無程式基礎網路爬蟲入門

課程講義 快速連結 QRCODE

https://bit.ly/498qPvO

不需要準備

✖☐ ~~程式語言~~
✖☐ ~~安裝爬蟲套件~~
✖☐ ~~本地開發環境~~
✖☐ ~~VPN 或代理伺服器~~
✖☐ ~~API 金鑰設定~~

課前準備

環境

作業系統

- Windows 10 / 11
- macOS 13+
- 筆電/桌機優先，不建議使用平板或手機

瀏覽器

- Google Chrome (推薦)
- Microsoft Edge

Google 帳號

- 若無帳號，請先完成申請（參考 建立 Google 帳戶）
- 請先確定帳號、密碼可正常登入 Google
- 可於 Gmail 正常收發信件

Apify 帳號

- 註冊網址：https://apify.com/
- 註冊方式：使用 Google 帳號快速註冊、登入
- 免費方案即可
- 確認可成功登入 Apify Console
- 進入 Actors（工具）頁面不會卡關

課前測試

- ☑ 可成功登入 Apify
- ☑ 能在 Actors Store 搜尋工具（如 Web Scraper）
- ☑ 瀏覽器可正常下載 CSV / Excel 檔案

爬蟲概念（快遞收件員 vs 自動化機器人）

如果以 「資料採集工廠 × 自動化小幫手」 作為核心比喻，
下面我們先簡要的做個名詞對照。

名詞對照說明

- 整個網路：一座超大的城市
- 每個網站：城市中的一棟建築
- 資料：建築裡的商品或文件
- 人力蒐集：人工快遞，一趟一趟去收件
- 爬蟲：不會疲倦的自動化機器人
- 儲存體：機器人身上的儲物箱、資料中心
- Apify 工具：機器人調度中心

對照組

- 人力收集資料：快遞員收取物件
- 爬蟲收集資料：自動運作的機器人收取物件

人力蒐集（快遞）

- 上班時間只能跑幾個地方
- 需要休息、可能錯漏
- 重複工作成本高、效率有限
- 派快遞員一棟一棟進去拿資料，再慢慢帶回來

爬蟲（機器人）

- 依照你事先設定的地圖與規則
- 在整座城市裡自動移動
- 精準找到指定的建築與資料
- 把資料整齊地放進自己的儲物箱中
- 告訴它「去哪裡、拿什麼、怎麼拿」

差別在於

- 人會累、會分心、可能會出錯
- 機器人可以 24 小時準時執行，每次都照規則來

爬蟲可以這樣理解

爬蟲是一個不需要休息的資料快遞機器人，
依照你設定好的路線與規則，
在整座網路城市中自動來回，
把指定的資料穩定又準確地帶回來。

所以我們要學的不是自己當快遞，
而是怎麼把任務交給這些機器人來完成。

爬蟲 vs API

- 爬蟲是一種「資料取得方式」
- API 是一種「資料提供服務」

API 是什麼？

通常是網站或平台「刻意提供」的資料服務，
資料結構清楚、規則明確、格式固定，對外的資料通道。

- 有文件說明
- 有欄位定義
- 有流量與權限控管

爬蟲是什麼？

是一種自動化的資料蒐集方式，可以從各種網路來源取得資料，
包含「網頁畫面」與「API 回傳的資料」。

只要資料在網路上、能被請求，爬蟲理論上都「有機會」去取得。

對照表

項目	爬蟲	API
本質	是種資料蒐集方式	通常為資料服務
資料結構化?	不一定	高度結構化
是否公開?	視情況	通常有規範
格式穩定?	易受版面影響	格式較為穩定

靜態網頁 vs 動態網頁

靜態網頁是什麼？

通常不太會變化，一個網址代表一個固定頁面內容，格式通常為 HTML。

例如：

- 公司簡介
- 新聞文章
- 活動公告

☞ 靜態網頁：因為固定通常比較容易抓

動態網頁是什麼？

通常是包含可互動的頁面內容，資料會因為互動行為（滑動、點擊、搜尋）而改變，格式基本仍是 HTML 但包含 JavaScript 的互動內容。

例如：

- 電商商品列表
- 社群平台
- 地圖、評論、搜尋結果頁

☞ 動態網頁：因為有互動行為，需要模擬「人操作瀏覽器」

Robots.txt 與法律／網站條款

爬蟲的世界也有一些規則需要遵守，在開始抓資料前需要先了解一下這些規則。

什麼是 Robots.txt？

是網站用來告訴爬蟲「哪些可以抓、哪些不行」
通常放在：網站網址/robots.txt

☞ Robots.txt 代表網站的使用規則（如果有的話）

法律與風險基本觀念

今天的課程重點是：

- 公開資料
- 不登入、不繞過驗證
- 不造成網站負載

請大家遵守三個原則：

- 只抓公開資料
- 不破壞、不干擾網站
- 資料用途合法合規

📌 如果資料用於商業或大量分析，一定要再三確認使用條款

適合使用無程式工具的情境

避免大家想像力太豐富，
今天的主題大致有一些先天條件和預期目標：

✔ 不會寫程式
✔ 想快速拿到資料

✔ 資料來源固定（網站不常改版）
✔ 需要排程、重複抓取
✔ 希望匯出 Excel / CSV

☞ 目標不是成為工程師，而是「拿到資料」

為何選擇 Apify？

Apify = 專為「實務資料抓取」設計的平台

Apify 的優點：

- 不寫程式也能用
- 支援動態網頁
- 可排程、自動化
- 直接輸出 CSV / Excel / JSON

非常適合：

- 行銷人員
- 研究人員
- 行政與企劃
- 教育與訓練課程

工具平台介紹：Apify

Apify 是雲端資料抓取與自動化平台，讓使用者能夠在無需撰寫程式碼或少量程式碼的情況下，自動抓取網站資料、處理網路任務、並設定排程執行。

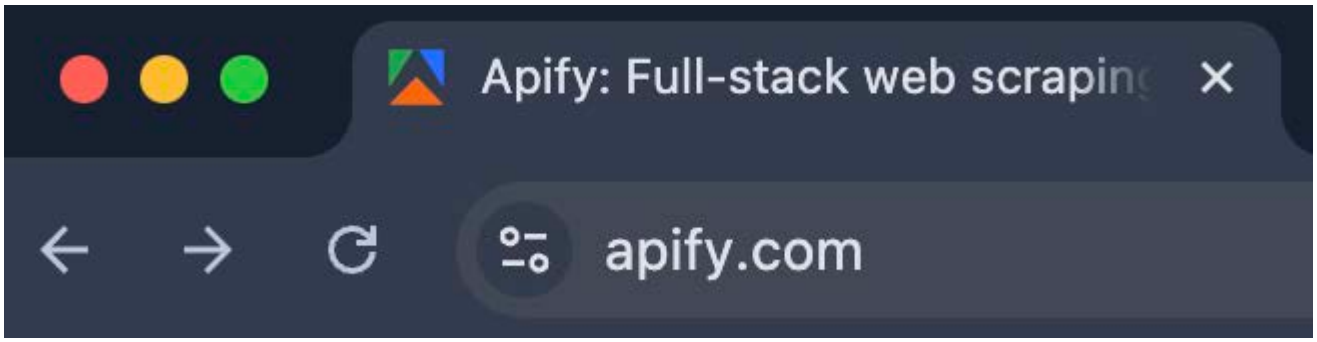
透過其 Actors Store，使用者可以直接使用現成的爬蟲工具或自訂配置腳本來抓取特定網站內容，如商品價格、評論、活動資訊等。

Apify 支援資料輸出為 CSV、Excel、JSON 等格式，並可與第三方服務（例如 Google Sheets 或 API）整合，便於後續資料分析與應用。

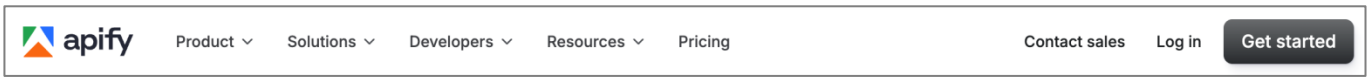
適合各種角色使用，包括行銷人員、產品分析師、研究人員等，不需要深厚程式背景也能快速掌握網路資料自動化流程。

註冊 Apify 帳號

前進 Apify 網站 https://apify.com/

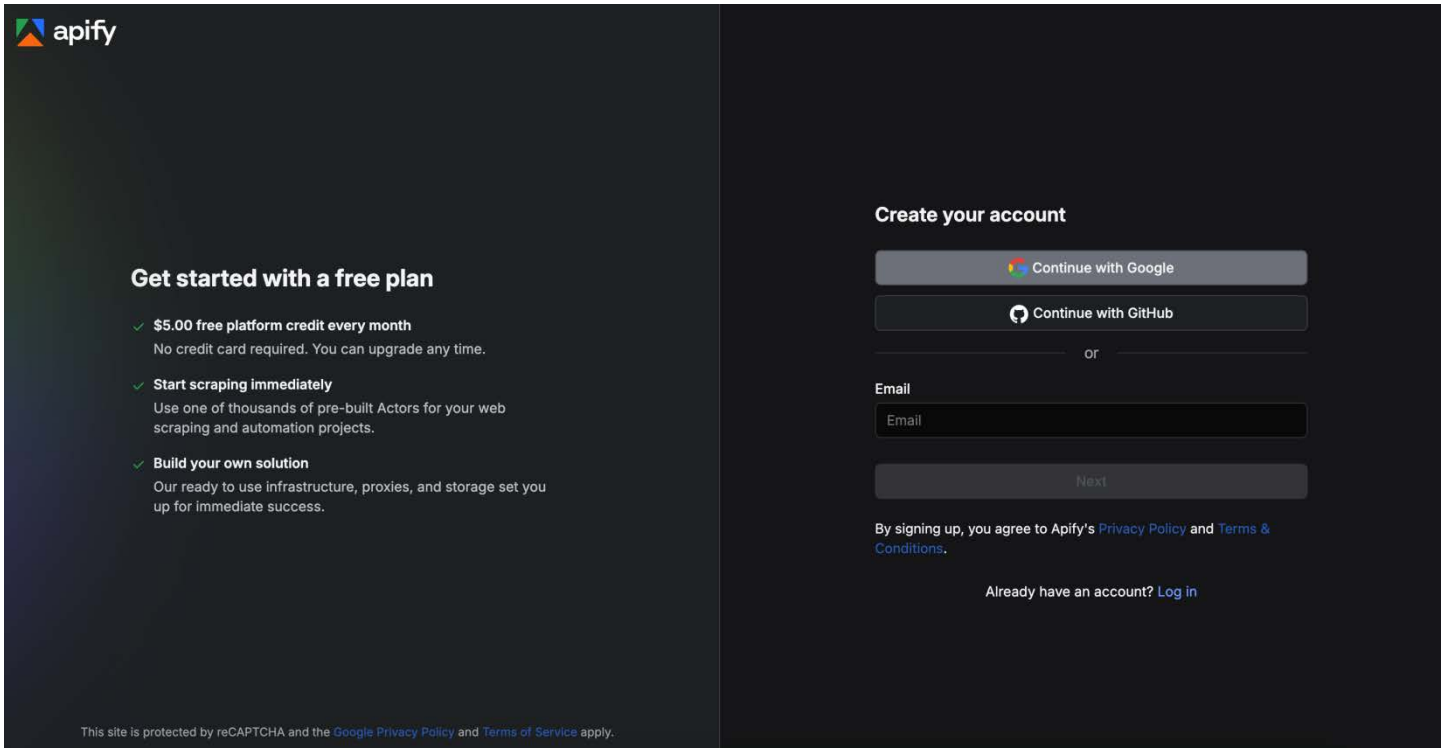


點擊 Get started 按鈕



透過 Google 帳號註冊

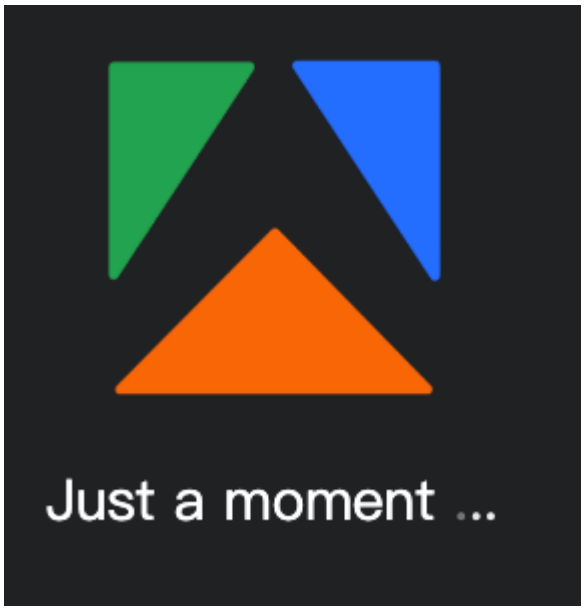
Create your account > Continue with Google



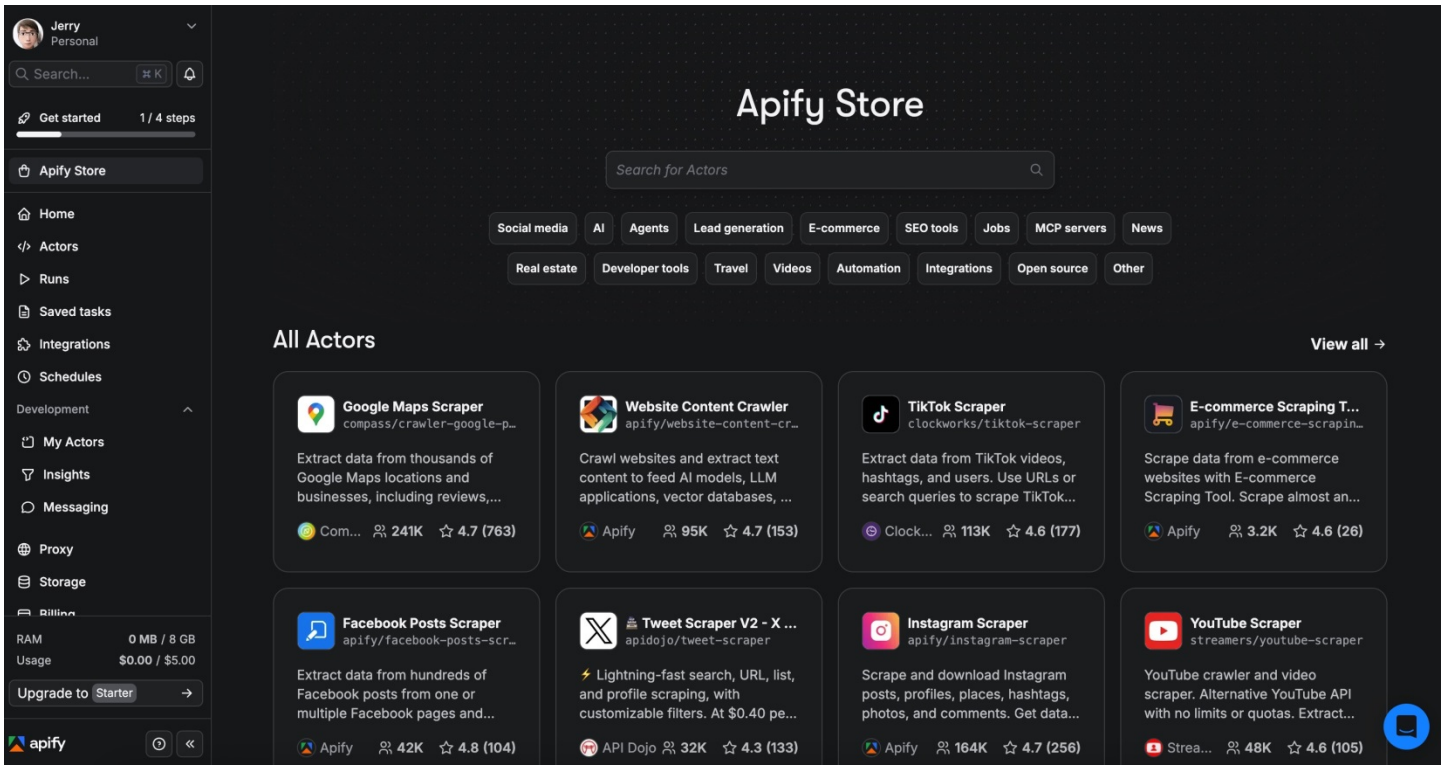
指定用來註冊的 Google 帳號

授權 Apify 進行驗證

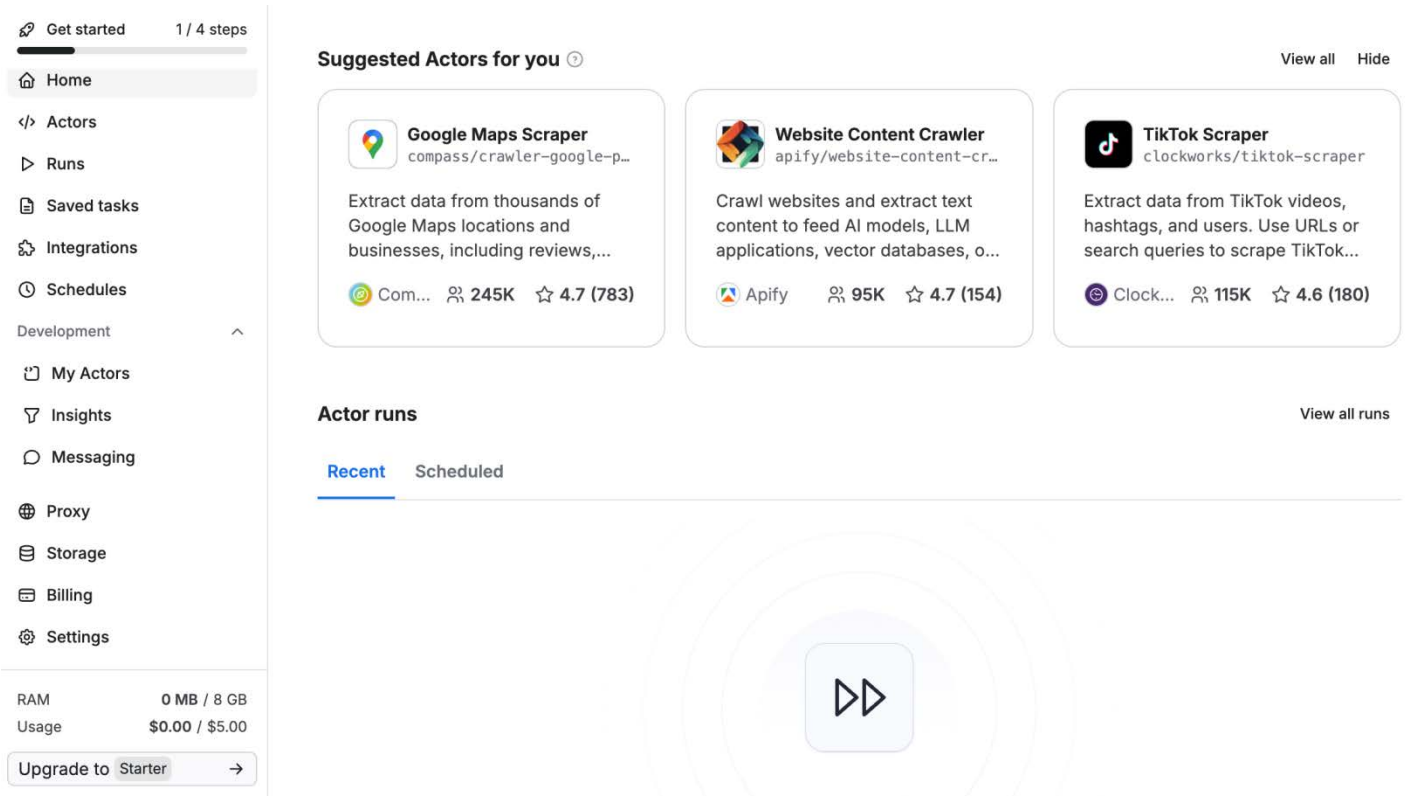
稍待片刻



自動登入首頁



Apify 平台基本介紹



Dashboard 儀表板

快速掌握：

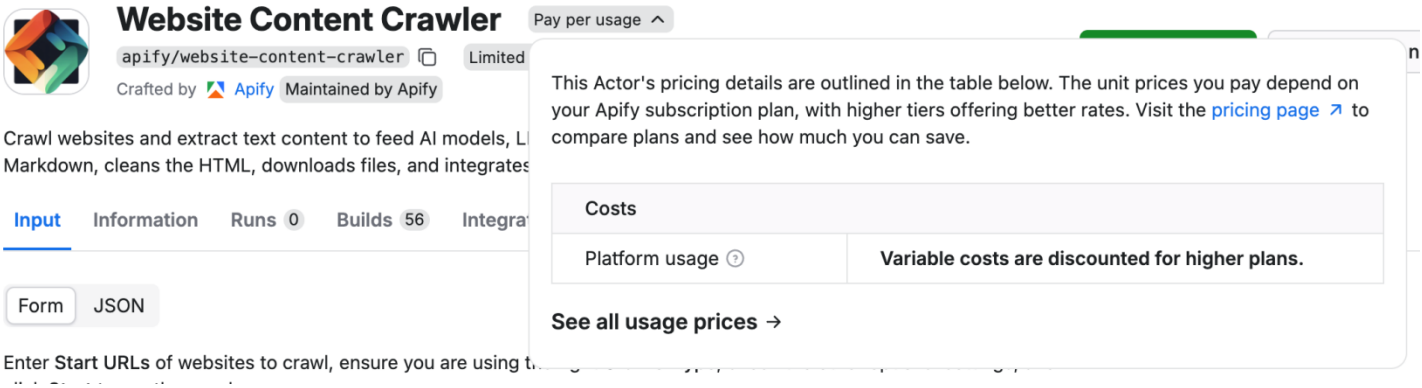
- 最近執行了哪些爬蟲
- 是否有正在跑的任務
- 執行成功 / 失敗狀態
- 使用額度

Actors 工具區

Actor Store：官方與社群提供的爬蟲工具庫

- Google Maps Scraper
- Website Content Crawler
- Instagram / Facebook / YouTube Scraper
- 電商平台（Amazon、Shopee）

注意費用描述



My actors：自己建立或 fork 的 Actor

Actor 執行流程

1. 選取 Actor
2. 進入 Input / Configuration（設定頁）
3. 填入：
    - 網址
    - 關鍵字
    - 抓取頁數
4. 按 Run
5. 等待狀態變成 Succeeded

☞ 關鍵觀念
「不需要寫程式，只是在填表單」

Storage

爬回來的資料倉庫

Datasets

- 儲存「結構化資料」
- 可直接下載成：
    - CSV
    - Excel
    - JSON

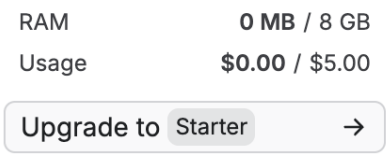
Key-Value Stores

儲存設定檔、HTML、單一內容

Request Queues

控制爬蟲要跑哪些網址，這是爬蟲內部使用

Usage & limits



簡易爬蟲流程

Actors（選工具）
↓
Run（執行）
↓
Dashboard（看狀態）
↓
Storage / Dataset（下載資料）

使用 Apify 完成第一個爬蟲

抓取新聞網頁資料

使用工具：
Web Scraper

示範網站：
中央社一手新聞 APP 即時新聞
https://www.cna.com.tw/list/aall.aspx

ChatGPT 提示詞

請幫我分析 https://www.cna.com.tw/list/aall.aspx 這個頁面的新聞標題、連結、時間、圖檔並提供 page function

轉換後的 Page Function

```
// Page function for Apify Web Scraper
async function pageFunction(context) {
    const $ = context.jQuery; // 使用 jQuery 方便操作 DOM
    const baseUrl = 'https://www.cna.com.tw';

    // 選取新聞列表的每一個 <li>
    const listItems = $('#jsMainList li').toArray();

    // 逐個抓取資訊
    const newsData = listItems.map(item => {
        const linkEl = $(item).find('a');
        const titleEl = $(item).find('h2 span');
        const dateEl = $(item).find('.date');
        const imgEl = $(item).find('img');

        return {
            title: titleEl.text().trim() || null,
            url: linkEl.attr('href') ? baseUrl + linkEl.attr('href') : null,
            date: dateEl.text().trim() || null,
            image: imgEl.attr('data-src') || imgEl.attr('src') || null
        };
    });

    // 如果頁面有下一頁，也可以加入抓取
    const nextPage = $('a.next'); // 假設下一頁按鈕有 .next class
    if (nextPage.length) {
        const nextUrl = baseUrl + nextPage.attr('href');
        await context.enqueueRequest({ url: nextUrl });
    }

    // 記錄 log
    context.log.info(`抓取 ${newsData.length} 筆新聞`);

    // 回傳資料給 dataset
    return newsData;
}
```

實作內容：

1. 輸入目標網址
   https://www.cna.com.tw/list/aall.aspx
2. 設定抓取深度（Max crawling depth）建議設定 2
3. 設定 Pseudo-URLs（有效目標頁）
   https://www.cna.com.tw/news/*
4. 設定目標欄位（Page function）：
   - 標題
   - 連結
   - 文章時間
   - 圖檔
5. 執行與查看結果（Dataset）
6. 匯出 CSV/Excel/JSON

學習成果：
✔ 完成一次完整的實際網頁抓取
✔ 學會匯出資料並使用 Dataset

抓取特定帳號的 Threads 資料

使用工具：
Threads Scraper

示範網站：
https://www.threads.net/@ashin_ig

實作內容：

1. 輸入目標網址或 username : @ashin_ig
2. 執行與查看結果（Dataset）
3. 匯出 CSV/Excel/JSON

抓取 Youtube 資料

使用工具：
Youtube Scraper (Pay Per Result)

示範網站：
https://www.youtube.com/results?search_query=powerbi

實作內容：

1. 輸入目標網址、頻道網址或查詢內容
2. 執行與查看結果（Dataset）
3. 匯出 CSV/Excel/JSON

排程、自動化

設定自動排程

Task 工作項目

特製的執行設定，Actor 是爬蟲，Task 是「執行這個爬蟲的指定方式或特別設定」。

Task 可以設定：

- 抓哪個網站或哪個分類
- 執行選項，例如：記憶體、逾時

Task 的好處：

- 同一個 Actor 建立多個 Task，每個 Task 執行不同的抓取範圍或參數
- 可以手動執行，或用排程自動執行

舉例：

- Actor：CNA 新聞爬蟲
- Task1：抓「即時新聞」
- Task2：抓「國際新聞」
- Task3：抓「財經新聞」

Schedule 排程計劃

是自動定時執行 Task 的機制，可以設定一個 Task 在特定時間自動執行。

舉例：

- Task1（政治新聞爬蟲）
    - Schedule1：每天 8:00 AM 自動執行
    - Schedule2：每小時抓一次
- Task2（國際新聞爬蟲）
    - Schedule：每天 9:00 AM 自動執行

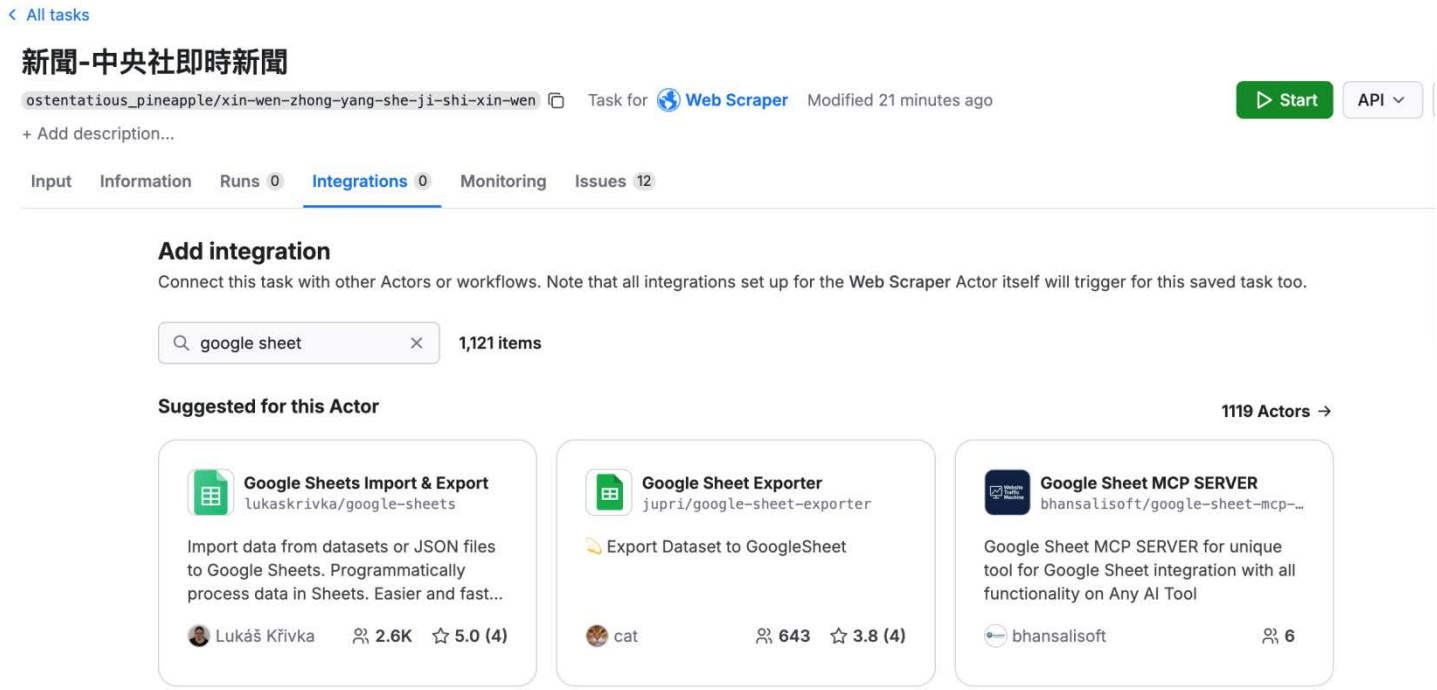
學習成果：

✔ 能建立「每日自動抓取」爬蟲

整合 Google Sheet

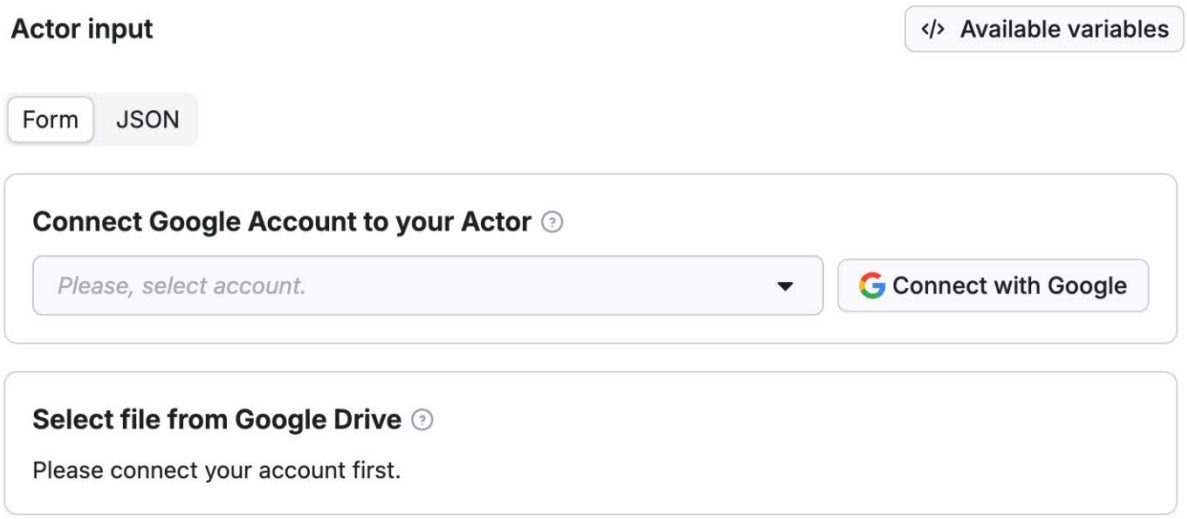
透過 Task 整合 Google sheet 將爬蟲爬回來的資料匯入至指定的 Google sheet 中。

選擇使用 Google Sheet Import & Export 整合



授予 Google Drive 存取權限

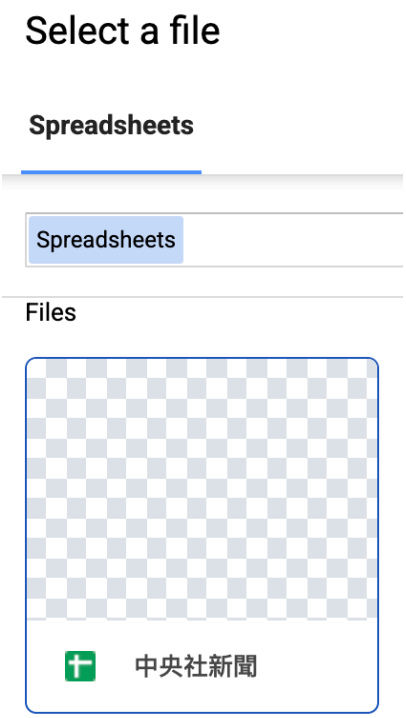
1. 連線至 Google 帳號



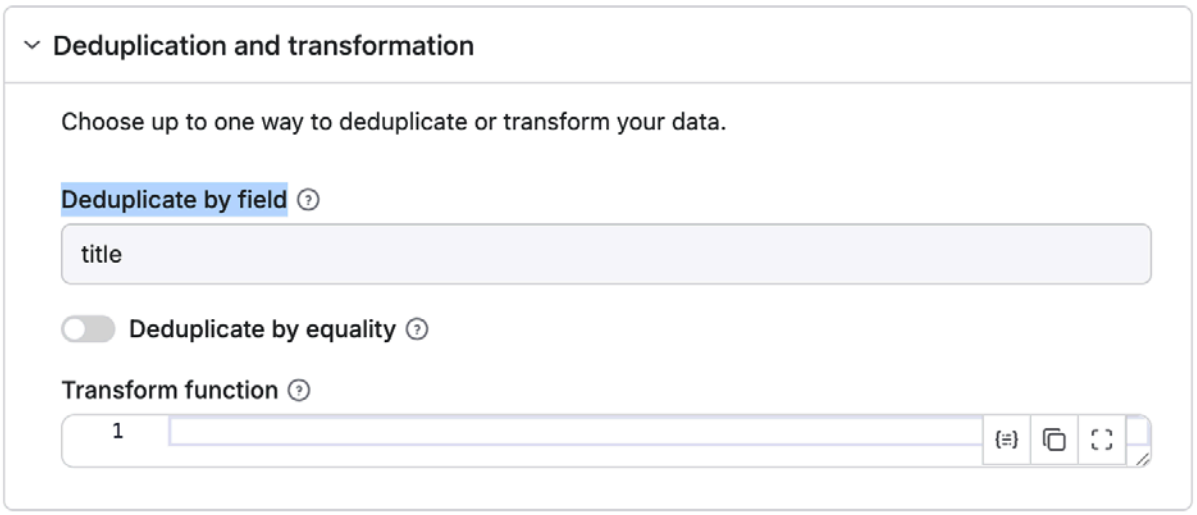
2. 授予帳號權限

3. 授予 Google Drive 權限

選取匯出檔案（請先建立中央社新聞 Google Sheet）



設定檢核是否重複的欄位（Deduplicate by field）



學習成果：
✔ 爬蟲抓取結果自動匯出到 Google Sheet 中

常見問題排除與最佳實務**

如何避免觸發反爬機制？

常見被阻擋原因

- 請求太快
- User-Agent 太明顯是爬蟲
- 連續存取相同路徑

No-code 最佳實務

1. User-Agent 偽裝
    - 使用 真實瀏覽器 UA
    - 避免預設 ApifyBot
2. 放慢速度（最重要）
    - 每頁間隔：2–5 秒
    - 限制同時請求數
3. 不要一次抓完
    - 分批抓（依日期、頁碼）
    - 使用排程慢慢累積資料

提醒

人類 3 秒看一頁，機器 0.1 秒看 100 頁 = 被封鎖

網站是動態渲染（JS）怎麼辦？

判斷

1. 開啟開發者工具檢視原始碼
2. 找不到內容文字，但畫面上「明明有」
3. 可判定為 JavaScript 動態載入

舉例

網站：https://web-scraping.dev/testimonials

- 首次載入只顯示部分 testimonial
- 往下滾動時會用 JS 發出 XHR 請求載入更多內容
  這種情況就是典型的「動態載入」

使用工具

Playwright Scraper

設定 JSON

```
{
  "startUrls": [
    {
      "url": "https://web-scraping.dev/testimonials"
    }
  ],
  "useHeadlessBrowser": true,
  "waitForSelector": ".testimonial",
  "maxRequestsPerCrawl": 1,
  "pageFunction": "async function pageFunction({ page }) {\n  // 模擬滑動觸發更多載入\n  for (let i = 0; i < 5; i++) {\n    await page.evaluate(() => window.scrollBy(0, window.innerHeight));\n    await page.waitForTimeout(1000);\n  }\n  // 抓動態載入 testimonial\n  const items = await page.$$eval('.testimonial', els =>\n els.map(el => ({\n      text: el.querySelector('p')?.innerText,\n      author: el.querySelector('h4')?.innerText\n    }))\n  );\n  return items;\n}"
}
```

抓到太多垃圾資料怎麼過濾？

常見垃圾資料

- 導覽列
- 頁尾
- 相關文章
- 廣告區塊

No-code 過濾策略

1. 從「源頭」選對範圍
    - 不要抓整頁
    - 只抓「文章區塊 container」
2. 善用 Include / Exclude
    - Include：.article-content
    - Exclude：nav, footer, aside, .ads
3. 後處理
    - 匯出 CSV / JSON
    - 用 Excel / Google Sheets 篩選

提醒

爬蟲不是一次就完美，而是 抓 → 看 → 修 → 再抓

學習成果：
✔ 能針對不同網站調整策略
✔ 能判斷 No-code 工具的可行性