

通訊網路實驗

REST API

112學年度 第一學期

Dept. of Electrical and Computer Engineering (ECE)
National Yang Ming Chiao Tung University

REST API

- API (application programming interface) 是應用程式的介面/接口
- Representational State Transfer , 簡稱 REST (RESTful) , 它是一種網路架構風格
 - ◆ 使用 HTTP 的協定完整定義 Web Service 在HTTP Request 的各種流程

RESTful API
GET PUT POST DELETE

RESTful 例子

一般

- 所有使用者資料 /getAllUsers
 - 獲取使用者資料 /getUser/1
 - 新增使用者資料 /createUser
 - 更新使用者資料 /updateUser/1
 - 刪除使用者資料 /deleteUser/1
-
- 不夠直觀簡潔
 - 使用拓展子分頁來達到 HTTP Verb (一個子分頁一個Verb)

REST 風格

- 所有使用者資料 /GET/users
 - 獲取使用者資料 /GET/user/1
 - 新增使用者資料 /POST/user
 - 更新使用者資料 /PUT/user/1
 - 刪除使用者資料 /DELETE/user/1
-
- 直觀簡潔的資源 URI
 - 善用 HTTP Verb 達到對資源的操作
 - 使用 Web 所接受的資料類型: JSON, XML, YAML 等，最常見的是 JSON



HTTP Request Method

- 常用的有 GET / PUT / POST / DELETE
- **GET** : 向指定的資源要求取得資料，並不會動到內部資源
- **PUT** : 向指定資源位置提交更新內容
- **POST** : 向指定資源提交資料
- **DELETE** : 向指定資源位置請求刪除內容

HTTP Request Example

```
POST /task/ HTTP/1.1
Host: localhost:9090
Content-Type: application/json;

{
  "text" : "AAA",
  "tags" : ["BBB", "BBB"],
  "due" : "2020-05-23T18:25:43.511Z"
}
```

Json 格式

```
{
  "String" : "This is a string." ,
  "Value" : 123456,
  "Boolean" : True,
  "Object" :
  {
    "var1" : "AAA" ,
    "var2" : "BBB"
  },
  "Array" :
  [
    "var3" : 0,
    "var4" : False
  ]
}
```

HTTP Response Example

MethodGETRequest URLhttps://gorest.co.in/public/v1/posts

SEND

Parameters

HeadersVariables

Toggle source modeInsert headers set

Header namecontent-typeHeader valueapplication/json

X

ADD HEADER

Headers are valid

Headers size: 30 bytes

200 OK69.00 ms

DETAILS

GEThttps://gorest.co.in/public/v1/posts

Response headers18Request headers2Redirects0Timings

```
server: nginx
date: Sun, 12 Dec 2021 13:44:13 GMT
content-type: application/json; charset=utf-8
transfer-encoding: chunked
connection: keep-alive
vary: Accept-Encoding; Origin
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
x-download-options: noopen
x-permitted-cross-domain-policies: none
referrer-policy: strict-origin-when-cross-origin
etag: W/"8dc328d87f0cf5aee23344ef287ad1a4"
cache-control: max-age=0, private, must-revalidate
set _gorest_co_in_session=CFTQBpU8liT5m%2FSXu0LUC62iWK1upYBT1eZRQVdhL6c61fXOX8b8lE3%2Fk5gE1cJRaoc52Tu9yRb0WtXpNqIm1DBop8udyfcUMgd5ikX%2B0
- zr0KX7Pod5%2FzE%2FwWwzLY%2Fp4fyz9vi3fGtWp4s4pGgFtkDvj7Xp4mCmRNdwal9ediJ4FzGN1pEiv2ExUjvQ%2Fm5jG3cVVIg%2FmOYihXfRpSCv1NA4qQBYsxHGtKqK3P
coo KXW0ou9kzjPjifXsoTthkmbnM98hnnhR04IHumIedmtArFWP7ncrPNPY1IbORErwo%3D--5LiKIkfQ81obumW1--vKP70N0MQ6qpkPwc%2BMN39A%3D%3D; path=/;
kie secure; HttpOnly; SameSite=Lax
:
x-request-id: 696184f8-706f-4d2e-a883-0006a2735e8f
```

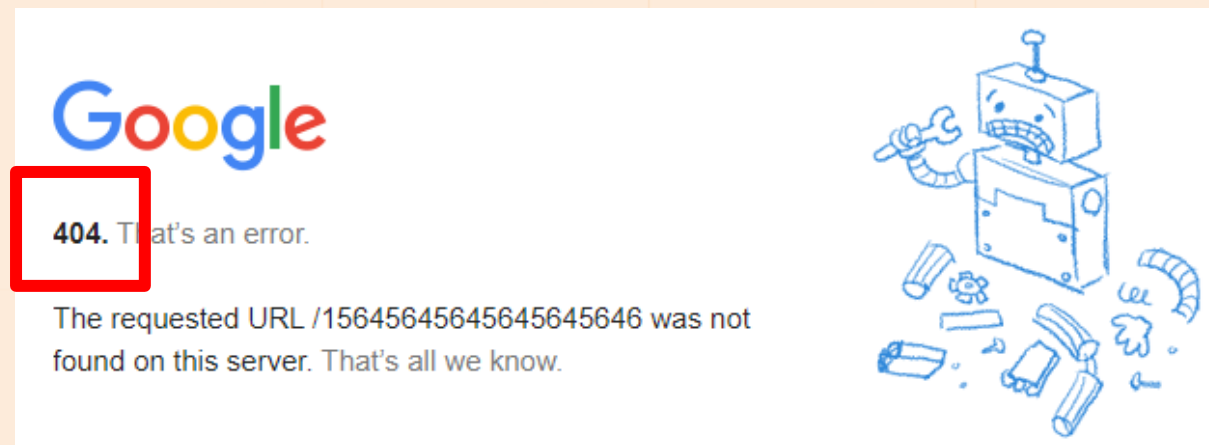
HTTP Response Example

```
{
  - "meta": {
    - "pagination": {
      "total": 180,
      "pages": 9,
      "page": 1,
      "limit": 20,
    - "links": {
      "previous": null,
      "current": "https://gorest.co.in/public/v1/posts?page=1",
      "next": "https://gorest.co.in/public/v1/posts?page=2"
    }
  }
},
- "data": [Array[20]
  - 0: {
    "id": 1539,
    "user_id": 5791,
    "title": "Test post",
    "body": "On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain. These cases are perfectly simple and easy to distinguish. In a free hour, when our power of choice is untrammelled and when nothing prevents our being able to do what we like best, every pleasure is to be welcomed and every pain avoided. But in certain circumstances and owing to the claims of duty or the obligations of business it will frequently occur that pleasures have to be repudiated and annoyances accepted. The wise man therefore always holds in these matters to this principle of selection: he rejects pleasures to secure other greater pleasures, or else he endures pains to avoid worse pains."
  },
  - 1: {
    "id": 1540,
    "user_id": 5791,
    "title": "Test post",
    "body": "On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain. These cases are perfectly simple and easy to distinguish. In a free hour, when our power of choice is untrammelled and when nothing prevents our being able to do what we like best, every pleasure is to be welcomed and every pain avoided. But in certain circumstances and owing to the claims of duty or the obligations of business it will frequently occur that pleasures have to be repudiated and annoyances accepted. The wise man therefore always holds in these matters to this principle of selection: he rejects pleasures to secure other greater pleasures, or else he endures pains to avoid worse pains."
  },
  ...
]
```

HTTP Status Codes

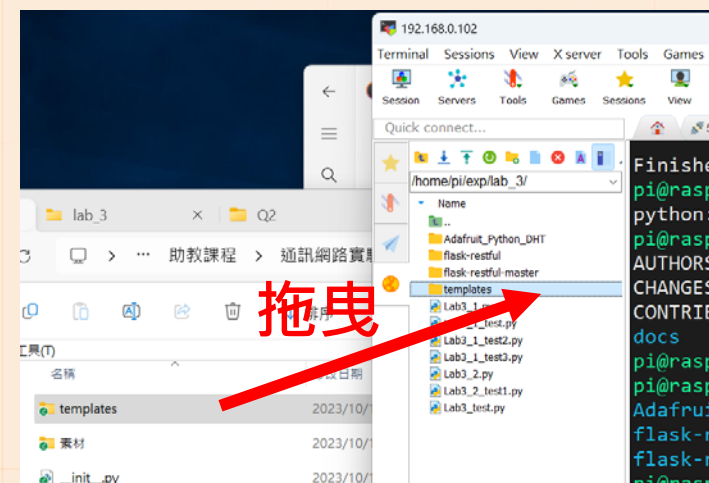
- 為從伺服器端回應的狀態，夾帶在 HTTP Response 中，由三個數字組成

🔒 google.com/15645645645645645646



下載本次實驗函式庫

- `sudo pip install flask`
- `git clone https://github.com/flask-restful/flask-restful.git`
 - ◆ `cd flask-restful`
 - ◆ `sudo python setup.py develop`
- 程式碼已放在 E3
 - ◆ `templates` 資料夾也需要下載，
並連同整個資料夾上傳至Rpi



Code 解釋

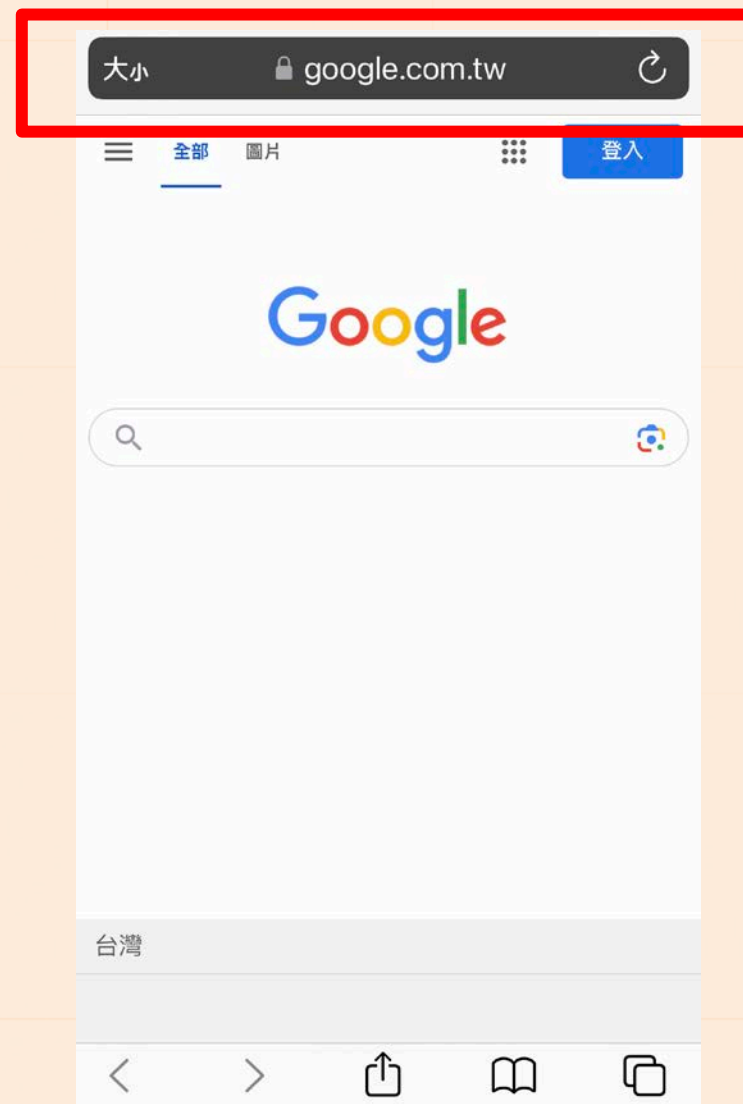
```
1  |from flask import Flask
2
3  app = Flask(__name__)
4
5  @app.route('/')
6  def index():
7      return 'hello!!'
8
9  if __name__ == '__main__':
10 |    app.run(host='0.0.0.0', port=3000, debug=True)
```

- from flask import Flask
 - ◆ 引入相關 Flask 函式庫
- app = Flask(__name__)
 - ◆ 初始化 Flask 物件，並貼上 app 這個 flag，之後就可以方便使用這個物件裡面的功能
- @app.route('/')
 - ◆ 創造出主網域下 "/" 這個網址，並定義 Request 該網址時可用的手段

- `def index():`
 - ◆ 定義一個函數，只要使用者請求相應的網址，就執行該函數
- `return`
 - ◆ 使用者請求該網址之後，根據函數運算之後回傳給使用者的資料，也就是網頁的資料
- `app.run()`
 - ◆ 使整個 Flask Web service 運作
 - ◆ 定義 host 地址與 port

如何發送 HTTP Request

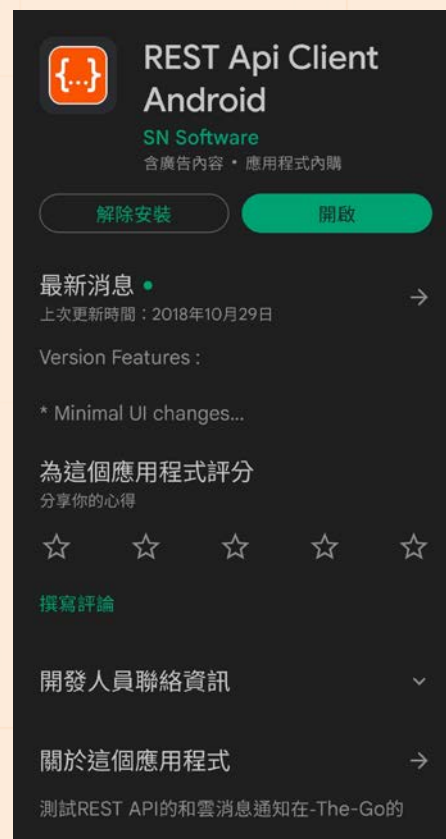
- 1. 使用手機或電腦網頁瀏覽器
 - ◆ 輸入 Server IP 地址或是域名即可使用
 - ◆ 最容易使用的方式



如何發送 HTTP Request

2. 使用手機 APP

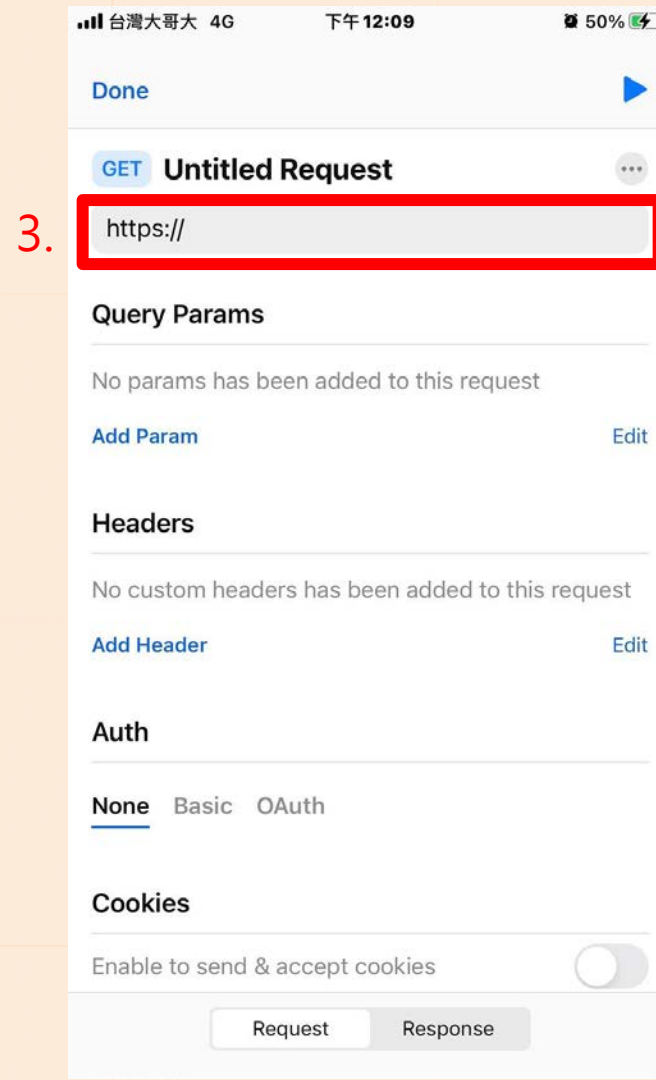
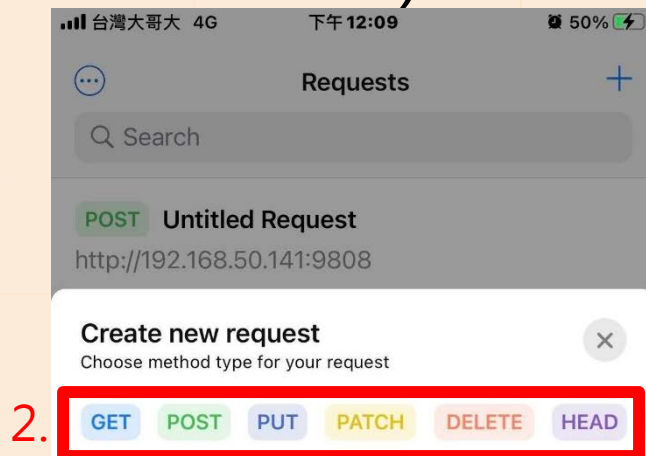
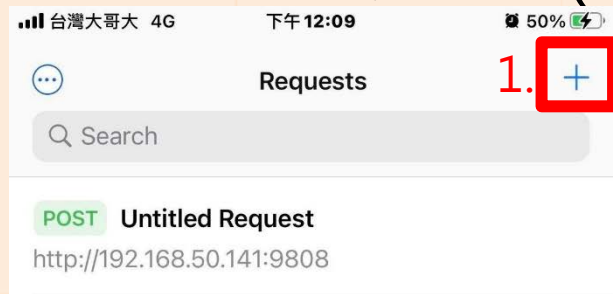
- ◆ 能根據需求進行 GET、POST、等 HTTP Request
- ◆ 能自己設定夾帶在 HTTP Request 中的 Json



※本次實驗使用 private IP 來讓手機與 Flask Web Service 互相連線，所以需要讓手機與 Rpi 板同時連接同一個無線網路分享器 (AP)

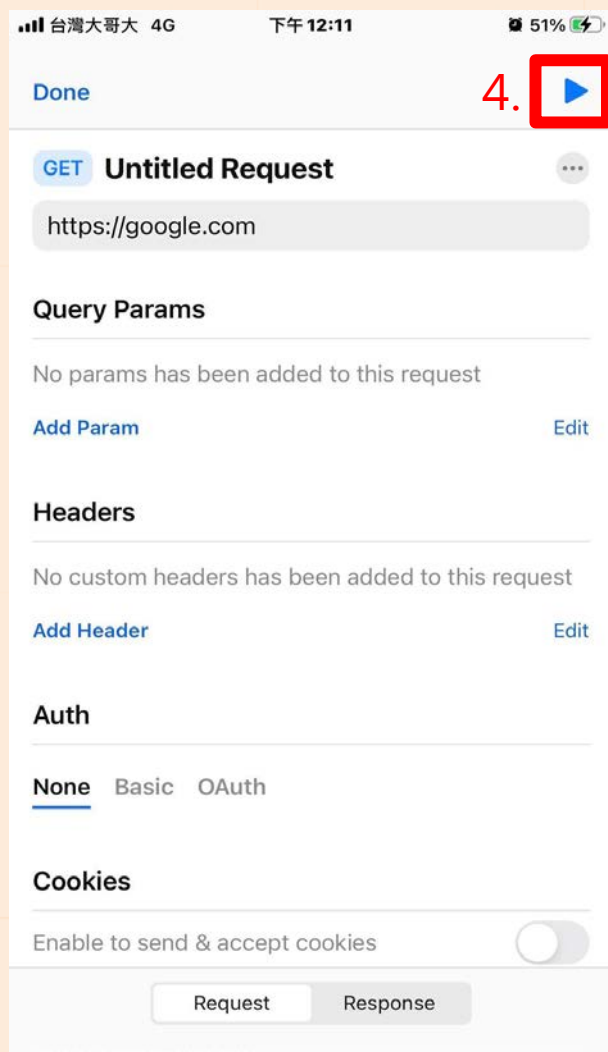
如何發送 HTTP Request

2. 使用手機 APP (IOS、Android)



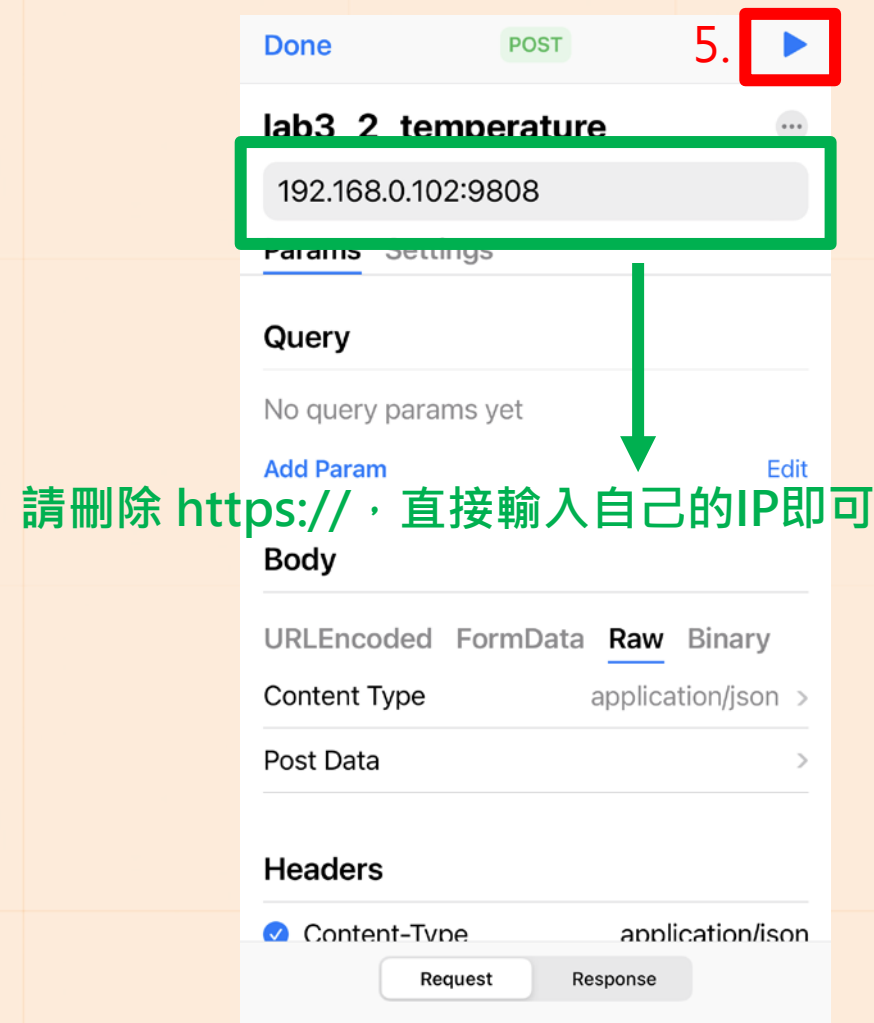
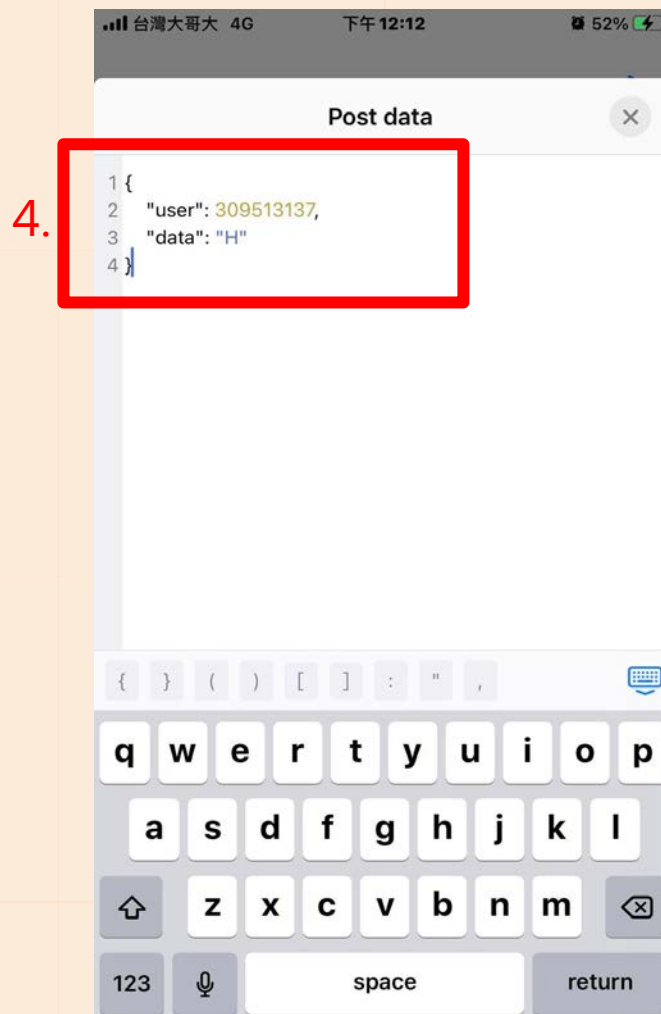
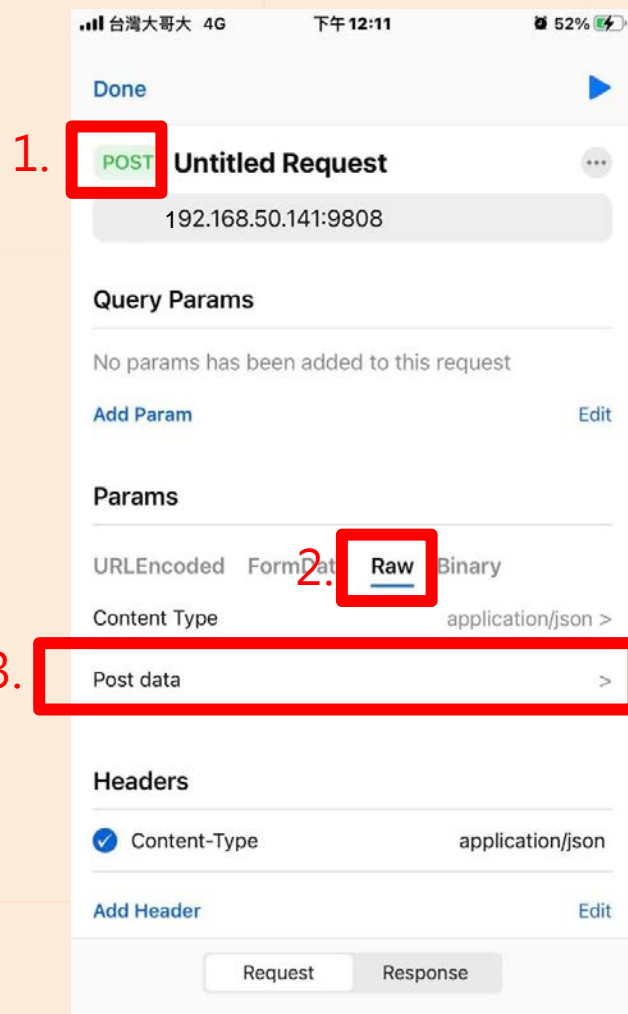
如何發送 HTTP Request

2. 使用手機 APP (IOS 、 Android)



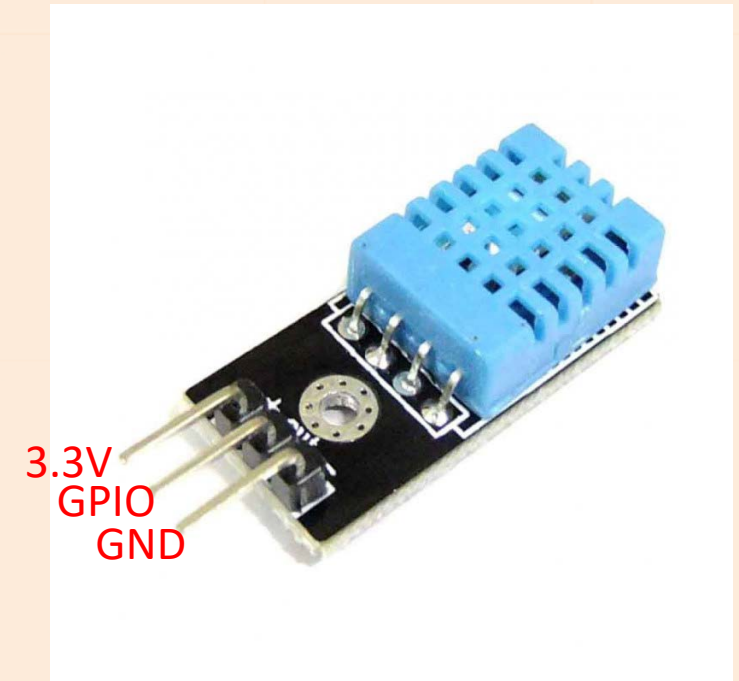
如何發送 HTTP Request

2. 使用手機 APP (IOS 、 Android)

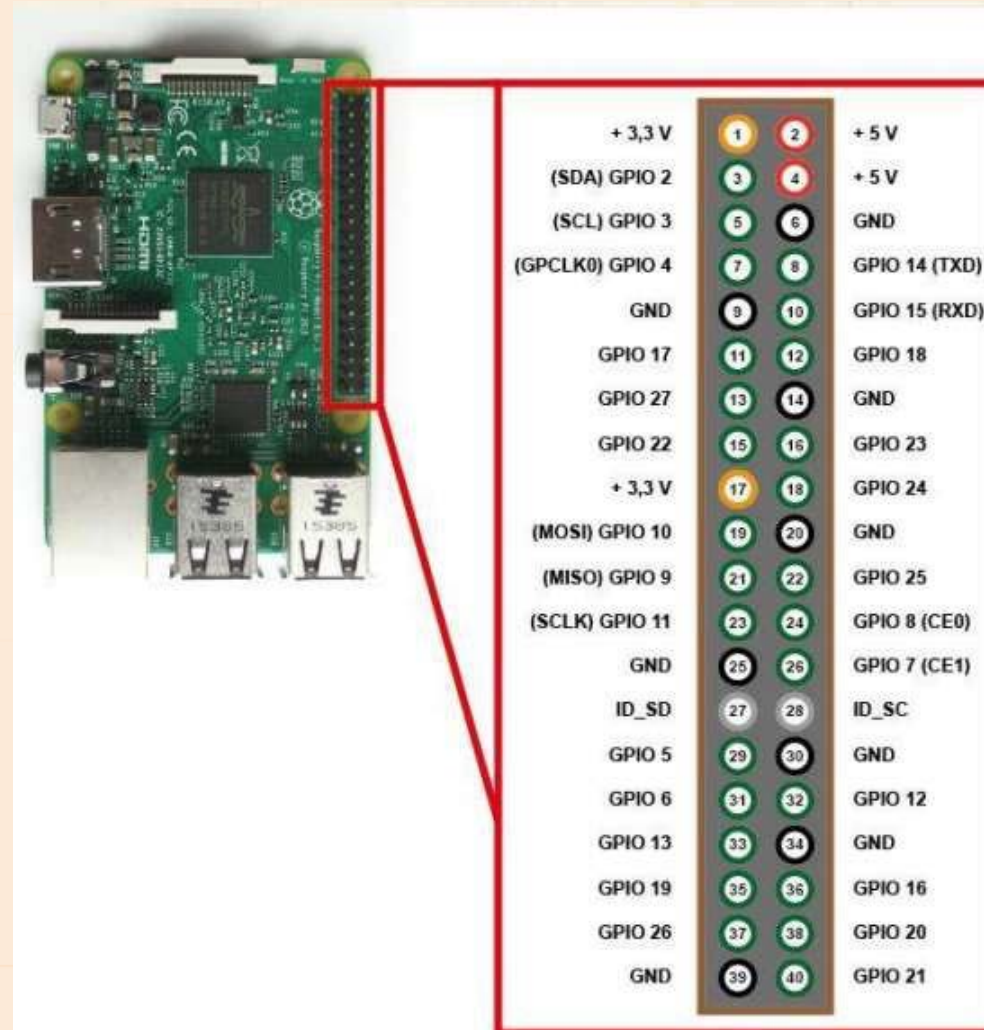


溫溼度感測器

- DHT11 溫溼度感測器
 - ◆ 溫度: $0 \sim 50^{\circ}\text{C}$, 誤差 $\pm 2^{\circ}\text{C}$
 - ◆ 濕度: $20 \sim 90\%$, 誤差 $\pm 5\%$
 - ◆ 使用三個腳位: Data , VCC , GND (out 、 + 、 -)
 - ◆ Data 腳位統一連接到 RPi 板上的 GPIO4 (Pin 7)
 - ◆ VCC 連接到 RPi 板上的 3.3 V 位置
 - ◆ GND \rightarrow 接地



腳位參考圖



下載本次實驗函式庫

- DHT11
 - ◆ git clone https://github.com/adafruit/Adafruit_Python_DHT.git
 - ◆ cd Adafruit_Python_DHT
 - ◆ sudo python setup.py install
- GPIO
 - ◆ sudo pip install rpi.gpio

溫溼度感測器功能測試

- 執行函式庫提供的測試檔
 - ◆ `cd Adafruit_Python_DHT/examples`
 - ◆ `sudo ./AdafruitDHT.py 11 4`
 - 11 為 DHT11 (也有 22 的型號)
 - 4 為 GPIO 4 (也就是Pin 7)

```
pi@raspberrypi ~ $ cd Adafruit_Python_DHT/examples/  
pi@raspberrypi ~/Adafruit_Python_DHT/examples $ sudo ./AdafruitDHT.py 11 4  
Temp=26.0* Humidity=37.0%
```

※若測試結果有任何錯誤或是無結果請先自行
檢查溫濕度計模組是否有接線錯誤，待確認後
再跟助教反應需更換材料或其他處理

Code 解釋

```
1  import Adafruit_DHT
2
3  # Setup DHT11
4  sensor_args = {'11' : Adafruit_DHT.DHT11,
5                 '22': Adafruit_DHT.DHT22,
6                 '2302': Adafruit_DHT.AM2302}
7
8  sensor = sensor_args['11']
9
10 # GPIO#, ex: GPIO4 = Pin7
11 gpio = 4
12
13 #Get humidity & temperature
14 humidity, temperature = Adafruit_DHT.read_retry(sensor, gpio)
15 humidity = str(humidity)
16 temperature = str(temperature)
```

溫溼度感測器 替代方案

- 若溫溼度感測器故障，即執行測試檔未出現溫溼度資訊，可嘗試以下方法
- 導入 random 函式庫

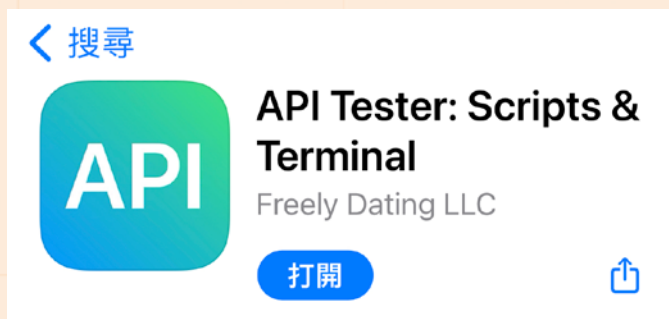
```
import random
```

- 以隨機數代替溫、濕度

```
humidity = random.randint(20, 30)  
temperature = random.randint(30, 40)
```

本次實驗 Demo

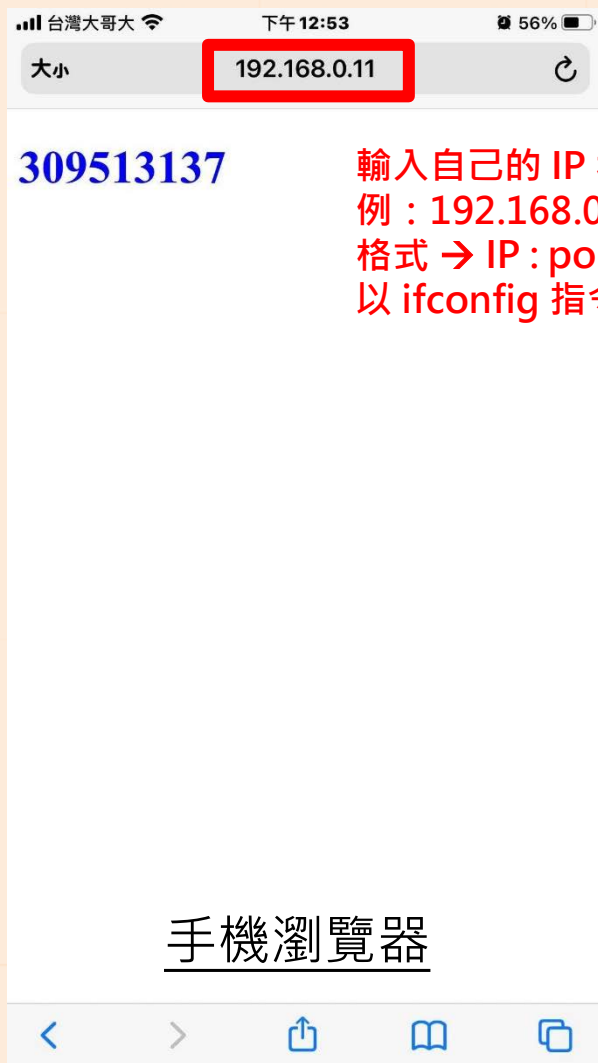
- Q1: 使用 GET Method 將自己的學號顯示出來
 - ◆ 根據助教提供的 **Lab3_1.py**，並自行完成
 - ◆ 請在自己手機或PC端利用網頁瀏覽器顯示出 HTTP Request 結果 (即學號)
 - ◆ 再使用手機 APP 顯示出 HTTP Request 結果
- ◆ IOS : 於 App Store 直接下載 API Tester，或搜尋 HTTP request 並下載相關 APP
- ◆ Android : 於 Google Play 商店直接下載 API Tester，或搜尋 HTTP request 並下載相關 APP



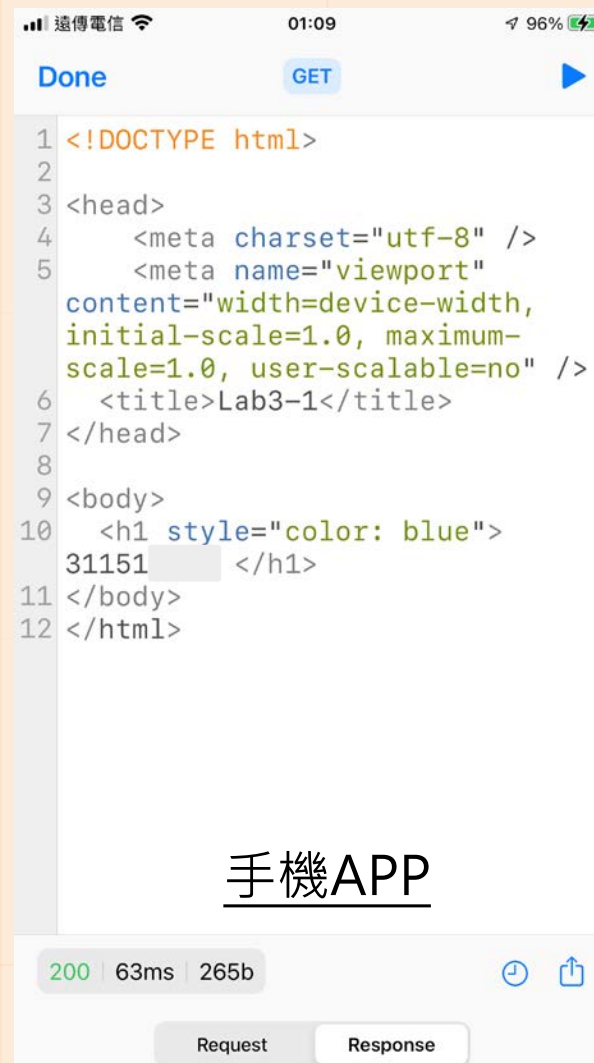
※本次實驗使用 **private IP** 來讓手機與 **Flask Web Service** 互相連線，所以需要讓手機與 **Rpi 板**同時連接同一個無線網路分享器 (AP)

本次實驗 Demo

Q1: 使用GET Method 將自己的學號顯示出來



輸入自己的 IP 和 port number
例：192.168.0.11:9808
格式 → IP : port
以 ifconfig 指令查詢 Rpi IP位址



手機APP

※本次實驗使用 private IP 來讓手機與 Flask Web Service 互相連線，所以需要讓手機與 Rpi 板同時連接同一個無線網路分享器 (AP)

本次實驗 Demo

● Q2: 完成一個能查閱 RPi 溫溼度模組數值 REST API

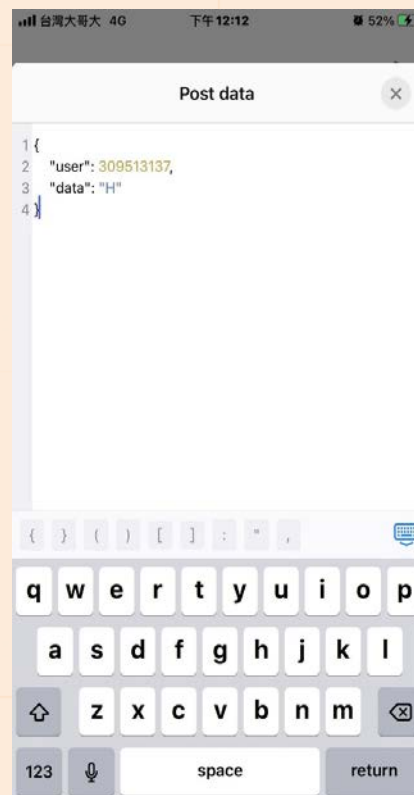
- ◆ 根據助教提供的 **Lab3-2.py**，並自行完成
- ◆ 使用者須使用 POST Request 提供使用者的**學號**與查詢**溫度**或**濕度**
- ◆ Json 格式 (Post data)

{

"user" : "309513137",

"data" : "H"

}



※本次實驗使用 private IP 來讓手機與 Flask Web Service 互相連線，所以需要讓手機與 Rpi 板同時連接同一個無線網路分享器 (AP)

本次實驗 Demo

- Q2: 完成一個能查閱 RPi 溫溼度模組數值 REST API
 - ◆ GET Method 部分已經完成，請參考 POST 部分完成剩下的 POST Method
 - GET Method 的內容有關於 POST Method 的實作提示
 - POST Method 溫溼度選擇以 T、H 代表即可，實作內容需包含下列4種情形：
 1. 如果傳入 **T**，應回傳 **溫度**、**學號**及指定格式的網頁內容
 2. 如果傳入 **H**，應回傳 **濕度**、**學號**及指定格式的網頁內容
 3. 使用者進行 POST Request 時並沒有提供 json 內容時，應回傳 **Status Code = 400** 的網頁內容
 4. 如果傳入非 H、T 變數時，應回傳 **Status Code = 400** 的網頁內容

本次實驗 Demo

- Q2 會使用到 jsonify 、 make_response 兩個 Flask 函式，使用方式請參考 Flask 函式，請點擊下列 API 網址參考使用方法：1. [jsonify](#) 2. [make_response](#)
- jsonify (**資料**)
 - ◆ 將資料轉換為 json 格式
- return make_response (**欲回傳的內容**, status code)
 - ◆ 回傳指定內容及狀態碼

本次實驗 Demo

- 若出現下列錯誤訊息，請執行以下步驟以排除編碼問題

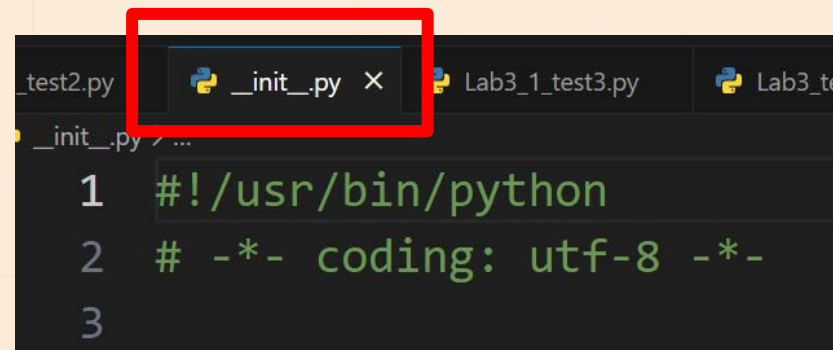
```
pi@raspberrypi:~/exp/lab_3 $ sudo python Lab3_2_test1.py
Traceback (most recent call last):
  File "Lab3_2_test1.py", line 4, in <module>
    from flask_restful import Api
  File "/home/pi/exp/lab_3/flask-restful/flask_restful/__init__.py", line 43
SyntaxError: Non-ASCII character '\xe2' in file /home/pi/exp/lab_3/flask-restful/flask_restful/__init__.py on line 44, but no encoding declared; see http://python.org/dev/peps/pep-0263/ for details
```

- 於執行 `git clone https://github.com/flask-restful/flask-restful.git` 的路徑

1. 輸入 `cd flask-restful/flask_restful`
2. 輸入 `nano __init__.py`
3. 務必於 `__init__.py` 最上方加上以下兩行註解：

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
```

- 詳情可參考[此網站](#)



```
_test2.py  _init_.py  Lab3_1_test3.py  Lab3_te
__init__.py / ...
1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3
```

本次實驗 Demo

1. 溫度

```
Post Data
1 {
2   "user": "31151",
3   "data": "T"
4 }
```

```
Done POST
1 {
2   "message": "Hi, 31151 the
3   current temperature is 29.0 °C"
}
```

200 | 808ms | 72b

Request Response

本次實驗 Demo

2. 濕度

```
Post Data ×
1 {
2   "data": "H",
3   "user": "31151"
4 }
```

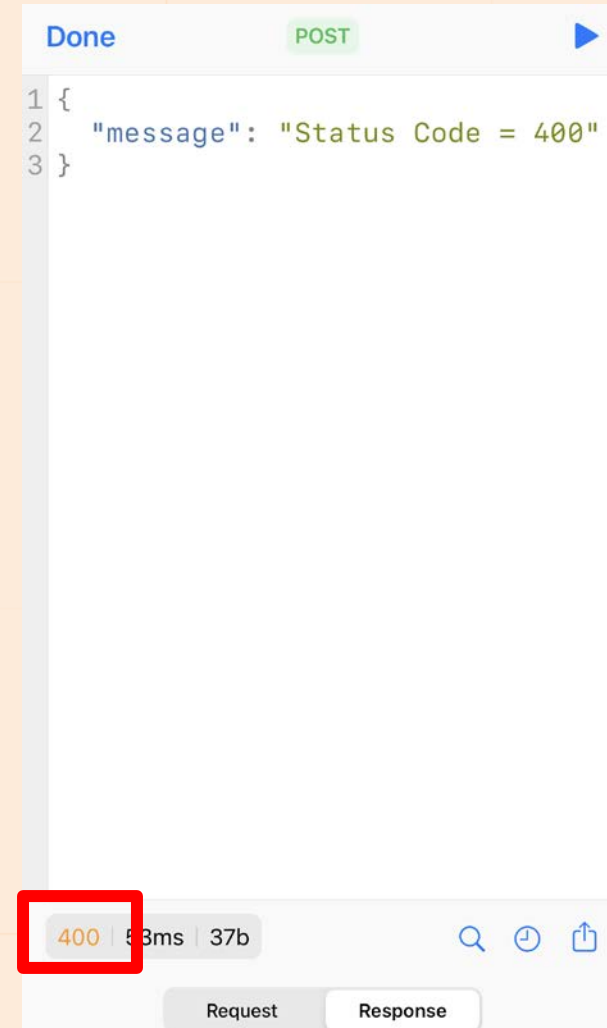
```
Done POST ▶
1 {
2   "message": "Hi, 31151 the
3   current humidity is 77.0 %"
```

200 | 705ms | 64b

Request Response

本次實驗 Demo

3. POST 時未提供 json 內容



本次實驗 Demo

4. 非溫度或濕度

```
Post Data
1 {
2   "data": "A",
3   "user": "31151"
4 }
```

```
Done POST
1 {
2   "message": "Status Code = 400"
3 }
```

400 68ms 37b

Request Response

本次結報內容

- 1. 請附上本次實驗 Q1 的手機 (或PC端) 瀏覽器截圖、APP 內容的截圖，以及 Q2 APP 內容的截圖，並詳細說明為何在 Q1 中兩種 Demo 方式顯示的文字內容不太一樣。另外，在 PC 瀏覽器中進行何項操作可使兩者顯示相同文字內容 (即 APP 中的顯示方式)？
- 2. 本次實驗內容 Q1，在手機端在進行 HTTP Request 時，Flask Web Service 會產生下列的訊息，請問紅色方框標示的數字分別代表著什麼意思？，請詳細說明。

```
192.168.0.104 - - [18/Oct/2023 13:11:17] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:19] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:19] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:20] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:20] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:21] "GET / HTTP/1.1" 200 -
192.168.0.104 - - [18/Oct/2023 13:11:32] "GET /pages HTTP/1.1" 404 -
192.168.0.104 - - [18/Oct/2023 13:11:36] "GET /pages HTTP/1.1" 404 -
192.168.0.104 - - [18/Oct/2023 13:11:37] "GET /pages HTTP/1.1" 404 -
```

本次結報內容

- 3. 若將實驗 Q1 中手機瀏覽器的網址內容更改成 **192.168.xxx.xxx:9808/pages** 並提交 GET Request 至 Flask Web Service 會產生什麼樣的訊息？請詳細說明原因。
(請附上網頁瀏覽器截圖 與 Flask Web Service [即Rpi終端機] 訊息截圖)
另外，請提供解決方法，於該網址恢復至顯示學號的頁面。
- 4. REST API 有哪些應用？(愈詳細且創新分數越高)
- 5. 本次實驗心得，你學到了什麼東西？

評分標準 & 注意事項

- 出席 30 %
- Demo 30 %
- 結報 40 %
- 請繳交 .pdf 檔，檔名為 學號_姓名_Labx.pdf
- 結報遲交一天，分數扣10%，以此類推

Reference

- <https://ithelp.ithome.com.tw/articles/10191925>
- <https://medium.com/%E4%B8%80%E5%80%8B%E4%BA%BA%E7%9A%84%E6%96%87%E8%97%9D%E5%BE%A9%E8%88%88/python-flask-rest-api%E7%AD%86%E8%A8%98-869c3d2fee3>
- https://ithelp.ithome.com.tw/articles/10202099?fbclid=IwAR1OSUaSK3yJj2FyyVCJQjfWtA_ucA8rPvpryamqs2E2QgnoQHwJ6nmuPQU
- <https://ithelp.ithome.com.tw/articles/10222132>
- <https://ithelp.ithome.com.tw/articles/10157431>