



# HW#6 Line AI Chatbot

## 作業說明

- 本作業需實作一個整合 Line Messaging API 的智慧聊天機器人系統，包含兩大組件：
  1. Webhook 式的 AI Bot 後端（接收 Line 訊息、呼叫 LLM、回覆使用者）
  2. Chat 管理後台（監控對話、檢視統計、管理歷程）
- 至少串接一個 LLM 供應商（可任一vendor）。若配額/流量被限或服務不可用，系統須能優雅降級並回覆合適訊息。
- 主題不限，但需要能夠讓使用者跟你的 AI chatbot 互動 — either 將使用者的對話轉換成適當的 prompt 去傳給 LLM，或者是結合事先設計好的腳本，向使用者提供合理的回應。  
一些建議的題目方向 (當然，不以此為限)：
  - AI 個人小幫手：讓 chatbot 幫忙完成資訊的整理、紀錄、搜尋、問題回應等功能
  - AI 智能客服：幫一個網站、機構、服務建立一個 Line 官方帳號 (可不用付費)，擔任客服
  - 遊戲類：讓 AI 在遊戲中扮演一個玩家或是莊家的角色，讓遊戲更加有趣好玩



建議不要一次把 chatbot 的對話流程弄得太複雜，除了會讓功能的正確性可能較難收斂之外，也可能反而讓 UI/UX 變爛。  
應該先實現一個有基礎功能版本的機器人，再來慢慢加上去，確保整個 chatbot 的體驗可以維持在良好的水準。

- 請以 Next.js (with TypeScript) 開發並部署至 Vercel，確保可直接透過 Line app 來評測。

## 基本功能要求 (Must Have)

---

- ☐ Line Bot 對話/功能設計：
  - ☐ 主題
  - ☐ 功能列表
  - ☐ 對話腳本 (文字、各種 Line reply templates、in-app browser page、多媒體等)
  - ☐ 對話脈絡：在回覆時維持上下文，讓回應更連貫
  - ☐ LLM prompt template 設計
  - ☐ 回應設計：根據預設腳本 and/or LLM 回覆，包裝成適當的回應
- ☐ Line Bot server
  - ☐ 從 Line Messaging API 接收使用者的訊息 (文字, or payload in general)
  - ☐ 實現上述功能設計與程式邏輯
  - ☐ 透過預先設計腳本 and/or 向 LLM 詢問，產生合適的回應
  - ☐ API for Line Messaging webhook
  - ☐ 對話管理與統計
- ☐ Line Bot 設定：建立 Line 官方帳號並設定 Line Channel，開啟 webhook 端點
- ☐ 資料庫整合：將完整對話（時間戳、使用者資訊、平台、額外中繼資料）持久化儲存
- ☐ 基礎管理後台：可在網頁後台檢視對話紀錄並提供基本篩選
- ☐ 錯誤處理：LLM/外部服務失效時，提供明確、友善的降級回覆
- ☐ LLM 配額與速率限制處理：偵測 quota/429 等錯誤並以清楚訊息與合理 fallback 應對
- ☐ 即時更新：後台可即時看到新訊息/新會話

## 可選延伸 (Nice to Have)

---

- ☐ 使用 Bottender 套件串接 LINE API 與對話資料庫
- ☐ 進階篩選：可依使用者、日期區間、平台、訊息內容搜尋

- ☐ Session 管理：追蹤對話流程與狀態機
- ☐ 回應客製化：後台可調整 AI 人設與回覆規則
- ☐ 效能/健康監控：回應時間、失敗率與健康檢查端點
- ☐ 多平台支援：在保留 Line 的前提下拓展至其他平台 (Messenger/Discord/Slack/Telegram...)
- ☐ 速率限制：對外 API 實作節流/限流以防濫用
- ☐ Webhook 健康檢查：提供可監控的狀態檢查
- ☐ 批次作業：後台多選與批次刪除對話
- ☐ 使用者分析：顯示總對話數、活躍使用者數、趨勢等統計數據

## 技術要求與建議

---

- 必要：
  - Next.js (with TypeScript)
  - 資料庫：MongoDB Atlas (free tier) + Mongoose ODM (or any other SQL/NoSQL DB)
  - 部署至 Vercel
  - 串接 Line Messaging API
  - 串接至任一 LLM
  - 環境變數：dotenv 或等效方案，避免將密鑰放入版本控制
- 建議 (可自行視需求調整)：
  - 樣式：Tailwind CSS
  - 架構：服務層 (Service Layer) + 資料存取層 (Repository Pattern)
  - 驗證：Zod 或 Joi (請對請求與回應做驗證)
  - 錯誤處理與紀錄：集中式錯誤處理與結構化日誌
  - 程式品質：ESLint + Prettier (可採 Next.js 或 Airbnb 風格)

## 評分方式與標準

---

☐ 功能完整性

- 有效的 Line 機器人並可回應訊息，讀懂問答脈絡
- 成功串接 LLM 並保存對話
- 管理後台可檢視、即時更新與基本篩選

☐ 功能正確性與強健性

- LLM/外部服務錯誤時之優雅降級
- 清楚一致的錯誤訊息

☐ 使用體驗

- 線上服務穩定可用
- 後台操作流暢、資訊清晰

☐ 進階可選延伸功能（加分）

## 繳交內容與方式

---

1. 部署連結：

- a. 可運行之 LINE bot URL (and/or QR code)
- b. 管理後台的 Production URL (如需帳密登入，請提供註冊服務)

2. Line Bot 對話/功能設計：請參考「基礎要求」，將 chatbot 之對話/功能設計以 MD 檔的形式編寫並儲存成 "chatbot-design.md", 放置 hw6 根目錄 (i.e. 跟 README.md 同一目錄)

3. 原始碼：上傳至 GitHub (hw6) 完整可讀之程式碼（請確認未上傳機密，如 .env、密鑰、logs...）

4. 環境設定：在README中提供在地端開發與部署所需之設定說明

## 注意事項

---

- 請務必確保部署連結「可存取、可互動」，評測以線上版本為主。
- 請勿將任何敏感金鑰放入git版本控制；部署時請以環境變數注入。

## Reviewer 注意事項

- 請根據跟專案之 Line Bot 對話/功能設計 ("chatbot-design.md"), 檢查該 chatbot 功能
- 請不要進行壓力測試, 或是過度使用 chatbot, 以免太快造成 LLM quota 使用完畢而無法繼續測試
- 每份 reviewers 的數量、分數計算方式, 大致上 follow HW#5。請參考 HW#5 之規定

## Deadlines



由於這次作業說明比較晚發出公告, 而這次作業也有一定的難度, 故 deadline 會延後兩天到 11/26 (三) 9pm

- 作業繳交 deadline: 9pm, Wednesday, 11/26
- Review deadline: 9pm, Wednesday, 12/03



### HW#6 Q&As

如果對於 HW#6 有任何的問題, 請至 NTU COOL 討論區提問:

[https://cool.ntu.edu.tw/courses/55502/discussion\\_topics/450133](https://cool.ntu.edu.tw/courses/55502/discussion_topics/450133)



### HW#6 TA Hours

本次作業設有助教諮詢時間 TA Hour, 有需要的同學請致信

**[eewebprogramming@googlegroups.com](mailto:eewebprogramming@googlegroups.com)**。

來信標題請務必以 [HW6 TA Hour] 開頭, 並在內文詳述您遇到的困難或問題, 助教將會儘速與您協調時間進行解答。