單元 17：系統部署與遠端存取配置

前面兩個單元介紹了兩套自動化系統：NotebookLM 自動化 Web 系統和簡報編修系統。這些系統在預設情況下都只能在安裝的那台電腦上使用。

但在很多情況下，你可能需要讓其他人也能使用這些系統：

- 團隊成員需要在自己的電腦上使用
- 你想在公司的不同辦公區域都能存取
- 你有多台電腦，想從任何一台都能使用

這個單元將介紹如何部署這些系統，讓它們可以在區域網路內分享，甚至透過網際網路遠端存取。

部署的基本概念

在開始之前，讓我們先理解幾個基本概念。

本機 vs. 網路存取

本機存取： 系統只監聽 `localhost`（127.0.0.1），只有執行系統的那台電腦可以連線。這是預設的方式，最安全但也最受限。

區域網路存取： 系統監聽 `0.0.0.0`，同一個區域網路內的其他電腦可以透過 IP 位址連線。例如，如果你的電腦 IP 是 192.168.1.100，其他人可以用 http://192.168.1.100:5000 連線。

網際網路存取： 透過各種技術（如 port forwarding、tunneling、雲端部署），讓在任何地方都能透過網際網路連線。

伺服器 vs. 客戶端

在這個架構中：

- **伺服器**：執行系統的電腦。它需要保持開機和運行，提供服務。
- **客戶端**：使用系統的電腦。只需要有瀏覽器就能使用。

當你把系統部署在一台電腦上，其他電腦都是「客戶端」——它們透過瀏覽器存取「伺服器」上的系統。

認證問題

這是一個需要特別注意的問題。

NotebookLM 自動化系統需要登入 Google 帳號。這個登入狀態是儲存在**伺服器電腦**上的。當其他人透過網路使用這個系統時，他們實際上是使用**伺服器上登入的 Google 帳號**。

這意味著：

- 你需要決定使用誰的 Google 帳號
- 所有使用者的操作都會影響同一個 NotebookLM 帳號
- 需要考慮是否信任所有使用者

對於簡報編修系統，情況比較簡單——它使用的是 Gemini API Key，不涉及個人帳號，只要 API Key 有效且有足夠的配額即可。

部署方式一：區域網路分享

這是最簡單的分享方式，適合同一個辦公室或家庭網路內的使用。

步驟一：修改監聽位址

預設情況下，Flask 應用程式只監聽 localhost。要讓區域網路內的其他電腦能連線，需要修改監聽位址。

NotebookLM 自動化系統：

編輯 `app.py`，找到最後的啟動部分，修改為：

```
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

簡報編修系統：

後端 `app.py` 可能已經設定為監聽 `0.0.0.0`，如果沒有，同樣修改：

```
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8099, debug=True)
```

步驟二：查詢伺服器的 IP 位址

在伺服器電腦上，開啟命令提示字元，輸入：

```
ipconfig
```

找到「IPv4 位址」，通常是類似 `192.168.x.x` 或 `10.x.x.x` 的格式。記下這個位址。

步驟三：設定 Windows 防火牆

Windows 的防火牆預設會阻擋外部連入的請求。你需要開放對應的 port。

方法一：透過 Windows 防火牆介面

1. 開啟「Windows Defender 防火牆」
2. 點擊「進階設定」
3. 選擇「輸入規則」
4. 點擊「新增規則」
5. 選擇「連接埠」
6. 選擇「TCP」，輸入特定的 port（如 5000 或 8099）
7. 選擇「允許連線」
8. 選擇適用的網路（網域、私人、公用）
9. 命名規則（如「Flask Server」）

方法二：使用命令列

```
netsh advfirewall firewall add rule name="Flask 5000" dir=in acti
netsh advfirewall firewall add rule name="Flask 8099" dir=in acti
```

步驟四：啟動系統

在伺服器電腦上啟動系統。

NotebookLM 自動化系統：

```
python app.py
```

簡報編修系統：

```
# 使用 start.bat 或手動啟動
start.bat
```

步驟五：從其他電腦連線

在區域網路內的其他電腦上，開啟瀏覽器，輸入：

```
http://伺服器IP:port
```

例如：

- NotebookLM 自動化系統：`http://192.168.1.100:5000`
- 簡報編修系統：`http://192.168.1.100:5173`

如果一切正常，你應該能看到系統的介面。

部署方式二：開機自動啟動

如果這個系統需要長期運行，你會希望它在電腦開機時自動啟動，而不是每次都要手動執行。

Windows：使用啟動資料夾

最簡單的方式是把啟動腳本放入 Windows 的啟動資料夾。

步驟一：建立啟動腳本

為 NotebookLM 自動化系統建立 `start_notebooklm.bat`：

```
@echo off
cd /d D:\NotebookLM課程\讓NotebookLM自動化
python app.py
pause
```

為簡報編修系統，直接使用現有的 `start.bat`。

步驟二：加入啟動資料夾

1. 按 `Win + R`
2. 輸入 `shell:startup`，按 Enter
3. 這會開啟啟動資料夾
4. 把你的 `.bat` 檔案（或它的捷徑）複製到這個資料夾

現在，每次 Windows 開機時，這些系統就會自動啟動。

Windows：使用 Windows 服務

如果你希望系統在背景執行（不顯示命令提示字元視窗），可以把它設定為 Windows 服務。這需要一些額外的工具，如 NSSM（Non-Sucking Service Manager）。

步驟一：下載 NSSM

前往 https://nssm.cc/download 下載 NSSM。

步驟二：安裝服務

```
nssm install NotebookLMService "D:\NotebookLM課程\讓NotebookLM自動
```

步驟三：啟動服務

```
nssm start NotebookLMService
```

這樣系統就會在背景執行，即使沒有使用者登入也會運行。

Linux/macOS：使用 systemd 或 launchd

如果你的伺服器是 Linux 或 macOS，可以使用系統原生的服務管理工具。

Linux (systemd)：

建立服務檔案 `/etc/systemd/system/notebooklm.service`：

```
[Unit]
Description=NotebookLM Automation Service
After=network.target

[Service]
Type=simple
User=your_username
WorkingDirectory=/path/to/notebooklm-automation
ExecStart=/usr/bin/python3 app.py
Restart=always

[Install]
WantedBy=multi-user.target
```

啟用並啟動：

```
sudo systemctl enable notebooklm
sudo systemctl start notebooklm
```

macOS (launchd)：

建立 plist 檔案並放入 `~/Library/LaunchAgents/`。

部署方式三：正式生產部署

前面介紹的方式都是使用 Flask 的開發伺服器。這個伺服器方便開發，但不適合正式的生產環境——它無法處理高併發，也沒有良好的錯誤處理。

如果你需要更穩定、更高效能的部署，應該使用專門的 WSGI 伺服器。

Windows：使用 Waitress

Waitress 是一個純 Python 的 WSGI 伺服器，特別適合 Windows 環境。

步驟一：安裝 Waitress

```
pip install waitress
```

步驟二：建立啟動腳本

建立 `run_server.py`：

```
from waitress import serve
from app import app

if __name__ == '__main__':
    print("伺服器啟動於 http://0.0.0.0:5000")
    serve(app, host='0.0.0.0', port=5000)
```

步驟三：執行

```
python run_server.py
```

Waitress 比 Flask 的開發伺服器更穩定，適合長期運行。

Linux/macOS：使用 Gunicorn

Gunicorn 是 Linux/macOS 上最常用的 Python WSGI 伺服器。

步驟一：安裝 Gunicorn

```
pip install gunicorn
```

步驟二：執行

```
gunicorn -w 4 -b 0.0.0.0:5000 app:app
```

參數說明：

- `-w 4`：使用 4 個 worker 進程（建議設定為 CPU 核心數的 2 倍）

- `-b 0.0.0.0:5000`：監聽所有網路介面的 5000 port
- `app:app`：應用程式的模組和變數名稱

使用 Docker

Docker 提供了一個隔離、可重複的部署環境。

步驟一：建立 Dockerfile

```
FROM python:3.11-slim

WORKDIR /app

# 安裝依賴
RUN pip install notebooklm-py flask
RUN pip install "notebooklm-py[browser]"
RUN playwright install chromium
RUN playwright install-deps

# 複製專案
COPY . .

# 啟動
EXPOSE 5000
CMD ["python", "app.py"]
```

步驟二：建立並執行 Docker 容器

```
docker build -t notebooklm-gui .
docker run -p 5000:5000 -v ~/.notebooklm:/root/.notebooklm notebo
```

注意 `-v` 參數：這是用來掛載登入狀態的目錄，讓容器可以存取已登入的 Google 帳號狀態。

部署方式四：網際網路存取

如果你需要從任何地方透過網際網路存取系統，有幾種方式可以實現。

使用 ngrok

ngrok 是一個簡單的隧道服務，可以把本機的服務暴露到網際網路上。

步驟一：下載並安裝 ngrok

前往 https://ngrok.com/ 下載，並註冊一個免費帳號。

步驟二：設定認證

```
ngrok config add-authtoken YOUR_AUTH_TOKEN
```

步驟三：啟動隧道

```
ngrok http 5000
```

ngrok 會提供一個公開的 URL（如 `https://xxxx.ngrok.io`），任何人都可以透過這個 URL 存取你的系統。

注意：免費版的 ngrok 每次啟動會產生不同的 URL，而且有流量限制。如果需要固定 URL，需要付費版。

使用 Cloudflare Tunnel

Cloudflare Tunnel 是另一個免費的隧道服務，比 ngrok 更穩定，而且可以使用自己的域名。

步驟一：註冊 Cloudflare 帳號並設定域名

步驟二：安裝 cloudflared

```
# Windows
winget install cloudflare.cloudflared

# macOS
brew install cloudflared
```

步驟三：登入並建立隧道

```
cloudflared tunnel login
cloudflared tunnel create my-notebooklm-tunnel
```

步驟四：設定隧道指向本機服務

```
cloudflared tunnel route dns my-notebooklm-tunnel notebooklm.your
cloudflared tunnel --url http://localhost:5000 run my-notebooklm-
```

現在你可以透過 `https://notebooklm.yourdomain.com` 存取系統。

使用 VPN

如果你不想把系統暴露在公開網際網路上，但又想從外部存取，可以使用 VPN。

當你連接到 VPN 後，就像是在區域網路內一樣，可以直接用內部 IP 存取系統。

常見的 VPN 解決方案：

- **Tailscale**：設定最簡單，免費版足夠個人使用
- **WireGuard**：效能最好，但需要一些設定
- **OpenVPN**：最成熟，但設定較複雜

安全考量

當你把系統暴露到網際網路時，安全性變得非常重要。

使用 HTTPS： 確保連線是加密的。ngrok 和 Cloudflare Tunnel 都會自動提供 HTTPS。

設定密碼保護： 可以在系統前面加上基本的密碼驗證，防止未授權的存取。

限制 IP： 如果知道會從哪些 IP 存取，可以設定防火牆只允許這些 IP。

監控存取日誌： 定期檢查存取日誌，看是否有異常的存取嘗試。

及時更新： 保持系統和所有依賴的更新，修補已知的安全漏洞。

多人使用的注意事項

當多人同時使用系統時，需要注意一些事項。

NotebookLM 帳號問題

如前所述，NotebookLM 自動化系統使用的是伺服器上登入的 Google 帳號。這意味著：

1. **所有人操作同一個帳號**：A 建立的筆記本，B 也看得到
2. **操作可能互相影響**：如果 A 正在處理某個筆記本，B 同時操作可能會產生衝突
3. **需要協調使用**：建議建立使用規範，避免同時對同一個筆記本操作

如果團隊成員需要各自獨立的 NotebookLM 帳號，可能需要部署多個系統實例，每個使用不同的帳號登入。

API 配額問題

簡報編修系統使用 Gemini API，API 有使用量的限制。

當多人同時使用時：

- API 呼叫量會增加
- 可能更快達到配額上限
- 可能需要購買更多配額

建議監控 API 使用量，必要時升級方案。

效能考量

系統的效能有限，特別是瀏覽器自動化（NotebookLM 自動化系統）比較耗資源。

當多人同時使用時：

- 伺服器電腦需要足夠的 RAM 和 CPU
- 可能需要限制同時使用的人數
- 考慮使用更強的硬體或雲端伺服器

故障排除

問題一：其他電腦無法連線

可能原因：

1. 防火牆阻擋
2. 監聽位址仍是 localhost
3. IP 位址不正確

解決步驟：

1. 確認已修改監聽位址為 `0.0.0.0`
2. 確認已在防火牆開放對應的 port
3. 確認使用正確的伺服器 IP 位址
4. 在伺服器上測試：`curl http://localhost:5000` 確認服務正常

問題二：連線很慢或經常斷線

可能原因：

1. 網路品質不佳
2. 伺服器負載過高
3. 使用的是開發伺服器，不適合多人使用

解決方法：

1. 檢查網路連線品質
2. 監控伺服器的 CPU 和記憶體使用
3. 改用 Waitress 或 Gunicorn 等正式的 WSGI 伺服器

問題三：登入狀態失效

可能原因： Cookie 過期或被清除。

解決方法： 在伺服器上重新執行 `notebooklm login`，完成登入流程。

本章小結

本章介紹了如何將 NotebookLM 相關系統部署為可分享的服務。主要內容包括：

區域網路分享：修改監聽位址、設定防火牆，讓同一網路內的電腦都能使用

開機自動啟動：使用啟動資料夾或系統服務，讓系統在電腦開機時自動運行

正式部署：使用 Waitress、Gunicorn 或 Docker，提供更穩定的服務

網際網路存取：使用 ngrok、Cloudflare Tunnel 或 VPN，從任何地方存取系統

安全與協作：考慮認證、安全、多人使用時的注意事項

選擇哪種部署方式取決於你的需求：

- 只有自己使用 → 本機執行即可
- 小團隊、同一辦公室 → 區域網路分享
- 需要隨時隨地存取 → 網際網路存取方案
- 大規模、穩定性要求高 → 正式部署 + 雲端伺服器

下一個單元，我們會深入探討簡報編修系統的前端架構，了解 Vue 3 和 Pinia 狀態管理是如何運作的。這對於想要客製化系統介面的人會很有幫助。