



A Multi-User RAG Chatbot Platform with Channel and Document Management

แพลตฟอร์มแชทบอท RAG แบบหลายผู้ใช้พร้อมการจัดการห้องสนทนาและเอกสาร

นายรชต เพชรรัตน์ รหัสประจำตัว 6703052411058

นายธิติพล เคารพธรรม รหัสประจำตัว 6703052411180

อาจารย์ที่ปรึกษาปริญญานิพนธ์ รองศาสตราจารย์ ดร.ชูพันธุ์ รัตนโกศา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรอุตสาหกรรมศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีอิเล็กทรอนิกส์ (คอมพิวเตอร์)

ภาควิชาเทคโนโลยีวิศวกรรมอิเล็กทรอนิกส์

วิทยาลัยเทคโนโลยีอุตสาหกรรม

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

พ.ศ. 2568

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของโครงการปริญญานิพนธ์

ในปัจจุบัน แม้เทคโนโลยีจะก้าวหน้าอย่างรวดเร็ว แต่หลายองค์กรและหน่วยงานต่างยังคงประสบปัญหาในการเข้าถึงข้อมูลที่ต้องการอย่างรวดเร็วและแม่นยำ โดยเฉพาะอย่างยิ่งเมื่อต้องจัดการกับข้อมูลจำนวนมากในรูปแบบของเอกสาร เช่น เอกสารทางราชการ เอกสารวิชาการ หรือคู่มือภายในองค์กร ปัญหาเหล่านี้มักทำให้การสืบค้นข้อมูลใช้เวลานาน เกิดความซ้ำซ้อน หรือสื่อสารผิดพลาด ส่งผลกระทบต่อประสิทธิภาพในการทำงานและการให้บริการ นอกจากนี้ ระบบแชทบอททั่วไปยังไม่สามารถตอบคำถามจากข้อมูลเฉพาะที่ผู้ใช้งานต้องการได้อย่างแม่นยำ เนื่องจากขาดความสามารถในการอ้างอิงจากแหล่งข้อมูลเฉพาะเจาะจง ทำให้ไม่สามารถประยุกต์ใช้ได้อย่างเต็มประสิทธิภาพในบางบริบทเฉพาะทาง

หนึ่งในเทคโนโลยีที่ได้รับความสนใจในการแก้ไขปัญหาข้างต้น คือ ปัญญาประดิษฐ์ (Artificial Intelligence: AI) โดยเฉพาะเทคโนโลยีโมเดลภาษา (Language Model) ที่สามารถประมวลผลภาษาธรรมชาติและโต้ตอบกับมนุษย์ได้อย่างชาญฉลาด ตัวอย่างที่โดดเด่น ได้แก่ ChatGPT ของ OpenAI, Gemini จาก Google และ Claude จาก Anthropic ซึ่งเป็นโมเดลภาษาใหญ่ (Large Language Model: LLM) ที่ถูกฝึกด้วยข้อมูลจำนวนมากมหาศาล ทำให้สามารถตอบคำถาม วิเคราะห์ข้อมูล และช่วยแก้ปัญหาได้อย่างมีประสิทธิภาพ ทั้งในระดับบุคคลและองค์กร นอกจากนี้ หลายหน่วยงานในประเทศไทยเริ่มนำ AI มาใช้ เช่น ระบบช่วยตอบคำถามในหน่วยงานราชการ ระบบตรวจสอบเอกสาร และแพลตฟอร์มการเรียนรู้อัตโนมัติ ซึ่งล้วนเป็นการสะท้อนแนวโน้มของการนำเทคโนโลยีมาเพิ่มประสิทธิภาพการทำงานในทุกภาคส่วน

จากแนวโน้มการใช้งานเทคโนโลยี AI ที่เพิ่มมากขึ้น จึงเกิดแนวคิดในการพัฒนาแพลตฟอร์มแชทบอทที่สามารถตอบคำถามจากเอกสารที่ผู้ใช้งานกำหนดเอง โดยอาศัยเทคนิค Retrieval-Augmented Generation (RAG) ซึ่งผสมผสานความสามารถของโมเดลภาษาในการสร้างคำตอบกับการค้นหาข้อมูลจากเอกสารเฉพาะ เช่น ไฟล์ที่ผู้ใช้อัปโหลด ระบบจะออกแบบให้รองรับการใช้งานแบบหลายผู้ใช้ (multi-user) โดยมีฟังก์ชันการจัดการห้องสนทนา แชนแนล เอกสาร การกำหนดสิทธิ์ และตั้งค่าความเป็นส่วนตัวอย่างครบถ้วน ผู้ใช้งานสามารถโต้ตอบกับ AI ที่ให้ข้อมูลตรงกับบริบทของตนเอง ทำให้สามารถนำแพลตฟอร์มนี้ไปประยุกต์ใช้ได้หลากหลายบริบท เช่น องค์กร สถาบันการศึกษา หรือกลุ่มวิชาชีพเฉพาะทางที่ต้องการระบบผู้ช่วยอัจฉริยะที่เข้าใจข้อมูลภายใน

1.2 วัตถุประสงค์ของโครงการงานปริญญานิพนธ์

- 1.2.1 เพื่อศึกษาพัฒนาเว็บแอปพลิเคชันด้วย Nuxt.js และ Tailwind CSS
- 1.2.2 เพื่อศึกษาการใช้ FastAPI พัฒนาส่วน Backend เว็บแอปพลิเคชันด้วยภาษา Python
- 1.2.3 เพื่อศึกษาการใช้งาน LlamaIndex
- 1.2.4 เพื่อศึกษาการทำ Text embedding
- 1.2.5 เพื่อศึกษาการเขียนโปรแกรมเพื่อเรียกใช้งาน Ollama
- 1.2.6 เพื่อศึกษาการออกแบบฐานข้อมูลและการจัดการด้วย MariaDB

1.3 ขอบเขตของโครงการงานปริญญานิพนธ์

1.3.1 ขอบเขตของระบบ

- ระบบสามารถเข้าใช้งานผ่าน ICIT Account
- บริการให้เป็นเว็บไซต์โดยใช้ FastAPI ในการทำ Back-End กับใช้ Nuxt.js ในการทำ Front-End และใช้ MariaDB เป็นระบบฐานข้อมูล
- ผู้ใช้งานที่ลงทะเบียน (Member) จะไม่สามารถเผยแพร่แชแนลของตนเองเป็นสาธารณะได้โดยตรง หากไม่ได้รับการอนุมัติจากผู้ดูแลระบบ (Admin) ก่อน
- มีการทำ Retrieval-Augmented Generation (RAG) สามารถดึงข้อมูลจากที่เกี่ยวข้องกับคำถาม โดยข้อมูลที่แสดงจะต้องสอดคล้องกับเอกสารที่ผู้ใช้อัปโหลดเท่านั้น
- ใน 1 แชแนลอนุญาตอัปโหลดเอกสารได้ไม่เกิน 50 ไฟล์ และในหนึ่งไฟล์ขนาดไม่เกิน 50 MB
- มีการให้บริการ API สำหรับเชื่อมต่อกับระบบภายนอก
- เปรียบเทียบประสิทธิภาพและค่าใช้จ่ายของโมเดล LLM ผ่าน Ollama อย่างน้อย 3 ตัวเพื่อเลือกโมเดลที่เหมาะสมที่สุดสำหรับโครงการงาน

1.3.2 ขอบเขตของผู้ใช้งานจะแบ่งประเภทออกเป็น 2 ประเภท คือ

1.3.2.1 ผู้ใช้งานไม่ผ่านการ Login

- สามารถเห็นและโต้ตอบกับ AI ที่เป็นแชแนลสาธารณะได้

1.3.2.2 ผู้ใช้งานผ่านการ Login จะแบ่งสิทธิ์เป็น 2 สิทธิ์ คือ

1.3.2.2.1 สิทธิ์ผู้ดูแลระบบ (Admin)

- สามารถสอบถาม AI เกี่ยวกับเอกสารได้ทุกแชแนลไม่จะเป็นส่วนตัวหรือสาธารณะ
- สามารถลบแชแนลและเอกสารได้ทุกแชแนล
- สามารถดูสถิติได้ เช่น จำนวนผู้ใช้งาน จำนวนคำถามที่ถามต่อวัน
- สามารถอนุมัติหรือปฏิเสธคำขอของผู้ใช้งานที่ต้องการตั้งค่าให้แชแนล

ของตนเองเป็นแบบสาธารณะได้

1.3.2.2.2 สิทธิผู้ใช้งาน (Member)

- สามารถสร้างแชแนลและลบของตนเองได้
- สามารถอัปโหลดเอกสาร PDF หรือ Text ลงในแชแนลที่ตนสร้าง
- สามารถพิมพ์ข้อความเพื่อสอบถามกับ AI เกี่ยวกับเนื้อหาในเอกสารที่อัปโหลด
- สามารถตั้งค่าความเป็นส่วนตัวหรือสาธารณะให้กับแชแนลของตนเองได้
- ระบบมีการจัดเก็บประวัติการสนทนา (Chat History)
 - เก็บประวัติสนทนาเพื่อสามารถดูข้อความที่เคยสนทนาได้
- สามารถตั้งชื่อแชแนลของตนเองได้

1.4 ประโยชน์ที่คาดว่าจะได้รับจากการทำโครงการปริญญาโท

- 1.4.1 เข้าใจการพัฒนาเว็บแอปพลิเคชันด้วยภาษา Nuxt.js และ Tailwind CSS
- 1.4.2 เข้าใจการใช้ FastAPI พัฒนาส่วน Backend เว็บแอปพลิเคชันด้วยภาษา Python
- 1.4.3 เข้าใจการใช้งาน LlamaIndex
- 1.4.4 เข้าใจการทำ Text embedding
- 1.4.5 เข้าใจการเขียนโปรแกรมเพื่อเรียกใช้งาน Ollama
- 1.4.6 เข้าใจการออกแบบฐานข้อมูลและจัดการด้วยโปรแกรม MariaDB

บทที่ 2

ทฤษฎีและเอกสารที่เกี่ยวข้อง

2.1 Nuxt.js

Nuxt.js คือเฟรมเวิร์กที่พัฒนาต่อยอดมาจาก Vue.js โดยออกแบบมาเพื่อช่วยให้การพัฒนาเว็บแอปพลิเคชัน (Web Application) เป็นไปอย่างง่ายขึ้น รวดเร็วขึ้น และได้ประสิทธิภาพมากขึ้น โดยเฉพาะในด้านการทำ Server-Side Rendering (SSR), Static Site Generation (SSG) และ Single Page Application (SPA)

2.2 RAG (Retrieval-Augmented Generation)

RAG (Retrieval-Augmented Generation) คือ แนวคิดหรือเทคนิคในระบบปัญญาประดิษฐ์ (AI) โดยเฉพาะในด้านการประมวลผลภาษาธรรมชาติ (Natural Language Processing – NLP) ที่นำเอาการดึงข้อมูล (Retrieval) และ การสร้างข้อความใหม่ (Generation) มาทำงานร่วมกัน เพื่อให้ได้คำตอบที่แม่นยำและตรงกับบริบทของคำถามมากที่สุด

2.3 MariaDB

MariaDB คือระบบบริหารจัดการฐานข้อมูล (Database Management System - DBMS) ที่ใช้ภาษา SQL (Structured Query Language) ในการจัดการข้อมูล โดยเป็นซอฟต์แวร์ประเภท ฐานข้อมูลเชิงสัมพันธ์ (Relational Database) ซึ่งหมายถึงการจัดเก็บข้อมูลเป็นตารางที่มีความสัมพันธ์กัน

2.4 Embeddings

Embeddings คือเทคนิคในการแปลงข้อมูล (เช่น คำ, ประโยค, หรือเอกสาร) ให้อยู่ในรูปของเวกเตอร์ตัวเลข (Vector) ที่สามารถนำไปใช้ในการคำนวณหรือประมวลผลทางคณิตศาสตร์ได้ เช่น การวัดความคล้ายคลึง (Similarity) หรือการค้นหาข้อมูลที่เกี่ยวข้อง

2.5 Vector Store

Vector Store คือระบบหรือฐานข้อมูลที่ใช้สำหรับ จัดเก็บและค้นหาเวกเตอร์ (vector) ซึ่งเป็นข้อมูลตัวเลขที่ได้จากการแปลงข้อความหรือภาพด้วย Embeddings เช่น ประโยค, เอกสาร หรือคำ ต่าง ๆ

จ: เจริญ Vector Store หรือ Vector Database
กนกพงศ์ เรือง Vector Database

บทที่ 3

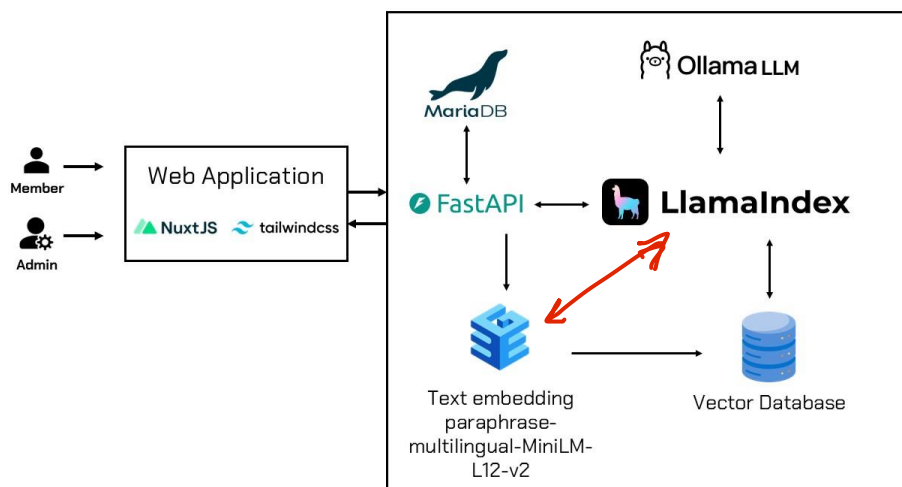
วิธีการดำเนินงาน

ในการออกแบบและพัฒนาแพลตฟอร์มแชทบอท RAG แบบหลายผู้ใช้พร้อมการจัดการห้องสนทนาและเอกสาร สามารถแบ่งขั้นตอนออกเป็น ส่วน ดังนี้

3.1 หลักการทำงานโดยรวมของระบบ

3.2 การออกแบบฐานข้อมูลที่ใช้ภายในเว็บไซต์

3.1 หลักการทำงานโดยรวมของระบบ



ภาพที่ 1 ภาพของการทำงานโดยรวมของเว็บไซต์

จัดอันดับได้ ๓

สถาปัตย์ 1

ฝั่งผู้ใช้ (Client Side)

Member และ Admin ใช้งานระบบผ่านเว็บแอปพลิเคชัน

เว็บแอปสร้างขึ้นด้วย

- Nuxt.js ซึ่งเป็นเฟรมเวิร์กสำหรับพัฒนาเว็บด้วย Vue.js
- TailwindCSS เฟรมเวิร์กสำหรับการจัดการรูปแบบและการแสดงผลของหน้าเว็บให้สวยงามและตอบสนองได้ดี

เจเนอเรชันระบบ
(backend) presentation

สถาปัตยกรรมระบบ AI-Chatbot ที่ฉลาดและใช้งานง่าย

ฝั่งเซิร์ฟเวอร์ (Backend)

ส่วนนี้ทั้งหมดทำงานกันผ่าน **FastAPI** ซึ่งเป็นเฟรมเวิร์กภาษา Python ที่ใช้สร้าง REST API ที่มีความเร็วสูงและใช้งานง่าย

1. MariaDB

ทำหน้าที่เป็นระบบฐานข้อมูล เก็บข้อมูลหลักของระบบ เช่น ข้อมูลผู้ใช้ ข้อมูลช่อง และข้อมูลไฟล์ต่างๆ

2. Text Embedding Model (paraphrase-multilingual-MiniLM-L12-v2)

เป็นโมเดลที่แปลงข้อความให้กลายเป็นเวกเตอร์ เพื่อให้ระบบสามารถเข้าใจความหมายของข้อความในเชิงคณิตศาสตร์ได้

3. Vector Database

เป็นฐานข้อมูลที่ใช้เก็บเวกเตอร์จากข้อความ ซึ่งช่วยให้สามารถค้นหาข้อมูลที่มีความหมายใกล้เคียงกันได้ ไม่จำกัดแค่คำตรงตัว

4. LlamaIndex

เป็นเฟรมเวิร์กที่ช่วยเชื่อมโยงโมเดลภาษา (LLM) เข้ากับฐานข้อมูลและเอกสารต่าง ๆ จัดการกระบวนการระหว่าง FastAPI, Vector Database และโมเดล LLM

5. Ollama LLM

แพลตฟอร์มซอฟต์แวร์แบบโอเพนซอร์สที่ช่วยให้สามารถรันโมเดลภาษาขนาดใหญ่ (LLM) ได้โดยตรง ทำหน้าที่สร้างคำตอบจากข้อมูลที่ได้จาก LlamaIndex

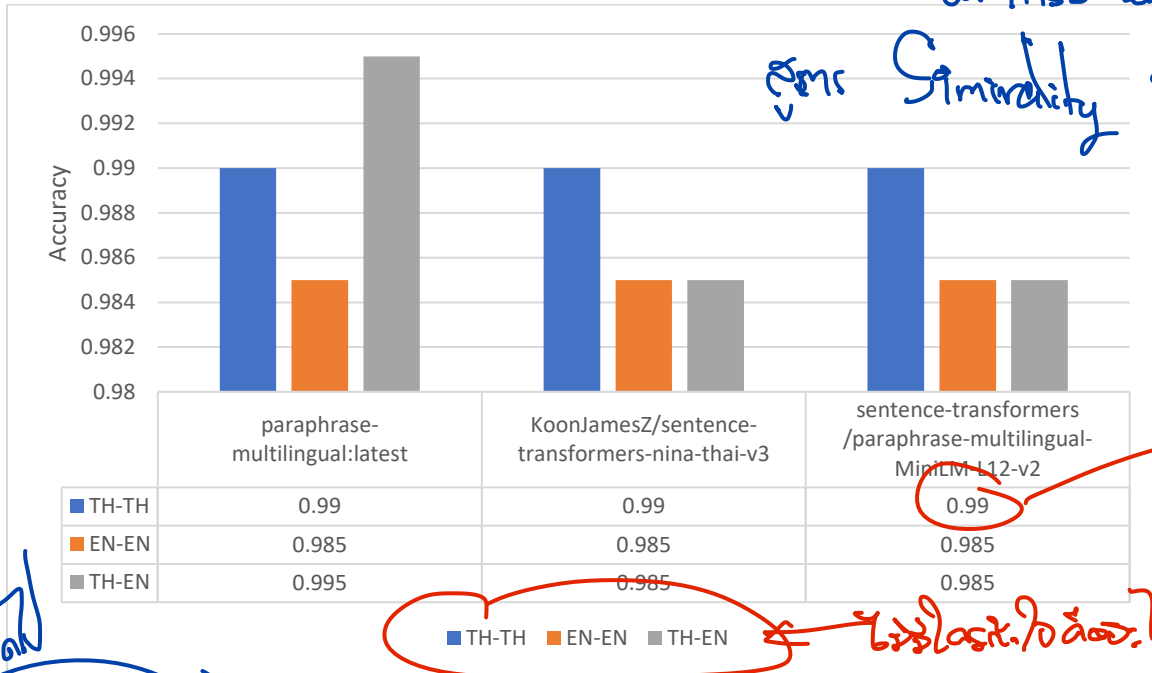
แพลตฟอร์มแชทบอท RAG แบบหลายผู้ใช้พร้อมการจัดการห้องสนทนาและเอกสารเป็นสื่อกลางระหว่างผู้ใช้งาน, ระบบหลังบ้าน (Backend) และ โมเดลภาษาขนาดใหญ่ (LLM) ผู้ใช้งานสามารถเข้าถึง แพลตฟอร์มผ่านเว็บเบราว์เซอร์ โดยแพลตฟอร์มนี้พัฒนาโดยใช้เฟรมเวิร์ก Nuxt.js ที่เขียนด้วยภาษา TypeScript ทำหน้าที่ประมวลผลคำร้องขอ (Request) จากผู้ใช้งานและเชื่อมต่อกับ ส่วนต่อประสานโปรแกรมประยุกต์ (API) ด้วย FastAPI เพื่อดึงหรือบันทึกข้อมูลตามที่ต้องการลงในฐานข้อมูล โดยใช้ MariaDB ซึ่งเป็นระบบจัดการฐานข้อมูลแบบเปิดที่มีความเร็วและความปลอดภัยสูง MariaDB ใช้สำหรับเก็บข้อมูลต่างๆ ของระบบ เช่น ข้อมูลผู้ใช้งาน ข้อมูลแชแนล และข้อความ ทำให้เป็นตัวเลือกที่ดีสำหรับการใช้งานร่วมกับ Nuxt.js และ FastAPI

1. สถาปัตยกรูปร่างนี้ ให้ระบบระบบ

ข้อสงสัย? ข้อควรระวัง

3.2 การทดสอบ Text Embedding Model (โมเดลฝังข้อความ)

Accuracy (ความแม่นยำ)



Dataset มาจากไหน
ตรงกับ Dataset ใด
ความเหมือน Si similarity

Similarity ?

คำแปล
หรือคำอื่น

ภาษาพูด

ภาษาพูด

ไม่ใช่ใครก็ได้จ้ะ

- ทั้งสามตัวแม่นยำมาก ($>0.98-0.99$) ในทุกภาษา ใช้ได้ดีทั้งไทย-ไทยและข้ามภาษา
- paraphrase-multilingual เหนือสุดใน TH-EN (0.995) เหมาะกับการค้นหาข้ามภาษาเช่น ถามไทย-ตอบอังกฤษ
- nina-thai-v3 และ MiniLM-L12-v2 แม่นเท่ากันใน TH-TH และ EN-EN

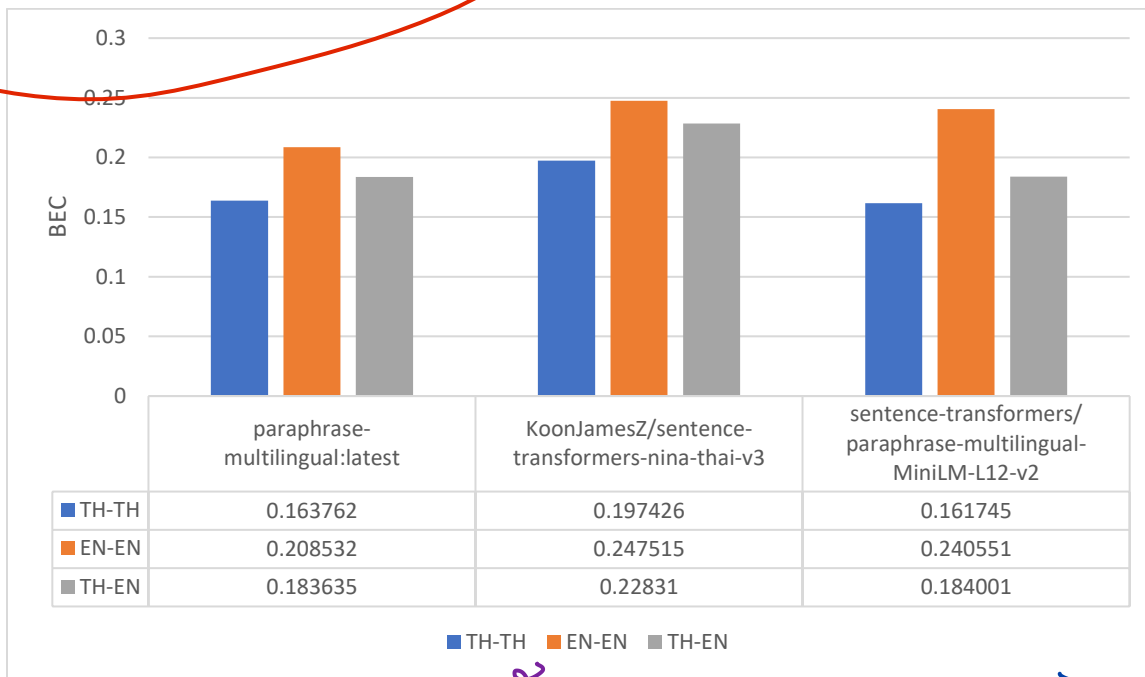
เราควรระวัง 3 ข้อ

1. ภาษาที่ใช้ในการทดลอง
2. ภาษาที่ใช้ในการค้นหา
3. ภาษาที่ใช้ในการแสดงผล

อย่าใช้ * หรือ * ในการค้นหา

ดี ๑๐๐% ๑

Bidirectional Embedding Correlation (ค่าความคล้ายคลึงระหว่างการฝัง)

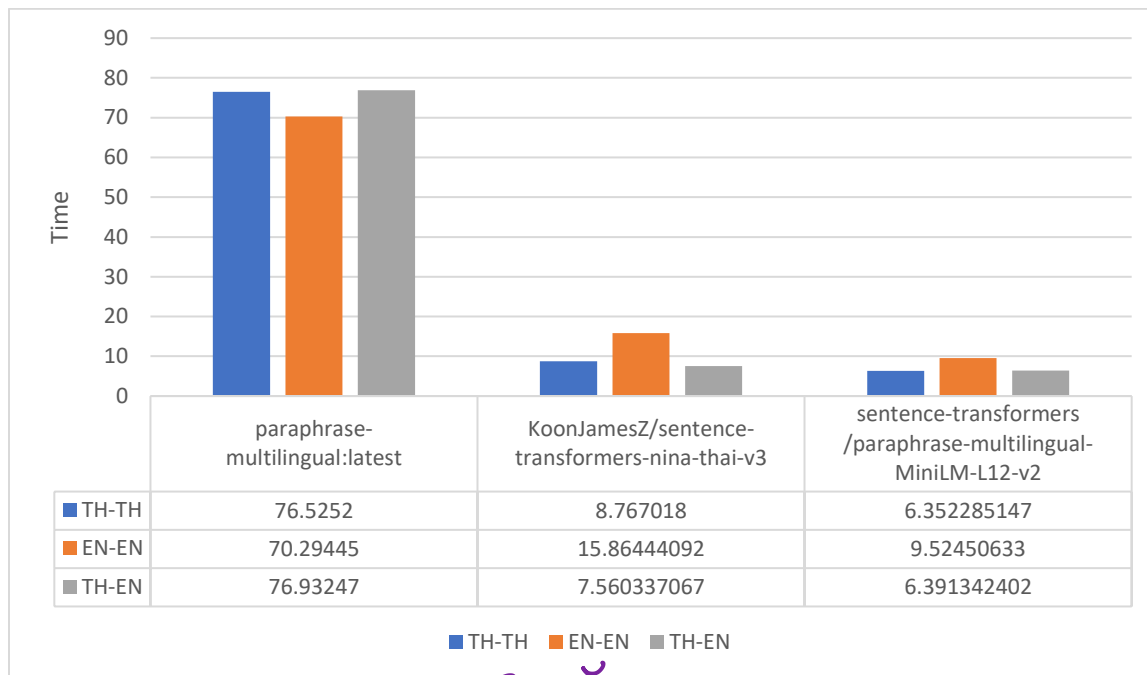


จาก

- BEC ค่าต่ำ = เวกเตอร์สองภาษาสอดคล้องกันดี
- ทั้งสามโมเดล BEC $\approx 0.16-0.24$ สอดคล้องกันค่อนข้างดีมาก
- paraphrase-multilingual และ MiniLM-L12-v2 ต่ำสุดแสดงว่ามีประสิทธิภาพการฝังที่จัดตำแหน่งกันดีที่สุด
- nina-thai-v3 สูงกว่าเล็กน้อย (≈ 0.22) อาจมีความเอนเอียงทางภาษาไทย

ผมยังไม่เข้าใจ

Processing Time (ระยะเวลาในการประมวลผล)



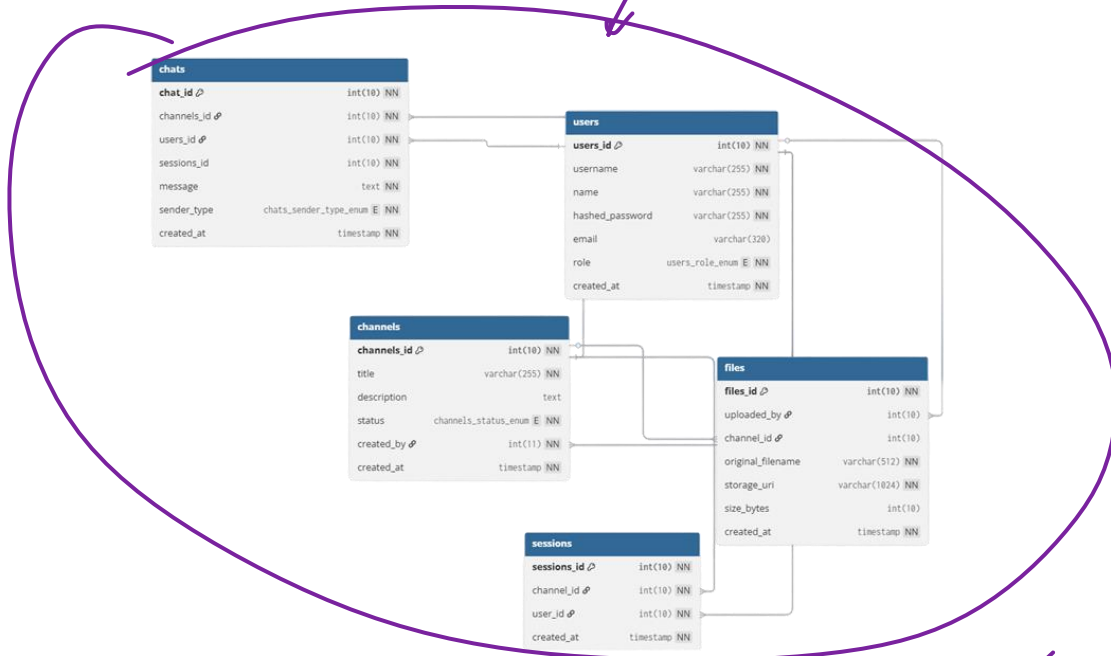
- MiniLM-L12-v2 เร็วที่สุด (6–9 ms)
- nina-thai-v3 เร็วกว่า paraphrase มาก $\approx 8\text{--}15$ ms
- paraphrase-multilingual ช้าที่สุด $\approx 70\text{--}77$ ms

ช้ากว่า

ดีกว่า

3.3 การออกแบบฐานข้อมูลที่ใช้ภายในเว็บไซต์

เริก / จกพเฑดลรล



ภาพที่ 2 ภาพของฐานข้อมูลโดยรวมของเว็บไซต์

ดัวลคพี ...

ในปริญญานิพนธ์นี้ใช้โปรแกรม MariaDB ในการจัดการฐานข้อมูล ซึ่งจะประกอบด้วยตารางต่างๆ ดังนี้

ได้แก่

- Users เก็บข้อมูลผู้ใช้งาน
- Chats เก็บข้อความในแต่ละช่อง
- Channels เก็บข้อมูลช่อง
- Files เก็บข้อมูลไฟล์ที่อัปโหลดในช่องหรือการสนทนา
- Sessions เก็บข้อมูลเซสชันการสนทนา

บทที่ 4 ผลการทดลอง

การทำงานของเว็บไซต์จะแบ่งออกเป็น 2 ส่วน ดังนี้

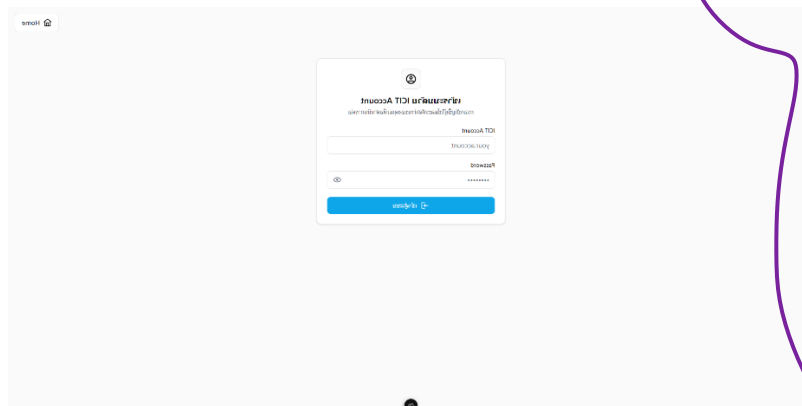
- 4.1 ผู้ใช้งานไม่ผ่านการ Login
- 4.2 ผู้ใช้งานผ่านการ Login จะแบ่งสิทธิ์เป็น 2 สิทธิ์ คือ
 - 4.2.1 ผู้ดูแลระบบ (Admin)
 - 4.2.2 ผู้ใช้งานทั่วไป (Member)

สิทธิ์แบ่งเป็น

Admin

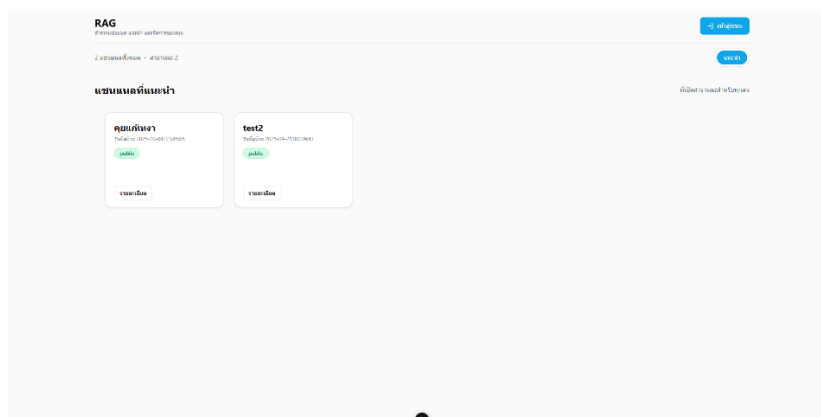
ถ้าไม่ Login

ไปก็ไม่ได้ใช้งาน



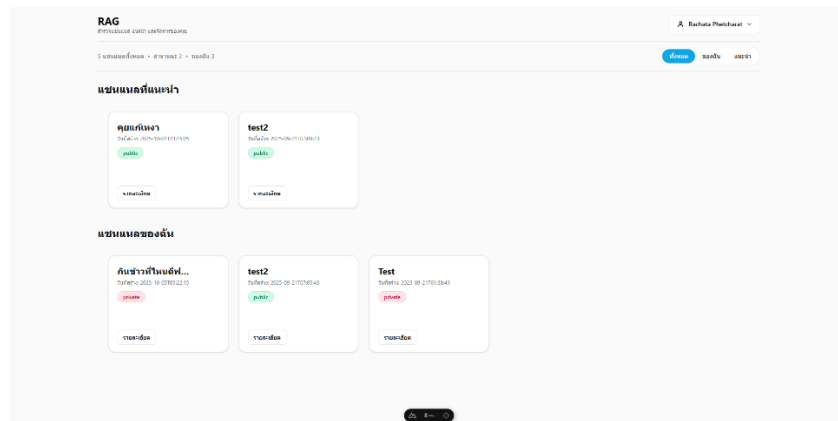
ภาพที่ 3 ภาพหน้า Login เข้าสู่ระบบ

4.1 ผู้ใช้งานไม่ผ่านการ Login



ภาพที่ 4 ภาพผู้ใช้งานไม่ผ่านการ Login

4.2 ผู้ใช้งานผ่านการ Login



ภาพที่ 5 ภาพผู้ใช้งานผ่านการ Login

4.2.1 ผู้ดูแลระบบ (Admin)

4.2.2 ผู้ใช้งานทั่วไป (Member)

บทที่ 5

สรุปผล

5.1 สรุปผล

5.2 ปัญหาที่เกิดขึ้นและการแก้ไข

5.3 แนวทางที่ทำต่อไป