報告題名：

旅遊推薦及快速旅程安排系統

Group Travel Recommendation and Trip Schedule System

作者：謝忠穎、張哲愷、鄭金君、陳佩貝

系級：資訊工程學系

學號：D0239438、D0239601、D0280170、D0208613

開課老師：陳奕中

課程名稱：專題研究

開課系所：資訊工程學系

開課學年：　105　學年度 第 1 學期

中文摘要

現在這個交通發達的時代，怎麼去玩已經不再是擔心的重點，大家反而是在煩惱哪裡可以去遊玩，為了解決大家這個困擾，我們設計這個網頁能夠讓大家輕鬆安排旅遊。為了要讓系統更完善，我們將目前市面上的旅遊規劃網站與我們撰寫的網頁來做衡量，將我們認為做得不夠好的地方改進，並且強化我們的優點，最後實作出旅遊推薦及快速行程安排系統。

我們設計的這個網站，主要包含三個部分－旅遊推薦、行程瀏覽以及旅遊書規劃，旅遊推薦是根據使用者(們)在一開始所填入的個人興趣指標，再從所有景點中找出最符合使用者興趣的地點，除了遊玩景點的推薦，更添加了附近飯店的資訊，讓使用者在安排行程的同時也能夠順邊尋找附近的飯店，不用在額外花時間在找尋飯店上。行程瀏覽是由使用者選擇有興趣的景點以及飯店之後，將所選擇的景點以及飯店按照出發的時間安排成一個完整的行程。而旅遊書規劃提供使用者自定義的旅遊書編輯畫面，讓使用者在實際出遊時能夠快速瀏覽而不必在自己製作筆記。且我們的旅遊書提供下載列印的功能，可以讓你在旅遊時帶著走順便記錄旅遊的過程。

關鍵字：行程安排、旅遊推薦、旅遊書、R-Tree、Skyline Queries

Abstract

Nowadays we have a well-developed transportation system. People never worried about where they can go, but where to go. In order to deal with this problem we design this website for people to schedule their trip more effectively . We compare all trip website on the internet with ours to improve the weakness and strengthen advantage for building this system.

There are three parts of our website. Trip Advisor、 Schedule Review、 Travel Book editor. Trip Advisor is according to the Interest of people fill in our system through our system. We can find the most suitable attractions to give people advice. Besides Trip Advisor we also add information about hostel nearby. Let our user more convenience. Schedule Review can help people schedule their trip. And easy to review it. Travel Book editor will auto-scan your schedule and convert your schedule to a beautiful travel book. You can also custom it by yourself. Our Travel Book providing you download it and you can take it with you on your trip.

Keyword：Schedule、Trip Advisor、Travel Book、R-Tree、Skyline Queries

目　次

第一章 序論

1.1 研究動機

在現在科技日新月異之下，科技的進步速度超乎我們想像，傳統的旅遊都是透過人們口耳相傳哪裡有好玩的地點，現代則是因為網路的出現而導致訊息量爆炸，網路上隨便搜索都有很多旅遊的資訊，這些資訊讓我們在選擇旅遊地點時有更多的選擇，但是要去過濾如此大量的訊息會耗費很多的時間，而且往往很難找到真正適合我們需求的資料。我們的專題計畫便是從這裡得到動機，想去設法解決人們在出遊時可能會遇到的問題，並透過科技的力量來解決傳統旅遊的問題。

根據觀光局的統計資料，國人如何規劃國內旅遊中，自助規劃比例將近有90%，幾乎每位國人都是自行規劃個人的旅遊行程的。但是網路上搜索到的資料往往都只有部落客或是一些內容農場的推薦，常常看了一堆內容卻沒有一個符合自己的興趣。在現有的系統中很少有針對個人興趣去做景點推薦系統，因此我們的首要目標便是建立個人化的旅遊推薦。除了推薦系統外之外我們在出遊時也需要一個可以方便規劃自己行程的旅遊手冊，除了能夠自己在旅遊時隨時查看外，也能夠提供給一同出遊的朋友。綜合了這些因素，我們決定開發一個能夠根據使用者的興趣去推薦他們景點，並能夠擁有安排旅遊行程功能的旅遊推薦及行程安排系統。

1.2 專題特色

以下是對我們從網頁搜尋所挑出來的兩個熱門網站與我們的旅遊系統所做的比較

1.2.1 旅遊資訊王

圖 1.1 旅遊資訊王網站

一進入網頁後，五花八門的資訊映入眼前，需要仔細的尋找才能夠找到我們所想要的景點搜尋，並且只能針對一個點或一個縣市來做搜尋的作用，完全無法針對使用者的個性來為給予完美的景點推薦。

1.2.2 雄獅旅遊網

圖 1.2 雄獅旅遊網

過多的商業化行程，沒有考慮到出遊最重要的品質，鮮少的輸入資訊，在輸入目的地後所得到的結果大多是除了主要景點一樣，其他的都被預訂好了，對有特定興趣的使用者來說相當不便。

1.2.3 本專題的旅遊推薦系統

為了因應每位使用者的興趣需求不同，我們系統在一開始即蒐集使用者對我們對既有景點所分成的四大類的個人評分，再藉由將使用者評分之結果來與所有景點來做相似度分析，進而產生出最符合每一位使用者的景點。因為台灣的景點非常多，市面上大部分的旅遊推薦採用的是暴力破解，將每一個景點的評分一一與使用者來做比較，那所花費的時間往往消磨使用者耐心，又或者是推薦給使用者的景點並沒有數據上的依據只是人工選擇或是隨機推薦的。這些都無法有效的提供給使用者他們喜歡的景點。因此我們做法是先以 R-tree 的資料結構將既有景點的分數來做安排，再藉由 The spatial Skyline Queries 來快速找出最符合使用者的景點，透過這些方法在搜尋的時間亦或是景點的精準度，都比市面上的產品來得更有優勢。

1.2.4 本專題的快速旅程安排系統

市面上的旅程安排系統大多不易操作且過程複雜，但是我們的系統僅須透過簡單的三個步驟，選擇景點、安排行程、選擇飯店即可快速完成旅遊安排，完程安排後系統會自動輸出每天的行程規劃以及的旅遊路線規劃。

1.2.5 本專題的專屬旅遊書

為了讓使用者省下製作旅遊書的時間成本，本系統將會利用使用者安排的景點自動生成一本旅遊書，旅遊書中也自備許多小工具，讓使用者可以自製專屬於的客製化旅遊書，並且保存起來，以便之後的修改。使用者也可以將自行製作的

旅遊書印出，讓旅遊時不需要攜帶厚重的書籍，或是依靠手機，也可以輕鬆出遊。旅遊過程中使用者也可以將自己的旅遊日記寫進旅遊書中，增添旅遊的趣味。

1.3 工作分配及甘特圖

表 1.1 工作分配表

工作內容	工作分配
需求分析	謝忠穎、陳佩貝、張哲愷、鄭金君
蒐集資料	謝忠穎、陳佩貝、張哲愷、鄭金君
網頁撰寫	謝忠穎、陳佩貝
PHP 撰寫	鄭金君、張哲愷
書面報告撰寫	謝忠穎、陳佩貝、張哲愷、鄭金君

表 1.2 甘特圖

工作項目	二月	三月	四月	五月	六月	七月	八月	九月	十月	十一月	十二月
與老師討論											
蒐集資料											
需求分析											
初步分析											
細部分析											
撰寫程式											

第二章 相關文獻回顧

2.1 Index Tree

2.1.1 M-tree

M-Tree 是依照各點的相似度做分組，並依照點和點之間的 Euclidean 距離，決定節點要被放入的圓圈為何，如圖 2.1 所示，為了說明方便，我們將以二維來論述，三為以上的例子則可由此例推導而得，假設二維空間上有兩點 $a(x_1,y_1)$與 $b(x_2,y_2)$，這兩點的 Euclidean 距離如公式 Eq. (1)

$$d(a,b) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

在建立樹狀結構時，M-Tree 會紀錄每一個圓圈的資訊，在 M-Tree 中的葉節點會紀錄節點以及節點與包含自己所屬的圓的圓心距離，在 M-Tree 的內部節點中會紀錄包含他們的圓心節點、圓的半徑以及那一個 level 中節點與包含自己所屬的圓的圓心距離，如圖 2.2 所示

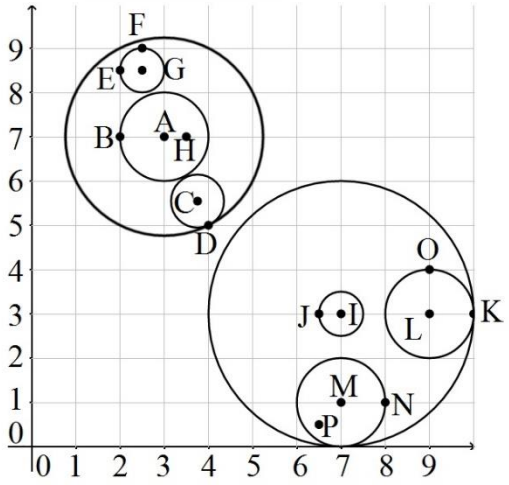

圖 2.1 M-Tree 索引結果

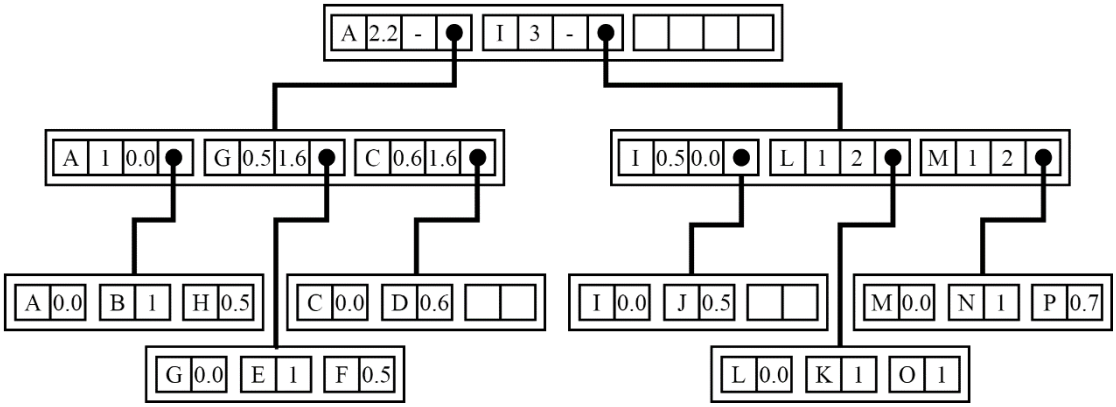

圖 2.2 M-Tree 樹狀架構圖

2.2 Heap

Heap 為資料結構中以樹型態為基礎儲存數據的結構，則對於整個 Heap 而言，假設 A 為 B 之父節點，則 A 之鍵值將被依大小相對於 B 節點排列，而 Heap 可分為最大 Heap 或是最小 Heap，再最大 Heap 內，父節點之鍵值必大於或是等於他的子節點，再最小 Heap 內，父節點之鍵值必小於或是等於他的子節點，而 Binary Heap 為 Heap 最常見亦是最有效率的一種並且 Heap 同時也是一棵完整樹，參考圖 2.3。

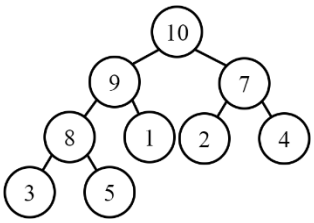

圖 2.3 Heap

2.2.1 Heap 新增

Heap 之新增為先將一欲新增的資料加到此樹的最後一個節點位置，接著再經由將最末節點逐一比對父元素與子元素的鍵值，大的當父節點，小的當子元素，以維持最大 Heap 的規則，詳細參考圖 2.4

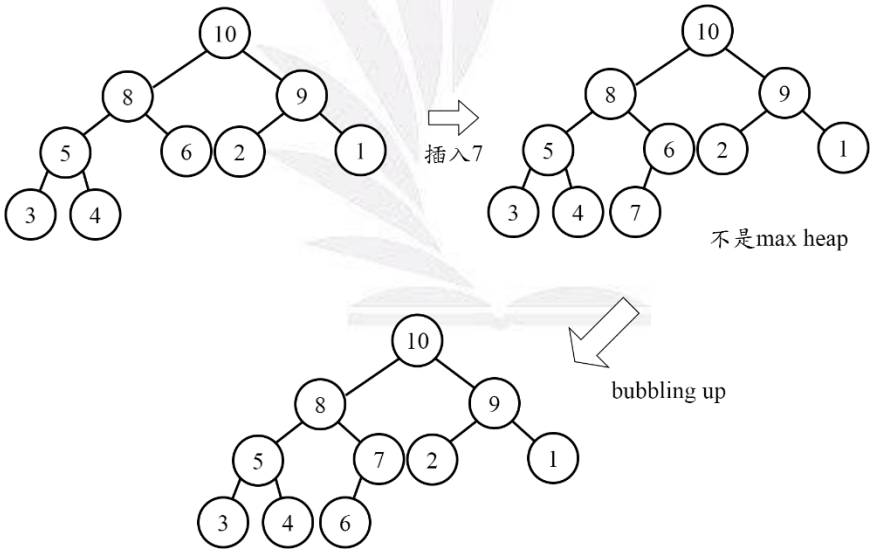

圖 2.4 Heap 之新增步驟

2.2.2 Heap 刪除

Heap 之刪除為先取出最大節點，接著取出最末的一個節點 Temp 加到根節點，由根節點往下找一個適合放置 Temp 之位置，每次判斷左右節點走向鍵值大的節點，詳細參考圖 2.5

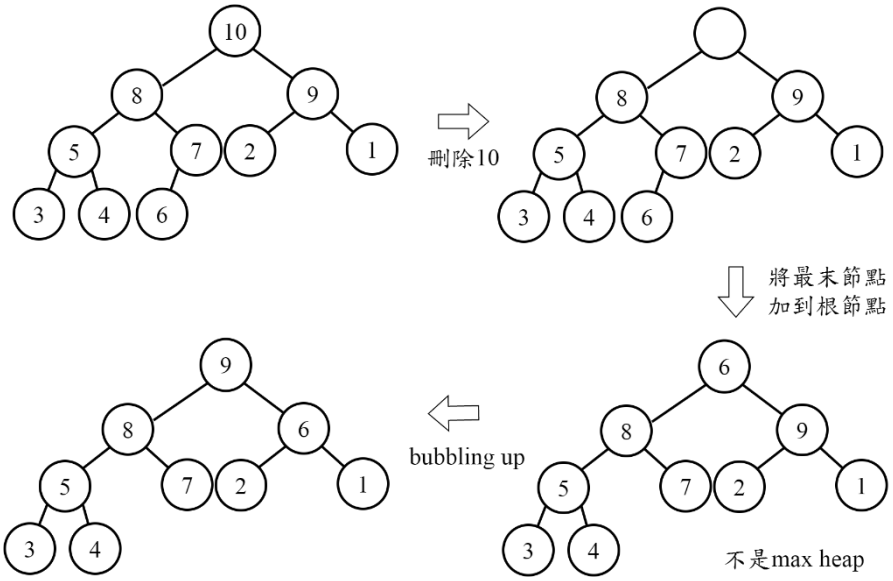

圖 2.5 Heap 之刪除步驟

2.3 The Skyline Queries

為了在大量數據中有效率且準確的找出目標，天際線查詢，給定一組欲查詢的指令資料與 R-tree 之 Root，將依照資料節點不會被其他的資料節點所支配，以表 2.1 各景點與海邊之距離及價錢的二維資料庫為範例，資料點 *F(3,75)*支配了資料點 *D(4,125)*，所以我們在選擇資料點的時候如果先取得了 *D(4,125)*點，再找到了 *F(3,75)*點，我們將會捨棄 *D(4,125)*點而保留 *F(3,75)*點，原因是因為 *D(4,125)* 資料點距離我們欲查詢目標點 4 英里，而花費為$125，接著當我們找尋到 *F(3,75)* 點距離我們現在 3 英里，花費為$75，在不考慮其他內在因素來說，*F(3,75)*點的價值度是比 *D(4,125)*點還要高的，故我們會選擇 *F(3,75)*點而不是 *D(4,125)*資料點，如圖 2.6 所示，因為 *F(3,75)*點在 *D(4,125)*左下方 ，也可以稱之為 *D(4,125)* 資料點被 *F(3,75)*支配。

表 2.1 各景點與海邊之距離及價錢

object	**distance(mile)**	**price($)**
A	0.5	200
B	2	150
C	2.5	25
D	4	125
E	1.5	100
F	3	75

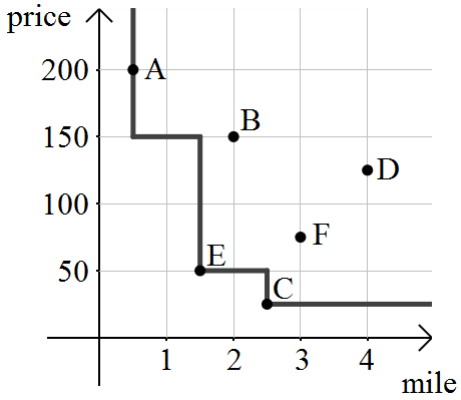

圖 2.6 各景點之天際線查詢

第三章 系統開發平台

現在手機市場的 OS 主要分為兩個主要的分類，由 Apple 的 IOS 以及 Google 的 Android 佔市佔率大宗，也因為如此，當你寫一個 App 時就必須面臨抉擇，到底要從 IOS 亦或是 Android 下手呢，因為這就將代表缺少你所選擇系統另一陣營的使用者，為了讓廣大使用者可以跨系統跨裝置的方便使用，我們選擇以 Visual Studio 來開發 Web Application 實作我們的旅遊推薦及快速旅程安排系統，並搭配一定 RWD 設計，讓使用者在手機端也可以輕鬆的使用我們的系統。

3.1 Visual Studio

Visual Studio 是微軟公司所開發的可插入套件的產品，意思就是，你欲完成哪一個方向的產品僅僅只要安裝那個產品的開發工具即可，Visual Studio 會幫忙處理環境的問題，涵蓋了大部分的軟體生命週期所需的工具，所以說 Visual Studio 是一個可完整開發的工具集，除了一些如 UML 或是 IDE 等工具外，亦支持 Web 應用程式的工具，譬如我們專題所用到的 PHP 即是一個例子，並提供開發及部屬相關的技術。

3.2 PHP

PHP 是一種通用的語言，適用於可嵌入 Html 的網路開發，而且繼承 C 語言及 JAVA 的一些特點，有助於開發者能夠快速上手，除此之外也被應用到許多領域上。

也因為 PHP 可以應用的範圍十分廣，在網頁伺服器上面執行更是佔絕大多數，因此使用者可以透過 PHP 程式碼來產生出可以瀏覽的網頁，也因為是免費的，吸引很多使用者來使用。

3.3 SQL Server

SQL server 是 Microsoft 公司所推出的關聯式資料庫，雖然資料庫語言原是採用國際標準組織所訂定之 SQL 語言，但是微軟對其擴充成大型之資料庫管理。

3.4 NuGet

NuGet 是一個開放原始碼的 package 的平台，而 NuGet 也是 Visual Studio 擴充的其中一種功能，而 NuGet 亦可在命令列中使用，並通過腳本自動執行。

第四章 系統架構與步驟

本章主要說明的 UML 圖有:使用案例圖、循序圖、狀態圖和類別圖,這些 圖用來說明我們系統的架構。還有包括資料庫的實體關係圖和關聯圖。

4.1 系統架構

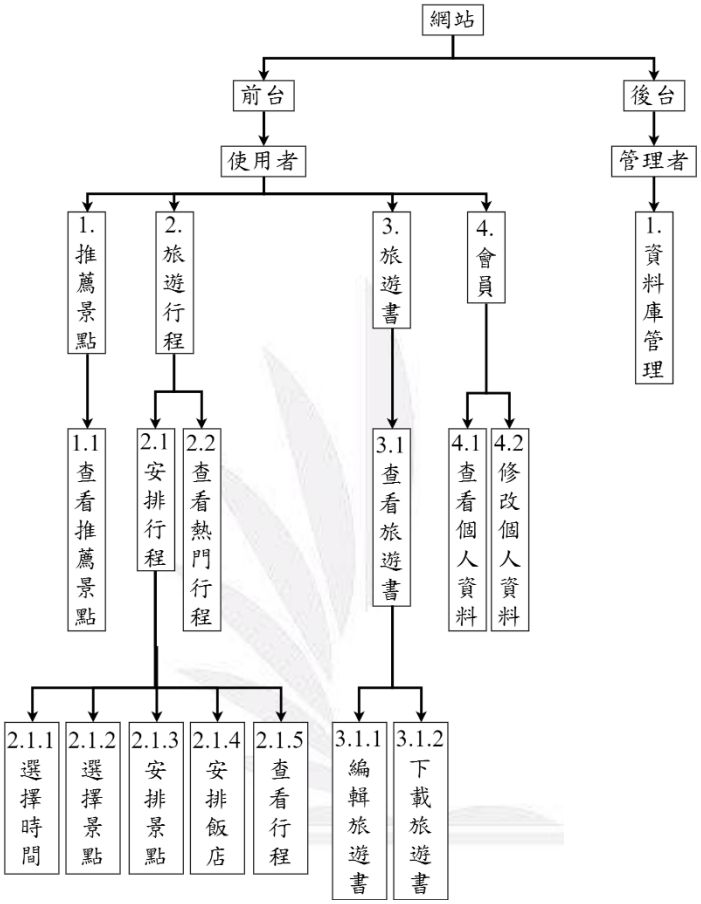

圖 4.1 系統架構圖

4.2 使用者需求 Use Case Diagram

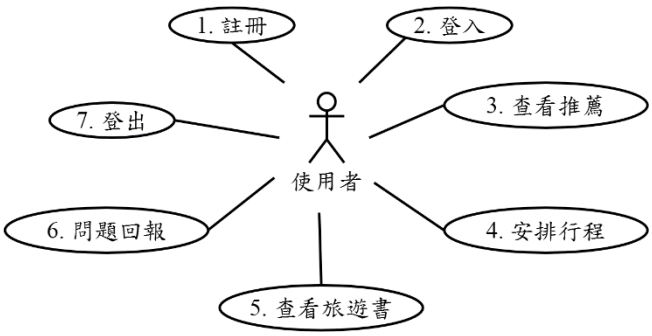

圖 4.2 使用者 Use Case Diagram

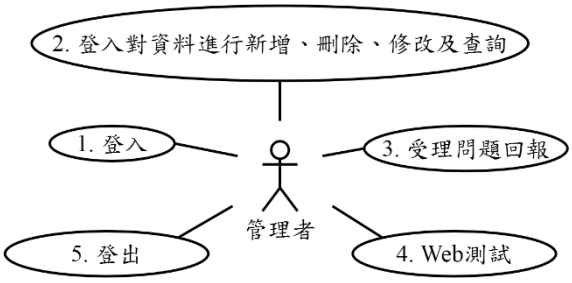

圖 4.3 管理者 Use Case Diagram

4.3 功能流程圖

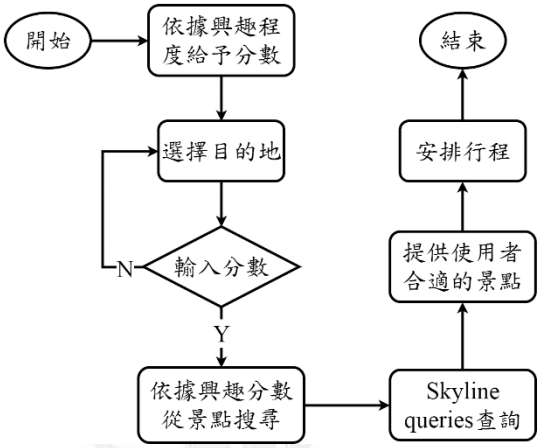

圖 4.4 推薦系統流程圖

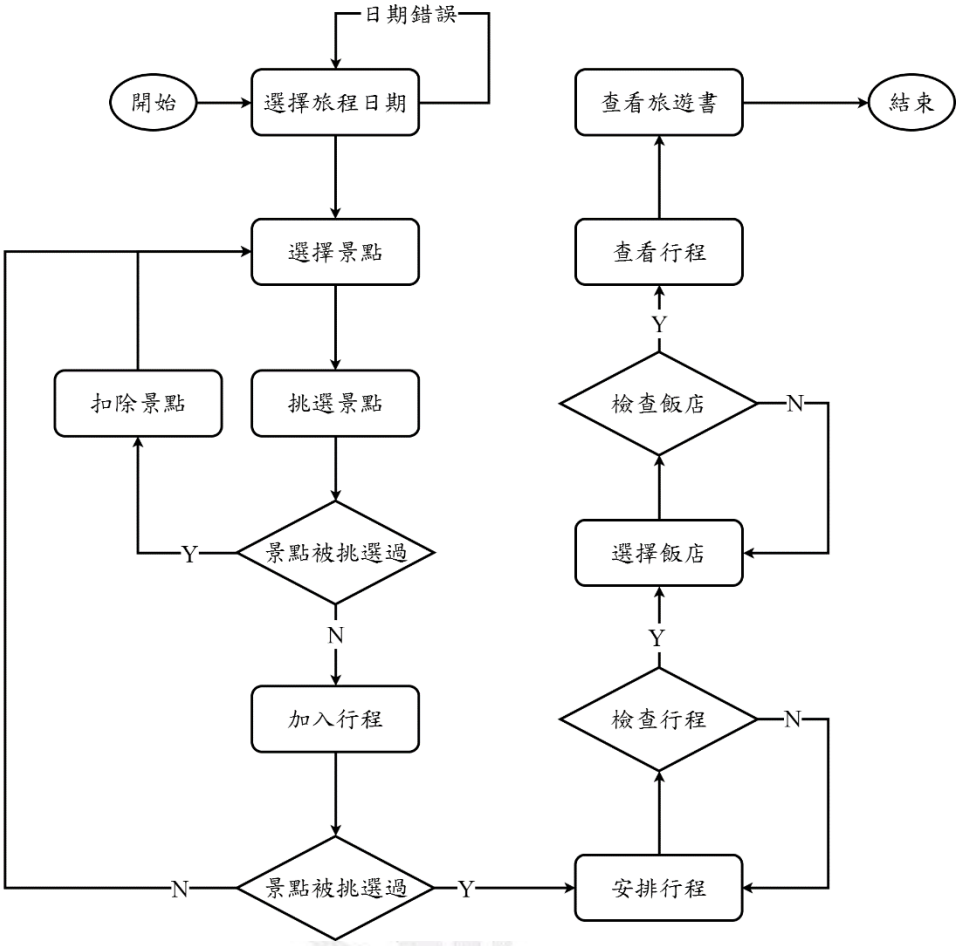

圖 4.5 旅遊行程安排系統的操作流程圖

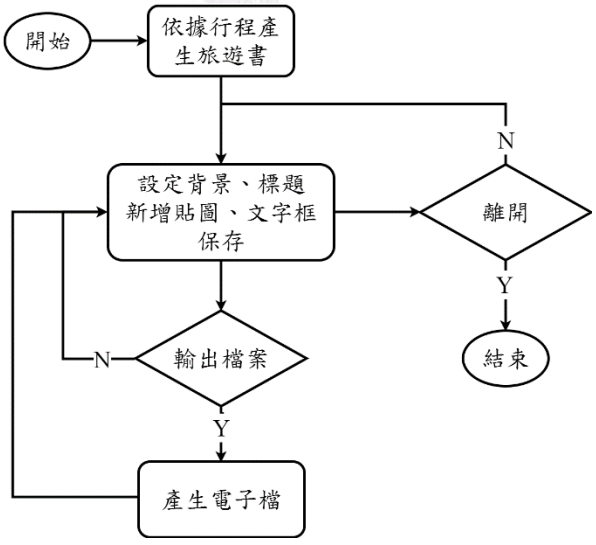

圖 4.6 客製化旅遊書系統流程圖

4.4 資料庫

4.4.1 資料庫關聯圖

根據圖 4.7，此關聯圖又稱為 ER-Model，是用來表述此種模型的每個資料的關聯，應用實體與關係以及屬性，可以快速的用來展示所有參與集合的關聯度。

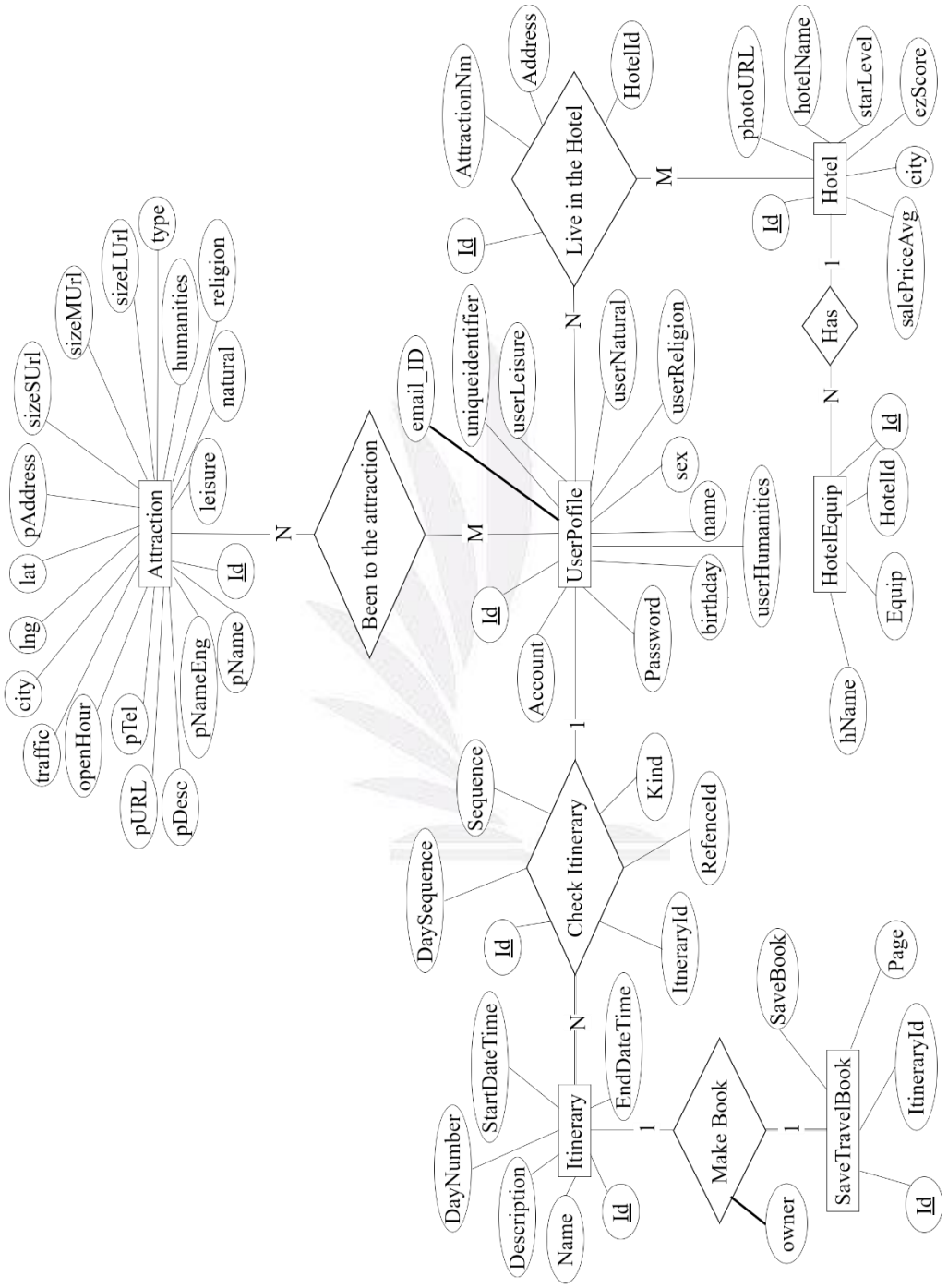

圖 4.7 旅遊推薦及快速旅程安排系統的資料庫關聯圖

4.4.2 資料庫大綱圖

根據圖 4.8，資料庫可以建立一個或多個圖表清楚的說明目前資料庫中的資料表、資料欄位、及索引鍵和他們的關聯性

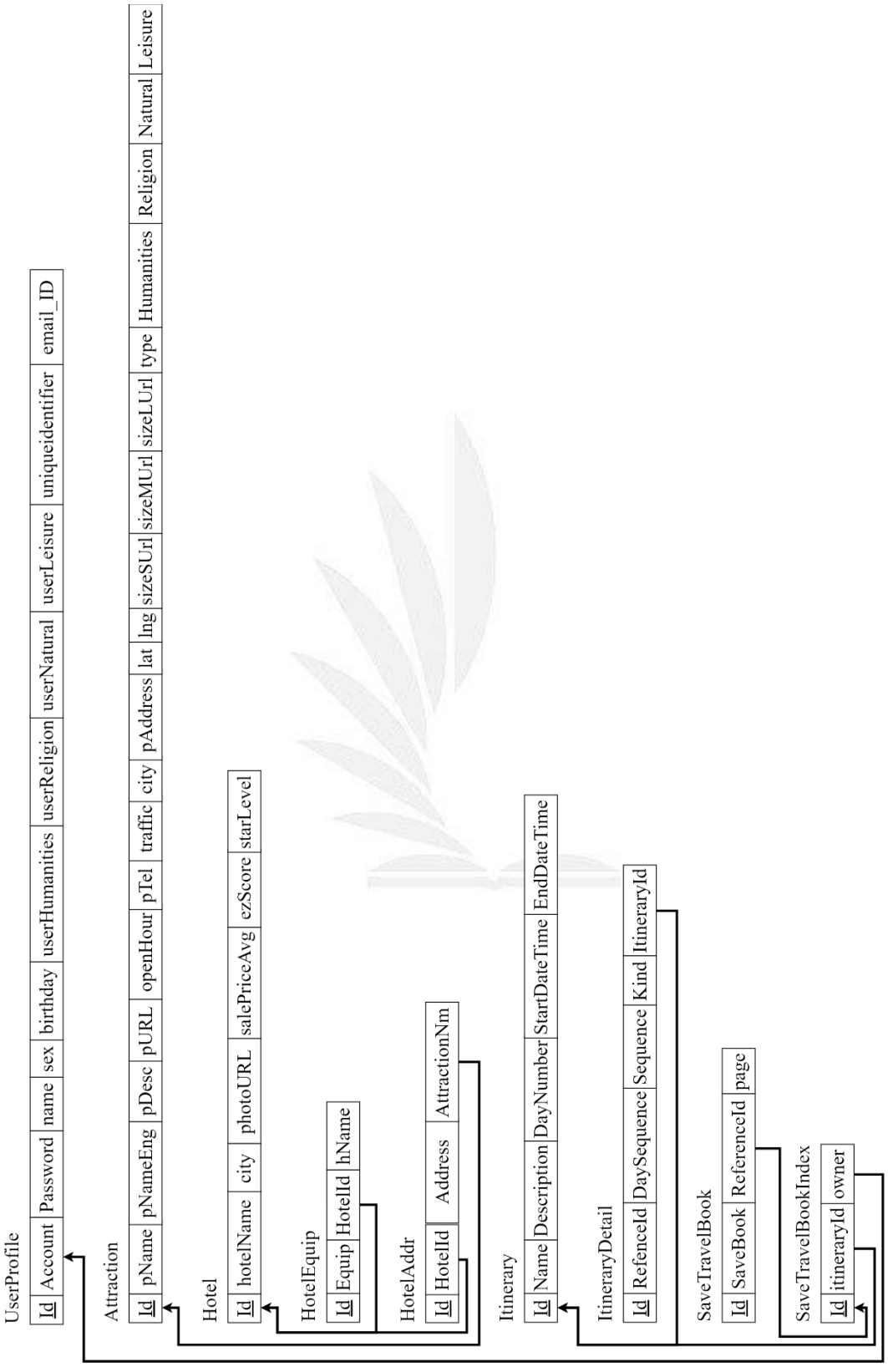

圖 4.8 旅遊推薦及快速旅程安排系統資料庫大綱圖

4.4.3 旅遊推薦及快速旅程安排系統的資料庫欄位

表 4.1 UserProfile 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
Account	nvarchar(50)	帳號	NOT NULL	
Password	nvarchar(50)	密碼	NOT NULL	
name	nvarchar(50)	使用者名稱	NOT NULL	
sex	nvarchar(50)	性別	NOT NULL	
birthday	date	出生年月日	NULL	
userHumanities	float(53)	人文興趣分數	NULL	
userReligion	float(53)	宗教興趣分數	NULL	
userNatural	float(53)	自然興趣分數	NULL	
userLeisure	float(53)	休閒興趣分數	NULL	
uniqueidentifier	nvarchar(50)	是否有確認認證信	NULL	有：true 沒有：NULL
email_ID	uniqueidentifier	識別碼	NOT NULL	

表 4.2 Attraction 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
pName	nvarchar(50)	中文名稱	NULL	
pNameEng	nvarchar(600)	英文名稱	NULL	
pDesc	nvarchar(MAX)	簡介	NULL	
pURL	nvarchar(600)	網址	NULL	
openHours	nvarchar(MAX)	開放時間	NULL	
pTel	nvarchar(50)	連絡電話	NULL	
traffic	nvarchar(MAX)	交通方式	NULL	
city	nvarchar(50)	位於城市	NULL	
pAddress	nvarchar(600)	地址	NULL	
lat	float(53)	緯度	NULL	
lng	float(53)	經度	NULL	
sizeSUrl	nvarchar(MAX)	小型照片	NULL	
sizeMUrl	nvarchar(MAX)	中型照片	NULL	
sizeLUrl	nvarchar(MAX)	大型照片	NULL	
type	nvarchar(50)	景點類型	NULL	
humanities	float(53)	人文興趣分數	NULL	
religion	float(53)	宗教興趣分數	NULL	
natural	float(53)	自然興趣分數	NULL	

leisure	float(53)	休閒興趣分數	NULL

表 4.3 Hotel 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
hotelName	nvarchar(50)	中文名稱	NULL	
city	nvarchar(50)	位於城市	NULL	
photoURL	nvarchar(600)	照片網址	NULL	
salePriceAvg	nvarchar(50)	平均價格	NULL	
ezScore	nvarchar(50)	飯店評鑑分數	NULL	
starLevel	nvarchar(50)	飯店星等	NULL	

表 4.4 HotelEquip 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
Equip	nvarchar(50)	設施	NULL	
HotelId	int	對應到 Hotel 資料表的 Id	NOT NULL	
hName	nvarchar(50)	飯店名稱	NULL	

表 4.5 HotelAddr 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
HotelId	int	參考到 Hotel 資料表的 Id	NOT NULL	
Address	nvarchar(600)	飯店地址	NULL	
AttractionNm	nvarchar(600)	對應到 Attraction 資料表的 Id	NULL	

表 4.6 Itinerary 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
Name	nvarchar(100)	行程名稱	NOT NULL	
Description	nvarchar(MAX)	行程簡介	NULL	
DayNumber	int	天數	NULL	
StartDateTime	datetime	開始日期	NULL	
EndDateTime	datetime	結束日期	NULL	

表 4.7 ItineraryDetail 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
RefenceId	int	參考到 Attraction 或是 Hotel 的 Id	NOT NULL	
DaySequence	int	第幾天	NOT NULL	
Sequence	int	早中晚	NOT NULL	早：0 中：1 晚：2
Kind	int	判斷是景點還是飯店	NOT NULL	飯店：0 景點：0 飯店：1
ItineraryId	int	參考到 Itinerary 資料表的 Id	NOT NULL	

表 4.8 SaveTravelBook 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
SaveBook	text	旅遊書該頁程式碼	NULL	
ReferenceId	int	參考到 SaveTravelBookIndex 資料表的 Id	NOT NULL	
Page	int	頁數	NOT NULL	

表 4.9 SaveTravelBookIndex 資料表

屬性名稱	資料格式	中文名稱	NULL	備註
Id	int		NOT NULL	Primary key
itineraryId	int	參考到 Itinerary 資料表的 Id	NOT NULL	
owner	nvarchar(50)	擁有者，參考到 UserProfile 的 Account	NULL	

第五章 演算法

本計畫的演算法包含兩個部分(1)對景點建立 R-tree，(2)利用熱門的多條件查詢演算法 —天際線查詢推薦適合使用者的節點及其包含的景點給使用者參考

5.1 建立 R-tree

5.1.1 R-tree 的架構

R-tree，是一種高度平衡的樹，用來做特殊空間搜尋動態索引的結構。其中心概念為將所有距離相近的節點聚在一起並且在父節點端紀錄其相聚節點所形成之最小矩形，可以用樹狀圖表示之，如圖 5.1。子節點上的每一個都代表是一堆矩形所聚合而成的矩形，在最底層才是一個個的資料點，由下而上代表著越來越大的矩形，由此可以推論出越上層的矩形與欲查詢的目標相似度越低。

R-tree 在每一層都有可儲存之容量上限，當超過上限就會分割成兩個最小矩形，進而保證此 MBR 內的每個點是更高度的相關。在查詢的過程中，與欲查詢的資料點的矩形沒有相交的查詢一定與資料點也都不相交，因此 R-tree 能夠大大的減少搜尋速度。

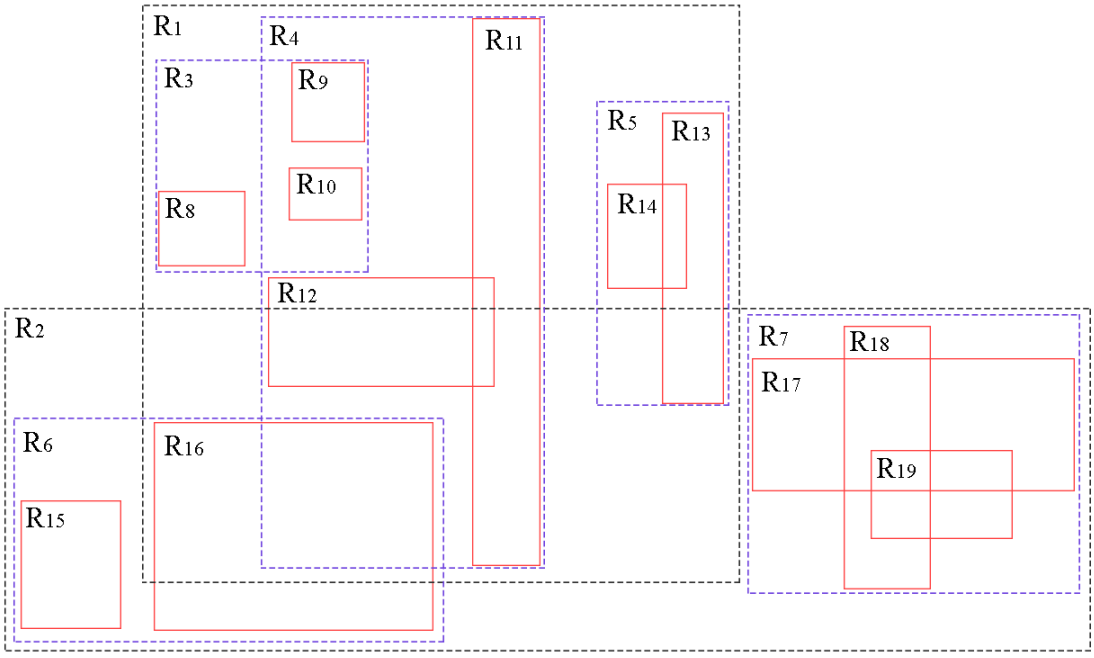

圖 5.1 R-tree 二維概念圖

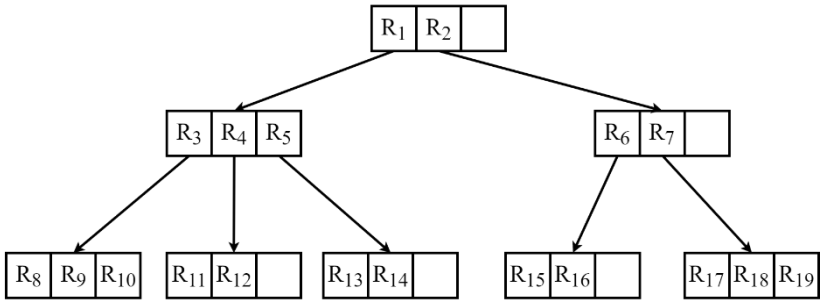

圖 5.2 R-tree 樹狀架構圖

5.1.2 R-tree 相關演算法

建立 R-tree 之後，對於每一個新加入的資料點，都必須要尋找合適的 MBR，再判斷此 MBR 是否超過上限，如果沒有滿就將資料點插入此 MBR，如果滿了就進行分割的動作，如圖 5.3 的流程圖所示。

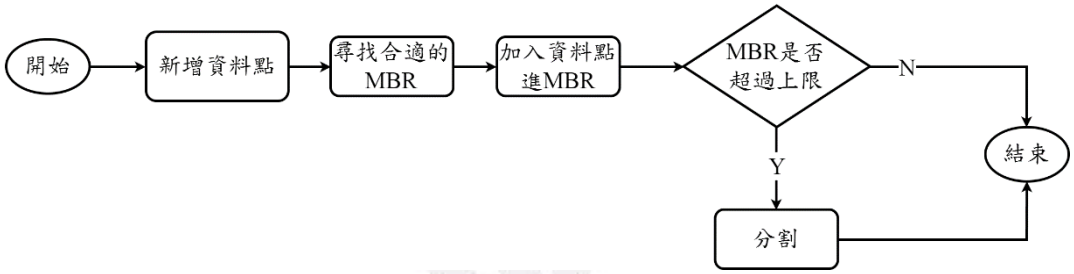

圖 5.3 建立 R-tree 流程圖

5.1.3 搜尋的演算法

當要搜尋一個資料點的時候，會從 root 開始往下搜尋，搜尋時只會搜尋和資料點範圍有重疊的 child node，直到找到該筆資料。

5.1.4 插入的演算法

插入資料點時，會從 root 開始做走訪，找到目前 node 裡增加面積最少的 rectangle，在從此 rectangle 所屬的 node 重複做此動作，直到 leaf node 為止，然後將此點加入找到的 rectangle 裡，若此 rectangle 滿了，就必須 spilt。

5.1.5 分割的演算法

當 rectangle 滿的時候，就會將此 node 裡的 rectangle(或資料點)平均分割，造出兩個最小的 rectangle，同時我們也要檢查看看 parent node 的 rectangle 的數量是否超過上限，若超過時，就必須重複此動作，直到數量都在規定的範圍裡。如圖 5.4 所示。

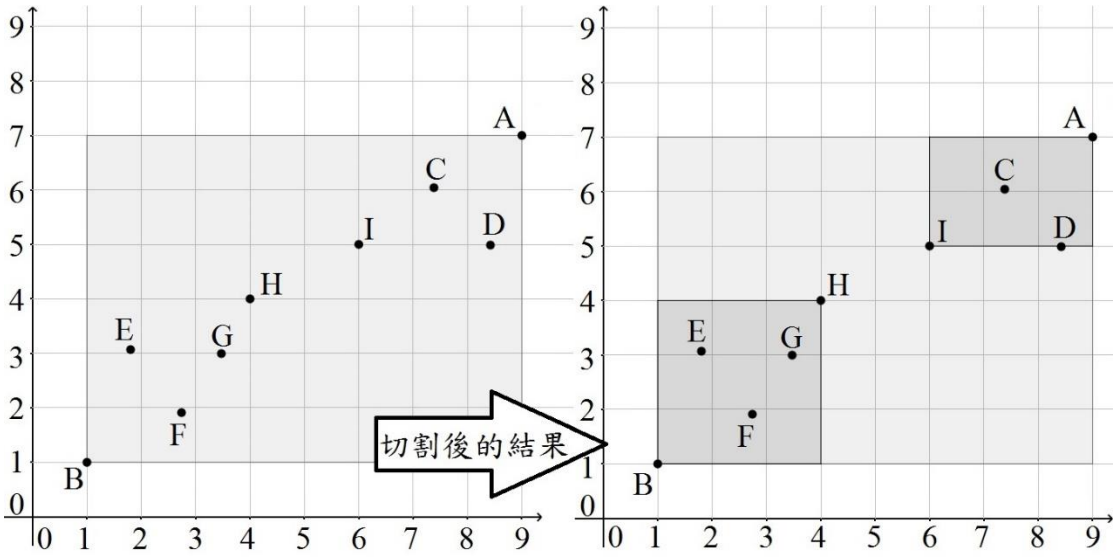

圖 5.4 分割 R-tree 範例

5.1.6 利用 R-tree 對地點資料建立索引

利用 R-tree 對各個景點做建立多維度的索引，包含類型評分，有人文、宗教、自然、休閒，還有地理位置，讓具有相關性的景點全都儲存在同一個矩形內，當使用者(團體)資料(興趣、想前往的地理位置)與某一節點高度相關時，即可保證此節點內的所有景點串成的行程將會讓使用者滿意。

5.2 利用天際線演算法搜尋

5.2.1 Branch and Bound Skyline algorithm (BBS)

BBS 演算法是目前最具效率的天際線搜尋方法之一，此方法是透過對資料建立索引來增加天際線的搜尋速度。BBS 演算法是基於利用 R-tree 的 Minimum Bounding Rectangle (MBR)將鄰近的資料點利用 MBR 包圍起來。如圖 5.5，只有與天際線有交集的 MBR 會被取出使用，而其他 MBR 裡的資料點都不會被檢查到，藉由此方法可以加快天際線的查詢速度。

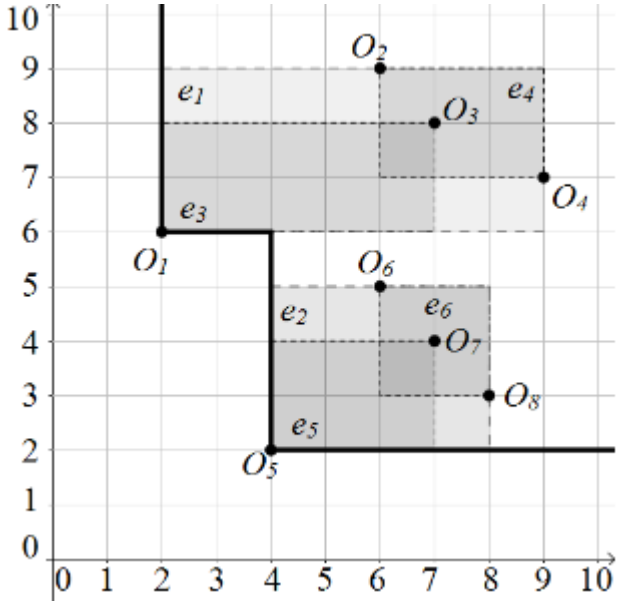

圖 5.5 利用 MBR 的 BBS 查詢

5.2.2 天際線演算法查詢

為了有效的使用天際線查詢，額外新增一條 Heap 所實作的為 S 之串列，我們將一開始 R-tree 的根結點放進 Heap 內，接著將 Heap 取出根節點所代表的最外一層矩形分解開來，意即將子節點一一放入 Heap 內，接著將 S 串列的內容與 Heap 所取出之資料點一一比較，看看有無被支配，如果被支配即將 S 串列內該資料點刪除，沒有被支配即將該資料點加入 S 串列，持續循環直到沒有任何點存在於 Heap 中。

第六章 實證分析

6.1 首頁

圖 6.1 首頁的快速安排

首頁我們在畫面正中央只放上一顆按鈕，讓第一次操作的使用者也能很直覺的知道該如何開始操作這個網頁。在主畫面的規劃我們也選用比較乾淨的色彩，希望使用者進入我們網頁時可以感覺到出遊一般的放鬆心情。

圖 6.2 首頁的超連結

此處是我們網站的三大重點分別是旅遊推薦，安排行程以及在線觀看旅遊書，如果是有操作我們網頁的使用者，便可以很輕鬆得再這裡直接進行自己要操作的功能。

6.2 熱門景點

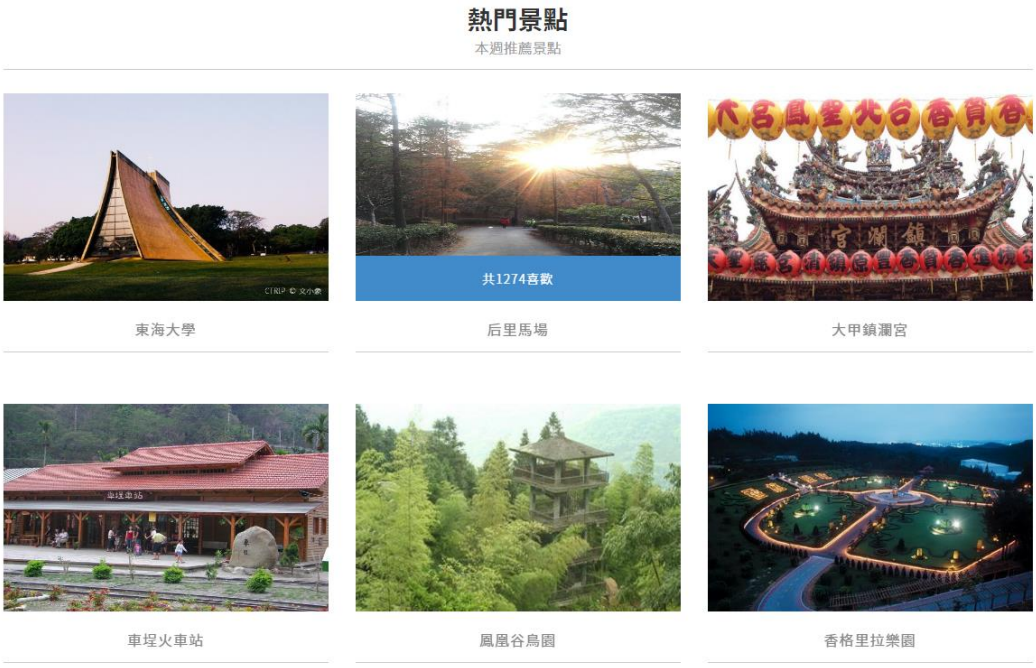

圖 6.3 熱門景點

在查詢熱門景點時，可以看到目前時下最多人喜歡的景點，在你想出去玩卻毫無頭緒時，熱門景點提供你一些點子也許能從中找到適合自己的景點。

圖 6.4 點擊景點後回傳之景點資訊

圖 6.5 熱門旅館

除了話題性景點之外，熱門旅館當然也少不了，一整天玩得開開心心想必住的地方也不能馬虎，而踩到了地雷壞了一整天的好心情，這裡可以幫使用者免除遇到評價不佳的旅店。

6.3 規劃行程

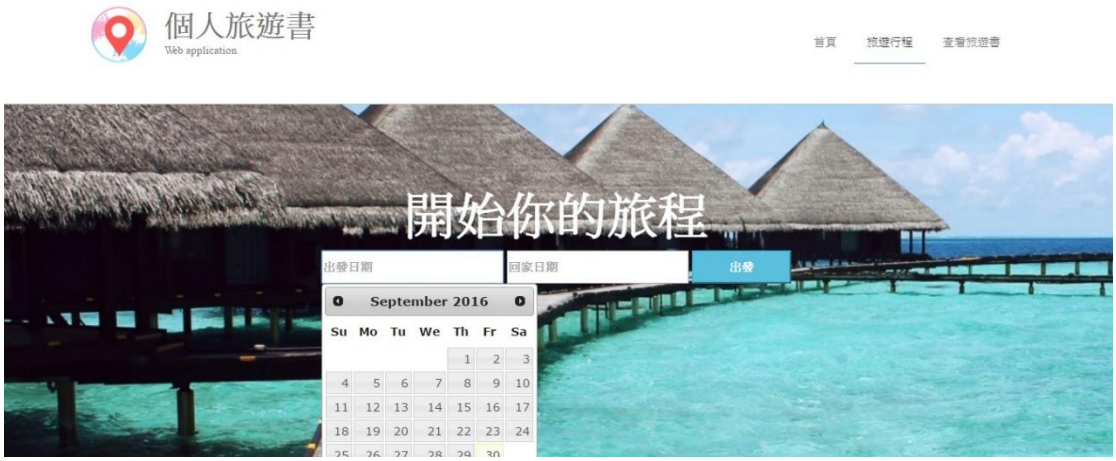

圖 6.6 選擇時間

選擇欲出遊的日期，以及回程的時間，我們系統就會在接下來的步驟幫你做旅遊排程的規劃。

圖 6.7 選擇景點

圖 6.8 加入景點

選擇好天數之後，接著在搜尋的地方選擇想要去的城市、或是直接鍵入想要去的景點，然後點下加入行程，即可將所有想去的地方通通加入。

圖 6.9 已選擇的景點

將所有想去的地方挑選起來之後，每個景點會呈現在最上方，如是想要刪除，僅需按下景點旁邊的叉叉即可。選擇完成後按下一步，即可進入下一個步驟。

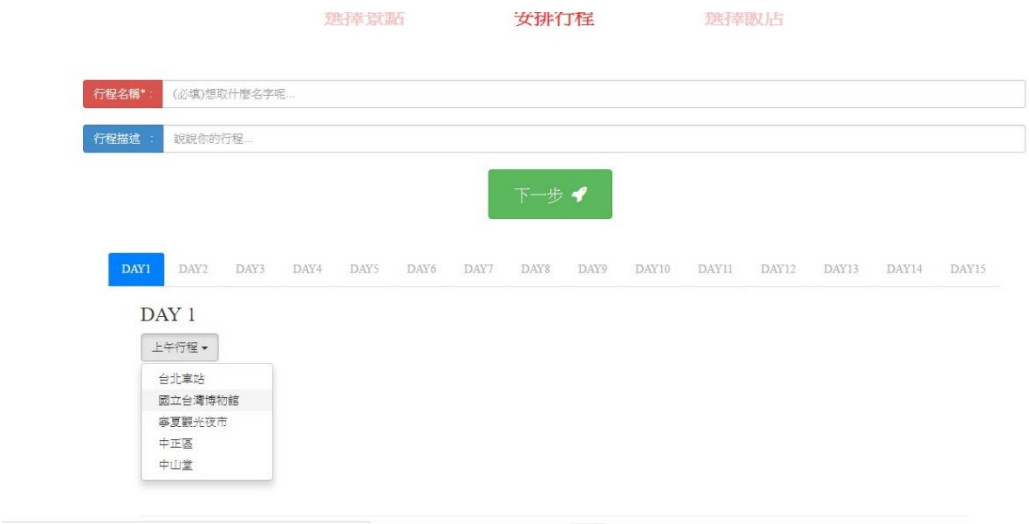

圖 6.10 安排早中晚細節

在安排行程的頁面中，首先使用者必須為這趟旅程取個響亮的名子，在未來的時候只要想到這個名子就會立刻想到這趟精彩的旅程。接著在下方的區塊安排行程，在這裡使用者將直接以每天的早中晚進行安排，系統會自動將使用者所選的景點條列出來，使用者僅需選擇先前所選取之景點並將之一一排進時間軸即可。

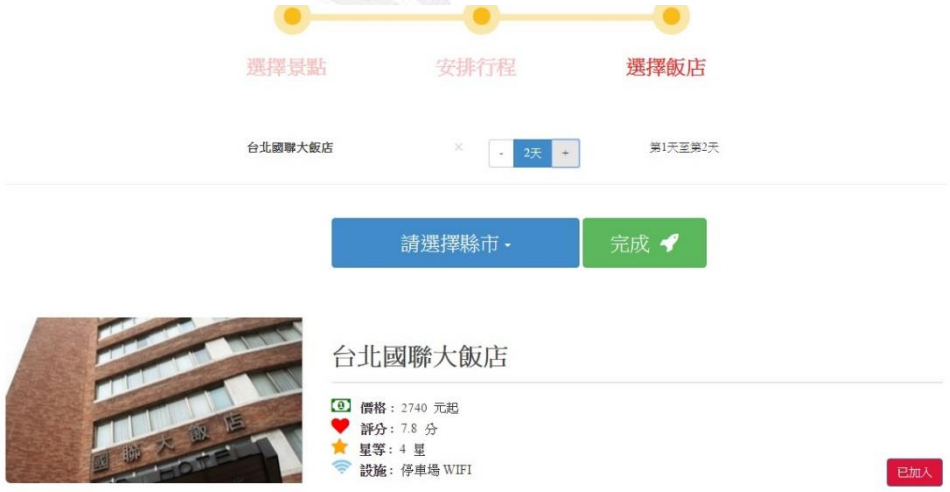

圖 6.11 選擇住宿飯店

將所有景點安排好之後，就來選取適合的飯店來為整天的旅程休息做充電，選擇居住的縣市後，即可將要入住的飯店都加入，並且選擇加入的每一間飯店要居住天數。

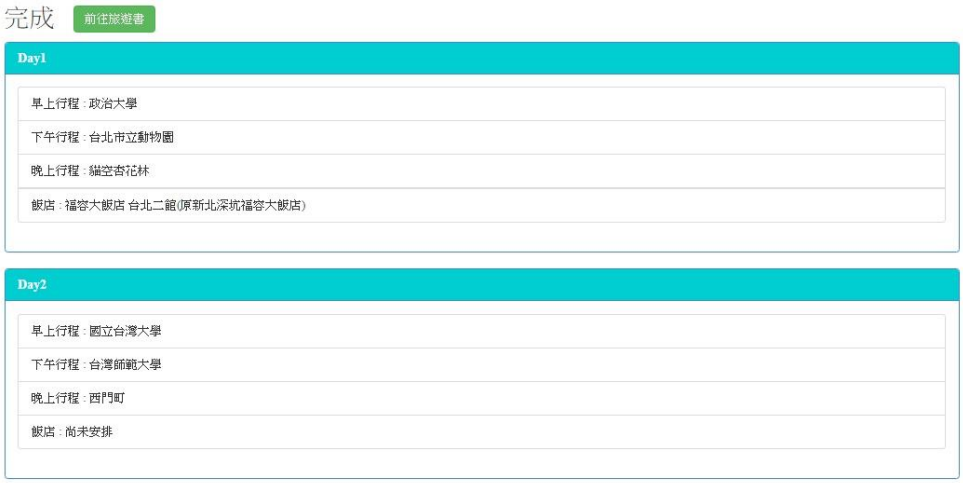

圖 6.12 簡潔文字顯示行程

安排完旅遊的行程以及飯店後，系統就會自動將規劃好行程並繪製成圖表讓使用者一目了然。若要生成旅遊書，僅需點擊前往旅遊書的按鈕，系統即會自動依照使用者規劃的行程來生成旅遊書。

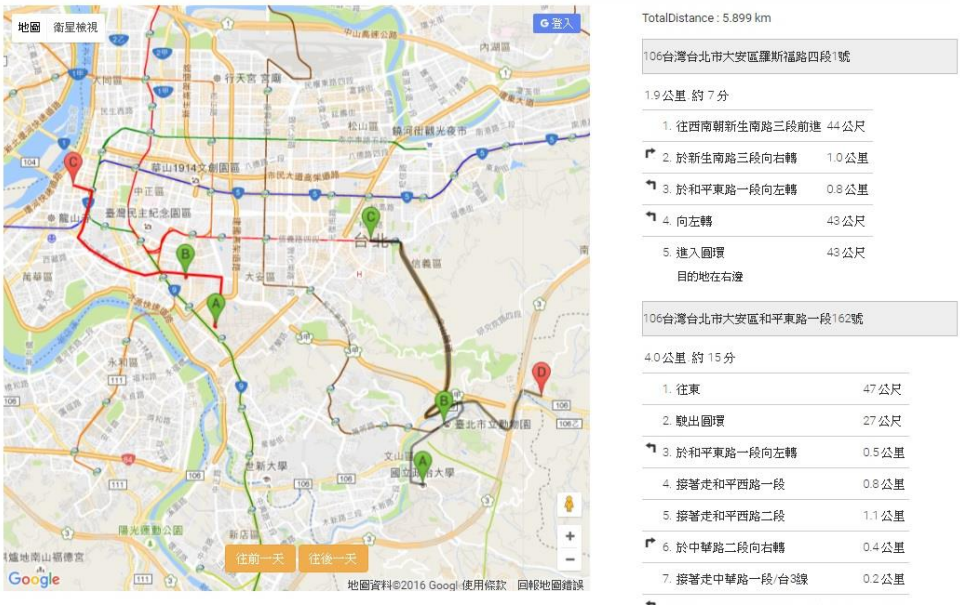

圖 6.13 地圖方式顯示每天的行程路徑

除了生成圖表之外我們還提供了每天的旅遊路徑給使用者查看，讓使用者可以預先了解路徑大概的規劃。

6.4 旅遊書

圖 6.14 旅遊書封面頁

進入旅遊書頁面後，系統會自動將使用者前面規劃好的行程放入旅遊書內，並做好旅遊書的基礎編排，讓使用者在製作時能夠更加輕鬆便利。

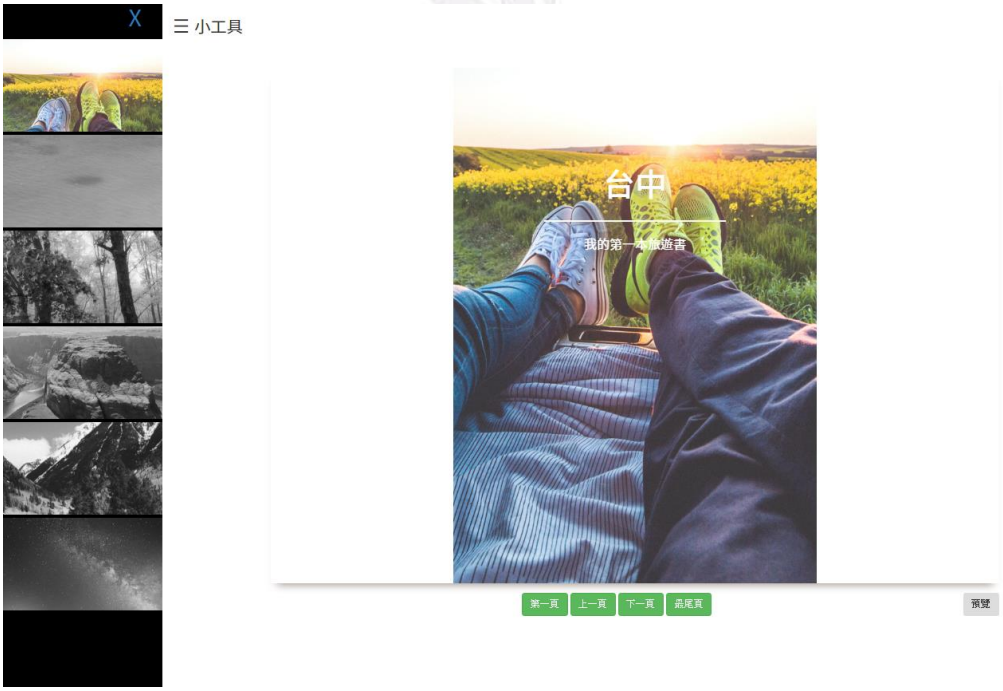

圖 6.15 旅遊書更改背景圖

我們提供很多不同的背景圖，讓使用者可以根據自己喜歡的風格來選擇背景圖片。

圖 6.16 旅遊書更改標題副標題

系統預設會將使用者前面輸入的行程名稱當作標題，但使用者也可在此頁面進行更改。

圖 6.17 旅遊書加入文字框

除了預設好的旅遊介紹景點資訊以及 Google Map 位置之外，使用者也可以加入自己想要的文字，提高客製化的程度。

圖 6.18 旅遊書加入貼圖

除了文字之外我們也提供使用者加入可愛的貼圖，這些貼圖可以讓旅遊書更加活潑生動，也讓每一本旅遊書都是特別且獨一無二的。

圖 6.19 旅遊書儲存

當使用者編輯完旅遊書之後，只要按下右下角的瀏覽按鈕，當頁的旅遊書就會呈現出來，然後再對圖片按下右鍵再按另存圖檔，即可儲存旅遊書，讓使使用者可以列印出來。

6.5 推薦系統

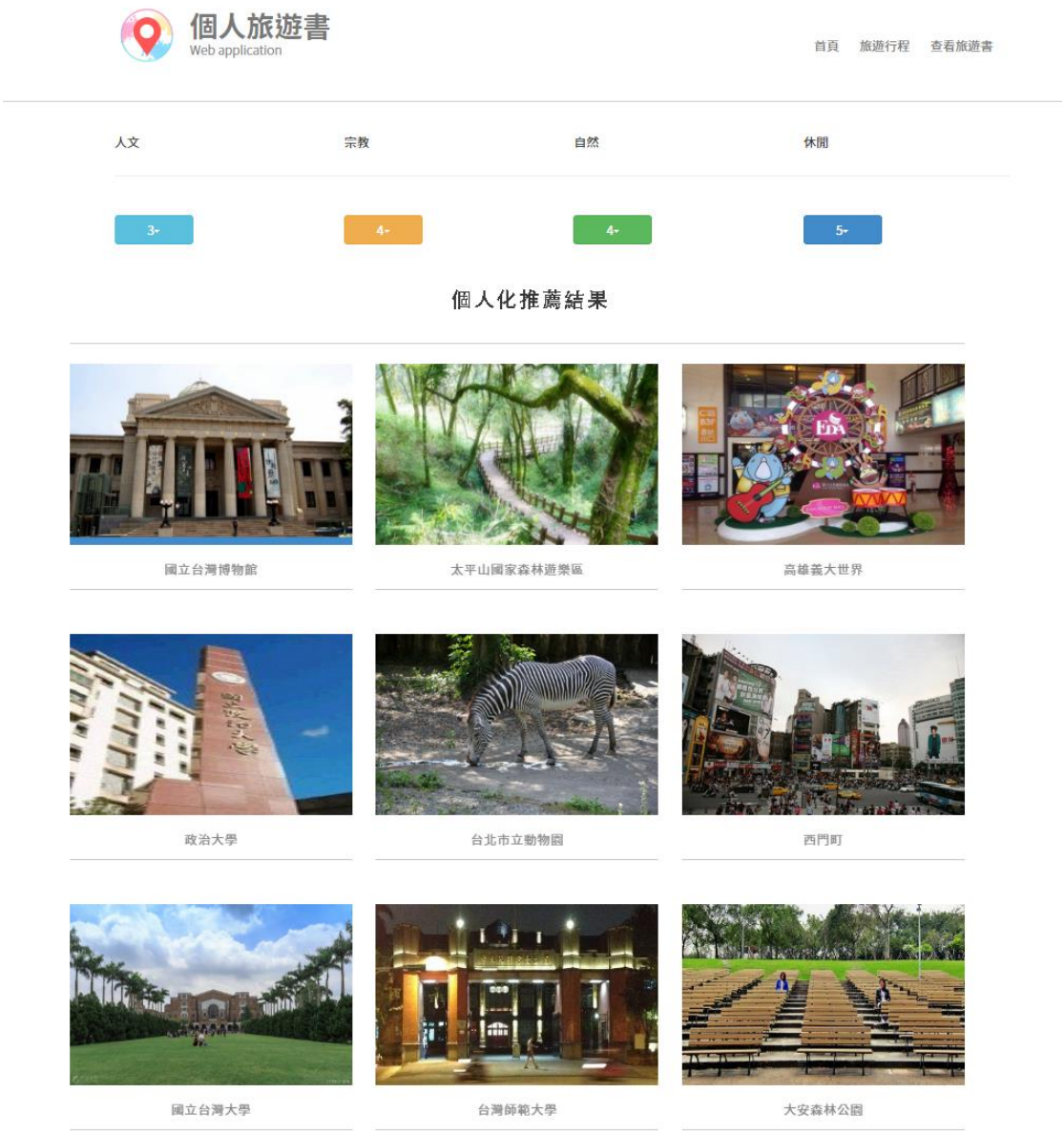

圖 6.20 個人化推薦結果

使用者輸入自己對每個類別(人文、宗教、自然、休閒)的興趣分數後，我們將會依照演算法的出的結果，將景點推薦給使用者觀看查詢。

第七章 結論與建議

7.1 結論

目前市面上的旅遊推薦系統常常推薦的是有名的景點，而非推薦真正符合使用者需求的地點，而我們做的旅遊推薦系統，是一個基於使用者個性資料所分析並推薦的系統，能夠人性化的推薦使用者符合的地點，並讓使用者簡單快速的安排自己的行程，而其中利用了 R-tree 以及天際線查詢的相關論文來減少執行時間查詢，R-tree 的功能在於將巨量的資料已以有規則的儲存方式來達到快速搜尋的功能，但是對於少量資料效果較不顯著，而天際線查詢則是以多個維度的興趣取向考慮來選出適合的景點，能有效率的挑出目標景點，而在資料量少的情況下也是效果不顯著。

但是在實際的情況下一個縣市的景點可能就高達數萬個資料點了，所以在儲存資料的方式以及分析必須採取最有效率的方式，所以我們使用了這 R-tree 以及天際線查詢這兩個方法來使用在我們的推薦系統上。

7.2 未來展望

在這個自助旅行的風氣逐漸上漲的世代，人們會越來越講求旅遊的品質以及時間的安排是否得當，不希望自己浪費額外的時間抑或是金錢，對此，我們討論了如何使我們的推薦系統更符合需求，就是將所有的景點資料所包含的資訊擴增，例如需要額外的花費、起迄時間，甚至是目前人數、年齡層等，希望能藉由額外的考量點來產生出一個獨一無二，讓使用者感到舒適的旅程。

7.3 專題心得

7.3.1 鄭金君

專題和以往的上課作業不同，並非用簡短的時間就能完成，而是需要長時間的研究與討論才有辦法達成，除此之外，很多的技術都是學校以前沒教過的，大部分的問題，都是需要透過網路上找資料，參考別人的過往經驗才得以解決。

這次的專題，我負責的部分主要是演算法的撰寫。要把一個討論出來的方法，透過程式寫出來，並非想像中的容易，其中需要結合很多的方法，以及考慮現有資料的類型，對資料做適當的分析，找出較佳的解。這些過程都需要反覆的嘗試，以及不斷的研究才能完成，從中真的能學到不少解決問題的方法。

做完這個專題，過程雖然很累，但卻能讓我們從中卻學到大量的課外知識。有了這次的專題經驗，也讓我們了解，很多技術都是要我們自己去學習的，學校只是一個教導我們理論基礎的地方，更專業的知識問題還是需要我們自己去探索學習。

7.3.2 張哲愷

這次的專題實作對我們來說是一個全新的展開，全新的工具，所有的一切都從頭摸索，無論是初步規劃、蒐集討論、初步實作以及後期的反覆測驗，都是一步一腳印，而需求是會因應時間、空間、以及各種因素有所改變，所以在規格方面我們修改及討論許多次，那也是經由一次次的修改及規劃明白了其中的重要性，研討二字是經由研究各種可能發生的成果以及討論所實作出來的，那當然一個好的專題是由各方面的人所匯集而成的，缺一不可，每個人都是很一個大型機器內的重要零件，缺少了即運轉不順，甚至是無法動作。

7.3.3 謝忠穎

專題從來不是一件簡單的事情，從一開始的規劃，到後來的製作，每個過程我都銘記在心中，這是我念資訊系以來最好玩的事情之一。專題的每個成員都很用心在製作自己的部分，所以我也督促自己要把份內工作做好，不要拖累大家的進度。

這次專題我負責的部分是網頁前端，要寫出一個能用的網頁前端也許沒什麼了不起，但是要做出一個實用的網站，需要付出的努力遠比我想像中的多更多，要先從使用者的需求分析開始，先思考使用者需要什麼樣的功能，我們要提供給他們什麼樣的服務，再到使用者在使用網站時，我們網站的流程以及視覺是否能夠給使用者提供直覺的管道，讓使用者在使用我們網站時能夠迅速上手。了解到成功的網站來自於背後大量的測試與分析，這是我這次專題最大的收穫。

經過這整年下來的努力，其中我學會了原本想都沒想過的想法，也自己上網學習很多知識與前端的技術，讓我發覺原來在學校所學到的知識，只是我們踏入外面職場的基石，要勾到頂端的開發者還有一大段的距離。希望自己可以汲取這次專題的經驗，勇於面對每一次的挑戰。

7.3.4 陳佩貝

這次專題從開始規劃到實作，都是很新奇有趣的經驗，以前都只是小小的程式，不會注意到太多的事情，但是專題卻需要結合很多之前學到的知識，而且有些還要自己再更深入研究才有比較了解。

這次的專題我負責的是 ASP.Net 的後端程式撰寫，以及資料庫管理的部分，這次遇到最大的麻煩就是，我在前期撰寫程式的時候，因為沒有將程式寫得很好，經常寫得很亂，造成之後要改一點點的程式碼就要改一大堆東西，以及與前端的溝通，因為程式寫得不好造成前端也要撰寫邏輯才呈顯資料，是一直到後期

有人跟我說我才了解，才開始改進，這經驗讓我了解一定要做好規劃以及溝通，之後也會比較好修改維護。

我很喜歡這次合作做專題的過程，大家一起討論規劃整個專案，討論的過程中也聽到很多與我不同的想法，一起學習合作的感覺真的很好，謝謝我們組員們。

參考文獻

[1] 中華民國 104 年國人旅遊狀況調查
URL:
http://admin.taiwan.net.tw/upload/statistic/20160810/5edad339-d16f-4933-982d-9c90c72f5739.pdf

[2] R-tree – Wikipedia
URL:https://en.wikipedia.org/wiki/R-tree

[3] M. Sharifzadeh and C. Shahabi, “The spatial skyline queries” in VLDB, 2006, pp. 751–762

[4] PHP - Wikipedia
URL:https://zh.wikipedia.org/wiki/PHP

[5] Microsoft SQL Server – Wikipedia
URL:https://zh.wikipedia.org/zh-hant/Microsoft_SQL_Server

[6] NuGet - Wikipedia
URL:https://en.wikipedia.org/wiki/NuGet

[7] 雄獅旅遊
URL: http://www.liontravel.com/

[8] 旅遊資訊王
URL:http://travel.network.com.tw/

[9] Heap (data structure) - Wikipedia
URL:https://en.wikipedia.org/wiki/Heap_(data_structure)

[10] M-tree – Wikipedia
URL:https://en.wikipedia.org/wiki/M-tree

[11] Database schema - Wikipedia
https://en.wikipedia.org/wiki/Database_schema

[12] Entity–relationship model
https://en.wikipedia.org/wiki/Entity-relationship_model

[13] ER 模型- 維基百科，自由的百科全書 – Wikipedia
https://zh.wikipedia.org/wiki/ER%E6%A8%A1%E5%9E%8B

[14] 實體關係模型(Entity-relationship model) | MySQL Taiwan
http://www.mysql.tw/2013/03/entity-relationship-model.html

[15] Use case diagram - Wikipedia
https://en.wikipedia.org/wiki/Use_case_diagram

[16] Flowchart - Wikipedia
https://en.wikipedia.org/wiki/Flowchart

[17] Microsoft Visual Studio - 維基百科，自由的百科全書 - Wikipedia
https://zh.wikipedia.org/wiki/Microsoft_Visual_Studio