HW1:

Source Reference

This project builds upon patterns and datasets related to the Spam Email problem from Chapter 3 of the Packt repository below. We used it to expand the preprocessing steps and add richer visualization work (step outputs, metrics, and CLI/Streamlit views).

https://github.com/PacktPublishing/Hands-On-Artificial-Intelligence-for-Cybersecurity.git

Using openspec and AI coding CLI to finish this project

Requirements：

1. need a github
   https://github.com/huanchen1107/2025ML-spamEmail

2. need a Demo site
   https://2025spamemail.streamlit.app/

[my GitHub websit]

[my Streamlit Demosite]

[執行模式，採二階段開發]

Phase 1：請 Gemini 根據 openspec 專案架構幫我產生初版

Phase 2：請 Gemini 升級我的專案

註 1：1st 是 openspec 搭配 GitHub Copilot，但 GitHub Copilot 不太聰明，執行過程不太像老師上課所教，及 AI 超元域的網路教學，放棄使用。

註 2：openspec 內建預設的 AI coding CLI 雖沒有 Gemini CLI，但我預選 GitHub Copilot 後，嘗試在 Termina 呼叫 Gemini，意外地發現 Gemini 也能依照 openspec 架構生成相關文件（如同老師上課所教，及 AI 超元域的網路教學），介紹如下：

[Phase 1：請 Gemini 根據 openspec 專案架構幫我產生初版]

1. 啟動 openspec 與 Gemini CLI 並請求產生專案提案的企劃案文件

圖 1: openspec init

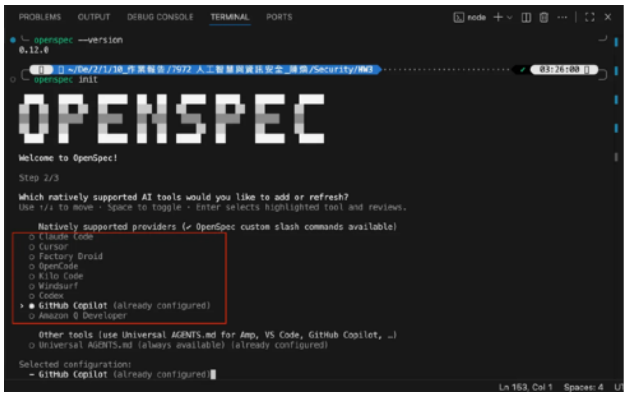

圖 2: Select GitHub Copilot

圖 3: Double confirm

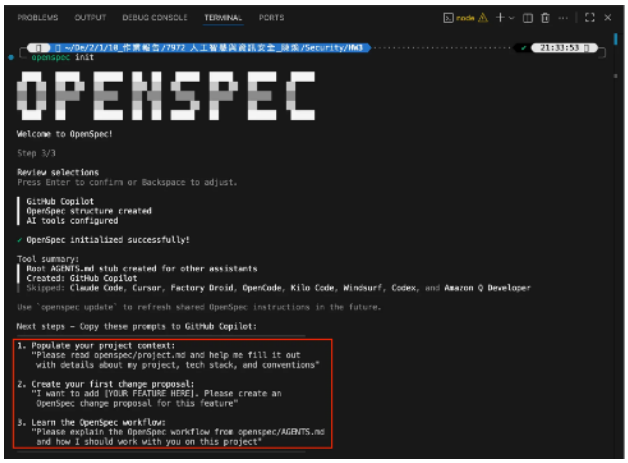

圖 4: Openspec 要求轉貼三個指令到 AI coding CLI

圖 5: 啟動 (呼叫) Gemini CLI

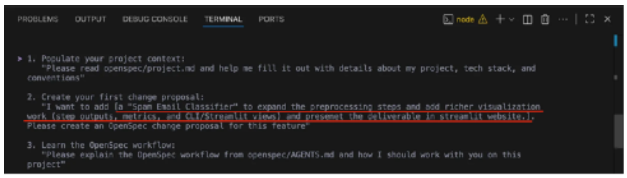

圖 6: 將 openspec 第二個指令的中括號 [YOUR FEATURE HERE] 改成我的專案內容 [a "Spam Email Classifier" to expand the preprocessing steps and add richer visualization work (step outputs, metrics, and CLI/Streamlit views) and presenet the deliverable in streamlit website.]。

如此，Gemini 更清楚我的專案需求，並依 openspec 架構自動產生專案提案文件：proposal.md、design.md、tasks.md、specs/spam-classifier

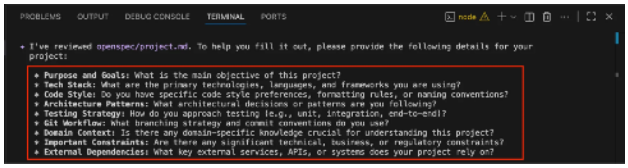

圖 7: Gemini CLI 列出 9 項問題要我澄清並回覆，以便它能更清楚掌握我的專案要求

圖 8: 我回覆給 Gemini CLI 的內容 (專案規範要求)

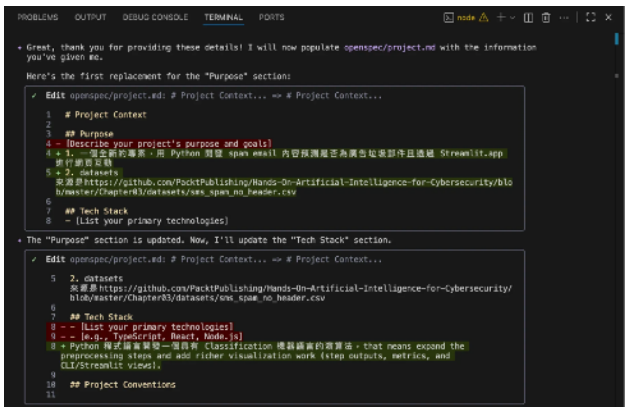

圖 9: Gemini CLI 接著會開始讀取分析並自動生成各資料夾或文件以符合專案需求，過程中會逐一再次詢問你所回覆的 9 項答覆 (專案規範要求) 的項次1、2

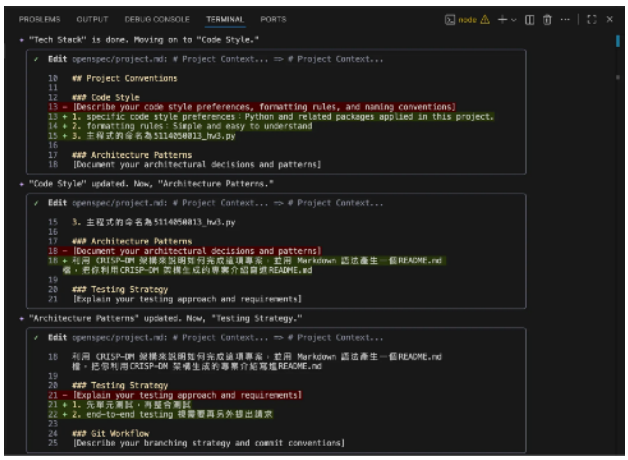

圖 10: Gemini CLI 逐一要我確認我回覆的 9 項答覆 (專案規範要求) 的項次 3～5

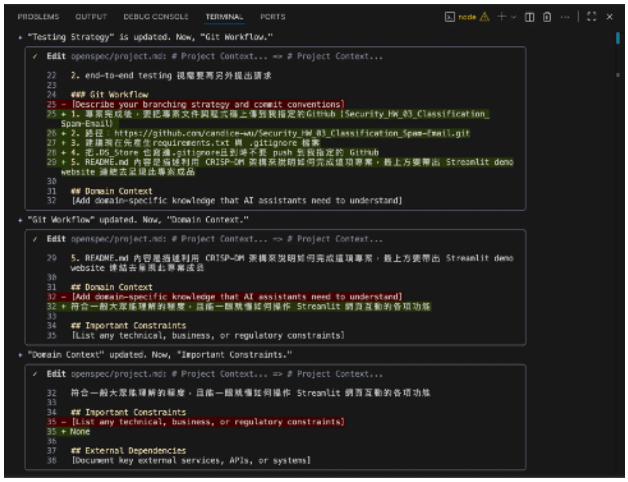

圖 11: Gemini CLI 逐一要我確認我回覆的 9 項答覆 (專案規範要求) 的項次 6～8

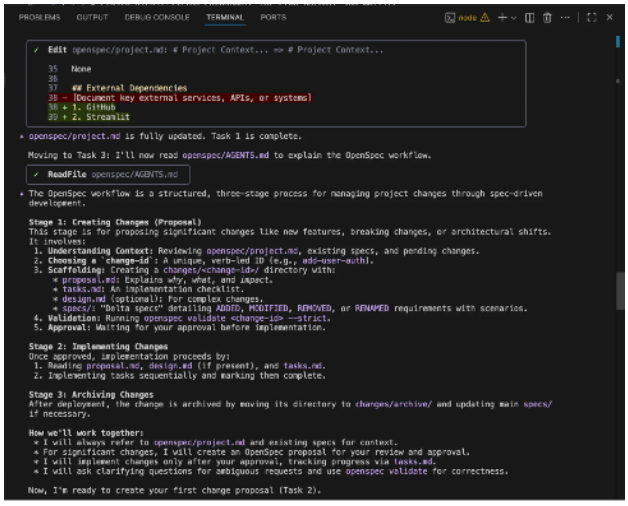

圖 12: Gemini CLI 逐一要我確認我回覆的 9 項答覆 (專案規範要求) 的項次 9。接著，它表示 Task 1 已完成，並主動先跳到 Task 3 解釋 OpenSpec workflow，最後表示說要開始進行 Task 2

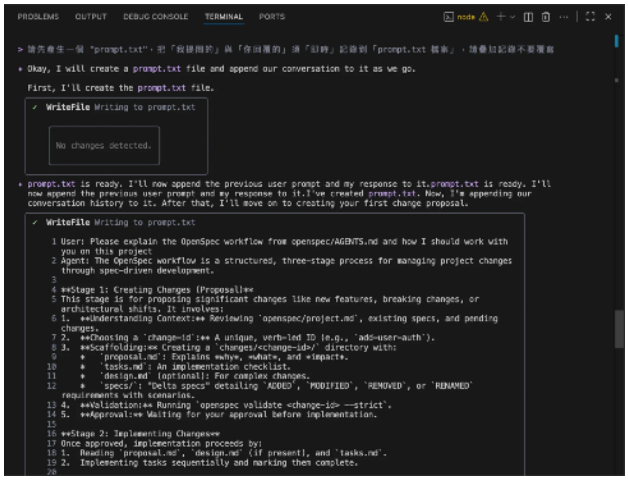

圖 13: Gemini CLI 接下來要進行 Task 2（create your first change proposal），但我先請它幫我先做對話紀錄並寫進 prompts.txt

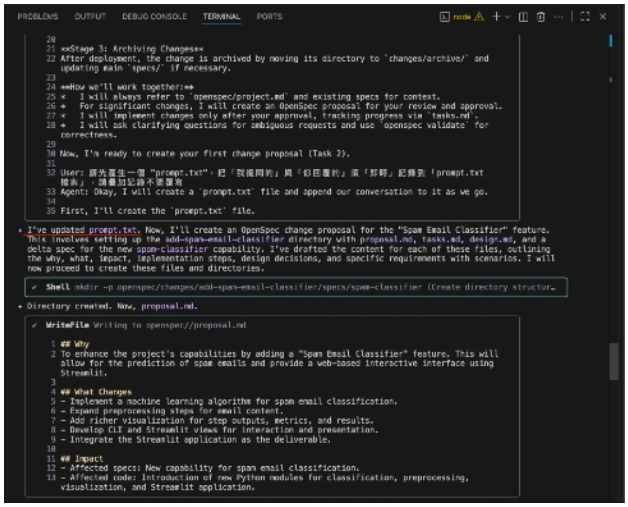

圖 14: Gemini CLI 幫忙生成對話記錄檔，也很聰明地把前面的對話自動回補到 prompts.txt。

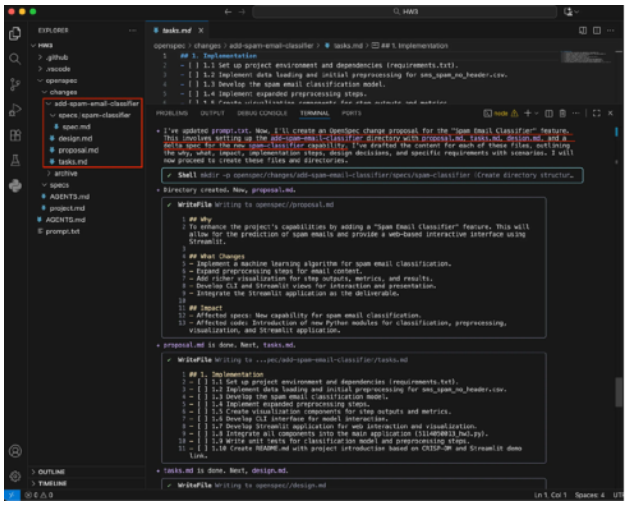

圖 15: Gemini CLI 完成對話記錄檔，自動回到 Task 2 依 openspec 架構去自動產生 OpenSpec change proposal，自動產生 add-spam-email-classifier 資料夾，底下有 proposal.md, tasks.md, design.md, and a delta spec，即自動地生成專案提案

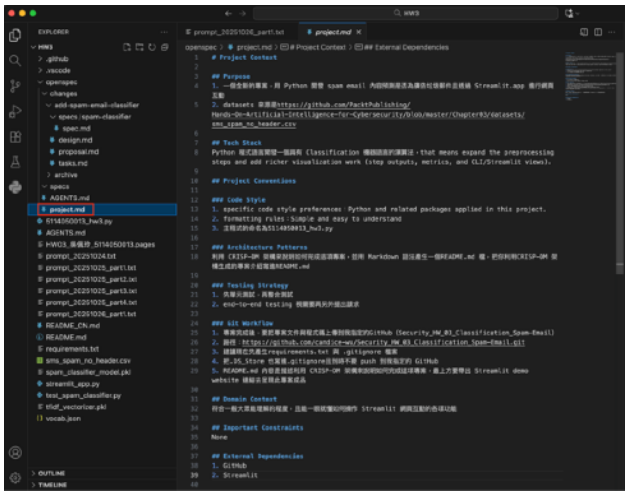

圖 16: 之前我回覆的9項專案規範要求，Gemini CLI 則產生一份 project.md 去記錄

註：可透過以下指令請 Gemini CLI 確認專案內容

指令	用途	常見參數 / 備註
openspec list	列出目前所有「變更 (changes)」資料夾和內容狀態（例如：哪些 change 尚未審核、哪些已完成）	
openspec show <資料夾名稱>	定義一個新的變更（例如 “add-spam-classifier”）後用來顯示詳細內容：Proposal、Tasks、Spec design 等	

openspec validate <資料夾名稱>	驗證指定變更的格式與結構是否正確（例如是否缺少 proposal.md、tasks.md、specs 目錄等）	
`openspec archive [--yes	-y]`	將已完成的變更封存（archive）並更新主規格（Specs）為最新狀態。

2. Gemini CLI 開始執行自動化程式開發

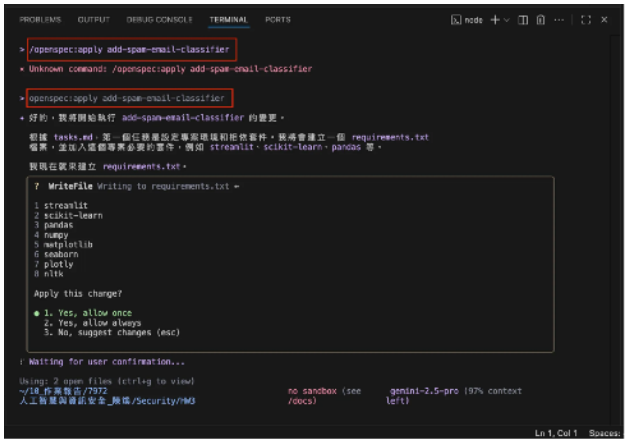	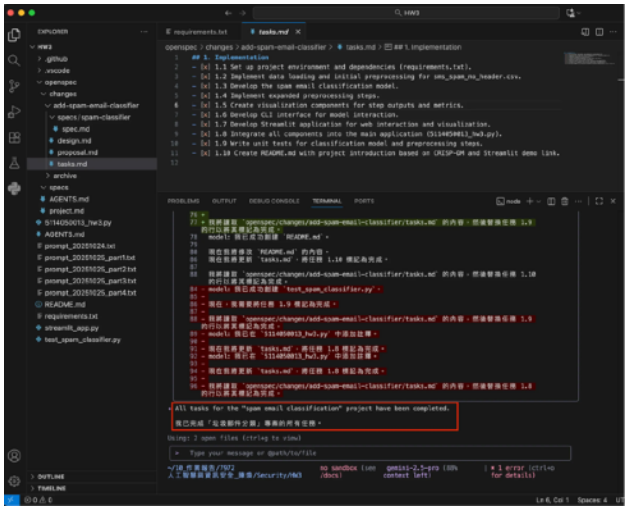
圖 1: 下開發指令： /openspec:apply add-spam-email-classifier 或 openspec:apply add-spam-email-classifier 註：加 slash 若失敗就不要加，再重下指令就會執行	圖 2: Gemini CLI 開發過程中會依照 task.md 清單逐步開發導入，完成的項目會打 X，最終完成十項

指令：/openspec:apply add-spam-email-classifier
或
openspec:apply add-spam-email-classifier

[Phase 2：請 Gemini 升級我的專案]

註：圖1～5同 Phase 1

圖 1: openspec init

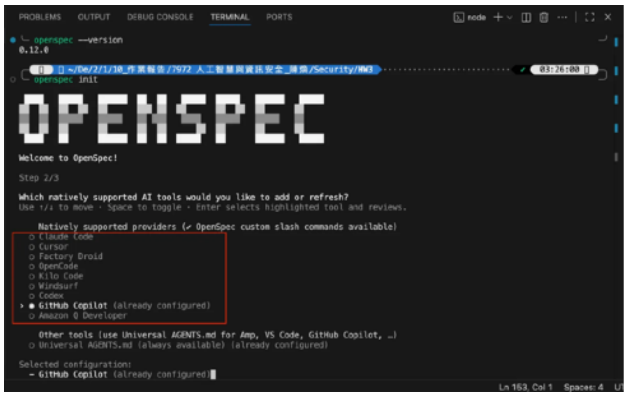

圖 2: Select GitHub Copilot

圖 3: Double confirm

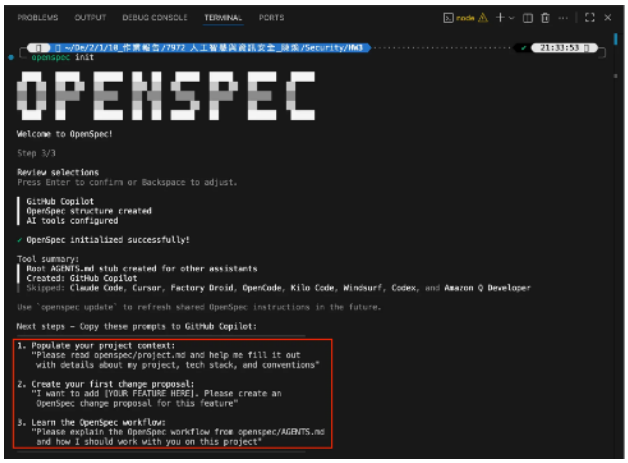

圖 4: Openspec 要求轉貼三個指令到 AI coding CLI

圖 5: 啟動 (呼叫) Gemini CLI

圖 6: 此時不用再轉貼 Openspec 要求的三指令，直接請 Gemini CLI 去讀取現有專案內容，請它幫忙做調整並提出建議

[Prompt]