專案開發與遷移規範 (Project Standards)

本文件定義了將 Google Apps Script (GAS) 遷移至 Node.js/Supabase 架構的標準作業流程與檔案結構。所有 AI Agent 與開發者必須嚴格遵守此規範。

1. 核心原則 (Core Principles)

- **服務化 (Service-Oriented)**：每一支獨立的 GAS 腳本應遷移為一個獨立的 `Service` 模組。
- **資產留存 (Asset Retention)**：嚴禁覆蓋舊的計畫書。所有遷移計畫必須歸檔。
- **關注點分離 (Separation of Concerns)**：資料庫連線、商業邏輯、測試腳本必須分開存放。

2. 目錄結構 (Directory Structure)

專案必須維持以下結構：

```
my-project/
├── docs/
│   └── plans/              # 存放所有遷移計畫書 (不可覆蓋)
├── src/
│   ├── lib/
│   │   └── db.ts           # 共用的 PostgreSQL (pg) 連線池實例
│   ├── services/           # 核心商業邏輯 (從 GAS 遷移而來)
│   │   ├── BolService.ts
│   │   ├── InventoryService.ts
│   │   └── ...
│   └── scripts/            # 單純的測試進入點 (Entry Points)
│       ├── 01_Test_BolService.ts
│       ├── 02_Test_InventoryService.ts
│       └── ...
└── .env                    # 環境變數 (嚴禁上傳 GitHub)
```

3. 命名慣例 (Naming Conventions)

3.1 遷移計畫書 (Migration Plans)

必須包含序號與功能名稱，存放於 `docs/plans/` 。

- 格式： `{Seq}_{FeatureName}_Migration.md`
- 範例： `01_BolTool_Migration.md` , `02_Inventory_Migration.md`

3.2 服務模組 (Service Files)

存放於 `src/services/` ，使用 PascalCase。

- 格式： `{FeatureName}Service.ts`
- 範例： `BolService.ts`

3.3 測試腳本 (Test Scripts)

存放於 `src/scripts/` ，對應計畫書序號。

- 格式： `{Seq}_Test_{FeatureName}Service.ts`
- 範例： `01_Test_BolService.ts`

4. 開發流程 (Development Workflow)

對於每一支新的 GAS 遷移任務，請依照以下步驟執行：

1. **初始化**：
   - 在 `docs/plans/` 建立新的計畫書 (例如 `02_Inventory_Migration.md`)。
   - **嚴禁** 修改或覆蓋 `src/scripts/` 中既有的測試腳本。
2. **實作 Service**：
   - 在 `src/services/` 建立對應的 Service 檔案。
   - 從 `src/lib/db.ts` 引入共用的 `pool` 物件。
   - 實作業務邏輯 (包含 SQL Transaction 處理)。
3. **建立測試**：
   - 在 `src/scripts/` 建立新的測試腳本。
   - 僅引入 Service 函式進行呼叫，保持腳本簡潔。
4. **驗收**：
   - 執行 `npx tsx src/scripts/{Seq}_Test_....ts` 確認無誤。