元智大學資訊管理學系
第三十屆專業實習報告

財團法人資訊工業策進會

競賽展分組代號：02

實習單位：資策會

輔導老師：王仁輔 教授

姓 名：邱子芸、任柏恩

學 號：1111707、1111727

壹、工作內容

貳、學習

一、技術研究

二、技術個案研究

三、專題應用實作

壹、工作內容

一、工作環境介紹

資策會數位轉型研究院（數轉院）致力於推動臺灣企業的數位轉型，提供從技術研發到應用落地的全方位支援。實習生在此環境中，與專業的研發團隊共同協作，參與實際專案，並在同事的指導下，透過實作中學習，累積職場經驗。實習期間涵蓋生成式 AI 應用的各個層面，從前期的技術爬蒐與應用分析、原型開發，到後期的系統整合與測試部署，提供了豐富且紮實的實戰機會。

二、工作詳述

在實習期間，主要負責以下是共同工作項目：

- **協助執行生成式 AI 相關資料與應用的爬蒐**：參與生成式 AI 的研究與實作，探索其在不同應用場景中的可能性，並定期評估最新工具與技術，如 ChatGPT、Claude、Stable Diffusion、Runway ML 等，記錄使用經驗並分享於團隊會議中。

- **協助開發生成式 AI 應用**：實際參與文本生成、圖像生成、語音合成等應用的原型設計與功能實作，涵蓋 LLM 開發（如 Llama2、OpenAI API）、圖片生成（如 Stable Diffusion）、TTS/Speech 合成服務等。

- **進行系統整合與 API 開發**：將模型包裝為 RESTful API，並使用 FastAPI、Docker 進行服務部署，提升服務的穩定性與擴展性，支援跨團隊的功能整合與共享。

- **撰寫技術文件與測試案例**：負責專案文件、技術筆記與測試案例設計，協助團隊建立可維運的技術基礎與知識庫，確保專案開發過程的品質控制。

- **跨部門協作與技術提案**：透過每週例會與技術簡報，與不同部門同仁交流專案成果與學習心得，協助理解業務需求並提出適合的生成式 AI 應用解決方案。

在技術方面，熟悉向量資料庫（如 Chroma、Pinecone）、圖形資料庫（Neo4j）、Python、Git/GitHub 等工具，並能運用 LangChain 框架整合各種 LLM 元件（如 PromptTemplates、Memory、Toolchains、Retrievers 等）進行應用開發。

三、實習期間完成之進度

子芸:

我工作進度大致可分為三個階段。初期主要著重於生成式 AI 技術的探索，實作包含圖片生成工具（如 Stable Diffusion）的應用，以及語音合成（TTS）模型的測試與整合，並開始接觸 Whisper 語音辨識模型，進行中英文語音辨識實驗。中期則深入語音處理領域，進行聲紋辨識（Speaker Recognition）研究，並以 SpeechBrain 框架實作語者註冊與辨識流程，設計自動儲存與信心門檻判斷邏輯。同時，我也嘗試進行台語語音轉台羅拼音的研究，結合 Whisper 模型與台灣語料庫，學習語言處裡的應用。而在語言模型應用方面，我使用 LangChain 框架實作資料檢索與問答功能，逐步建立對 Retrieval-Augmented Generation（RAG）結構的理解。後期的部分，我將重心轉向以 n8n 為核心的自動化流程設計，成功整合表單輸入、主題分析、資料查詢、段落撰寫、HTML 組版、PDF 匯出與 Gmail 寄送等多項模組，完成一條龍的研究報告產生流程。

柏恩:

實習初期參與與 AI 圖像生成相關的專案，熟悉 Stable Diffusion 等技術，但因任務性質較偏研究測試，學習深度有限。後續在同事的指導下，轉向 LLM 應用開發方向，開始學習 LangChain 框架，深入理解其核心元件（如 Chat Models、Prompts、Chains、Memory、Agents、Retrieval 等）與應用場景。

四、工作當中扮演的角色

在實習期間，我們扮演以下角色：

- **技術協作者**：與團隊成員共同協作，從技術討論、系統設計、模組開發到測試部署等階段，均積極參與並提供可行建議，協助專案順利推進。
- **資料爬蒐 :** 定期察看 github 上的開源專案，作為開拓者去試用，紀錄使用體驗，並在每周的會議上進行匯報。
- **文件與報告整理者**：協助彙整專案過程中的設計流程圖、研究成果與進度紀錄，並將研究內容轉化為報告架構與簡報資料，用以支援團隊對外展示與階段性彙報。

貳、學習

一、技術研究

LangChain

LangChain 是一套專為大型語言模型（LLM）應用所設計的開源開發框架，其核心目標是讓開發者更容易將語言模型整合到實際應用中。它不僅支援與外部工具、資料庫、API 等進行串接，也能建立多步驟的對話邏輯與決策流程，廣泛應用於智慧問答、資料查詢、工作自動化與報告生成等場景。

1. 核心概念與模組結構理解

- **Models（模型）**：使用 ChatModel 處理多輪對話，能正確辨別角色（System、User、AI）。
- **PromptTemplate（提示語模板）**：學習如何撰寫具一致性與重用性的提示語格式。
- **Chains（鏈式流程）**：了解如何將多個處理步驟串接為可維護的流程，例如「接收問題 → 檢索資料 → 生成回答」。
- **Memory（記憶模組）**：實作能保留對話歷史的功能，提升用戶體驗的連貫性。
- **Agents（智能代理）**：學習如何讓 AI 根據需求自動決定使用哪些工具解決問題。
- **Tools（工具擴充）**：是我最投入且實作最多的部分，能讓 AI 具備額外能力。
- **Retrieval（資訊檢索）**：整合外部資料庫以提升回答的準確性與專業度。
- **Cache（快取）**：初步理解其在大流量場景中加速回應與節省資源的應用。

2. 工具 Tool 模組用運補充

- **使用內建工具**：
    - get_current_time：提供時間查詢功能。

- get_weather：整合即時天氣查詢 API。
- currency_converter：將匯率查詢納入對話中，實現動態資訊查詢。

- **開發自訂工具**：
  - **PDF 知識問答助手**：透過將 PDF 轉為向量並整合 FAISS，再設計工具供 LLM 查詢回應。
  - **Google Search API 查詢器**：實作一個工具讓 LLM 可動態搜尋網頁資料作為回答依據。
  - **資料庫查詢工具**：讓 AI 可針對公司內部的產品資料庫查詢條目，回傳條件篩選結果。

3.LangChain 技術應用實作：客服知識問答流程

在掌握 LangChain 各模組後，我們實作了一套整合了 Retrieval、Tool 與 Agent 等技術的智慧客服問答系統，此研究主要是模擬企業內部常見的客服應答情境，目的是將原本分散在 FAQ 表格中的知識轉換為結構化、可查詢的知識庫，並透過語言模型實現自然語言問答回應。

實作流程：

1. **資料處理**：將常見客服問答表彙整至 Excel，作為知識來源。
2. **向量化儲存**：將問題欄位向量化，並用 FAISS 建立檢索系統。
3. **LangChain 結構設計**：

- 使用 Retrieval 模組建立向量查詢邏輯。
- 用 Tool 包裝查詢功能，供 Agent 呼叫。
- Agent 分析提問語意，自動判斷是否需要查詢知識庫。

4. **AI 回應設計**：

- 使用 OutputParser 將查詢結果轉為簡潔口語化回答。
- 模型能進行類似「客服互動」的回應模式。

4.成果特色

- **支援自然語言提問**：使用者可直接以口語化方式提出問題，例如「公開標案日期是什麼時候？」或「如何申請廠商名稱變更？」系統能理解語意並進行相關知識檢索。
- **提升查詢準確度**：系統能從知識庫中找出語意最接近的內容，提供具參考價值的回應，有效降低關鍵字誤判的情況。
- **模擬真實客服互動**：AI 回覆內容條理清晰、語氣自然，具備實用性與親和感，接近實際客服的回應模式。
- **強化作業效率與應用潛力**：透過語言模型與向量檢索的結合，大幅降低人工查閱 FAQ 或內部文件的時間。

---

n8n

n8n 是一款開源的工作流程自動化工具，支援無程式碼與低程式碼操作。使用者可透過視覺化介面，將各種應用程式與服務串聯，建立自動化流程，如資料同步、通知發送等，靈活且易於擴充，適合開發者與企業使用。

1. 核心概念與模組理解

- **節點（Node）**：n8n 中的每個步驟都是一個功能節點，支援 HTTP 請求、讀寫 Google Sheets、執行 JavaScript 等。
- **觸發器（Trigger）**：如 Webhook、時間排程等，可設定流程啟動條件。
- **流程邏輯（Workflow）**：透過拖拉方式設計資料流與條件判斷，具備高度彈性與可視化優勢。
- **環境變數與資料傳遞**：了解如何在節點間傳遞 JSON 格式資料，及如何存取前一個節點的輸出。

2. n8n 技術應用實作: 智慧問答 with Line

在熟悉 n8n 的流程設計與資料傳遞邏輯後，我們嘗試實作了一個**文字查詢流程**，讓使用者可以透過 LINE Bot 提問，系統會回到向量資料庫進行語意比對，並傳回最相關的資料作為回應。

流程說明

1. **接收提問文字 :** 使用者透過Line 介面輸入問題，透過 webhook 傳入 n8n 系統。

2. **轉換向量嵌入:** 使用 Google Gemini Embeddings 模組將輸入文字轉為向量表示。

3. **語意比對與資料檢索 :** 在 Pinecone Vector Store 中進行相似語意查詢，找出最接近的段落作為回應依據。

4. **整合回答與回傳 :** 啟用 OpenRouter ChatModel 模型，將檢索結果組合並生成自然語言回答，再回傳給使用者。記憶模組（Simple Memory）可記錄上下文資訊，支援後續多輪提問。

5. **輔助查詢擴充（例: SerpAPI）:**
   若向量庫查詢未命中，系統亦可自動觸發 SerpAPI 進行外部網路搜尋作為補充來源。

二、技術個案研究

研究個案:聲紋辨識模型（子芸）

在技術個案研究部分，我選擇探索語音領域中的「聲紋辨識（Speaker Recognition）」技術，並以開源語音處理框架 SpeechBrain 為基礎進行實驗與測試。聲紋辨識是一種利用個人聲音中獨特的特徵來識別說話者身份，與語音辨識（辨內容）並不同相同，其核心目的是「辨人」而非「辨語」。此技術主要可分為語者驗證（Speaker Verification）與語者辨識（Speaker Identification）兩大類，廣泛應用於語音安全驗證、個人化語音助理、智慧監控與語音存取控制等情境。

1.研究工具與技術架構

SpeechBrain 是一套基於 PyTorch 的開源語音處理框架，支援語音辨識、聲紋辨識、語音合成等多種應用。這次主要使用內建的 voiceprint recognition 模組，透過 CLI 介面進行以下操作：

- 語者註冊（via microphone）
- 聲紋辨識
- 即時語者辨識迴圈（real-time loop）
- 從檔案進行語者辨識（e.g., test.wav）

2.技術細節與門檻判定設計

實作過程中，我使用 SpeechBrain 提供的 ECAPA-TDNN 模型進行語者嵌入（voice embedding）與相似度比對，使用的相似性指標為**餘弦距離（cosine distance）**，數值越小代表語音越相近。

為提升實用性與自動化程度，我設計了一些辨識邏輯與門檻條件：

- **比對方式**：將輸入語音與資料庫中的語者進行逐一比對，選出餘弦距離最小者作為預測結果。

- **信心門檻設定**：
    - 若距離分數小於 0.60（即相似度大於 0.40），系統視為有效辨識。
    - 若分數高於此門檻，則標記為「Unknown」，並顯示信心分數。
- **自動註冊功能**：若辨識為「Unknown」且信心分數低於 0.40，系統會自動將該語音儲存為 auto_user_x.wav，保留未知說話者資料供後續訓練或人工標註。

3.實作流程

- **語者註冊**：透過麥克風輸入語音，並儲存為 .wav 檔案，以建立語說話者的聲紋資料庫。
- **語者辨識**：輸入語音樣本（即時錄製或來自檔案），系統進行嵌入比對與語者預測。
- **結果輸出**：於 CLI 介面即時顯示辨識結果與信心指標，可視情況啟用自動儲存功能。

4.研究成果與反思

研究成果

透過本次個案研究，我成功建置了一套基於 SpeechBrain 的說話辨識原型流程，完整涵蓋語者註冊、辨識判定、信心分數計算與自動儲存等功能模組。系統能夠即時辨識語音來源是否為已登錄說話者，並於辨識信心不足時啟用自動註冊機制，具備初步的延用與實用性。經過多次測試，在語音清晰、背景噪音低的情況下，辨識準確率表現穩定，展示了說話者嵌入模型在小型資料集下的可行性。此外，本專案亦加深了我對語音處理流程、特徵向量比對與模型判斷邏輯的理解。

研究反思

在研究與實作過程中，我發現說話辨識雖然流程看似簡單，但實際操作時需要考慮多種細節與限制。例如：錄音品質對辨識準確率的影響極大，背景雜訊、語速變化或發音不清楚都可能導致模型誤判。另外，在處理語音比對邏輯時，我也學習到如何設計合理的信心門檻，避免過度誤判或錯誤接受。

此外，雖然 SpeechBrain 提供了高度整合的模型與 API，但在理解其內部嵌入流程與相似度判斷邏輯上仍需要一定的深度學習背景。這讓我意識到在實作 AI 專案時，僅僅呼叫模型並不足夠，理解背後原理與數據處理方式同樣重要。

整體而言，這次研究不僅讓我學習到語音處理與辨識技術的基礎知識，也訓練了我在實務應用中解決問題、設計流程邏輯與進行錯誤處理的能力，是一次具挑戰性但收穫豐富的實驗經驗。

研究個案:LLM 記帳機器人(柏恩)

這款 LLM 記帳機器人旨在提供一種直覺、輕鬆的記帳方式，讓使用者能夠直接以日常語言記錄消費。例如，只需輸入「今天午餐花了 120 元」，系統便能自動辨識金額與分類，並支援同時處理多筆消費紀錄。為了提高使用者的信任與便利性，每次記帳都會透過 LINE 的 Flex Message 主動回饋與確認，讓記帳過程更安心可靠。

圖. 透過自然語言記帳，並回傳確認按鈕

1. 研究工具與技術架構

本系統採用標準的 LINE Bot 架構：

使用者將訊息傳送至機器人的 LINE 帳號後，LINE 平台會觸發 webhook event 並將其傳送至後端伺服器。伺服器接收事件並進行處理，再透過 LINE 平台回應訊息給使用者，形成完整互動流程。

後端技術細節:

- **Bot Server 架設**：採用 FastAPI 作為主要伺服器框架，提供輕量、高效的 API 處理能力。
- **LINE Bot SDK**：使用 LINE 官方提供的 Python SDK
- **大型語言模型整合**：透過 LangChain 串接 OpenAI GPT-4o mini，進行自然語言理解與消費分類。
- **資料儲存與快取**：使用 SQLite 作為主資料庫儲存使用者帳務資料，Redis 快取處理。

2.技術細節

為了讓記帳機器人不僅能記錄消費，還能回應使用者查詢如「這週花了多少」或「這個月的收入是多少」等問題，我們設計並實作了一套完整的「意圖分析與處理系統」，結合大型語言模型的理解能力與明確的指令工具，讓機器人可以理解多樣的使用者需求並做出相應回應。

意圖分類與處理機制

我們將使用者意圖分為兩類：

- **需要使用者確認的操作（會寫入或刪除資料）**：
  包括 create_record（新增紀錄）、delete_record（刪除紀錄）、create_category（新增分類）、edit_category（編輯分類名稱）、delete_category（刪除分類）。這類操作需透過 LINE Flex Message 回傳確認按鈕，避免 LLM 錯誤理解使用者的需求。

- **不需使用者確認的查詢操作**：
  包括 query_bytime（查詢指定時段內紀錄）、query_latest（查詢最近幾筆紀錄）、list_categories（列出所有分類）等。

為實作這些功能，我們在 tools.py 中設計了對應的工具函式，並擴充 LLMProcessor 來解析使用者輸入的語意並指派正確的工具處理。處理邏輯則實作於 line_bot_service，涵蓋意圖分流、回應生成與確認流程。

使用者確認與快取策略

由於 LINE 平台不像瀏覽器那樣支援 session 機制，Flex Message 傳送後 bot 就無法記住當下的操作上下文。為了解決這個問題，我們除了主資料庫 SQLite 之外，額外引入 Redis 作為快取式資料庫，用來暫存待確認的操作內容。當使用者點擊 Flex Message 的確認按鈕時，bot 可從 Redis 中讀取原始指令並完成資料寫入或刪除操作

3.研究成果與反思

研究成果

本專案成功開發一款支援自然語言記帳與查詢的 LINE 機器人，整合 GPT-4o mini 模型與 FastAPI 架構。使用者可透過簡單語句（如「今天午餐 120 元」）快速記帳，並查詢本週或本月收支情況。

技術上採用兩層資料架構（SQLite + Redis），解決 LINE 無 session 問題，確保操作確認機制穩定運作。意圖系統區分「需確認」與「免確認」操作，提升準確性與安全性，同時導入 Flex Message 設計優化互動體驗。

開發反思

現今的 LLM 能力越來越強，但仍會出現一些預期之外的錯誤。因此，我認為當任務涉及「修改資料」時，增加一層確認機制是非常必要的。以本記帳系統為例，若 LLM 將使用者原本想「刪除一筆記帳紀錄」的意圖誤解為「刪除整個分類」，那麼後果將會嚴重影響使用者體驗。這不只是針對本系統，而是所有 LLM 相關的系統!

這也是當我發現 LINE Bot 不支援 session，導致無法記住訊息上下文時，仍決定設法解決這個問題的原因。雖然一開始感覺技術門檻很高、甚至一度想放棄，但透過一步步拆解問題，最後發現其實實作確認機制並沒有想像中困難。成功完成後，當下甚至一度覺得沒什麼難得倒我了（並沒有）。

同時，我也很慶幸當初選擇了模組化設計。雖然初期開發比較繁瑣，但如今要新增功能，只需要撰寫一小段函式就能快速擴充，不僅開發效率高，出錯機率也大幅降低。

三、研究專題應用實作

主題：用 n8n 生成深度研究報告（子芸）

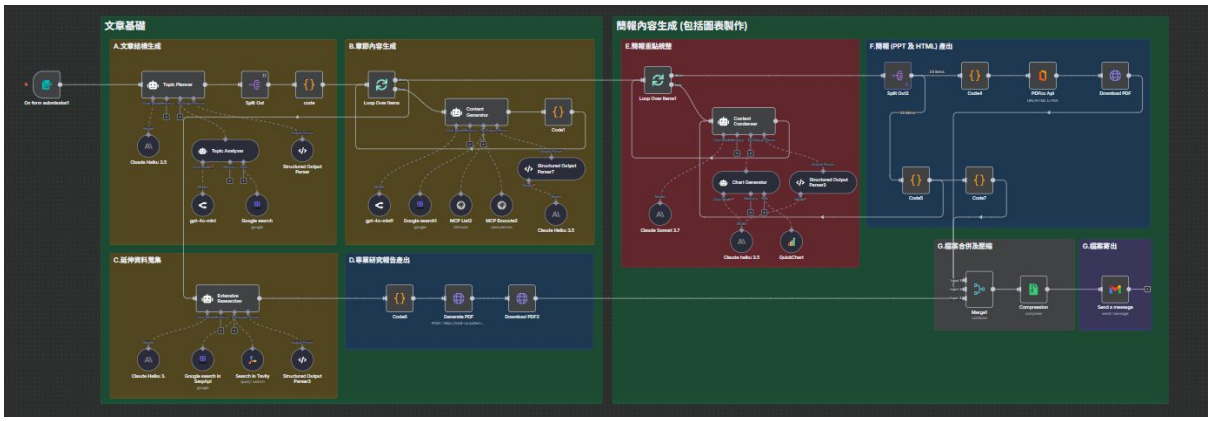
1. 系統設計與規劃

此專題以 n8n 作為主要的流程設計與自動化平台，主要打造一套能根據使用者輸入的主題，協助完成研究報告並產出的自動流程。整體架構整合了語言生成模型（如 Claude 3.5 / GPT 4o-mini）、即時資料擷取 API（Tavily 及 Serp API），並透過工具代理（Tools Agent）完成從主題拆解、資料搜尋、段落撰寫到報告整合的各項任務。除了文字內容外，系統也能辨識適合視覺化的資訊，並透過圖表生成工具產生統計圖、流程圖與架構圖等視覺化輔助資料，讓研究內容更完整、易讀且具展示效果。最後，所有產出的內容會依照不同展示需求，自動輸出成三種成果檔案：正式版研究報告（PDF）、簡報檔案（PPT）以及可於瀏覽器播放的 HTML 網頁版簡報。

此流程可大幅簡化學生或研究人員在撰寫研究內容初期所面臨的資訊收集與結構建立之壓力，透過自動化的方式提供具邏輯性、參考依據並包含圖表輔助的初步內容提升寫作效率。

而在 n8n 的實作中，整套系統由**八個主要模組（A～H）**所組成，每個模組負責不同任務，從文章規劃到簡報輸出都包含在內：

A. 文章結構生成　**B.** 章節內容生成　**C.** 延伸資料蒐集　**D.** 專業研究報告產出
E. 簡報重點統整　**F.** 簡報（PPT 及 HTML）產出　**G.** 內容合併及壓縮　**H.** 檔案寄送



這八個模組依照流程被整理成四個主要階段架構，如下：

1.1 使用者輸入與文章結構規劃（模組 A：文章結構生成）

使用者透過表單輸入研究主題、Email 與簡短說明後，系統流程會自動啟動。在這個階段，主要由兩個 Agent 負責規劃整份文章的框架：

- **Topic Analyzer（資料理解與內容分析 Agent）**：此 Agant 會先透過搜尋工具（如 Google Search API 及 Tavily）查找與主題相關的背景資訊、定義、常見問題與討論方向，並根據結果摘要出主題的核心概念與重要面向。
- **Topic Planner（文章架構規劃 Agent）**：
  根據 Topic Analyzer 所整理出的資訊，將主題拆解成數個主要章節，並建立文章的大綱結構，包含章節標題與內容方向。

這個階段主要是先協助使用者釐清「文章要寫什麼」，並建立後續內容生成的基礎骨架，使整份研究報告能保持清楚的邏輯結構。

1.2 章節內容生成與延伸資料蒐集（模組 B、C、D）

在文章架構建立後，系統會進入內容撰寫階段。這個階段由 Content Generator 主導撰寫每個章節的內容，而在正式生成段落之前，會先透過底下的工具與 Agent 協助理解資料與補充內容，使生成的文本更完整、更有根據。

- **Content Generator（章節內容撰寫 Agent）**：此 Agent 是各章節的主要撰寫核心。在開始生成段落內容之前，Content Generator 會先透過 **MCP Tools** 查找主題相關的外部資料，包括論文摘要、新聞內容、產業與公司資訊，以及相關技術概念的背景與定義。

  而在掌握這些資訊後，Content Generator 會依照章節大綱進行撰寫，將背景脈絡、問題分析、概念比較、實際案例等內容整理成具體且完整的段落，使文章在敘事上更具專業性與邏輯性。

- **Extensive Researcher（延伸內容補強 Agent）:**此 Agent 會在 Content Generator 完成初稿後，根據章節內容進行補充查詢，包括延伸研究、相關案例、背景資料等資訊。這些補充資訊會被整理成研究報告後段的「**延伸閱讀**」區塊，提供讀者在需要時進一步探索主題、取得更多背景脈絡與參考來源。

完成後，所有章節的草稿內容會以標準格式輸出，並交由 **模組 D** 進行整合與排版，最終生成完整的 PDF 研究報告版本，作為正式的成果文件。

1.3 簡報重點與圖表內容生成（模組 E、模組 F）

專業研究報告內容完成後，系統會進入簡報內容的整理階段。本階段主要由模組 E 負責，將原始篇幅較長的研究內容轉換為適合簡報呈現的重點與圖像化資料，使使用者能以更精簡、直觀的方式展示研究成果。

- **Chart Generator（圖表生成 Agent）**:此 Agent 會先閱讀由前一階段所生成的完整研究報告內容，並分析哪些段落適合透過視覺化方式呈現。Chart Generator 能從文字中辨識出可圖像化的數據結構，例如比例比較、趨勢變化、分類架構或流程模型等。在完成分析後，Chart Generator 會使用內建工具**Quick Chart**及提詞提供的**Mermaid 圖表模板**，自動生成相對應的圖表內容，並輸出包含圖表、說明文字與圖像檔案等內容。

- **Content Condenser（簡報重點統整 Agent）**: 此 Agent 會根據 Chart Generator 所產生的圖表內容與研究報告的段落，將圖表分配到最適合的簡報章節位置，例如背景介紹、問題分析、結果比較或結論段落。在處理圖表分配後，Content Condenser 會進一步將篇幅較長的報告內容濃縮成精簡且符合投影片呈現的條列式重點。

在簡報重點與圖表內容整理完成後，模組 F 會將所有簡報文字、圖表資料與段落架構套用到既定的版面格式中，並生成 PPT 簡報檔以及可於瀏覽器直接展示的 HTML 網頁版簡報（HTML Slide）。

1.4 多格式成果輸出與寄送（模組 G、模組 H）

在研究報告（PDF）與簡報內容（PPT、HTML Slide）完成格式整理後，系統會進入最終的成果輸出與寄送階段。此階段的流程主要包含檔案整合、壓縮與寄送，確保使用者能一次取得所有成果。

- **檔案整合（模組 G）**
  此階段會將產出的所有檔案一併整理，包括 PDF 研究報告、PPT 簡報與 HTML 網頁版簡報，統整成完整的成果資料集，方便後續進行壓縮與寄送。

- **壓縮與寄送（模組 H）**
  整合後的檔案會被打包成 ZIP，並寄送到使用者表單中提供的 Email。信件內容會附上說明文字與下載連結，讓使用者能快速取得並使用所有成果。

2. 技術與方法

- **資料擷取：**使用 Tavily API、SerpAPI 與 MCP Tools 取得主題相關的新聞、研究、數據與背景資訊。

- **語意生成：**透過 Claude 3.5、Claude 3.7 Sonnet、GPT-4o mini 與多個 Tools Agent 完成文章架構規劃、內容撰寫與延伸資料整理。

- **圖表生成:**使用 QuickChart 製作數據圖，並依據預先設計的 Mermaid 模板自動生成流程圖與結構圖。

- **流程控制與格式整理：**使用 JavaScript 節點、Merge、Loop Over Items、HTML Formatter 等模組處理資料並控制流程。

- **輸出與傳送：**使用 PDF/PPT 轉換工具輸出研究報告（PDF）、簡報（PPT、HTML Slide），並以 Email 自動寄送與 ZIP 壓縮下載。

3. 預期研究效益

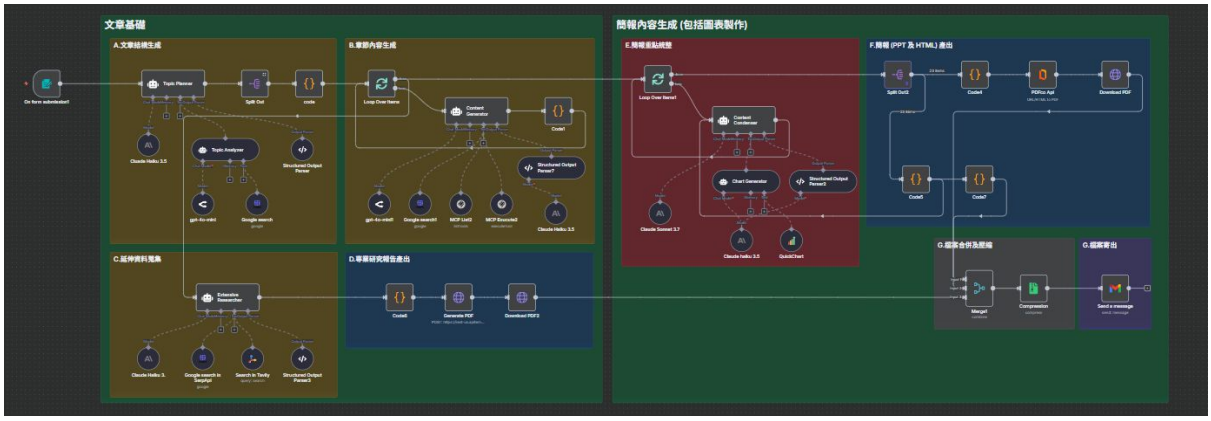
- **提升撰寫效率：**將研究報告的初步內容生成交由自動化流程執行，可大幅減少使用者在主題拆解、資料搜尋、內容組織與段落撰寫上的時間成本。

- **強化內容品質與資訊完整性：**系統結合即時搜尋（Tavily、SerpAPI）、資料查詢工具（MCP Tools）以及延伸閱讀資料蒐集等功能，讓內容在產製時能同時具備背景脈絡、案例輔助與相關參考資訊，大幅提升研究的可信度與完整性。

- **提升研究成果的閱讀性與展示效果：**系統會自動生成適合簡報的重點條列與視覺化圖表，並輸出 PDF、PPT 與 HTML Slide，使研究成果更易於在課堂、會議或展示場合中說明與呈現。

- **多元應用潛力：**此流程框架可依不同主題靈活套用，並能延伸至簡報內容整理、教學素材製作、專題初稿撰寫等多種情境，具有豐富的跨領域應用價值。

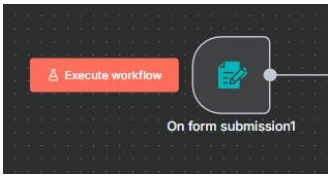
4. 系統實作流程

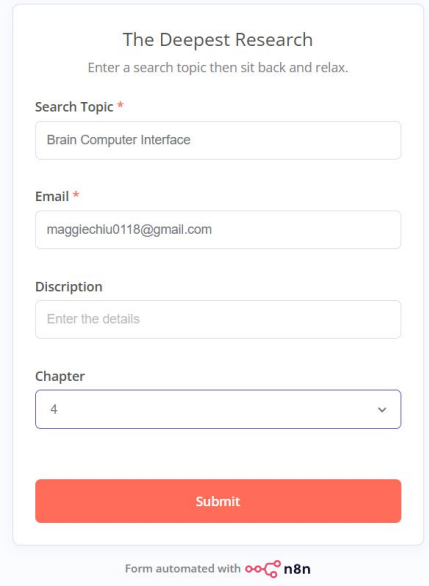


整體流程展示

4.1 使用者輸入介面（Form Interface）

本系統的操作入口是線上表單介面，使用者只需填入研究主題、Email 與主題說明，即可啟動完整的研究生產流程。介面設計採用簡潔、容易理解的格式，讓使用者在無須額外設定的情況下，就能開始生產研究內容。





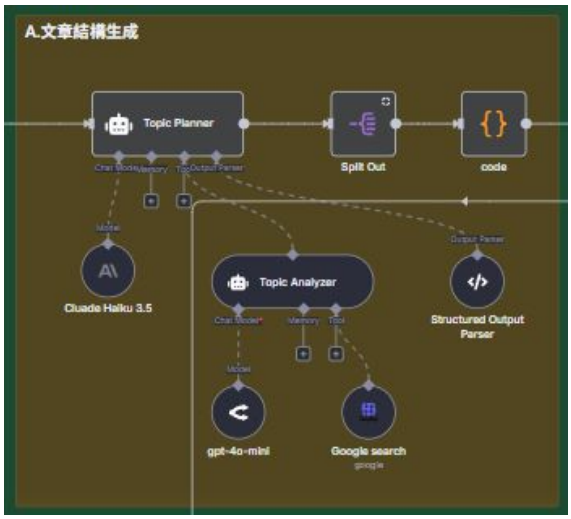
表單共有四個輸入欄位：

- **Search Topic**：使用者希望研究的主題或關鍵字。
- **Email**：系統在完成所有成果產出後會寄送至此信箱。
- **Description**：讓使用者補充研究方向、需求或背景資訊。
- **Chapter**：可選擇研究方向或章節類型，讓系統更準確規劃文章架構。

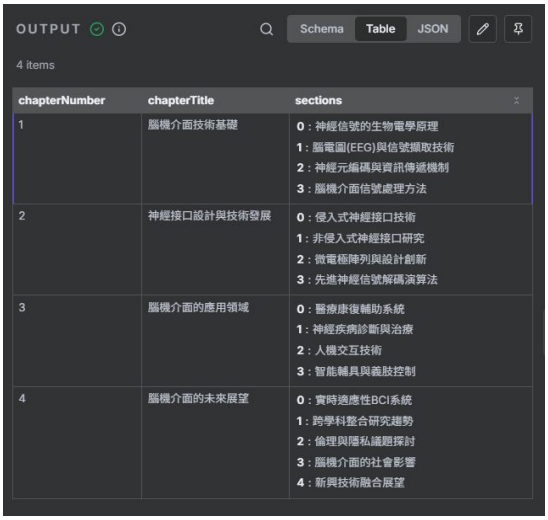
所有資料送出後，表單會透過 n8n Webhook 觸發後續流程，並將輸入內容傳入模組 A 進行文章架構規劃。

4.2 系統建立文章架構（模組 A 實作流程）

在使用者提交研究主題後，系統會進入模組 A，由 Topic Analyzer 先透過搜尋工具蒐集與分析主題相關資訊，並整理出重要概念與研究方向。接著由 Topic Planner 根據分析結果，產生完整的文章架構，包括章節標題與各小節項目。



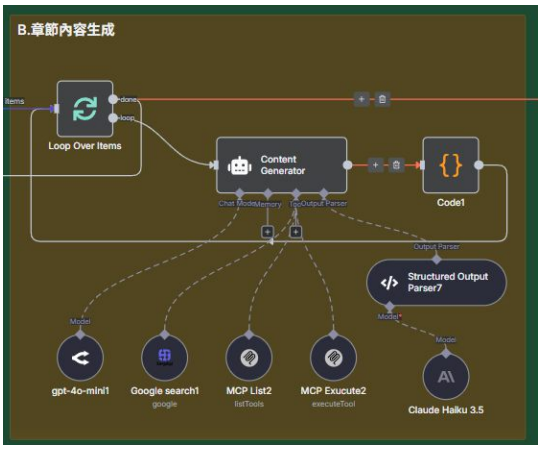
模組 A 的 n8n 實際流程圖



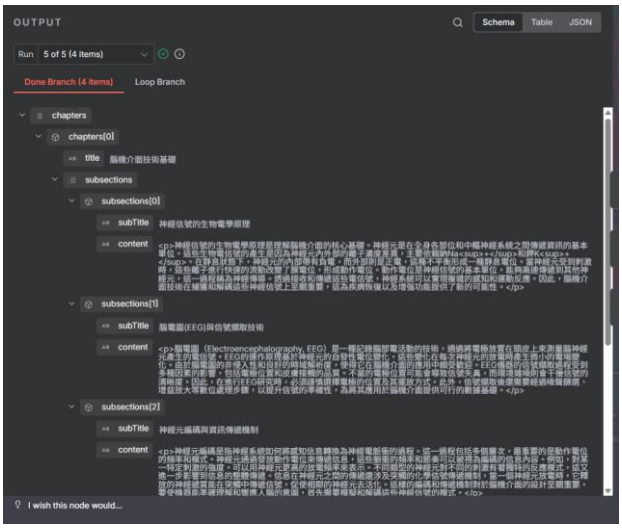
章節架構輸出結果

4.3 系統撰寫章節內容（模組 B、C、D 實作流程）

在完成文章架構後，系統會依序進入模組 B、C、D，開始撰寫各章節的實際內容。此階段以 Content Generator 作為核心，依照先前生成的章節標題與小節項目逐段撰寫。為了使內容更完整，系統會在撰寫前先透過搜尋工具(Serp API}及延伸查詢模組(MCP Tool)取得相關背景資訊、案例輔助等資料，確保文字具有充分的參考依據。

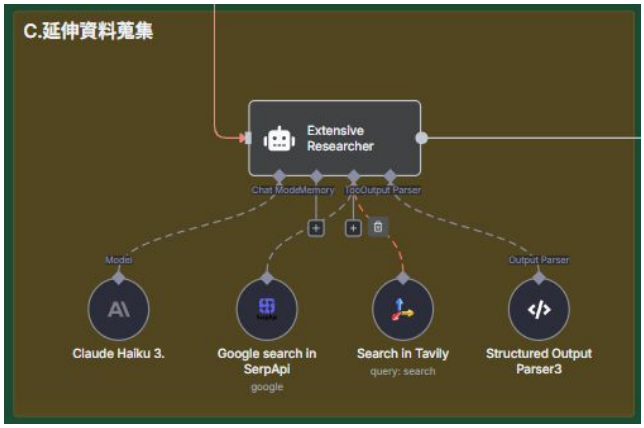


模組 B 的 n8n 實際流程圖



章節內容輸出結果

在初稿內容生成後，系統會進一步進入延伸內容補強模組（Extensive Researcher），針對每個章節進行補充查詢。此模組會透過 SerpAPI 與 Tavily 搜尋相關技術背景、概念來源、案例說明，並將取得的資訊整理成章節後段的延伸閱讀內容，補足初稿中的背景脈絡與參考來源，讓整體內容更加完整。

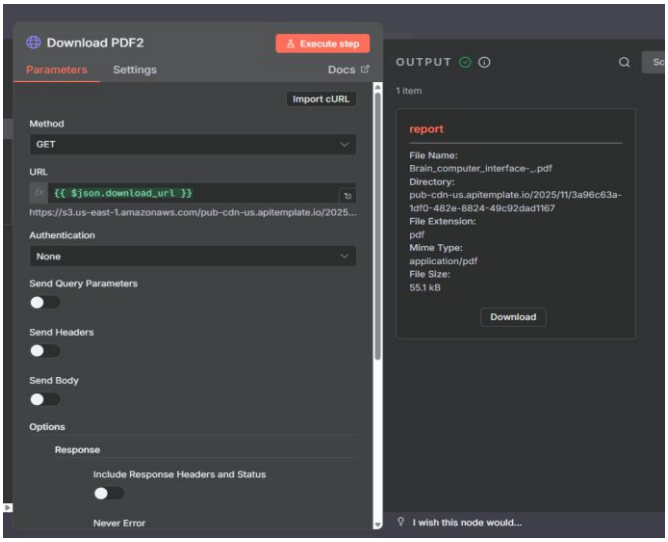


模組 C 的 n8n 實際流程圖

延伸閱讀輸出結果

最後，所有完成的章節內容會交由模組 D 進行格式統整，將內容轉換為統一的資料格式，並生成 PDF 研究報告完整檔。

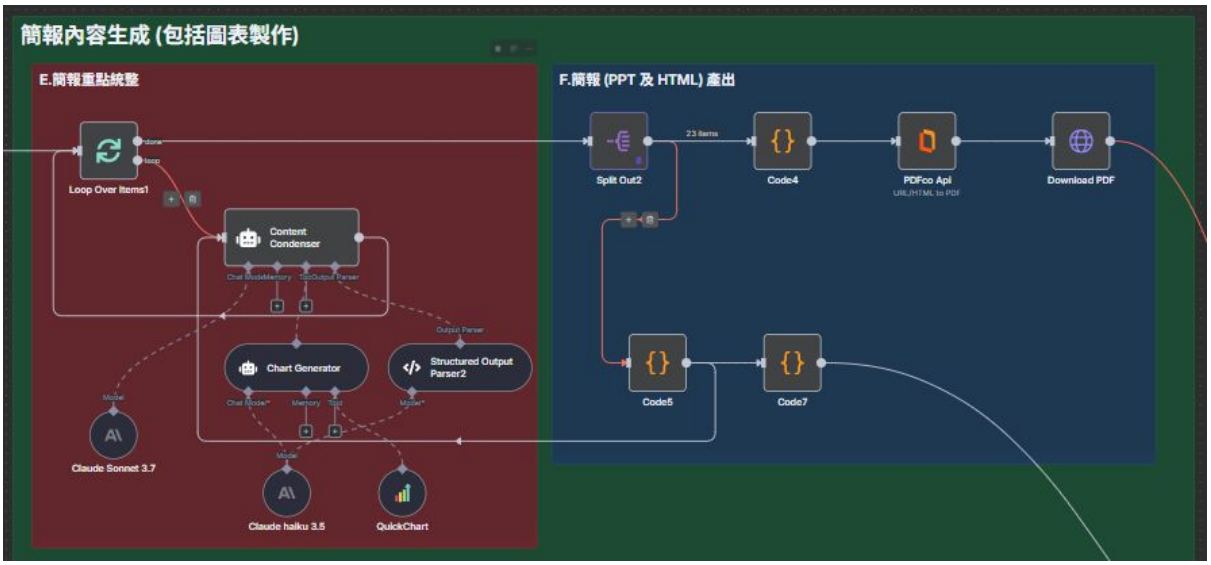
模組 D 的 n8n 實際流程圖



延伸閱讀輸出結果

4.4 系統生成簡報內容（模組 E、F 實作流程）

在研究報告內容完成並轉換為標準文件後，系統會進入模組 E 與 F，負責簡報重點整理與圖表生成。本階段的目的，是將篇幅較長的研究內容轉換為適合簡報呈現的形式，使使用者能以更精簡、直觀的方式展示研究成果。

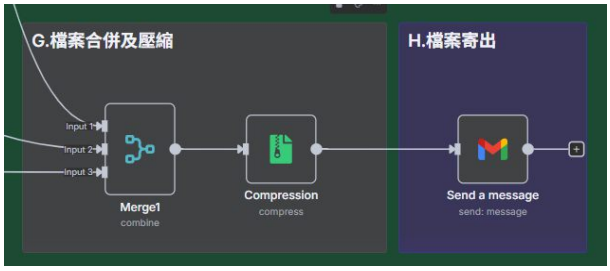


模組 E 及 F 的 n8n 實際流程圖

首先，由 Chart Generator 讀取完整研究報告內容，分析各段落中可視覺化的資訊結構，例如趨勢比較、分類架構、流程步驟或層級關係。系統會依據分析結果，透過 QuickChart 與事先定義的 Mermaid 模板，自動產生相對應的圖表內容，並輸出圖表描述與圖像檔。最後，由 **模組 F** 將所有處理後的內容套用到預先設定的 PPT 與 HTML Slide 模板中，並產出兩種最終版本的簡報檔案，提供使用者直接下載或展示。

4.5 系統整合、壓縮與成果寄送（模組 G、H 實作流程）

在研究報告（PDF）與簡報內容（PPT、HTML Slide）產出後，系統會進入成果整合階段，由模組 G 與模組 H 負責檔案整理與寄送。模組 G 會自動收集所有生成的檔案，統整至同一資料夾並進行命名與版本整理；接著模組 H 會將這些檔案壓縮成 ZIP 檔，並寄送至使用者於表單填寫的 Email。使用者可透過郵件中的下載連結一次取得所有最終成果。



模組 G 及 H 的 n8n 實際流程圖

5. 成果展示

Brain computer interface

章節 1：腦機介面概述

腦機介面的定義

腦機介面（Brain-Computer Interface，簡稱BCI）是一種創新的科技系統，能夠在人腦與外部設備之間建立直接的通訊管道。這種先進技術允許大腦的電氣活動直接與電腦或機器人設備進行即時互動，無需通過傳統的神經肌肉路徑。

BCI系統主要包含四個核心元件：神經信號採集裝置（如腦電圖電極）、信號放大電路、類比數位轉換器，以及用於解碼神經活動的計算演算法。這些技術元件共同工作，將腦部電氣信號轉換為可操作的數位指令，從而實現人腦直接控制外部設備的革命性目標。

根據信號採集方式，BCI可分為侵入性和非侵入性兩大類。非侵入性方法如腦電圖（EEG）使用貼附在頭皮上的電極，而侵入性方法則需要將電極直接植入腦組織，以獲得更高精度的神經信號。

腦機介面的歷史發展

腦機介面的發展歷程可追溯至20世紀60年代和70年代，當時科學家開始嘗試探索人腦與計算機的通訊可能性。早期研究主要集中在理解如何捕捉和解讀腦部電氣活動信號，這是一個充滿挑戰的科學探索過程。

一個重要的里程碑發生在2004年，Blackrock Neurotech成功將首個腦機介面設備植入患者體內，標誌著這項技術邁入實質性突破階段。隨後的幾十年間，技術不斷進步。2020年代更是見證了腦機介面領域的快速發展，包括Synchron在2022年為肌萎縮性側索硬化症（ALS）患者植入Stentrode腦機介面，以及Neuralink在2024年1月啟動首個人體臨床試驗。

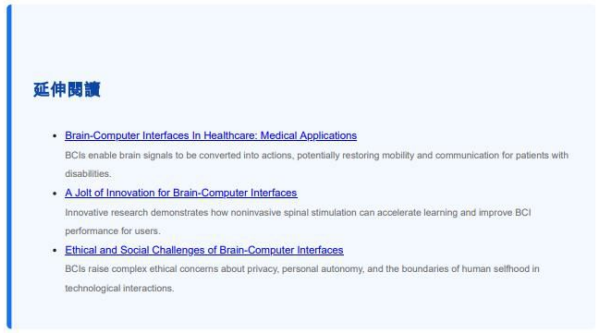
目前，像Blackrock Neurotech、Synchron和Neuralink等公司正在推動腦機介面技術的創新發展，已經成功在40多名患者身上實施了相關設備，為殘障人士重建功能、改善生活質量提供了前所未有的可能。

腦機介面的基本原理

腦機介面的基本原理圍繞著捕捉、分析和解讀神經信號這一核心過程展開。當人腦產生電氣活動時，這些微弱但具有重要信息的信號會被專門設計的電極捕捉。不同類型的電極（如表面電極或植入式電極）可以檢測不同精度和範圍的神經活動。

信號獲取後，需要經過複雜的處理流程。首先，信號會被放大和濾波，以消除雜訊並提取關鍵特徵。然後，先進的機器學習和神經網絡演算法會對這些信號進行解碼，將其轉換為可執行的指令。這個過程類似於將外語翻譯成可理解的母語，將腦部的『神經語言』轉換為機器可理解的『指令語言』。

人工智能的引入使得現代腦機介面變得更加智能和自適應。新一代系統能夠快速學習用戶的獨特腦electrical模式，顯著減少校準時間，並實現更直觀、更自然的人機交互。未來，這些技術有望在醫療康復、輔助科技和人機交互等領域發揮革命性作用。



延伸閱讀

- Brain-Computer Interfaces In Healthcare: Medical Applications
  BCIs enable brain signals to be converted into actions, potentially restoring mobility and communication for patients with disabilities.
- A Jolt of Innovation for Brain-Computer Interfaces
  Innovative research demonstrates how noninvasive spinal stimulation can accelerate learning and improve BCI performance for users.
- Ethical and Social Challenges of Brain-Computer Interfaces
  BCIs raise complex ethical concerns about privacy, personal autonomy, and the boundaries of human selfhood in technological interactions.

專業研究報告: Brain Computer Interface（可以開連結參考文檔）

PPT 及網頁檔

6.開發反思

在本次專題的開發過程中，我對流程自動化、資料整合與模組化架構設計有了比以往更深刻的理解。這套系統橫跨主題規劃、資料蒐集、段落撰寫、延伸閱讀整理、圖表生成、到最終成果輸出，每個階段都需要穩定銜接與一致的資料格式。也因為流程從簡單到逐步複雜，我更能體會到前期規劃的重要性，架構規劃越清楚，後續串接與擴充就越順利。

在後段的簡報與圖表生成流程中，我首次實作「從文字推導可視覺化資訊」的轉換方式。系統需要先分析研究報告的段落，再利用 Mermaid 與 QuickChart 產生相對應的視覺圖表。這個過程讓我發現，視覺化並不是將資料畫成圖而已，而是必須判斷資訊是否適合圖像呈現、是否具有代表性，以及圖表在簡報結構中的位置安排是否合理。同時，在內容生成的品質上，我也逐漸理解到提示語（prompt）設計的關鍵並非「寫越多越好」，而是在指令明確度、模型自由度與引用資料間取得平衡。

雖然 n8n 具有高度彈性與擴充性，但在大型流程中節點管理變得較為複雜，未來若要進一步提升可維護性，或許可考慮將常用模組包裝成子流程（Subworkflow），提升整體流程的整齊度與流程優化。

總體而言，此次開發不僅提升了我對資料流程與 AI 工具整合的掌握度，也讓我實際體會到從使用者需求到最終輸出之間的每一個細節環節，都需要嚴謹設計與不斷測試，才能確保成果是符合現實且實際的。

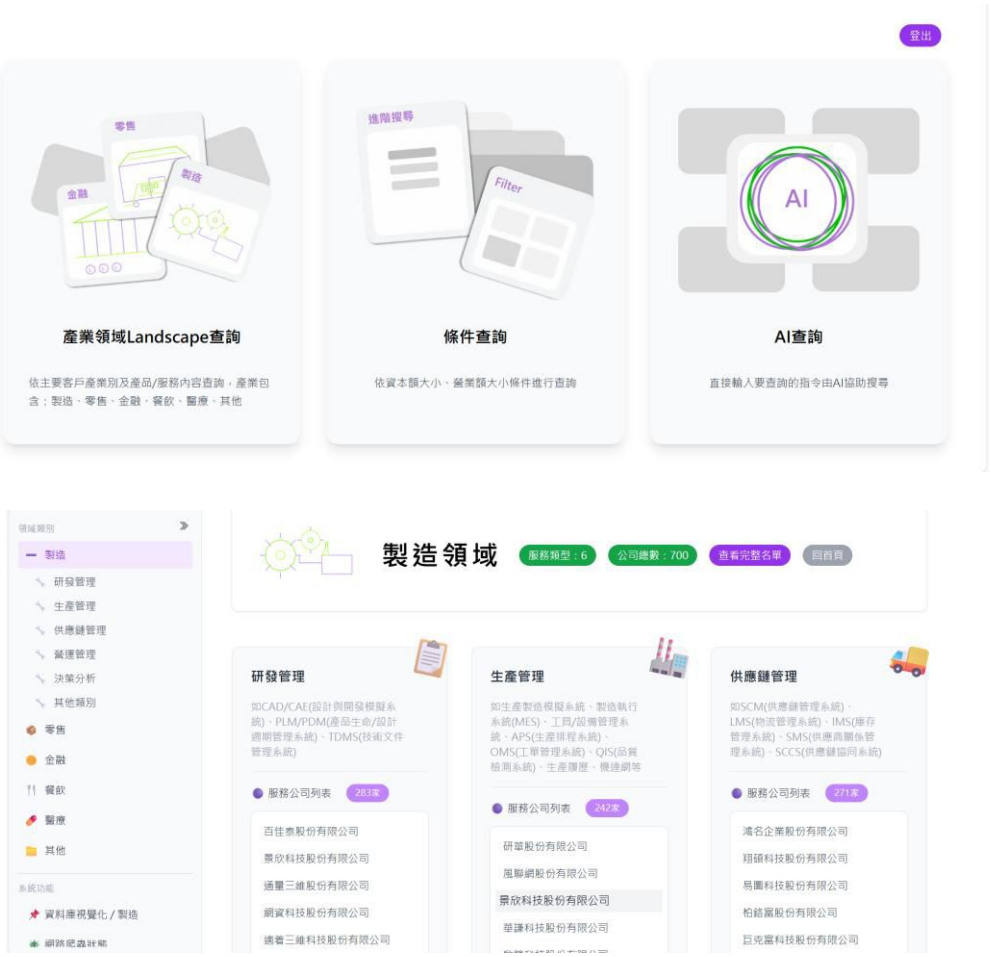
主題：LandScape（柏恩）

1. 系統設計與規劃

本系統旨在建構一個專為台灣科技產業打造的資料探勘平台。核心理念是以圖資料庫（Neo4j）取代傳統關聯式資料庫，以提升複雜關聯資料的查詢效率與語意理解能力。圖資料庫的結構天然適合關係密集的產業資料，使大型語言模型（LLM）能透過 RAG（Retrieval-Augmented Generation，檢索增強生成）更準確掌握資料脈絡。

使用者可以透過自然語言查詢資料，由系統以 GraphRAG 進行檢索與回應。同時，平台也保留傳統的關鍵字搜尋、條件過濾等查詢方式，提供多樣化的查詢體驗。在前端部分，我們採用 Neo4j 官方提供的 React SDK，讓使用者能以視覺化方式瀏覽資料節點與關聯，提升互動性與可理解性。

未來規劃包括整合即時網路爬蟲與使用者自有資料匯入功能，擴大資料來源，進而提升查詢的完整度與應用範圍。



landscape 主頁

2. 技術與方法

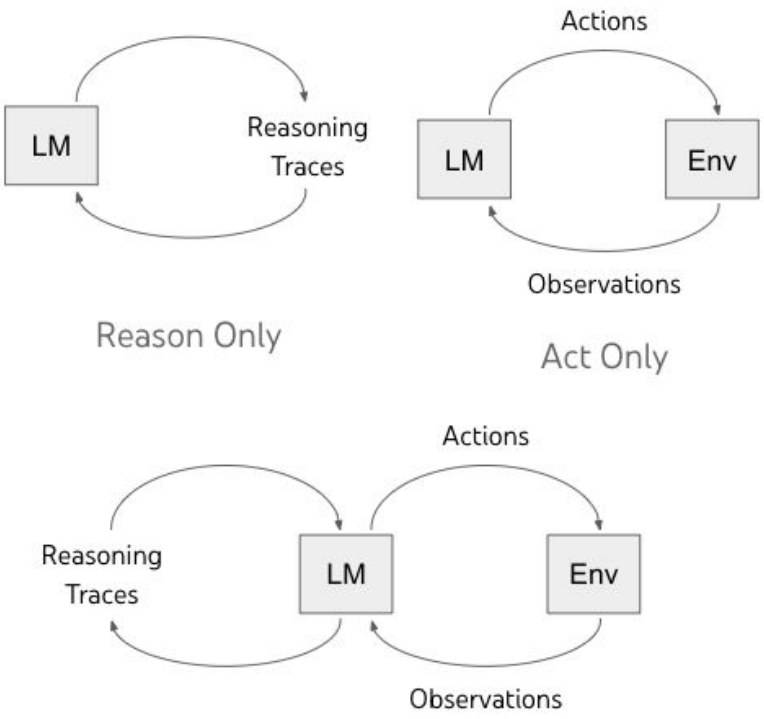
本章節說明系統核心功能：AI 智能查詢模組。此模組結合大型語言模型（LLM）的推理能力與圖資料庫的結構優勢，使系統能處理多步驟、跨資料來源的複雜查詢。與傳統 RAG 的單一流程（檢索 → 生成）不同，本系統採用 Agentic RAG，使模型能夠在有限的資料下規劃步驟、反思結果、使用多個工具，並反覆調整查詢策略，進而提升查詢品質。

2.1 ReAct Agent

ReAct（Reasoning + Acting）是本系統採用的關鍵技術之一。它讓 LLM 能在推理過程中同時產生：

- reasoning traces（推理軌跡）
- text actions（動作指令）

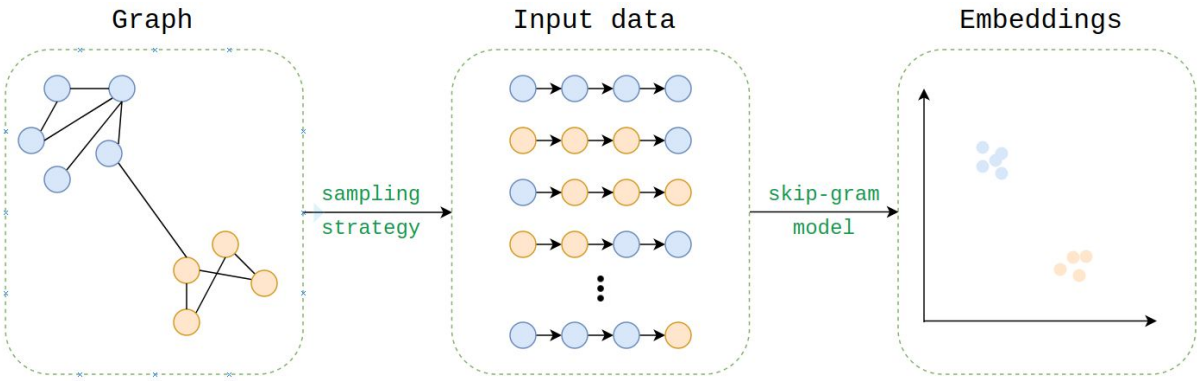
這種交錯式的推理方式使模型可以在查詢中反覆思考、修正策略，並從上下文中擷取更有效的資訊。ReAct 的優勢在於它能讓模型具有「規劃能力」，而不是被動執行單次查詢，對於多步驟任務尤其重要。



2.3 Vector Retriever

Node2Vec：將圖結構轉換成向量

Node2Vec 並非一般語言模型的文字嵌入，而是專為圖結構設計的向量方法。它透過隨機遊走模擬「節點在圖中的位置與鄰近關係」，再轉換成向量，讓系統可以計算節點之間的相似度。



Node2Vec

Vector Index：節點向量索引

系統為每個節點建立向量索引，使查詢時可以利用 cosine similarity 進行相似節點搜尋。此外，我們將查詢範圍設計為「以公司節點為中心，擴展 1-hop 的子圖」，使查詢既保留語意，又具備結構背景。

使用者不需輸入完全相同的關鍵字，也能透過語意相似度找到相關企業、技術或事件。

2.4 Cypher Retriever

CypherQAChain 是 LangChain 針對圖資料庫（Neo4j）設計的查詢模組。此模組能將使用者的自然語言問題轉換為 Cypher 語法，執行查詢後，再結合查詢結果與原始問題重新生成答案。

這種方式結合了：

- LLM 的語言理解能力
- Neo4j 的關聯查詢能力

讓使用者以自然語言查詢複雜的圖結構，而不需直接撰寫 Cypher。

3. 預期研究效益

本研究所建構的「台灣科技產業圖資料庫探勘系統」，在資料架構、查詢模式與人工智慧整合三方面均具創新性，預期可帶來以下幾項具體效益：

1. **提升知識查詢的效率與深度**
   相較於傳統關聯式資料庫，本系統以圖資料庫結構呈現產業知識脈絡，配合 LLM 與 GraphRAG 的結合，使使用者能以自然語言進行語意導向的查詢，不需熟悉資料庫語法即可取得複雜資訊。

2. **促進圖資料庫與生成式 AI 的融合應用**
   實踐圖資料與大型語言模型的整合查詢機制，具備高度的參考價值，未來可廣泛應用於金融、法律、醫療等多種領域的知識圖譜系統。

3. **建立可擴充、可模組化的查詢框架**
   本系統採用模組化設計，未來能輕易擴增資料來源（如網路爬蟲、用戶資料匯入），也能引入不同類型的 Agent 技術進行深層推理與自動化分析，具有高度擴充性。

4. 系統實作與成果展示

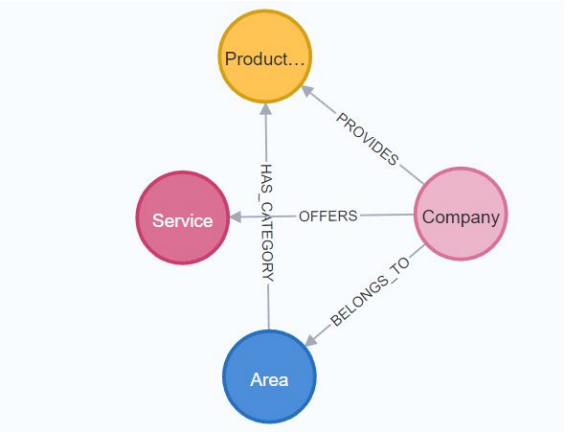
本系統的實作核心為 Neo4j 圖資料庫，搭配大型語言模型（LLM）、自然語言查詢介面與向量檢索機制，建置一套能針對台灣科技產業知識圖譜進行語意查詢與視覺探索的平台。系統分為後端架構、AI 查詢模組與前端介面三個部分，並整合多種技術以達成穩定且可擴充的查詢流程。

- **後端架構**：以 FastAPI 為主要框架，連接 Neo4j 圖資料庫與輕量化向量資料庫，支援 Cypher 查詢與向量檢索整合流程。
- **AI 查詢模組**：本系統的查詢核心採用 ReAct Agent，透過「推理＋動作」的交錯流程動態規劃查詢策略。Agent 在處理自然語言問題時會整合兩類 Retriever：**Cypher Retriever** 負責將問題轉為 Cypher 並回傳精確的圖資料；**Vector Retriever**（Node2Vec＋向量索引）則用於處理模糊查詢，透過節點向量與 1-hop 子圖語意空間比對相似度。Agent 會依照問題內容自動選擇或交替使用這兩種檢索方式，使查詢同時具備結構精準度與語意容錯能力，提升整體查詢品質與靈活度。
- **前端介面**：採用 Neo4j 官方 React SDK，提供圖形結構瀏覽、關鍵字篩選、自然語言查詢等功能，讓使用者能以視覺化方式探索企業關係與技術鏈結。

成果展示重點如下：

- **自然語言查詢**：使用者可輸入自然語言，系統可產生對應 Cypher 語法並回傳關聯企業結果。
- **圖形化展示**：查詢結果以互動式圖形節點展示，方便使用者進一步探索節點間關係。

自然語言查詢



圖形化展示

- **語意精準強化**：引入向量檢索提升模糊查詢精準度，尤其在公司名稱不完整或描述性查詢情境中效果良好。
- **模組擴充性驗證**：已完成用戶資料匯入測試，以及初步網路爬蟲串接，驗證系統具有良好的擴充能力。

5. 開發反思

在本次系統的開發過程中，我更加理解圖資料庫與大型語言模型結合後所能展現的真正潛力，也面臨了許多比預期更複雜的挑戰。原以為只要讓模型產生 Cypher 查詢就能完成自然語言查詢功能，但實作後才發現還涉及命名一致性、語意轉換精準度以及查詢上下文不足等問題。隨著資料量成長，模型輸入限制與查詢效率更成為核心瓶頸。

在這一年中，我隨著自己對 LLM、RAG、Agent 與圖資料庫的理解逐步深化，也同步持續迭代系統設計：從早期的固定模板式查詢流程，發展到目前能透過 Agent 動態選擇檢索策略的架構，使查詢更具彈性與智慧化。這段經驗讓我意識到資料建模、AI 模組與系統架構之間的緊密依存關係，也促使我開始思考如何構築更可擴充、可學習、可長期演進的知識系統。