



GraphRAG and Agent Memory

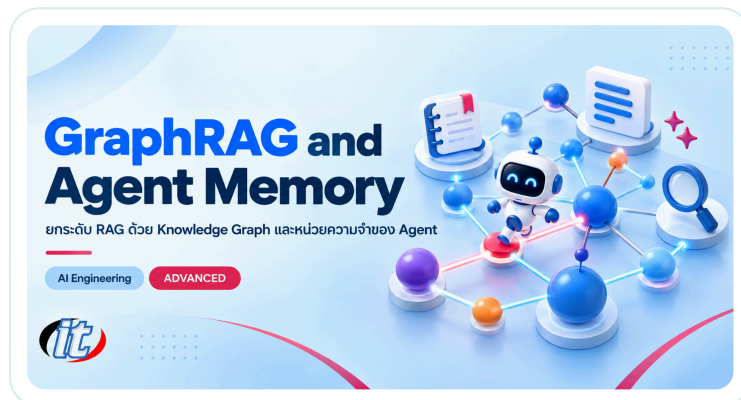
ยกระดับ RAG ด้วย Knowledge Graph และหน่วยความจำของ Agent

12 ชั่วโมง

2 วัน

พื้นฐาน Python และ RAG

ระดับ Advanced



ระยะเวลา: 2 วัน

12 ชั่วโมง (2 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

ระบบ RAG ที่ใช้ Vector Search อย่างเดียวตอบคำถามที่คำตอบอยู่ในย่อหน้าเดียวได้ดี แต่มีผลพลานเมื่อคำถามต้องเชื่อมข้อมูลหลายเอกสาร เช่น ลुकคำรายนี้ใช้สินค้านั้นที่ได้รับผลจากประกาศฉบับล่าสุด หรือถามภาพรวมของข้อมูลทั้งหมด ขณะเดียวกัน AI Agent ที่ทำงานกับผู้ใช้ต่อเนื่องก็ต้องจำได้ว่าใครเคยพูดอะไร และรู้ว่าข้อมูลใดล้าสมัยไปแล้ว Knowledge Graph จึงกลายเป็นชั้นข้อมูลสำคัญของทั้ง RAG และหน่วยความจำของ Agent

หลักสูตรนี้เป็นขั้นที่ 2 ต่อจาก RAG & Knowledge Base พาผู้เรียนสร้าง GraphRAG และ Agent Memory ด้วย Neo4j และ Python เริ่มจากเขียน Cypher ออกแบบ Graph Schema แล้วสร้าง Knowledge Graph จากเอกสารด้วย LLM ผ่านแพ็คเกจ neo4j-graphrag เปรียบเทียบ Retriever แบบ Vector, Vector ร่วมกับ Graph และ Text2Cypher และเขียนแนวคิด Community Summary ของ GraphRAG จากนั้นต่อหน่วยความจำของ Agent ทั้ง Short-term และ Long-term ด้วย LangGraph และ Temporal Knowledge Graph ด้วย Graphiti วัดผลเทียบกับ Vector RAG วางความปลอดภัยและ PDPA แล้วปิดท้ายด้วย Capstone GraphRAG Agent (อบรม 2 วัน วันละ 6 ชั่วโมง รวม 12 ชั่วโมง ระดับ Advanced)

เครื่องมือ

N4 Neo4j

RAG GraphRAG

Ag Agent Memory

Py Python

LG LangGraph

ราคาค่าอบรม · PUBLIC / IN-HOUSE

7,900 บาท / ท่าน

ราคายังไม่รวมภาษีมูลค่าเพิ่ม (VAT 7%) · รวมเอกสารประกอบการอบรม Lab Guide โค้ด Python และ Cypher ครบทุก Lab ไฟล์ docker-compose สำหรับ Neo4j และ Certificate · ผู้เรียนเตรียม Notebook และบัญชี API ของตนเอง · รับผิดชอบ In-house ขอบใบเสนอราคาได้

สมัครผ่านเว็บไซต์ ลด 10%



เพิ่มเพื่อน
LINE

01 วัตถุประสงค์ Objectives

- อธิบายได้ว่าเมื่อไร GraphRAG คุ่มค่ากว่า Vector RAG และเมื่อไรไม่จำเป็น
- เขียน Cypher สร้างและค้นข้อมูลบน Neo4j รวมถึง Vector Index
- ออกแบบ Graph Schema และสร้าง Knowledge Graph จากเอกสารด้วย LLM
- ใช้ Retriever แบบ Vector, Vector ร่วมกับ Graph และ Text2Cypher ให้เหมาะกับคำถาม
- เข้าใจ Community Summary และโหมดค้นหาแบบ Local และ Global ของ GraphRAG
- ออกแบบ Agent Memory แบบ Short-term และ Long-term ด้วย LangGraph
- สร้าง Temporal Memory ที่รู้ว่าข้อเท็จจริงใดเปลี่ยนไปแล้วด้วย Graphiti
- วัดผล GraphRAG เทียบกับ Vector RAG และวางความปลอดภัยและ PDPA

02 กลุ่มเป้าหมาย Target Audience

- AI Engineer และนักพัฒนา Python ที่สร้างระบบ RAG แล้วเจอข้อจำกัดของ Vector Search
- ผู้ที่ผ่านหลักสูตร RAG & Knowledge Base และต้องการต่อยอดขั้นถัดไป
- นักพัฒนา AI Agent ที่ต้องการให้ Agent จำผู้ใช้และบริบทได้ข้ามหลาย Session
- Data Engineer ที่ต้องเชื่อมข้อมูลหลายแหล่งให้ AI ใช้ได้
- Solution Architect ที่ต้องเลือกสถาปัตยกรรม RAG และ Memory ให้องค์กร

03 จุดเด่นของหลักสูตร Highlights

- **ขึ้นต่อจาก RAG พื้นฐาน** แก่คำถามหลายชั้นที่ Vector RAG ตอบพลาด
- **ลงมือกับ Neo4j จริง** Cypher, Vector Index และ neo4j-graphrag
- **Retriever หลายแบบ** Vector, Graph Traversal และ Text2Cypher
- **Agent ที่ทำได้จริง** LangGraph Memory และ Graphiti
- **วัดผลก่อนเลือกใช้** เทียบ GraphRAG กับ Vector RAG ด้วยตัวเลข

04 ความรู้พื้นฐาน Prerequisites

- เขียน Python ได้ในระดับใช้งาน เช่น Function, Class และติดตั้ง Package
- เข้าใจหลักการ RAG, Embedding และ Vector Search หรือผ่านหลักสูตร RAG & Knowledge Base
- ใช้ Docker, Command Line และเรียก REST API ขึ้นพื้นฐานได้
- โน้ตบุ๊กที่ติดตั้ง Docker Desktop, Python และ VS Code ได้ พร้อมสิทธิ์ติดตั้งโปรแกรม

05 สิ่งที่ต้องเตรียมและเครื่องมือ Setup & Tools

รายการ	รายละเอียด
อุปกรณ์	Notebook เชื่อมต่ออินเทอร์เน็ต ติดตั้ง Docker Desktop, Python และ VS Code ได้
Graph Database	Neo4j บน Docker หรือ Neo4j Aura แบบใช้ฟรี พร้อม Neo4j Browser
ไลบรารี	neo4j-graphrag, LangGraph และ Graphiti สำหรับ Python
LLM และ Embedding	Gemini API หรือ OpenAI API
ข้อมูลตัวอย่าง	คู่มือสินค้า นโยบาย และประวัติลูกค้าจำลองสำหรับทุก Lab

หมายเหตุ: ผู้เรียนใช้บัญชีของตนเองหรือของบริษัท Gemini API และ OpenAI API มีค่าใช้จ่ายตามการใช้งาน (Gemini API มี Free Tier) การสร้าง Knowledge Graph ด้วย LLM ใช้ Token มากกว่า RAG ทั่วไป จึงใช้ชุดเอกสารขนาดเล็กในห้อง Neo4j ใช้รุ่น Community บน Docker หรือ Neo4j Aura แบบใช้ฟรี หลักสูตรนี้เป็นขั้นที่ 2 ต่อจาก RAG & Knowledge Base ซึ่งปูพื้นฐาน Vector RAG ไว้แล้ว ชื่อเครื่องมือและฟีเจอร์ของ AI เปลี่ยนเร็ว เนื้อหาจะอิงเวอร์ชันล่าสุด ณ วันอบรม

06 ภาพรวมการเรียนรู้ 2 วัน Big Picture

วัน	หัวข้อหลัก
Day 1 Section 1-6	เข้าใจข้อจำกัดของ Vector RAG, เขียน Cypher บน Neo4j, ออกแบบ Graph Schema, สร้าง Knowledge Graph จากเอกสารด้วย LLM, ใช้ Retriever แบบ Vector, Graph และ Text2Cypher และตอบคำถามภาพรวมด้วย Community Summary
Day 2 Section 7-12	ออกแบบ Agent Memory, สร้าง Agent ด้วย LangGraph ที่ใช้ Graph เป็น Tool, ทำ Temporal Memory ด้วย Graphiti, วัดผล GraphRAG เทียบ Vector RAG, วางความปลอดภัยและ PDPA แล้วปิดท้ายด้วย Capstone

DAY
1

Knowledge Graph และ GraphRAG

เนื้อหาการอบรม · Section 1-6

รวม
6 ชั่วโมง

1 ทำไม Vector RAG อย่างเดียวไม่พอ

- คำถามที่ Vector RAG ตอบพลาด: เชื่อมหลายเอกสาร ความสัมพันธ์หลายชั้น และคำถามภาพรวม
- Knowledge Graph คืออะไร: Node, Relationship และ Property
- รูปแบบ GraphRAG: Graph เสริม Vector, Text2Cypher และ Community Summary
- เมื่อไรกลุ่มที่จะใช้ Graph และเมื่อไร Vector RAG ก็พอ
- Lab: ติดตั้ง Neo4j ด้วย Docker หรือใช้ Neo4j Aura แบบใช้ฟรี

2 Lab: Cypher สำหรับงาน RAG

- สร้าง Node และ Relationship ด้วย CREATE และ MERGE
- ค้นด้วย MATCH, WHERE และ Path หลายชั้น
- Constraint, Index และ Vector Index ใน Neo4j
- Query ที่ใช้บ่อยใน GraphRAG และการอ่านผลบน Neo4j Browser

3 ออกแบบ Graph Schema จากเอกสาร

- กำหนด Entity และ Relationship จากคำถามที่ธุรกิจต้องการคำตอบ
- Lexical Graph: เชื่อม Document และ Chunk เข้ากับ Entity
- Entity Resolution: รวมชื่อที่เขียนต่างกัน รวมถึงชื่อไทยและอังกฤษ
- Lab: ออกแบบ Schema ของคู่มือสินค้าและนโยบายบริษัทจำลอง

4 Lab: สร้าง Knowledge Graph ด้วย LLM

- ใช้ SimpleKGPipeline ของแพ็คเกจ neo4j-graphrag ดึง Entity จากข้อความและ PDF
- กำหนด Schema ให้ LLM ดึงเฉพาะสิ่งที่ต้องการ
- ตรวจสอบคุณภาพ Graph ที่ได้และแก้ Entity ที่ซ้ำกัน
- ต้นทุนและเวลาเมื่อสร้าง Graph จากเอกสารจำนวนมาก

5 Lab: Retriever แบบต่าง ๆ

- Vector Retriever และ Hybrid Retriever บน Neo4j
- VectorCypherRetriever: หา Chunk แล้วขยายบริบทตาม Relationship
- Text2CypherRetriever: แปลงคำถามเป็น Cypher พร้อมข้อควรระวัง
- Lab: ระบบถามตอบที่เลือก Retriever ตามประเภทคำถาม

6 คำถามภาพรวมและ Community Summary

- แนวคิดของ Microsoft GraphRAG: Community Detection และสรุปรายกลุ่ม
- Local Search, Global Search และ DRIFT Search ต่างกันอย่างไร
- แนวทางลดต้นทุนการทำ Index อย่าง LazyGraphRAG และ LightRAG
- Lab: ตอบคำถามภาพรวมของชุดเอกสารด้วย Community Summary



DAY
2

Agent Memory และระบบใช้งานจริง

เนื้อหาการอบรม · Section 7-12

รวม
6 ชั่วโมง

7 พื้นฐาน Agent Memory

- Short-term กับ Long-term Memory และ Memory แบบ Semantic, Episodic และ Procedural
- สิ่งที่ต้องจำ สิ่งที่ไม่ควรจำ และเมื่อไรควรลืม
- Memory แบบ Vector เทียบกับแบบ Graph
- ภาพรวมเครื่องมือ: LangGraph Store, Mem0 และ Graphiti

8 Lab: Agent ด้วย LangGraph ที่ใช้ Graph เป็น Tool

- Agent ที่เลือกใช้ Vector Search, Graph Query หรือ Text2Cypher เป็น Tool
- Checkpointer สำหรับจำบทสนทนาภายใน Thread
- Long-term Memory ด้วย Store แยกตามผู้ใช้
- ทดสอบ Agent กับคำถามที่ต้องค้นหลายขั้นตอน

9 Lab: Temporal Memory ด้วย Graphiti

- เก็บข้อเท็จจริงพร้อมช่วงเวลาที่มึผล และรู้ว่าข้อมูลใดถูกแทนที่แล้ว
- เพิ่ม Episode จากบทสนทนาและเหตุการณ์ทางธุรกิจ
- ค้นหาแบบผสม Semantic, Keyword และ Graph Traversal
- Agent ที่จำความต้องการและประวัติของลูกค้าข้ามหลาย Session

10 Lab: วัดผล GraphRAG และ Memory

- ชุดคำถามทดสอบแบบ Single-hop, Multi-hop และคำถามภาพรวม
- วัดความถูกต้อง ความครบถ้วน และการอ้างอิงแหล่งที่มา
- เทียบ Vector RAG กับ GraphRAG บนชุดคำถามเดียวกัน
- ตรวจสอบ Memory ที่ผิดหรือล้าสมัย และกลไกแก้ไข

11 ความปลอดภัยและการใช้งานจริง

- ป้องกัน Cypher Injection ด้วยสิทธิ์อ่านอย่างเดียวและตรวจสอบ Query ก่อนรับ
- สิทธิ์เข้าถึงข้อมูลตามผู้ใช้ทั้งใน Graph และ Memory
- PDPA: ข้อมูลส่วนบุคคลใน Memory สิทธิ์ขอลบ และระยะเวลาเก็บ
- อัปเดต Graph แบบ Incremental เมื่อเอกสารเปลี่ยน พร้อมเรื่อง Latency และต้นทุน

12 Workshop: Capstone GraphRAG Agent

- เลือกโจทย์: ผู้ช่วยคู่มือเทคนิค ผู้ช่วยด้านนโยบาย หรือผู้ช่วยลูกค้าที่จำประวัติ
- สร้าง Graph, Retriever และ Memory ของระบบ
- วัดผลเทียบกับ Vector RAG บนคำถามชุดเดียวกัน
- นำเสนอผลงานและแผนต่อยอด

07 เรียนจบแล้วได้อะไร Outcomes

- เลือกใช้ Graph เป็น รู้ว่าเมื่อไรคุ้มกว่า Vector RAG
- ใช้ Neo4j ได้คล่อง Cypher และ Vector Index
- สร้าง Knowledge Graph ได้ จากเอกสารด้วย LLM
- ใช้ Retriever ได้ตรงงาน Vector, Graph และ Text2Cypher
- สร้าง Agent ที่ทำได้ LangGraph Memory และ Graphiti
- วัดผลและใช้อย่างปลอดภัย Evaluation, สิทธิ์ และ PDPA

08 สิ่งที่คุณเรียนจะได้รับ Deliverables

- เอกสารประกอบการอบรมฉบับสมบูรณ์ครบทั้ง 2 วัน และ Lab Guide ที่ละเอียด
- โค้ด Python และ Cypher ตัวอย่างครบทุก Lab ทั้ง GraphRAG, LangGraph และ Graphiti
- ไฟล์ docker-compose สำหรับ Neo4j พร้อมชุดข้อมูลจำลองและชุดคำถามทดสอบ
- ประกาศนียบัตรจบหลักสูตร (Certificate of Completion) และช่องทางสอบถามวิทยากรหลังจบหลักสูตร

09 ข้อมูลผู้สอน Instructor

อ.สามิตร โทยม ผู้สอนและนักพัฒนาที่ออกแบบระบบ RAG และ AI Agent สำหรับองค์กร สอนโดยเริ่มจากคำถามจริงที่ Vector RAG ตอบพลาด แล้วให้ผู้เรียนเห็นด้วยตัวเองว่า Graph ช่วยตรงไหนและเพิ่มต้นทุนตรงไหน เน้นการวัดผลก่อนตัดสินใจเลือกสถาปัตยกรรม และการออกแบบ Memory ที่ปลอดภัยต่อข้อมูลส่วนบุคคล เพื่อให้นำไปใช้กับระบบขององค์กรได้อย่างมีประสิทธิภาพ

ราคาค่าอบรม 7,900 บาท/ท่าน · สนใจ IN-HOUSE ขอใบเสนอราคา

7,900 บาท / ท่าน (ยังไม่รวม VAT 7%)

สมัครผ่านเว็บไซต์ ลด 10%

☎ 02-570-8449

☎ 088-807-9770

LINE @itgenius

✉ contact@itgenius.co.th



เพิ่มเพื่อน
LINE