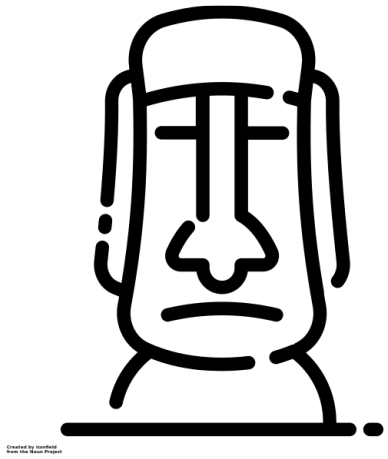
第五組

目錄



- 組員介紹
- 創作動機
- Line Bot操作說明
- 程式碼
- 相關應用
- 心得

一.組員介紹

巨資二B 陳尚恩

從小我就嚮往能坐在電腦前操作一般人看不懂的程式語言，所以對coding一直感到好奇與充滿熱忱。進入東吳後，讓我開始接觸程式，隨著code內容越來越多，完成一些語法複雜、燒腦的題目，即使每次debug都絞盡腦汁，但我非常享受這個過 程，看著自己作出來的成果日漸豐富，真的很有成就感。希望藉由這次的專題，累積實作的經驗，加強coding的能力，順利的結束這個課程。

日文四B 劉于孺

雖然不是本科系的，但在輔系的學習下漸 漸發現自己對程式有很大的興趣，當初會 想輔系是因為覺得日文是一項加分的工具 ，應該要再多學習第二專長，未來在就業 的時候才能有更多選擇。 希望在學生時期的最後一個報告能運用自 己所學，或是透過這次的報告學習新技能 跟相關知識，能讓自己更加進步。

巨資二B 王奕心

以前常常看到巨資碩士的哥哥在寫論文做專 題，對這方面產生了興趣，且每次看到哥哥 做完專題後的成就感，也讓我對這方面產生了憧憬。以前從未接觸過任何程式語言，上 學期才開始接觸，雖然現在也還算初學者，上課時跟著老師的操作，課後也會繼續鑽 研、複習，讓我更加熟悉python 語法及其 應用。希望透過這次的專題，能夠將所學與 專題融會貫通，並有個滿意的成果，也可以 透過這次更精進自己的python 語法等知識。

心理三 陳品妤

父母從事資訊相關工作，所以從以前就對 這一塊有點好奇（雖然沒有去特別研究） 在大學時期才開始接觸，幫忙家裡打打零 工寫一點簡單的程式碼，因此開始發現寫 程式的好玩之處，最喜歡的就是寫完一長 串，最後跑出來正確結果的成就感。 希望透過這門課的專案，可以讓我好好將 課堂上所學運用在其中，並且交出一份好 的專案作為這門課的完美ending

巨資二B 林羿帆

大學之前也從沒碰過程式，只覺得寫程式 好像很酷很厲害。於是之後便選擇了資訊 相關的科系，在這次專題實作中讓我更加 明白，原來程式不單單只是出現在電腦螢幕，做些簡單的數值運算而已，更可以結 合生活的情境使我們生活更佳的便利。

二.創作動機

二.創作動機

在當學生的時候，常常為了三餐要吃什麼而苦惱許久，尤其身為東吳大學的學生更加感同身受，校門外只有外雙溪和大馬路。在此為了解決大家的煩惱，我們決定以雙北的大學 （台大、政大、師大、北大、國北教、北市大、台科大、北科大、國北 護、北商、輔大、東吳、淡江、文化、世新、銘傳、實踐、北醫）為對象，依照使用者的查詢時間過濾掉目前未營業的店家，再以價格、距 離以及食物分類，整理各校周遭的各類美食，以供學生們選擇。

我們主要使用視覺化功能以及line bot聊天功能來提供服務，並透過上網、查找資料以及實地走訪，將查找的資料製作成資料庫來使用，以便推薦給大家值得一試的美食。

三.Line Bot操作說明

here

Step1:

選擇指定大學的所在縣市

here

Step2:

並選擇該縣市欄位以下的國立/私立大學類，即可在聊天室中呈現所在位置的大學

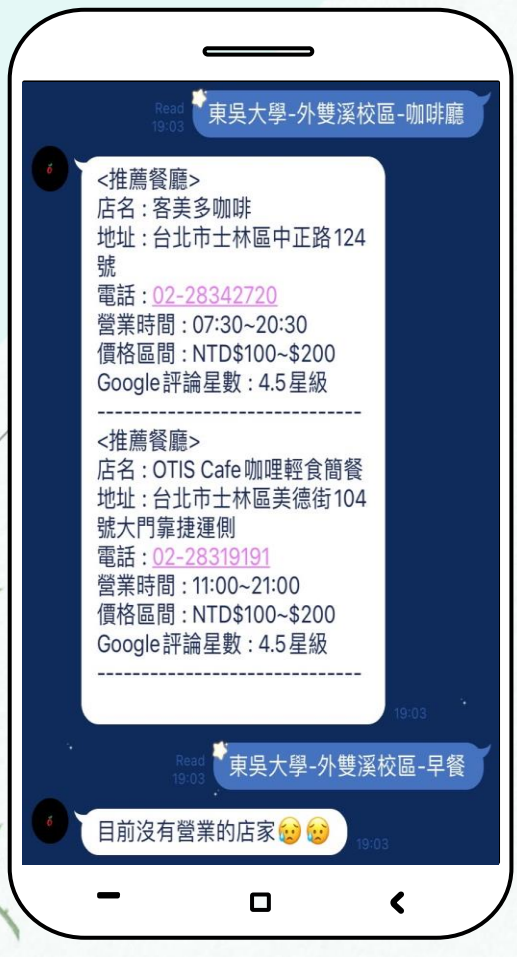
here

Step 3:

點擊完想查看的大學後，若該所大學有分校區，會再做區分
(ex:台北市立大學-天母、博愛
東吳大學-城中、雙溪
臺北大學-台北、三峽)

Step 4:

選擇完大學相關資訊後，即可開始瀏覽美食分群。共區分成多種類別，點擊看更多，即可呈現該分群的店家資訊。
(日式、韓式、台式、南洋、美式、西式、早餐、甜點、咖啡廳、手搖飲)



Step 5:

根據使用者所使用的時間，列出符合該時間內的所有指定分類的店家，若不符合該店家營業時間則會告知。


四.程式碼

```python
#匯入需要用到的套件
from flask import Flask,request,abort
from linebot import LineBotApi,WebhookHandler
from linebot.exceptions import InvalidSignatureError
from linebot.models import (MessageEvent,TextMessage,TextSendMessage , TemplateSendMessage , ButtonsTemplate ,
    PostbackTemplateAction , MessageTemplateAction , URITemplateAction , CarouselTemplate , CarouselColumn , FlexSendMessage)
import pandas as pd
import time
import datetime
```

A1 | name

	A	B	C	D	E	F	G	H	I	J	K	L
1	name	address	phone	open	close	rest	class	price	id	star	price_interval	
2	吐司覺得熱	台北市士林區士東路266巷3弄15號1樓	02-88661152	07:00	14:00	7	早餐	b	0	4.6星級	NTD$100~$200	
3	食指大井 丼飯專賣店	台北市士林區士東路336號	02-28385689	11:30/17:30	14:00/21:00	0	日式	d	1	4.6星級	Over NTD$300	
4	O PA 輕食簡餐	台北市士林區士東路260-1號	02-28317836	11:00	20:00	6	西式	b	2	4.3星級	NTD$100~$200	
5	LA PASTA 義麵屋	台北市士林區天母東路102號	02-28721738	11:00/17:00	14:30/21:00	0	西式	c	3	4.2星級	NTD$200~$300	
6	蕭家牛肉麵	台北市士林區德行東路331巷50弄6號	936192276	11:00	20:30	0	台式	b	4	4.8星級	NTD$100~$200	
7	滷味香排骨店(自助餐)	台北市士林區士東路200巷79號	02-28326522	11:00/16:30	14:00/19:30	7	台式	a	5	4.4星級	Under NTD$100	
8	松三町 丼飯、咖哩、烏龍麵	台北市士林區天母東路50巷22之8號	02-28771788	11:30/17:00	14:00/20:00	1	日式	b	6	4.5星級	NTD$100~$200	
9	天東86牛肉麵	台北市士林區天母東路86號	02-28734649	11:30/17:30	14:30/20:30	0	台式	b	7	4.5星級	NTD$100~$200	
10	品卉小吃	台北市士林區士東路286巷1-16號	02-28335297	11:00	20:00	7	台式	b	8	3.9星級	NTD$100~$200	
11	方家小館(上海名菜)	台北市士林區天母東路7號	02-28728402	11:15/17:00	14:30/21:00	0	中式	d	9	4.1星級	Over NTD$300	
12	女娘的店(滷肉飯)	台北市士林區天母東路97號	02-28741981	11:30/17:00	14:00/22:00	1	台式	d	10	4.3星級	Over NTD$300	
13	世界一粥	台北市士林區士東路200巷81號	02-28377867	16:00	23:00	6	台式	a	11	4.4星級	Under NTD$100	
14	親愛的小籠湯包	台北市士林區德行東路244號	02-28321886	10:30	22:30	0	台式	a	12	4.2星級	Under NTD$100	
15	天母德州牛排	台北市士林區德行東路205號	916179119	11:00/17:00	14:00/21:00	0	美式	c	13	4.3星級	NTD$200~$300	
16	秀緣鮮果汁	台北市士林區士東路286巷1-16號	02-28353626	10:00	20:00	7	飲料	a	14	4.4星級	Under NTD$100	

UTTfinally

```python
import requests
from bs4 import BeautifulSoup
from selenium import webdriver
import time
import pandas as pd


stars = []

options = webdriver.ChromeOptions()


from webdriver_manager.chrome import ChromeDriverManager
browser = webdriver.Chrome(ChromeDriverManager().install())

for nc in name_crawler:

    url = "https://www.google.com.tw/maps/place/" + str(nc)

    browser.implicitly_wait(1)

    browser.get(url)

    browser.find_element_by_class_name("pzfvzf").click()

    time.sleep(2)

    tmp = browser.find_elements_by_xpath("//div[@class='ODSEW-ShBeI-content']")

    soup = BeautifulSoup(browser.page_source, 'html.parser')

    star = soup.find_all('div', {'class': 'PPCwl'})

    star = str(star).split("=")[99].replace('"','').replace('jstcache','').replace(' ',

    stars.append(star)

    print(star)
```

```python
#台式料理
def TaiwanUTT():
    df = pd.read_csv("UTTfinally.csv")
    C0 = 0
    C1 = 0
    C2 = 0
    recount0 = []
    recount1 = []
    name = []
    address = []
    phone = []
    Time = []
    price_interval = []
    stars = []
    #食物風味篩選
    for classify in df['class']:
        if classify != "台式":
            others = df[df['id'] == C0].index
            df.drop(others , inplace=True)  #將不是此分類的店家從DataFrame刪除, inplace=True代表會修改原來的資料
            C0 += 1
        else:
            C0 += 1
    #id重置
    for a in range(len(df['rest'])):  #取得新DataFrame的長度將id重置, 以利後續篩選
        recount0.append(a)
    df['id'] = recount0
```

```python
    #公休日篩選
    for rest in df['rest']:
        if rest != '0':  #由於datetime他的星期是以數字表達, 故我們資料中的0代表店家沒有公休
            if len(str(rest)) != 1:  #若公休日長度不等於1, 代表這家店公休日有兩天
                R = rest.split('_')  #將兩天公休日分隔
                today = datetime.date.today()  #取得今天日期
                weekday = today.isoweekday()  #取得今天星期
                if R[0] == str(weekday) or R[1] == str(weekday):  #判斷公休日有無等於今天的星期
                    rr = df[df['id'] == C1].index
                    df.drop(rr , inplace=True)  #將公休的店家從DataFrame刪除, inplace=True代表會修改原來的資料
                    C1 += 1
                else:
                    C1 += 1
            else:  #其他就是每周有固定公休日的店家
                today = datetime.date.today()  #取得今天日期
                weekday = today.isoweekday()  #取得今天星期
                r = df[ df['rest'] == str(weekday)].index
                df.drop(r , inplace=True)  #將公休的店家從DataFrame刪除, inplace=True代表會修改原來的資料
                C1 += 1
        else:
            C1 += 1
    #id重置
    for b in range(len(df['rest'])):  #取得新DataFrame的長度將id重置, 以利後續篩選
        recount1.append(b)
    df['id'] = recount1
```

```python
    #營業時間篩選
    for O,C in zip(df['open'],df['close']):  #資料中營業時間分為開始和打烊
        if len(O) >10 and len(C) > 10:  #許多店家為兩段式營業的店家, 故在資料中他的長度會是11, 如:11:00/17:00(11)~14:00/21:00
            morning_open = O.split('/')[0]  #將白天與晚上的營業時間做切割
            night_open = O.split('/')[1]  #將白天與晚上的營業時間做切割
            morning_close = C.split('/')[0]  #將白天與晚上的營業時間做切割
            night_close = C.split('/')[1]  #將白天與晚上的營業時間做切割
            now = time.strftime('%H:%M' , time.localtime())  #取得現在時間
            if now > morning_open and now < morning_close:  #判斷現在時間有無在白天的營業時間內
                C2 += 1
            elif now > night_open and now < night_close:  #判斷現在時間有無在晚上的營業時間內
                C2 += 1
            else:
                t = df[df['id'] == C2].index
                df.drop(t , inplace=True)  #將現在打烊或休息的店家從DataFrame刪除, inplace=True代表會修改原來的資料
                C2 += 1
        else:  #其他為一個時段營業到底的店家, 如:11:00~22:00
            now = time.strftime('%H:%M' , time.localtime())  #取得現在時間
            if now > O and now < C:  #判斷現在時間有無在營業時間內
                C2 += 1
            else:
                t = df[df['id'] == C2].index
                df.drop(t , inplace=True)  #將現在打烊或休息的店家從DataFrame刪除, inplace=True代表會修改原來的資料
                C2 += 1
```

```
    #將篩選完的資料，依類別分別放入每個list中
    for n in df['name']:
        name.append(n)  #將篩選完的DataFrame資料依他的column來丟到新的list中
    for ad in df['address']:
        address.append(ad)  #將篩選完的DataFrame資料依他的column來丟到新的list中
    for c in df['phone']:
        if len(c) == 9:  #若店家電話為手機號碼，csv檔不會顯示第一個0
            c = '0'+str(c)
            phone.append(c)
        elif c == 'None' or c == 'none':  #有些店家沒有留電話
            c = 'NaN'
            phone.append(c)
        else:
            phone.append(c)
    for O,C in zip(df['open'],df['close']):
        if len(O) >10 and len(C) > 10:
            morning_open = O.split('/')[0]
            night_open = O.split('/')[1]
            morning_close = C.split('/')[0]
            night_close = C.split('/')[1]
            OC0 = morning_open+"~"+morning_close+"/"+night_open+"~"+night_close  #將營業時間拆成大眾較易理解的型態
            Time.append(OC0)
        else:
            OC1 = O+"~"+C
            Time.append(OC1)
    for st in df['star']:
        stars.append(st)  #將篩選完的DataFrame資料依他的column來丟到新的list中
    for pi in df['price_interval']:
        price_interval.append(pi)  #將篩選完的DataFrame資料依他的column來丟到新的list中
```

```
    contents = []
    final = ""
    for count in range(len(df['id'])):
        content = ("<推薦餐廳>\n店名 : {}\n地址 : {}\n電話 : {}\n營業時間 : {}\n價格區間 : {}\nGoogle評論星數 : {}\n{}\n"
                        .format(name[count],address[count],phone[count],Time[count],price_interval[count],stars[count],"-"*30)
                            )
        contents.append(content)  #回覆給使用者的訊息寫好新增到list中
    for con in contents:
        final += con  #再利用迴圈將list的每則訊息全部轉為字串聯在一起
    if df.empty:
        final = "目前沒有營業的店家😳😳"  #若DataFrame中沒有資料的話，將回覆使用者訊息
    return final
```

```
app = Flask(__name__)

line_bot_api = LineBotApi("aF4BT0fW4qs/5UXgD+ZefJl/hP7bW93yuAFF0T6CEkSqXHCxtcEmofST2ClN3/twm6X7P2OUqkNqQdAIQvZe7rKAeSoE/
    PedIqQ1xiiN+LCf8QThaVof0btMXKuoiY9fK5vf7c7bB0lwPdM1AgUdlwdB04t89/1O/w1cDnyilFU=")
handler = WebhookHandler("2dd8ddcbc32eca207b2e6da318692267")

@app.route("/callback" , methods=['POST'])
def callback():
    signature = request.headers['X-Line-Signature']
    body = request.get_data(as_text=True)
    app.logger.info("Request body:" + body)
    try:
        handler.handle(body,signature)
    except InvalidSignatureError:
        abort(400)
    return 'OK'
```

```python
@handler.add(MessageEvent , message=TextMessage)
def handle_message(event):
    line=event.message.text
    if line=="台北市國立大學":
        carousel_template_message = TemplateSendMessage(  #接收指令後, LineBot回覆的訊息
        alt_text = "Carousel template",
        template = CarouselTemplate(  #使用Carousel Template
            columns=[
                CarouselColumn(
                    thumbnail_image_url="https://i.imgur.com/NvgMHu3.jpg",
                    title = "台灣大學 NTU",
                    text = "請選擇您大學",
                    actions = [
                        MessageTemplateAction(  #回覆text:台灣大學
                            label="台灣大學",
                            text="台灣大學",
                        ),
                        URITemplateAction(   #利用google map查看台大的地理位置
                            label="查看位址",
                            uri="https://www.google.com/maps/search/NTU/"
                        )
                    ]
                ),
```


```python
    elif line=="東吳大學":
        buttons_template_message = TemplateSendMessage(
        alt_text = "Buttons template",
        template = ButtonsTemplate(
            thumbnail_image_url="https://i.imgur.com/VXn5ID3.jpg",
            title = "東吳大學",
            text = "請選擇您的校區",
            actions = [
                MessageTemplateAction(
                    label="外雙溪校區",
                    text="東吳大學-外雙溪校區",
                ),
                MessageTemplateAction(
                    label="城中校區",
                    text="東吳大學-城中校區",
                )
            ]
        )
    )
        line_bot_api.reply_message(event.reply_token , buttons_template_message)
        return 0
#----------------------------ButtonTemplate-----------------------------
```

```
elif line=="台北市立大學-天母校區":
    line_bot_api.reply_message(event.reply_token,messages=FlexSendMessage(
        alt_text="bubble message",
        contents={
            "type": "carousel",
            "contents":[
                {  #日式
                    "type": "bubble",
                    "body": {
                        "type": "box",
                        "layout": "vertical",
                        "contents": [
                            {
                                "type": "image",
                                "url": "https://i.imgur.com/ZH7asXM.png",
                                "size": "full",
                                "aspectMode": "cover",
                                "aspectRatio": "260:194",
                                "gravity": "top"
                            },
```

```
{
    "type": "box",
    "layout": "vertical",
    "contents": [
        {
            "type": "text",
            "text": "#日本",
            "color": "#ffffff",
            "align": "center",
            "size": "xs",
            "offsetTop": "3px",
            "wrap": True
        }
    ],
    "position": "absolute",
    "cornerRadius": "20px",
    "offsetTop": "18px",
    "backgroundColor": "#ff334b",
    "offsetStart": "18px",
    "height": "25px",
    "width": "100px"
},
```

```
{
    "type": "box",
    "layout": "vertical",
    "contents": [
        {
            "type": "text",
            "text": "#にっぽん",
            "align": "center",
            "size": "xs",
            "offsetTop": "3px",
            "wrap": True
        }
    ],
    "position": "absolute",
    "cornerRadius": "20px",
    "offsetTop": "18px",
    "backgroundColor": "#a6ed8e",
    "height": "25px",
    "width": "100px",
    "offsetEnd": "18px",
}
```


```
        "type": "box",
        "layout": "baseline",
        "contents": [
            {
                "type":"icon",
                "url":"https://i.imgur.com/L2yKIMz.jpg",
                "size":"xl",
                "aspectRatio":"640:427"
            },
            {
                "type": "text",
                "text": " 日式料理",
                "size": "xxl",
                "color": "#FFFFFF",
                "weight": "bold"
            }
        ]
    },
```

```
{
    "type": "box",
    "layout": "baseline",
    "contents": [
        {
            "type": "text",
            "text": "如 : 拉麵、燒烤、壽司、丼飯、居酒屋、鰻魚飯、烏龍麵、和食",
            "color": "#FFFFFF",
            "size": "sm",
            "flex": 0,
            "wrap": True
        }
    ],
    "spacing": "lg"
},
```

```
{
    "type": "filler"
},
{
    "type": "box",
    "layout": "baseline",
    "contents": [
        {
            "type": "filler"
        },
        {
            "type": "text",
            "text": "點我看更多日式料理",
            "color": "#FFFFFF",
            "flex": 0,
            "offsetTop": "-2px"
        },
        {
            "type": "filler"
        }
    ],
    "spacing": "sm",
    "action": {
        "type": "message",
        "label":"日式料理",
        "text": "台北市立大學-天母校區-日式"
    }
},
{
    "type": "filler"
}
```


```
    elif line == "東吳大學-外雙溪校區-日式":
        content = JapanSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-韓式":
        content = KoreaSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-台式":
        content = TaiwanSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-南洋":
        content = SEAsiaSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-美式":
        content = AmericaSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-西式":
        content = WesternSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-早餐":
        content = breakfastSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-飲料":
        content = beverageSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-甜點":
        content = dessertSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
    elif line == "東吳大學-外雙溪校區-咖啡廳":
        content = cafeSCUS()
        line_bot_api.reply_message(event.reply_token,TextSendMessage(text=content))
#----------------------------東吳大學-外雙溪校區-----------------------------
```

```
    else:
        line_bot_api.reply_message(event.reply_token ,
                                   TextSendMessage(text="請利用圖文選單輔助操作🤗\n選擇您的學校後🏫\n自行決定您今天的用餐風味🍜"))

if __name__ == "__main__":
    app.run()
```

五.相關應用

技術原理:

使用者透過LINE發送訊息時，LINE Platform將會進行接收，並且傳遞至我們所開發的LINE Bot執行邏輯運算後，透過LINE所提供的Messaging API回應訊息給LINE Platform，最後再將訊息傳遞給使用者。

Line Bot 相關應用:

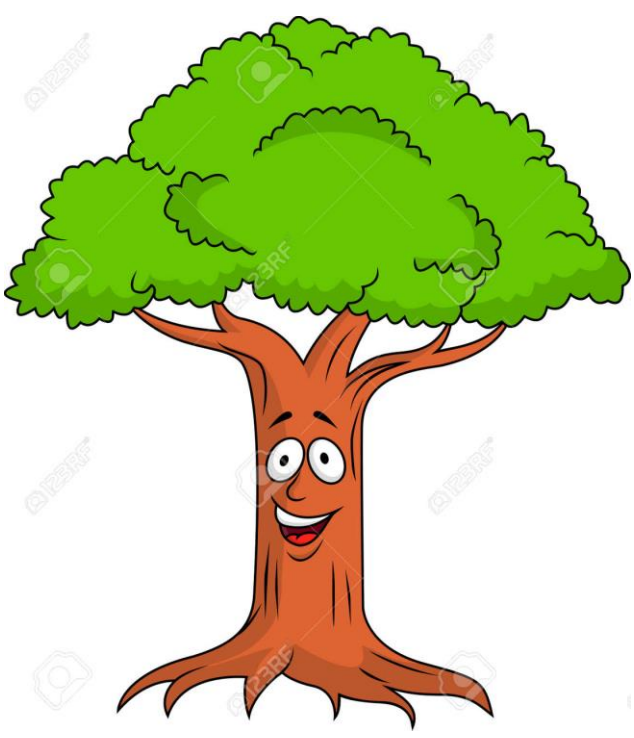
Line Bot不僅可以做為店家的客服外，
也可以針對各個領域去作為此領域的聊天機器人

調酒的聊天機器人（很適合夜店、餐廳來開發），可以幫助用戶在酒吧選擇雞尾酒，除了透過雞尾酒的名稱、酒單選擇以外，還可以依照心情、口味、約會日來推薦。

針對扔垃圾問題而開發的聊天機器人，專門用來管理垃圾收集計畫和垃圾稅，透過註冊垃圾收集日的日期，它就會自動通知，或告知可燃垃圾的收集日、哪些地方可以扔垃圾。同時，也可以依照你的區域，告訴你該區的垃圾分類訊息。

LINE Bot做成遊戲的形式，更容易受到用戶喜愛，而且像這樣的遊戲，只要透過聊天機器人開發平台當中的『圖像按鈕』功能，就可以做得出來。

六.心得



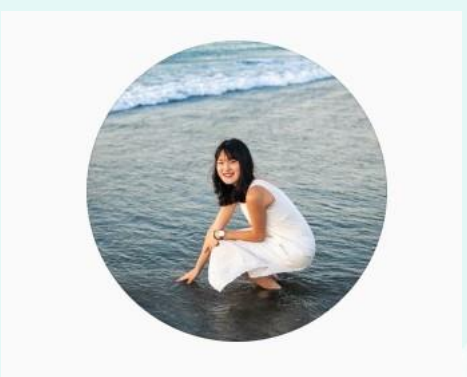
這次的專案做起來對我們來說跟上課的感覺很不一樣，因爲遇到不會的問題需要自己找答案，也因為後來變成遠距的方式，在討論上就變得沒有那麼方便。剛開始會有很多挫折感，處理了一次又一次的error ，好幾次都很想放棄，相對之下，最後真的寫出來後，成就感是遠遠大於那些挫折感的，這個專案讓我們看到寫程式的另外艱辛的一面，學到很多。

可以優化的地方：

1. 希望能夠做到在使用者搜尋的同時，計算出使用者目前與該店家的距離
2. 一並列出是否有外送平台的服務
3. 能夠讓使用者在電腦上也能使用這個功能

巨資二B 陳尚恩

在開始操作之前設想可以做出很豐富的成果，但沒想到實際要的東西比想像中多很多，類似的文章也不好找，要花大量時間解讀原文和程式碼各個函數的功能。但在成功做出來後，成就感遠大於過程中error的挫折感，總共57000多行的程式碼，也算是人生中的很好的經驗。透過這次的專題讓我學到很多（爬蟲真的好方便），未來希望可以經歷更多python的各種應用，讓自己的程式能力不斷的進步。



日文四B 劉于孺

我覺得這次的專題對我來說是一個很特別的經驗，從一開始的發想到最後看到成果做出來的那種成就感非常感動，但中間免不了出現程式的error，導致挫折感滿滿，但也因為中間的辛苦，才有我們這次這麼完整的專案，要謝謝自己的努力以及組員們的幫忙。

巨資二B 王奕心

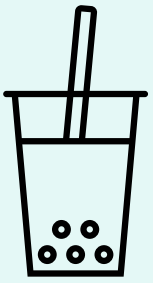
經過這一學年的課程後，以這次的期末專題作為本課程的美好結尾，雖然過程真的很辛苦，對程式碼也都不是那麼熟悉，每次看到error都很想放棄，慶幸自己有堅持下來，以及在後續的專題中，也跟組員學習到很多，本次專題不僅是呈現這一年多學習成果外，也學到了團隊合作的重要性，希望在未來，能夠將課程所學應用在生活中。

心理三 陳品妤

之前上課做作業時，一直覺得寫程式好像沒有那麼困難，只要根據當堂課的上課內容，推敲一下 ，就能夠完成，但是直到開始寫專案後，要從無到有，寫之前再也沒有參考的程式碼，就開始會有很多挫折感，處理了一次又一次的error ，好幾次都很想放棄，但對之下，最後真的寫出來後，成就感是遠遠大於課堂作業的，這個專案讓我看到寫程式的另外艱辛的一面，學到很多。

巨資二B 林羿帆

在這次專題學到很多東西，原來不用許多繁瑣的程序、程式碼也能做到人機互動的介面，相信經歷過這次的專題後，對於我未來將會有很大的助益。



組員	分工
陳尚恩	Line Bot程式碼撰寫， 資料前處理、查找
劉于孺	資料前處理、查找、簡報製作
王奕心	資料前處理、查找
陳品妤	資料前處理、查找
林羿帆	資料前處理、查找、報告書

Thank YOU