# twMVC #51

## 《GitHub Copilot 徹底改變開發模式，探索 AI 驅動的智慧程式碼協作》




歡迎參加我們的每週四固定聚會

# GitHub Copilot 徹底改變開發模式，探索 AI 驅動的智慧程式碼協作

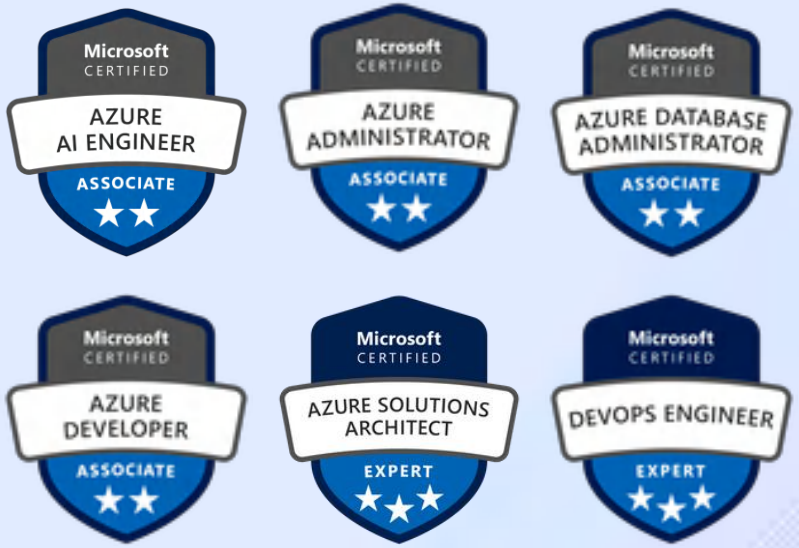
AnYun Liao

# 講師資訊



## 黯雲 AnYun Liao

### 資深雲端技術顧問




「黯雲端記事錄」部落格（https://dotblogs.com.tw/anyun）作者，主要分享關於開發以及 Azure 相關的主題，有多張微軟 Azure 和 AI 相關證照，擅長微軟解決方案的開發技術和 Microsoft Azure 關於成本管理、AI 服務等議題。

# 講師資訊



- 黯雲端記事錄 (https://dotblogs.com.tw/anyun)
- 黯雲端記事錄 (https://www.facebook.com/AnYunCloudNote)
- GitHub (https://github.com/anyun)

CONTENTS


GitHub Copilot 基本介紹 01

GitHub Copilot 基本用法 02

GitHub Copilot Enterprise 03

# GitHub Copilot 基本介紹

# GitHub Copilot 基本介紹

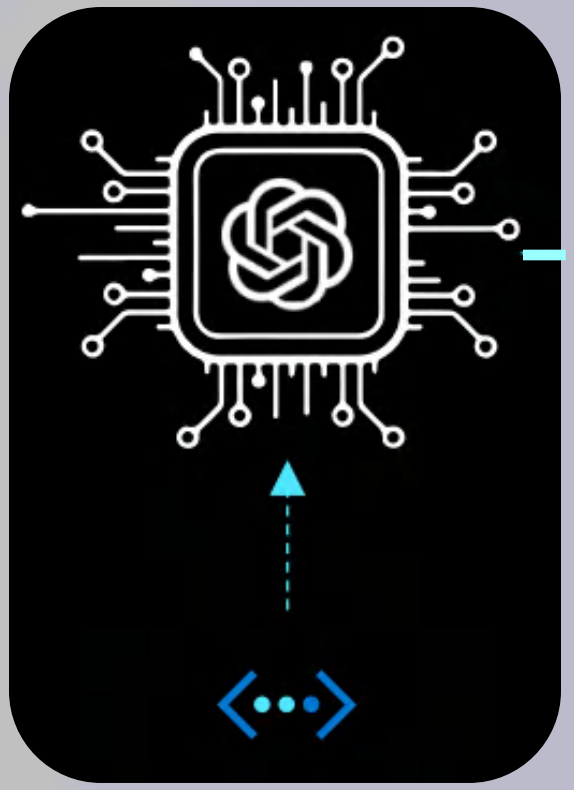
- Your AI pair programmer
- Pair programming



OpenAI 模型

GitHub

專用程式碼

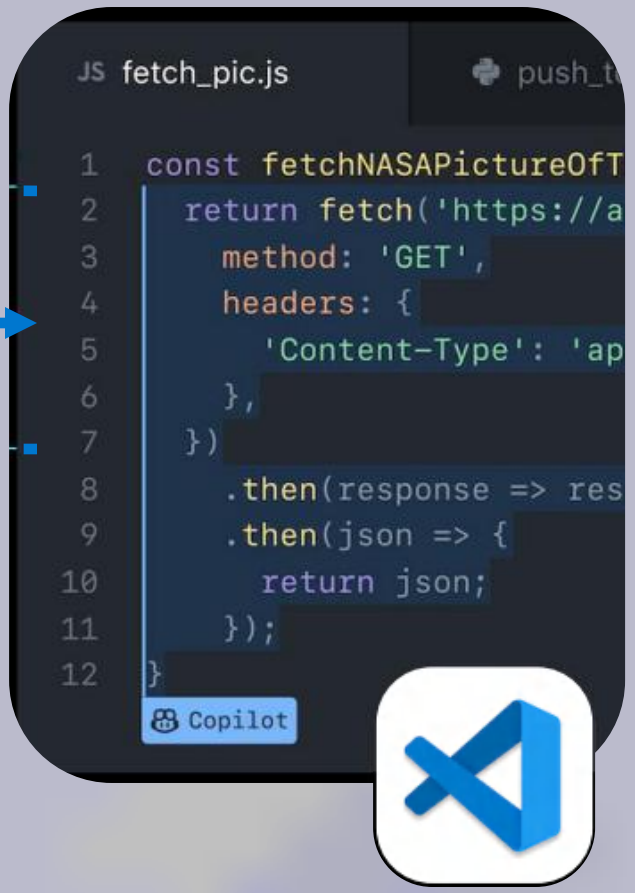





提供文字編輯

提供建議

改善建議




網路上的文字
和公開的程式碼

# GitHub Copilot 環境設定




- Visual Studio 2022 17.10 或以上的版本內建套件
- Visual Studio 2022 17.8 ~ 17.9 透過擴充功能安裝




- Visual Studio Code Marketplace 安裝




- Vim 版本 9.0.0185
- Neovim 版本 0.6 或更高版
- Node.js 版本 18 或更高版本

# GitHub Copilot 環境設定




**JetBrains IDEs**

- IntelliJ IDEA (Ultimate, Community, Educational)
- Android Studio
- AppCode
- CLion
- Code With Me Guest
- DataGrip
- DataSpell
- GoLand
- JetBrains Client
- MPS
- PhpStorm
- PyCharm (Professional, Community, Educational)
- Rider
- RubyMine
- RustRover
- WebStorm



**Xcode**

- Xcode 版本 8.0 或更高版本以及 macOS Monterey (12.0)

# GitHub Copilot 基本介紹

**46%**

的新程式碼現在
由 AI 撰寫

**55%**

整體開發人員生
產力快了

**74%**

的開發人員覺得
可更專注在帶來
滿足感的工作

# GitHub Copilot




## 維護階段

- 程式碼重構
- 程式碼轉換
- 解釋程式碼
- 程式碼註解
- 命令和腳本說明

## 測試階段

- 單元測試
- 找出程式問題
- 除錯
- Code review

## 開發階段

- 將註解轉成程式碼
- 自動根據上下文，建議程式碼
- 給十個建議

## 生成程式碼

- 自動產生程式
  - 書本上有的
  - 書本上沒有的
- 產生數個建議的程式碼
- 轉換程式碼語言
  - 轉換成指令
  - 轉換成文件
  - Oracle SQL to T-SQL
- 根據公司的Coding style生成程式碼
- 使用不同版本的SDK, 來生成程式碼
- 根據API文件產生代碼
- 產生韌體代碼與Makefile
- Reference Code

## 測試程式碼

測試程式碼

- 生成測試程式碼
  - 提升測試覆蓋度
  - 提升測試複雜度
- Test-Driven Development (TDD)

## 優化程式碼

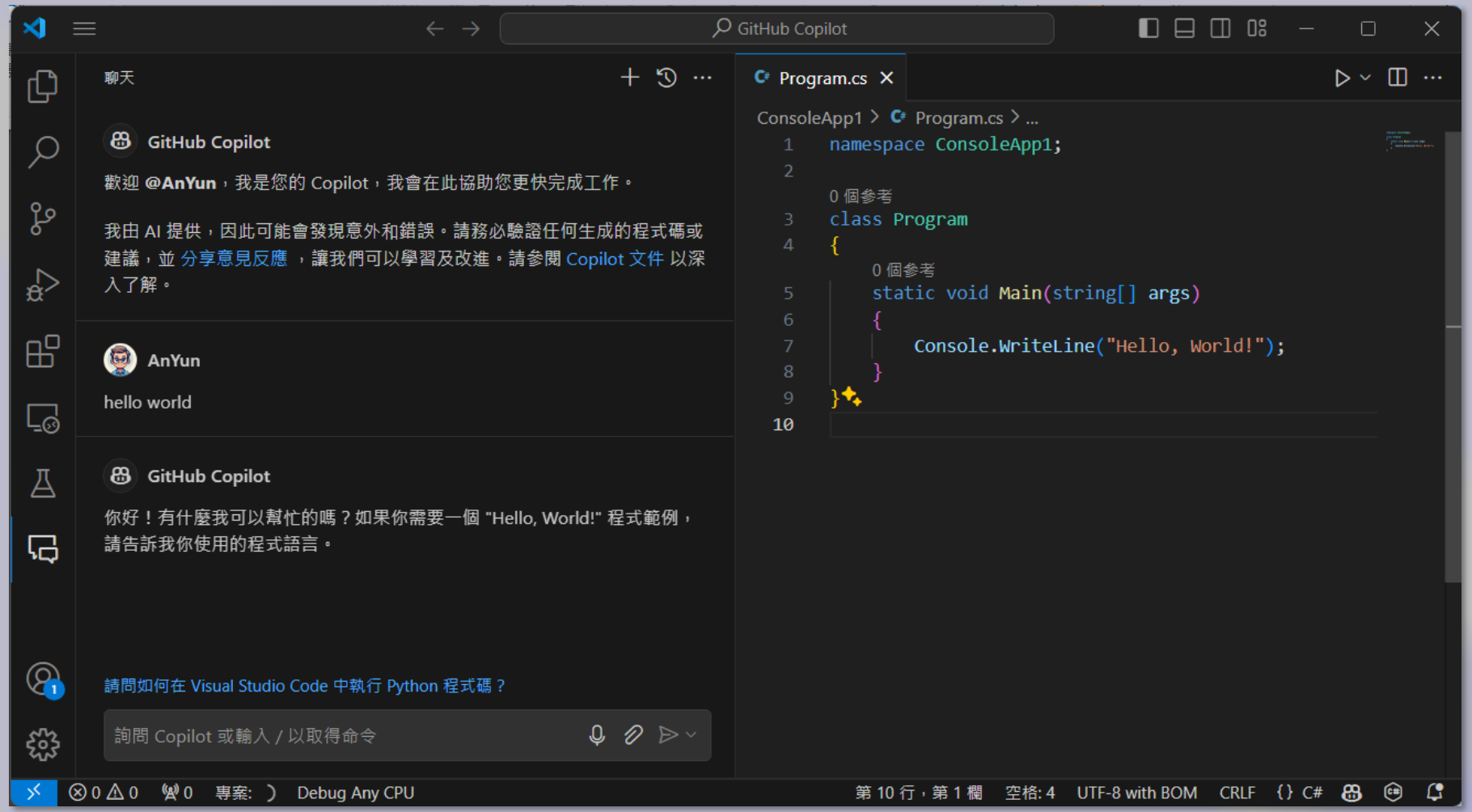
- 解釋程式碼行為
- 解釋看不懂的程式碼
- 生成註解文件
- 加入除錯的資訊
- 檢查程式碼錯誤
- 優化或重構程式碼
- SQL效能調教
- 程式碼分檔
- 檢查程式碼安全性
  - XSS (Cross-Site Scripting)
  - CSRF (Cross-Site Request Forgery)
- SQL Injection
- 檢查Memory Leak and Segment fault
- 時間複雜度與空間複雜度

## 其他

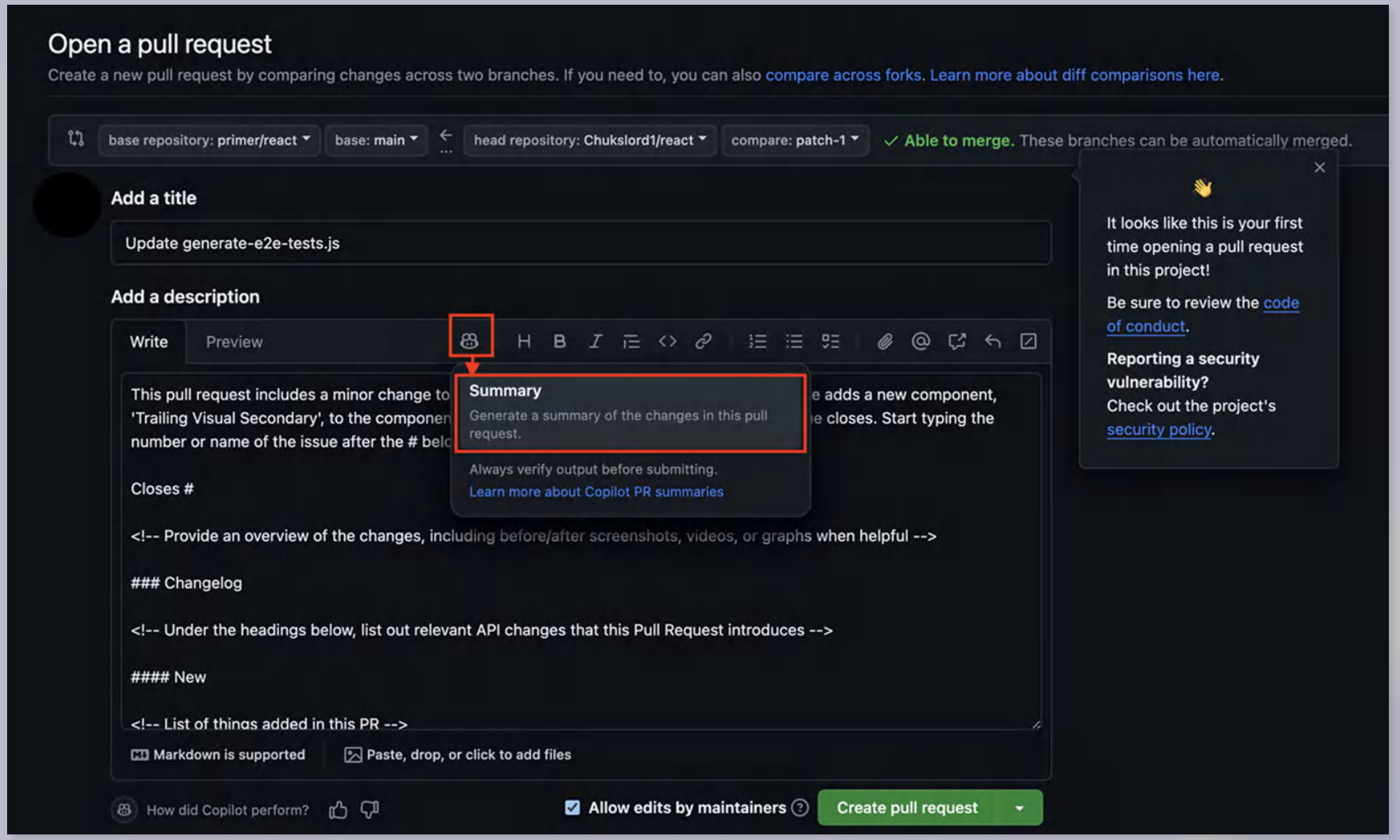
其他

- Docker與K8S語法
- Terraform
- ARM (Azure Resource Management), Bicep
- Azure Pipeline
- 快速學習各種技術內容
- ABAP開發 (SAP)
- HTML開發
- Databricks開發
- 跟AI對話

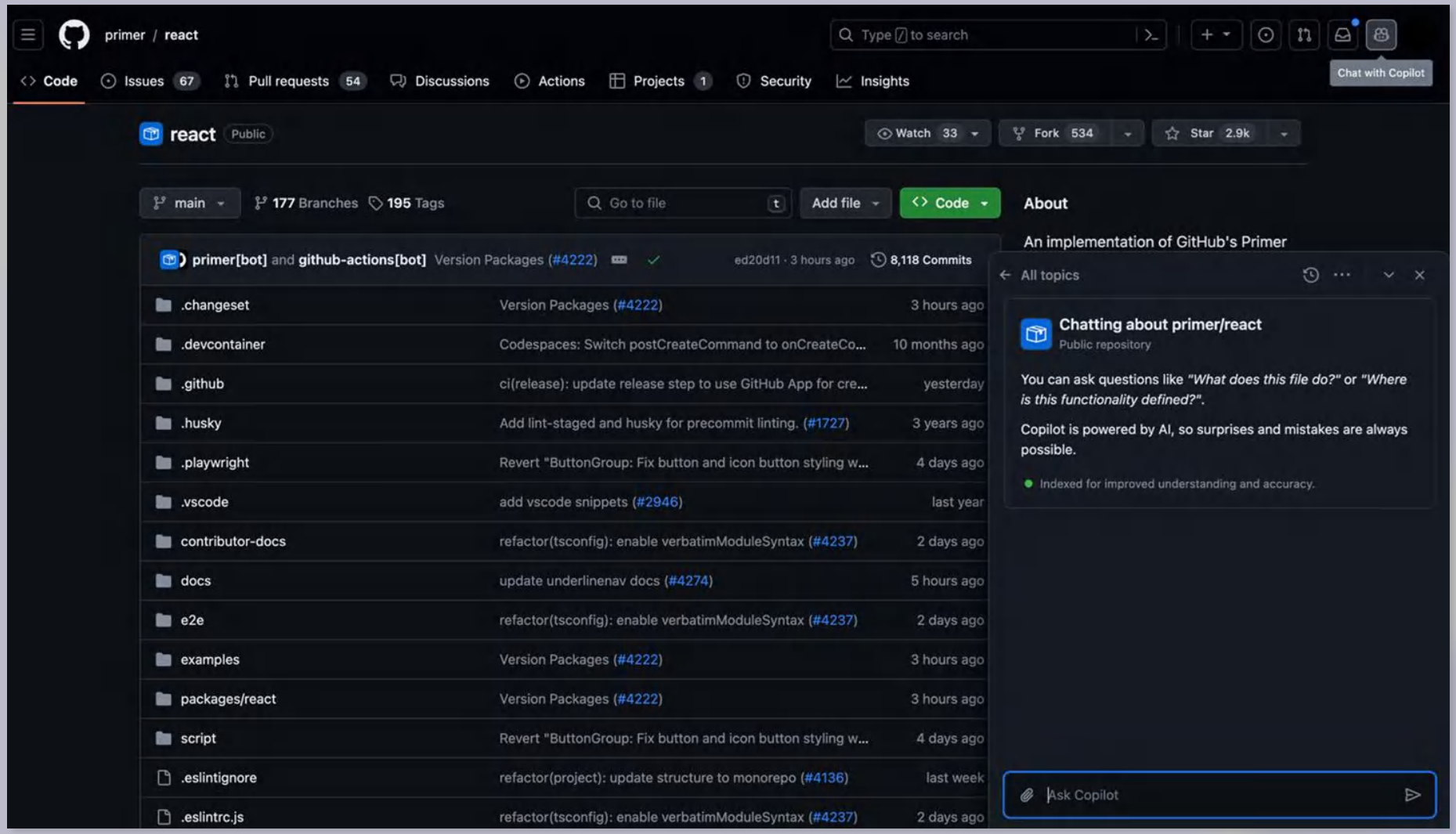
# 您的編輯器中體驗類似 ChatGPT 的 GitHub Copilot Chat 模式



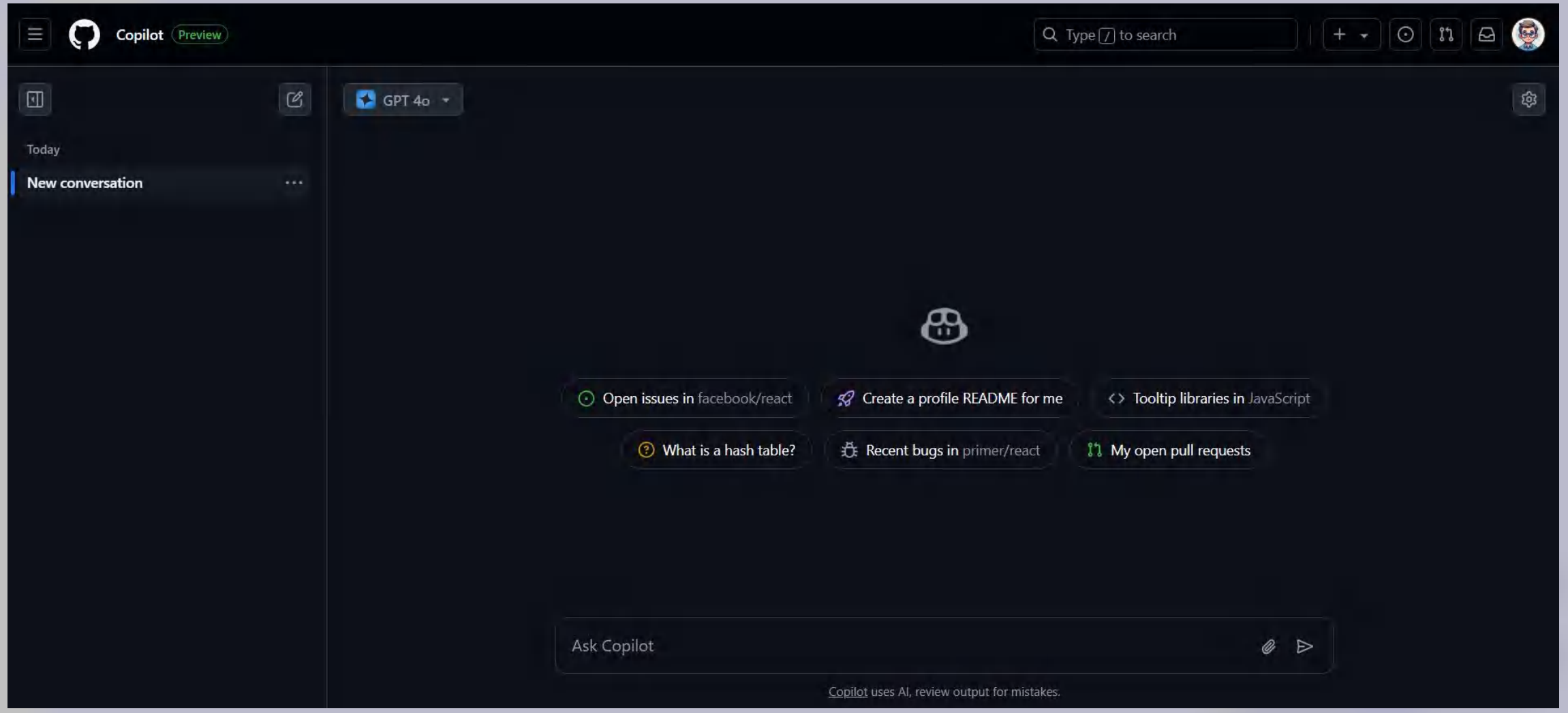
# 提取要求摘要 Pull Request Summaries (GitHub.com)



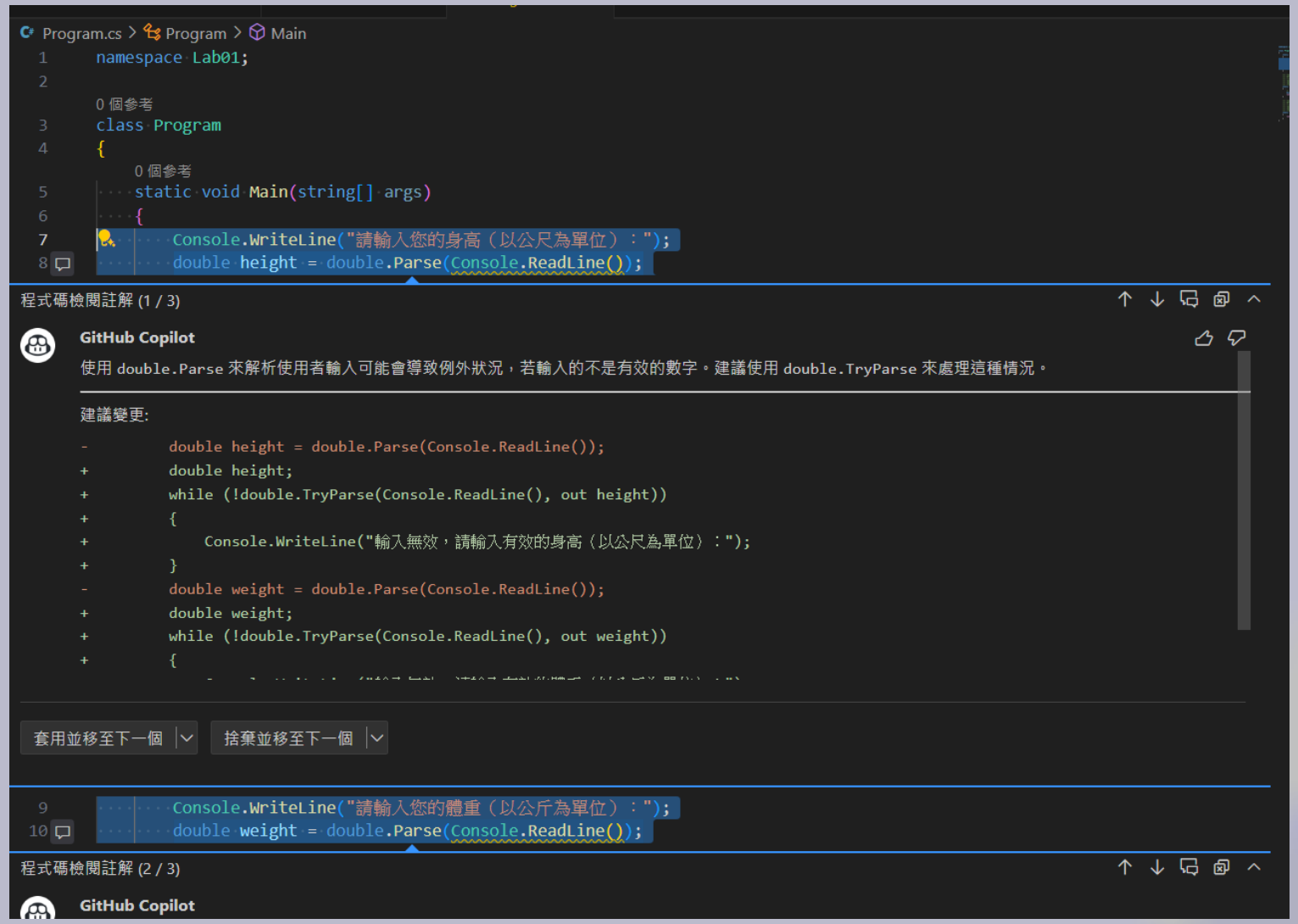
# 自訂至程式碼基底的聊天 Chat Customized to Your Codebase（GitHub.com)



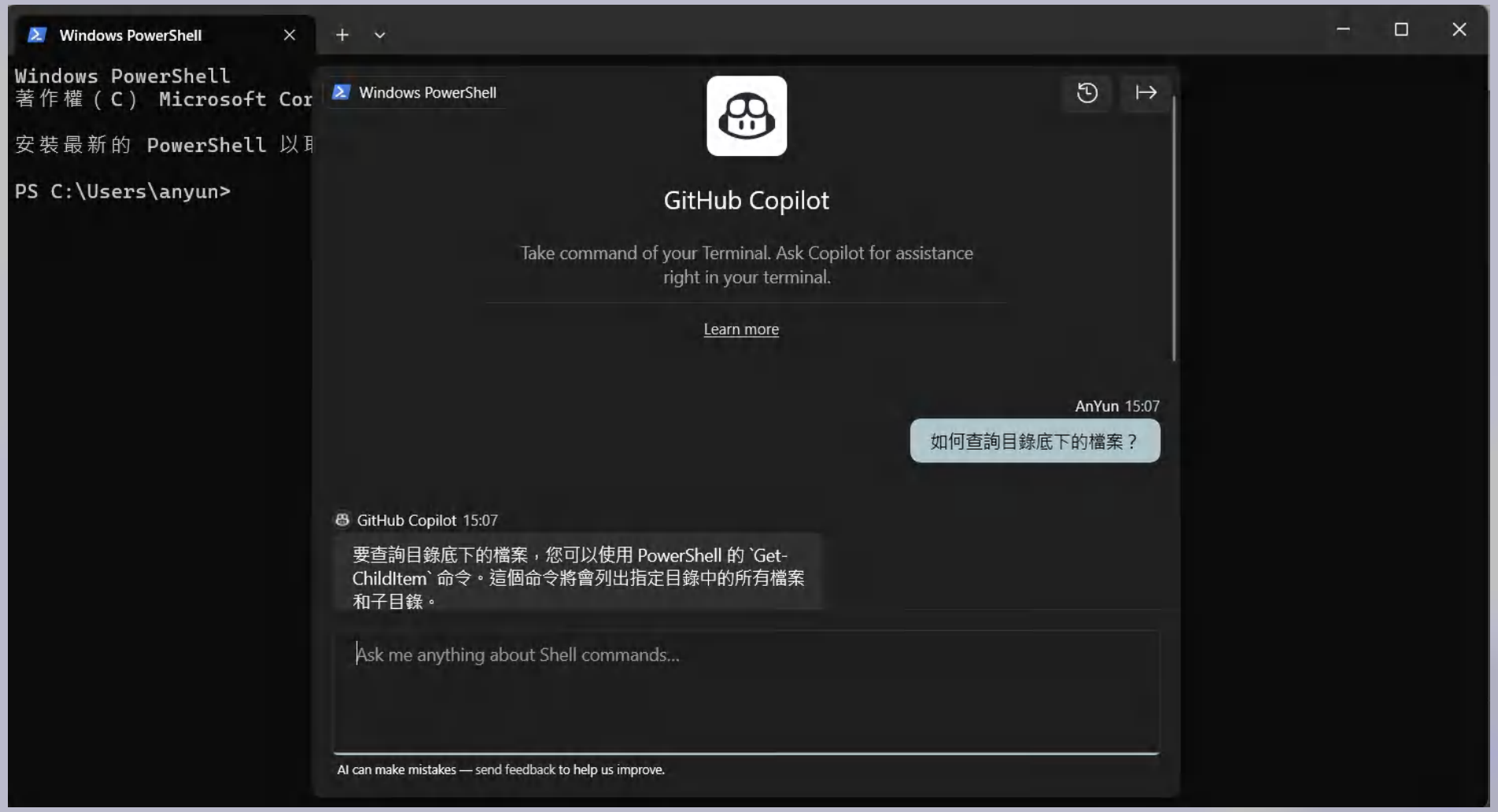
# 網頁版 GitHub Copilot (GitHub.com)



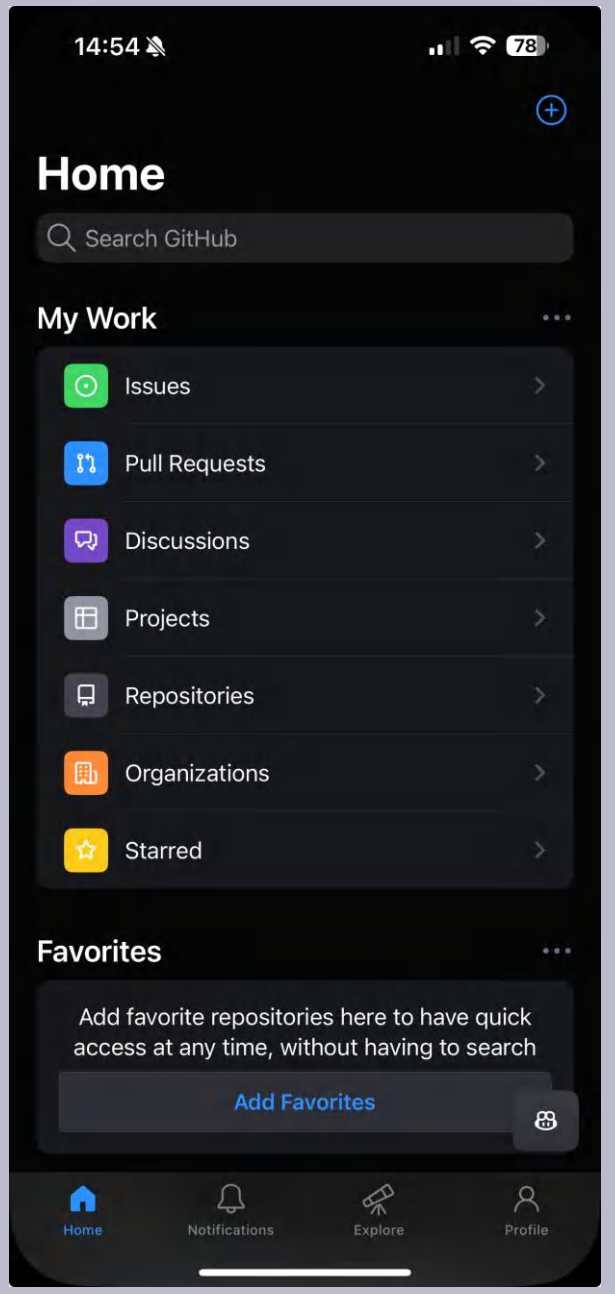
# 程式碼檢閱 Code Review

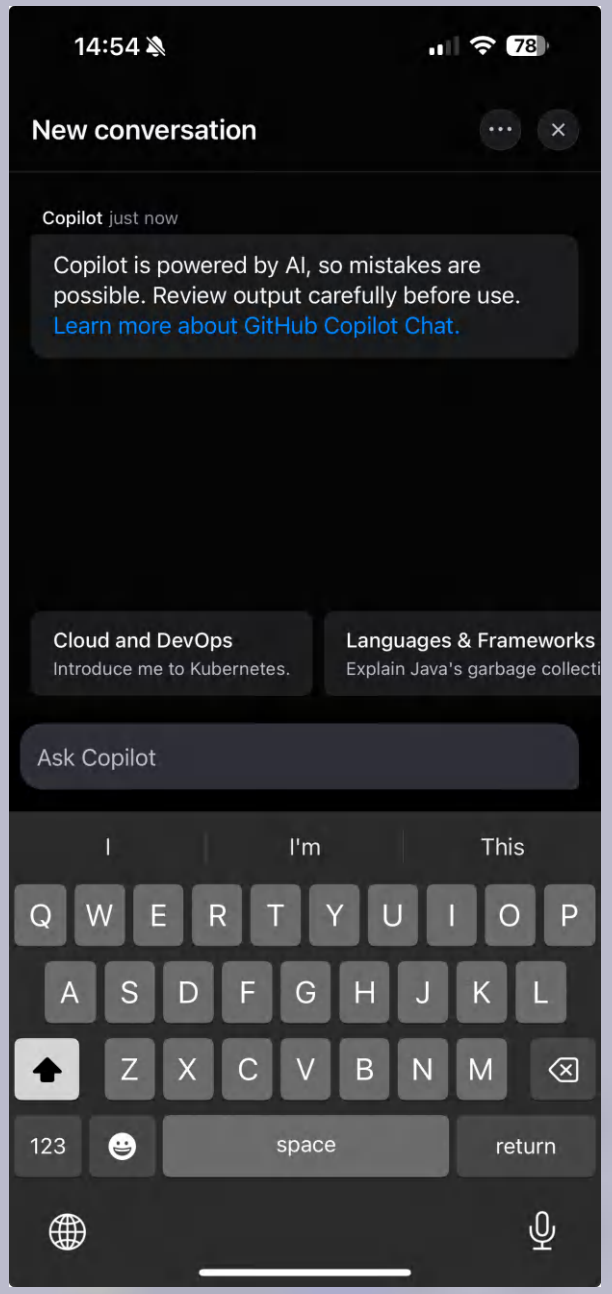


# 透過 GitHub Cli 來使用 GitHub Copilot

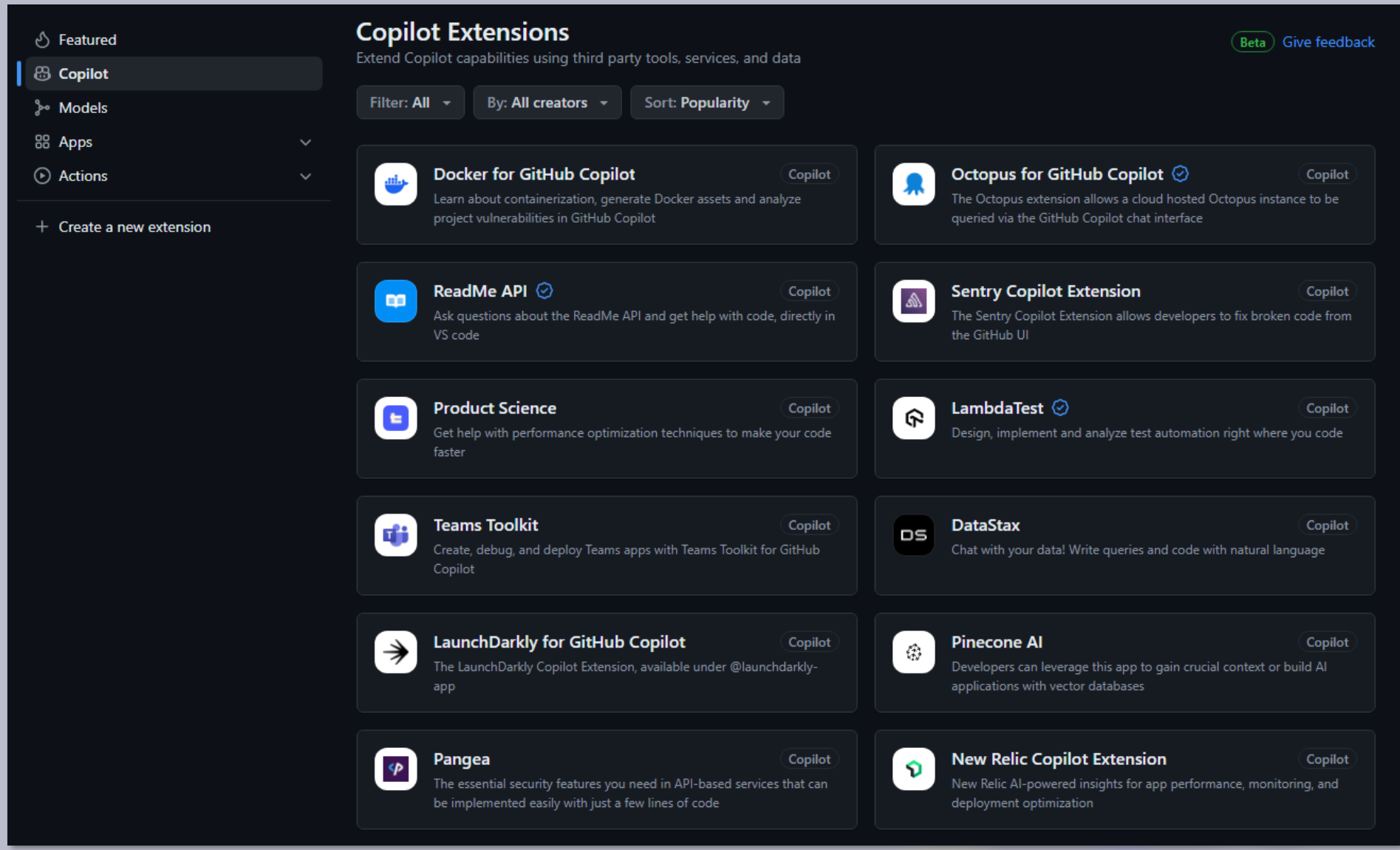


# GitHub Copilot Chat in GitHub Mobile



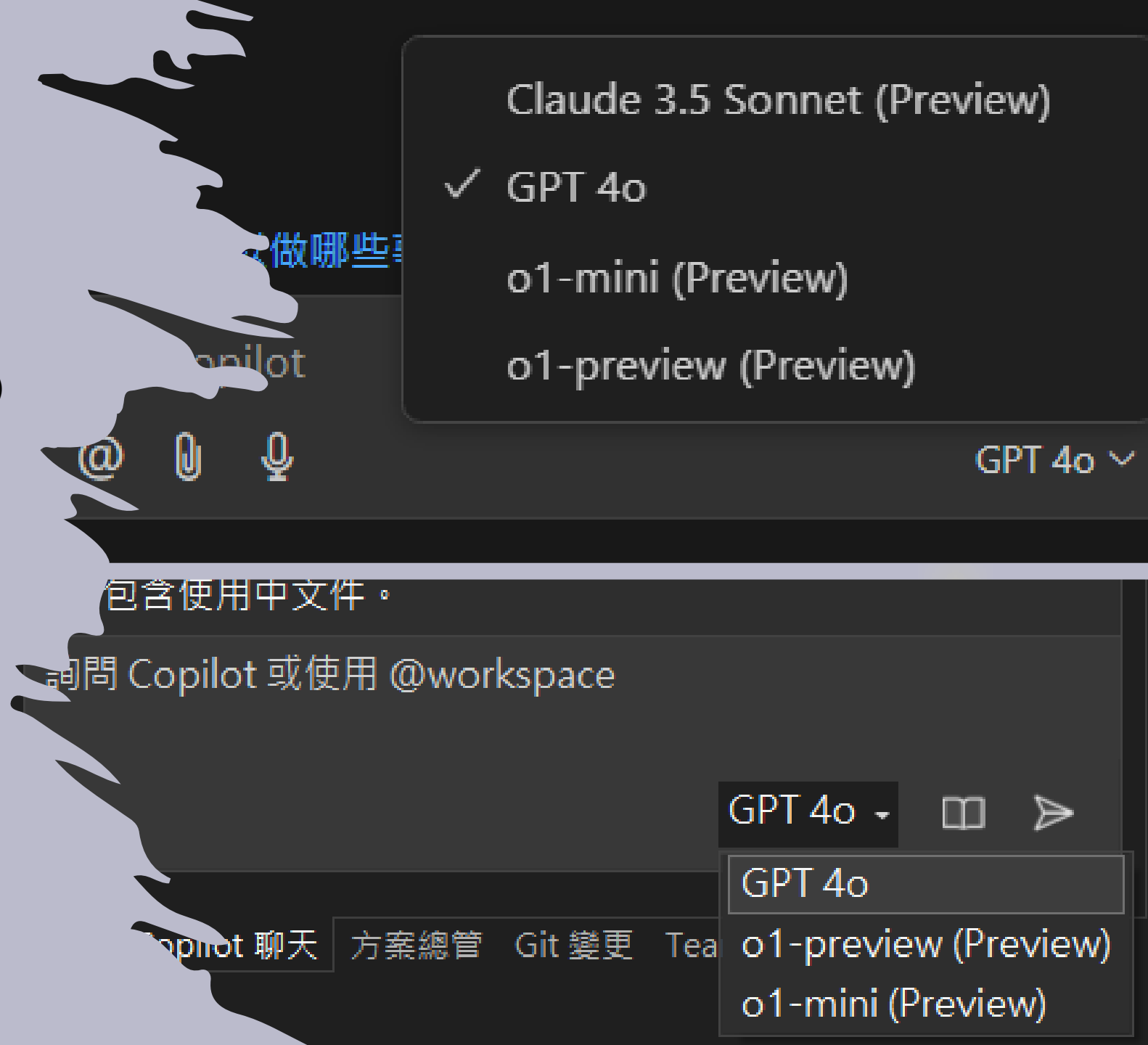
# GitHub Copilot Extensions




Copilot Extensions

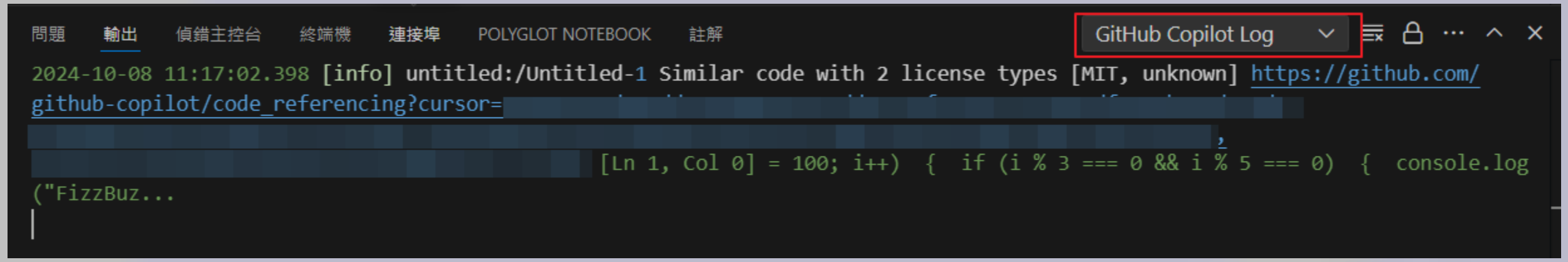
# 支援多種模型

- 支援的模型
  - Anthropic Claude 3.5 Sonnet
  - Google Gemini 1.5 Pro (尚未正式推出)
  - OpenAI o1-preview
- 需從 Copilot Policy 去啟用



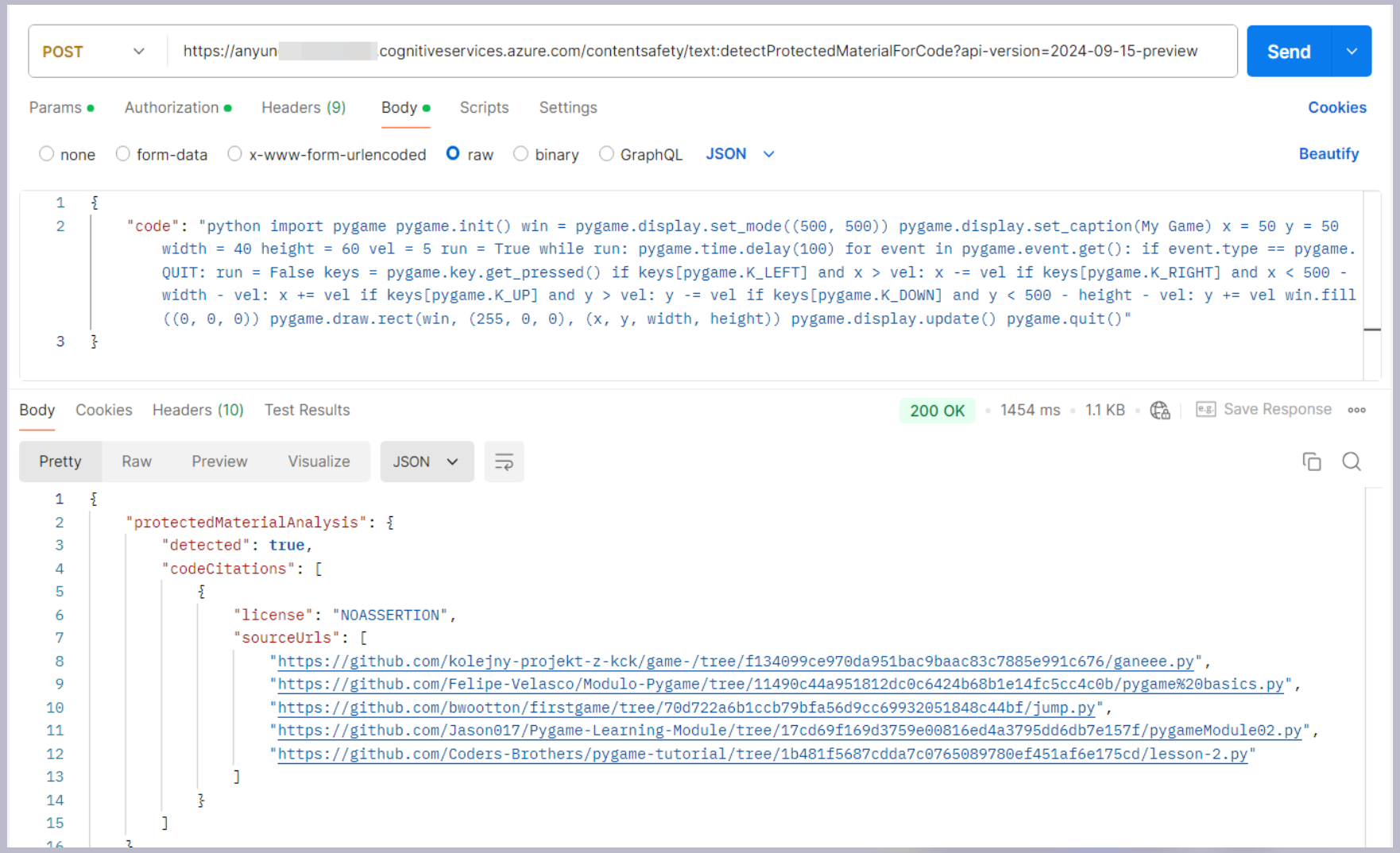
# 程式碼參考 Code referencing

- 程式碼參考可以在產生程式碼建議時候提示有使用到 Public Code
- 目前僅支援 VS Code
- 底層透過 Azure AI Content Safety

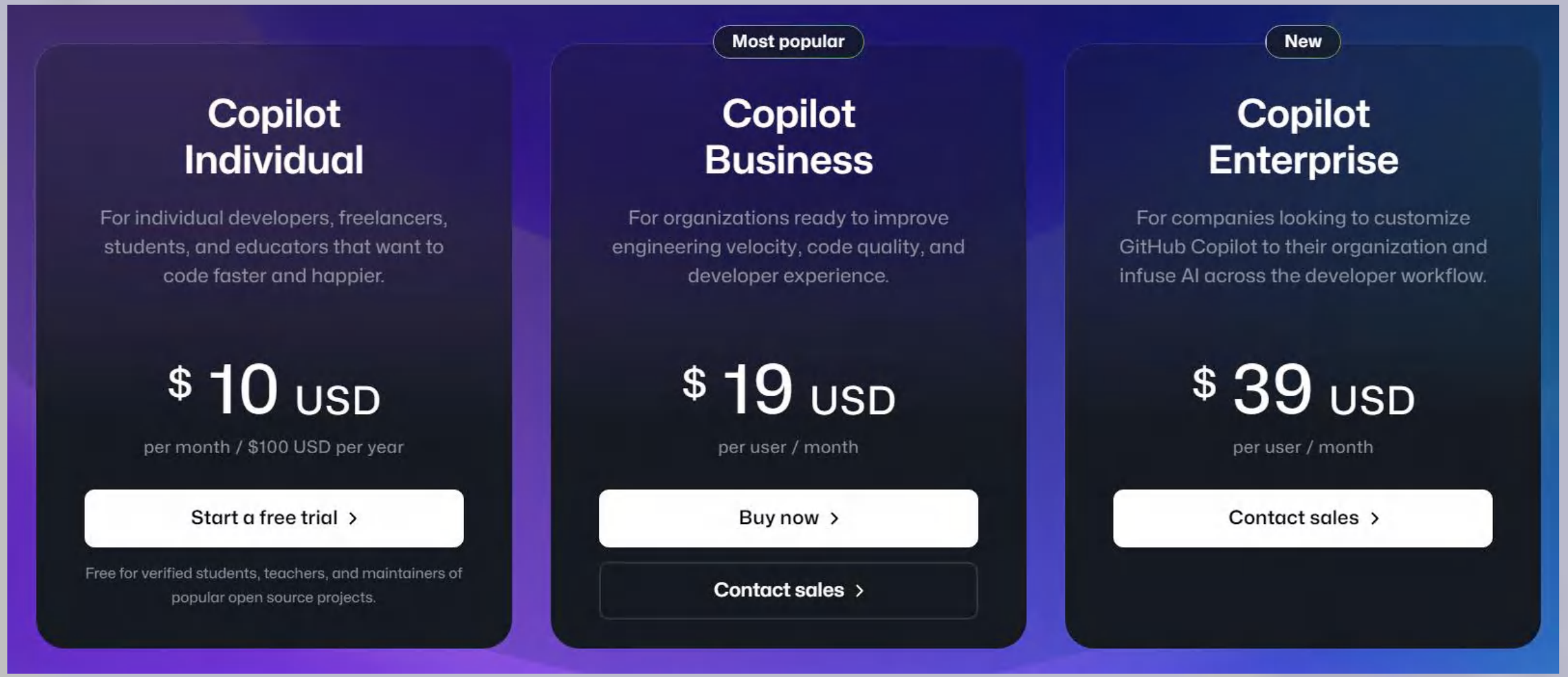


# 程式碼參考 Code referencing

- Azure AI Content Safety
  - 程式碼的受(版權)保護素材偵測
  - 資料只到 2021 年 11 月 6 日以前的程式碼



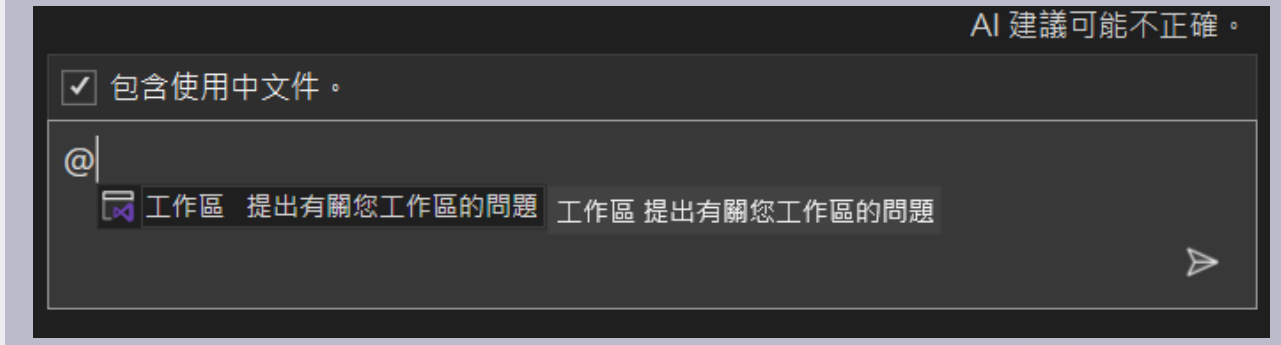
# GitHub Copilot 基本介紹

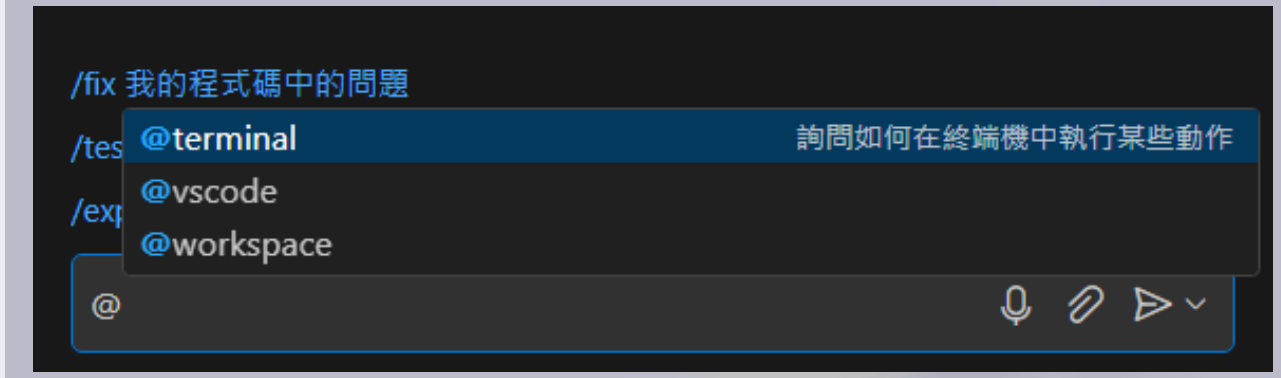


# GitHub Copilot 基本用法

# GitHub Copilot 基本用法 : Agent Command

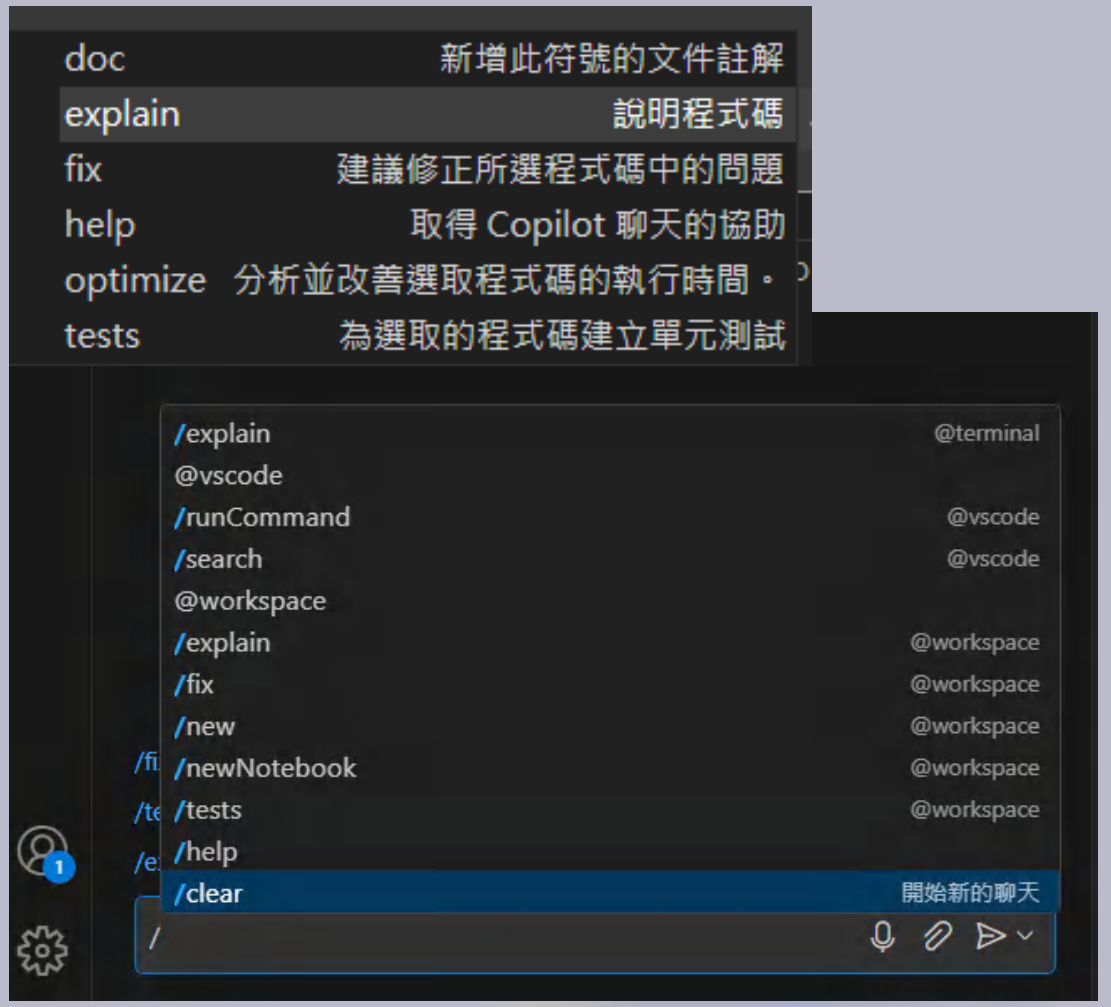
| | |
|---|---|
| @terminal | 協助處理終端命令及其運作 |
| @vscode | 提供使用 VS Code 的技巧與功能建議 |
| @workspace | 工作區中的程式碼內容 |
| @AGENT | 新增程式碼註解 |



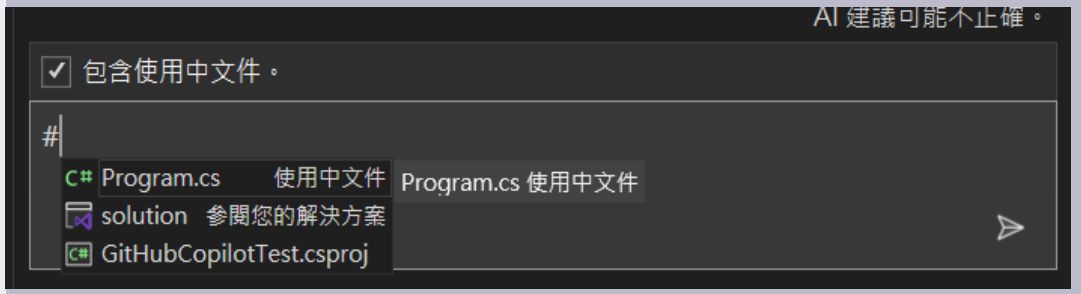
# GitHub Copilot 基本用法 : Slash Command

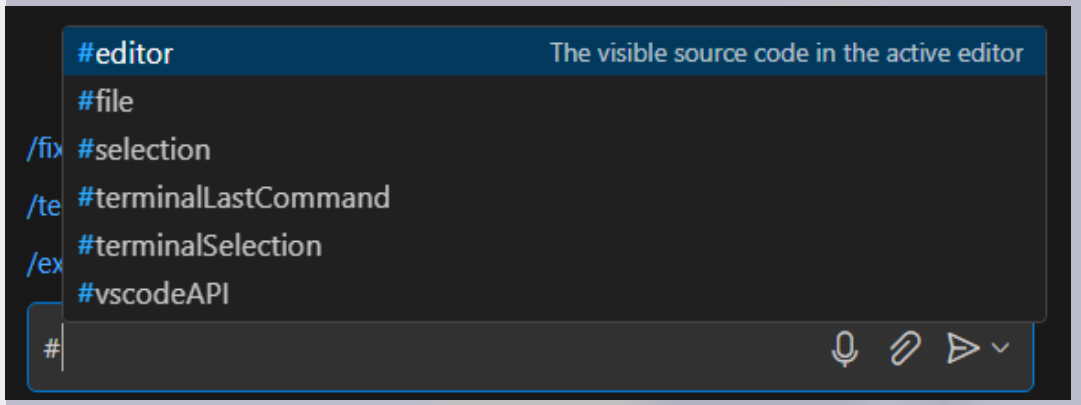
| | |
|---|---|
| /explain | 說明程式碼 |
| /tests | 為選取的程式碼產生單元測試 |
| /fix | 建議修正所選程式碼中的問題 |
| /doc | 新增程式碼註解 |
| /help | 取得 GitHub Copilot 的聊天協助 |



# GitHub Copilot 基本用法 : Hash Command

| | |
|---|---|
| #editor | 使用中編輯器中的目前內容 |
| #file | 指定檔案內容 |
| #selection | 選取的內容 |
| #terminalSelection | 終端中選取的內容 |
| #terminalLastCommand | 使用中終端的最後一個執行命令 |



# GitHub Copilot 基本用法

## 快速鍵 (VS Code / VS)

- Ctrl+I / Alt+/：inline Chat
- Alt+[ / Alt+,：上一個建議
- Alt+] / Alt+.：下一個建議
- Tab：接受建議
- Ctrl+Enter：Suggestions Panel (僅 Vs Code 有)

- Configuring GitHub Copilot in your environment

# DEMO

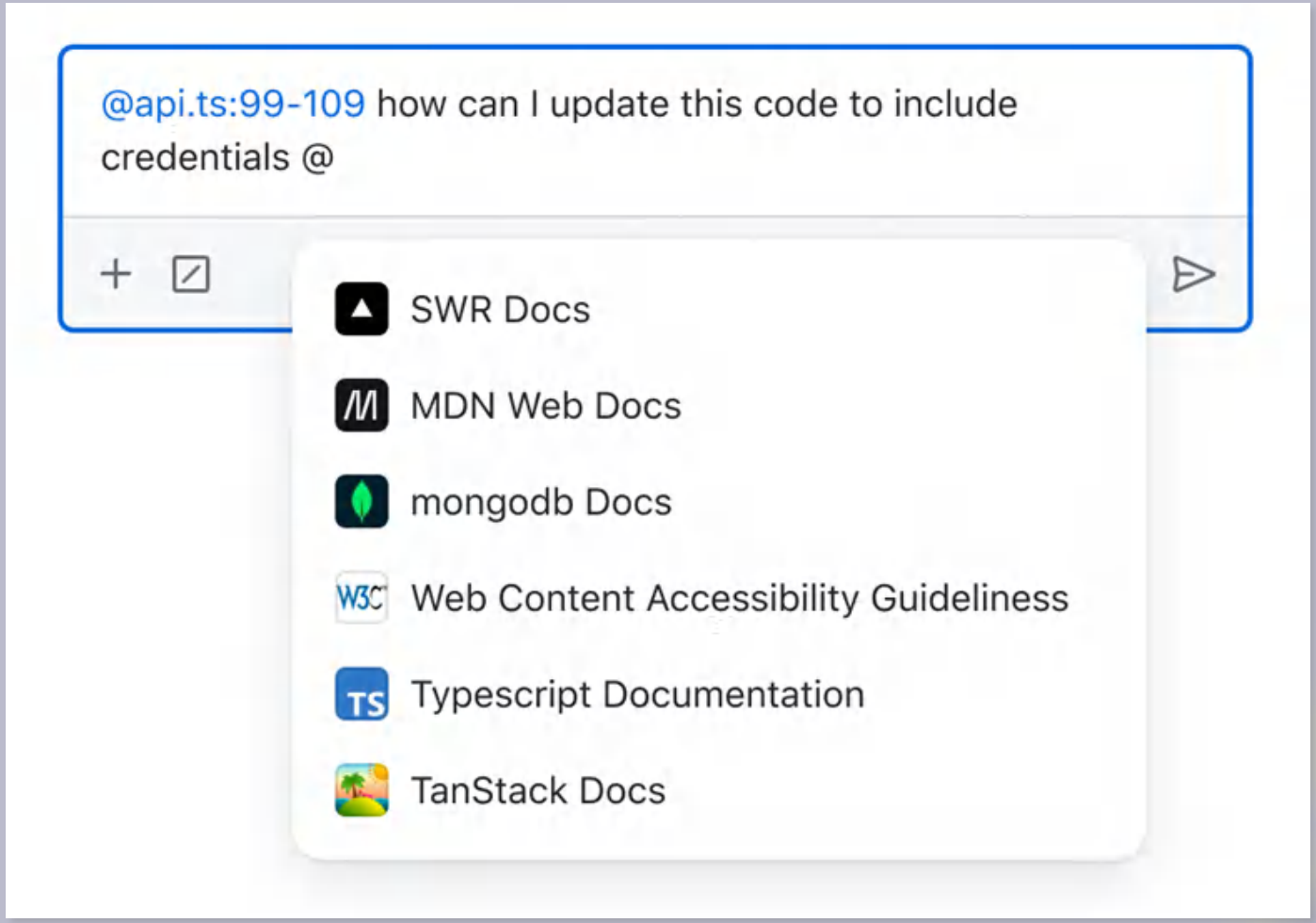
# GitHub Copilot Enterprise

# GitHub Copilot Enterprise：功能比較

GitHub Copilot Enterprise 簡介

| 功能 | Enterprise | Business | Individual |
| --- | --- | --- | --- |
| 提取要求內的共同聊天 | ✓ | | |
| Lives on Github.com | ✓ | ✓ | ✓ |
| 直接插入編輯器 | ✓ | ✓ | ✓ |
| 提取要求摘要 | ✓ | | |
| Copilot 文件集管理 | ✓ | | |
| Copilot 程式碼檢閱 | ✓ | | |
| 程式碼片段和使用方式遙測的零資料保留 | ✓ | ✓ | ✓ |
| 組織範圍的原則管理 | ✓ | ✓ | |
| 與安全性工具整合 | ✓ | 有限 | |
| 稽核記錄和報告 | ✓ | 有限 | |
| 透過自我簽署憑證支援 VPN Proxy | ✓ | ✓ | |

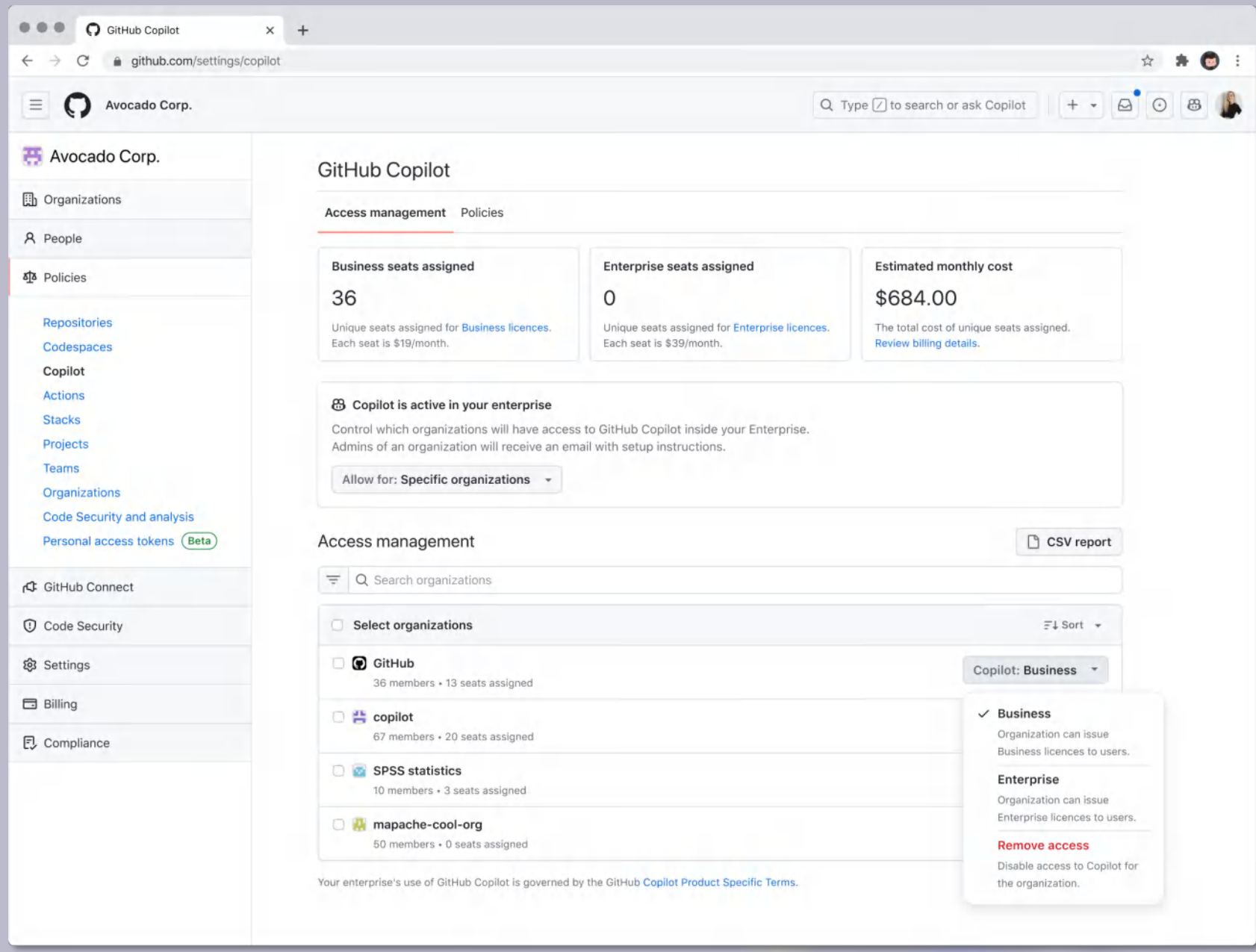
# 使用文件集的文件搜尋和摘要
# Documentation Search and Summaries Using Docsets



# Copilot Enterprise Mixed Licensing

可以更靈活地混合選擇

- Copilot Business
- Copilot Enterprise

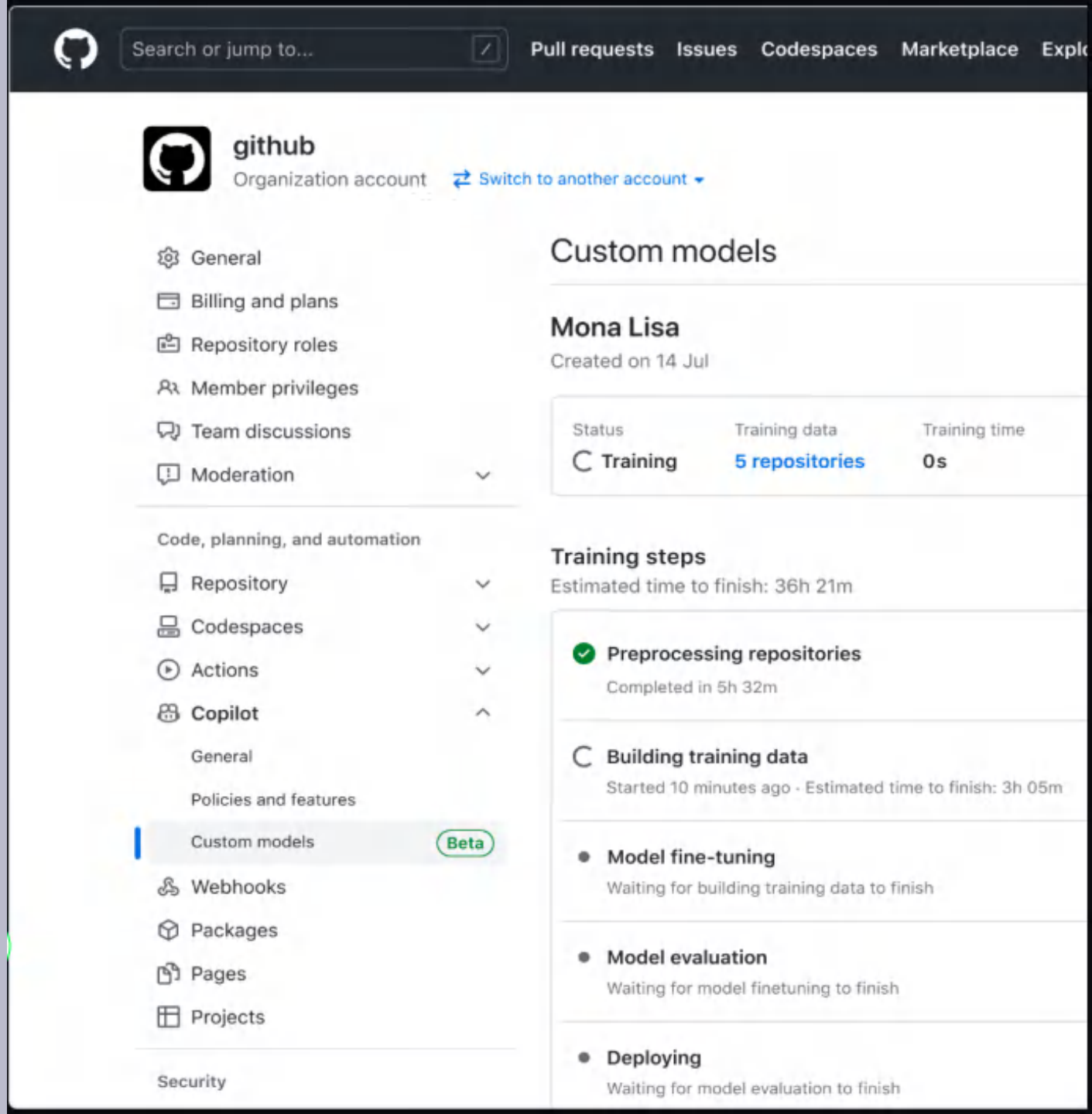


# 個人化的 fine-tuned models

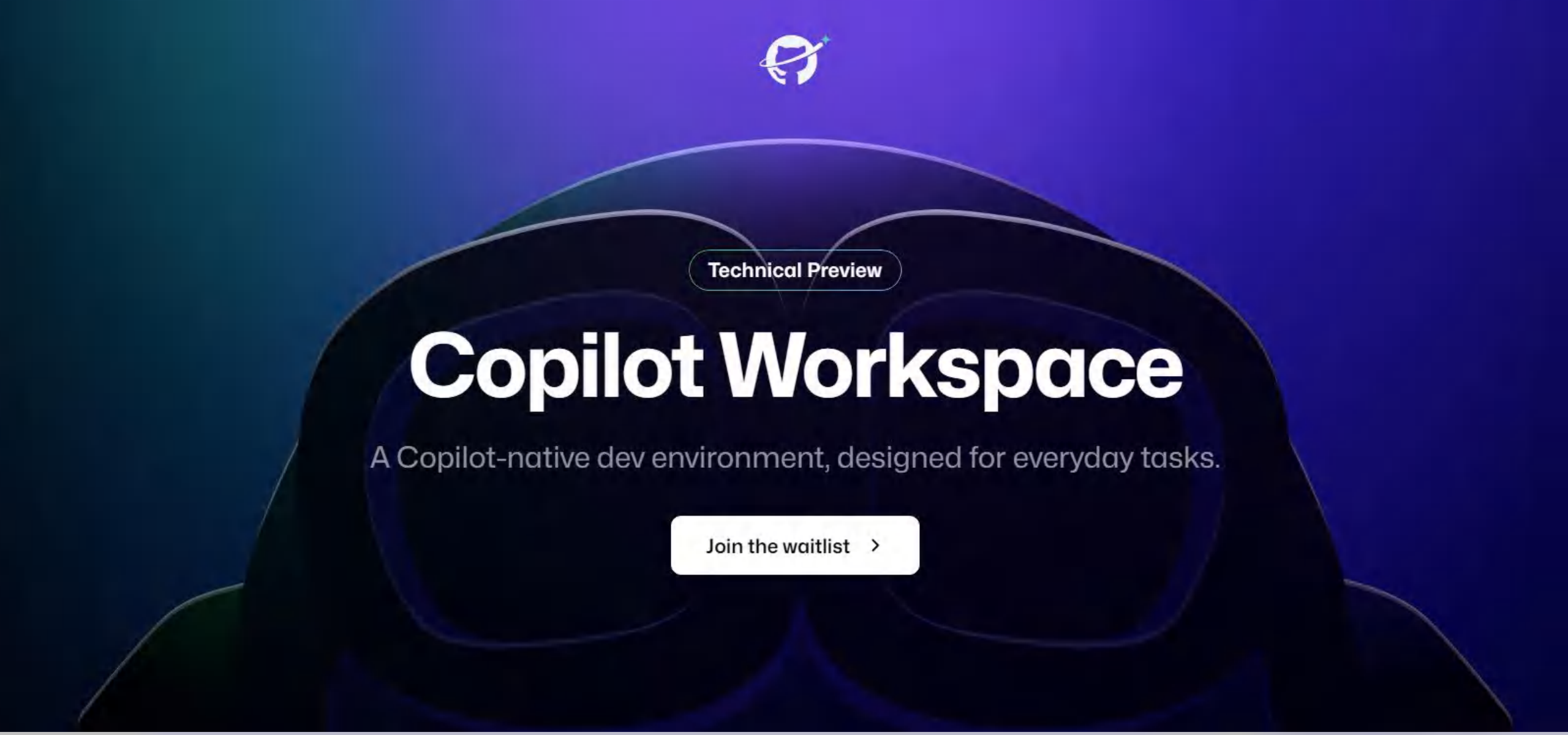
**如果您需要更精確的企業內部程 式碼建議，可以選擇fine-tune model, 客製化自己的Copilot**

- 顯示更多相關和量身定製的代碼建議
- 開發人員可以更快地編寫代碼，減少錯誤
- 程式碼的參考、建模、託管和推論均屬企業自己的
- 程式碼建議基於訓練出來的模型

Creating a custom model for GitHub Copilot



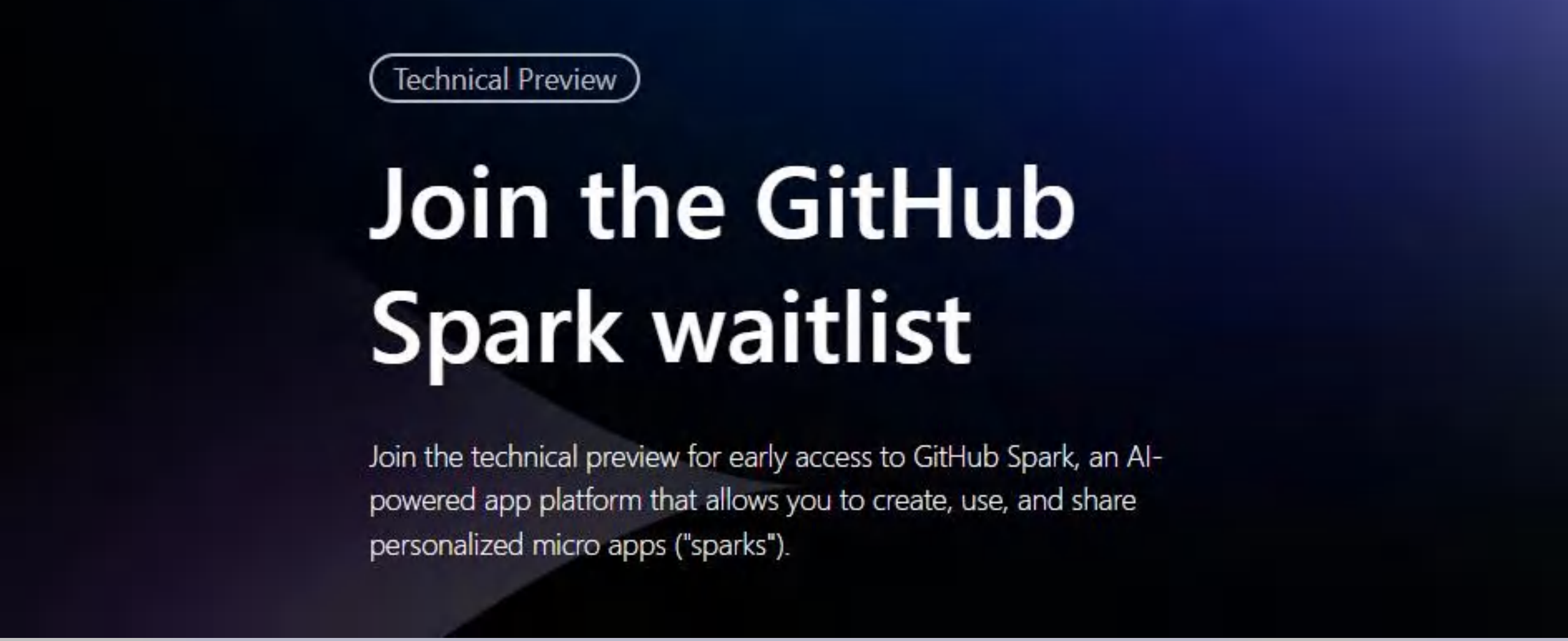
# Bonus




GitHub Next | Copilot Workspace



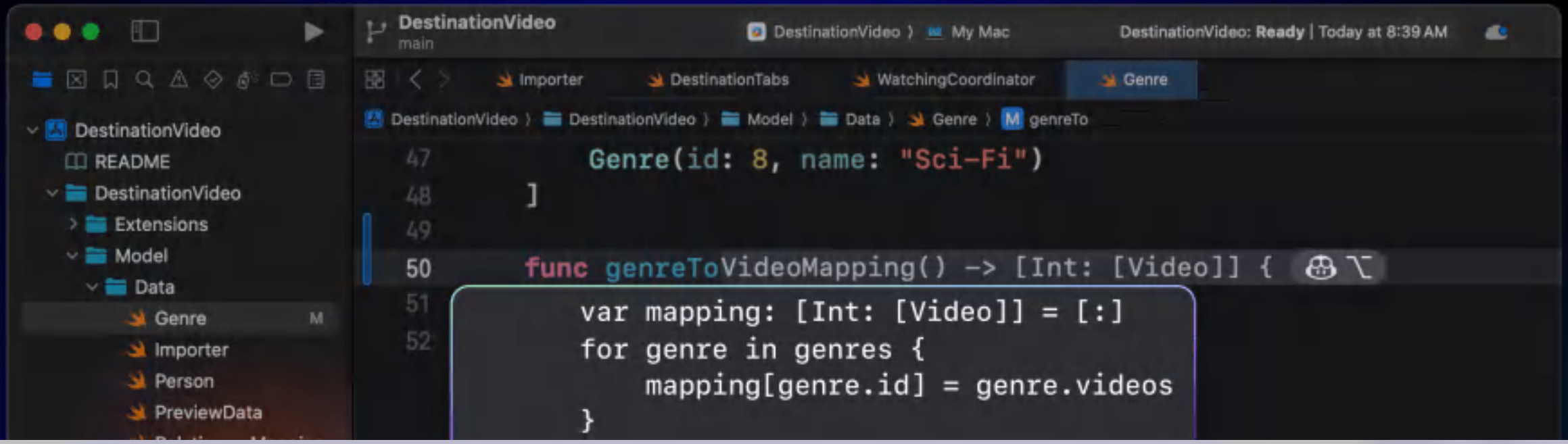
[Try out OpenAI o1 in GitHub Copilot and Models]




GitHub Spark

Public Preview

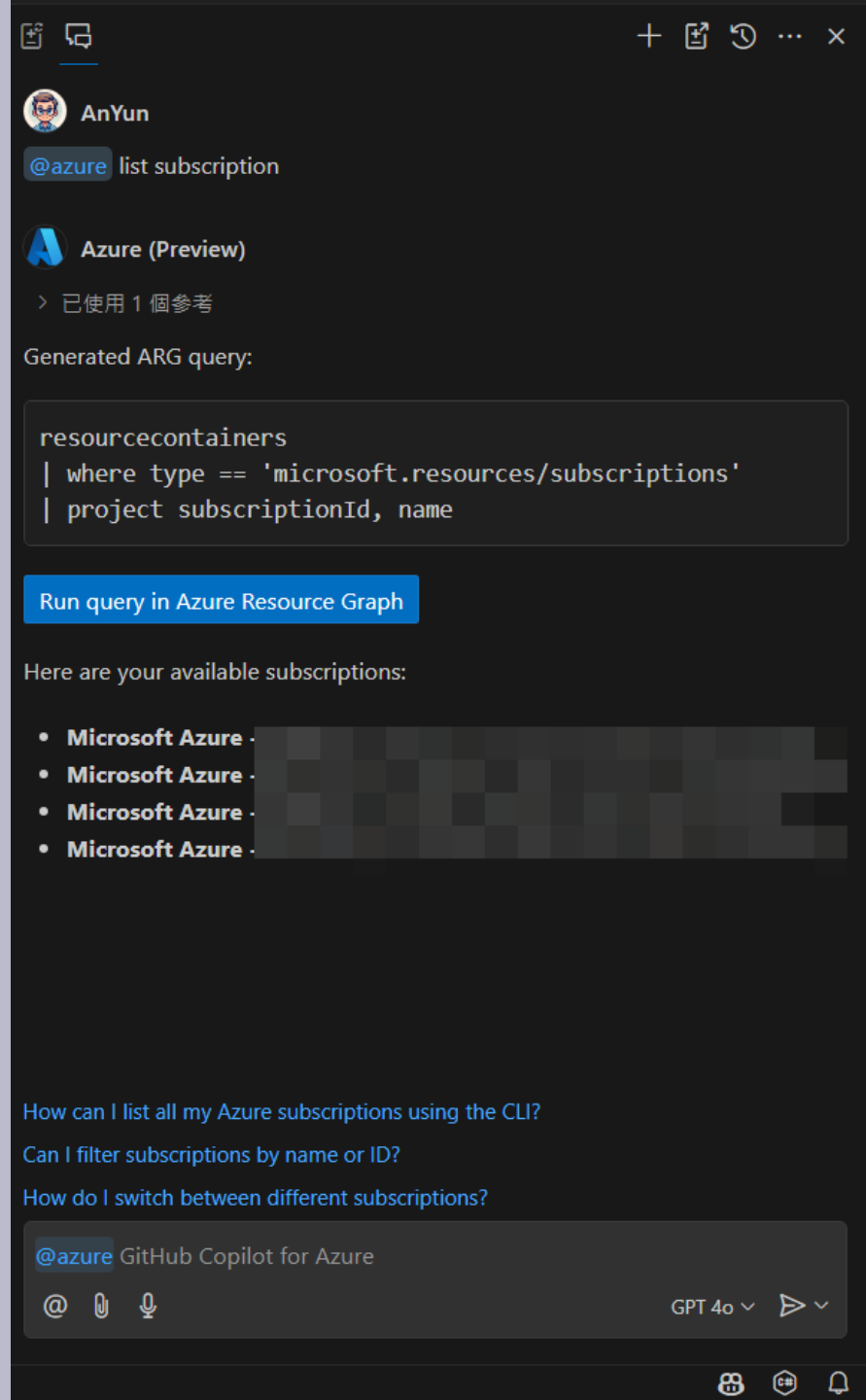
# GitHub Copilot for Xcode




GitHub copilot code completion in Xcode

# GitHub Copilot for Azure

- Introducing GitHub Copilot for Azure: Your Cloud Coding Companion in VS Code!
  - https://techcommunity.microsoft.com/t5/microsoft-developer-community/introducing-github-copilot-for-azure-your-cloud-coding-companion/ba-p/4127644
  - https://techcommunity.microsoft.com/t5/microsoft-developer-community/github-copilot-for-azure-6-must-try-features/ba-p/4283126

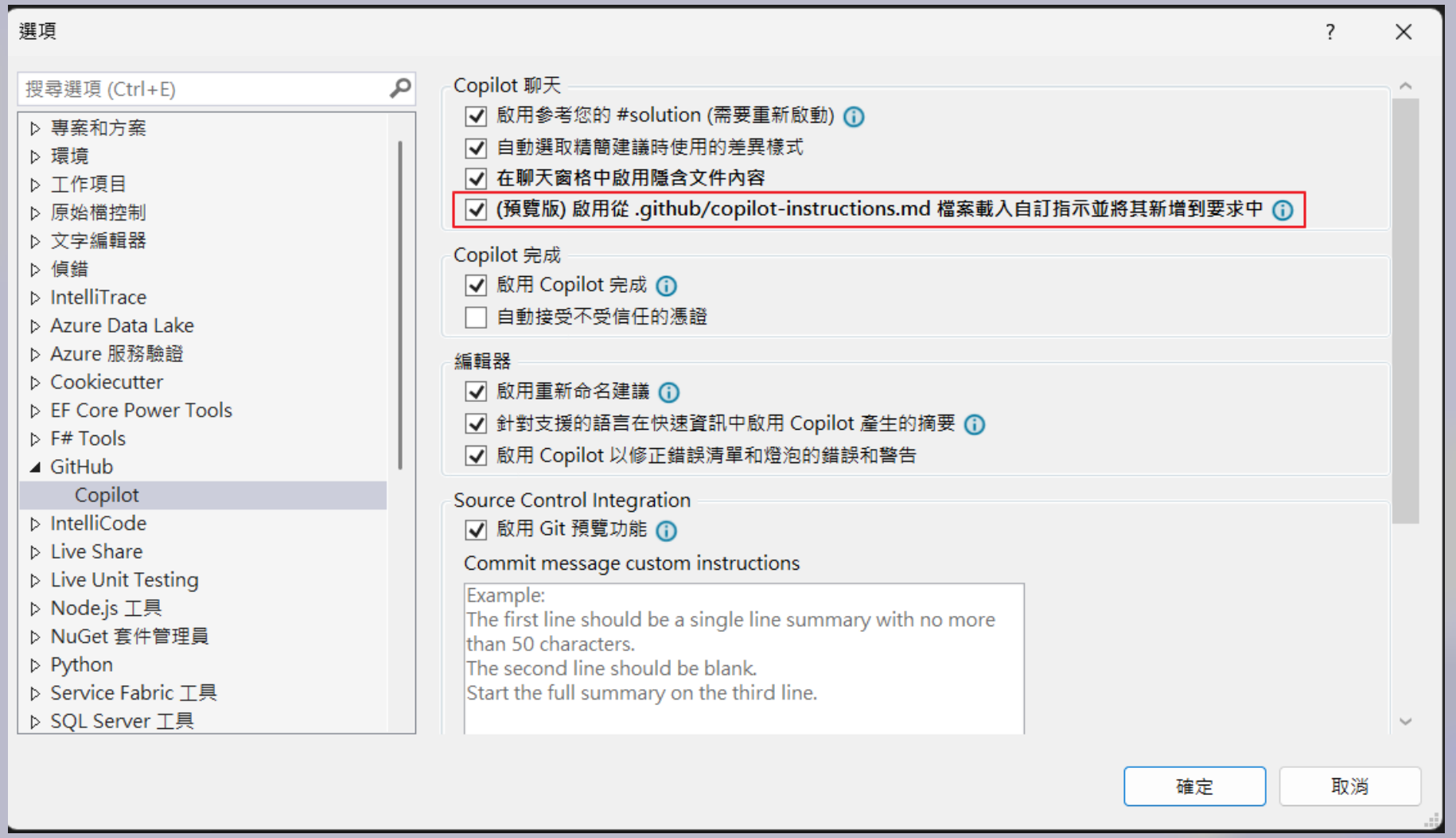


# 自定義指示詞

- 開啟使用自定義指示詞功能
- 在專案資料夾底下加入 .github/copilot-instructions.md
- Custom instructions for GitHub Copilot in VS Code



# 自定義指示詞


選項
搜尋選項 (Ctrl+E)
專案和方案
環境
工作項目
原始檔控制
文字編輯器
偵錯
IntelliTrace
Azure Data Lake
Azure 服務驗證
Cookiecutter
EF Core Power Tools
F# Tools
GitHub
Copilot
IntelliCode
Live Share
Live Unit Testing
Node.js 工具
NuGet 套件管理員
Python
Service Fabric 工具
SQL Server 工具
Copilot 聊天
啟用參考您的 #solution (需要重新啟動)
自動選取精簡建議時使用的差異樣式
在聊天窗格中啟用隱含文件內容
(預覽版) 啟用從 .github/copilot-instructions.md 檔案載入自訂指示並將其新增到要求中
Copilot 完成
啟用 Copilot 完成
自動接受不受信任的憑證
編輯器
啟用重新命名建議
針對支援的語言在快速資訊中啟用 Copilot 產生的摘要
啟用 Copilot 以修正錯誤清單和燈泡的錯誤和警告
Source Control Integration
啟用 Git 預覽功能
Commit message custom instructions
Example:
The first line should be a single line summary with no more than 50 characters.
The second line should be blank.
Start the full summary on the third line.
確定
取消

# 小結

- AI 是否會取代開發人員？
  - 最終佈署程式還是得人做最後的決定 (負責任的 AI – 當責)
  - 開發人員仍然需要會開發技能，但 Junior 跟 Senior 定義跟界線模糊
- AI 世代開發人員需要的技能？
  - 基本功 (物件導向、設計模式等) 仍須紮實
  - 溝通的能力 (會說人話)

# FAQ