

資料庫期末報告

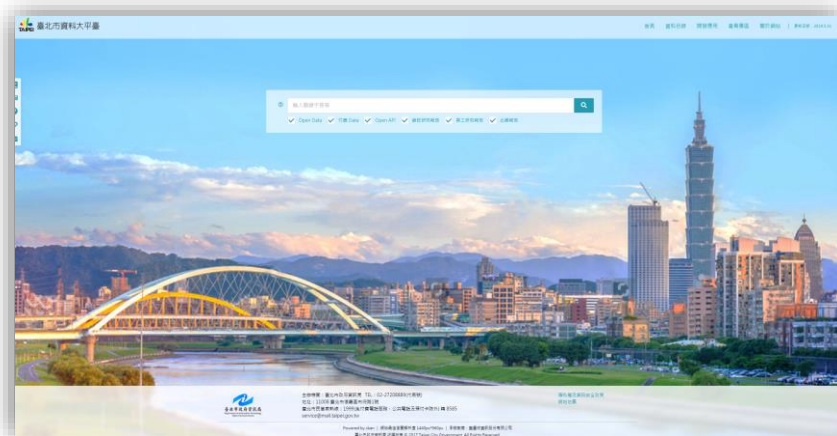
第二組

題目

台北市竊盜案件大數據平台

資料來源

資料取自台北市資料大平台



選擇原因:

因為台北市提供的資料相較於其他縣市的資料完整許多 (其他縣市只有台中市以及新竹縣有統計資料，新竹縣資料不完整，台中市上一次更新已經是去年一月) 因此我們選擇使用台北市資料大平台的資料來做為我們這次project的主要data。

再加上台北市有監視器點位相關資訊，我們想到可以用地圖，結合警察局位置以及監視器點位，提供使用者一個可以立即查詢的互動介面。

資料選擇

臺北市自行車竊盜點位資訊

臺北市自行車竊盜點位資訊

☆☆☆☆
平均0 (0人/次投票)

分享：f t

加入收藏

資料項目

臺北市104年1月-110年5月份自行車竊盜點位資訊

智慧繪圖 下載 (1563次) API 預覽

詮釋資料

主題分類	治安
政府網站分類	生活安全及品質
資料集描述	臺北市104年1月-110年5月份自行車竊盜點位資訊
關鍵詞	竊盜,自行車竊盜
主要欄位說明	編號、案類、發生日期、發生時段、發生地點
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-06-07 16:04:28
資料量	
發布時間	2021-06-07
更新頻率	每月

顯示較多

臺北市機車竊盜點位資訊

臺北市機車竊盜點位資訊

☆☆☆☆
平均0 (0人/次投票)

分享：f t

加入收藏

資料項目

臺北市107年1月-110年5月份機車竊盜點位資訊

智慧繪圖 下載 (899次) API 預覽

詮釋資料

主題分類	治安
政府網站分類	生活安全及品質
資料集描述	臺北市107年1月-110年5月份機車竊盜點位資訊
關鍵詞	機車竊盜,竊盜
主要欄位說明	編號、案類、發生日期、發生時段、發生地點
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-06-07 16:04:15
資料量	
發布時間	2021-06-07
更新頻率	每月

顯示較多

資料選擇

臺北市汽車竊盜點位資訊

臺北市汽車竊盜點位資訊

平均5.0 (1人/次投票)

分享：f t

加入收藏

資料項目

臺北市104年1月-110年5月份汽車竊盜點位資訊

智慧地圖 下載 (1391次) API 預覽

詮釋資料

主題分類	治安
政府網站分類	生活安全及品質
資料集描述	臺北市104年1月-110年5月份汽車竊盜點位資訊
關鍵詞	汽車竊盜,竊盜
主要欄位說明	編號、案類、發生日期、發生時段、發生地點
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-06-07 16:04:39
資料量	
發布時間	2021-06-07
更新頻率	每月

顯示較多

臺北市住宅竊盜點位資訊

臺北市住宅竊盜點位資訊

平均4.0 (7人/次投票)

分享：f t

加入收藏

資料項目

臺北市104年1月-110年5月份住宅竊盜點位資訊

智慧地圖 下載 (2440次) API 預覽

詮釋資料

主題分類	治安
政府網站分類	生活安全及品質
資料集描述	臺北市104年1月-110年5月份住宅竊盜點位資訊
關鍵詞	住宅竊盜,竊盜
主要欄位說明	編號、案類、發生日期、發生時段、發生地點
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-06-07 16:03:01
資料量	
發布時間	2021-06-07
更新頻率	每月

顯示較多

資料選擇

臺北市政府警察局名稱及地址

臺北市府警察局名稱及地址

OD

☆☆☆☆☆
平均0 (0人/次投票)

分享：[f](#)[t](#)

加入收藏

資料項目

派出所地址

智慧地圖 下載 (470次) API 預覽

詮釋資料

主題分類	交通
政府網站分類	公共資訊
資料集描述	警察局各分局、派出所地址
關鍵詞	分局,地址,派出所
主要欄位說明	name(標題)、content(內容)、traffic_information(交通資訊)、display_addr(顯示用地址)、poi_addr(系統辨識用地址)
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-04-27 08:29:55
資料量	114
發布時間	2017-02-17
更新頻率	不定期

顯示較多

臺北市政府警察局錄影監視系統設置區位

臺北市府警察局錄影監視系統設置區位

OD


☆☆☆☆☆
平均3.0 (2人/次投票)

分享：

 加入收藏

資料項目

臺北市府警察局錄影監視系統設置區位

 下載 (701次)  預覽

詮釋資料

主題分類	治安
政府網站分類	生活安全及品質
資料集描述	警察局錄影監視系統設置區位
關鍵詞	監視器,警察局,錄影監視系統
主要欄位說明	編號、所屬單位、安裝地址、攝影方向
國發會政府資料開放平台資料集類型	原始資料
最後更新時間	2021-04-07 19:04:51
資料量	15639
發布時間	2019-11-07
更新頻率	每半年







顯示較多

Data Tables Schema


以下列出所有資料的個別Schema。

因為這次採用Ruby on Rails 的原因，使用的Model創建工具是 ActiveRecord。他在新增每一筆新增資料都會加上一個 id 作為Primary Key 還會在最後加上創建時間與更新時間，方便資料的更新以及查找。但缺點是隨著資料量變大，那些增加的 column 可能會佔據一定的資料量。

Table: scooter_stoles







 Expand	 Records	 Schema	 SQL Query	 Export	 Functions
--	---	--	---	--	---

Schema

PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	case	varchar	true				string	
	date	varchar	true				string	
	time	varchar	true				string	
	location	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Data Tables Schema

Table: bike_stoles

 Expand	 Records	 Schema	 SQL Query	 Export	 Functions
--	---	--	---	--	---

Schema




PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	case	varchar	true				string	
	date	varchar	true				string	
	time	varchar	true				string	
	location	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Table: cctvs

 Expand	 Records	 Schema	 SQL Query	 Export	 Functions
--	---	--	---	--	---

Schema

PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	ps	varchar	true				string	
	addr	varchar	true				string	
	direction	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Data Tables Schema

Table: car_stoles

[Expand](#)[Records](#)[Schema](#)[SQL Query](#)[Export](#)[Functions](#)

Schema



PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	case	varchar	true				string	
	date	varchar	true				string	
	time	varchar	true				string	
	location	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Table: home_stoles







[Expand](#)[Records](#)[Schema](#)[SQL Query](#)[Export](#)[Functions](#)

Schema

PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	case	varchar	true				string	
	date	varchar	true				string	
	time	varchar	true				string	
	location	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Data Tables Schema

Table: cctv_coors

 Expand	 Records	 Schema	 SQL Query	 Export	 Functions
--	---	--	---	--	---

Schema









PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	lat	float	true				float	
	lng	float	true				float	
	dir	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Table: histories

 Expand	 Records	 Schema	 SQL Query	 Export	 Functions
--	---	--	---	--	---

Schema

PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	q	varchar	true				string	
	lat	float	true				float	
	lng	float	true				float	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

Data Tables Schema

Table: ps_coors

[Expand](#)[Records](#)[Schema](#)[SQL Query](#)[Export](#)[Functions](#)

Schema



PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	no	varchar	true				string	
	lat	float	true				float	
	lng	float	true				float	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

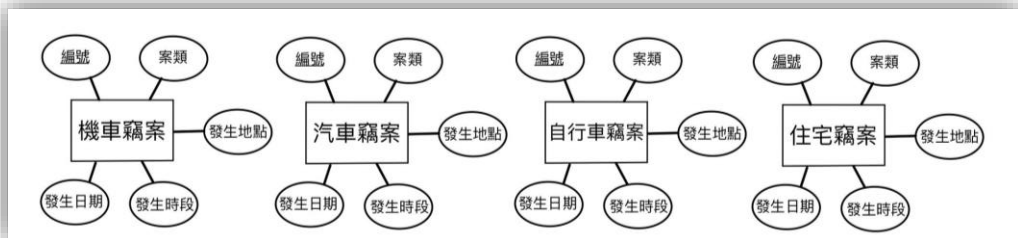
Table: ps_addrs

[Expand](#)[Records](#)[Schema](#)[SQL Query](#)[Export](#)[Functions](#)

Schema

PK	NAME	SQL TYPE	NULL	LIMIT	PRECISION	SCALE	TYPE	DEFAULT
	id	integer	false				integer	
	name	varchar	true				string	
	content	varchar	true				string	
	traffic_information	varchar	true				string	
	display_addr	varchar	true				string	
	poi_addr	varchar	true				string	
	created_at	datetime(6)	false		6		datetime	
	updated_at	datetime(6)	false		6		datetime	

ER Diagram

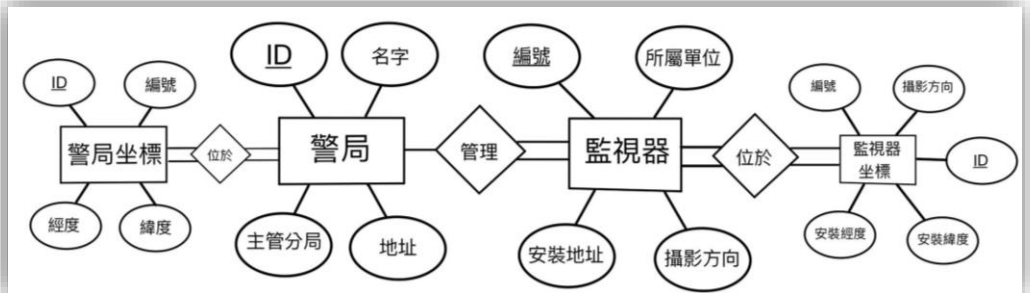


上面的4個 table 分別是機車竊案、汽車竊案、自行車竊案和住宅竊案，每個 table 都包含了編號、案類、發生地點、發生日期和發生時段共5個 attributes。



這個搜尋紀錄的 table 是將使用者的搜尋紀錄存取下來之後可以做使用，這個 table 包含了搜尋時間、搜尋地址、地址經度和地址緯度4個 attributes。

ER Diagram



此 ER Model 包含了警局、警局坐標、監視器和監視器坐標4個 table。其中用一對多的 relationship 連結警局和監視器兩個 table，因為一個監視器只隸屬於一個警局，但一個警局可以管理多個監視器。用一對一的 relationship 連結警局和警局坐標兩個 table，因為每個警局與警局位址都是相對應而且固定的。用一對一的 relationship 連結監視器和監視器坐標兩個 table，因為每個監視器與監視器位址都是相對應而且固定的。

Normalization

我們試著把我們的資料正規化，經過觀察之後，所有我們運用到的 table 中，可被 normalize 的有下面兩個 table，分別是 ps_addrs 跟 cctvs。

cctvs

id	no	ps	addr	direction	created_at	updated_at
1	LAAA001-01	大同分局 寧夏路派出所	民生西路231號	西	2021-05-19 21:10:53.529477	2021-05-19 21:10:53.529477
2	LAAA002-01	大同分局 寧夏路派出所	民生西路257號	西	2021-05-19 21:10:53.539937	2021-05-19 21:10:53.539937
113	LAAB001-01	大同分局 民族路派出所	迪化街2段246號	南	2021-05-19 21:10:54.555502	2021-05-19 21:10:54.555502
664	LABA001-01	萬華分局 漢中街派出所	西園路1段34巷口	西	2021-05-19 21:10:59.718462	2021-05-19 21:10:59.718462



normalize

id	no	addr	direction	created_at	updated_at
1	LAAA001-01	民生西路231號	西	2021-05-19 21:10:53.529477	2021-05-19 21:10:53.529477
2	LAAA002-01	民生西路257號	西	2021-05-19 21:10:53.539937	2021-05-19 21:10:53.539937
113	LAAB001-01	迪化街2段246號	南	2021-05-19 21:10:54.555502	2021-05-19 21:10:54.555502
664	LABA001-01	西園路1段34巷口	西	2021-05-19 21:10:59.718462	2021-05-19 21:10:59.718462



no	ps
LAAA001-01	大同分局 寧夏路派出所
LAAA002-01	大同分局 寧夏路派出所
LAAB001-01	大同分局 民族路派出所
LABA001-01	萬華分局 漢中街派出所

Normalization

ps_addrs

id	name	content	traffic_information	display_addr	poi_addr	created_at	updated_at
562	臺北市政府警察局總局	臺北市政府警察局總局	null	臺北市中正區延平南路96號	臺北市中正區延平南路96號	2021-05-19 15:42:11.287075	2021-05-19 15:42:11.287075
563	大同分局	大同分局	null	臺北市大同區錦西街200號	臺北市大同區錦西街200號	2021-05-19 15:42:11.295380	2021-05-19 15:42:11.295380
585	民族路派出所	隸屬大同分局	null	臺北市大同區重慶北路三段168號	臺北市大同區重慶北路三段168號	2021-05-19 15:42:11.533980	2021-05-19 15:42:11.533980



normalize

id	name	content	display_addr	created_at	updated_at
562	臺北市政府警察局總局	臺北市政府警察局總局	臺北市中正區延平南路96號	2021-05-19 15:42:11.287075	2021-05-19 15:42:11.287075
563	大同分局	大同分局	臺北市大同區錦西街200號	2021-05-19 15:42:11.295380	2021-05-19 15:42:11.295380
585	民族路派出所	隸屬大同分局	臺北市大同區重慶北路三段168號	2021-05-19 15:42:11.533980	2021-05-19 15:42:11.533980



display_addr	poi_addr
臺北市中正區延平南路96號	臺北市中正區延平南路96號
臺北市大同區錦西街200號	臺北市大同區錦西街200號
臺北市大同區重慶北路三段168號	臺北市大同區重慶北路三段168號

Database 選擇

因為 web framework 採用 Rails 的關係，這次project我們選擇的資料庫是 Rails 的預設資料庫組態: SQLite3



SQLite 遵守 ACID，實現了大多數 SQL 標準。它使用動態的、弱類型的 SQL 語法。它作為嵌入式資料庫，是應用程式，如網頁瀏覽器，在本地/客戶端儲存資料的常見選擇。它可能是最廣泛部署的資料庫引擎，因為它正在被一些流行的瀏覽器、作業系統、嵌入式系統所使用。同時，它有許多程式設計語言的語言繫結。

Framework 選擇

這次沒有選擇傳統的 php 來建構網頁，而是選擇了近幾年才逐漸成熟的 Ruby on Rails



Ruby on Rails 是一個使用 Ruby 語言寫的開源 Web 應用框架，它是嚴格按照 MVC 結構開發，努力使自身保持簡單，使實際應用開發時的代碼更少，使用最少的組態。

Rails 的設計原則包括：

- 「[不做重複的事](#)」（Don't Repeat Yourself）
- 「[慣例優於設定](#)」（Convention Over Configuration）

簡單來說，採用此框架能做到快速部屬的目標。裡面有許多慣例可以採用，而不用辛辛苦苦的手刻所有底層的程式碼。

MVC 架構模式

也就是模型(Model)-視圖(View)-控制器(Controller) 架構由以下各部分組成：

Model 模型:

模型包含著應用的狀態，狀態可能是臨時的也可能是長久性儲存在資料庫中的。模型包含資料與資料代表的邏輯。在 Rails 中，模型通常是由一些代表關聯式資料庫中 RDBMS 表的類組成的。在 Rails 中，模型類是通過 **Active Record** 模式進行處理的。一般來說，程式設計師要做的是繼承 **Active Record** 類，同時程式會自動計算出要使用哪個 RDBMS 表，這個表有哪些列。表與表之間的關係通過簡單的命令來指明。

我們採用 => **Active Record**

View 視圖:

View 負責根據 Model 中的資料顯示使用者介面。作為 web 應用，Rails 裡的 View 通常是生成整個或者部分網頁。

我們採用 => **Javascript**、**內嵌 Ruby 的 HTML**

Controller 控制器:

控制器將使用者介面和資料模型關聯起來，並充當協調運作的角色。它接收各種使用者操作，更新資料模型，並用合適的view展示結果給使用者。像他的名字一樣，可以說應用的主要控制中心就是各個控制器。

CRUD

在處理資料的部分我們只有在查詢紀錄那個部分做到完整的CRUD，其他的部分都是撈取資料來做應用。

Create 增加:

1. 將監視器與警局的地址裡用 google map api 匯入新的table
2. 新增查詢紀錄

Delete 刪除:

1. 刪除查詢紀錄

Read 讀取:

1. 根據查詢的地址附近找出方圓一公里以內的監視器以及警局，並列出相關資訊
2. 針對年份月份竊盜類型以及行政區做查詢
3. 統計每個行政區的資料並製作成圖表

Update 更新：

1. 更新查詢紀錄

APPLICATION

應用方面，我們主要做了三大功能。
以下會詳盡的依序作介紹，並補充說明關於資料庫的處理。



地圖查詢



清單查詢



圖表查詢

地圖查詢

前置作業:

因為 google map api 免費查詢額度只有四萬筆的原因，因此我們先將將近兩萬筆的監視器以及警局的地址透過預先查詢的方式，存入一個新的 table，以減少重複查詢的次數。

```

2   Geocoder.configure(lookup: :google, api_key: "AIzaSyBc8zmz3YQWK0t6hCBRCVamuepPjLfcVPk")
3   Cctv.all.each do |i|
4     results = Geocoder.search(i.addr).first
5     next if results.nil?
6     results = results.coordinates
7     d = ""
8     d = d + "s" if i.direction.include?("南")
9     d = d + "n" if i.direction.include?("北")
10    d = d + "w" if i.direction.include?("西")
11    d = d + "e" if i.direction.include?("東")
12
13    CctvCoord.create(
14      no: i.no.to_s,
15      lat: results[0].to_f,
16      lng: results[1].to_f,
17      dir: d
18    )
19  end

```

這邊是監視器的部分，我們還有針對監視器面對的方位將中文改寫成英文，一並存入新的 table 之中，方便圖像轉換。

```

22  Geocoder.configure(lookup: :google, api_key: "AIzaSyBc8zmz3YQWK0t6hCBRCVamuepPjLfcVPk")
23  PsAddr.all.each do |i|
24    results = Geocoder.search(i.display_addr).first
25    next if results.nil?
26    results = results.coordinates
27
28    PsCoord.create(
29      no: i.id,
30      lat: results[0].to_f,
31      lng: results[1].to_f,
32    )
33  end
34

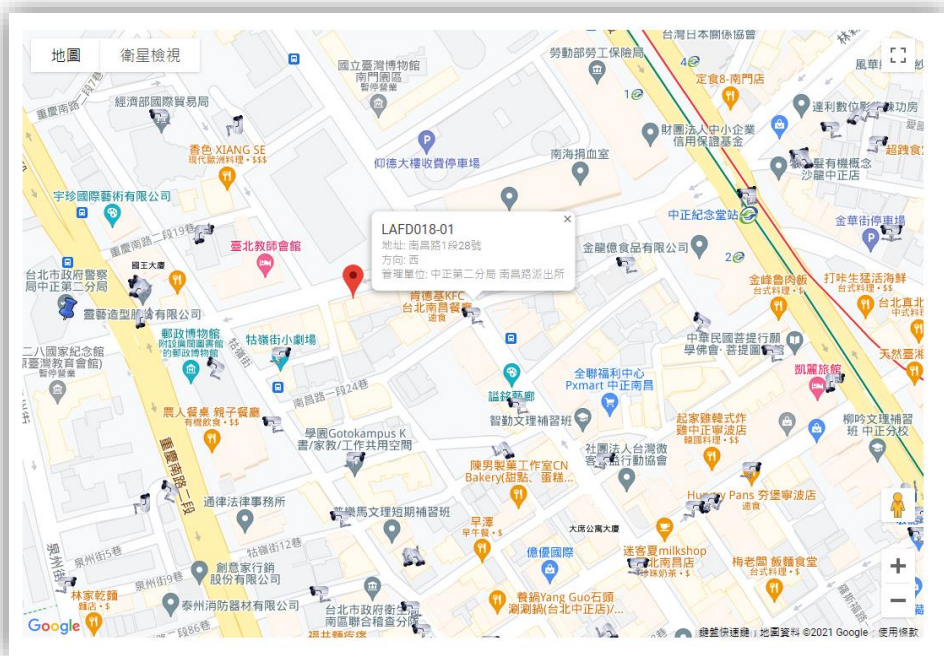
```

這邊是警局的部分，用警局的 ID 做為 Primary Key 讓兩個 table 可以互相應用。

地圖查詢

監視器顯示：

在地圖上除了標上監視器地點以外，點擊圖標還可以看到該監視器的相關資訊。



另外，針對監視器方位的不同，我們也有對圖像進行轉向。



```

155 CctvCoord.all.each do |cc|
156   next if Geocoder::Calculations.distance_between(results, [cc.lat, cc.lng], :units => :km) > 0.5
157   info = Cctv.where(no: cc.no).first
158   @hash << {
159     :lat => cc.lat,
160     :lng => cc.lng,
161     :infowindow => "<h3>#{info.no}</h3>地址: #{info.addr}<br>方向: #{info.direction}<br>管理單位: #{info.ps}",
162     :picture => {
163       :url => "/images/cctv_#{cc.dir}.png",
164       :width => 40,
165       :height => 40
166     }
167   }
168 end

```

在資料庫裡查詢方圓500公尺內的監視器，並把圖像以及資訊附上。

地圖查詢

查詢歷史紀錄：

我們將已查詢過的地址和座標一起存進 table 裡，除了可以方便使用者快速獲取歷史紀錄以外，還可以節省 api 的用量。

```
121     exi = History.where(q: params[:q])
122     if exi != []
123       results = [exi.take.lat, exi.take.lng]
124     else
125       results = Geocoder.search(params[:q]) if not params[:q].nil?
126       results = results.first.coordinates
127       if (results[0]<n[0] && results[0]>s[0] && results[1]>w[1] && results[1]<e[1] && params[:q] != "請輸入台北市以內的地址")
128         h = History.new
129         h.q = params[:q]
130         h.lat = results[0]
131         h.lng = results[1]
132         h.save
133       else
134         results = []
135       end
136     end
```

因為我們的資料的範圍只在台北市以內，因此不會出現台北市以外的查詢結果。一旦使用者輸入台北市以外的地址或是輸入非地址的字串，我們將不會顯示，且提示使用者只能輸入台北市以內的地址。在使用者輸入地址時，我們會先在資料庫中搜索有無一模一樣的關鍵字，如果有，就直接取用那筆資料的座標，無須再利用 api 查詢。

搜尋紀錄	Lat	Lng	時間	
台北市	25.0329694	121.5654177	2021-06-07 07:01	✖
台北市大安區和平東路二段108號	25.0248648	121.5422482	2021-06-07 07:08	✖
台北市南海路38號	25.03216	121.515801	2021-06-07 12:26	✖

也可以對查詢的歷史紀錄做刪除的動作。

清單查詢

搜尋欄位：

為防止使用者輸入了不如預期的關鍵字，我們採用下拉式選單的方式供使用者查詢。

搜尋您附近的失竊案件

年分	月份	類型	行政區
<input type="text" value="2021"/>	<input type="text" value="整年"/>	<input type="text" value="自行車竊盜"/>	<input type="text" value="全台北市"/>
<input type="button" value="搜尋"/>			

```
72 def search
73   if params[:month]!="整年"
74     params[:month] = "_" +format("%02d",params[:month]).to_s
75     date_search = (params[:year].to_i-1911).to_s+params[:month][1..2]
76   else
77     date_search = (params[:year].to_i-1911).to_s
78   end
79   @data = case params[:type]
80     when "自行車竊盜" then BikeStole.all
81     when "機車竊盜" then ScooterStole.all
82     when "汽車竊盜" then CarStole.all
83     when "住宅竊盜" then HomeStole.all
84     #else BikeStole.all+ScooterStole.all+CarStole.all+HomeStole.all
85   end
86   if params[:district] != ""
87     @data = @data.where("location LIKE '%#{params[:district]}%'")
88   end
89   @data = @data
90     .where("date LIKE '#{date_search}%'")
91     .order(:date)
92   @count = @data.count()
93   @data = @data.page(params[:page])
94 end
```

利用 GET 的方式分析使用者查詢了什麼，逐步找出目標資訊。

1. 先將西元年轉為民國年，如有選取月份則用串接在後面
2. 依照類型選擇要查詢的 table
3. 在地址中找尋有該行政區出現的資料

清單查詢

搜尋結果分頁：

由於搜尋結果常常都是上百條起跳，我們在清單中加入了分頁功能，方便閱讀。

以下為 [2020 整年] 在 [台北市] 的 [自行車竊盜] 事件 (共 291 筆資料)		
日期	時間	地點
1090102	14~16	臺北市松山區民生東路5段212巷1~30號
1090103	18~20	臺北市士林區岩山里忠義街91巷1~30號
1090105	12~14	臺北市中正區黎明里公園路31~60號
1090106	06~08	臺北市大安區敦安里(仁愛路四段)121~150號
1090106	08~10	臺北市北投區大同里光明路1~30號
1090106	12~14	臺北市中山區林森北路121~150號
1090108	16~18	臺北市大同區鄭州路121~150號
1090109	06~08	臺北市北投區溫泉里溫泉路58巷1~30號
1090110	00~02	臺北市信義區國興里虎林街222巷31~60號
1090110	04~06	臺北市大安區住安里信義路4段30巷27弄1~30號
1090111	08~10	臺北市大安區敦煌里(仁愛路4段)222巷1~30號
1090111	12~14	臺北市信義區松山路151~180號
1090113	20~22	臺北市大安區住安里瑞安街1~30號
1090114	16~18	臺北市北投區文林里石牌路1段166巷2弄1~30號
1090115	18~20	臺北市大安區學府里羅斯福路4段1~30號
1090117	06~08	臺北市大安區龍陣里和平東路2段175巷31~60號
1090118	08~10	臺北市南港區聯成里東新街83巷1~30號
1090120	08~10	臺北市文山區興光里興隆路3段1~30號
1090122	12~14	臺北市南港區中南里市民大道8段511~540號
1090123	12~14	臺北市士林區福德里基河路271~300號
1090123	12~14	臺北市大同區赤峰街1~30號
1090124	18~20	臺北市士林區德行里中山北路6段61~80號
1090125	02~04	臺北市大安區龍生里和平東路2段91~120號
1090125	14~16	臺北市北投區開明里15鄰珠海路91~120號
1090127	10~12	臺北市信義區忠孝東路5段780巷23弄1~30號
1 2 3 4 5 ... Next Last		

利用 Ruby 插件 : [Kaminari](#) 來實作這個功能

Kaminari

build passing

maintainability B

docs

A Scope & Engine based, clean, powerful, customizable and sophisticated paginator for modern web app frameworks and ORMs

圖表查詢

搜尋欄位：

一樣採用下拉式選單供使用者做查詢的動作

台北市各區失竊案件數量圖表

年分

2021

月份

整年

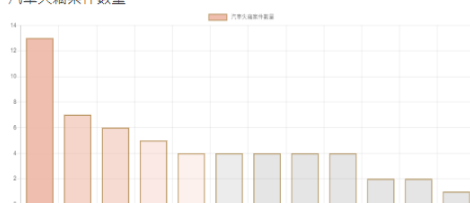
搜尋

以下為 [2019 整年] 的台北市各區失竊事件數量圖表

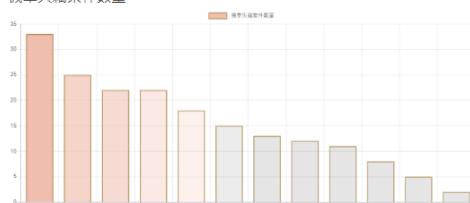
腳踏車失竊案件數量



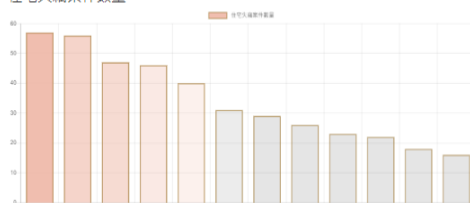
汽車失竊案件數量



機車失竊案件數量



住宅失竊案件數量



腳踏車失竊案件數量



汽車失竊案件數量



機車失竊案件數量



住宅失竊案件數量



圖表查詢

圖表顯示：

依照使用者的查詢內容，撈取目標資料，並製作圖表。我們利用 [chart.js](#) 來製作圖表。



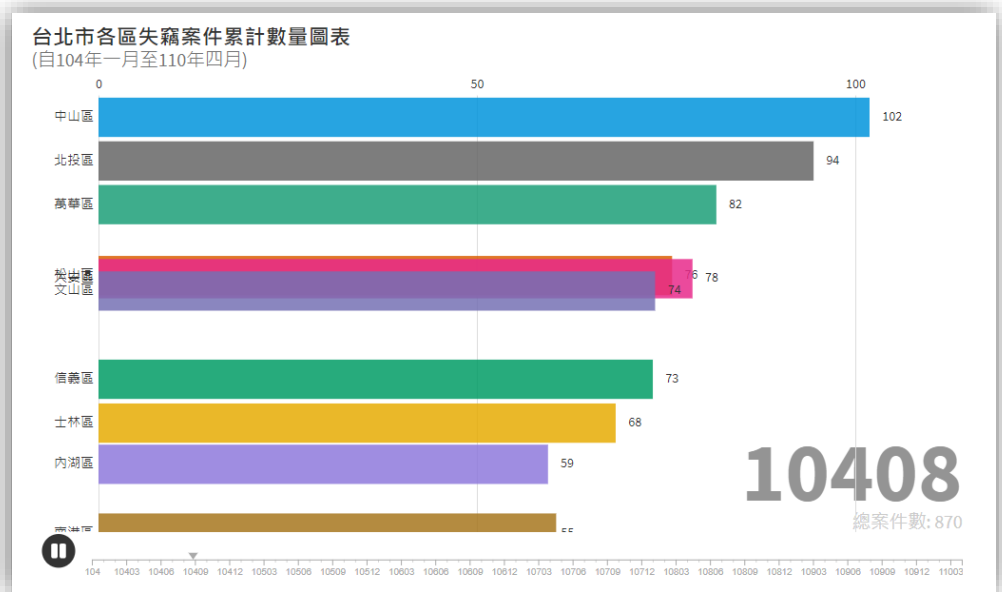
```
216 @chart = []
217 @name = ["腳踏車失竊案件數量", "汽車失竊案件數量", "機車失竊案件數量", "住宅失竊案件數量"]
218 tmp = Hash.new
219 BikeStole.where("date LIKE '#{date_search}%'").select("substr(location,4,3) as d,count(*) as n").group("d").order("n desc").each do |s|
220   tmp[s.d] = s.n
221 end
222 @chart << tmp
223 tmp = Hash.new
224 CarStole.where("date LIKE '#{date_search}%'").select("substr(location,4,3) as d,count(*) as n").group("d").order("n desc").each do |s|
225   tmp[s.d] = s.n
226 end
227 @chart << tmp
228 tmp = Hash.new
229 ScooterStole.where("date LIKE '#{date_search}%'").select("substr(location,4,3) as d,count(*) as n").group("d").order("n desc").each do |s|
230   tmp[s.d] = s.n
231 end
232 @chart << tmp
233 tmp = Hash.new
234 HomeStole.where("date LIKE '#{date_search}%'").select("substr(location,4,3) as d,count(*) as n").group("d").order("n desc").each do |s|
235   tmp[s.d] = s.n
236 end
237 @chart << tmp
```

這邊是圖表資料撈取的過程：

1. 針對每一個 table 進行搜尋
2. 篩選時間範圍
3. 字串切割出行政區
4. Group By 行政區
5. 對累積數量做降冪排序

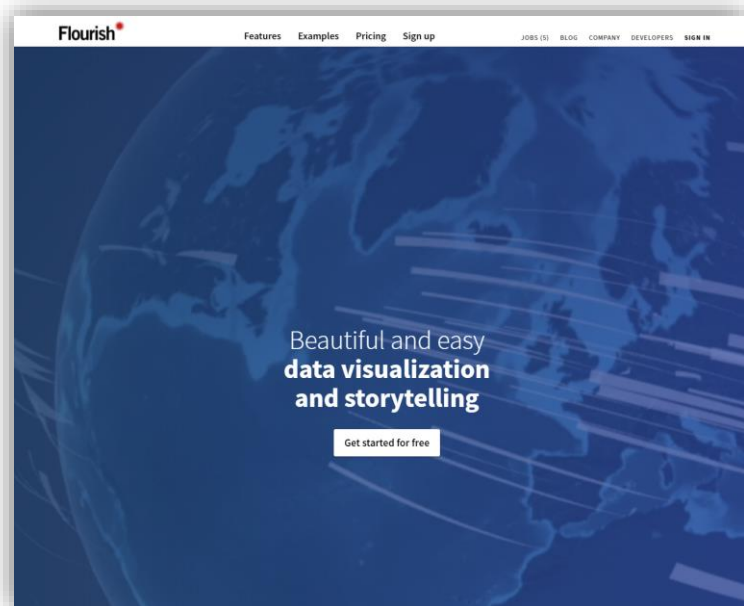
圖表查詢

動態圖表製作：



撈取所需資料，存入 excel，並匯入 Flourish Studio 網站中。
再將圖表以 iframe 的方式嵌入網頁當中。

[Flourish | Data Visualization & Storytelling](https://flourish.studio)



圖表查詢

動態圖表資料撈取：

```

45 t = []
46 ["萬華區", "松山區", "文山區", "大安區", "大同區", "士林區", "南港區", "北投區", "內湖區", "信義區", "中正區", "中山區"].each do |dis|
47   tmp = [0]
48   104.upto(110) do |y|
49     1.upto(12) do |m|
50       break if (y==110&m==5)
51       date_search = y.to_s+format("%02d",m).to_s
52
53       count = BikeStole.where("date LIKE '#{date_search}%'").where("location LIKE '%#{dis}%'").count
54       count += ScooterStole.where("date LIKE '#{date_search}%'").where("location LIKE '%#{dis}%'").count
55       count += HomeStole.where("date LIKE '#{date_search}%'").where("location LIKE '%#{dis}%'").count
56       count += CarStole.where("date LIKE '#{date_search}%'").where("location LIKE '%#{dis}%'").count
57
58       tmp << count+tmp[-1]
59     end
60   end
61   t << tmp
62 end
63 end

```

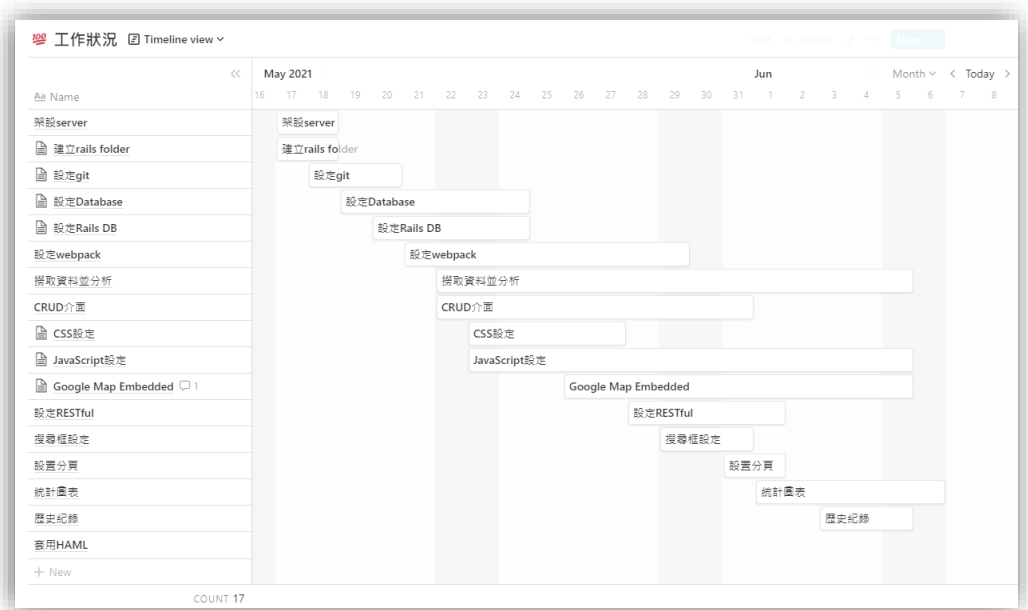
從104的一月開始，逐月搜尋各行政區的竊案數量。

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	區域	104	10401	10402	10403	10404	10405	10406	10407	10408	10409	10410	10411	10412	10501
2	萬華區	0	20	27	40	50	55	57	69	74	83	95	98	109	113
3	松山區	0	13	21	31	38	49	53	67	69	77	81	85	93	102
4	文山區	0	13	20	28	38	50	57	63	71	74	79	83	89	95
5	大安區	0	10	19	32	39	45	54	58	70	80	87	94	104	114
6	大同區	0	3	6	7	16	22	25	28	35	44	55	62	71	74
7	士林區	0	3	13	19	25	34	40	45	59	70	78	86	96	110
8	南港區	0	14	18	28	35	37	43	47	55	57	61	66	68	72
9	北投區	0	10	23	31	41	49	62	71	86	96	99	105	121	134
10	內湖區	0	8	18	23	38	46	49	51	56	60	65	74	85	98
11	信義區	0	12	25	27	30	43	51	58	69	74	84	93	100	105
12	中正區	0	9	14	21	22	31	39	46	52	66	85	93	103	107
13	中山區	0	12	19	25	37	44	64	78	90	104	114	122	137	146

匯入 Flourish Studio 製作圖表

Area	104	10401	10402	10403	10404	10405	10406	10407	10408	10409	10410	10411	10412	10501
萬華區	0	20	27	40	50	55	57	69	74	83	95	98	109	113
松山區	0	13	21	31	38	49	53	67	69	77	81	85	93	102
文山區	0	13	20	28	38	50	57	63	71	74	79	83	89	95
大安區	0	10	19	32	39	45	54	58	70	80	87	94	104	114
大同區	0	3	6	7	16	22	25	28	35	44	55	62	71	74
士林區	0	3	13	19	25	34	40	45	59	70	78	86	96	110
南港區	0	14	18	28	35	37	43	47	55	57	61	66	68	72
北投區	0	10	23	31	41	49	62	71	86	96	99	105	121	134
內湖區	0	8	18	23	38	46	49	51	56	60	65	74	85	98
信義區	0	12	25	27	30	43	51	58	69	74	84	93	100	105
中正區	0	9	14	21	22	31	39	46	52	66	85	93	103	107
中山區	0	12	19	25	37	44	64	78	90	104	114	122	137	146

Project 進度



上圖為實際的工作進度圖表，跟原本計畫的相去不遠。有考慮到 google map 的功能可能會遇上重重困難，因此給它很多的緩衝時間。另外，在後面的工作大致完成的時候，突然覺得可以加上圖表的功能，算是沒有在原本的計畫之中。在排定流程的時候，都有為可能遇到未知的 bug 做好時間上的緩衝，幸好，在遇到問題的時候都能在緩衝的時間內解決，或是改採用其他方案在時限內趕上進度。

分工狀況

陳伯庭：

- 資料庫建立
- 簡報、影片製作
- 搜尋列表製作

高靖：

- 簡報製作
- 網頁前端設計及維護
- Google Map 置入

郭晉維：

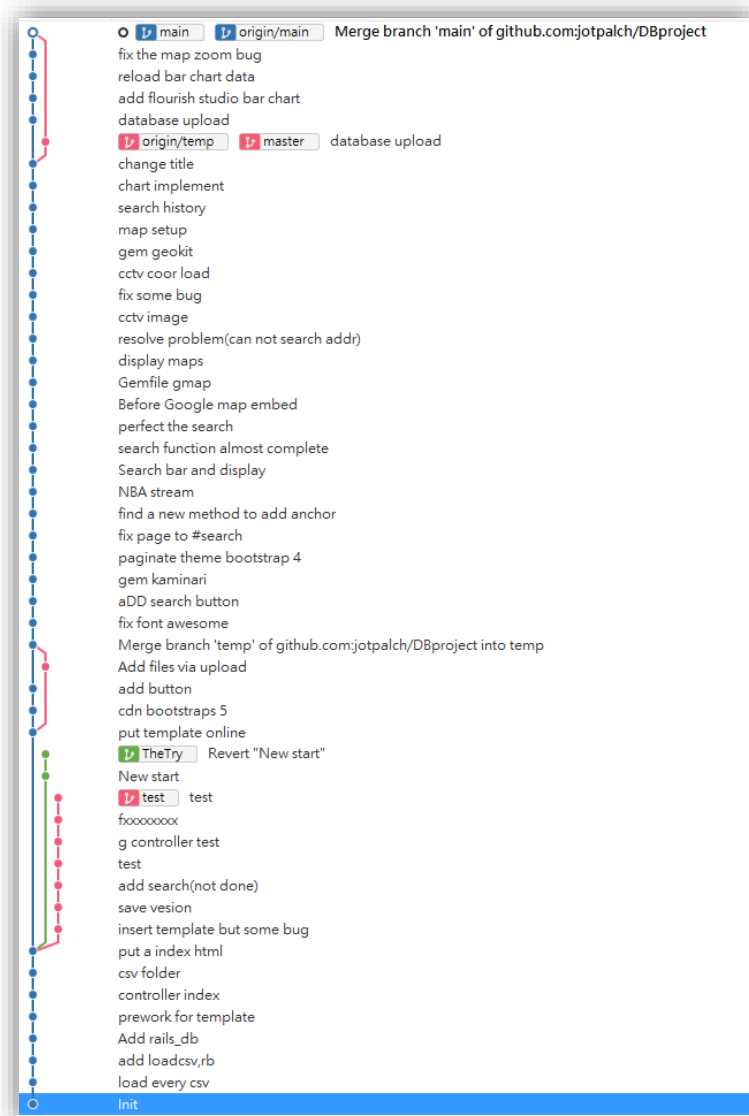
- ER Diagram 繪製
- Query 設計
- 歷史紀錄製作

陳璋誠：

- 整合工作、雜務
- 網頁前後端設計及維護
- 搜尋圖表製作
- 影片製作

基本上，我們每個組員都盡心盡力地完成這次的專案。大家各司其職，各盡其責，組內氣氛融洽。遇到問題時也會一起討論，共同尋求答案。在完成這個專案的過程中一起學習、一起成長。

版本管理



第一次使用 git 來做版本管理，發覺這個功能真的是開發必備。爬了許多文，才好不容易弄懂如何操作，雖然每隔一段時間就要記得 git 一次，但如果後面把網頁搞爆了就不用擔心，可以直接回復就好。另外，還可以做分支處理，讓開發新功能移到分支去做，不會影響到主幹。

更重要的是，可以進行多人共同協作一份專案，每個人在做完自己的部分都可以上傳到 github 讓其他組員接續工作。

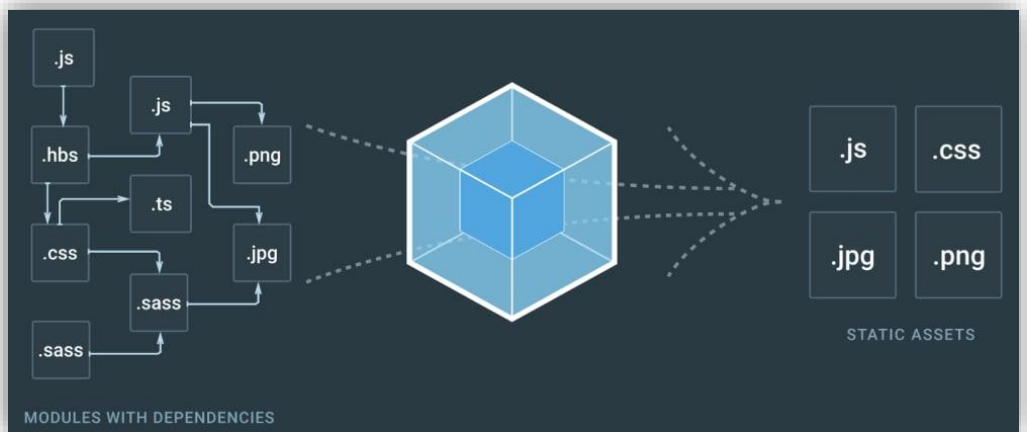
[jotpalch/DBproject: 2021 資料庫Final Project \(github.com\)](https://github.com/jotpalch/DBproject)

Ruby on Rails 問題

在專案進行的同時，經常需要加入許多插件。但有些插件年久失修，不支援最新的 Ruby on Rails 版本，因此需要降低版本才可使用。後來發現，雖然 Ruby on Rails 標榜快速部屬，但那是依照他的慣例的情況之下。一旦跳脫慣例，就有成千上萬的 bug 等著你解決，有些明明用 php 就可以輕鬆解決的問題，往往要搞上好幾個小時。除此之外，相較於 php，Ruby on Rails 的線上資源少了太多，甚至有時候連 Stack Overflow 都沒有人問過，只能慢慢地從操作守則以及說明文件當中找尋答案。中間甚至一度有放棄使用 Ruby on Rails 的念頭，信好在組員以及網路的協助之下，順利的找到了解決的方案。

Webpacker 問題

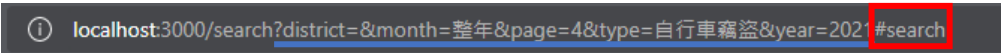
在 Ruby on Rails 裡面，預設有一個專門打包圖像、CSS 和 Javascript 檔案的程式叫 webpacker。它可以壓縮文件的大小，並以湊雜的方式為檔案命名。



這個看似很方便的工具，卻有一大堆的漏洞。常常無法編譯就算了，還沒有辦法移除。最後，我們用 cdn 的方式匯入 css, js 檔，避開 webpacker 的打包，圖像就放入 public 資料夾，方便存取。

前端網頁問題

在開發網頁的過程中，我們在外框架套用了模板。但是這個模板似乎對 Rails 不是那麼的友善，放上去之後完全無法顯示，經過了好一番功夫才得以解決，始能開發我們的新功能。

A screenshot of a web browser's address bar. The URL is `localhost:3000/search?district=&month=整年&page=4&type=自行車竊盜&year=2021#search`. The `#search` part at the end of the URL is highlighted with a red rectangular box.

在處理網址分配的時候也是上了問題，由於有做搜尋的部分，所以網址後方常常會綴有 query，但我們切換頁面是利用 hashtag 來實現。本以為只是在網址後方加上字串就能解決，但事情不如預期，在 query 後方家的字都會自動消失。後來是發現有一個可以加上 anchor 的功能，而這個 anchor 指的就是那個 hashtag。

影片製作問題

本來要採用 Animaker 製作影片，輸出之後發現效果不好，因此轉而使用 Adobe Premiere 來製作影片。

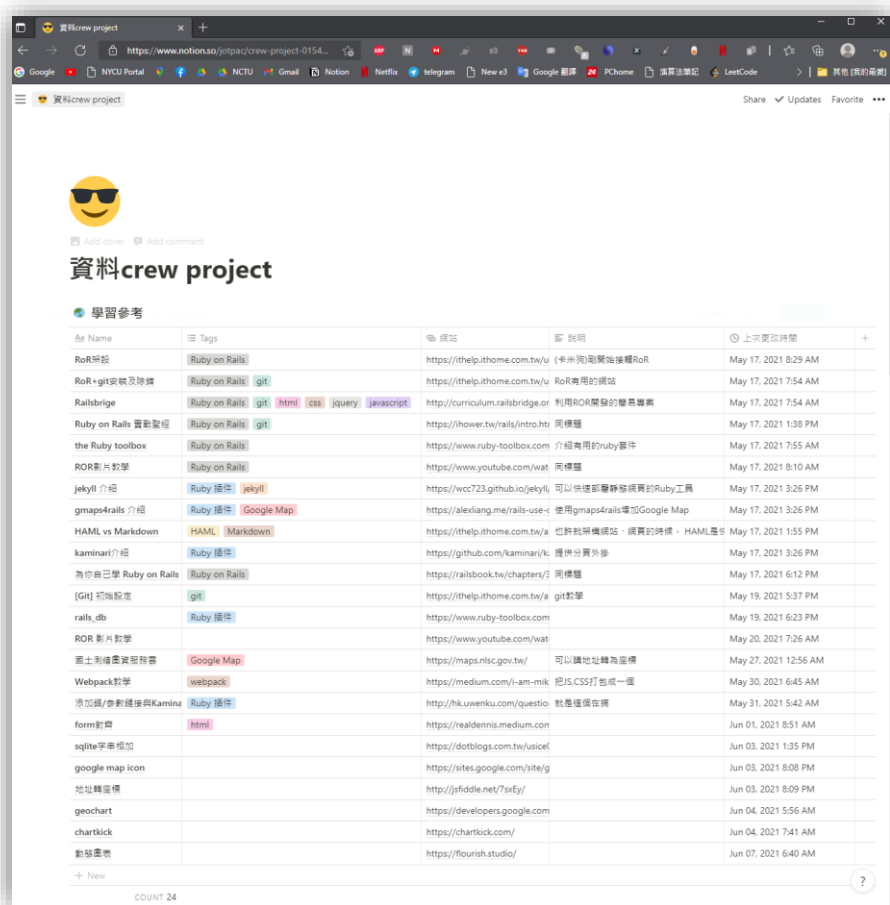


<https://youtu.be/8-N4cKgbnK0>

配音的部分，本來要採用 Google 小姐的語音，但發現好像過於俗套。最後採用 Edge 瀏覽器的內建 pdf 讀稿語音。

討論平台

由於我們四個組員都住在同一間或是隔壁寢室，因此我們的問題常常都是當面討論解決，討論平台的使用可能就沒有那麼頻繁。但我們還是有開創一個屬於這個專案的資源蒐集分頁，讓大家可以把找過的資源蒐集在一起，方便往後的查詢。



<https://www.notion.so/jotpac/crew-project-01549d00f13449c897e9d10ff2c0e891>