貨件追蹤系統 專案技術與需求文件

專案名稱：TailorMed Tracking System
適用對象：專案管理與開發團隊
文件性質：技術與需求存檔
更新日期：2026-01
文件版本：v1.0

本文件彙整專案之需求概覽、開發說明與時程規劃，
用於專案交付存檔與後續維護參考。

TailorMed Tracking System 專案技術與需求說明

> *本文件提供需求、設計與時程參考，不含前台使用教學內容。*

一、系統需求概覽

1. 專案簡介與目標

- 目的：提供收送件之貨況查詢（客戶前台）與狀態資料查詢（API），提升透明度、降低內部人工查詢作業及提升客戶服務品質。
- 交付：前端查貨頁（index.html）、時間軸模組（兩種樣式與事件節點）、Netlify Functions API、Airtable 連結、監控 Dashboard（可選）。

2. 角色與利害關係人

- 最終用戶：Shipper/Consignee（查詢貨況）。
- 後台營運人員：維護資料與回覆需求。
- 系統維運：萬能數維有限公司。
- 第三方：Netlify、Airtable。

3. 使用情境（User Stories）

- 作為顧客，輸入 Job No. 與 Tracking No. 查詢貨件狀態與時間軸。
- 作為營運，可在 Airtable 更新各階段時間欄位，前台即時反映。
- 作為維運，可透過監控頁了解 API 成功率、平均回應時間與最近請求（需有監控 Dashboard 功能）。

4. 範圍與不在範圍

- 範圍：國內及國際運輸單件查詢、時間軸視覺、基礎監控（可選）、自訂網域部署、基本安全驗證與限流。
- 不在範圍：會員系統、權限分級、支付、第三方物流 API 聚合、報表 BI、長期監控資料庫持久化（可選）。

5. 功能需求（Functional）

- F1 前台查詢頁：表單（Job No./Tracking No.）。
- F2 時間軸模組：依資料呈現 Executed/Processing/Pending；支援 Domestic, Import/Export/Cross 樣式分歧；特殊事件 Dry Ice Refilled(Terminal), Dry Ice Refilled。
- F3 後端 API（可選）：
  - GET /api/tracking?orderNo&trackingNo：提供客戶點選連結後可直接查貨。
  - GET /api/health：健康檢查。
  - GET /api/monitoring/Admin：監控統計與最近請求。
- F4 資料來源：Airtable（Tracking 表單），若無資料回傳 404。
- F5 安全與限流：API key（可選）與 IP-based rate limit；CORS 設定。
- F6 部署與網域：Netlify Pro，redirects 與 netlify.toml 管理路由。

6. 非功能需求（NFR）

- 可用性：P95 回應 < 1.5s（Airtable 正常時）。
- 可維運性：環境變數管理、日誌可追蹤 requestId/IP。
- 可擴充性：時間軸節點可擴增；支援更多運輸型別。
- 安全性：CORS 白名單、API Key 驗證（如開啟）、不回傳敏感欄位。
- 可測試性：Staging 與 Production；接口以例外/404 區分。

7. 業務規則（BR）

- BR1 以 Airtable Tracking 表中欄位為單一真實來源（SSOT）。
- BR2 若 A 階段時間存在即為 Executed，下一未填階段為 Processing...，再往後為 Pending...。

- BR3 Dry Ice Refilled
  - Dry Ice Refilled(Terminal)：僅在 In Transit 之後且事件為 Yes/True 才顯示。
  - Dry Ice Refilled：僅在 Destination Customs Process 之後且事件為 Yes/True 才顯示。
- BR4 查無資料即回 404；Airtable 失效時回 500 並記錄錯誤。

8. 介面與規格摘要

- 前端：Pug + Stylus（RWD），URL 支援 ?orderNo=XXXXXX & trackingNo=XXXXXX（可選）。
- 後端：Netlify Functions（Node），單入口 routing。
- 資料：Airtable REST SDK。
- 監控（可選）：/api/monitoring/stats + Admin 頁（Chart.js）。

9. 驗收條件（UAT）

- 依 3 組以上有效資料可正確呈現兩種樣式時間軸與事件節點。
- API 正確回 200/404/429/500 各場景。
  - 200：請求成功
  - 404：查無資料
  - 429：查詢過於頻繁
  - 500：內部伺服器錯誤
- 網域可用、redirects 與 netlify.toml 路由生效。
- 監控頁（可選）可顯示今日/月累計、平均回應、成功率、最近 20 筆請求。

10. 上線與維運

- 部署：GitHub → Netlify 自動部署；Staging → Production。
- 環境變數：AIRTABLE_API_KEY / BASE_ID / TABLE / API_KEY_xxx。
- 監控：Netlify logs + Admin 監控頁（可選）。
- 備援：Airtable 異常回退 500。

二、TailorMed 貨件追蹤系統開發說明文件

技術架構

- 前端框架: HTML5 + CSS3 + JavaScript
- 模板引擎: Pug
- 樣式預處理器: Stylus
- 字型: Noto Sans (Google Fonts)
- 部署平台: Netlify
- 版本控制: Git

功能需求

1. 時間軸追蹤系統 (Timeline Tracking System)

1.1 支援的追蹤類型

- DOMESTIC (國內運輸) 4 個節點
- IMPORT/EXPORT (進出口運輸/跨國貿易) 7 個節點 + 2 個事件
- ORDER COMPLETION (訂單完成) 3 個節點

1.2 節點狀態系統

- Completed (已執行): 藍色實心圓 + 白色實心圓 + 勾選圖標
- Processing (處理中): 桃色實心圓 + 白色實心圓 + 卡車圖標 + 狀態文字有 Gradient Shine 動畫
- In Transit (轉運中): 桃色實心圓 + 白色實心圓 + 飛機運輸圖標 + 狀態文字有 Gradient Shine 動畫
- Pending (待處理): 灰色實心圓圈 + 白色實心圓 + 無圖標
- Acomplishment (已完成): 藍色圓圈 + 白色勾選圖標

- Shipment Delivered (貨件已完成遞交): 藍色圓角卡 + TM Mark

1.3 互動效果

- 已執行節點 Hover 效果: 圓圈放大 1.10 倍 + 陰影效果
- 已執行完成狀態 Hover: :before 從白色變為 accent-color，圖標從藍色變為白色
- 處理中/轉運中節點 Hover 效果: 圓圈放大 1.10 倍 + 陰影效果
- 處理中/轉運中完成狀態 Hover: :before 從白色變為 accent-color，圖標從變為白色
- Processing/In Transit 狀態: 文字有 Gradient Shine 動畫效果

2. 視覺設計系統

2.1 顏色規範

- Primary Color: 主要藍色 (用於完成狀態、連接線)
- Accent Color: 輔助桃色 (用於漸層效果)
- Neutral Colors: 淺灰藍色系 (用於待處理狀態、文字)

2.2 連接線樣式

- DOMESTIC: 漸層連接線 (primary-color → accent-color)
- IMPORT/EXPORT/CROSS: 漸層連接線 (primary-color → accent-color)
- SHIPMENT DELIVERED: 純色連接線 (primary-color)

2.3 節點樣式

- 圓圈尺寸: 40x40px
- 連接線高度: 5px
- 日期字體: 1.1rem, Title Case (如 "Oct 13")
- 狀態文字: 0.9rem
- 時間文字: 0.85rem
- 狀態區域寬度: 110px

3. 動畫效果

3.1 Gradient Shine 動畫

- 觸發條件: Processing 和 In Transit 狀態
- 動畫效果: 白色漸層從左到右掃過文字
- 動畫時長: 2 秒循環
- 動畫範圍: 僅限於狀態文字區域

3.2 Hover 動畫

- 節點圓圈: 縮放效果 + 陰影
- 過渡時間: 0.3 秒
- 顏色變化: 完成狀態的顏色反轉效果

4. 響應式設計

4.1 桌面版本

- 最大寬度: 1200px
- 間距: 40px padding
- 字體大小: 標準尺寸

4.2 移動版本

- 媒體查詢: @media (max-width: 768px)
- 字體縮放: 相對縮小
- 間距調整: 減少 padding

具體實現需求

1. 節點內容結構

1.1 標準節點

![][image1]

1.2 Order Completion 特殊節點

![][image2]

2. 特殊組件需求

2.1 Order Completion 容器

- 尺寸: 55x40px
- 圓角: 5px
- 背景: primary-color
- 圖標: TailorMed mark (36x24px, 原色)

2.2 事件標籤 (Event Tags)

- 乾冰補充事件: 垂直連接線 + 標籤文字
- 位置: 時間軸下方
- 樣式: 淺藍色背景 + 白色圖標

3. 狀態流程定義

3.1 DOMESTIC 流程

1. Order Created (訂單建立)
2. Shipment Collected (貨件收取) Processing 狀態
3. In Transit (運輸中) Pending 狀態
4. Shipment Delivered (貨件送達) Pending 狀態

3.2 IMPORT/EXPORT 流程

1. Order Created (訂單建立)
2. Shipment Collected (貨件收取)
3. Origin Customs Process (起點海關處理)
4. In Transit (運輸中) Processing 狀態
5. Destination Customs Process (目的地海關處理)
6. Out for Delivery (配送中)
7. Shipment Delivered (貨件送達)

技術實現細節

1. 檔案結構

```
src/Projects/TailorMed/track/
├── Templates/
│ ├── tracking_ui.pug (主頁面)
│ └── components/
│ └── timelineTrack.pug (時間軸組件)
├── Styles/
│ ├── main.styl (主樣式)
│ ├── variables.styl (變數定義)
│ └── components/
│ └── timelineTrack.styl (時間軸樣式)
├── Javascript/
│ ├── app.js (應用邏輯)
│ ├── config.js (配置檔案)
│ └── interaction.js (互動邏輯)
├── Assets/
│ ├── icon-check.svg (勾選圖標)
│ ├── icon-truck.svg (卡車圖標)
```

| ├── icon-inTransit.svg (飛機圖標)
| └── tailormed-mark.svg (品牌標誌)
└── compile.js (編譯腳本)

2. 編譯流程

- Pug → HTML: 模板編譯
- Stylus → CSS: 樣式編譯
- 資源複製: 圖片和 JavaScript 檔案複製到 dist 目錄

3. 部署需求

- 靜態檔案服務: 支援 HTML, CSS, JS, 圖片
- 路由設定: SPA 路由支援 (Netlify _redirects)
- CDN: 字型載入優化

效能需求

1. 載入效能

- 字型預載: Google Fonts preconnect
- 圖片優化: SVG 格式，適當尺寸
- CSS 優化: Stylus 編譯，移除未使用樣式

2. 互動效能

- 動畫流暢度: 60fps 動畫效果
- 響應時間: Hover 效果 < 100ms
- 記憶體使用: 最小化 DOM 操作

3. 瀏覽器支援

- 現代瀏覽器: Chrome, Firefox, Safari, Edge
- CSS 支援: Flexbox, CSS Grid, CSS Variables
- JavaScript: ES6+ 語法支援

維護需求

1. 程式碼品質

- 模組化設計: 組件化架構
- 樣式分離: 變數統一管理
- 註解完整: 中文註解說明

2. 擴展性

- 新增時間軸類型: 易於擴展新的追蹤流程
- 狀態系統: 支援新增節點狀態
- 主題系統: 支援顏色主題切換

3. 版本控制

- Git 管理: 功能分支開發
- 部署自動化: Netlify 自動部署
- 回滾機制: 快速回滾到穩定版本

驗收標準

1. 功能驗收

- [ ] 輸入 Order No. 及 Tracking No. 正確可顯示貨件狀況
- [ ] 兩種運輸狀態以及運輸完成下之時間軸類型正常顯示

- [ ] 時間軸可以正確反應當下訂單狀態及時間戳記
- [ ] 節點狀態依訂單狀態進行正確切換
- [ ] 動畫效果流暢運行
- [ ] Hover 互動正常響應

2. 視覺驗收

- [ ] 顏色規範正確應用
- [ ] 字體大小符合設計
- [ ] 間距對齊準確
- [ ] 響應式設計正常

3. 效能驗收

- [ ] 頁面載入時間 < 3 秒
- [ ] 動畫幀率 ≥ 50fps
- [ ] 記憶體使用穩定
- [ ] 跨瀏覽器相容性

前期工作成果

1. 需求文件整理

基於客戶訪談結果，我們整理了完整的系統需求文件，包含：

1.1 功能需求分析

- 核心功能識別: 貨件追蹤系統需要支援兩種不同的運輸流程類型
- 使用者故事: 從訂單客戶角度定義完整的追蹤體驗需求
- 業務流程梳理: 詳細分析了 DOMESTIC、IMPORT/EXPORT、CROSS TRADE、ORDER COMPLETION 三種狀態
- 狀態定義: 明確定義了 Completed、Processing、In Transit、Pending 四種節點狀態

1.2 非功能性需求

- 效能要求: 頁面載入時間 < 3 秒，動畫幀率 ≥ 50fps
- 可用性要求: 支援現代瀏覽器，響應式設計
- 維護性要求: 模組化設計，易於擴展新的追蹤流程

1.3 需求優先級排序

- 高優先級: 時間軸顯示、節點狀態切換、基本互動效果
- 中優先級: 動畫效果、響應式設計、特殊節點樣式
- 低優先級: 主題切換、進階互動功能

2. 技術可行性評估

2.1 技術選型分析

- 前端框架: 選擇 HTML5 + CSS3 + JavaScript 原生技術棧
  - 評估理由: 輕量級、快速載入、易於維護、相容性佳
  - 替代方案: React/Vue.js (考慮到專案規模，原生技術更適合)
- 模板引擎: 採用 Pug
  - 評估理由: 簡潔語法、組件化支援、編譯時優化
  - 效能優勢: 預編譯模板，減少運行時處理
- 樣式預處理器: 選擇 Stylus
  - 評估理由: 簡潔語法、變數管理、嵌套支援
  - 維護優勢: 統一的樣式變數管理，易於主題切換

2.2 技術風險評估

- 瀏覽器相容性: 低風險 使用標準 Web 技術

- 效能風險: 低風險 輕量級實現，優化動畫效果
- 維護風險: 低風險 清晰的代碼結構和文檔

2.3 技術債務管理

- 代碼品質: 建立統一的代碼規範和註解標準
- 文檔維護: 完整的技術文檔和使用說明
- 版本控制: Git 分支管理策略

3. 系統架構設計

3.1 整體架構

![][image3]

3.2 組件架構設計

- 模組化設計: 每個時間軸類型為獨立組件
- 狀態管理: 統一的節點狀態系統
- 樣式系統: 基於變數的主題系統
- 互動系統: 分離的 JavaScript 互動邏輯

3.3 資料流設計

- 靜態資料: 時間軸配置和節點資訊
- 動態狀態: 節點狀態和動畫效果
- 使用者互動: Hover 效果和狀態切換

3.4 擴展性設計

- 新增時間軸類型: 通過配置檔案快速添加
- 狀態擴展: 統一的狀態系統支援新狀態
- 主題系統: 基於 CSS 變數的主題切換

3.5 資料流程

1. **使用者請求** → 前端發送 HTTP 請求到 /api/tracking
2. **CDN 處理** → Netlify CDN 處理靜態資源和路由
3. **Function 執行** → Netlify Function 處理 API 請求
4. **資料查詢** → Function 呼叫 Airtable API 查詢資料
5. **資料處理** → Function 處理和轉換資料格式
6. **回應資料** → Function 返回 JSON 回應給前端
7. **前端顯示** → 前端接收資料並顯示時間軸

4. 資料庫設計

4.1 資料庫平台

- 平台: Airtable
- 資料表: Tracking
- 認證方式: Personal Access Token (PAT)

4.2 資料結構設計

由於此專案主要為前端展示系統，資料庫設計主要針對靜態配置資料：
![][image4]

4.3 資料管理策略

- 靜態配置: 時間軸結構和節點資訊
- 動態狀態: 即時追蹤狀態更新
- 快取策略: 本地儲存常用配置
- 版本控制: 配置檔案版本管理

4.4 資料驗證

- 結構驗證: 確保時間軸配置完整性
- 狀態驗證: 驗證節點狀態的有效性

- 一致性檢查: 確保資料間的一致性

5. 介面設計規範

5.1 視覺設計系統

色彩系統

- Primary Color: 主要藍色 (#143463)
- Accent Color: 輔助藍色 (#97d3df)
- Neutral Colors: 灰色系 (#666, #999, #ccc)

字型系統

- 主要字型: Noto Sans (Google Fonts)
- 字重: 300, 400, 500, 600, 700
- 字級: 0.85rem 1.1rem

間距系統

- 基礎單位: 8px
- 常用間距: 10px, 16px, 24px, 32px, 40px

5.2 組件設計規範

節點設計

- 圓圈尺寸: 40x40px
- 狀態指示: 顏色 + 圖標組合
- 互動效果: 縮放 + 陰影

連接線設計

- 線條高度: 5px
- 漸層效果: 支援純色和漸層兩種模式
- 動畫效果: 進度條動畫

文字設計

- 日期格式: Title Case (Oct 13)
- 狀態文字: 0.9rem
- 時間文字: 0.85rem

5.3 互動設計規範

Hover 效果

- 節點圓圈: scale(1.10) + box-shadow
- 過渡時間: 0.3s ease
- 顏色變化: 完成狀態反轉效果

動畫效果

- Gradient Shine: 2 秒循環，白色漸層掃過
- 適用狀態: Processing, In Transit
- 動畫範圍: 僅限狀態文字區域

5.4 響應式設計規範

- 桌面版本: 最大寬度 1200px，標準間距
- 平板版本: 768px - 1024px，調整間距
- 手機版本: < 768px，緊湊佈局

5.5 無障礙設計規範

- 色彩對比: 確保文字與背景有足夠對比度
- 鍵盤導航: 支援 Tab 鍵導航
- 螢幕閱讀器: 適當的 ARIA 標籤
- 動畫控制: 支援使用者偏好設定

專案里程碑

Phase 1: 基礎架構

- [ ] 專案結構建立
- [ ] 基礎樣式系統
- [ ] 時間軸組件開發

Phase 2: 核心功能

- [ ] 三種時間軸類型實現
- [ ] 節點狀態系統
- [ ] 互動效果開發

Phase 3: 優化完善

- [ ] 動畫效果優化
- [ ] 響應式設計
- [ ] 效能優化

Phase 4: 部署上線

- [ ] 生產環境部署
- [ ] 效能監控設定
- [ ] 使用者測試反饋

文件版本: 1.0
最後更新: 2025 年 11 月
維護者: 萬能數維

三、工作時程及內容規劃

1. 需求分析與資料庫設計

時程：11/05(三) ～ 11/07(五)
工作內容：

1. 訪談
2. 需求文件整理
3. 技術可行性評估
4. 系統架構設計
5. 資料庫設計
6. 介面設計規範

2. 前端開發（Frontend）

時程：11/10(一) ～ 11/14(五)
工作內容：

1. HTML/CSS/JS 結構設計
2. 查詢輸入介面（單號 + 查詢碼）
3. 查詢結果卡片樣式（含 10 欄資料欄位）

4. RWD 響應式設計（桌機/手機）

3. 時間軸模組（Tracking Timeline UI）

時程：11/10(一) ～ 11/14(五)
工作內容：

1. 後端時間軸資料處理邏輯開發
2. Airtable 欄位映射與狀態判斷
3. 前端時間軸 UI 元件開發
4. 動態樣式切換邏輯
5. 響應式設計（桌機/手機）
6. 前後端整合測試
7. 效能優化與錯誤處理

4. 品牌設計整合（UI/UX + Theme）

時程：11/10(一) ～ 11/14(五)
工作內容：

1. 配色與字型符合客戶品牌規範
2. 按鈕、卡片、圖示設計統一
3. 依品牌風格調整前端樣式（亮度、邊角、陰影）

5. 後端與 API 串接（Backend / Integration）

時程：11/17(一) ～ 11/21(五)
工作內容：

1. Node.js 後端 API 設計（接收單號與查詢碼）
2. 連線 Airtable API（含驗證、錯誤處理）
3. 回傳資料 JSON 結構規範化
4. 基礎安全防護：API key 加密、限制查詢次數、CORS 設定

6. 部署與測試（Deployment & QA）

時程：11/24(一) ～ 11/28(五)
工作內容：

1. Netlify 前端部署與 Functions 後端連接測試
2. API 錯誤與異常查詢測試
3. RWD 測試（桌機/手機）
4. Airtable 權限與資料驗證測試
5. 程式碼部署自動化測試
6. 跨瀏覽器相容性測試
7. 負載測試

7. 文件與交付（Documentation & Handover）

時程：12/01(一) ～ 12/02(二)
工作內容：

1. 系統架構文件（API 結構、欄位對應表）
2. 前後端部署說明文件
3. 管理者維護指南
4. 部署指南
5. 用戶操作手冊

8. 網站主機申購

時程：12/01(一)
工作內容：

1. 子網域設定
2. DNS 指向
3. SSL 部署及測試

使用範圍與說明

本文件為 TailorMed Tracking System 之專案交付與技術需求存檔文件，
用於保留需求、設計、規格與時程之記錄。

如需調整系統行為或新增功能，
請另行提出需求並進行評估與報價。