全球軟硬體資訊互動平台專案規格書

版本：1.0 日期：2025-08-16 專案代號：SoftHub

### 

### **專案目標**

**建立一個多語言、使用者友善的全球軟硬體互動資訊與學習平台。此平台旨在讓普通用戶、技術愛好者及專業開發者能夠輕鬆地發現、評價、比較及交流各種軟硬體產品的使用心得與學習經驗。**

### **核心定位**

* **內容聚合與發現： 專注於聚合來自全球頂尖來源的軟硬體資訊、開源專案（Repo）及 IT 新聞。**
* **社群互動與評價： 提供一個中立的平台，讓用戶可以發表評論、評分，並圍繞特定產品或技術進行討論。**
* **輕量化託管： 初期僅託管文字、圖片及外部影片連結。我們不託管原始程式碼或影片檔案，以降低複雜度和維護成本。**
* **全球化與親民化： 支援多語言（優先支援中文、英文、日文），並採用直觀、簡潔的 UI/UX 設計，降低使用門檻。**

## 

## 爬蟲技術選型

* 爬蟲定時任務與自動化測試框架:Playwright,Puppeteer,Selenium, Github Action
* 爬蟲資料庫寄存:

1. 原始資料寄存：Supabase Storage。 優點：與主資料庫 Supabase (PostgreSQL) 無縫整合，可透過 Supabase 的 API 和權限管理（RLS）進行存取，方便後續的數據追蹤與管理。 流程：爬蟲任務抓取原始資料（HTML, JSON），直接上傳至 Supabase Storage 的一個特定 Bucket（例如 raw-data），並以 來源/日期/uuid.html 的形式命名。
2. 清洗後資料寄存：Supabase (PostgreSQL)。MDX 檔案可以視為結構化資料的一部分，儲存在 text 類型的欄位中。

## 網頁技術選型

* 全端框架: [NEXT.JS](http://next.js)，使用的庫: app router, 國際化庫:React-i18next
* CSS+UI庫 : Tailwind CSS, Ant Design, Radix UI + 自定義
* 前端認證：Supabase Auth
* 後端網頁寄存:Supabase

其它技術選型:

* 語法檢查:ESLint + Prettier
* 性能測試框架:Lighthouse CI
* 分割工具：Next.js内置代码分割 + @loadable/component
* 预加载库：quicklink（Google出品）
* 分析工具：Lighthouse CI + Treo（可视化监控）

## 系統架構

本專案採用基於 Next.js 和 Supabase 的現代化 Serverless 架構，以實現高效能、高擴展性與低維護成本。

* **數據爬取層 (Crawler Service):**
* **後端服務層 (Backend Service):**
* **前端應用層 (Frontend Application):**
* **數據儲存層 (Data Storage):**

## 各層執行細節與步驟

* 數據爬取層 (Crawler Service):
  + 觸發器: 使用 GitHub Actions 作為排程器 (Scheduler)，每日定時觸發爬蟲任務。執行環境：爬蟲腳本可以打包成 Docker 容器，在 GitHub Actions 的虛擬環境中執行。這樣可以確保執行環境的一致性。
  + 錯誤處理與日誌：爬蟲任務必須包含完整的錯誤處理機制。每次執行都應產生詳細日誌（成功爬取了多少條、哪些失敗了、失敗原因），並將日誌上傳至 Supabase Storage 或專門的日誌服務（如 Logtail）。
  + 數據校驗：在存入 PostgreSQL 之前，應使用 Zod 或類似的校驗庫對清洗後的 JSON 數據進行嚴格的結構與類型校驗，確保數據庫的完整性。
* 後端服務層 (Backend Service):
  + 由 Next.js 的 Server Components 和 API Routes 構成。
  + API Routes 負責處理前端的動態請求、使用者認證、資料庫寫入操作（如發表評論）。
  + Server Components 負責在伺服器端直接讀取 Supabase 資料庫，進行伺服器端渲染 (SSR) 或靜態網站生成 (SSG)，將預先渲染好的頁面發送給客戶端，以達到最佳的載入效能 (Performance) 和 SEO。
* 前端應用層 (Frontend Application):
  + 由 Next.js 的 Client Components 構成，使用 React 進行開發。
  + 負責處理使用者互動、動態 UI 更新和客戶端數據請求。
  + 透過 Supabase Auth 處理使用者登入、註冊和會話管理。
  + 使用 React-i18next 實現多語言切換。
* 數據儲存層 (Data Storage):
  + Supabase (PostgreSQL): 作為主資料庫，儲存所有爬取的資訊、使用者資料、評論等。
  + Supabase Storage（圖片資源）: 用於儲存使用者上傳的頭像或評論中的圖片。

## 

## **網站各頁面配置與說明**

* 登入頁 (LOGIN PAGE):
  + 佈局： 左右分欄。左側為品牌 Logo 和 Slogan；右側為登入/註冊表單，使用者帳號、密碼、忘記密碼。支援 OAuth (Email,Google, GitHub) 登入。
  + 導航： 頁面頂部保留一個極簡的 Navbar，僅包含 Logo 和語言切換選項。
* 首頁 (HOME PAGE):
  + 頂部導航欄 (Navbar):Logo（鏈回首頁）、學習進度（用戶專屬）、認證
  + 頂部搜索欄：智能搜索建議，過濾選項（內容類型、時間範圍），高級搜索鏈接
  + 徽章（展示用戶認證狀態）、語言切換下拉菜單、用戶頭像/登入按鈕
  + 左側 (Sidebar)分類:
  + 右側邊欄 (Sidebar):
    - 動態細項篩選： 根據左側選擇的大分類，動態顯示相關的細項標籤 (Tags)。例如，選擇「軟體工程師」，右側顯示 Python, Java, Rust, Go 等語言標籤供使用者篩選。
  + 中間主內容區 (Layout):
    - 今日頭條（輪播展示）
    - 排行榜模塊（帶時間篩選）：
      * 硬體：CPU 天梯圖、GPU 天梯圖。
      * 軟體/Repo：本週、本月熱門 Repo 榜（基於 Star 增長量）。
      * 熱門開源項目（按star數、貢獻者數）
      * 學習資源熱度榜
    - 個性化推薦（基於用戶興趣）
  + 底部Footer：網站導航、關於我們、隱私政策、服務條款、社交媒體鏈接、版權信息

詳情頁:

* Repo 詳情頁： 顯示 Repo 的詳細資訊（Logo、描述、Readme.md 內容）、統計數據（License, Stars, Forks, Contributors）、主要使用的程式語言、相關評論和討論。
* 硬體詳情頁： 顯示硬體的規格、評測文章連結、相關新聞、使用者評價。
* 使用者個人頁 :顯示使用者基本資料、發表的評論、收藏的項目、學習進度、各科考試題目、已獲得的個人分數與認證等。

## 組件說明:

# **Server Components**：主要負責**數據讀取與展示**。首頁的排行榜、Repo 詳情頁的 Readme 內容、文章列表等。它們在伺服器端直接存取 Supabase，將渲染好的靜態 HTML 發送給客戶端，實現極速載入和優異的 SEO。

# **API Routes / Server Actions**：主要負責**數據寫入與用戶操作**。例如：用戶登入/註冊、發表評論、收藏項目、更新個人資料等。Next.js 13+ 的 Server Actions 是處理表單提交和數據變更的更現代化選擇，能簡化客戶端與伺服器之間的代碼。

# **數據快取 (Caching)**：對於不常變動但請求頻繁的數據（如 CPU/GPU 天梯圖、熱門 Repo 榜），應在後端進行積極的快取策略。可利用 Next.js 的 fetch 快取或 ISR (Incremental Static Regeneration) 機制，設定一個合理的過期時間（例如每小時或每半天重新生成一次），大幅降低資料庫負載。

# **Client Components**：主要負責**互動性強**的 UI 元素。例如：需要使用 useState, useEffect 的組件、互動式表單、語言切換下拉菜單、響應點擊事件的按鈕等。

# **狀態管理**：

# **伺服器狀態**：推薦使用 React Query (TanStack Query) 或 SWR 來管理來自 API 的數據。它們能自動處理快取、重新驗證和樂觀更新，與 Server Actions/API Routes 完美配合。

# **全域客戶端狀態**：對於如用戶認證狀態、主題（深/淺色模式）等全域狀態，可使用 Zustand 或 Jotai，它們比 Redux 更輕量、更易用。

# **UI/UX 優化**：使用 Radix UI 作為無頭 (Headless) UI 元件庫，可以確保可訪問性 (Accessibility) 並提供最大的自定義彈性，再結合 Tailwind CSS 進行樣式設計。Ant Design 則可用於後台管理頁面，快速搭建功能複雜的介面。

## 網站內容的來源網站

* REPO來源:Github API,Hugging Face API,GitLab API,SourceForge API, Youtube API
* IT新聞來源: <https://juejin.cn/>, <https://www.tomshardware.com/>, <https://www.wired.com/>, <https://www.cnet.com/>
* 學習資源來源:Udemy API、Coursera API
* CPU GPU天梯排名https://www.passmark.com/
* 資料庫排名: <https://db-engines.com/en/ranking>
* LLMs排名https://huggingface.co/spaces/open-llm-leaderboard/open\_llm\_leaderboard#/
* 台灣TechOrange 科技報橘<https://techorange.com/>
* INSIDE 硬塞的網路趨勢觀察<https://www.inside.com.tw/>
* 電腦王阿達的3C胡言亂語<https://www.kocpc.com.tw/>
* T客邦<https://www.techbang.com/>
* 36氪<https://36kr.com/>
* 掘金https://juejin.cn/

## 商業模式

1. **初期 (上線後 0-6 個月):**
   1. **原生廣告（與內容融合）:將廣告設計得與普通內容卡片樣式一致，僅在角落標註「廣告」或「贊助」。例如，在 Repo 列表中的特定位置插入一個軟體廠商的工具介紹。**
   2. **廣告收入:** 在頁面的非核心位置（如右側邊欄、文章底部）放置 Google AdSense 或類似的內容相關廣告。
   3. **聯盟行銷 (Affiliate Marketing):聯盟行銷： 透明化：在頁面底部或相關連結旁明確告知用戶「本頁面包含聯盟連結，您的購買將支持本站運營」，以建立信任。 硬體比價：在硬體詳情頁提供來自多個電商（Amazon, Newegg 等）的帶有聯盟 ID 的連結，方便用戶比價。**
2. **中期 (上線後 6-18 個月):**
   1. **會員登錄+付費訂閱模式:**
      1. **登入訂閱者(免費):**每日免費觀看3篇付費文章。第四篇開始只能看到一部分內容。
      2. **進階付費使用者:**免費觀看所有付費文章。
      3. **白金付費使用者:**免費觀看所有付費文章，**除進階付費使用者有的功能以外，還有**進階篩選功能（如按公司、貢獻者地理位置篩選 Repo。搶先體驗新功能。提供「數據導出 (CSV/JSON)」功能，讓他們可以對感興趣的數據進行離線分析。
3. **長期 (上線後 18+ 個月):**
   1. **企業服務 (B2B):**
      1. **贊助內容:** 允許軟硬體廠商付費發布「精選文章」或在排行榜中獲得「贊助商」標示。
      2. **人才招聘:** 開設招聘板塊，企業可付費發布與平台技術標籤相關的職位。**人才招聘**：可以與用戶的「技能標籤」和「認證」掛鉤，為企業提供更精準的人才篩選服務。
      3. **數據 API 服務**：將脫敏、聚合後的趨勢數據以 API 形式提供給市場分析機構或企業。例如，提供「特定技術（如 Rust）在過去一年內的 Repo 增長趨勢」的 API，供投資機構或市場研究公司訂閱。

其它現金流來源:

* 技術認證體系：基礎認證（免費）專業認證（$49/次）企業認證（$299/帳號）數據服務：
* 企業市場趨勢報告:技術棧分析，薪資水平數據
* 招聘解決方案: 智能人崗匹配,技術能力評估,招聘流程管理

競品優缺點分析與借鏡

* <https://www.producthunt.com/>
* <https://sourceforge.net/>
* <https://www.w3schools.com/>
* <https://openrouter.co/appstore>

網站頁面配置說明:

**預設語言**：偵測 navigator.language 或 HTTP Accept-Language 請求頭，自動切換至對應語言，並允許用戶手動覆蓋。

1. 登入頁面(Login Page):只分左右兩頁，左邊一個簡潔的搜尋欄位navbar。右邊使用者登入頁面欄位，佈局：左側除了 Logo 和 Slogan，可以加入一個動態輪播，展示平台的核心價值（例如：「發現下一個偉大的開源專案」、「追蹤最新的硬體性能排行」）。安全性：所有密碼傳輸必須使用 HTTPS，並在後端進行安全的雜湊儲存 (hashing)。
2. **首頁 (Home Page)**：**個人化推薦**：初期可基於用戶「關注的標籤」和「瀏覽歷史」進行簡單的內容推薦。引入協同過濾演算法，推薦與其行為相似的用戶所喜歡的內容。**權限顯示**：**管理員**：登入後導航欄增加「後台管理」入口，進入數據儀表板、用戶管理、內容審核等頁面。**付費/一般用戶**：首頁佈局相同，但在內容區塊，付費文章對一般用戶顯示「訂閱以閱讀全文」的遮罩。
3. **使用者個人頁 (User Profile Page)**： 公開頁面和私有頁面。**公開頁面**：其他人可見，展示用戶頭像、暱稱、公開發表的評論和獲得的徽章。**私有儀表板 (Dashboard)**：僅用戶自己可見，包含「我的收藏」、「學習進度」、「帳號設定」、「訂閱管理」等。

* **頂部導航欄位：首頁、logo、學習進度、認證、語言切換**
* **頂部中間搜尋欄位:各類排行榜**
* **左側Sidebar分類欄位:各類工程師**
* **右側Sidebar欄位:各工程師的專業細項**
* **中間Layout 欄位:今日科技頭條、底部Footer欄位:**

### 

附件:

各工程師的專業細項 (右側邊欄篩選標籤)

資料來源與爬取策略

## 任務說明:

這份PRD我寫得不夠完善。

你現在是資深全棧架構師，請完成以下任務，文字說明用 Markdown, MDX格式，圖說用mermaid語法:

任務 1 補充、修正此專案PRD規格書:

1. 修正、補充各層執行細節與步驟
2. 修正、補充各頁面的配置與說明
3. 修正、補充商業模式
4. 網站頁面配置說明
5. 修正、補充各工程師的專業細項，常用軟體、專業...等。比如常用軟體Revit、Ansys..等。
6. 修正、補充Server component vs. Client component 分配說明
7. 根據修正後的內容修改各流程圖:

* 系統架構圖([Architecture Diagrams](https://mermaid.js.org/syntax/architecture.html))
* 爬蟲運作流程-序列圖([Sequence Diagram](https://mermaid.js.org/syntax/sequenceDiagram.html))
* 網頁運作流程-序列圖([Sequence Diagram](https://mermaid.js.org/syntax/sequenceDiagram.html))
* 爬蟲程式測試流程-序列圖([Sequence Diagram](https://mermaid.js.org/syntax/sequenceDiagram.html))
* 網頁測試流程-序列圖([Sequence Diagram](https://mermaid.js.org/syntax/sequenceDiagram.html))
* 專案文件與目錄檔案結構樹狀圖([Treemap Diagram](https://mermaid.js.org/syntax/treemap.html#treemap-diagram))
* Server component vs. Client component 結構樹狀圖([Treemap Diagram](https://mermaid.js.org/syntax/treemap.html#treemap-diagram))