Youbike 1.5

Team45

邱鈺傑 b06303085

創作理念

對於學生來說，Youbike　是日常通勤、以及課間趕堂的好工具，在日常通勤的情境中，一下了公車、捷運時，如果直接走去附近站點，會有站點沒車的可能性，騎到目的地也可能沒有車位可停，通勤時間會多出許多變數而導致難以控管。而在課間趕堂的情境中，Youbike2.0　在校園內分布密集，是趕堂的好幫手，但也同樣會有出發站點沒車、目的地沒車位的可能性。

針對以上使用情境，現在其實已有開發相關功能的 app，以下我們比較 app 各自的缺陷，並且在我們推出的 Youbike1.5 提供解決方案。

1. Youbike1.0 & Youbike2.0 官方 app

兩個 app 其實功能上差不多，官方 app 會提供一個地圖上面有全部的 1.0 站點(2.0　站點)，使用者手動點選上面的站點，查看剩餘車輛、車位，路徑規劃的部份，僅會從使用者現在位置在地圖上以粉紅色的線導到站點。

我們在使用上發現了三個缺點:

(1) 沒有幫忙規劃完整路線:

使用者必須得先自行判斷要去哪個站點借還車，再來路線規劃很陽春，只能把使用者從所在位置導引到一個站點，在我個人的體驗上，如果從公館要騎 1.0 到科技大樓，出捷運站到 1.0 站要導一次路線，從 1.0 站到科技大樓附近的 1.0 站又要導一次，科技大樓附近的 1.0 站到捷運站又要導一次，使用這個 app 要重導三次路線，非常麻煩。

(2) 沒有計算通勤時間:

Youbike1.0 app 完全不會提供時間資訊，Youbike2.0 app 則是結合了 google map 的導航功能，會顯示騎車時間，但因為前述第一個缺點，路線非常零散，使用者必須跑很多段並把時間加總，才能得到通勤時間，使用上相當不便。

(3) 1.0 和 2.0 的 app 是完全獨立的:

1.0app 上面只會有 1.0 的資訊，2.0 的完全不會有，2.0 app 也同樣部會有任何 1.0 的資訊，假設有學生是要從古亭(附近僅有 1.0 站點)騎到總圖書館(附近僅有 2.0 站點)，那麼他必須打開兩個 app，慢慢規劃、比對路線。

2. Google map

Google map 也有結合共享單車，但是 google map 上目前僅有極少的 2.0 站點(約七到八站)，1.0　站點雖然相對足夠，然而站點剩餘車輛、剩餘車位的更新延遲過長，和 Youbike 官網公布的數字往往相去甚遠。

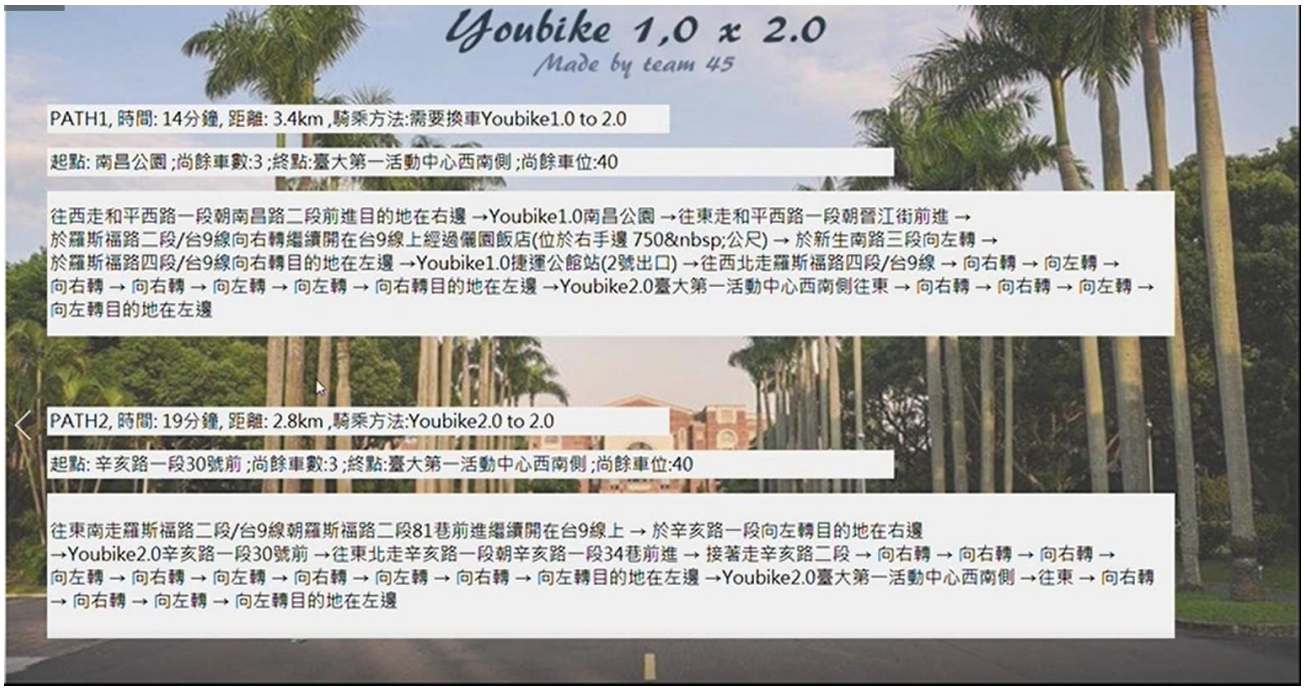
為了解決學生的痛點，讓通勤時間變得精確可控，以及截長補短，補足上面 app 的缺陷，我們希望可以做出一個程式幫忙做出結合 Youbike1.0 及 2.0 的通勤規劃，只要使用者輸入出發地點以及目的地，程式會幫忙選定出發站點有車可借、目的地站點有車位可停，並且通勤時間最短的完整路徑，其中也會考量轉乘

的可能性，最後在介面輸出兩條路徑，顯示所需時間、剩餘車輛、剩餘車位、路徑的文字導航。

成果說明

進到程式之後，第一個介面會顯示開始旅途以及離開按鍵，按下開始旅途以執行後續程式。

接著介面會有兩列顯示，分別要求使用者輸入所在地以及目的地，由於程式會連結到 google map 爬取路徑資訊，這邊的輸入可以是**確切的地址**或是 **google map　能正確搜尋到地標的關鍵字**，如果僅輸入總圖而非台大總圖，則可能會被判定成其他地標。為了下一步路徑的展示，我們在此輸入所在地「捷運古亭站」以及目的地「台大總圖」，並且按下 go 等待結果。



在短暫的等待過後，程式會輸出最佳的兩條路徑，在介面顯示此路徑的所需時間、距離、騎乘方法[1]，起點 Youbike 站點名稱以及剩餘車輛、終點 Youbike 站點名稱以及剩餘車位，再來是路徑指示，會先導航使用者從古亭站走到 Youbike 南昌公園站，再騎到捷運公館站還 1.0、借 2.0(中繼站在演算法判定當下也確定會有 2.0 車可借、有 1.0 車位可還)，最後騎到臺大第一活動中心南側還車，總圖就在眼前了。

系統設計

一、 整體架構

我們的程式會先爬 Youbike 官網上所有站點的經緯度資料，並在 google map 建立這些站點，取得使用者所在地、目的地之後，程式會連結到自建的

[1] 騎乘方法共分:騎 1.0、騎 2.0，或是需要轉乘，先騎 1.0 再轉 2.0、先騎 2.0 再轉 1.0。

google map，找到距離最近的數個站點，再連結到 Youbike 官網取得站點即時資訊，淘汰掉(起始點)沒車可借以及(終點)沒車位的站點，之後進入演算法透過 google map 計算零碎路徑的騎車或是走路時間，加總後進行排序，最後輸出所需時間最短的兩條路徑的所有資訊，介面也配合程式的輸入、輸出做版面的配置及美化。

二、 爬蟲

1. Youbike1.0 與 Youbike2.0 官網爬蟲:

透過 requests 和 BeautifulSoup 兩個套件爬取台北市各站點的「區域名稱」、「租賃站名」、「剩餘車數」、「剩餘車位」以及「經緯度」，以 list 的形式輸出各站點的以上資訊。

2. Googlemap API 爬蟲:

以 class Youbike_Crawler:包裝針對 Google map API 爬蟲部分的程式碼，其中包含__init_、route_plan、top_choice 三個函數。_init_用於在每次呼叫 class 時初始化 class，得到新的目標位置資訊（起點或終點的位置）；route_plan 用於獲得目標位置（起點或終點的位置）與附近某 youbike1.0 或 youbike2.0 站點的距離；top_choice 則是比較 youbike 站點與目標位置（起點或終點）的距離，並將 youbike1.0 和 youbike2.0 分組，各自依距離由近到遠排序，最後分別輸出各自的排序結果，以便接下來執行演算法的同學使用。

三、 演算法

最重要的基礎是 route_plan2 函數和 borrowable 函數，route_plan2 函數把出發地點和目標地點丟進去，這個函數就會連結到 google map，輸出兩點的騎車時間或走路時間，以及距離和導航的路徑。
再來是 borrowable 函數，把任意一個 Youbike 站點丟進去，就能知道有沒有剩餘車輛或者是車位。

有了這兩個函數，我們衍伸出了三個演算法來算出最佳路徑:
youbike1_to_1、optimal_midway_ubike1_start 以及 optimal_midway_ubike2_start， youbike1_to_1 是假設使用者會騎 1.0 並且在終點還 1.0，用 route_plan2 函數來計算並且按照所需時間排序每條路徑，使用者騎 2.0、還 2.0 也是使用此函式做類似處理，而 optimal_midway_ubike1_start 以及 optimal_midway_ubike2_start 是假設使用者會騎 1.0 出發，到了中繼站轉 2.0，最後在終點還 2.0 的車，在這個演算法裡我們會考量所有台大周圍的 1.0 站做為中繼站，然後借由 midway_time_cost 函式來算出跑每個中繼站的所需時間，最後輸出一條時間最短的中繼站路徑。

經過這三條演算法，會篩選出許多條不同的路徑，最後我們再一次以時間排序，輸出耗時最少的兩條路徑。

四、 介面

在介面的使用套件部分，總共會使用三個套件:

1. "import tkinter as tk, from tkinter import *, from tkinter import ttk ":
   建立介面的基本架構所需的函式庫。
2. "import tkinter.font as tkFont":
   用來更改介面的字體。
3. "from PIL import ImageTk, Image":
   輸入圖片時需要的套件。

之後我們利用"class MainPage(tk.Frame):"寫「歡迎光臨」及「輸入路徑」兩個介面，其中 def_init_ 函式創造出一個介面的框架，再用 def createWidgets 加上按鈕等細節， button 中的 command 函數能夠連結到欲前往的畫面或程式，使用者按下去才能成功執行程式。start 這個框界主要是以.entry 讓使用者輸入起點與終點，再利用.get 取得字串以輸入給主程式使用；「輸入所在地」及「輸入目的地」則是以 label.() 呈現。而細節上.pack()是為了讓目標物呈現，.place()是設定目標物的大小及位置。

最後的"class MainPage1(tk.Frame):"則是將演算法算出的結果利用 label 呈現在介面上。