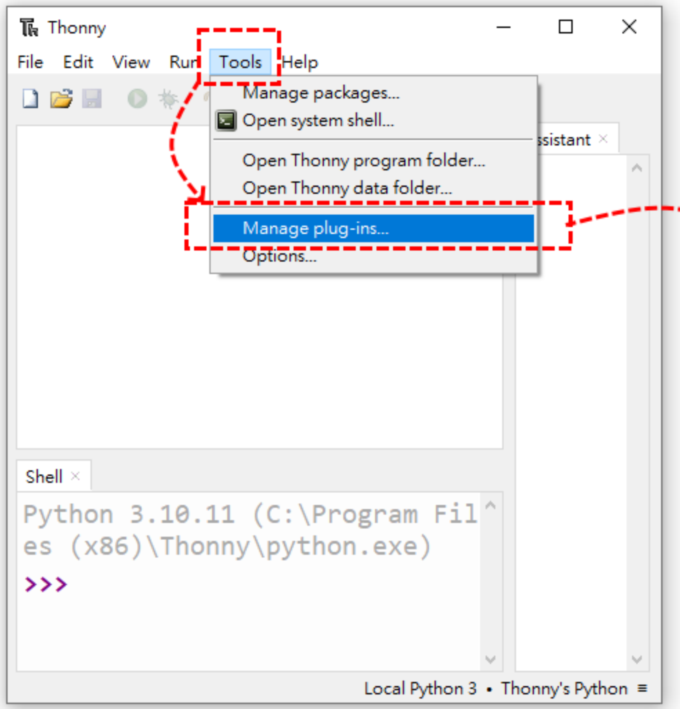
Flask Server App Example

1.Install Python IDE--Thonny

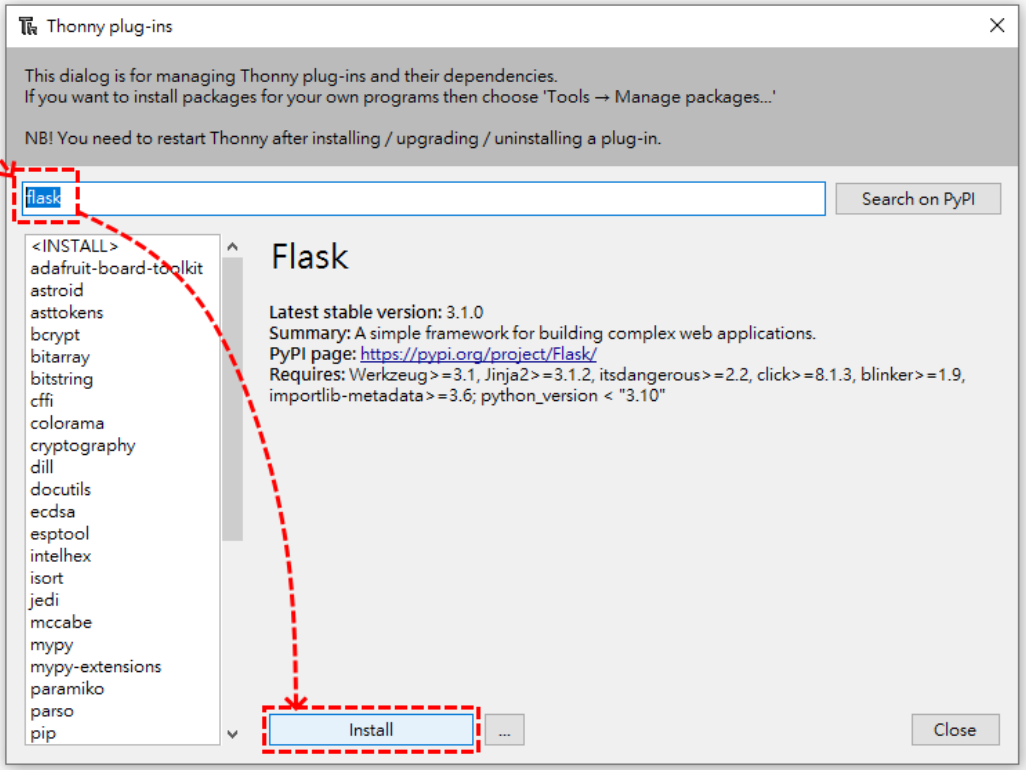
Not necessarily thonny, you can choose other familiar IDE for you.

IDE Download: https://thonny.org/

2.Install Flask



```
Python 3.10.11 (C:\Program Files (x86)\Thonny\python.exe)
>>>
```



This dialog is for managing Thonny plug-ins and their dependencies.
If you want to install packages for your own programs then choose 'Tools → Manage packages...'

NB! You need to restart Thonny after installing / upgrading / uninstalling a plug-in.

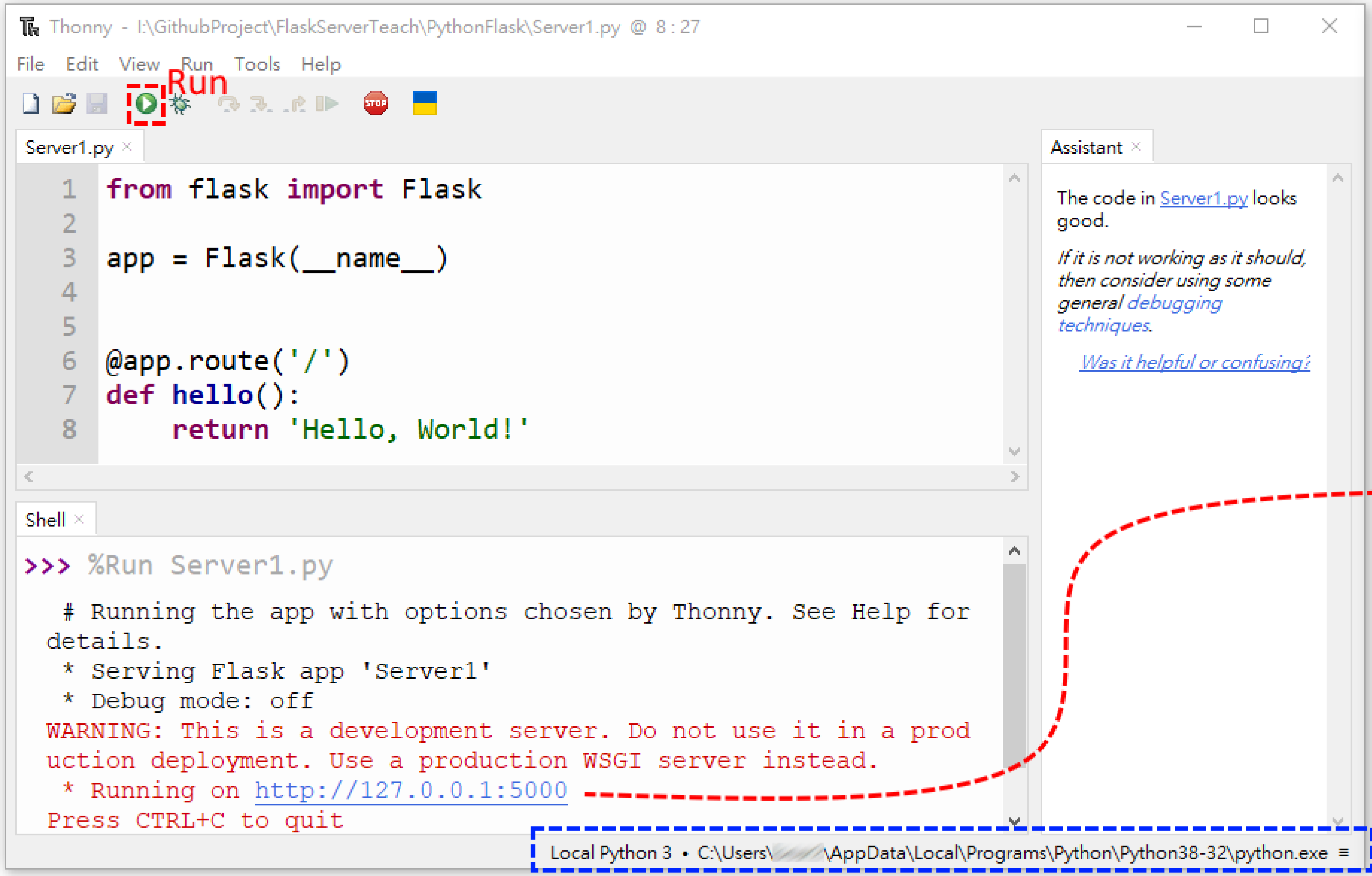
<INSTALL>
adafruit-board-toolkit
astroid
asttokens
bcrypt
bitarray
bitstring
cffi
colorama
cryptography
dill
docutils
ecdsa
esptool
intelhex
isort
jedi
mccabe
mypy
mypy-extensions
paramiko
parso
pip

Flask

Latest stable version: 3.1.0
Summary: A simple framework for building complex web applications.
PyPI page: https://pypi.org/project/Flask/
Requires: Werkzeug>=3.1, Jinja2>=3.1.2, itsdangerous>=2.2, click>=8.1.3, blinker>=1.9, importlib-metadata>=3.6; python_version < "3.10"

3.Write Code: Server1.py and Run

Library: https://flask.palletsprojects.com/en/stable/tutorial/layout/


Thonny - I:\GithubProject\FlaskServerTeach\PythonFlask\Server1.py @ 8 : 27
File Edit View Run Tools Help
Run
Server1.py
Assistant
Shell

```python
from flask import Flask

app = Flask(__name__)


@app.route('/')
def hello():
    return 'Hello, World!'
```

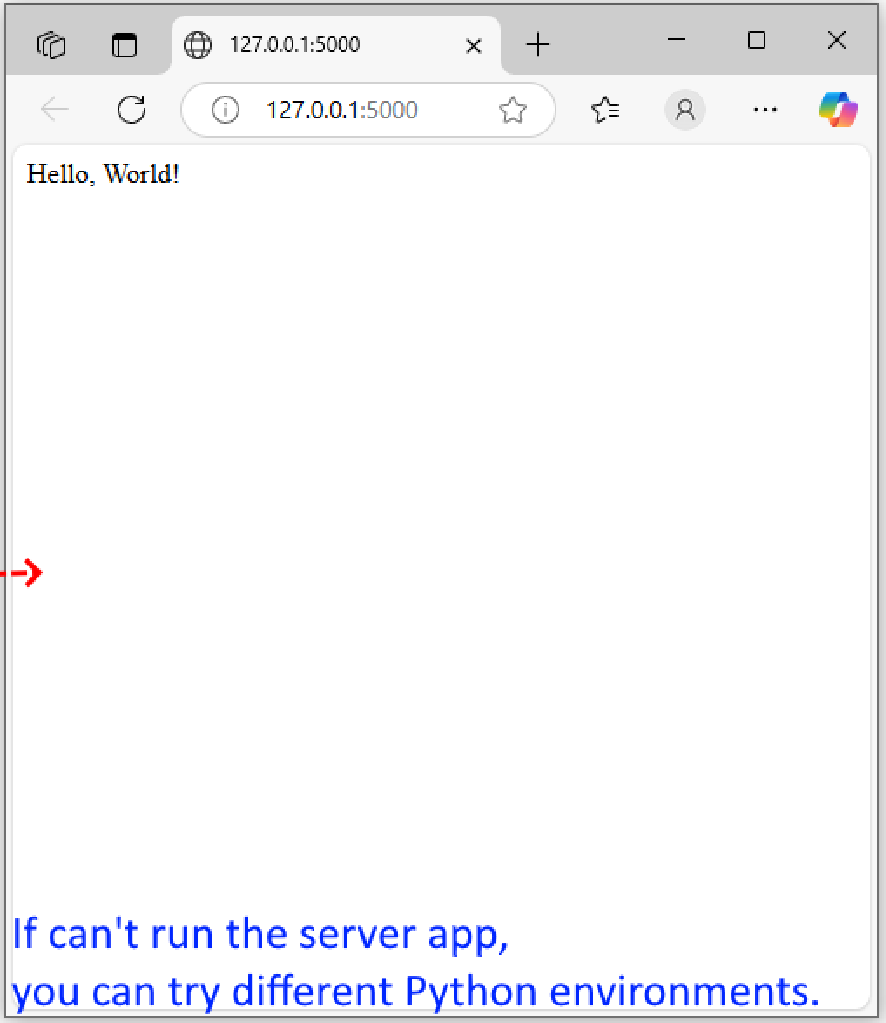
The code in Server1.py looks good.

If it is not working as it should, then consider using some general debugging techniques.

Was it helpful or confusing?

```
>>> %Run Server1.py

 # Running the app with options chosen by Thonny. See Help for
details.
 * Serving Flask app 'Server1'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a prod
uction deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

Local Python 3 • C:\Users\[redacted]\AppData\Local\Programs\Python\Python38-32\python.exe ≡



Hello, World!

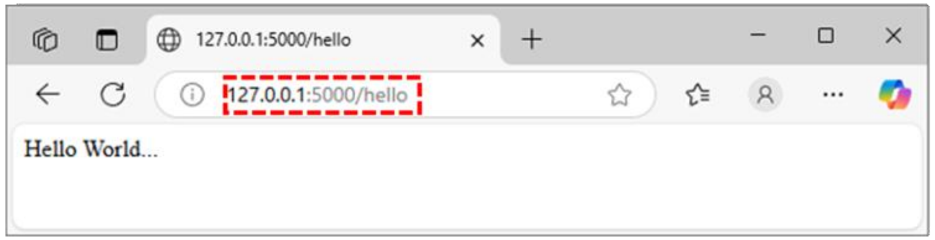
If can't run the server app,
you can try different Python environments.

4.Write Code: Get Method

Library: form AI

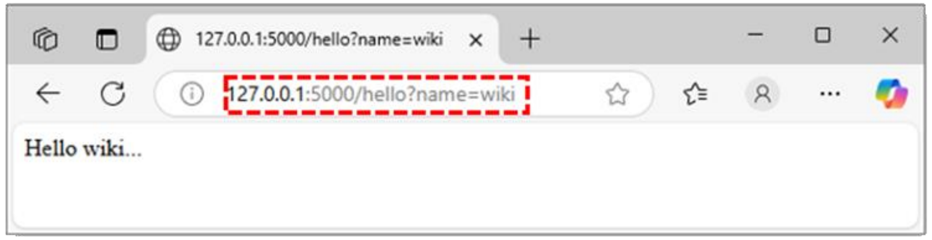
```
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/')
def hello():
    return 'Hello World!'

# GET Method
@app.route('/hello', methods=['GET'])
def hello_get():
    name = request.args.get('name', 'World') #'World' is default
    password = request.args.get('password', 'None') #‘None' is default
    if password=='None':
        return f'Hello {name}...'
    else:
        if name=='wiki' and password=='123456':
            return f'Hello {name}, welcome to system.'
        elif name=='chen' and password=='456789':
            return f'Hello {name}, welcome to system.'
        else:
            return f'password error!!!'
```

Web Output


127.0.0.1:5000/hello

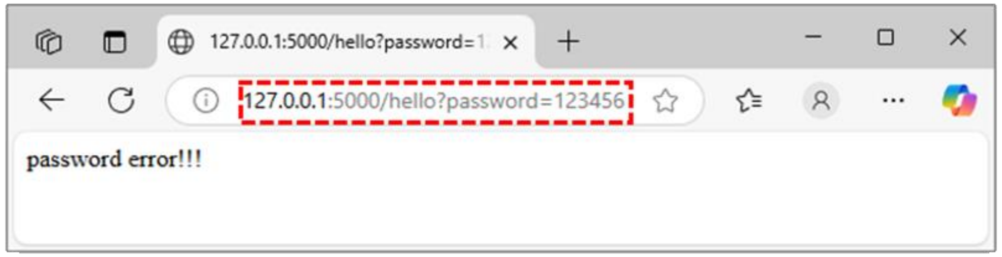
127.0.0.1:5000/hello

Hello World...


127.0.0.1:5000/hello?name=wiki

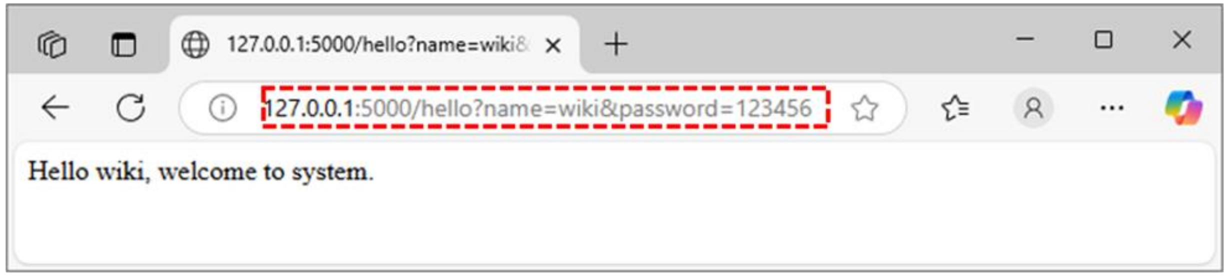
127.0.0.1:5000/hello?name=wiki

Hello wiki...


127.0.0.1:5000/hello?password=1

127.0.0.1:5000/hello?password=123456

password error!!!


127.0.0.1:5000/hello?name=wiki&

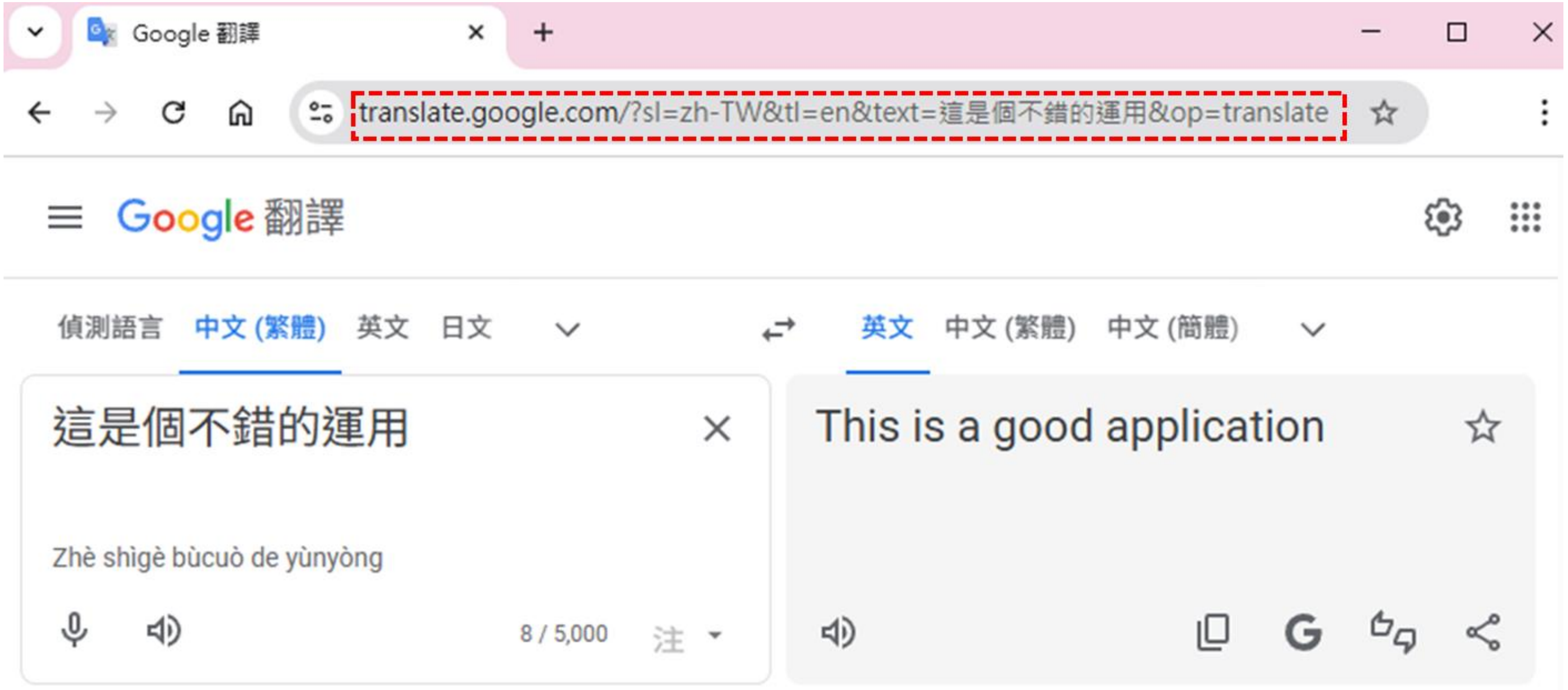
127.0.0.1:5000/hello?name=wiki&password=123456

Hello wiki, welcome to system.

This a fast network transmission method. If you have passwords or personal information, please don't use this method. So this example is a bad application.

4.Write Code: Get Method

This is a good application.



Google 翻譯

translate.google.com/?sl=zh-TW&tl=en&text=這是個不錯的運用&op=translate

Google 翻譯

這是個不錯的運用

Zhè shìgè bùcuò de yùnyòng

8 / 5,000

This is a good application

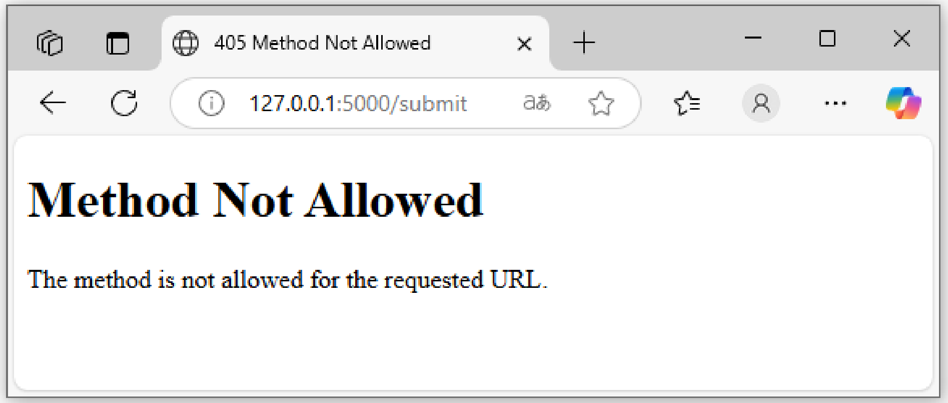
5.Write Code: Post Method

Library: form AI

```
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/')
def hello():
   return 'Hello World!'

# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
   data = request.get_json()
   name = f"{data.get('name')}"
   password = f"{data.get('password')}"
   return jsonify(message=f'Your name is {name},password is
{password}')
```

```
fetch('http://127.0.0.1:5000/submit', {  method: 'POST',
headers: {    'Content-Type': 'application/json'  },  body:
JSON.stringify({ name: 'wiki', password:
'123456' })}).then(response => response.json()).then(data =>
console.log(data)).catch(error => console.error('Error:', error));
```

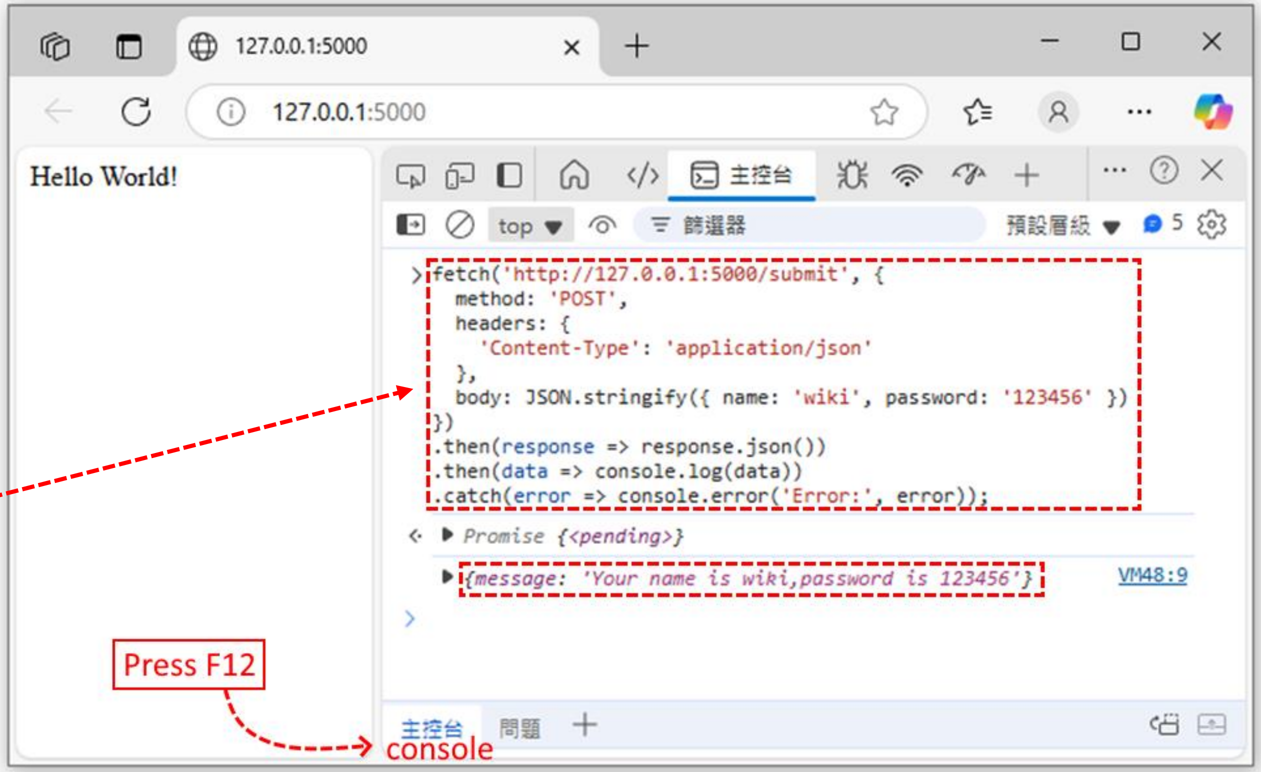
Web Output



Method Not Allowed

The method is not allowed for the requested URL.

Web Console Output

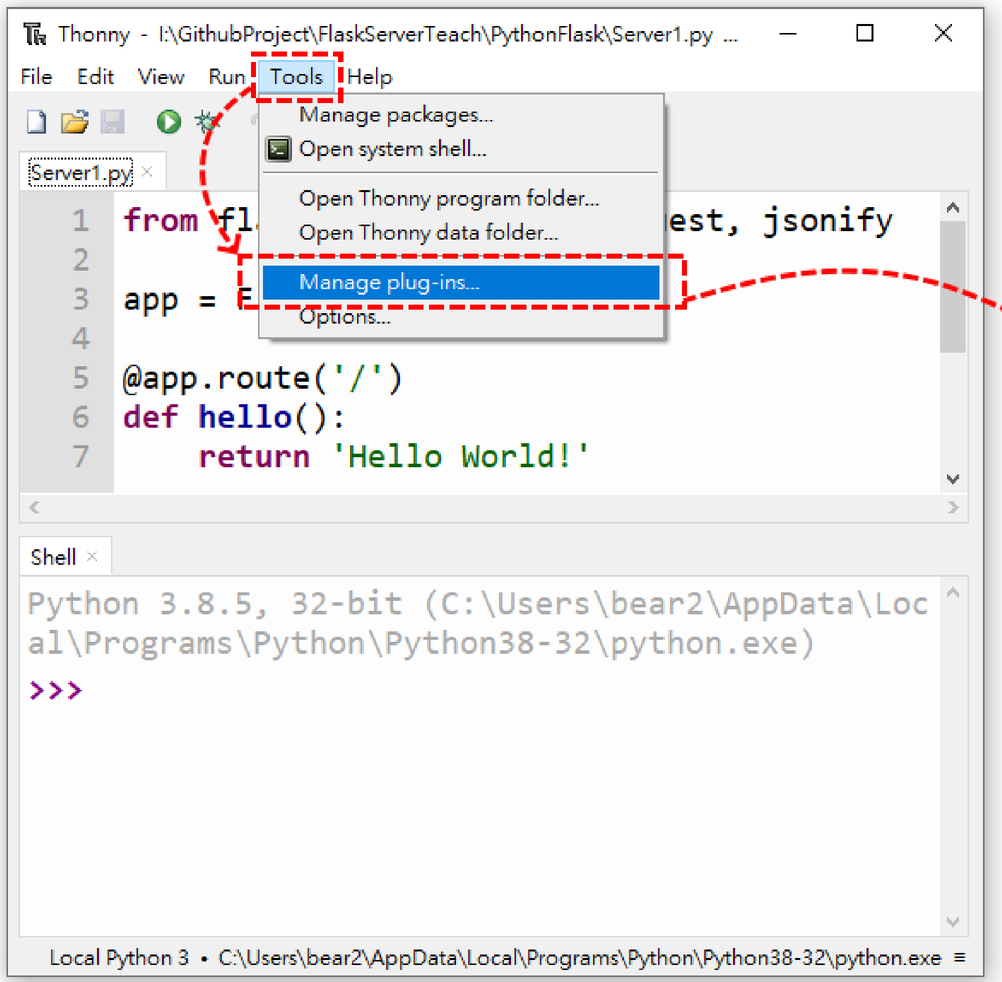


Hello World!

```
>fetch('http://127.0.0.1:5000/submit', {
   method: 'POST',
   headers: {
     'Content-Type': 'application/json'
   },
   body: JSON.stringify({ name: 'wiki', password: '123456' })
 })
 .then(response => response.json())
 .then(data => console.log(data))
 .catch(error => console.error('Error:', error));
<· ▶ Promise {<pending>}
   ▶ {message: 'Your name is wiki,password is 123456'}          VM48:9
>
```

6.Write Code: Post Method Test

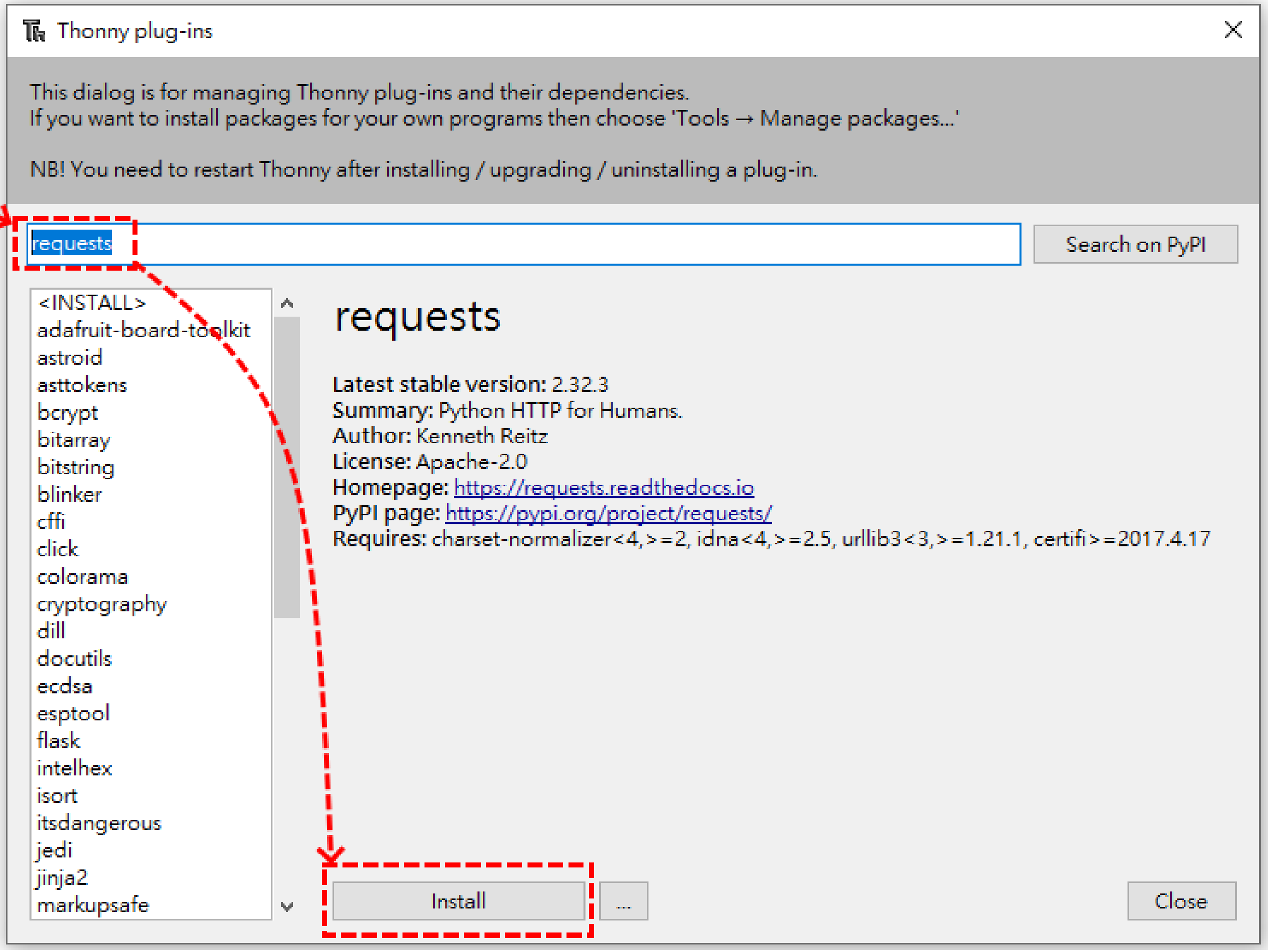
Step1:install requests


Thonny - I:\GithubProject\FlaskServerTeach\PythonFlask\Server1.py ...
File
Edit
View
Run
Tools
Help
Manage packages...
Open system shell...
Open Thonny program folder...
Open Thonny data folder...
Manage plug-ins...
Options...
Server1.py
Shell

```
from fl                                  est, jsonify

app = F

@app.route('/')
def hello():
    return 'Hello World!'
```

```
Python 3.8.5, 32-bit (C:\Users\bear2\AppData\Local\Programs\Python\Python38-32\python.exe)
>>>
```

Local Python 3 • C:\Users\bear2\AppData\Local\Programs\Python\Python38-32\python.exe



This dialog is for managing Thonny plug-ins and their dependencies.
If you want to install packages for your own programs then choose 'Tools → Manage packages...'

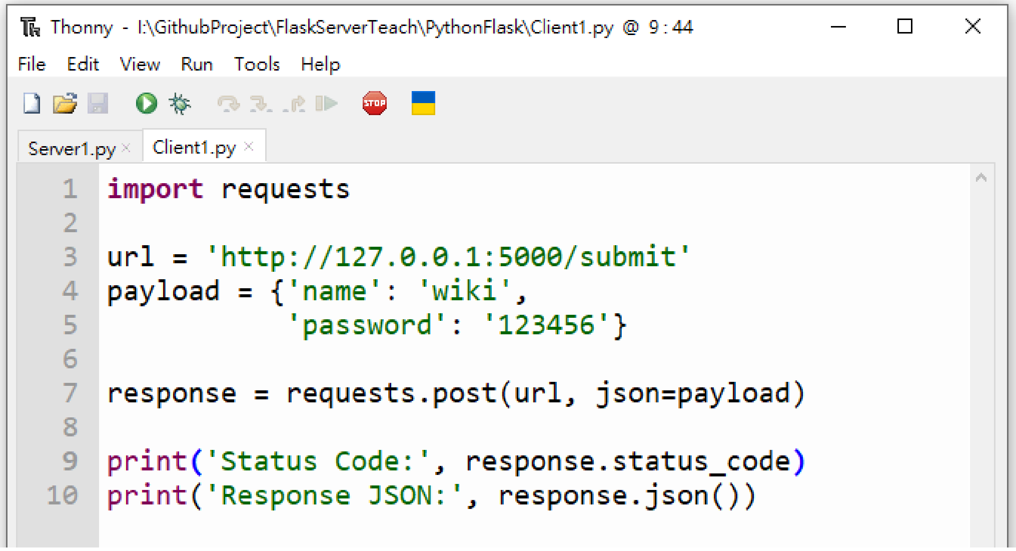
NB! You need to restart Thonny after installing / upgrading / uninstalling a plug-in.

requests

Latest stable version: 2.32.3
Summary: Python HTTP for Humans.
Author: Kenneth Reitz
License: Apache-2.0
Homepage: https://requests.readthedocs.io
PyPI page: https://pypi.org/project/requests/
Requires: charset-normalizer<4,>=2, idna<4,>=2.5, urllib3<3,>=1.21.1, certifi>=2017.4.17

6.Write Code: Post Method Test

Step2: Write Code on Client1.py

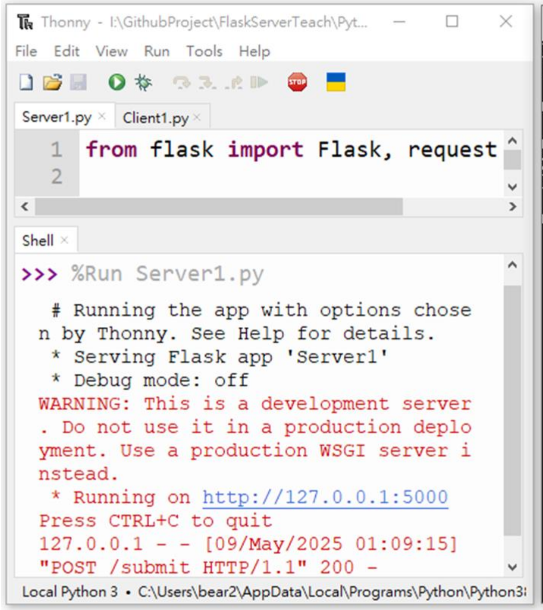

Thonny - I:\GithubProject\FlaskServerTeach\PythonFlask\Client1.py @ 9 : 44
File Edit View Run Tools Help
Server1.py
Client1.py

```python
import requests

url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
           'password': '123456'}

response = requests.post(url, json=payload)

print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

```python
import requests

url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
        'password': '123456'}

response = requests.post(url, json=payload)

print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

Step3: Run Server1.py and Client1.py

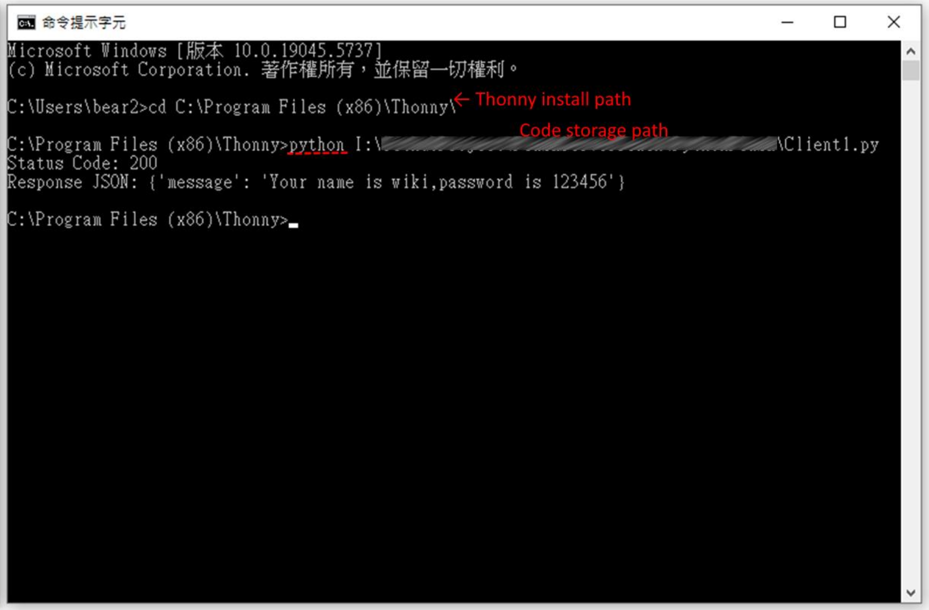
Server1.py Run on IDE Shell.


Thonny - I:\GithubProject\FlaskServerTeach\Pyt...
File Edit View Run Tools Help
Server1.py
Client1.py
Shell
Local Python 3 • C:\Users\bear2\AppData\Local\Programs\Python\Python3

```python
from flask import Flask, request
```

```
>>> %Run Server1.py

  # Running the app with options chose
 n by Thonny. See Help for details.
  * Serving Flask app 'Server1'
  * Debug mode: off
 WARNING: This is a development server
 . Do not use it in a production deplo
 yment. Use a production WSGI server i
 nstead.
  * Running on http://127.0.0.1:5000
 Press CTRL+C to quit
 127.0.0.1 - - [09/May/2025 01:09:15]
 "POST /submit HTTP/1.1" 200 -
```

Client1.py Run on Terminal.



```
Microsoft Windows [版本 10.0.19045.5737]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\bear2>cd C:\Program Files (x86)\Thonny\

C:\Program Files (x86)\Thonny>python I:\[redacted]\Client1.py
Status Code: 200
Response JSON: {'message': 'Your name is wiki,password is 123456'}

C:\Program Files (x86)\Thonny>_
```

Client1.py is run on terminal, because Thonny IDE can’t run two code at same time.

If Client1.py can't run, you may need to change the Python environment.

7.Write Code: Post Method Test

Step1: Write Code on Server1.py

```
from flask import Flask, request, jsonify
from datetime import datetime

app = Flask(__name__)

@app.route('/')
def hello():
   return 'Hello World!'

# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
   data = request.get_json()
   name = data.get('name')
   password = data.get('password')
   birthday = datetime.strptime((f"{data.get('birthday')}"), "%Y-%m-%d")
   gender = data.get('gender')
   height = data.get('height')
   weight = data.get('weight')

   # bmi count
   bmi = round(weight / height **2,2)

   # age count
   today = datetime.today()
   age = today.year - birthday.year - ((today.month, today.day) < (birthday.month, birthday.day))

   return jsonify(message=f'Your AGE={age},BMI={bmi}')
```

Step2: Write Code on Client1.py

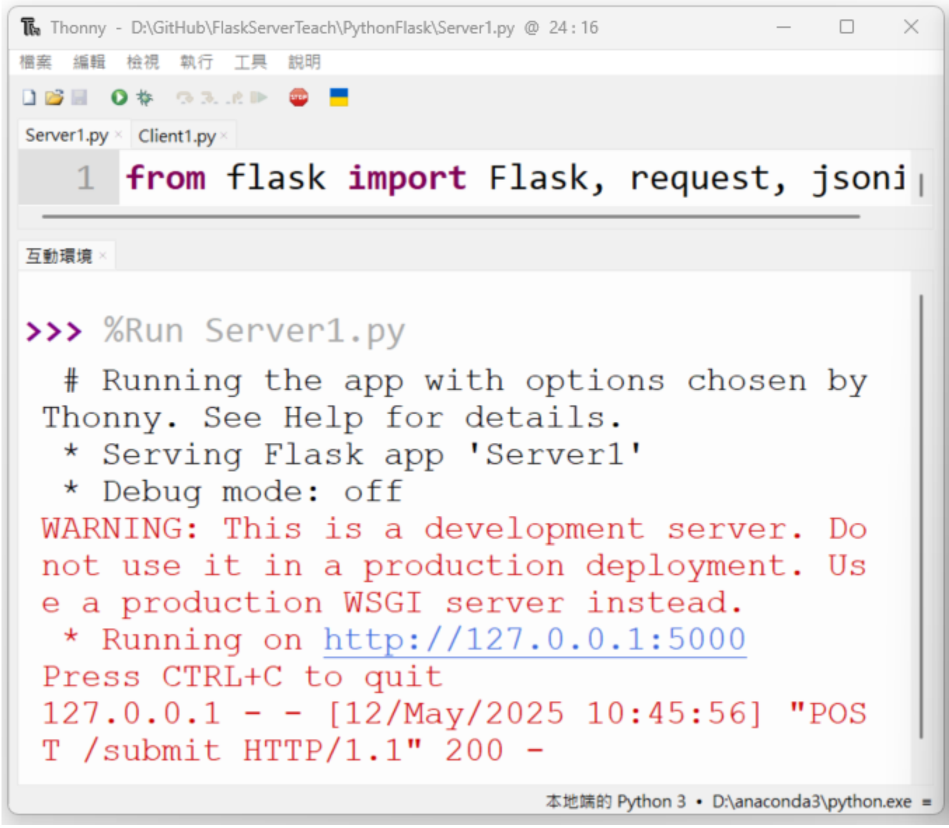
```
import requests

url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
        'password': '123456',
        'birthday':'1990-04-24',
        'height':1.75,
        'weight':60}

response = requests.post(url, json=payload)

print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

7.Write Code: Post Method Test

Step3 Run Server1.py and Client1.py

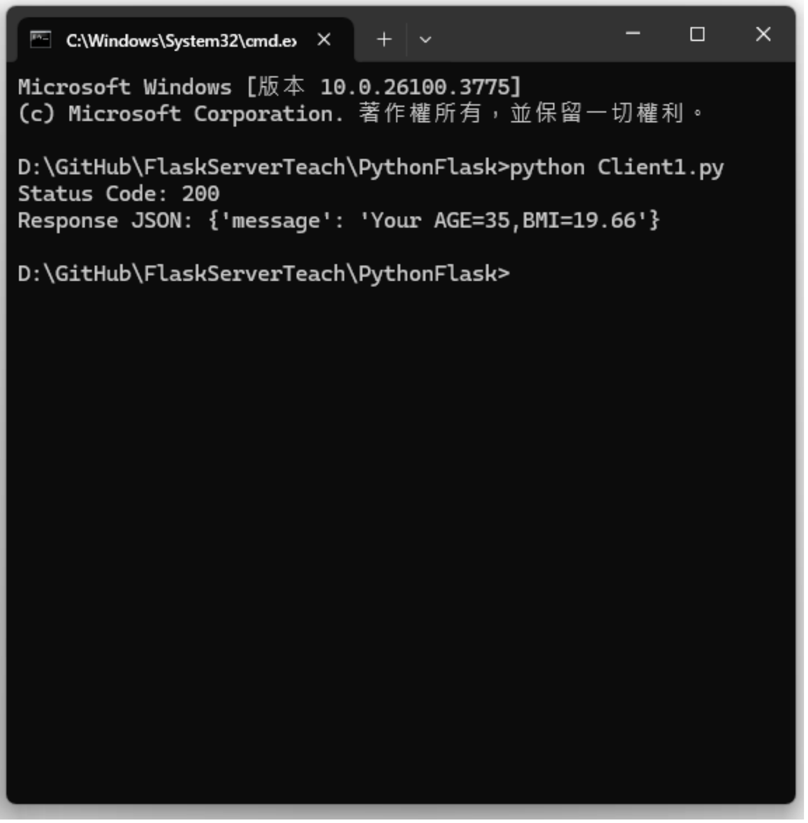
Server1.py Run on IDE Shell.

Client1.py Run on Terminal.


Thonny - D:\GitHub\FlaskServerTeach\PythonFlask\Server1.py @ 24 : 16
檔案 編輯 檢視 執行 工具 說明
Server1.py
Client1.py
互動環境
本地端的 Python 3 • D:\anaconda3\python.exe

```
from flask import Flask, request, jsoni
```

```
>>> %Run Server1.py
 # Running the app with options chosen by
Thonny. See Help for details.
 * Serving Flask app 'Server1'
 * Debug mode: off
WARNING: This is a development server. Do
not use it in a production deployment. Us
e a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [12/May/2025 10:45:56] "POS
T /submit HTTP/1.1" 200 -
```



```
Microsoft Windows [版本 10.0.26100.3775]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=19.66'}

D:\GitHub\FlaskServerTeach\PythonFlask>
```

Client1.py is run on terminal, because Thonny IDE can’t run two code at same time.

If Client1.py can't run, you may need to change the Python environment.

8.Write Code: Post Method Test

Step1: Write Code on Server1.py

```
from flask import Flask, request, jsonify
from datetime import datetime
import math
import pandas as pd
app = Flask(__name__)
# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
    data = request.get_json()
    name = data.get('name')
    password = data.get('password')
    birthday = datetime.strptime((f"{data.get('birthday')}"), "%Y-%m-%d")
    gender = data.get('gender')
    height = data.get('height')
    weight = data.get('weight')
    bmi = round(weight / height **2,2)# bmi count
    today = datetime.today()# age count
    age = today.year - birthday.year - ((today.month, today.day) < (birthday.month, birthday.day))
    df = pd.read_csv('BMI_Normal.csv')
    matched_rows = df[df['age'] == age].iloc[0]
    print(matched_rows)
    bmi_judge = 'wrong data'
    if gender=='male':
        if bmi<matched_rows['male_min']:
            bmi_judge='Too thin'
        elif bmi>=matched_rows['male_min'] and bmi<=matched_rows['male_max']:
            bmi_judge='Normal'
        else:
            bmi_judge='Too fat'
    elif gender=='female':
        if bmi<matched_rows['female_min']:
            bmi_judge='Too thin'
        elif bmi>=matched_rows['female_min'] and bmi<=matched_rows['female_max']:
            bmi_judge='Normal'
        else:
            bmi_judge='Too fat'
    return jsonify(message=f'Your AGE={age},BMI={bmi},{bmi_judge}.')
```

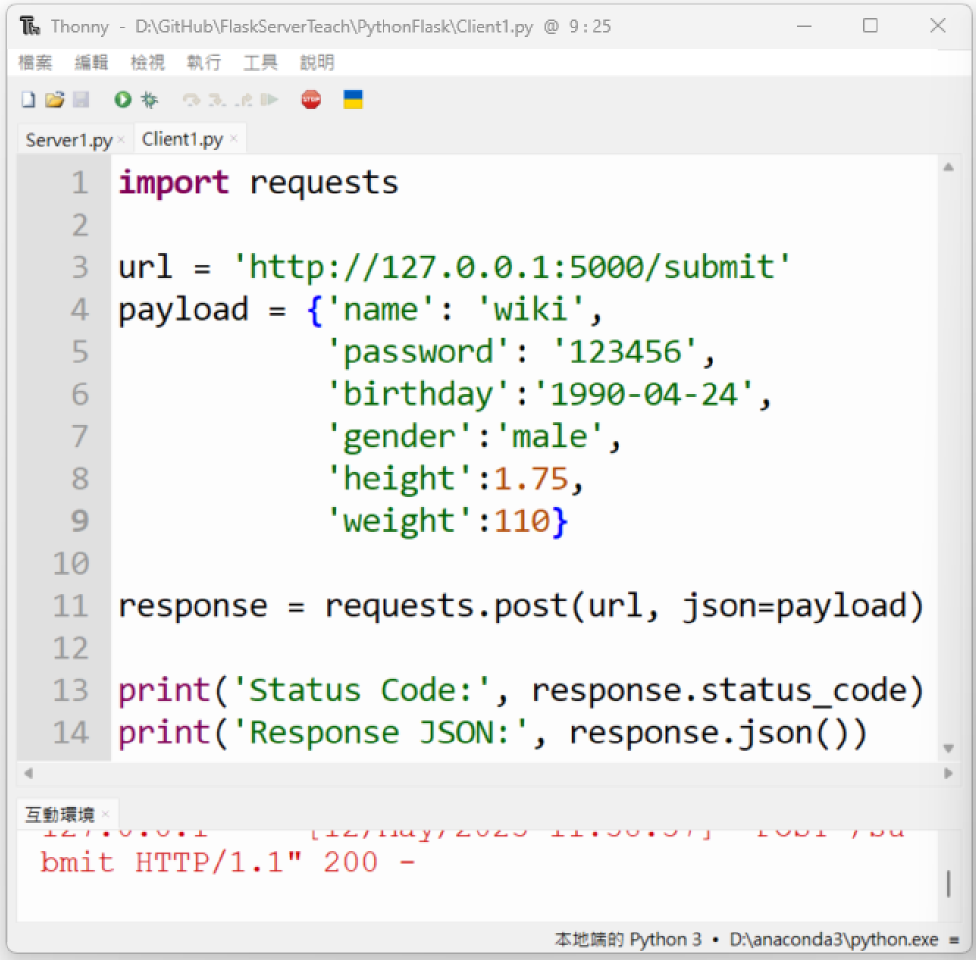
Step2: Write Code on Client1.py

```
import requests

url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
          'password': '123456',
          'birthday':'1990-04-24',
          'gender':'male’,
          'height':1.75,
          'weight':60}

response = requests.post(url, json=payload)

print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

8.Write Code: Post Method Test

Step3 Run Server1.py and Client1.py

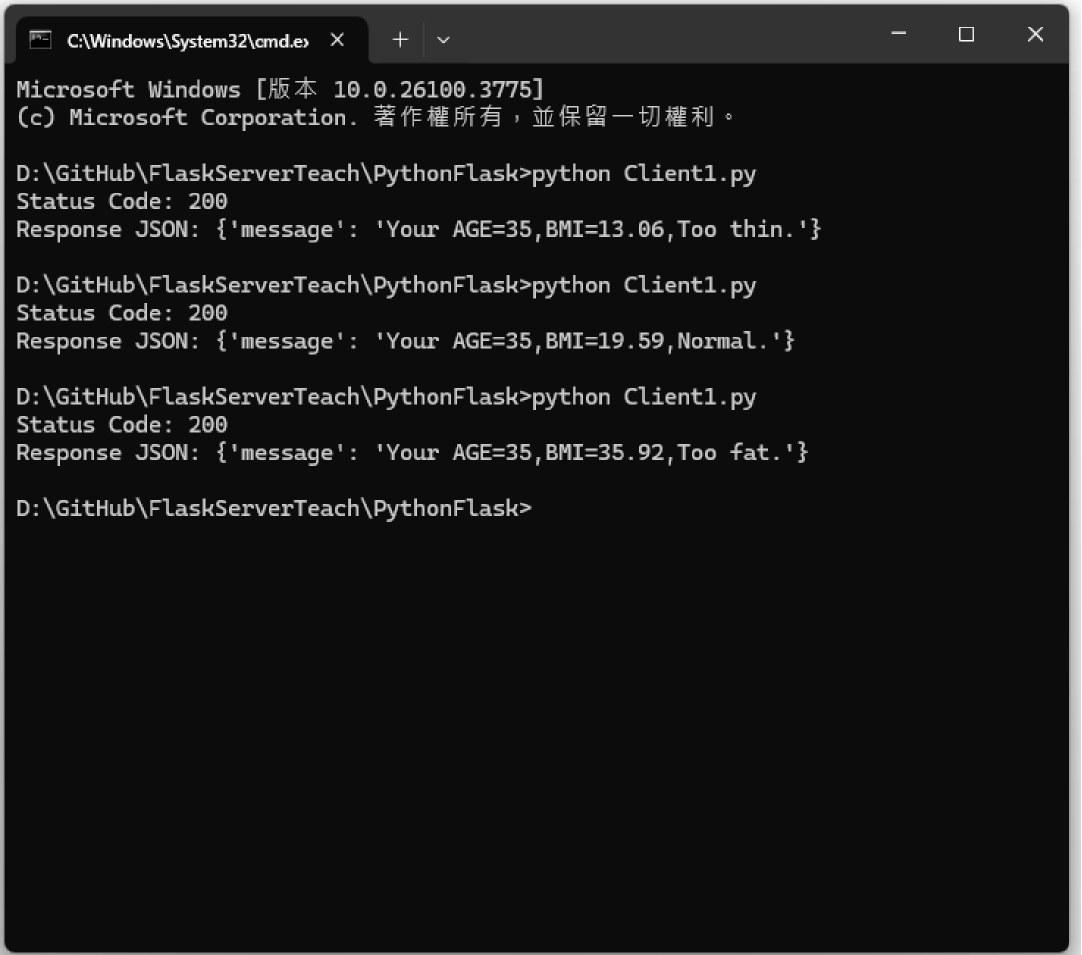
Server1.py Run on IDE Shell.


Thonny - D:\GitHub\FlaskServerTeach\PythonFlask\Client1.py @ 9 : 25
檔案 編輯 檢視 執行 工具 說明
Server1.py
Client1.py
互動環境
本地端的 Python 3 • D:\anaconda3\python.exe

```python
import requests

url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
           'password': '123456',
           'birthday':'1990-04-24',
           'gender':'male',
           'height':1.75,
           'weight':110}

response = requests.post(url, json=payload)

print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

```
bmit HTTP/1.1" 200 -
```

Client1.py Run on Terminal.



```
Microsoft Windows [版本 10.0.26100.3775]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=13.06,Too thin.'}

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=19.59,Normal.'}

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=35.92,Too fat.'}

D:\GitHub\FlaskServerTeach\PythonFlask>
```

Client1.py is run on terminal, because Thonny IDE can’t run two code at same time.
If Client1.py can't run, you may need to change the Python environment.

9.Write Code: Post Method Test

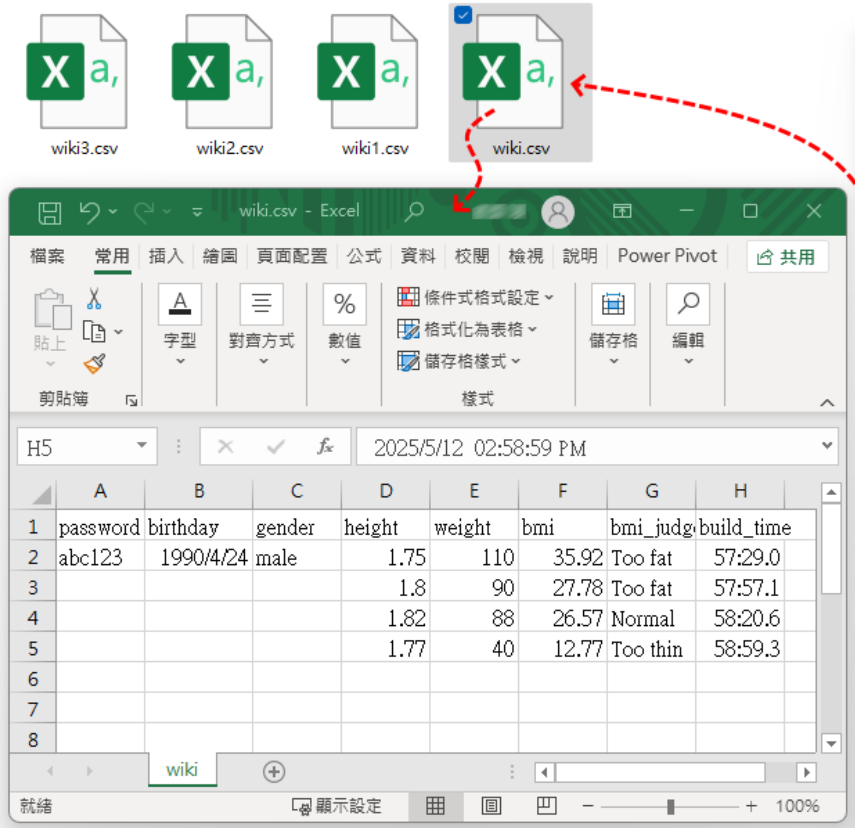
Step1: Write Code on Server1.py

```
from flask import Flask, request, jsonify
from datetime import datetime
import math
import pandas as pd
import os

app = Flask(__name__)

# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
   data = request.get_json()
   name = data.get('name')
   password = data.get('password')
   birthday = datetime.strptime((f"{data.get('birthday')}"), "%Y-%m-%d")
   gender = data.get('gender')
   height = data.get('height')
   weight = data.get('weight')
   bmi = round(weight / height **2,2)# bmi count
   today = datetime.today()
   age = today.year - birthday.year - ((today.month, today.day) < (birthday.month,
birthday.day))
   df = pd.read_csv('BMI_Normal.csv')
   matched_rows = df[df['age'] == age].iloc[0]
   #print(matched_rows)
   bmi_judge = 'wrong data'
   if gender=='male':
      if bmi<matched_rows['male_min']:
         bmi_judge='Too thin'
      elif bmi>=matched_rows['male_min'] and bmi<=matched_rows['male_max']:
         bmi_judge='Normal'
      else:
         bmi_judge='Too fat'
```

```
 elif gender=='female':
      if bmi<matched_rows['female_min']:
         bmi_judge='Too thin'
      elif bmi>=matched_rows['female_min'] and bmi<=matched_rows['female_max']:
         bmi_judge='Normal'
      else:
         bmi_judge='Too fat'
filename = f"UserData\{name}.csv"
   if os.path.exists(filename): # file exists
      load_df = pd.read_csv(filename)# file load
      stored_password = str(load_df.iloc[0]['password'])# password load
      if stored_password != password:# password incorrect
         return jsonify(message="password error!"), 403
      else:# password correct
         user_data = { #Save Data
            'height': height,
            'weight': weight,
            'bmi': bmi,
            'bmi_judge':bmi_judge,
            'build_time':today
         }
         load_df = pd.concat([load_df, pd.DataFrame([user_data])], ignore_index=True)
         load_df.to_csv(filename, index=False)
   else:# no files
      user_data = {#Save Data
         'password': password,
         'birthday': birthday,
         'gender': gender,
         'height': height,
         'weight': weight,
         'bmi': bmi,
         'bmi_judge':bmi_judge,
         'build_time':today
      }
      save_df = pd.DataFrame([user_data])
      save_filename = f"UserData\{name}.csv".replace("/", "_")
      save_df.to_csv(save_filename, index=False)
   return jsonify(message=f'Your AGE={age},BMI={bmi},{bmi_judge}.')
```

9.Write Code: Post Method Test

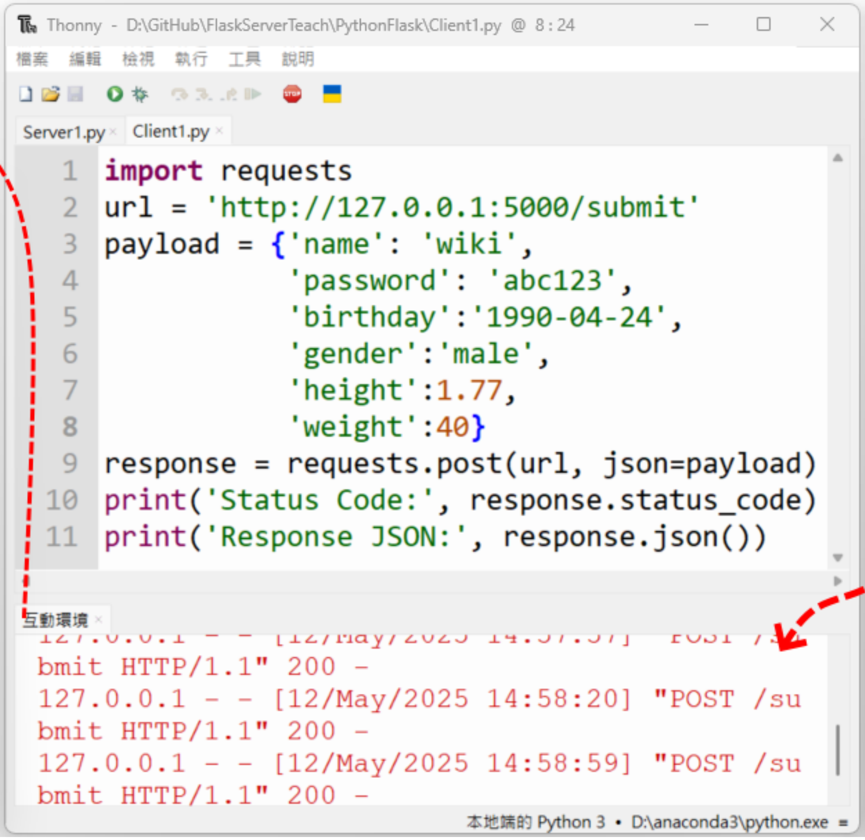
Step2: Run Server1.py and Client1.py



wiki3.csv
wiki2.csv
wiki1.csv
wiki.csv
wiki.csv - Excel
檔案 常用 插入 繪圖 頁面配置 公式 資料 校閱 檢視 說明 Power Pivot
共用
貼上 剪貼簿 字型 對齊方式 數值 條件式格式設定 格式化為表格 儲存格樣式 樣式 儲存格 編輯
wiki
就緒 顯示設定 100%

H5 2025/5/12 02:58:59 PM

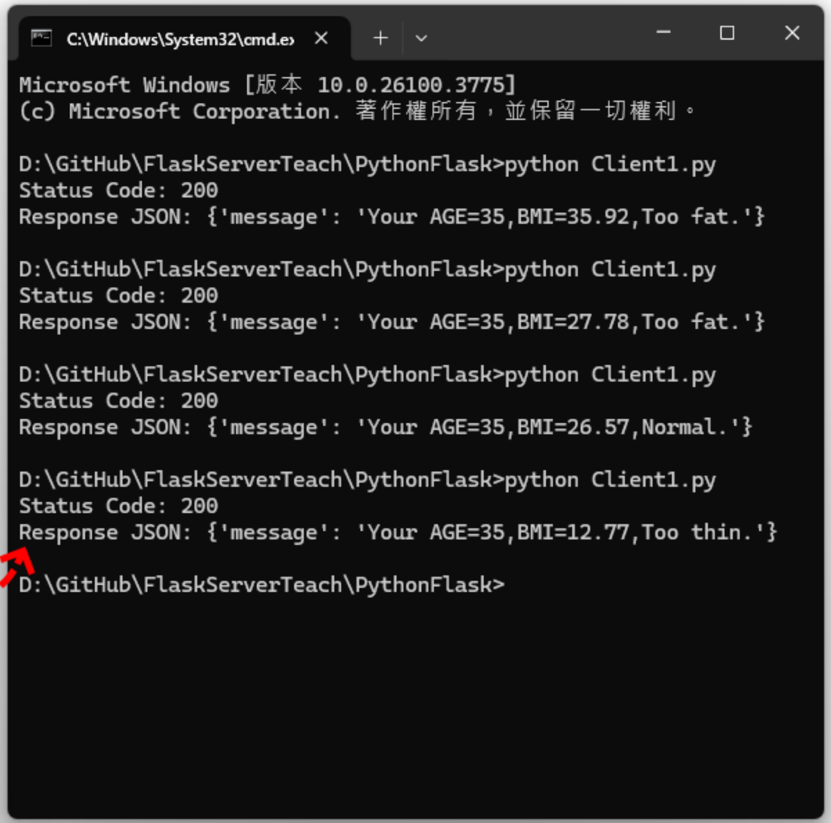
	A	B	C	D	E	F	G	H
1	password	birthday	gender	height	weight	bmi	bmi_judg	build_time
2	abc123	1990/4/24	male	1.75	110	35.92	Too fat	57:29.0
3				1.8	90	27.78	Too fat	57:57.1
4				1.82	88	26.57	Normal	58:20.6
5				1.77	40	12.77	Too thin	58:59.3
6								
7								
8								



Thonny - D:\GitHub\FlaskServerTeach\PythonFlask\Client1.py @ 8 : 24
檔案 編輯 檢視 執行 工具 說明
Server1.py
Client1.py
互動環境
本地端的 Python 3 • D:\anaconda3\python.exe

```python
import requests
url = 'http://127.0.0.1:5000/submit'
payload = {'name': 'wiki',
           'password': 'abc123',
           'birthday':'1990-04-24',
           'gender':'male',
           'height':1.77,
           'weight':40}
response = requests.post(url, json=payload)
print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

```
bmit HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 14:58:20] "POST /su
bmit HTTP/1.1" 200 -
127.0.0.1 - - [12/May/2025 14:58:59] "POST /su
bmit HTTP/1.1" 200 -
```

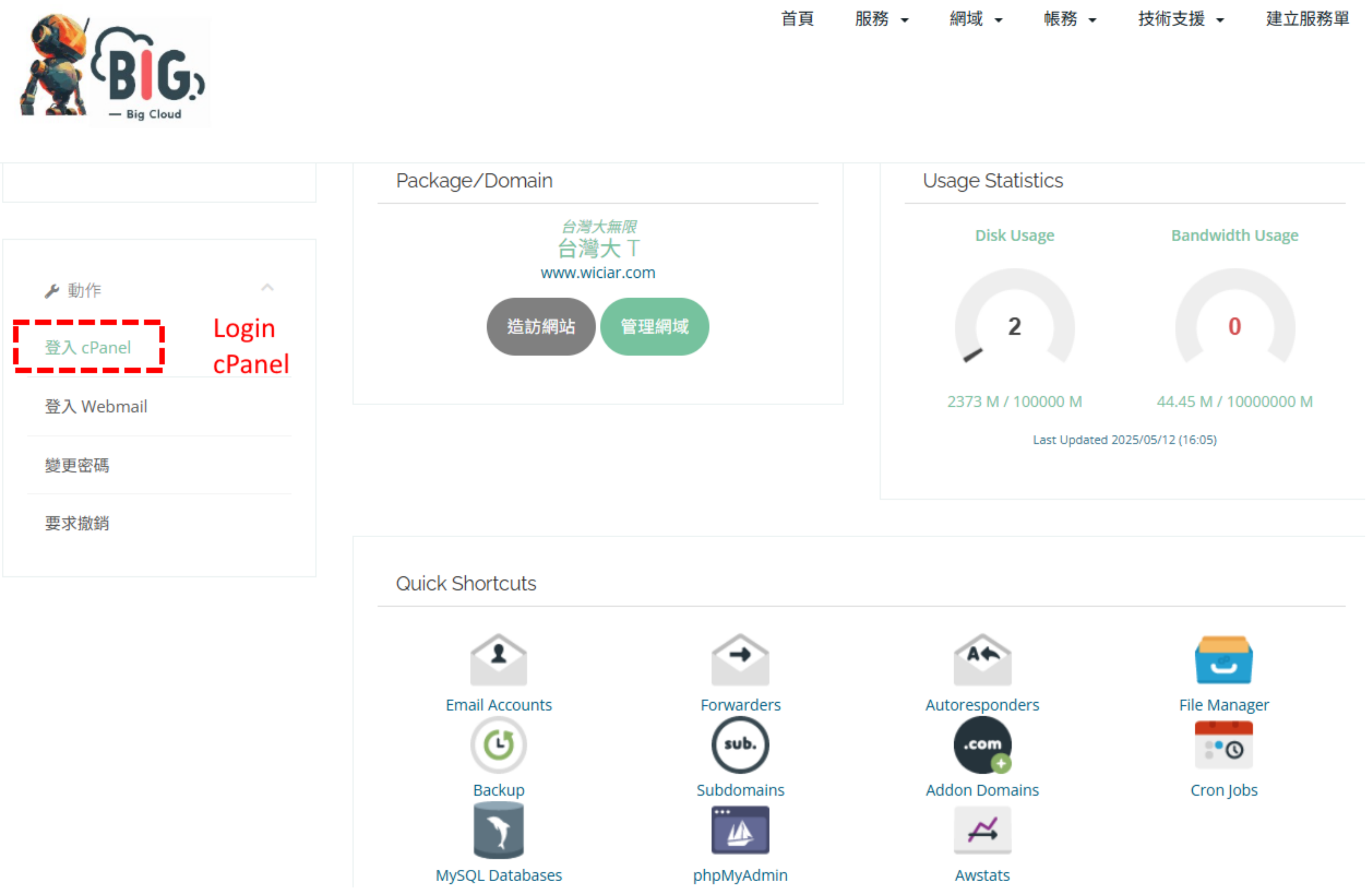


```
Microsoft Windows [版本 10.0.26100.3775]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=35.92,Too fat.'}

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=27.78,Too fat.'}

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=26.57,Normal.'}

D:\GitHub\FlaskServerTeach\PythonFlask>python Client1.py
Status Code: 200
Response JSON: {'message': 'Your AGE=35,BMI=12.77,Too thin.'}

D:\GitHub\FlaskServerTeach\PythonFlask>
```

Client1.py is run on terminal, because Thonny IDE can't run two code at same time.
If Client1.py can't run, you may need to change the Python environment.

10.Web Server Building for cPanel

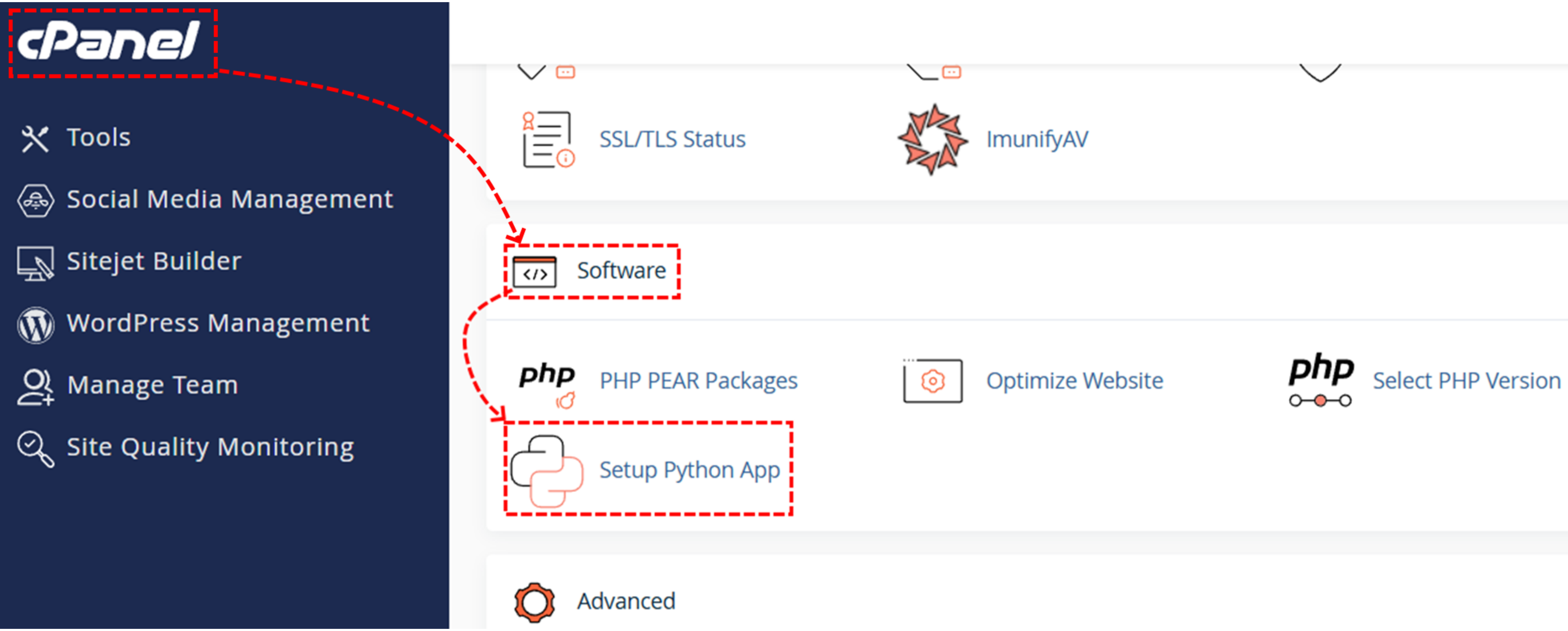
Step1: Paid and apply for a cloud server. (My teaching uses bigcloud.com.tw)



Not necessarily Bigcloud, you can use AWS, Azure, firebase or Alibaba Cloud…
But when you want to bid for the R.O.C gov. Project, Data center must for Chunghwa Telecom's Device.

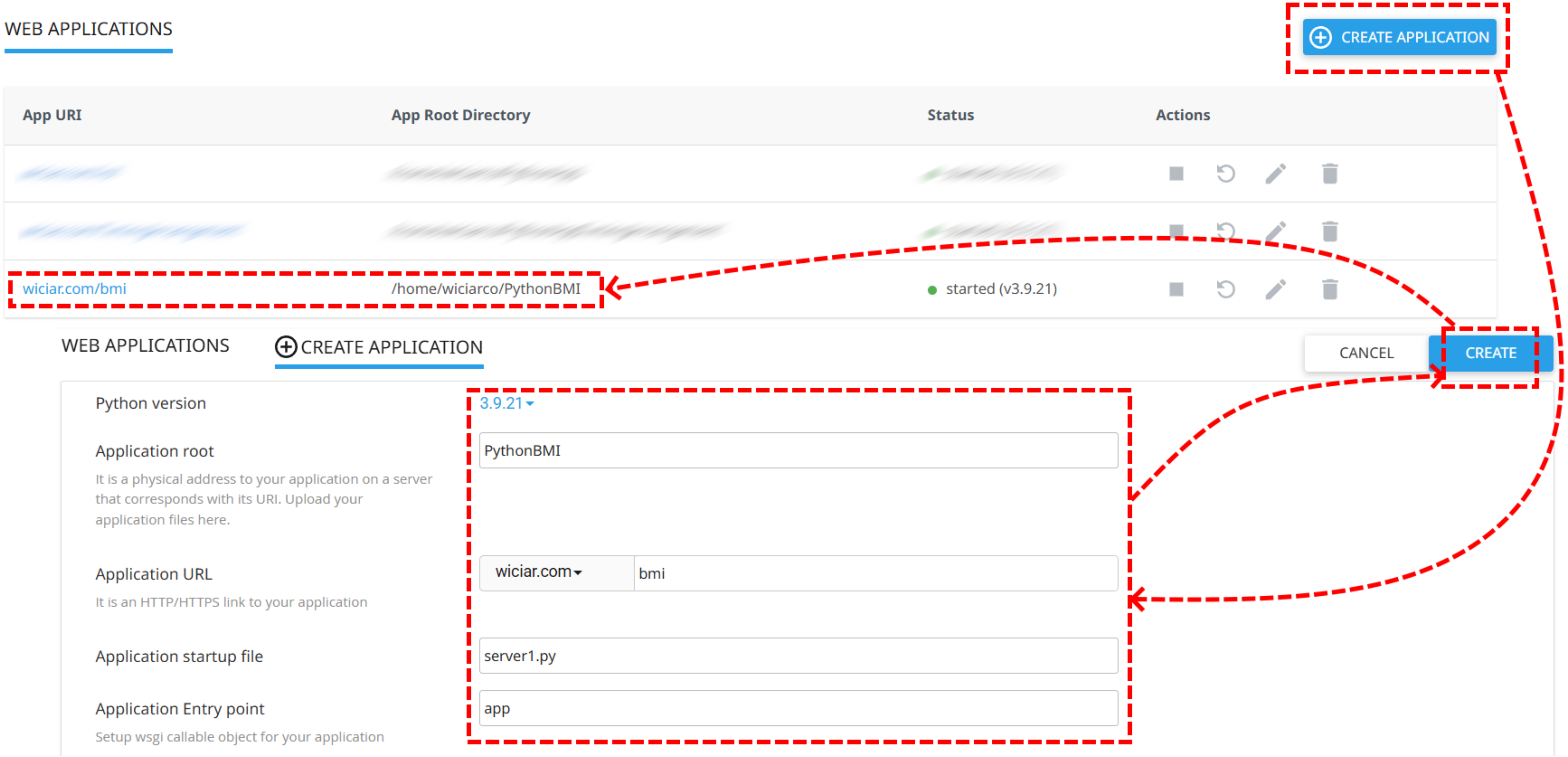
10.Web Server Building for cPanel

Step2: Create python server on cPanel.



10.Web Server Building for cPanel

Step2: Create python server on cPanel.



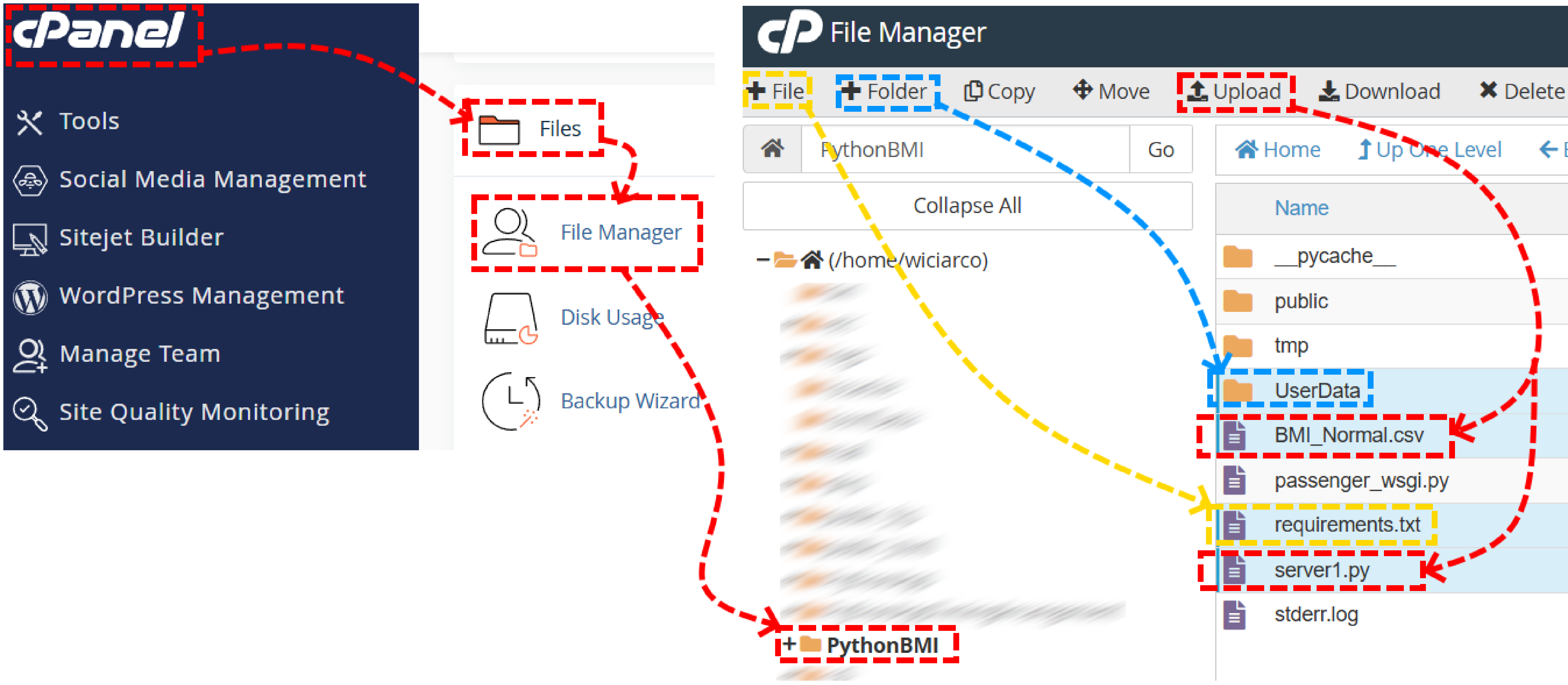
WEB APPLICATIONS

App URI	App Root Directory	Status	Actions
wiciar.com/bmi	/home/wiciarco/PythonBMI	● started (v3.9.21)	

Python version	3.9.21
Application root It is a physical address to your application on a server that corresponds with its URI. Upload your application files here.	PythonBMI
Application URL It is an HTTP/HTTPS link to your application	wiciar.com / bmi
Application startup file	server1.py
Application Entry point Setup wsgi callable object for your application	app

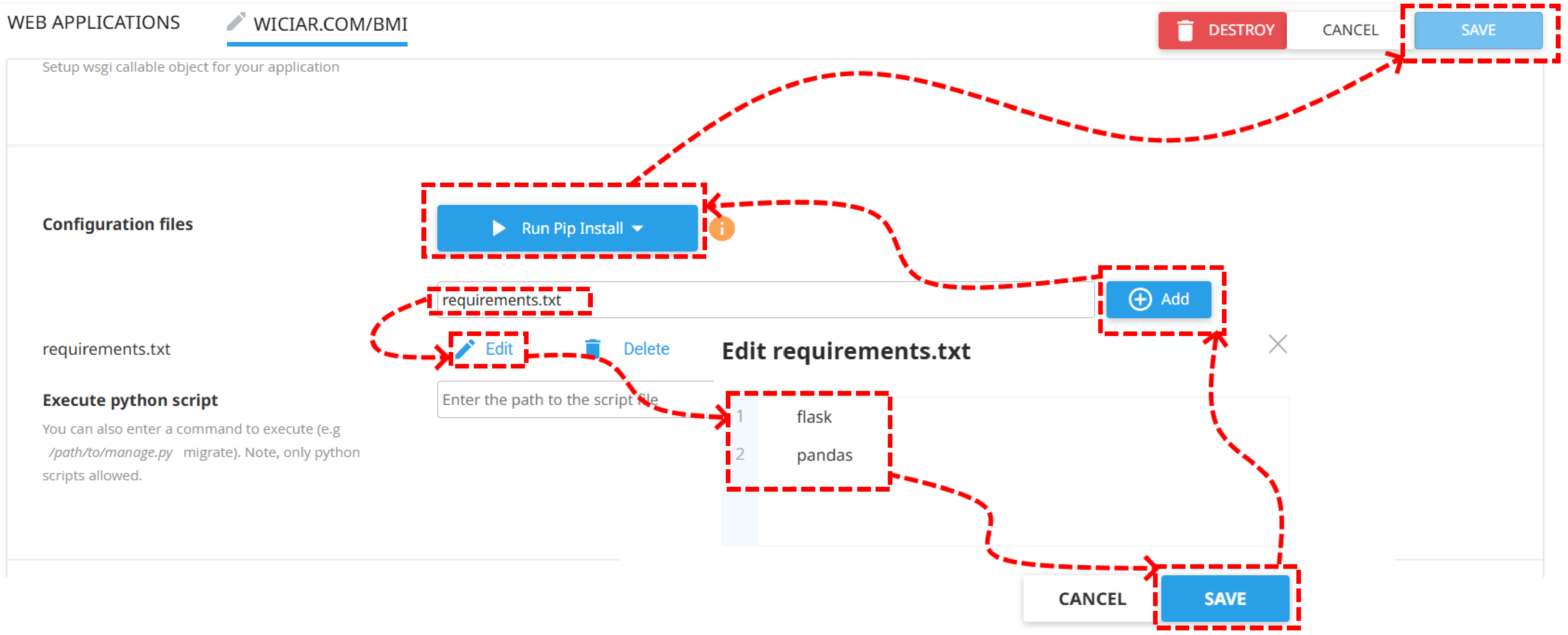
10.Web Server Building for cPanel

Step3: upload and Create app root of files & folders.



10.Web Server Building for cPanel

Step4:updata python server environmental from requirements.txt



Setup wsgi callable object for your application

Configuration files

requirements.txt

requirements.txt

Execute python script

Enter the path to the script file

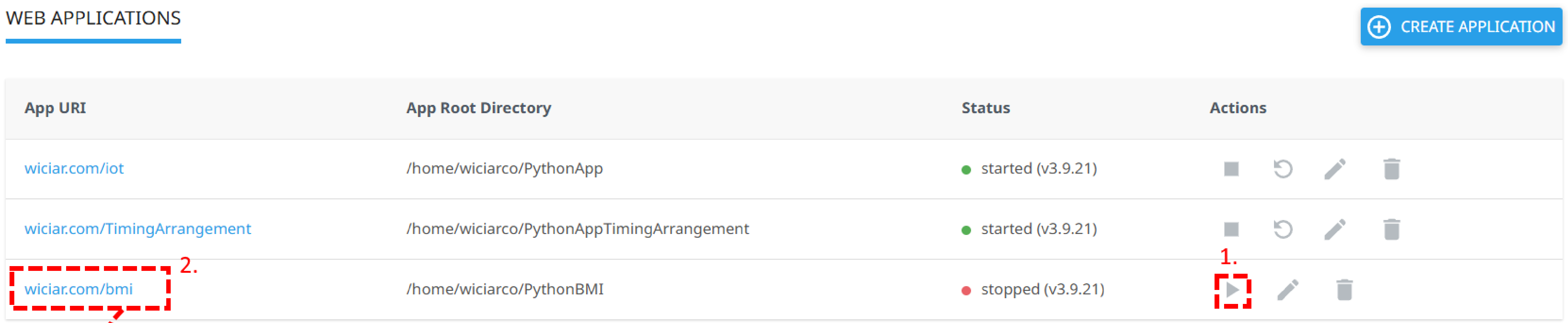
You can also enter a command to execute (e.g */path/to/manage.py* migrate). Note, only python scripts allowed.

Edit requirements.txt

```
flask
pandas
```

10.Web Server Building for cPanel

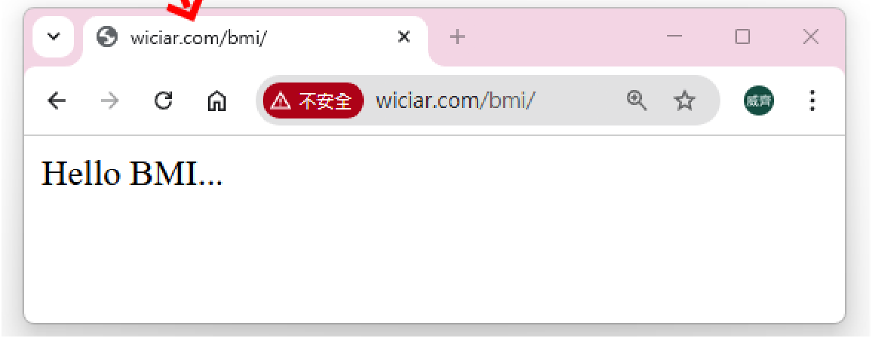
Step5:Run Server



WEB APPLICATIONS

CREATE APPLICATION

App URI	App Root Directory	Status	Actions
wiciar.com/iot	/home/wiciarco/PythonApp	● started (v3.9.21)	
wiciar.com/TimingArrangement	/home/wiciarco/PythonAppTimingArrangement	● started (v3.9.21)	
wiciar.com/bmi	/home/wiciarco/PythonBMI	● stopped (v3.9.21)	

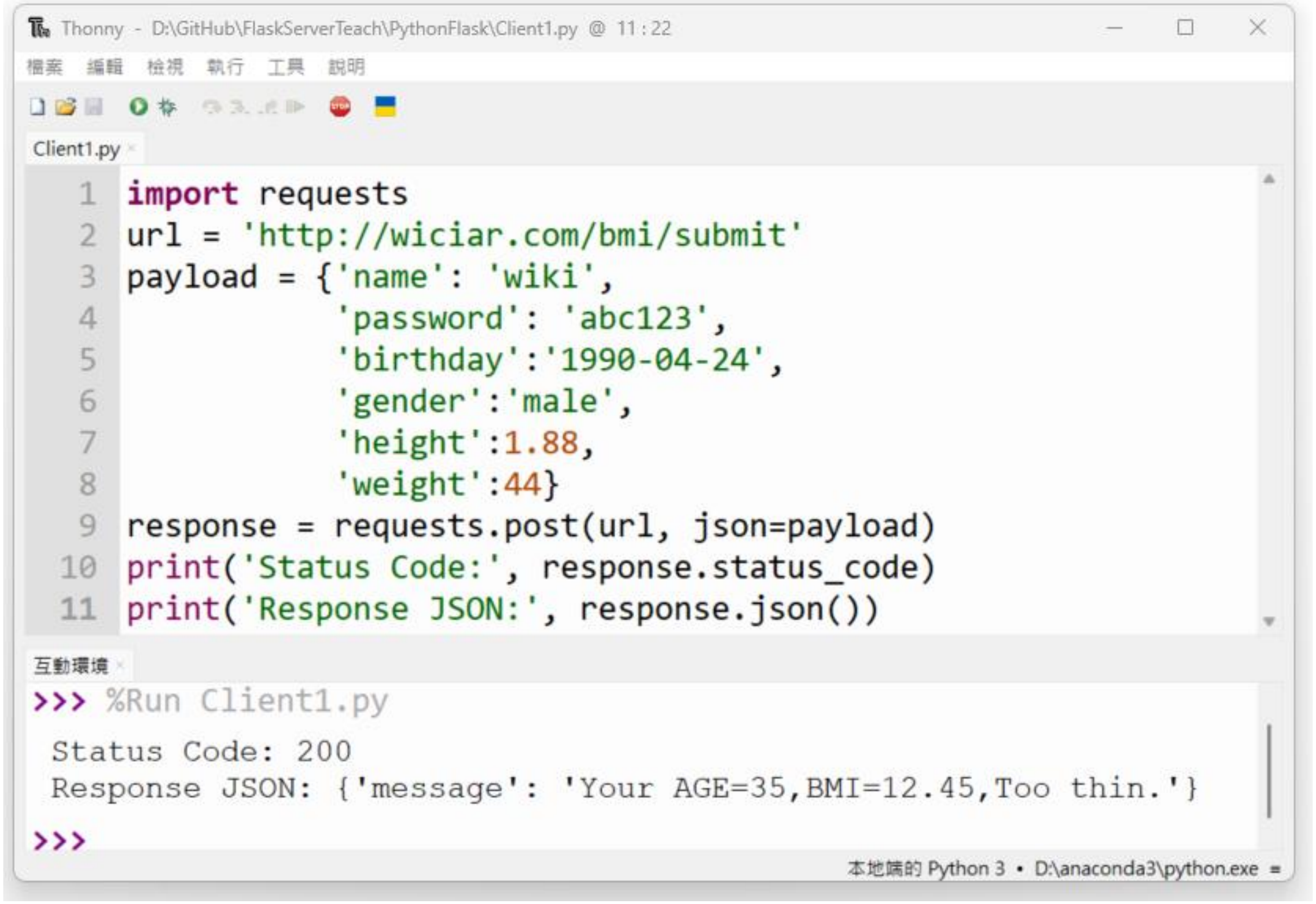


Hello BMI...

10.Web Server Building for cPanel

Step6: Client1.py Code & Test Run

```
import requests
url = 'http://wiciar.com/bmi/submit'
payload = {'name': 'wiki',
          'password': 'abc123',
          'birthday':'1990-04-24',
          'gender':'male',
          'height':1.88,
          'weight':44}
response = requests.post(url, json=payload)
print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```


Thonny - D:\GitHub\FlaskServerTeach\PythonFlask\Client1.py @ 11 : 22
檔案 編輯 檢視 執行 工具 說明
Client1.py
互動環境
本地端的 Python 3 • D:\anaconda3\python.exe

```
import requests
url = 'http://wiciar.com/bmi/submit'
payload = {'name': 'wiki',
           'password': 'abc123',
           'birthday':'1990-04-24',
           'gender':'male',
           'height':1.88,
           'weight':44}
response = requests.post(url, json=payload)
print('Status Code:', response.status_code)
print('Response JSON:', response.json())
```

```
>>> %Run Client1.py
 Status Code: 200
 Response JSON: {'message': 'Your AGE=35,BMI=12.45,Too thin.'}
>>>
```

10.Web Server Building for cPanel

Server1.py Code

```
from flask import Flask, request, jsonify
from datetime import datetime
import math
import pandas as pd
import os

app = Flask(__name__)

# GET Method
@app.route('/')
def hello():
    return 'Hello BMI...'

# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
    data = request.get_json()
    name = data.get('name')
    password = data.get('password')
    birthday = datetime.strptime((f"{data.get('birthday')}"), "%Y-%m-%d")
    gender = data.get('gender')
    height = data.get('height')
    weight = data.get('weight')
    bmi = round(weight / height **2,2)# bmi count
    today = datetime.today()
    age = today.year - birthday.year - ((today.month, today.day) < (birthday.month, birthday.day))
    df = pd.read_csv('BMI_Normal.csv')
    matched_rows = df[df['age'] == age].iloc[0]
    #print(matched_rows)
    bmi_judge = 'wrong data'
    if gender=='male':
        if bmi<matched_rows['male_min']:
            bmi_judge='Too thin'
        elif bmi>=matched_rows['male_min'] and bmi<=matched_rows['male_max']:
            bmi_judge='Normal'
        else:
            bmi_judge='Too fat'
    elif gender=='female':
        if bmi<matched_rows['female_min']:
            bmi_judge='Too thin'
        elif bmi>=matched_rows['female_min'] and bmi<=matched_rows['female_max']:
            bmi_judge='Normal'
        else:
            bmi_judge='Too fat'
    filename = f"UserData//{name}.csv"
    if os.path.exists(filename): # file exists
        load_df = pd.read_csv(filename)# file load
        stored_password = str(load_df.iloc[0]['password'])# password load
        if stored_password != password:# password incorrect
            return jsonify(message="password error!"), 403
        else:# password correct
            user_data = { #Save Data
                'height': height,
                'weight': weight,
                'bmi': bmi,
                'bmi_judge':bmi_judge,
                'build_time':today
            }
            load_df = pd.concat([load_df, pd.DataFrame([user_data])], ignore_index=True)
            load_df.to_csv(filename, index=False)
    else:# no files
        user_data = {#Save Data
            'password': password,
            'birthday': birthday,
            'gender': gender,
            'height': height,
            'weight': weight,
            'bmi': bmi,
            'bmi_judge':bmi_judge,
            'build_time':today
        }
        save_df = pd.DataFrame([user_data])
        #save_filename = f"UserData\{name}.csv".replace("/", "_")
        save_df.to_csv(filename, index=False)
    return jsonify(message=f'Your AGE={age},BMI={bmi},{bmi_judge}.')
```

11.Client UI Design for Python

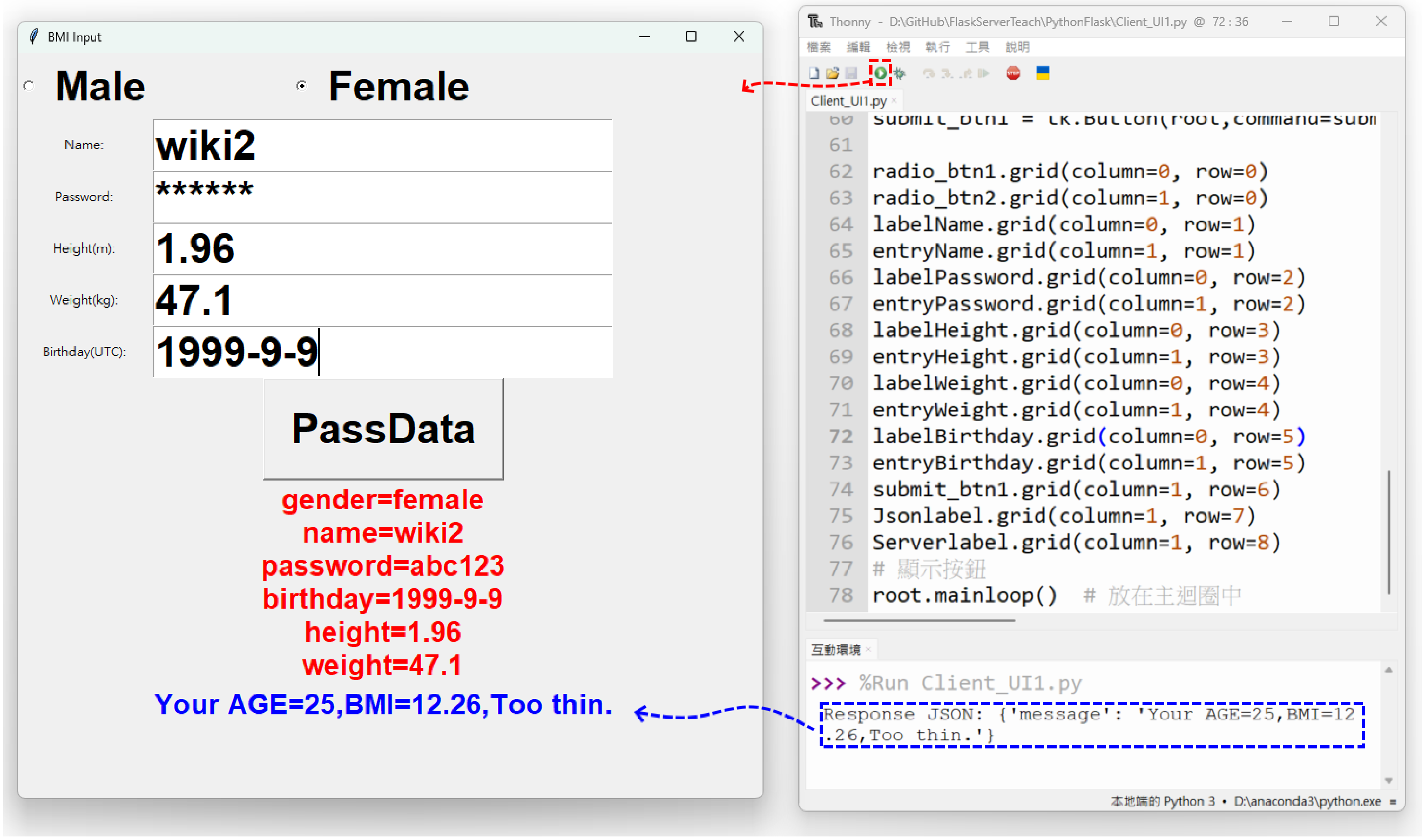
Step1: Write Code--Client_UI1.py

```
import tkinter as tk
import requests

root = tk.Tk()
root.title('BMI Input')
root.geometry('720x720')

gender = tk.StringVar()
name = tk.StringVar()
password = tk.StringVar()
birthday = tk.StringVar()
height = tk.StringVar()
weight = tk.StringVar()
jsonText = tk.StringVar()
serverText = tk.StringVar()

radio_btn1 = tk.Radiobutton(root,text='Male',font=('Arial',30,'bold'),variable=gender,value='male')
radio_btn2 = tk.Radiobutton(root,text='Female',font=('Arial',30,'bold'),variable=gender,value='female')
radio_btn1.select()

labelName = tk.Label(root, text='Name:')
entryName = tk.Entry(root,font=('Arial',30,'bold'),textvariable=name)

labelPassword = tk.Label(root, text='Password:')
entryPassword = tk.Entry(root,show='*',font=('Arial',30,'bold'),textvariable=password)

labelHeight = tk.Label(root, text='Height(m):')
entryHeight = tk.Entry(root,font=('Arial',30,'bold'),textvariable=height)

labelWeight = tk.Label(root, text='Weight(kg):')
entryWeight = tk.Entry(root,font=('Arial',30,'bold'),textvariable=weight)

labelBirthday = tk.Label(root, text='Birthday(UTC):')
entryBirthday = tk.Entry(root,font=('Arial',30,'bold'),textvariable=birthday)

Jsonlabel = tk.Label(root,textvariable=jsonText,font=('Arial',20,'bold'),fg='#f00')

Serverlabel = tk.Label(root,textvariable=serverText,font=('Arial',20,'bold'),fg='#00f')
```

```
def submit():
    global gender,name,password,birthday,height,weight,jsonText
    temp
=f'gender={gender.get()}\nname={name.get()}\npassword={password.get()}\nbi
rthday={birthday.get()}\nheight={height.get()}\nweight={weight.get()}'
    jsonText.set(temp)
    url = 'http://wiciar.com/bmi/submit'
    jsonSet = {'name': name.get(),
               'password': password.get(),
               'birthday':birthday.get(),
               'gender':gender.get(),
               'height':float(height.get()),
               'weight':float(weight.get())}
    #print(jsonSet)
    try:
        response = requests.post(url, json=jsonSet)
        serverText.set(response.json()['message'])
        print('Response JSON:', response.json())
    except:
        serverText.set('Server Link error!!!')
        print('Status Code:', response.status_code)

submit_btn1 =
tk.Button(root,command=submit,text='PassData',font=('Arial',30,'bold'),padx=1
0,pady=10,activeforeground='#f00')

radio_btn1.grid(column=0, row=0)
radio_btn2.grid(column=1, row=0)
labelName.grid(column=0, row=1)
entryName.grid(column=1, row=1)
labelPassword.grid(column=0, row=2)
entryPassword.grid(column=1, row=2)
labelHeight.grid(column=0, row=3)
entryHeight.grid(column=1, row=3)
labelWeight.grid(column=0, row=4)
entryWeight.grid(column=1, row=4)
labelBirthday.grid(column=0, row=5)
entryBirthday.grid(column=1, row=5)
submit_btn1.grid(column=1, row=6)
Jsonlabel.grid(column=1, row=7)
Serverlabel.grid(column=1, row=8)
root.mainloop()
```

11.Client UI Design for Python

Step2: Run Client_UI1.py


BMI Input
Thonny - D:\GitHub\FlaskServerTeach\PythonFlask\Client_UI1.py @ 72 : 36
檔案 編輯 檢視 執行 工具 說明
Client_UI1.py
互動環境
本地端的 Python 3 • D:\anaconda3\python.exe

Male [✓ Female]

Name:	wiki2
Password:	******
Height(m):	1.96
Weight(kg):	47.1
Birthday(UTC):	1999-9-9

PassData

gender=female
name=wiki2
password=abc123
birthday=1999-9-9
height=1.96
weight=47.1

Your AGE=25,BMI=12.26,Too thin.

```python
submit_btn1 = tk.Button(root,command=subm

radio_btn1.grid(column=0, row=0)
radio_btn2.grid(column=1, row=0)
labelName.grid(column=0, row=1)
entryName.grid(column=1, row=1)
labelPassword.grid(column=0, row=2)
entryPassword.grid(column=1, row=2)
labelHeight.grid(column=0, row=3)
entryHeight.grid(column=1, row=3)
labelWeight.grid(column=0, row=4)
entryWeight.grid(column=1, row=4)
labelBirthday.grid(column=0, row=5)
entryBirthday.grid(column=1, row=5)
submit_btn1.grid(column=1, row=6)
Jsonlabel.grid(column=1, row=7)
Serverlabel.grid(column=1, row=8)
# 顯示按鈕
root.mainloop()  # 放在主迴圈中
```

```
>>> %Run Client_UI1.py
Response JSON: {'message': 'Your AGE=25,BMI=12
.26,Too thin.'}
```

12.Client get User historical information

Step1: Write Code--Client2.py

```
import requests
import pandas as pd
import matplotlib.pyplot as plt

url = 'http://wiciar.com/bmi/data_get'
payload = {'name': 'wiki2',
        'password': 'abc123'}
print('Json Code:', payload)
response = requests.post(url, json=payload)
print('Status Code:', response.status_code)
#print('Response JSON:', response.json()['message'])
#Data visualization output
data_list = response.json()['message']
df = pd.DataFrame(data_list)
df['build_time'] = pd.to_datetime(df['build_time'])
df = df.dropna(subset=['bmi', 'build_time'])
df = df.sort_values(by='build_time')
plt.figure(figsize=(10, 5))
plt.plot(df['build_time'], df['bmi'], marker='o', linestyle='-', color='blue', label='BMI')
plt.title('BMI Trend Over Time')
plt.xlabel('Build Time')
plt.ylabel('BMI')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.legend()
plt.show()
```

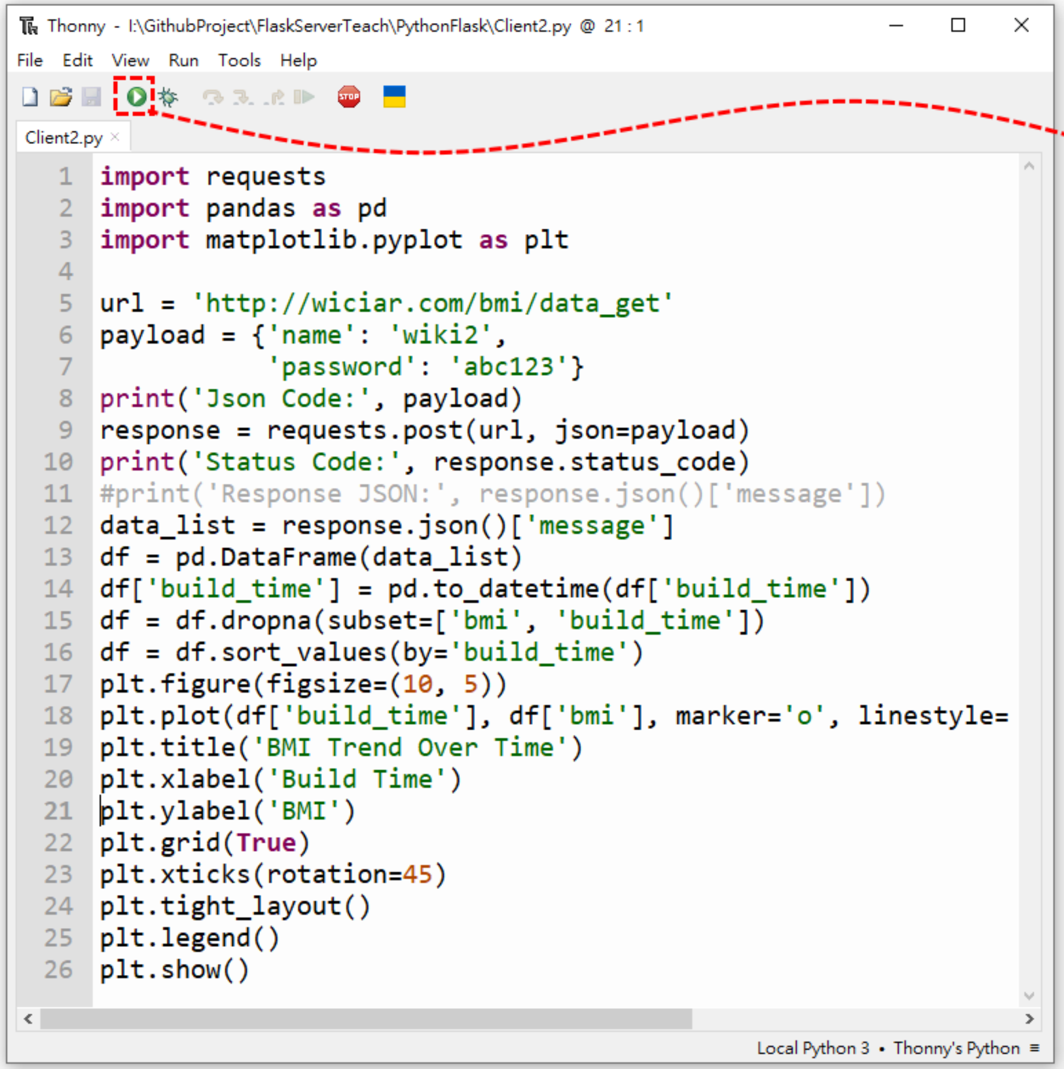
Server1.py Code Design

```
#omit…
# GET Method
@app.route('/')
def hello():
  return 'Hello BMI…'

# POST Method
@app.route('/data_get', methods=['POST'])
def UserDataGet():
  data = request.get_json()
  name = data.get('name')
  password = data.get('password')

  if not name or not password:
    return jsonify(message="get me 'name' & 'password'."), 400

  filename = f"UserData//{name}.csv"

  if os.path.exists(filename):
    df = pd.read_csv(filename)
    if (df["password"] == password).any():
      df = pd.concat([df, pd.DataFrame()], ignore_index=True)
    else:
      return jsonify(message="Password error!"), 403

    df = df.drop(columns=["password"], errors="ignore")
    print(df)
    return jsonify(message=df.to_dict(orient="records"))

  else:
    return jsonify(message="No User Data!"), 403

# POST Method
@app.route('/submit', methods=['POST'])
def submit_form():
#omit…
```

#omit… this can Refer to Section 10

12.Client get User historical information

Step2: Run Code--Client2.py

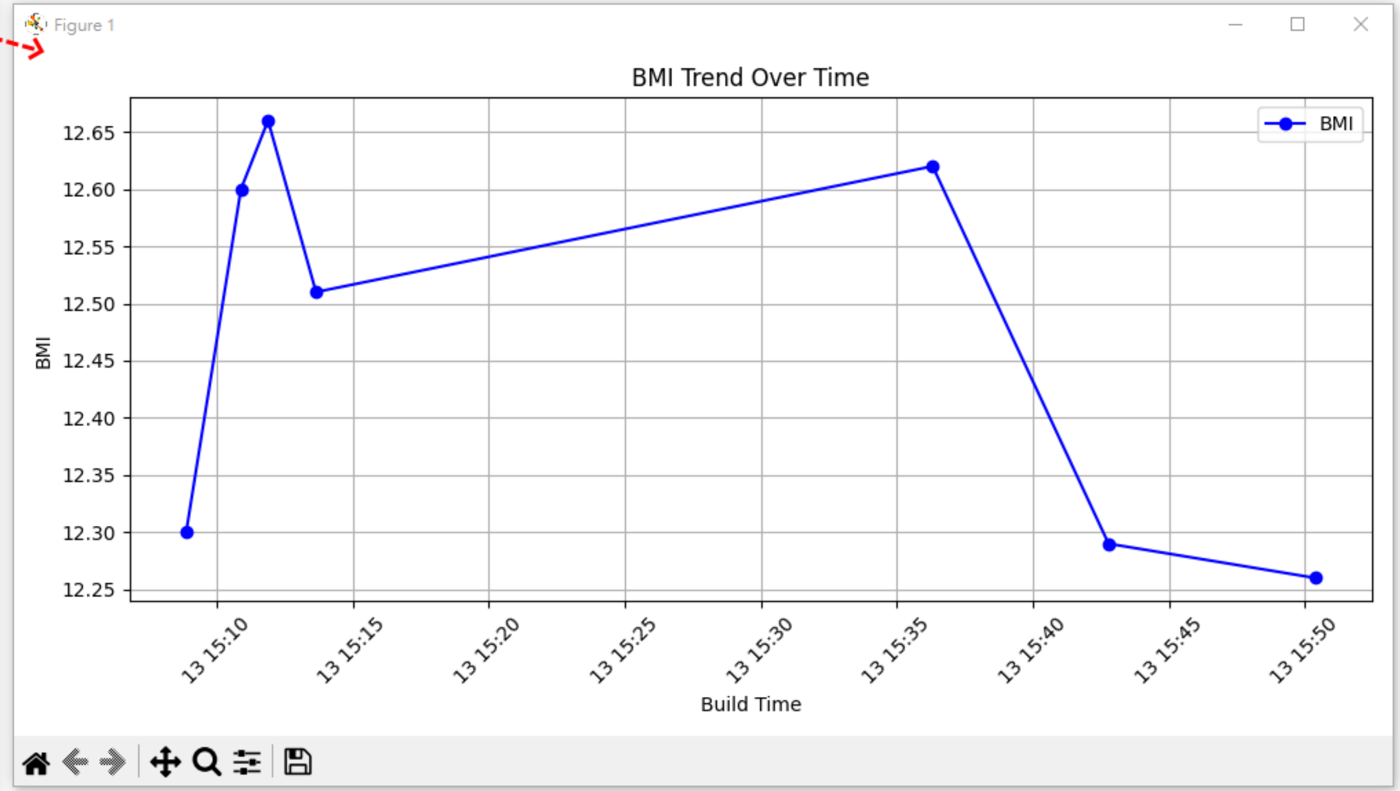


Thonny - I:\GithubProject\FlaskServerTeach\PythonFlask\Client2.py @ 21 : 1

Client2.py

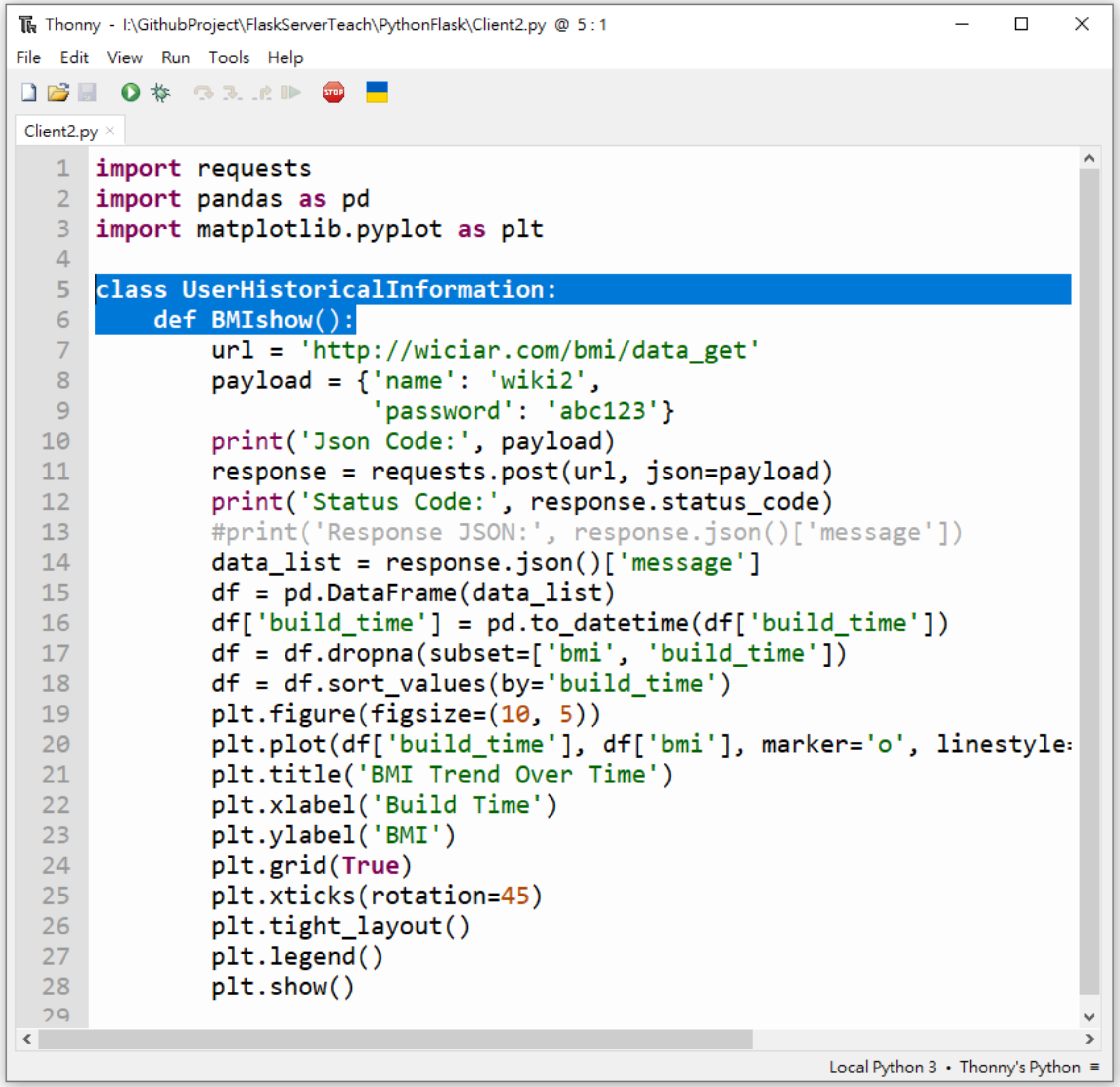
```python
import requests
import pandas as pd
import matplotlib.pyplot as plt

url = 'http://wiciar.com/bmi/data_get'
payload = {'name': 'wiki2',
           'password': 'abc123'}
print('Json Code:', payload)
response = requests.post(url, json=payload)
print('Status Code:', response.status_code)
#print('Response JSON:', response.json()['message'])
data_list = response.json()['message']
df = pd.DataFrame(data_list)
df['build_time'] = pd.to_datetime(df['build_time'])
df = df.dropna(subset=['bmi', 'build_time'])
df = df.sort_values(by='build_time')
plt.figure(figsize=(10, 5))
plt.plot(df['build_time'], df['bmi'], marker='o', linestyle=
plt.title('BMI Trend Over Time')
plt.xlabel('Build Time')
plt.ylabel('BMI')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.legend()
plt.show()
```

Local Python 3 • Thonny's Python



13.Client get User historical information

Step1: Client2.py Redesign


Thonny - I:\GithubProject\FlaskServerTeach\PythonFlask\Client2.py @ 5 : 1
File Edit View Run Tools Help
Client2.py
Local Python 3 • Thonny's Python

```python
import requests
import pandas as pd
import matplotlib.pyplot as plt

class UserHistoricalInformation:
    def BMIshow():
        url = 'http://wiciar.com/bmi/data_get'
        payload = {'name': 'wiki2',
                   'password': 'abc123'}
        print('Json Code:', payload)
        response = requests.post(url, json=payload)
        print('Status Code:', response.status_code)
        #print('Response JSON:', response.json()['message'])
        data_list = response.json()['message']
        df = pd.DataFrame(data_list)
        df['build_time'] = pd.to_datetime(df['build_time'])
        df = df.dropna(subset=['bmi', 'build_time'])
        df = df.sort_values(by='build_time')
        plt.figure(figsize=(10, 5))
        plt.plot(df['build_time'], df['bmi'], marker='o', linestyle:
        plt.title('BMI Trend Over Time')
        plt.xlabel('Build Time')
        plt.ylabel('BMI')
        plt.grid(True)
        plt.xticks(rotation=45)
        plt.tight_layout()
        plt.legend()
        plt.show()
```

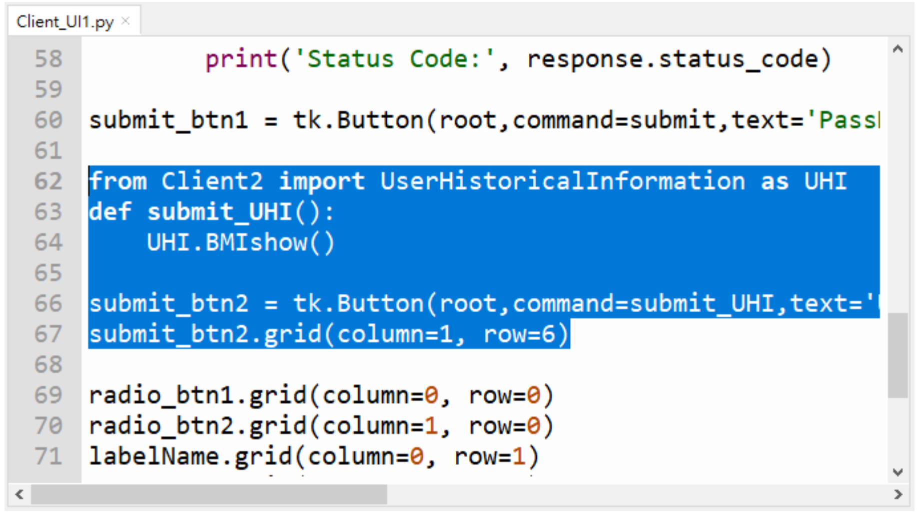
```python
import requests
import pandas as pd
import matplotlib.pyplot as plt

class UserHistoricalInformation:
    def BMIshow():
        url = 'http://wiciar.com/bmi/data_get'
        payload = {'name': 'wiki2',
                   'password': 'abc123'}
        print('Json Code:', payload)
        response = requests.post(url, json=payload)
        print('Status Code:', response.status_code)
        #print('Response JSON:', response.json()['message'])
        data_list = response.json()['message']
        df = pd.DataFrame(data_list)
        df['build_time'] = pd.to_datetime(df['build_time'])
        df = df.dropna(subset=['bmi', 'build_time'])
        df = df.sort_values(by='build_time')
        plt.figure(figsize=(10, 5))
        plt.plot(df['build_time'], df['bmi'], marker='o', linestyle='-', color='blue', label='BMI')
        plt.title('BMI Trend Over Time')
        plt.xlabel('Build Time')
        plt.ylabel('BMI')
        plt.grid(True)
        plt.xticks(rotation=45)
        plt.tight_layout()
        plt.legend()
        plt.show()
```

13.Client get User historical information

Step2: Client_UI1.py Redesign

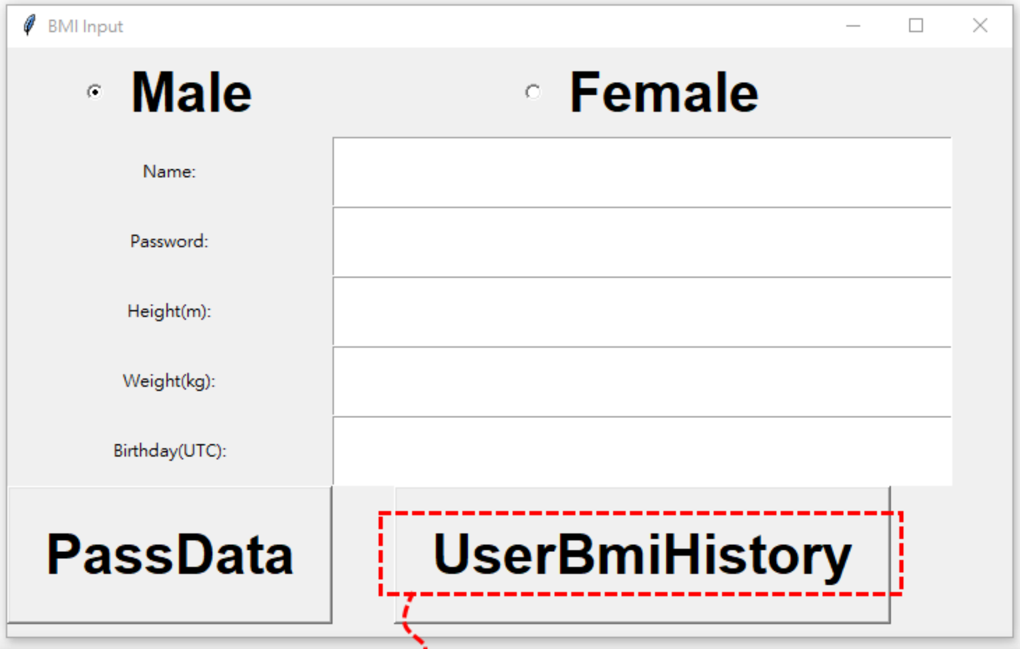
```
#omit...
submit_btn1 =
tk.Button(root,command=submit,text='PassData',font=('Arial',3
0,'bold'),padx=10,pady=10,activeforeground='#f00')

from Client2 import UserHistoricalInformation as UHI
def submit_UHI():
   UHI.BMIshow()

submit_btn2 =
tk.Button(root,command=submit_UHI,text='UserBmiHistory',fo
nt=('Arial',30,'bold'),padx=10,pady=10,activeforeground='#f00')
submit_btn2.grid(column=1, row=6)

radio_btn1.grid(column=0, row=0)
radio_btn2.grid(column=1, row=0)
#omit...
```

#omit... this can Refer to Section 11



```
        print('Status Code:', response.status_code)

submit_btn1 = tk.Button(root,command=submit,text='Pass

from Client2 import UserHistoricalInformation as UHI
def submit_UHI():
    UHI.BMIshow()

submit_btn2 = tk.Button(root,command=submit_UHI,text='
submit_btn2.grid(column=1, row=6)

radio_btn1.grid(column=0, row=0)
radio_btn2.grid(column=1, row=0)
labelName.grid(column=0, row=1)
```

Step3: Run Client_UI1.py


BMI Input
Male
Female
Name:
Password:
Height(m):
Weight(kg):
Birthday(UTC):
PassData
UserBmiHistory

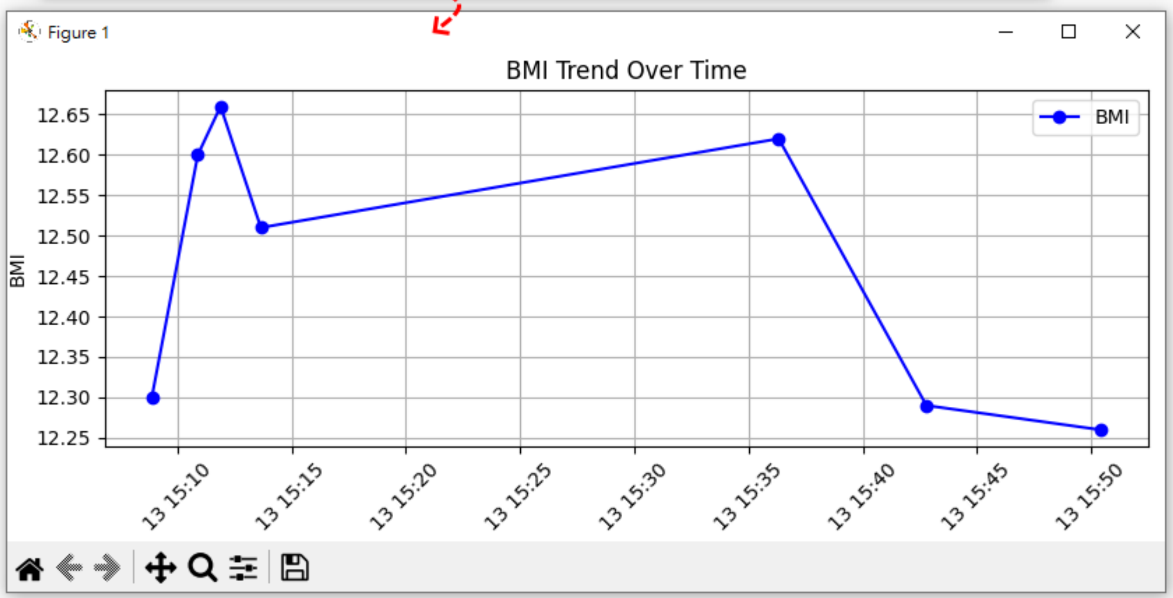


Figure 1
BMI Trend Over Time
BMI
13 15:10
13 15:15
13 15:20
13 15:25
13 15:30
13 15:35
13 15:40
13 15:45
13 15:50
BMI