



สถาบันไอจีเนียส
IT GENIUS INSTITUTE

COURSE ID
AIC-91

COURSE OUTLINE · รายละเอียดหลักสูตรอบรม

AI / LLM ENGINEERING

LLM Application Engineering Foundations

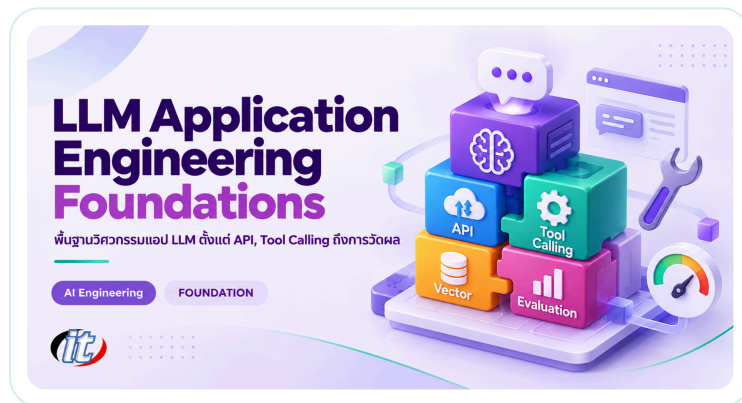
พื้นฐานวิศวกรรมแอป LLM ตั้งแต่ API, Tool Calling ถึงการวัดผล

• 18 ชั่วโมง

• 3 วัน

• พื้นฐาน เขียน Python ได้

• ระดับ Beginner - Inter



ระยะเวลา: 3 วัน

18 ชั่วโมง (3 วัน วันละ 6 ชม.) · 09:00-16:00 น.



รูปแบบ: Classroom / Hybrid

Onsite หรือ Online ผ่าน Microsoft Teams · Public / In-house



รายละเอียดหลักสูตร

www.itgenius.co.th

การเรียก LLM API ให้ได้คำตอบกลับมานั้นใช้โค้ดไม่กี่บรรทัด แต่การสร้างแอปที่ใช้งานได้จริงเป็นอีกเรื่อง นักพัฒนาต้องรับมือกับคำตอบที่รูปแบบไม่แน่นอน ค่าใช้จ่ายที่บานปลาย Rate Limit ข้อมูลที่โมเดลไม่รู้ และคำถามว่าแค่ Prompt แล้วดีขึ้นจริงหรือไม่ หลายทีมจึงกระโดดไปใช้เฟรมเวิร์ก RAG หรือ Agent แทนที่โดยยังไม่มีพื้นฐานที่แน่น ทำให้แก้ปัญหาได้ยากเมื่อระบบไม่ทำงานอย่างที่ตั้ง

หลักสูตรนี้วางพื้นฐานวิศวกรรมแอป LLM ให้นักพัฒนาแบบลงมือเขียนโค้ดด้วย Python เริ่มจากเข้าใจ Token, Context Window และพารามิเตอร์สำคัญ เรียก OpenAI, Claude และ Gemini API จัดการ Prompt แบบโค้ด และบังคับผลลัพธ์ด้วย Structured Output กับ Pydantic จากนั้นให้ LLM เรียก Tool และ API ภายใน เข้าใจ Embedding กับ Semantic Search บน Vector Database และเขียนโค้ดที่สลับผู้ให้บริการหรือโมเดลภายในได้ ก่อนเข้าสู่เรื่องที่ทำให้แอปพร้อมใช้งาน ทั้งการวัดผลด้วยชุดทดสอบ การคุมต้นทุนและความเร็ว การจัดการ Error และความปลอดภัย และการเก็บ Log ปิดท้ายด้วย Capstone แอป LLM ที่พร้อมต่อยอดสู่หลักสูตร RAG และ AI Agent (อบรม 3 วัน วันละ 6 ชั่วโมง รวม 18 ชั่วโมง ระดับ Beginner - Intermediate)

เครื่องมือ

GPT OpenAI API

CI Claude API

G Gemini API

Py Python

VDB Vector DB

ราคาค่าอบรม · PUBLIC / IN-HOUSE

9,900 บาท / ท่าน

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide โค้ด Python ตัวอย่างครบทุก Lab โปรเจกต์ Capstone ต้นแบบ และ Certificate · ค่าใช้ API ของผู้ให้บริการ AI ผู้เรียนรับผิดชอบเอง · รับจัด In-house ขอใบเสนอราคาได้

สมัครผ่านเว็บไซต์ ลด 10%



เพิ่มเพื่อน
LINE

☎ 02-570-8449

☎ 088-807-9770

LINE @itgenius

🌐 www.itgenius.co.th

01 วัตถุประสงค์ Objectives

- อธิบายการทำงานของ LLM ในมุมมองนักพัฒนา ทั้ง Token, Context Window และพารามิเตอร์สำคัญได้
- เรียก OpenAI, Claude และ Gemini API ด้วย Python ทั้งแบบปกติและแบบ Streaming ได้
- บังคับผลลัพธ์ให้ตรง Schema ด้วย Structured Output และตรวจสอบด้วย Pydantic ได้
- ออกแบบ Tool Calling ให้ LLM เรียก API และฐานข้อมูลภายในได้อย่างปลอดภัย
- ใช้ Embedding และ Vector Database ทำ Semantic Search ได้
- วัดผลแอป LLM ด้วยชุดทดสอบ การตรวจอัตโนมัติ และ LLM-as-a-Judge ได้
- คำนวณต้นทุน ความเร็ว Error และความปลอดภัยของแอป LLM ได้

02 กลุ่มเป้าหมาย Target Audience

- นักพัฒนา Backend และ Full-stack ที่เริ่มนำ LLM เข้ามาใช้ในแอป
- นักพัฒนา Python ที่อยากวางพื้นฐานก่อนเรียน RAG และ AI Agent
- Data Engineer และ Data Scientist ที่ต้องใช้ LLM ดึงหรือจัดหมวดข้อมูล
- Tech Lead ที่ต้องกำหนดแนวทางการเขียนโค้ดที่ใช้ LLM ของทีม
- นักศึกษาและผู้เปลี่ยนสายงานที่มีพื้นฐาน Python และอยากเป็น AI Engineer

03 จุดเด่นของหลักสูตร Highlights

- **ใช้ได้หลายผู้ให้บริการ** OpenAI, Claude, Gemini และโมเดลภายใน
- **ผลลัพธ์ที่โค้ดใช้ได้** Structured Output และ Pydantic
- **Tool Calling และ Embedding** พื้นฐานก่อนไป Agent และ RAG
- **วัดผลได้ ไม่ใช่แค่** ชุดทดสอบและ LLM-as-a-Judge
- **คำนวณต้นทุนตั้งแต่แรก** Token, Caching และการเลือกโมเดล

04 ความรู้พื้นฐาน Prerequisites

- เขียน Python ได้ เช่น Function, Class, Virtual Environment และการติดตั้ง Package
- เข้าใจ REST API และ JSON และใช้ Git ขั้นพื้นฐานได้
- เคยใช้ ChatGPT, Claude หรือ Gemini ในการทำงาน
- โน้ตบุ๊กที่ติดตั้ง Python และ VS Code ได้ พร้อม API Key ของผู้ให้บริการอย่างน้อยหนึ่งราย

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook ติดตั้ง Python, VS Code และ Docker Desktop ได้
LLM API	OpenAI API, Claude API และ Gemini API จากบัญชีของผู้เรียนเอง
ไลบรารี	SDK ทางการของแต่ละผู้ให้บริการ, Pydantic, FastAPI และ LiteLLM SDK
Vector Database	Chroma หรือ PostgreSQL กับ pgvector บน Docker
โมเดลภายใน	Ollama กับโมเดลขนาดเล็กสำหรับทดลองแบบไม่เสียค่า API

หมายเหตุ: ผู้เรียนใช้ API Key จากบัญชีของตนเองหรือขององค์กร Gemini API มีโควตาใช้ฟรีตามที่ Google กำหนด ส่วน OpenAI และ Claude API คิดค่าใช้จ่ายตามการใช้งานซึ่งผู้เรียนรับผิดชอบเอง โดย Lab ทั้งหมดใช้ปริมาณน้อย และทำด้วยผู้ให้บริการรายเดียวหรือโมเดลภายในผ่าน Ollama ก็ได้ หลักสูตรนี้เป็นพื้นฐานก่อนหลักสูตร RAG & Knowledge Base และ AI Agents for Business ส่วนการวัดผลและ Monitoring ระดับ Production เชิงลึกอยู่ในหลักสูตร LLMops ชื่อโมเดล ราคา และฟีเจอร์ของ API เปลี่ยนเร็ว เนื้อหาจะอัปเดตตามเวอร์ชันล่าสุด ณ วันอบรม

06 ภาพรวมการเรียนรู้ 3 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-5	เข้าใจ LLM ในมุมมองพัฒนา, เรียก OpenAI, Claude และ Gemini API ด้วย Python, จัดการ Prompt แบบโค้ด, บังคับผลลัพธ์ด้วย Structured Output และสร้าง API ดึงข้อมูลจากเอกสารด้วย FastAPI
Day 2 Section 6-10	ใช้ LLM เรียก Tool และ API ภายใน, สร้างผู้ช่วยที่ทำงานหลายขั้นตอน, เข้าใจ Embedding และ Semantic Search, ใช้ Vector Database และเขียนโค้ดที่สลับผู้ให้บริการหรือโมเดลภายในได้
Day 3 Section 11-15	วัดผลด้วยชุดทดสอบและ LLM-as-a-Judge, คุมต้นทุนและความเร็ว, จัดการ Error, Rate Limit และความปลอดภัย, เก็บ Log และ Trace แล้วปิดท้ายด้วย Capstone แอป LLM ที่พร้อมต่อยอดสู่ RAG และ Agent



DAY
1

LLM API และผลลัพธ์ที่โค้ดใช้ได้

เนื้อหาการอบรม · Section 1-5

รวม
6 ชั่วโมง

1 LLM ในมุมมองนักพัฒนา

- LLM ทำงานอย่างไรแบบที่นักพัฒนาต้องรู้: Token, การทำนายคำถัดไป และ Context Window
- Temperature, Max Tokens และพารามิเตอร์ที่มีผลต่อคำตอบ
- เลือกโมเดลตามงาน: รุ่นใหญ่ รุ่นเล็ก และ Reasoning Model
- ค่าใช้จ่ายแบบนับ Token และการประมาณต้นทุนตั้งแต่ออกแบบ
- Lab: นับ Token ของข้อความไทยและอังกฤษแล้วเทียบต้นทุน

2 Lab: เรียก LLM API ด้วย Python

- ตั้งโปรเจกต์ Python และเก็บ API Key อย่างปลอดภัยด้วย Environment Variable
- เรียก OpenAI, Claude และ Gemini API ผ่าน SDK ทางการ
- Message Role: System, User และ Assistant และการเก็บประวัติบทสนทนา
- Streaming คำตอบเพื่อให้ผู้ใช้เห็นผลเร็วขึ้น
- เปรียบเทียบรูปแบบ API ของแต่ละผู้ให้บริการ

3 Prompt แบบโค้ด

- แยก Prompt ออกจากโค้ดด้วย Template และตัวแปร
- เขียน System Prompt ที่ชัดเจน และใช้ตัวอย่างแบบ Few-shot
- จัดการเวอร์ชันของ Prompt ด้วย Git เหมือนโค้ด
- ส่งข้อมูลยาวเข้า Context อย่างมีโครงสร้างและไม่เกิน Context Window
- Lab: สร้างไลบรารี Prompt ของทีมพร้อมตัวอย่างการใช้งาน

4 Lab: Structured Output

- ทำไมการขอให้โมเดลตอบ JSON ด้วย Prompt อย่างเดียวยังไม่พอ
- กำหนด Schema ด้วย Pydantic และใช้ Structured Output ของแต่ละผู้ให้บริการ
- ตรวจสอบผลลัพธ์ จัดการกรณีไม่ตรง Schema และลองใหม่อย่างมีขอบเขต
- ออกแบบ Schema ให้โมเดลตอบง่าย เช่น Enum และฟิลด์ที่อธิบายชัด
- Lab: ดึงข้อมูลจากอีเมลและใบเสนอราคาภาษาไทยเป็น JSON

5 Lab: บริการดึงข้อมูลด้วย FastAPI

- ห่อการเรียก LLM เป็น REST API ด้วย FastAPI
- รับไฟล์หรือข้อความ แล้วคืนผลลัพธ์ตาม Schema
- ตั้ง Timeout และส่ง Error ที่ผู้เรียกใช้เข้าใจได้
- ทดสอบ API ด้วย Swagger UI และ pytest
- Lab: สร้างบริการจัดหมวดหมู่และสรุปคำร้องเรียนลูกค้า

DAY
2

Tool Calling, Embedding และการสลับโมเดล

เนื้อหาการอบรม · Section 6-10

รวม
6 ชั่วโมง

6 Tool Calling

- Tool Calling ทำงานอย่างไร: โมเดลเลือก Tool แต่โค้ดของเราเป็นผู้เรียกจริง
- เขียนคำอธิบาย Tool และ Parameter Schema ให้โมเดลเลือกถูก
- วงรอบการเรียก Tool หลายครั้ง และการเรียกหลาย Tool พร้อมกัน
- จัดการกรณี Tool ล้มเหลวหรือได้ Argument ที่ไม่ถูกต้อง
- Lab: ให้ LLM เรียก Function คำนวณและคืนข้อมูลสินค้า

7 Lab: ผู้ช่วยที่เรียก API ภายใน

- ต่อ LLM เข้ากับฐานข้อมูลคำสั่งชื่อและ API ภายในของธุรกิจตัวอย่าง
- จำกัดสิทธิ์ของ Tool: อ่านอย่างเดียว และขอยืนยันก่อนทำรายการที่เปลี่ยนข้อมูล
- จำกัดจำนวนรอบเพื่อไม่ให้วนไม่สิ้นสุดและคุมค่าใช้จ่าย
- ภาพรวม MCP ในฐานมาตรฐานเชื่อม Tool และจุดที่ต่อยอดไปสู่ Agent
- Lab: ผู้ช่วยตอบสถานะคำสั่งชื่อจากข้อมูลจริงในฐานข้อมูล

8 Embedding และ Semantic Search

- Embedding คืออะไร และวัดความคล้ายของข้อความอย่างไร
- เลือก Embedding Model ที่รองรับภาษาไทย และเรื่องจำนวนมิติกับต้นทุน
- แบ่งเอกสารเป็น Chunk และผลของขนาด Chunk ต่อคุณภาพการค้นหา
- การใช้งานนอกเหนือจาก RAG: ค้นหาคำถามซ้ำ จัดกลุ่ม และแนะนำเนื้อหา
- Lab: สร้าง Embedding ของคำถามที่พบบ่อยแล้วค้นหาด้วยความหมาย

9 Lab: Vector Database เบื้องต้น

- ภาพรวม Vector Database และการเลือกใช้ เช่น Chroma และ pgvector
- บันทึกเอกสารพร้อม Metadata และค้นหาแบบมีเงื่อนไข
- Index สำหรับค้นหาเร็ว และข้อแลกเปลี่ยนเรื่องความแม่นยำ
- รวมผลค้นหาเข้ากับคำตอบของ LLM เป็นรูปแบบ RAG อย่างง่าย
- Lab: ระบบค้นหาคู่มือสินค้าด้วยความหมายบน pgvector

10 Lab: เขียนโค้ดที่สลับโมเดลได้

- ความต่างของผู้ให้บริการที่ต้องระวัง: รูปแบบ Message, Tool และ Structured Output
- ใช้ OpenAI-compatible Endpoint และ LiteLLM SDK เพื่อเรียกหลายผู้ให้บริการด้วยโค้ดเดียว
- รันโมเดลภายในด้วย Ollama สำหรับพัฒนาและงานที่ข้อมูลห้ามออกนอกเครื่อง
- ออกแบบชั้น Abstraction ของทีมโดยไม่ผูกติดกับผู้ให้บริการรายเดียว
- Lab: สลับบริการจัดหมวดจาก Day 1 ไปใช้อีกผู้ให้บริการและโมเดลภายใน



DAY
3

วัดผล คุณต้นทุน และทำให้พร้อมใช้งาน

เนื้อหาการอบรม · Section 11-15

รวม
6 ชั่วโมง

11 Lab: วัดผลแอป LLM ขั้นพื้นฐาน

- ทำไมต้องมีชุดทดสอบ ก่อนแก้ Prompt หรือเปลี่ยนโมเดล
- สร้างชุดทดสอบจากตัวอย่างจริง พร้อมคำตอบที่คาดหวัง
- ตรวจสอบอัตโนมัติด้วยกฎ เช่น ตรง Schema, มีคำสำคัญ และค่าตัวเลขถูกต้อง
- LLM-as-a-Judge สำหรับคำตอบแบบข้อความ และข้อจำกัดที่ต้องรู้
- Lab: รันชุดทดสอบใน pytest เพื่อเทียบ Prompt สองเวอร์ชัน

12 คุณต้นทุนและความเร็ว

- วัดต้นทุนต่อคำขอและต่อผู้ใช้จากจำนวน Token จริง
- Prompt Caching และการจัดลำดับ Prompt ให้ใช้ Cache ได้
- Batch API สำหรับงานที่ไม่ต้องได้ผลทันที
- เลือกโมเดลเล็กสำหรับงานง่ายและส่งงานยากไปโมเดลใหญ่
- Lab: ลดต้นทุนของบริการจาก Day 1 แล้ววัดผลก่อนและหลัง

13 Error, Rate Limit และความปลอดภัย

- จัดการ Rate Limit และ Error ชั่วคราวด้วย Retry แบบ Exponential Backoff
- Timeout, Fallback ไปโมเดลสำรอง และข้อความแจ้งผู้ใช้เมื่อระบบขัดข้อง
- Prompt Injection เบื้องต้น และการแยกคำสั่งออกจากข้อมูลผู้ใช้
- ข้อมูลส่วนบุคคลกับ PDPA: ส่งอะไรได้ ปกปิดอะไร และเก็บ Log อย่างไร
- จัดการ API Key ในทีม และการหมุนเวียน Key

14 Lab: Log และ Trace ของแอป LLM

- บันทึก Prompt, คำตอบ, Token, เวลา และต้นทุนของทุกคำขอ
- ตั้ง Trace ให้เห็นขั้นตอนของ Tool Calling และการค้นหา
- ใช้ Log หากคำถามที่ระบบตอบไม่ได้เพื่อเพิ่มเข้าชุดทดสอบ
- เลือกระดับรายละเอียดของ Log ให้สมดุลกับความเป็นส่วนตัว

15 Capstone: แอป LLM ของตัวเอง

- ออกแบบแอป LLM ขนาดเล็กจากโจทย์งานของตนเองหรือโจทย์ตัวอย่าง
- ใช้ Structured Output, Tool Calling และ Semantic Search อย่างน้อยสองอย่าง
- สร้างชุดทดสอบ วัดต้นทุน และจัดการ Error ให้ครบ
- นำเสนอผลงาน รีวิวร่วมกัน และวางแผนทางต่อยอดสู่ RAG และ AI Agent

07 เรียนจบแล้วได้อะไร Outcomes

- เรียก LLM API คล่อง OpenAI, Claude และ Gemini
- ได้ผลลัพธ์ที่เชื่อถือได้ Structured Output และ Pydantic
- ต่อ LLM กับระบบเดิม Tool Calling และ API ภายใน
- ค้นหาด้วยความหมาย Embedding และ Vector Database
- วัดผลก่อนเปลี่ยน ชุดทดสอบและ LLM-as-a-Judge
- พร้อมใช้งานจริง ต้นทุน, Error, ความปลอดภัย และ Log

08 สิ่งที่คุณเรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมฉบับสมบูรณ์ครบทั้ง 3 วัน และ Lab Guide ที่ละเอียด
- โค้ด Python ตัวอย่างครบทุก Lab สำหรับ OpenAI, Claude, Gemini และ Ollama
- โปรเจกต์ Capstone ต้นแบบพร้อม FastAPI, ชุดทดสอบ และไฟล์ docker-compose ของ Vector Database
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สามิตร โทยม ผู้สอนและนักพัฒนาที่สร้างแอปพลิเคชันที่ใช้ LLM ในงานจริง ทำงานดึงข้อมูลจากเอกสาร ผู้ช่วยตอบคำถาม และการเชื่อม AI เข้ากับระบบเดิมขององค์กร จุดเด่นคือการสอนแบบวิศวกรรมซอฟต์แวร์ เน้นโค้ดที่อ่านง่าย ทดสอบได้ และวัดผลได้ ไม่ผูกกับผู้ให้บริการรายใดรายหนึ่ง พาผู้เรียนเห็นทั้งสิ่งที่ LLM ทำได้ดีและจุดที่มักพัง เพื่อให้กลับไปสร้างแอป LLM ของทีมบนพื้นฐานที่แน่นก่อนต่อยอดสู่ RAG และ Agent

ราคาค่าอบรม 9,900 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

9,900 บาท / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

☎ 02-570-8449

📞 @itgenius

☎ 088-807-9770

✉ contact@itgenius.co.th



เพิ่มเพื่อน
LINE