

# Week 5: Agentic Workflow & LLM Integration

Building AI Travel Planner Bot with n8n

วิชา n8n Workflow Automation

ภาคการศึกษาที่ 2/2568

# สารบัญ

- 1 Introduction & Recap
- 2 Agentic Workflow
- 3 n8n LLM Integration
- 4 AI Extract Info
- 5 Prompt Engineering
- 6 APIs: Weather & Places
- 7 Data Flow & Architecture
- 8 Discord Embed Design
- 9 bot.py: ส่งข้อความดิบ
- 10 Webhook: test vs production
- 11 API Key Security
- 12 Summary

# Recap: สิ่งที่เราเรียนมาแล้ว

## Lab 01-03: n8n Basic Workflow

- Workflow, Node, Connection, JSON Data Flow
- HTTP Request Node (GET & POST)
- External API Integration

## Lab 04: Discord Earthquake Alert Bot

- Webhook Node (Chat Command Bot)
- USGS Earthquake API (GeoJSON)
- Discord Webhook URL + Bot Token + bot.py
- Code Node (Discord Embed)

# วันนี้จะเรียนอะไร?

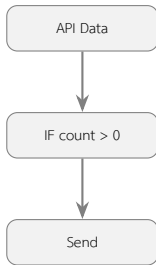
## Agentic Workflow — ใช้ AI เป็นตัวตัดสินใจ

### หัวข้อที่จะเรียน:

- 1 Agentic Workflow vs Rule-based Workflow
- 2 n8n Basic LLM Chain + OpenRouter Chat Model
- 3 AI Extract Info — ให้ AI แยกข้อมูลจากข้อความอิสระ
- 4 Prompt Engineering (System & User Messages)
- 5 Multi-source Data Aggregation
- 6 OpenWeatherMap API + Geoapify Places API
- 7 สร้าง AI Travel Planner Bot

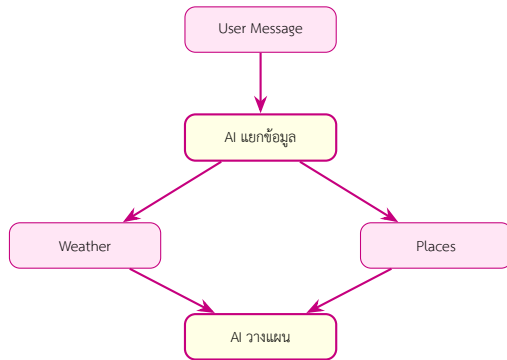
# Rule-based vs Agentic Workflow

## Rule-based (Lab 04):



คนเขียน logic ทั้งหมด

## Agentic (Lab 05):



AI รับข้อความ → แยกข้อมูล + คิดเอง

Agentic = AI เป็น Agent ที่ตัดสินใจ ไม่ใช่แค่ทำตามกฎ

# ตัวอย่าง Agentic Decision ใน Lab นี้

สถานการณ์: User พิมพ์: “อยากไปเที่ยวเชียงใหม่ 3 วัน”

AI ทำงาน 2 ขั้นตอน:

1. AI Extract Info — แยกข้อมูลจากข้อความ:

- Input: “อยากไปเที่ยวเชียงใหม่ 3 วัน”
- Output: {"city": "Chiang Mai", "days": 3}

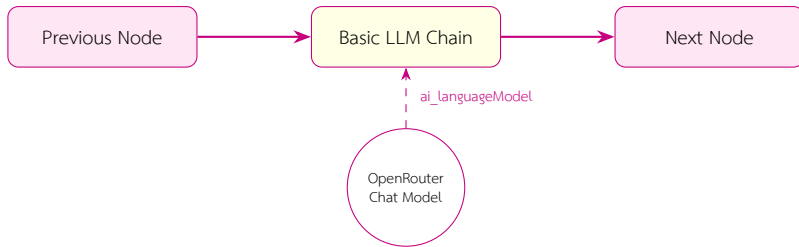
2. AI Plan Trip — วางแผนทริปจากข้อมูลจริง:

- อากาศ: 25°C, ฝนตกช่วงบ่าย
- สถานที่: วัดพระธาตุดอยสุเทพ, ย่านนิมมาน, ไนท์บาซาร์
- AI เลือก: เข้าไปดอยสุเทพ, บ่ายเดินนิมมาน (ในร่ม), เย็นไนท์บาซาร์

ไม่ต้อง parse command เอง — AI ทำให้ทั้งหมด!

# n8n Basic LLM Chain + OpenRouter Chat Model

**Basic LLM Chain** = n8n node สำหรับเรียก AI/LLM โดยตรง

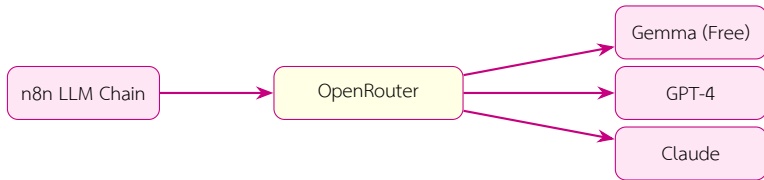


ข้อดีเทียบกับ HTTP Request:

- ไม่ต้องเขียน JSON body เอง — แค่พิมพ์ prompt
- ไม่ต้องจัดการ header/auth — ใช้ n8n Credentials
- เปลี่ยน model ง่าย — แค่เลือกจาก dropdown
- Output: `$json.text` หรือ `$json.output` (ไม่ต้อง parse `choices[0]`)

# OpenRouter คืออะไร?

OpenRouter = API Gateway สำหรับ LLM หลายตัว



ข้อดี:

- Credentials เดียวใช้ได้กับ LLM หลายตัว
- เปลี่ยน model แค่เปลี่ยนชื่อใน dropdown
- มี Free models: **google/gemma-3-27b-it:free**



# AI Extract Info — หัวใจของ Agentic Workflow

ให้ AI แยกข้อมูลจากข้อความอิสระของ User

**ปัญหาเดิม:** User ต้องพิมพ์ตาม format เช่น !plan Bangkok 3

**แนวทางใหม่:** User พิมพ์อะไรก็ได้:

- “อยากไปเที่ยวเชียงใหม่ 3 วัน” → {"city":"Chiang Mai","days":3}
- “Plan a trip to Tokyo” → {"city":"Tokyo","days":3}
- “แนะนำที่เที่ยวกรุงเทพ” → {"city":"Bangkok","days":3}

**Prompt สำคัญ:**

- บอก AI ว่า “ตอบเป็น JSON เท่านั้น”
- ให้ตัวอย่าง input/output หลายๆ แบบ
- กำหนด default (ถ้าไม่ระบุวัน = 3 วัน)
- city ต้องเป็นภาษาอังกฤษ (สำหรับ API)

# Parse Extract Result — Code Node

## ทำไมต้องมี Code Node หลัง AI Extract?

- AI อาจตอบ JSON ไม่สมบูรณ์ หรือมีข้อความอื่นปน
- Code Node ทำหน้าที่ parse + fallback

### Logic:

- 1 หา JSON pattern {...} จาก AI response
- 2 Parse เป็น object ดึง city, days
- 3 ถ้า parse ไม่ได้ → fallback เป็น Bangkok, 3 วัน
- 4 จำกัด days: 1-7 วัน

### Output:

- \$json.city = “Chiang Mai” (ใช้ใน Get Weather)
- \$json.days = 3 (ใช้ใน AI Plan Trip)

# Prompt Engineering: System vs User Message

Role	หน้าที่	ตัวอย่าง
system	กำหนดบทบาท/กฎ	"You are a travel planner"
user	ส่งข้อมูล/คำถาม	"Plan 3 days in Bangkok..."
assistant	คำตอบจาก AI	(AI ตอบกลับ)

## เคล็ดลับ System Prompt ที่ดี:

- 1 ระบุ **บทบาท** ชัดเจน (expert travel planner)
- 2 ระบุ **ภาษา** ที่ต้องการ (Thai language)
- 3 ระบุ **format** ผลลัพธ์ (day-by-day, morning/afternoon/evening)
- 4 ระบุ **ข้อจำกัด** (concise, consider weather)

# Prompt Engineering: ตัวอย่างจริง

## AI Extract Info (Prompt):

*“Extract city name and days from user message. Respond with ONLY JSON: {”city”:”CityName”,”days”:3}. City must be in English. Default days = 3.”*

## AI Plan Trip (Prompt):

*“You are an expert travel planner. Create a day-by-day itinerary in Thai language based on weather data and tourist attractions. Consider weather for outdoor vs indoor activities.”*

ยิ่ง prompt ชัด ยิ่งได้ผลลัพธ์ตรงประเด็น!

# OpenWeatherMap API

## ดึงข้อมูลสภาพอากาศปัจจุบัน

รายการ	ค่า
URL	api.openweathermap.org/data/2.5/weather
Method	GET
Auth	API Key (query param)
Free Quota	1,000 calls/day

## ข้อมูลสำคัญ:

- main.temp — อุณหภูมิ (°C)
- main.humidity — ความชื้น (%)
- weather[0].description — คำอธิบาย (เมฆกระจาย)
- coord.lon, coord.lat — พิกัด (ใช้ต่อกับ Geoapify)

# Geoapify Places API

## ค้นหาสถานที่ท่องเที่ยวใกล้เคียง

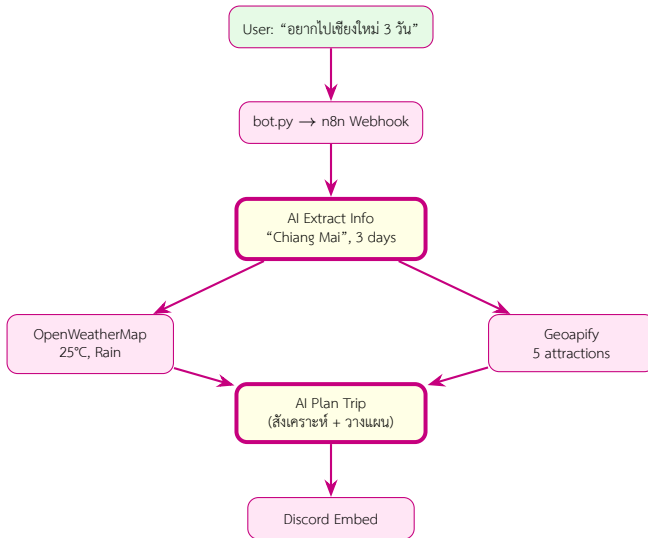
รายการ	ค่า
URL	api.geoapify.com/v2/places
Method	GET
Auth	API Key (query param)
Free Quota	3,000 calls/day

## Parameters สำคัญ:

- categories — tourism.sights (สถานที่ท่องเที่ยว)
- filter — circle:lon,lat,radius (ค้นหารอบจุด)
- limit — จำนวนผลลัพธ์ (5)

ใช้ coord จาก OpenWeather เป็น filter ของ Geoapify!

# Data Flow: Agentic Travel Planner



## Discord Embed: สีตามสภาพอากาศ

สภาพอากาศ	Emoji (Code)	สี	Decimal
Clear (แจ่มใส)	sun	Gold	15844367
Clouds (มีเมฆ)	cloud	Gray	9807270
Rain (ฝนตก)	rain	Blue	3447003
Thunderstorm	storm	Purple	7419530
Snow (หิมะ)	snow	White-Blue	12370112
Default	partly cloudy	Teal	1752220

### เปรียบเทียบกับ Lab 04:

- Lab 04: สีตาม **alert level** (green/yellow/orange/red)
- Lab 05: สีตาม **สภาพอากาศ** (Clear/Clouds/Rain/...)



# bot.py: ส่งข้อความดิบให้ AI แยกข้อมูล

**Lab 04:** bot.py parse command เอง (!earthquake)

**Lab 05:** bot.py ส่งข้อความดิบ — AI แยกข้อมูลให้

User types anything:

อยากไปเที่ยวเชียงใหม่" 3 วัน"

"Plan a trip to Tokyo for 5 days"

แนะนำที่เที่ยวกรุงเทพ"

Payload ที่ bot.py ส่งไป n8n:

```
{  
  "content": "อยากไปเที่ยวเชียงใหม่" 3 วัน",  
  "author": { "username": "student01" },  
  "channel_id": "123...",  
  "guild_id": "456..."  
}
```

ไม่มี city/days ใน payload — AI Extract Info จะแยกให้!

## n8n Webhook: test vs production (ทบทวน)

Mode	URL Path	เมื่อไหร่
Testing	/webhook- <b>test</b> /travel-planner	กด Test workflow
Production	/webhook/travel-planner	Activate Workflow

bot.py ต้องใช้ /webhook/ + Workflow ต้อง Active!

# API Key Security

Lab นี้มี API Key 3 ตัว — ต้องจัดการให้ปลอดภัย!

## ห้ามทำ:

- ใส่ key ใน README.md
- Commit .env ลง Git
- แชร์ key ใน Discord
- Hardcode ใน code ที่ push

## ควรทำ:

- เก็บใน .env file
- ใส่ .env ใน .gitignore
- ใช้ n8n Credentials
- ใช้ Environment Variables

## วันนี้เรียนรู้:

- 1 Agentic Workflow — AI ตัดสินใจ ไม่ใช่แค่ IF/ELSE
- 2 Basic LLM Chain — n8n node สำหรับเรียก AI โดยตรง
- 3 OpenRouter Chat Model — Sub-node เชื่อมต่อ LLM
- 4 AI Extract Info — ให้ AI แยก city/days จากข้อความอิสระ
- 5 Prompt Engineering — System vs User message
- 6 Multi-source Aggregation — Weather + Places → AI
- 7 OpenWeatherMap API — สภาพอากาศ
- 8 Geoapify Places API — สถานที่ท่องเที่ยว
- 9 API Key Security — .env, .gitignore, Credentials

## Lab 05: AI Travel Planner Bot

### สิ่งที่ต้องทำ:

- ❶ สร้าง Discord Server + Channel #travel-plans
- ❷ สมัคร API Keys 3 ตัว (OpenWeather, Geoapify, OpenRouter)
- ❸ สร้าง Workflow (Chat Command) — 8 Nodes + 2 Sub-nodes
- ❹ ตั้งค่า bot.py → ทดสอบพิมพ์อะไรก็ได้
- ❺ Export workflow.json → Push ขึ้น GitHub

คะแนน: 100 คะแนน (10 Tests)

## มีคำถามไหม?

ดู Lab Instructions: README.md

ดูคู่มือตั้งค่า: SETUP\_GUIDE.md