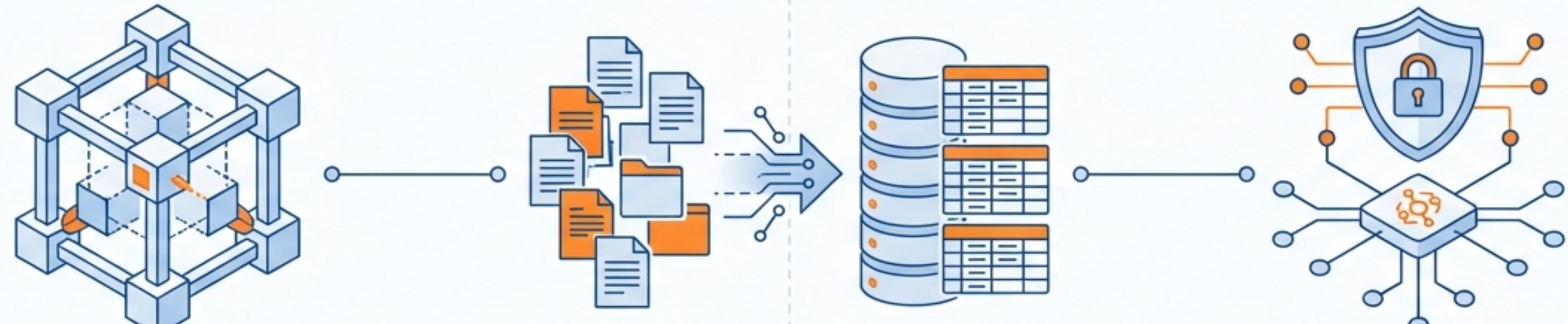


現代化 SaaS 架構轉型計畫：從 GLA Firebase 邁向 Keypiz L1 與 PostgreSQL

打造基於標準化託管、強型別資料與事件驅動身份驗證的堅實地基



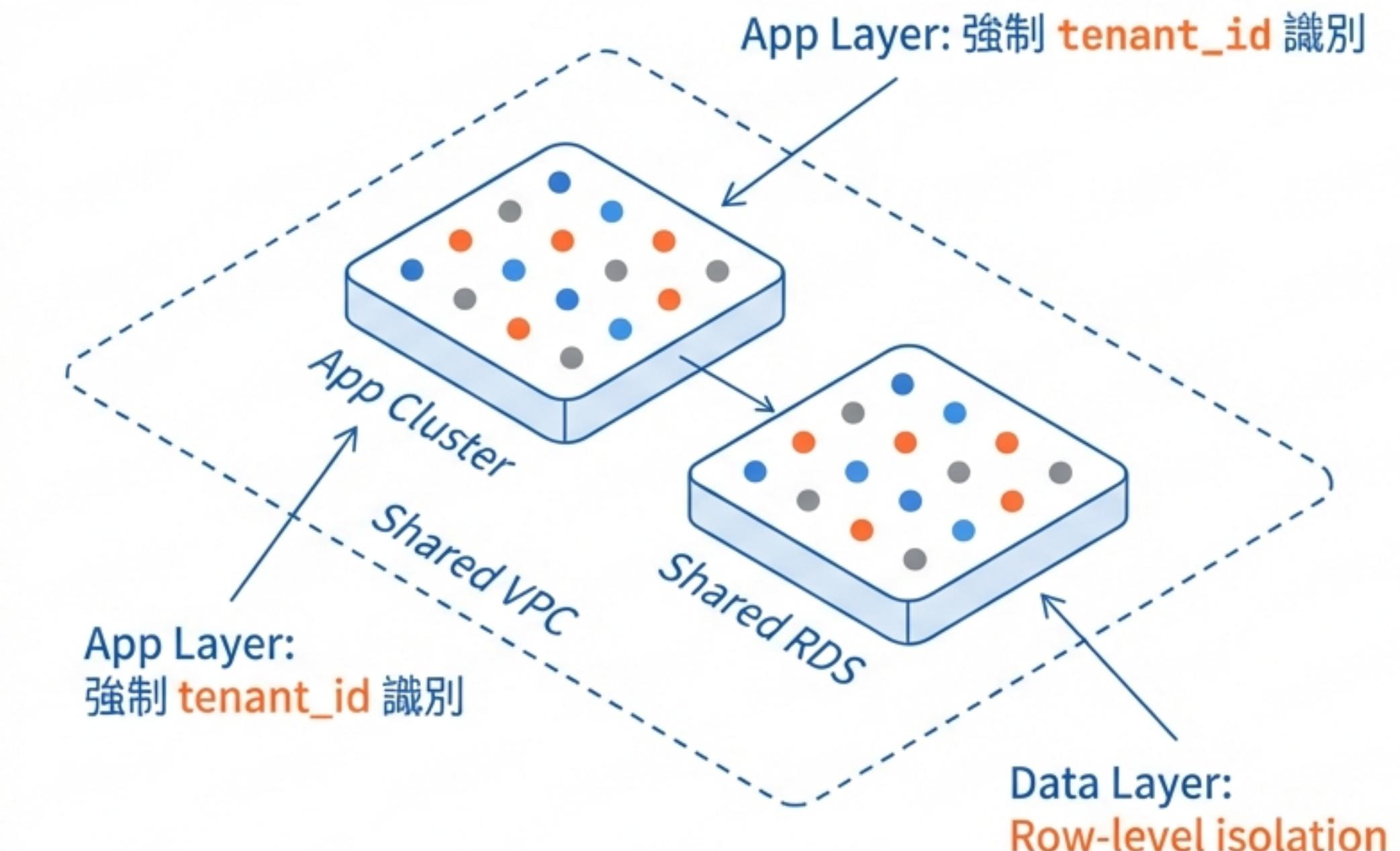
基礎設施
(Keypiz L1)

資料核心
(PostgreSQL)

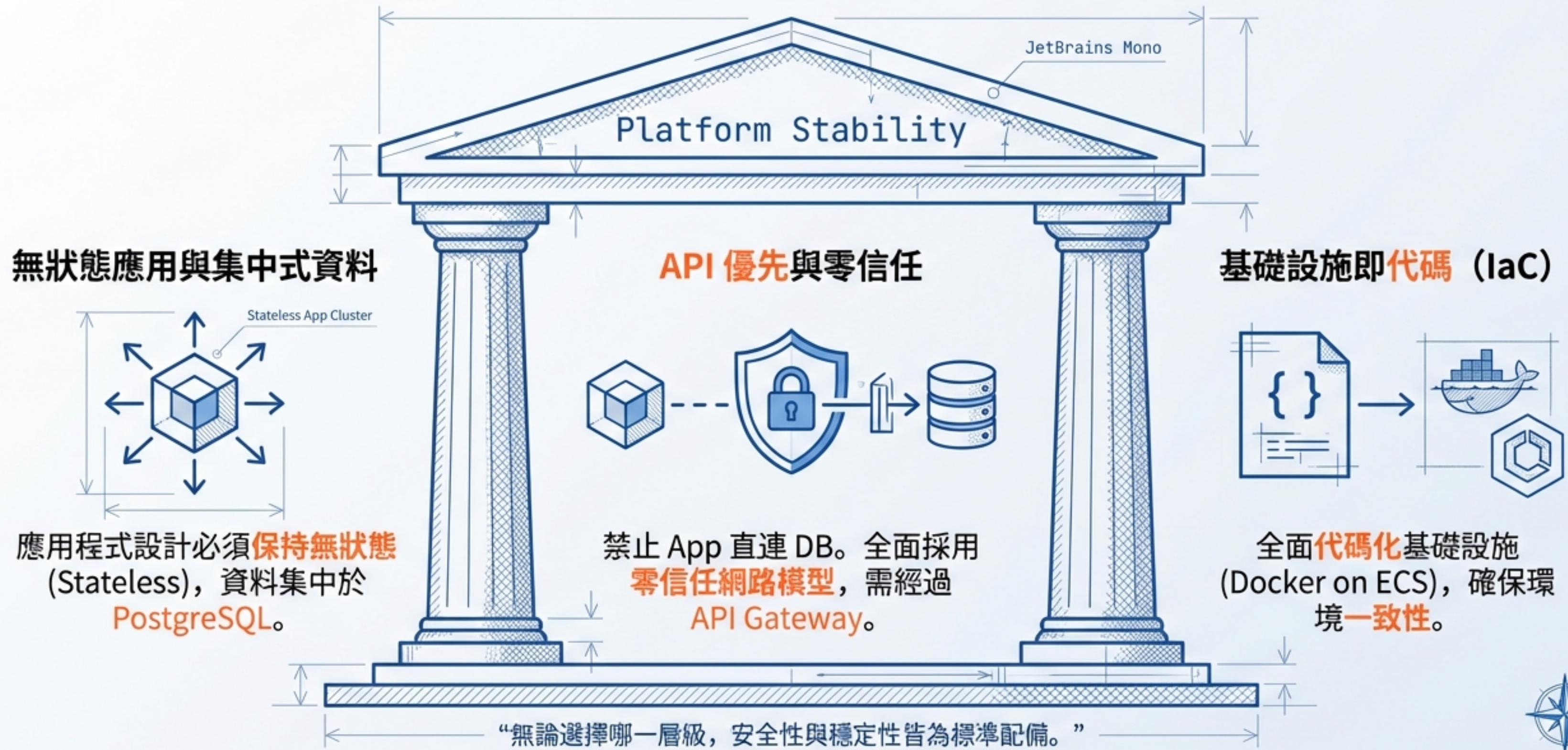
身份與整合
(Cognito + EventBridge)

目標架構定義：Keypiz Level 1 標準共享平台

Level 1 Shared SaaS
<ul style="list-style-type: none">• SLA: 99.5%• 隔離機制: 邏輯隔離 (Logical Isolation)• 資料層: Shared RDS (共用資料庫實例)• 應用層: 強制 tenant_id 識別與 Middleware 阻擋



通用工程設計原則：貫穿架構的安全與穩定基石



資料策略轉型：從 Firebase 到 Keypiz PostgreSQL

Source: GLA / Firebase (NoSQL)



Destination: Keypiz / PostgreSQL (Relational)

'tenant_id'	'user_id'	'data'	'created_at'
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27
1001	u-123	{json_data}	2023-10-27

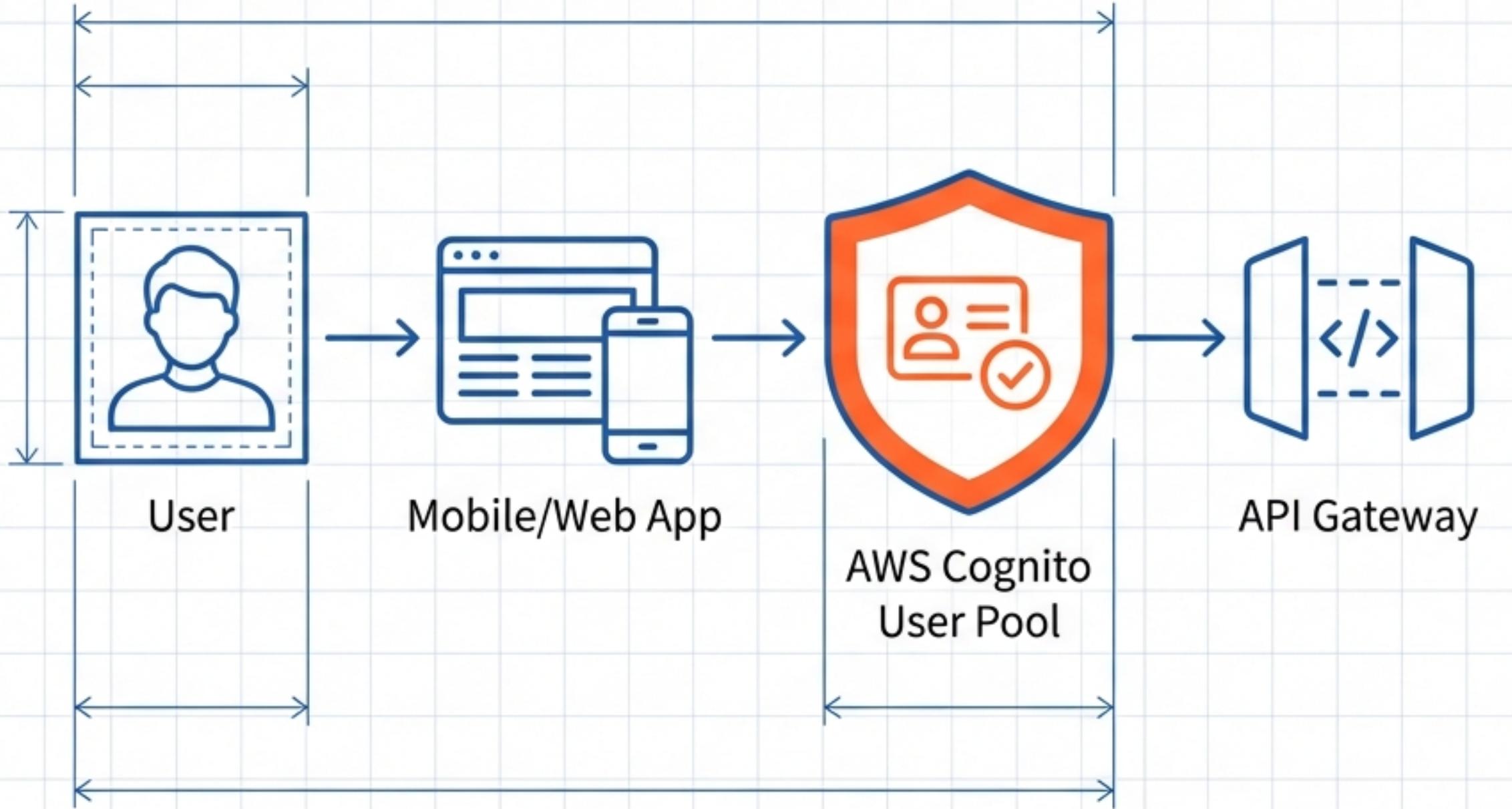
Shared RDS (RLS Enabled)

關鍵實作
(Implementation Key)
注入 **tenant_id**
這是 Shared RDS 模型下取代
Security Rules 的核心機制

非結構化文件。結構彈性但缺乏
強制關聯。

強型別關聯式資料庫。Row-Level
Security (RLS) 強制隔離。

身份驗證層：以 AWS Cognito 建立託管式安全基石



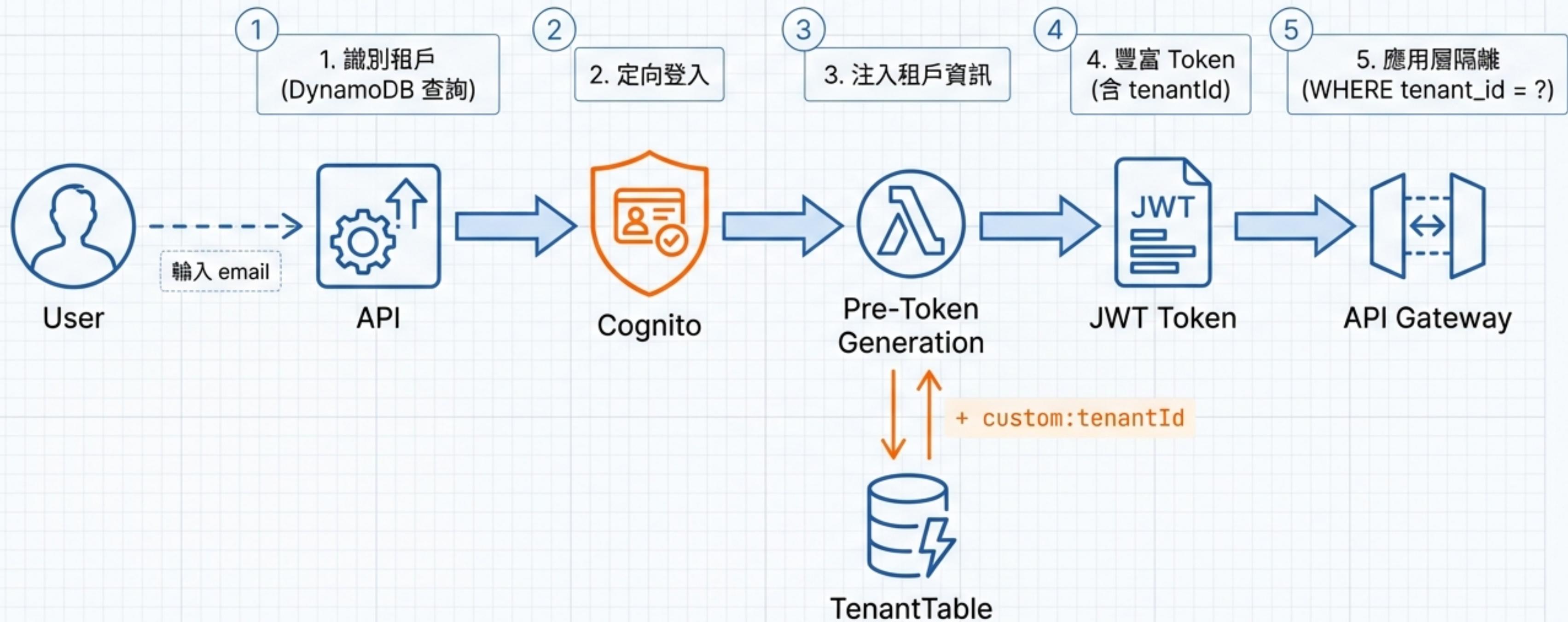
為何選擇 Cognito?

- **全託管服務**：處理註冊、登
登入、**MFA**、**Token** 管理
- **標準化 JWT**：簽發符合業界
標準的 JSON Web Token
- **安全屏障**：保護後端 API 與
Keypiz 資源
- **整合性**：與 API Gateway
原生整合

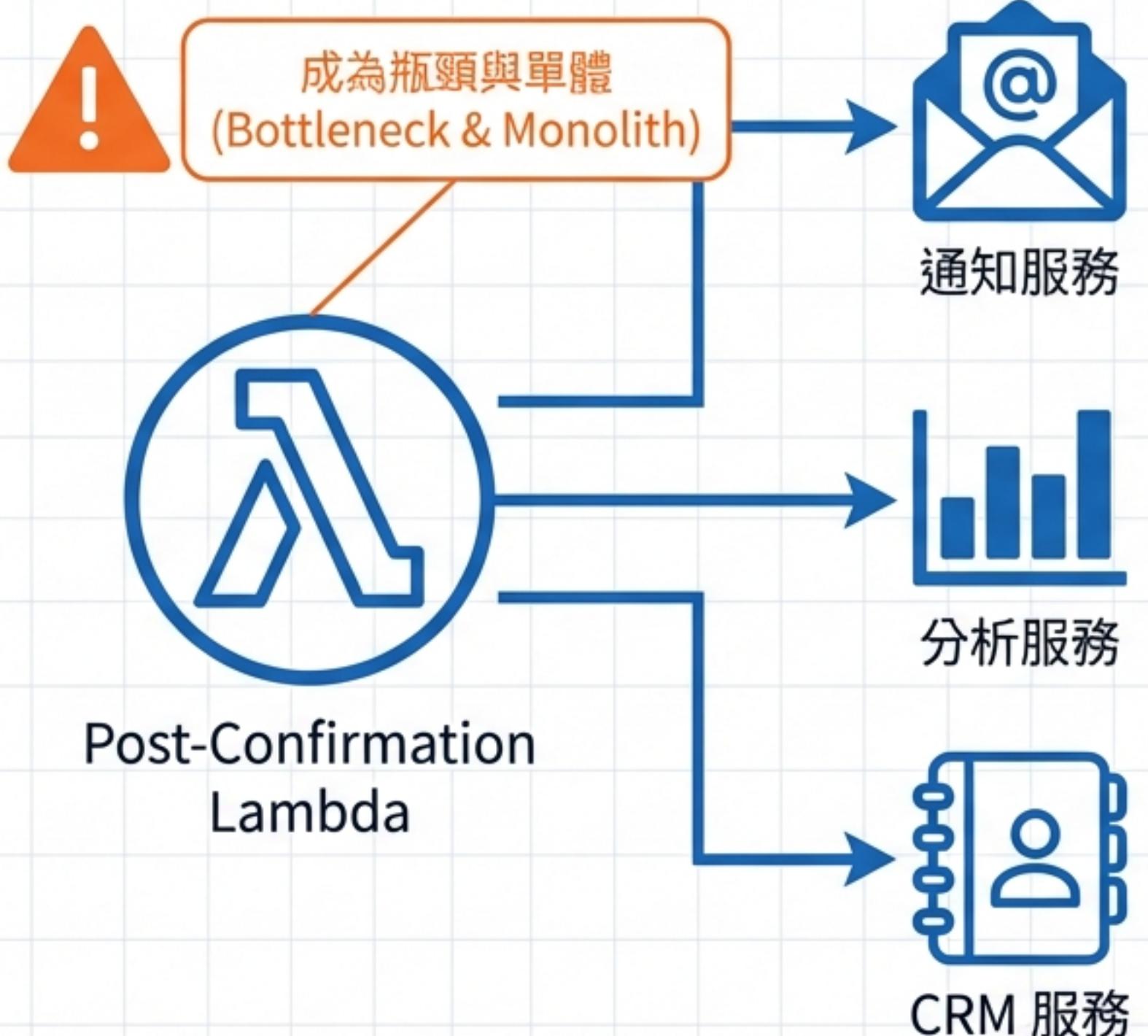
B2B 多租戶策略：單一 User Pool 與自訂屬性模型

模型 (Model)	優點 (Pros)	缺點 (Cons)
每租戶一個 User Pool	隔離度最高	管理成本高、難以維運
共用 User Pool + App Client 分租戶	共用使用者資料	App Client 變多、設定較複雜
共用 User Pool + 自訂屬性分租戶	擴展性與體驗的最佳平衡。 Sign-up / Sign-in URL 完全共用，新增租戶無需基礎設施變更。 ↳ Implementation Note: 使用 <code>custom:tenantId</code> 屬性標記組織	授權與資料隔離需在應用和 API 層嚴格實作

深度解析：身份驗證與租戶識別流程



超越點對點觸發：系統級解耦的需求

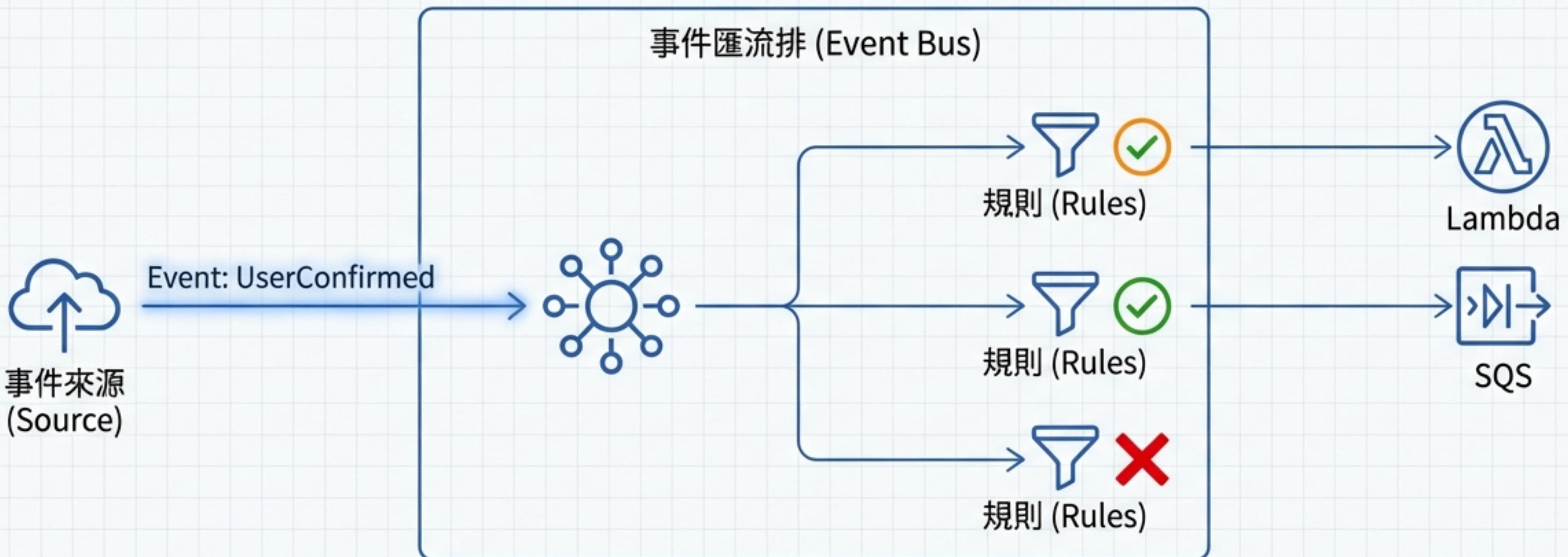


問題：若將所有邏輯（建立 Postgres 使用者、發送 Email、通知 CRM）都塞入單一 Lambda Trigger...

- **同步阻塞**: 拖慢使用者註冊體驗
- **脆弱性**: 任何一個下游服務失敗都會導致流程中斷

解決方案：我們需要一個更強大的機制來廣播事件——EventBridge。

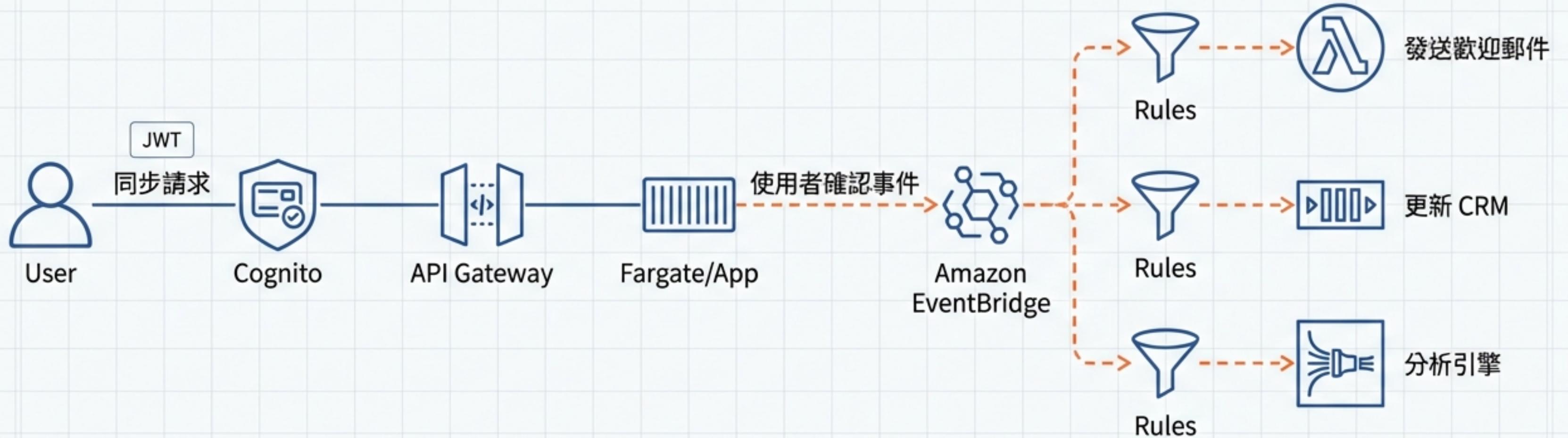
架構神經中樞：Amazon EventBridge



核心概念

- **解耦:** 身份層只負責發布事件，不需知道誰在監聽。
- **路由:** 使用 JSON 規則過濾並分發事件到正確的目標。

完整架構全景：從身份驗證到事件驅動



流程：使用者登入 (JWT) -> 應用層產生事件 -> EventBridge 接收 -> 平行觸發下游服務 (Email, CRM, Analytics)。

現實考量：大規模下的成本與效能管理



本地驗證 JWT (Local Validation)

在 [API Gateway](#) 或後端本地驗證 Token 簽名，而非每次呼叫 [Cognito API](#)。大幅降低延遲與成本。



批次處理 (Batching)

利用 [SQS](#) 緩衝 [EventBridge](#) 的下游寫入操作，減少資料庫連線壓力並優化吞吐量。



L1 資源監控

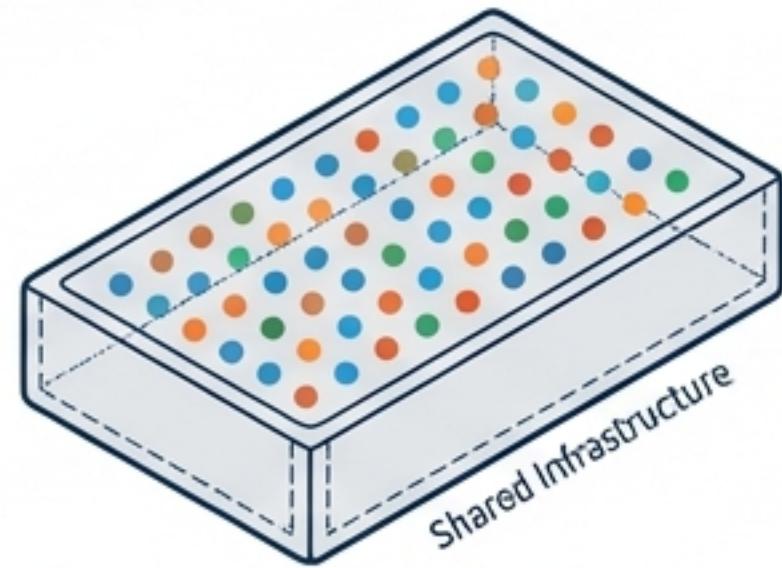
密切監控『吵鬧鄰居』([Noisy Neighbor](#)) 效應，確保 [Shared RDS](#) 在多租戶環境下的效能穩定。

未來展望：從 L1 彈性擴展至 L2/L3



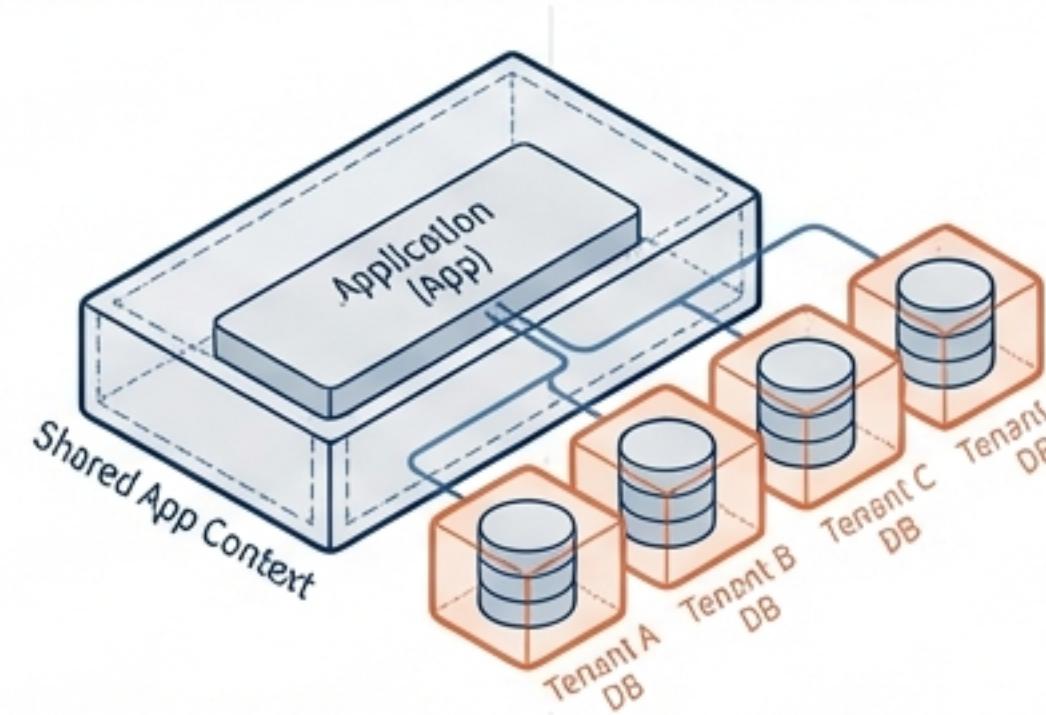
Phase 1 (Left): Start Fast (L1)

成本最低，共用基礎設施。適合快速遷移與 MVP。



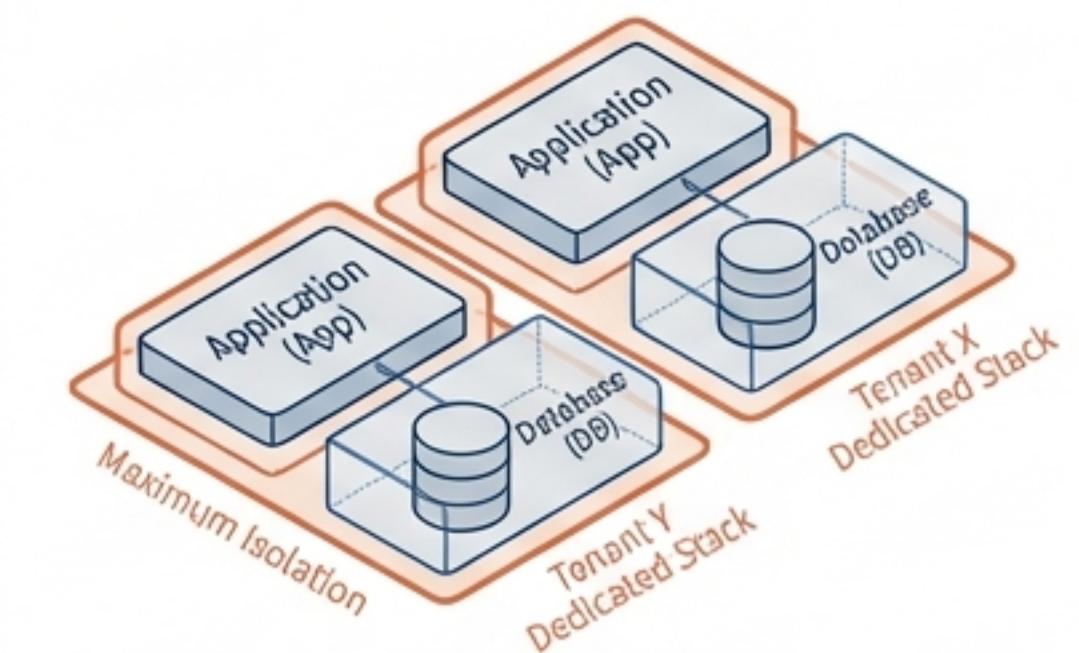
Phase 2 (Center): Scale Securely (L2)

資料實體隔離。適合高合規需求客戶。



Phase 3 (Right): Future (L3)

全環境獨立。針對銀行/政府級專案。



Technical Promise: 單一程式碼庫 (Single Codebase)。升級無需重寫應用邏輯。

執行總結：構建現代化 SaaS 的三大支柱

1



從安全的身份層開始

以 AWS Cognito 取代自建系統，實作『**單一 User Pool + Tenant ID**』的最佳實踐。

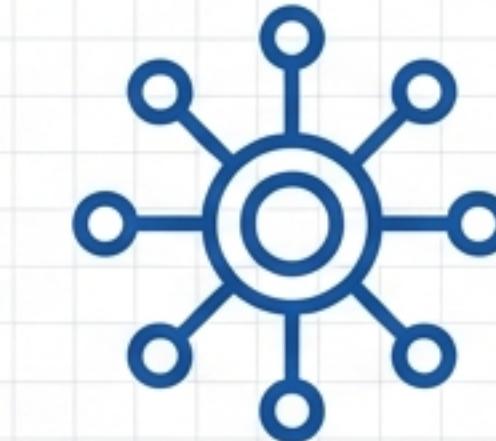
2



標準化 L1 資料架構

完成從 Firebase 到 PostgreSQL 的遷移，利用 Keypiz L1 規範確保多租戶資料的嚴格邏輯隔離。

3



用事件匯流排解耦一切

引入 Amazon **EventBridge** 作為系統神經中樞，確保身份事件與後端業務邏輯的鬆散耦合。

Outcome: 一個安全、可擴展且符合 Keypiz 標準的企業級 SaaS 平台。