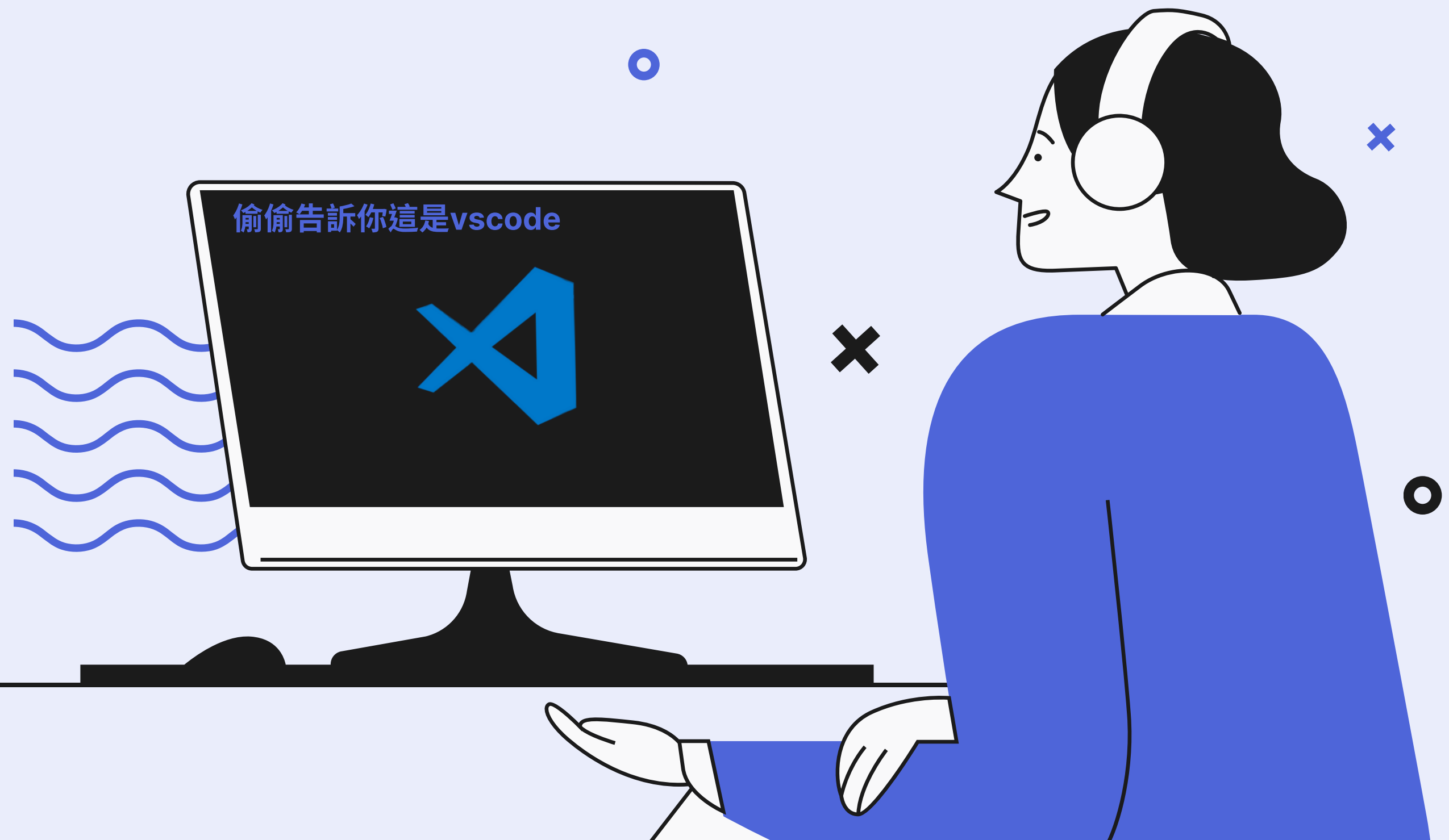


Python Flask

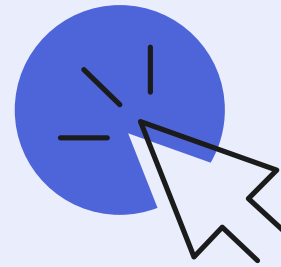
與

HTML



聽不懂想睡覺很正常XD

簡報內容



前情提要 and 補科
Python Flask
開始前的環境建置
部分code白話文解說
示範-Demo
心得感想與反思
Open Source-將程式碼開源
結尾



Q: 前後端資料互傳很重要?

A: 是，而且是非常重要的

**舉個例子:
平時逛購物網站、IG都要登入自己的帳號，那後端系統是如何驗證你輸入的帳號密碼是否正確?**



What is Python?

一種廣泛使用的直譯式、進階程式語言。因內建多函式庫且模組化，近年崛起成最受歡迎的程式語言。常被用於運算、開發、API撰寫等等。

What is HTML?

HyperText Markup Language
嚴格來說不算是程式語言，屬於
"超文本標記語言"，雖然不是程式
語言，但卻和coding有極大的關聯。

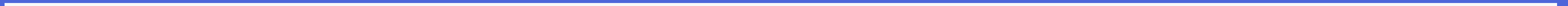
知道了Python 那什麼是Flask?

Python基於和前端溝通的簡易函式庫。Flask也被稱為「微框架」，因使用簡單的核心，能用擴充方式增加其他功能。

何謂「微框架」?(將於稍後介紹)



喇賽完ㄌ~進入主題



環境建置(1)

下載Python

WAY 1:瀏覽

<https://www.python.org/downloads/>

WAY 2:使用CMD(命令提示字元)打上:

winget install python

(如右圖，再打上"Y"即會進行自動下載)

```
命令提示字元 - winget install python
C:\Users\wang8>winget install python
找到 Python 3.10 [9PJW5LDXLZ5] 版本 Unknown
此套件是透過 Microsoft Store 所提供。winget 可能需要代表目前的使用者從 Microsoft Store 取得套件。
版本: Unknown
發行者: Python Software Foundation
發行者 URL: https://www.python.org/
發行者支援 URL: https://www.python.org/doc/
描述: Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

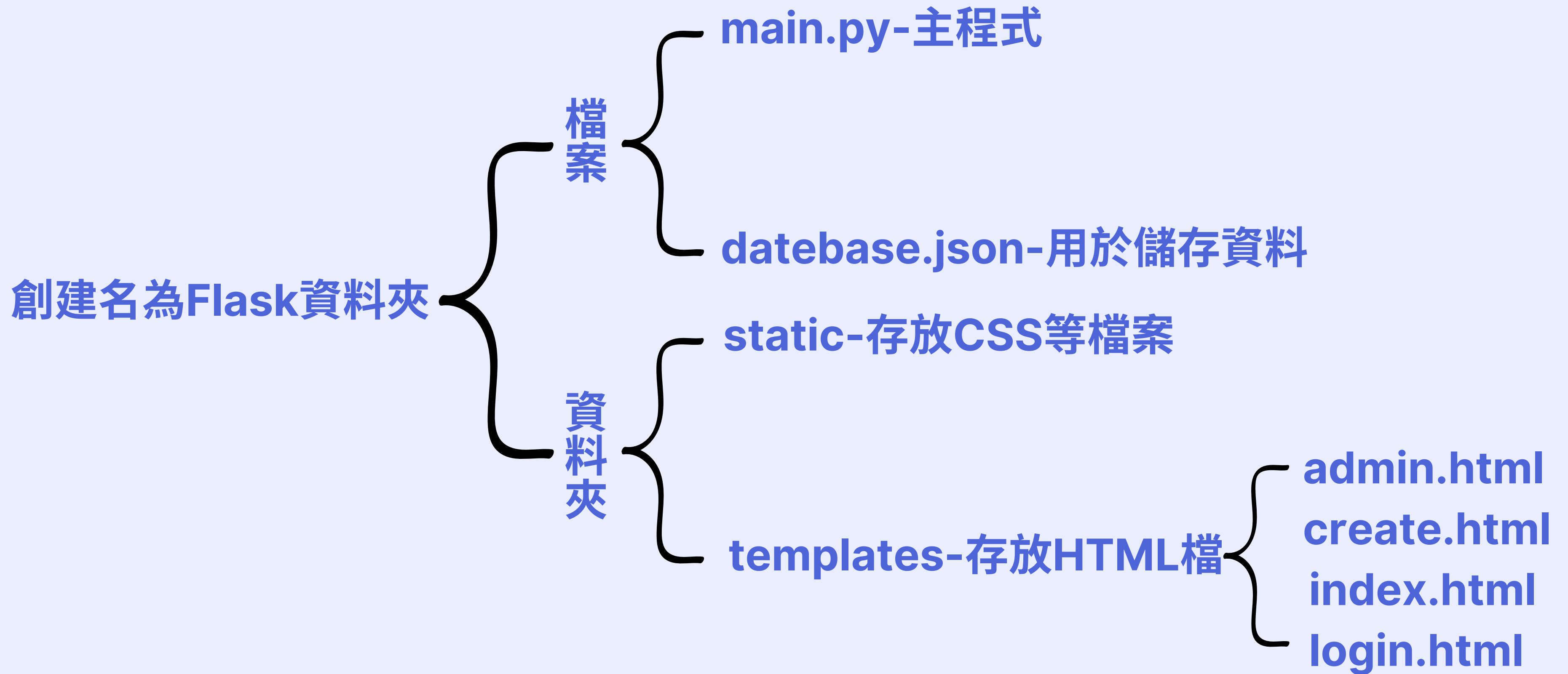
Expected your existing copy of Python to run? Either update your PATH to raise its priority, or open "Manage App Execution Aliases" from Start to disable the shortcuts.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.
授權: https://docs.python.org/3.10/license.html
隱私權 Url: https://www.python.org/privacy/
著作權: (c) Python Software Foundation
合約:
Category: Developer tools
Pricing: Free
Free Trial: No
Terms of Transaction: https://aka.ms/microsoft-store-terms-of-transaction
Seizure Warning: https://aka.ms/microsoft-store-seizure-warning
Store License Terms: https://aka.ms/microsoft-store-license

發行者要求您檢視上述資訊並接受合約，然後再安裝。
是否同意這些條款？
[Y] 是 [N] 否: y
```

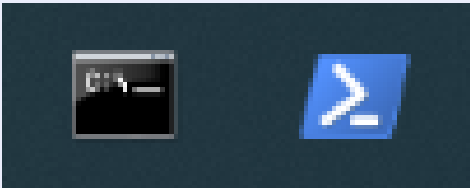
環境建置(2)



環境建置(3)

於CMD/PowerShell分別輸入以安裝套件

pip install Flask
pip install requests



```
PS C:\Users\wang8> pip install Flask
Collecting Flask
  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)
    ----- 101.5/101.5 kB 1.5 MB/s eta 0:00:00
Collecting Werkzeug>=2.2.2
  Downloading Werkzeug-2.2.2-py3-none-any.whl (232 kB)
    ----- 232.7/232.7 kB 1.6 MB/s eta 0:00:00
Requirement already satisfied: Jinja2>=3.0 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from Flask) (3.1.2)
Requirement already satisfied: click>=8.0 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from Flask) (8.1.3)
Requirement already satisfied: itsdangerous>=2.0 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from Flask) (2.1.2)
Requirement already satisfied: importlib-metadata>=3.6.0 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from Flask) (4.12.0)
Requirement already satisfied: colorama in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from click>=8.0->Flask) (0.4.5)
Requirement already satisfied: zipp>=0.5 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from importlib-metadata>=3.6.0->Flask) (3.8.1)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\wang8\appdata\local\programs\python\python38\lib\site-packages (from Jinja2>=3.0->Flask) (2.1.1)
Installing collected packages: Werkzeug, Flask
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 2.1.2
    Uninstalling Werkzeug-2.1.2:
      Successfully uninstalled Werkzeug-2.1.2
Successfully installed Flask-2.2.2 Werkzeug-2.2.2
```

Package	Version
aiohttp	3.8.3
aiosignal	1.2.0
altgraph	0.17.3
asciichartpy	1.5.25
async-timeout	4.0.2
attrs	21.2.0
certifi	2022.6.15
chardet	4.0.0
charset-normalizer	2.1.0
click	8.1.3
colorama	0.4.5
Flask	2.1.3
Flask-SQLAlchemy	2.5.1
frozenset	1.3.1
future	0.18.2
greenlet	1.1.2
idna	3.3
importlib-metadata	4.12.0
itsdangerous	2.1.2
Jinja2	3.1.2
line-bot-sdk	1.18.0
MarkupSafe	2.1.1
multidict	5.2.0
nextcord	2.2.0
pefile	2022.5.30
pip	22.3.1
psutil	5.9.1
psycpg2	2.9.3
pyinstaller	5.6.2
pyinstaller-hooks-contrib	2022.12
pywin32-ctypes	0.2.0
requests	2.28.1
setuptools	41.2.0
SQLAlchemy	1.4.39
typing_extensions	4.4.0
urllib3	1.26.10
Werkzeug	2.1.2
yaml	1.7.0
you-get	0.4.1620
zipp	3.8.1

安裝完後可使用**pip list**查看是否安裝成功以及安裝版本-->

環境建置(4)

使用import導入相關套件
(Flask/main.py)

```
1 from flask import Flask
2 from flask import request
3 from flask import render_template
4
5 import datetime
6 import json
```

於json寫入標籤，值為保留
(Flask/database.json)

```
1 {
2     "username": "",
3     "password": ""
4 }
```



正式開始 勿前情提要



- 1.本程式碼僅針對前後端回傳進行研究和實作，並無針對網頁美化、預防資安事件(例XSS、injection...等)。
- 2.本程式碼為求簡單使用Json作為資料儲存的地方
並非常見的SQL(資料庫)。



微框架？ 有多微？

啟動環境只需要7行!!
(Flask/main.py)

```
1  from flask import Flask
2  app = Flask(__name__)
3  @app.route("/")
4  def index():
5      return ""
6  if __name__ == "__main__":
7      app.run(host="127.0.0.1", port=5000, debug=True)
```

```
* Serving Flask app 'file' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 312-902-132
```

<--成功啟動
提示說僅是測試人員環境

此時可用任意browser訪問localhost:5000

先從創建帳號開始(白話文翻譯版)!

1~4:導入相關套件

6:設定URL路徑為/create，資料傳輸方式使用**GET**和**POST**

8:如果傳輸方式為**POST**

9~14:宣告username和password的變數並用於接收前端回傳的資訊

16~18:開啟Json檔且將前端回傳後端資料以**UTF-8**寫入

19~27:檔案開啟時將輸入框回傳至前端頁面顯示

```
1  from flask import Flask
2  from flask import request
3  import json
4  app = Flask(__name__)
5
6  @app.route("/create", methods = ['GET', 'POST'])
7  def create():
8      if request.method == 'POST':
9          create_user = request.values["createuser"]
10         create_pw = request.values["createpw"]
11         create = {
12             "username" : create_user,
13             "password" : create_pw
14         }
15
16         with open('database.json', mode = 'w', encoding = 'utf-8') as database:
17             json.dump(create, database, indent=4)
18             return "帳號創建完畢!"
19     else:
20         return ''
21     <form method="post" action="/create">
22         <p>創建帳號</p>
23         <input type="text" name="createuser">
24         <p>創建密碼</p>
25         <input type="text" name="createpw">
26         <input type="submit" name="submit" value="send">
27     </form>
28     ...
```

備註:標記紅色為網概內容

登入帳號(白話文翻譯版)!

1~5:導入相關套件

7:設定URL路徑為/login，資料傳輸方式使用**GET**和**POST**

9:如果傳輸方式為**POST**

10~11:宣告name和pw變數並用於接收前端回傳的資訊

12~13:開啟Json檔且將前端回傳後端資料以**UTF-8**讀出

14~15:如果前端輸入的帳號和密碼"完全"等於Json檔裡的值的話進入admin.html，如不是則顯示"帳號或密碼錯誤!"

19~25:檔案開啟時將輸入框回傳至前端頁面顯示

```
1  from flask import Flask
2  from flask import request
3  from flask import render_template
4  import json
5  app = Flask(__name__)
6
7  @app.route("/login", methods = ['GET', 'POST'])
8  def login():
9      if request.method == 'POST':
10         name = request.values["name"]
11         pw = request.values["pw"]
12         with open('database.json', mode = 'r', encoding = 'utf-8') as database:
13             date = json.load(database)
14             if name == (date["username"]) and pw == (date["password"]):
15                 return render_template("admin.html")
16             else:
17                 return "帳號或密碼錯誤!"
18         return ''
19     <form method="post" action="/login">
20         <p>登入帳號</p>
21         <input type="text" name="name">
22         <p>登入密碼</p>
23         <input type="text" name="pw">
24         <input type="submit" name="submit" value="send">
25     </form>''
```

備註:標記紅色為網概內容



示範



創建帳號

(Flask/database.json)

```
1 {  
2   "username": "",  
3   "password": "",  
4 }
```

確認資料為空值

進入創建帳號頁面輸入資料

(Flask/templates/create.html)

創建帳號

創建密碼

輸入完畢按下送出

(Flask/database.json)

```
1 {  
2   "username": "admin",  
3   "password": "test",  
4 }
```

回去查看資料是否被寫入

帳號創建完畢!

登入帳號(1)

(Flask/templates/login.html)

(Flask/database.json)

```
1 {  
2   "username": "admin",  
3   "password": "test"  
4 }
```

確認資料是否已被創建

輸入錯誤帳號或密碼

(Flask/templates/login.html)

登入帳號

admin

登入密碼

1111

send

(Flask/templates/login.html)

帳號或密碼錯誤！

回傳對應結果(登入失敗)

登入帳號(2)

(Flask/database.json)

```
1 {  
2   "username": "admin",  
3   "password": "test"  
4 }
```

確認資料是否已被創建

輸入正確帳號和密碼

(Flask/templates/login.html)

登入帳號

登入密碼

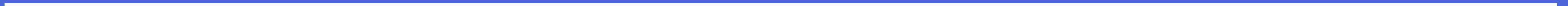
(Flask/templates/admin.html)

Hello

回傳對應結果(成功登入)



心得感想與反思



心得感想~~

雖然接觸Python已經長達一段時間了，但嚴格來說，這是我第一份正式完成並公開的Python程式專案，完成時的心情只有感動和歡喜，因為這代表我在程式設計的路上成功了第一步，也給未來的我一個方向和走下去的動力。

補充一下：

Q:為何接觸那麼久才完成一份專案？

A:因為之前都把時間花在學網概和HTML等其他東西

反思和待改進~~

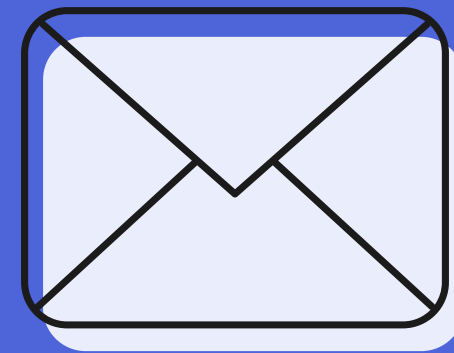
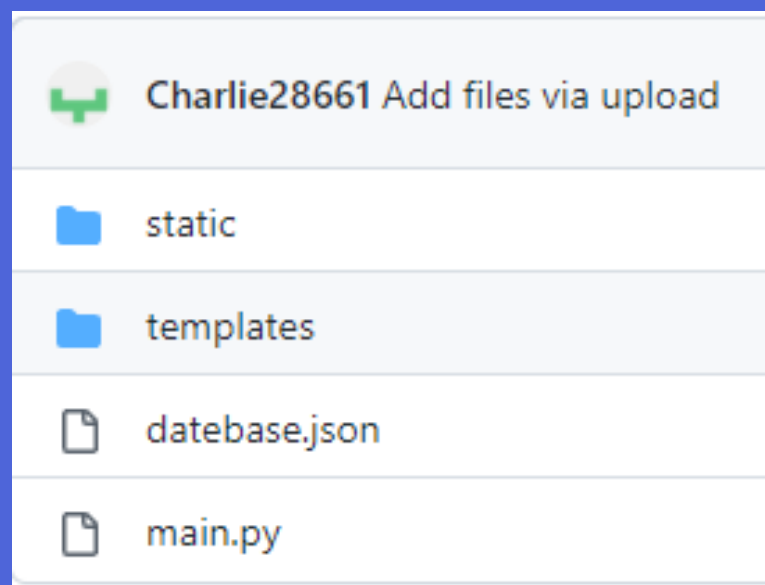
1.程式題目決定過久
(造成壓縮到編程的時間)

2.缺乏經驗
(許多語法幾乎都是陌生的，需要花時間學習)

Ex.全域變數、區域變數的使用時機

相關連結、 意見回饋、 Q&A問答?

Github開源專案(Open Source):



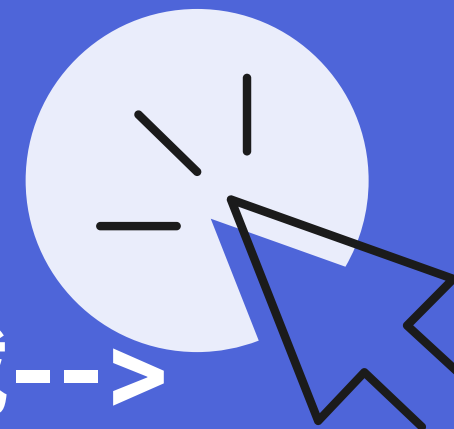
Email

service@ncves.com
u111004@nhes.edu.tw



Website(自我介紹網站)

ncves.com



NCVES資料下載網站

download.ncves.com

如果對這份PPT有興趣，歡迎至此下載-->