

教學生 AI 輔助開發：教學現場工具與實作指南

蔡宗榮

臺南市綜合教研中心執行秘書

前言

我們都有這樣的共識：「未來的學習者，將不再是單純的知識接收者，而是能夠駕馭 AI、主動建構知識的探險家。」這更是當今教育現場的現實寫照。當人工智慧（AI）以驚人的速度滲透到我們生活的每個角落，教育工作者面臨一個關鍵課題：我們該如何教導學生，不懼怕 AI，而是將其視為一個強大工具，將他們從傳統的被動學習者，轉變為能夠駕馭知識、擁有深度學習能力的超級學習者？

AI 支援可以增強學習成果，人類夥伴可以提供更豐富的社交參與。隨著人工智慧能力的進步，教育工作者應該深思熟慮地整合這些工具，確保技術擴充輔助而不是取代有效程式教育中人際關係和技能發展的核心（Fan et al., 2025）。

本文介紹 AI 輔助開發（AI-assisted development），就是中學資訊科技相關課程及社團中，如何用 AI 來賦能軟體工程的開發過程。特別針對沒有程式基礎的人，筆者把許多步驟都一步步寫出來，特別是在教學時學生特別容易遇到的錯誤。

環境設定

在踏入 AI 輔助開發的世界前，首要任務是建置一個高效且順暢的開發環境。

一、核心工具介紹與安裝

- ❖ GitHub Copilot：作為目前市場上最普及的 AI 程式碼助手，GitHub Copilot 能夠提供即時的程式碼建議、自動完成程式碼片段，甚至生成完整的函數與檔案。訂閱方面，除了付費版本，學生與教師可以透過 GitHub Education 申請，享有免費的 Copilot Pro 服務。建議在提交相關證明文件（如學生證、教職證明）後，耐心等待約一週的審核時間。
- ❖ Git：儘管 AI 能夠生成程式碼，版本控制仍然是軟體開發的基石。Git 允許開發者追蹤程式碼的變更歷史，並在多人協作時有效管理不同版本的程式碼。Git 的安裝過程在不同作業系統上有所差異，本文以 Windows 為例，提供了詳細的下載與安裝指引，確保讀者能順利完成。
- ❖ Visual Studio Code (VSCode)：VSCode 以其輕量、高效及豐富的延伸模組（Extensions）生態系統，成為現代開發者的首選編輯器。在安裝時，建

議勾選所有選項以確保最佳體驗 (圖 1)。安裝完成後，務必安裝以下兩個關鍵延伸模組 (圖 2)：

- ❖ Chinese (Traditional) Language Pack for Visual Studio Code：讓介面顯示為繁體中文，降低初學者上手難度。
- ❖ Live Preview：提供即時網頁預覽功能，讓開發者在編寫前端程式時能立即看到成果，大幅提升開發效率。

圖 1

Visual Studio Code (VSCode) 安裝畫面，建議全部勾選安裝

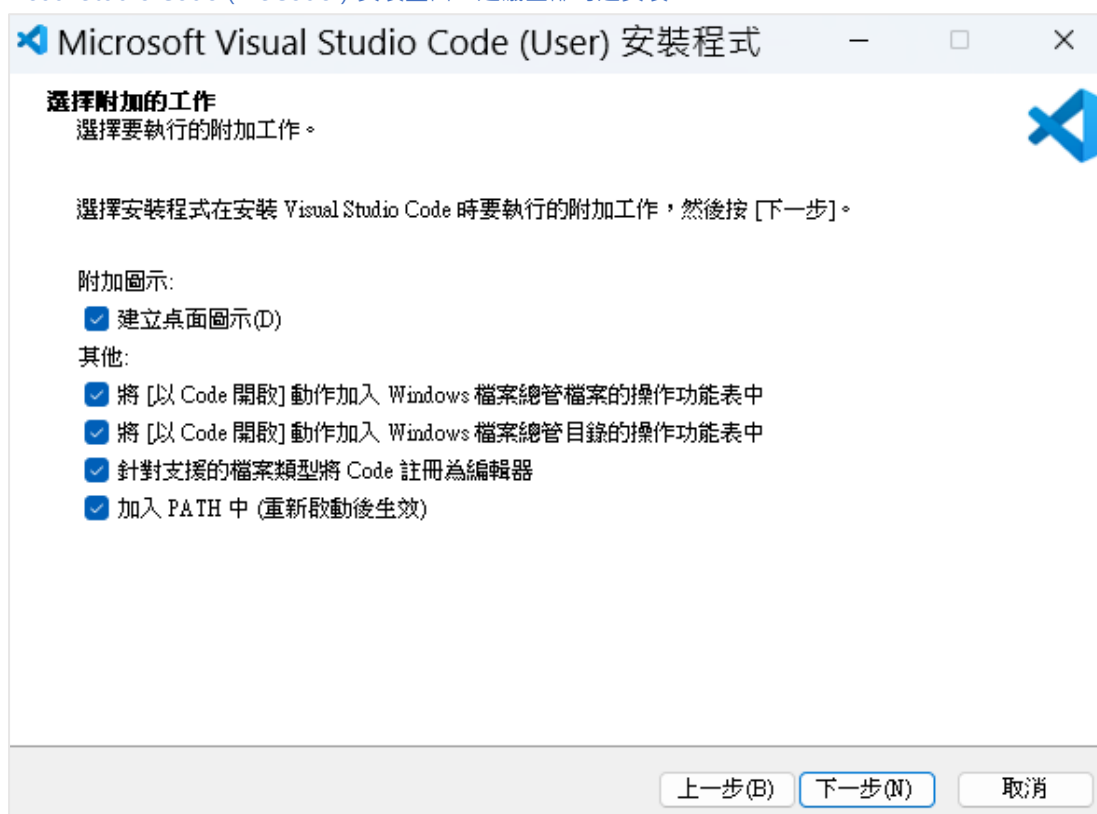
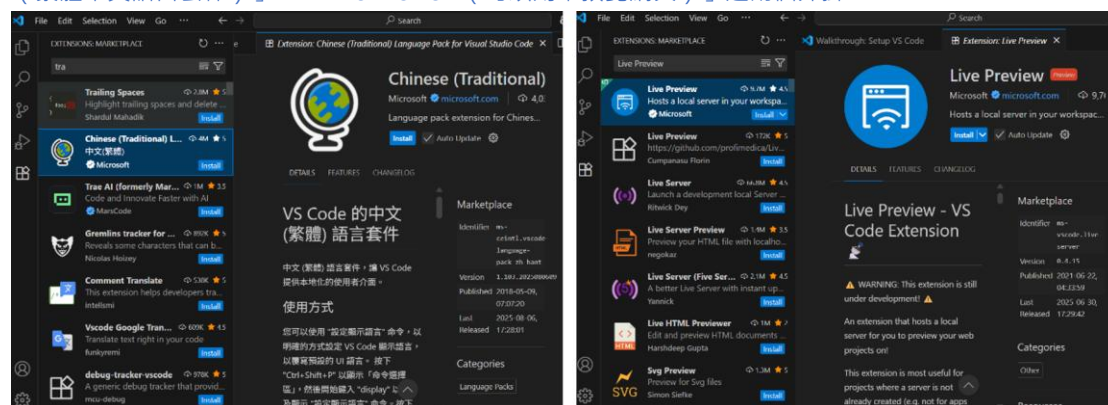


圖 2

Extensions (延伸模組) 搜尋安裝「Chinese (Traditional) Language Pack for Visual Studio Code (繁體中文語言套件)」、「Live Preview (可以用來預覽網頁)」這兩個外掛



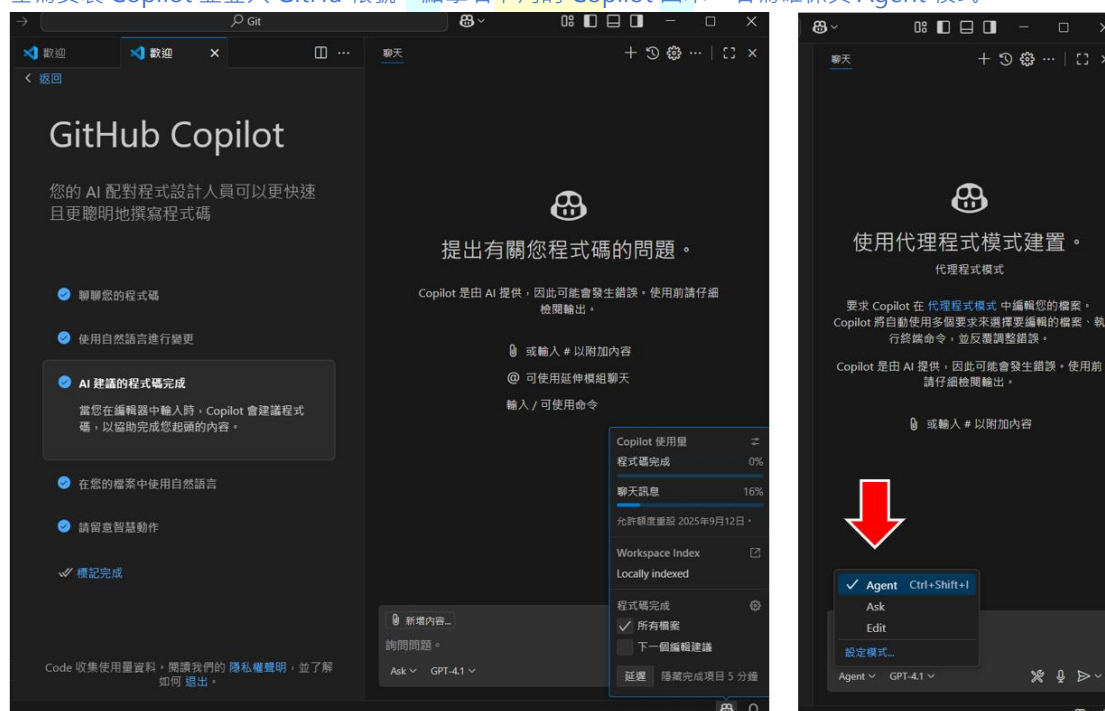
二、GitHub Copilot 的進階設定與模型選擇

在 VSCode 中，成功登入後，需進行以下關鍵設定（圖 3）：

- ❖ 切換至 Agent 模式：點擊右下角的 Copilot 圖示，並確保其處於「Agent 模式」。與傳統的「聊天模式」不同，Agent 模式賦予了 AI 更高的自主權，使其能夠理解複雜的指令並自動執行多個步驟，例如一次性生成多個檔案與目錄結構。
- ❖ 模型選擇：GitHub Copilot 內部整合了多個大型語言模型。根據筆者實測經驗，Claude 系列模型在程式碼邏輯與複雜度理解方面表現最為出色；Google Gemini 則緊追其後，在部分場景下有不俗表現；而 GPT 系列模型則可作為點數耗盡時的備用選項。使用者可以根據專案需求與個人偏好進行切換，以實現最佳的開發體驗。

圖 3

左為安裝 Copilot 並登入 GitHub 帳號，點擊右下角的 Copilot 圖示，右為確保其 Agent 模式



三、AI 輔助開發之核心思維與方法論

AI 輔助開發不僅是工具的改變，更是思維方式的轉變。本節將深入探討其背後的兩個核心概念：Vibe Coding 與 Agentic Coding。

- ❖ Vibe Coding（氛圍程式設計）：這是一種以人為主導的協作模式。開發者提

供意圖 (intent) ，AI 則根據程式碼上下文提供即時的程式碼建議。這種模式的優勢在於快速原型設計與探索性開發。當開發者腦海中只有一個模糊的想法，或者需要快速驗證某個技術可行性時，Vibe Coding 能夠幫助他們迅速將概念轉化為可視化的程式碼，建立「氛圍」般的開發流暢感。

- ❖ Agentic Coding (代理式程式設計)：這是一種更自動化的模式。開發者提供一個高層次的目標 (objective) ，例如「創建一個支援多人即時通訊的網站」，AI 代理則會自主地規劃、分工、執行，甚至自行除錯。這種模式的精髓在於解耦 (decoupling) 人類與程式碼實現的過程。開發者從具體的程式碼實現者轉變為「專案經理」，負責監控 AI 的進度並給予宏觀指導。Agentic Coding 尤其適用於大型、自動化的專案，能夠顯著降低重複勞動。

這兩種模式並非互斥，而是互補的。在專案初期，開發者可能採用 Vibe Coding 快速搭建原型；在進入詳細開發階段後，則可切換至 Agentic Coding，讓 AI 代理自動完成更多繁瑣的任務。

提示詞工程 (Prompt Engineering)

提示詞 (prompt) 是人類與 AI 代理溝通的橋樑，其品質直接決定了專案的成功與否。如果有人說他在 Google 查東西卻得不到他想要的結果，那可能是因為他用的關鍵字不夠精確，或是提問的方式不夠清楚。這個情況同樣可以套用在 AI 應用中，因此 Prompt Engineering 誕生了。

一、Prompt Engineering 簡介

“Prompt Engineering” 中文稱作「提示工程」，簡單來說 Prompt Engineering 意思是通過設計、改良，生成更精確的提問方式，讓 AI 更懂你的需求。一份有效的提示詞應具備以下三要素「契約-工具-需求」：

- ❖ 契約 (Contract)：這是專案的「長期記憶體」。AI 代理雖強大，但其上下文窗口 (context window) 有限。為了確保專案在多次會話間的一致性與連續性，我們需要建立一個契約檔案，如 AGENTS.md。這個檔案記載了專案的目標、進度與已完成事項。每次啟動專案時，只需提示 AI 「請閱讀 AGENTS.md 並繼續」，即可讓其無縫接軌。這種方法有效地解決了 AI 遺忘專案細節的問題，類似於檢索增強生成 (RAG) 在程式碼領域的應用。
- ❖ 工具 (Tools)：在專案開始前，明確指定 AI 應使用的技術堆疊 (tech stack)。例如，若要開發一個無需後端的靜態網頁，可以指示 AI 使用 CDN 版本的 JavaScript 套件並選擇輕量級的前端分散式資料庫如 GUN.js。這個步

驟不僅能確保 AI 選擇最適合的技術，還能讓開發者了解其背後的技术原理。經特別探討純前端技術的潛力，例如利用 WebAssembly、Web API（如 Three.js、WebRTC）等技術，可實現高性能與多功能的網頁應用。

- ❖ 需求 (Requirements)：這是提示詞的核心。有效的需求描述應明確、具體、可量化。初學者常犯的錯誤是給予模糊的指令，導致 AI 產出不符合預期的結果。建議採用螺旋式開發 (spiral development) 模式，即逐步引導 AI。從一個簡單的原型開始，然後在每次迭代中，根據 AI 產出的結果進行更精準的描述與重構。這種方法不僅降低了單次失敗的風險，也培養了開發者將抽象需求轉化為具體邏輯的運算思維。

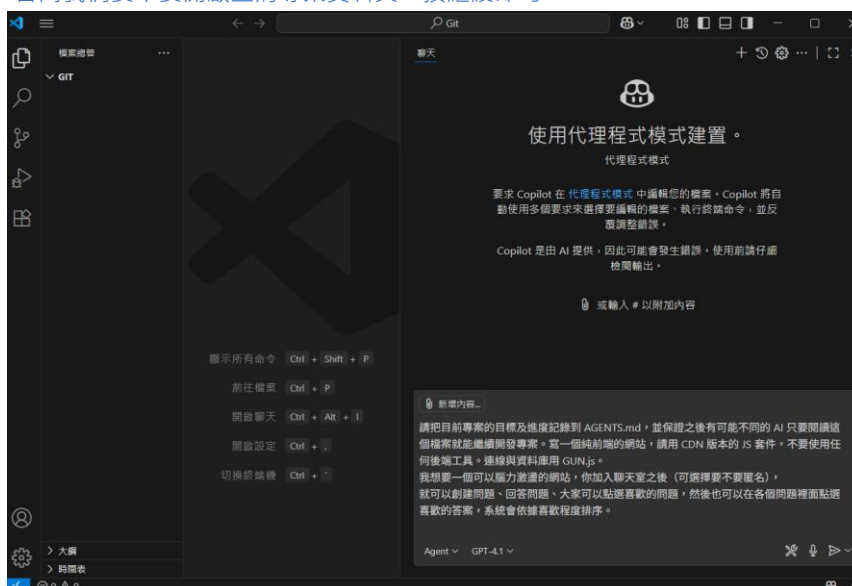
二、實作

範例：「請把目前專案的目標及進度記錄到 AGENTS.md，並保證之後有可能不同的 AI 只要閱讀這個檔案就能繼續開發專案。寫一個純前端的網站，請用 CDN 版本的 JS 套件，不要使用任何後端工具。連線與資料庫用 GUN.js。我想要一個可以腦力激盪的網站，你加入聊天室之後（可選擇要不要匿名），就可以創建問題、回答問題、大家可以點選喜歡的問題，然後也可以在各個問題裡面點選喜歡的答案，系統會依據喜歡程度排序。」

如圖 4，按繼續設定就可以了！接著欣賞 AI 如何寫程式。等 AI 完成之後，就可以找到 index.html 這個檔案，這是我們網站的首頁，開啟檔案之後，點擊右上方的圖示可以預覽網頁，執行到此，學生就完成 AI 輔助開發的第一步了（圖 5-6）。

圖 4

在旁邊的 GitHub Copilot 聊天視窗輸入初始專案的提示詞 (Prompt)，接著 AI 會問我們要不要開啟空的專案資料夾，按繼續即可



後續還可以加深加廣教學生：初始化存放庫、版本提交與控制、發佈專案、CI/CD（持續整合/持續部署）、分工架構與解耦（decouple）等等。

圖 5

Live Preview 功能有提供一些工具，點擊右上方三條線的工具箱，就可以在瀏覽器中開啟網頁，這在測試多人連線的時候很好用

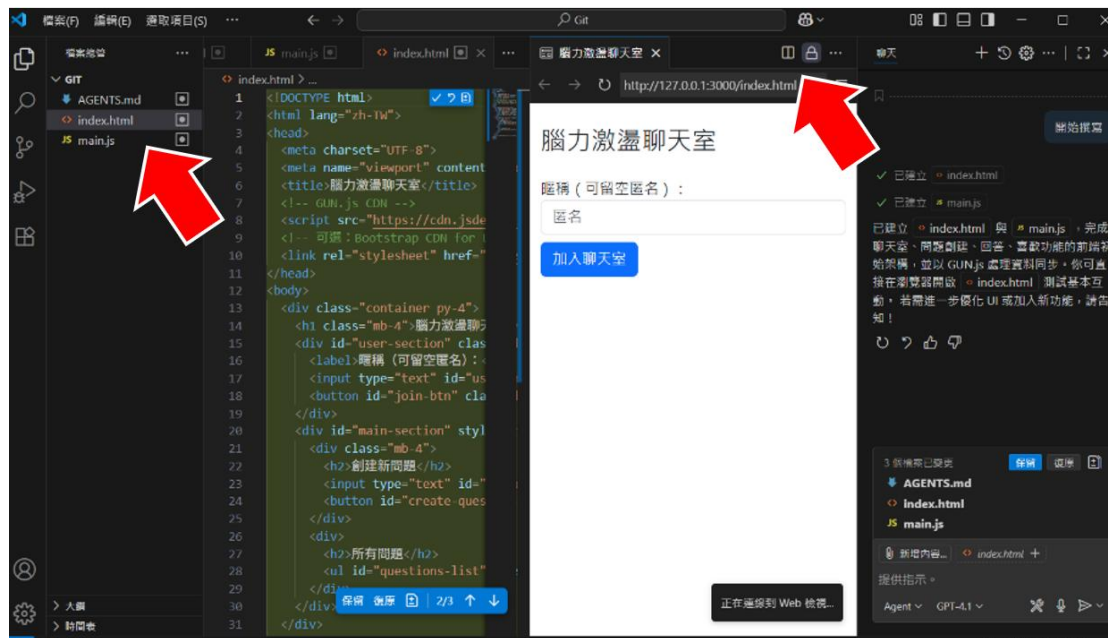
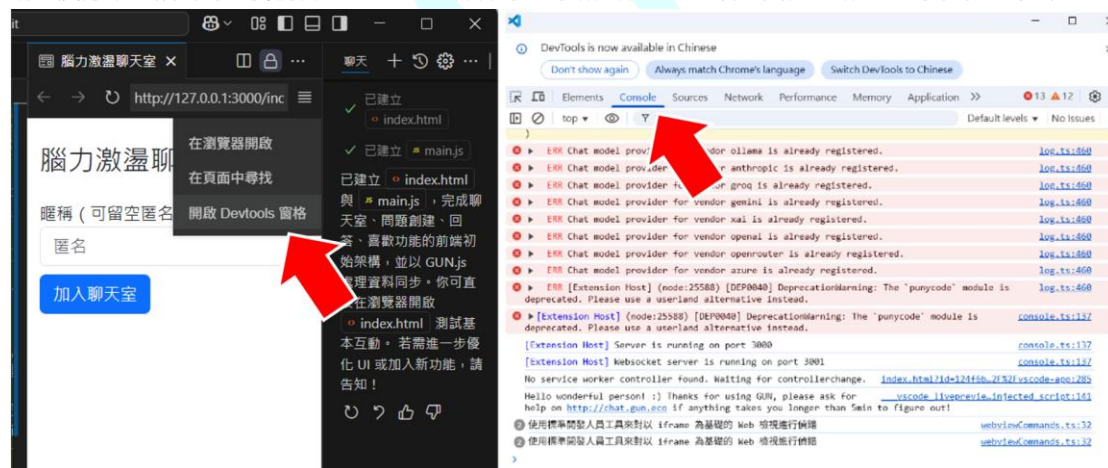


圖 6

網頁反應不如預期時，我們會去 Console 看看有沒有錯誤訊息，可以直接複製給 AI，尋求解決方法



教學結果與省思

課程實施對象為國中階段 7 年級 511 名學生，於下學期資訊科技課程及社團課進行，藉由教師引導，採取差異化教學搭配媒體輔助，92%以上的一般學生可以完成基礎的 AI 輔助開發。本文提供的 AI 輔助開發課堂指南，從概念思維到具體實踐，系統性地闡述了其核心價值。我們可以看到，AI 不僅是加速程式碼編寫的工具，更是改變整個軟體工程生態系統的關鍵驅動力。

課程實施後，筆者綜合整理學生回饋及教師教學省思，得到 AI 輔助開發未來將呈現以下趨勢：

- 一、更強大的 AI 代理：未來的 AI 代理將更具備自主學習與適應能力，能夠處理更為複雜的專案，甚至能與其他 AI 代理協作，形成一個「代理社會」。
- 二、人機協作的新型態：程式開發將不再是單純的人類活動，而是人機協作的混合模式。開發者將轉型為更高層次的「架構師」與「產品經理」，專注於定義問題、設計架構與維護核心價值，而將程式碼實現的細節交由 AI 完成。
- 三、教育與人才培養：未來的程式教育將不再側重於程式碼語法記憶，而是更強調運算思維、邏輯推導與問題解決能力。學生需要學習如何有效地與 AI 溝通，並利用其作為強大的生產力工具。

AI 輔助開發是一場不可逆的變革。在科技領域教導學生掌握這項技術，不僅能提升工作效率，更能在未來保持競爭力，更符合課綱所提及的在透過營造適性與友善的學習環境，使每一位孩子都能具備基本的科技素養，並且在支持的環境下，因應科技發展帶來的新世代生活方式，掌握、分析、運用科技的能力，啟發與開展孩子的天賦，為自己開創一個充滿可能性的全新道路。

參考文獻

教育部 (2018)。十二年國民基本教育課程綱要國民中小學暨普通型高級中等學校-科技領域。 <https://reurl.cc/oYGRl3>

Fan, G., Liu, D., Zhang, R., Pan, L. (2025). The impact of AI-assisted pair programming on student motivation, programming anxiety, collaborative learning, and programming performance: A comparative study with traditional pair programming and individual approaches. *International Journal of STEM Education*, 12 (16) . <https://reurl.cc/A3ZjGp>

