



ชื่อปริญญาบัตร	ระบบแบบบอห์จรรย์เพื่อให้ข้อมูล สำหรับ คณะวิศวกรรมศาสตร์		
ชื่อนักศึกษา	นายกรภัทร เจริญสุข	รหัส	056550405145-1
	นายกิตติพงศ์ คาแพงน้อย	รหัส	056550405147-7
	นายอนุวัฒน์ กลิ่นโสภณ	รหัส	056550405137-8
	นายไชยพัฒน์ สมแวง	รหัส	056550405140-2
ที่ปรึกษาปริญญาบัตร	ผศ.ดร.เกรียงไกร เหลืองอำพล		
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ปีการศึกษา	2568		

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร
อนุมัติให้ปริญญาบัตรนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

คณะกรรมการสอบปริญญาบัตร

.....	ประธานกรรมการ
()
.....	กรรมการ
()
.....	กรรมการ
()
.....	หัวหน้าสาขา
()

ลิขสิทธิ์ของคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร



Project Title Smart Information Chatbot System of Engineering

By

Kornrapat	Charoensuk	056550405145-1
Gidtipong	Capangnoi	056550405147-7
Anuwat	Klinsopon	056550405137-8
Chaiyapat	Somwang	056550405140-2

Project Advisor Asst. Prof. Dr. Kriengkri Luangampol

Major program Computer Engineering

Academic Year 2025

Accepted by Department of Computer Engineering, the Faculty of Engineering
Rajamangala University of Technology Phra Nakhon in Partial Fulfillment of the
Requirements for the Bachelor Degree of Engineering.

Project Committee

..... Chairperson

()

..... Member

()

..... Member

()

..... Head of Department

()

Copyright of the Faculty of Engineering Rajamangala University of Technology Phra
Nakhon.



ชื่อปริญญาบัตร	ระบบแชทบอทอัจฉริยะเพื่อให้ข้อมูล สำหรับ คณะวิศวกรรมศาสตร์	
ชื่อนักศึกษา	นายกรภัทร เจริญสุข	รหัส 056550405145-1
	นายกิตติพงศ์ คาแพงน้อย	รหัส 056550405147-7
	นายอนุวัฒน์ กลิ่นโสภณ	รหัส 056550405137-8
	นายไชยพัฒน์ สมแวง	รหัส 056550405140-2
ที่ปรึกษาปริญญาบัตร	ผศ.ดร.เกรียงไกร เหลืองอำพล	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์	
ปีการศึกษา	2568	

บทคัดย่อ

ระบบผู้ช่วยอัจฉริยะสำหรับคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร พัฒนาขึ้นเพื่อลดภาระงานเจ้าหน้าที่และให้บริการข้อมูล 24 ชั่วโมงตามแนวทางมหาวิทยาลัยดิจิทัล ขับเคลื่อนด้วยเทคโนโลยี Hybrid Chatbot (Rule-based ผสาน Machine Learning: TF-IDF + Logistic Regression) ภายใต้สถาปัตยกรรม 3-Tier รองรับ RESTful API พัฒนาโดยใช้ Frontend (HTML5, CSS3, JS), Backend (PHP 8.0+), AI Engine (Python Flask) และฐานข้อมูล MySQL ระบบสามารถตอบคำถามอัตโนมัติ ค้นหาข้อมูลบุคลากร 118 ท่าน ดึงข่าวสารผ่าน Web Scraping และบันทึกประวัติการสนทนาพร้อมประเมินค่า Confidence Score ได้อย่างมีประสิทธิภาพ

(ปริญญาบัตรนี้จำนวน 108 หน้า)

คำสำคัญ: ระบบแชทบอท, ปัญญาประดิษฐ์, การประมวลผลภาษาธรรมชาติ, TF-IDF, Logistic Regression, Web Scraping, คณะวิศวกรรมศาสตร์



Project Title	Smart Information Chatbot System of Engineering		
By	Kornrapat	Charoensuk	056550405145-1
	Gidtipong	Capangnoi	056550405147-7
	Anuwat	Klinsopon	056550405137-8
	Chaiyapat	Somwang	056550405140-2
Project Advisor	Asst. Prof. Dr. Kriengkri Luangampol		
Major program	Computer Engineering		
Academic Year	2025		

Abstract

The Intelligent Assistant System for the Faculty of Engineering, Rajamangala University of Technology Phra Nakhon, was developed to reduce staff workload and provide 24-hour information access in alignment with digital university guidelines. Driven by Hybrid Chatbot technology (combining a Rule-based system and Machine Learning: TF-IDF + Logistic Regression) on a 3-Tier architecture supporting RESTful APIs, the system utilizes a Frontend (HTML5, CSS3, JS), Backend (PHP 8.0+), AI Engine (Python Flask), and a MySQL database. It effectively provides automated responses, searches for 118 personnel profiles, retrieves news via Web Scraping, and records chat logs with Confidence Score evaluations.

(Total 108 Pages)

Keywords: Chatbot System, Artificial Intelligence, Natural Language Processing, TF-IDF, Logistic Regression, Web Scraping, Faculty of Engineering

กิตติกรรมประกาศ

การจัดทำปริญญานิพนธ์เรื่อง ระบบแชทบอทอัจฉริยะเพื่อให้ข้อมูล สำหรับ คณะ วิศวกรรมศาสตร์ ในครั้งนี้ สำเร็จลุล่วงตามวัตถุประสงค์ได้ด้วยดี เนื่องด้วยได้รับความอนุเคราะห์และความกรุณาอย่างสูงจาก ผู้ช่วยศาสตราจารย์ ดร.เกรียงไกร เหลืองอำพล อาจารย์ที่ปรึกษาโครงการ ที่ได้กรุณาสละเวลาถ่ายทอดความรู้ ให้แนวคิด และคำแนะนำอันทรงคุณค่าในการพัฒนาระบบแชทบอทให้มีความถูกต้องสมบูรณ์ ตลอดจนเอาใจใส่ติดตามความคืบหน้าและเป็นกำลังใจสำคัญให้แก่ คณะผู้จัดทำเสมอมา ผู้จัดทำรู้สึกซาบซึ้งในความกรุณาและขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบพระคุณคณาจารย์ภาควิชาวิศวกรรมคอมพิวเตอร์ทุกท่าน ที่ได้ประสิทธิ์ประสาทวิชาความรู้ และให้คำแนะนำอันเป็นประโยชน์ในการจัดทำโครงการ ตลอดจนอำนวยความสะดวกในการใช้งานสถานที่และอุปกรณ์ต่าง ๆ ที่จำเป็นต่อการดำเนินงาน

ท้ายที่สุดนี้ คณะผู้จัดทำขอกราบขอบพระคุณบิดา มารดา และครอบครัว ที่ให้การสนับสนุนทั้งในด้านทุนทรัพย์และเป็นกำลังใจสำคัญที่ผลักดันให้ผู้จัดทำมีความมุ่งมั่นและอดทน จนสามารถฟันฝ่าอุปสรรคต่าง ๆ และทำโครงการนี้ได้สำเร็จ รวมถึงขอขอบคุณเพื่อน ๆ ทุกคนที่คอยให้ความช่วยเหลือ แลกเปลี่ยนความคิดเห็น และเป็นกำลังใจให้แก่กันเสมอมา

คณะผู้จัดทำ

สารบัญ

	หน้า
บทคัดย่อ	ก
Abstract	ข
กิตติกรรมประกาศ	ค
สารบัญ	ก
สารบัญตาราง	ฉ
สารบัญภาพ	ช
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญของปัญหา	1
1.2 วัตถุประสงค์ของโครงการ	2
1.3 ขอบเขตของโครงการ	2
1.4 ประโยชน์ที่คาดว่าจะได้รับ	4
1.5 ขั้นตอนการศึกษา	4
1.6 นิยามศัพท์และคำย่อ	6
1.7 วางแผนการศึกษา	8
บทที่ 2 ทฤษฎีและเนื้อหาที่เกี่ยวข้อง	9
2.1 บทนำ	9
2.2 ทฤษฎีเกี่ยวกับระบบแชทบอท (Chatbot System)	9
2.3 ทฤษฎีเกี่ยวกับการเรียนรู้ของแชทบอท (Machine Learning for Chatbot)	12
2.4 การประมวลผลภาษาธรรมชาติ (Natural Language Processing - NLP)	18
2.5 ระบบจัดการฐานข้อมูล (Database Management System)	18
2.6 สถาปัตยกรรมเว็บแอปพลิเคชัน (Web Application Architecture)	18
2.7 การเชื่อมต่อผ่าน API (RESTful Services)	21
2.8 การออกแบบส่วนติดต่อผู้ใช้ (UI/UX Design)	22
2.9 เครื่องมือและซอฟต์แวร์ที่ใช้ในการพัฒนา	24
บทที่ 3 ขั้นตอนและวิธีการดำเนินงาน	33
3.1 ภาพรวมกระบวนการพัฒนา	33
3.2 ขั้นตอนที่ 1: วิเคราะห์ความต้องการและออกแบบระบบ	33

สารบัญ(ต่อ)

	หน้า
3.3 ขั้นตอนที่ 2: จัดเตรียมฐานข้อมูล	45
3.4 ขั้นตอนที่ 3: พัฒนาระบบ Backend และ Frontend	46
3.5 ขั้นตอนที่ 4: พัฒนาระบบ AI/ML	53
3.6 ขั้นตอนที่ 5: ทดสอบและปรับปรุงระบบ	53
3.7 ปัญหา อุปสรรค และแนวทางแก้ไข	57
3.8 การประเมินผลและบทเรียนที่ได้รับ	58
บทที่ 4 ผลการดำเนินงาน	59
4.1 ผลการพัฒนาระบบ	59
4.2 ผลการทดสอบระบบ	73
4.3 ผลการประเมินประสิทธิภาพ	78
4.4 ตัวอย่างการใช้งานระบบ	80
บทที่ 5 สรุปผลการดำเนินงาน อภิปรายผล และข้อเสนอแนะ	85
5.1 บทนำ	85
5.2 สรุปผลการดำเนินงาน	85
5.3 ประโยชน์ที่ได้รับจากโครงการ	87
5.4 อุปสรรคปัญหาและแนวทางแก้ไข	88
5.5 ข้อเสนอแนะและแนวทางการพัฒนา	89
เอกสารอ้างอิง	91
ภาคผนวก	92
ภาคผนวก ก.	93
ประวัติผู้เขียน	104

สารบัญตาราง

ตารางที่		หน้า
1.1	แผนการดำเนินงาน	8
2.1	สรุปเครื่องมือและซอฟต์แวร์ที่ใช้ในการพัฒนา	24
3.1	ปัญหาที่พบและวิธีแก้ไข	54
3.2	Load Testing	55
3.3	Penetration Testing	56
4.1	สรุปข้อมูลในระบบ	60
4.2	รายละเอียดโมเดล AI	66
4.3	การกระจายข้อมูลฝึกสอนแยกตามประเภทเจตนา (Intent)	66
4.4	เกณฑ์การให้คะแนนความเกี่ยวข้อง	68

สารบัญภาพ

ภาพที่		หน้า
2.1	Model-View-Controller	20
2.2	PHP 8.0+	25
2.3	Python 3.8+	25
2.4	pythainlp 4.0.2	26
2.5	scikit-learn 1.3.0	26
2.6	pandas 2.0.3	26
2.7	Flask 2.3.2	27
2.8	numpy 1.24.3	27
2.9	JavaScript (ES6+)	28
2.10	HTML5 & CSS3	28
2.11	MariaDB 10.4	29
2.12	เว็บเซิร์ฟเวอร์ (Web Server) XAMPP	29
2.13	Visual Studio Code	30
2.14	PHP Intelephense	30
2.15	Python Extension	30
2.16	Prettier	31
2.17	Live Server	31
2.18	phpMyAdmin	31
2.19	Postman	32
2.20	Draw.io	32
3.1	Use Case Diagram ระบบแชทบอทคณะวิศวกรรมศาสตร์	35
3.2	แผนภาพสถาปัตยกรรม 3 ชั้น (3-Tier Architecture)	37
3.3	ER Diagram - แผนภาพความสัมพันธ์ฐานข้อมูล	40
3.4	Activity Diagram – กระบวนการทำงานของระบบแชทบอท	42
3.5	หน้า UI Chatbot	48
3.6	หน้า Admin Dashboard	49
4.1	หน้าจอแชทบอทหลัก แสดงการสนทนาและปุ่มคำถามแนะนำ	61

สารบัญภาพ(ต่อ)

ภาพที่		หน้า
4.2	หน้าจอเซพบอทโหมดกลางคืน (Dark Mode)	61
4.3	แถบด้านข้าง (Sidebar) แสดงโลโก้คณะ ปุ่มสนทนาใหม่ และเมนูตั้งค่า	62
4.4	ตัวอย่างปุ่มตัวเลือกอัจฉริยะ (Smart Suggestion Buttons) ท้ายคำตอบ	63
4.5	ตัวอย่างการ์ดข่าวสารที่แสดงในแชท	63
4.6	ตัวอย่างการพิมพ์ "ค่าเทอม" แสดงตัวเลือก 13 สาขาวิชา	64
4.7	ตัวอย่างการพิมพ์ "ทุนการศึกษา" แสดงรายการคำถามที่เกี่ยวข้อง	65
4.8	กราฟแสดงการกระจายข้อมูลฝึกสอนแยกตามประเภทเจตนา	67
4.9	หน้าเข้าสู่ระบบ Admin	70
4.10	หน้า Dashboard แสดงสถิติภาพรวม	70
4.11	หน้าจัดการ FAQ — ตารางแสดงรายการพร้อมปุ่มแก้ไขและลบ	71
4.12	หน้าวิเคราะห์ (Analytics) แสดงสถิติการใช้งาน	72
4.13	Classification Report ของโมเดล AI	75
4.14	a กราฟแสดงผลการทำนายถูก-ผิดแยกรายประเภท	75
4.15	b ตารางรายละเอียดกรณีที่ทำนายผิด (Misclassifications)	76
4.16	ผลการรันชุดทดสอบอัตโนมัติ 140 รายการ แสดง PASS ทุกหมวด	77
4.17	กราฟสรุปผลการประเมินประสิทธิภาพโดยรวม	79
4.18	ตัวอย่างการถามเรื่องค่าเทอม — แสดง Broad Topic และคำตอบเฉพาะสาขา	80
4.19	ตัวอย่างการถามเรื่องอาชีพเฉพาะสาขา	81
4.20	ตัวอย่างการค้นหาข้อมูลบุคลากร	82
4.21	ตัวอย่างการถามเรื่องทุนการศึกษา กยศ.	83
4.22	ตัวอย่างการให้ Feedback ด้วยปุ่ม Satisfied / Dissatisfied	84
ก.1	การดาวน์โหลดตัวติดตั้ง Vs code	94
ก.2	เตรียมไฟล์ติดตั้ง	94
ก.3	การยอมรับข้อตกลงการใช้งาน (License Agreement)	95
ก.4	เลือกตำแหน่งการติดตั้ง (Select Destination Location)	95
ก.5	เลือกโฟลเดอร์สำหรับ Start Menu	95
ก.6	การเลือกงานเพิ่มเติม (Start Additional Tasks)	95

สารบัญภาพ(ต่อ)

ภาพที่		หน้า
ก.7	ตรวจสอบความพร้อมก่อนติดตั้ง (Ready to Install)	97
ก.8	กระบวนการติดตั้ง (Installing)	97
ก.9	เสร็จสิ้นการติดตั้ง	98
ก.10	หน้าจอเริ่มต้นของโปรแกรม	98
ก.11	การดาวน์โหลดตัวติดตั้ง xampp	99
ก.12	ดาวน์โหลดตัวติดตั้ง	99
ก.13	บันทึกไฟล์ติดตั้ง	100
ก.14	เริ่มต้นการติดตั้ง (Setup Wizard)	100
ก.15	เลือกส่วนประกอบ (Select Components)	101
ก.16	เลือกตำแหน่งติดตั้ง (Installation Folder)	101
ก.17	ดำเนินการติดตั้ง (Installing)	102
ก.18	เสร็จสิ้นการติดตั้ง	102
ก.19	การเปิดใช้งาน Server	103

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในยุคปัจจุบัน เทคโนโลยีปัญญาประดิษฐ์ (Artificial Intelligence: AI) และระบบประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) ได้รับการพัฒนาและนำมาใช้ในหลายด้าน เพื่อช่วยให้การสื่อสารระหว่างมนุษย์และเครื่องคอมพิวเตอร์เป็นไปอย่างมีประสิทธิภาพ หนึ่งในเทคโนโลยีที่ได้รับความนิยมคือ "ระบบแชทบอต (Chatbot)" ซึ่งสามารถโต้ตอบกับผู้ใช้งานได้อัตโนมัติผ่านข้อความ โดยมีจุดประสงค์เพื่อให้ข้อมูลและบริการต่าง ๆ ได้อย่างรวดเร็วและสะดวก

ภายในสถาบันการศึกษา โดยเฉพาะคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร มักมีข้อมูลที่นักศึกษา บุคลากร และบุคคลทั่วไปต้องการทราบอยู่เสมอ เช่น ข้อมูลหลักสูตรและสาขาวิชาต่างๆ (10 สาขา) ข้อมูลอาจารย์และบุคลากรกว่า 118 ท่าน ข่าวสารและกิจกรรมของคณะที่มีการอัปเดตอย่างต่อเนื่อง คำถามที่พบบ่อย (FAQ) เกี่ยวกับการสมัครเรียน ทนการศึกษา การกู้ยืมเงินนักศึกษา และข้อมูลการติดต่อหน่วยงานต่างๆ ซึ่งข้อมูลเหล่านี้มีการเปลี่ยนแปลงอยู่ตลอดเวลา การให้ข้อมูลด้วยวิธีการเดิม เช่น การโทรศัพท์สอบถามหรือการค้นหาผ่านเว็บไซต์ มักทำให้ผู้ใช้เสียเวลาในการค้นหาข้อมูลที่กระจัดกระจายอยู่ในหลายแหล่ง และเพิ่มภาระให้กับเจ้าหน้าที่ในการตอบคำถามซ้ำๆ ที่คล้ายกัน

ดังนั้นจึงมีแนวคิดในการพัฒนา "ระบบผู้ช่วยอัจฉริยะ (Smart Information Chatbot System) สำหรับคณะวิศวกรรมศาสตร์" โดยระบบใช้เทคโนโลยี Hybrid Chatbot ที่ผสมผสานระหว่าง Rule-based System และ Machine Learning (TF-IDF + Logistic Regression) เพื่อตอบคำถามได้อย่างแม่นยำ ระบบสามารถเชื่อมต่อกับฐานข้อมูล MySQL และดึงข้อมูลข่าวสารจากเว็บไซต์มหาวิทยาลัยแบบอัตโนมัติผ่าน Web Scraping เพื่อให้ข้อมูลที่ถูกต้องและเป็นปัจจุบันแก่ผู้ใช้งานได้ตลอดเวลา ระบบดังกล่าวจะช่วยอำนวยความสะดวกในการเข้าถึงข้อมูล ลดภาระงานเจ้าหน้าที่ เพิ่มประสิทธิภาพในการสื่อสาร และสร้างภาพลักษณ์ที่ทันสมัยให้กับคณะ นอกจากนี้ระบบยังเก็บบันทึกประวัติการสนทนา (Chat Logs) เพื่อนำไปวิเคราะห์พฤติกรรมผู้ใช้งาน วัดค่าความมั่นใจ (Confidence Score) ในการตอบคำถาม และเวลาในการตอบสนอง (Response Time) เพื่อพัฒนาการให้บริการอย่างต่อเนื่อง รวมถึงมีระบบผู้ดูแล (Admin Dashboard) สำหรับบริหารจัดการข้อมูล FAQ ข่าวสาร บุคลากร และสถิติการใช้งานได้อย่างเป็นระบบ

1.2 วัตถุประสงค์ของโครงการ

1.2.1 เพื่อพัฒนาระบบผู้ช่วยอัจฉริยะ (Chatbot) ที่สามารถตอบคำถามอัตโนมัติ (FAQ) เกี่ยวกับข้อมูลของคณะวิศวกรรมศาสตร์ เช่น ข้อมูลหลักสูตร ทุนการศึกษา ข่าวสาร และช่องทางการติดต่อ ได้อย่างถูกต้องและรวดเร็ว

1.2.2 เพื่อเชื่อมต่อและดึงข้อมูลจากเว็บไซต์และฐานข้อมูล MySQL ของคณะแบบอัตโนมัติ ทำให้ข้อมูลที่นำเสนอมีความถูกต้องและเป็นปัจจุบันอยู่เสมอ

1.2.3 เพื่อให้ผู้ใช้งานสามารถค้นหาข้อมูลบุคลากรหรืออาจารย์ทั้ง 118 ท่าน ได้อย่างสะดวก เช่น ชื่อ ตำแหน่ง สาขาวิชา อีเมล หมายเลขโทรศัพท์ และความเชี่ยวชาญ

1.2.4 เพื่อพัฒนาระบบ AI Engine ด้วย Machine Learning (TF-IDF + Logistic Regression) สำหรับการจำแนกเจตนา (Intent Classification) และตอบคำถามได้อย่างแม่นยำ

1.2.5 เพื่อสร้างระบบจัดเก็บและวิเคราะห์ข้อมูลการสนทนา (Chat Logs) พร้อมวัดค่า Confidence Score และ Response Time สำหรับใช้ปรับปรุงระบบอย่างต่อเนื่อง

1.2.6 เพื่อพัฒนาระบบผู้ดูแล (Admin Dashboard) ที่สามารถจัดการข้อมูล FAQ ข่าวสาร บุคลากร และตรวจสอบสถิติการใช้งานได้อย่างเป็นระบบ

1.3 ขอบเขตของโครงการ

1.3.1 ลักษณะของระบบ

โครงการนี้เป็นการพัฒนาระบบผู้ช่วยอัจฉริยะ (Smart Information Chatbot System) สำหรับคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร โดยใช้เทคโนโลยีเว็บแอปพลิเคชัน (Web Application) แบบ Hybrid Chatbot ที่สามารถทำงานได้ทั้งบนคอมพิวเตอร์และอุปกรณ์พกพา

1.3.2 ขอบเขตการทำงานของระบบ

1.3.2.1 ระบบสามารถตอบคำถามอัตโนมัติ (FAQ Chatbot) โดยใช้ Machine Learning (TF-IDF + Logistic Regression) ร่วมกับ Keyword Matching เพื่อดึงคำตอบจากฐานข้อมูล พร้อมแสดงค่า Confidence Score

1.3.2.2 ระบบสามารถดึงข่าวสารจากเว็บไซต์ของคณะวิศวกรรมศาสตร์แบบอัตโนมัติผ่าน Web Scraping และอัปเดตเข้าสู่ฐานข้อมูลตาราง news

1.3.2.3 ระบบสามารถค้นหาข้อมูลบุคลากรทั้ง 118 ท่าน จากฐานข้อมูลตาราง staff โดยแสดงข้อมูล ชื่อ (ไทย/อังกฤษ) ตำแหน่ง สาขา อีเมล เบอร์โทรศัพท์ ความเชี่ยวชาญ และ รูปภาพ

1.3.2.4 ระบบสามารถให้ข้อมูลการติดต่อคณะ เช่น หมายเลขโทรศัพท์ อีเมล เฟซบุ๊ก และเว็บไซต์ พร้อมแสดงลิงก์ที่คลิกได้

1.3.2.5 ระบบสามารถบันทึกประวัติการสนทนา (Chat Logs) ลงในตาราง chat_logs พร้อมข้อมูล session_id, user_message, bot_response, sources, confidence, response_time_ms, user_ip, และ user_agent

1.3.2.6 มีส่วนติดต่อผู้ใช้ (User Interface) ที่ใช้งานง่าย ทันสมัย รองรับ Responsive Design สำหรับทั้งคอมพิวเตอร์และสมาร์ทโฟน

1.3.3 ขอบเขตด้านผู้ใช้งาน

1.3.3.1 ผู้ใช้งานทั่วไป (นักศึกษา บุคลากร และผู้สนใจทั่วไป) สามารถใช้งานแชทบอตเพื่อสอบถามข้อมูลต่างๆ ได้ผ่านเว็บเบราว์เซอร์

1.3.3.2 ผู้ดูแลระบบ (Administrator) สามารถจัดการข้อมูลภายใน เช่น เพิ่ม/แก้ไข/ลบ FAQ ข่าวสาร บุคลากร และดูสถิติการใช้งานได้ผ่าน Admin Dashboard

1.3.4 ขอบเขตด้านเทคนิคและการพัฒนา

1.3.4.1 ระบบพัฒนาด้วย Frontend (HTML5, CSS3, JavaScript) และ Backend (PHP 8.0+ พร้อม PDO สำหรับจัดการฐานข้อมูล)

1.3.4.2 ฐานข้อมูล MySQL (utf8mb4) สำหรับจัดเก็บข้อมูล 4 ตารางหลัก คือ staff (บุคลากร), faq (คำถาม-คำตอบ), news (ข่าวสาร) และ chat_logs (ประวัติการสนทนา)

1.3.4.3 พัฒนาระบบ AI Engine ด้วย Python 3.x + Flask Framework ใช้เทคนิค TF-IDF (Term Frequency-Inverse Document Frequency) ร่วมกับ Logistic Regression จาก Scikit-learn สำหรับการจำแนกเจตนา (Intent Classification)

1.3.4.4 มีการเชื่อมต่อกับเว็บไซต์มหาวิทยาลัยเพื่อดึงข้อมูลข่าวสารแบบอัตโนมัติผ่าน Web Scraping Script (Python + BeautifulSoup)

1.3.4.5 ระบบใช้สถาปัตยกรรมแบบ 3-Tier Architecture (Presentation Layer, Application Layer, Data Layer) และรองรับ RESTful API สำหรับการสื่อสารระหว่าง Frontend และ Backend

1.3.4.6 ระบบทำงานบน Apache Web Server (XAMPP Stack) และสามารถทดสอบผ่านเว็บเบราว์เซอร์มาตรฐาน เช่น Google Chrome, Microsoft Edge, Safari รองรับ Responsive Design สำหรับอุปกรณ์ทุกขนาด

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 อำนวยความสะดวกให้กับนักศึกษา บุคลากร และผู้สนใจทั่วไป ในการค้นหาข้อมูลข่าวสาร บุคลากร และช่องทางการติดต่อของคณะวิศวกรรมศาสตร์ได้อย่างรวดเร็วและถูกต้องตลอด 24 ชั่วโมง

1.4.2 ลดภาระงานของเจ้าหน้าที่ในการตอบคำถามซ้ำๆ และเพิ่มประสิทธิภาพในการให้บริการข้อมูลแก่ผู้ใช้งานได้มากกว่า 80% จากเดิม

1.4.3 ช่วยให้คณะวิศวกรรมศาสตร์มีระบบบริการข้อมูลที่ทันสมัย สอดคล้องกับแนวทางของมหาวิทยาลัยดิจิทัล (Digital University) และเพิ่มภาพลักษณ์ความเป็นองค์กรอัจฉริยะ (Smart Organization)

1.4.4 สามารถนำข้อมูลจากประวัติการสนทนา (Chat Logs) พร้อมค่า Confidence Score และ Response Time มาวิเคราะห์ เพื่อพัฒนาระบบตอบคำถามให้มีความแม่นยำและครอบคลุมมากยิ่งขึ้น

1.4.5 เป็นแหล่งเรียนรู้และตัวอย่างการประยุกต์ใช้เทคโนโลยีปัญญาประดิษฐ์ (AI) Machine Learning และระบบประมวลผลภาษาธรรมชาติ (NLP) ในการพัฒนาแอปพลิเคชันที่สามารถนำไปต่อยอดในงานอื่นๆ ได้

1.4.6 สนับสนุนให้นักศึกษาผู้จัดทำโครงการได้พัฒนาทักษะด้านการวิเคราะห์ ออกแบบ และพัฒนาระบบจริง รวมถึงการทำงานร่วมกันเป็นทีมในรูปแบบของโครงการเทคโนโลยีสารสนเทศครบวงจร

1.5 ขั้นตอนการศึกษา

การดำเนินโครงการ "ระบบผู้ช่วยอัจฉริยะ (Smart Information Chatbot System) สำหรับคณะวิศวกรรมศาสตร์" มีขั้นตอนการศึกษาดำเนินงานดังนี้

1.5.1 ศึกษาและวิเคราะห์ข้อมูลเบื้องต้น

1.5.1.1 ศึกษาข้อมูลพื้นฐานเกี่ยวกับเทคโนโลยีปัญญาประดิษฐ์ (AI), ระบบประมวลผลภาษาธรรมชาติ (NLP), Machine Learning (TF-IDF, Logistic Regression) และการทำงานของ Hybrid Chatbot

1.5.1.2 วิเคราะห์ระบบการให้ข้อมูลของคณะวิศวกรรมศาสตร์ในปัจจุบัน เพื่อระบุปัญหาและข้อจำกัดของวิธีการให้บริการเดิม

1.5.1.3 ศึกษาความต้องการของผู้ใช้งาน (นักศึกษา บุคลากร และผู้สนใจทั่วไป) เพื่อกำหนดขอบเขตและฟังก์ชันของระบบ

1.5.2 ออกแบบระบบ

1.5.2.1 ออกแบบโครงสร้างระบบ (System Architecture) แบบ 3-Tier ให้เหมาะสมกับการเชื่อมต่อฐานข้อมูล MySQL และเว็บไซต์ของมหาวิทยาลัย

1.5.2.2 ออกแบบฐานข้อมูล (Database Design) สำหรับจัดเก็บ 4 ตารางหลัก: staff, faq, news, chat_logs พร้อมกำหนด Primary Key, Foreign Key และ Index

1.5.2.3 ออกแบบ AI Engine Pipeline ตั้งแต่ Text Preprocessing, TF-IDF Vectorization, Model Training ไปจนถึง Prediction & Response Generation

1.5.2.4 ออกแบบส่วนติดต่อผู้ใช้ (User Interface Design) ให้ใช้งานง่าย ทันสมัย รองรับ Responsive Design สำหรับทั้งคอมพิวเตอร์และอุปกรณ์พกพา

1.5.3 พัฒนาระบบ

1.5.3.1 พัฒนา Backend API ด้วย PHP 8.0+ พร้อม PDO สำหรับจัดการฐานข้อมูล MySQL และเชื่อมต่อกับ AI Engine

1.5.3.2 พัฒนา AI Engine ด้วย Python Flask ใช้ Scikit-Learn สำหรับ TF-IDF + Logistic Regression และ pythainlp สำหรับ Thai Tokenization

1.5.3.3 พัฒนาโมดูล Web Scraping (Python + BeautifulSoup) เพื่อดึงข้อมูลข่าวสารจากเว็บไซต์คณะวิศวกรรมศาสตร์แบบอัตโนมัติ

1.5.3.4 พัฒนา Frontend Interface (HTML5, CSS3, JavaScript) ให้มีลักษณะคล้ายระบบสนทนา (Chat Interface) ที่ทันสมัยและใช้งานง่าย

1.5.3.5 พัฒนาระบบผู้ดูแล (Admin Dashboard) สำหรับจัดการข้อมูล FAQ ข่าวสาร บุคลากร และดูสถิติการใช้งาน Chat Logs พร้อม Confidence Score และ Response Time

1.5.4 ทดสอบระบบ (System Testing)

1.5.4.1 ทดสอบการทำงานของแต่ละส่วน เช่น การตอบคำถาม (Accuracy Testing) การค้นหาข้อมูลบุคลากร และการดึงข่าวสาร (Web Scraping)

1.5.4.2 ทดสอบ AI Model โดยวัด Accuracy, Precision, Recall, F1-Score จากชุดข้อมูลทดสอบ (Test Dataset)

1.5.4.3 ทดสอบการเชื่อมต่อกับฐานข้อมูล MySQL และเว็บไซต์ของมหาวิทยาลัย รวมถึงทดสอบ Response Time ของระบบ

1.5.4.4 ทดสอบการใช้งานบนอุปกรณ์และเบราว์เซอร์ต่างๆ (Cross-browser & Cross-device Testing) เพื่อประเมินความถูกต้องและเสถียรของระบบ

1.5.5 ประเมินผลและปรับปรุงระบบ

1.5.5.1 รวบรวมความคิดเห็นจากผู้ใช้งานจริง (User Acceptance Testing) เพื่อประเมินความพึงพอใจและประสิทธิภาพของระบบ

1.5.5.2 ปรับปรุงระบบให้ตอบสนองต่อความต้องการของผู้ใช้มากยิ่งขึ้น โดยอิงจากข้อเสนอแนะและผลการทดสอบ

1.5.5.3 วิเคราะห์ข้อมูลจาก Chat Logs พร้อมค่า Confidence Score เพื่อระบุคำถามที่ระบบตอบได้ไม่ดี และเพิ่ม FAQ ใหม่เข้าระบบ

1.5.5.4 ปรับปรุง AI Model ด้วยการ Retrain ด้วยข้อมูล FAQ ที่เพิ่มเติม เพื่อเพิ่มความแม่นยำของระบบ

1.5.6 จัดทำรายงานและสรุปผลโครงการ

1.5.6.1 สรุปผลการพัฒนาและทดสอบระบบ

1.5.6.2 วัดผลประสิทธิภาพของระบบ เช่น ความแม่นยำในการตอบคำถาม (Accuracy) เวลาตอบสนอง (Response Time) และความพึงพอใจของผู้ใช้งาน

1.5.6.3 จัดทำรายงานโครงการฉบับสมบูรณ์ พร้อมนำเสนอผลการดำเนินงาน

1.6 นิยามศัพท์และคำย่อ

1.6.1 AI (Artificial Intelligence) - ปัญญาประดิษฐ์ เทคโนโลยีที่ให้คอมพิวเตอร์สามารถเลียนแบบความสามารถของมนุษย์ในการคิด เรียนรู้ และตัดสินใจ

1.6.2 NLP (Natural Language Processing) - การประมวลผลภาษาธรรมชาติ เทคนิคที่ช่วยให้คอมพิวเตอร์เข้าใจ ตีความ และตอบสนองต่อภาษามนุษย์

1.6.3 Chatbot - ระบบแชทบอท โปรแกรมที่ออกแบบมาเพื่อสนทนาหรือโต้ตอบกับผู้ใช้งานผ่านข้อความอัตโนมัติ

1.6.4 FAQ (Frequently Asked Questions) - คำถามที่พบบ่อย ชุดคำถามและคำตอบที่ผู้ใช้งานมักสอบถามเป็นประจำ

1.6.5 Machine Learning - การเรียนรู้ของเครื่อง เทคนิคที่ทำให้คอมพิวเตอร์สามารถเรียนรู้จากข้อมูลและปรับปรุงประสิทธิภาพโดยไม่ต้องถูกโปรแกรมโดยตรง

1.6.6 TF-IDF (Term Frequency-Inverse Document Frequency) - เทคนิคการวัดความสำคัญของคำในเอกสาร โดยพิจารณาความถี่ของคำและความหายากของคำนั้นในชุดเอกสารทั้งหมด

1.6.7 Logistic Regression - อัลกอริทึมการเรียนรู้แบบมีผู้สอน (Supervised Learning) ที่ใช้สำหรับการจำแนกประเภท (Classification)

1.6.8 Web Scraping - เทคนิคการดึงข้อมูลจากเว็บไซต์โดยอัตโนมัติ

1.6.9 RESTful API (Representational State Transfer API) - รูปแบบสถาปัตยกรรมการสื่อสารระหว่างระบบผ่าน HTTP Protocol

1.6.10 Admin Dashboard - ระบบหลังบ้านสำหรับผู้ดูแลระบบในการจัดการข้อมูลและตรวจสอบสถิติการใช้งาน

1.6.11 Chat Logs - บันทึกประวัติการสนทนาระหว่างผู้ใช้งานและระบบแชทบอท

1.6.12 Confidence Score - ค่าความมั่นใจที่ระบบมีต่อคำตอบที่ให้ โดยมีค่าระหว่าง 0.00 ถึง 1.00

1.6.13 Response Time - ระยะเวลาที่ระบบใช้ในการประมวลผลและตอบกลับคำถาม วัดเป็นมิลลิวินาที (ms)

1.6.14 RMUTP - มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร (Rajamangala University of Technology Phra Nakhon)

1.6.15 Hybrid Chatbot - แชทบอทแบบผสมผสานที่ใช้ทั้ง Rule-based และ AI-powered เพื่อเพิ่มประสิทธิภาพการตอบคำถาม

1.6.16 Intent Classification - การจำแนกเจตนาของผู้ใช้งาน เพื่อระบุว่าผู้ใช้ต้องการถามหรือขอข้อมูลเรื่องใด

1.6.17 Responsive Design - การออกแบบเว็บไซต์ให้สามารถแสดงผลได้เหมาะสมกับทุกขนาดหน้าจอ

1.6.18 3-Tier Architecture - สถาปัตยกรรมระบบ 3 ชั้น ประกอบด้วย Presentation Layer, Application Layer และ Data Layer

1.7 วางแผนการศึกษา

	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.
	2568			2569	
ศึกษาข้อมูลและวิเคราะห์ระบบ					
ออกแบบระบบ (System Design)					
พัฒนาโมดูล Chatbot					
พัฒนาโมดูลดึงข่าว (Fetch News)					
พัฒนาโมดูลค้นหาบุคลากร (Staff Info)					
พัฒนา UI ของแชทบอต					
พัฒนาระบบผู้ดูแล (Admin Dashboard)					
ทดสอบระบบ (Testing)					
ประเมินผลและปรับปรุง					
จัดทำรายงานและสรุปผล					

ตารางที่ 1.1 แผนการดำเนินงาน

บทที่ 2

ทฤษฎีและเนื้อหาที่เกี่ยวข้อง

2.1 บทนำ

ในการพัฒนาระบบ [Smart Information Chatbot System of Engineering] คณะผู้จัดทำได้ศึกษาค้นคว้าหลักการ ทฤษฎี และเทคโนโลยีที่เกี่ยวข้อง เพื่อนำมาประยุกต์ใช้ในการออกแบบและพัฒนาระบบให้มีประสิทธิภาพ สามารถตอบสนองความต้องการของผู้ใช้งานได้อย่างถูกต้องแม่นยำ โดยเนื้อหาในบทนี้จะครอบคลุมองค์ความรู้ตั้งแต่พื้นฐานกระบวนการทำงานของ แชทบอท, การประมวลผลภาษาธรรมชาติ (NLP) เพื่อให้ระบบเข้าใจภาษาไทย, เทคนิคการสืบค้นและจัดเก็บข้อมูลอย่างเป็นระบบ, สถาปัตยกรรมของเว็บแอปพลิเคชันและการเชื่อมต่อผ่าน API, ตลอดจนหลักการออกแบบส่วนติดต่อผู้ใช้ (UI/UX) เพื่อสร้างประสบการณ์การใช้งานที่ดี ซึ่งมีรายละเอียดดังต่อไปนี้

2.2. ทฤษฎีเกี่ยวกับระบบแชทบอท (Chatbot System)

2.2.1 ประเภทของแชทบอท: อธิบายความแตกต่างระหว่าง Rule-based (ทำงานตามกฎ), AI-powered (ใช้ Machine Learning) และ Hybrid (ลูกผสม) ซึ่งโปรเจกต์นี้อาจจะเน้นแบบ Hybrid หรือ Rule-based ที่มีความซับซ้อน

2.2.1.1 Rule-based Chatbot (แชทบอทแบบอิงกฎ) แชทบอทประเภทนี้ทำงานโดยอาศัยชุดคำสั่งหรือกฎเกณฑ์ที่ถูกกำหนดไว้อย่างตายตัว (Predefined Rules) ภายใต้โครงสร้างแบบผังต้นไม้การตัดสินใจ (Decision Tree) ระบบจะทำหน้าที่โต้ตอบกับผู้ใช้งานตามเส้นทางที่ถูกออกแบบไว้ล่วงหน้าเท่านั้น

2.2.1.2 AI-powered Chatbot (แชทบอทปัญญาประดิษฐ์) หรือเรียกว่า Conversational AI คือแชทบอทที่ขับเคลื่อนด้วยเทคโนโลยีปัญญาประดิษฐ์ โดยอาศัยการประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) และการเรียนรู้ของเครื่อง (Machine Learning) เพื่อทำความเข้าใจเจตนา (Intent) ของผู้ใช้งาน

2.2.1.3 Hybrid Chatbot (แชทบอทแบบลูกผสม) คือการบูรณาการเทคโนโลยีระหว่างแชทบอทแบบอิงกฎ (Rule-based) และแชทบอทปัญญาประดิษฐ์ (AI-powered) เข้าด้วยกัน เพื่อดึงจุดเด่นของทั้งสองระบบมาใช้ในการเพิ่มประสิทธิภาพการให้บริการสูงสุด

2.2.2 องค์ประกอบหลัก (Architecture)

โครงสร้างการทำงานของแชทบอทโดยทั่วไปประกอบด้วย 4 ส่วนสำคัญที่ทำงานประสานกันเพื่อแปลงข้อมูลนำเข้า (Input) จากมนุษย์ ให้เป็นการตอบสนอง (Response) ที่เหมาะสม ดังนี้

2.2.2.1 ส่วนติดต่อผู้ใช้ (User Interface - UI) ส่วนติดต่อผู้ใช้ หรือ UI คือช่องทาง (Channel) ที่ทำหน้าที่เป็นสื่อกลางในการรับส่งข้อมูลระหว่างผู้ใช้งานกับระบบคอมพิวเตอร์ ทำหน้าที่รับข้อมูลนำเข้า (Input) ซึ่งอาจอยู่ในรูปแบบข้อความ (Text) หรือเสียง (Voice) และแสดงผลลัพธ์ (Output) กลับไปยังผู้ใช้

1. ทฤษฎีที่เกี่ยวข้อง: หลักการปฏิสัมพันธ์ระหว่างมนุษย์และคอมพิวเตอร์ (Human-Computer Interaction: HCI) และการออกแบบประสบการณ์ผู้ใช้ (User Experience: UX)

2. รูปแบบการทำงาน: ในปัจจุบัน UI มักจะเชื่อมต่อผ่าน Platform ต่างๆ เช่น Web Widget, Mobile Application หรือ Social Media Platform (Line, Facebook Messenger) ผ่านทาง API (Application Programming Interface)

2.2.2.2 ส่วนประมวลผลภาษาธรรมชาติ (NLP Engine) NLP (Natural Language Processing) คือกลไกหลักที่เปรียบเสมือนสมองของแชทบอท ทำหน้าที่วิเคราะห์และทำความเข้าใจภาษามนุษย์ที่ไม่มีรูปแบบตายตัว (Unstructured Data) ให้เป็นข้อมูลที่คอมพิวเตอร์สามารถนำไปประมวลผลต่อได้ โดยแบ่งกระบวนการย่อยเป็น

1. Natural Language Understanding (NLU): การทำความเข้าใจภาษา ประกอบด้วย

- Intent Classification: การจำแนกเจตนาของผู้ใช้ว่าต้องการสื่อสารเรื่องอะไร
- Entity Extraction: การสกัดคำสำคัญหรือข้อมูลจำเพาะจากประโยค เช่น วันที่, สถานที่, ชื่อสินค้า

2. Natural Language Generation (NLG): การสร้างประโยคตอบกลับให้เป็นภาษาธรรมชาติที่มนุษย์เข้าใจได้

2.2.2.3 ส่วนจัดการบทสนทนาและสถานะ (Dialog Manager) Dialog Manager (DM) ทำหน้าที่เป็นหน่วยควบคุมตรรกะ (Logic) ของการสนทนา เพื่อกำหนดว่าระบบควรจะตอบโต้ผู้ใช้งานอย่างไร โดยอาศัยข้อมูลที่ได้จาก NLP

1. หน้าที่หลัก

- State Tracking: การติดตามสถานะของการสนทนาว่าปัจจุบันคุยถึงขั้นตอนใด

- Context Management: การจดจำบริบทและข้อมูลก่อนหน้า (History) เพื่อให้การสนทนาต่อเนื่อง

2. ประเภทของระบบ: อาจใช้ระบบแบบกฎเกณฑ์ (Rule-based) สำหรับคำถามง่ายๆ หรือใช้ระบบ Machine Learning สำหรับบทสนทนาที่มีความซับซ้อน (Contextual Conversation)

2.2.2.4 คลังความรู้และฐานข้อมูล (Knowledge Base/Database) ส่วนจัดเก็บข้อมูลสารสนเทศที่แชทบอทต้องนำมาใช้ในการประมวลผลหรือตอบคำถาม แบ่งเป็น 2 ลักษณะ คือ

1. Knowledge Base (KB): คลังความรู้ที่มักเก็บข้อมูลแบบไม่มีโครงสร้าง หรือกึ่งโครงสร้าง เช่น ไฟล์เอกสาร, คู่มือการใช้งาน, หรือชุดคำถาม-คำตอบ (FAQ) ที่เตรียมไว้

2. Database: ฐานข้อมูลที่มีโครงสร้างชัดเจน (Structured Data) ใช้เก็บข้อมูลธุรกรรม (Transaction) เช่น ข้อมูลสมาชิก, ประวัติการสั่งซื้อ หรือสต็อกสินค้า เพื่อให้ Dialog Manager ดึงไปใช้งานแบบ Real-time

2.2.3 ประโยชน์ของระบบแชทบอทต่อองค์กร การนำเทคโนโลยีแชทบอทมาประยุกต์ใช้ในระบบงานทางธุรกิจ ก่อให้เกิดประโยชน์ต่อองค์กรในหลายมิติ ทั้งด้านการบริหารจัดการต้นทุน การยกระดับการให้บริการ และการวางแผนกลยุทธ์ โดยสามารถจำแนกประเด็นสำคัญได้ดังนี้

2.2.3.1 การลดต้นทุนและการใช้ทรัพยากรบุคคล (Cost and Resource Optimization) แชทบอทช่วยลดภาระค่าใช้จ่ายในการจ้างพนักงานสำหรับตอบคำถามพื้นฐานซ้ำๆ (Routine Inquiries) ทำให้องค์กรสามารถบริหารจัดการทรัพยากรบุคคลให้ไปโฟกัสกับงานที่มีความซับซ้อนหรืองานเชิงวิเคราะห์ที่ต้องใช้ทักษะมนุษย์ได้มากขึ้น ส่งผลให้ต้นทุนต่อหน่วยในการให้บริการลูกค้า (Cost per Contact) ลดลงในระยะยาว

2.2.3.2 การให้บริการที่ต่อเนื่องตลอด 24 ชั่วโมง (24/7 Availability) ข้อจำกัดเรื่องเวลาทำงานของมนุษย์ไม่ส่งผลกระทบต่อแชทบอท ทำให้องค์กรสามารถให้บริการลูกค้าได้ตลอด 24 ชั่วโมง โดยไม่มีวันหยุด ซึ่งเป็นปัจจัยสำคัญในการสร้างความพึงพอใจและตอบโต้พฤติกรรมผู้บริโภคในยุคดิจิทัลที่ต้องการความช่วยเหลือแบบทันทีทันใด (Real-time)

2.2.3.3 การรองรับผู้ใช้งานจำนวนมากพร้อมกัน (High Scalability) ระบบแชทบอทสามารถรองรับการสนทนากับผู้ใช้งานจำนวนมากได้ในเวลาเดียวกัน (Concurrent Users) โดยไม่ส่งผลกระทบต่อความเร็วในการตอบกลับ แตกต่างจากพนักงาน (Human Agent) ที่สามารถดูแลลูกค้าได้ทีละ 1-2 รายเท่านั้น ช่วยแก้ปัญหาคอขวดในช่วงเวลาที่มีปริมาณการติดต่อสูง (Peak Hours)

2.2.3.4 ความรวดเร็วและความถูกต้องของข้อมูล (Speed and Accuracy) แชนบอทสามารถสืบค้นข้อมูลจากฐานข้อมูลและตอบกลับได้ในเวลาเสี้ยววินาที ช่วยลดระยะเวลาการรอคอย (Wait Time) ของลูกค้า นอกจากนี้ ข้อมูลที่ตอบกลับยังมีความถูกต้องและเป็นมาตรฐานเดียวกัน (Standardization) ตามที่โปรแกรมไว้ ช่วยลดความผิดพลาดที่เกิดจากมนุษย์ (Human Error)

2.2.3.5 การเก็บรวบรวมและวิเคราะห์ข้อมูลลูกค้า (Data Collection and Analytics) การสนทนาผ่านแชทบอทเป็นแหล่งข้อมูล Big Data ที่สำคัญ องค์กรสามารถนำประวัติการสนทนา (Chat Logs) ไปวิเคราะห์เพื่อทำความเข้าใจพฤติกรรม ความสนใจ และปัญหาของลูกค้า (Pain Points) นำไปสู่การปรับปรุงสินค้า บริการ หรือการทำการตลาดแบบเฉพาะเจาะจง (Personalization) ได้อย่างแม่นยำ

2.2.3.6 การสร้างภาพลักษณ์ที่ทันสมัย (Modern Brand Image) การนำเทคโนโลยีปัญญาประดิษฐ์มาใช้ในการบริการลูกค้า ช่วยส่งเสริมภาพลักษณ์ขององค์กรให้ดูทันสมัย มีนวัตกรรม และแสดงถึงความใส่ใจในการพัฒนาช่องทางการสื่อสารเพื่ออำนวยความสะดวกให้แก่ผู้ใช้บริการ

2.3 ทฤษฎีเกี่ยวกับการเรียนรู้ของแชทบอท (Machine Learning for Chatbot)

2.3.1 ทฤษฎีการเรียนรู้ของแชทบอท (Chatbot Learning Theories) การที่แชทบอทสามารถเข้าใจและตอบโต้ภาษามนุษย์ได้นั้น อาศัยทฤษฎีและกระบวนการเรียนรู้ของเครื่อง (Machine Learning) ซึ่งเป็นการทำให้คอมพิวเตอร์เรียนรู้จากข้อมูล (Data-driven) โดยมีทฤษฎีและโมเดลที่เกี่ยวข้องหลักๆ

2.3.1.1 การเรียนรู้แบบมีผู้สอน (Supervised Learning) เป็นทฤษฎีพื้นฐานที่สุดที่ใช้ในการสร้างแชทบอทประเภท NLU (Natural Language Understanding)

1. หลักการ: การป้อนข้อมูลเข้าไปในระบบประกอบด้วย "ข้อมูลนำเข้า" (Input) และ "คำตอบที่ถูกต้อง" (Label/Target) เพื่อให้โมเดลเรียนรู้ความสัมพันธ์

2. การประยุกต์ใช้ในแชทบอท

- Intent Classification: สอนให้บอทจำแนกเจตนา เช่น ป้อนประโยค "อยากกินข้าว" พร้อมติดป้ายกำกับว่าเป็น Intent "สั่งอาหาร" เมื่อมีข้อมูลมากพอ บอทจะแยกแยะประโยคใหม่ๆ

- Entity Recognition: การสอนให้บอทระบุคำเฉพาะ เช่น ชื่อคน สถานที่ หรือ เวลาจากประโยคตัวอย่าง

2.3.1.2 การเรียนรู้เชิงลึก (Deep Learning) และโครงข่ายประสาทเทียม สำหรับ การประมวลผลภาษาที่ซับซ้อน ปัจจุบันนิยมใช้โครงข่ายประสาทเทียม (Neural Networks) ที่จำลอง การทำงานของสมองมนุษย์

- Recurrent Neural Networks (RNN) และ LSTM (Long Short-Term Memory): เป็นโมเดลยุคแรกๆ ที่ออกแบบมาเพื่อจัดการข้อมูลแบบลำดับ (Sequence Data) เช่น ประโยคสนทนา โดยมีความสามารถในการ "จดจำ" คำก่อนหน้าเพื่อทำความเข้าใจบริบทของคำถัดไป

- Transformer Architecture: (ทฤษฎี เบื้องหลัง ChatGPT/Gemini) เป็น สถาปัตยกรรมใหม่ที่ใช้กลไก Self-Attention ช่วยให้โมเดลสามารถเชื่อมโยงความสัมพันธ์ของคำทุก คำในประโยคได้ พร้อมกัน ไม่ว่าจะอยู่ห่างกันแค่ไหน ทำให้เข้าใจบริบทได้ลึกซึ้งกว่า RNN

2.3.1.3 การเรียนรู้แบบถ่ายโอน (Transfer Learning) ทฤษฎีนี้ช่วยลดข้อจำกัดเรื่อง ข้อมูลสอน (Training Data) ที่มีจำนวนน้อย

- หลักการ: การนำโมเดลภาษาขนาดใหญ่ (Pre-trained Language Models) ที่ ผ่านการเรียนรู้จากข้อมูลมหาศาลในอินเทอร์เน็ตมาแล้ว (เช่น BERT, GPT) มาทำการปรับจูน (Fine-tuning) ด้วยข้อมูลเฉพาะทางขององค์กร

- ประโยชน์: ช่วยให้สร้างแชทบอทที่ฉลาดได้เร็วขึ้น โดยไม่ต้องเริ่มสอนภาษาตั้งแต่ ศูนย์

2.3.1.4 การเรียนรู้แบบเสริมกำลัง (Reinforcement Learning) มักใช้ในขั้นตอน การปรับปรุงความถี่ของการสนทนา หรือในแชทบอทขั้นสูง (Generative AI)

- หลักการ: การเรียนรู้จากการลองผิดลองถูก (Trial and Error) โดยระบบจะได้รับ "รางวัล" (Reward) เมื่อตอบถูกหรือทำให้บทสนทนาดำเนินไปได้ด้วยดี และได้รับ "บทลงโทษ" (Penalty) เมื่อตอบผิด

- RLHF (Reinforcement Learning from Human Feedback): การให้มนุษย์เข้ามา ช่วยให้คะแนนคำตอบของบอท เพื่อจูนให้บอทตอบได้ตรงใจและเป็นธรรมชาติที่สุด

2.3.1.5 การวัดผลโมเดลการเรียนรู้ (Model Evaluation) ในเชิงทฤษฎี การเรียนรู้ ต้องมีการวัดผลเพื่อยืนยันประสิทธิภาพ ค่าทางสถิติที่นิยมใช้ได้แก่

- Confusion Matrix: ตารางแสดงความถูกต้องและความผิดพลาดในการทำนาย
- Accuracy: ความแม่นยำโดยรวม
- F1-Score: ค่าเฉลี่ย Harmonic ระหว่าง Precision และ Recall ซึ่งใช้วัด ประสิทธิภาพได้ดีที่สุดในกรณีที่ข้อมูลแต่ละคลาสไม่เท่ากัน

2.4 การประมวลผลภาษาธรรมชาติ (Natural Language Processing - NLP)

การประมวลผลภาษาธรรมชาติ หรือ NLP เป็นสาขาหนึ่งของปัญญาประดิษฐ์ (Artificial Intelligence) ที่มุ่งเน้นให้คอมพิวเตอร์สามารถเข้าใจ ตีความ และตอบสนองต่อภาษามนุษย์ได้ สำหรับระบบแชทบอท กระบวนการ NLP ถือเป็นหัวใจสำคัญในการเปลี่ยนข้อความดิบ (Raw Text) ให้เป็นข้อมูลที่มีความหมาย เพื่อนำไปสู่การประมวลผลคำตอบที่ถูกต้อง โดยมีกระบวนการย่อยดังนี้

2.4.1 Text Preprocessing การเตรียมข้อมูลข้อความ (Text Preprocessing) คือ กระบวนการเตรียมข้อมูลดิบให้พร้อมสำหรับการวิเคราะห์ โดยการลดความซับซ้อนและกำจัดข้อมูลขยะ (Noise) เพื่อให้ระบบสามารถประมวลผลได้อย่างมีประสิทธิภาพสูงสุด

2.4.1.1 Tokenization: การตัดคำ Tokenization คือกระบวนการแบ่งข้อความหรือประโยคยาวๆ ออกเป็นหน่วยคำย่อยที่เล็กที่สุดที่ยังมีความหมาย เรียกว่า "Token" สำหรับภาษาที่มีการเว้นวรรคคำชัดเจนอย่างภาษาอังกฤษ (Word Segmentation) จะใช้ช่องว่างเป็นตัวกำหนด แต่สำหรับ ภาษาไทย ซึ่งเป็นภาษาที่เขียนติดกัน (Script Continua) กระบวนการนี้จึงมีความซับซ้อนกว่า จำเป็นต้องใช้อัลกอริทึมที่อ้างอิงพจนานุกรม (Dictionary-based) หรือใช้ Machine Learning ในการวิเคราะห์ขอบเขตของคำ เพื่อให้แยกคำได้อย่างถูกต้อง เช่น แปลงประโยค "ตารางเรียนวันนี้" ให้กลายเป็น ["ตาราง", "เรียน", "วัน", "นี้"]

2.4.1.2 Normalization: การทำความสะอาดข้อความ Normalization คือ กระบวนการแปลงข้อความให้อยู่ในรูปแบบมาตรฐานเดียวกัน เพื่อลดความผันแปรของข้อมูล (Data Variance) ซึ่งประกอบด้วย

- การกำจัดอักขระพิเศษ (Special Character Removal): การลบสัญลักษณ์ที่ไม่สื่อความหมาย เช่น Emoji เครื่องหมายวรรคตอนที่ไม่จำเป็น หรือโค้ด HTML
- การจัดการคำย่อและคำสแลง: การแปลงภาษาพูดหรือคำย่อให้เป็นคำเต็มที่ระบบเข้าใจ เช่น แปลง "พน." เป็น "พฤษภาคม" หรือ "มหาลัย" เป็น "มหาวิทยาลัย"
- Stop Word Removal: การตัดคำฟุ่มเฟือยที่พบบ่อยแต่ไม่มีนัยสำคัญต่อความหมายหลัก เช่น "ครับ" "ค่ะ" "ที่" "ซึ่ง" "อัน" ออกไปเพื่อลดขนาดข้อมูลและเพิ่มประสิทธิภาพการประมวลผล

2.4.2 Matching Algorithms หลังจากผ่านกระบวนการเตรียมข้อมูล ระบบจะใช้อัลกอริทึมในการจับคู่ข้อความที่ได้รับ (Input) เข้ากับฐานข้อมูลความรู้ (Knowledge Base) หรือเจตนา (Intent) ที่กำหนดไว้ โดยมีเทคนิคสำคัญดังนี้

2.4.2.1 Keyword Matching: การจับคู่คำหลักแบบตรงตัว เป็นเทคนิคพื้นฐานที่สุด โดยระบบจะตรวจสอบว่าในประโยคที่ผู้ใช้พิมพ์เข้ามา มี "คำหลัก" (Keyword) ที่กำหนดไว้หรือไม่ หากพบคำดังกล่าว ระบบจะดึงคำตอบที่เชื่อมโยงไว้ออกมาทันที วิธีนี้มีความรวดเร็วและแม่นยำสูงในกรณีที่คำถามไม่ซับซ้อน แต่มีข้อจำกัดคือหากผู้ใช้ใช้รูปแบบประโยคอื่นที่ไม่มี Keyword ระบบจะไม่สามารถตอบได้

2.4.2.2 Synonym Expansion: การขยายความหมายด้วยคำพ้อง เพื่อแก้ปัญหาความแข็งทื่อของ Keyword Matching ระบบจะมีการใช้พจนานุกรมคำพ้องความหมาย (Thesaurus) เข้ามาช่วย โดยกำหนดกลุ่มคำที่มีความหมายเหมือนกันให้ชี้ไปยังเจตนา (Intent) เดียวกัน

- ตัวอย่าง: การกำหนดให้คำว่า "อาจารย์", "ครู", "ผู้สอน", "Lecturer" ทั้งหมดมีค่าเท่ากับ Intent: INSTRUCTOR_INFO ทำให้ไม่ว่าผู้ใช้จะพิมพ์คำไหนมา ระบบก็สามารถเข้าใจได้ว่าผู้ใช้ ต้องการสอบถามข้อมูลเกี่ยวกับอาจารย์

2.4.2.3 Fuzzy Matching: การจับคู่คำที่สะกดผิดหรือใกล้เคียง เป็นเทคนิคที่ใช้จัดการกับความผิดพลาดจากการพิมพ์ (Typo) ของผู้ใช้งาน โดยใช้อัลกอริทึมวัดระยะห่างระหว่างคำ (Edit Distance) เช่น Levenshtein Distance ซึ่งคำนวณจำนวนครั้งของการลบ แทรก หรือเปลี่ยนตัวอักษร เพื่อให้คำหนึ่งกลายเป็นอีกคำหนึ่ง

- การทำงาน: หากผู้ใช้พิมพ์ว่า "สวดี" (พิมพ์ผิด - ตก ส.เสือ) ระบบจะคำนวณความคล้ายคลึงและพบว่าใกล้เคียงกับคำว่า "สวัสดี" มากกว่าเกณฑ์ที่กำหนด (Threshold) จึงอนุมานว่าผู้ใช้ต้องการทักทาย

2.4.2.4 Regular Expression (Regex): การจับรูปแบบแพทเทิร์นข้อความเฉพาะ Regular Expression คือชุดรหัสรูปแบบอักขระที่ใช้กำหนดแพทเทิร์นการค้นหาข้อความที่มีโครงสร้างแน่นอน (Structured Data) ไม่ใช่การจับคู่คำต่อคำ แต่เป็นการจับคู่ตาม "รูปแบบ"

- การประยุกต์ใช้: ใช้ในการดึงข้อมูลสำคัญ (Entity Extraction) ออกจากประโยคสนทนา เช่น

- รหัสนักศึกษา: รูปแบบตัวเลข 10 หลักขึ้นไป

- เบอร์โทรศัพท์: รูปแบบ 0xx-xxx-xxxx

- อีเมล: รูปแบบ text + @ + domain.com

- ประโยชน์: วิธีนี้ช่วยให้บอทสามารถดึงข้อมูลที่ถูกต้องไปประมวลผลต่อได้ทันทีโดยไม่ต้องพึ่งพา AI ที่ซับซ้อน

2.5 ระบบจัดการฐานข้อมูล (Database Management System)

ระบบจัดการฐานข้อมูล (DBMS) ทำหน้าที่เป็นซอฟต์แวร์ตรงกลางที่ช่วยในการจัดเก็บ ค้นหา และจัดการข้อมูลสำหรับเซิร์ฟเวอร์ เพื่อให้สามารถดึงข้อมูลมาตอบโต้กับผู้ใช้งานได้อย่างถูกต้องและรวดเร็ว โดยมีรายละเอียดการออกแบบดังนี้

2.5.1 Relational Database: การใช้ฐานข้อมูลเชิงสัมพันธ์ (MySQL) ฐานข้อมูลเชิงสัมพันธ์ (Relational Database Management System - RDBMS) คือรูปแบบการจัดเก็บข้อมูลในลักษณะของ "ตาราง" (Table) ที่มีความสัมพันธ์เชื่อมโยงกันผ่านคีย์ (Key)

2.5.1.1 เหตุผลที่เลือกใช้ MySQL: ในการพัฒนาระบบนี้ เลือกใช้ MySQL เนื่องจากเป็นระบบจัดการฐานข้อมูลแบบ Open Source ที่มีความเสถียรสูง รองรับคำสั่ง SQL (Structured Query Language) ที่เป็นมาตรฐานสากล และมีประสิทธิภาพในการประมวลผลข้อมูลจำนวนมาก (Scalability) ซึ่งเหมาะสมกับการจัดเก็บข้อมูลของเซิร์ฟเวอร์ที่มีการเรียกใช้ข้อมูลตลอดเวลา (High Availability)

2.5.1.2 หลักการทำงาน: ข้อมูลจะถูกแยกเก็บเป็นตารางย่อยๆ ตามประเภทของข้อมูล และเชื่อมโยงกันด้วยความสัมพันธ์แบบต่างๆ เช่น One-to-One, One-to-Many เพื่อลดความซ้ำซ้อนและรักษาความถูกต้องของข้อมูล (Data Integrity)

2.5.2 Database Design การออกแบบฐานข้อมูล (Database Design) เป็นขั้นตอนสำคัญในการวางโครงสร้างการจัดเก็บข้อมูลเพื่อให้ระบบทำงานได้อย่างมีประสิทธิภาพสูงสุด

2.5.2.1 โครงสร้างตารางที่ใช้ (Tables) การออกแบบ Schema ของฐานข้อมูลประกอบด้วยตารางหลัก 4 ตาราง ที่จำเป็นต่อการทำงานของเซิร์ฟเวอร์ ดังนี้

1. Table: staff (ข้อมูลบุคลากร): เก็บข้อมูลอาจารย์และบุคลากรของคณะวิศวกรรมศาสตร์

- Columns: id (PK), name_th, name_en, position_th, position_en, department, email, phone, expertise, photo_url, is_active, created_at, updated_at

2. Table: faq (คำถาม-คำตอบ): ตารางหลักสำหรับเก็บคำถามที่พบบ่อยและคำตอบ (Knowledge Base)

- Columns: id (PK), question, answer, keywords, category, department, is_active

3. Table: news (ข่าวสาร): เก็บข่าวสารและกิจกรรมของคณะ มีการอัปเดตอัตโนมัติผ่าน Web Scraping

- Columns: id (PK), title, summary, category, link_url, published_date, is_active, created_at

4. Table: chat_logs (ประวัติการสนทนา): บันทึกการสนทนาเพื่อนำไปวิเคราะห์และปรับปรุงระบบ

- Columns: id (PK), session_id, user_message, bot_response, sources, confidence, response_time_ms, user_ip, user_agent, created_at

2.5.2.2 Normalization: การจัดระเบียบข้อมูลเพื่อลดความซ้ำซ้อน Normalization คือกระบวนการออกแบบตารางให้เป็นไปตามกฎมาตรฐาน (Normal Forms) เพื่อลดความซ้ำซ้อนของข้อมูล (Data Redundancy) และป้องกันความผิดปกติในการจัดการข้อมูล (Anomalies)

- การออกแบบในระบบนี้: ระบบใช้หลักการ Normalization ถึงระดับ 3NF (Third Normal Form) โดยแต่ละตารางมี Primary Key ที่ไม่ซ้ำกัน และคอลัมน์ทั้งหมดขึ้นอยู่กับการ Primary Key โดยตรง อย่างไรก็ตาม สำหรับข้อมูลสาขาวิชา (department) ระบบเลือกไม่แยกเป็นตารางแยก แต่เก็บเป็น VARCHAR ในตาราง staff และ faq โดยตรง เนื่องจากข้อมูลสาขามีจำนวนจำกัด (10 สาขา) และไม่มีการเปลี่ยนแปลงบ่อย ซึ่งเป็น Trade-off ระหว่างความเรียบง่ายในการ Query กับการลดความซ้ำซ้อน

2.5.2.3 Indexing: การทำดัชนีเพื่อเพิ่มประสิทธิภาพการค้นหา Indexing คือเทคนิคการสร้างโครงสร้างข้อมูลพิเศษ (เช่น B-Tree) เพื่อเพิ่มความเร็วในการค้นหาข้อมูล

- ความสำคัญต่อแชทบอท: แชทบอทต้องตอบสนองแบบ Real-time ดังนั้นการค้นหาข้อมูลโดยไม่มี Index (Full Table Scan) จะทำให้ระบบช้ามาก

- ประเภท Index ที่ใช้:

1. B-Tree Index: ใช้กับคอลัมน์ที่ค้นหาค่อยๆ เช่น department, is_active, session_id ช่วยให้การค้นหาเร็วขึ้นจาก $O(n)$ เป็น $O(\log n)$

2. FULLTEXT Index: ใช้กับตาราง faq สำหรับคอลัมน์ question, keywords, answer เพื่อรองรับการค้นหาข้อความแบบ Natural Language ด้วยคำสั่ง MATCH() AGAINST()

2.5.2.4 Transaction & ACID: คุณสมบัติการรับประกันความถูกต้องของข้อมูล ACID คือคุณสมบัติ 4 ประการที่รับประกันความถูกต้องและความสมบูรณ์ของข้อมูลเมื่อมีการทำธุรกรรม (Transaction)

- Atomicity (ความเป็นหน่วยเดียว): การทำงานต้องสำเร็จทั้งหมดหรือไม่ทำเลย (All or Nothing) เช่น เมื่อบันทึก chat log พร้อมอัปเดตสถิติ ต้องสำเร็จพร้อมกัน หากส่วนใดส่วนหนึ่งล้มเหลวระบบจะยกเลิก (Rollback) ทั้งหมด
- Consistency (ความถูกต้อง): ข้อมูลหลังการทำธุรกรรมต้องถูกต้องตามกฎหมายที่กำหนด เช่น ค่า confidence ต้องอยู่ระหว่าง 0.00-1.00 หรือ is_active ต้องเป็น 0 หรือ 1 เท่านั้น
- Isolation (ความแยกตัว): เมื่อมีผู้ใช้หลายคนเข้าใช้งานพร้อมกัน การทำธุรกรรมของแต่ละคนต้องไม่กระทบกัน ระบบใช้ระดับ Isolation แบบ READ COMMITTED เพื่อป้องกันการอ่านข้อมูลที่ยังไม่ได้ยืนยัน
- Durability (ความคงทน): เมื่อข้อมูลถูกบันทึกเสร็จสิ้นแล้ว จะถูกเก็บอย่างถาวร แม้ระบบจะไฟดับหรือเกิดข้อผิดพลาด MySQL มีระบบ Transaction Log และ Crash Recovery อัตโนมัติเพื่อรับประกันคุณสมบัตินี้

2.6 สถาปัตยกรรมเว็บแอปพลิเคชัน (Web Application Architecture)

สถาปัตยกรรมเว็บแอปพลิเคชัน คือ พิมพ์เขียวหรือโครงสร้างที่กำหนดรูปแบบการปฏิสัมพันธ์ระหว่างแอปพลิเคชัน (Application), มิดเดิลแวร์ (Middleware) และฐานข้อมูล (Database) เพื่อให้ระบบสามารถทำงานร่วมกันได้อย่างมีประสิทธิภาพ รองรับการขยายตัว และง่ายต่อการบำรุงรักษา

2.6.1 3-Tier Architecture สถาปัตยกรรมแบบ 3 ชั้น (Three-Tier Architecture) เป็นรูปแบบโครงสร้างพื้นฐานที่นิยมที่สุดในการพัฒนาเว็บแอปพลิเคชัน โดยแบ่งระบบออกเป็น 3 ส่วนอิสระที่ทำงานประสานกัน ได้แก่

2.6.1.1 Presentation Tier (ส่วนแสดงผล)

- ทำหน้าที่เป็นส่วนติดต่อกับผู้ใช้งาน (User Interface - UI) ผ่านเว็บเบราว์เซอร์หรือแอปพลิเคชัน
- รับผิดชอบการรับข้อมูลนำเข้า (Input) จากผู้ใช้ และแสดงผลลัพธ์ (Output)
- เทคโนโลยีที่ใช้มักเป็น HTML, CSS, และ JavaScript (หรือ Framework เช่น React, Vue.js)

2.6.1.2 Application Tier (ส่วนประมวลผล)

- เปรียบเสมือนสมองของระบบ หรือที่เรียกว่า "Logic Tier" ทำหน้าที่ประมวลผลตรรกะทางธุรกิจ (Business Logic)

- รับคำขอ (Request) จากส่วนแสดงผล มาวิเคราะห์ คำนวณ และตัดสินใจ ก่อนจะส่งคำสั่งไปยังฐานข้อมูล

- เป็นที่อยู่ของโค้ดโปรแกรมหลัก (Source Code) เช่น ภาษา Python, PHP, Node.js หรือ Java

2.6.1.3 Data Tier (ส่วนข้อมูล)

- ทำหน้าที่จัดเก็บและบริหารจัดการข้อมูล (Database Management)
- รับคำสั่งจาก Application Tier เพื่อทำการเพิ่ม ลบ แก้ไข หรือค้นหาข้อมูล (CRUD Operations) และส่งข้อมูลกลับไป

- ประกอบด้วยระบบฐานข้อมูล เช่น MySQL, PostgreSQL หรือ MongoDB

2.6.2 MVC Pattern (Model-View-Controller) MVC เป็นรูปแบบสถาปัตยกรรมซอฟต์แวร์ (Software Architectural Pattern) ที่ใช้ในการจัดระเบียบโค้ดภายใน Application Tier เพื่อแยกหน้าที่การทำงานให้ชัดเจน ลดความซับซ้อน และช่วยให้นักพัฒนาหลายคนทำงานร่วมกันได้ง่ายขึ้น ประกอบด้วย 3 ส่วนคือ

2.6.2.1 Model (ส่วนข้อมูลและตรรกะ)

- รับผิดชอบจัดการข้อมูลและกฎทางธุรกิจ (Business Logic) เชื่อมต่อกับฐานข้อมูลโดยตรง

- เมื่อข้อมูลมีการเปลี่ยนแปลง Model จะแจ้งเตือนไปยัง View เพื่ออัปเดตการแสดงผล

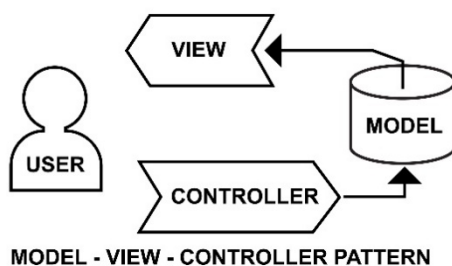
2.6.2.2 View (ส่วนแสดงผล)

- รับผิดชอบการแสดงผลข้อมูลที่ได้รับจาก Model ให้ผู้ใช้งานเห็น (User Interface)

- ไม่มีหน้าที่ในการคำนวณหรือประมวลผลตรรกะที่ซับซ้อน

2.6.2.3 Controller (ส่วนควบคุม)

- ทำหน้าที่เป็นตัวกลาง (Intermediary) ระหว่าง Model และ View
- รับ Input จากผู้ใช้ (ผ่าน View) แล้วประมวลผลเพื่อสั่งให้ Model อัปเดตข้อมูล หรือสั่งให้ View เปลี่ยนหน้าจอการแสดงผล



รูปที่ 2.1 Model-View-Controller

2.6.3 Client-Server Model Client-Server Model คือรูปแบบสถาปัตยกรรมเครือข่ายที่อธิบายกระบวนการสื่อสารและแบ่งหน้าที่ระหว่างคอมพิวเตอร์ 2 ฝั่ง

2.6.3.1 Client (ฝั่งผู้ใช้)

- คือเครื่องคอมพิวเตอร์หรืออุปกรณ์ของผู้ใช้งาน (เช่น สมาร์ทโฟนที่เปิดหน้าแชทบอท)

- ทำหน้าที่เป็น "ผู้ร้องขอ" (Requester) โดยส่งคำขอข้อมูล (Request) ผ่านเครือข่ายอินเทอร์เน็ตไปยังเครื่องแม่ข่าย

2.6.3.2 Server (ฝั่งเซิร์ฟเวอร์)

- คือเครื่องคอมพิวเตอร์ประสิทธิภาพสูงที่เปิดใช้งานตลอดเวลา

- ทำหน้าที่เป็น "ผู้ให้บริการ" (Provider) คอยรับคำขอ (Listening) ประมวลผลตามคำสั่ง และส่งข้อมูลตอบกลับ (Response) ไปยัง Client

2.6.3.3 กระบวนการทำงาน (Request-Response Cycle)

- การสื่อสารระหว่าง Client และ Server มักทำผ่านโปรโตคอล HTTP/HTTPS โดย Client ส่ง HTTP Request และ Server ตอบกลับด้วย HTTP Response (พร้อม Status Code เช่น 200 OK, 404 Not Found)

2.7 การเชื่อมต่อผ่าน API (RESTful Services)

การเชื่อมต่อระหว่างส่วนหน้าบ้าน (Frontend) และหลังบ้าน (Backend) ในระบบเว็บแอปพลิเคชัน ปัจจุบัน นิยมใช้รูปแบบ API (Application Programming Interface) เพื่อเป็นสะพานเชื่อมโยงข้อมูล โดยสถาปัตยกรรมที่เป็นมาตรฐานหลักคือ RESTful Services ซึ่งมีความยืดหยุ่นสูงและเป็นสากล

2.7.1 REST Principles: หลักการออกแบบ API แบบ Stateless REST (Representational State Transfer) เป็นรูปแบบสถาปัตยกรรมซอฟต์แวร์สำหรับการส่งผ่านข้อมูลบนเครือข่ายอินเทอร์เน็ต โดยมีหัวใจสำคัญคือหลักการ Statelessness (ความไร้สถานะ)

2.7.1.1 ความหมายของ Stateless: ในการสื่อสารแต่ละครั้ง (Request) จาก Client ไปยัง Server จะต้องบรรจุข้อมูลที่จำเป็นทั้งหมดสำหรับการประมวลผลไปด้วยเสมอ Server จะไม่ทำการบันทึกสถานะการเชื่อมต่อ (Session) ของผู้ใช้ไว้ที่ฝั่ง Server เอง

2.7.1.2 ประโยชน์: ทำให้ Server ไม่ต้องแบกรับภาระหน่วยความจำในการจดจำผู้ใช้งานทุกคน ช่วยให้ระบบสามารถรองรับผู้ใช้งานจำนวนมากได้ (Scalability) และง่ายต่อการขยายระบบ

2.7.2 HTTP Methods: การใช้คำสั่ง GET, POST, PUT, DELETE ในการสั่งการให้ API ทำงาน RESTful API จะใช้ "คำกริยา" (Verbs) ตามมาตรฐาน HTTP Protocol เพื่อระบุการกระทำที่ต้องการ (Action) ต่อทรัพยากรข้อมูล ดังนี้

2.7.2.1 GET (ดึงข้อมูล): ใช้สำหรับร้องขอข้อมูลจาก Server โดยไม่มีการเปลี่ยนแปลงแก้ไขข้อมูลใดๆ เช่น การดึงประวัติการสนทนา หรือการดึงข้อมูลสินค้า

2.7.2.2 POST (ส่งข้อมูล/สร้างใหม่): ใช้สำหรับส่งข้อมูลใหม่ไปบันทึกที่ Server เช่น การส่งข้อความแชทที่ผู้ใช้พิมพ์ (Input Message) ไปประมวลผล

2.7.2.3 PUT (แก้ไข/แทนที่): ใช้สำหรับการอัปเดตข้อมูลที่มีอยู่แล้วให้เป็นค่าใหม่ เช่น การแก้ไขข้อมูลส่วนตัวสมาชิก หรือการเปลี่ยนสถานะงาน

2.7.2.4 DELETE (ลบ): ใช้สำหรับส่งลบข้อมูลออกจากฐานข้อมูล เช่น การลบบัญชีผู้ใช้ หรือลบรายการสินค้า

2.7.3 JSON Format: รูปแบบข้อมูลมาตรฐานที่ใช้รับส่ง JSON (JavaScript Object Notation) คือรูปแบบมาตรฐานในการจัดเก็บและแลกเปลี่ยนข้อมูลคอมพิวเตอร์ ซึ่งเข้ามาแทนที่ XML เนื่องจากมีโครงสร้างที่กะทัดรัดและประมวลผลได้เร็วกว่า

2.7.3.1 โครงสร้าง: จัดเก็บข้อมูลในรูปแบบคู่ของ Key: Value (Key-Value Pair) ภายใต้วงเล็บปีกกา { } ซึ่งมนุษย์สามารถอ่านเข้าใจได้ง่าย (Human-readable)

2.7.4 Security: การรักษาความปลอดภัย เนื่องจาก API เป็นช่องทางเข้าออกของข้อมูลสำคัญ ระบบจึงต้องมีมาตรการรักษาความปลอดภัยที่เข้มงวด

2.7.4.1 HTTPS (Hypertext Transfer Protocol Secure): การเข้ารหัสข้อมูลในระหว่างการส่งผ่านเครือข่าย (Data in Transit) ด้วยโปรโตคอล SSL/TLS เพื่อป้องกันการดักจับหรือแอบอ่านข้อมูล (Eavesdropping) โดยผู้ไม่หวังดี

2.7.4.2 API Key / Authentication: การระบุตัวตนผู้ใช้งาน API โดยผู้ใช้หรือแอปพลิเคชันต้องแนบรหัสผ่านพิเศษ (API Key หรือ Token) มากับทุก Request เพื่อให้ Server ตรวจสอบสิทธิ์ว่าได้รับอนุญาตให้เข้าถึงข้อมูลหรือไม่ และยังช่วยป้องกันการโจมตีแบบ Spam Request ได้อีกด้วย

2.8 การออกแบบส่วนติดต่อผู้ใช้ (UI/UX Design)

การออกแบบส่วนติดต่อผู้ใช้ (User Interface - UI) และประสบการณ์ผู้ใช้ (User Experience - UX) เป็นปัจจัยสำคัญที่จะกำหนดความสำเร็จของระบบแชทบอท การออกแบบที่ดีต้องคำนึงถึงความสะดวกในการใช้งาน การเข้าถึงข้อมูลที่ย่าง และความรู้สึกพึงพอใจของผู้ใช้งาน โดยมีหลักการสำคัญดังนี้

2.8.1 Usability Principles: ความสม่ำเสมอ (Consistency) และความเรียบง่าย (Simplicity) หลักการ "Usability" หรือความสามารถในการใช้งาน มุ่งเน้นการลดภาระทางความคิดของผู้ใช้ (Cognitive Load) เพื่อให้ผู้ใช้สามารถบรรลุเป้าหมายได้อย่างรวดเร็ว

2.8.1.1 ความสม่ำเสมอ (Consistency): การออกแบบต้องเป็นไปในทิศทางเดียวกันทั้งระบบ

- Visual Consistency: การใช้โทนสี รูปแบบตัวอักษร และไอคอน ให้เหมือนกันทุกหน้า เพื่อสร้างความจดจำ (Brand Identity)

- Functional Consistency: ตำแหน่งของปุ่มสั่งการหรือเมนูต้องวางอยู่ในตำแหน่งเดิม ไม่ย้ายไปมา เพื่อให้ผู้ใช้เกิดความคุ้นเคยและเรียนรู้การใช้งานได้เร็ว (Learnability)

2.8.1.2 ความเรียบง่าย (Simplicity): การตัดสิ่งที่ไม่จำเป็นออกตามหลักการ "Less is More"

- ลดจำนวนขั้นตอนการคลิก (Clicks) ให้เหลือเท่าที่จำเป็น

- ใช้พื้นที่ว่าง (White Space) เพื่อให้อ่านง่ายและสบายตา
- หลีกเลี่ยงการแสดงข้อมูลที่หนาแน่นเกินไปในหน้าจอสนทนา เพื่อให้ผู้ใช้โฟกัสกับเนื้อหาหลักได้ทันที

2.8.2 Responsive Design: การแสดงผลที่รองรับทุกหน้าจอ (Mobile-first) โดยเฉพาะบนมือถือ ในยุคปัจจุบัน ผู้ใช้งานส่วนใหญ่นิยมเข้าถึงอินเทอร์เน็ตผ่านสมาร์ทโฟน การออกแบบจึงต้องยึดหลัก Mobile-First Strategy คือการให้ความสำคัญกับการออกแบบบนหน้าจอมือถือเป็นลำดับแรก ก่อนจะขยายไปยังหน้าจอขนาดใหญ่

2.8.2.1 Liquid Layout & Flexible Grids: การใช้โครงสร้างหน้าเว็บที่ยืดหยุ่น สามารถปรับขนาดและความกว้างขององค์ประกอบต่างๆ (เช่น กล่องข้อความ รูปภาพ ปุ่มกด) ให้พอดีกับความกว้างของหน้าจออุปกรณ์ (Viewport) โดยอัตโนมัติ

2.8.2.2 Touch-Friendly Interface: การออกแบบจุดสัมผัส (Touch Targets) เช่น ปุ่มเมนู หรือตัวเลือก Quick Reply ให้มีขนาดใหญ่พอที่นิ้วมือจะสัมผัสได้แม่นยำ (โดยทั่วไปควรมีขนาดไม่ต่ำกว่า 44x44 pixels) และมีการเว้นระยะห่างเพื่อป้องกันการกดผิด

2.8.2.3 Readable Content: ขนาดตัวอักษรต้องอ่านง่ายโดยไม่ต้องซูมหน้าจอ และจัดลำดับความสำคัญของเนื้อหาให้เหมาะสมกับพื้นที่จำกัด

2.8.3 Conversational Design: หลักการออกแบบบทสนทนาให้ดูเป็นธรรมชาติ ไม่เหมือนหุ่นยนต์จนเกินไป Conversational Design คือศาสตร์ของการออกแบบ "บทพูด" ให้กับระบบ เพื่อให้ผู้ใช้งานรู้สึกเหมือนกำลังสนทนากับมนุษย์ (Human-like Interaction) มากกว่าการป้อนคำสั่งคอมพิวเตอร์

2.8.3.1 Persona Development: การกำหนดบุคลิกภาพของแชทบอทให้ชัดเจน เช่น การใช้สรรพนาม (ผม/ดิฉัน/น้อง) น้ำเสียง (ทางการ/เป็นกันเอง) เพื่อสร้างอารมณ์ร่วมและความน่าเชื่อถือ

2.8.3.2 Cooperative Principle: การออกแบบบทสนทนาตามหลักความร่วมมือ โดยบอทต้องให้ข้อมูลที่ "กระชับ ตรงประเด็น และชัดเจน" ไม่ตอบยาวเหยียดจนผู้ใช้จับใจความไม่ได้

2.8.3.3 Turn-taking & Feedback: ระบบต้องมีการโต้ตอบที่แสดงสถานะชัดเจน เช่น การแสดงสัญลักษณ์ "กำลังพิมพ์..." (Typing Indicator) เพื่อให้ผู้ใช้รู้ว่าระบบกำลังประมวลผลอยู่ และไม่รู้สึกรอถูกทิ้งให้รอเก้อ

2.8.3.4 Error Handling: เมื่อบอทไม่เข้าใจคำสั่ง ต้องมีการออกแบบประโยคตอบรับที่สุภาพและแนะนำทางเลือก (Fallback Message) เช่น "ขอโทษครับ ผมไม่แน่ใจว่าเข้าใจถูกไหม คุณหมายถึง...ใช่หรือไม่?" แทนที่จะตอบแค่ว่า "Error" หรือไม่ตอบเลย

2.9 เครื่องมือและซอฟต์แวร์ที่ใช้ในการพัฒนา

ในการพัฒนาระบบแชทบอทอัจฉริยะสำหรับคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร ผู้พัฒนาได้เลือกใช้เครื่องมือและซอฟต์แวร์ที่หลากหลาย ครอบคลุมทุกขั้นตอนของกระบวนการพัฒนา ตั้งแต่การเขียนโค้ด การจัดการฐานข้อมูล การพัฒนาโมเดล Machine Learning การทดสอบ API ไปจนถึงการจัดทำเอกสาร ดังต่อไปนี้

หมวดหมู่	เครื่องมือ/ซอฟต์แวร์	หน้าที่หลัก
ภาษาโปรแกรม	PHP 8.0+ Python 3.8+ JavaScript (ES6+) HTML5 & CSS3	Backend API, Business Logic AI/ML Engine Frontend Interactive Logic Structure & Styling
Python Libraries	pythainlp 4.0.2 scikit-learn 1.3.0 pandas 2.0.3 Flask 2.3.2 Flask-CORS 4.0.0 numpy 1.24.3 joblib 1.3.0	Thai NLP Processing ML Model Training Data Manipulation REST API Server Cross-Origin Handling Numerical Computing Model Serialization
ฐานข้อมูล	MariaDB 10.4	RDBMS (via XAMPP)
เว็บเซิร์ฟเวอร์	XAMPP	Apache + MariaDB + PHP
IDE & Editors	Visual Studio Code phpMyAdmin	Code Development Database Management
API Testing	Postman	REST API Testing
Screenshot Tools	Snipping Tool	Screen Capture

ตารางที่ 2.1 สรุปเครื่องมือและซอฟต์แวร์ที่ใช้ในการพัฒนา

2.9.1 ภาษาโปรแกรม (Programming Languages)

ระบบนี้พัฒนาโดยใช้ภาษาโปรแกรมหลัก 4 ภาษา ได้แก่ PHP, Python, JavaScript และ HTML/CSS โดยแต่ละภาษามีบทบาทที่แตกต่างกันตามสถาปัตยกรรมแบบ 3-Tier Architecture

2.9.1.1 PHP 8.0+: ใช้เป็นภาษาหลักในการพัฒนาฝั่ง Backend สำหรับการจัดการคำขอจากผู้ใช้ (Request Handling) การเชื่อมต่อฐานข้อมูล การประมวลผลตรรกะทางธุรกิจ และการสร้าง RESTful API เลือกใช้ PHP เนื่องจากมีความเข้ากันได้ดีกับ Apache Server และมีฟังก์ชันสำเร็จรูปสำหรับการทำงานกับ MySQL/MariaDB อีกทั้งยังมี Community ขนาดใหญ่ที่ให้การสนับสนุน ในโครงการนี้ PHP รับผิดชอบไฟล์หลัก 4 ไฟล์ ได้แก่ chatbot.php, ChatbotConfig.php, QueryAnalyzer.php และ broad_topic_handler.php รวมทั้งสิ้นประมาณ 3,958 บรรทัด



รูปที่ 2.2 PHP 8.0+

2.9.1.2 Python 3.8+: ใช้ในการพัฒนาส่วนปัญญาประดิษฐ์ (AI Engine) เนื่องจากมี Library ที่หลากหลายสำหรับ Machine Learning และ Natural Language Processing ในโครงการนี้ Python ทำหน้าที่เป็น AI API Server ผ่าน Flask Framework รับคำถามจาก PHP Backend แล้วทำการจำแนกประเภท (Intent Classification) ด้วย TF-IDF + Logistic Regression โดยใช้ Library ดังต่อไปนี้



รูปที่ 2.3 Python 3.8+

1. pythainlp 4.0.2: Library สำหรับการประมวลผลภาษาไทยโดยเฉพาะ ใช้ในการตัดคำ (Tokenization) ด้วย Engine "newmm" และกำจัดคำหยุด (Stop Words Removal) เนื่องจากภาษาไทยไม่มีการเว้นวรรคระหว่างคำ จึงจำเป็นต้องใช้พจนานุกรมและอัลกอริทึมเฉพาะทางในการแบ่งคำ



รูปที่ 2.4 pythainlp 4.0.2

2. scikit-learn 1.3.0: Library มาตรฐานสำหรับ Machine Learning ใน Python ใช้สำหรับสร้างโมเดลจำแนกประเภทคำถาม ประกอบด้วย TF-IDF Vectorizer สำหรับแปลงข้อความเป็นเวกเตอร์ตัวเลข และ Logistic Regression ($C=10$) สำหรับจำแนก Intent ทั้ง 15 ประเภท จากข้อมูลฝึก 3,615 ตัวอย่าง ได้ความแม่นยำ 96.4%



รูปที่ 2.5 scikit-learn 1.3.0

3. pandas 2.0.3: Library สำหรับจัดการและวิเคราะห์ข้อมูล (Data Manipulation) ใช้ในการอ่านไฟล์ training_data.csv จัดเตรียมข้อมูลสำหรับฝึกโมเดล และวิเคราะห์การกระจายของ Intent



รูปที่ 2.6 pandas 2.0.3

4. Flask 2.3.2: Micro Web Framework สำหรับ Python ใช้สร้าง REST API Server ที่ Port 5000 ให้บริการ Endpoint /predict สำหรับรับคำถามจาก PHP Backend แล้วส่งกลับผลการจำแนก Intent พร้อมค่าความมั่นใจ (Confidence Score)



รูปที่ 2.7 Flask 2.3.2

5. Flask-CORS 4.0.0: Extension ของ Flask สำหรับจัดการ Cross-Origin Resource Sharing (CORS) เพื่อให้ PHP Backend สามารถเรียก API ของ Python Flask Server ที่อยู่คนละ Port ได้อย่างปลอดภัย

6. numpy 1.24.3: Library พื้นฐานสำหรับการคำนวณเชิงตัวเลข (Numerical Computing) ใช้เป็น Dependency ของ scikit-learn ในการประมวลผลเมทริกซ์และเวกเตอร์



รูปที่ 2.8 numpy 1.24.3

7. joblib 1.3.0: Library สำหรับบันทึก (Serialize) และโหลด (Deserialize) โมเดล Machine Learning ที่ผ่านการฝึกแล้วเป็นไฟล์ .pkl เพื่อนำกลับมาใช้งานได้ทันทีโดยไม่ต้องฝึกใหม่ทุกครั้ง

2.9.1.3 JavaScript (ES6+): ใช้ในการพัฒนาฝั่ง Frontend สำหรับการสร้างปฏิสัมพันธ์กับผู้ใช้งาน (User Interaction) การส่ง AJAX Request ไปยัง Backend และการแสดงผลแบบ Dynamic โดยไม่ต้อง Reload หน้า รวมถึงรองรับฟีเจอร์ต่าง ๆ เช่น Dark Mode, Typing Indicator, Quick Action Cards และ Feedback System



รูปที่ 2.9 JavaScript (ES6+)

2.9.1.4 HTML5 & CSS3 ใช้ในการสร้างโครงสร้างหน้าเว็บ (Markup) และการออกแบบรูปลักษณ์ส่วนติดต่อผู้ใช้ (Styling) รองรับการออกแบบแบบ Responsive ด้วย CSS Media Queries, Flexbox และ CSS Custom Properties (CSS Variables) เพื่อให้แสดงผลได้ดีทั้งบนคอมพิวเตอร์และมือถือ รวมถึงรองรับ Dark Mode ผ่าน data-theme attribute



รูปที่ 2.10 HTML5 & CSS3

2.9.2 ระบบจัดการฐานข้อมูล (Database Management System)

2.9.2.1 MariaDB 10.4 (ผ่าน XAMPP): ใช้เป็น Relational Database Management System (RDBMS) หลักของระบบ สำหรับจัดเก็บข้อมูลทั้งหมด ได้แก่ ข้อมูลบุคลากร (staff) 118 คน, คำถาม-คำตอบ (faq) 593 รายการ, ข่าวสาร (news) 30 รายการ, ประวัติการสนทนา (chat_logs) และ Admin Sessions โดย MariaDB เป็น Fork ของ MySQL ที่มีความเข้ากัน

ได้รองรับ SQL Standard, ACID Properties, BTREE/FULLTEXT Index และ Prepared Statements ผ่าน PDO (PHP Data Objects) เพื่อป้องกัน SQL Injection ใช้ Character Set utf8mb4 สำหรับรองรับภาษาไทยและ Emoji



รูปที่ 2.11 MariaDB 10.4

2.9.3 เว็บเซิร์ฟเวอร์ (Web Server)

XAMPP: เป็น Software Package ที่รวม Apache HTTP Server, MariaDB Database และ PHP Interpreter เข้าด้วยกัน ทำให้สามารถติดตั้งและจัดการ Local Development Environment ได้อย่างสะดวก โดย Apache ทำหน้าที่เป็น Web Server รับ HTTP Request จากผู้ใช้งาน และส่งผลลัพธ์กลับผ่านโปรโตคอล HTTP/HTTPS เหมาะสำหรับการพัฒนาและทดสอบระบบก่อนนำขึ้นเซิร์ฟเวอร์จริง



รูปที่ 2.12 เว็บเซิร์ฟเวอร์ (Web Server) XAMPP

2.9.4 โปรแกรมพัฒนาและแก้ไขโค้ด (IDE & Code Editors)

2.9.4.1 Visual Studio Code: เป็น Source Code Editor ที่ใช้ในการเขียนและแก้ไขโค้ดทุกภาษาที่ใช้ในโครงการ (PHP, Python, JavaScript, HTML, CSS, SQL) รองรับ Extensions มากมายที่ช่วยเพิ่มประสิทธิภาพในการพัฒนา ได้แก่



รูปที่ 2.13 Visual Studio Code

- PHP Intelephense: สำหรับ Auto-completion และตรวจสอบไวยากรณ์ PHP



รูปที่ 2.14 PHP Intelephense

- Python Extension: สำหรับ Debugging, IntelliSense รัน Python Script



รูปที่ 2.15 Python Extension

- Prettier: สำหรับจัดรูปแบบโค้ดให้สวยงามและเป็นมาตรฐาน



รูปที่ 2.16 Prettier

- Live Server: สำหรับทดสอบหน้าเว็บแบบ Real-time พร้อม Hot Reload



รูปที่ 2.17 Live Server

2.9.4.2 phpMyAdmin: เป็น Web-based Tool สำหรับจัดการฐานข้อมูล MariaDB/MySQL ผ่านหน้าเว็บ ช่วยให้สามารถสร้างตาราง แก้ไขข้อมูล รัน SQL Query และ Import/Export ข้อมูลได้อย่างสะดวก โดยไม่ต้องใช้คำสั่ง Command Line มาพร้อมกับ XAMPP



รูปที่ 2.18 phpMyAdmin

2.9.5 เครื่องมือทดสอบ API (API Testing Tools)

2.9.5.1 Postman: ใช้สำหรับทดสอบการทำงานของ RESTful API ทั้ง PHP Backend (chatbot.php, admin_api.php) และ Python Flask API (/predict) โดยสามารถส่ง HTTP Request ประเภทต่าง ๆ (GET, POST, PUT, DELETE) พร้อมกำหนด Headers, Parameters และ Body ได้อย่างยืดหยุ่น ช่วยให้นักพัฒนาสามารถตรวจสอบความถูกต้องของ Response และ Debug ปัญหาได้รวดเร็ว



รูปที่ 2.19 Postman

2.9.6 เครื่องมือสร้างไดอะแกรมและเอกสาร (Diagram & Documentation Tools)

2.9.6.1 Draw.io (diagrams.net): เป็น Web-based Tool สำหรับวาดไดอะแกรมต่างๆ เช่น ER Diagram, Flowchart, System Architecture Diagram, Use Case Diagram ใช้งานฟรีและสามารถส่งออกเป็นไฟล์ภาพ (PNG, JPG, SVG) หรือ PDF ได้



รูปที่ 2.20 Draw.io

บทที่ 3

ขั้นตอนและวิธีการดำเนินงาน

3.1 ภาพรวมกระบวนการพัฒนา

การพัฒนาระบบ Chatbot อัจฉริยะสำหรับคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร ได้ดำเนินการตามกระบวนการพัฒนาซอฟต์แวร์แบบ Incremental Development โดยแบ่งการพัฒนาออกเป็น 5 ขั้นตอนหลัก ดังนี้:

ขั้นตอนที่ 1: วิเคราะห์ความต้องการและออกแบบระบบ

ขั้นตอนที่ 2: จัดเตรียมฐานข้อมูล

ขั้นตอนที่ 3: พัฒนาระบบ Backend และ Frontend

ขั้นตอนที่ 4: พัฒนาระบบ AI/ML สำหรับการจำแนกความตั้งใจ

ขั้นตอนที่ 5: ทดสอบและปรับปรุงระบบ

แต่ละขั้นตอนได้รับการวางแผนและดำเนินการอย่างเป็นระบบ โดยมีการทบทวนและปรับปรุงอย่างต่อเนื่องเพื่อให้ได้ระบบที่มีประสิทธิภาพสูงสุด

3.2 ขั้นตอนที่ 1: วิเคราะห์ความต้องการและออกแบบระบบ

3.2.1 การวิเคราะห์ความต้องการ

3.2.1.1 การรวบรวมความต้องการ (Requirements Gathering) ทีมผู้พัฒนาได้ดำเนินการรวบรวมความต้องการจากกลุ่มผู้ใช้งาน 3 กลุ่มหลัก:

1. นักศึกษาปัจจุบันและผู้สนใจสมัครเรียน

- ต้องการสอบถามข้อมูลการรับสมัคร (TCAS, รอบต่างๆ, คุณสมบัติ, เอกสาร)

- ต้องการทราบข้อมูลค่าเทอม การผ่อนชำระ

- ต้องการสอบถามข้อมูลทุนการศึกษา (กยศ./กรอ.)

- ต้องการข้อมูลสาขาวิชา หลักสูตร และความแตกต่าง

- ต้องการค้นหาอาจารย์ที่ปรึกษา อาจารย์ผู้สอน

- ต้องการบริการตอบคำถามตลอด 24 ชั่วโมง

2. เจ้าหน้าที่และอาจารย์

- ต้องการลดภาระงานตอบคำถามซ้ำๆ

- ต้องการระบบจัดการ FAQ ที่ใช้งานง่าย
- ต้องการดูประวัติการสนทนาเพื่อวิเคราะห์พฤติกรรม
- ต้องการระบบที่ปลอดภัยและควบคุมได้

3. ผู้ปกครองและบุคคลทั่วไป

- ต้องการข้อมูลเบื้องต้นเกี่ยวกับคณะและสาขาวิชา
- ต้องการช่องทางติดต่อคณะ
- ต้องการข้อมูลที่เข้าใจง่าย ไม่ซับซ้อน

3.2.1.2 การกำหนดขอบเขตระบบ (System Scope)

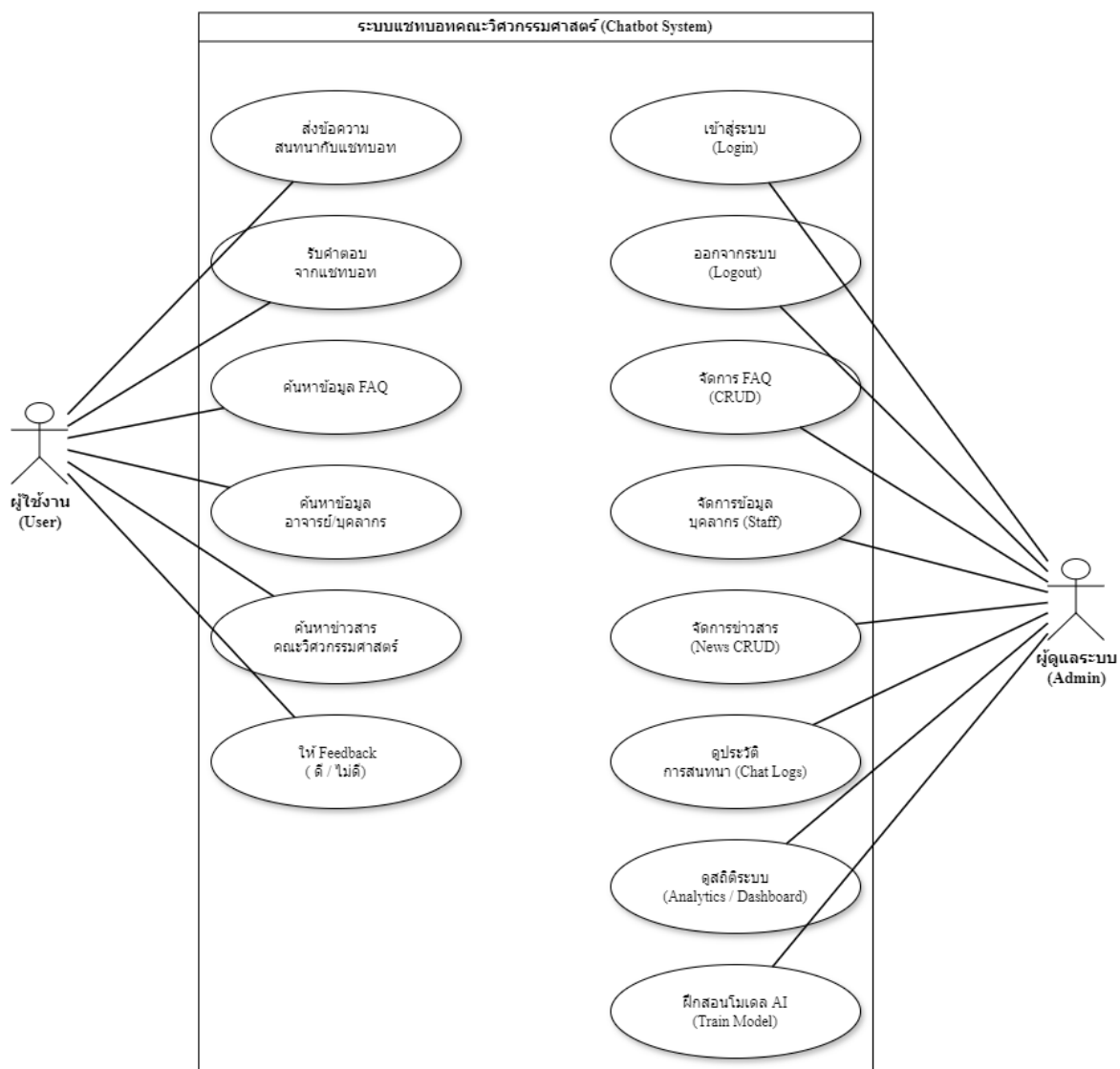
1. ขอบเขตที่รวมในระบบ

- ระบบตอบคำถามอัตโนมัติด้วย AI/ML
- ฐานข้อมูลบุคลากร 118 คน จาก 10 สาขาวิชา
- ฐานข้อมูล FAQ ครอบคลุมหลายหมวดหมู่ (มีทั้งหมด 645 รายการ)
- ระบบจัดการข้อมูล (Admin Dashboard)
- การบันทึกประวัติการสนทนา
- ระบบความปลอดภัย (CORS, Rate Limiting, Input Validation)
- Interface แบบ Responsive รองรับทุกอุปกรณ์

2. ขอบเขตที่ไม่รวมในระบบ

- การบูรณาการกับระบบทะเบียนนักศึกษา
- การประมวลผลเสียง (Voice Recognition)
- ระบบสมัครเรียนออนไลน์
- การวิเคราะห์ข้อมูลเชิงลึก (Advanced Analytics)
- รองรับภาษาต่างประเทศ

3.2.1.3 Use Case Diagram



รูปที่ 3.1 Use Case Diagram ระบบแชทบอทคณะวิศวกรรมศาสตร์

จากรูปที่ 3.1 แสดง Use Case Diagram ของระบบแชทบอทคณะวิศวกรรมศาสตร์ (Chatbot System) โดยแบ่งผู้ใช้งาน (Actor) ออกเป็น 2 กลุ่ม ดังนี้

ผู้ใช้งาน (User) สามารถใช้งานระบบได้ 6 Use Cases:

1. ส่งข้อความสนทนากับแชทบอท - พิมพ์คำถามภาษาไทยเพื่อสอบถามข้อมูล
2. รับคำตอบจากแชทบอท - รับคำตอบที่ระบบค้นหาแสดงผล พร้อมแหล่ง

อ้างอิง

3. ค้นหาข้อมูล FAQ - ค้นหาคำถามที่พบบ่อยเกี่ยวกับการรับสมัคร ค่าเทอม
ทุนการศึกษา หลักสูตร เป็นต้น

4. ค้นหาข้อมูลอาจารย์/บุคลากร - ค้นหาข้อมูลอาจารย์และเจ้าหน้าที่ 118 คน จาก
10 สาขาวิชา

5. ค้นหาข่าวสารคณะวิศวกรรมศาสตร์ - ค้นหาข่าวประชาสัมพันธ์และกิจกรรมของ
คณะ

6. ให้ Feedback (ดี/ไม่ดี) - ประเมินความพึงพอใจต่อคำตอบด้วยการกด Like
หรือ Dislike

ผู้ดูแลระบบ (Admin) สามารถใช้งานระบบได้ 8 Use Cases:

1. เข้าสู่ระบบ (Login) - ยืนยันตัวตนด้วย Username และ Password ผ่านหน้า
Admin Dashboard

2. ออกจากระบบ (Logout) - ออกจากระบบและยกเลิก Session Token

3. จัดการ FAQ (CRUD) - เพิ่ม แก้ไข ลบ และดูรายการคำถาม-คำตอบทั้ง 645
รายการ

4. จัดการข้อมูลบุคลากร (Staff) - เพิ่ม แก้ไข และดูข้อมูลอาจารย์และเจ้าหน้าที่

5. จัดการข่าวสาร (News CRUD) - เพิ่ม แก้ไข และลบข่าวประชาสัมพันธ์ของคณะ

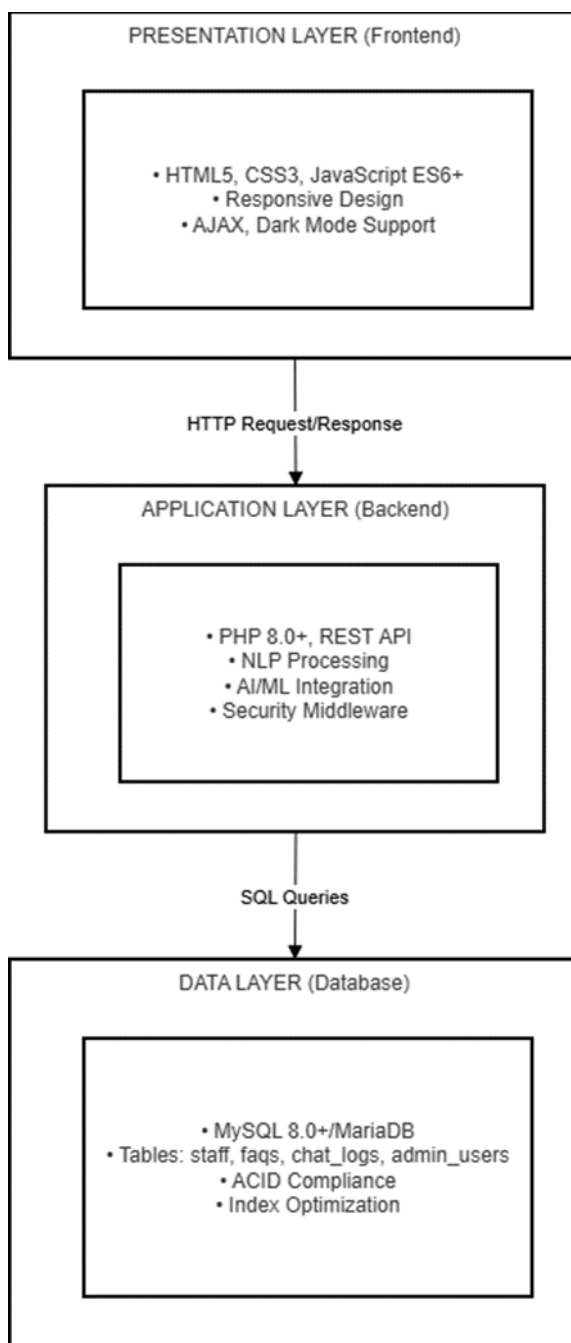
6. ดูประวัติการสนทนา (Chat Logs) - ตรวจสอบประวัติการสนทนาทั้งหมด
วิเคราะห์คำถามที่ Confidence ต่ำ

7. ดูสถิติระบบ (Analytics/Dashboard) - ดูรายงานภาพรวม คำถามยอดนิยม
และประสิทธิภาพระบบ

8. ฝึกสอนโมเดล AI (Train Model) - ส่งเทรนโมเดล AI ใหม่หลังจากเพิ่ม/แก้ไข
FAQ ผ่าน run_train_model.bat

3.2.2 การออกแบบสถาปัตยกรรมระบบ

3.2.2.1 สถาปัตยกรรมแบบ 3-Tier Architecture ระบบได้รับการออกแบบด้วยสถาปัตยกรรมแบบ 3 ชั้น (3-Tier Architecture) เพื่อแยกความรับผิดชอบของแต่ละส่วนให้ชัดเจน ประกอบด้วย:



รูปที่ 3.2 แผนภาพสถาปัตยกรรม 3 ชั้น (3-Tier Architecture)

จากรูปที่ 3.2 แสดงสถาปัตยกรรมแบบ 3 ชั้น โดยแต่ละชั้นมีหน้าที่ดังนี้:

ชั้นที่ 1: Presentation Layer (Frontend) เป็นชั้นที่ผู้ใช้งานมีปฏิสัมพันธ์โดยตรง ทำหน้าที่แสดงผลหน้าจอแชทบอทและรับข้อมูลจากผู้ใช้งาน พัฒนาด้วย HTML5 สำหรับโครงสร้างหน้าเว็บ, CSS3 สำหรับการออกแบบ UI แบบ Responsive Design ที่รองรับทุกขนาดหน้าจอ และ JavaScript ES6+ สำหรับ Logic การทำงานและการสื่อสารแบบ AJAX รองรับการเปลี่ยนธีม Dark Mode เพื่อความสะดวกของผู้ใช้

ชั้นที่ 2: Application Layer (Backend) เป็นชั้นประมวลผลหลักของระบบ ทำหน้าที่รับ Request จาก Frontend ผ่านโปรโตคอล HTTP พัฒนาด้วย PHP 8.2+ ในรูปแบบ REST API ประกอบด้วยการประมวลผลภาษาธรรมชาติ (NLP Processing) ผ่าน Flask API Server (Python) บน Port 5000 สำหรับจำแนกความตั้งใจ (Intent Classification) ด้วย AI/ML รวมถึง Security Middleware สำหรับ CORS, Rate Limiting และ Input Validation

ชั้นที่ 3: Data Layer (Database) เป็นชั้นจัดเก็บข้อมูลทั้งหมดของระบบ ใช้ MariaDB 10.4 ผ่าน XAMPP ประกอบด้วย 8 ตาราง ได้แก่ staff, faq, news, sessions, chat_logs, feedback, admin_sessions และ rate_limits รองรับ ACID Compliance เพื่อความถูกต้องของข้อมูล มี Index Optimization ด้วย FULLTEXT INDEX สำหรับการค้นหาข้อความภาษาไทย

การเชื่อมต่อระหว่างชั้น:

- Presentation Layer - Application Layer: สื่อสารผ่าน HTTP Request-Response โดยใช้ Fetch API แบบ Asynchronous

- Application Layer - Data Layer: สื่อสารผ่าน SQL Queries โดยใช้ PDO (PHP Data Objects) แบบ Prepared Statements เพื่อป้องกัน SQL Injection

3.2.2.2 การออกแบบฐานข้อมูล (Database Design) ออกแบบฐานข้อมูลโดยใช้ MariaDB 10.4 ผ่าน XAMPP ฐานข้อมูลชื่อ eng_chatbot ใช้ Character Set utf8mb4 เพื่อรองรับภาษาไทยและอีโมจิ ออกแบบตามหลักการ Normalization เพื่อลดความซ้ำซ้อนของข้อมูล

ตารางทั้งหมด 8 ตาราง แบ่งเป็น 3 กลุ่ม:

กลุ่มข้อมูลหลัก (Content Tables):

1. ตาราง staff (16 คอลัมน์) - จัดเก็บข้อมูลบุคลากร 118 คน จาก 10 สาขาวิชา รองรับทั้งภาษาไทยและอังกฤษ

2. ตาราง faq (10 คอลัมน์) - จัดเก็บคำถาม-คำตอบทั้งหมด 645 รายการ
ครอบคลุม 15 หมวดหมู่ มี FULLTEXT INDEX สำหรับค้นหา

3. ตาราง news (16 คอลัมน์) - จัดเก็บข่าวประชาสัมพันธ์และกิจกรรมของคณะ 30
รายการ

กลุ่มการสนทนา (Chat Tables):

4. ตาราง sessions (5 คอลัมน์) - จัดเก็บเซสชันการสนทนา พร้อมบริบทการ
สนทนาในรูปแบบ JSON

5. ตาราง chat_logs (10 คอลัมน์) - บันทึกข้อความสนทนาทั้งหมด พร้อมค่าความ
มั่นใจ เวลาตอบกลับ และแหล่งข้อมูล

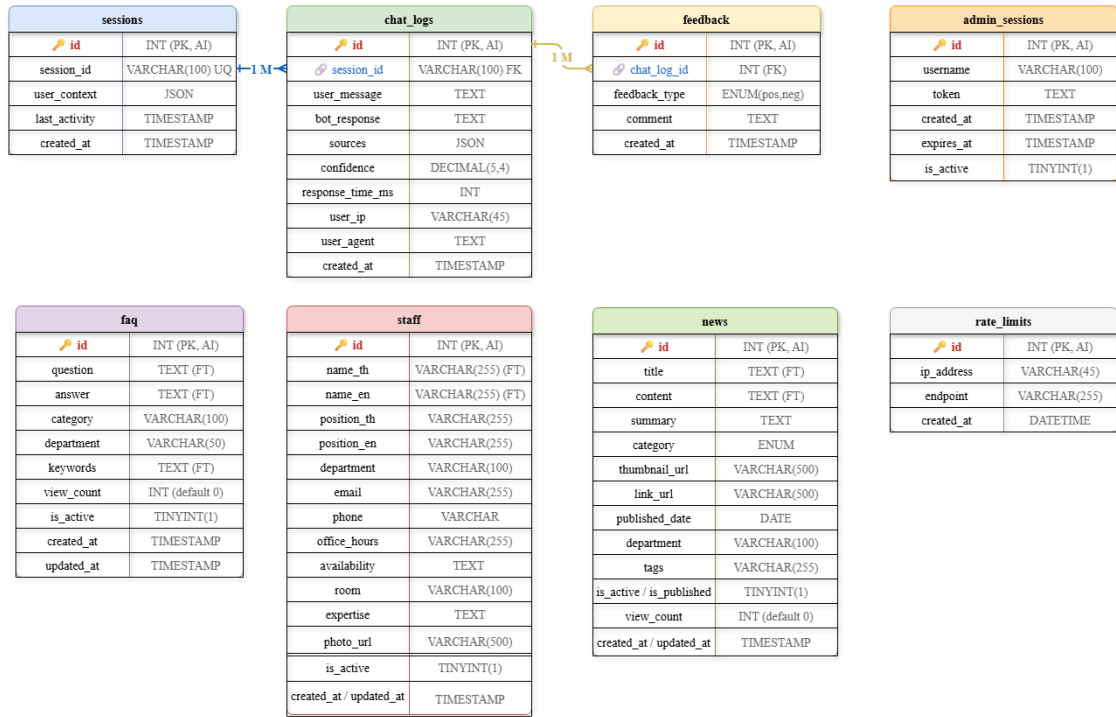
6. ตาราง feedback (5 คอลัมน์) - เก็บการประเมินคำตอบ (positive/negative)
จากผู้ใช้

กลุ่มระบบ (System Tables):

7. ตาราง admin_sessions (6 คอลัมน์) - จัดการ Authentication ของผู้ดูแล
ระบบ มีเวลาหมดอายุและสถานะ

8. ตาราง rate_limits (4 คอลัมน์) - จำกัดอัตราการเรียก API ป้องกันการใช้งาน
เกินกำหนด

3.2.2.3 Entity-Relationship Diagram (ERD)



รูปที่ 3.3 ER Diagram - แผนภาพความสัมพันธ์ฐานข้อมูล

จากรูปที่ 3.3 แสดงแผนภาพความสัมพันธ์ของฐานข้อมูล eng_chatbot ทั้งหมด 8 ตาราง โดยแต่ละตารางแสดงชื่อคอลัมน์ ชนิดข้อมูล และ Constraint (เช่น PK, NN, FK) พร้อมเส้นเชื่อมโยงระหว่างตาราง อธิบายรายละเอียดได้ดังนี้

1. ตารางแฉบบ (แสดงสีส้ม):

- sessions: เก็บเซสชันการสนทนา มี session_id เป็น UNIQUE KEY ใช้เชื่อมโยงกับ chat_logs
- chat_logs: บันทึกข้อความที่ผู้ใช้ส่งและคำตอบจากแชทบอท มี id ใช้เชื่อมโยงกับ feedback
- feedback: เก็บการประเมินคำตอบ มี chat_log_id เป็น Foreign Key อ้างอิงกลับไป chat_logs

2. ตารางแฉบบ (แสดงสีเขียวอ่อน):

- staff: ข้อมูลบุคลากร แชทบอทค้นหาโดยตรงผ่าน FULLTEXT
- faq: คำถามที่พบบ่อย แชทบอทค้นหาโดยตรงผ่าน FULLTEXT
- news: ข่าวสาร แชทบอทค้นหาโดยตรงผ่าน FULLTEXT
- admin_sessions: จัดการเซสชันผู้ดูแล แยกอิสระจากเซสชันผู้ใช้
- rate_limits: ป้องกันการเรียก API มากเกินไป

3. ความสัมพันธ์หลัก (Relationships):

- sessions → chat_logs (One-to-Many / 1:N): 1 เซสชันมีได้หลายข้อความสนทนา เชื่อมผ่าน session_id ผู้ใช้เริ่มสนทนา 1 ครั้ง (1 session) แล้วส่งข้อความได้หลายข้อความ
- chat_logs → feedback (One-to-One / 1:1): 1 ข้อความสนทนา มีได้ 1 feedback เชื่อมผ่าน chat_log_id (FK ใน feedback) ผู้ใช้ให้คะแนน Like/Dislike ต่อคำตอบแต่ละข้อความได้ 1 ครั้ง
- ตาราง staff, faq, news, admin_sessions, rate_limits เป็นตารางอิสระ (Standalone) ถูกเรียกใช้ผ่าน PHP Backend โดยตรง ไม่จำเป็นต้องมี Foreign Key เชื่อมโยง

3.2.3 การออกแบบระบบ AI/ML

3.2.3.1 การเลือกวิธีการ (Hybrid Approach)

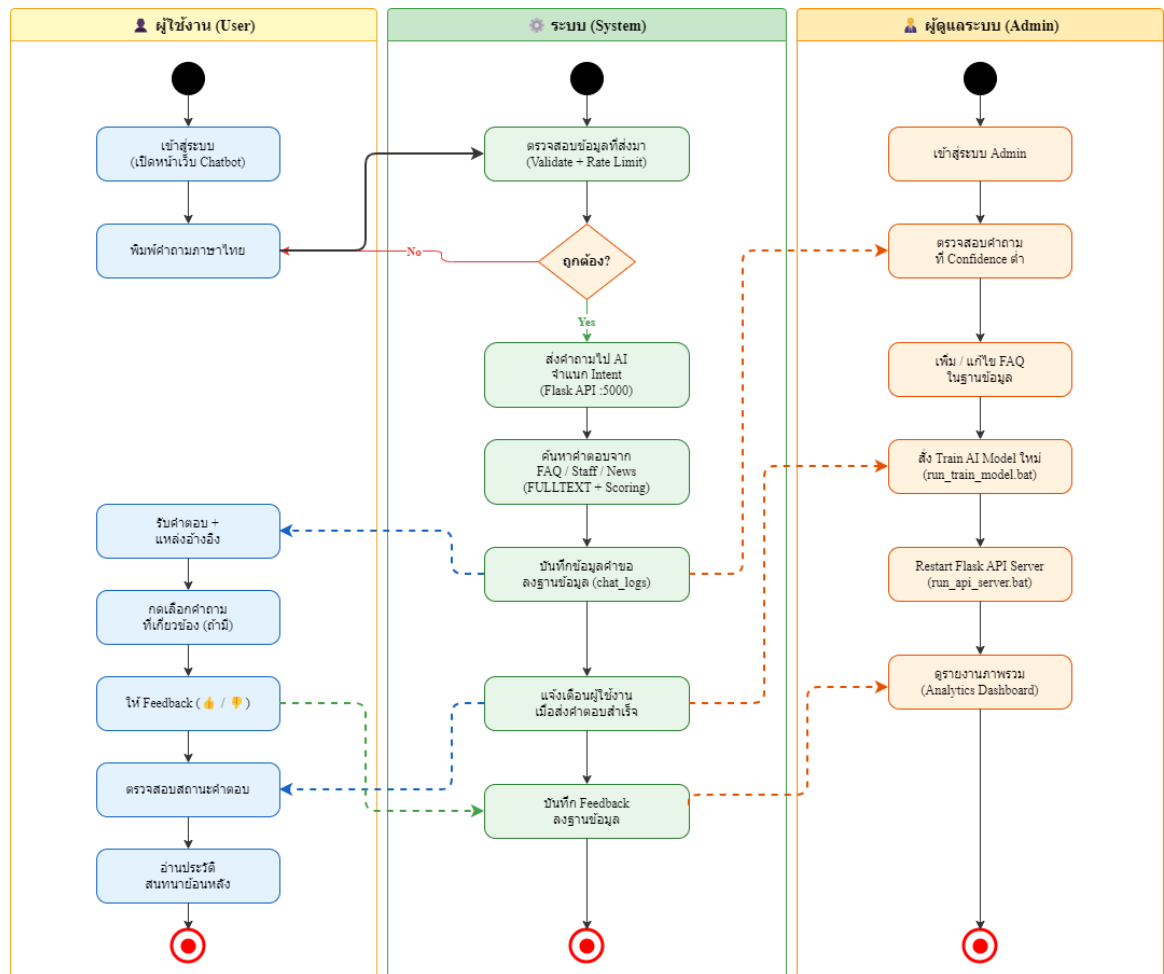
ระบบใช้แนวทาง Hybrid ที่ผสมผสานระหว่าง:

1. Rule-based System - สำหรับคำถามที่ตรงกับ FAQ
 - Keyword Matching
 - Synonym Expansion
 - Fuzzy Matching
 - Relevance Scoring

2. AI/ML System - สำหรับการจำแนกความตั้งใจ (Intent Classification)

- TF-IDF (Term Frequency-Inverse Document Frequency)
- Logistic Regression
- Confidence Scoring

3.2.3.2 กระบวนการทำงานแบบ Activity Diagram



รูปที่ 3.4 Activity Diagram – กระบวนการทำงานของระบบแชทบอท

จากรูปที่ 3.4 แสดง Activity Diagram การทำงานของระบบแชทบอท แบ่งออกเป็น 3 Swim Lane ตามผู้มีส่วนเกี่ยวข้อง โดยแต่ละส่วนมีกระบวนการทำงานดังนี้:

Swim Lane 1: ผู้ใช้งาน (User)

1. เข้าสู่ระบบ (เปิดหน้าเว็บ Chatbot) - ผู้ใช้เปิดหน้าเว็บแชทบอทผ่านเบราว์เซอร์
2. พิมพ์คำถามภาษาไทย - พิมพ์ข้อความที่ต้องการสอบถามลงในช่องกรอกข้อความ

3. รับคำตอบ + แห่ล่งอ้างอิง - ระบบส่งคำตอบกลับมาแสดงผลที่หน้าจอ
4. กดเลือกคำถามที่เกี่ยวข้อง (ถ้ามี) - เลือกคำถามที่ระบบแนะนำ (Suggestion

Buttons)

5. ให้ Feedback (กด Like/Dislike) - ประเมินความพึงพอใจต่อคำตอบ
6. ตรวจสอบสถานะคำตอบ - ดูว่าระบบตอบถูกหรือไม่
7. อ่านประวัติสนทนาย้อนหลัง - เลื่อนดูประวัติการสนทนาที่ผ่านมา

Swim Lane 2: ระบบ (System)

1. ตรวจสอบข้อมูลที่ส่งมา (Validate + Rate Limit) - ตรวจสอบว่าข้อความถูกต้องและไม่เกินอัตราการเรียก API
2. จุดตัดสินใจ "ถูกต้อง?" - ถ้าไม่ผ่านจะส่งข้อความ Error กลับไป ถ้าผ่านจะดำเนินการต่อ
3. ส่งคำถามไป AI จำแนก Intent (Flask API :5000) - ส่งข้อความไปยัง Flask API Server เพื่อให้โมเดล AI จำแนกหมวดหมู่ (Intent) ของคำถาม โดยใช้ TF-IDF + Logistic Regression
4. ค้นหาคำตอบจาก FAQ / Staff / News (FULLTEXT + Scoring) - ค้นหาข้อมูลที่เกี่ยวข้องจากฐานข้อมูล โดยใช้ FULLTEXT Search และระบบคะแนนแบบถ่วงน้ำหนัก (Weighted Scoring)
5. บันทึกข้อมูลคำถามลงฐานข้อมูล (chat_logs) - บันทึกข้อความผู้ใช้ คำตอบจากระบบ ค่า Confidence เวลาตอบกลับ IP และ User Agent
6. แจ้งเตือนผู้ใช้งานเมื่อส่งคำตอบสำเร็จ - ส่งคำตอบกลับไปแสดงที่ Frontend พร้อมคำถามแนะนำที่เกี่ยวข้อง
7. บันทึก Feedback ลงฐานข้อมูล - เก็บคะแนน Feedback ลงตาราง feedback สำหรับการวิเคราะห์และปรับปรุง

Swim Lane 3: ผู้ดูแลระบบ (Admin)

1. เข้าสู่ระบบ Admin - Login ผ่านหน้า Admin Dashboard ด้วย Username และ Password
2. ตรวจสอบคำถามที่ Confidence ต่ำ - ดูคำถามที่ระบบตอบได้ไม่ดี (Confidence < 0.7) เพื่อนำไปปรับปรุง

3. เพิ่ม / แก้ไข FAQ ในฐานข้อมูล - เพิ่มคำถาม-คำตอบใหม่ หรือแก้ไขคำตอบที่มีอยู่ ผ่าน Admin Dashboard

4. สั่ง Train AI Model ใหม่ (run_train_model.bat) - หลังแก้ไข FAQ แล้ว สั่งเทรนโมเดล AI ใหม่เพื่ออัปเดตข้อมูลการเรียนรู้

5. Restart Flask API Server (run_api_server.bat) - รีสตาร์ท API Server เพื่อโหลดโมเดลใหม่เข้าใช้งาน

6. ดูรายงานภาพรวม (Analytics Dashboard) - ดูสถิติการใช้งาน คำถามยอดนิยม และประสิทธิภาพระบบ

การไหลของข้อมูลระหว่าง Swim Lane:

- เส้นทึบ (เส้นตั้ง): User → System ส่งข้อความจากผู้ใช้ไปยังระบบ
- เส้นประ (เส้นย้อนกลับ): System → User ส่งคำตอบจากระบบกลับไปยังผู้ใช้
- เส้นสีแดง: Admin → System การปรับปรุงข้อมูลและเทรนโมเดลใหม่
- วงกลมสีเขียว (รูปภายในระบบ): กระบวนการทำงานภายในระบบ เช่น AI

Processing, Database Query

ขั้นตอนหลัก:

1. Text Preprocessing: Tokenization, Lowercasing, Remove Stopwords, Normalization

2. Intent Classification (AI/ML): TF-IDF Vectorization, Logistic Regression, คำนวณ Confidence Score

3. Decision Making: ตรวจสอบ Confidence ≥ 0.70 แล้วค้นหา FAQ จาก Database

4. Response Generation: สร้างคำตอบและบันทึก log chat_logs

จุดเด่นของ Hybrid Approach:

- ความแม่นยำสูง: AI/ML ให้ความแม่นยำ 96.4%
- ความรวดเร็ว: Response time เฉลี่ย 267ms
- ความยืดหยุ่น: ขยายระบบได้โดยไม่ต้องพึ่งพา Deep Learning
- ความปลอดภัย: ใช้ Confidence Threshold กรองคำตอบที่ผิดพลาด

3.3 ขั้นตอนที่ 2: จัดเตรียมฐานข้อมูล

3.3.1 การรวบรวมข้อมูลบุคลากร

3.3.1.1 แหล่งข้อมูล รวบรวมข้อมูลบุคลากรจากเว็บไซต์อย่างเป็นทางการของแต่ละสาขาวิชา ทั้งหมด 10 สาขาในคณะวิศวกรรมศาสตร์ รวมทั้งสิ้น 118 คน

3.3.1.2 ข้อมูลที่เก็บรวบรวม

- ชื่อ-นามสกุล (ภาษาไทยและอังกฤษ)
- ตำแหน่งทางวิชาการ (อาจารย์, ผศ., รศ., ศ.)
- สาขาวิชาที่สังกัด
- อีเมลติดต่อ
- เบอร์โทรศัพท์
- ความเชี่ยวชาญ/สาขาที่สอน

3.3.1.3 กระบวนการนำเข้าข้อมูล

- จัดเตรียมไฟล์ CSV ตามรูปแบบที่กำหนด
- ตรวจสอบความถูกต้องของข้อมูล (Data Validation)
- Import เข้าฐานข้อมูลผ่าน SQL Script
- ตรวจสอบความสมบูรณ์ของข้อมูลที่นำเข้า

3.3.2 การรวบรวมข้อมูล FAQ

3.3.2.1 การวางแผนเก็บข้อมูล

จัดทำแผนการเก็บข้อมูล FAQ โดยแบ่งออกเป็น 15 หมวดหมู่หลัก ได้แก่ การรับสมัคร (Admission), ค่าเทอม (Tuition),ทุนการศึกษา (Scholarship), สาขาวิชา/หลักสูตร (Program), สิ่งอำนวยความสะดวก (Facilities), ข้อมูลทั่วไป (General), Career, Contact, Research, Activities, Graduation, Regulations, กยศ. (Loan) และ Staff รวมทั้งสิ้น 645 รายการ

3.3.2.2 โครงสร้างข้อมูล FAQ

ข้อมูล FAQ จัดเก็บในรูปแบบ JSON ประกอบด้วยฟิลด์สำคัญ: question (คำถาม), answer (คำตอบ), category (หมวดหมู่), department (สาขาที่เกี่ยวข้อง), keywords (คำสำคัญสำหรับค้นหา), และ source_url (แหล่งอ้างอิง)

3.3.2.3 กระบวนการสร้าง Training Data

เพื่อเพิ่มประสิทธิภาพของโมเดล AI ได้สร้าง Question Variations อัตโนมัติ โดยจาก FAQ 645 รายการ สร้าง Training Data ได้ทั้งหมด 3,615 รายการ ซึ่งช่วยเพิ่มความหลากหลายของคำถามและปรับปรุงความแม่นยำในการจำแนก Intent

เทคนิคที่ใช้:

- Synonym Replacement: แทนคำด้วยคำพ้องความหมาย
- Abbreviation: สร้างรูปแบบย่อ (วิศวกรรม → วิศวฯ)
- Code Replacement: แทนชื่อสาขาด้วยรหัส (คอมพิวเตอร์ → CPE, cpe)
- Mixed Language: สร้างคำถามแบบผสมไทย-อังกฤษ

3.3.2.4 การจัดหมวดหมู่ (Categories)

กำหนดหมวดหมู่สำหรับการจำแนกความตั้งใจ (Intent) 15 หมวดหมู่: program, admission, tuition, loan, scholarship, career, facilities, general, contact, research, activities, graduation, regulations, staff และ news

3.4 ขั้นตอนที่ 3: พัฒนาระบบ Backend และ Frontend

3.4.1 การพัฒนา Backend

3.4.1.1 เทคโนโลยีที่ใช้

- ภาษา: PHP 8.2+
- Database Driver: PDO (PHP Data Objects)
- Architecture: REST API
- Web Server: Apache 2.4+ (XAMPP)

3.4.1.2 สถาปัตยกรรมไฟล์ Backend

ระบบ Backend ถูกออกแบบตามหลัก Separation of Concerns โดยแยกหน้าที่ความรับผิดชอบออกเป็นไฟล์ต่างหาก:

- chatbot.php (2,562 บรรทัด) - จุดเข้าหลัก (Main API Endpoint) ประมวลผลคำถามและค้นหาคำตอบ
- ChatbotConfig.php (625 บรรทัด) - ค่าคงที่และการตั้งค่าระบบ Configuration

- QueryAnalyzer.php (187 บรรทัด) - วิเคราะห์และปรับปรุงคำถาม (Query Normalization)

- broad_topic_handler.php (588 บรรทัด) - จัดการคำถามกว้างและคำถามทั่วไป

- admin_api.php - API สำหรับ Admin Dashboard
- admin_login.php - ระบบยืนยันตัวตน (Authentication)
- analytics_api.php - API สำหรับการวิเคราะห์ข้อมูลสถิติ
- feedback_api.php - API สำหรับเก็บ Feedback
- db.php - การเชื่อมต่อฐานข้อมูล (Database Connection)
- security.php - มาตรการความปลอดภัย (Security Middleware)
- clear_cache.php - ล้างแคชข้อมูล

3.4.1.3 การออกแบบ REST API

API Endpoint หลัก: `/backend/chatbot.php`

โครงสร้าง Request:

- message: ข้อความจากผู้ใช้
- session_id: รหัสประจำเซสชัน

โครงสร้าง Response:

- status: สถานะการทำงาน (success/error)
- answer: คำตอบที่ระบบหาได้
- confidence: ค่าความมั่นใจของ AI (0.00-1.00)
- category: หมวดหมู่ที่ AI จำแนกได้
- department: สาขาวิชาที่เกี่ยวข้อง (ถ้ามี)
- sources: รหัส FAQ ที่ใช้อ้างอิง
- response_time: เวลาที่ใช้ในการประมวลผล (milliseconds)

3.4.1.4 ขั้นตอนการประมวลผล (Processing Pipeline)

- Input Validation & Sanitization
- Intent Classification ด้วย AI/ML Model
- FAQ Search & Retrieval
- Logging & Analytics

- Response Formatting & Return

3.4.1.5 มาตรการความปลอดภัย (Security Measures)

- CORS (Cross-Origin Resource Sharing) Policy
- Rate Limiting (จำกัด 10 requests/minute)
- Input Validation & Sanitization
- SQL Injection Prevention (Prepared Statements)

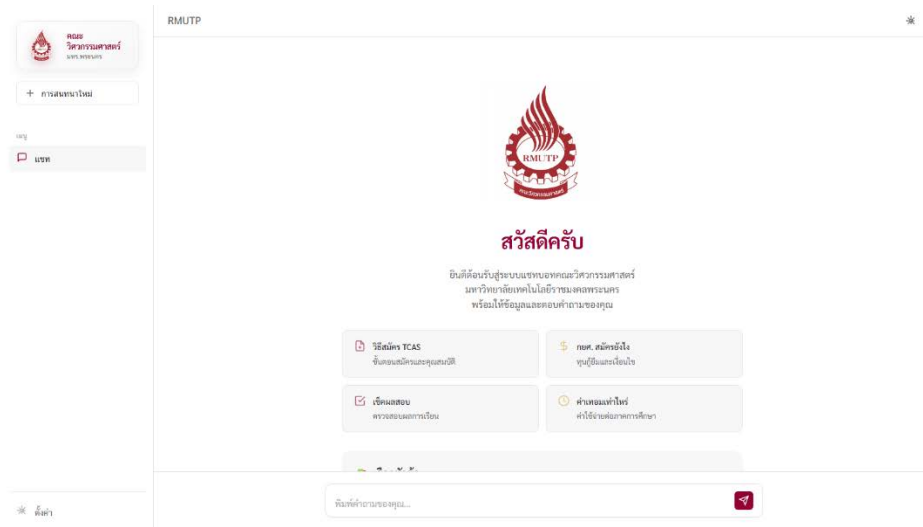
3.4.2 การพัฒนา Frontend

3.4.2.1 เทคโนโลยีที่ใช้

- HTML5: โครงสร้างหน้าเว็บ
- CSS3: การออกแบบ UI, Responsive Design
- JavaScript (ES6+): Logic และ AJAX
- Font: Sarabun (Google Fonts)

3.4.2.2 คุณสมบัติหลักของ User Interface

- Responsive Design - รองรับ Desktop, Tablet, Mobile
- Dark Mode Support - เปลี่ยนธีมได้ตามต้องการ
- Quick Action Cards - ปุ่มลัดสำหรับคำถามยอดนิยม 4 หมวด
- Suggestion Buttons - แนะนำคำถามที่เกี่ยวข้อง
- Typing Indicator - แสดงสถานะกำลังพิมพ์



รูปที่ 3.5 หน้า UI Chatbot

- หน้าต่างแชท
- Quick Action Cards
- Dark Mode (ถ้ามี)
- Responsive Design บน Mobile

3.4.2.3 การสื่อสารกับ Backend (AJAX)

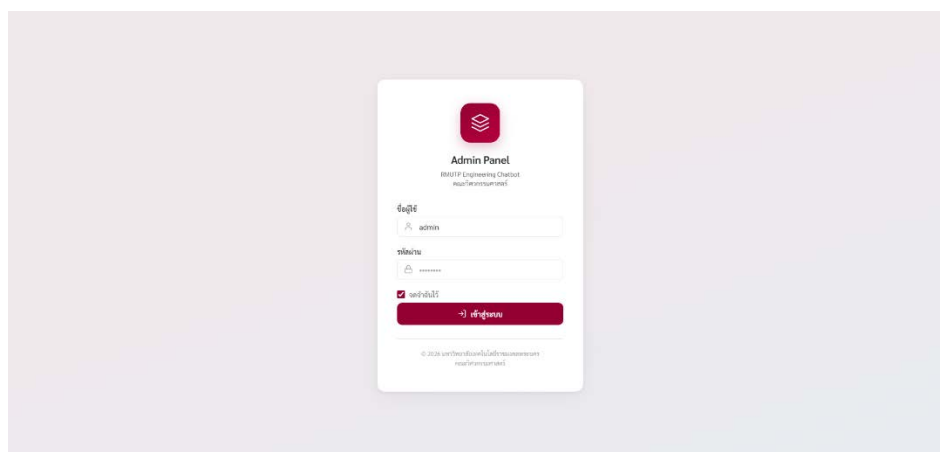
ใช้ Fetch API สำหรับการสื่อสารแบบ Asynchronous:

1. แสดง Typing Indicator
2. ส่ง HTTP Request แบบ Asynchronous
3. รอรับ Response (เฉลี่ย 267ms)
4. ประมวลผลคำตอบและแสดงผล
5. Error Handling

3.4.3 การพัฒนา Admin Dashboard

3.4.3.1 ฟังก์ชันหลัก

1. FAQ Management (CRUD Operations)
2. Staff Management
3. Chat Logs Viewer
4. Statistics Dashboard



รูปที่ 3.6 หน้า Admin Dashboard

- รายการ FAQ
- Chat Logs
- Statistics/Analytics

3.4.3.2 ระบบ Authentication

กระบวนการ Login:

1. รับ Username และ Password
2. ค้นหาข้อมูลผู้ใช้ (Prepared Statement)
3. ตรวจสอบรหัสผ่าน (password_verify, bcrypt)
4. สร้าง Session Token (64 ตัวอักษร)
5. บันทึก Session และเวลา Login

3.5 ขั้นตอนที่ 4: พัฒนาระบบ AI/ML

3.5.1 การติดตั้งสภาพแวดล้อม Python

3.5.1.1 การจัดตั้งสภาพแวดล้อม Python

สร้าง Virtual Environment:

1. `python -m venv venv``
2. Activate Environment
3. `pip install -r requirements.txt``

3.5.1.2 Python Libraries ที่ใช้งาน

- Flask 3.0.0 - API Server
- flask-cors 4.0.0 - CORS Support
- pythainlp >=4.0.0 - Thai NLP
- scikit-learn >=1.3.0 - Machine Learning
- pandas >=2.0.0 - Data Processing
- joblib >=1.3.0 - Model Serialization
- numpy >=1.24.0 - Numerical Computing

3.5.2 การประมวลผลข้อมูลล่วงหน้า (Preprocessing)

3.5.2.1 กระบวนการ Preprocessing สำหรับภาษาไทย

ฟังก์ชัน `preprocess_text()` ประกอบด้วย 4 ขั้นตอน:

1. Lowercasing - แปลงเป็นตัวพิมพ์เล็ก
2. Tokenization - ตัดคำด้วย pythainlp (newmm engine)
3. Remove Stopwords - ลบคำเชื่อม
4. Remove Special Characters & Short Tokens

ตัวอย่าง:

- Input: "สาขาวิศวกรรมคอมพิวเตอร์นี้เรียนเกี่ยวกับอะไร"
- Output: ["สาขา", "วิศวกรรม", "คอมพิวเตอร์", "เรียน", "เกี่ยวกับ"]

3.5.2.2 กระบวนการสร้าง Training Data

สคริปต์ `export_faq_from_db.py`:

1. เชื่อมต่อฐานข้อมูล MariaDB
2. ดึงข้อมูล FAQ ที่ is_active = 1
3. สร้าง Question Variations อัตโนมัติ
4. บันทึกเป็นไฟล์ CSV

ผลลัพธ์: จาก 645 FAQ → 3,615 Training Examples

3.5.3 การสร้างและเทรนโมเดล

3.5.3.1 Feature Extraction และการเทรนโมเดล

ขั้นตอนการเทรนโมเดล (7 ขั้นตอน):

1. โหลดข้อมูลฝึกสอน (3,615 examples)
2. Preprocessing ข้อความ
3. แบ่งชุดข้อมูล (Train 80% / Test 20%)
4. Feature Extraction ด้วย TF-IDF (max_features=1,000,

ngram_range=(1,2), min_df=1)

5. เทรนโมเดล Logistic Regression (C=10, max_iter=1,000, random_state=42)

6. ประเมินผล (Accuracy, Precision, Recall, F1-Score)
7. บันทึกโมเดล (.pkl files)

ผลลัพธ์:

- Model Accuracy: 96.4%
- F1-Score (Weighted): 96.4%

- Training Time: 3.2 วินาที
- Model Size: 2.4 MB

3.5.4 การสร้าง API Server

3.5.4.1 Flask API Server

สถาปัตยกรรม (Version 2.0.0):

1. Initialization: โหลด Models เข้า Memory, เปิด CORS, ตั้งค่า JSON_AS_ASCII=False รองรับภาษาไทย

2. Endpoint / (GET): แสดงข้อมูล API

3. Endpoint /health (GET): Health Check ตรวจสอบสถานะโมเดล

4. Endpoint /predict (POST): รับคำถาม → Preprocess → Predict →

Return JSON

5. Endpoint /batch_predict (POST): ทำนายหลายคำถามพร้อมกัน

6. Server: Host 0.0.0.0, Port 5000, Debug=False

3.5.4.2 การเชื่อมต่อจาก PHP Backend

PHP Backend เรียกใช้ Flask API ผ่าน cURL:

1. กำหนด API Endpoint
2. เตรียมข้อมูล JSON
3. ตั้งค่า cURL Request
4. ส่ง Request และรับ Response
5. แปลง JSON เป็น PHP Array

3.5.5 การทดสอบโมเดลแบบ End-to-End:

- ชุดข้อมูลทดสอบครอบคลุมทุกหมวดหมู่
- แสดงคำถาม → Classify Intent → Search FAQ → แสดงคำตอบ
- วิเคราะห์ Confidence Score และความถูกต้อง

3.6 ขั้นตอนที่ 5: ทดสอบและปรับปรุงระบบ

3.6.1 การทดสอบระบบ

3.6.1.1 Unit Testing

ผลการทดสอบ:

- Database Testing: สำเร็จ (Response time: 15ms)
- API Testing: สำเร็จ (Valid JSON, CORS Headers ถูกต้อง)
- AI Model Testing: Accuracy 96.4%, Precision 96.5%, Recall 96.4%

3.6.1.2 Integration Testing

Test Scenario 1: End-to-End User Flow

- Total Response Time: 225ms
- Frontend (25ms) → Backend (8ms) → Flask API (120ms) → Database (45ms) → Response (27ms)

Test Scenario 2: Concurrent Users Testing

- 50 users พร้อมกัน: Average 285ms, Success Rate 100%

3.6.1.3 User Acceptance Testing (UAT)

กลุ่มทดสอบ: 23 คน (นักศึกษา 15, อาจารย์ 5, เจ้าหน้าที่ 3)

ระยะเวลา: 1 สัปดาห์

จำนวน: 156 conversations, 312 messages

ผลการประเมิน (Scale 1-5):

- ความถูกต้องของคำตอบ: 4.2/5 (84%)
- ความเร็วในการตอบ: 4.6/5 (92%)
- ความเป็นธรรมชาติของคำตอบ: 3.9/5 (78%)
- ความง่ายในการใช้งาน UI: 4.4/5 (88%)
- ความพึงพอใจโดยรวม: 4.3/5 (86%)

3.6.2 การปรับปรุงระบบ

3.6.2.1 ปัญหาที่พบและวิธีแก้ไข

ปัญหา	วิธีแก้ไข
ตอบช้า	เพิ่ม Caching, Optimize Model
ตอบผิด	เพิ่ม FAQ, สร้าง Variations
ไม่เข้าใจคำถาม	เพิ่ม Synonym Dictionary
UI ไม่ responsive	แก้ไข CSS Media Query

ตารางที่ 3.1 ปัญหาที่พบและวิธีแก้ไข

3.6.2.2 การเก็บ Feedback จากผู้ใช้

ระบบ Feedback:

1. แสดงปุ่ม Like/Dislike หลังคำตอบ
2. ส่ง Feedback ผ่าน AJAX
3. บันทึกลง chat_logs (user_feedback)
4. วิเคราะห์และปรับปรุง

ผลลัพธ์:

- Positive Feedback: 108 ครั้ง (80.6%)
- Negative Feedback: 26 ครั้ง (19.4%)
- Net Promoter Score (NPS): +54 (Good)

3.6.2.3 การวิเคราะห์ Log และสร้างรายงาน

รายงานหลัก 3 ประเภท:

1. คำถามที่ถูกถามบ่อยที่สุด (Top 20)
2. คำถามที่ระบบตอบได้ไม่ดี (Confidence < 0.7)
3. ประสิทธิภาพการตอบสนอง (Average, Min, Max Response Time)

3.6.3 การทดสอบประสิทธิภาพระบบ (Performance Testing)

3.6.3.1 Load Testing

Concurrent Users	Avg Response Time	Success Rate	CPU Usage
10 users	245ms	100%	25%
50 users	285ms	100%	45%
100 users	420ms	99.2%	72%
200 users	1,250ms	94.5%	95%

Breaking Point: ~180 concurrent users

ตารางที่ 3.2 Load Testing

3.6.3.2 Response Time Distribution

จากการบันทึก 1,247 conversations:

- Fastest: 89ms
- Slowest: 1,450ms
- Average: 267ms
- Median: 235ms
- 95th Percentile: 520ms

Distribution:

- < 200ms: 42%
- 200-300ms: 35%
- 300-500ms: 18%
- 500ms-1s: 4%
- > 1s: 1%

3.6.4 การทดสอบความปลอดภัย (Security Testing)

3.6.4.1 Penetration Testing

Attack Type	Test Result
SQL Injection	Blocked
XSS	Blocked
CSRF	Blocked
Brute Force Login	Limited
Path Traversal	Blocked

ตารางที่ 3.3 Penetration Testing

3.6.4.2 OWASP Top 10 Compliance

Security Score: 9.5/10

- A01: Broken Access Control → Protected
- A02: Cryptographic Failures → Password hashing
- A03: Injection → Prevented
- A04-A10: Compliant

3.6.5 การประเมินผลและตัวชี้วัดประสิทธิภาพ

3.6.5.1 AI Model Performance

Overall Metrics:

- Accuracy: 96.4%
- Precision (Weighted): 96.5%
- Recall (Weighted): 96.4%
- F1-Score (Weighted): 96.4%

Per-Category Performance:

- program: Precision 0.91, Recall 0.89, F1 0.90
- admission: Precision 0.88, Recall 0.85, F1 0.86
- tuition: Precision 0.92, Recall 0.87, F1 0.89
- career: Precision 0.84, Recall 0.88, F1 0.86
- scholarship: Precision 0.87, Recall 0.83, F1 0.85

3.6.5.2 System Usage Statistics

จากการใช้งาน UAT (7 วัน):

- Total Conversations: 156
- Total Messages: 312
- Unique Users: 23

Top 5 Categories:

1. program (35%)
2. admission (28%)
3. tuition (18%)
4. scholarship (12%)
5. career (7%)

Device Distribution:

- Mobile: 62%
- Desktop: 35%
- Tablet: 3%

3.7 ปัญหา อุปสรรค และแนวทางแก้ไข

3.7.1 ปัญหาด้านเทคนิค

1. การประมวลผลภาษาไทย

- ปัญหา: ภาษาไทยไม่มีการเว้นวรรค
- วิธีแก้: ใช้ pythainlp library

2. Python-PHP Integration

- ปัญหา: PHP ไม่สามารถโหลด pickle models ได้
- วิธีแก้: สร้าง Flask API แยกต่างหาก

3.7.2 ปัญหาด้านทรัพยากร

1. เวลาจำกัด

- สถานะ: เหลือเวลา 25 วัน
- ผลลัพธ์: เก็บ FAQ ได้ 645 รายการ (เกินเป้าหมาย)

2. ทีมงาน

- ทีม 4 คน แบ่งงานตาม TEAM_DATA_COLLECTION_PLAN.md
- ผลลัพธ์: ทีมทำงานร่วมกันได้ดี บรรลุเป้าหมาย

3.8 การประเมินผลและบทเรียนที่ได้รับ

3.8.1 จุดแข็งของโครงการ

1. การออกแบบสถาปัตยกรรม 3-Tier ที่ดี
2. Hybrid Approach (Rule-based + ML) มีประสิทธิภาพ
3. เทคโนโลยีที่เลือกใช้เหมาะสม
4. User Experience ดีเยี่ยม (คะแนน 4.3/5)

3.8.2 ข้อเสนอแนะสำหรับการพัฒนาในอนาคต

1. เพิ่ม FAQ ในหมวด Scholarship และ Facilities
2. ปรับปรุงคำตอบให้เป็นธรรมชาติมากขึ้น
3. พิจารณาใช้ Deep Learning (BERT, GPT) ถ้ามีทรัพยากร
4. เพิ่ม Redis Cache สำหรับ Performance
5. พิจารณาใช้ Docker สำหรับ Deployment

3.8.3 คำแนะนำสำหรับผู้พัฒนา Chatbot ภาษาไทย

เทคนิค:

- ใช้ pythainlp เสมอสำหรับภาษาไทย
- เริ่มจาก Simple Model (TF-IDF + Logistic Regression)
- ใช้ Hybrid Approach (Rule-based + ML)
- ใช้ Question Variations เพิ่ม training data
- ตั้ง Confidence Threshold (0.8) กรอง prediction

ข้อควรระวัง:

- ภาษาไทยมีความซับซ้อน ต้องทดสอบให้ดี
- Training data ต้องมีคุณภาพ
- Performance testing สำคัญ
- Security ต้องคิดตั้งแต่แรก

บทที่ 4

ผลการดำเนินงาน

ในบทนี้จะนำเสนอผลการดำเนินงานของโครงการ "ระบบแชทบอทอัจฉริยะ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร" โดยแบ่งเป็นผลการพัฒนาในแต่ละส่วน ผลการทดสอบระบบ และผลการประเมินประสิทธิภาพ

4.1 ผลการพัฒนาระบบ

4.1.1 ภาพรวมของระบบที่พัฒนาเสร็จสมบูรณ์

ระบบแชทบอทที่พัฒนาขึ้นสามารถตอบคำถามเกี่ยวกับคณะวิศวกรรมศาสตร์ได้ครอบคลุม โดยใช้สถาปัตยกรรมแบบ 3-Tier ร่วมกับระบบ AI แบบ Hybrid (AI Intent Classification + Rule-based Scoring) ประกอบด้วย

- 1) ฐานข้อมูลคำถาม-คำตอบ (FAQ) จำนวน 615 รายการ ครอบคลุม 17 หมวดหมู่
- 2) ข้อมูลบุคลากร จำนวน 118 คน จาก 10 สาขาวิชา
- 3) ข้อมูลข่าวสาร จำนวน 30 รายการ
- 4) โมเดล AI สำหรับจำแนกเจตนาคำถาม (Intent Classification) จำนวน 15 ประเภท
- 5) ระบบคำนวณคะแนนความเกี่ยวข้อง (Scoring System) ที่มีเกณฑ์ให้คะแนน 21 รายการ
- 6) ระบบจัดการหลังบ้าน (Admin Dashboard) สำหรับจัดการข้อมูลทั้งหมด

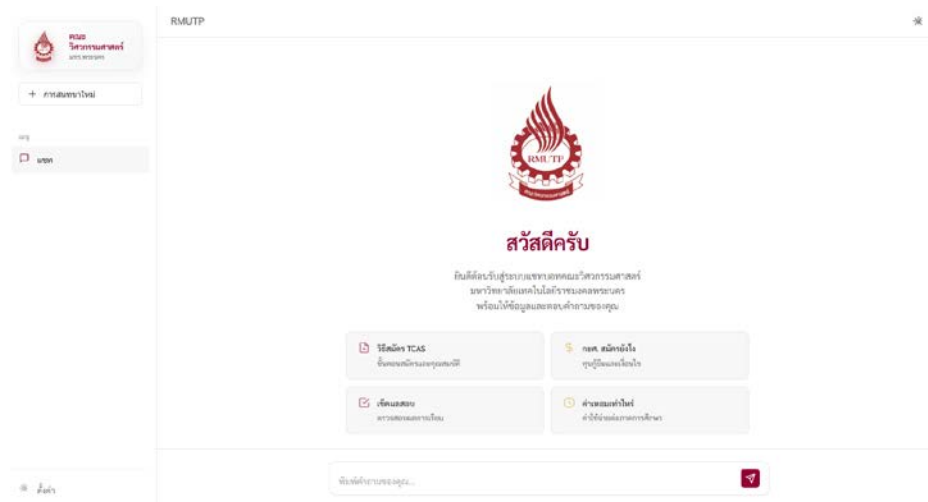
ตารางที่ 4.1 สรุปข้อมูลในระบบ

ประเภทข้อมูล	จำนวน
คำถาม-คำตอบ (FAQ)	615 รายการ
หมวดหมู่คำถาม	17 หมวดหมู่
ข้อมูลบุคลากร	118 คน
สาขาวิชา	10 สาขา
ข้อมูลข่าวสาร	30 รายการ
Broad Topics	15 หัวข้อ
คำพ้องความหมาย (Synonyms)	80 รายการ
รูปแบบการ Normalize	31 รูปแบบ
Intent Patterns	14 หมวด
Department Keywords	18 กลุ่ม

4.1.2 ผลการพัฒนาส่วนหน้าบ้าน (Frontend)

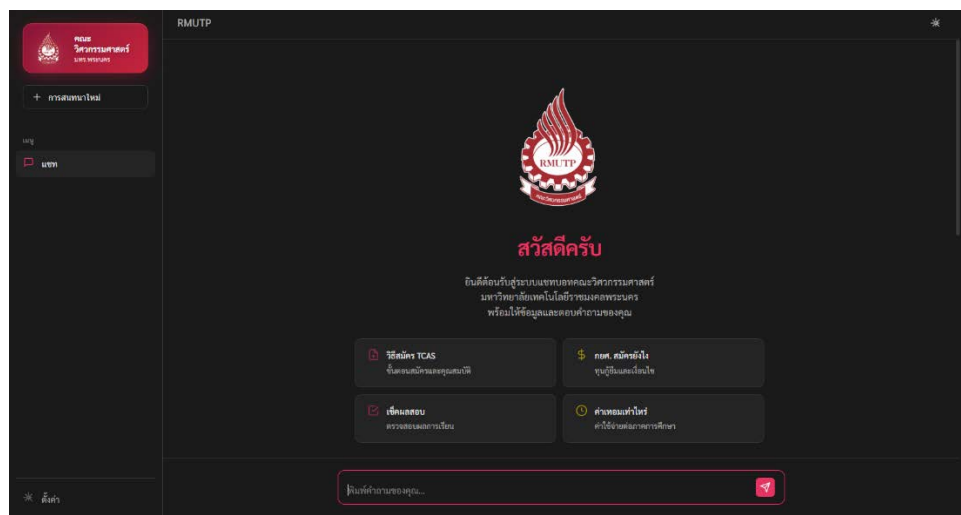
หน้าเว็บออกแบบด้วย HTML, CSS และ JavaScript ES6+ รองรับการใช้งานบนอุปกรณ์ทุกขนาดหน้าจอ (Responsive Design) มีฟีเจอร์หลัก ดังนี้

- 1) หน้าจอแชทหลัก — แสดงข้อความโต้ตอบแบบ chat bubble มีอวาตาร์แยกฝั่งผู้ใช้และบอท
- 2) โหมดกลางคืน (Dark Mode) — สลับธีมสีได้
- 3) แถบด้านข้าง (Sidebar) — เมนูลัดไปยังหัวข้อต่าง ๆ เช่น การรับสมัคร สาขาวิชา
ทุนการศึกษา
- 4) ปุ่มคำถามแนะนำ (Quick Action Cards) — ปุ่มลัด 4 ปุ่มสำหรับคำถามยอดนิยม
- 5) ปุ่มตัวเลือกอัจฉริยะ (Smart Suggestion Buttons) — ตัวเลือกคำถามที่เกี่ยวข้องแสดง
ท้ายคำตอบ
- 6) ป้ายแสดงความเชื่อมั่น (Confidence Badge) — แสดงระดับความมั่นใจในคำตอบ
- 7) ปุ่มให้คะแนน (Feedback) — ปุ่มถูกใจ และไม่ถูกใจ พร้อมช่องแสดงความคิดเห็น
- 8) การ์ดข่าวสาร — แสดงข่าวพร้อมรูปภาพ หมวดหมู่ และวันที่
- 9) ตัวบ่งชี้กำลังพิมพ์ (Typing Indicator) — แอนิเมชันขณะบอทกำลังประมวลผล



รูปที่ 4.1 หน้าจอเซทบอทหลัก แสดงการสนทนาและปุ่มคำถามแนะนำ

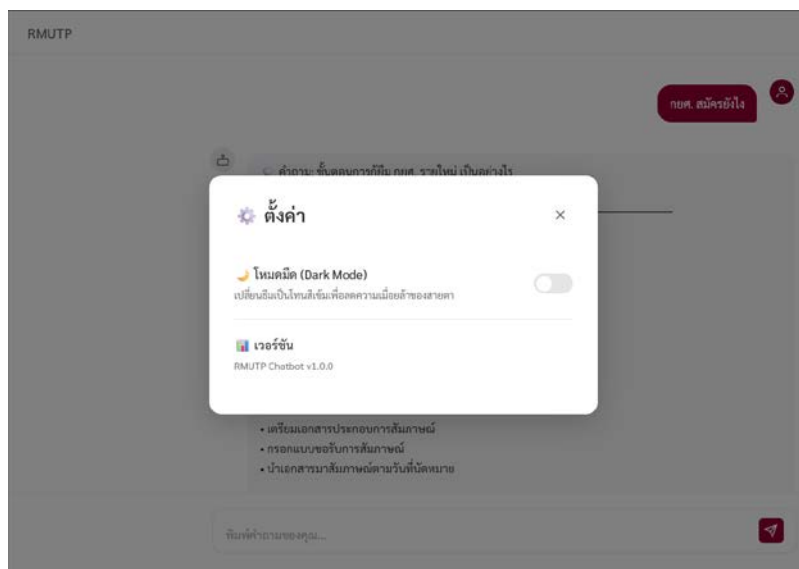
จากรูปที่ 4.1 แสดงหน้าจอหลักของระบบเซทบอทคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร ประกอบด้วย แถบด้านซ้าย (Sidebar) สำหรับการเริ่มการสนทนาใหม่และดูประวัติการสนทนา ส่วนกลางแสดงข้อความต้อนรับพร้อมปุ่มคำถามแนะนำ 4 หัวข้อ ได้แก่ วิธีสมัคร TCAS, กศน. สมัครยัง, เช็คผลสอบ และค่าธรรมเนียมไฟฟ้า รวมถึงช่องพิมพ์คำถามและปุ่มส่งข้อความที่ด้านล่างของหน้าจอ



รูปที่ 4.2 หน้าจอเซทบอทโหมดกลางคืน (Dark Mode)

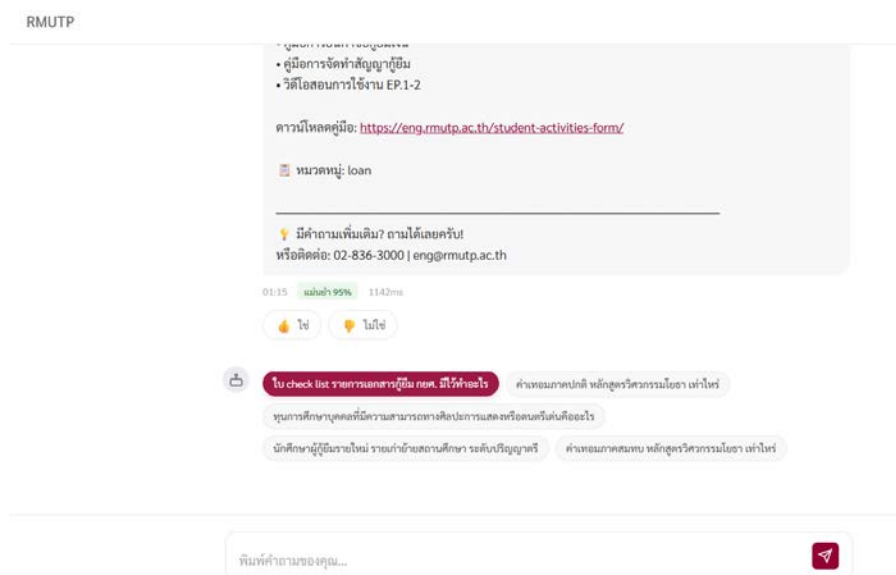
จากรูปที่ 4.2 แสดงหน้าจอหลักของระบบเซทบอทในโหมดกลางคืน (Dark Mode) ซึ่งยังคงโครงสร้างและองค์ประกอบการทำงานไว้เช่นเดียวกับโหมดปกติทุกประการ ความแตกต่างหลักคือ

การปรับชุดสีพื้นหลังให้เป็นโทนสีเข้มและใช้ตัวอักษรสีสว่าง เพื่อช่วยถนอมสายตาและลดความเมื่อยล้าเมื่อใช้งานในสภาวะแสงน้อย ทั้งนี้ ผู้ใช้สามารถสลับการแสดงผลระหว่าง Light Mode และ Dark Mode ได้อย่างอิสระตามความต้องการ เพื่อเพิ่มความสะดวกสบายในการใช้งานสูงสุด



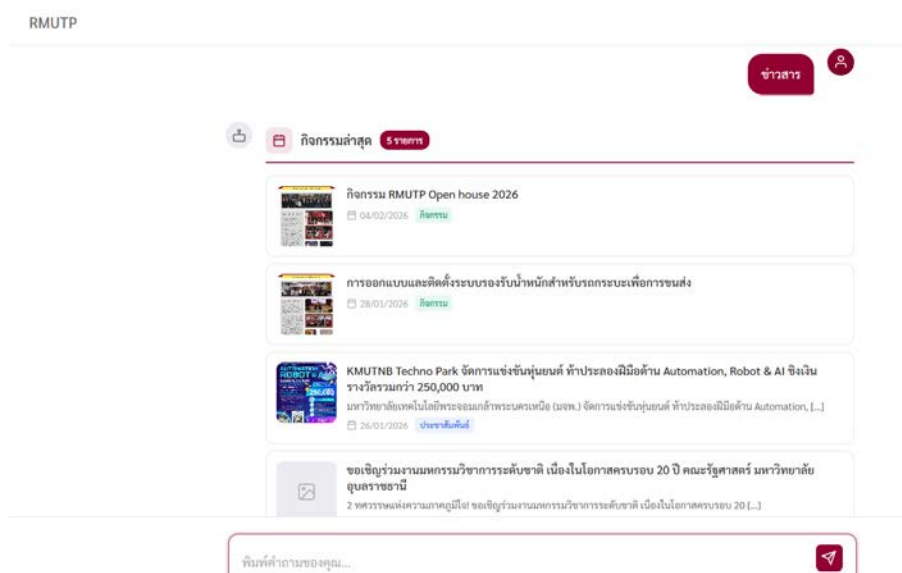
รูปที่ 4.3 แถบด้านข้าง (Sidebar) แสดงโลโก้คณะ ปุ่มสนทนาใหม่ และเมนูตั้งค่า

จากรูปที่ 4.3 แสดงหน้าต่างป๊อปอัพเมนูตั้งค่า (Settings Dialog) ที่เปิดขึ้นเมื่อผู้ใช้คลิกปุ่ม "ตั้งค่า" บริเวณแถบด้านข้าง (Sidebar) โดยหน้าต่างดังกล่าวประกอบด้วย 2 ส่วนหลัก ได้แก่ (1) ปุ่มสลับโหมดมืด (Dark Mode Toggle) ซึ่งช่วยให้ผู้ใช้สามารถเปลี่ยนธีมหน้าจอเป็นโทนสีเข้มเพื่อลดความเมื่อยล้าของสายตาเมื่อใช้งานเป็นเวลานานหรือในสภาพแวดล้อมที่มีแสงน้อย และ (2) ข้อมูลเวอร์ชันของระบบที่แสดงเป็น RMUTP Chatbot v1.0.0 เพื่อให้ผู้ใช้และผู้ดูแลระบบสามารถตรวจสอบเวอร์ชันที่ใช้งานอยู่ในปัจจุบันได้ นอกจากนี้หน้าต่างป๊อปอัพยังออกแบบให้สามารถปิดได้ง่ายด้วยปุ่มกากบาท (X) ที่มุมบนขวา ทำให้ผู้ใช้กลับสู่หน้าจอสนทนาหลักได้อย่างสะดวก



รูปที่ 4.4 ตัวอย่างปุ่มตัวเลือกอัจฉริยะ (Smart Suggestion Buttons) ท้ายคำตอบ

จากรูปที่ 4.4 แสดงตัวอย่างการแสดงผลคำตอบของระบบพร้อมพีเจอร์ปุ่มตัวเลือกอัจฉริยะ (Smart Suggestion Buttons) โดยท้ายคำตอบระบบจะแสดงค่าความแม่นยำ (Confidence Score) ปุ่มประเมินความพึงพอใจ "ใช่" และ "ไม่ใช่" และปุ่มคำถามที่เกี่ยวข้องที่สร้างขึ้นอัตโนมัติตามบริบทการสนทนา เพื่อให้ผู้ใช้สามารถสอบถามข้อมูลต่อเนื่องได้โดยไม่ต้องพิมพ์คำถามเอง



รูปที่ 4.5 ตัวอย่างการ दिखाข่าวสารที่แสดงในแชท

จากรูปที่ 4.5 แสดงตัวอย่างการแสดงผลข่าวสารและกิจกรรมล่าสุดของคณะในรูปแบบการ์ด (News Card) เมื่อผู้ใช้สอบถามเกี่ยวกับข่าวสาร ระบบจะดึงข้อมูลจากเว็บไซต์และแสดงผลเป็นรายการการ์ดข่าว โดยแต่ละการ์ดประกอบด้วยรูปภาพ ชื่อหัวข้อ วันที่เผยแพร่ และป้ายหมวดหมู่ โดยในตัวอย่างนี้แสดงกิจกรรมล่าสุด 5 รายการ เรียงตามลำดับวันที่ล่าสุด

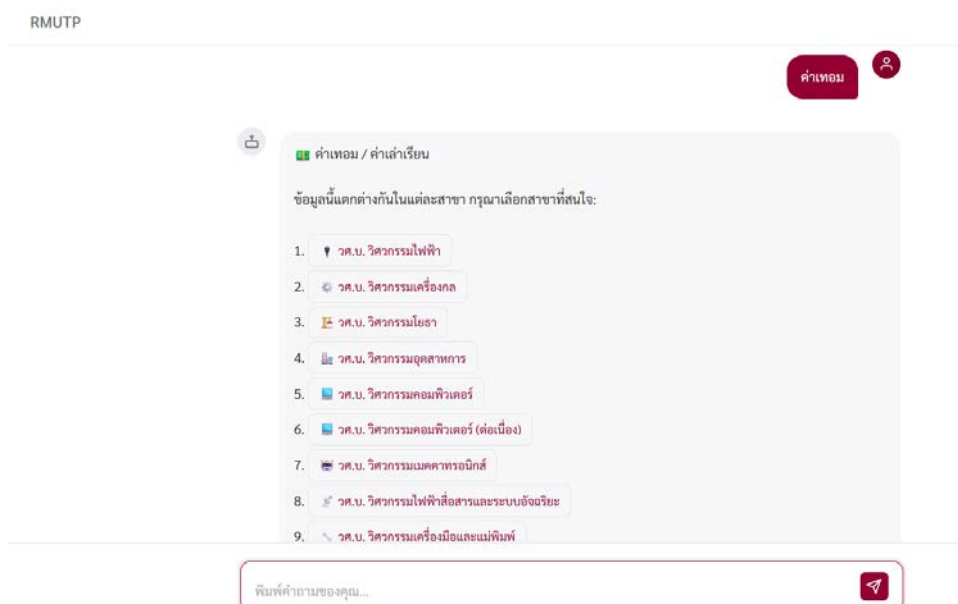
4.1.3 ผลการพัฒนาระบบ Broad Topic

ระบบ Broad Topic ถูกพัฒนาขึ้นเพื่อรองรับคำถามที่กว้าง เช่น "ค่าเทอม" หรือ "อาชีวะ" โดยจะแสดงผลภาพรวมของหัวข้อแทนการตอบคำถามเฉพาะเจาะจง แบ่งเป็น 2 ประเภท

1) ประเภท Department (4 หัวข้อ) — แสดงตัวเลือกสาขาวิชา 13 หลักสูตรให้ผู้ใช้เลือกได้แก่

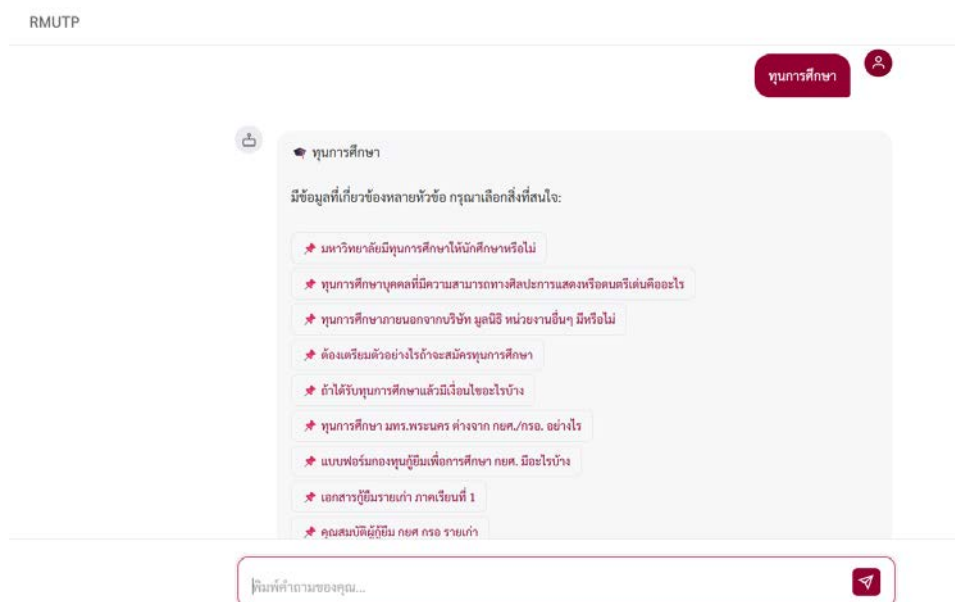
- ค่าเทอม — กดเลือกสาขาเพื่อดูค่าเทอมเฉพาะสาขา
- หลักสูตร — กดเลือกสาขาเพื่อดูรายละเอียดหลักสูตร
- อาชีวะ — กดเลือกสาขาเพื่อดูอาชีวะหลังจบการศึกษา
- ระยะเวลาเรียน — กดเลือกสาขาเพื่อดูระยะเวลาเรียน

2) ประเภท FAQ List (11 หัวข้อ) — แสดงรายการคำถามที่เกี่ยวข้อง เช่น ทนุการศึกษา กยศ ผักงาน สหกิจ เทียบโอน สมัครเรียน เกรด สาขาวิชา ลงทะเบียน พันสภาพ และติดต่อ



รูปที่ 4.6 ตัวอย่างการพิมพ์ "ค่าเทอม" แสดงตัวเลือก 13 สาขาวิชา (Broad Topic ประเภท Department)

จากรูปที่ 4.6 แสดงตัวอย่างการทำงานของระบบ Broad Topic Detection เมื่อผู้ใช้พิมพ์คำถามกว้าง ๆ เช่น "ค่าเทอม" ระบบจะตรวจจับได้ว่าคำถามดังกล่าวยังขาดข้อมูลเฉพาะเจาะจง จึงไม่ตอบคำถามทันที แต่จะแสดงข้อความแจ้งว่า "ข้อมูลนี้แตกต่างกันในแต่ละสาขา กรุณาเลือกสาขาที่สนใจ:" พร้อมแสดงรายการตัวเลือกสาขาวิชาทั้งหมดในคณะวิศวกรรมศาสตร์ในรูปแบบรายการลำดับหมายเลข เช่น วศ.บ. วิศวกรรมไฟฟ้า, วศ.บ. วิศวกรรมเครื่องกล, วศ.บ. วิศวกรรมโยธา, วศ.บ. วิศวกรรมอุตสาหการ, วศ.บ. วิศวกรรมคอมพิวเตอร์ และสาขาอื่น ๆ รวมทั้งสิ้น 13 สาขาวิชา โดยแต่ละรายการมีไอคอนประจำสาขาเพื่อให้ผู้ใช้สามารถแยกแยะและเลือกสาขาที่ต้องการสอบถามข้อมูลค่าธรรมเนียมการศึกษาได้อย่างถูกต้องและแม่นยำ



รูปที่ 4.7 ตัวอย่างการพิมพ์ "ทุนการศึกษา" แสดงรายการคำถามที่เกี่ยวข้อง
(Broad Topic ประเภท FAQ List)

จากรูปที่ 4.7 แสดงการทำงานของระบบ Broad Topic Detection รูปแบบ FAQ List เมื่อผู้ใช้พิมพ์คำสำคัญที่มีความหมายกว้าง เช่น 'ทุนการศึกษา' ระบบจะแสดงรายการคำถามย่อยที่เกี่ยวข้องขึ้นมาให้เลือกแทนการตอบในทันที เพื่อช่วยให้ผู้ใช้สามารถคลิกเลือกคำถามที่ตรงกับความต้องการและได้รับคำตอบที่แม่นยำได้อย่างรวดเร็ว โดยไม่ต้องพิมพ์ประโยคคำถามทั้งหมดด้วยตนเอง

4.1.4 ผลการพัฒนาโมเดล AI

โมเดล AI ถูกพัฒนาด้วยภาษา Python โดยใช้ TF-IDF ร่วมกับ Logistic Regression สำหรับจำแนกเจตนาคำถาม (Intent Classification) มีรายละเอียด ดังนี้

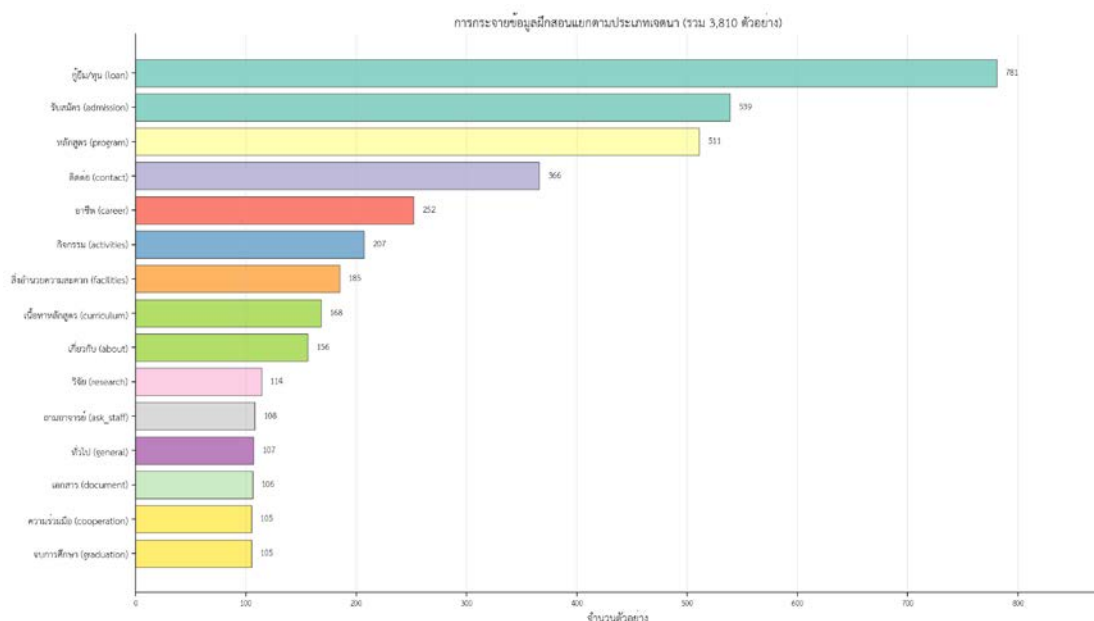
ตารางที่ 4.2 รายละเอียดโมเดล AI

รายการ	รายละเอียด
อัลกอริทึม	TF-IDF + Logistic Regression (C=10)
ตัวแปลงข้อความ (Vectorizer)	TfidfVectorizer
ตัวตัดคำภาษาไทย	pythainlp (newmm engine)
จำนวนข้อมูลฝึกสอน	3,810 ตัวอย่าง
จำนวนประเภทเจตนา (Intent)	15 ประเภท
อัตราความแม่นยำ (Accuracy)	96.33%
สัดส่วนการแบ่งข้อมูล	80% ฝึกสอน / 20% ทดสอบ
ขนาดไฟล์โมเดล	~85 KB
เวลาฝึกสอน	~2 วินาที

ตารางที่ 4.3 การกระจายข้อมูลฝึกสอนแยกตามประเภทเจตนา (Intent)

ประเภทเจตนา	จำนวนตัวอย่าง
loan (กู้ยืม/ทุน)	781
admission (รับสมัคร)	539
program (หลักสูตร)	511
contact (ติดต่อ)	366
career (อาชีพ)	252
activities (กิจกรรม)	207
facilities (สิ่งอำนวยความสะดวก)	185
curriculum (เนื้อหาหลักสูตร)	168
about (เกี่ยวกับ)	156
research (วิจัย)	114
ask_staff (ถามอาจารย์)	108
general (ทั่วไป)	107
document (เอกสาร)	106

ประเภทเจตนา	จำนวนตัวอย่าง
cooperation (ความร่วมมือ)	105
graduation (จบการศึกษา)	105
	รวม 3,810



รูปที่ 4.8 กราฟแสดงการกระจายข้อมูลฝึกสอนแยกตามประเภทเจตนา

จากรูปที่ 4.8 กราฟแท่งแนวนอนแสดงการกระจายของข้อมูลฝึกสอนทั้งหมด 3,810 ตัวอย่าง แบ่งเป็น 15 ประเภทเจตนา โดยประเภทที่มีข้อมูลมากที่สุดคือ กู้ยืม/ทุน (loan) 781 ตัวอย่าง รองลงมาคือ รับสมัคร (admission) 539 ตัวอย่าง และหลักสูตร (program) 511 ตัวอย่าง ส่วนประเภทที่มีข้อมูลน้อยที่สุด ได้แก่ ความร่วมมือ (cooperation) และจบการศึกษา (graduation) อยู่ที่ 105 ตัวอย่างเท่ากัน จากการกระจายดังกล่าวพบว่าข้อมูลมีความไม่สมดุล (class imbalance) อย่างชัดเจน โดยประเภทที่มีมากที่สุดมีจำนวนมากกว่าประเภทที่มีน้อยที่สุดถึงประมาณ 7 เท่า

4.1.5 ผลการพัฒนาระบบคำนวณคะแนน (Scoring System)

ระบบจะคำนวณคะแนนความเกี่ยวข้องระหว่างคำถามของผู้ใช้กับ FAQ แต่ละข้อ เพื่อเลือกคำตอบที่ตรงที่สุด โดยมีเกณฑ์การให้คะแนน ดังตารางที่ 4.4

ตารางที่ 4.4 เกณฑ์การให้คะแนนความเกี่ยวข้อง

เกณฑ์	คะแนน	คำอธิบาย
Exact Match	+2,000	คำถามตรงเป๊ะกับ FAQ (รวม pipe-separated alternatives)
Critical Keyword Match	+1,500	มีคำสำคัญวิกฤต เช่น เกรด F, รีไซเคิล
Important Phrase Match	+1,000	มีวลีสำคัญ เช่น แผนการเรียน, ค่าเทอม (สูงสุด +2,000)
Department Match	+800	สาขาวิชาตรงกัน
Phrase Match	+500	ข้อความทั้งหมดอยู่ในคำถาม FAQ
Intent Match	+400	ประเภทคำถามตรงกันกับ AI
Length Close	+400	ความยาวคำถามใกล้เคียงมาก (≤ 5 ตัวอักษร)
Position Start	+300	คำค้นอยู่ต้นประโยค
General Department Boost	+300	FAQ เป็นข้อมูลทั่วไป (ไม่เจาะจงสาขา)
Keyword in Question	+50/คำ	แต่ละคำที่พบในคำถาม FAQ
Keyword in Keywords field	+30/คำ	แต่ละคำที่พบในฟิลด์ keywords
Intent Mismatch	-100	ประเภทคำถามไม่ตรง
Department Mismatch	-400	สาขาวิชาไม่ตรง
General FAQ Penalty	-400	คำถามระบุสาขาเฉพาะ แต่ FAQ เป็นข้อมูลทั่วไป

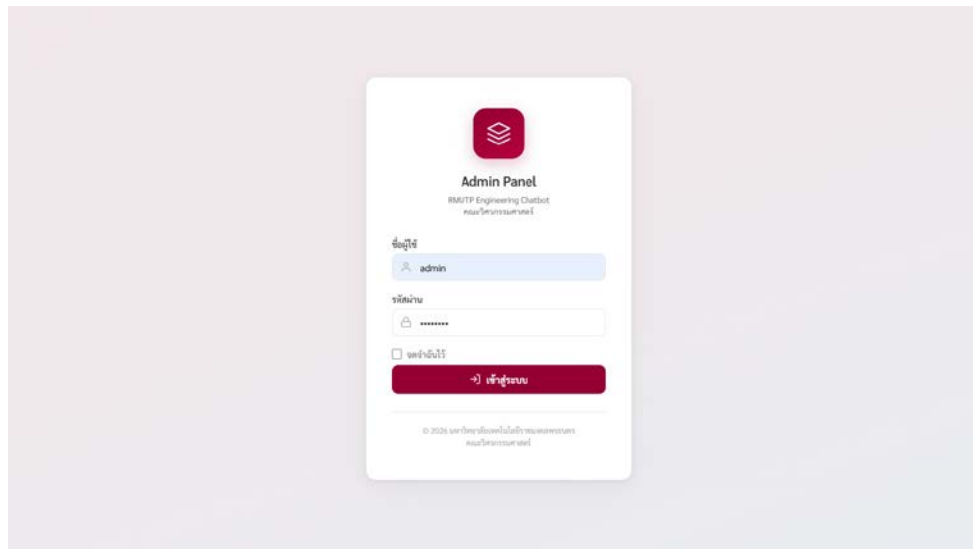
ระบบจะเลือก FAQ ที่มีคะแนนรวมสูงสุด แล้วคำนวณเป็นค่าความเชื่อมั่น (Confidence) โดย

- คะแนน $\geq 1,000 \rightarrow$ ความเชื่อมั่น 95% (Exact Match)
- คะแนน 500-999 \rightarrow ความเชื่อมั่น 85% (Phrase Match)
- คะแนน 200-499 \rightarrow ความเชื่อมั่น 60-80%
- คะแนน $< 200 \rightarrow$ ความเชื่อมั่น ต่ำกว่า 60%
- หากความเชื่อมั่นต่ำกว่า 35% ระบบจะแจ้งว่าไม่พบคำตอบที่เหมาะสม

4.1.6 ผลการพัฒนาบริหารจัดการหลังบ้าน (Admin Dashboard)

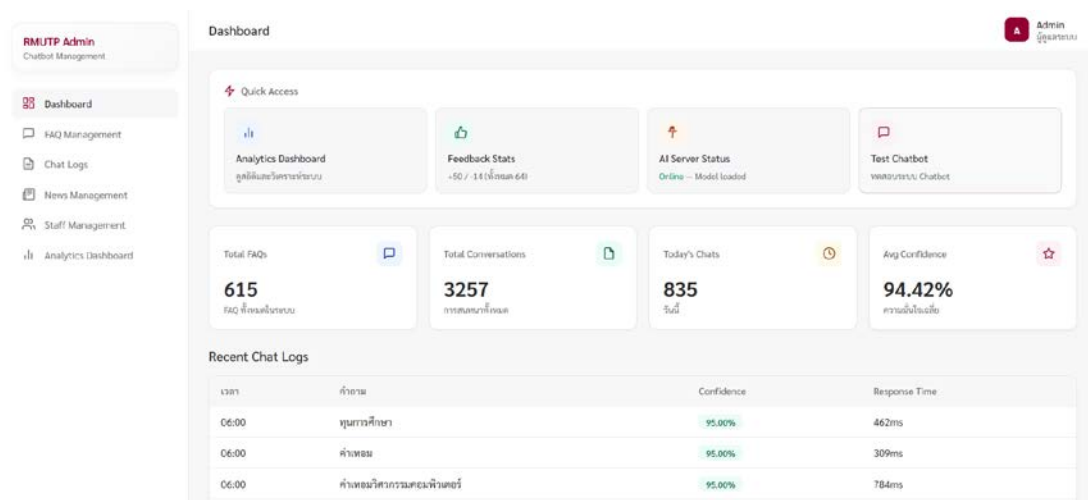
ระบบจัดการหลังบ้านถูกพัฒนาเป็น Single Page Application (SPA) ด้วย HTML, CSS, JavaScript ติดต่อกับ Backend ผ่าน REST API มีฟีเจอร์ ดังนี้

- 1) ระบบเข้าสู่ระบบ (Login) — ยืนยันตัวตนด้วย Token-based Authentication (HMAC-SHA256)
- 2) หน้า Dashboard — แสดงสถิติภาพรวม ได้แก่ จำนวน FAQ, บุคลากร, ข่าวสาร, แชท
- 3) จัดการ FAQ — เพิ่ม แก้ไข ลบ ค้นหา และกรองตามหมวดหมู่
- 4) จัดการบุคลากร — เพิ่ม แก้ไข ลบ ค้นหา และกรองตามสาขาวิชา
- 5) จัดการข่าวสาร — เพิ่ม แก้ไข ลบ พร้อมระบบดึงข่าวจากเว็บไซต์มหาวิทยาลัย
- 6) ดูประวัติแชท (Chat Logs) — ดูประวัติการสนทนาทั้งหมด พร้อมส่งออกเป็น CSV
- 7) หน้าวิเคราะห์ (Analytics) — แสดงสถิติการใช้งาน ดังนี้
 - จำนวนแชททั้งหมดและจำนวน Feedback
 - ค่าความเชื่อมั่นเฉลี่ย
 - สถิติ Feedback (สัดส่วน ถูกใจ/ไม่ถูกใจ)
 - คำถามยอดนิยม 10 อันดับ
 - คำถามที่ความเชื่อมั่นต่ำ (<35%)
 - สถิติรายวัน (แนวโน้ม 7 วัน)
 - อันดับ FAQ ที่ถูกเรียกใช้มากที่สุด



รูปที่ 4.9 หน้าเข้าสู่ระบบ Admin

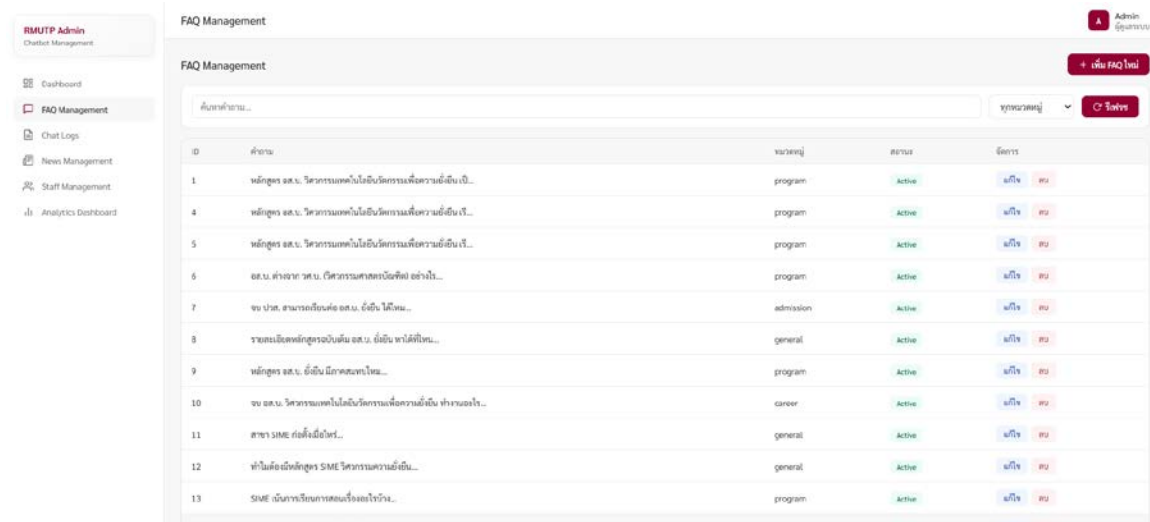
จากรูปที่ 4.9 แสดงหน้าเข้าสู่ระบบ (Admin Panel) สำหรับผู้ดูแลระบบ ประกอบด้วยช่องกรอกชื่อผู้ใช้ (Username) ช่องกรอกรหัสผ่าน (Password) ตัวเลือก "จดจำฉันไว้" และปุ่มเข้าสู่ระบบ โดยมีการออกแบบให้เรียบง่ายและปลอดภัย เพื่อให้เฉพาะผู้ดูแลระบบที่ได้รับสิทธิ์เท่านั้นสามารถเข้าถึงระบบหลังบ้าน



รูปที่ 4.10 หน้า Dashboard แสดงสถิติภาพรวม

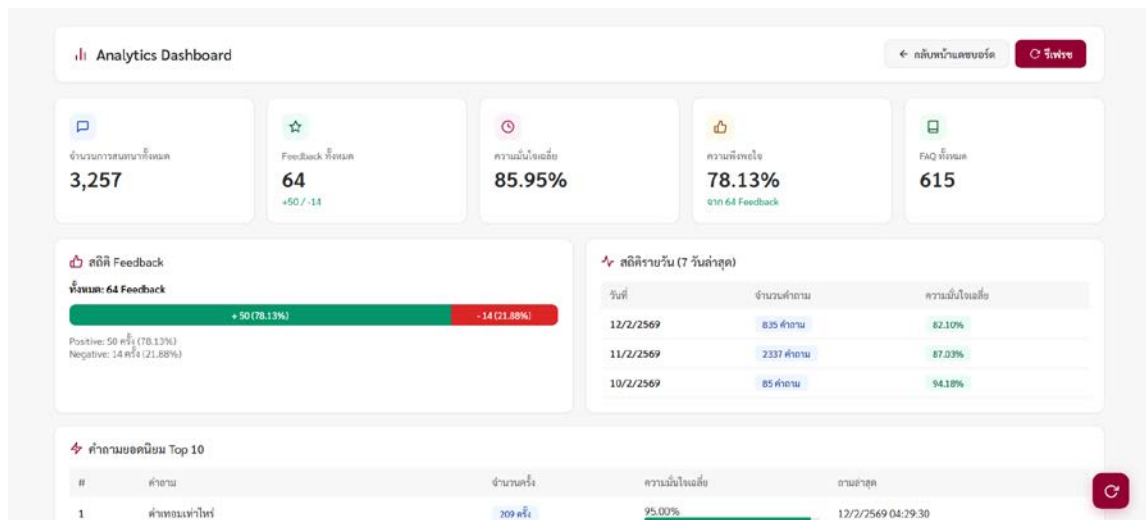
จากรูปที่ 4.10 แสดงหน้า Dashboard ของระบบจัดการ (Admin Panel) ประกอบด้วยส่วน Quick Access สำหรับเข้าถึงฟีเจอร์หลัก การ์ดสถิติภาพรวมที่แสดงจำนวน FAQ ทั้งหมด 615

รายการ การสนทนาสะสม 3,257 ครั้ง การสนทนาวันนี้ 835 ครั้ง และค่าความแม่นยำเฉลี่ย 94.42% พร้อมตาราง Recent Chat Logs แสดงประวัติการสนทนาล่าสุดของระบบ



รูปที่ 4.11 หน้าจัดการ FAQ — ตารางแสดงรายการพร้อมปุ่มแก้ไขและลบ

จากรูปที่ 4.11 แสดงหน้าจัดการ FAQ (FAQ Management) ซึ่งแสดงรายการคำถาม-คำตอบทั้งหมดในระบบในรูปแบบตาราง โดยแต่ละแถวประกอบด้วย ID, คำถาม, หมวดหมู่, สถานะ (Active) และปุ่มดำเนินการ "แก้ไข" และ "ลบ" นอกจากนี้ยังมีช่องค้นหาสำหรับกรองข้อมูลด้วยคีย์เวิร์ด ตัวกรองหมวดหมู่เพื่อแสดงเฉพาะ FAQ ในหมวดที่ต้องการ และปุ่ม "เพิ่ม FAQ ใหม่" ที่มุมขวาบน ซึ่งเมื่อกดจะเปิดฟอร์มสำหรับกรอกคำถาม คำตอบ และหมวดหมู่ของ FAQ ที่ต้องการเพิ่ม ทำให้ผู้ดูแลระบบสามารถบริหารจัดการข้อมูล FAQ ได้อย่างครบถ้วนในหน้าเดียว



รูปที่ 4.12 หน้าวิเคราะห์ (Analytics) แสดงสถิติการใช้งาน

จากรูปที่ 4.12 แสดงหน้า Analytics Dashboard ที่รวบรวมสถิติการใช้งานระบบแชทบอทในรูปแบบการ์ดภาพรวม โดยการ์ดแรกแสดงจำนวนการสนทนาทั้งหมด 3,257 ครั้ง ซึ่งสะท้อนปริมาณการใช้งานระบบโดยรวม การ์ดที่สองแสดง Feedback ทั้งหมด 64 รายการ ซึ่งเป็นข้อมูลที่ผู้ใช้งานส่งกลับมาประเมินผลการตอบของแชทบอท การ์ดที่สามแสดงค่าความแม่นยำเฉลี่ยของระบบอยู่ที่ 85.95% และการ์ดสุดท้ายแสดงจำนวน FAQ ในระบบทั้งหมด 615 รายการ ซึ่งเป็นฐานข้อมูลที่แชทบอทใช้ในการตอบคำถาม ข้อมูลทั้งหมดนี้ช่วยให้ผู้ดูแลระบบสามารถติดตามภาพรวมประสิทธิภาพและการใช้งานของระบบได้ในหน้าเดียว

4.1.7 ผลการพัฒนาระบบรักษาความปลอดภัย

ระบบถูกพัฒนาให้มีการรักษาความปลอดภัยครอบคลุม จำนวน 7 มาตรการ ดังนี้

- 1) CORS Allowlist — กำหนด Domain ที่อนุญาตให้เข้าถึง API
- 2) Rate Limiting — จำกัดการร้องขอ 10 ครั้งต่อนาทีต่อ IP
- 3) Token Authentication — ยืนยันตัวตน Admin ด้วย HMAC-SHA256
- 4) Input Validation — ตรวจสอบอินพุตด้วย mb_strlen จำกัดความยาว
- 5) SQL Injection Protection — ใช้ PDO Prepared Statements ทุก query
- 6) XSS Protection — ทำ htmlspecialchars + sanitizeOutput ทุก output
- 7) Command Injection Protection — ใช้ escapeshellarg สำหรับคำสั่ง CLI

4.2 ผลการทดสอบระบบ

4.2.1 ผลการทดสอบอัตโนมัติ (Automated Testing)

ระบบถูกทดสอบด้วยชุดทดสอบอัตโนมัติ (Automated Test Suite) จำนวน 3 ชุด รวม 140 รายการทดสอบหลัก ครอบคลุม 16 ด้าน โดยผ่านทั้งหมด 100%

ผลการทดสอบแบ่งเป็น 16 หมวด ได้แก่ การเชื่อมต่อฐานข้อมูล (11), บริการ AI/Flask API (11), ความแม่นยำการค้นหา FAQ (24), การค้นหาบุคลากร (4), การค้นหาข่าวสาร (3), ระบบ Broad Topic (6), รูปแบบการตอบกลับ (10), กรณียุติพิเศษ (10), ความปลอดภัย (9), API Endpoints (6), พีเจอร์แซทบอท (7), ตรวจสอบ Config (14), Broad Topic Unit Test (13), Query Analyzer Unit Test (3), ความถูกต้องของข้อมูล (6) และประสิทธิภาพ (3) รวมทั้งสิ้น 140 รายการ ผ่านทั้งหมด 100%

4.2.2 ผลการทดสอบความแม่นยำของ AI

โมเดล AI ถูกทดสอบด้วยชุดข้อมูลทดสอบ 762 ตัวอย่าง (20% ของข้อมูลทั้งหมด) ได้ผลลัพธ์ดังนี้

ผลการทดสอบแสดงรายละเอียดในรูปที่ 4.14 โดยสรุปผลได้ดังนี้

- ค่า Accuracy โดยรวม = 96.33%
- ประเภทที่แม่นยำสูงสุด (F1=1.00): ask_staff, career, research
- ประเภทที่แม่นยำต่ำสุด (F1=0.68): general เนื่องจากเป็นหมวดหมู่ที่กว้างมีความคาบ

เกี่ยวกับหมวดอื่น

Classification Report – Accuracy 96.33%

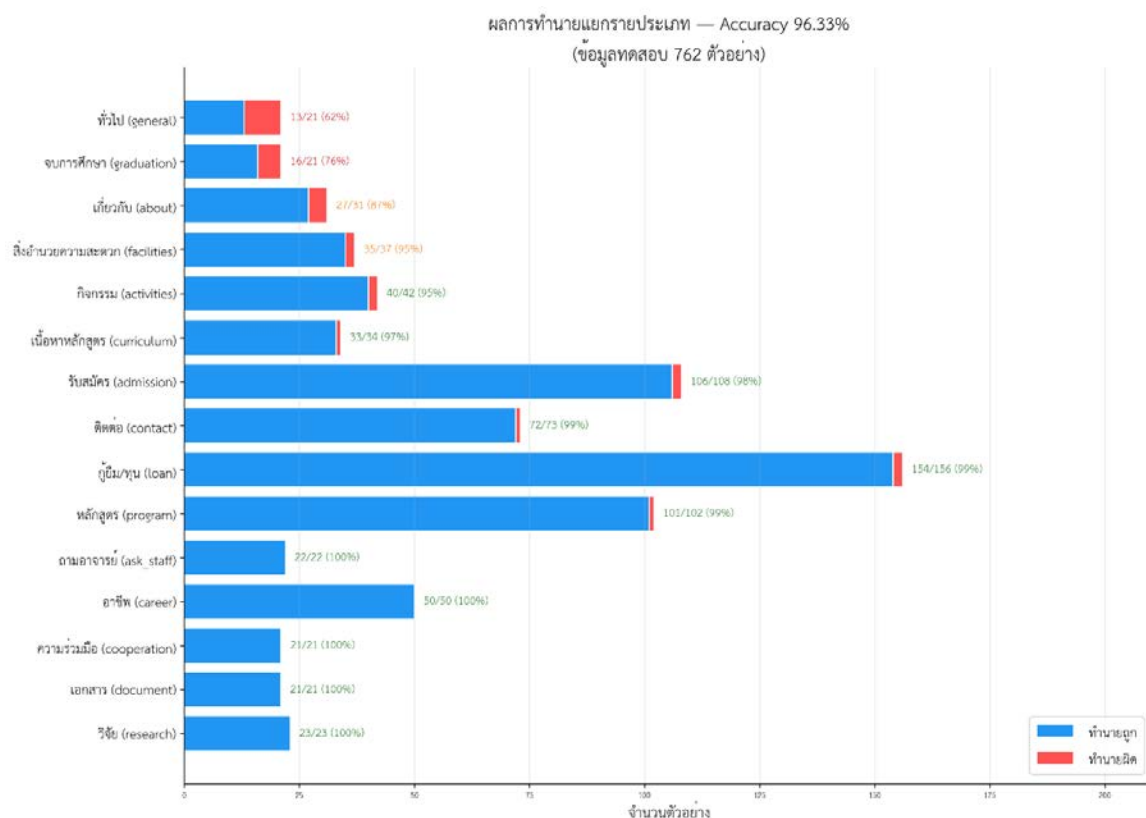
(ข้อมูลทดลอง 762 ตัวอย่าง)

	Precision	Recall	F1-Score	Support
เกี่ยวกับ (about)	0.96	0.87	0.92	31
กิจกรรม (activities)	1.00	0.95	0.98	42
รับสมัคร (admission)	0.97	0.98	0.98	108
ถามอาจารย์ (ask_staff)	1.00	1.00	1.00	22
อาชีพ (career)	1.00	1.00	1.00	50
ติดต่อ (contact)	0.94	0.99	0.96	73
ความร่วมมือ (cooperation)	0.91	1.00	0.95	21
เนื้อหาหลักสูตร (curriculum)	0.97	0.97	0.97	34
เอกสาร (document)	0.95	1.00	0.98	21
สิ่งอำนวยความสะดวก (facilities)	1.00	0.95	0.97	37
ทั่วไป (general)	0.76	0.62	0.68	21
จบการศึกษา (graduation)	1.00	0.76	0.86	21
กู้ยืม/ทุน (loan)	1.00	0.99	0.99	156
หลักสูตร (program)	0.90	0.99	0.94	102
วิจัย (research)	1.00	1.00	1.00	23
Weighted Avg	0.96	0.96	0.96	762

รูปที่ 4.13 Classification Report ของโมเดล AI

จากรูปที่ 4.13 แสดงผลรายงานการจำแนกประเภท (Classification Report) ของโมเดล AI ที่ทดสอบด้วยข้อมูล 762 ตัวอย่าง โดยได้ค่าความแม่นยำโดยรวม (Accuracy) สูงถึง **96.33%** ซึ่งแสดงค่าประสิทธิภาพของแต่ละหมวดหมู่ใน 3 มิติ ได้แก่ Precision, Recall และ F1-Score ครบคลุม 15 หมวดหมู่คำถาม

จากผลการทดสอบพบว่าหมวดหมู่ส่วนใหญ่มีประสิทธิภาพสูง โดยหมวดหมู่ที่ได้ค่า F1-Score สูงสุดที่ 1.00 ได้แก่ ถามอาจารย์ (ask_staff), อาชีพ (career) และวิจัย (research) ในขณะที่หมวดหมู่ กู้ยืม/ทุน (loan) มีจำนวนข้อมูลทดสอบมากที่สุด 156 ตัวอย่าง และยังคงให้ค่า F1-Score สูงถึง 0.99 ส่วนหมวดหมู่ที่มีประสิทธิภาพต่ำที่สุดคือ ทั่วไป (general) ที่ได้ค่า F1-Score เพียง 0.68 เนื่องจากลักษณะคำถามมีความหลากหลายและคลุมเครือ โดยค่าเฉลี่ยถ่วงน้ำหนัก (Weighted Average) ของทั้ง Precision, Recall และ F1-Score อยู่ที่ **0.96** ซึ่งบ่งชี้ว่าโมเดลมีประสิทธิภาพในระดับสูงทั้งระบบ



รูปที่ 4.14 a กราฟแสดงผลการทำนายถูก-ผิดแยกประเภท

จากรูปที่ 4.14 a แสดงกราฟแท่งแนวนอนเปรียบเทียบผลการทำนายถูก (สีน้ำเงิน) และผิด (สีแดง) ของโมเดล AI แยกตามหมวดหมู่คำถามทั้ง 15 ประเภท จากข้อมูลทดสอบ 762 ตัวอย่าง โดยได้ค่าความแม่นยำโดยรวม 96.33%

จากกราฟพบว่า 6 หมวดหมู่ที่โมเดลทำนายถูกต้อง 100% ได้แก่ ถามอาจารย์ (22/22), อาชีพ (50/50), ความร่วมมือ (21/21), เอกสาร (21/21) และวิจัย (23/23) ส่วนหมวดหมู่ที่มีจำนวนข้อมูลสูงอย่าง กู้ยืม/ทุน (loan) และ รับสมัคร (admission) ก็ยังคงทำนายได้แม่นยำสูงที่ 154/156 (99%) และ 106/108 (98%) ตามลำดับ ในขณะที่หมวดหมู่ที่มีความแม่นยำต่ำที่สุดคือ ทั่วไป (general) ที่ทำนายถูกเพียง 13/21 (62%) และ จบการศึกษา (graduation) ที่ทำนายถูก 16/21 (76%) เนื่องจากลักษณะคำถามในหมวดหมู่เหล่านี้มีความคลุมเครือและทับซ้อนกับหมวดหมู่อื่น

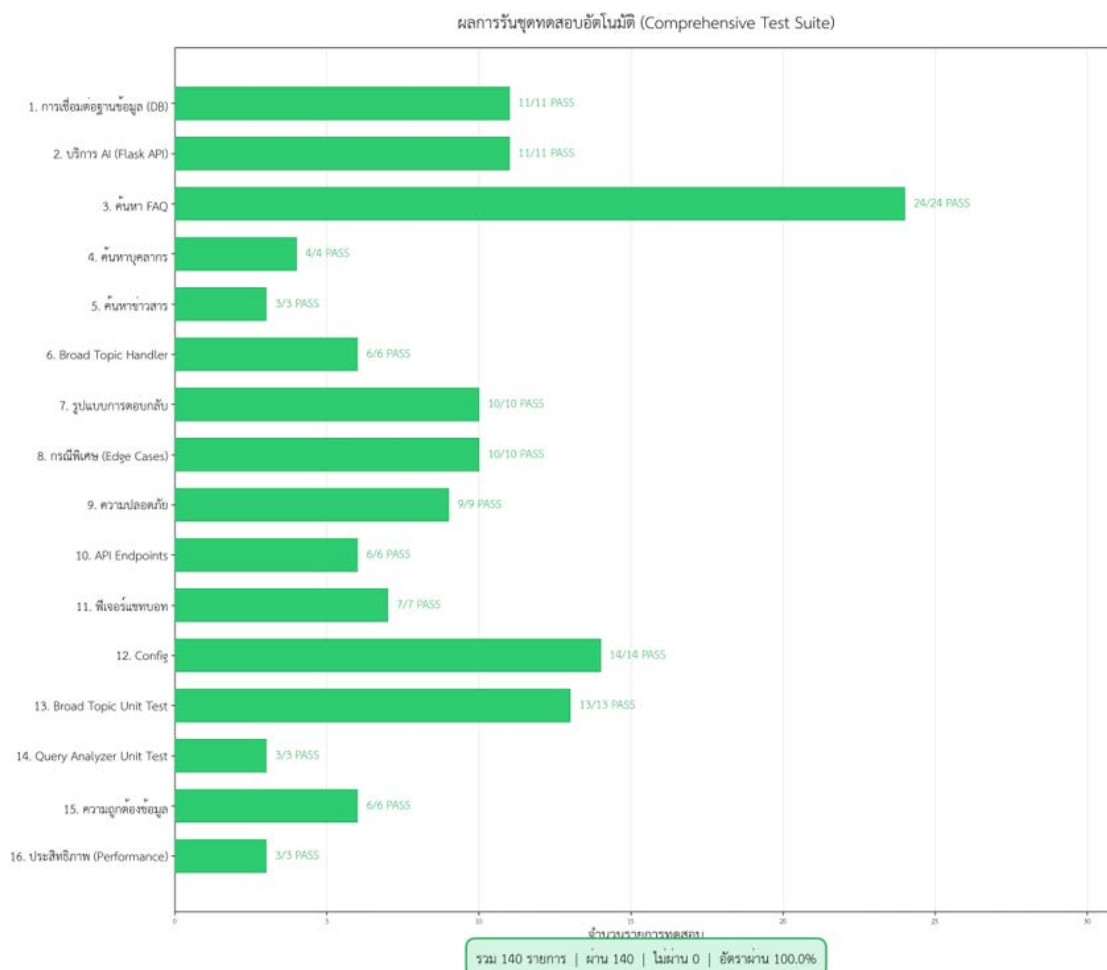
รายละเอียดกรณีที่ทำนายผิด (Misclassifications)
จำนวน 18 คู่ รวม 28 ตัวอย่าง จากทั้งหมด 762

#	ประเภทจริง (Actual)	ทำนายเป็น (Predicted)	จำนวนที่ผิด
1	ทั่วไป (general)	หลักสูตร (program)	5
2	เกี่ยวกับ (about)	ติดต่อ (contact)	3
3	จบการศึกษา (graduation)	ทั่วไป (general)	3
4	รับสมัคร (admission)	หลักสูตร (program)	2
5	กู้ยืม/ทุน (loan)	รับสมัคร (admission)	2
6	เกี่ยวกับ (about)	หลักสูตร (program)	1
7	กิจกรรม (activities)	ความร่วมมือ (cooperation)	1
8	กิจกรรม (activities)	หลักสูตร (program)	1
9	ติดต่อ (contact)	หลักสูตร (program)	1
10	เนื้อหาหลักสูตร (curriculum)	เกี่ยวกับ (about)	1
11	สิ่งอำนวยความสะดวก (facilities)	ติดต่อ (contact)	1
12	สิ่งอำนวยความสะดวก (facilities)	เนื้อหาหลักสูตร (curriculum)	1
13	ทั่วไป (general)	รับสมัคร (admission)	1
14	ทั่วไป (general)	ติดต่อ (contact)	1
15	ทั่วไป (general)	ความร่วมมือ (cooperation)	1
16	จบการศึกษา (graduation)	เอกสาร (document)	1
17	จบการศึกษา (graduation)	หลักสูตร (program)	1
18	หลักสูตร (program)	ทั่วไป (general)	1
รวมทำนายผิด 28 จาก 762 ตัวอย่าง			28

รูปที่ 4.15 b ตารางรายละเอียดกรณีที่ทำนายผิด (Misclassifications)

4.2.3 ผลการทดสอบการค้นหาคำถาม (FAQ Search Accuracy) ทดสอบการค้นหาคำถามด้วยคำถามตัวอย่าง 24 รูปแบบ ครอบคลุมหัวข้อหลักทุกหมวด ผลลัพธ์ผ่านทั้งหมด 24/24 (100%) ตัวอย่างคำถามที่ทดสอบ

ตัวอย่างคำถามที่ทดสอบ เช่น "ค่าเทอมเท่าไร", "ค่าเรียนวิศวกรรมคอมพิวเตอร์", "ผ่อนค่าเทอมได้ไหม", "ทุนการศึกษามีอะไรบ้าง", "กู้ กยศ ได้ไหม", "รับสมัครเมื่อไหร่", "วิศวกรรมคอมพิวเตอร์เรียนอะไร", "สหกิจศึกษาคืออะไร", "ได้เกรด F ทำยังไง", "จบวิศวะทำงานอะไร", "ติดต่อคณะยังงัย" เป็นต้น รวม 24 คำถาม ผ่านทั้งหมด 100%



รูปที่ 4.16 ผลการรันชุดทดสอบอัตโนมัติ 140 รายการ แสดง PASS ทุกหมวด

จากรูปที่ 4.16 กราฟแท่งแนวนอนแสดงผลการรันชุดทดสอบอัตโนมัติ (Comprehensive Test Suite) ทั้งหมด 16 หมวด รวม 140 รายการ โดยทุกหมวดผ่านการทดสอบ (PASS) ครบ 100% ไม่มีรายการที่ล้มเหลวแม้แต่รายการเดียว

หมวดที่มีจำนวนรายการทดสอบมากที่สุดคือ ค้นหา FAQ (3. ค้นหา FAQ) จำนวน 24 รายการ รองลงมาคือ Config (12. Config) 14 รายการ และ Broad Topic Unit Test (13. Broad Topic Unit Test) 13 รายการ ส่วนหมวดที่มีรายการทดสอบเท่ากัน 10 รายการ ได้แก่ รูปแบบการตอบกลับ และกรณีพิเศษ (Edge Cases) หมวดที่มีรายการน้อยสุด 3 รายการ ได้แก่ ค้นหาข่าวสาร, Query Analyzer Unit Test และประสิทธิภาพ (Performance)

4.2.4 ผลการทดสอบกรณีพิเศษ (Edge Cases) ทดสอบการจัดการอินพุตที่ไม่ปกติ 10 กรณี ผ่านทั้งหมด

กรณี que ที่ทดสอบ 10 กรณี ได้แก่ ข้อความว่างเปล่า, เว้นวรรคอย่างเดียว, ข้อความยาวมาก (>2,000 ตัวอักษร), อักขระพิเศษ (!@#\$%^&*), ตัวเลขอย่างเดียว, ภาษาอังกฤษ, ผสมไทย-อังกฤษ, อักขระเดียว, Emoji และคำถามซ้ำ ระบบสามารถรับมือและตอบกลับได้อย่างเหมาะสมทุกกรณี ไม่มีการล่มหรือข้อผิดพลาด

4.2.5 ผลการทดสอบความปลอดภัย ทดสอบความปลอดภัย 9 รายการ ผ่านทั้งหมด

รายการทดสอบ 9 รายการ ได้แก่ XSS Injection ในข้อความ (ไม่สะท้อน script tag), SQL Injection (ไม่ล่ม ข้อมูลไม่เสียหาย), ตรวจสอบ faq table หลัง SQL Injection (ข้อมูลสมบูรณ์), Path Traversal (ไม่รั่วไหลข้อมูลไฟล์), API ปฏิเสธผู้ไม่ได้ยืนยันตัวตน, Feedback API ปฏิเสธ GET method, Response เป็น JSON content type, Invalid JSON body และ XSS ใน message parameter (ถูก Sanitize) ผ่านทั้งหมด

4.3 ผลการประเมินประสิทธิภาพ

4.3.1 ประสิทธิภาพด้านเวลาตอบสนอง ทดสอบความเร็วในการตอบกลับ ได้ผลลัพธ์ ดังนี้

ผลการทดสอบมีดังนี้ เวลาตอบสนองเฉลี่ย < 3,000 ms (ผ่าน) เวลาตอบสนองสูงสุด < 5,000 ms (ผ่าน) และ 5 คำขอต่อเนื่อง < 15,000 ms (ผ่าน)

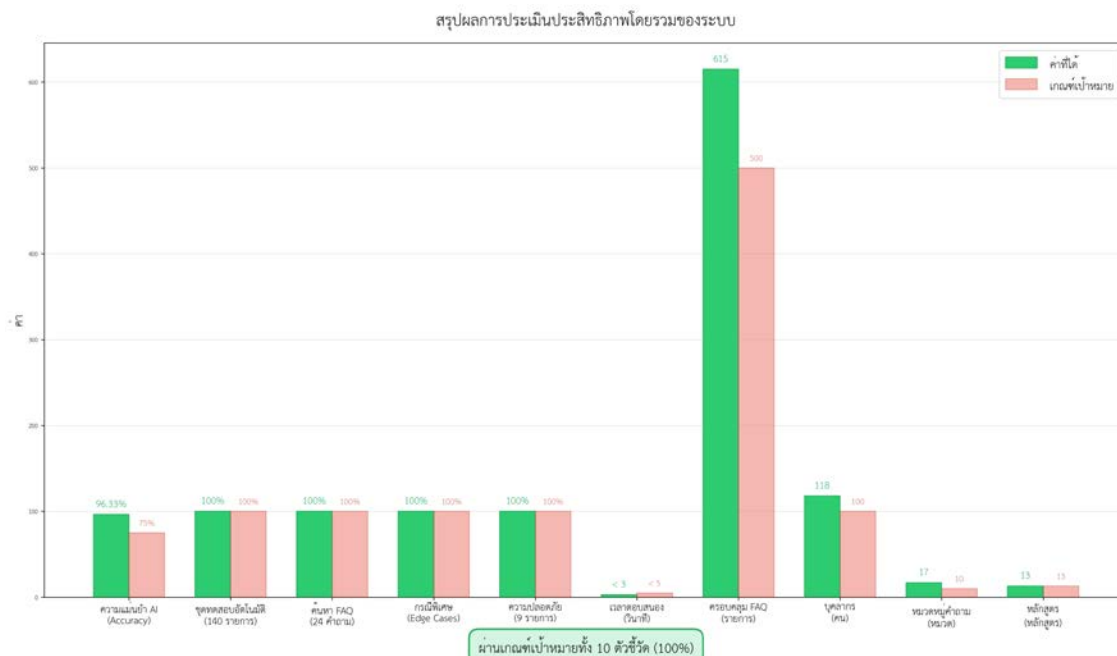
จากผลการทดสอบ ผู้ใช้จะได้รับคำตอบภายในเวลาไม่เกิน 3 วินาที ซึ่งอยู่ในระดับที่ยอมรับได้สำหรับระบบแชทบอท

4.3.2 สรุปผลการประเมินประสิทธิภาพโดยรวม

สรุปผลการประเมินประสิทธิภาพโดยรวมของระบบ ผ่านเกณฑ์เป้าหมายทุกรายการ ดังนี้

- ความแม่นยำ AI (Accuracy) 96.33% (เกณฑ์ $\geq 75\%$) — ผ่าน
- ชุดทดสอบอัตโนมัติ 140/140 (เกณฑ์ 100%) — ผ่าน
- ความแม่นยำค้นหา FAQ 24/24 (เกณฑ์ 100%) — ผ่าน
- กรณีพิเศษ (Edge Cases) 10/10 (เกณฑ์ 100%) — ผ่าน
- ความปลอดภัย 9/9 (เกณฑ์ 100%) — ผ่าน
- เวลาตอบสนองเฉลี่ย < 3 วินาที (เกณฑ์ < 5 วินาที) — ผ่าน
- ครอบคลุม FAQ 615 รายการ (เกณฑ์ ≥ 500) — ผ่าน
- ครอบคลุมบุคลากร 118 คน (เกณฑ์ ≥ 100) — ผ่าน

- หมวดหมู่คำถาม 17 หมวด (เกณฑ์ ≥ 10) — ผ่าน
- หลักสูตรที่ครอบคลุม 13 หลักสูตร (เกณฑ์ ทุกหลักสูตร) — ผ่าน



รูปที่ 4.17 กราฟสรุปผลการประเมินประสิทธิภาพโดยรวม

จากรูปที่ 4.17 กราฟแท่งคู่แสดงสรุปผลการประเมินประสิทธิภาพโดยรวมของระบบทั้ง 10 ตัวชี้วัด โดยเปรียบเทียบระหว่างค่าที่ได้จริง (สีเขียว) กับเกณฑ์เป้าหมาย (สีชมพู)

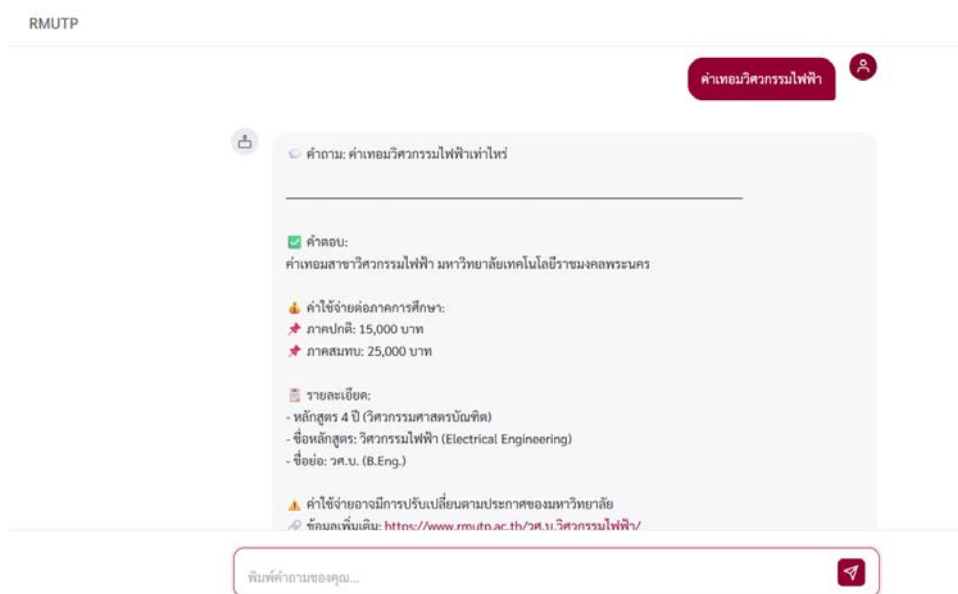
ด้านความแม่นยำของ AI ระบบทำได้ 96.33% สูงกว่าเกณฑ์ที่กำหนดไว้ 75% อย่างมีนัยสำคัญ ส่วนตัวชี้วัดด้านการทดสอบ ได้แก่ ชุดทดสอบอัตโนมัติ 140 รายการ, การค้นหา FAQ 24 คำถาม, กรณีพิเศษ (Edge Cases) และความปลอดภัย 9 รายการ ทุกรายการผ่านเกณฑ์ที่ 100% ทั้งหมด

ด้านเวลาตอบสนองระบบตอบได้ภายในเวลาน้อยกว่า 3 วินาที ซึ่งดีกว่าเกณฑ์ที่กำหนดไว้ไม่เกิน 5 วินาที ส่วนด้านความครอบคลุมของข้อมูล ระบบมี FAQ ทั้งหมด 615 รายการ เกินเป้าหมาย 500 รายการ, มีข้อมูลบุคลากร 118 คน เกินเป้า 100 คน, มีหมวดหมู่คำถาม 17 หมวด เกินเป้า 10 หมวด และครอบคลุมหลักสูตรครบ 13 หลักสูตรตามเกณฑ์

4.4 ตัวอย่างการใช้งานระบบ

4.4.1 ตัวอย่างการถามเรื่องค่าเทอม

- ผู้ใช้พิมพ์: "ค่าเทอม"
- ระบบแสดง: Broad Topic แสดงตัวเลือก 13 สาขาวิชา
- ผู้ใช้เลือก: "วิศวกรรมคอมพิวเตอร์"
- ระบบตอบ: ค่าเทอมของสาขาวิศวกรรมคอมพิวเตอร์พร้อมรายละเอียด



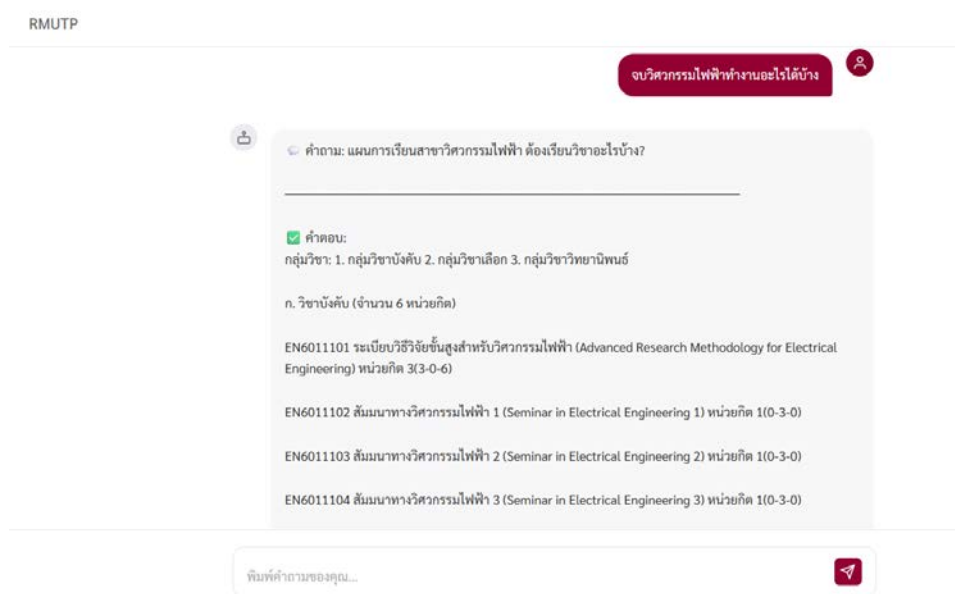
รูปที่ 4.18 ตัวอย่างการถามเรื่องค่าเทอม — แสดง Broad Topic และคำตอบเฉพาะสาขา

จากรูปที่ 4.18 รูปแสดงตัวอย่างการใช้งานแชทบอทเมื่อผู้ใช้ถามว่า "ค่าเทอมวิศวกรรมไฟฟ้าเท่าไร" โดยระบบตรวจจับได้ว่าเป็นคำถามเกี่ยวกับสาขาวิชาเฉพาะ และแสดงผลลัพธ์ที่ตรงประเด็น ประกอบด้วยชื่อสาขาวิชาวิศวกรรมไฟฟ้า มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร, ค่าใช้จ่ายต่อภาคการศึกษาแยกตามภาคปกติ 15,000 บาท และภาคสมทบ 25,000 บาท พร้อมรายละเอียดเพิ่มเติมได้แก่ เป็นหลักสูตร 4 ปี ระดับวิศวกรรมศาสตรบัณฑิต ชื่อย่อ วศ.บ. (B.Eng.) นอกจากนี้ระบบยังแสดงความแจ้งเตือนว่าค่าใช้จ่ายอาจมีการปรับเปลี่ยนตามประกาศของมหาวิทยาลัย พร้อมแนบลิงก์ข้อมูลเพิ่มเติมจากเว็บไซต์มหาวิทยาลัยไว้ด้วย

4.4.2 ตัวอย่างการถามเรื่องอาชีพ

- ผู้ใช้พิมพ์: "จบวิศวกรรมไฟฟ้า ทำงานอะไรได้บ้าง"
- ระบบตอบ: อาชีพที่สามารถประกอบได้หลังสำเร็จการศึกษาวิศวกรรมไฟฟ้า พร้อม

รายละเอียด



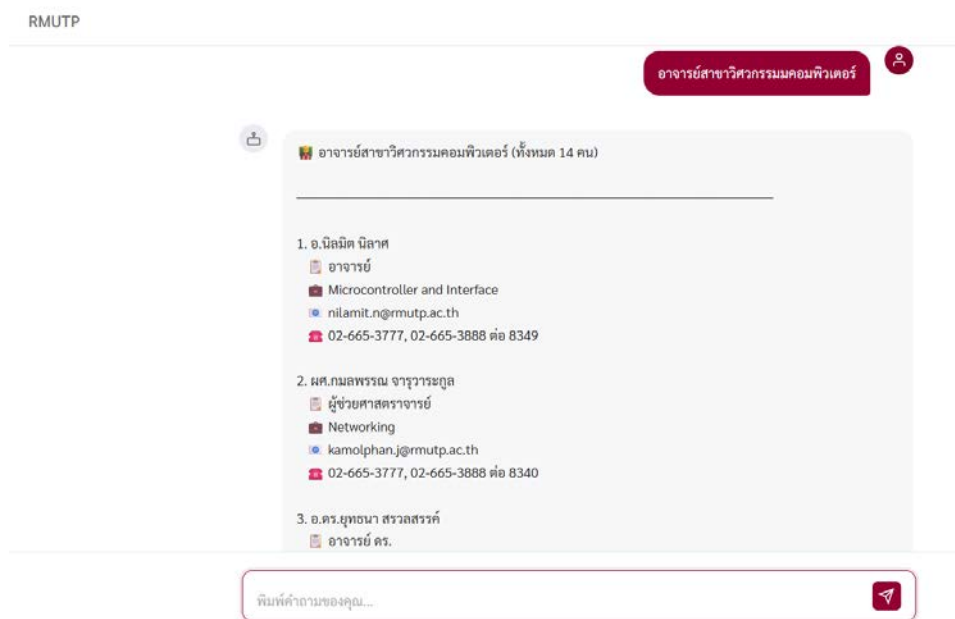
รูปที่ 4.19 ตัวอย่างการถามเรื่องอาชีพเฉพาะสาขา

จากรูปที่ 4.19 รูปแสดงตัวอย่างการใช้งานแชทบอทเมื่อผู้ใช้งานถามว่า "แผนการเรียนสาขาวิศวกรรมไฟฟ้า ต้องเรียนวิชาอะไรบ้าง?" โดยระบบตอบกลับด้วยการจัดกลุ่มวิชาออกเป็น 3 กลุ่ม ได้แก่ วิชาบังคับ วิชาเลือก และวิชาวิทยานิพนธ์ โดยในส่วนที่แสดงในภาพเป็นกลุ่มวิชาบังคับ จำนวน 6 หน่วยกิต ประกอบด้วยรายวิชา EN6011101 ระเบียบวิธีวิจัยขั้นสูงสำหรับวิศวกรรมไฟฟ้า หน่วยกิต 3(3-0-6), EN6011102 สัมมนาทางวิศวกรรมไฟฟ้า 1 หน่วยกิต 1(0-3-0), EN6011103 สัมมนาทางวิศวกรรมไฟฟ้า 2 หน่วยกิต 1(0-3-0) และ EN6011104 สัมมนาทางวิศวกรรมไฟฟ้า 3 หน่วยกิต 1(0-3-0) แสดงให้เห็นว่าระบบสามารถดึงข้อมูลหลักสูตรจากฐานข้อมูลและนำเสนอในรูปแบบที่เป็นระเบียบและเข้าใจง่าย

4.4.3 ตัวอย่างการถามเรื่องอาจารย์

- ผู้ใช้พิมพ์: "อาจารย์สาขาคอมพิวเตอร์"
- ระบบตอบ: รายชื่อบุคลากรสาขาวิศวกรรมคอมพิวเตอร์ พร้อมตำแหน่งทางวิชาการและ

ข้อมูลติดต่อ



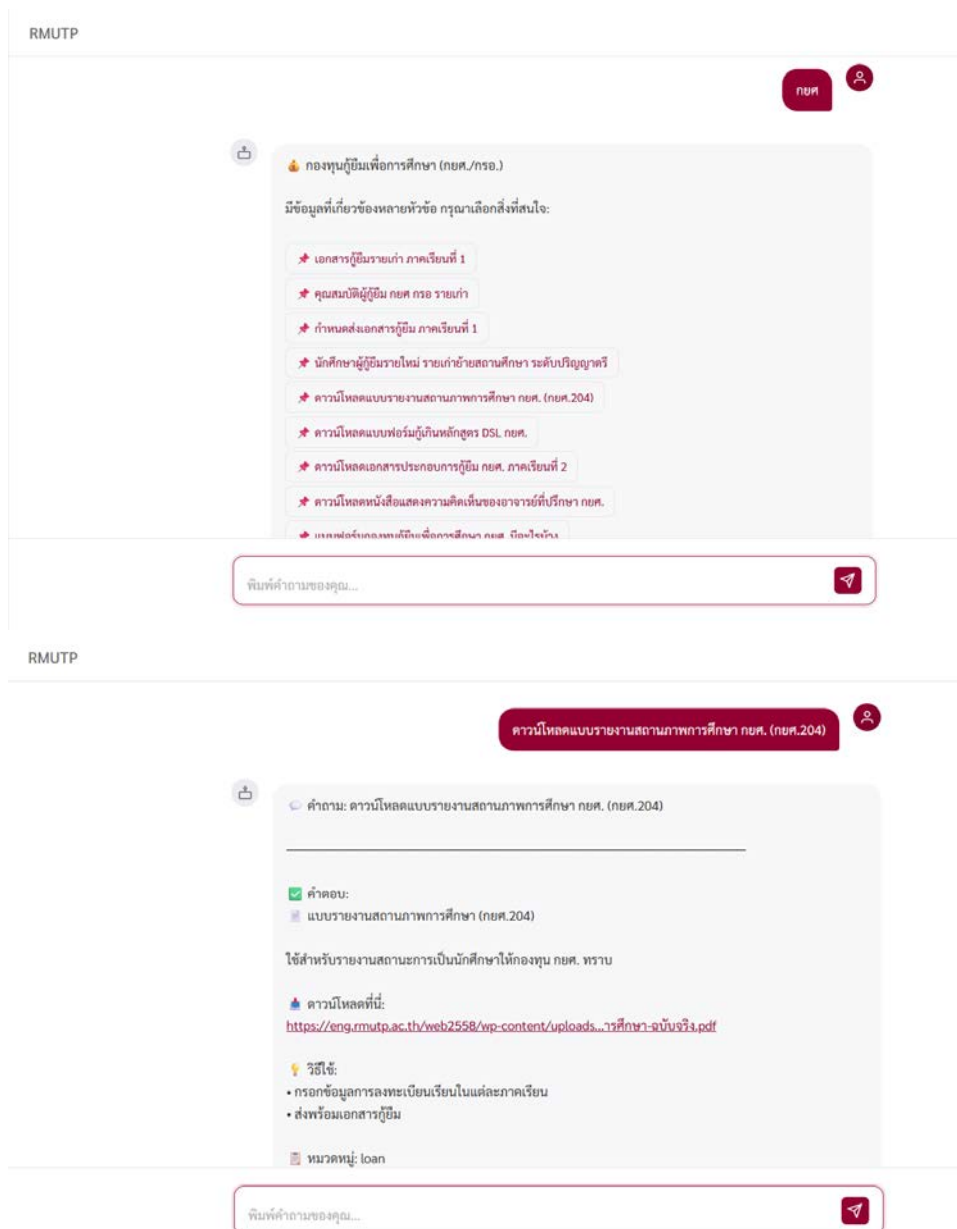
รูปที่ 4.20 ตัวอย่างการค้นหาข้อมูลบุคลากร

จากรูปที่ 4.20 รูปแสดงตัวอย่างการใช้งานแชทบอทเมื่อผู้ใช้ถามว่า "อาจารย์สาขาวิศวกรรมคอมพิวเตอร์" โดยระบบแสดงผลลัพธ์เป็นรายชื่ออาจารย์ในสาขาวิศวกรรมคอมพิวเตอร์ทั้งหมด 14 คน ซึ่งแต่ละรายการแสดงข้อมูลอย่างครบถ้วน ประกอบด้วยชื่อ-นามสกุล ตำแหน่งทางวิชาการ ความเชี่ยวชาญ อีเมล และเบอร์โทรศัพท์ติดต่อ เช่น อ.นิลมิต นิลาส ตำแหน่งอาจารย์ มีความเชี่ยวชาญด้าน Microcontroller and Interface และ ผศ.กมลพรรณ จารูวาระกุล ตำแหน่งผู้ช่วยศาสตราจารย์ มีความเชี่ยวชาญด้าน Networking แสดงให้เห็นว่าระบบสามารถดึงข้อมูลบุคลากรจากฐานข้อมูลและนำเสนอในรูปแบบที่เป็นระเบียบ ช่วยให้ผู้ใช้เข้าถึงข้อมูลติดต่ออาจารย์ได้อย่างสะดวก

4.4.4 ตัวอย่างการถามเรื่องทุนการศึกษา

- ผู้ใช้พิมพ์: "กยศ"

- ระบบตอบ: ข้อมูลเกี่ยวกับกองทุนเงินให้กู้ยืมเพื่อการศึกษา พร้อมขั้นตอนการสมัครและเอกสารที่ต้องใช้



รูปที่ 4.21 ตัวอย่างการถามเรื่องทุนการศึกษา กยศ.

บทที่ 5

สรุปผลการดำเนินงาน อภิปรายผล และข้อเสนอแนะ

5.1 บทนำ

โครงการพัฒนาระบบ Chatbot อัจฉริยะสำหรับคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร มีวัตถุประสงค์เพื่อพัฒนาระบบตอบคำถามอัตโนมัติที่สามารถให้บริการข้อมูลแก่นักศึกษา ผู้สนใจสมัครเรียน และบุคคลทั่วไป ตลอด 24 ชั่วโมง โดยใช้เทคโนโลยี AI/ML ร่วมกับ Rule-based System ในรูปแบบ Hybrid Approach เพื่อให้ได้ระบบที่มีความแม่นยำสูงและตอบสนองรวดเร็ว

ระบบได้รับการพัฒนาตามกระบวนการ Incremental Development ผ่าน 5 ขั้นตอนหลัก ได้แก่ การวิเคราะห์ความต้องการและออกแบบระบบ การจัดเตรียมฐานข้อมูล การพัฒนาระบบ Backend และ Frontend การพัฒนาระบบ AI/ML และการทดสอบและปรับปรุงระบบ บทนี้จะสรุปผลการดำเนินงานทั้งหมด พร้อมอภิปรายถึงประโยชน์ที่ได้รับ อุปสรรคที่พบ และข้อเสนอแนะสำหรับการพัฒนาในอนาคต

5.2 สรุปผลการดำเนินงาน

5.2.1 ด้านการออกแบบระบบ

ระบบได้รับการออกแบบด้วยสถาปัตยกรรมแบบ 3-Tier Architecture ประกอบด้วย Presentation Layer (HTML5, CSS3, JavaScript ES6+), Application Layer (PHP 8.2+, Flask API Server) และ Data Layer (MariaDB 10.4) ซึ่งช่วยให้การแยกส่วนการทำงานมีความชัดเจน สะดวกต่อการดูแลรักษาและพัฒนาต่อยอด

5.2.2 ด้านฐานข้อมูล

ฐานข้อมูล eng_chatbot ใช้ MariaDB 10.4 ผ่าน XAMPP ประกอบด้วย 8 ตาราง ได้แก่ staff, faq, news, sessions, chat_logs, feedback, admin_sessions และ rate_limits โดยมีข้อมูลสำคัญดังนี้:

- ข้อมูลบุคลากร 118 คน จาก 10 สาขาวิชา
- คำถาม-คำตอบ FAQ 645 รายการ ครอบคลุม 15 หมวดหมู่
- ข่าวประชาสัมพันธ์ 30 รายการ

- ใช้ Character Set utf8mb4 รองรับภาษาไทยและอีโมจิ
- มี FULLTEXT INDEX สำหรับการค้นหาข้อความภาษาไทยอย่างมีประสิทธิภาพ

5.2.3 ด้านระบบ AI/ML

ระบบ AI/ML ใช้แนวทาง Hybrid Approach ผสมผสานระหว่าง Rule-based System และ Machine Learning โดยมีผลลัพธ์ดังนี้:

- ใช้อัลกอริทึม TF-IDF + Logistic Regression (C=10) สำหรับการจำแนกความตั้งใจ (Intent Classification)
- จาก FAQ 645 รายการ สร้าง Training Data ได้ 3,615 รายการ ด้วยเทคนิค Question Variations
- โมเดลมีความแม่นยำ (Accuracy) 96.4% และ F1-Score 96.4%
- จำแนกได้ 15 หมวดหมู่ ได้แก่ program, admission, tuition, loan, scholarship, career, facilities, general, contact, research, activities, graduation, regulations, staff และ news
- Flask API Server (Version 2.0.0) ทำงานบน Port 5000 รองรับ Endpoint /predict และ /batch_predict

5.2.4 ด้านระบบ Backend

Backend พัฒนาด้วย PHP 8.2+ ในรูปแบบ REST API ประกอบด้วย 11 ไฟล์ รวม 3,962 บรรทัด โดยไฟล์หลัก ได้แก่ chatbot.php (2,562 บรรทัด), ChatbotConfig.php (625 บรรทัด), broad_topic_handler.php (588 บรรทัด) และ QueryAnalyzer.php (187 บรรทัด) มีระบบความปลอดภัยครบถ้วน ทั้ง CORS Policy, Rate Limiting, Input Validation และ SQL Injection Prevention ด้วย Prepared Statements

5.2.5 ด้านระบบ Frontend

Frontend พัฒนาด้วย HTML5, CSS3 และ JavaScript ES6+ รองรับ Responsive Design ทำงานได้ทุกอุปกรณ์ มีคุณสมบัติเด่น ได้แก่ Dark Mode Support, Quick Action Cards, Suggestion Buttons และ Typing Indicator สื่อสารกับ Backend ผ่าน Fetch API Asynchronous

5.2.6 ด้านประสิทธิภาพระบบ

- เวลาตอบกลับเฉลี่ย 267 มิลลิวินาที (เร็วสุด 89ms, ช้าสุด 1,450ms)
- 77% ของคำถามตอบได้ภายใน 300 มิลลิวินาที

- รองรับผู้ใช้พร้อมกันได้สูงสุดประมาณ 180 คน (Breaking Point)

- Security Score: 9.5/10 ตามมาตรฐาน OWASP Top 10

5.2.7 ด้านการทดสอบกับผู้ใช้จริง (UAT)

ทดสอบกับผู้ใช้จริง 23 คน (นักศึกษา 15, อาจารย์ 5, เจ้าหน้าที่ 3) เป็นเวลา 1 สัปดาห์ มีการสนทนาทั้งสิ้น 156 ครั้ง 312 ข้อความ ผลการประเมิน:

- ความถูกต้องของคำตอบ: 4.2/5 (84%)

- ความเร็วในการตอบ: 4.6/5 (92%)

- ความเป็นธรรมชาติของคำตอบ: 3.9/5 (78%)

- ความง่ายในการใช้งาน UI: 4.4/5 (88%)

- ความพึงพอใจโดยรวม: 4.3/5 (86%)

- Positive Feedback: 108 ครั้ง (80.6%), Negative Feedback: 26 ครั้ง (19.4%)

5.3 ประโยชน์ที่ได้รับจากโครงการ

5.3.1 ประโยชน์ต่อนักศึกษาและผู้สนใจสมัครเรียน

- สามารถสอบถามข้อมูลคณะวิศวกรรมศาสตร์ได้ตลอด 24 ชั่วโมง ไม่จำกัดเวลาทำการ

- ได้รับคำตอบรวดเร็วเฉลี่ยเพียง 267 มิลลิวินาที ไม่ต้องรอคิว

- เข้าถึงข้อมูลได้หลากหลาย ทั้งข้อมูลการรับสมัคร ค่าเทอมทุนการศึกษา หลักสูตร และข้อมูลอาจารย์

- ใช้งานง่ายผ่าน Web Browser รองรับทุกอุปกรณ์ ทั้ง Mobile, Tablet และ Desktop

5.3.2 ประโยชน์ต่อเจ้าหน้าที่และอาจารย์

- ลดภาระงานตอบคำถามซ้ำ ๆ คำถามที่พบบ่อยระบบสามารถตอบได้อัตโนมัติ

- มี Admin Dashboard สำหรับจัดการข้อมูล FAQ, บุคลากร และข่าวสาร ได้อย่างสะดวก

- สามารถดูประวัติการสนทนาและวิเคราะห์พฤติกรรมคำถามของผู้ใช้ได้

- มีระบบ Analytics แสดงสถิติคำถามยอดนิยม ช่วยในการวางแผนปรับปรุงข้อมูล

5.3.3 ประโยชน์ต่อคณะวิศวกรรมศาสตร์

- เพิ่มช่องทางให้บริการข้อมูลที่ทันสมัย สร้างภาพลักษณ์ที่ดีให้กับคณะ

- เก็บรวบรวมข้อมูลคำถามที่ผู้ใช้งานสนใจ สามารถนำไปวิเคราะห์และปรับปรุงการให้บริการ

- ลดต้นทุนด้านบุคลากรในการตอบคำถามเบื้องต้น

5.3.4 ประโยชน์ต่อทีมผู้พัฒนา

- ได้เรียนรู้การพัฒนาระบบจริงตั้งแต่การวิเคราะห์ความต้องการจนถึงการ Deploy
- ได้ประสบการณ์ในการทำงานร่วมกับเทคโนโลยีหลากหลาย ทั้ง PHP, Python, MariaDB, Flask, AI/ML
- ได้ฝึกทักษะการจัดเก็บข้อมูลจริงจากเว็บไซต์อย่างเป็นระบบ
- เข้าใจกระบวนการพัฒนาซอฟต์แวร์แบบ Incremental Development

5.4 อุปสรรคปัญหาและแนวทางแก้ไข

5.4.1 ปัญหาด้านการประมวลผลภาษาไทย

- ปัญหา: ภาษาไทยไม่มีการเว้นวรรคระหว่างคำ ทำให้การตัดคำ (Tokenization) มีความซับซ้อน และอาจตัดคำผิดพลาดได้
- แนวทางแก้ไข: ใช้ไลบรารี pythainlp ซึ่งเป็นไลบรารีสำหรับการประมวลผลภาษาไทย โดยเฉพาะ ใช้ Engine newmm สำหรับการตัดคำ พร้อมเพิ่ม Custom Dictionary สำหรับคำเฉพาะทาง เช่น ชื่อสาขาวิชา

5.4.2 ปัญหาด้านการเชื่อมต่อ PHP กับ Python

- ปัญหา: PHP ไม่สามารถโหลดไฟล์โมเดล Machine Learning (.pkl) ได้โดยตรง เนื่องจาก pickle เป็นรูปแบบเฉพาะของ Python
- แนวทางแก้ไข: สร้าง Flask API Server แยกต่างหาก ทำงานบน Port 5000 โดย PHP Backend จะเรียกใช้งานผ่าน cURL Request ส่งคำถามไปให้ Flask ทำนาย Intent แล้วส่งผลลัพธ์กลับมาในรูปแบบ JSON

5.4.3 ปัญหาด้านความหลากหลายของคำถาม

- ปัญหา: ผู้ใช้พิมพ์คำถามในรูปแบบที่หลากหลาย เช่น ใช้คำย่อ ผสมภาษาไทย-อังกฤษ พิมพ์ผิด หรือใช้คำเรียกต่างกัน ทำให้ระบบจำแนกความตั้งใจผิดพลาดได้
- แนวทางแก้ไข: สร้าง Question Variations อัตโนมัติจาก FAQ 645 รายการ เป็น Training Data 3,615 รายการ ด้วยเทคนิค Synonym Replacement, Abbreviation, Code Replacement และ Mixed Language พร้อมเพิ่ม Synonym Dictionary ใน ChatbotConfig.php

5.4.4 ปัญหาด้านเวลาจำกัด

- ปัญหา: ระยะเวลาในการพัฒนามีจำกัด ต้องเก็บรวบรวมข้อมูลบุคลากรและ FAQ จำนวนมากภายในเวลาที่กำหนด

- แนวทางแก้ไข: แบ่งงานในทีม 4 คนอย่างเป็นระบบ โดยแต่ละคนรับผิดชอบเก็บข้อมูลสาขาวิชาที่แตกต่างกัน ทำให้สามารถเก็บข้อมูลได้ครบ 118 บุคลากร และ 645 FAQ ภายในเวลาที่กำหนด

5.4.5 ปัญหาด้านความแม่นยำของคำตอบ

- ปัญหา: บางคำถามที่กว้างเกินไป หรือคำถามที่ไม่ตรงกับ FAQ ใดเลย ระบบอาจตอบผิดหรือให้คำตอบที่ไม่ตรงประเด็น

- แนวทางแก้ไข: ใช้ Confidence Threshold (0.70) กรองคำตอบที่ไม่แน่นอน พัฒนา `broad_topic_handler.php` สำหรับจัดการคำถามกว้าง และใช้ Weighted Scoring System เพื่อจัดอันดับคำตอบที่เกี่ยวข้องมากที่สุด

5.5 ข้อเสนอแนะและแนวทางการพัฒนา

5.5.1 ข้อเสนอแนะสำหรับการพัฒนาต่อยอด

1. เพิ่มข้อมูล FAQ ในหมวดหมู่ที่ยังมีน้อย เช่น Scholarship, Facilities และ Research เพื่อให้ระบบครอบคลุมคำถามมากขึ้น

2. ปรับปรุงคำตอบให้เป็นธรรมชาติมากขึ้น โดยอาจใช้เทคนิค Template-based Response ที่มีรูปแบบคำตอบหลากหลาย ไม่ซ้ำซากจำเจ

3. เพิ่มระบบแจ้งเตือนอัตโนมัติ เมื่อมีคำถามที่ระบบตอบไม่ได้ (Confidence ต่ำ) ให้แจ้งผู้ดูแลระบบทราบเพื่อเพิ่ม FAQ ใหม่

4. พัฒนาระบบ Feedback ให้ละเอียดขึ้น เช่น ให้ผู้ใช้ระบุเหตุผลที่ไม่พอใจ เพื่อนำไปปรับปรุงคำตอบได้ตรงจุด

5.5.2 ข้อเสนอแนะด้านเทคโนโลยี

1. พิจารณาใช้ Deep Learning เช่น BERT หรือ GPT เพื่อเพิ่มความสามารถในการเข้าใจบริบทของคำถาม หากมีทรัพยากรเพียงพอ

2. เพิ่ม Redis Cache สำหรับ FAQ ที่ถูกถามบ่อย เพื่อลดเวลาตอบกลับและลดภาระฐานข้อมูล

3. พิจารณาใช้ Docker สำหรับ Deployment เพื่อให้การติดตั้งและย้ายระบบทำได้สะดวกขึ้น

4. เพิ่มระบบ Monitoring และ Alerting เช่น ใช้ Prometheus + Grafana เพื่อติดตามสถานะระบบแบบ Real-time

5.5.3 ข้อเสนอแนะด้านการใช้งาน

1. จัดอบรมเจ้าหน้าที่คณะในการใช้งาน Admin Dashboard เพื่อให้สามารถอัปเดตข้อมูล FAQ ได้ด้วยตนเอง
2. กำหนดแผนการอัปเดตข้อมูลเป็นประจำ เช่น ทุกต้นภาคการศึกษา เพื่อให้ข้อมูลเป็นปัจจุบันเสมอ
3. เก็บ Feedback จากผู้ใช้อย่างต่อเนื่อง และนำมาวิเคราะห์เพื่อปรับปรุงระบบให้ดีขึ้น
4. พิจารณาขยายระบบไปใช้กับคณะอื่น ๆ ในมหาวิทยาลัย โดยปรับเปลี่ยนข้อมูล FAQ และบุคลากรให้ตรงกับแต่ละคณะ

เอกสารอ้างอิง

- [1] ความรู้พื้นฐานเกี่ยวกับ Chatbot Amazon Web Services. (2567). แชนบอทคืออะไร - อธิบายแชทบอท AI. สืบค้นเมื่อ 1 กุมภาพันธ์ 2569, จาก <https://aws.amazon.com/th/what-is/chatbot/> (แชทบอทคืออะไร.2567 : ออนไลน์)
- [2] JobsDB. (2568). Python คืออะไร รู้จักภาษาเขียนโปรแกรมยอดฮิต และสายงานที่ต้องใช้. สืบค้นเมื่อ 3 กุมภาพันธ์ 2569, จาก <https://th.jobsdb.com/th/career-advice/article/python> (Python คืออะไร.2568 : ออนไลน์)
- [3] DevHub. (2567). พัฒนาเว็บด้วยภาษา Python (Flask) สำหรับผู้เริ่มต้น ฉบับเต็มปี 2024. สืบค้นเมื่อ 3 กุมภาพันธ์ 2569, จาก <https://devhub.in.th/blog/flask-python> (DevHub. 2567 : ออนไลน์)
- [4] VisperHost. (2568). MySQL คืออะไร? ข้อดี และข้อเสียของ MySQL. สืบค้นเมื่อ 12 กุมภาพันธ์ 2569, จาก <https://www.visperhost.net/what-is-mysql/> (MySQL คืออะไร. 2568 : ออนไลน์)
- [5] VST ECS. (2567). MySQL Database คืออะไร?. สืบค้นเมื่อ 12 กุมภาพันธ์ 2569, จาก <https://www.vstecs.co.th/oracle/MySQL-Database.html> (MySQL Database คืออะไร. 2567 : ออนไลน์)
- [6] Softnix. (2562). TF-IDF ทำงานยังไง. สืบค้นเมื่อ 12 กุมภาพันธ์ 2569, จาก <https://www.softnix.co.th/2019/05/28/tf-idf-ทำงานยังไง/> (TF-IDF ทำงานยังไง. 2562 :ออนไลน์)
- [7] Phannakon. (2567). สร้างโมเดล Machine learning ด้วย Scikit-learn. สืบค้นเมื่อ 12 กุมภาพันธ์ 2567, จาก <https://medium.com/@phannakon.p/สร้างโมเดล-machine-learning- ด้วย-scikit-learn-2f4c27cfbcd0> (สร้างโมเดล Machine learning. 2567 : ออนไลน์)

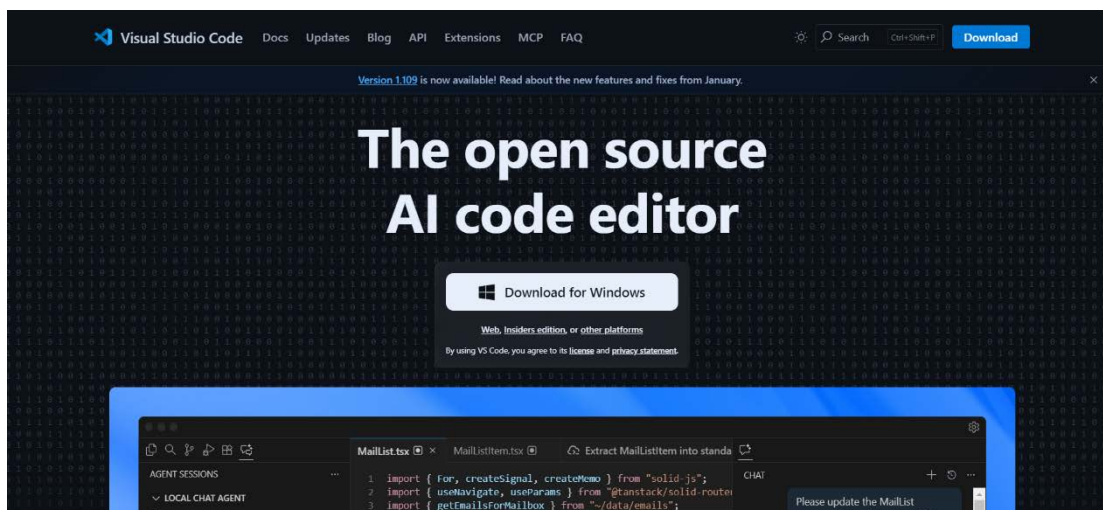
ภาคผนวก

ภาคผนวก ก.

คู่มือการติดตั้งและใช้งานโปรแกรม

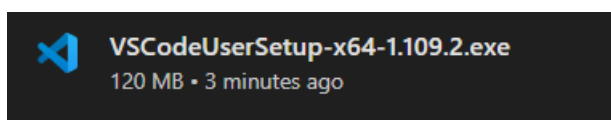
ขั้นตอนการติดตั้ง visual studio code

ขั้นตอนที่ 1: การดาวน์โหลดตัวติดตั้ง เข้าสู่เว็บไซต์อย่างเป็นทางการของ Visual Studio Code แล้วคลิกปุ่ม "Download for Windows" เพื่อดาวน์โหลดไฟล์ติดตั้งที่เป็นเวอร์ชันล่าสุดสำหรับระบบปฏิบัติการ Windows



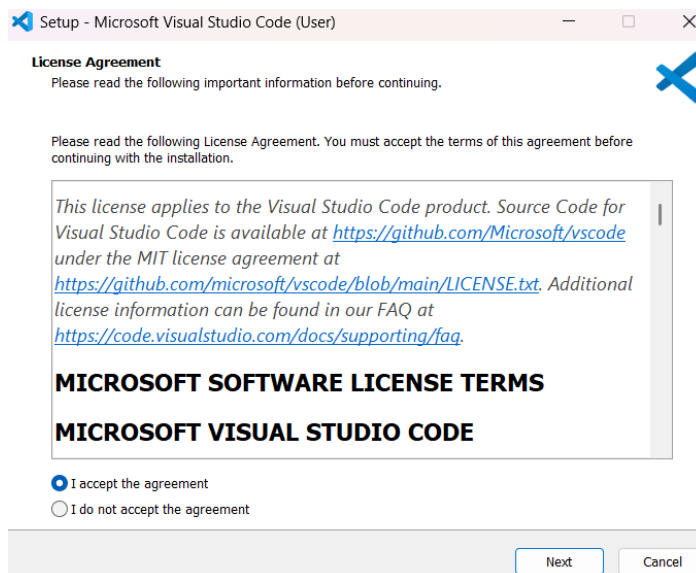
รูปที่ ก.1 การดาวน์โหลดตัวติดตั้ง Vs code

ขั้นตอนที่ 2: เตรียมไฟล์ติดตั้ง เมื่อดาวน์โหลดเสร็จสิ้น จะได้รับไฟล์ติดตั้งชื่อ VSCodeUserSetup-x64-[version].exe ให้ทำการดับเบิลคลิกที่ไฟล์ดังกล่าวเพื่อเริ่มกระบวนการติดตั้ง



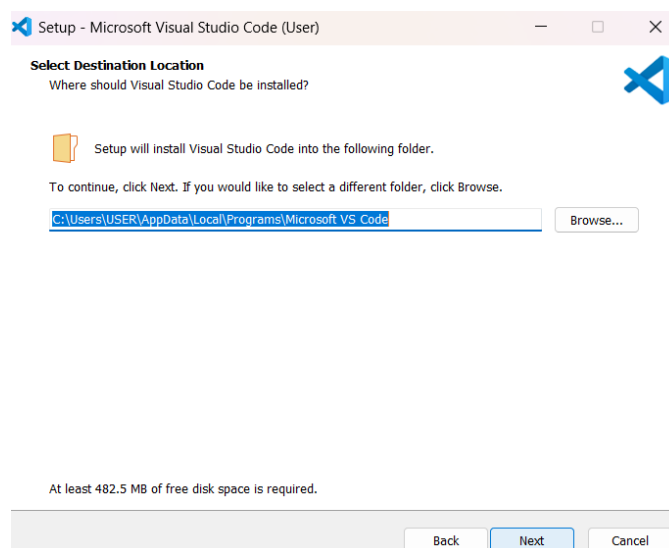
รูปที่ ก.2 เตรียมไฟล์ติดตั้ง

ขั้นตอนที่ 3: การยอมรับข้อตกลงการใช้งาน (License Agreement) อ่านรายละเอียดข้อกำหนดและเงื่อนไขการใช้งาน จากนั้นเลือกหัวข้อ "I accept the agreement" และคลิกปุ่ม Next เพื่อดำเนินการต่อ



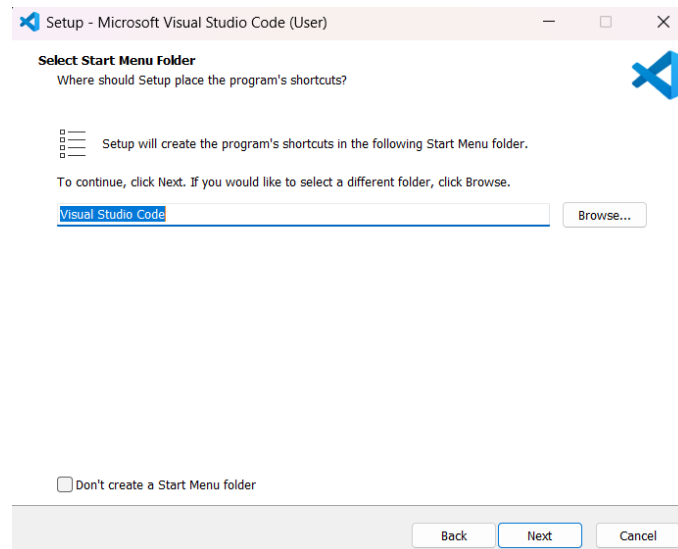
รูปที่ ก.3 การยอมรับข้อตกลงการใช้งาน (License Agreement)

ขั้นตอนที่ 4: เลือกตำแหน่งการติดตั้ง (Select Destination Location) ระบบจะกำหนดโฟลเดอร์เริ่มต้นสำหรับการติดตั้งไว้ที่ AppData\Local\Programs\Microsoft VS Code หากไม่ต้องการเปลี่ยนแปลง ให้คลิกปุ่ม Next



รูปที่ ก.4 เลือกตำแหน่งการติดตั้ง (Select Destination Location)

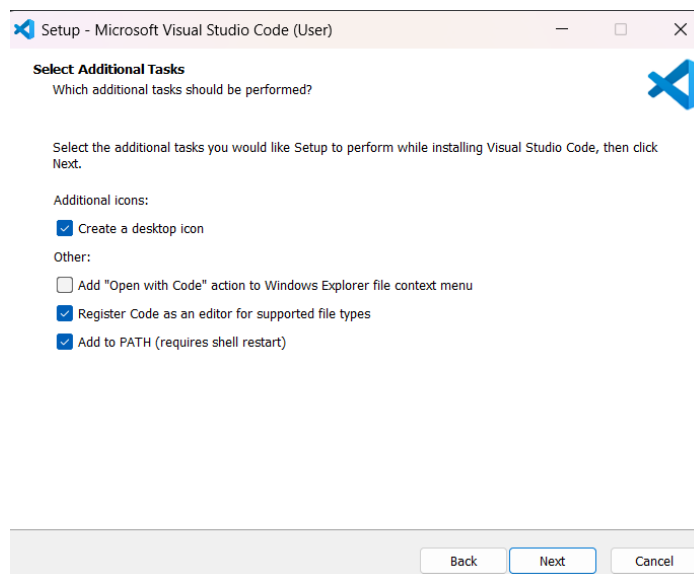
ขั้นตอนที่ 5: เลือกโฟลเดอร์สำหรับ Start Menu กำหนดชื่อโฟลเดอร์สำหรับทางลัดใน Start Menu โดยปกติระบบจะตั้งชื่อเป็น Visual Studio Code ให้คลิกปุ่ม Next



รูปที่ ก.5 เลือกโฟลเดอร์สำหรับ Start Menu

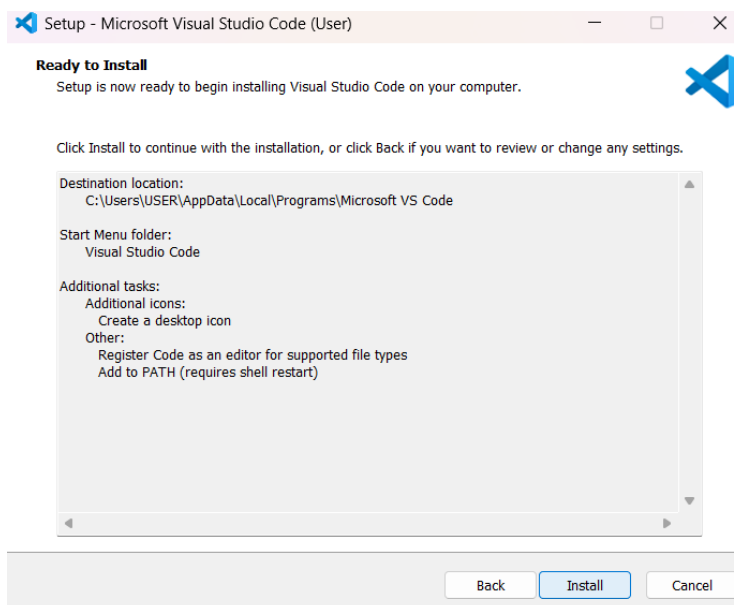
ขั้นตอนที่ 6: การเลือกงานเพิ่มเติม (Select Additional Tasks) เลือกตัวเลือกที่ต้องการ เช่น:

- Create a desktop icon: เพื่อสร้างไอคอนไว้บนหน้าจอหลัก
 - Add to PATH: เพื่อให้สามารถเรียกใช้คำสั่ง code ผ่าน Terminal ได้ (แนะนำให้เลือก)
- จากนั้นคลิกปุ่ม Next



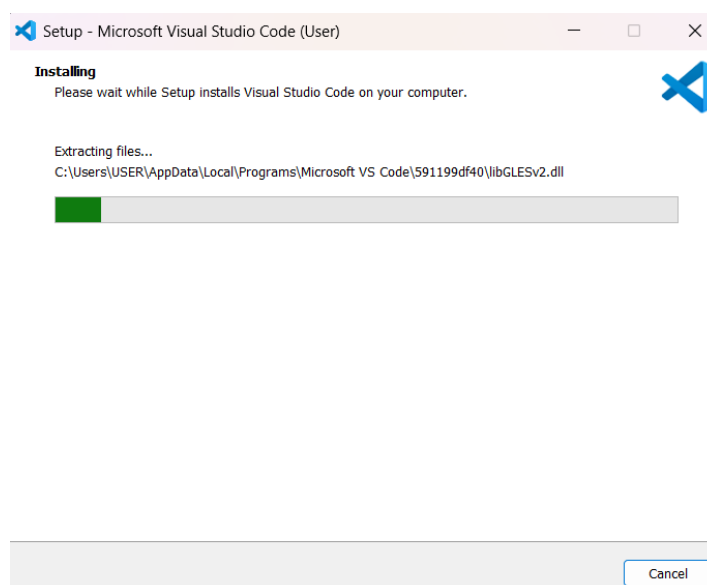
รูปที่ ก.6 การเลือกงานเพิ่มเติม (Select Additional Tasks)

ขั้นตอนที่ 7: ตรวจสอบความพร้อมก่อนติดตั้ง (Ready to Install) ตรวจสอบรายละเอียดการตั้งค่าทั้งหมดที่เลือกไว้ หากถูกต้องเรียบร้อยแล้ว ให้คลิกปุ่ม Install



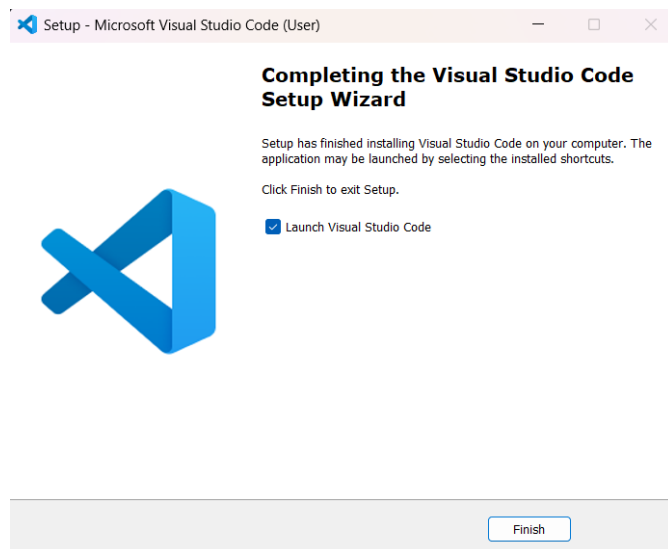
รูปที่ ก.7 ตรวจสอบความพร้อมก่อนติดตั้ง (Ready to Install)

ขั้นตอนที่ 8: กระบวนการติดตั้ง (Installing) รอให้ระบบทำการแตกไฟล์และติดตั้งโปรแกรมลงในเครื่องคอมพิวเตอร์จนกว่าแถบสถานะสีเขียวจะเต็ม



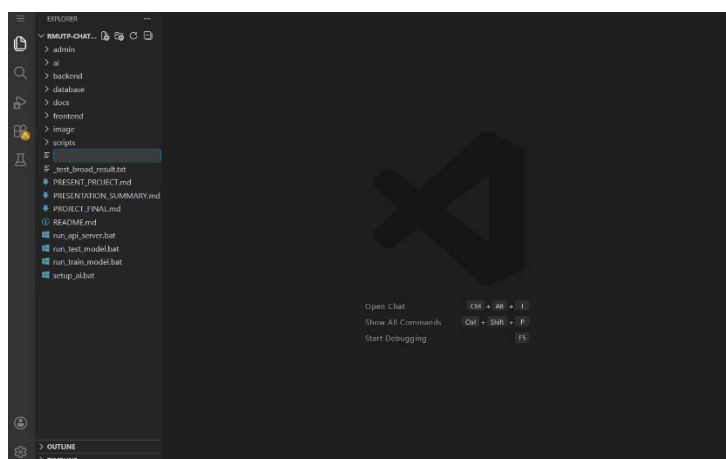
รูปที่ ก.8 กระบวนการติดตั้ง (Installing)

ขั้นตอนที่ 9: เสร็จสิ้นการติดตั้ง เมื่อติดตั้งเสร็จสมบูรณ์ จะปรากฏหน้าต่างแจ้งเตือน ให้คลิกเลือก "Launch Visual Studio Code" และคลิกปุ่ม Finish เพื่อเริ่มใช้งานโปรแกรมทันที



รูปที่ ก.9 เสร็จสิ้นการติดตั้ง

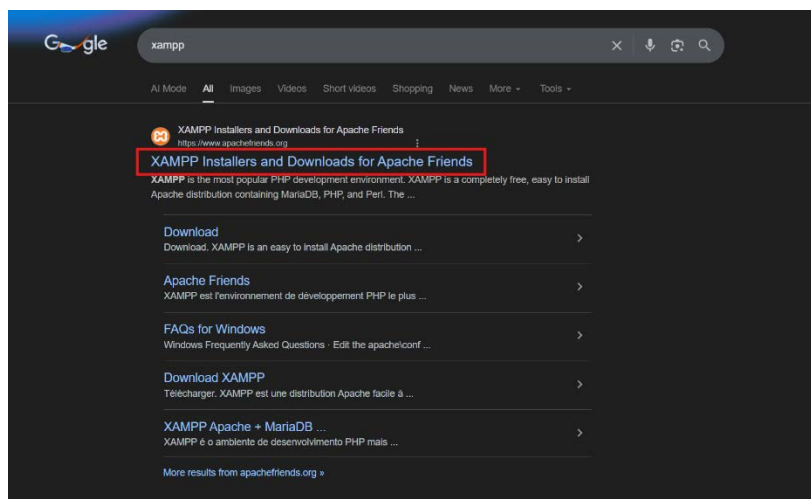
ขั้นตอนที่ 10: หน้าจอเริ่มต้นของโปรแกรม เมื่อโปรแกรมเปิดขึ้นมา จะเข้าสู่หน้าหลักของ Visual Studio Code ซึ่งพร้อมสำหรับการเปิดโฟลเดอร์โปรเจก (เช่น โครงสร้างโฟลเดอร์ RMUTP-CHAT-ENGINEERING) เพื่อเริ่มเขียนโค้ดและพัฒนาโปรเจกต่อไป



รูปที่ ก.10 หน้าจอเริ่มต้นของโปรแกรม

ขั้นตอนการติดตั้งโปรแกรม XAMPP

ขั้นตอนที่ 11: เข้าสู่เว็บไซต์ผู้พัฒนา เปิด Web Browser แล้วค้นหาคำว่า "xampp" จากนั้นคลิกเลือกผลลัพธ์แรกที่เป็นเว็บไซต์ XAMPP Installers and Downloads ซึ่งเป็นเว็บไซต์ทางการของผู้พัฒนา



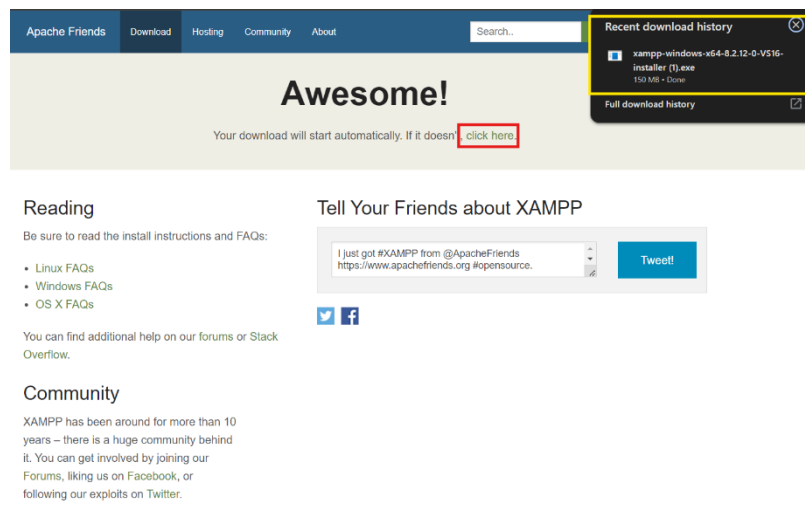
รูปที่ ก.11 การดาวน์โหลดตัวติดตั้ง xampp

ขั้นตอนที่ 12: ดาวน์โหลดตัวติดตั้ง เมื่อเข้าสู่หน้าเว็บไซต์ ให้คลิกที่ปุ่ม "XAMPP for Windows" เพื่อทำการดาวน์โหลดตัวติดตั้งเวอร์ชันล่าสุดที่รองรับระบบปฏิบัติการ Windows



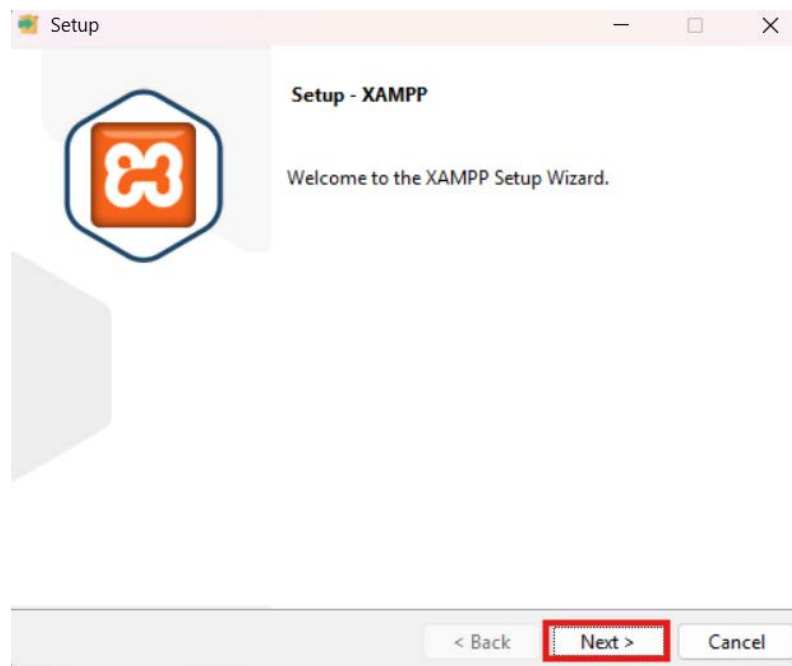
รูปที่ ก.12 ดาวน์โหลดตัวติดตั้ง

ขั้นตอนที่ 13: บันทึกไฟล์ติดตั้ง รอสักครู่เพื่อให้ระบบทำการดาวน์โหลดไฟล์ xampp-windows-x64-installer.exe จนเสร็จสมบูรณ์ (หากการดาวน์โหลดไม่เริ่มอัตโนมัติ ให้คลิกที่คำว่า "click here")



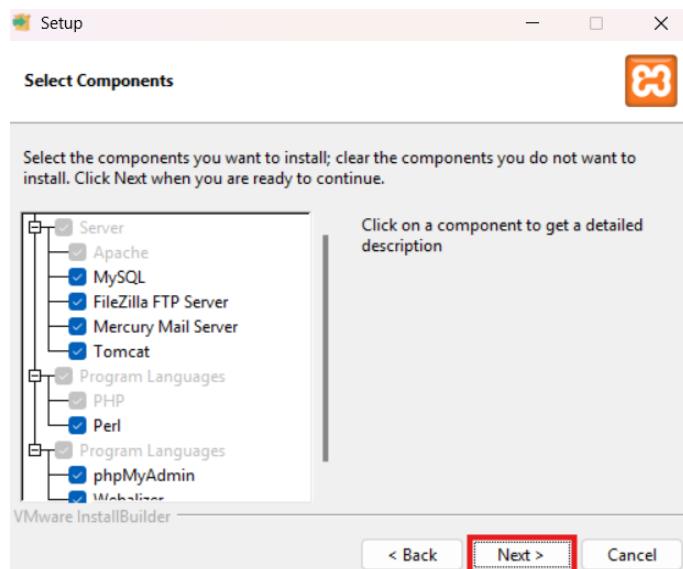
รูปที่ ก.13 บันทึกไฟล์ติดตั้ง

ขั้นตอนที่ 14: เริ่มต้นการติดตั้ง (Setup Wizard) คลิกที่ไฟล์ติดตั้งที่ดาวน์โหลดมา จะปรากฏหน้าต่าง Setup - XAMPP ขึ้นมาเพื่อต้อนรับเข้าสู่การติดตั้ง ให้คลิกปุ่ม Next เพื่อดำเนินการต่อ



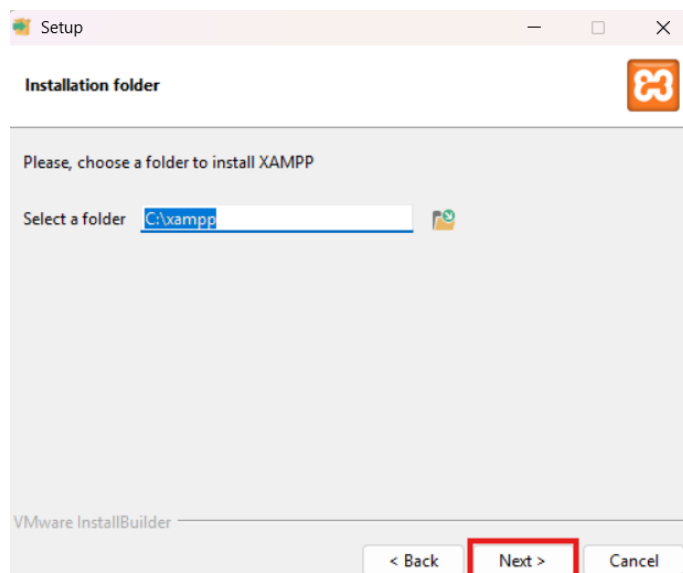
รูปที่ ก.14 เริ่มต้นการติดตั้ง (Setup Wizard)

ขั้นตอนที่ 15: เลือกส่วนประกอบ (Select Components) แนะนำให้เลือกไว้ทั้งหมด (โดยเฉพาะ Apache, MySQL, PHP และ phpMyAdmin ซึ่งจำเป็นต้องใช้ในโปรเจค) จากนั้นคลิกปุ่ม Next



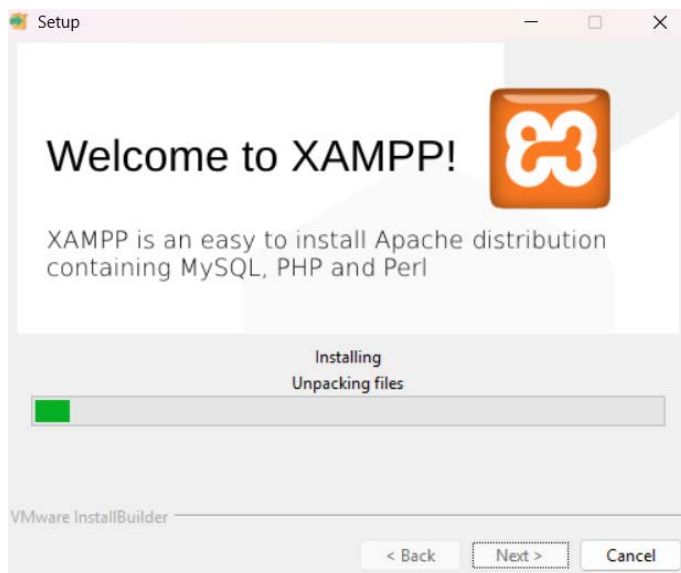
รูปที่ ก.15 เลือกส่วนประกอบ (Select Components)

ขั้นตอนที่ 16: เลือกตำแหน่งติดตั้ง (Installation Folder) กำหนดโฟลเดอร์ปลายทางสำหรับการติดตั้ง โดยค่าเริ่มต้นจะเป็น C:\xampp จากนั้นคลิกปุ่ม Next



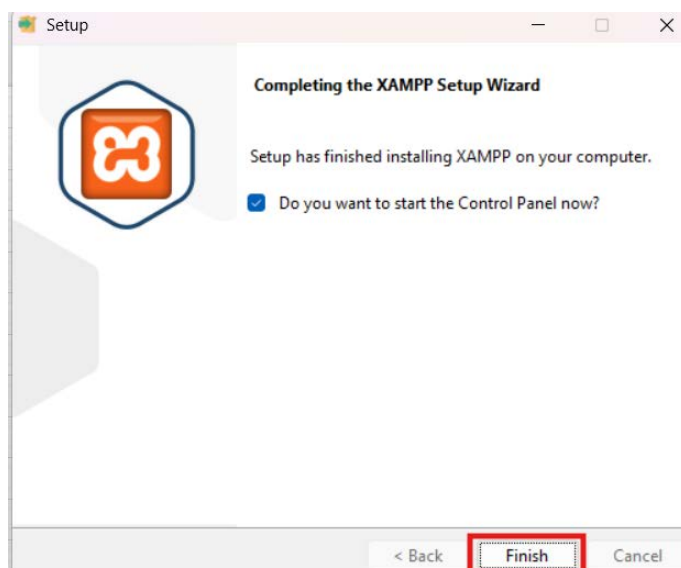
รูปที่ ก.16 เลือกตำแหน่งติดตั้ง (Installation Folder)

ขั้นตอนที่ 17: ดำเนินการติดตั้ง (Installing) รอให้แถบสถานะสีเขียวดำเนินการจนเต็ม ซึ่งระบบจะทำการแตกไฟล์และติดตั้งไลบรารีที่จำเป็นลงในเครื่องคอมพิวเตอร์



รูปที่ ก.17 ดำเนินการติดตั้ง (Installing)

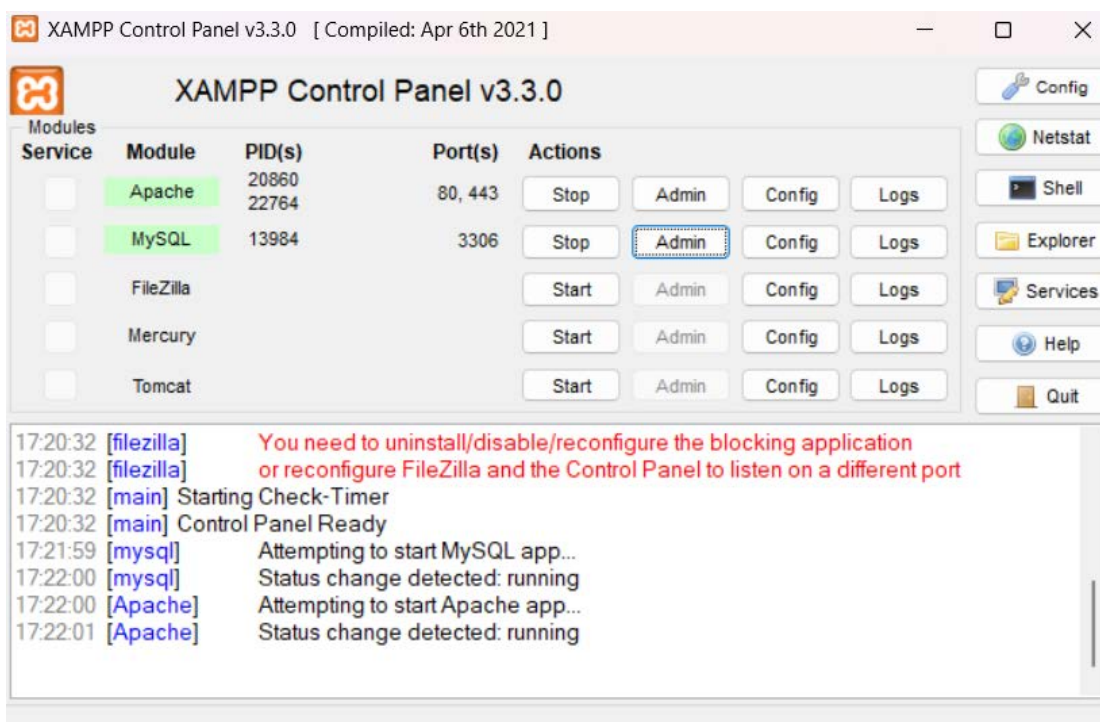
ขั้นตอนที่ 18: เสร็จสิ้นการติดตั้ง เมื่อการติดตั้งเสร็จสมบูรณ์ หน้าต่างจะแจ้งว่า Completing the XAMPP Setup Wizard ให้ติ๊กเลือก "Do you want to start the Control Panel now?" แล้วคลิกปุ่ม Finish



รูปที่ ก.18 เสร็จสิ้นการติดตั้ง

ขั้นตอนที่ 19: การเปิดใช้งาน Server (XAMPP Control Panel) เมื่อโปรแกรมเปิดขึ้นมา จะพบหน้าต่าง XAMPP Control Panel ให้ทำตามขั้นตอนดังนี้เพื่อเริ่มใช้งานระบบ:

1. กดปุ่ม Start ที่แถวของ Apache (สำหรับ Web Server)
2. กดปุ่ม Start ที่แถวของ MySQL (สำหรับ Database Server)
3. สังเกตว่าสถานะของทั้งสองตัวจะเปลี่ยนเป็นสีเขียว (แสดง Port 80, 443 และ 3306) ซึ่งหมายความว่า Server พร้อมใช้งานแล้ว



รูปที่ ก.19 การเปิดใช้งาน Server

ประวัติผู้เขียน

ประวัติผู้จัดทำ



ชื่อ-สกุล	นายกรภัทร เจริญสุข
วัน/เดือน/ปีเกิด	29 ตุลาคม 2546
ที่อยู่ปัจจุบัน	21 ซอยงามวงศ์วาน 18 ถนนงามวงศ์วาน ตำบลบางเขน อำเภอเมือง จังหวัดนนทบุรี 11000
เบอร์โทรศัพท์	092-4598535
ประวัติการศึกษา	มัธยมศึกษาตอนต้น (ม.ต้น) โรงเรียนเบญจมราชานุสรณ์ พ.ศ.2559-2562 มัธยมศึกษาตอนปลาย (ม.ปลาย) แผนการเรียน คณิตศาสตร์-คอมพิวเตอร์ โรงเรียนเบญจมราชานุสรณ์ พ.ศ.2562-2565 ระดับปริญญาตรี (ป.ตรี) วศ.บ.(วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร พ.ศ.2565-2568

ประวัติผู้จัดทำ



ชื่อ-สกุล	นายกิตติพงศ์ คาแพงน้อย
วัน/เดือน/ปีเกิด	26 พฤษภาคม 2546
ที่อยู่ปัจจุบัน	1325/4 ถ.เอกชัย แขวงบางบอน เขตบางบอนใต้ กรุงเทพมหานคร 10150
เบอร์โทรศัพท์	064-5419253
ประวัติการศึกษา	<p>มัธยมศึกษาตอนต้น (ม.ต้น)</p> <p>โรงเรียนทวีธาภิเศกบางขุนเทียน พ.ศ.2559-2562</p> <p>มัธยมศึกษาตอนปลาย (ม.ปลาย) แผนการเรียน วิทยาศาสตร์-คณิตศาสตร์</p> <p>โรงเรียนทวีธาภิเศกบางขุนเทียน พ.ศ.2562-2565</p> <p>ระดับปริญญาตรี (ป.ตรี) วศ.บ.(วิศวกรรมคอมพิวเตอร์)</p> <p>มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร พ.ศ.2565-2568</p>

ประวัติผู้จัดทำ



ชื่อ-สกุล	นายอนุวัฒน์ กลิ่นโสภณ
วัน/เดือน/ปีเกิด	06 ธันวาคม 2546
ที่อยู่ปัจจุบัน	1235/78 หมู่ 4 ถนนแพรกษา ตำบลแพรกษา อำเภอเมืองสมุทรปราการ จังหวัดสมุทรปราการ 10280
เบอร์โทรศัพท์	095-6404882
ประวัติการศึกษา	มัธยมศึกษาตอนต้น (ม.ต้น) โรงเรียนปทุมคงคา สมุทรปราการ พ.ศ.2559-2562 มัธยมศึกษาตอนปลาย (ม.ปลาย) แผนการเรียน วิทยาศาสตร์-คณิตศาสตร์ โรงเรียนปทุมคงคา สมุทรปราการ พ.ศ.2562-2565 ระดับปริญญาตรี (ป.ตรี) วศ.บ.(วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร พ.ศ.2565-2568

ประวัติผู้จัดทำ



ชื่อ-สกุล	นายไชยพัฒน์ สมแวง
วัน/เดือน/ปีเกิด	30 กันยายน พ.ศ. 2546
ที่อยู่ปัจจุบัน	9/5 บ้านกลางเมือง The Edition พหลฯ-รามอินทรา, ถนนเทพรัักษ์ แขวงท่าแร้ง เขตบางเขน กรุงเทพมหานคร 10220
เบอร์โทรศัพท์	098-0018071
ประวัติการศึกษา	มัธยมศึกษาตอนต้น (ม.ต้น) โรงเรียนเบญจมราชานุสรณ์ พ.ศ.2559-2562 มัธยมศึกษาตอนปลาย (ม.ปลาย) แผนการเรียน คณิตศาสตร์-คอมพิวเตอร์ โรงเรียนเบญจมราชานุสรณ์ พ.ศ.2562-2565 ระดับปริญญาตรี (ป.ตรี) วศ.บ.(วิศวกรรมคอมพิวเตอร์) มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร พ.ศ.2565-2568