🚀 一個月後端 MVP 衝刺計畫 (主線與延伸)

目標： 在 60 小時內，實現一個具備使用者認證和單一核心資源 CRUD 的 RESTful API 服務。

📌 第一部分：主線任務 (MVP - 必須完成)

🗓️ 第 1 週：基礎架構與使用者認證 (約 14 小時)

任務分類	每日焦點 (2 小時)	關鍵知識點	達成標準
規劃與初始化	**Day 1-2：** 確定 MVP 範圍、設計 **Users** 和 **核心資源** 的 ERD 和 API 規格。啟動 Django 專案和 App。	RESTful 原則、Django 專案/App 結構。	專案骨架建立，API 規格定稿。
認證 Model	**Day 3：** 閱讀 Custom User Model 文件。實作 Custom User Model，運行 Migration。	`AbstractUser` vs `AbstractBaseUser` 差異。	User Model 實作完成。
認證 API	**Day 4-5：** 安裝 Simple JWT。實作 **註冊** Serializer 和 View。定義路由。	DRF Serializer 基礎、Simple JWT 配置。	註冊 API 測試通過。
認證 API	**Day 6-7：** 配置 **登入/Token 路由**。測試註冊、登入功能，確保 JWT 返回正常。	JWT Token 機制、Token 獲取路由。	登入 API 測試通過。

🗓️ 第 2 週：核心資源 CRUD 與權限控制 (約 14 小時)

任務分類	每日焦點 (2 小時)	關鍵知識點	達成標準
核心 Model	**Day 8-9：** 建立 **核心資源 Model** (e.g., `Todo`)。建立與 `User` 的 **ForeignKey** 關聯。	Model 關聯 (ForeignKey)、`on_delete` 參數。	核心 Model 定義完成。
CRUD 實作	**Day 10-12：** 為核心資源建立 Serializer 和 ViewSets (Create, Read, Update, Delete)。	DRF ViewSets/Generic Views 的用法。	未經權限控制的 CRUD 功能實現。
權限控制	**Day 13-14：** 為 ViewSets 增加 `IsAuthenticated` 權限。實作 `IsOwner` 或 `IsAuthor` 的自定義權限，確保資料隔離。	DRF Permissions 機制、`request.user` 使用。	基本的資料隔離和權限控制實現。

🗓️ 第 3 週： API 測試與業務邏輯 (約 14 小時)

任務分類	每日焦點 (2 小時)	關鍵知識點	達成標準
測試入門	**Day 15-17**： 學習 DRF 測試 Client 的使用。為 **註冊/登入** 和 **CRUD 功能** 撰寫單元測試。	Django Test Client、測試資料準備 (setup)。	核心功能單元測試完成。
業務邏輯	**Day 18-20**： 加入一個簡單的業務邏輯 (e.g., 在 `Todo` 完成時，自動設定 `completed_at` 時間戳)。優化 View/Serializer。	Serializer 的 `validate` 和 `update` 方法。	業務邏輯實作並通過測試。
代碼審查	**Day 21**： 進行一次程式碼回顧，清理註釋、確保命名規範。	Python PEP 8 規範。	程式碼可讀性提升。

🗓️ 第 4 週：文件、部署準備與總結 (約 18 小時)

任務分類	每日焦點 (2 小時)	關鍵知識點	達成標準
API 文件	**Day 22-23**： 安裝並配置 `drf-spectacular`。生成 **Swagger/OpenAPI** 文件。	drf-spectacular 的配置與使用。	產出可瀏覽的 API 文件頁面。
部署準備	**Day 24-25**： 配置 **環境變數** (`.env`)。準備部署所需文件 (`requirements.txt`, `Procfile` 等)。	`python-decouple` 或 `django-environ` 使用。	專案具備部署至雲平台的條件。
總結與回顧	**Day 26-28**： 修復最後的 Bug。撰寫 **README.md** 專案說明文件。	-	MVP 最終版本和完整文件。

🚀 第二部分：延伸任務 (如果時間充裕)

如果主線任務超前完成，優先執行 **延長任務一**。

延長任務一：嘗試線上部署 (高價值)

- **目標：** 將您的 API 部署到線上環境。
- **步驟：**
    1. 配置 **PostgreSQL** 或 **MySQL** 資料庫。
    2. 將專案部署到 **Render.com** 或其他 PaaS 平台。
    3. 配置 Gunicorn (WSGI Server) 確保服務穩定運行。

延長任務二：進階功能與安全性

- **目標：** 提升使用者體驗和安全性。
- **步驟：**
    1. 實作 **Token Refresh** 機制。
    2. 加入 **List API 篩選** (Filtering) 和 **分頁** (Pagination) 功能。
    3. 增加 **Email 欄位唯一性** 檢查。
