

Building Web APIs with FastAPI

Lecturer: Bessy

Date: Sep. 1st, 2022



Whoami

- Name: Bessy, Chia, 丰嘉
- Experience:
 - 國衛院 生物資訊工程師 兼計畫助理 (現任)
 - 學生資訊社群 SIRLA、Project Abyss 共同創辦人 (現任)
 - 國衛院 Digi⁺ Talent 研習生
 - 專題「基因體大數據分析與精準醫療應用」



Outline

- Introduction to FastAPI
 - What is FastAPI ? What is an API ?
 - HTTP Methods for RESTful API
- Build a basic Web API with FastAPI & Google Sheet
 - Get all data, Query data
 - Insert data, Update data, Delete data (補充)

Introduction to FastAPI

What is FastAPI ?



FastAPI

web framework

FastAPI framework, high performance, easy to learn, fast to code, ready for production



Test

passing

coverage

100%

pypi package

v0.79.1

python

3.6 | 3.7 | 3.8 | 3.9 | 3.10

What is FastAPI ?

- Key features
 - 1. **Fast:** Very high performance, on par with NodeJS and Go (thanks to Starlette and Pydantic). **One of the fastest Python frameworks available.**
 - 2. **Easy:** Designed to be easy to use and learn. Less time reading docs.
 - 3. **Fast to code:** Increase the speed to develop features by about 200% to 300%.
 - 4. **Robust:** Get production-ready code. **With automatic interactive documentation.**
 - 5. **Standards-based:** Based on (and fully compatible with) the open standards for APIs: **OpenAPI** (previously known as Swagger) and **JSON Schema**.



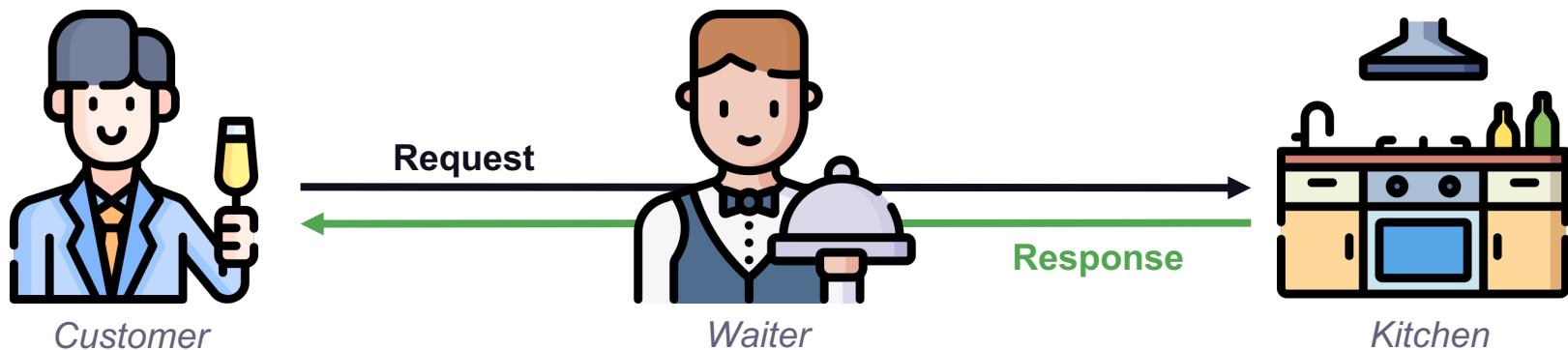
What is an API ?

What is an API ?

“ Software products exchange data and functionalities via machine-readable interfaces – APIs (application programming interfaces). ”

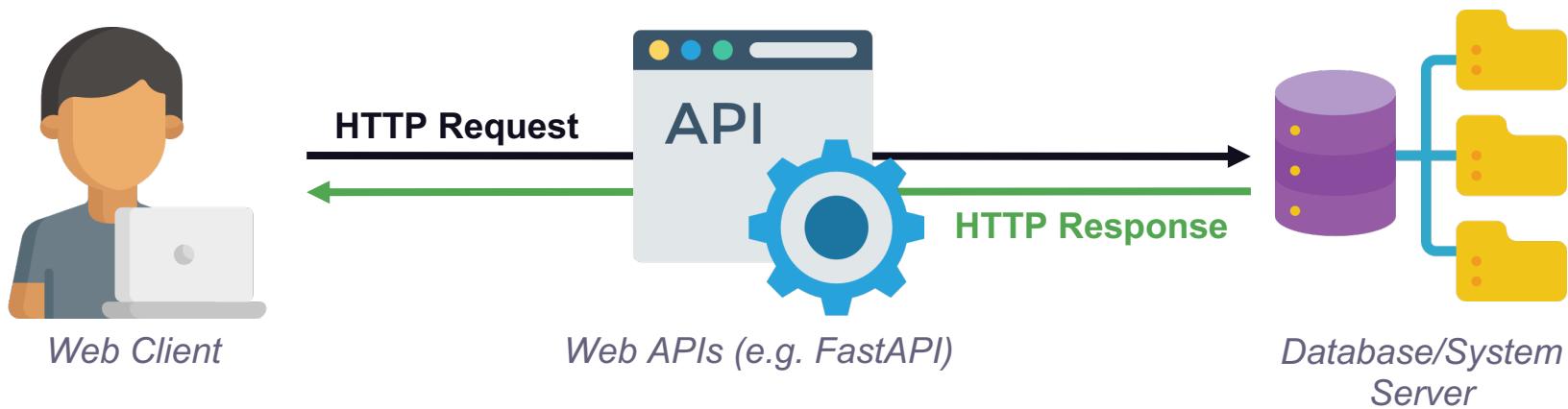
What is an API ?

Real World



What is an API ?

Web Application Development World



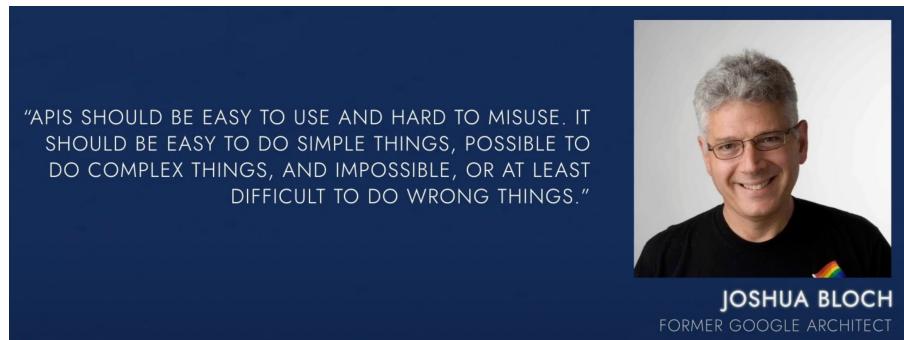
API Principles



1. Limited access



2. System-Independent



JOSHUA BLOCH
FORMER GOOGLE ARCHITECT

3. Simple in use

HTTP Methods for RESTful API

So a client makes a request to the server,
how would the server know which operation to perform ?



Commonly-used HTTP Methods

CRUD: Read

To read/retrieve Data



CRUD: Create

To insert data



To update data

CRUD: Update/Replace

To delete data

CRUD: Delete

HTTP Status Code



HTTP Status Code	Description
1xx: Informational	The request was received and the process is continuing.
2xx: Success	The action was successfully received, understood, and accepted.
3xx: Redirection	Further action must be taken in order to complete the request.
4xx: Client Error	The request contains incorrect syntax or cannot be fulfilled.
5xx: Server Error	The server failed to fulfill an apparently valid request.

Build a basic Web API with FastAPI & Google Sheet

Overview

1. Installation
2. Run a basic Web API with FastAPI (Official sample code)
3. Setup and connect with Google Sheets API
4. Run a Web API with FastAPI & Google Sheet (Database)
 - [GET] Get all data
 - [GET] Query data
 - [POST] Insert data
 - [PUT] Update data
 - [DELETE] Delete data

1. Installation (Windows)

- 請先確定已安裝 Python 3.x 版本。
- 建立專案資料夾、Python 虛擬環境，並在虛擬環境內進行專案開發。

```
# Create isolated Python environment  
$ python -m venv [MyVenv-name]
```

```
# Activate virtual environment  
$ [MyVenv-name]\Scripts\activate.bat
```

```
# Deactivate virtual environment (Select 1 to use)  
(MyVenv-name)$ deactivate  
(MyVenv-name)$ [MyVenv-name]\Scripts\deactivate
```

1. Installation (MacOS)

- 請先確定已安裝 Python 3.x 版本。
- 建立專案資料夾、Python 虛擬環境，並在虛擬環境內進行專案開發。

```
# Create project folder & isolated Python environment
$ mkdir [Project-name]
$ cd [Project-name]
$ python3 -m venv [MyVenv-name]
```

```
# Activate virtual environment
$ source [MyVenv-name]/bin/activate
```

```
# Deactivate virtual environment
(MyVenv-name)$ deactivate
```

1. Installation (Windows/MacOS)

```
# Upgrade pip  
(MyVenv-name)$ python -m pip install --upgrade pip  
  
# Install packages  
(MyVenv-name)$ pip install fastapi  
(MyVenv-name)$ pip install "uvicorn[standard]"
```

2. Run a basic Web API with FastAPI

```
from typing import Union
from fastapi import FastAPI
import uvicorn

app = FastAPI()

@app.get("/")
def read_root():
    return {"Hello": "World"}

@app.get("/items/{item_id}")
def read_item(item_id: int, q: Union[str, None] = None):
    return {"item_id": item_id, "q": q}

if __name__ == "__main__":
    uvicorn.run(app='main:app', host="127.0.0.1", port=8000, reload=True, debug=True)
```

main.py

2. Run a basic Web API with FastAPI

```
(MyVenv-name)$ python main.py
```



← → ⌂ 127.0.0.1:8000/docs

FastAPI 0.1.0 OAS3

/openapi.json

default ^

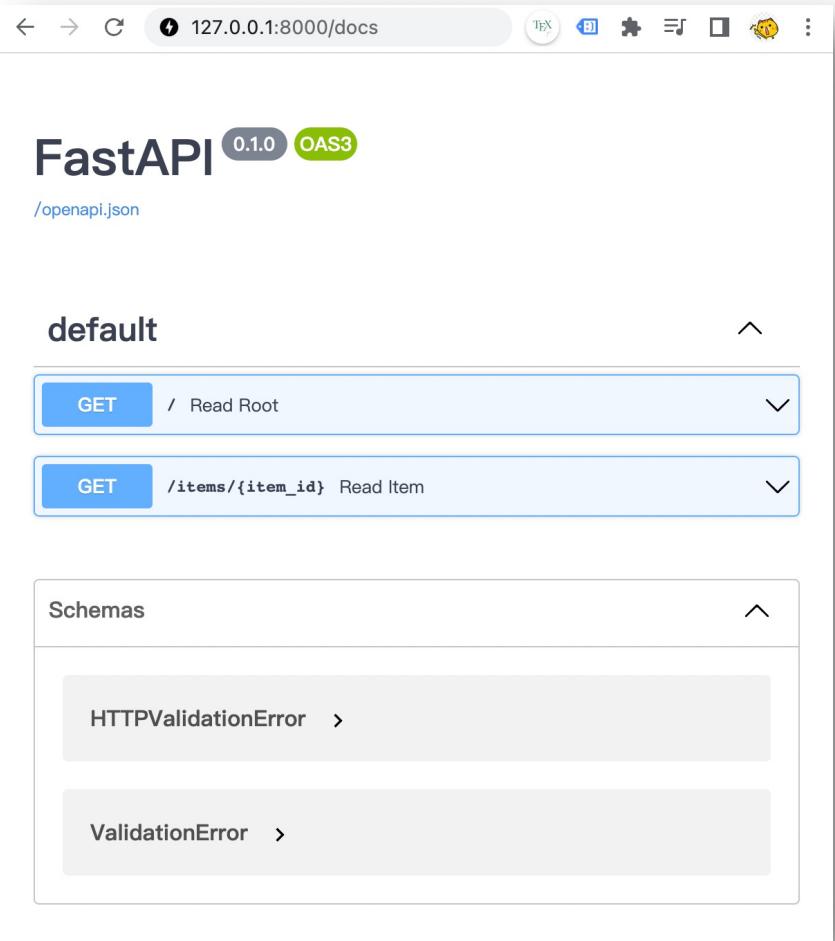
GET / Read Root ▾

GET /items/{item_id} Read Item ▾

Schemas ^

HTTPValidationError >

ValidationError >



Automatic interactive API documentation

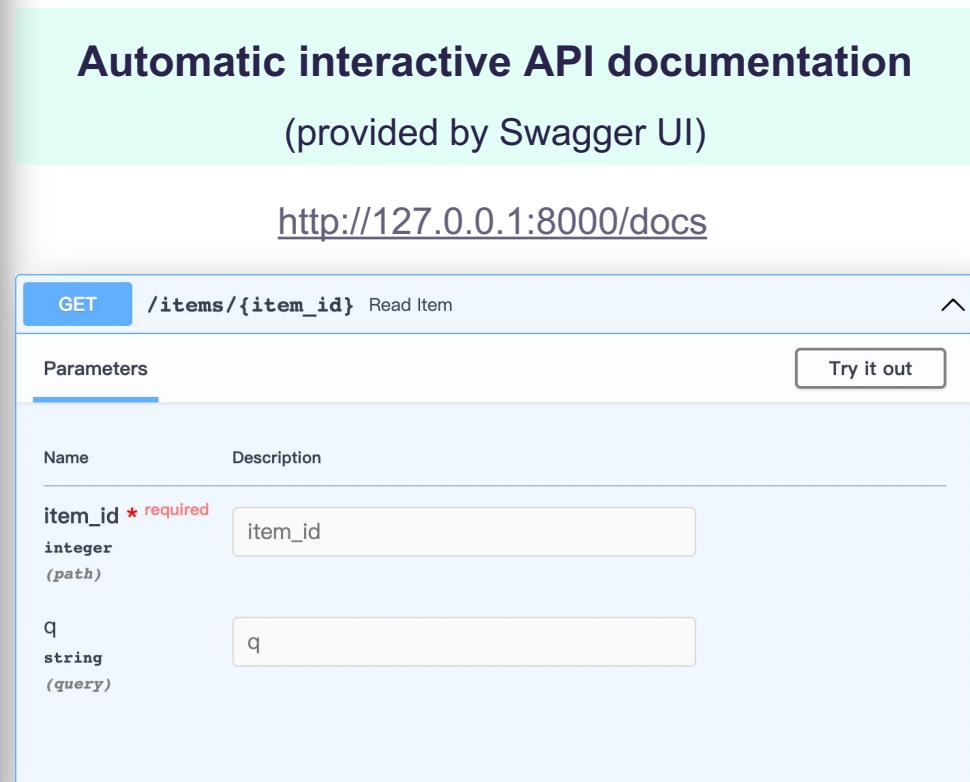
(provided by Swagger UI)

<http://127.0.0.1:8000/docs>

GET /items/{item_id} Read Item ^

Parameters Try it out

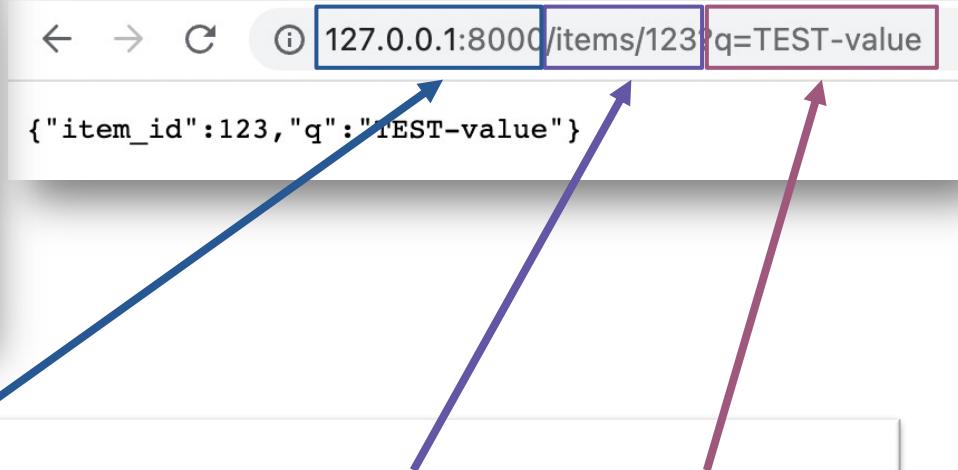
Name	Description
item_id * required integer (path)	<input type="text" value="item_id"/>
q string (query)	<input type="text" value="q"/>



GET /items/{item_id} Read Item

Parameters

Name	Description
item_id * required integer (path)	item_id 123
q string (query)	q TEST-value



http://www.example.com/photos?page=1

HTTP
通訊協定

DNS 網域名稱

Path
路徑

Parameter
參數

Resources:

<https://www.semrush.com/blog/url-parameters/>

Alternative automatic documentation (provided by ReDoc)

<http://127.0.0.1:8000/redoc>

The screenshot shows the ReDoc interface for a FastAPI application. On the left, there's a sidebar with a search bar and two main sections: "Read Root" and "Read Item".

Read Root

- Responses**
 - > 200 Successful Response

Read Item

- PATH PARAMETERS**
 - item_id (required) integer (Item Id)
- QUERY PARAMETERS**
 - q string (Q)

On the right, there are two detailed views of API endpoints:

- Response samples for / (GET)**: Shows a 200 response with content type application/json and a null value. The status code "200" is highlighted with a red box.
- Response samples for /items/{item_id} (GET)**: Shows both 200 and 422 responses with content type application/json and a null value. Both status codes "200" and "422" are highlighted with red boxes.

200
OK

422
Unprocessable Entity

3. Setup & connect with Google Sheets API

- 目的：將 Google Sheet 當作資料庫來使用。
- 步驟：
 1. 申請 [Google Sheet API](#) > 新增專案



3. Setup & connect with Google Sheets API

- 步驟：

2. 建立新專案

新增專案

⚠ 您的配額還可供建立 23 projects。建議您要求增加配額或刪除專案。[瞭解](#)
[詳情](#)

[MANAGE QUOTAS](#)

專案名稱 * —————

Python-FastAPI-GoogleSheet [?](#)

專案 ID：axiomatic-folio-359516。專案 ID 設定完成後即無法變更。 [編輯](#)

位置 * —————

無機構 [瀏覽](#)

上層機構或資料夾

[建立](#) [取消](#)

3. Setup & connect with Google Sheets API

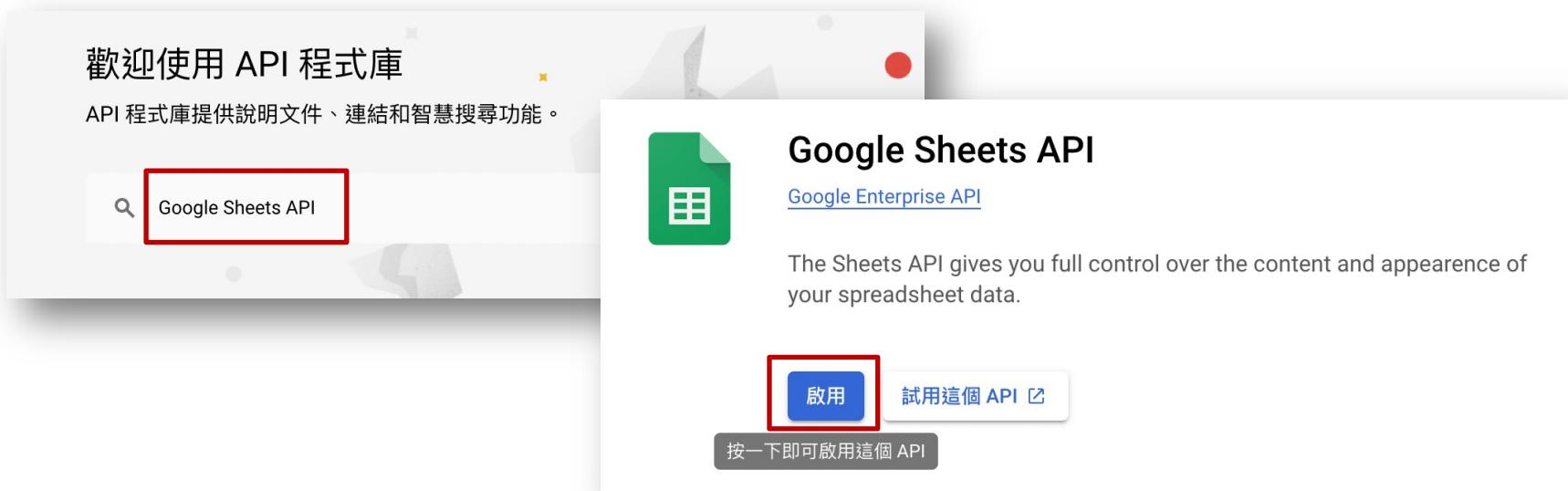
- 步驟：

3. 選取「Python-FastAPI-GoogleSheet」專案，並點選上方的「啟用 API 和服務」

The screenshot shows the Google Cloud Platform interface for managing APIs. On the left, there's a sidebar with various service categories like '程式庫', '憑證', 'OAuth 同意畫面', '網域驗證', and '頁面使用協議'. The main area is titled 'API 和服務' and shows a list of enabled APIs. A prominent red box highlights the '啟用 API 和服務' (Enable API and services) button at the top right of this section. Below this, there are two cards: '流量' (Traffic) and '錯誤' (Errors), both of which display a warning message: 'No data is available for the selected time frame.' The traffic chart has a scale from 0.05/s to 0.25/s. The error chart has a scale from 20% to 100%.

3. Setup & connect with Google Sheets API

- 步驟：
 4. 於搜尋框輸入「Google Sheets API」> 點選「啟用」



3. Setup & connect with Google Sheets API

- 步驟：

5. 啟用後，點選畫面左方的「憑證」，再點選上方「建立憑證(+CREATE CREDENTIALS)」>「服務帳戶」> 填寫「服務帳務詳細資料」

The screenshot shows the Google Cloud Platform interface for managing APIs and services. On the left, a sidebar lists various options: 已啟用的 API 和服務, 程式庫, 憑證 (selected), OAuth 同意畫面, 網域驗證, and 頁面使用協議. The main content area is titled 'API 和服務' and shows the '憑證' tab selected. A red box highlights the 'Python-FastAPI-GoogleSheet' project dropdown in the top navigation bar. Below it, a blue button labeled '+ CREATE CREDENTIALS' is also highlighted with a red box. A modal window is open, showing two options: 'API 金鑰' (API Key) and '服務帳戶' (Service Account). The '服務帳戶' option is highlighted with a red box. The modal also contains a warning message about OAuth user consent and a '請幫我選擇' (Help me choose) button.

3. Setup & connect with Google Sheets API

建立服務帳戶

1 服務帳戶詳細資料

服務帳戶名稱
admin-bessy

這個服務帳戶的顯示名稱

服務帳戶 ID *
admin-bessy

電子郵件地址 : admin-bessy@axiomatic-folio-359516.iam.gserviceaccount.com

服務帳戶說明

請描述這個服務帳戶的用途

建立並繼續

2 將專案存取權授予這個服務帳戶 (選用)

3 將這個服務帳戶的存取權授予使用者 (選用)

完成 取消

建立服務帳戶

1 服務帳戶詳細資料

2 將專案存取權授予這個服務帳戶 (選用)

3 將這個服務帳戶的存取權授予使用者 (選用)

允許不填寫

Grant access to users or groups that need to perform actions as this service account. [Learn more](#)

服務帳戶使用者角色

將透過這個服務帳戶部署工作和 VM 的權限授予使用者

服務帳戶管理員角色

將管理這個服務帳戶的權限授予使用者

完成 取消

29

3. Setup & connect with Google Sheets API

- 步驟：

- 點選服務帳戶的電子郵件 > 金鑰 > 建立新的金鑰

請「複製」電子郵件，後續會用到！

The screenshot shows the Google Cloud Platform Service Accounts interface. In the top navigation bar, there is a green button labeled "請「複製」電子郵件，後續會用到！". Below the navigation bar, the "admin-bessy" service account is selected. The "Keys" tab is active, showing two existing keys: "電子郵件" and "admin-bessy@admin-bessy.iam.gserviceaccount.com". A red box highlights the second key. A green arrow points from the text above to this key. Below the keys, there is a warning message about the security risks of key theft and a link to Workload Identity Federation. At the bottom, there are buttons for "新增金鑰" (Create New Key), "建立新的金鑰" (Create New Key), and "上傳現有金鑰" (Upload Existing Key). The "Create New Key" button is also highlighted with a red box.

3. Setup & connect with Google Sheets API

- 步驟：

7. 選擇 JSON 後按「建立」，將下載的 .json 存到專案的資料夾中



3. Setup & connect with Google Sheets API

- 步驟：

8. 請至 Google Drive 新增一張 google sheet，「共用人員」貼上剛剛複製的電子郵件，選擇「編輯者」> 傳送！



3. Setup & connect with Google Sheets API

```
# Install packages  
(MyVenv-name)$ pip install gspread oauth2client
```

GoogleSheet Key

The screenshot shows a Google Sheets interface. At the top, there is a dark header with the command `# Install packages` and `(MyVenv-name)$ pip install gspread oauth2client`. To the right of this, a green box contains the text "GoogleSheet Key". A green arrow points from this text towards the URL bar. The URL bar displays the URL `https://docs.google.com/spreadsheets/d/1PpPQVnKh4lwhivQ9sF95Lmk7PhcF8S19vDdUVLsk-Oc/edit#gid=0`, with the sheet ID part (`1PpPQVnKh4lwhivQ9sF95Lmk7PhcF8S19vDdUVLsk-Oc`) highlighted by a red box. Below the URL bar, the title bar shows the spreadsheet name "new-fastapi-example" and various menu options like 檔案, 編輯, 查看, 插入, 格式, 資料, 工具, 擴充功能, 說明. The main workspace shows a grid with columns A through I and rows 1 through 3. Cell A1 is currently selected.

3. Setup & connect with Google Sheets API

main.py

```
from oauth2client.service_account import ServiceAccountCredentials as SAC
import gspread

# ...

Json = "/Users/bessyhuang/Downloads/axiomatic-folio-359516-0f98f8adb09d.json"
Url = ["https://spreadsheets.google.com/feeds"]
Connect = SAC.from_json_keyfile_name(Json, Url)
GoogleSheets = gspread.authorize(Connect)
Sheet = GoogleSheets.open_by_key("1PpPQVnKh4IwhivQ9sF95Lmk7PhcF8S19vDdUVLsk-Oc")
Sheets = Sheet.sheet1

app = FastAPI()

# ...
```

GoogleSheet Key

3. Setup & connect with Google Sheets API

main.py

```
# ...

if __name__ == "__main__":
    dataTitle = ["name", "account", "password"]
    datas = ["Alex", "qaz", "111"]
    Sheets.append_row(dataTitle)
    Sheets.append_row(datas)
    print("寫入成功")
    print(Sheets.get_all_values())

uvicorn.run(app='main:app', host="127.0.0.1", port=8000, reload=True, debug=True)
```

3. Setup & connect with Google Sheets API

```
(MyVenv-name)$ python main.py
```

	A	B	C	D	E	F	
1	name	account	password				
2	Alex	qaz	111				
3							

4. Run a Web API with FastAPI & Google Sheet (Database)

[GET] Get all data



[GET] Get all data

main.py

```
from oauth2client.service_account import ServiceAccountCredentials as SAC
from fastapi import FastAPI
import uvicorn
import gspread

Json = "json-file-path"
Url = ["https://spreadsheets.google.com/feeds"]
Connect = SAC.from_json_keyfile_name(Json, Url)
GoogleSheets = gspread.authorize(Connect)
Sheet = GoogleSheets.open_by_key("google-sheet-key")
Sheets = Sheet.sheet1

# ...
```

[GET] Get all data

main.py

```
# ...

app = FastAPI()

@app.get("/users/")
async def getAllData():
    return Sheets.get_all_records()

if __name__ == "__main__":
    uvicorn.run(app='main:app', host="127.0.0.1", port=8000, reload=True, debug=True)
```

4. Run a Web API with FastAPI & Google Sheet (Database)

[GET] *Query data*



[GET] Query data

main.py

```
import re

# ...
app = FastAPI()

@app.get("/users/{userName}")
async def FindUsernameInfo(userName: str):
    userName_re = re.compile(r'^{}'.format(userName))
    cell = Sheets.find(userName_re)
    if cell == None:
        return "Username {} doesn't exist in Google Sheet.".format(userName)
    else:
        r = cell.row
        x = [item for item in Sheets.row_values(r) if item]
        return f"在第{cell.col}直行, 第{cell.row}橫列, {x}"
# ...
```

❖備註：僅查詢重複紀錄中的第一筆。

4. Run a Web API with FastAPI & Google Sheet (Database)

[POST] Insert data



[POST] Insert data

main.py

```
from pydantic import BaseModel
from typing import List

# ...
class Info(BaseModel):
    id: int
    data: List[str] = ["Bessy", "test", "321"]

app = FastAPI()

@app.post("/users/")
async def AddNewUsernameInfo(info: Info):
    Sheets.append_row(info.data)
    return {"status": "SUCCESS", "data": info}

# ...
```

4. Run a Web API with FastAPI & Google Sheet (Database)

[PUT] Update data



[PUT] Update data

main.py

```
from typing import Union

# ...
app = FastAPI()

@app.put("/users/{userName}")
async def UpdateExistUsernameInfo(userName: str, modified_account: Union[str, None] = None, modified_password: Union[str, None] = None):
    cell = Sheets.find(userName)
    if cell == None:
        return "Username {} doesn't exist in Google Sheet.".format(userName)
    else:
        r = cell.row
        Sheets.update(f"B{r}:C{r}", [[modified_account, modified_password]])
        x = [item for item in Sheets.row_values(r) if item]
        return {"msg": x}
# ...
```

❖備註：僅修改重複紀錄中的第一筆。

4. Run a Web API with FastAPI & Google Sheet (Database)

[DELETE] Delete data



[DELETE] Delete data

main.py

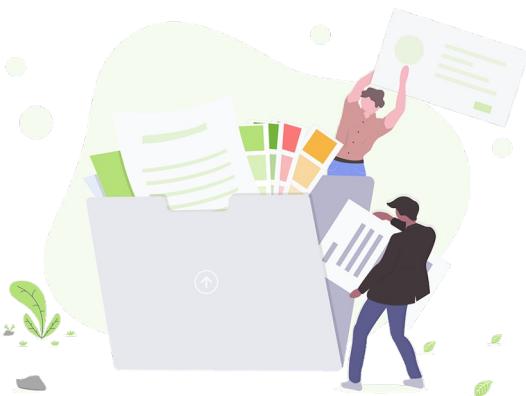
```
from typing import Union

# ...
app = FastAPI()

@app.delete("/users/{userName}")
async def DeleteExistUsernameInfo(userName: str):
    userName_re = re.compile(r'^{}'.format(userName))
    cell = Sheets.find(userName_re)
    if cell == None:
        return "Username {} doesn't exist in Google Sheet.".format(userName)
    else:
        r = cell.row
        Sheets.delete_row(r)
        return f"UserName {userName} 原本位於第 {cell.row} 橫列，現已刪除！"
# ...
```

❖備註：僅修改重複紀錄中的第一筆。

Reference



- [What is API: Definition, Types, Specifications, Documentation](#)
- [HTTP Tutorial](#)
- [Learn REST: A RESTful Tutorial](#)
- [什麼是真正的 RESTful API? #RESTful API應該長什麼樣子?](#)
- [API是什麼？有什麼應用？學會串接 Web API](#)
- [網路基礎概論 - HTTP 協定、TCP/IP、API](#)
- [FastAPI + Google Sheet \(Database\)](#)

Thanks!

Any questions?

You can find me at:

- @bessyhuang (GitHub)

