# 技術架構文件 (Technical Architecture Document)

**專案名稱**：Bombus 企業管理系統

**版本**：V6.0 (PRD Alignment Ver.)

**文件日期**：2025-11-21

**狀態**：正式版 (Official)

**關聯文件**：PRD, FDD, IDP

**撰寫人**：Gemini (資深系統設計規劃師)

## 1. 功能模組架構圖 (Functional Module Architecture)

本架構圖嚴格依據《Bombus V6.0 PRD》定義之 L0.0 - L6.6 編號結構展開，作為開發範疇的最高指導藍圖。

mindmap  
 root((Bombus V6.0<br>企業管理系統))  
 L0 系統核心與儀表層  
 L0.0 企業管理儀表板  
 L0.1 系統管理與權限控制  
 L1 員工管理模組  
 L1.1 招募與候選人管理  
 L1.2 員工檔案與歷程管理  
 L1.3 人才庫與再接觸管理  
 L1.4 職涯晉升與接班規劃  
 L1.5 會議管理  
 L2 職能管理模組  
 L2.1 職等職級管理  
 L2.2 職務說明書管理 (JD)  
 L2.3 職能框架開發  
 L2.4 職能評估系統  
 L2.5 職能落差分析  
 L2.6 AI 職能生成引擎  
 L3 教育訓練管理模組  
 L3.1 培訓計畫管理  
 L3.2 課程與報名管理  
 L3.3 線上測驗系統  
 L3.4 培訓成效追蹤與回饋  
 L4 專案管理模組  
 L4.1 專案與任務管理  
 L4.2 專案協作與進度追蹤  
 L4.3 專案績效與毛利  
 L4.4 專案報表與分析  
 L4.5 專案自動化引擎  
 L5 績效管理模組  
 L5.1 過程管理系統與毛利計算  
 L5.2 考核週期管理  
 L5.3 360 度回饋系統  
 L5.4 績效紀錄與日誌系統  
 L5.5 績效分析與改善計畫  
 L6 文化管理模組  
 L6.1 企業文化手冊管理  
 L6.2 獎項資料庫管理  
 L6.3 文件儲存庫  
 L6.4 AI 申請助理  
 L6.5 智慧文件分析  
 L6.6 影響力評估引擎

**架構整合說明**：

這張架構圖（Functional Map）是系統開發的「地圖骨架」，它確保了技術實作與產品需求（PRD）的一一對應。我們從最底層的 **L0 系統核心** 構建基礎設施與全局儀表板，接著向上堆疊 **L1 至 L6** 的業務模組。

這種嚴格的編號結構（如 L1.4 對應週休三日、L4.3 對應未來損益）不僅是為了文件整齊，更是為了：

1. **開發追蹤**：Jira/Trello 的任務票（Ticket）可直接使用 L4.3 作為前綴，方便追蹤進度。
2. **模組化設計**：後端 API 的路由設計（Route Design）將直接參考此結構（如 /api/v6/l4/profit），確保程式碼結構清晰。
3. **依賴管理**：透過此圖可清晰識別模組邊界。例如 L2.5 (職能落差) 必然依賴於 L2.4 (職能評估) 的產出，而 L3.2 (課程管理) 則需要 L2.5 的輸入來推薦課程。

## 2. 系統技術架構圖 (System Technical Architecture)

本圖表定義系統的分層架構 (Layered Architecture)，展示從使用者端到基礎設施的技術堆疊與服務劃分。

### 2.1 架構設計說明

* **前端層 (Client Layer)**: 採用 Vue.js/React 框架，實作 Soft UI 與 i18n 多語系。
* **網關層 (Gateway Layer)**: Nginx/Kong 負責負載平衡、SSL 卸載與第一層 Rate Limiting。
* **應用服務層 (App Service)**: 採微服務或模組化單體 (Modular Monolith) 架構，Node.js 處理核心業務，Python 處理 AI 運算。
* **資料儲存層 (Data Layer)**: PostgreSQL 存儲關聯資料，MongoDB 存儲非結構化日誌，Redis 負責快取，NAS 負責文件冷儲存。

graph TD  
 subgraph "Client Layer (前端互動)"  
 Web[Web Browser<br>(React/Vue, Soft UI)]  
 Mobile[Mobile Web<br>(RWD Design)]  
 end  
  
 subgraph "Gateway Layer (安全與路由)"  
 LB[Load Balancer<br>(Nginx)]  
 Auth[Auth Gateway<br>(JWT, RBAC Check)]  
 WAF[Web App Firewall]  
 end  
  
 subgraph "Application Layer (核心業務)"  
 CoreService[Node.js Core Service<br>(NestJS/Go)]  
   
 subgraph "Module Logic (L0-L6)"  
 M\_L1[L1 員工管理]  
 M\_L2[L2 職能管理]  
 M\_L3[L3 教育訓練]  
 M\_L4[L4 專案管理]  
 M\_L5[L5 績效管理]  
 M\_L6[L6 文化管理]  
 end  
   
 AIService[Python AI Engine<br>(FastAPI + LLM SDK)]  
 subgraph "AI Agents"  
 AI\_Interview[L1.1 面試分析]  
 AI\_FuturePL[L4.3 未來損益]  
 AI\_TalentMap[L3.2 人才地圖]  
 AI\_JD[L2.6 職能生成]  
 end  
 end  
  
 subgraph "Data Storage Layer (資料持久化)"  
 PG[(PostgreSQL<br>核心業務數據)]  
 Mongo[(MongoDB<br>日誌/面試逐字稿)]  
 Redis[(Redis<br>Session/Cache)]  
 NAS[(NAS Storage<br>文件備份/歸檔)]  
 end  
  
 subgraph "External Services (第三方整合)"  
 API\_104[104 Job Bank API]  
 API\_GCal[Google Calendar API]  
 API\_LLM[LLM Model API]  
 end  
  
 %% Connections  
 Web --> LB  
 Mobile --> LB  
 LB --> WAF --> Auth --> CoreService  
   
 CoreService --> M\_L1 & M\_L2 & M\_L3 & M\_L4 & M\_L5 & M\_L6  
   
 M\_L1 --> AI\_Interview  
 M\_L4 --> AI\_FuturePL  
 M\_L3 --> AI\_TalentMap  
 M\_L2 --> AI\_JD  
   
 AI\_Interview & AI\_FuturePL & AI\_TalentMap & AI\_JD --> AIService  
   
 CoreService --> PG  
 CoreService --> Mongo  
 CoreService --> Redis  
 CoreService --> NAS  
   
 AIService --> API\_LLM  
 M\_L1 --> API\_104  
 M\_L1 --> API\_GCal

**技術整合說明**：

本系統採用了 **分層式微服務架構 (Layered Microservices Architecture)**，旨在平衡系統的穩定性、擴展性與智能化需求。核心業務邏輯（L0-L6 CRUD）選用 **Node.js (NestJS)** 或 **Go**，以確保高併發下的效能與回應速度；而針對 V6.0 重點強化的 AI 功能（如面試分析、損益預測），則獨立部署為 **Python (FastAPI)** 服務，以便靈活調用各類 LLM 模型與數據分析庫。

在資料儲存層，我們採用了 **Polyglot Persistence (多種資料儲存)** 策略：結構化且關聯性強的核心業務數據（如員工資料、財務紀錄）存放於 **PostgreSQL**，確保 ACID 事務一致性；非結構化數據（如面試逐字稿、系統日誌）則存放於 **MongoDB**，提供靈活的讀寫效能；文件與備份則依賴 **NAS** 進行冷儲存。此外，透過 **Gateway Layer** 的統一入口設計，我們能有效實施安全策略（如 WAF、RBAC），並在混合雲環境中靈活調度地端與雲端的資源，實現企業級的資安防護與營運彈性。

## 3. 系統資料流向架構圖 (High-Level Data Flow)

本圖表展示 Bombus V6.0 最核心的 **「人才價值鏈 (Talent Value Chain)」** 與 **「利潤計算鏈 (Profit Chain)」** 如何跨模組流動。

### 3.1 核心資料流說明

1. **人才流入**: L1.1 招募引入人才，建立 L1.2 員工檔案。
2. **能力轉化**: L2.5 職能落差分析觸發 L3 教育訓練，完成後更新 L3.2 人才地圖與 L2 職能狀態。
3. **價值創造**: 員工投入 L4.1 專案執行，產生工時與產出，AI 進行 L4.3 未來損益預測。
4. **績效變現**: L5.1 收集 L4 成本與財務營收，計算毛利與獎金。
5. **文化循環**: L6.1 EAP 支撐心理健康，L6.6 評估文化對留任率影響，回饋至 L0.0 儀表板。

graph LR  
 %% L1 & External  
 Job104(104 職缺) -->|匯入| L1\_Recruit[L1.1 招募與候選人管理]  
 L1\_Recruit -->|AI面試評分| L1\_Emp[L1.2 員工檔案與歷程]  
   
 %% L2  
 L1\_Emp -->|職位資訊| L2\_Comp[L2 職能管理]  
 L2\_Comp -->|L2.5 職能落差 Gap| L3\_Trigger{觸發培訓?}  
   
 %% L3  
 L3\_Trigger -->|Yes| L3\_Train[L3 教育訓練]  
 L3\_Train -->|訓後成效/轉化率| L2\_Update[更新職能等級]  
 L2\_Update -.-> L2\_Comp  
 L3\_Train -->|能力數據| TalentMap[L3.2 人才地圖]  
   
 %% L4  
 L1\_Emp -->|人力資源| L4\_Proj[L4 專案管理]  
 L4\_Proj -->|工時/任務| Cost\_Data[L4.1 隱形成本]  
 L4\_Proj -->|專案進度| Future\_PL[L4.3 未來損益預測]  
   
 %% L5  
 Cost\_Data --> L5\_Calc[L5.1 毛利計算引擎]  
 Revenue(財務營收) --> L5\_Calc  
 L5\_Calc -->|毛利與獎金池| L5\_Perf[L5.2 考核與獎金]  
 L3\_Train -->|ROI數據| L5\_Perf  
   
 %% L6  
 L1\_Emp -->|使用紀錄| L6\_EAP[L6.1 EAP/文化手冊]  
 L6\_EAP -->|留任率/滿意度| L6\_Impact[L6.6 影響力評估]  
   
 %% L0 Output  
 TalentMap --> Dashboard[L0.0 儀表板]  
 Future\_PL --> Dashboard  
 L5\_Perf --> Dashboard  
 L6\_Impact --> Dashboard  
   
 style Dashboard fill:#667eea,color:#fff,stroke:#333  
 style L5\_Calc fill:#f56565,color:#fff  
 style AI\_FuturePL fill:#9f7aea,color:#fff

**數據流向整合說明**：

這張圖表揭示了 Bombus V6.0 的核心價值邏輯：**「人才驅動績效，數據引導決策」**。資料流並非單向線性的，而是一個不斷優化迭代的閉環。

首先，**L1** 引入的人才數據是整個系統的起點，這些數據流入 **L2** 與 **L3**，透過「評估-培訓-再評估」的機制，動態更新員工的職能等級與人才地圖。具備更強職能的員工進入 **L4** 執行專案，此時系統會實時搜集工時（成本）與進度（產出），並透過 AI 進行未來損益預測。這些營運數據最終匯流至 **L5**，結合財務資料進行精確的毛利計算與獎金分配，實現「績效變現」。同時，**L6** 的文化數據（如 EAP 使用率）作為關鍵的環境變數，影響著員工的留任與滿意度。所有模組的關鍵產出最終匯聚於 **L0.0 儀表板**，為高階主管提供全景式的決策依據，讓每一次的管理動作都能基於數據，而非直覺。

## 4. 基礎設施與部署策略 (Infrastructure Strategy)

### 4.1 混合雲部署架構 (Hybrid Cloud Deployment) - 針對目標客戶需求

由於 PRD 2.1 提到需支援混合雲，以下為建議架構：

* **地端 (On-Premises / Private Cloud)**:
  + **部署內容**: Database (PostgreSQL - 薪資/個資 schema), NAS (機密文件), Auth Server (AD/LDAP).
  + **目的**: 確保敏感資料物理隔離，符合法規。
* **公有雲 (Public Cloud - AWS/GCP)**:
  + **部署內容**: Web Server, AI Engine (GPU Instances), Logging (MongoDB), Public API Gateway.
  + **目的**: 彈性擴充運算資源，處理 AI 模型推論與大量併發請求。
* **連接方式**: Site-to-Site VPN (IPsec) 或 AWS Direct Connect，確保內外網資料傳輸加密 (TLS 1.3)。

### 4.2 災難復原 (Disaster Recovery)

* **RPO (Recovery Point Objective)**: < 1 小時 (透過 WAL Log Archiving)。
* **RTO (Recovery Time Objective)**: < 4 小時。
* **備份策略**:
  + 資料庫：每 6 小時全量快照，每 15 分鐘增量備份。
  + NAS 文件：每日凌晨 03:00 同步至異地冷儲存 (S3 Glacier)。