



自然語言處理

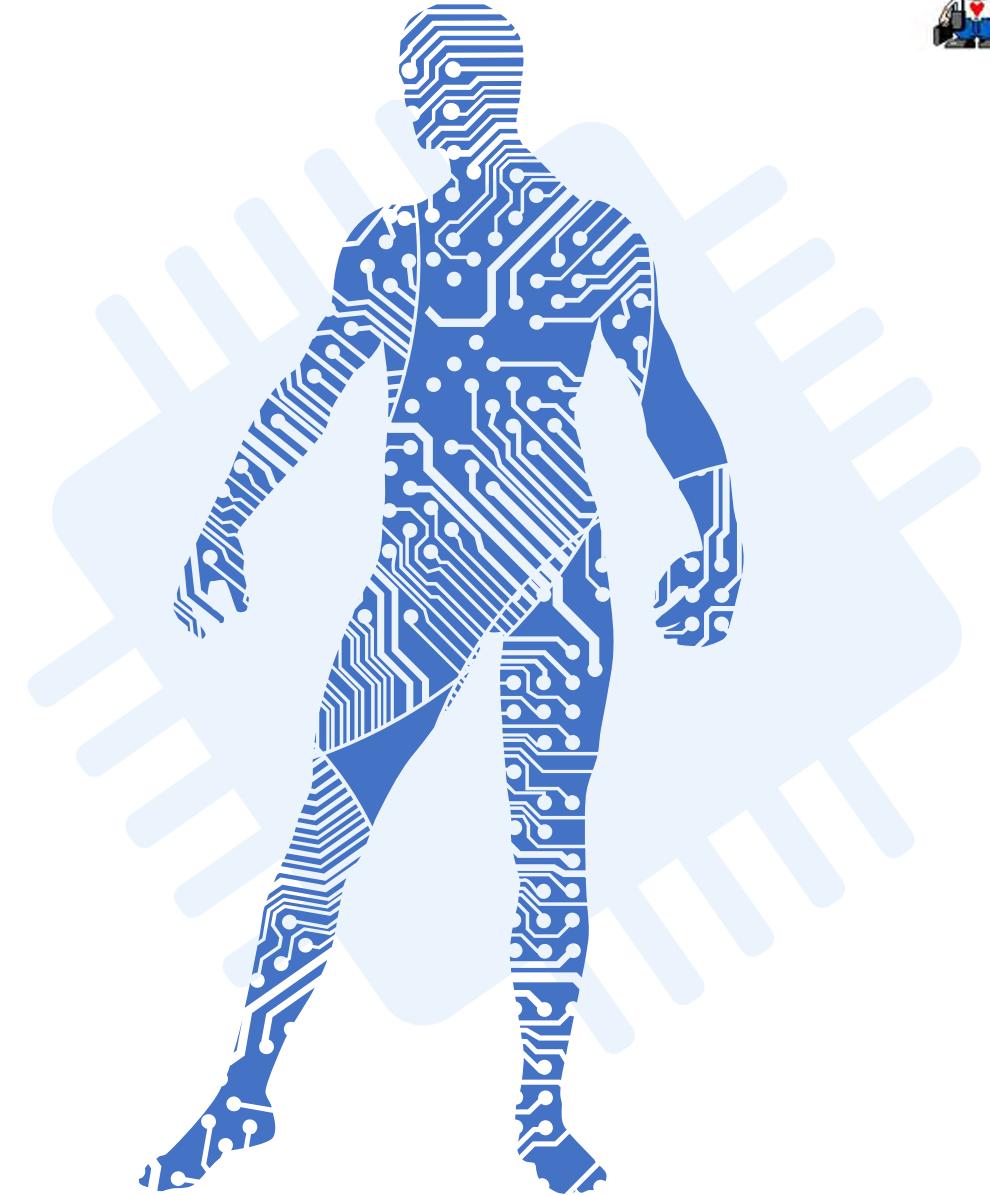
第 10 章 大語言模型專案 LLM Projects

講師：紀俊男



本章大綱

- 專案簡介
- 建立 LINE 開發者帳號
- 建立 Provider & Channel
- 加入好友與調整回應訊息
- 啟動 Webhook

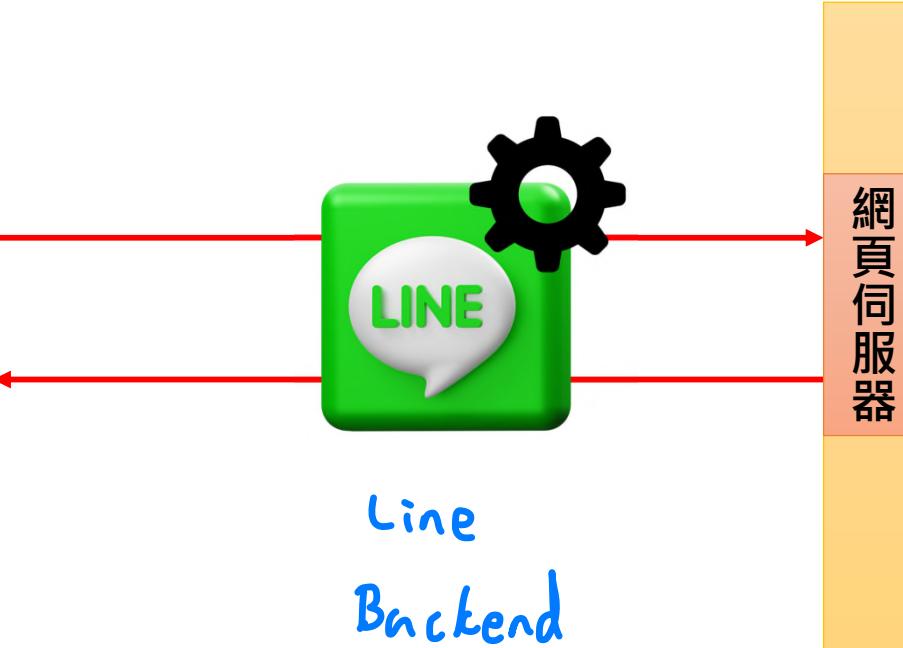
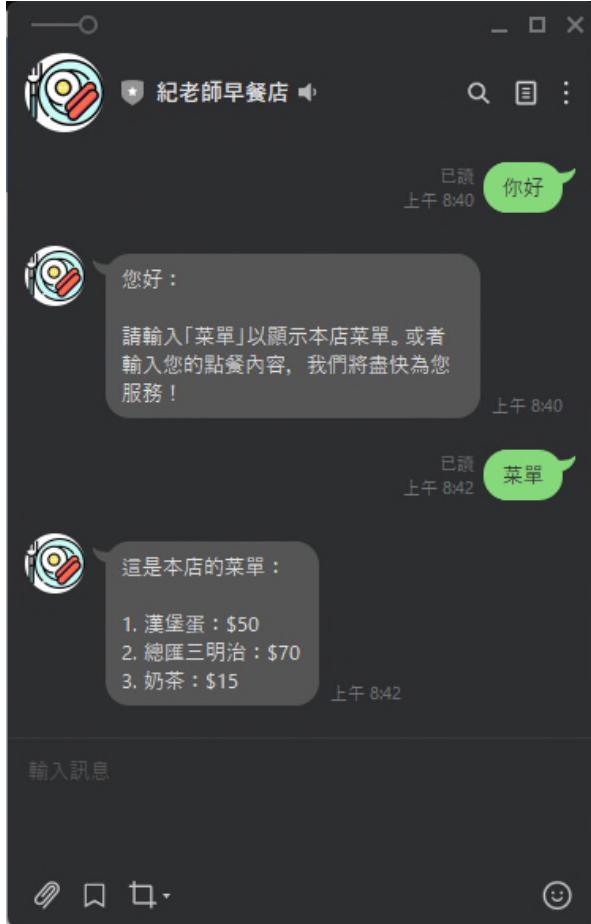




專案簡介



專案完成樣貌



Colab
程式碼

大語言
模型





專案所需元件

- **LINE 開發者帳號**
 - 讓我們有個能夠管控 LINE 訊息的後台。
 - 允許我們把 LINE 訊息轉發至另一個網址。
- **網頁伺服器**
 - 本專案使用 ngrok。
 - 需要申請 ngrok 免費帳號。
- **LINE Bot SDK**
 - 讓 Colab 程式碼能看懂 LINE 傳送過來的封包。
- **大語言模型**
 - 讓我們的聊天機器人更聰明。





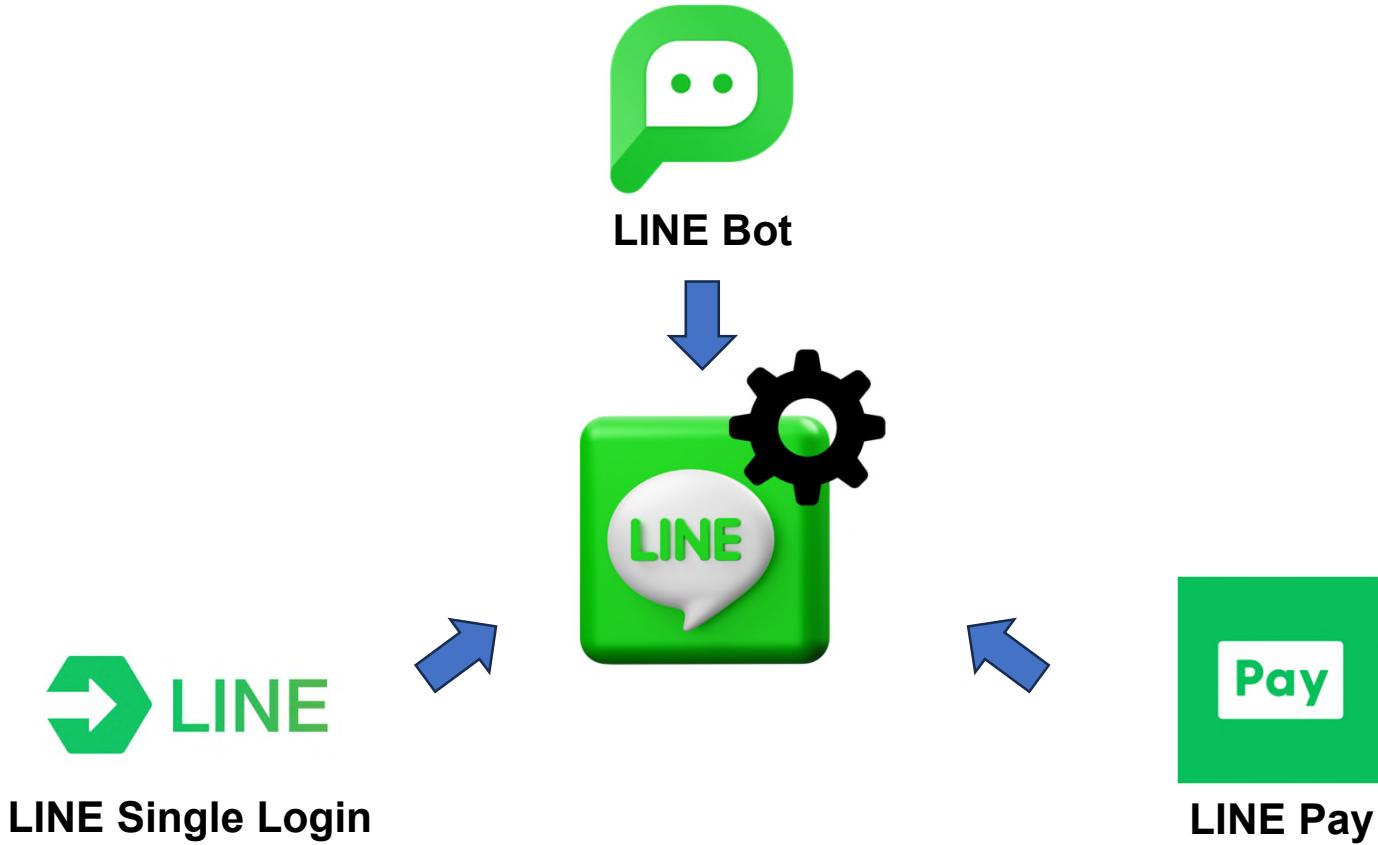
**建立 LINE
開發者帳號**



何謂 LINE 開發者帳號



- 允許你用**程式碼**，**介入**、**管理**下列**三種 LINE 服務的帳號**





如何建立 LINE 開發者帳號

- <https://developers.line.biz/zh-hant/>

The screenshot shows the homepage of the LINE Developers site. At the top, there is a navigation bar with links for 'LINE Developers', '產品', '文件', '最新消息', '英文版首頁', '其他', a search icon, and a language dropdown set to '繁體中文'. A red circle highlights the 'Log in' button. Below the navigation bar, a banner says 'Easy registration with your LINE Account!' with a link. The main heading 'Connect with LINE Developers' is displayed above a laptop and a smartphone displaying developer tools. Three buttons below the heading are 'Console', 'Documentation', and 'About LINE Developers site'. At the bottom right of the page, there is a '繁體中文' dropdown menu.

LINE Business ID

使用LINE帳號登入

或

使用商用帳號登入

建立帳號

使用LINE商用ID須遵從服務條款等規定，登入後即視為您同意相關條款內容。

② 關於LINE商用ID

繁體中文

Help 服務條款 © LY Corporation





- 首次申請 Developers 帳號，需輸入姓名與 Email

Confirm that the following information is correct and select Register.
Developer information can be modified after registration.

Name ② Max. 200 characters

Email address ② Max. 100 characters

LINE Developers Agreement: I have read and agree to the LINE Developers Agreement.

Confirm



如何建立 LINE 開發者帳號

- 進入 Developers 帳號首頁

The screenshot shows the LINE Developers Console homepage. At the top, there is a navigation bar with links for Products, Documents, News, FAQ, Community, and Blog. On the right side, there is a search bar and a user profile icon. The main content area is titled "Provider List" and features a sub-header "Welcome to LINE Developers Console !". It includes a brief introduction: "Let's develop an app that connects people with people using your development technology and LINE Platform! A provider is a service provider (company / individual), and we begin by creating a provider." Below this, there are three steps illustrated with icons: STEP 1 shows a folder with a green checkmark; STEP 2 shows a folder and three circular icons representing different services; STEP 3 shows a smartphone with a code editor interface. Each step has a corresponding explanatory text below it. A prominent blue button at the bottom center says "+ Create New Provider".

LINE Developers Products Documents News FAQ Community Blog

Welcome 紀俊男 (Robert)

Providers

Has not provider

Tools

Provider List

Welcome to LINE Developers Console !

Let's develop an app that connects people with people using your development technology and LINE Platform!

A provider is a service provider (company / individual), and we begin by creating a provider.

STEP 1

Let's start by creating a new provider

STEP 2

For each provider, you can create a channel for LINE login and Messaging API

STEP 3

Let's finish setting up the application on each channel and develop your own application

+ Create New Provider





隨堂練習：建立開發者帳號



- 請先開啟瀏覽器，拜訪下列網址：
 - <https://developers.line.biz/zh-hant/>
- 依照前述方法，建立一個屬於您自己的「**開發者帳號**」。





建立 Provider &
Channel



何謂 LINE Provider & Channel

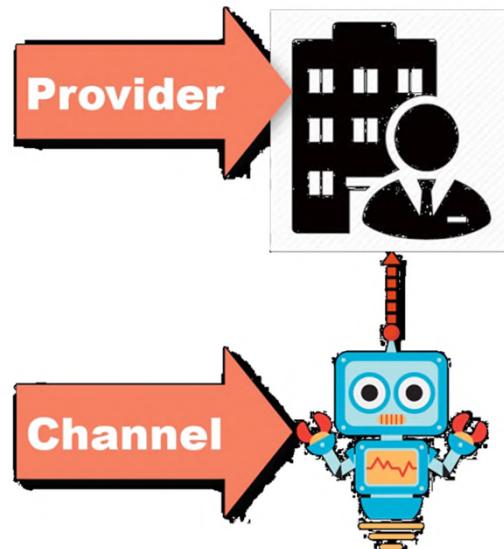


- **Provider**

- 1 Provider = 一家公司名稱
- 如：「[紀老師餐飲有限公司](#)」
- 一個帳號，可以擁有多個 Provider

- **Channel**

- 1 Channel = 一個品牌、服務
- 如：「[紀老師早餐店](#)」
- 一個 Provider，可以擁有多個 Channel





建立 Provider

Providers (1) Create

Create a new provider

Provider name ?

紀老師餐飲有限公司

- ✓ Don't leave this empty
- ✓ Don't use special characters (4-byte Unicode)
- ✓ Enter no more than 100 characters

A provider is an individual developer, company, or organization that provides services. For more details, see the [documentation](#).

Cancel

Create





建立 Channel



TOP > 紀老師餐飲有限公司

紀老師餐飲有限公司

Channels Roles Settings

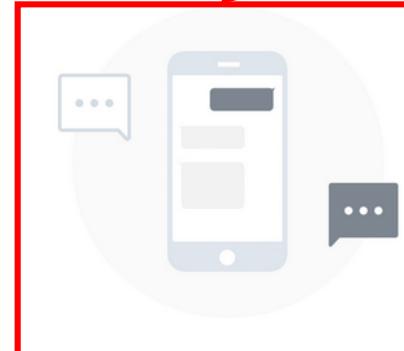
一定要選「Messaging API」才是 Chatbot

This provider doesn't have any channels yet

To create one, choose a channel type below



Create a LINE Login channel



Create a Messaging API channel



Create a Blockchain Service channel



Create a LINE MINI App channel

Available features might defer based on the account with which you are currently logged in. For details, check the [document](#).





建立 Channel

Create a new channel

Channel type

Messaging API

Don't leave this empty

Provider

紀老師餐飲有限公司

Don't leave this empty

Company or owner's country or region

Taiwan

Corporations should select their company residence.

Don't leave this empty

Channel icon
optional



4

Channel name

紀老師早餐店

Note: The channel name can't be changed for several days.
 Don't leave this empty
 Don't use special characters (4-byte Unicode)
 Enter no more than 20 characters

Email address

contact@robertchi.tw

Don't leave this empty
 Enter a valid email address
 Enter no more than 100 characters

5

Channel description

這是「紀老師早餐店」的聊天機器人

Don't leave this empty
 Don't use special characters (4-byte Unicode)
 Enter no more than 500 characters

Privacy policy URL

optional

Enter privacy policy URL

Enter a valid HTTPS URL
 Enter no more than 500 characters

6

Category

餐飲

Don't leave this empty

8

I have read and agree to the [LINE Official Account Terms of Use](#)
 I have read and agree to the [LINE Official Account API Terms of Use](#)

Select the checkbox after reading the related document

7

Subcategory

早餐店

Don't leave this empty

9

Create





建立 Channel

同意我們使用您的資訊

LY Corporation為了完善本公司服務，需使用企業帳號（包括但不限於LINE官方帳號及其相關API產品；以下合稱「企業帳號」）之各類資訊。若欲繼續使用企業帳號，請確認並同意下列事項。

■ 我們將會蒐集與使用的資訊

- 用戶傳送及接收的傳輸內容（包括訊息、網址資訊、影像、影片、貼圖及效果等）。
- 次數、時間長度及接收發送對象等（下稱「格式等資訊」）。
- 內容格式等資訊。
- 已接收內容是否已讀、網址的點選等（包括但不限於）、及LY Corporation隱私權政策所述的其他資訊。

Create a Messaging API channel with the following details?

Channel name: 紀老師早餐店
Official Account name: 紀老師早餐店
Provider: 紀老師餐飲有限公司

- If you proceed, an official account will be created with the same name as the messaging API channel above.
- You cannot change the channel provider after the channel is created. Make sure that the provider and official account owner are the same individual developer, company or organization.
- For the handling of LINE user information, please refer to [User Data Policy](#).

OK

Cancel

- 提供最佳服務，包括呈現LY Corporation所瞭解用戶偏好的廣告。

若您對本同意書內容有任何問題或意見，請透過[聯絡表單](#)與我們聯繫。

如果授予此處同意的人不是企業帳號所有人所授權之人，請事先取得該被授權人的同意。如果LY Corporation得中止該企業帳號的使用，且不為因此而生的任何情事負責。

註：本規則以日文版做成，英文版僅供參考，如有任何歧異應以日文版內容為準。

Agree



紀老師早餐店

Admin

Messaging API

Basic settings

Basic information

Channel ID: 2005521558

Channel icon





隨堂練習：建立 Provider & Channel



- 請依照前述方法，建立您自己的「Provider」& 「Channel」。

紀老師餐飲有限公司

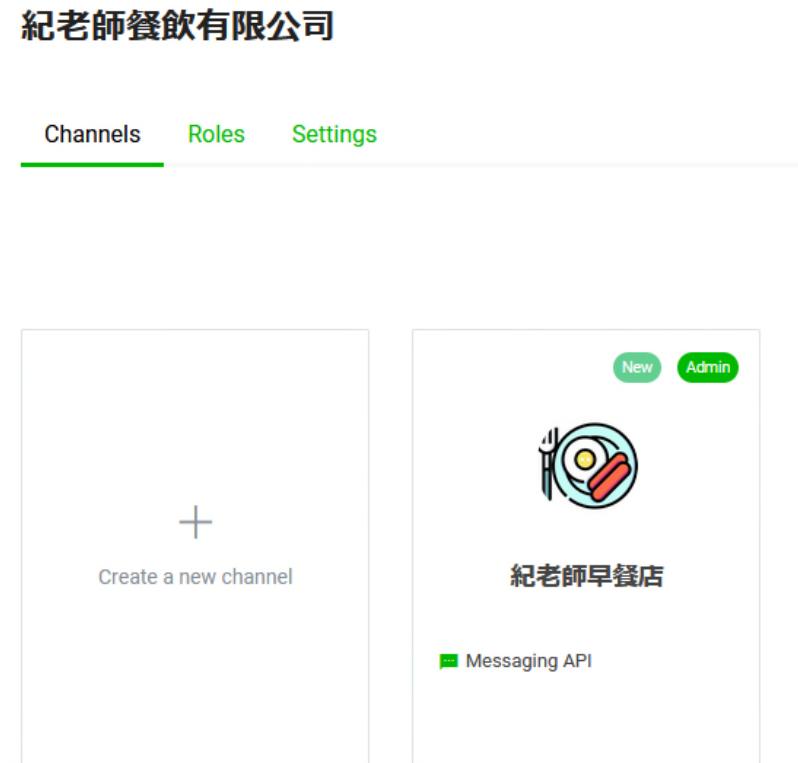
Channels Roles Settings

Create a new channel

New Admin

紀老師早餐店

Messaging API





加入好友與
調整回應訊息



加入好友

紀老師早餐店

Admin | Messaging API

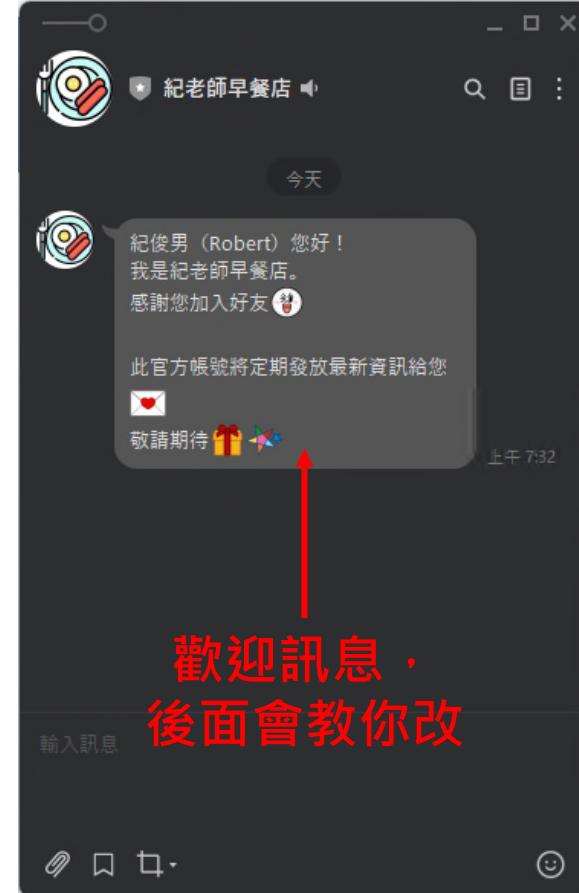
Basic settings **Messaging API** LIFF Security Statistics Roles

Messaging API settings

Bot information

Bot basic ID @741seseh

QR code



第一次加入好友時才會出現



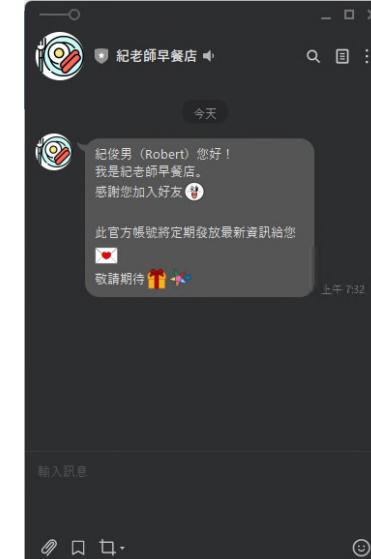
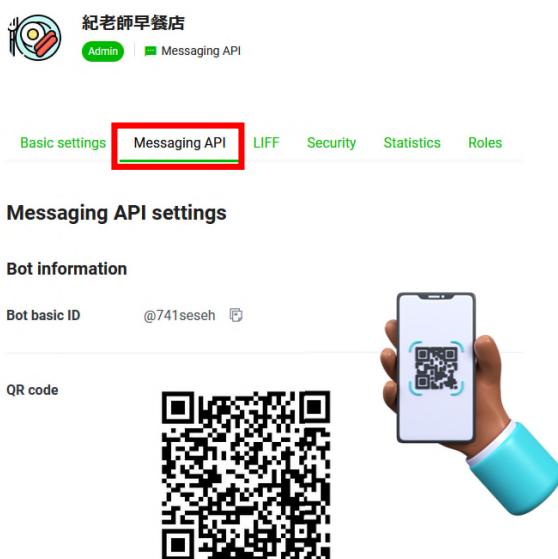
輸入訊息看不懂時會出現





隨堂練習：加入好友

- 請依照前述方法，將您新建的 LINE 帳號加為好友。
- 請觀察加入好友那一剎那的「歡迎訊息」，之後會教你修改方法。
- 隨便輸入任何字串，觀察「自動回覆訊息」，之後會教你修改方法。





修改「歡迎訊息」

Basic settings **Messaging API** LIFF Security Statistics Roles

LINE Official Account features

Edit the message text and other settings for these features in the LINE Account Manager

Allow bot to join group chats **Disabled**

Auto-reply messages **Enabled**

Greeting messages **Enabled**

回應設定

可配合帳號的經營目的，設定聊天及自動訊息的回應方式。

回應功能

聊天

可透過聊天與好友互動。

加入好友的歡迎訊息

當用戶將本帳號加為好友時，可自動傳送訊息內容
[開啟「加入好友的歡迎訊息」設定](#)

Webhook

當用戶傳送訊息給本帳號或將本帳號加為好友時
[開啟Messaging API的設定](#)

自動回應訊息

可使用事先設定好的訊息內容進行自動回覆。
[開啟自動回應訊息的設定](#)

Edit

訊息設定

範本

好友的顯示名稱，早安！
歡迎加入 **帳號名稱**。
感謝您加入好友

您可以輸入與「菜單」有關的文字，來取得菜單

或者直接點擊，我會告訴您餐點價格，以及可以用餐的大概時間喔～

表情貼 好友的顯示名稱 帳號名稱

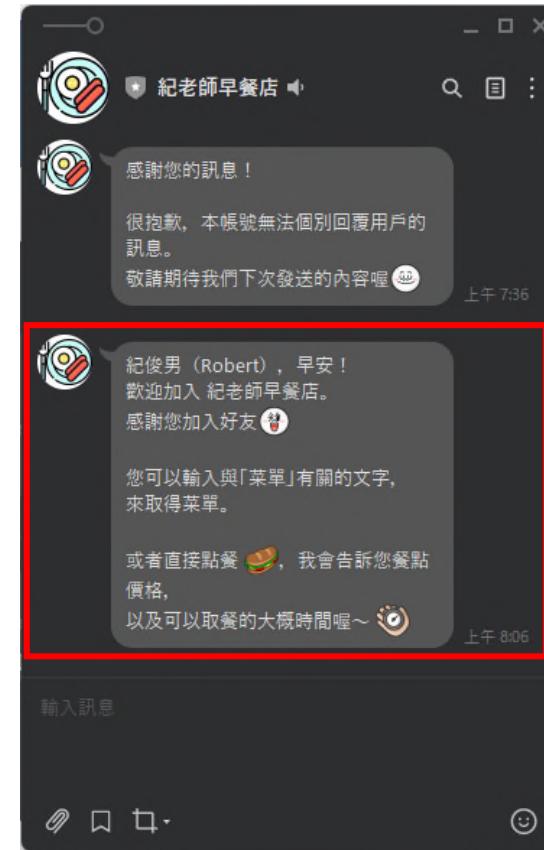
若於訊息中加入「好友的顯示名稱」，則只會傳送給已允許您存取其個人檔案的用戶。

122/500





驗證「歡迎訊息」





隨堂練習：修改歡迎訊息



- 請依照前述方法，修改您的歡迎訊息。
 - 接著用下列方法，驗證您修改的歡迎訊息：
 - 封鎖該聊天帳號。
 - 重新「解除封鎖」該帳號。





修改「自動回覆訊息」

Basic settings **Messaging API** LIFF Security Statistics Roles

LINE Official Account features

Edit the message text and other settings for these features in the LINE Official Account Manager

Allow bot to join group chats Disabled

[Edit](#)

Auto-reply messages Enabled

[Edit](#)

Greeting messages Enabled

[Edit](#)

回應設定

可配合帳號的經營目的，設定聊天及自動訊息的回應方式。

回應功能

聊天



可透過聊天與好友互動。

加入好友的歡迎訊息



當用戶將本帳號加為好友時，可自動傳送訊息內容。

[開啟「加入好友的歡迎訊息」設定](#)

Webhook



當用戶傳送訊息給本帳號或將本帳號加為好友時，從LINE平台傳送Webhook事件至Webhook網址。

[開啟Messaging API的設定](#)

自動回應訊息



可使用事先設定好的訊息內容進行自動回應。

[開啟自動回應訊息的設定](#)

內容	標題	回應類型	指定日期或時間	使用
感謝您的訊息！	Default	一律回應	永遠	<input checked="" type="checkbox"/>





修改「自動回覆訊息」

回應設定

回應類型 一律回應
系統會回覆所有的訊息。

關鍵字回應
系統會在收到與關鍵字完全一致的訊息內容時進行回覆。※若已登錄多個關鍵字，會在內容時進行回覆。

選項設定 指定日期或時間
僅希望系統在特定期間或時段進行回覆時選擇。若已設定回應時間，則不需選擇此選項

訊息設定

訊息內容：
您好：
請輸入「菜單」以顯示本店菜單。或者輸入您的點餐內容，我們將盡快為您服務！|

表情貼 好友的顯示名稱 帳號名稱

若於訊息中加入「好友的顯示名稱」，則只會傳送給已允許您存取其個人檔案的用戶。

自動回應訊息

若在此事先建立回應內容，當用戶傳訊息給您且符合設定條件時，系統將會自動回傳訊息。

建立

標題

menu | 4/20

回應設定

回應類型 一律回應
系統會回覆所有的訊息。

關鍵字回應
系統會在收到與關鍵字完全一致的訊息內容時進行回覆。※若已登錄多個關鍵字，會在收到與任一關鍵字完全一致的內容時進行回覆。

訊息設定

訊息內容：
這是本店的菜單：
1. 漢堡蛋：\$50
2. 總匯三明治：\$70
3. 奶茶：\$15 | 43/500

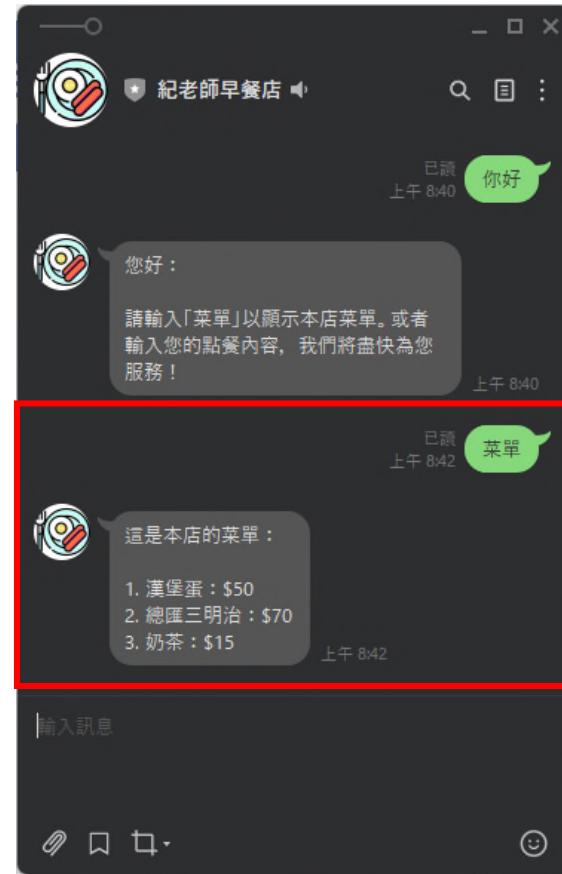
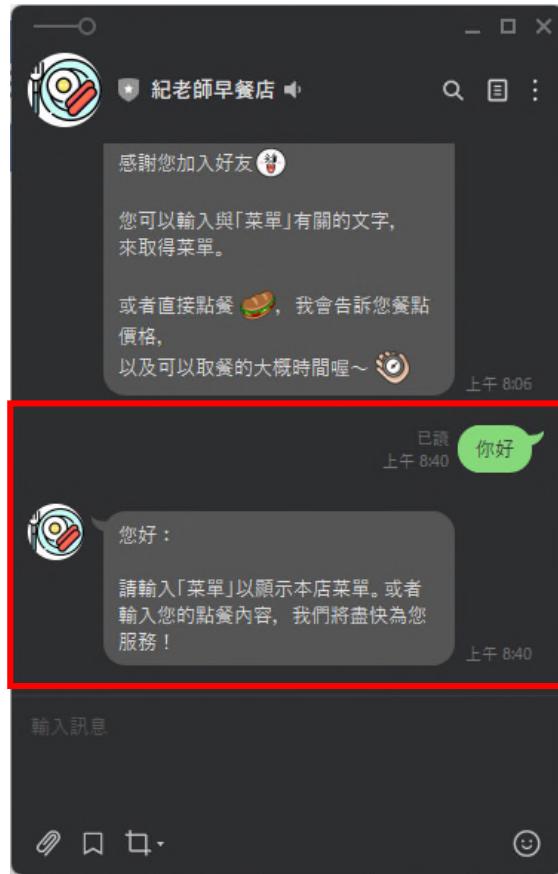
表情貼 好友的顯示名稱 帳號名稱

若於訊息中加入「好友的顯示名稱」，則只會傳送給已允許您存取其個人檔案的用戶。





驗證「自動回覆訊息」





隨堂練習：修改自動回覆訊息



- 請依照前述方法，添加一個「一律回應」的自動回覆訊息。
- 再添加一個輸入「菜單」，就能回應本店菜單的「關鍵字回應」訊息。
- 接著到 LINE 聊天畫面，驗證此兩個訊息是否正常運作？

訊息設定

您好：
請輸入「菜單」以顯示本店菜單。或者輸入您的點餐內容，我們將盡快為您服務！|

41/500

表情貼 好友的顯示名稱 帳號名稱

若於訊息中加入「好友的顯示名稱」，則只會傳送給已允許您存取其個人檔案的用戶。

訊息設定

這是本店的菜單：

1. 漢堡蛋 : \$50
2. 總匯三明治 : \$70
3. 奶茶 : \$15

43/500

表情貼 好友的顯示名稱 帳號名稱

若於訊息中加入「好友的顯示名稱」，則只會傳送給已允許您存取其個人檔案的用戶。





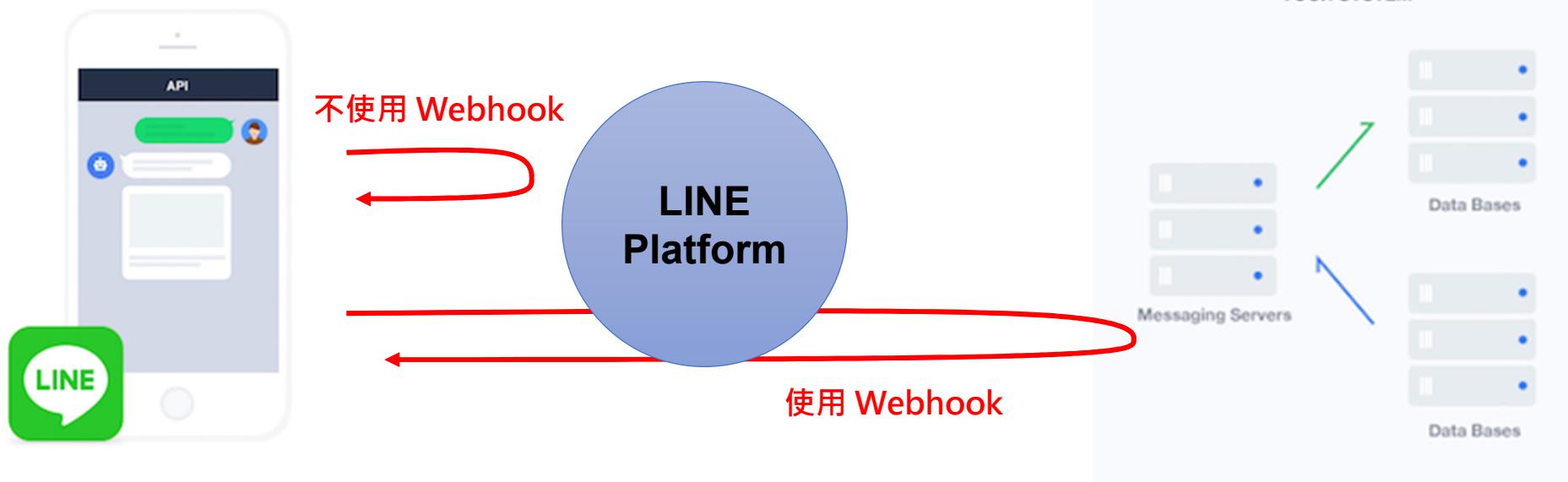
啟動 Webhook

範例完整原始碼：
<https://url.cc/i5phgl>





何謂 Webhook ?





開啟 Webhook 步驟



安裝相關套件

- Ollama
- LangChain
- ...

撰寫攔截程式

- LINE Bot
SDK

將攔截程式掛載
為 Webhook

- ngrok 網頁
伺服器





安裝 GPU 驅動相關套件

```
1 # 更新 Linux 內的套件清單至最新版
2 !apt-get update
3
4 # 安裝 PCI 匯流排工具 (PCI Utility) 與 lshw (LiSt HardWare), 以便能偵測到 GPU
5 # -y : 遇到詢問是否安裝, 一律自動回答 yes
6 !apt-get install -y pciutils lshw
7
8 # 用 nVidia 的 System Management Interface (SMI) 確認 GPU 的確抓得到
9 !nvidia-smi
```

```
+-----+
| NVIDIA-SMI 535.104.05      Driver Version: 535.104.05 CUDA Version: 12.2 |
+-----+
| GPU  Name                  Persistence-M | Bus-Id     Disp.A  | Volatile Uncorr. ECC | | | | |
| Fan  Temp     Perf          Pwr:Usage/Cap | Memory-Usage | GPU-Util  Compute M. |
| |           |             |              |           |          | MIG M.   |
+-----+
| 0  Tesla T4                Off  00000000:00:04.0 Off   0 |
| N/A   65C     P8            11W / 70W |    0MiB / 15360MiB |    0%     Default |
| |           |             |              |           |          | N/A      |
+-----+
+
| Processes:
| GPU  GI  CI          PID  Type  Process name        GPU Memory |
|       ID  ID          ID   ID    name                 Usage  |
|-----+
| No running processes found
+-----+
```





安裝 Ollama



```
1 # 至 https://ollama.com/download/linux
2 # 直接將安裝 Ollama 於 Linux 的指令貼上
3 !curl -fsSL https://ollama.com/install.sh | sh
4
5 # 啟動 Ollama，讓它執行於背景中
6 # ollama serve: 用 Server 模式、而非互動模式執行 Ollama
7 # > server.log：將本應顯示於螢幕的訊息，轉向輸出至 server.log 這個檔備查
8 # 2>&1：2 為 stderr。將所有錯誤訊息，轉向 &1 (stdout, 螢幕) 輸出。
9 # &：將程式啟動之後，馬上返回，不要等該程式執行完成
10 !ollama serve > server.log 2>&1 &
11
12 # 將 Llama-3 模型下載，並做為此次的大語言模型
13 # ollama run <模型名稱>：下載並執行特定 LLM
14 # > model.log：將本應顯示於螢幕的訊息，轉向輸出至 model.log 這個檔備查
15 !ollama run gemma:7b > model.log 2>&1 &
16
17 # 注意：上述兩指令皆以「&」後綴，告知 Colab「不用等 Linux 執行完」。
18 # 但事實上，不論啟動為 Server，或下載 LLM，皆須 1~5 分鐘不等的時間。
19 # 可以查看 server.log、model.log 兩檔案內容，得知當前執行狀況。
```





安裝 LangChain 與撰寫相關函數



```
1 # 下載 LangChain, 一套專門連上各種大語言模型的 Python 套件
2 !pip install langchain # LangChain 核心元件
3 !pip install langchain-community # 各種開源大語言模型連接函數套件
4
5 # 載入 ollama 以便連上後端 LLM
6 from langchain_community.llms import Ollama
7
8 # LLM 名稱需與 ollama run 後方名稱相同
9 llm = Ollama(model="gemma:7b")
10
11 # 定義一個以 LLM 處理使用者輸入的函數
12 def llm_chat(input_text):
13     msg = llm.invoke(input_text)
14
15     return msg
```





隨堂練習：設定相關環境



- 請依照前述投影片內容，分別撰寫好下列程式碼：

- 安裝 GPU 驅動相關套件
- 安裝 Ollama
- 安裝 LangChain 與撰寫相關函數

```
1 # 安裝 GPU 驅動相關套件
2 !apt-get update
3 !apt-get install -y pciutils lshw
4 !nvidia-smi
5
6 # 安裝 Ollama
7 !curl -fsSL https://ollama.com/install.sh | sh
8 !ollama serve > server.log 2>&1 &
9 !ollama run gemma:7b > model.log 2>&1 &
```

```
11 # 下載 LangChain
12 !pip install langchain
13 !pip install langchain-community
14
15 # 載入 Ollama 以便連上後端 LLM
16 from langchain_community.llms import Ollama
17
18 # LLM 名稱需與 ollama run 後方名稱相同
19 llm = Ollama(model="gemma:7b")
20
21 # 定義一個以 LLM 處理使用者輸入的函數
22 def llm_chat(input_text):
23     msg = llm.invoke(input_text)
24
25     return msg
```





安裝 ngrok

• 何謂 ngrok ?

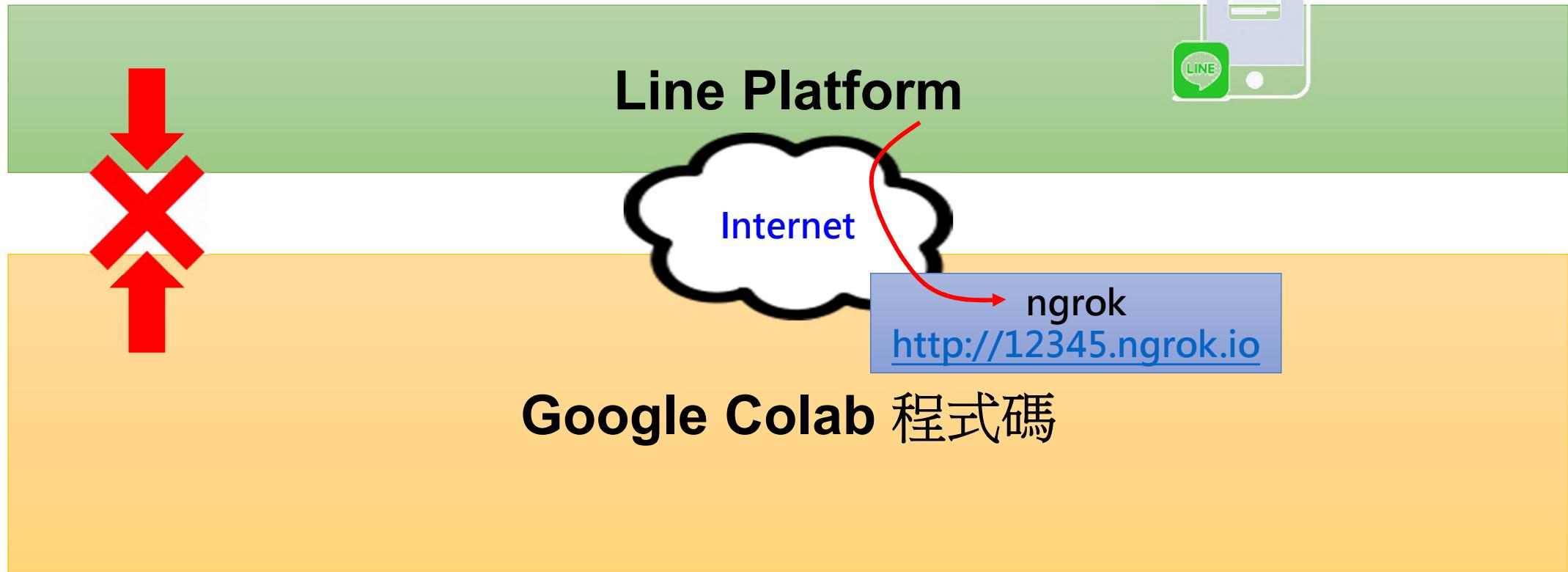
ngrok

- 讓你把執行於本地端的程式，暴露給公共網路。
- 執行方法
 - ngrok http <http://localhost:5000>
- 執行後，會以亂數配發一個公共網址。
 - 如：<http://12345.ngrok.io> -> <http://localhost:5000>
- 外界可以用 <http://12345.ngrok.io> 連向 localhost 服務。



安裝 ngrok

- 為何要使用 ngrok ?





安裝 ngrok

- 申請 ngrok 帳號：<https://ngrok.com/>

The screenshot shows the ngrok homepage with a dark background. At the top, there's a navigation bar with links: Platform, Use cases, Blog, Resources, Docs, Pricing, and Get ngrok. To the right of the navigation bar are two buttons: 'Log in' and 'Sign up'. The 'Sign up' button is highlighted with a red circle. Below the navigation bar, the text 'Unified Ingress Platform for developers' is displayed in large white font. Underneath this text, a subtext reads: 'ngrok combines your reverse proxy, firewall, API gateway, and global load balancing to deliver apps and APIs.' At the bottom of the page, there are two buttons: 'Sign up for free' and 'Technical documentation →'. To the right of the 'Sign up for free' button, there's a large graphic consisting of several overlapping colored rectangles (blue, yellow, red) forming a geometric pattern.

The screenshot shows the 'Sign up' form on the ngrok website. It has fields for Name (filled with 'Robert Chi'), Email (filled with 'contact@robertchi.tw'), and Password (a series of dots). There's also a reCAPTCHA field with a green checkmark and the text '我不是機器人'. Below the fields are two buttons: 'Sign up' and 'or', followed by 'Sign up with GitHub' and 'Sign up with Google' buttons.



安裝 ngrok

- 拷貝安裝指令與登入密鑰

Welcome

ngrok is your app's front door—a globally distributed reverse proxy that secures, protects and accelerates your applications and network services, no matter where you run them.

Agents

- MacOS
- Windows
- Linux** (highlighted with a red box)
- Docker
- RaspberryPi
- FreeBSD

SDKs

- Go
- Kubernetes
- Node.js

Infrastructure

Looking for something?

Agent

Linux

Choose another platform

Installation

Apt Download Snap

Install ngrok via Apt with the following command:

```
curl -s https://ngrok-agent.s3.amazonaws.com/ngrok.asc \
| sudo tee /etc/apt/trusted.gpg.d/ngrok.asc >/dev/null \
&& echo "deb https://ngrok-agent.s3.amazonaws.com buster main" \
| sudo tee /etc/apt/sources.list.d/ngrok.list \
&& sudo apt update \
&& sudo apt install ngrok
```

Run the following command to add your authtoken to the default `ngrok.yml` configuration file.

```
ngrok config add-authtoken 1TWJSwmSTyaHLvn0JhnjV1pFP5W_5XaVvtwaze1ACiQMDBdm7
```





安裝 ngrok

• 完整程式碼

```
1 # 安裝 ngrok
2 !curl -s https://ngrok-agent.s3.amazonaws.com/ngrok.asc \
3 | sudo tee /etc/apt/trusted.gpg.d/ngrok.asc >/dev/null \
4 && echo "deb https://ngrok-agent.s3.amazonaws.com buster main" \
5 | sudo tee /etc/apt/sources.list.d/ngrok.list \
6 && sudo apt update \
7 && sudo apt install ngrok
8
9 # 將你自己的 ngrok 登入 Token 設定好
10 # ngrok 登入 Token 每人不同，可在登入 ngrok 後，由安裝或左側 Your Authtoken 取得
11 !ngrok config add-authtoken 1TWJSwmSTyaHLvn0JhnjV1pFP5W_5XaVvtwaze1ACiQMDBdm7
```





隨堂練習：安裝 ngrok

- 請先至 ngrok 網站申請帳號：
 - <https://ngrok.com/>
- 登入後，在左側欄中，選擇 Getting Started > Setup & Installation。
- 選擇 Linux 為安裝平台。
- 將屬於自己的安裝指令，拷貝至 Colab 如下：

```
1 # 安裝 ngrok
2 !curl -s https://ngrok-agent.s3.amazonaws.com/ngrok.asc \
3     | sudo tee /etc/apt/trusted.gpg.d/ngrok.asc >/dev/null \
4     && echo "deb https://ngrok-agent.s3.amazonaws.com buster main" \
5     | sudo tee /etc/apt/sources.list.d/ngrok.list \
6     && sudo apt update \
7     && sudo apt install ngrok
8
9 # 將你自己的 ngrok 登入 Token 設定好
10 # ngrok 登入 Token 每人不同，可在登入 ngrok 後，由安裝或左側 Your Authtoken 取得
11 !ngrok config add-authtoken 1TWJSwmSTyaHLvn0JhnjV1pFP5W_5XaVvtwaze1ACiQMDBdm7
```





攔截 LINE 訊息



- 請先至 LINE Bot SDK 網站：
 - <https://github.com/line/line-bot-sdk-python>

The screenshot shows the GitHub repository page for the LINE Messaging API SDK for Python. The page has a dark theme. At the top, there are links for README, Code of conduct, and Apache-2.0 license. Below that, the title "LINE Messaging API SDK for Python" is displayed, along with a "pypi package 3.11.0" badge. A brief description follows: "SDK of the LINE Messaging API for Python." Under the "Introduction" section, it says: "The LINE Messaging API SDK for Python makes it easy to develop bots using LINE Messaging API, and you can create a sample bot within minutes." The "Documentation" section provides links to official documentation in English (<https://developers.line.biz/en/docs/messaging-api/overview/>) and Japanese (<https://developers.line.biz/ja/docs/messaging-api/overview/>). The "Requirements" section lists "Python >= 3.8".





攔截 LINE 訊息



- 拷貝「安裝指令」與「攔截程式碼」至 Colab

Installation

```
$ pip install line-bot-sdk
```

Synopsis

Usage:

```
from flask import Flask, request, abort

from linebot.v3 import (
    WebhookHandler
)
from linebot.v3.exceptions import (
    InvalidSignatureError
)
```



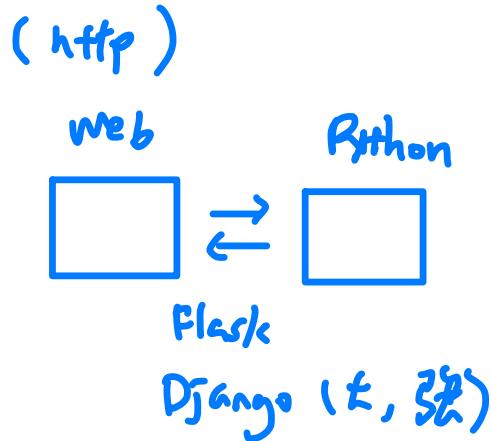


攔截 LINE 訊息

• 程式碼逐行解釋 (1)：套件安裝、引入、建立

```
1 # 安裝 Line Bot SDK  
2 !pip install line-bot-sdk  
3  
4 # 攔截 LINE 訊息主程式，可由 LINE Bot SDK 網站拷貝得到  
5 from flask import Flask, request, abort  
6  
7 from linebot.v3 import (  
8     WebhookHandler  
9 )  
10 ... (略) ...  
11 from linebot.v3.webhooks import (  
12     MessageEvent,  
13     TextMessageContent  
14 )  
15  
16 # 產生一個 Flask 物件，連上 Web 伺服器  
17 app = Flask(__name__)
```

引入必要套件





攔截 LINE 訊息

- 程式碼逐行解釋 (2)：取得 Access Token & Channel Secret

```
19 # Channel Acess Token 類似登入帳號, Channel Secret 類似登入密碼  
20 # 兩者請至 LINE Developer 網站, 相應 Channel 內取得  
21 configuration = Configuration(access_token='YOUR_CHANNEL_ACCESS_TOKEN')  
22 handler = WebhookHandler('YOUR_CHANNEL_SECRET')
```

可變成 username

Colab $\xrightarrow{\text{login}}$ Line

Basic settings Messaging API LIFF Security Statistics Roles

Channel secret ⓘ 713802b414dcef46b41a97429900187b

Issue

Basic settings **Messaging API** LIFF Security Statistics Roles

Channel access token

Channel access token (long-lived) ⓘ

oEkXqVZnTrJBVnMZ1pNZ6wYu+6nWbBiZCJZ8/cpUxwShq51QqrqmsetJbN2MEL1AA30MoUk3g6Yv2f8zQtitwi9yyzVVQrKes6zQu+o4cGv11ErVXsASXVYmiD/FBrxAwpMRjzs7Edj5L9ck8vKirQdB04t89/10/w1cDnyil FU=

Reissue



攔截 LINE 訊息

• 程式碼逐行解釋 (3)：網頁封包處理函數

```
24 # 網頁封包處理函數，不要更動
25 @app.route("/callback", methods=['POST'])
26 def callback():
27     # get X-Line-Signature header value
28     signature = request.headers['X-Line-Signature']
29
30     # get request body as text
31     body = request.get_data(as_text=True)
32     app.logger.info("Request body: " + body)
33
34     # handle webhook body
35     try:
36         handler.handle(body, signature)
37     except InvalidSignatureError:
38         app.logger.info("Invalid signature. Please check your channel access token/channel secret.")
39         abort(400)
40
41     return 'OK'
```





攔截 LINE 訊息

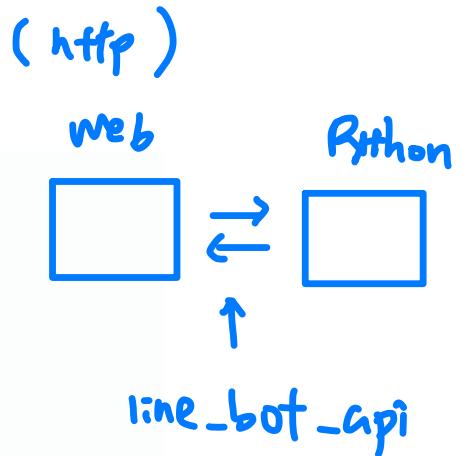
- 程式碼逐行解釋 (4)：訊息處理函數，可客製化

```
1 @handler.add(MessageEvent, message=TextMessageContent)
2 def handle_message(event): ← 收到使用者發過來的一則訊息
3     with ApiClient(configuration) as api_client:
4         line_bot_api = MessagingApi(api_client)
5         line_bot_api.reply_message_with_http_info(
6             ReplyMessageRequest(
7                 reply_token=event.reply_token,
8                 messages=[TextMessage(text=event.message.text)])
9         )
10    )
11
12 if __name__ == "__main__":
13     app.run()
```

將訊息原封不動回傳

Flask

假設 → MessageEvent → ngrok → Python





攔截 LINE 訊息

- 程式碼逐行解釋（4）：訊息處理函數，可客製化

```
43 # 網頁訊息處理函數，可以客製化
44 # event.message.text 就是使用者輸入的文字
45 @handler.add(MessageEvent, message=TextMessageContent)
46 def handle_message(event): ← 收到使用者發過來的一則訊息
47     with ApiClient(configuration) as api_client:
48         line_bot_api = MessagingApi(api_client)
49
50     # 將使用者輸入的訊息，送往大語言模型處理
51     reply_message = llm_chat(event.message.text)
52
53     # 回覆使用者輸入的訊息
54     line_bot_api.reply_message_with_http_info(
55         ReplyMessageRequest(
56             reply_token=event.reply_token,
57             messages=[TextMessage(text=reply_message)])
58     )
59
60
61 if __name__ == "__main__":
62     app.run()
```

將 LLM 訊息回傳





攔截 LINE 訊息

- 將服務掛載至 LINE 後台

取得服務執行於本機的 URL

```
* Serving Flask app '__main__'  
* Debug mode: off  
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.  
* Running on http://127.0.0.1:5000  
INFO:werkzeug:Press CTRL+C to quit
```

終端機 X

```
/content# ngrok http http://localhost:5000
```

Try our new Traffic Inspector: <https://ngrok.com/r/ti>

Session	Status					
Account	online					
Version	Robert Chi (Plan: Free)					
Region	3.10.0					
Latency	United States (us)					
Web Interface	29ms					
Forwarding	http://127.0.0.1:4040 https://a6e5-34-42-61-141.ngrok-free.app					
Connections	ttl	opn	rt1	rt5	p50	p90
	0	0	0.00	0.00	0.00	0.00

記下此網址





攔截 LINE 訊息

- 將服務掛載至 LINE 後台

Webhook settings

記得多加一個
/callback

輸入上一頁的網址

Webhook URL ②

✓ Don't leave this empty
✓ Enter a valid HTTPS URL
✓ Enter no more than 500 characters

Update **Cancel**

Use webhook ② 記得打開這個開關

Webhook settings

Webhook URL ② https://a6e5-34-42-61-141.ngrok-free.app/callback

Verify Edit

Success

OK

Remember to disable auto reply & default reply





隨堂練習：攔截 LINE 訊息



- 請依照先前投影片指示，攔截 LINE 訊息。

