่อระหว่างฐานข้อมูลและเว็บแอปพลิเคชัน โดยโค้ดจะทำการดึงข้อมูลจากฐานข้อมูลไปแสดงผลบนเว็บแอปพลิเคชัน

โค้ดสำหรับระบบค้นหาเส้นทาง pgrouting.php เชื่อมต่อฐานข้อมูลของเส้นทางจากนั้นรับพิกัดจุดเริ่มต้นและจุดหมายปลายทางและประเภทพาหนะจาก Frontend เพื่อใช้ฟังก์ชัน pgr_dijkstra ในการคำนวณเส้นทางที่สั้นสุด ผลลัพธ์ส่งคืนข้อมูล GeoJSON ของเส้นทางที่คำนวณได้กลับไปยังเว็บแอปพลิเคชัน

```php
//  การตั้งค่าและเชื่อมต่อฐานข้อมูล
define("PG_DB", "test_pgrouting");
define("PG_HOST", "localhost");
define("PG_USER", "postgres");
define("PG_PORT", "5432");
define("PG_PASS", "postgres");

// กำหนดค่าคงที่สำหรับการคำนวณ Cost
define('BIKE_SPEED_KPH', 15);
define('WALK_SPEED_KPH', 5);
define('DEFAULT_VEHICLE_SPEED_KPH', 30);

// เชื่อมต่อฐานข้อมูล
$conn = pg_connect("dbname=" . PG_DB . " host=" . PG_HOST . " password=" . PG_PASS . " user=" . PG_USER . " port=" . PG_PORT);

if (!$conn) {
    echo json_encode(['error' => 'Database connection failed']);
    exit;
}

// รับพารามิเตอร์จาก Frontend
$route_type = $_GET['route_type'] ?? '';
$start_node = (int)($_GET['start_node'] ?? 0);
$end_node = (int)($_GET['end_node'] ?? 0);

// ตรวจสอบพารามิเตอร์ที่จำเป็น
if (empty($route_type) || !$start_node || !$end_node) {
    echo json_encode(['error' => 'Missing required parameters']);
    exit;
}

// กำหนด Routing Profile และ Filter
$filter_column = '';
$cost_kph = 0;
switch ($route_type) {
    case 'car':
        $filter_column = 'is_car_allowed';
        $cost_kph = DEFAULT_VEHICLE_SPEED_KPH;
        break;
    case 'motorcycle':
        $filter_column = 'is_motorcycle_allowed';
        $cost_kph = DEFAULT_VEHICLE_SPEED_KPH;
        break;
    case 'bike':
        $filter_column = 'is_bicycle_allowed';
        $cost_kph = BIKE_SPEED_KPH;
        break;
    case 'walk':
        $filter_column = 'is_walkable';
        $cost_kph = WALK_SPEED_KPH;
        break;
    default:
        echo json_encode(['error' => 'Invalid route type']);
        exit;
}
// 5. สร้าง Query สำหรับ pgRouting
$sub_query_raw = "SELECT gid AS id, source, target, {$cost_calc} AS cost, {$reverse_cost_calc} AS reverse_cost
FROM public.ways WHERE {$filter_column} = TRUE";
$sub_query = pg_escape_string($conn, $sub_query_raw);
// 6. เรียกใช้ pgr_dijkstra Algorithm
$sql_route = "
    SELECT
        ST_AsGeoJSON(ST_Collect(w.the_geom)) AS route_geojson,
        SUM(route.cost) AS total_cost
    FROM public.ways AS w
    JOIN
        pgr_dijkstra(
            '{$sub_query}',
            {$start_node},
            {$end_node},
            FALSE
        ) AS route
    ON
        w.gid = route.edge;
";
```

ภาพที่ 21 ตัวอย่างโค้ด pgrouting.php การคำนวณเส้นทาง

โค้ด weather.php ทำหน้าที่ดึงข้อมูลสภาพอากาศล่าสุดแบบเรียลไทม์จากฐานข้อมูล PostgreSQL เพื่อแสดงผลบนแผนที่และป๊อปอัพดึงข้อมูลสภาพอากาศล่าสุด จากตาราง weather_station ซึ่งรวมข้อมูลค่าฝุ่นPM, อุณหภูมิ, ความชื้น, ความเร็วลมและปริมาณฝน

```php
$sql = "
    SELECT json_agg(row_to_json(result))
    FROM (
        SELECT DISTINCT ON (i.name)
            n.eui, n.wind_direct, n.wind_speed, n.temperature, n.humidity,
            n.pm, n.rain, n.rainacc, n.date_time, n.rssi, n.snr,
            i.name, i.location_name, i.latitude, i.longitude
        FROM weather_station1 n, node_info i
        WHERE n.eui = i.eui and i.name IN ('NU-01', 'NU-02', 'NU-03', 'NU-04', 'NU-05') and n.pm > 0
        ORDER BY i.name, n.date_time DESC
    ) AS result;
";


// Perform database query
$query = pg_query($con, $sql);

if (!$query) {
    echo json_encode(["error" => "Database query failed"]);
    exit;
}

// Fetch the JSON result
$row = pg_fetch_assoc($query);
$jsonResult = $row['json_agg']; // PostgreSQL returns a JSON array

// Close the database connection
pg_close($con);

// Return JSON response
header('Content-Type: application/json');
echo $jsonResult;
?>
```

ภาพที่ 22 ตัวอย่างโค้ด weather.php ดึงข้อมูลสภาพอากาศจากฐานข้อมูล

โค้ดสำหรับข้อมูลย้อนหลัง weather_history.php ไฟล์เหล่านี้รองรับการดึงข้อมูลตามช่วงเวลาที่กำหนดเพื่อใช้ในการสร้างกราฟข้อมูลย้อนหลังรับพารามิเตอร์ eui (รหัสสถานี), start (วันที่เริ่มต้น), และ end (วันที่สิ้นสุด) และใช้คำสั่ง SQL เพื่อดึงข้อมูลในช่วงเวลาที่กำหนดจากตารางข้อมูลสภาพอากาศที่เกี่ยวข้อง

```php
// รับค่าพารามิเตอร์ที่จำเป็นจาก URL
$eui = $_GET['eui'] ?? null;
$start_date = $_GET['start'] ?? null;
$end_date = $_GET['end'] ?? null;

// ตรวจสอบความถูกต้องของพารามิเตอร์
if (!$eui || !$start_date || !$end_date) {
    http_response_code(400); // ส่งสถานะ Bad Request
    echo json_encode(['error' => 'พารามิเตอร์ที่จำเป็นไม่ครบถ้วน (eui, start, หรือ end)']);
    exit;
}

try {
    // สร้างคำสั่ง SQL โดยใช้ Prepared Statement
    // การใช้ JOIN กับ node_info จะทำให้โค้ดมีความยืดหยุ่นมากขึ้น
    // ใช้ตัวแปร $1, $2, $3 สำหรับการแทนค่า ซึ่งปลอดภัยจาก SQL Injection
    $sql = "
        SELECT
            n.eui,
            n.date_time,
            n.pm,
            n.temperature,
            n.humidity,
            n.wind_speed,
            n.wind_direct,
            n.rain,
            n.rainacc
        FROM weather_station1 n
        JOIN node_info i ON n.eui = i.eui
        WHERE n.eui = $1
          AND n.date_time BETWEEN $2 AND $3
        ORDER BY n.date_time ASC
    ";
```

ภาพที่ 23 ตัวอย่างโค้ด weather_history.php ข้อมูลย้อนหลัง

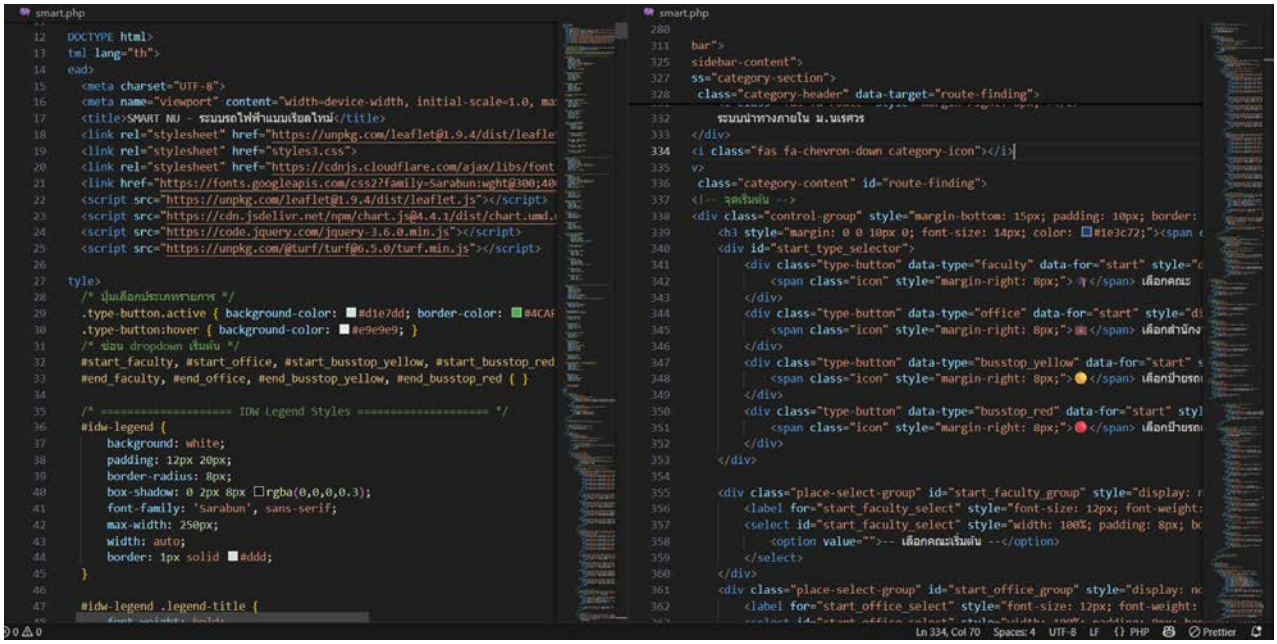
โค้ด carapi.php ทำการดึงตำแหน่งรถโดยสารแบบเรียลไทม์จาก API ภายนอก

```php
<?php
header('Content-Type: application/json');

$url = "https://transit.nu.ac.th/allGPS"; // ลิงก์ API ตำแหน่งรถแบบเรียลไทม์

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response = curl_exec($ch);

if (curl_errno($ch)) {
    echo json_encode(["error" => curl_error($ch)]);
}
curl_close($ch);

echo $response;
?>
```

ภาพที่ 24 ตัวอย่างโค้ด carapi.php ตำแหน่งรถโดยสาร

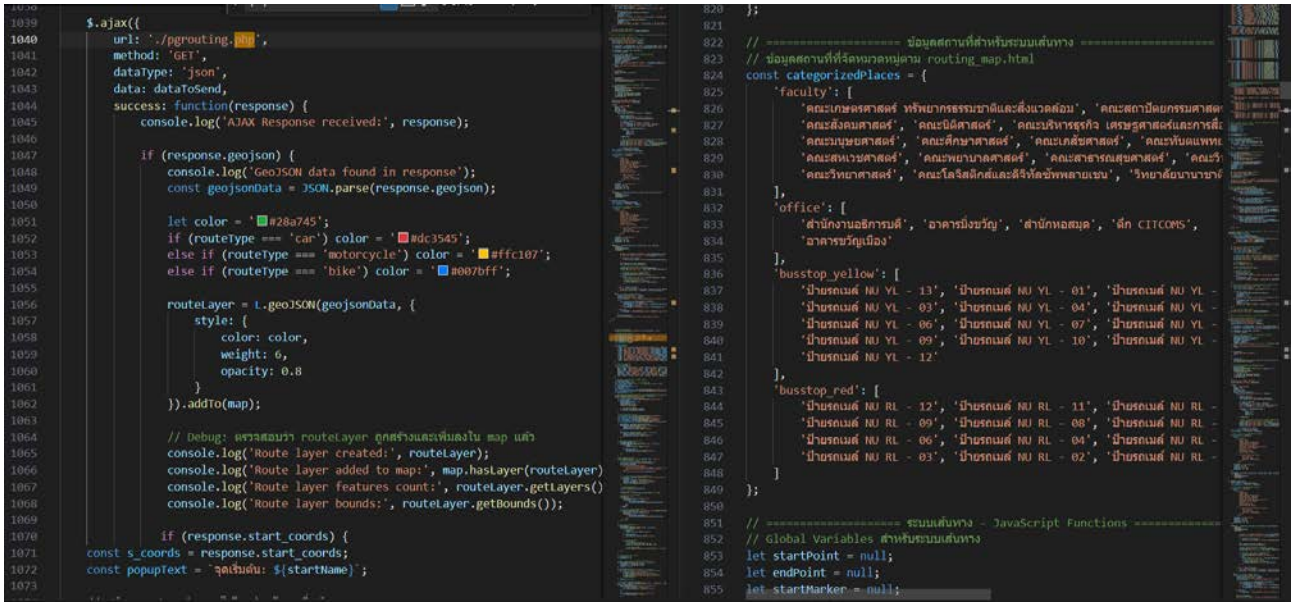
3.2.8 การพัฒนาเว็บแอปพลิเคชัน

หลังจากที่ทำการพัฒนาสร้างโค้ดที่เชื่อมต่อกับฐานข้อมูลแล้ว จึงเข้าสู่กระบวนการเขียนโค้ดเพื่อพัฒนาหน้าแอปพลิเคชันเป็นการออกแบบและการตกแต่งหน้าเว็บแอปพลิเคชันเพื่อแสดงผลข้อมูล เช่น การรคำนวณเส้นทางและข้อมูลสภาพอากาศ ใช้ HTML กำหนดโครงสร้างและเนื้อหาของหน้าเว็บแอปพลิเคชันทั้งหมด เช่น การสร้างส่วน Header Sidebar และส่วนแสดงผลแผนที่หลัก CSS ใช้ในการออกแบบและตกแต่งหน้าเว็บแอปพลิเคชัน เพื่อให้เว็บไซต์มีความสวยงามและเป็นระเบียบ เช่น การกำหนดสี, ฟอนต์, โครงร่าง (Layout) และการจัดองค์ประกอบให้เหมาะสมกับหน้าจอที่แตกต่างกัน JavaScript เป็นภาษาหลักในการสร้างการโต้ตอบ (Interactivity) และตรรกะฝั่งไคลเอนต์ เช่น การจัดการการคลิกปุ่ม การแสดงหรือซ่อน Layer ข้อมูลและการประมวลผลข้อมูลที่ได้รับมาจากเซิร์ฟเวอร์ และระบบเลือกใช้ไลบรารี Leaflet.js ซึ่งเป็นไลบรารี JavaScript ใช้ในการสร้างแผนที่แบบโต้ตอบที่ผู้ใช้สามารถซูม, เลื่อน, เลือกชั้นข้อมูล (Layers) และแสดงข้อมูลแบบไดนามิกได้อย่างมีประสิทธิภาพและ JavaScript จะทำการเรียกใช้โค้ด PHP ฝั่งเซิร์ฟเวอร์ (เช่น pgrouting.php) ผ่าน AJAX เพื่อดึงผลลัพธ์เป็น GeoJSON และใช้ Leaflet.js ในการวาดรูปทรงเรขาคณิต (เช่น เส้นทางที่คำนวณได้) และ Marker ต่าง ๆ บนแผนที่



ภาพที่ 25 ตัวอย่างโค้ดหน้าเว็บหลัก

จากนั้นผู้วิจัยได้ ทำการเรียกใช้ pgrouting.php ที่สร้างไปก่อนหน้านี้ เพื่อคำนวณเส้นทางที่สั้นที่สุดลงบนหน้าเว็บแอปพลิเคชันหลัก



ภาพที่ 26 ตัวอย่างโค้ดการคำนวณเส้นทาง

ทำการเรียกข้อมูล weather.php สถานีตรวจสอบสภาพอากาศเพื่อเชื่อมเข้ากับหน้าเว็บหลักการแสดงผล ข้อมูลที่ถูกดึงมาจะถูกใช้ในการวาง Marker บนแผนที่ตามพิกัดของแต่ละสถานี สีของ Marker จะถูกกำหนดตามค่าฝุ่น PM2.5 (ฟังก์ชัน pmColor()) ตามมาตรฐานคุณภาพอากาศไทยและมีข้อมูลอุณหภูมิ ความชื้น ความเร็วลม ปริมาณฝนและสร้างกราฟเส้นของข้อมูลสภาพอากาศเพื่อให้ผู้ใช้รับรู้สถานการณ์ได้อย่างรวดเร็ว

```
// 1. Weather System Constants - ใช้ endpoint เดียวสำหรับข้อมูล weather ทั้งหมด
const ENDPOINT_WEATHER_LATEST = './weather.php';
const ENDPOINT_W1_HISTORY = './weather_history.php';


const REFRESH_MS = 30_000; // อัปเดตทุก 30 วินาที

// 2. Station Config - ใช้ endpoint เดียวสำหรับข้อมูล weather ทั้งหมด
const STATIONS = [
    { name: 'คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อม', eui: '0000fcc23d222cb9', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7
    { name: 'แปลงเกษตร', eui: '0000fcc23d223e2f', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7423, lng: 100.1985 },
    { name: 'อาคาร KNECC', eui: '0000fcc23d22ac5d', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7460, lng: 100.1955 },
    { name: 'คณะวิศวกรรมศาสตร์', eui: '0000fcc23d22b5a1', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7424, lng: 100.1973 },
    { name: 'คณะวิทยาศาสตร์', eui: '0000fcc23d22b5a2', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7426, lng: 100.1954 },
    { name: 'คณะวิทยาศาสตร์(สาขาเคมี)', eui: '0000fcc23d22b5a3', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7420, lng: 100.1957
    { name: 'คณะวิทยาศาสตร์(สาขาคณิตศาสตร์)', eui: '0000fcc23d22b5a4', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7425, lng: 10
    { name: 'คณะวิทยาศาสตร์(สาขาฟิสิกส์)', eui: '0000fcc23d22b5a5', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7428, lng: 100.19
    { name: 'คณะวิทยาศาสตร์(สาขาชีววิทยา)', eui: '0000fcc23d22b5a6', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7430, lng: 100.
    { name: 'คณะวิทยาศาสตร์(สาขาเคมี)', eui: '0000fcc23d22b5a7', latest_endpoint: ENDPOINT_WEATHER_LATEST, history_endpoint: ENDPOINT_W1_HISTORY, lat: 16.7422, lng: 100.1956
];

// 3. PM Color function (ปรับแต่งสีตามค่าฝุ่น PM2.5)
function pmColor(v) {
    if (v == null || isNaN(v)) return '#7f8c8d';
    v = Number(v);

    // ตามมาตรฐาน WHO และกรมควบคุมมลพิษ
    if (v <= 12.0) return '#007BFF';      // ฟ้า - ดีมาก (0-12 µg/m³)
    if (v <= 25.0) return '#2BA745';      // เขียว - ดี (12.1-25 µg/m³)
    if (v <= 37.5) return '#FFC107';      // เหลือง - ปานกลาง (25.1-37.5 µg/m³)
    if (v <= 50.0) return '#FD7E14';      // ส้ม - เริ่มมีผลกระทบ (37.6-50 µg/m³)
    if (v <= 75.0) return '#FF6B35';      // ส้มแดง - มีผลกระทบ (50.1-75 µg/m³)
    return '#DC3545';                     // แดง - อันตราย (>75 µg/m³)
}
```

ภาพที่ 27 ตัวอย่างโค้ดเรียกข้อมูล weather.php

การสร้างแผนที่ประมาณค่าฝุ่น PM2.5 (IDW Interpolation)เพื่อแสดงรูปแบบการกระจายตัวของฝุ่น PM2.5 ทั่วพื้นที่มหาวิทยาลัยอย่างต่อเนื่อง

ขั้นตอนการทำงาน รวบรวมข้อมูล ดึงพิกัดและค่า PM2.5 จากสถานีตรวจวัดทั้งหมด สร้าง Grid ใช้ฟังก์ชัน turf.hexGrid เพื่อสร้างตารางทรงหกเหลี่ยม (Hexagonal Grid) คลุมพื้นที่มหาวิทยาลัย คำนวณค่าใช้ฟังก์ชันที่กำหนดเอง (idwAtCoord) ซึ่งอิงหลักการ IDW เพื่อประมาณค่า PM2.5 ให้กับจุดศูนย์กลางของแต่ละเซลล์ใน Grid โดยให้ความสำคัญกับค่าจากสถานีที่อยู่ใกล้เคียงมากกว่า แสดงผลวาด Polygon ของตารางหกเหลี่ยมแต่ละเซลล์บนแผนที่ Leaflet โดยใช้สีที่แตกต่างกันตามระดับค่า PM2.5 ที่ประมาณได้

```
function showIDWInterpolation() {
    // เก็บข้อมูล PM2.5 จากสถานีที่มีข้อมูล
    const pm25Data = [];
    // สร้าง FeatureCollection สำหรับ turf.js
    const pointFeatures = pm25Data.map(point =>
        turf.point([point.lng, point.lat], {
            pm: point.pm25,
            name: point.name,
            eui: point.eui
        })
    );
    const featuresFC = turf.featureCollection(pointFeatures);
    // สร้าง hexagonal grid
    const hexGrid = buildHexGrid(featuresFC, 0.05);

    // คำนวณ PM2.5 สำหรับแต่ละ hexagon ใช้ IDW
    for (const cell of hexGrid.features) {
        const center = turf.centerOfMass(cell).geometry.coordinates;
        const pmVal = idwAtCoord(center, featuresFC, 2, 3);
        cell.properties.pm = pmVal;
    }
}
function buildHexGrid(featuresFC, cellSideKm) {
    const bbox = turf.bbox(featuresFC);
    // เพิ่ม padding ให้ bbox
    const padLng = (bbox[2] - bbox[0]) * 0.05 || 0.001;
    const padLat = (bbox[3] - bbox[1]) * 0.05 || 0.001;
    const bboxP = [bbox[0]-padLng, bbox[1]-padLat, bbox[2]+padLng, bbox[3]+padLat];

    const hex = turf.hexGrid(bboxP, cellSideKm, {units:'kilometers'});
    return hex;
}
function idwAtCoord(coord, featuresFC, power=2, maxDistanceKm=0) {
    const pts = featuresFC.features.filter(f => Number.isFinite(f.properties.pm));
    let num = 0, den = 0;
    for (const s of pts) {
        const d = turf.distance(turf.point(coord), s, {units:'kilometers'});
        if (maxDistanceKm && d > maxDistanceKm) continue;
        // สูตร IDW: w = 1 / d^power
        const w = 1 / Math.pow(Math.max(d, 1e-6), power);
        num += w * s.properties.pm;
        den += w;
    }

    return num / den;
}
function renderHexGrid(hexFC) {
    hexFC.features.forEach(feature => {
        const pm = feature.properties.pm;
        if (pm !== null && !isNaN(pm)) {
            const color = getIDWColor(pm);

            const hexPolygon = L.polygon(
                feature.geometry.coordinates[0].map(coord => [coord[1], coord[0]]),
                {
                    fillColor: color,
                    color: '#555',
                    weight: 0.3,
                    fillOpacity: 0.75
                }
            );
        }
    });
}
```

ภาพที่ 28 ตัวอย่างโค้ดการสร้างแผนที่ประมาณค่าฝุ่น PM2.5 (IDW)

สร้างการแสดงผลตำแหน่งรถโดยสารแบบเรียลไทม์หน้าเว็บแอปพลิเคชันหลักใช้โค้ด JavaScript (AJAX) ในการเรียกข้อมูลตำแหน่งปัจจุบันของรถโดยสารจากไฟล์ carapi.php ซึ่งทำหน้าที่เป็น API Endpoint สำหรับรับข้อมูลจากระบบ GPS ที่ติดตั้งในรถโดยสารที่แยกสีรถเป็นรถสีนำเงิน รถสีเหลืองและรถสีแดง

```
// 10. ฟังก์ชันสำหรับการแสดงรถโดยสาร
function showBusStatistics() {
    fetch('./carapi.php')
        .then(response => response.json())
        .then(data => {
            if (data && Array.isArray(data)) {
                const totalBuses = data.length;
                const yellowLineBuses = data.filter(car =>
                    car.plateNumber.includes('9)40-0202') ||
                    car.plateNumber.includes('(7)40-0203') ||
                    car.plateNumber.includes('(12)40-0193')
                ).length;
                const redLineBuses = data.filter(car =>
                    car.plateNumber.includes('(1)40-0191') ||
                    car.plateNumber.includes('(3)40-0199') ||
                    car.plateNumber.includes('(4)40-0204')
                ).length;
                const blueLineBuses = totalBuses - yellowLineBuses - redLineBuses;

                console.log('Bus Statistics:', {
                    total: totalBuses,
                    yellowLine: yellowLineBuses,
                    redLine: redLineBuses,
                    blueLine: blueLineBuses
                });
            }
        })
        .catch(error => {
            console.error('Error fetching bus statistics:', error);
        });
}
```

ภาพที่ 29 ตัวอย่างโค้ดตำแหน่งรถแบบเรียลไทม์

บทที่ 4

ผลการวิจัย

SMART NU การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร โดยผลการวิจัยครอบคลุมการพัฒนาประกอบไปด้วยฟังก์ชันคำนวณเส้นทางด้วย pgRouting ที่แยกประเภทของเส้นทางและการแสดงข้อมูลสภาพอากาศภายในมหาวิทยาลัยนเรศวรรวมถึงการแสดงตำแหน่งรถโดยสาร ระบบนี้สามารถช่วยนักศึกษา บุคลากร ผู้มาเยือน ให้มีความสะดวกสบายและความแม่นยำในการค้นหาเส้นทางและคำนวณเส้นทาง โดยระบบเว็บแอปพลิเคชันมีระบบนี้มีฟังก์ชันหลักๆ ดังนี้

4.1 หน้าเว็บแอปพลิเคชันหลัก

หน้าเว็บแอปพลิเคชันหลักของระบบถูกพัฒนาขึ้นในรูปแบบของ Web Application โดยใช้ภาษา HTML, CSS, และ JavaScript เพื่อให้ผู้ใช้งานสามารถเข้าถึงระบบได้ง่ายผ่านเว็บเบราว์เซอร์ โครงสร้างหน้าเว็บถูกแบ่งออกเป็นสองส่วนหลักได้แก่

1. แถบควบคุม (Sidebar) อยู่ทางด้านซ้ายของหน้าจอ ทำหน้าที่เป็นจุดรวมของฟังก์ชันการทำงานหลักและเครื่องมือควบคุมทั้งหมด ผู้ใช้สามารถควบคุมชั้นข้อมูล (Layers) และเข้าถึงฟังก์ชันการวิเคราะห์ต่าง ๆ ได้จากส่วนนี้ แถบควบคุมแบ่งออกเป็นสามหมวดหมู่หลัก ได้แก่ 1. ระบบนำทางภายใน ม.นเรศวร สำหรับฟังก์ชันค้นหาและคำนวณเส้นทาง 2. สถานีตรวจสอบสภาพอากาศ สำหรับการควบคุมการแสดงผลข้อมูลสภาพอากาศและกราฟ 3. ระบบรถโดยสารในมหาวิทยาลัย สำหรับการควบคุมการแสดงผลเส้นทางและตำแหน่งรถโดยสาร

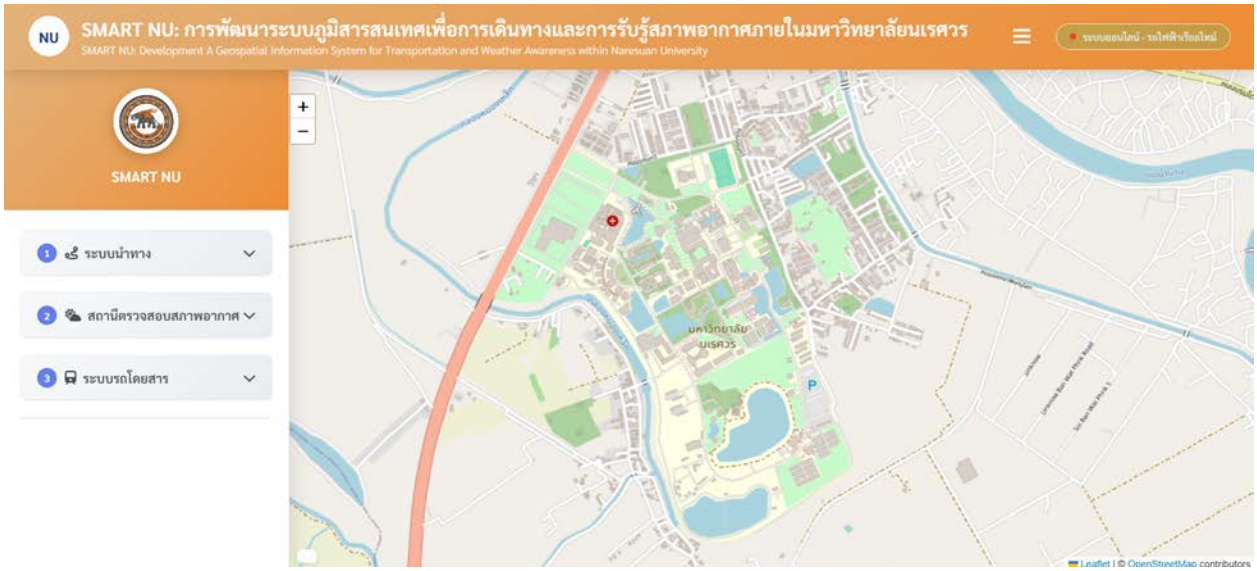
2. ส่วนแสดงผลแผนที่ (Map View) อยู่ทางด้านขวาของหน้าจอเป็นพื้นที่แสดงผลข้อมูลเชิงพื้นที่ทั้งหมดของมหาวิทยาลัยนเรศวร โดยใช้ไลบรารี Leaflet.js ข้อมูลที่แสดงผล ได้แก่

1. แผนที่ฐาน (Base Map) แผนที่พื้นหลังของมหาวิทยาลัยนเรศวร

2. ระบบนำทางภายใน ม.นเรศวร แสดงเส้นทางผลลัพธ์จากฟังก์ชัน pgRouting

3. สถานีตรวจสอบสภาพอากาศ แสดง Marker ข้อมูลสภาพอากาศแบบเรียลไทม์

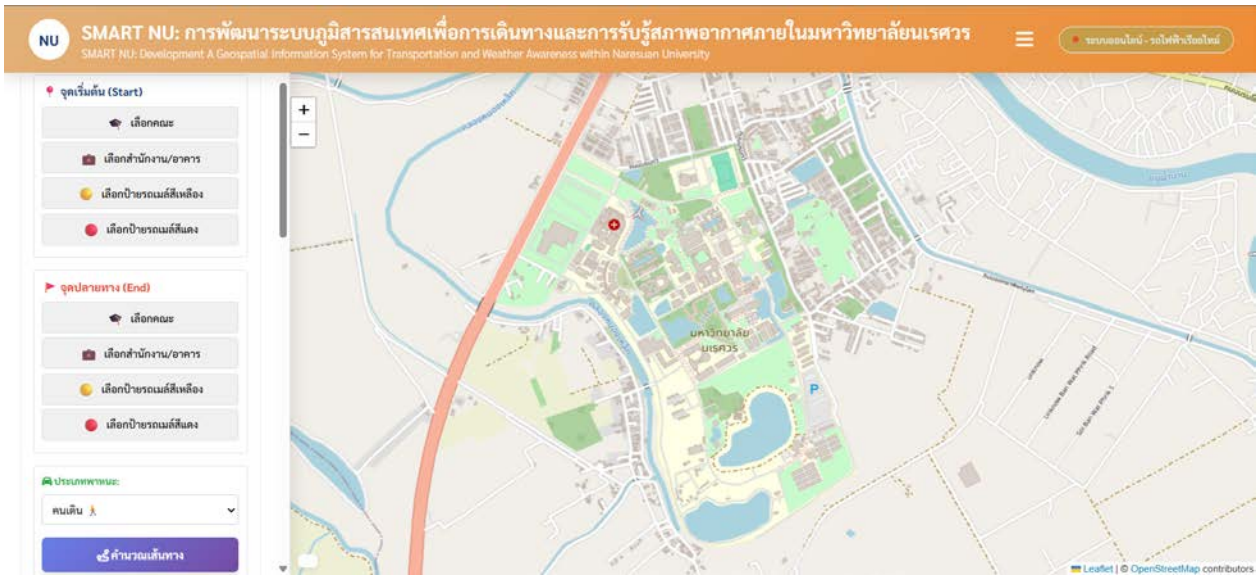
4. ระบบรถโดยสารในมหาวิทยาลัย แสดงตำแหน่งปัจจุบันของรถโดยสารในมหาวิทยาลัย



ภาพที่ 30 ตัวอย่างหน้าเว็บแอปพลิเคชัน

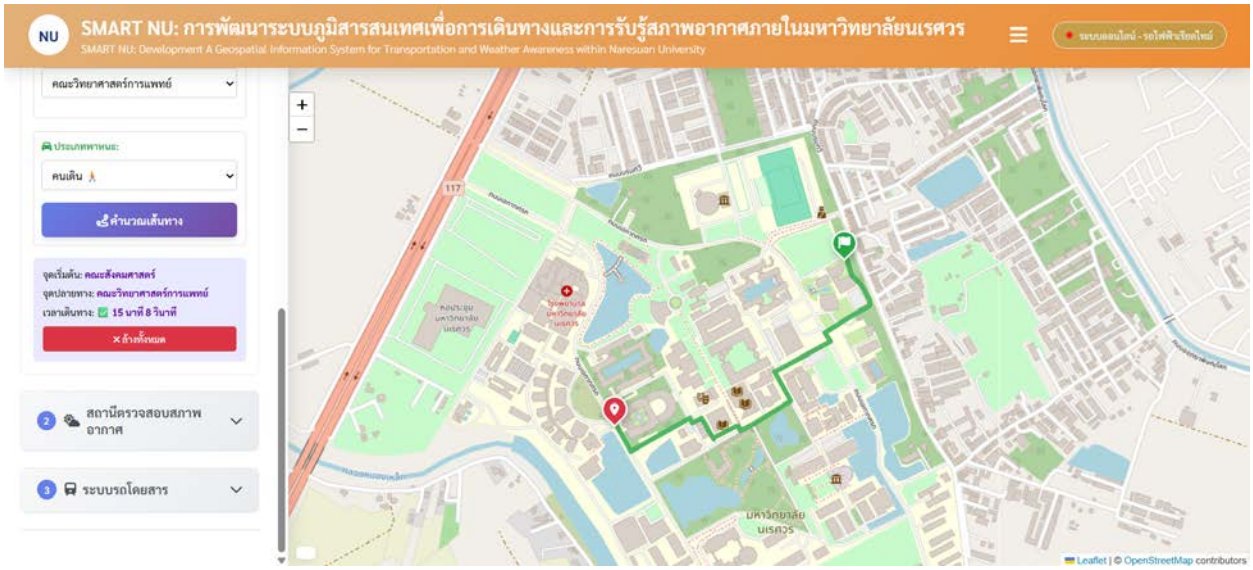
4.2 หมวดหมู่ระบบนำทางภายใน ม.นเรศวร

ระบบนำทางได้รับการพัฒนาโดยใช้ pgRouting อัลกอริทึม Dijkstra ผู้ใช้งานสามารถเลือก จุดเริ่มต้น และ จุดปลายทาง ระบบมีรายการให้เลือก ได้แก่ คณะ สำนักงาน ป้ายรถโดยสารสีเหลือง และป้ายรถโดยสารสีแดง จากนั้นผู้ใช้งานต้องเลือกประเภทพาหนะที่ใช้ในการเดินทาง เช่น เส้นทางคนเดิน (สีเขียว), เส้นทางมอเตอร์ไซค์(สีเหลือง), เส้นทางจักรยาน(สีฟ้า), และเส้นทางรถยนต์(สีแดง) เมื่อเลือกข้อมูลเสร็จแล้วระบบจะมีให้กดคำนวณเส้นทาง



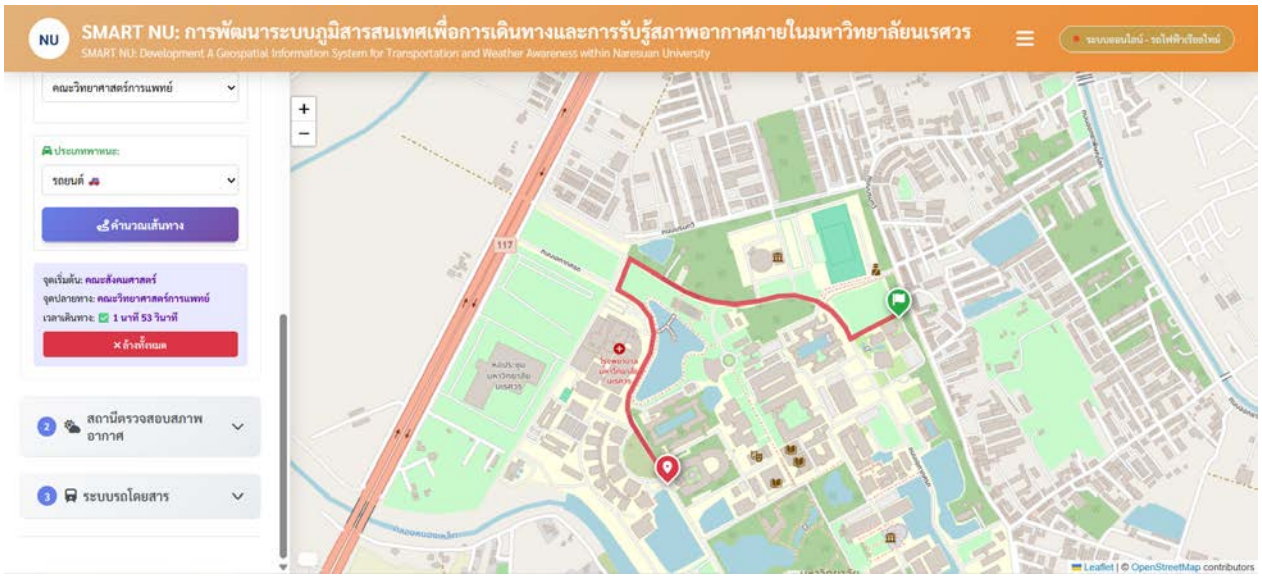
ภาพที่ 31 ตัวอย่างหน้าระบบนำทางบนเว็บแอปพลิเคชัน

การคำนวณและการแสดงเส้นทางบนแผนที่จะมีรายละเอียดบอกจุดเริ่มต้นที่ผู้ใช้งานเลือกและจุดหมายปลายทางมีเวลาการเดินทางระบุ ดังภาพที่32 เป็นการเลือกจุดเริ่มต้นจากคณะไปยังจุดหมายปลายทางที่เป็นคณะโดยจะเป็น จุดเริ่มต้นคณะคณะสังคมศาสตร์ จุดหมายปลายทางคณะวิทยาศาสตร์การแพทย์ เลือกประเภทเส้นทางเป็นคนเดินจากนั้นเส้นทางจะแสดงบนแผนที่ซึ่งเป็นเส้นทางสีเขียวคือเส้นทางคนเดิน



ภาพที่ 32 การคำนวณเส้นทางคนเดินจากคณะไปยังจุดหมายปลายทางที่เป็นคณะ

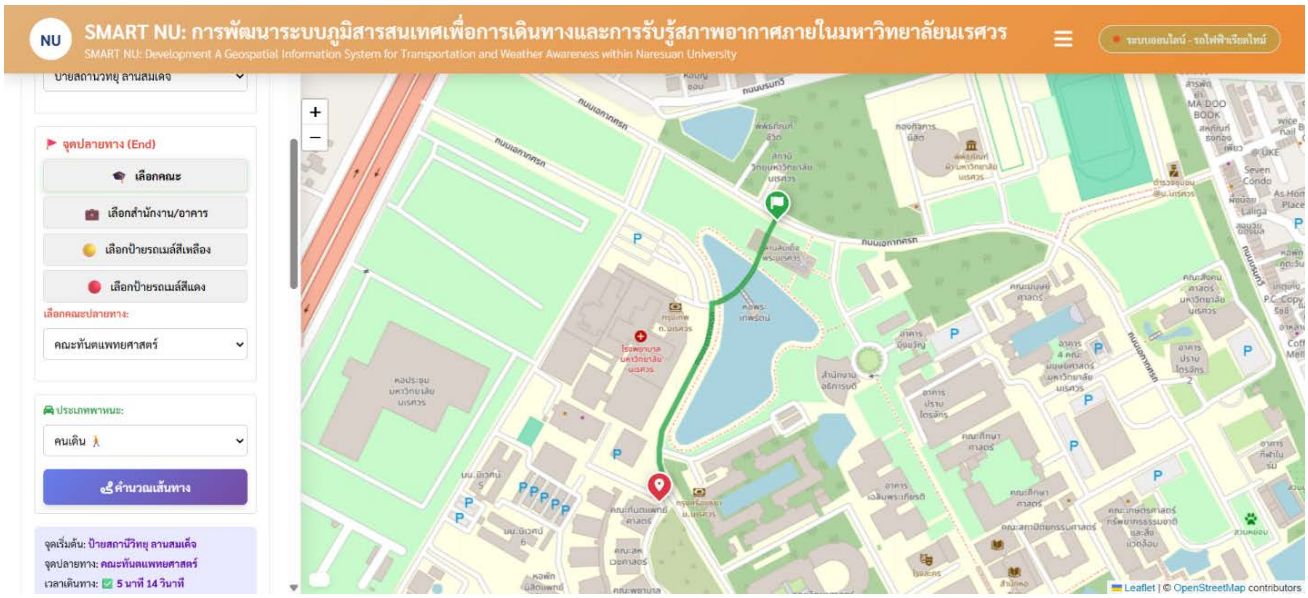
ทำการคำนวณเส้นทางโดยเลือกจุดเริ่มต้นและจุดหมายปลายทางเหมือนกัน จากนั้นเลือกเป็นประเภทเส้นทางเป็นรถยนต์ ดังรูปภาพที่33 จะเห็นได้ว่าผลของเส้นทางจะเปลี่ยนไปตามเส้นสีแดงเพราะเป็นถนนสำหรับรถยนต์วิ่ง ซึ่งเส้นทางจะไม่สามารถผ่านไปได้เหมือนกับเส้นทางคนเดิน



ภาพที่ 33 การคำนวณเส้นทางรถยนต์จากคณะไปยังจุดหมายปลายทางที่เป็นคณะ

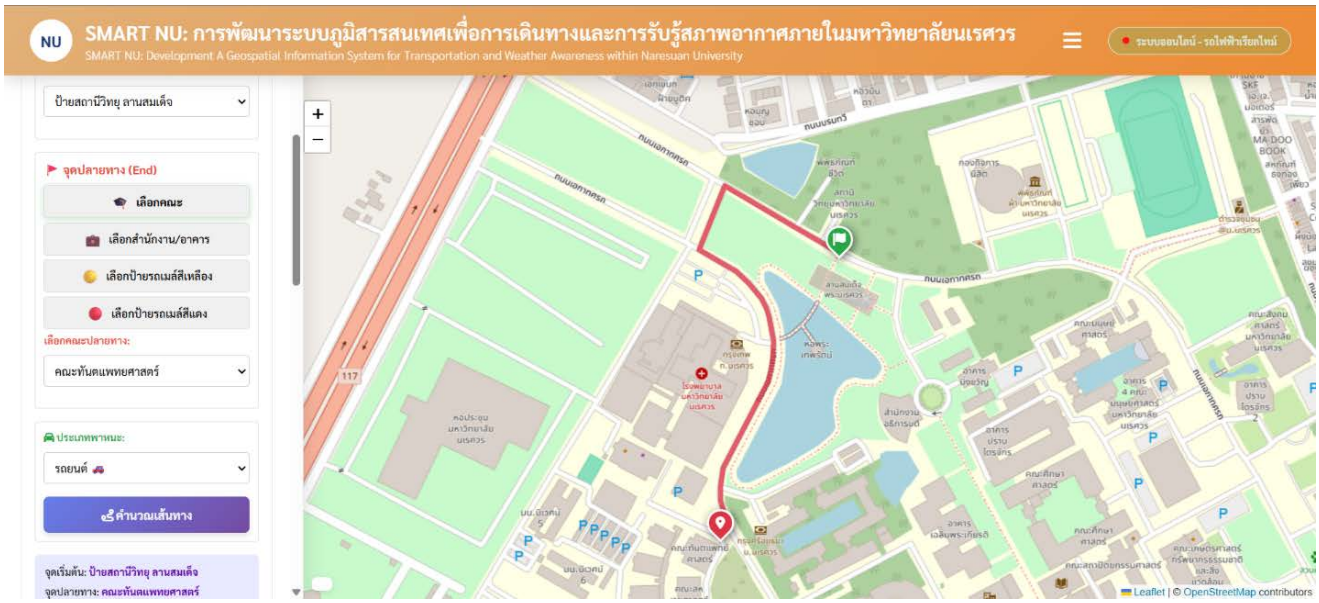
ตัวอย่างการคำนวณเส้นทาง จุดเริ่มต้นป้ายรถโดยสารสีเหลืองลานสมเด็จ จุดหมายปลายทางคณะแพทยศาสตร์เลือกประเภทเส้นทางทั้งสี่ประเภท ได้แก่ เส้นทางคนเดิน เส้นทางรถยนต์ เส้นทางมอเตอร์ไซค์ เส้นทางจักรยาน จากนั้นเส้นทางจะแสดงบนแผนที่

เส้นทางคนเดิน



ภาพที่ 34 การคำนวณเส้นทางคนเดิน

เส้นทางรถยนต์

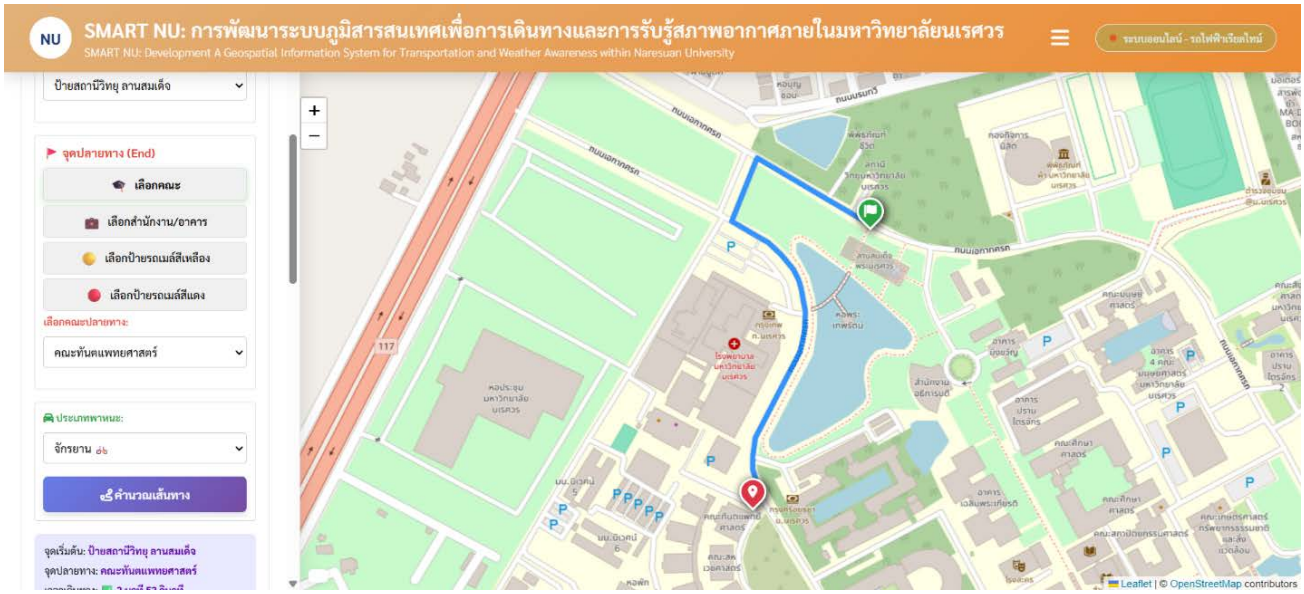


ภาพที่ 35 การคำนวณเส้นทางรถยนต์

เส้นทางมอเตอร์ไซค์

ภาพที่ 36 การคำนวณเส้นทางมอเตอร์ไซค์

เส้นทางจักรยาน



ภาพที่ 37 การคำนวณเส้นทางจักรยาน

4.3 หมวดหมู่สถานีตรวจสอบสภาพอากาศ

หมวดหมู่สถานีตรวจสอบสภาพอากาศได้รับการพัฒนาเพื่อนำเสนอข้อมูลสภาพแวดล้อมที่สำคัญแก่ผู้ใช้งานแบบเรียลไทม์ โดยใช้ข้อมูลจากเซนเซอร์ 10 สถานีที่ติดตั้งอยู่ภายในมหาวิทยาลัยระบบแสดงข้อมูลสภาพอากาศแบบเรียลไทม์จากตำแหน่งสถานี โดยผู้ใช้สามารถควบคุมการแสดงผลผ่านสวิตช์ควบคุมในแถบควบคุม ได้แก่

1. ค่าฝุ่น (PM2.5) สามารถเลือกดูค่า PM2.5 ที่แสดงผลเป็นตัวเลขบน Marker

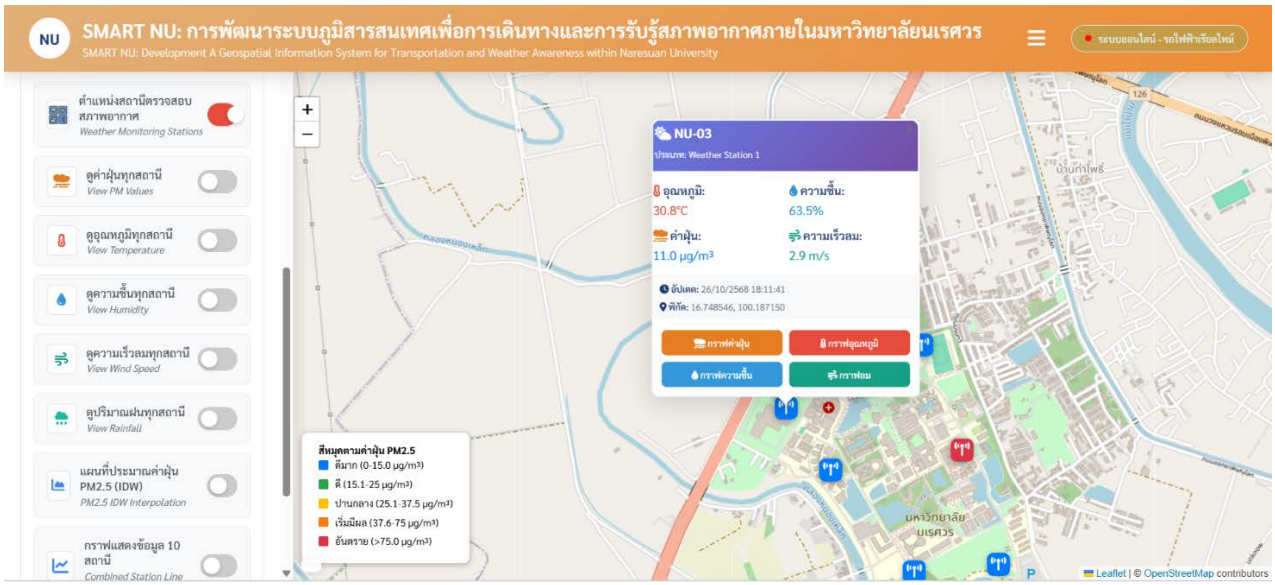
2. อุณหภูมิ สามารถเลือกดูค่าอุณหภูมิ (°C) ของแต่ละสถานี

3. ความชื้น สามารถเลือกดูค่าความชื้น (%) ของแต่ละสถานี

4. ความเร็วลม สามารถเลือกดูค่าความเร็วลม (m/s) และทิศทางลม

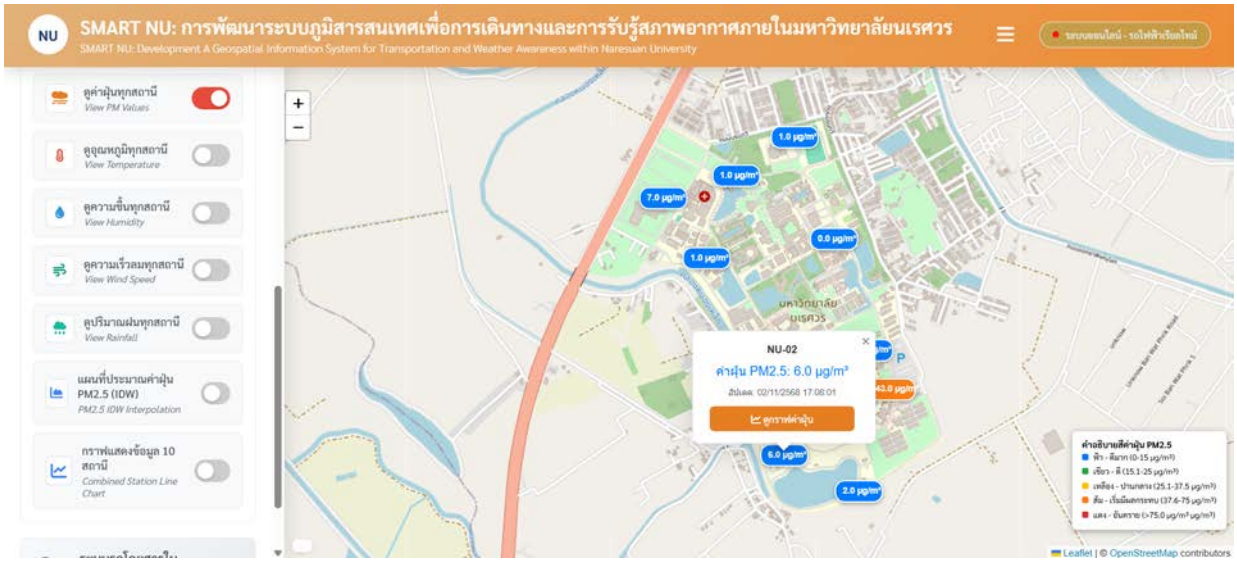
5. ปริมาณฝน สามารถเลือกดูค่าปริมาณฝน (mm) ของแต่ละสถานี

ตำแหน่งสถานีตรวจสอบสภาพอากาศ ตำแหน่งของสถานีตรวจวัดทั้ง 10 สถานีภายในมหาวิทยาลัยจะถูกแสดงบนแผนที่ในรูปของ Marker เมื่อผู้ใช้งานคลิกที่ Marker ของสถานีใดสถานีหนึ่งบนแผนที่ (เช่น สถานี NU-03), จะปรากฏ Popup แสดงข้อมูลโดยละเอียดของสถานีนั้น ๆ ข้อมูลเรียลไทม์ที่แสดงใน Popup ชื่อสถานี อุณหภูมิ ความชื้น ค่าฝุ่น PM2.5 ความเร็วลม วันและเวลาอัปเดตล่าสุด พิกัด (ละติจูด, ลองจิจูด) และปุ่มสำหรับเข้าถึงกราฟ



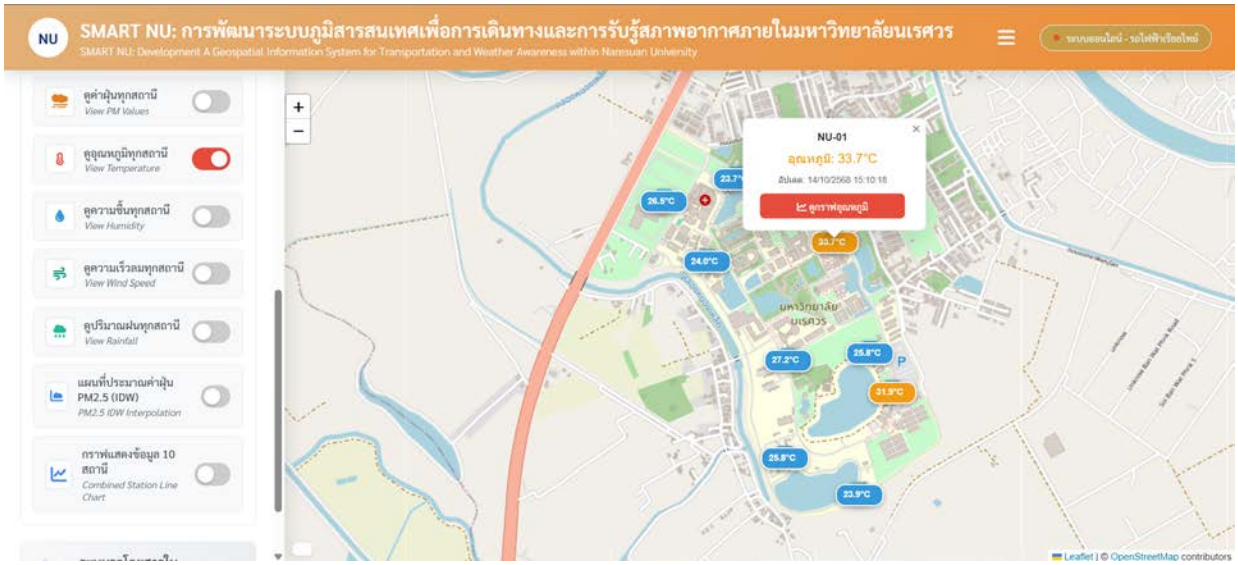
ภาพที่ 38 ตำแหน่งสถานีตรวจสอบสภาพอากาศ

การแสดงผลค่าฝุ่น PM2.5 เมื่อผู้ใช้งานเปิดดูค่าฝุ่น Marker ของสถานีจะแสดงบนแผนที่และสีของ Marker จะถูกกำหนดแบบเรียลไทม์ตามระดับค่าฝุ่น PM2.5 ที่ตรวจวัดได้ ระบบมีคำอธิบายสีของค่าฝุ่น (Legend) ตามเกณฑ์คุณภาพอากาศไทย เพื่อให้ผู้ใช้รับรู้สถานการณ์ได้อย่างรวดเร็ว (ฟ้าดีมาก, เขียวดี,เหลืองปานกลาง,ส้มเริ่มมีผลกระทบ, แดงอันตราย)



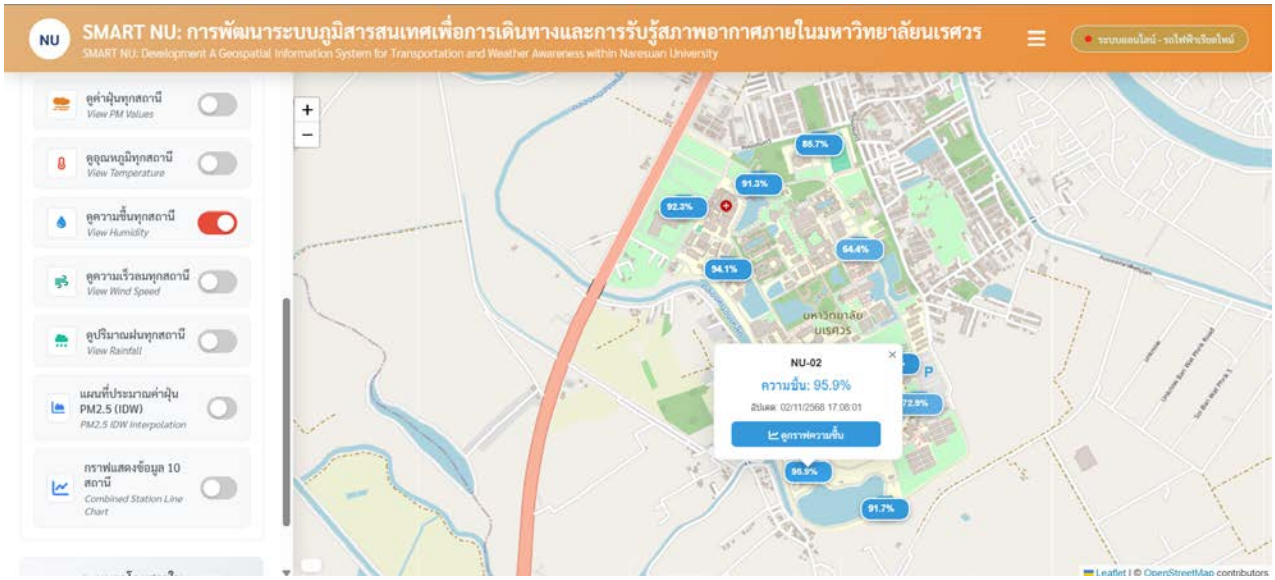
ภาพที่ 39 การแสดงผลค่าฝุ่น PM2.5

การแสดงผลค่าอุณหภูมิ เมื่อผู้ใช้งานเปิดสวิตช์ดูอุณหภูมิทุกสถานี Marker จะแสดงผลค่าอุณหภูมิ ณ ตำแหน่งพิกัดของสถานีตรวจวัดแต่ละแห่งค่าอุณหภูมิจะแสดงผลเป็นหน่วยองศาเซลเซียส (°C) และจะใช้สีที่เหมาะสมตามช่วงค่าอุณหภูมิ เช่น สีน้ำเงินสำหรับอุณหภูมิต่ำและสีส้มสำหรับอุณหภูมิสูง



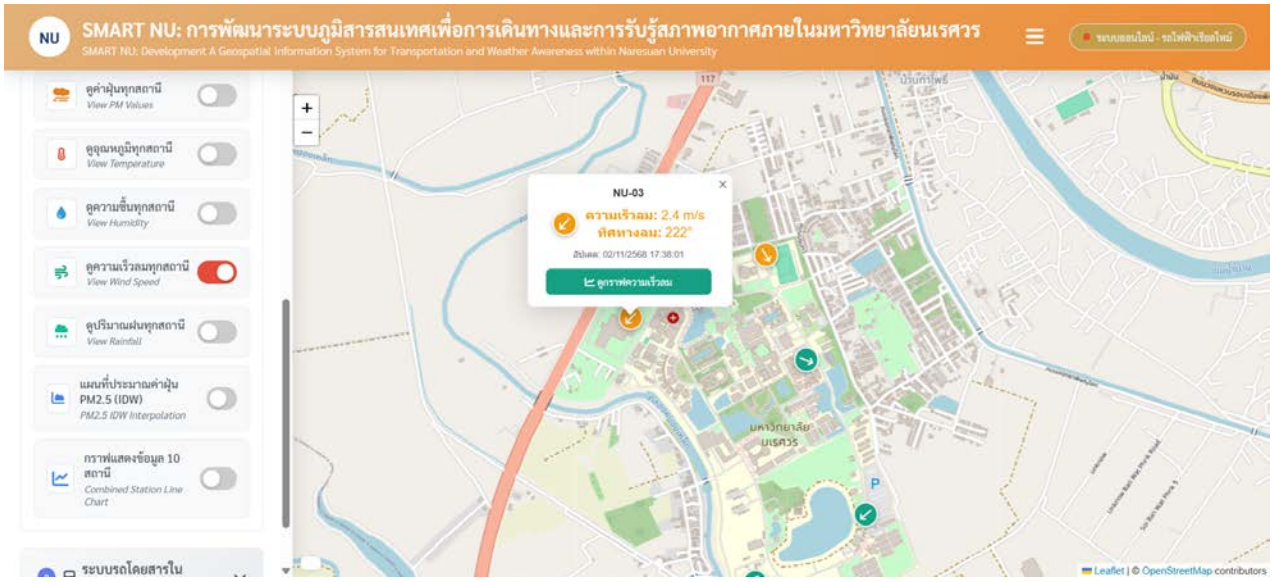
ภาพที่ 40 การแสดงผลค่าอุณหภูมิ

การแสดงผลค่าความชื้น สามารถเลือกดูค่าความชื้น (%) ที่แสดงผลเป็นตัวเลขบนแผนที่ ณ ตำแหน่งของแต่ละสถานี เมื่อคลิกที่ Marker จะมี Popup แสดงข้อมูลโดยละเอียดของสถานีนั้น ๆ พร้อมปุ่มสำหรับเข้าสู่กราฟความชื้น



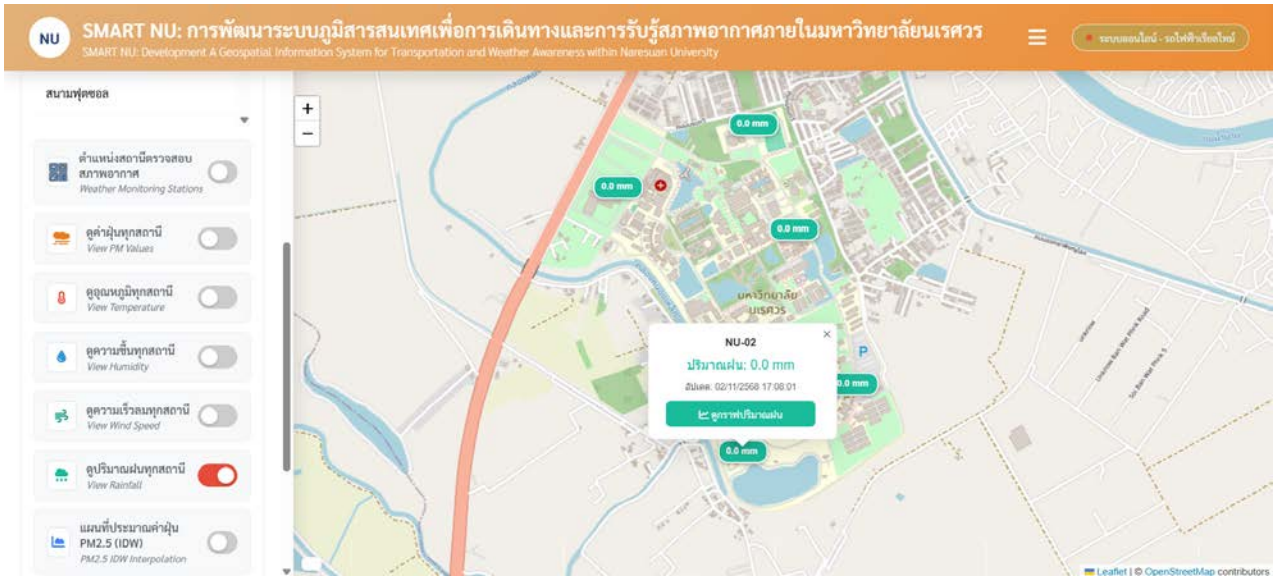
ภาพที่ 41 การแสดงผลค่าความชื้น

การแสดงผลค่าความเร็วลม สามารถเลือกดูค่าความเร็วลม (m/s) และทิศทางลมของแต่ละสถานีที่มีลักษณะเป็นลูกศรทิศทางลม ซึ่งจะถูกหมุนตามทิศทาง (องศา) ที่ตรวจวัดได้ เพื่อแสดงผลทิศทางการพัดของลมแบบเรียลไทม์ เมื่อกด Popup จะแสดงข้อมูลความเร็วลมและทิศทางลมเป็นตัวเลขพร้อมหน่วย



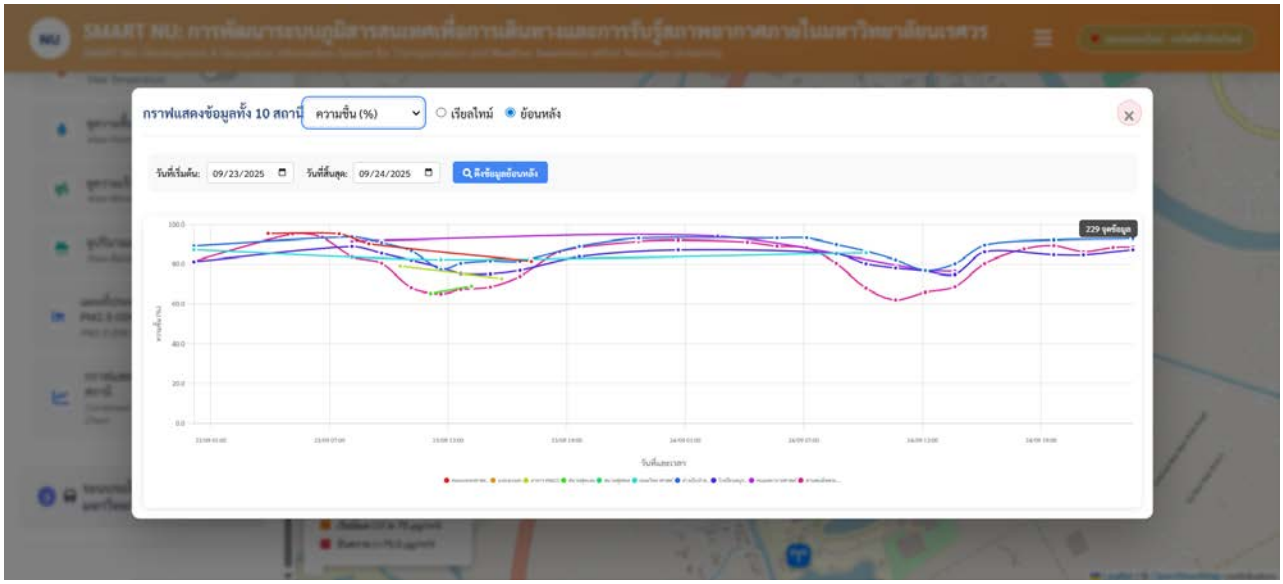
ภาพที่ 42 การแสดงผลค่าความเร็วลม

การแสดงผลค่าปริมาณฝน สามารถเลือกดูค่าปริมาณฝน (mm) ที่แสดงผลเป็นตัวเลขบนแผนที่ Marker ณ ตำแหน่งของแต่ละสถานีเมื่อคลิกที่ Marker จะมี Popup แสดงข้อมูลโดยละเอียดของสถานีนั้น ๆ พร้อมปุ่มสำหรับเข้าสู่กราฟปริมาณฝนย้อนหลัง



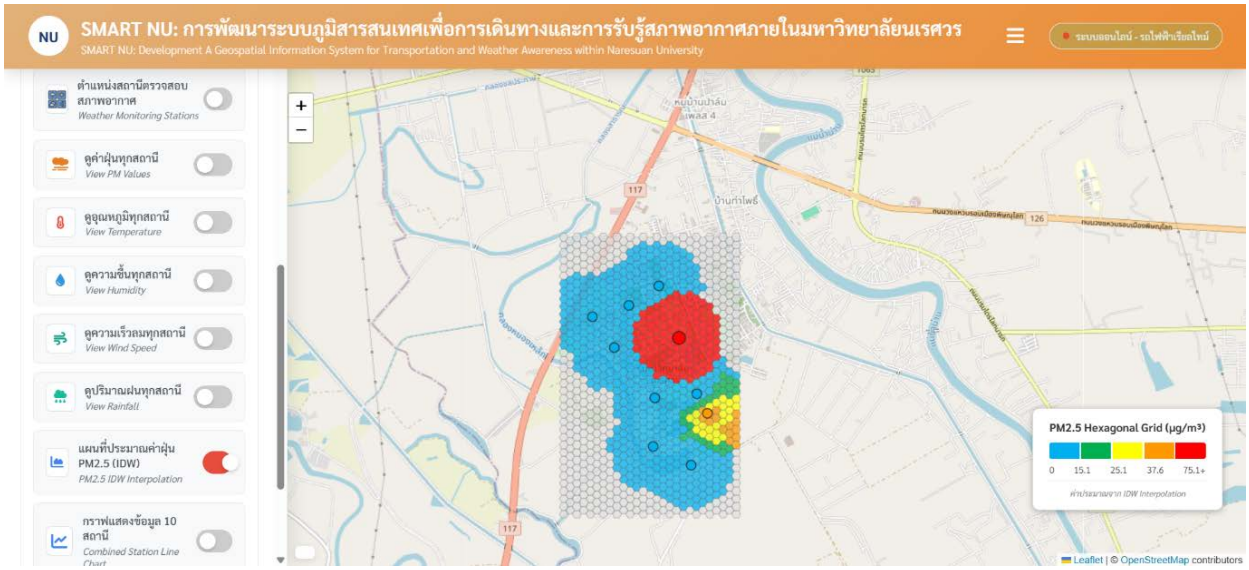
ภาพที่ 43 การแสดงผลค่าปริมาณฝน

กราฟแสดงข้อมูลทั้ง10 สถานี กราฟแสดงข้อมูลแบบเรียลไทม์และย้อนหลังที่ให้ผู้ใช้งานสามารถเลือกวันเดือนที่ต้องการดูได้



ภาพที่ 44 กราฟแสดงข้อมูลทั้ง10 สถานี

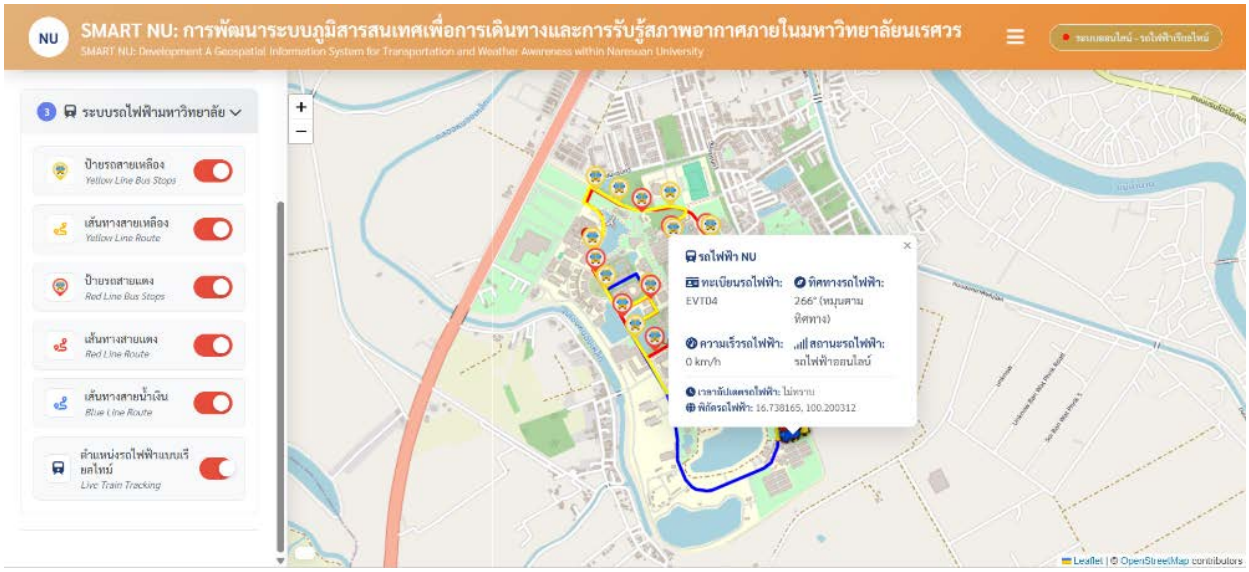
แผนที่ประมาณค่าฝุ่น PM2.5 IDW (Inverse Distance Weighting) เป็นวิธีการประมาณค่าเชิงพื้นที่ที่ใช้ในการคำนวณค่าจากจุดข้อมูลใกล้เคียง (สถานีตรวจวัด) โดยให้น้ำหนักผกผันตามระยะทางและผลลัพธ์ของการประมาณค่านี้จะถูกแสดงผลบนแผนที่ในรูปแบบของตารางหกเหลี่ยม (Hexagonal Grid)



ภาพที่ 45 แผนที่ประมาณค่าฝุ่น PM2.5 IDW

4.4 หมวดหมู่ระบบรถโดยสารในมหาวิทยาลัย

ระบบนี้ดึงข้อมูลตำแหน่งรถไฟฟ้าแบบเรียลไทม์ผ่าน API การแสดงผลข้อมูลจะถูกแยกออกเป็นเลเยอร์ต่างๆ บนแผนที่เพื่อให้ผู้ใช้เลือกเปิดดูได้ เส้นทางรถวิ่งแบ่งตามสายสี ได้แก่ สายสีเหลือง, สายสีแดง, และสายสีน้ำเงิน ป้ายรถโดยสาร แสดงตำแหน่งของป้ายที่จอดรับส่งผู้โดยสาร โดยแยกเป็น ป้ายสีเหลือง และป้ายสีแดงและแสดงตำแหน่งปัจจุบันของรถไฟฟ้าแบบเรียลไทม์



ภาพที่ 46 ตำแหน่งรถโดยสารในมหาวิทยาลัย

บทที่ 5

สรุปผลการวิจัยและอภิปราย

5.1 สรุปผลการวิจัย

ผลการวิจัยเรื่อง “SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร” พบว่าระบบที่พัฒนาขึ้นสามารถตอบสนองวัตถุประสงค์ที่กำหนดไว้ได้ครบถ้วน ได้แก่ 1.ระบบนำทางอัจฉริยะ ระบบสามารถคำนวณเส้นทางที่สั้นที่สุดได้อย่างถูกต้อง โดยประยุกต์ใช้ข้อมูลโครงข่ายถนนจาก OpenStreetMap ร่วมกับฟังก์ชันของ pgRouting และใช้อัลกอริทึม Dijkstra บนฐานข้อมูล PostgreSQL/PostGIS ในการวิเคราะห์และประมวลผลเส้นทาง ทำให้ผู้ใช้สามารถเลือกเส้นทางการเดินทางภายในมหาวิทยาลัยได้อย่างรวดเร็ว และช่วยลดความผิดพลาดในการเดินทาง โดยเฉพาะนิสิตใหม่และผู้มาเยือน นอกจากนี้ยังมีการแยกประเภทเส้นทางให้เหมาะสมกับรูปแบบการเดินทาง เช่น เส้นทางคนเดิน เส้นทางรถยนต์ เส้นทางรถมอเตอร์ไซค์ และเส้นทางจักรยาน เพื่อให้การแนะนำเส้นทางสอดคล้องกับบริบทพื้นที่จริง 2.ประสิทธิภาพของระบบรับรู้สภาพอากาศแบบเรียลไทม์ ระบบสามารถแสดงข้อมูลสภาพอากาศแบบเวลาจริงจากเซนเซอร์ภายในมหาวิทยาลัยได้อย่างถูกต้อง ผ่านการเชื่อมต่อด้วย Node-RED และจัดเก็บข้อมูลในฐานข้อมูล PostgreSQL ซึ่งผู้ใช้สามารถตรวจสอบค่าฝุ่นละออง PM2.5 อุณหภูมิ ความชื้น ความเร็วลม และปริมาณน้ำฝนได้ตามเวลาจริง ตอบสนองต่อการใช้งานเชิงสิ่งแวดล้อมอย่างมีประสิทธิภาพ การนำเสนอข้อมูลผ่านระบบ Web Map Application ที่พัฒนาด้วย PHP ร่วมกับ Leaflet ทำให้การแสดงผลข้อมูลเชิงพื้นที่มีความถูกต้อง สวยงาม และใช้งานง่ายโดยไม่ต้องติดตั้งโปรแกรมเพิ่มเติม พร้อมทั้งรองรับการแสดงสถานการณ์คุณภาพอากาศด้วยรหัสสีตามมาตรฐานของประเทศไทย รวมถึงมีการประยุกต์ใช้เทคนิค Interpolation แบบ IDW เพื่อแสดงแผนที่การกระจายตัวของค่าฝุ่น PM2.5 ให้ครอบคลุมทุกพื้นที่ 3.การบูรณาการระบบขนส่งภายในมหาวิทยาลัย ระบบสามารถบูรณาการข้อมูลตำแหน่งรถโดยสารภายในมหาวิทยาลัยแบบเรียลไทม์ ทำให้ผู้ใช้งานสามารถติดตามตำแหน่งรถได้บนแผนที่และวางแผนการขึ้นรถได้อย่างเหมาะสมส่งผลให้การเดินทางด้วยรถโดยสารสาธารณะมีความต่อเนื่อง คล่องตัวและลดเวลารอรถของผู้ใช้งานยกระดับคุณภาพการเดินทางภายในพื้นที่มหาวิทยาลัย ความสามารถในการเข้าถึงระบบปัจจุบัน ระบบได้ทำให้เข้าใช้งานได้ทั้งมือถือและคอมพิวเตอร์ โดยมีการออกแบบส่วนติดต่อผู้ใช้งาน (User Interface) ซึ่งปรับเปลี่ยนหน้าตาและจัดเรียงองค์ประกอบต่าง ๆ ให้เหมาะสมกับการแสดงผลบนจอภาพขนาดเล็ก เช่น สมาร์ทโฟนหรือแท็บเล็ต และจอภาพขนาดใหญ่คอมพิวเตอร์ตั้งโต๊ะทำให้ผู้ใช้สามารถเข้าถึงบริการนำทางและข้อมูลสภาพอากาศได้ทุกที่ทุกเวลาผ่านเว็บเบราว์เซอร์

5.2 อภิปรายผลการวิจัย

อภิปรายผลการวิจัย SMART NU เป็นระบบที่ผสานเทคโนโลยี Web GIS เข้ากับระบบตรวจวัดสภาพอากาศแบบเรียลไทม์เพื่อเพิ่มประสิทธิภาพในการเดินทางและสนับสนุนการเข้าถึงข้อมูลสิ่งแวดล้อมภายในมหาวิทยาลัยนเรศวร ผลการวิจัยสะท้อนให้เห็นถึงศักยภาพของระบบในการตอบสนองความต้องการของผู้ใช้งานในหลายมิติ ดังนี้

1. ด้านระบบนำทางและการประมวลผลเชิงพื้นที่ ระบบสามารถคำนวณและนำเสนอเส้นทางภายในมหาวิทยาลัยได้อย่างถูกต้อง โดยใช้งานฐานข้อมูลโครงข่ายถนนจาก OpenStreetMap ร่วมกับการวิเคราะห์เส้นทางด้วยฟังก์ชัน pgr_dijkstra ใน pgRouting บน PostgreSQL/PostGIS ซึ่งช่วยให้ระบบสามารถประมวลผลได้รวดเร็วและรองรับการใช้งานแบบเวลาจริง การจัดแบ่งประเภทเส้นทางตามรูปแบบการเดินทาง เส้นทางคนเดิน เส้นทางรถยนต์ เส้นทางรถมอเตอร์ไซค์ และเส้นทางจักรยาน ทำให้ผลลัพธ์มีความเหมาะสมกับข้อจำกัดพื้นที่จริง ลดความสับสนในการเดินทางของผู้ใช้งาน และเพิ่มความสะดวกในการเข้าถึงสถานที่ต่างๆ ภายในมหาวิทยาลัยนเรศวร

2. ด้านการรับรู้สภาพอากาศและการประยุกต์ใช้เทคโนโลยี IoT ระบบสามารถดึงข้อมูลจากเซนเซอร์ตรวจวัดสิ่งแวดล้อมหลายสถานีภายในพื้นที่แบบเวลาจริงผ่าน Node-RED และบันทึกลงในฐานข้อมูลได้อย่างมีเสถียรภาพ การนำเสนอข้อมูล PM2.5 อุณหภูมิ ความชื้น ปริมาณฝน และความเร็วลม ทำให้ผู้ใช้สามารถรับรู้สถานการณ์สิ่งแวดล้อมได้ทันที โดยมีการประยุกต์ใช้เทคนิค IDW Interpolation เพื่อแสดงการกระจายตัวของค่าฝุ่น PM2.5 และการใช้ Color Coding ตามเกณฑ์คุณภาพอากาศของประเทศไทย ซึ่งช่วยสนับสนุนการตัดสินใจในการเดินทางและกิจกรรมกลางแจ้งอย่างเหมาะสม

3. ด้านการบูรณาการระบบและประสบการณ์การใช้งาน ระบบสามารถแสดงตำแหน่งรถโดยสารภายในมหาวิทยาลัยแบบเรียลไทม์บนแผนที่ ทำให้ผู้ใช้สามารถวางแผนการขึ้นรถได้รวดเร็วยิ่งขึ้นและลดเวลารอคอย การสร้าง Web Map Application ด้วย Leaflet และ PHP ช่วยให้ระบบมีความยืดหยุ่นในการปรับปรุงต่อยอด รองรับการเข้าถึงผ่านเว็บเบราว์เซอร์ได้ทันทีโดยไม่ต้องติดตั้งโปรแกรมเพิ่มเติม ส่งผลต่อความสะดวกในการใช้งานของนิสิต บุคลากร และผู้มาเยือนโดยระบบใช้เวลาเฉลี่ยในการประมวลผลไม่เกิน 2 วินาที

5.3 ข้อเสนอแนะ

1. ควรเพิ่มฟังก์ชันการใช้งานที่ทำให้ผู้ใช้สามารถเห็นตำแหน่งปัจจุบันของตนเองได้บนแผนที่โดยไม่ต้องเลือกตำแหน่ง การแสดงตำแหน่งปัจจุบันโดยอัตโนมัติ (ผ่าน Geolocation API ของเบราว์เซอร์) จะช่วยให้การใช้งานหน้าเว็บง่ายขึ้น และเป็นประโยชน์อย่างยิ่งสำหรับผู้ที่ไม่ทราบตำแหน่งปัจจุบันของตนเอง เช่น บุคคลภายนอก หรือผู้มาเยือนที่ไม่คุ้นเคยกับพื้นที่มหาวิทยาลัยซึ่งจะสามารถนำตำแหน่งนั้นไปใช้เป็นจุดเริ่มต้นของการค้นหาเส้นทางได้ทันที

2. เพิ่มจำนวนสถานีตรวจวัดให้เยอะและครอบคลุมยิ่งขึ้นควรพิจารณาเพิ่มจำนวน สถานีตรวจวัดสภาพอากาศ (เซนเซอร์ IoT) ให้ครอบคลุมพื้นที่ที่ยังขาดข้อมูลหรือบริเวณที่มีความอ่อนไหวสูง เช่น พื้นที่ใกล้แหล่งจราจรหนาแน่น หรือพื้นที่พักอาศัยของนิสิต

บรรณานุกรม

อรัญญา จันแดง (2023) **"OnTimeBus: แอปพลิเคชันตรวจสอบตำแหน่งรถบัสตามเวลาจริงและเวลาที่มาถึง"**คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อม มหาวิทยาลัยนเรศวร วิทยานิพนธ์ระดับปริญญาตรี สาขาวิชาภูมิศาสตร์ https://ww2.agi.nu.ac.th/nred/Document/isPDF/2566/geo_2566_011_FullPaper.pdf

ธัญพร ศรีดอกไม้(2010) **“ระบบค้นหาอาคารสถานที่และบุคลากรภายในมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือโดยใช้พื้นฐานทาง GIS”** คณะเทคโนโลยีสารสนเทศมหาวิทยาลัยเทคโนโลยีพระ จอมเกล้าพระนครเหนือ https://ph01.tci-thaijo.org/index.php/IT_Journal/article/view/73235

เนื่องวงศ์ ทวยเจริญ (2018) **“การพัฒนาโปรแกรมประยุกต์บนเว็บและเว็บเซอร์วิสเพื่อบันทึกและค้นหาเส้นทางขนส่งสาธารณะในกรุงเทพมหานคร”** มหาวิทยาลัยธุรกิจบัณฑิตย์ https://libdoc.dpu.ac.th/research/Nuengwong.Tua.61.pdf

สุภัสสรา เมืองนันต์ (2022) **“การพัฒนาระบบฐานข้อมูลและระบบค้นหาเส้นทางในการเดินทางภายในมหาวิทยาลัยนเรศวร”** คณะเกษตรศาสตร์ทรัพยากรธรรมชาติและสิ่งแวดล้อม มหาวิทยาลัยนเรศวร วิทยานิพนธ์ระดับปริญญาตรี สาขาวิชาภูมิศาสตร์ https://ww2.agi.nu.ac.th/nred/Document/is-PDF/2565/geo_2565_011_FullPaper.pdf

Abdullah Alattas(2022) **“3D pgRouting and Visualization in Cesium JS Using the Integrated Model of LADM and IndoorGML”** https://gdmc.nl/3dcadastre/literature/3Dcad_2021_29.pdf

Taranjot Singh Bhatia, Harpinder Singh, P.K Litoria, Brijendra Pateriya(2018) **“Web GIS Development using Open Source Leaflet and Geoserver Toolkit”** https://www.researchgate.net/publication/326971834_Web_GIS_Development_using_Open_Source_Leaflet_and_Geoserver_Toolkit

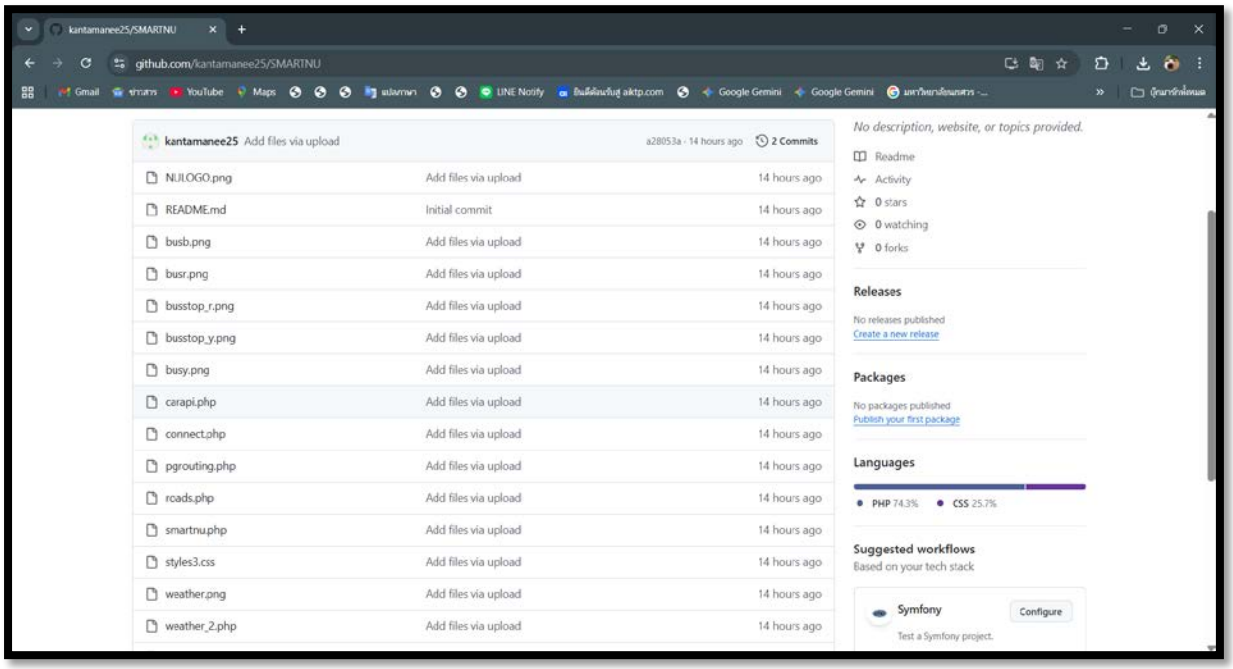
ภาคผนวก

ภาคผนวก

ภาคผนวกโค้ดที่ใช้ในการพัฒนาระบบ

ผู้วิจัยได้เผยแพร่โค้ดต้นฉบับทั้งหมดที่ใช้ในการพัฒนาระบบภูมิสารสนเทศ “SMART NU: การพัฒนาระบบภูมิสารสนเทศเพื่อการเดินทางและการรับรู้สภาพอากาศภายในมหาวิทยาลัยนเรศวร” บนแพลตฟอร์ม GitHub เพื่อให้สามารถตรวจสอบอ้างอิง และนำไปต่อยอดการพัฒนาได้ การเผยแพร่โค้ดในรูปแบบดิจิทัลนี้ครอบคลุมไฟล์ที่เกี่ยวข้องทั้งหมด ได้แก่ ไฟล์สำหรับส่วนหน้า (Frontend: HTML, CSS, JavaScript/Leaflet) และไฟล์สำหรับส่วนหลัง (Backend: PHP/SQL Query Logic) รวมถึงไฟล์ฐานข้อมูลเริ่มต้นที่ใช้ในการพัฒนาระบบ

ที่อยู่ (URL) สำหรับการเข้าถึงโค้ด https://github.com/kantamanee25/SMARTNU



ภาพที่ 47 โค้ดที่ใช้ในการพัฒนาระบบ

ประวัติผู้วิจัย

ชื่อ – ชื่อสกุล กัณฐมณี วรอินทร์

วัน เดือน ปี เกิด 2 พฤษภาคม 2546

ที่อยู่ปัจจุบัน 31 หมู่ 6 ตำบลเขาท่าพระ อำเภอเมืองชัยนาท จังหวัดชัยนาท 17000

ประวัติการศึกษา

พ.ศ. 2565 – ปัจจุบัน ระดับอุดมศึกษา วท.บ. สาขาภูมิศาสตร์ มหาวิทยาลัยนเรศวร อำเภอเมืองพิษณุโลก จังหวัดพิษณุโลก 65000 เกรดเฉลี่ย 3.43

พ.ศ. 2562 – 2564 ระดับมัธยมศึกษาตอนปลาย (วิทย์-คณิต) โรงเรียนองค์การบริหารส่วนจังหวัดชัยนาท ต.เขาท่าพระ อ.เมืองชัยนาท จ.ชัยนาท 17000

เกรดเฉลี่ย 3.57

พ.ศ. 2559 – 2561 ระดับมัธยมศึกษาตอนต้น (วิทย์-คณิต) โรงเรียนองค์การบริหารส่วนจังหวัดชัยนาท ต.เขาท่าพระ อ.เมืองชัยนาท จ.ชัยนาท 17000

พ.ศ. 2553 – 2558 ระดับประถมศึกษา โรงเรียนเทศบาลตำบลหางน้ำสาคร ต.หางน้ำสาคร อ.มโนรมย์ จ.ชัยนาท 17170

กิจกรรมที่เข้าร่วม

1) ดำรงตำแหน่งประธานค่าย จัดเตรียมโครงการค่ายภูมิศาสตร์ ณ อุทยานแห่งชาติภูหินร่องกล้า อำเภอนครไทยจังหวัดพิษณุโลก พ.ศ.2566

2) ดำรงตำแหน่งประธานชมรมภูมิศาสตร์ พ.ศ.2567

3) เข้าร่วมอบรมการใช้เครื่องมือ GNSS คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

4) เข้าร่วมงาน GeoHackathon ณ หมู่บ้านนครชุม อำเภอนครไทย จังหวัดพิษณุโลก

5) ลงพื้นที่สำรวจการเรียนรู้นอกห้องเรียนด้วยการออกภาคสนามจริง สถานีปฏิบัติการบึงราชนก โดยใช้ Trimble GNSS RTK ทำการปักพิกัด GCPs และถ่ายภาพจากโดรน Multispectral Camera อำเภอวังทอง จังหวัดพิษณุโลก

กิจกรรมที่เข้าร่วม (ต่อ)

6) เข้าร่วมงานประชุมสมาคมภูมิศาสตร์แห่งประเทศไทย

7) เข้าร่วมโครงการพัฒนาความรู้และทักษะภาษาอังกฤษเพื่อการเรียนรู้ภูมิศาสตร์

8) อบรมเชิงปฏิบัติการ หัวข้อ “IoT และ LoRawan Gateways”

9) เข้ารับฟัง Special Seminar Theme: From Climate Change to Sustainable Futures: Geoinformatics in Action for SDGs ณ มหาวิทยาลัยนเรศวร

10) เข้ารับฟังบรรยายพิเศษหัวข้อ “Leveraging Geospatial Technologies Using Free & Open Source Software and Open Data” และพูดคุยแลกเปลี่ยนความร่วมมือทางวิชาการระหว่างมหาวิทยาลัยนเรศวรและ Osaka Metropolitan University (OMU) ณ มหาวิทยาลัยนเรศวร

11) ช่วยงานสโมสรนิสิต ฝ่ายโสตทัศนศึกษา

12) เข้าร่วมการแข่งขันโครงการมหกรรมกีฬา (เสลาเกมส์) ประเภทกีฬาฟุตซอล

รางวัลที่ได้รับ

1) เกียรติบัตรด้านผลการเรียนดี ประจำปีการศึกษา 2566 สาขาภูมิศาสตร์

2) เกียรติบัตรด้านผลการเรียนดี ประจำปีการศึกษา 2567 สาขาภูมิศาสตร์

3) เกียรติบัตรด้านผลการเรียนดีเด่น ประจำปีการศึกษา 2567 สาขาภูมิศาสตร์

4) เกียรติบัตรด้านผลการเรียนดี ประจำปีการศึกษา 2567 โดยมหาวิทยาลัยนเรศวร

5) เกียรติบัตรด้านผลการเรียนดีเด่น ประจำปีการศึกษา 2567 โดยมหาวิทยาลัยนเรศวร

6) ได้รับรางวัลรองชนะเลิศอันดับหนึ่ง กิจกรรมการประกวดนวัตกรรมและโครงงานด้านการเกษตร วันที่ 22 กันยายน 2568