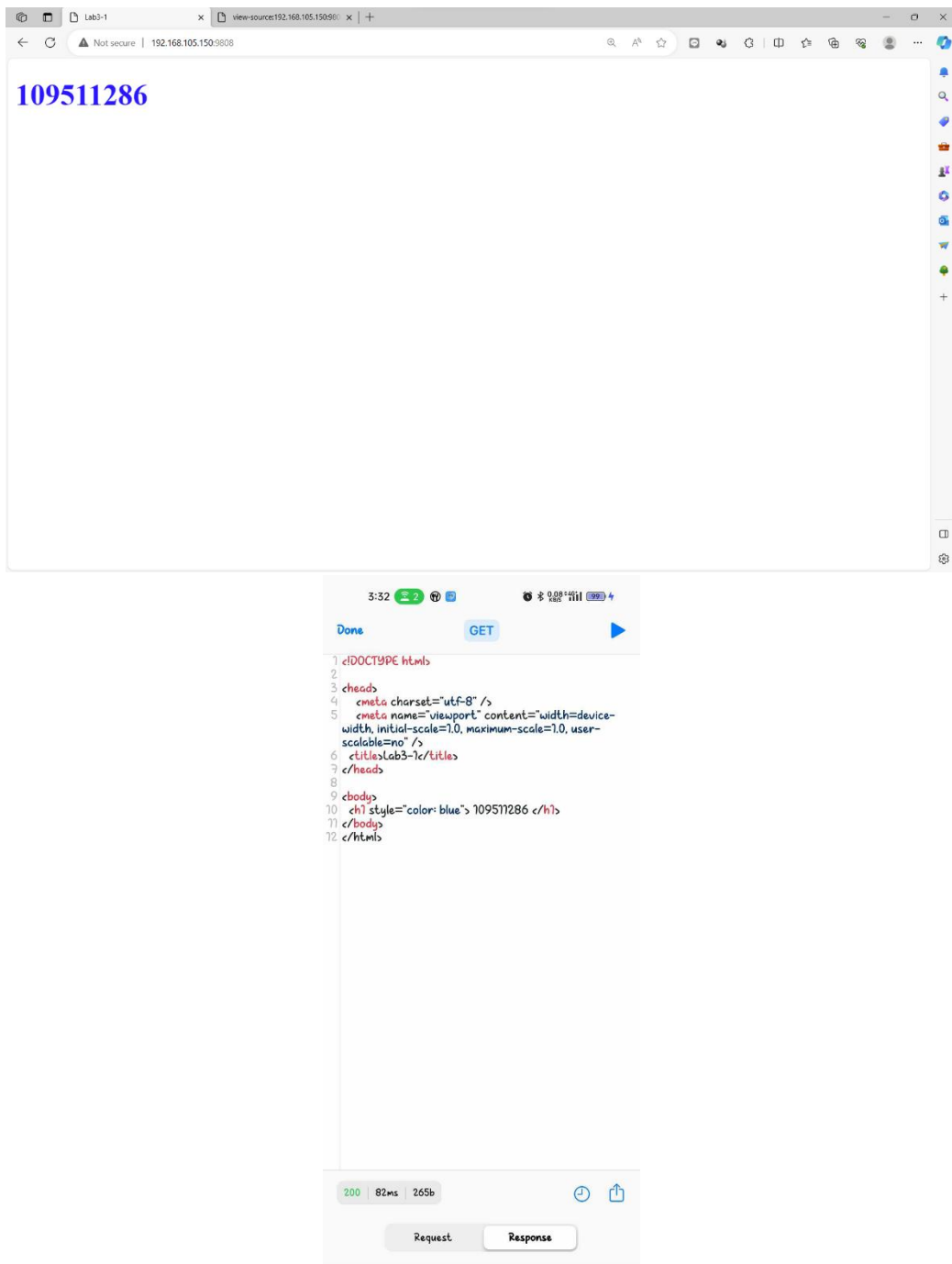


通訊網路實驗

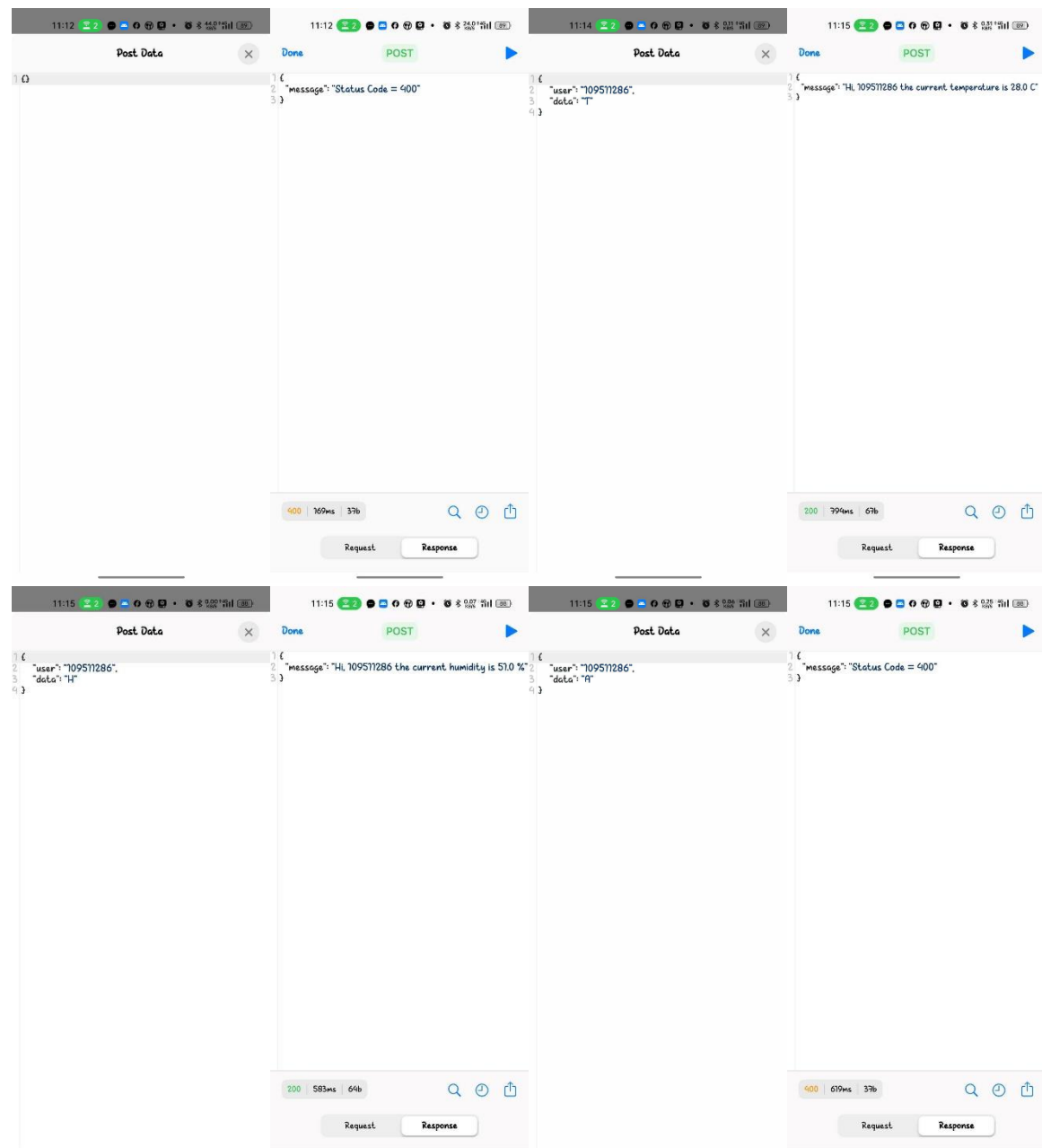
Lab 3 REST API 結報

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)？

Q1：

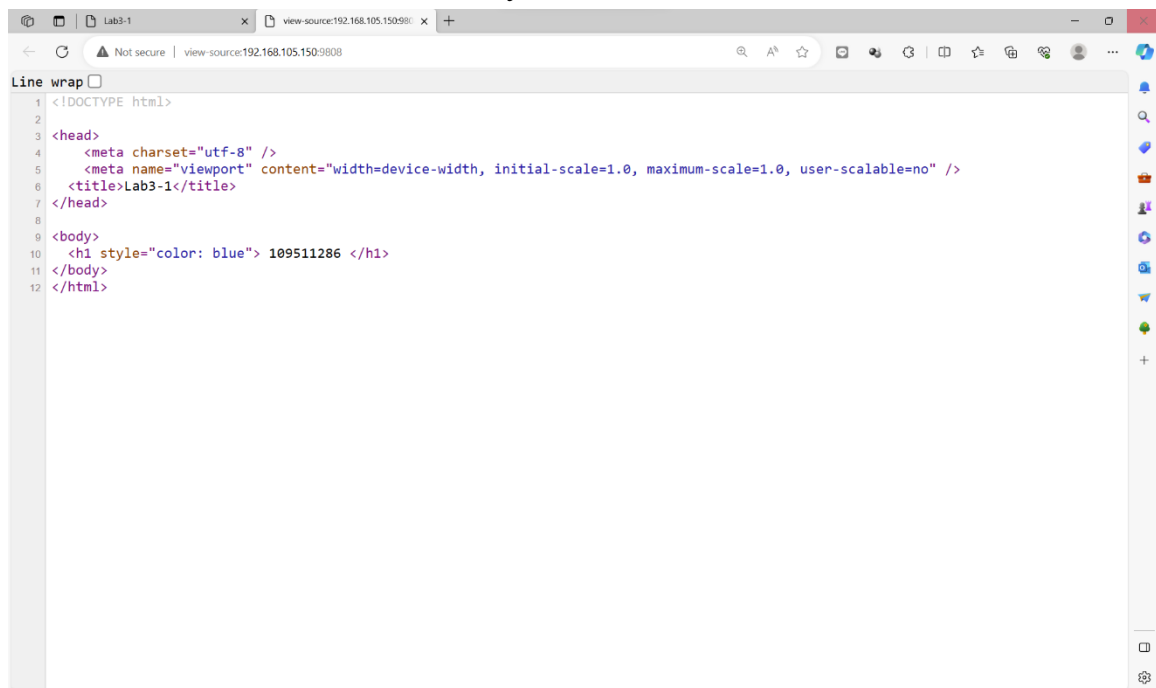


Q2 :



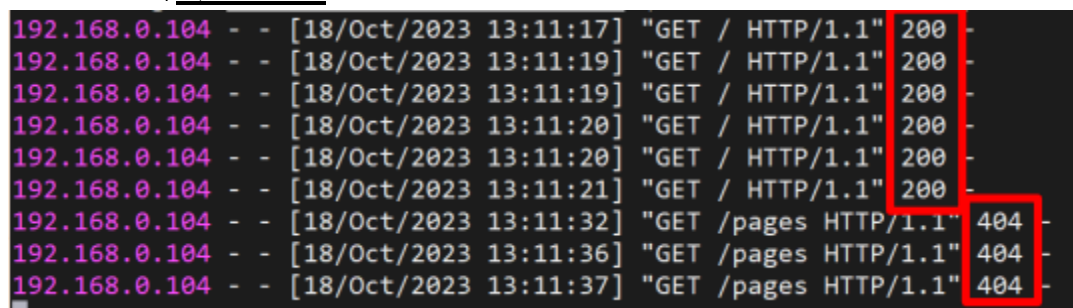
Q1 兩種 demo 方式出來的文字內容不相同的原因為 Rpi 板回傳的是一個 HTML 模板加上學號。HTML 會告訴瀏覽器該如何呈現該網頁。因此，瀏覽器上會呈現 HTML 檔中<body>部分的內容（style=" color : blue" 的學號）。使用 APP demo 時，則會顯示出 HTML 檔中全部的 code 。

在 PC 瀏覽器中使用 shortcut key (Ctrl + U) 就可顯示 HTML 原代碼。



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no" />
    <title>Lab3-1</title>
  </head>
  <body>
    <h1 style="color: blue"> 109511286 </h1>
  </body>
</html>
```

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 -
```

紅色方框標示的數字是 HTTP 狀態碼，用來表示 HTTP 請求的結果。

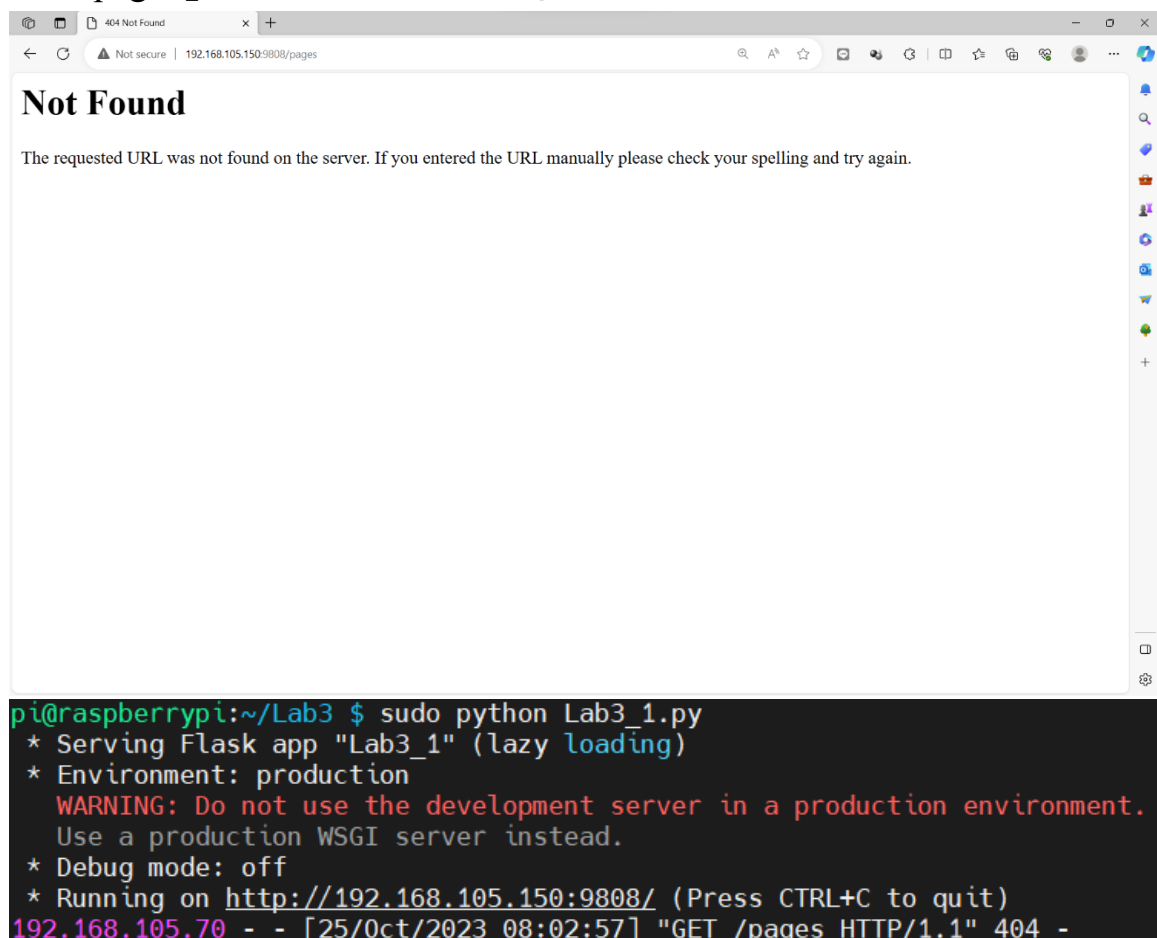
- HTTP 200 OK:
"200 OK" 表示請求成功，伺服器已成功處理請求，並返回所需的數據或內容。
- HTTP 404 Not Found:
"404 Not Found" 表示所請求的資源或頁面未找到。這通常意味著伺服器無法找到任何與請求 URI 相符的內容。

3. 若將實驗 Q1 中手機瀏覽器的網址內容更改成

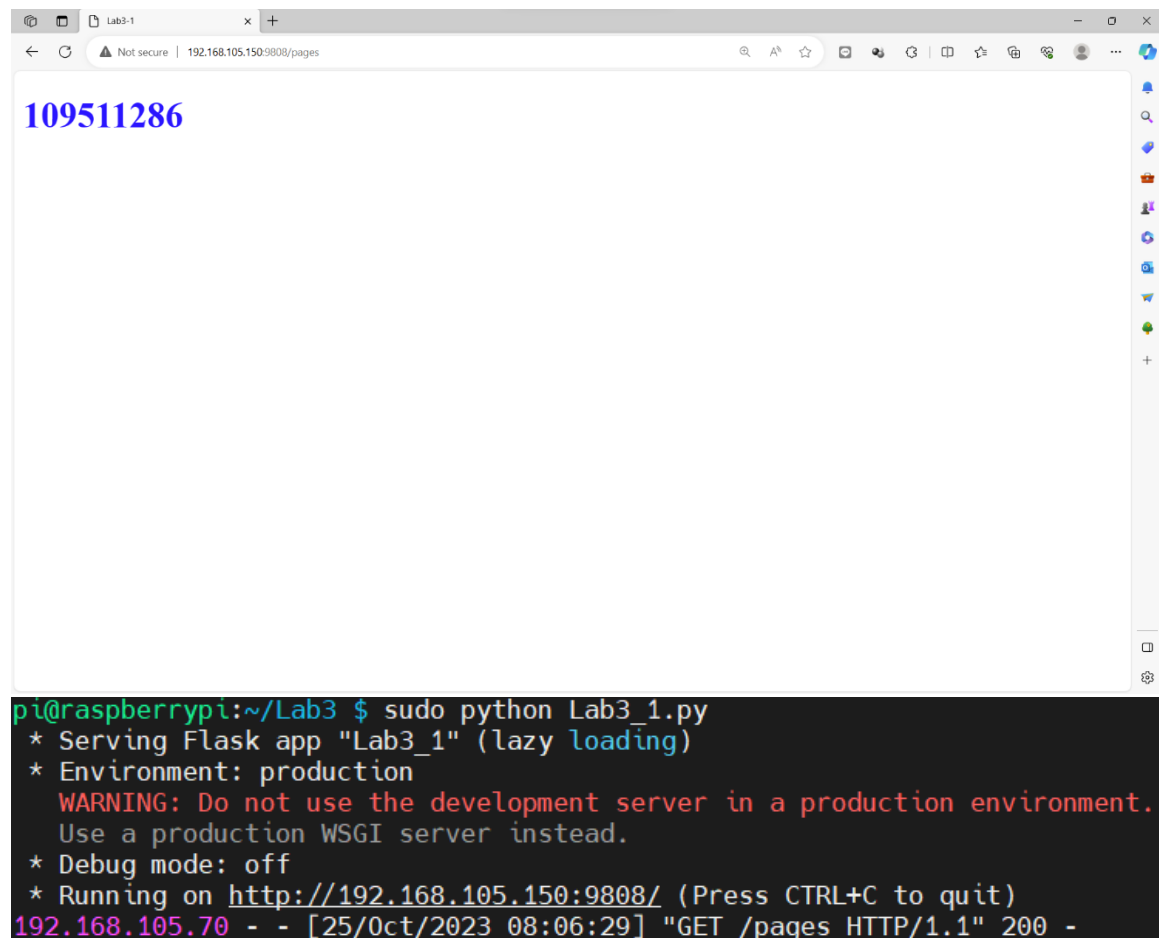
192.168.xxx.xxx:9808/pages 並提交 GET Request 至 Flask Web Service 會產生什麼樣的訊息? 請詳細說明原因。(請附上網頁瀏覽器截圖 與 Flask Web Service [即 Rpi 終端機] 訊息截圖)

另外，請提供解決方法，於該網址恢復至顯示學號的頁面。

在 Lab3_1.py 中，`@app.route('/')` 的後面所接的是 index function。因此，當連接到「/」的時候，路由就知道要執行後面的 function 了。當瀏覽器的網址內容更改成 192.168.105.150:9808/pages 時，由於主網域下並無「/pages」這個網址，所以瀏覽器會顯示 404 Not Found。



解決方法：在 Lab3_1.py 中再增加一個 `@app.route('/pages')` 且後面接 pages function（與 `index()` 內容相同），該網址將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)

- REST API 可用於整合不同社交媒體平台，例如 Facebook、Twitter 和 Instagram。這使得用戶可以在一個應用程序中查看和發布跨不同平台的內容。
- REST API 可用於連接智能家居設備，例如智能燈、智能恆溫器和智能門鎖。用戶可以透過應用程序控制和監視這些設備，並建立自動化場景。
- 電子商務平台如 Amazon 使用 REST API 來允許第三方開發者建立電子商務應用。這些 API 可以實現產品目錄、購物車、支付處理和訂單管理功能。這使得用戶能夠購買產品並管理其訂單。

- 地圖應用程序如 Google Maps 使用 REST API 來提供地理定位、路線導航、地理資訊和位置數據。開發者可以使用這些 API 來創建地圖應用、GPS 導航和位置追蹤應用。
- 醫療應用程序使用 REST API 來實現醫療記錄管理、預約掛號、處方藥物信息、電子病歷共享和遠程診斷。這有助於改進醫療服務的可及性和效率，使用戶更容易訪問和管理其健康信息。
- 學校和教育機構可以使用 REST API 來提供在線學習資源、學生信息系統、課程管理和教學工具。
- 運輸和物流公司使用 REST API 來實現貨運追蹤、路線優化、車隊管理和配送。
- 區塊鏈平台如 Ethereum 使用 REST API 來訪問區塊鏈上的智能合同。這使得開發者能夠創建去中心化應用程式，實現不需要中間人的交易。
- 遊戲開發者可以使用 REST API 來實現多人遊戲、排行榜、成就和虛擬物品交易。這擴展了遊戲的互動性。

5. 本次實驗心得，你學到了什麼東西？

API 是應用程式介面（Application Programming Interface），它定義程式之間的互動，以及可以進行的呼叫或請求的種類，如何進行呼叫或發出請求，應使用的資料格式，應遵循的慣例等。

RESTful 最常使用在各種 Web Service 中，透過 URI 的方式，由客戶端發送要求，伺服器再回傳資料。

- 要求資料時：使用 URI，將資料放在 HTTP 的 BODY
- 返回資料時：可使用 HTML、XML、JSON 或其他各種格式

由於 RESTful 的特性，讓呼叫 API 非常便利，通常制定好變數後，幾乎就可以知道怎麼呼叫了，無狀態的特性也讓開發時省去許多麻煩。總結一句：看名詞，就知道要存取什麼內容、看使用的方法，就知道要做什麼處理、看結果的狀態碼，就知道成功或失敗。