

Docker Hermes Container 系統架構圖解

v5

NORTHWND MCP Server 運行架構 × LINE Webhook API 架構 — 以 Microsoft Azure 架構圖風格繪製與說明

nousresearch/hermes-agent:latest · Hermes Agent v0.21.5 · 模型 glm-5.3 (ollama-cloud)

盤點時間 2026-10-03 (依 docker-compose.yml 、 config.yaml 、 LINE adapter 原始碼 、 gateway.log 與 state.db 實際檢視)



閱讀指引：兩條架構路徑，依 Azure Architecture Center 結構組織

先看端到端全貌，再分別拆解 LINE Webhook 與 NORTHWND MCP，最後以實測資料檢視整合與改善

1

Architecture 端到端架構

全貌圖、編號工作流程與元件、部署與網路路徑

第 3-5 頁

2

LINE Webhook API

入口與驗證管線、訊息轉換與 session、回覆策略、互動時序

第 6-9 頁

3

NORTHWND MCP Server

運行架構、設定與 MCP 全景、工具規格、資料模型、呼叫時序

第 10-14 頁

4

Integration 整合實測

LINE → Agent → NORTHWND 的實際執行路徑與回應時間

第 15-16 頁

5

Considerations · Next steps

安全與信任邊界、限制與改善路線、維運速查、重點回顧

第 17-20 頁



v5 範圍與資料來源

- LINE：容器內 plugins/platforms/line/adapters.py 原始碼與 gateway.log
- 隧道：主機 ngrok.exe (本機 API :4040 確認轉送 localhost:8646)
- MCP：/opt/data/config.yaml、mcp/northwind/server.py、註冊日誌
- 實測：state.db 中 LINE session 的訊息與工具呼叫紀錄
- 遮蔽：ngrok 子網域、LINE userId 與所有密碼皆不列出

圖例

LINE 平台 / 使用者

容器 · Gateway

資料庫 / 儲存

邊界 / 網路區域

資料流步驟

隧道 / 風險

MCP · 工具

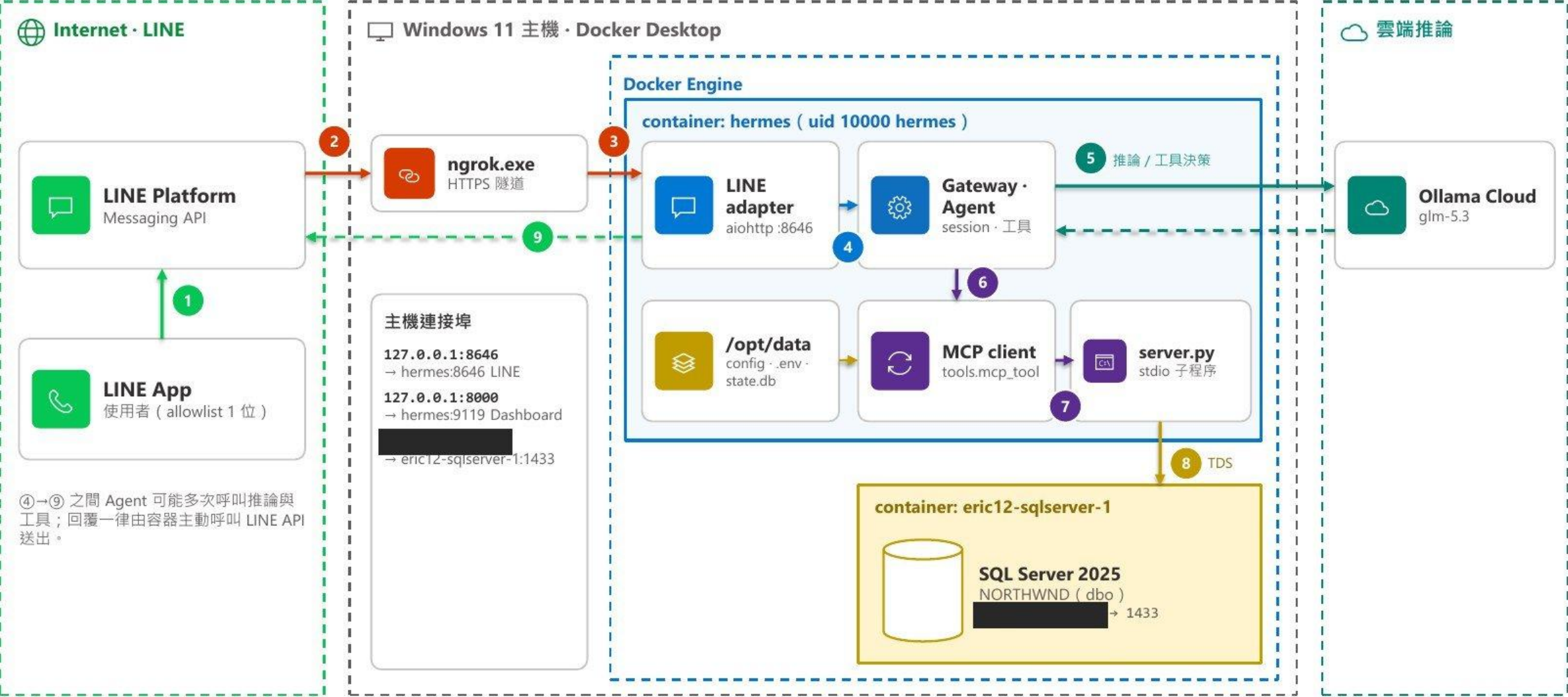
雲端推論

容器

回應 / 非同步

端到端架構：LINE → 隧道 → 容器 → MCP → NORTHWND

LINE 訊息經 ngrok 進入 hermes 容器，Agent 透過 stdio MCP 查詢 NORTHWND；編號對應下一頁



工作流程與元件：九個編號步驟與各元件職責

左：資料流（對應上一頁編號）；右：元件的實作方式與職責

- 1
- 使用者在 LINE 傳送訊息給官方帳號
- 2
- LINE Platform 以 HTTPS POST 呼叫 webhook URL (帶 X-Line-Signature)
- 3
- ngrok 轉送到 localhost:8646 · Docker Desktop 轉進容器
- 4
- LINE adapter 驗簽、去重、白名單後轉成 MessageEvent 交給 Gateway
- 5
- Agent 把對話與工具清單送給 glm-5.3，取得回覆或工具呼叫
- 6
- 需要資料時，Agent 呼叫 mcp_northwind_* 工具
- 7
- MCP client 經 stdin/stdout (JSON-RPC) 呼叫 server.py
- 8
- server.py 以 pymssql 參數化 SELECT 查詢 NORTHWND
- 9
- LINE adapter 以 Reply API (失效改 Push) 送回答案

元件	實作	職責
LINE Platform	Messaging API (雲端)	收發使用者訊息、呼叫 webhook
ngrok.exe	主機程序 · 免費網域	公開 HTTPS → localhost:8646
LINE adapter	aiohttp · plugins/platforms/line	驗簽、白名單、轉換、回覆
Gateway · Agent	hermes gateway run	session 管理、推論、工具派送
Ollama Cloud	provider ollama-cloud	glm-5.3 推論
MCP client	tools.mcp_tool (mcp 2.0)	spawn 子程序、tools/list、tools/call
northwind server.py	mcp 2.3.0 MCPServer · pymssql	4 個唯讀查詢工具
eric12-sqlserver-1	SQL Server 2025	NORTHWND 資料庫
/opt/data volume	hermes-data	config.yaml、.env、state.db、venv

部署與網路：兩個本機入口、兩個 Docker 網路、一條主機轉發

hermes 未加入 eric12_eric 網路，因此資料庫連線走 host.docker.internal 經主機連接埠

```
docker-compose.yml ( 節錄 )

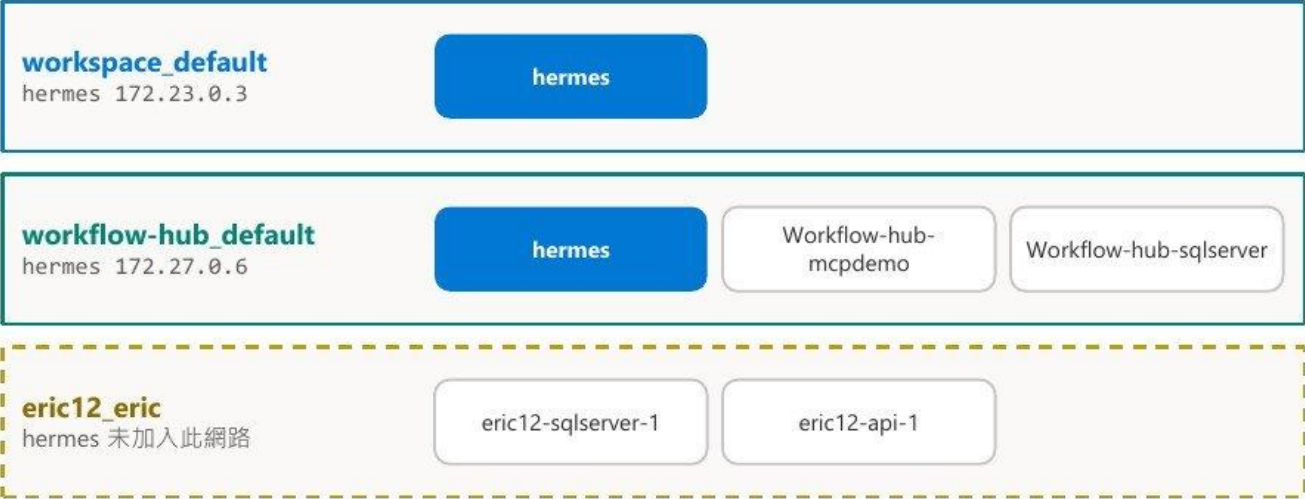
services:
  hermes:
    image: nousresearch/hermes-agent:latest
    command: gateway run
    ports:
      - "127.0.0.1:8000:9119"
      - "127.0.0.1:8646:8646" # LINE
    volumes:
      - hermes-data:/opt/data
    networks:
      - default # workspace_default
      - workflow-hub # mcpdemo:5301
```

 外網入口只有 LINE

8646 與 8000 皆綁 127.0.0.1；LINE 由 ngrok 從主機內部轉入，Dashboard 只供本機。

 跨 compose 存取資料庫

hermes 不加入 eric12_eric，改走 [redacted] compose。

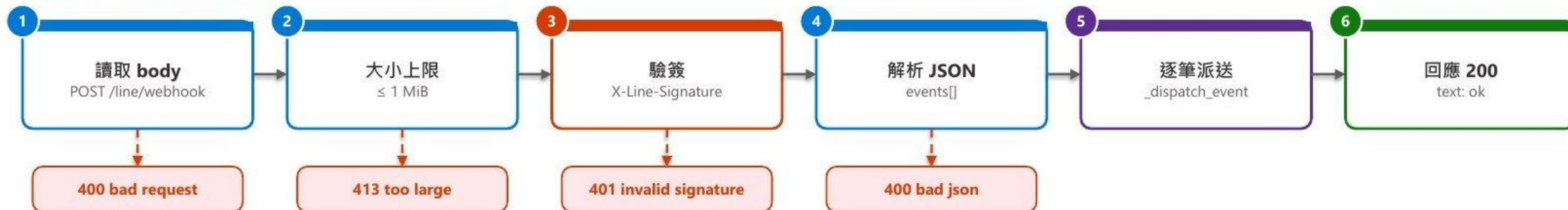


方向	路徑	協定 / 驗證
入站	LINE → <子網域>.ngrok-free.app → ngrok → localhost:8646 → 容器 :8646	HTTPS→HTTP · HMAC 驗簽
入站	瀏覽器 → 127.0.0.1:8000 → 容器 :9119 Dashboard	HTTP · Basic Auth
出站	LINE adapter → api.line.me (reply / push / loading)	HTTPS · Bearer token
出站	Agent → Ollama Cloud (glm-5.3)	HTTPS · API key
出站	server.py → [redacted] → eric12-sqlserver-1:1433	[redacted]

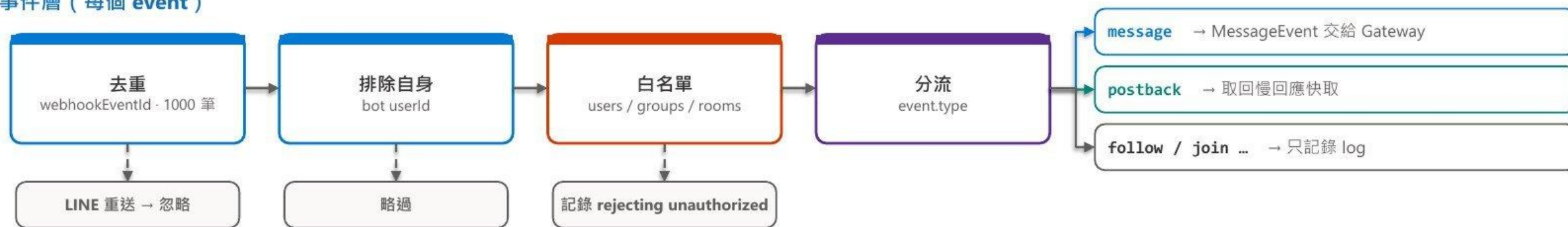
LINE Webhook ①：入口路由與請求驗證管線

aiohttp 綁定所有介面 (IPv4+IPv6) :8646；每個請求先過四道檢查，再逐筆派送事件

請求層 (`_handle_webhook`)



事件層 (每個 event)



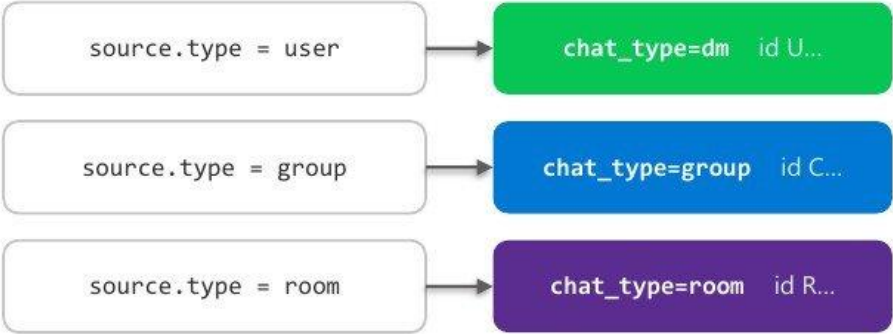
路由	方法	用途
/line/webhook	POST	LINE Platform 事件入口 (Developers Console 設定的 Webhook URL)
/line/webhook/health	GET	隧道 / 反向代理健康探測 · 回 {"status":"ok","platform":"line"}
/line/media/{token}/{name}	GET	外送圖片 / 影音的暫存 URL (30 分鐘)；需 LINE_PUBLIC_URL，目前未設定

LINE Webhook ②：訊息轉換、對話識別與 session

不同訊息型別轉成統一的 MessageEvent；chat_id 決定 session，DM 另會觸發 loading 動畫

LINE message.type	轉換結果	說明
text	原文字	直接作為使用者輸入
image / video / file	下載到本地路徑 + "[image]" 等	由 api-data.line.me 取內容，供視覺模型使用
audio	VOICE (語音轉文字路徑)	LINE 錄音視為語音訊息
sticker	"[sticker: 關鍵字]"	以貼圖 keywords 描述
location	"[location: 標題 地址]"	以文字描述位置
其他	"[unsupported message type]"	不中斷對話

對話識別 (_resolve_chat)



Gateway session key

agent:main:line:dm: [redacted]



同一使用者共用同一 session

今天 13 則 LINE 訊息都落在 session 20261003_115454 (11:54 建立)；對話歷史寫入 /opt/data/state.db，跨訊息保留脈絡。



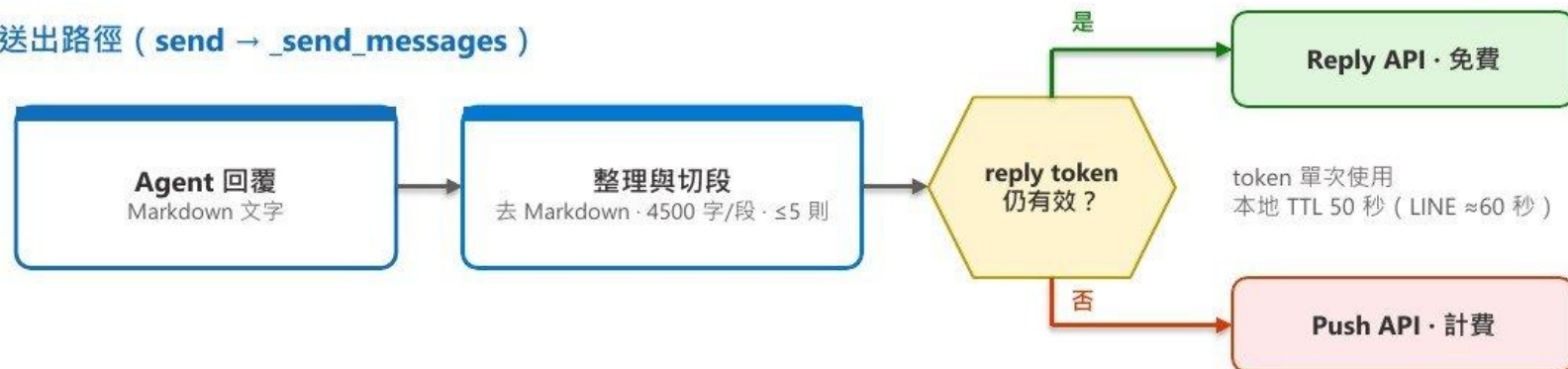
loading 動畫

DM 收到訊息即呼叫 chat/loading/start (5-60 秒，以 5 秒為單位)，讓使用者知道正在處理。

LINE Webhook ③：回覆策略與慢回應按鈕

免費 Reply 優先、計費 Push 備援；回覆一律由容器主動呼叫 api.line.me，不經過隧道（依 adapter.py 原始碼）

送出路徑（send → `_send_messages`）



訊息上限

每則氣泡硬上限 5000 字，adapter 以 4500 字切段；
每次呼叫最多 5 個訊息物件，超過的尾段以「...」截斷。

慢回應機制（`slow_response_threshold = 45 秒`）



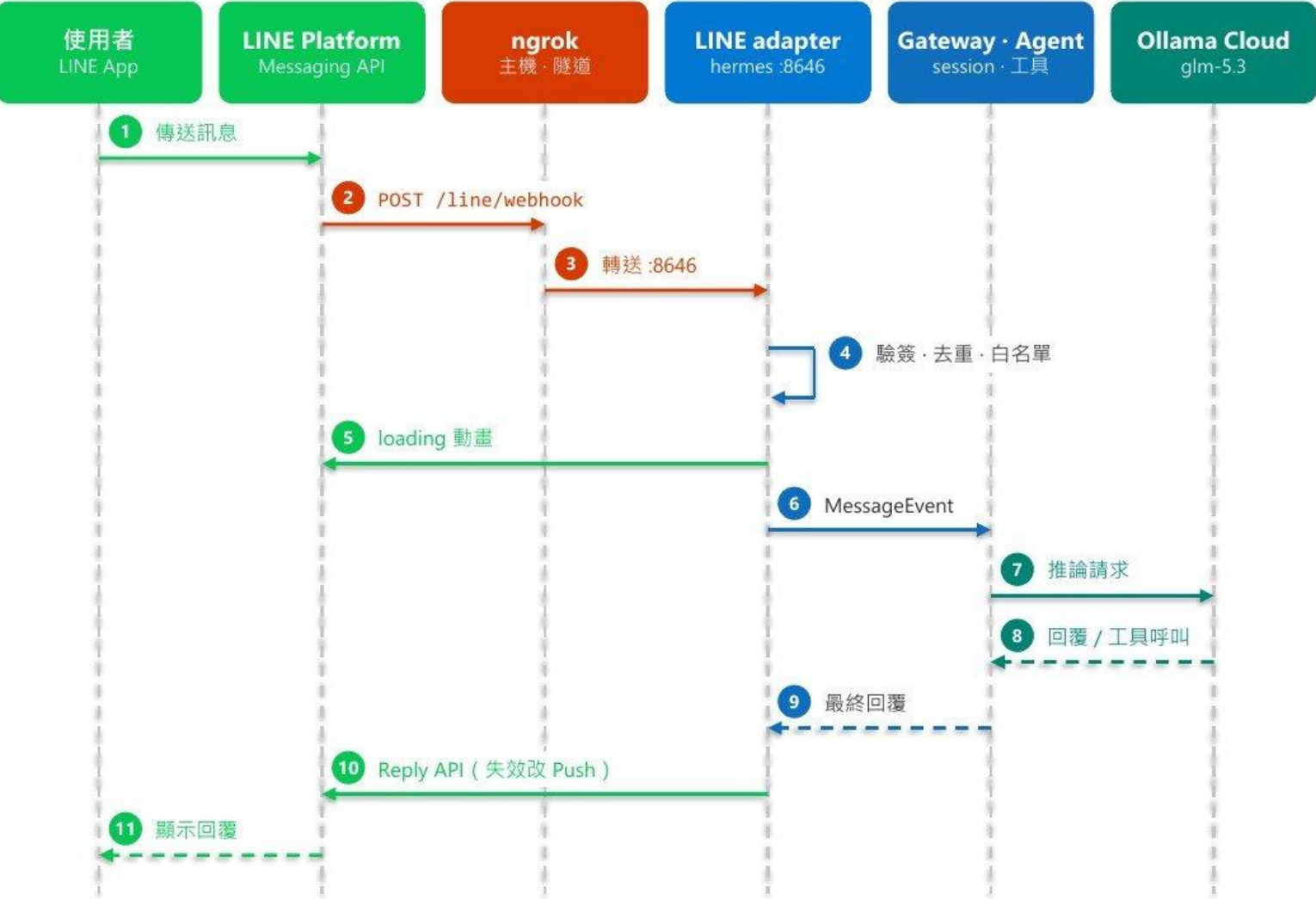
觀察：今天 13 則回覆中有 5 則超過 45 秒，但 gateway.log 沒有 postback 事件紀錄；本頁按鈕流程依原始碼描述，實際是否觸發需另行驗證。

時間軸

- 1 收到訊息，暫存 reply token
- 2 45 秒仍未完成：用 token 送 Template Buttons「Get answer」
- 3 答案完成：存入快取（READY）
- 4 使用者點按：postback 帶來新的免費 reply token
- 5 以新 token 回覆答案（DELIVERED）

LINE Webhook ④：一則訊息的互動時序

UML 時序圖；入站只有 webhook 一條路，回覆由容器主動呼叫 LINE Messaging API



入口

LINE Developers Console 的 Webhook URL 指向 ngrok 公開網域；ngrok 轉送 localhost:8646。



驗證

base64(HMAC-SHA256(Channel Secret, body)) 常數時間比對；來源須在 LINE_ALLOWED_USERS 內。



200 時機

事件派送後回 200 ok；Agent 推論在背景進行，不阻塞 LINE 的 webhook 請求。



回覆

優先使用免費、單次的 reply token；逾時改用計費的 Push API。



實測

今天 13 則訊息回應 3.8–197.3 秒；工具呼叫越多越慢（詳見第 16 頁）。

NORTHWND MCP ①：運行架構 — Gateway 啟動的 stdio 子程序

不開連接埠；Gateway 依 config.yaml 啟動 server.py，以 stdin / stdout 傳 JSON-RPC



NORTHWND MCP ②：設定方式與 MCP Server 全景

config.yaml 共 13 個 mcp_servers (12 個啟用) ; northwind 是唯一在容器內執行的 stdio Server

```
/opt/data/config.yaml ( 節錄 )

mcp_servers:
  northwind:
    command: /opt/data/mcp-servers/
      northwind/.venv/bin/python
    args: ["/opt/data/mcp-servers/
      northwind/server.py"]
    env:
      NORTHWIND_DB_HOST: host.docker.internal
      NORTHWIND_DB_PORT:
      NORTHWIND_DB_NAME: NORTHWND
      NORTHWIND_DB_USER:
      timeout: 60 # 每次 tools/call
      connect_timeout: 30
      enabled: true
```

傳給子程序。



為何選 stdio

需求是在 hermes 容器內執行；不開連接埠，由 Gateway 依設定自動啟動、重啟與關閉。

stdio · 容器內子程序

1 個

northwind

HTTP · Docker 網路 · API key

1 個

workflow-hub

HTTPS · OAuth

7 個

canva

figma

gamma

hugging_face

postman

stripe

supabase

HTTPS · 無驗證

3 個

kiwi

microsoft-learn

trivago

已停用

1 個

context7

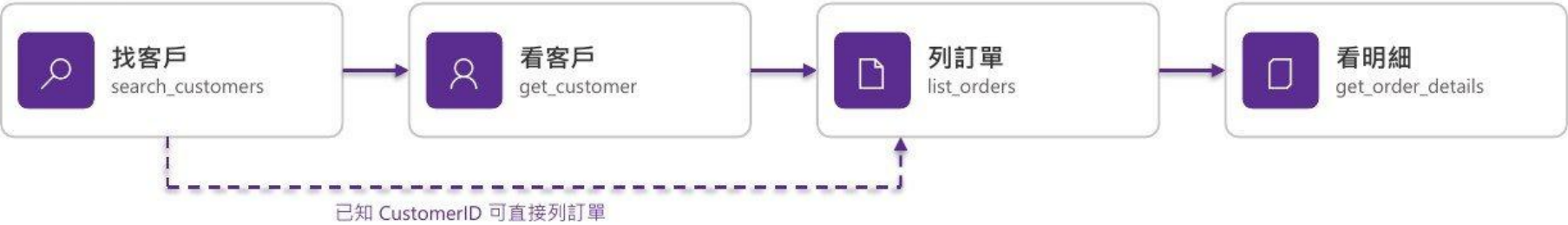
外部 MCP 皆為遠端 URL；只有 northwind 在 hermes 容器內以子程序執行，並直接連線內部資料庫。

NORTHWND MCP ③：工具規格 — 4 個業務工具 + 4 個通用工具

Gateway 日誌顯示註冊 8 個工具：server.py 定義 4 個，Hermes 另為每個 MCP Server 加上 resources / prompts 通用工具

工具 (mcp_northwind_...)	參數	回傳
search_customers	keyword、country、city、limit=20	count、customers[]
get_customer	customer_id	客戶欄位 + OrderCount、TotalSales、首末訂單日
list_orders	customer_id、date_from、date_to、ship_country、limit=20	count、orders[] (含 LineCount、Subtotal)
get_order_details	order_id	表頭 + Lines[] + Subtotal、GrandTotal

典型查詢鏈



防護	做法
SQL 注入	所有值以 pymysql %s 參數帶入；TOP (n) 先轉整數並限制 1–100
萬用字元	模糊查詢跳脫 [% _
錯誤訊息	預期錯誤丟 ToolError，模型才看得到原因

Hermes 通用工具

list_resources

read_resource

list_prompts

get_prompt

全部唯讀

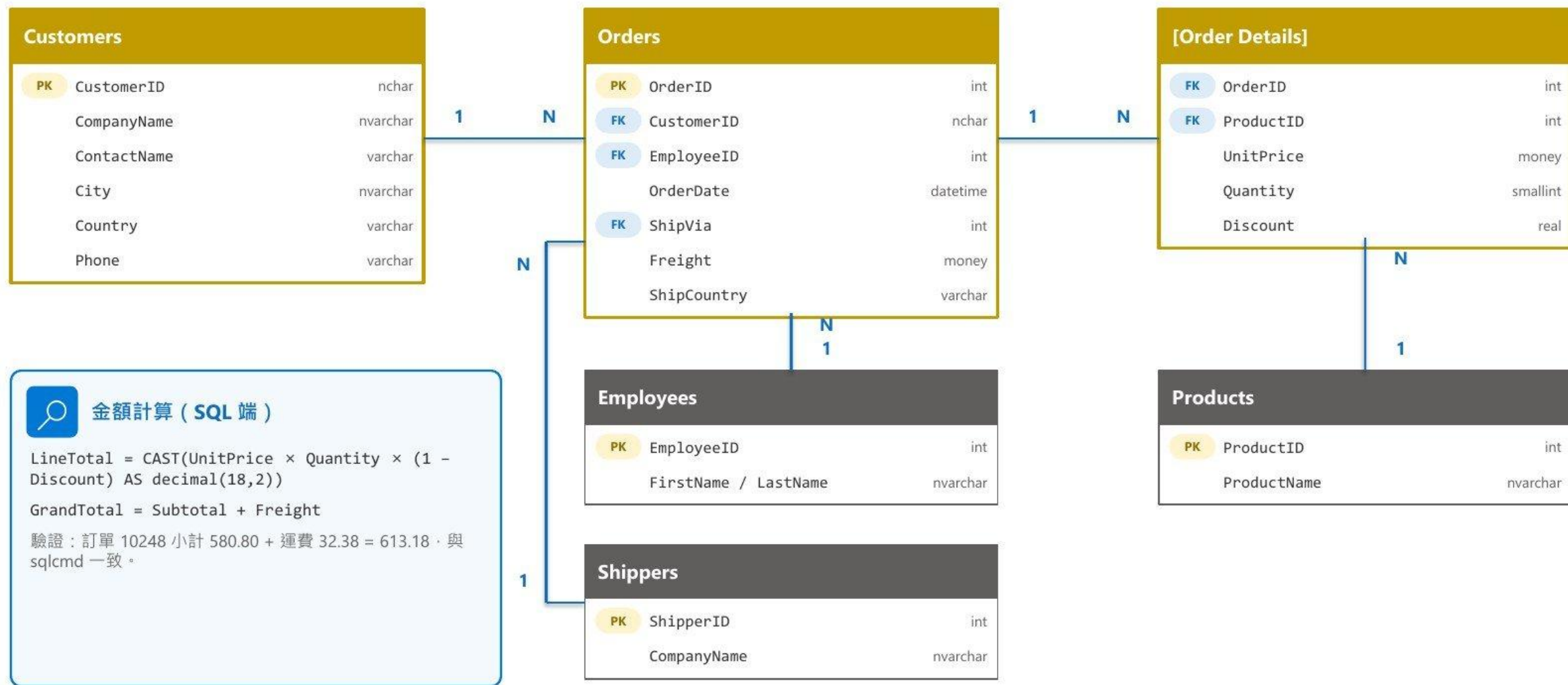
readOnlyHint=True；只有 SELECT。

結果上限

limit 上限 100，避免結果過大。

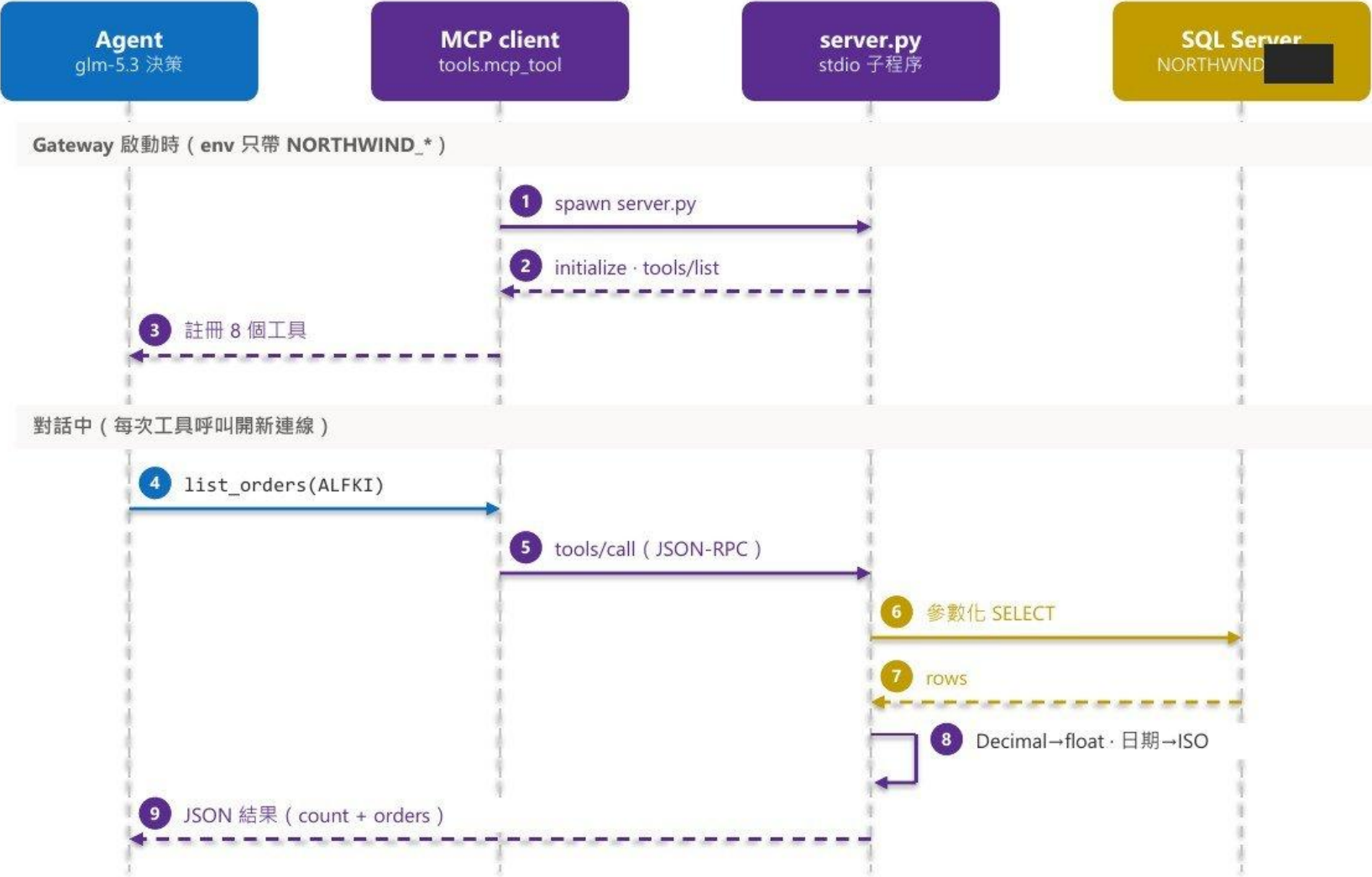
NORTHWND MCP ④：查詢涉及的資料模型 (dbo schema)

工具一律寫明 dbo.，避免誤用 khh / Owasp / tpi / SOUTHWND 等 schema 的同名資料表



NORTHWND MCP ⑤：從啟動到一次查詢的呼叫時序

Gateway 啟動時 spawn 並列出工具；對話中每次 tools/call 都開一條新的資料庫連線



啟動驗證

hermes mcp test northwind : Connected (1181 ms)
Tools discovered: 4 .



逾時

tools/call 60 秒、連線 30 秒；DB 登入 10 秒、查詢 30 秒。



錯誤回報

ToolError 訊息交給模型；DB 錯誤包成 "Database error: ...".



重新註冊

今天 gateway / agent 日誌共 23 筆 northwind 註冊紀錄
· 首筆 13:22:56 .

整合實測：LINE 查詢 NORTHWND 的實際執行路徑

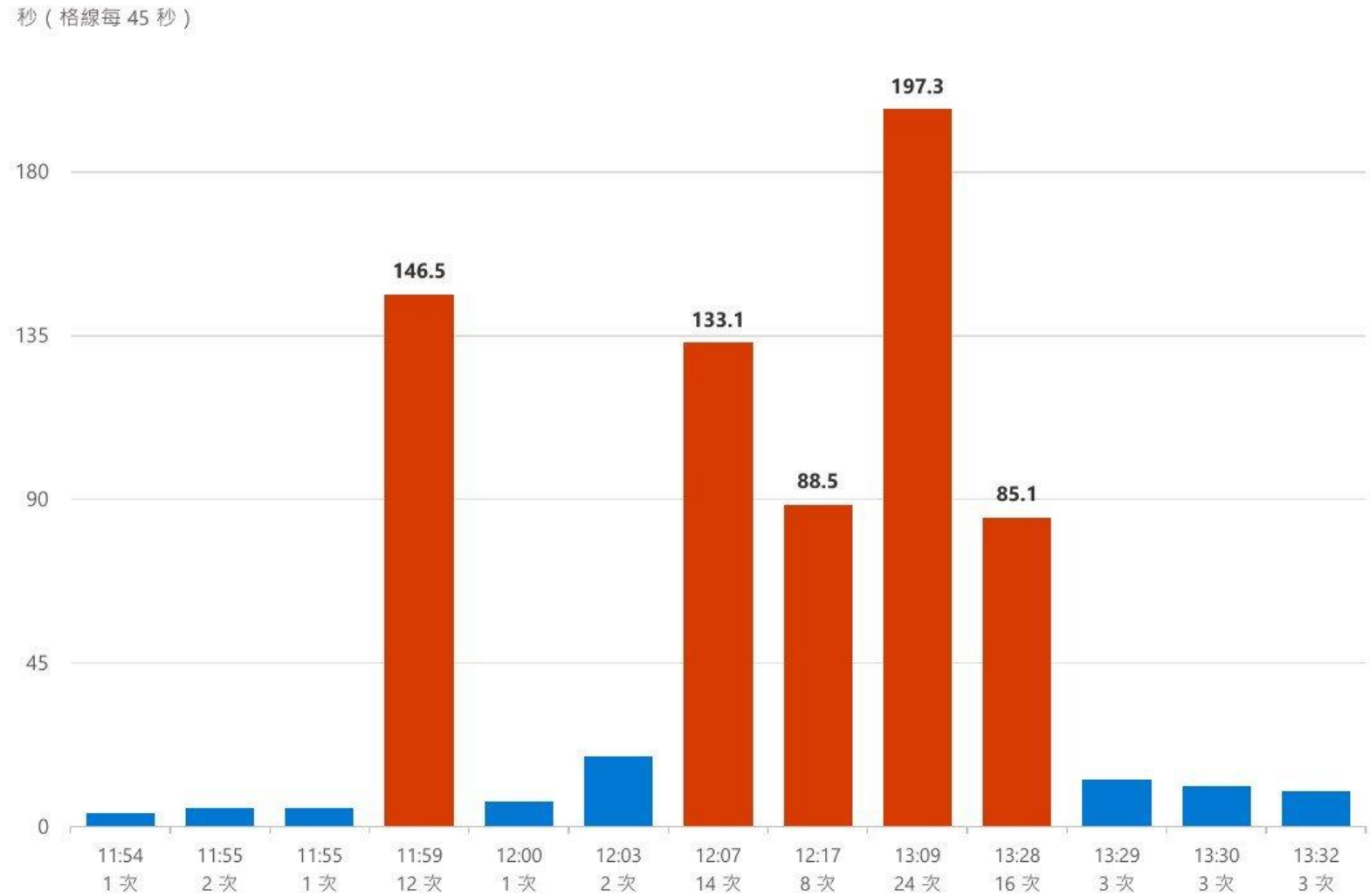
依 state.db 中 LINE session 的工具呼叫紀錄；設計路徑與實測路徑不同



實測：13:22 註冊後的 4 則查詢，state.db 中沒有任何 mcp_northwind_* 呼叫；13:29 起每則都是 write_file → terminal 兩步。
推測原因：此 LINE session 於 11:54 建立，早於 13:22 的工具註冊，模型的工具清單可能未更新。

回應時間：今天 13 則 LINE 訊息，5 則超過 45 秒門檻

依 gateway.log 的 response ready 紀錄；橫軸為回覆完成時間與 api_calls 次數，橘色為超過慢回應門檻



5 / 13

超過 45 秒

3 次

最近 3 則 NORTHWND 查詢的 api_calls

≤ 45 秒

> 45 秒 (慢回應門檻)

🕒

時間跟著 api_calls 走

平均每次 api_call 約 4-12 秒；超過 45 秒的 5 則都在 8 次以上。

🔍

首次查詢最慢

13:06 與 13:27 的 ANTON 都在摸索資料位置；腳本寫好後同類查詢降到 10-13 秒。

✅

改善方向

讓模型直接用 MCP 工具，可再省去 write_file 與 terminal 兩步。

安全與信任邊界：從公開網址到資料庫的每一道關卡

綠色為已落實的控制，橘色為需要注意的風險



考量與後續步驟：已知限制與改善路線

每項標註對應的 Azure Well-Architected 支柱

#	限制 / 問題	建議	支柱	優先順序
1	LINE session 未使用已註冊的 MCP 工具，改自寫腳本	LINE 送 /new 重測；必要時以 SOUL.md 指示優先用 mcp_northwind_*	效能	立即 #1 #2
2			安全	
3	ngrok 免費網域重啟即變，需手動改 Webhook URL	改用固定網域（ngrok 保留網域或 Cloudflare Tunnel）	可靠	
4	eric12-sqlserver-1 停止時查詢回傳 Database error	在工具說明與回覆中提示；監控 eric12 容器健康	可靠	本週 #3 #4 #7
5	host.docker.internal 僅 Docker Desktop 支援	移到 Linux 時加 extra_hosts: host-gateway	營運	
6	未設定 LINE_PUBLIC_URL，無法外送圖片 / 影音	設為 ngrok 公開網址即可啟用 /line/media	營運	規劃 #5 #6 #8
7	5 / 13 則回覆超過 45 秒，可能觸發 Push 計費	減少工具輪次；確認慢回應按鈕實際運作	成本	
8	資料品質：45 筆客戶欄位為？，含 2026 測試訂單	資料庫端改 nvarchar 並重寫資料	營運	

維運速查

PowerShell / Git Bash 皆可；操作 Hermes CLI 一律加 -u hermes

用途	指令
測試 MCP 連線	<code>docker exec -u hermes -e HERMES_HOME=/opt/data hermes /opt/hermes/.venv/bin/hermes mcp test northwind</code>
MCP 註冊 / 錯誤 log	<code>docker exec hermes sh -c 'grep -h northwind /opt/data/logs/gateway.log /opt/data/logs/mcp-stderr.log tail -20'</code>
LINE 進出訊息	<code>docker exec hermes sh -c 'grep platform=line /opt/data/logs/gateway.log tail -20'</code>
LINE 健康探測	<code>curl http://127.0.0.1:8646/line/webhook/health</code>
ngrok 目前網址	Invoke-RestMethod http://127.0.0.1:4040/api/tunnels
更新 server.py	<code>docker cp mcp/northwind/server.py hermes:/opt/data/mcp-servers/northwind/server.py</code>
修正擁有者	<code>docker exec hermes chown hermes:hermes /opt/data/mcp-servers/northwind/server.py</code>
套用設定變更	<code>docker restart hermes (.env / config.yaml 變更都需要)</code>



ngrok 網址變更時

到 LINE Developers Console 更新 Webhook URL 並按 Verify。



更換資料庫密碼



暫停 MCP

config.yaml 將 northwind.enabled 改 false 後重啟。

重點回顧

NORTHWND MCP × LINE Webhook 架構一頁帶走

1

條入站路徑

LINE → ngrok →
127.0.0.1:8646 → LINE
adapter

4

道請求檢查

大小、HMAC 驗簽、JSON
、白名單 (另有去重)

1

個 **stdio MCP**

northwind 是 13 個 MCP
設定中唯一的容器內子程序

8

個註冊工具

4 個唯讀業務工具 + 4 個
Hermes 通用工具

5/13

則超過 45 秒

回應時間隨 api_calls 增加
；最近查詢約 10–13 秒

架構重點：LINE 只靠一條 **HTTPS webhook** 進來，回覆由容器主動呼叫 **LINE API**；**NORTHWND** 經 **Gateway** 管理的 **stdio** 子程序存取，密碼不進設定檔。

優先處理：讓 **LINE session** 直接使用 **MCP 工具** (**/new** 重測)、改用唯讀資料庫帳號、改用固定隧道網域。

資料來源：docker-compose.yml、config.yaml、plugins/platforms/line/adapter.py、mcp/northwind/server.py、gateway.log、state.db、ngrok 本機 API；盤點時間 2026-10-03。