探食舍

TFB103 G1

組員：張為勳、陳儷文、林德發、黃彥宗、李昀諭

指導老師：郭惠民、蔡昀翰、蔡政廷

緯育 TibaMe

成員介紹

黃彥宗

張為勳

陳儷文

林德發

李昀諭

專題介紹

Introduction

報告人: 黃彥宗

專題大綱

1.專題介紹	專題說明、專題整體結構、工作分配
2.環境建置	資料庫配置、資料串流方法
3.資料處理	全案資料策略說明
4.資料清洗	網路爬蟲、資料評估、資料清洗
5.資料分析	資料分析、ML建模、演算法運用
6.成果展示	前端功能展示
7.未來展望	未來發展與計畫

專題發想

好多種房間

市面上訂房網站只提供飯店自身條件(價格、設施)，真的適合我嗎?

好多篇文章

透過部落格的介紹、看評論，想知道最近大家討論的熱點，想跟上流行。

我們聽到你的心聲

依使用者客制喜好，找到貼近需求的住宿點。

提案調查

1.

自助方式

國人旅遊方式多以「自行規劃行程旅遊」(90.6%)。

- 109年臺灣旅遊狀況調查報告

2.

資訊取得

旅遊資訊來源主要以「網路網絡與社群媒體」52.8%

- 109年臺灣旅遊狀況調查報告

3.

合法旅館數

依交通部觀光局2021年8月統計共1079家，台北市(600)、新北市(235)、宜蘭(244)。

目標族群

熱愛自助旅行

家庭|情侶|親友團

背包客

出遊沒想法的旅客

說走就走團

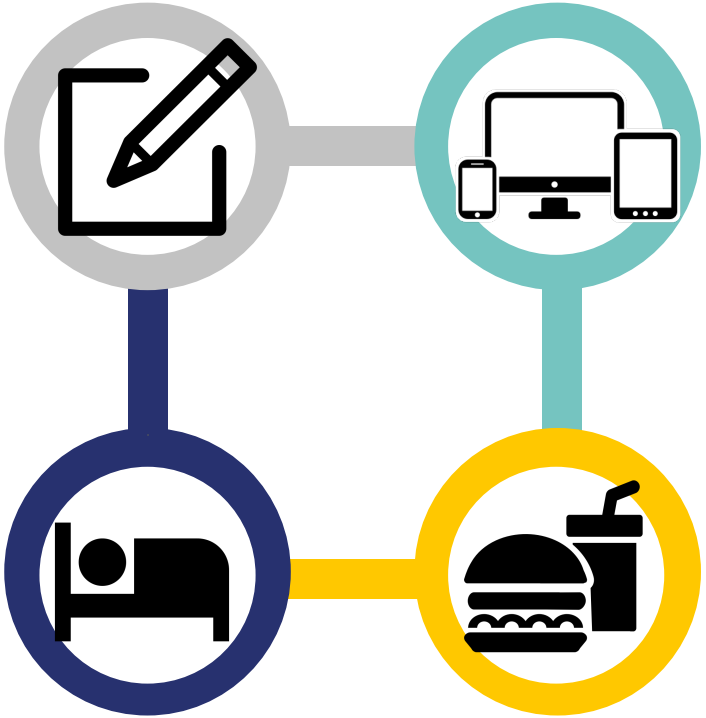
產品流程

使用者喜好資訊

藉由問卷圖文搭配，逐步建立使用者，的個性化標籤

網路與社群媒體資訊

結合訂房網站、遊記、評價，對使用者標籤做群聚分析與協同過濾等



個性房型推薦

搭配使用者旅遊偏好找出心目中的住宿點

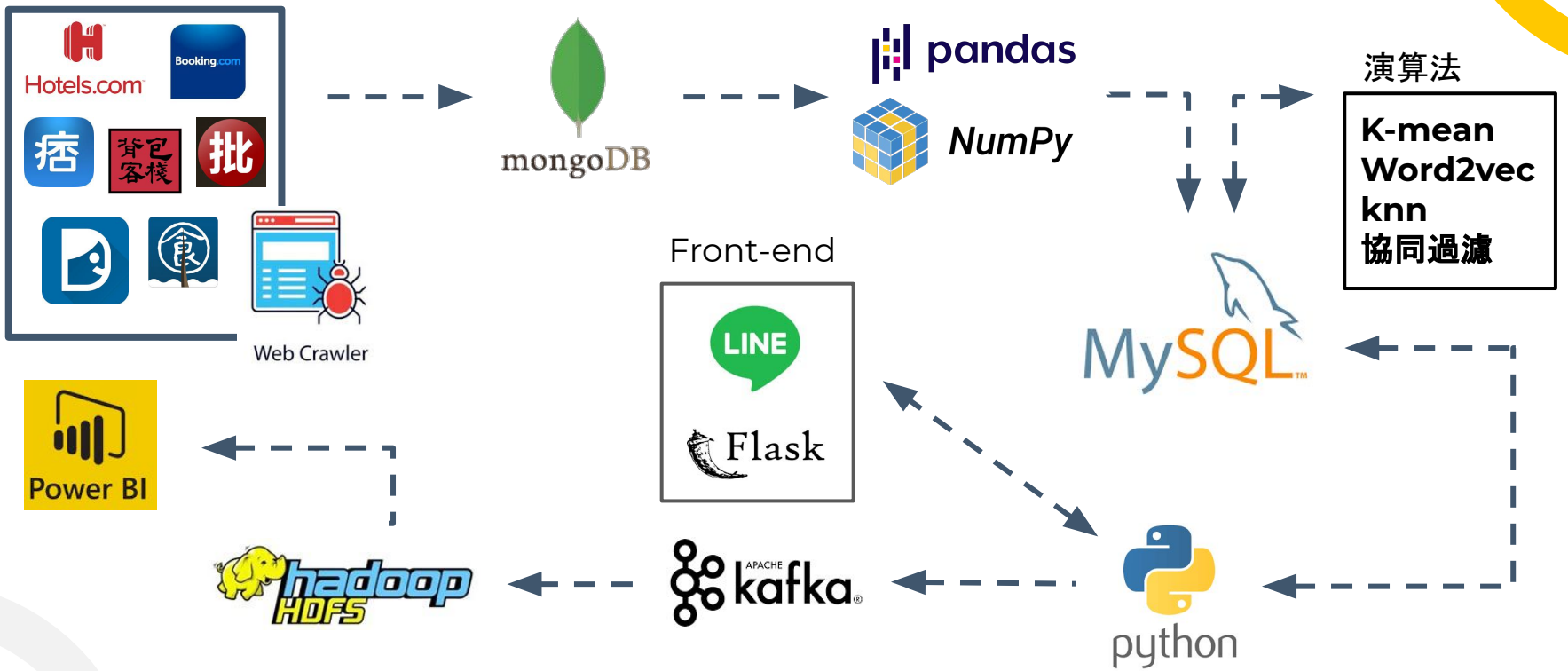
餐廳推薦

根據所在地區、價格、評分、分類(特色小吃、西式、日式、東南亞料理)

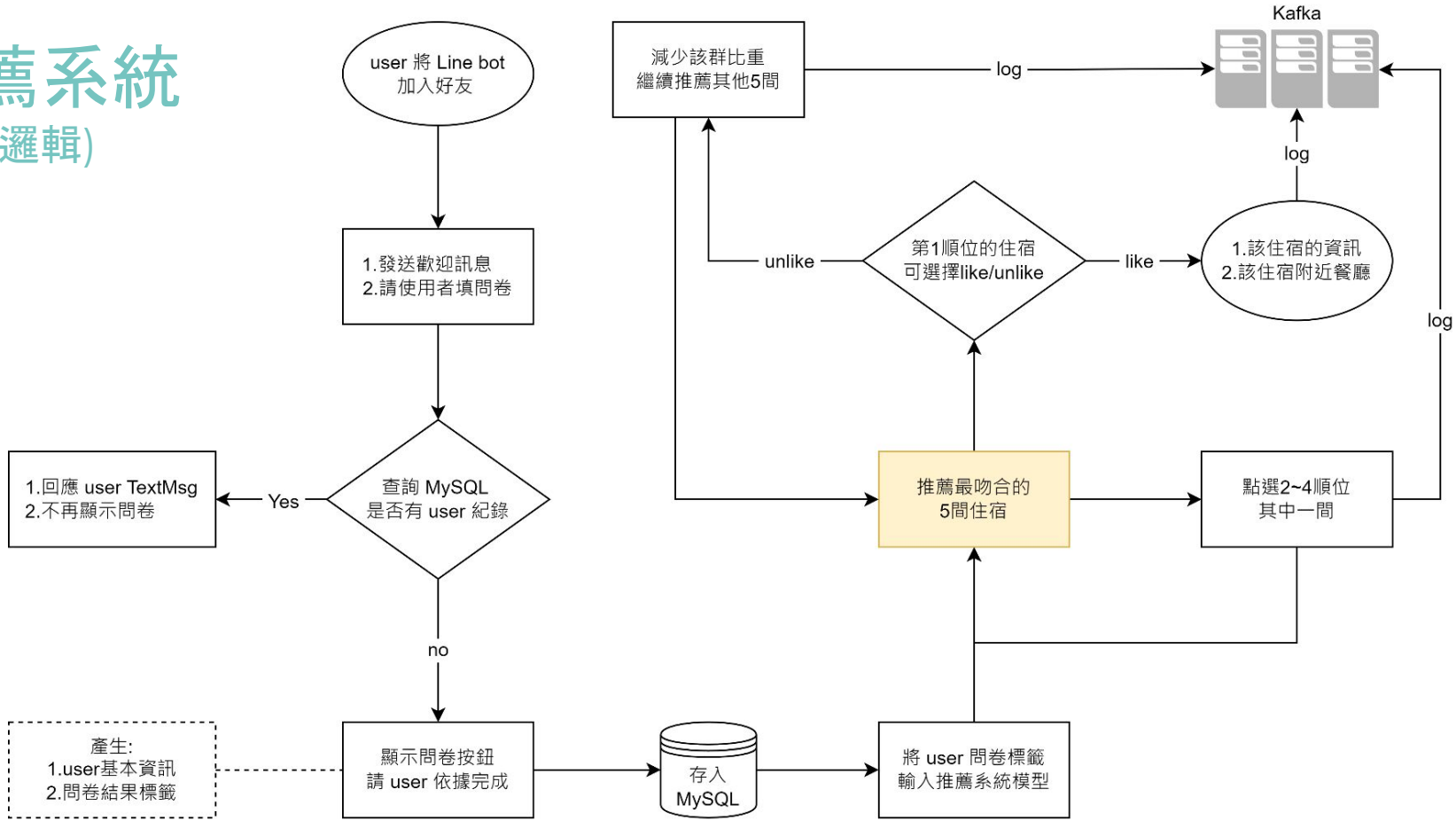
專題分工

	網路爬蟲	資料清洗	資料庫 建置/管理	特徵工程	演算法 建模/分析	APP 功能開發	前後端 串接
張為勳	●	●		●	●		
陳儷文	●	●				●	●
林德發	●	●		●	●		
黃彥宗	●	●	●			●	●
李昀諭	●	●		●	●		

專題架構


Hotels.com
Booking.com
Web Crawler
mongoDB
pandas
NumPy
演算法
K-mean
Word2vec
knn
協同過濾
Front-end
LINE
Flask
MySQL
Power BI
hadoop
HDFS
APACHE kafka
python

推薦系統
(運作邏輯)


user 將 Line bot
加入好友
1.發送歡迎訊息
2.請使用者填問卷
查詢 MySQL
是否有 user 紀錄
Yes
1.回應 user TextMsg
2.不再顯示問卷
no
產生:
1.user基本資訊
2.問卷結果標籤
顯示問卷按鈕
請 user 依據完成
存入
MySQL
將 user 問卷標籤
輸入推薦系統模型
推薦最吻合的
5間住宿
減少該群比重
繼續推薦其他5間
log
Kafka
log
第1順位的住宿
可選擇like/unlike
unlike
like
1.該住宿的資訊
2.該住宿附近餐廳
log
點選2~4順位
其中一間
log

資料串流環境建置

Data Streaming Environment

報告人: 黃彥宗

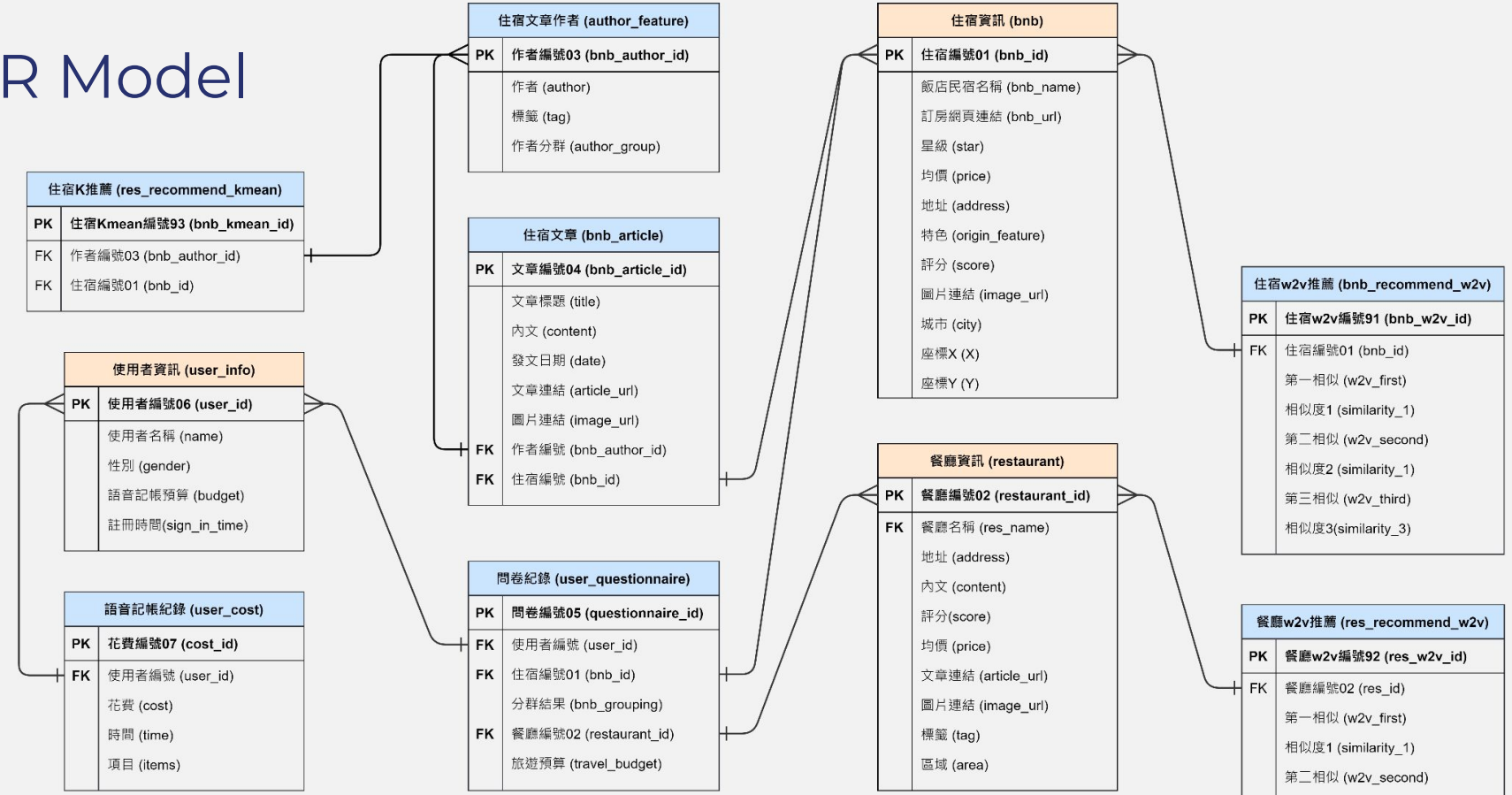
資料倉儲配置

- 爬蟲 Raw data

- 飯店、餐廳資訊
- 使用者基本資訊
- 使用者問券調查紀錄
- 演算法模型結果

- 使用者行為 Log (.csv)

E-R Model


住宿文章作者 (author_feature)
PK 作者編號03 (bnb_author_id)
作者 (author)
標籤 (tag)
作者分群 (author_group)
住宿K推薦 (res_recommend_kmean)
PK 住宿Kmean編號93 (bnb_kmean_id)
FK 作者編號03 (bnb_author_id)
FK 住宿編號01 (bnb_id)
住宿文章 (bnb_article)
PK 文章編號04 (bnb_article_id)
文章標題 (title)
內文 (content)
發文日期 (date)
文章連結 (article_url)
圖片連結 (image_url)
FK 作者編號 (bnb_author_id)
FK 住宿編號 (bnb_id)
住宿資訊 (bnb)
PK 住宿編號01 (bnb_id)
飯店民宿名稱 (bnb_name)
訂房網頁連結 (bnb_url)
星級 (star)
均價 (price)
地址 (address)
特色 (origin_feature)
評分 (score)
圖片連結 (image_url)
城市 (city)
座標X (X)
座標Y (Y)
住宿w2v推薦 (bnb_recommend_w2v)
PK 住宿w2v編號91 (bnb_w2v_id)
FK 住宿編號01 (bnb_id)
第一相似 (w2v_first)
相似度1 (similarity_1)
第二相似 (w2v_second)
相似度2 (similarity_1)
第三相似 (w2v_third)
相似度3(similarity_3)
使用者資訊 (user_info)
PK 使用者編號06 (user_id)
使用者名稱 (name)
性別 (gender)
語音記帳預算 (budget)
註冊時間(sign_in_time)
問卷紀錄 (user_questionnaire)
PK 問卷編號05 (questionnaire_id)
FK 使用者編號 (user_id)
FK 住宿編號01 (bnb_id)
分群結果 (bnb_grouping)
FK 餐廳編號02 (restaurant_id)
旅遊預算 (travel_budget)
餐廳資訊 (restaurant)
PK 餐廳編號02 (restaurant_id)
FK 餐廳名稱 (res_name)
地址 (address)
內文 (content)
評分(score)
均價 (price)
文章連結 (article_url)
圖片連結 (image_url)
標籤 (tag)
區域 (area)
語音記帳紀錄 (user_cost)
PK 花費編號07 (cost_id)
FK 使用者編號 (user_id)
花費 (cost)
時間 (time)
項目 (items)
餐廳w2v推薦 (res_recommend_w2v)
PK 餐廳w2v編號92 (res_w2v_id)
FK 餐廳編號02 (res_id)
第一相似 (w2v_first)
相似度1 (similarity_1)
第二相似 (w2v_second)
相似度2 (similarity_2)
第三相似 (w2v_third)
相似度3 (similarity_3)

MySQL 資料庫應用

class: MysqlDataFrame

```python
class MysqlDataFrame:
    def __init__(self, user, pwd, db='tfb1031_project', ip='10.2.16.174'):
        self.user = user
        self.pwd = pwd
        self.db = db
        self.__conn_ip = ip
        self.__stopWords = [
            'ALTER', 'UPDATE', 'DELETE', 'DROP', 'INSERT',
            'TABLE', 'DATABASE'
            ]
```

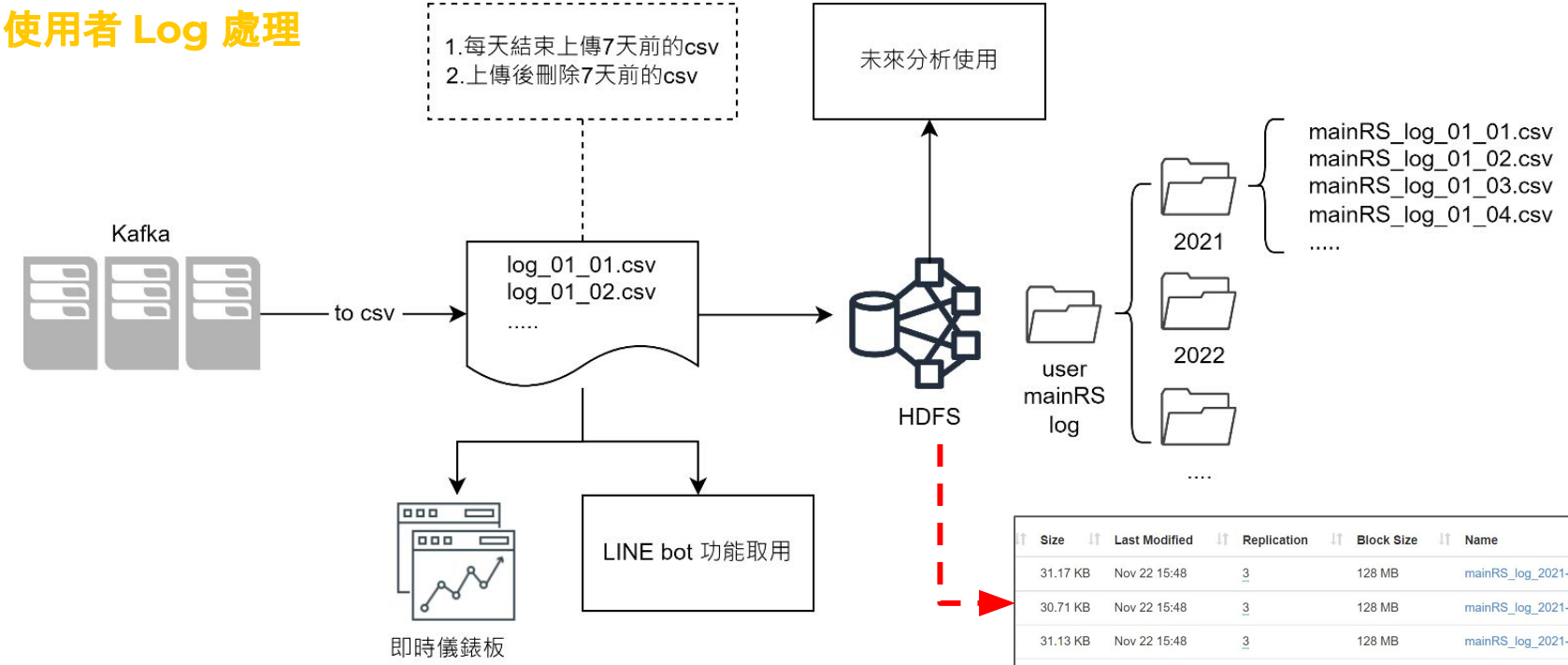
- 建立 class 把連線 db 包在裡面
- 埋入 SQL 語法
- 以 pandas.DataFrame 方式，直接存取資料庫的資料
- 設定 stopWords 以防修改資料庫

```python
# Get sql table to pd.DataFrame
def get_pandas_df(self, table='test'):
    engine = self.__create_conn()
    sql = f'SELECT * FROM {table};'
    try:
        df = pd.read_sql_query(sql, engine)
        return df
    except Exception as err:
        print(logging.error(str(err)))

# Use user-defined SQL
def use_sql_query(self, input_sql):
    engine = self.__create_conn()
    for word in self.__stopWords:
        if word in input_sql.upper():
            return print("Please don't alter the table.")
    try:
        df = pd.read_sql_query(input_sql, engine)
        return df
    except Exception as err:
        print(logging.error(str(err)))
```

Kafka 應用

使用者 Log 處理



Kafka 應用

Kafka producer

```python
def kafka_producer(topic, user_id, *data):
    producer = conn_kafka_producer()
    producer.produce(topic, key=user_id, value=f'{data}')
    producer.poll(0)
    print(f'key = {user_id},\nvalue = {data}')
    producer.flush(10)
```

- 把 producer 埋在前端的推薦系統內觸發
- key = user_id
- value = 點擊的飯店名稱, 點擊時間

user_id	click	time
U8a6e3915313149a9d5b445862b79767a	府中棧精品商旅	16:28:15
U8a6e3915313149a9d5b445862b79774a	台北東旅	16:28:20
U8a6e3915313149a9d5b445862b79793a	城市商旅北門館	16:28:20
U8a6e3915313149a9d5b445862b79767a	函舍旅館	16:28:20

Kafka 應用

Kafka consumer

```
# Create a new file when 00:00:01 everyday
if nowTime == saveFileTime:
    with open(f'data/user_mainRS_log/{year}/mainRS_log_{todayDate}.csv',
              'w', encoding='utf-8')as f:
        f.write(f'{msg_key},{msg_value},{nowTime}\n')
    # Put last week file to HDFS
    client.upload(f'/project/tfb1031/user_mainRS_log/{year}',
                  f'data/user_mainRS_log/{year}/mainRS_log_{lastWeek}.csv')
    # Remove last week file after putting it on HDFS
    os.remove(f'data/user_mainRS_log/{year}/mainRS_log_{lastWeek}.csv')
    continue
else:
    print(nowTime)
    with open(f'./data/user_mainRS_log/{year}/mainRS_log_{todayDate}.csv',
              'a', encoding='utf-8')as f:
        f.write(f'{msg_key},{msg_value},{nowTime}\n')
```

Log 自動化流程:

- 每天 00:00:01 建立當日 csv
- 上傳7天前的 csv 到 HDFS
- 刪除7天前的 csv

(本地端只留7日內的 log,
提供前端做即時統計)

資料處理

Data strategy

報告人: 黃彥宗

推薦系統

訂房網站

- booking.com
- hotel.com

1400筆

部落客住宿文章

- 背包客棧
- 痞客邦, Dcard

8000筆

餐廳食記

- 愛食記
- Ptt

33000筆

使用演算法

- Word2vec

使用演算法

- Word2vec
- KNN
- 協同過濾

使用演算法

- Word2vec
- KNN
- K-means

涉及個人著作隱私
該部分以口頭說明

資料分析

Data Analysis

報告人: 張為勳
報告人: 林德發
報告人: 李昀諭

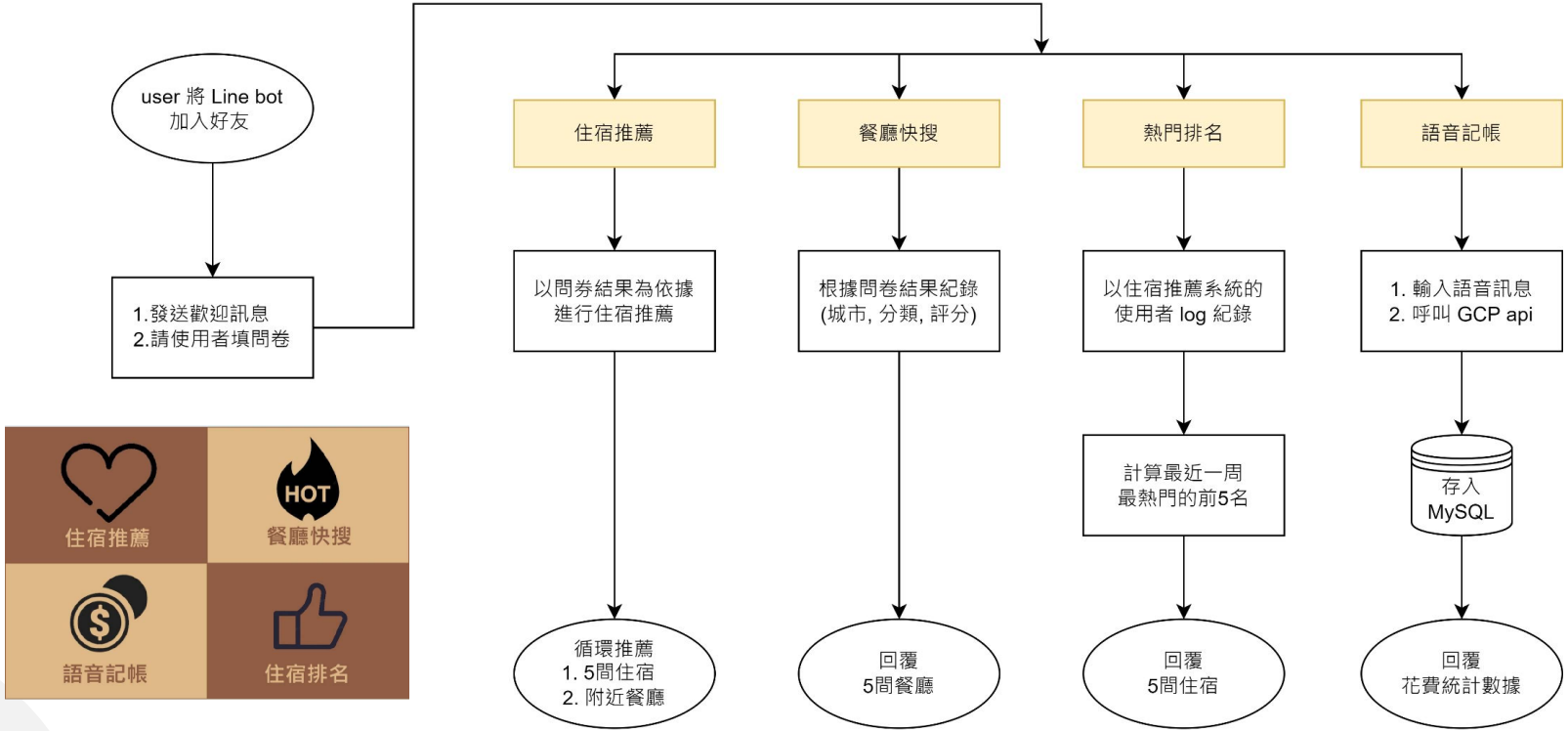
前端串接

Front-end with LINE bot

報告人: 黃彥宗
報告人: 陳儷文

涉及個人著作隱私
該部分以口頭說明

LINE bot 架構


user 將 Line bot
加入好友
1.發送歡迎訊息
2.請使用者填問卷
住宿推薦
餐廳快搜
熱門排名
語音記帳
以問券結果為依據
進行住宿推薦
根據問卷結果紀錄
(城市, 分類, 評分)
以住宿推薦系統的
使用者 log 紀錄
1. 輸入語音訊息
2. 呼叫 GCP api
計算最近一周
最熱門的前5名
存入
MySQL
循環推薦
1. 5間住宿
2. 附近餐廳
回覆
5間餐廳
回覆
5間住宿
回覆
花費統計數據
住宿推薦
HOT
餐廳快搜
語音記帳
住宿排名

功能展示: 語音記帳

遇到的困難:

- GCP api **只接受.wav檔**

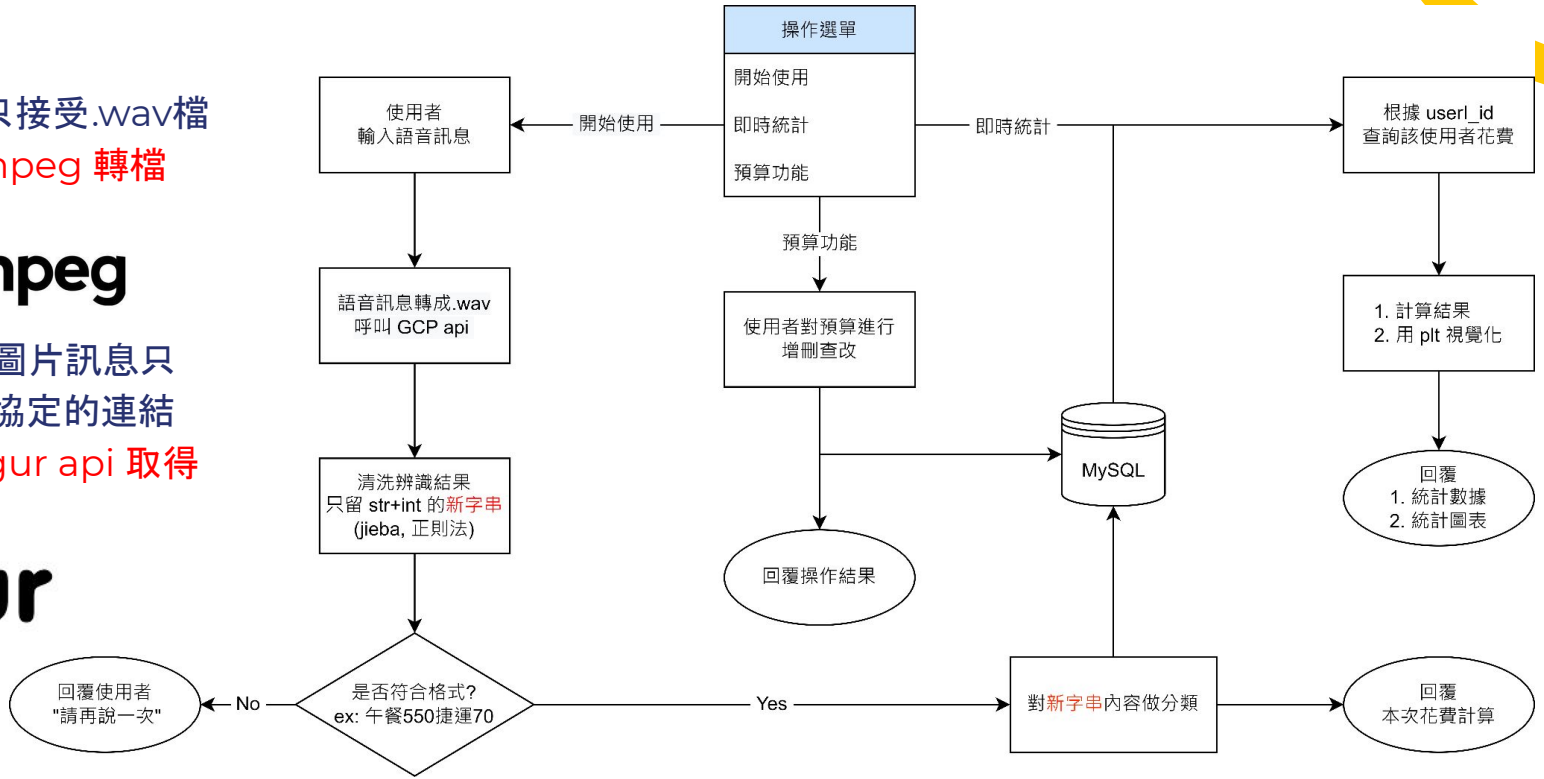
>> 使用FFmpeg 轉檔

FFmpeg

- LINE bot **圖片訊息只能放 https 協定的連結**

>> 使用 imgur api 取得圖片連結

imgur



語音記帳

FFmpeg

- 使用 user_id 命名音訊檔
- 在本地端進行轉檔 (.acc to .wav)

```python
# Convert .acc to .wav for GCP speech-to-text api
def covert_audio(user_id, rawAudio):
    rawAudioPath = './static/audio/{}.aac'.format(user_id)
    with open(rawAudioPath, 'wb') as f:
        for chunk in rawAudio.iter_content():
            f.write(chunk)
    newAudio = AudioSegment.from_file_using_temporary_files(rawAudioPath)
    newAudioPath = './static/audio/{}.wav'.format(user_id)
    newAudio.export(newAudioPath, format='wav')
    return newAudioPath
```

語音記帳

- 埋入 GCP speech-to-text api

```python
from google.cloud import speech_v1p1beta1 as speech

def call_gcp_speech_to_ext(audio_wav):
    client = speech.SpeechClient()
    with io.open(audio_wav, "rb") as audio_file:
        content = audio_file.read()
    audio = speech.RecognitionAudio(content=content)
    config = speech.RecognitionConfig(
        encoding=speech.RecognitionConfig.AudioEncoding.LINEAR16,
        sample_rate_hertz=16000,
        language_code="zh-TW",
        enable_automatic_punctuation=True
        )
    response = client.recognize(config=config, audio=audio)
    for i, result in enumerate(response.results):
        alternative = result.alternatives[0]
        return alternative.transcript
```

語音記帳 (即時統計)

imgur

- 使用 user_id 命名 jpg 檔案
- 將 jpg 上傳至 imgur
- 透過 imgur 取得 https 協定的圖片連結

```
# Get image URL with imgur_api
def get_image_url(user_id):
    imgur = pyimgur.Imgur(client_id=imgurClientId)
    image1 = imgur.upload_image(f'./static/app_accounting/pic_{user_id}.jpg',
                                title=f'pie_{user_id}_1')
    image2 = imgur.upload_image(f'./static/app_accounting/bar_{user_id}.jpg',
                                title=f'pie_{user_id}_2')
    image1Url = image1.link
    image2Url = image2.link
    return image1Url, image2Url
```

功能展示: 住宿排名

- 取得過去7天的使用者的log (住宿推薦系統的行為)
- 讀入成 pandas.DataFrame
- 進行統計計算

```
# Get last 7 days data from today
def get_week_log_data():
    today = datetime.today()
    year = datetime.today().year
    yesterdayData = pd.read_csv(
        f'./data/user_mainRS_log/{year}/mainRS_log_{today.year}-{today.month}-{today.day-1}.csv',
        encoding='utf-8',
        header=None)
    # Append last 6 days data to yesterdayData
    for i in range(2, 8):
        try:
            data = pd.read_csv(f'./data/user_mainRS_log/{year}/mainRS_log_{today.year}-{today.month}-{today.day-i}.csv',
                               encoding='utf-8', header=None)
            yesterdayData = pd.concat([yesterdayData, data], axis=0)
        except FileNotFoundError:
            pass
    return yesterdayData
```

功能展示: 住宿排名

- 取得過去 7 天的"**飯店名稱**", "**飯店網址**"
- 填入 LINE message 所需的 json

```
# Insert top5, top5_url into button json
def get_hotRank_button():
    hotRank_mkdir()
    top5 = get_top5_hotel()
    # top5 = [hotel_1, hotel_2, hotel_3, hotel_4, hotel_5]
    top5_url = get_top5_url(top5)
    # url  = [url_1, url_2, url_3, url_4, url_5]
    button = json.load(open('./line_bot_card/card_hotRank_button.json',
                            'r', encoding='utf-8'))
    for i in range(0, 5):
        button['footer']['contents'][i]['action']['label'] = top5[i]
        button['footer']['contents'][i]['action']['uri'] = top5_url[i]
    return button
```

功能展示: 餐廳快搜

- 依據使用者選取的住宿點的 **"所在區域"**
- 推薦該區域評分最高的5間餐廳

```
# Get area from user where he/she had chosen
def get_restaurant_data(area):
    host, user, pwd, db = get_mysql_config()
    conn, cursor = conn_mysql(host=host, user=user, pwd=pwd, db=db)
    sql = f'''
    SELECT res_name, image_url, article_url, address FROM test_restaurant
        WHERE area = "{area}"
        ORDER BY score DESC
        LIMIT 5;'''
    cursor.execute(sql)
    result = cursor.fetchall()
    restaurant_dict = {key[0]: key[1:] for key in result}
    close_conn_mysql(conn, cursor)
    return restaurant_dict
```

未來展望

Future Prospects

報告人: 黃彥宗

未來展望

建立大數據環境

將整體部屬在 HDFS, Spark 環境，以應對更大量的數據

根據使用者行為優化

將使用者 log 紀錄餵給模型持續訓練優化，讓推薦結果更貼近使用者

建立完整的即時儀表板

追蹤所有的使用者行為，並視覺化呈現，以做更完整的行為分析

開發影像辨識功能

透過使用者拍攝的景點照片搜尋附近適合的飯店,民宿等

THANKS

Email Address

黃彥宗 - davis0515@gmail.com

Github

張為勳	陳儷文	林德發	黃彥宗	李昀諭
QRcode	QRcode	QRcode		QRcode