

程式設計（二）

期末專題書面報告



學號：410410032、410410034、410410044

組員：溫旻軒、林士恆、張証傑

魔女食堂

目錄:

(一):

呈現手法----- (1)

(二):

程式架構設計----- (2)

(三):

程式組成細節----- (6)

(四):

進度紀錄----- (28)

(五):

參考資料----- (30)

(六):

後記----- (31)

(一):呈現手法

•手法選擇

在規劃時期我們就決定要建構一個有顧客系統、地圖功能、還有極高機動性的 App，此我們就必須有資料庫、網頁、和方便的機器人(discord,line bot...)

•網頁

為了讓大家都成為魔女食堂的一員，我們選擇網路來貼近魔女們和使用者的距離，而網頁化的設計不僅能讓使用者清楚的看到地圖，更能提供直覺化的顧客體驗，簡單的表單回傳、分頁切換就能觸發各式各樣的功能。

•Heroku-

因為我們需要資料庫和網頁，而 heroku 能同時提供我們這兩個功能，且雲端資料庫比起存在本機更為安全，版本控式的功能也能讓我們在製作時能有更大的彈性。

•line bot-

line 無疑是現在最普及的通訊軟體了，我們相信在大家傳送訊息之餘，能利用小魔女 app 尋找附近好吃的食物來品嚐。

(二):程式架構設計

•檔案配置

```
ccu-111-final
|   Procfile
|   ccu-111-final.py
|   requirements.txt
|   runtime.txt
|
|   g
|   |   aqua.png
|   |   cat_bad.png
|   |   logo2.png
|   |---smoking.png
|
|---templates
|   account.html
|   base.html
|   home.html
|   recommend.html
|---record.html
```

•程式架構

Python




```
| database
| | history_eat    #使用者歷史紀錄
| | restaurant_data  #使用者給店家的評分
| | restaurant_id   #店家的restaurant_id
| |---user_data     #使用者相關資訊
|
| website
| | account.html    #登入頁面
| | base.html       #網頁模板
| | home.html       #首頁
| | recommend.html   #推薦店家頁面
| |---record.html    #歷史紀錄
|
|---line bot
    |---finalproject
```

• 資料庫

history_eat

Data Output		Explain	Notifications	Messages	
	<div><div><div><div></div></div><div>user_name</div><div>text</div></div></div>	<div><div><div><div></div></div><div>restaurant_name</div><div>text</div></div></div>	<div><div><div><div></div></div><div>rank</div><div>bigint</div></div></div>	<div><div><div><div></div></div><div>day</div><div>timestamp with time zone</div></div></div>	
1	2	真香牛肉麵	5	2022-06-03 02:09:30+00	<div></div>
2	2	真香牛肉麵	5	2022-06-03 02:13:12+00	
3	2	真香牛肉麵	3	2022-06-03 02:20:51+00	
4	123	北門滷味	6	2022-06-03 22:23:07+00	
5	2	小妞滷味	4	2022-06-03 22:45:23+00	
6	2	異香屋平價美食	6	2022-06-03 22:46:24+00	
7	2	舜御食坊	6	2022-06-03 22:47:27+00	
8	2	永富川味麵館	1	2022-06-03 22:48:08+00	
9	2	辰品日式料理	2	2022-06-03 22:48:39+00	
10	2	古早味伙食工坊	6	2022-06-03 22:50:03+00	

restaurant_id

Data Output	Explain	Notifications	Messages
	 name text	 id text	
1	民雄照記芋圓	ChIJ35ze1929bjQRNLx34fBv2nw	
2	北門滷味	ChIJ-S7lkt69bjQRlonmkjxRyZ0	
3	真香牛肉麵	ChIJ9xgHvd69bjQRVJ5aFQRi-Lk	
4	允好食堂	ChIJpZj-V3C-bjQRTM640MHvZz4	
5	異香屋平價美食	ChIJHYXOLty9bjQReo-n9LRSgh0	
6	小妞滷味	ChIJJ9eExry9bjQRGGQi69-lkug	
7	舜御食坊	ChIJleWjVJq_bjQRaggcQjAsSm4	
8	永富川味麵館	ChIJx_vfn-e9bjQRbeMSoUyLYAY	
9	辰品日式料理	ChIJybPX__m9bjQRmvcBANtcuzl	
10	民雄廟口破烤雞排	ChIJVVVVVRW-bjQRuaZ9WaK86vg	

restaurant_data

Data Output		Explain	Notifications	Messages				
	<div><div>user_name</div><div>[PK] text</div></div>	<div><div>Abao</div><div>house民雄東榮店</div></div> <div>bigint</div>	<div><div>民雄</div><div>照記芋圓</div></div> <div>bigint</div>	<div><div>北門</div><div>滷味</div></div> <div>bigint</div>	<div><div>真香</div><div>牛肉麵</div></div> <div>bigint</div>	<div><div>允好</div><div>食堂</div></div> <div>bigint</div>	<div><div>異香</div><div>屋平價美食</div></div> <div>bigint</div>	<div><div>小妞</div><div>滷味</div></div> <div>bigint</div>
1	1	[null]	1	3	8	[null]	[null]	[null]
2	123	[null]	1	6	2	3	[null]	[null]
3	2	[null]	9	4	10	10	6	[null]
4	25	[null]	[null]	[null]	[null]	[null]	[null]	[null]
5	26	[null]	[null]	[null]	[null]	[null]	[null]	[null]
6	27	[null]	[null]	[null]	[null]	[null]	[null]	[null]
7	3	[null]	[null]	2	2	[null]	[null]	[null]
8	30	[null]	10	[null]	[null]	[null]	[null]	[null]
9	4	[null]	6	3	[null]	[null]	[null]	[null]
10	40	[null]	[null]	[null]	[null]	[null]	[null]	[null]

user_data

	Data Output	Explain	Notifications	Messages
	<div><div>▲</div><div>user_name</div><div>text</div></div>	<div><div>user_password</div><div>text</div></div>	<div><div>favorite</div><div>text</div></div>	<div><div>line_userid</div><div>text</div></div>
1	5	456	[null]	[null]
2	6	456	[null]	[null]
3	123	456	[null]	[null]
4	4	11	[null]	U05740e333d1bd05e4e0bedb0b7c94fe0
5	3	456	雞排	[null]
6	1	123	雞排	[null]
7	7	456	[null]	[null]
8	8	456	[null]	[null]
9	10	10	[null]	[null]
10	11	11	[null]	[null]

(三). 程式組成細節

•python

python 程式碼主要由 7 個 route 組成。分別是首頁、找附近美食、歷史紀錄、登入、新增帳號、登出、推薦系統。

•首頁

記得要先確認登入方式，因為 post 要處理表單傳過來的資料

```
@app.route('/', methods=['GET','POST'])
def home():
    #檢查一下請求的方式是什麼
    if request.method == 'GET':
        login_status,login_account=check_login()
        return render_template("home.html",login_status=login_status)
    else:
```

如果是 Post 則要使用 request.value[""]抓取資料同時也把資料庫連線打開，我們要新增飲食紀錄

```
restaurant_name=request.values['eat']
rank=request.values['rate']

#資料庫連線
conn = psycopg2.connect(database_url,sslmode='require')
cursor=conn.cursor()
```

新增(insert)歷史紀錄(使用者名稱、餐廳名、評分、時間)

```
sql="INSERT INTO history_eat(user_name,restaurant_name,rank,day) VALUES(%s,%s,%s,%s)"
cursor.execute(sql,(user_name,restaurant_name,rank,time_text))
conn.commit()
```


先抓取(select)資料庫內所有使用者以及餐廳，如果表單中的使用者及餐廳都存在資料庫內的話，更新(update)評價

```
#找出所有的使用者名稱
sql="SELECT user_name FROM restaurant_data "
cursor.execute(sql)
all_user_name=cursor.fetchall()

#找出所有餐廳
sql="SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='restaurant_data'"
cursor.execute(sql)
all_restaurant_name=cursor.fetchall()

if (user_name in all_user_name and restaurant_name in all_restaurant_name ):
    sql="UPDATE restaurant_data SET {restaurant_name} = {rank} WHERE user_name='{user_name}'"\
        .format(restaurant_name=str(restaurant_name),rank=rank,user_name=user_name)
    cursor.execute(sql)
    conn.commit()
```

•找附近美食

利用 google 提供的 place api 中的 nearby search 尋找附近的店家，同時我們可以設定經緯度、關鍵字

```
#尋找附近的店家
url = "https://maps.googleapis.com/maps/api/place/nearbysearch/json?location={latitude}, {longitude}&radius=2000\
    .format(latitude=lat,longitude=lng,keyword=eat)
payload={}
headers = {}

#尋找附近店家
response = requests.request("GET", url, headers=headers, data=payload)
response.json()
res=json.loads(response.text)
```

但 nearby search 給的資訊非常少，於是我們利用 place api 中 place detail 來獲取更多資訊(照片、官網等)

```

url = "https://maps.googleapis.com/maps/api/place/details/json?place_id={place_id}&language=zh-Tw&
.format(place_id=i['place_id'])
payload={}
headers = {}

#抓取餐廳資訊並改成json格式
ress = requests.request("GET", url, headers=headers, data=payload)
ress.json()
detail=json.loads(ress.text)

```

而我們在抓取到的照片其實是一串 `photo_reference`，必須再用 `place api` 中 `place photo` 將實際照片找出

```

for i in range(5,8):
    #如果圖片不存在則利用指定圖片代替
    if(data_web[i]=="無資料"):
        data_web[i]=="https://media.istockphoto.com/vectors/open-source-concept-trendy-icon-
    else:
        data_web[i]="https://maps.googleapis.com/maps/api/place/photo?maxwidth=300&maxheight=

```

●歷史紀錄

在這邊我們利用排序(`order by`)和找前面幾筆(`limit`)來找出最近十筆評分資料。

```

#這邊試著把離現在最近的十筆資料抓出來
conn = psycopg2.connect(database_url,sslmode='require')
cursor=conn.cursor()
sql="SELECT * FROM history_eat WHERE user_name='{user_name}' ORDER BY day DESC LIMIT 10 ".format(user_name=login_account)
cursor.execute(sql)
all_data=cursor.fetchall()

```

●登入

登入系統中我們分成兩步驟確定使用者身分，先看 `id` 是否存在，如果存在再進一步 `select` 密碼來確認使用者是否本人，避免查詢多餘的資料

```
conn = psycopg2.connect(database_url,sslmode='require')
cursor=conn.cursor()
sql="SELECT user_name FROM user_data"
cursor.execute(sql)
all_user_name=cursor.fetchall()
```

```
if(user_name in all_user_name):
    sql="SELECT user_password FROM user_data WHERE user_name='{user_id}'".format(user_id=user_name)
    cursor.execute(sql)
    password=cursor.fetchall()
```

確定是合法的登入後，將使用者資訊寫在 cookie 中的 session 裡，之後進入其他分頁可以直接 0 抓取使用者名稱來操作

```
if(user_password==password[0]):
    #抓取使用者名稱
    session['session_password']=user_name
    return render_template("home.html",login_status="yes")
else:
    return render_template("account.html",login_status="error")
```

•新增帳號

線看看此帳號是否存在，如果已經存在則失敗，不存在則用

insert 新增

```
sql="SELECT user_name FROM user_data"
cursor.execute(sql)
all_user_name=cursor.fetchall()
```

```

#如果名稱已存在則更新失敗
if(user_name in all_user_name):
    cursor.close()
    conn.close()
    return render_template("account.html",login_status="error",aqua="bad")
#新的名稱則把資料記錄在user_data和restaurant_data裡
sql="INSERT INTO user_data(user_name,user_password) VALUES(%s,%s)"
val=(user_name,user_password)
cursor.execute(sql,val)
conn.commit()

```

•登出

因為我們是利用 cookie 的 session 來儲存登入狀態，所以登出時要記得清除 session

```

#登出系統
@app.route("/logout",methods=['GET','POST'])
def logout():
    session.clear()
    return render_template("account.html")

```

•推薦系統

利用協同過濾找出使用者可能會喜歡的食物。

先找出所有使用者的評分紀錄，再進一步找出最相似的 5 名使用者，最後找出最相近的五名使用者評分平均最高的 10 間餐廳再推薦給使用者。

而計算相似度的方式是採用餘弦相似度，利用公式

$$\text{Cos}=(a*b)/(|a|*|b|)$$

兩個 if 分別是計算其他顧客向量的 norm 和計算兩向量內積

```
for j in range(1,restaurant_len):
    #計算長度以及cos
    if(i[j]!=None):
        other_len+=int(i[j])*int(i[j])
    if(i[j]==None or target_data[j]==None):
        continue
    all_cos+=int(i[j])*int(target_data[j])
```

算完之後再加入記錄前五名資料的 list 中，如果人數已經滿了則比大小

```
#如果人數小於五就增加，大於則比較
if(len(max_cos)>=5):
    if(final_cos>min(max_cos)):
        switch=max_cos.index(min(max_cos))
        max_cos[switch]=final_cos
        max_name[switch]=i[0]
elif(other_len>0 and all_cos>0):
    max_cos.append(final_cos)
    max_name.append(i[0])
```

接著再利用 column_name from information_schema.columns 找出每個 column 對應到的餐廳名稱

```
sql="SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='restaurant_data'"
cursor.execute(sql)
all_restaurant_name=cursor.fetchall()
all_restaurant_name=[i[0] for i in all_restaurant_name]
```

接著簡單的比大小找出評分最高的五家餐廳，這時要去

restaurant_id 這個資料庫找出餐廳對應的 restaurant_id，再利用
place api 中 place detail 來獲取更多資訊(照片、官網等)

```
for name in best_rest_name:
    sql="SELECT id FROM restaurant_id WHERE name='{name}'".format(name=name)
    cursor.execute(sql)
    id=cursor.fetchall()
    rest_id.append(id[0][0])
```

```
for id in rest_id :
    url = "https://maps.googleapis.com/maps/api/place/details/json?place_id={place_id}".format(place_id=id)
    payload={}
    headers = {}
```


•line bot

透過 line bot 我們製作了以下功能，包括帳號登入、透過輸入位置資訊推薦選定類型的餐廳(Recommend)，以及紀錄(Record)等。

•帳號登入功能

利用 re 尋找是否有命令字串，當輸入的指令符合格式時
match != "None"，這裡我們使用 lstrip()將命令字元過濾掉

```
match = re.search(r'^([A|a]){1}(ccount:){1}', tmp_text)
```

```
tmp = tmp_text.lstrip('Aaccount')  
tmp = tmp.lstrip(':')
```

在和資料庫比對完帳號後，如果正確及回傳"success"，反之，則回傳"小魔女最討厭來路不明的怪叔叔了"，回傳則使用自動回覆訊息 reply_message()回傳。

```
if (str(tmp) in all_user_name):  
    #存在則新增line_userid  
    sql="UPDATE user_data SET line_userid = '{user_id}' WHERE user_name='{user_name}'"\br/>        .format(user_id=user_id,user_name=tmp)  
    cursor.execute(sql)  
    conn.commit()  
    return_text='success'  
else:  
    return_text='小魔女最討厭來路不明的怪叔叔了'
```

```
message = TextSendMessage(text = return_text)  
line_bot_api.reply_message(event.reply_token, message)
```

而我們則會把使用者的 line user_id 記錄在 user_data 的資料庫中，之後就利用這個來判斷 line 的使用者身分

•飲食偏好設定

初次使用時必續設定，不然會無法推薦餐廳

利用 re 尋找是否有命令字串，當輸入的指令符合格式時

recommend != "None"，這裡我們使用 lstrip()將命令字元過濾掉

```
recommend=re.search(r'^(\R|r){1}(ecommand:){1}', tmp_text)
```

```
tmp = tmp_text.lstrip('Rrecomend')  
tmp = tmp.lstrip(':')
```

如果成功設定則回傳"success"，回傳則使用自動回覆訊息

reply_message()回傳。

```
message = TextSendMessage(text = return_text)  
line_bot_api.reply_message(event.reply_token, message)
```

•輸入位置資訊的功能

如果輸入的訊息是位置資訊時，它會獲取那個位置的經緯度。

我們首先先設定一個抓位置的副函式，並利用

event.message.latitude 或 longitude 找出經緯度並回傳。

```
def message_location(event):  
    lat = event.message.latitude  
    lng = event.message.longitude  
    return lat,lng
```


•推薦功能

在回傳一個座標後我們利用"座標"+"飲食偏好"來找出附近相似的店家並回傳(同 python 介紹內容)，飲食偏好則用 line user_id 回去 user_data 資料庫找出

```
sql="SELECT favorite FROM user_data where line_userid='{user_id}' "\
    .format(user_id=user_id)
```

而我們呈現店家的方式則是用 CarouselTemplate

```
line_bot_api.reply_message( event.reply_token,TemplateSendMessage(
    alt_text='CarouselTemplate',
    template=CarouselTemplate(
        columns =[message_link(i) for i in data_for_line]
    )
))
```

•歷史紀錄

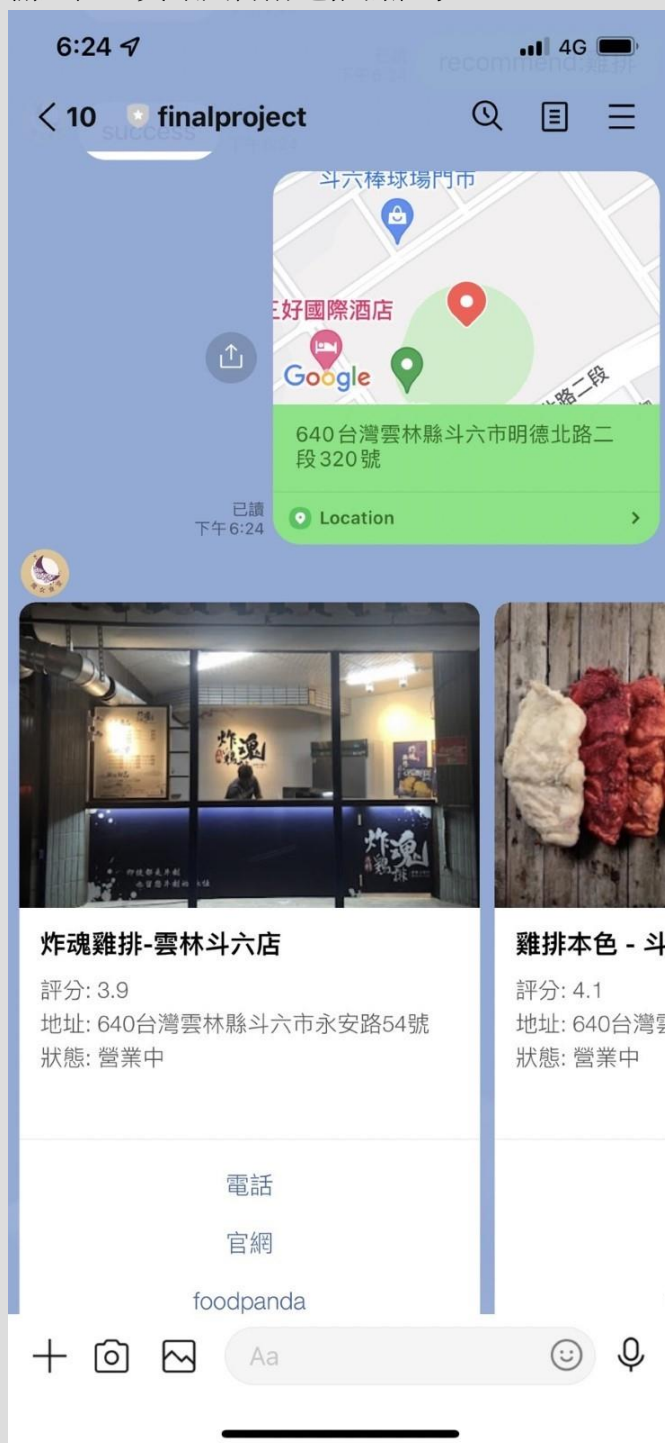
利用 line user_id 去 user_data 找相對應的帳號，再利用帳號去 history_eat 找出最近十筆評分紀錄(同 python 介紹內容) 回傳則使用自動回覆訊息 reply_message()回傳。

操作範例:

登入帳號選擇食物喜好成功範例:



輸入位置資訊回傳附近推薦店家:



• 網頁前端

account.html
base.html
home.html
recommend.html
record.html

因為四個網頁都有相同的部分，為了省去更改的麻煩，因此把重複的部分拿出來做成「模板」。

```
{%extends "base.html"%}
```

如此便能在每一個網頁引用此模板。

• 導覽列介紹

```
<nav class="navbar navbar-expand-sm bg-dark-pink navbar-dark">
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#collapsibleNavbar">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="collapsibleNavbar">
    <ul class="navbar-nav">
      <li class="nav-item">
        <a class="nav-link" href="/">首頁</a>
      </li>
      <li class="nav-item login-before">
        <a class="nav-link" href="account">登入/註冊</a>
      </li>
      <li class="nav-item login-after">
        <a class="nav-link" href="record">飲食紀錄</a>
      </li>
      <li class="nav-item login-after">
        <a class="nav-link" href="recommend">推薦</a>
      </li>
      <li class="nav-item dropdown login-after">
        <a class="nav-link dropdown-toggle" id="navbardrop" data-toggle="dropdown">
          使用者: {{login_account}}
        </a>
        <div class="dropdown-menu">
          <a class="dropdown-item" href="logout">登出</a>
        </div>
      </li>
    </ul>
  </div>
</nav>
```

這裡使用了 bootstrap 的 navbar，幫助我們快速做出導覽列的外型。而圖中框起來的部分為後端 python-flask 的功能，可以傳入變數，這裡則是使用者的名稱。

```
<div id="loginReady" style="display: none;">{{login_status}}</div>

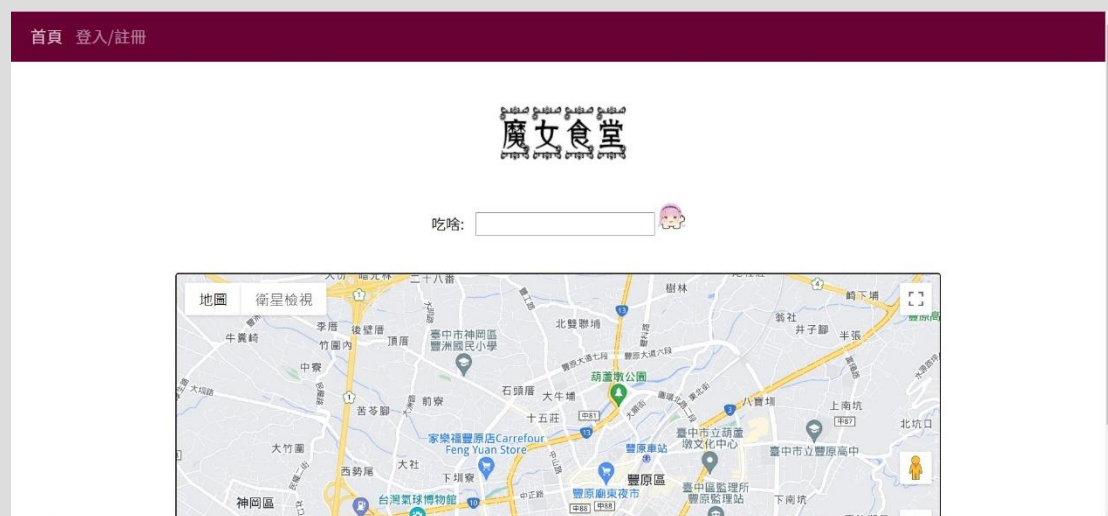
//detect login or not
if(document.querySelector('#loginReady').innerText == 'yes'){
  let ele = document.querySelector('.login-before');
  ele.style.display = 'none';
  let eles = document.querySelectorAll('.login-after');
  for(let i=0; i<eles.length; i++){
    eles[i].style.display = 'block';
  }
}
```

接著使用 div 標籤接收參數，並由 js 的 if 來判斷使用者是否已登入，若已登入，導覽列會顯示四個項目(左圖)，若未登入，則只會剩首頁及登入(右圖)。

首頁 飲食紀錄 推薦 使用者: ▼

首頁 登入/註冊

• 首頁



這是未登入前的樣子

```
//if login then show the rate-field
if(document.querySelector('#loginReady').innerText == 'yes'){
  map = document.querySelector('#map-field');
  map.style.width = '60%';
  rate = document.querySelector('#rate-field');
  rate.style.display = 'flex';
}
```

接著用 js 一樣地從後端拿變數，而我們的做法都統一為先將變數以{{變數名稱}}的形式傳入 html，再用 js 抓取標籤，如此就能利用裡面剛傳入的變數。若偵測到使用者已登入，則我們的首頁會變為：

魔女食堂

吃啥:



評分

餐廳名稱:

評分:

因為在登入後才能使用評分功能，因此我們在登入前的畫面將它隱藏，登入後就顯示出來。

```
<div style="margin-bottom: 40px; display: flex; justify-content: center;">
  <form action="nearby">
    <label for="eat" style="font-size: 18px; margin-right: 10px;">吃啥:</label>
    <input name="eat" id="eat" />
    <input type="hidden" id="now_lat" name="now_lat">
    <input type="hidden" id="now_lng" name="now_lng">
    <input type="image" alt="按我" src="/g/smoking.png" height="35px">
  </form>
</div>
```

吃啥:



使用者在填寫此表單後按下阿夸的頭，會把內容送到後端，接著後端會送出推薦資料給我們。

```
//generate the recommend-field by the input number
let input = document.querySelector('#input').innerText;
let array = input.split(',');
let runtime = array[0];
for(let i=0; i<runtime; i++){
  let j = i*8;
  let newdiv = document.createElement('div');
  newdiv.innerHTML = '<div class="rcd-field"><div class='
  document.querySelector('#new-rcd').append(newdiv);
}
```


並使用 js 將獲得的資料放入 html 中，成果如右圖。



而地圖則是利用 Google 提供的 GoogleMap API 自動抓取使用者的位置。

```

navigator.geolocation.getCurrentPosition(successCallback);
var msg='';
var now_lat='';
var now_lng='';
function successCallback(position){
    now_lat=position.coords.latitude;
    now_lng=position.coords.longitude;
    document.querySelector('#now_lat').value = now_lat;
    document.querySelector('#now_lng').value = now_lng;
    initMap();
}
function initMap() any
    var map = new google.maps.Map(document.getElementById("map") , {
        center: { lat: now_lat, lng: now_lng },//中心位置
        zoom: 14,
    });
    //-----下面是呼叫一個新marker-----
    var marker = new google.maps.Marker({
        position: { lat: now_lat, lng: now_lng }, //marker的放置位置
        map: map //這邊的map指的是第3行的map變數
    });
}

```

評分

餐廳名稱:

評分:



使用者可以在這裡填寫評分，送出後會記錄在線上資料庫內，就算登出後也會保有紀錄。

這裡是做成可以搜尋關鍵字的功能，由於餐廳數量太多，因此用 js 幫忙把字

```

//turn those shop-name into list, and put them in #restaurant
let shops_name = document.querySelector('#shop-name');
let shops = shops_name.innerHTML.split('\n').slice(1, -1);
let res = document.querySelector('#restaurant');
for(let i=0; i<shops.length; i++){
    let new_shop = document.createElement('option');
    new_shop.value = shops[i];
    res.append(new_shop);
}

```


餐廳名稱: 壽司

評分: 10

評分: 10

1

2

3

4

5

6

7

8

9

10

串處理成搜尋內容。

```
<label for="rate" style="font-size: 18px; margin-right: 10px;">評分:</label>
<select name="rate" id="rate">
  <option value="1">1</option>
  <option value="2">2</option>
  <option value="3">3</option>
  <option value="4">4</option>
  <option value="5">5</option>
  <option value="6">6</option>
  <option value="7">7</option>
  <option value="8">8</option>
  <option value="9">9</option>
  <option value="10" selected>10</option>
</select>
```

評分則是做成下拉式選單，提供 1-10 給使用者選擇。

• 評分紀錄

```
//generate the shop-field by the input number
let input = document.querySelector('#input').innerText;
let array = input.split(',');
let runtime = array[0];
for(let i=0; i<runtime; i++){
  let j = i*3;
  let newdiv = document.createElement('div');
  newdiv.innerHTML = '<div class="shop-field"><div class=
document.querySelector('#new-shop').append(newdiv);
}
```

用 js 來抓取後端傳過來的資料，並動態產生記錄用的格子。

首頁 飲食紀錄 推薦 使用者: ▾		
評分歷史紀錄		
時間: 2022-06-08 19:59:02+00:00	我的評分: 10	
餐廳名稱: 真香牛肉麵		
時間: 2022-06-08 19:58:12+00:00	我的評分: 4	
餐廳名稱: 北門滷味		
時間: 2022-06-08 19:36:35+00:00	我的評分: 1	
餐廳名稱: 北門滷味		
時間: 2022-06-08 18:56:56+00:00	我的評分: 10	
餐廳名稱: 允好食堂		
時間: 2022-06-08 18:14:41+00:00	我的評分: 3	
餐廳名稱: 真香牛肉麵		
時間: 2022-06-08 17:17:58+00:00		

記錄用格子範例 html:

```
<div class="shop-field">
  <div class="upper-field">
    | <span>時間: 我怎麼知道</span>
  </div>
  <div class="downer-field">
    | <span style="width: 85%;">餐廳名稱: 小魔女</span><span>我的評分: 滿分</span>
  </div>
</div>
```

```
//if no result
let no_result = document.querySelector('#no-result');
if(!(runtime>0)){
  no_result.style.display = 'flex'
}
```

並且判斷是否有歷史紀錄存在，若否，則顯示 no results，如下圖。

首頁 飲食紀錄 推薦 使用者: ▾		
評分歷史紀錄		
No results found 🐾		

• 餐飲推薦

```
//generate the recommend-field by the input number
let input = document.querySelector('#input').innerText;
let array = input.split(',');
let runtime = array[0];
for(let i=0; i<runtime; i++){
  let j = i*8;
  let newdiv = document.createElement('div');
  newdiv.innerHTML = '<div class="rcd-field"><div class="';
  document.querySelector('#new-rcd').append(newdiv);
}
```

一樣利用 js 來抓取後端傳送過來的資料，並動態產生推薦用的格子。

特別的是，首頁的推薦是尋找附近的餐飲，而這裡是根據使用者以往的評分紀錄，以及其他使用者給予的評價來產生推薦的。



紅框處
範例
html:

```
<div class="rcd-field">
  <div class="upper-field">
    <a style="cursor: pointer; font-size:24px; color: □black;" href="https://www.google.com">餐廳名稱: 小魔女</a>
  </div>
  <div class="downer-field">
    <div style="flex: none; width: 50%;">電話: 0987-666-321/(03)5412-287</div>
    <div style="flex: auto;">評分: 滿分</div>
    <div style="flex: auto;">營業狀態: 營業中</div>
  </div>
</div>

<div class="pictures">
  
  
  
</div>
```

一樣會判斷是否有推薦存在，如果沒有歷史評分，由於無法判別使用者喜好，同樣也不會有推薦，顯示 no results，如下圖。



• 帳號管理

登入(Login)

帳號：

密碼：

沒有帳號? [點擊註冊](#)

註冊(Register)

帳號：

密碼：

確認密碼：

已經有帳號了? [點擊登入](#)

```
function register(){//switch to regist page
    let login = document.querySelector("#login");
    let register = document.querySelector("#register");
    login.style.display = "none";
    register.style.display = "block";
    not_error();
}
function login(){//switch to login page
    let login = document.querySelector("#login");
    let register = document.querySelector("#register");
    login.style.display = "block";
    register.style.display = "none";
    not_error();
}
```

當使用者點擊藍色字體的區域，切換登入與註冊畫面。

註冊(Register)

帳號：
879

密碼：
7

確認密碼：
...

你故意輸錯密碼嗎???

REGISTER

已經有帳號了? [點擊登入](#)

```
//detect whether the password is same with confirm-password when regist
let password = document.querySelector('#password');
let password_confirm = document.querySelector('#password-confirm');
function passwordConfirm(){
  if(password.value == password_confirm.value){
    password_confirm.setCustomValidity('');
  }else{
    password_confirm.setCustomValidity('你故意輸錯密碼嗎???');
  }
}
password.onkeyup = passwordConfirm;
password_confirm.onkeyup = passwordConfirm;
```

若密碼與確認密碼不同，則顯示錯誤訊息。

登入(Login)

帳號：
Account

帳號或密碼錯誤

註冊(Register)

帳號：
Account

帳號已被使用

```
function error() {//show the error message
  error = document.querySelectorAll('.error-msg');
  for(let i=0; i<error.length; i++){
    error[i].style.display = 'block';
  }
}
function not_error() {//hide the error message
  error = document.querySelectorAll('.error-msg');
  for(let i=0; i<error.length; i++){
    error[i].style.display = 'none';
  }
}

if(document.querySelector('#input').innerText == 'bad'){
  register();
}
if(document.querySelector('#loginReady').innerText == 'error'){
  error();
}
```

若使用者的輸入格式正確，但登入或註冊失敗，後端會告訴前端登入/註冊失敗，前端就可以顯示錯誤訊息。

最後，使用者在使用完後可以登出。就算登出了，我們的資料庫仍會保存使用者的資料。

(四). 進度紀錄

	証	士	溫
5/25	註冊完 github 和 heroku 帳號	久違的打開期末 project 的檔案，今天把 postgres 資料庫和 pqadmin 4 連起來了，感動。	創建 heroku 帳號並綁定 github
5/27	被組長提醒開始寫期末 project，今天將 line bot 帳號生出來了並連接 github 和 heroku。	新增了一個表單，並把資料庫與 python 連上	大致學習 html 的語法及常用標籤，還有 css 的用法
5/28	和組長討論了一下要做的功能，今天新增自動回覆訊息的功能，有了初步成果，感動。	一起討論了網頁底層要如何架構，以及希望有什麼功能	和士恆討論了網頁的基本組成架構
5/29	學習網路上 line bot 的語法和相關的功能	和証傑研究了 line bot 語法和連接 heroku 的方式，在 callback 那邊卡了很久。也成功用 git 將証傑的程式部屬到 heroku。	做出網頁的導覽列
5/30	同上	和旻軒討論了一下網頁需求。我也在試著找出在隨機經緯度附近的店家資訊。	製作 html 架構
5/31	今天將讀入位置資訊的主函式和回傳經緯度的副函式寫出來了。	和証傑研究如何使用 line 回傳經緯度。畫出了網頁介面的雛形。此外也建立了顧客對餐廳評分的資料庫。	把網頁的外觀正式訂出來，也把需要的功能決定好了
6/1	研究完 Carousel template 並將相關的功能完成成功將士恆給的資料印出來。	訂出了民雄餐廳名單，和要傳給 line 的資料格式。建立了儲存顧客資料、歷史評分兩個資料庫。	學習 js 基本文法，還有一些需要的函式，做了一些登入畫面
6/2	和士恆討論剩下要做的	進入三人整合階段!!!	把登入畫面做出來了!

	功能，順便幫士恆把餐廳名單分類。	做出登入系統雛形，還有把協同過濾演算法寫了出來，也把要傳給 line 的資料弄出來了。也將我想看到的 Line 介面畫來了。	
6/3	士恆幫我 debug，並將帳號和推薦指令的功能寫出來	傳給網頁的資料格式改動，網頁細節討論。建立了儲存店家餐廳 id 的資料庫。	決定了推薦跟評分用的資料格式，並且把這兩份網頁做出來
6/4	將 line bot 的 code 完成並傳給士恆整合	將 line bot 程式碼加入主程式中，實現了在手機登入帳號、設定喜歡的食物、找出附近和自己喜歡的食物相關的店家，以及將歷史評分紀錄顯示在 Line 中。	把首頁做出來
6/5		更新餐廳名單。	完成網頁前端!!
6/6		將旻軒的網頁部屬到 heroku 上，再次修改我的資料格式。	修改網頁的排版細項以及一些沒發現的小錯誤
6/7			新增看版圖
6/8	模擬報告	整理報告內容+小練一下。	新增找無資料時的圖片，將按鈕換成 aqua

(五). 參考資料

[第 21 天：Flask：裝飾我們的網頁](#)

[LINE BOT 教學 \(Python \) | STEAM 教育學習網 \(oxxostudio.tw\)](#)

[\[Day 26\]用 Django 架構建置專屬的 LINEBOT 吧 - 網路爬蟲\(III\)位置訊息應用 - iT 邦幫忙::一起幫忙解決難題，拯救 IT 人的一天 \(ithome.com.tw\)](#)

[\[Python+LINE Bot 教學\]6 步驟快速上手 LINE Bot 機器人 \(learncodewithmike.com\)](#)

[\[Python 爬蟲教學\]教你如何部署 Python 網頁爬蟲至 Heroku 雲端平台 \(learncodewithmike.com\)](#)

[以 Python 實現推薦系統的協同過濾算法](#)

[歐氏距離與餘弦相似度的比較](#)

[以 Python 打造簡單實用的電影推薦系統](#)

(六). 後記

夏夜晚風，吹亮了繁星點點。

站在 B 棟頂樓的我們舒展著僵硬的身軀

士恆：「你們看!今天的星星好清楚啊」

証傑：「是魔女送餐時的暗號嗎?」

旻軒：「那流星就是魔女留下的軌跡ㄟ」

証傑：「那我們和小魔女又有什麼不同呢?」

是阿，我們也只是在搬運著散落各地的知識而已

而如今任務告一段落，魔女食堂不僅在此發光發熱，也在等著下個新手魔女將它拾起，前往未知的旅程。

：「誰又不是如此呢?」

三人相視笑了

雖然一路走來跌跌撞撞，雖然未知多的難以掌控

不過看到成果後總覺得一切都值得了

2022_06_17 魔女隨筆