CN332 - PROJECT

Project overview

ชื่อโปรเจกต์: ระบบบริหารจัดการนิติบุคคลอัจฉริยะ (Smart Juristic AI Agent)

1.แนวคิดและหลักการ (Core Concept)

พัฒนาระบบบริหารจัดการที่พักอาศัย (หมู่บ้าน) ที่เน้นการโต้ตอบผ่านอินเทอร์เฟซที่ผู้ใช้งานคุ้นเคยที่สุด คือ LINE Messenger โดยใช้เทคโนโลยี Artificial Intelligence (AI) เป็นตัวกลางในการคัดกรอง วิเคราะห์ และจัดการข้อมูลโดยอัตโนมัติ เพื่อลดภาระงานของเจ้าหน้าที่นิติบุคคลและเพิ่มความสะดวกสบายให้แก่ลูกบ้าน

2.โครงสร้างสถาปัตยกรรมระบบ (System Architecture)

ระบบถูกออกแบบโดยแบ่งออกเป็น 3 ส่วนหลัก (Three-Tier Architecture):

ส่วนที่ 1: Resident & Technician Interface (Frontend - LINE OA)

เน้นการเข้าถึงง่าย (Accessibility) โดยใช้ LINE Official Account เป็นช่องทางหลักสำหรับลูกบ้านและช่าง

ส่วนที่ 2: Admin Dashboard (Frontend - Web Application)

ระบบบริหารจัดการสำหรับเจ้าหน้าที่นิติบุคคล เพื่อควบคุมและติดตามภาพรวมของโครงการ

ส่วนที่ 3: Smart Central Core (Backend - Django & AI Integration)

ส่วนประมวลผลหลักที่ใช้ Framework Django (Python) ในการเชื่อมต่อฐานข้อมูลและโมเดลปัญญาประมวลผล

AI Intelligence Capabilities:

- NLP Support (Juristic Bot): ใช้ Natural Language Processing เพื่อตอบคำถามจากระเบียบข้อบังคับของหมู่บ้าน หรือคู่มือการใช้งานเบื้องต้นได้ตลอด 24 ชม.
- Auto-Categorization: AI วิเคราะห์รูปภาพและข้อความที่ลูกบ้านส่งมา เพื่อแยกประเภทงาน (เช่น ไฟฟ้า, ประปา, โครงสร้าง) และประเมินความเร่งด่วนโดยอัตโนมัติ
- Speech-to-Text Conversion: แปลงคำสั่งเสียงจากลูกบ้านเป็นข้อความบันทึกในระบบ Ticket โดยตรง

โดยมีระบบประมวลผลกลางที่พัฒนาด้วย Django Framework โดยบูรณาการเทคโนโลยีปัญญาประมวลผล (AI) เพื่อวิเคราะห์คัดกรองข้อมูล แจกจ่ายงานอัตโนมัติ และให้บริการโต้ตอบข้อมูลอัจฉริยะแบบ Real-time

3.วัตถุประสงค์ของโครงการ (Project Objectives)

• เพื่อพัฒนาระบบบริหารจัดการที่พักอาศัยอัจฉริยะ ที่บูรณาการเทคโนโลยีปัญญาประดิษฐ์ (AI) เข้ากับระบบหลังบ้าน (Django) เพื่อเพิ่มประสิทธิภาพในการดำเนินงานของนิติบุคคลและลดความผิดพลาดที่เกิดจากมนุษย์ (Human Error)

• เพื่อเพิ่มความสะดวกและรวดเร็วในการสื่อสารระหว่างลูกบ้านและนิติบุคคล ผ่านแพลตฟอร์ม LINE Messaging API ซึ่งช่วยให้การแจ้งซ่อมและการรับทราบข้อมูลข่าวสารเป็นไปอย่างง่ายดายและเข้าถึงได้ทุกที่ทุกเวลา

• เพื่อสร้างระบบการจัดการงานซ่อมและข้อร้องเรียนแบบอัตโนมัติ (Automated Ticket & Tracking) ที่สามารถวิเคราะห์ประเภทปัญหาและติดตามสถานะการดำเนินงานได้อย่างเป็นระบบและโปร่งใส

• เพื่อนำเทคโนโลยีการประมวลผลภาษาธรรมชาติ (NLP) มาใช้ในการให้บริการตอบคำถามและให้ข้อมูลเบื้องต้นแก่ลูกบ้านตลอด 24 ชั่วโมง เพื่อลดภาระการตอบคำถามซ้ำซ้อนของเจ้าหน้าที่นิติบุคคล

Use case and scenario

1. Line OA

Use Cases

Category A: ถ่ายรูปแจ้งซ่อม, รายงานปัญหา เช่น แจ้งรถจอดขวางทาง ปัญหาขยะรายงานปัญหาข้างบ้าน, แจ้งขอติดต่อนิติโดยตรง, แจ้งขอใช้สถานที่หรืออาจใช้เสียงในการทำกิจกรรมใดๆ

Scenario

สถานการณ์: ข้างบ้านส่งเสียงดังรบกวนตอนกลางดึก เพื่อให้รปภหรือนิติมาเตือน

User: ลูกบ้านพิมมาใน Line: บ้านเลข14/7(สมมติ)ส่งเสียงดังมากนอนไม่หลับ

Ai: อ่านข้อความและตีความว่าลูกบ้านต้องการอะไรแล้วควรจะติดต่อ. ฝ่ายไหน

Result: ติดต่อยามให้ลูกบ้าน

Category B: ลูกบ้านต้องการทราบถึงสถานการณ์ซ่อมแซม

Scenario

User: "เรื่องไฟกิ่งที่แจ้งไป ถึงไหนแล้ว?"

AI: ค้นหา Ticket ID จากประวัติแชทของ User คนนี้ แล้วไปดึงข้อมูลจาก Database

Result: ตอบกลับว่า "ตอนนี้ช่างกำลังเบิกหลอดไฟครับ คาดว่าจะซ่อมเสร็จภายใน 16:00 น. วันนี้ครับ"

Category C: การให้บัตรผ่านเข้าออกหมู่บ้าน

Scenario

User: "พรุ่งนี้แม่จะมาหาตอน 10 โมง ทะเบียน กก-1234"

AI: ขอหลักฐานยืนยันตัวตน เช่น บัตรประชาชน บันทึกข้อมูลบัตรประชาชนข้อมูลทะเบียนรถลงระบบ White-list ของไม้กั้น

Result: ส่ง QR Code สำหรับผู้มาติดต่อ (Visitor Pass) ให้ลูกบ้าน เพื่อส่งต่อให้แม่ เมื่อรถแม่มาถึง แม่ไม้จะนำQR codeไปสกนที่ทางเข้าหมู่บ้านและไม้กั้นจะเปิดเองอัตโนมัติ พร้อมส่งไลน์บอกลูกบ้านว่า “คุณแม่มาถึงแล้วครับ”

Category D: การชำระเงินค่าส่วนกลางของหมู่บ้าน

Scenario

User: ส่งรูปสลิปโอนค่าส่วนกลางเข้าแชท

AI: ใช้ OCR (Optical Character Recognition) ดึงข้อมูล วันที่, เวลา, จำนวนเงิน, และ QR Code จากสลิป ไปเทียบกับ API ของธนาคาร

Result: หากยอดตรง AI จะบันทึกบัญชี และส่งใบเสร็จ E-Receipt กลับมาให้ในแชททันที ภายใน 3 วินาที

2.Detected maintenance

Use Cases

Category A: เซ็นเซอร์ตรวจสอบเก็บข้อมูลของท่อน้ำประปากลางที่ใช้กันในหมู่บ้าน โดยจะเก็บข้อมูลที่ได้ลงใน Database โดยใช้ Django จากนั้น import library pytorch เข้ามาเพื่อวิเคราะห์ข้อมูลจากเซ็นเซอร์วัดน้ำที่เราได้มานำไปanalysis และ Machine Learning (LSTM Autoencoder) บน PyTorch ว่าน้ำที่ใช้มีสิ่งผิดปกติหรือไม่ หรือคาดการณ์ว่าในอนาคตน้ำที่ใช้มีสิ่งผิดปกติหรือไม่ก็ได้เช่นกันโดยการAnomaly Detection แบบdynamic threshold และ Predictive Maintenance ถ้ามีก็จะส่งการแจ้งเตือนไปที่ลูกบ้านโดยSmart Juristic AI Agent

Scenario

สถานการณ์: น้ำมีสารปนเปื้อน
การใช้งาน: เซ็นเซอร์วัดค่าน้ำส่งเข้าdatabase จากนั้นpytorchวิเคราะห์ข้อมูล ประมวลผลกราฟข้อมูลสถิติต่างๆที่ส่งมา พบว่าน้ำที่ใช้วันนี้มีสารปนเปื้อน
AI: รับข้อมูลจากpytorchและส่งไปยังลูกบ้าน
Result: ติดต่อแจ้งเตือนลูกบ้านอัตโนมัติด้วยผ่านline OA

Category B: การเฝ้าระวังระดับน้ำและอัตราการไหล (Real-time Monitoring)
1. รับค่าระดับน้ำจากเซ็นเซอร์ทุกจุดแบบ Real-time
2. คำนวณ อัตราการเปลี่ยนแปลง ว่าน้ำสูงขึ้นกี่ ซม./นาที
3. แสดงผลบนแผนที่หมู่บ้าน โดยใช้สีแบ่งความรุนแรง (เช่น เขียว=ปกติ, เหลือง=ขึ้นเล็กน้อย, แดง=ขึ้นเร็วมาก)
4. Monitoring Officer ดูภาพรวมบน Dashboard

Scenario

สถานการณ์: ฝนตกหนักต่อเนื่อง 30 นาทีในโซนท้ายหมู่บ้าน
การใช้งาน: เซ็นเซอร์วัดระดับน้ำที่ติดตั้งอยู่ตามจุดต่างๆ (เช่น ท้ายซอย, ประตูน้ำ) ทำการวัดค่าระดับน้ำอย่างต่อเนื่องทุกๆ 1 นาที ท่ามกลางฝนที่ตกหนัก

AI: รับค่าระดับน้ำปัจจุบันเทียบกับค่าเมื่อ 5 นาทีก่อน จากนั้นใช้ Logic คำนวณ อัตราการเปลี่ยนแปลง แล้ววิเคราะห์ พบว่าระดับน้ำสูงขึ้น 5 ซม. ภายใน 2 นาที ซึ่งเกินกว่าค่ามาตรฐานที่ตั้งไว้ว่า "ปลอดภัย" ระบบจึงระบุสถานะเป็น "Critical Warning"

Result: บนแผนที่ Dashboard ของเจ้าหน้าที่ หมุดตำแหน่งนั้นเปลี่ยนจาก สีเขียว เป็น สีแดงกระพริบ ทันที ระบบส่ง Alert แจ้งเตือนข้อความ: "ระวังน้ำท่วมฉับพลัน! โซน A ระดับน้ำเพิ่มสูงผิดปกติ" ทำให้เจ้าหน้าที่เตรียมรับมือได้ทัน

Category C: AI ตรวจจับการอุดตันด้วยเสียง

1.AI รับเสียงจากท่อระบายน้ำและตะแกรง

2.เปรียบเทียบเสียงปัจจุบันกับฐานข้อมูลเสียงปกติ

3.ถ้าเสียงเปลี่ยนไป (เช่น จากเสียงน้ำไหลซู่ๆ กลายเป็นเสียงอู้อี้) AI จะวิเคราะห์ว่า เป็นแพทเทิร์นของ "การอุดตัน" หรือไม่

4.ถ้าพบความผิดปกติ ระบบจะส่งแจ้งเตือนไปยังศูนย์ควบคุม

Scenario

สถานการณ์ : มีถุงพลาสติกและกิ่งไม้ไหลมาปิดตะแกรงรับน้ำจนมิด

การใช้งาน: ไมโครโฟนที่ติดตั้งอยู่เหนือตะแกรงรับน้ำ ทำการดักฟังเสียงสภาพแวดล้อมเป็นระยะ (เช่น อัดเสียง 3 วินาที ทุก 1 นาที) ในขณะที่น้ำกำลังไหล

AI : ให้ AI แปลงไฟล์เสียงที่อัดได้เป็นรูปภาพกราฟความถี่ จากนั้นนำภาพเสียงนั้นไปเทียบกับโมเดลเสียงน้ำไหลปกติที่เรียนรู้มา แล้วจึงตัดสินใจว่า เสียงซ่าของน้ำหายไป กลายเป็น เสียงเงียบหรือเสียงหยดน้ำเบาๆ โดยตีค่าความน่าจะเป็นว่า ท่อน้ำตัน = 92%

Result : ระบบสร้างใบงานแจ้งซ่อมอัตโนมัติ ระบุหัวข้อ "มีความเสี่ยงสิ่งกีดขวางทางน้ำ"ส่งพิกัด GPS ของตะแกรงนั้นไปยังแอปฯ ของทีมช่าง เพื่อให้รีบไปเขี่ยขยะออกก่อนที่น้ำจะท่วมถนน

3.Weather report

Use cases

Category A: Real-time Weather Quick Menu (ปุ่มกดดูรายงานด่วน) ส่วนนี้คือปุ่ม Menu ใน Line OA ที่ลูกบ้านกดปุ๊บ ข้อมูลจะเด้งขึ้นมาทันทีเมื่อลูกบ้านกดปุ่มนี้ ระบบจะส่ง Flex Message ที่อ่านง่าย

ส่วนประกอบของ Flex Message:

1. หัวข้อ (Header):

- "รายงานอากาศรอบหมู่บ้าน [ชื่อโครงการ]"
- อัปเดตล่าสุด: 18:42 น. (ใช้ Time.now ในโค้ด)

2. ส่วนข้อมูลหลัก (Body):

- อุณหภูมิ: 31°C (ดึงจาก OpenWeatherMap API พิกัดหมู่บ้าน)
- ค่าฝุ่น PM 2.5: 12 μg/m³ (สถานะ: ดีมาก/สีเขียว)
- ความชื้น: 65% (เหมาะกับการตากผ้า)

3. ส่วนคำแนะนำด่วน (Footer/Action Button):

- ปุ่ม "วิเคราะห์กิจกรรม": กดเพื่อให้ Chatbot ถามต่อว่า "วันนี้คุณลูกบ้านวางแผนจะทำอะไรครับ ให้ผมช่วยแนะนำไหม?"

(ใช้ chatbot ในการรายงานสภาพอากาศตาม prompt)

Category B: Interactive Activity

ใช้ AI เป็นตัวกลางในการรับค่าจาก API/Sensor แล้วแปลงเป็นคำแนะนำ มี AI chatbot ที่ทำหน้าที่วิเคราะห์สภาพอากาศรอบโครงการจากค่า JSON ที่ได้รับ (Temp, Humidity, PM2.5, Rain Chance) เพื่อแนะนำลูกบ้าน

สิ่งที่ต้องตอบคำถาม:

1. เป้าหมาย: วิเคราะห์ว่าสภาพอากาศ 'เหมาะกับกิจกรรมที่ลูกบ้านถามหรือไม่'
2. เกณฑ์ PM 2.5: ถ้าเกิน 37.5 (สีส้ม) ให้เตือนให้สวมหน้ากากหรือเลี่ยงกิจกรรมกลางแจ้ง
3. เกณฑ์ความร้อน: ถ้า Temp เกิน 35°C ให้แนะนำให้รอช่วงเย็น
4. เกณฑ์ฝน: ถ้า Rain Chance > 40% ให้เตือนเรื่องการพกร่มหรือเก็บผ้า
5. บอกข้อมูลที่จำเป็นทั้งหมด เพื่อให้ลูกบ้านสามารถนำไปตัดสินใจได้เอง

Scenario

1. สถานการณ์: ออกกำลังกาย outdoor

- User: "ตอนนี้ไปวิ่งรอบหมู่บ้านได้ไหม"
- Input ไปยัง AI: { "temp": 36, "pm25": 42, "rain_chance": 10 }

• AI ตอบกลับ: "ช่วงนี้ 'ไม่แนะนำ' ให้วิ่งกลางแจ้งครับ เนื่องจากอุณหภูมิค่อนข้างสูง (36°C) และค่าฝุ่น PM 2.5 อยู่ในระดับสีส้ม (42) ซึ่งอาจส่งผลต่อสุขภาพได้ และฝน (10%) อยู่ในเกณฑ์ที่รับได้ แนะนำให้ใช้ลู่วิ่งในห้อง Fitness ของสโมสรแทนซึ่งสามารถ walk in ได้เลยครับ

2. สถานการณ์: การตากผ้า

• User: "ควรจะซักผ้าตอนนี้ไหม จะแห้งทันหรือเปล่า"

• Input ไปยัง AI: { "temp": 28, "humidity": 85, "pm25": 15, "rain_chance": 65 }

• AI วิเคราะห์: ความชื้น (Humidity) สูงมาก และมีโอกาสฝนตกเกิน 40%

• Result: "วันนี้ความชื้นในอากาศค่อนข้างสูงถึง 85% และมีโอกาสฝนตกสูงถึง 65% ในช่วงบ่ายครับ แนะนำว่า 'ไม่ควรตากผ้ากลางแจ้ง' เพราะผ้าอาจมีกลิ่นอับหรือเปียกฝนได้ หากจำเป็นต้องซัก แนะนำให้ตากในที่ร่มที่มีอากาศถ่ายเท หรือใช้เครื่องอบผ้าแทนครับ

3. สถานการณ์: วางแผนที่จะไปว่ายน้ำที่สโมสร

• User: "วันนี้กะว่าจะไปว่ายน้ำที่ส่วนกลางตอนบ่ายสอง แดดร้อนไปไหม"

• Input ไปยัง AI: { "temp": 37, "pm25": 10, "rain_chance": 0 }

• AI วิเคราะห์: อุณหภูมิ (Temp) เกิน 35°C

• Result: "ขณะนี้อุณหภูมิในโครงการสูงถึง 37°C ซึ่งอยู่ในระดับที่ 'ร้อนจัด' หากไปว่ายน้ำตอนนี้ไม่แนะนำครับ ให้รอช่วงเย็นหลัง 16:30 น. เป็นต้นไป

Features

1. Line OA
   - แจ้งปัญหาภายในหมู่บ้าน และแจ้งสถานะการซ่อมแซม
   - ติดตามการชำระเงินค่าส่วนกลางของลูกบ้าน
   - ติดตาม และสร้างบัตรผ่านสำหรับ visitor
   - ให้ข้อมูล และตอบคำถามเรื่องกฎของหมู่บ้าน และเรื่องจิปาถะอื่นๆ
2. Detected maintenance
   - แจ้งนิติบุคคลเพื่อทำการลอกท่อ เมื่อตรวจพบสิ่งอุดตันในท่อ
   - ตรวจสอบคุณภาพน้ำ และแจ้งเตือนลูกบ้าน เมื่อพบสารปนเปื้อนของแต่ละบ้าน

- ใช้ข้อมูลจากเซ็นเซอร์อัลตราโซนิค สำหรับตรวจสอบระดับน้ำในท่อ และวิเคราะห์ความเร็วของน้ำที่เพิ่มสูงขึ้น เพื่อแจ้งเตือนว่าน้ำจะท่วมตัวบ้านเมื่อใด
- ตรวจสอบก๊าซพิษ หรือกลิ่นไม่พึงประสงค์ที่ออกมาจากท่อ เพื่อเฝ้าระวัง และแจ้งนิติบุคคลในการปรับปรุง
- แจ้งเตือนเส้นทางในหมู่บ้านที่น้ำท่วม และแนะนำเส้นทางใหม่ที่น้ำไม่ท่วม
- วิเคราะห์ผลกระทบต่อสิ่งของในหมู่บ้านหลังจากน้ำท่วม และความเสียหายของถนนหลังจากน้ำท่วม เพื่อให้นิติบุคคลดำเนินการซ่อมแซมต่อไป

3. Weather report
    - มีปุ่ม "เช็กอากาศหมู่บ้านตอนนี้" เพื่อดูข้อมูล อุณหภูมิ, ความชื้น และค่าฝุ่น PM 2.5 ในพื้นที่หมู่บ้านได้ทันที
    - ใช้การรายงานผลที่เข้าใจง่าย เพื่อให้ลูกบ้านเข้าใจระดับของสภาพอากาศ
        - AI ที่เป็น Advisior วิเคราะห์สภาพอากาศขณะนั้นตามคำถามลูกบ้าน
        - มีการตั้งเกณฑ์ความปลอดภัยอัตโนมัติ เช่น แจ้งเตือนเมื่ออุณหภูมิหรือค่าฝุ่น PM 2.5 เกิน
        - ลูกบ้านสามารถพิมพ์สอบถามด้วยภาษาพูด (NLP) (เช่น "ซักผ้าได้ไหม", "อากาศตอนนี้ร้อนไปหรือเปล่า")

Technology

1. Line OA

- LLM backend: langchain, langchain-community, langchain-core ทำหน้าที่เชื่อมต่อ LLM กับ Tools ต่างๆ, จัดการ Memory (จำประวัติการคุย), และทำ Chain การทำงาน
- Knowledge base: RAG ทำหน้าที่จัดเก็บกฎระเบียบของหมู่บ้าน vector database:ChromaDB เก็บข้อมูลกฎระเบียบในรูปแบบ Vector (ตัวเลข) เพื่อให้ค้นหาด้วยความหมาย (Semantic Search) ได้
- Utilities & tools:
    - Environment Variables: python-dotenv หน้าที่: เก็บ API Key และ Token ต่างๆ ในไฟล์ .env ไม่ให้หลุดไปใน Code

- Local Tunneling: ngrok (Software, ไม่ใช่ Lib) หน้าที่: จำเป็นมากตอน Dev เพราะ LINE Webhook ต้องยิงเข้า HTTPS URL (ngrok จะแปลง Localhost ของเราให้เป็น Public URL ชั่วคราว)
- Async/Task Queue: Celery + Redis (Optional แต่แนะนำสำหรับ Production)
- หน้าที่: ถ้า AI คิดนานเกิน 30 วินาที LINE จะตัดการเชื่อมต่อ เราต้องโยนงานให้ Celery ทำเบื้องหลัง แล้วค่อย Push Message กลับไปหา User เมื่อเสร็จ

2. Detected maintenance

- **Time-Series Anomaly Detection ด้วย LSTM Autoencoder**
  - ใช้ Deep Learning วิเคราะห์ข้อมูลเซ็นเซอร์แบบต่อเนื่อง
  - ตรวจจับ “รูปแบบผิดปกติ” ไม่ใช่แค่ค่าเกิน threshold ธรรมดา
- **Dynamic Threshold (Adaptive Anomaly Thresholding)**
  - Threshold ปรับเปลี่ยนตามสภาพแวดล้อมและฤดูกาลไม่ใช้ค่าคงที่ (Static Threshold)
- **Predictive Maintenance สำหรับระบบน้ำประปาชุมชน**
  - คาดการณ์ความผิดปกติล่วงหน้า
- **Hardware & IoT**
  - Sensors
    - Ultrasonic Sensor: สำหรับวัดระดับน้ำ
    - MEMS Microphone: สำหรับรับเสียงการไหลของน้ำและเสียงปั๊ม
- **AI & Data Processing**
  - Signal Processing
    - Librosa: ไลบรารี Python สำหรับแปลงไฟล์เสียงเป็นภาพ Mel-Spectrogram หรือค่า MFCCs (Mel-frequency cepstral coefficients)
  - Model Architecture
    - Convolutional Neural Network (CNN): โมเดล Deep Learning ที่ใช้จำแนกแพทเทิร์นของภาพ Spectrogram เพื่อระบุสถานะท่อน้ำว่า "ปกติ" หรือ "อุดตัน"

3. Weather report

1. Communication & Interface (Front-end)

- **LINE Messaging API:** ช่องทางหลักในการสื่อสารกับลูกบ้าน
- **LINE Rich Menu:** เมนูลัดด้านล่างหน้าจอแชท สำหรับกดเช็กสภาพอากาศด่วนหรือแจ้งปัญหา
- **LINE Flex Message:** การแสดงผลข้อมูลสภาพอากาศและสถานะพัสดุในรูปแบบ Card ที่สวยงามและอ่านง่าย (ใช้ LINE Flex Message Simulator ในการออกแบบ)

2. AI & Cognitive Logic (The Brain)

- **LLM (Large Language Model):** ใช้ GPT- 4o (OpenAI) หรือ Gemini 1.5 Flash เป็นหลักในการตีความคำถามและสร้างคำแนะนำ
- **LangChain Framework:** เครื่องมือสำหรับจัดการ AI Logic เช่น การทำ Memory, Chains (ลำดับขั้นตอนการคิด), และ Function Calling (สั่งให้ AI ไปดึงค่าอากาศมาวิเคราะห์)
- **Prompt Engineering:** การออกแบบคำสั่ง (System Prompt) เพื่อกำหนดบทบาทให้ AI เป็นผู้ช่วยนิติบุคคลและผู้เชี่ยวชาญด้านสภาพอากาศ

3. Data & APIs (Hyper-local Data Sources)

- **OpenWeatherMap API: ดึง**ข้อมูลอุณหภูมิ, ความชื้น และการพยากรณ์อากาศล่วงหน้าตามพิกัดหมู่บ้าน
- **IQAir (AirVisual) API:** ดึงค่าฝุ่น PM 2.5 จากสถานีที่ใกล้โครงการที่สุด

4. Backend & Integration (System Orchestrator)

- **FastAPI (Python):** Web Framework ทำหน้าที่เป็นศูนย์กลางคอยรับ Webhook จาก LINE และเชื่อมต่อกับ AI Backend
- **Supabase (PostgreSQL):** Cloud Data สำหรับเก็บ Log อากาศ, ข้อมูลลูกบ้าน

5. Management & Monitoring (Admin Tools)

- **Streamlit:** Framework สำหรับสร้าง Admin Dashboard แบบง่าย เพื่อให้นิติบุคคลดูสถิติการใช้งานและรายงานสภาพอากาศย้อนหลัง
- **Pydantic:** ไลบรารีสำหรับตรวจสอบความถูกต้องของข้อมูล (Data Validation) ก่อนส่งเข้า AI เพื่อป้องกัน Error จากข้อมูลที่ผิดพลาด