台北 先看才行

交通安全資訊網

2021 資料庫系統概論 第五組
李璨宇 張辰宇 黃則維 劉承遠
期末報告（第七版）

0 目錄

1 Datasets

1.1 A1A2 交通事故

資料表結構

欄位名稱	類型	是否為 primary key
事故編號	Int	True
發生年	Int	False
發生月	Int	False
發生日	Int	False
發生時	Int	False
發生分	Int	False
處理別	Varchar(5)	False
區序	Varchar(50)	False
肇事地點	Int	False
死亡人數	Int	False
受傷人數	Int	False
當事人數	Bigint	False
天候	Int	False
速限	Int	False
道路型態	Int	False
事故位置	Int	False

欄位說明

提供臺北市 A1 及 A2 類交通事故事件發生紀錄，包含發生時間、傷亡人數、天候、速限等資訊，已正規化。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
101 年-臺北市 A1 及 A2 類交通事故明細.csv

102 年-臺北市 A1 及 A2 類交通事故明細.csv
103 年-臺北市 A1 及 A2 類交通事故明細.csv
104 年-臺北市 A1 及 A2 類交通事故明細.csv
105 年-臺北市 A1 及 A2 類交通事故明細.csv
106 年-臺北市 A1 及 A2 類交通事故明細.csv
107 年-臺北市 A1 及 A2 類交通事故明細.csv
108 年-臺北市 A1 及 A2 類交通事故明細.csv
109 年-臺北市 A1 及 A2 類交通事故明細.csv

1.2 A1A2 交通事故明細

資料表結構

欄位名稱	類型	是否為 primary key
明細編號	Int	True
事故編號	Int	False
當事人序	Int	False
車種	Varchar(5)	False
性別	Int	False
年齡	Int	False
受傷程度	Int	False

欄位說明

資料來源同 A1A2 交通事故，但把每起事故中的每個當事人都當作一筆資料去做正規化後的資料分割。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
101 年-臺北市 A1 及 A2 類交通事故明細.csv
102 年-臺北市 A1 及 A2 類交通事故明細.csv
103 年-臺北市 A1 及 A2 類交通事故明細.csv
104 年-臺北市 A1 及 A2 類交通事故明細.csv
105 年-臺北市 A1 及 A2 類交通事故明細.csv

106 年-臺北市 A1 及 A2 類交通事故明細.csv
107 年-臺北市 A1 及 A2 類交通事故明細.csv
108 年-臺北市 A1 及 A2 類交通事故明細.csv
109 年-臺北市 A1 及 A2 類交通事故明細.csv

1.3 處理別

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar (4)	True
對應項目	Varchar (12)	False
備註	Varchar (20)	False

欄位說明

本表對應處理別的各種代碼之涵義。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.4 天候

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar (4)	True
對應項目	Varchar (12)	False
備註	Varchar (20)	False
cat	Int	False

欄位說明

本表對應天候的各種代碼之涵義，cat 是為了占卜而對資料做進一步分類用的代碼。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.5 道路型態

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar(4)	True
對應項目	Varchar(12)	False
備註	Varchar(20)	False

欄位說明

本表對應道路型態的各種代碼之涵義。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.6 事故位置

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar (4)	True
對應項目	Varchar (15)	False
備註	Varchar (20)	False

欄位說明

本表對應事故位置的各種代碼之涵義。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.7 車種

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar (4)	True
對應項目	Varchar (12)	False
備註	Varchar (20)	False
cat	Int	False

欄位說明

本表對應車種的各種代碼之涵義，cat 是為了占卜而對資料做進一步分類用的代碼。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.8 性別

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar(4)	True
對應項目	Varchar(12)	False
備註	Varchar(20)	False

欄位說明

本表對應性別的各種代碼之涵義。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.9 受傷程度

資料表結構

欄位名稱	類型	是否為 primary key
代碼	Varchar(4)	True
對應項目	Varchar(12)	False
備註	Varchar(20)	False

欄位說明

本表對應受傷程度的各種代碼之涵義。

資料來源

A1 及 A2 交通事故資料
https://data.gov.tw/dataset/130110
交通事故代碼對照表.csv

1.10 自訂地標

資料表結構

欄位名稱	類型	是否為 primary key
地標編號	Int	True
會員編號	Int	False
地點	Varchar(200)	False
lat	Double	False
lng	Double	False
類型	Int	False
建立時間	Datetime	False

欄位說明

提供會員自行新增各類型地標，包含經緯度、地點、建立時間等。

資料來源

為註冊會員自行新增之地標的資料。

1.11 交通事故斑點圖

資料表結構

欄位名稱	類型	是否為 primary key
發生時間	Varchar(15)	False
處理別	Int	False
肇事地點	Varchar(50)	False
X	Double	False
Y	Double	False

欄位說明

提供臺北市道路交通事故發生時間、處理別、肇事地點、座標點位資料。

資料來源

臺北市道路交通事故斑點圖
https://data.gov.tw/dataset/136123
108 年臺北市道路交通事故斑點圖.csv
109 年臺北市道路交通事故斑點圖.csv

1.12 大隊網站公布固定桿地點表

資料表結構

欄位名稱	類型	是否為 primary key
編號	Int	True
功能	Varchar(30)	False
設置路段	Varchar(20)	False
設置地點	Varchar(20)	False
轄區	Varchar(3)	False
拍攝方向	Varchar(5)	False
速限	Int	False

lat	Double	False
lng	Double	False

欄位說明

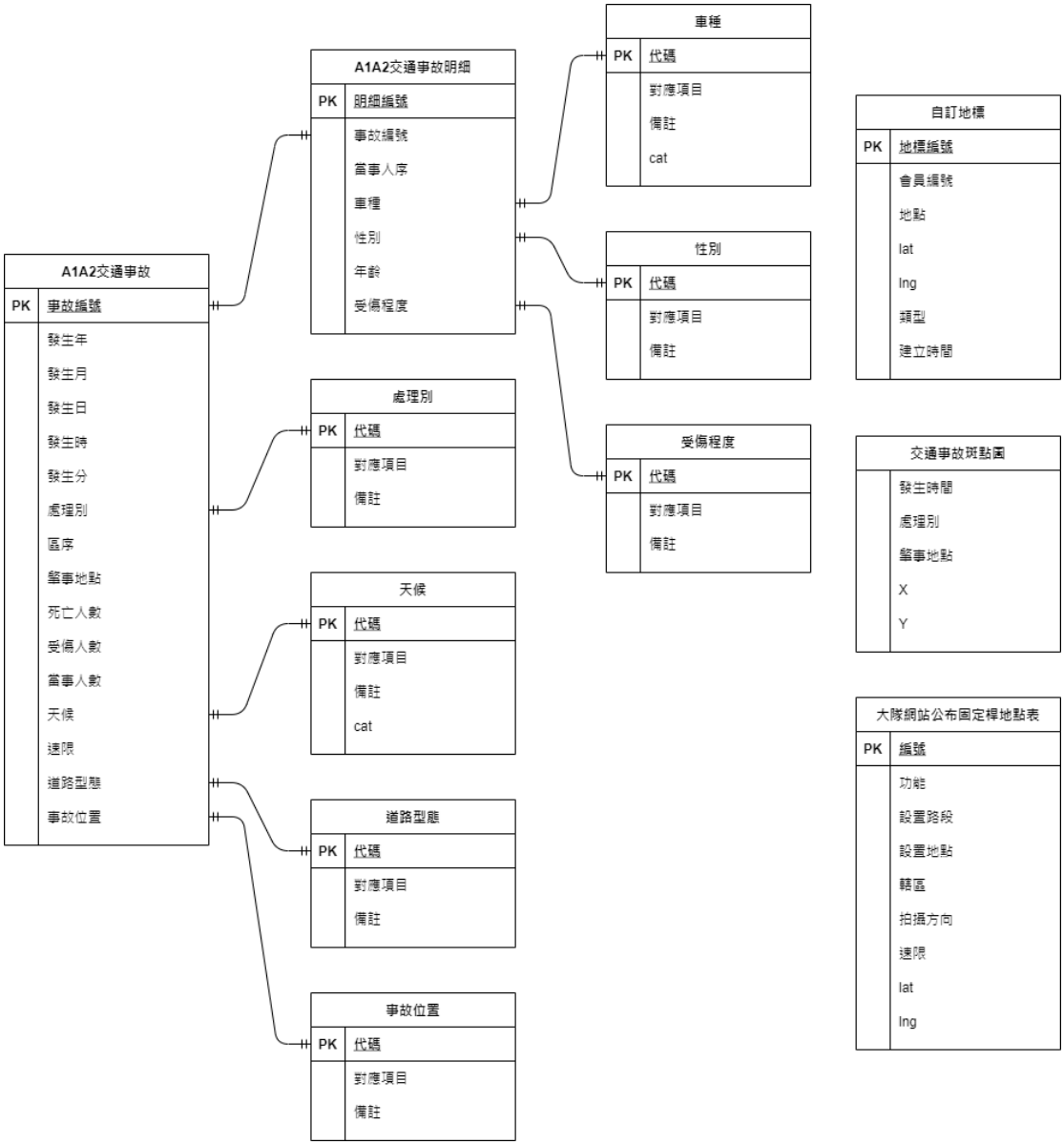
提供臺北市現有車輛超速自動照相設備之設置地點及地點路段之限速值等資料。

資料來源

臺北市固定測速照相地點表
https://data.gov.tw/dataset/130111
臺北市固定測速照相地點表 110.4.23.csv

2 Database Design

2.1 正規化

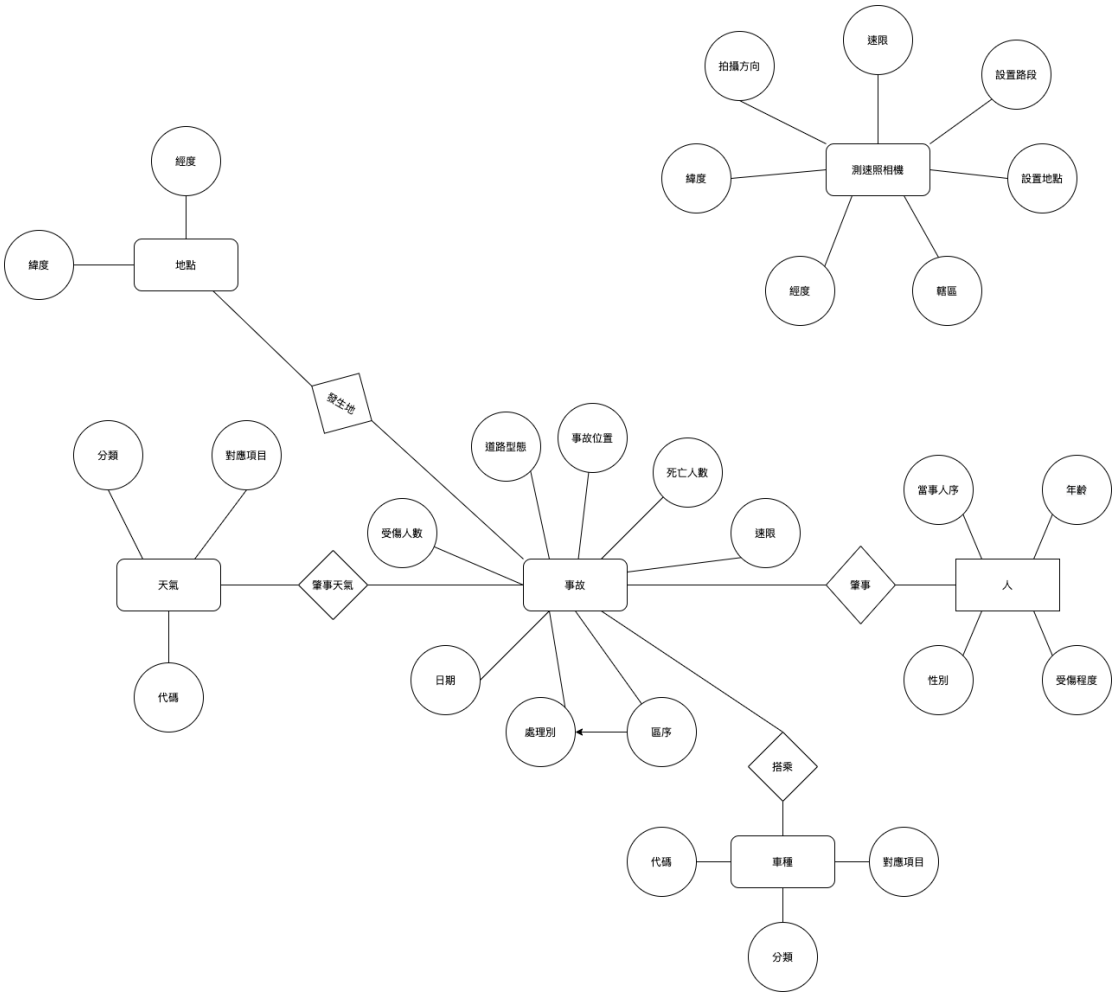


在一開始我們取得的資料並未正規化，其中的台北市 A1A2 交通事故表甚至未符合 1NF,也就是說他一個事件會有好幾個 row(同一個肇事案件裡每個肇事者都有一個 row,且表中也沒有 id 作為 key)。因此我們決定把台北市 A1A2 交通事故正規化成 A1A2 交通事故以及 A1A2 交通事故明細,其中交通事故以事故編號為主鍵而交通事故明細則以明細編號為主鍵（使其滿足 2NF)。

而原本資料平台提供的代碼對應表我們也用正規化的方式把它拆成幾個較為獨

立的表，包括處理別、天候、道路型態、車種、性別受傷程度以及事故位置。這些表都以代碼為主鍵以利我們之後利用 SQL 查詢時可以方便對應（滿足 3NF）。

2.2 ER Model



我們以每起事故為中心，向外延伸出車種，肇事人，肇事地點以及天氣。每個 entity 都具備其對應到的 attribute，其中車種以及天氣都具有分類這項 attribute，這是我們為了縮小分類範圍而為其額外加上的屬性。

2.3 預存程序與查詢

預存程序（Stored Procedure），是在資料庫儲存複雜的 SQL 語句，方便外部程式呼叫的資料庫物件，可以視為資料庫的一種函式或子程式。
我們將所有 SQL 查詢全部以預存程序來實現，原因有以下四點：

建立標準化查詢

使用預存程序後，寫 SQL 查詢語句的開發者與寫應用程式（PHP）的開發者不需要是同一人，就可以輕易使用對方所寫的查詢，這尤其在查詢語句複雜時（可能超過 30 行）特別有效，不需要理解內容即可輕易使用。

方便管理傳入參數

可以預先設定預存程序的變數類型與數量，使查詢時能保證不會有格式不相容的問題。

讓程式碼更簡潔

利用預存程序，可以將在 PHP 實作的查詢字串長度降低，更簡潔美觀，而且不需要在撰寫 PHP 時需要同時為 SQL 語法偵錯，使開發進行更有效率。尤其當重複的資料要被不同頁面取得時，透過預存程序可以更快速的重現已使用過的查詢。

查詢維護更加容易

若有一個 SQL 語句需要修改，不需要改動 PHP 網頁的查詢命令，只需要修改相應預存程序的內容即可，實現 SQL 設計與 PHP 開發的高度分治，效率提升。

在這些基礎上，我們又將查詢依照回傳值分為三種類型，分別是 2.4 的統計查詢、2.5 的列表查詢以及 2.6 的純量查詢。

2.4 統計查詢

統計查詢指的是資料經過彙總函式（AVG、COUNT、SUM 等函數）聚合所成的結果。

年齡事故統計圖

由於一開始取得的資料一起事故會有多個當事者，而每個當事人會有一筆資料，所以無法由表直接統計年齡事故，因此必須先把同一起事故的肇事者全部合併為一筆資料以利我們統計數字。
我們再以 select case 的方式把每個年齡分類到其所屬的年齡層，再分別統計每個年齡層所包含的人數來依此排名。

地點事故死亡統計圖

由於一開始取得的資料一起事故會有多個當事者，而每個當事人會有一筆資料，所以無法由表直接統計地點事故，因此必須先把同一起事故的肇事者全部合併為一筆資料以利我們統計數字。
我們以 group by 的方式先將不同地點分類，再依此統計死亡數量。值得注意的是，因為對應項目是以代碼作為紀錄，因此必須藉由事故對照表先做對應的轉換（型態也不同，所以要用到 cast）方可做統計。

車種事故統死亡率統計圖

由於一開始取得的資料一起事故會有多個當事者，而每個當事人會有一筆資料，所以無法由表直接統計車種事故，因此必須先把同一起事故的肇事者全部合併為一筆資料以利我們統計數字。
由於統計人數的事故明細並未包含車種的資料，所以必須先用 left join 來取得資料。再分別統計好總人數以及死亡人數後，我們再相除算得死亡率。

```
select t3.`對應項目`, t3.cnt/t4.cnt as p
from(
select `對應項目`, t1.cnt
from
(select A1A2交通事故明細.`車種`,count(DISTINCT(`A1A2交通事故`.`事故編號`)) as cnt
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故.`死亡人數` > 0
group by A1A2交通事故明細.`車種`) as t1,
車種 as t2
where t1.`車種` = t2.`代碼`
)as t3,(
  select `對應項目`, t1.cnt
from
(select A1A2交通事故明細.`車種`,count(DISTINCT(`A1A2交通事故`.`事故編號`)) as cnt
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`

group by A1A2交通事故明細.`車種`) as t1,
車種 as t2
where t1.`車種` = t2.`代碼`
) as t4
where t3.`對應項目` = t4.`對應項目`
order by p desc
```

天候事故統計圖

由於一開始取得的資料一起事故會有多個當事者，而每個當事人會有一筆資料，所以無法由表直接統計天候事故，因此必須先把同一起事故的肇事者全部合併為一筆資料以利我們統計數字。
我們以 group by 的方式先將不同天候分類，再依此統計死亡數量以及總數量。值得注意的是，因為對應項目是以代碼作為紀錄，因此必須藉由事故對照表先做對應的轉換（型態也不同，所以要用到 cast）方可做統計。最後再以相除的方式計算死亡率。

```
事故天候（死亡機率）排名
create procedure '事故天候（死亡機率）排名'
AS
select t3.死亡數量/t4.事故數量  as 死亡機率 , t4.`天候` as  天候
from (
select t1.cnt as 死亡數量, t2.`對應項目` as 天候
from (SELECT COUNT(*) as cnt , `天候` FROM `A1A2交通事故明細` as t1 WHERE `死亡人數`>0 GROUP BY`天候`) as t1, `天候` as t2
WHERE  cast(t1.`天候` as char) = cast(t2.`代碼` as char)
) as t3, (
select t1.cnt as 事故數量, t2.`對應項目` as 天候
from (SELECT COUNT(*) as cnt , `天候` FROM `A1A2交通事故明細` as t1 GROUP BY`天候`) as t1, `天候` as t2
WHERE cast(t1.`天候` as char) = cast(t2.`代碼` as char)
) as t4
where t3.`天候` = t4.`天候`
order by  死亡機率 desc
```

2.5 列表查詢

列表查詢指的是會傳回多個資料列，每列有多個資料值的查詢。

事故查詢

事故查詢透過預存程序可以實現按照不同模式（mode）來達到以不同排序方法呈現資料，其中 IF 語句使 SQL 語法更接近一般程式語言的判斷邏輯。

```
IF ord = 1 THEN BEGIN
        SELECT `事故編號`,`發生年`, `發生月`, `發生日`, `發生時`,
                        `發生分`,`處理別`,`區序`, `肇事地點`, `死亡人數`,
                        `受傷人數`, `當事人數`
        FROM `A1A2交通事故`
        WHERE `發生年` = year AND `區序` LIKE CONCAT('%',region,'%')
        ORDER BY `受傷人數` DESC, `死亡人數` DESC, `當事人數` DESC,
                 `發生月` ASC, `發生日` ASC, `發生時` ASC,
                 `發生分` ASC;
END;
ELSEIF ord = 2 THEN BEGIN
        SELECT `事故編號`,`發生年`, `發生月`, `發生日`, `發生時`,
                        `發生分`,`處理別`,`區序`, `肇事地點`, `死亡人數`,
                        `受傷人數`, `當事人數`
        FROM `A1A2交通事故`
        WHERE `發生年` = year AND `區序` LIKE CONCAT('%',region,'%')
        ORDER BY `死亡人數` DESC, `受傷人數` DESC, `當事人數` DESC,
                 `發生月` ASC, `發生日` ASC, `發生時` ASC,
                 `發生分` ASC;
END;
ELSEIF ord = 3 THEN BEGIN
        SELECT `事故編號`,`發生年`, `發生月`, `發生日`, `發生時`,
                        `發生分`,`處理別`,`區序`, `肇事地點`, `死亡人數`,
                        `受傷人數`, `當事人數`
        FROM `A1A2交通事故`
        WHERE `發生年` = year AND `區序` LIKE CONCAT('%',region,'%')
        ORDER BY `當事人數` DESC, `死亡人數` DESC, `受傷人數` DESC,
                 `發生月` ASC, `發生日` ASC, `發生時` ASC,
                 `發生分` ASC;
END;
ELSE BEGIN
        SELECT `事故編號`,`發生年`, `發生月`, `發生日`, `發生時`,
                        `發生分`,`處理別`,`區序`, `肇事地點`, `死亡人數`,
                        `受傷人數`, `當事人數`
        FROM `A1A2交通事故`
        WHERE `發生年` = year AND `區序` LIKE CONCAT('%',region,'%')
        ORDER BY `發生月` ASC, `發生日` ASC, `發生時` ASC,
                 `發生分` ASC;
END;
END IF
```

事故明細查詢

經過預先建立的事故編號（自動遞增，Auto Increment），查詢對應的當事人資訊。

2.6 純量查詢

純量查詢指的是查詢只會回傳 1 個值（純量）。

安全占卜

占卜的部分有四個參數，分別是性別、年齡、天候以及車種。年齡的部分由於我們不想把分類分得這麼細因此我們設計四個模式分別代表四個年齡層，接著就可以用 if else 函式來分別處理不同年齡層。在天候以及車種方面我們同樣的因為不想把分類分得這麼細，因此另外將多個車種以及多個天後合併，接著再用子查詢的方式來取得參數對應的天候以及車種在用 in 的方式取得滿足條件的車種以及天候。另一方面由於用來統計人數的交通事故表並未包含各種參數屬性，因此我們必須先用 left join 的方式取得資訊接著在查找符合傳入參數的事件，最後在以 distinct 函式來計算事故件數，再把符合條件的死亡人數除上符合條件的事故件數來算的改參數條件下的死亡率。求得死亡率後我們在和平均肇事死亡率比較，用此結果算出在使用者輸入的參數條件下占卜出來的結果如果機率比平均死亡率高則為兇，若較低者為吉。

```
占卜
if mode = 1 then
select
(select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=18  and A1A2交通事故明細.`年齡`>= 0 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather)  and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) and A1A2交通事故.`死亡人數` > 0
)/(
select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=18  and A1A2交通事故明細.`年齡`>= 0 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) )
;

elseif mode = 2 then
select
(select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=40  and A1A2交通事故明細.`年齡`>= 19 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) and A1A2交通事故.`死亡人數` > 0
)/(
select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=40  and A1A2交通事故明細.`年齡`>= 19 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) )
;

elseif mode = 3 then
select
(select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=60  and A1A2交通事故明細.`年齡`>= 41 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) and A1A2交通事故.`死亡人數` > 0
)/(
select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=60  and A1A2交通事故明細.`年齡`>= 41 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) )
;

else
select
(select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=100  and A1A2交通事故明細.`年齡`>= 61 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) and A1A2交通事故.`死亡人數` > 0
)/(
select count(DISTINCT(`A1A2交通事故`.`事故編號`))
FROM A1A2交通事故
left join A1A2交通事故明細
on A1A2交通事故.`事故編號` =  A1A2交通事故明細.`事故編號`
where A1A2交通事故明細.`性別` = gender and A1A2交通事故明細.`年齡`<=100  and A1A2交通事故明細.`年齡`>= 61 and A1A2交通事故.`天候` in
(select `代碼` from `天候` where cat = weather) and A1A2交通事故明細.`車種` in (select `代碼` from `車種` where cat = vehicle) )
;
END IF;
```

參考資料

預存程序 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/%E5%AD%98%E5%82%A8%E7%A8%8B%E5%BA%8F

3 Database Management

3.1 建置環境

使用MySQL（版本8.0.25-0ubuntu0.20.04.1)進行資料庫建置。透過phpMyAdmin（版本4.9.5deb2）可以使授權的使用者（僅開發成員）由遠端連入指定資料庫進行CRUD與管理功能。
（詳細拓樸圖可至4.1章節查看）

3.2 應用程式與資料庫的連接方式

透過MySQLi連接我們使用的MySQL資料庫，MySQLi（MySQL Improved）是MySQL針對 PHP 所設計的一個擴充模組，我們利用它來建立與資料庫的連線（exec/sql.php），並可執行SQL查詢並返回結果。
使用的連接字串如下：

exec/sql.php

```
<? php
  $cn = mysqli_connect([Host IP], [username], [password],
  "DBProject")
  or die ("mysqli_connect failed!");
  $cn->set_charset('utf8mb4');
?>
```

因為包含資料庫的帳密資訊，因此在此僅提供使用範例，GitHub 上傳的版本也沒有包含實際使用的exec/sql.php。
若要進行查詢，只需要require連接字串檔案(exec/sql.php）即可。
以下提供查詢範例：

example.php

```
<? php
  require('exec/sql.php');
  $query = [SQL Query];
  $data = mysqli_query($cn, $query);
?>
```

3.3 資安防範

我們關注於 SQL Injection 的防範，所謂 SQL 注入（Injection）攻擊就是利用應用程式查詢資料庫時的指令，注入特定惡意程式碼（Query），以非法手段獲得資料庫內的內容，或更進一步地竄改數據，甚至刪除資料表，造成資料遺失。
我們採用以下三種方法防範此類問題。

資料庫端防護：限制權限

透過網站存取資料庫時，是使用我們另外建立的特別使用者，透過將該使用者權限降低，僅提供執行特定資料庫中特定預存程序（Procedure）的權限，就可有效防範未經授權的修改。

後端防護：跳脫字元

SQL Injection 最廣為人知的防範方法就是透過應用程式判斷並阻絕非法 Query 字串，我們透過 PHP MySQLi 提供的 mysqli_real_escape_string 函式實作跳脫字元處理。
範例如下：

example.php

```
<? php
  require('exec/sql.php');
  $query = sprintf("CALL AccidentRank('%s','%s','%s');",
    mysqli_real_escape_string($cn,$_REQUEST["year"]),
    mysqli_real_escape_string($cn,$_REQUEST["region"]),
    mysqli_real_escape_string($cn,$_REQUEST["mode"]));
  $data = mysqli_query($cn, $query);
?>
```

範例程式碼是擷取肇事查詢中的查詢字串，可見三個輸入$_REQUEST["year"]、$_REQUEST["region"]與$_REQUEST["mode"]皆經過跳脫字元處理，所有我們執行的查詢若有附加可變參數，皆使用以上規格撰寫。

物理防護：定期備份

異地定期備份是最古老的防護方式，可以防止許多異常狀況所造成的資料損失，缺點是還原的資料可能有不一致的問題，算是我們設下的最後一道防線。

可以改進的地方：參數化查詢

關於第二點的後端防護，其實除了跳脫字元外改以參數化查詢方式才能實現更有效的防範 SQL 注入攻擊，未來若要繼續開發此應用程式，對於安全部分則需要更加改進。

3.4 其他資料（非 SQL）

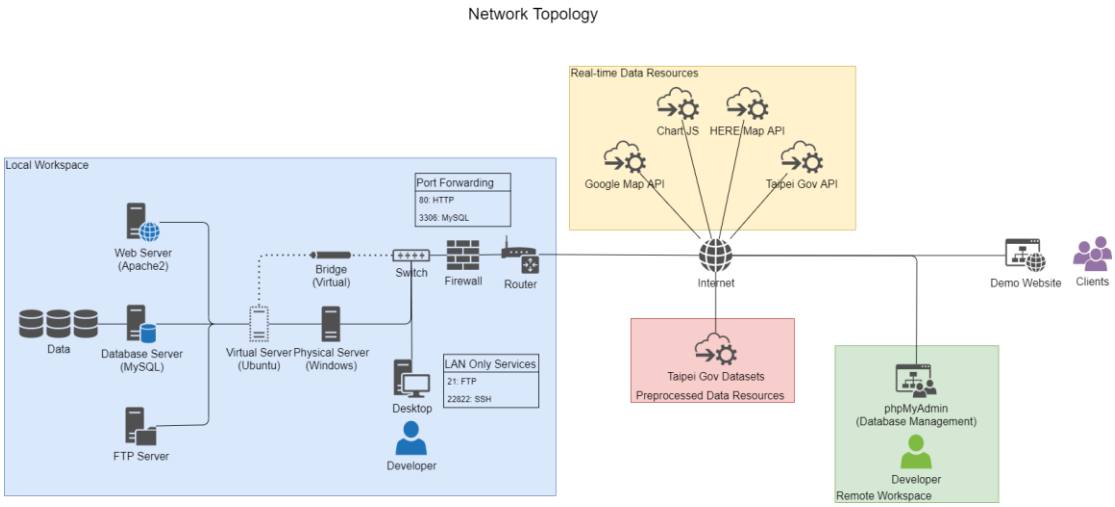
除了儲存在 MySQL 資料庫的資料表以外，還有一些資料是透過即時從 API 獲得的，例如在首頁的最新消息，其實是透過臺北市政府交通局所提供的「臺北市即時交通訊息」介接 API 取得 JSON 格式資料所建立的，這些即時性的資料就不會經過資料庫的處理，直接由網頁取得與分析處理，這些部分的詳細說明會在 4.5 章節提到。

參考資料

MySQLi - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/MySQLi
SQL 注入 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/SQL%E6%B3%A8%E5%85%A5
參數化查詢 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/%E5%8F%83%E6%95%B8%E5%8C%96%E6%9F%A5%E8%A9%A2

4 Application Architecture

4.1 架構介紹



上圖為我們整個系統的架構展示圖，顯示了裝置與服務之間的關係。以下則分別介紹四大區域的功能，以及使用者的如何透過網際網路連入我們的網站。

4.2 Local Workspace

Local Workspace 含有我們存放資料庫與網頁的伺服器：Ubuntu 虛擬機。建於 Windows 結構之上的 Ubuntu 虛擬機，提供了 HTTP、FTP、SSH、MySQL 等服務，方便我們進行網站建置與維護。之所以決定使用虛擬機的原因，是因為虛擬機提供快照功能，若不小心操作失誤導致系統出錯時可以很快還原成原始狀態。

其中虛擬機是以 Bridge 方式上網，路徑上不需要經過 Host（Windows 系統），在網路中可以視為一台真正的裝置。

所有外部網路傳送的封包都會經過 Router 內建的防火牆篩選並依照預先設定的規則轉送至內部網路（NAT 架構），此次使用的 Ubuntu 虛擬機對外僅開放 HTTP（Port 80）及 MySQL（Port 3306），其他非上述類型的外部封包都會被防火牆規則擋住。

於內網的開發者可以透過 FTP 服務將 PHP 文件、HTML 文件或多媒體檔案上傳至虛擬機，進而發布網站。

4.3 Remote Workspace

由於安全性的問題，我們並沒有開放所有 local workspace 提供的服務如 SSH 與 FTP…等，所有開發者只能透過 phpMyAdmin 或 MySQL Client 透過帳號密碼連入資料庫進行 CRUD 動作。

4.4 Preprocessed Data Resources

Taipei Gov Datasets

所有資料庫的資料含「A1A2 交通事故」、「A1A2 交通事故明細」、「交通事故斑點圖」等 12 個上列資料表，皆由政府資料平台中的台北市政府提供。其中「大隊網站公布固定桿地點表」之經緯度則是由 Google Geocoding API 查詢後儲存至資料庫。

4.5 Real-Time Data Resources

Taipei Gov API

首頁的最新消息是直接透過臺北市政府交通局提供的「臺北市即時交通訊息」介接 API 取得 JSON 格式資料所建立的。透過 Javascript 的 document.body.onload 事件觸發 AJAX 透過提供的 JSON API 取得資料，回傳的資料經過格式化後就可以以表格形式呈現在網頁上。

HERE Map API

由於 Google Map 提供的 API 流量限制較嚴格，因此這次我們並沒有在線上使用 Google 的服務，改採用 HERE Global B.V. 公司旗下的 HERE Map 地圖服務。雖然查詢準確度與執行效能不如 Google 服務，多方考量後使用此折衷方案。

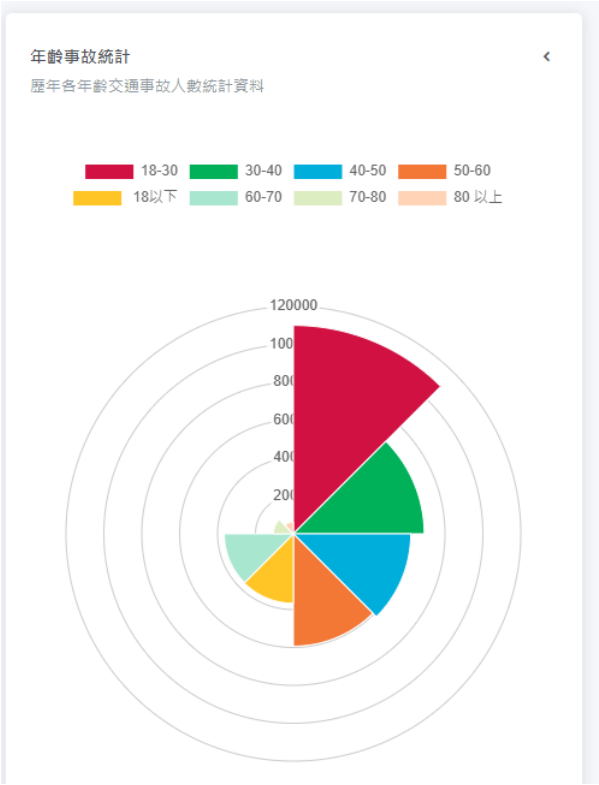
HERE Geocoder API

HERE Geocoder API 提供地址、路口、地標轉經緯度與反向查詢。我們將其用於「事故詳細資料」網頁中，事發地圖中的車禍地點經緯度定位。

由下圖下方可見「本 GPS 資訊由 Here Geocoding API 提供」表示此座標資訊由 Here Geocoder API 即時查詢與顯示。

Chart JS

Chart.js 是 Javascript 中一個可調整性極高的圖表 Library，支援八種常見的統計圖表樣式，製作出簡單明瞭的圖表以呈現資料視覺化。



4.6 Client

使用者透過瀏覽器經由網際網路瀏覽我們的網站(Demo 網址:demo.exodus.tw),首先會連線到 DNS 伺服器（Domain Name System）將域名（demo.exodus.tw）解析為目標伺服器的 IP 位址，再連線至我們的伺服器。可以將 DNS 想像成電話簿的功能，使用 DNS 的好處是使用者如果想要瀏覽某個網站，不需要記下數字形式的 IP 位址，只需要輸入域名就可以找到對應的 IP，方便許多。
使用者向我們的伺服器發送 HTTP 請求後,我們就會將相對應的網頁應答使用者,其中這些網頁若是動態網頁（PHP 程式），會先經過伺服器運算後才會回應給使用者。

4.7 使用軟體列表

由於使用軟體眾多，在此僅列出幾項比較重要的應用軟體，但實際要建置一個網站其實要透過多種服務才能順利完成。例如透過 SSH、VPN 及 FTP 實現遠端管理，以及經由 DNS 服務（不論代管或自架）解析域名，這些都是我們開發時不可或缺的部分，但由於篇幅的關係，我們只介紹與資料庫架構有直接關係的程式：

Oracle VirtualBox

用來虛擬化 Ubuntu 環境的軟體，可實現快照還原功能，讓我們伺服器系統錯誤時離線維護的時間降低。

Apache HTTP Server

架設 HTTP 網站的伺服器軟體，透過 PHP 語言可以實現動態網頁的功能，我們採用的版本是 Apache 2.4.41,並已關閉目錄瀏覽功能,以防止目錄遍歷(Directory traversal）攻擊。

MySQL

MySQL 作為資料庫是這次報告不可或缺的重要部分，我們採用的版本是 8.0.25-0ubuntu0.20.04.1。

PHP

PHP（PHP：Hypertext Preprocessor）是一種開源的程式語言，常用於網路開發並可嵌入 HTML 中使用。PHP 的語法借鑑吸收 C 語言、Java 和 Perl 等流行電腦語言的特點，易於一般程式設計師學習。PHP 的主要目標是允許網路開發人員快速編寫動態頁面，同時也被用於很多其他的領域。
作為這次網站專題的後端語言，是我們網頁與資料庫互動的重要對接介面，我們這次採用了 PHP 版本 7.4.3。

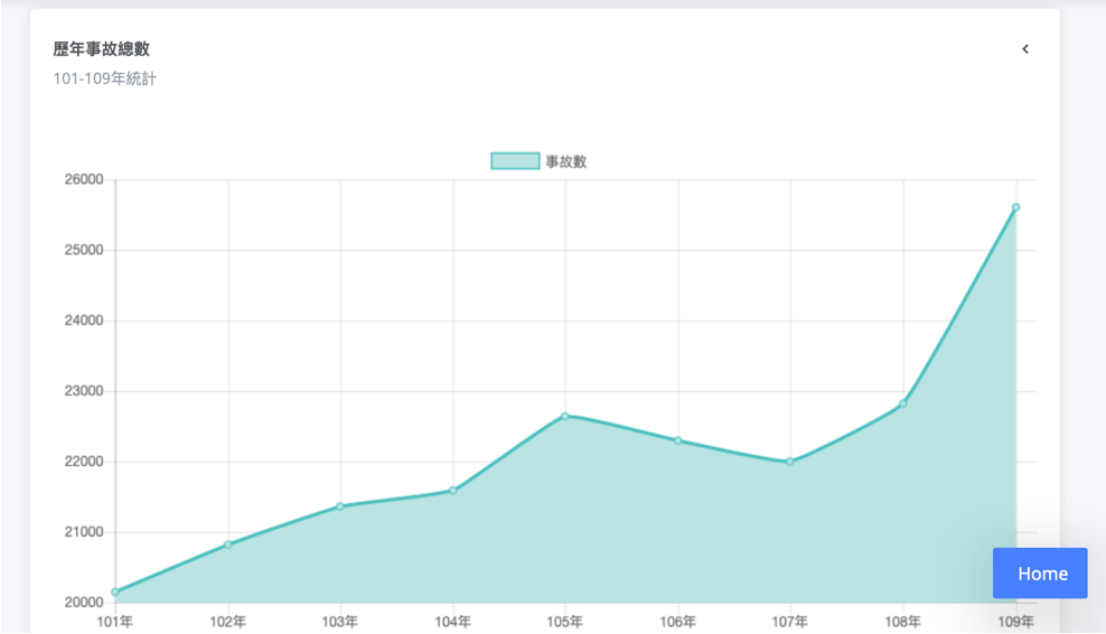
參考資料

[Day 30]Chart.js - 輕鬆完成資料視覺化 - iT 邦幫忙::一起幫忙解決難題，拯救 IT 人的一天
https://ithelp.ithome.com.tw/articles/10188031
域名系統 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/%E5%9F%9F%E5%90%8D%E7%B3%BB%E7%BB%9F
目錄遍歷 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/%E7%9B%AE%E5%BD%95%E9%81%8D%E5%8E%86
PHP - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/PHP

5 User Interface

5.1 基本介紹

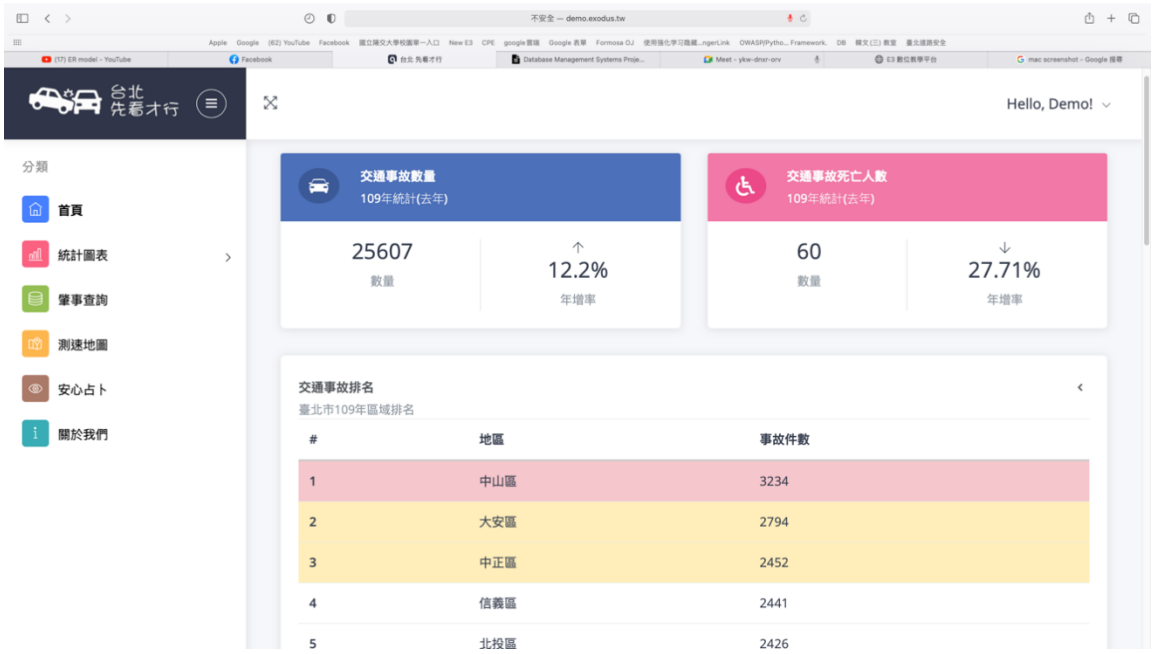
首先左上角放置我們的 Logo，並且右上方的導覽列會顯示使用者是否登入，左邊的導覽列則分別連結到各種功能的選項，右下角有一個浮動按鈕可以直接回首頁，我們認為這樣的設計讓使用者在使用上會更加直覺。

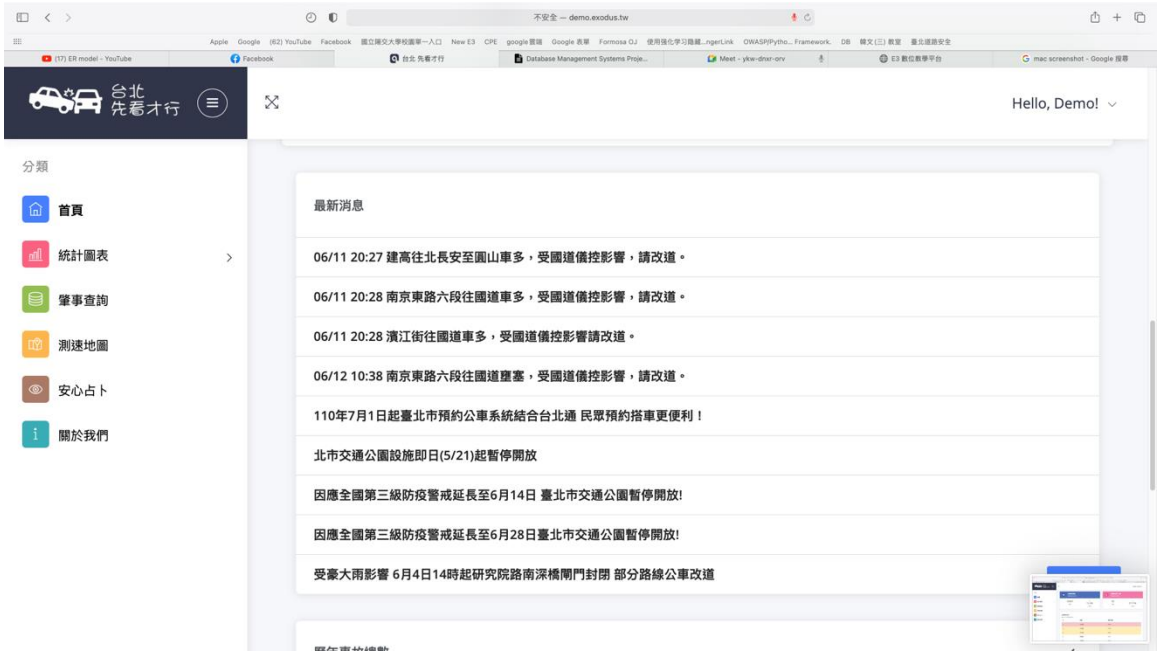


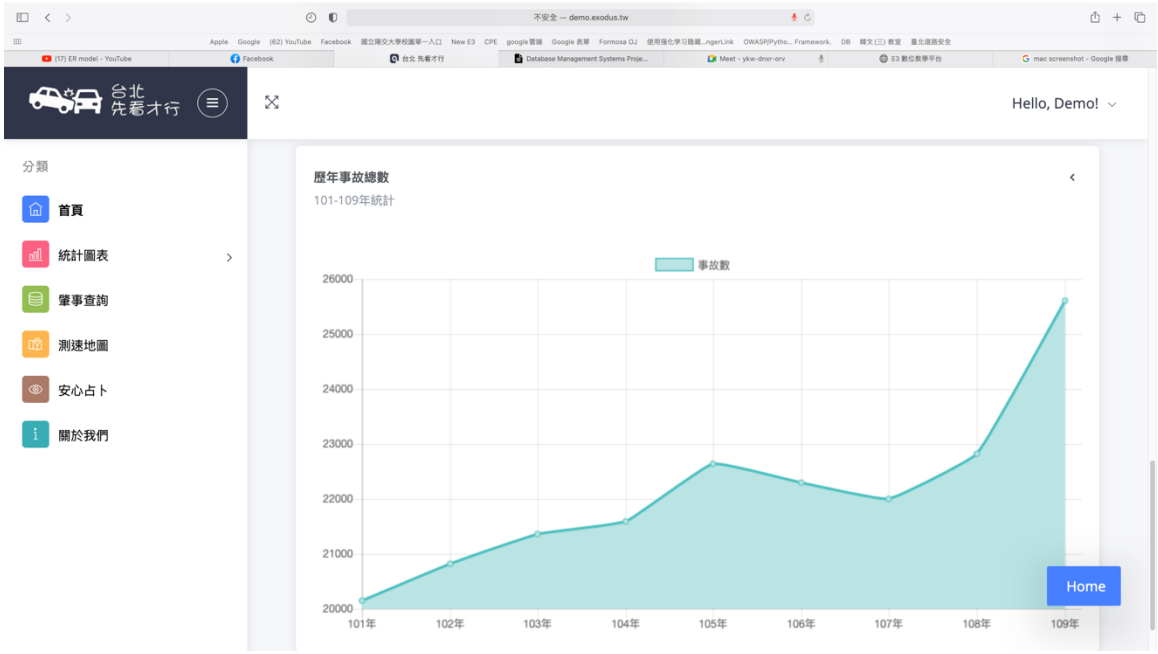
如上圖所示，使用者可以得知臺北市過去十年的交通事故資訊，另外我們也有分別以不同的類別做統計以及分析，這些使用者都可以透過網站知道。使用者還可以知道測速相機的位置以及我們已過去的事故資料來為使用者進行的安全預測（皆為台北市）。

5. 2 首頁

首頁的地方設計上方兩個小卡顯示數量與死亡人數的變化，接著列出區域交通事故的排名，再來是現在臺北市公布的最新消息，最後則是事故數量的統計折線圖。

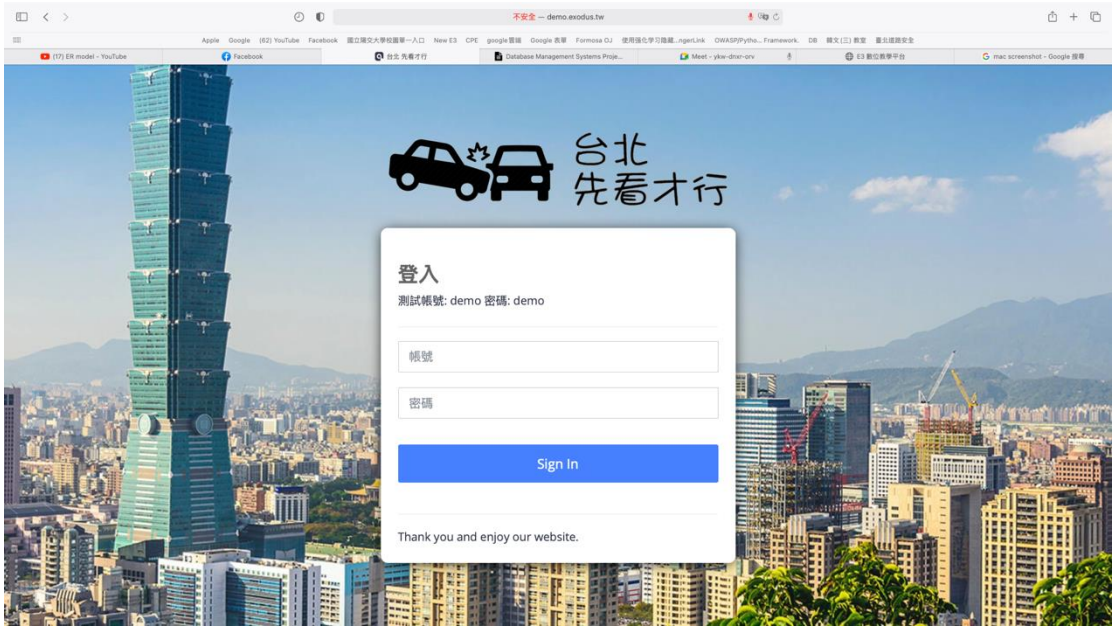






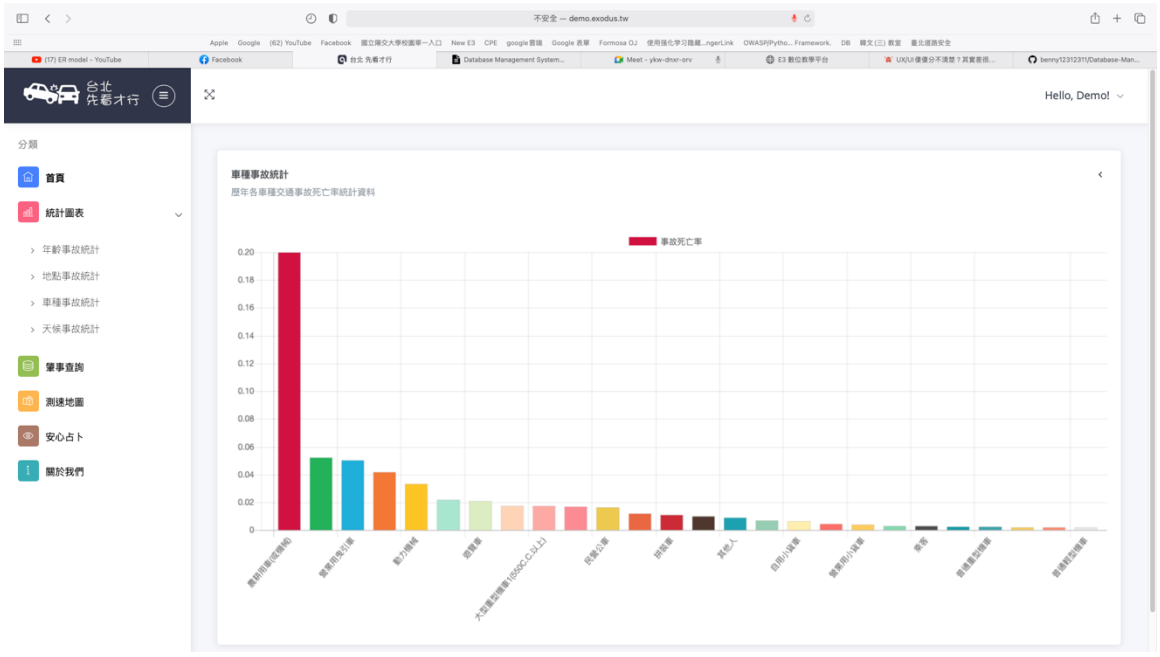
首頁會這樣設計是想讓大家點進來可以先看到一些整體數據的統計分析以及最新的消息。

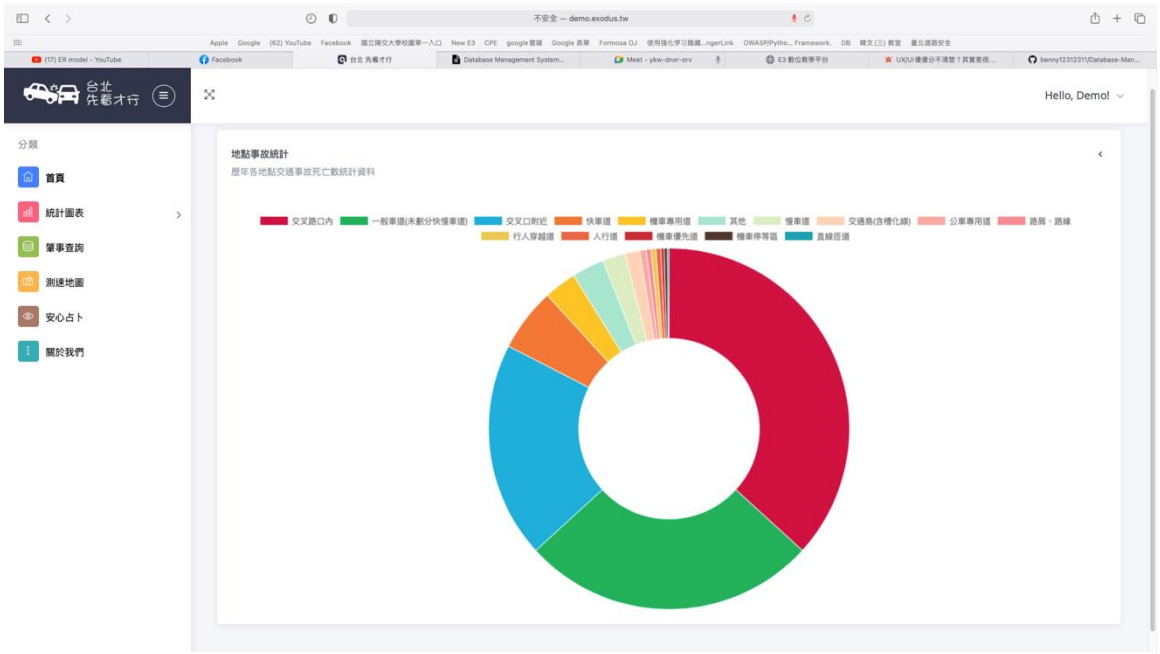
5.3 登入畫面



可以看到登入畫面採用城市圖當作背景，中間置放我們的 logo，還有輸入帳號密碼的欄位。

5. 4 統計圖表

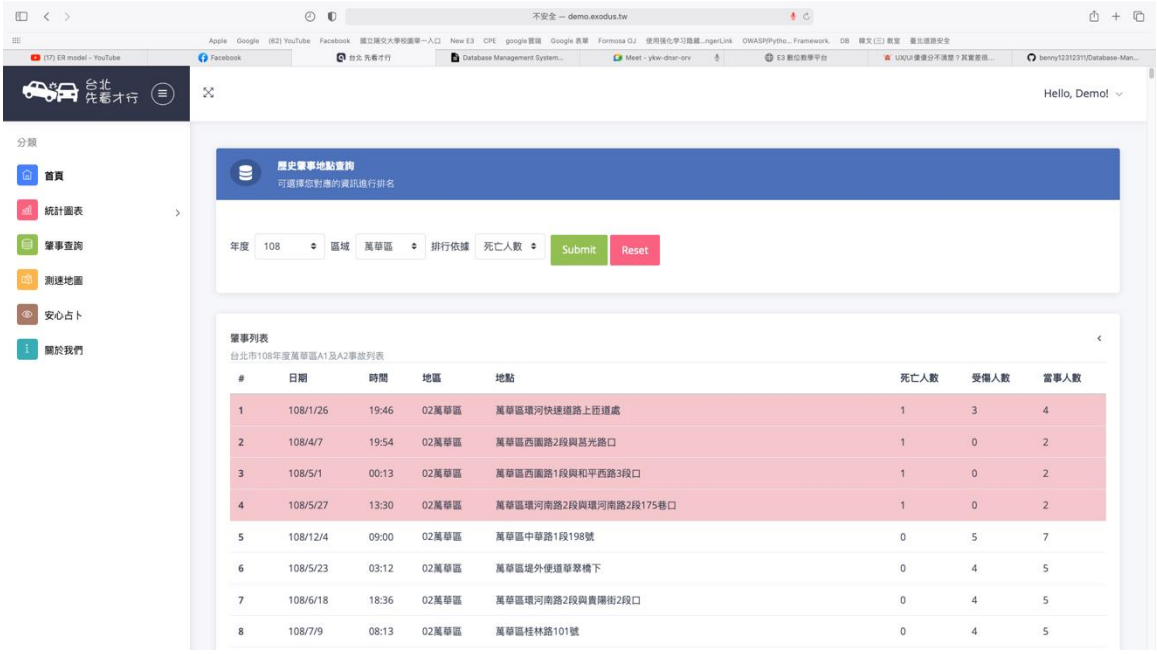




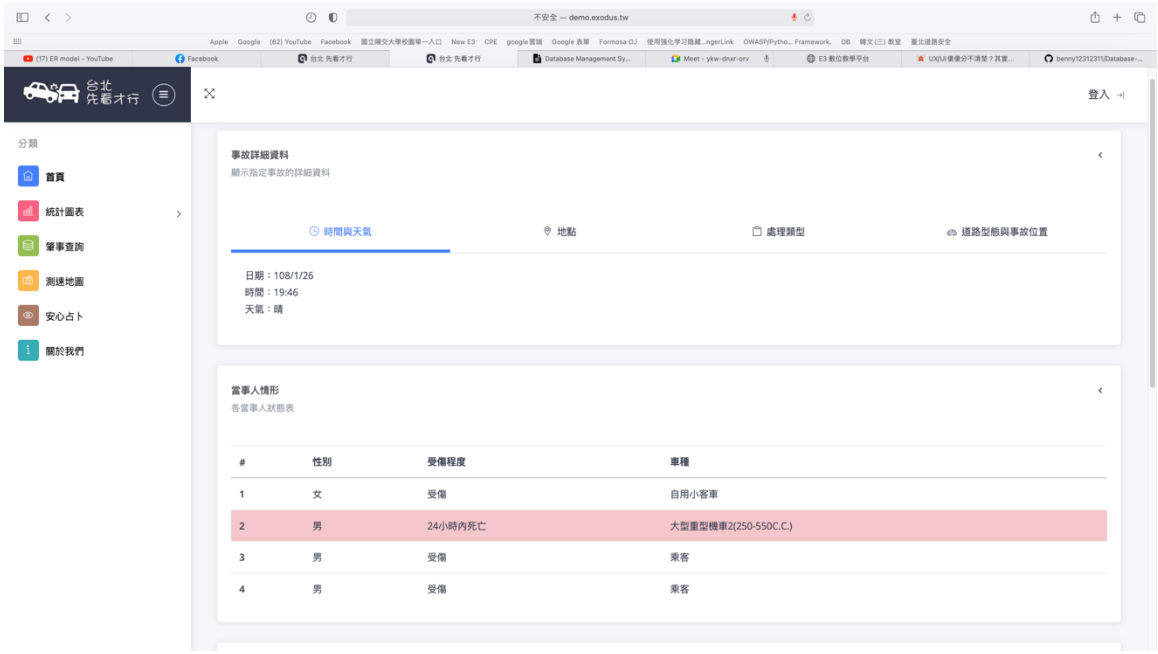
統計圖表的部分我們分成三種類別的統計圖（長條圖、圓餅圖與極坐標圖），每一種都分別以不同的統計方式與著色顯示，讓使用者在視覺上有更多元的體驗。

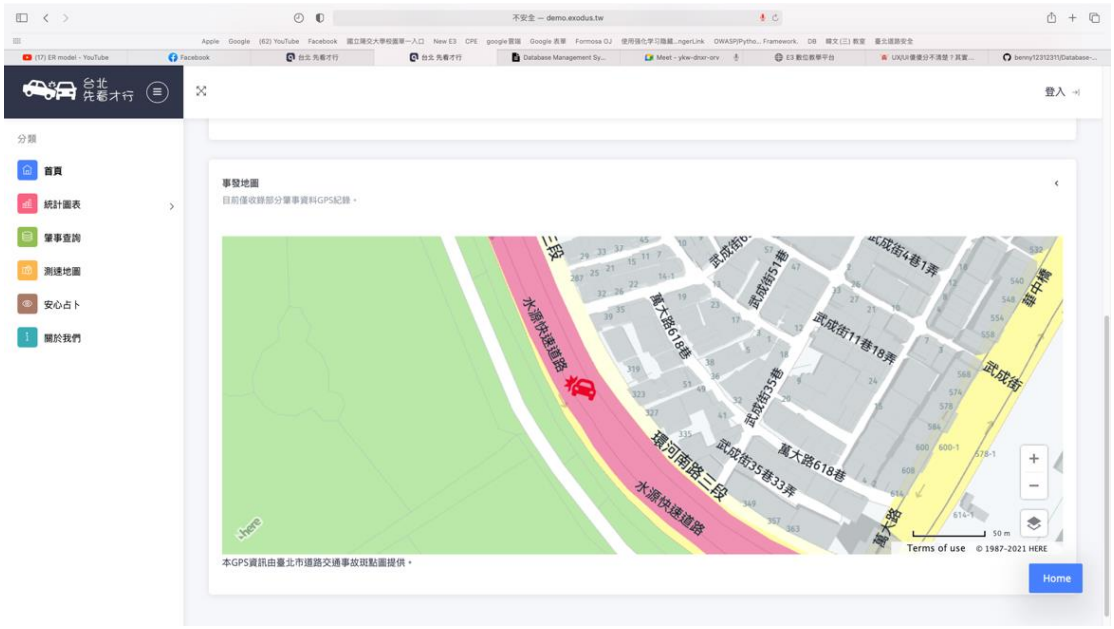
5.5 肇事查詢

肇事查詢的網頁設計為可以依照使用者想查詢的條件進行查詢，按鈕以綠色顯示「開始」與紅色顯示「重設」較符合人類的點選直覺。查詢送出後，會在下方以列表形式顯示事故資料，如果事故有造成死亡（死亡人數>=1）的就會以紅色為底色顯示。

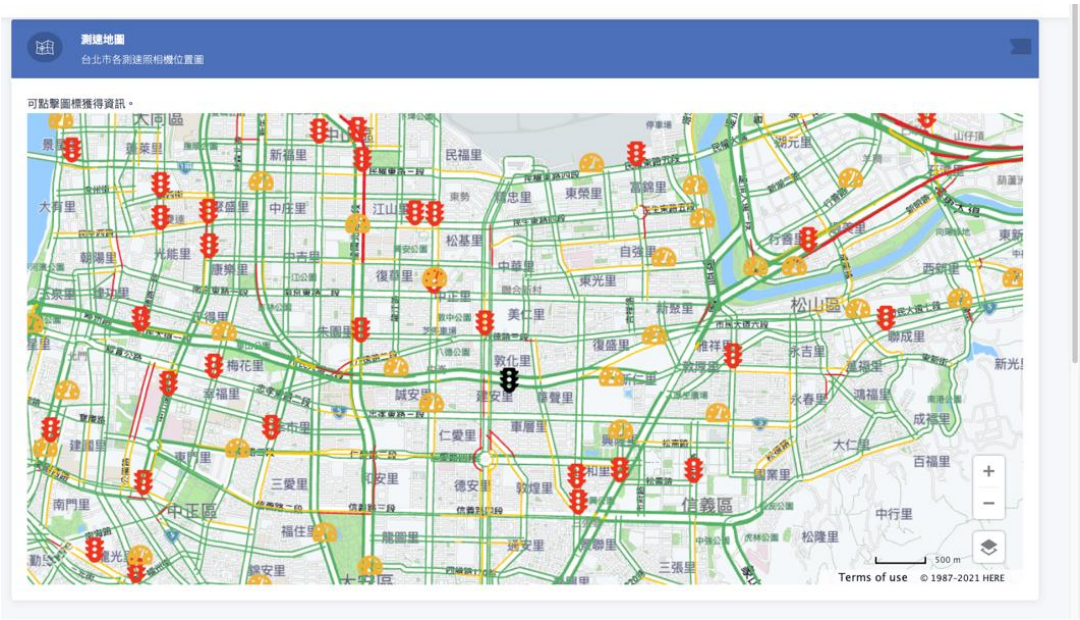


點進個別事故後會顯示該筆事故的詳細資訊，我們將時間、地點等文字顯示在上方，接者以表格的形式列出當事人，最下方則是地點的定位資訊（以地圖形式展現）。

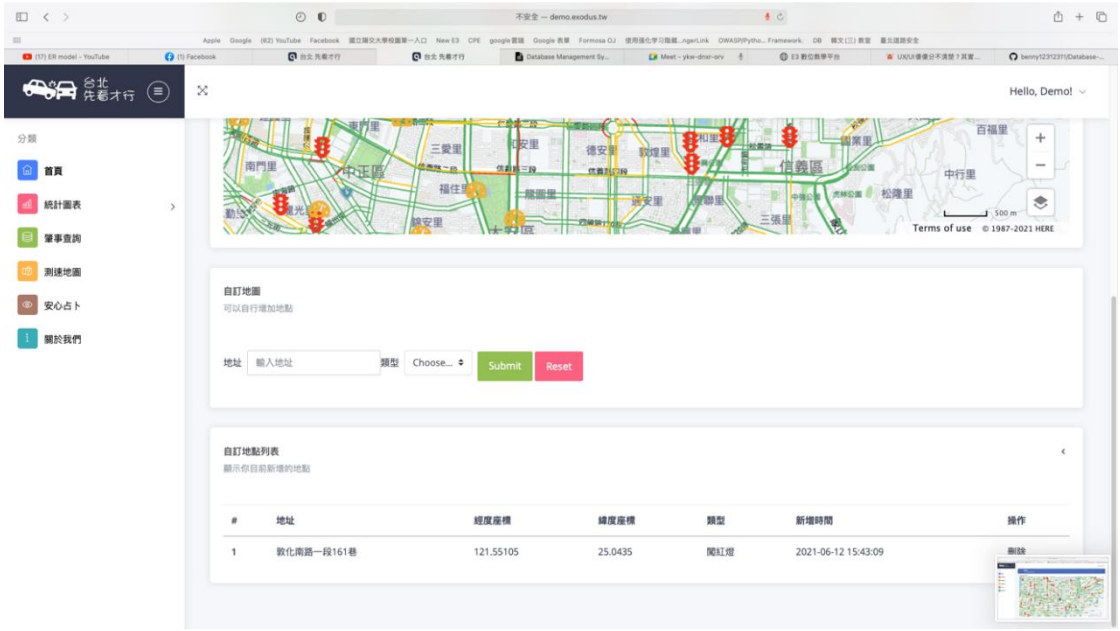

台北
先看才行
登入
分類
首頁
統計圖表
肇事查詢
測速地圖
安心占卜
關於我們
事故詳細資料
顯示指定事故的詳細資料
時間與天氣
地點
處理類型
道路型態與事故位置
日期：108/1/26
時間：19:46
天氣：晴
當事人情形
各當事人狀態表
性別 受傷程度 車種
1 女 受傷 自用小客車
2 男 24小時內死亡 大型重型機車2(250-550C.C.)
3 男 受傷 乘客
4 男 受傷 乘客


台北
先看才行
登入
分類
首頁
統計圖表
肇事查詢
測速地圖
安心占卜
關於我們
事發地圖
目前僅收錄部分肇事資料GPS紀錄。
水源快速道路
環河南路三段
萬大路618巷
武成街51巷
武成街11巷18弄
武成街35巷
武成街35巷33弄
武成街4巷1弄
武成街
Terms of use © 1987-2021 HERE
本GPS資訊由臺北市道路交通事故斑點圖提供。
Home

5.6 測速地圖



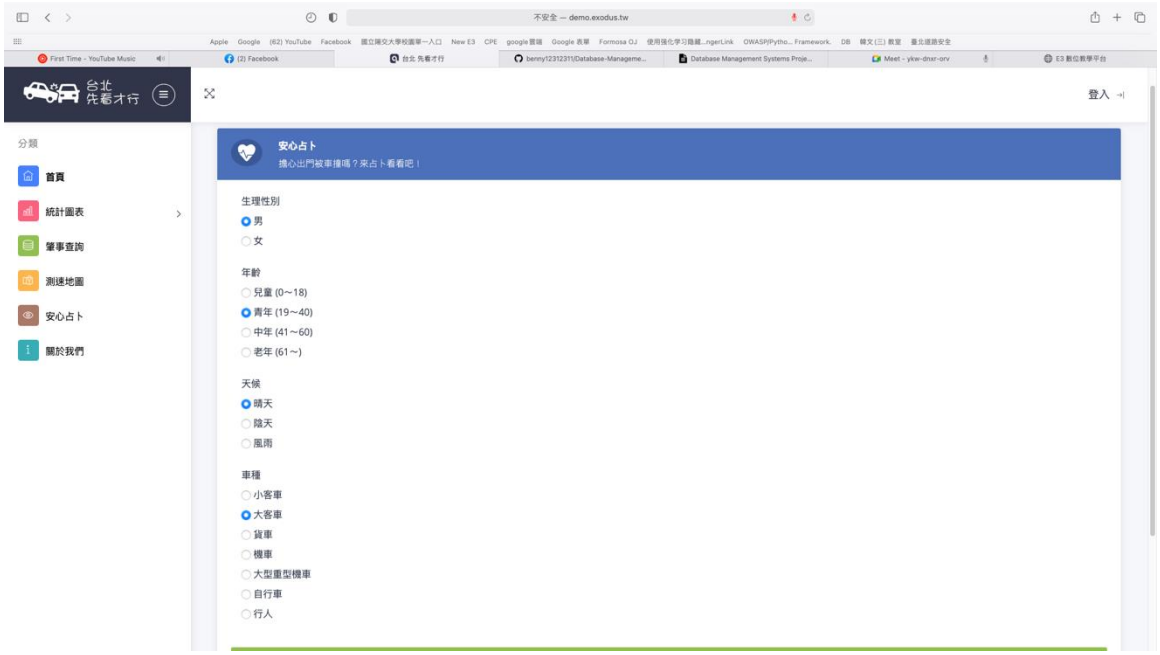
這個部分我們以紅色圖案顯示闖紅燈的照相、黃色顯示測速照相、粉色顯示區間測速照相，而黑色的則為使用者自行新增的（因圖上大多都是暖色調，所以以黑色區別），點擊這些圖案可顯示該地標的詳細資料。

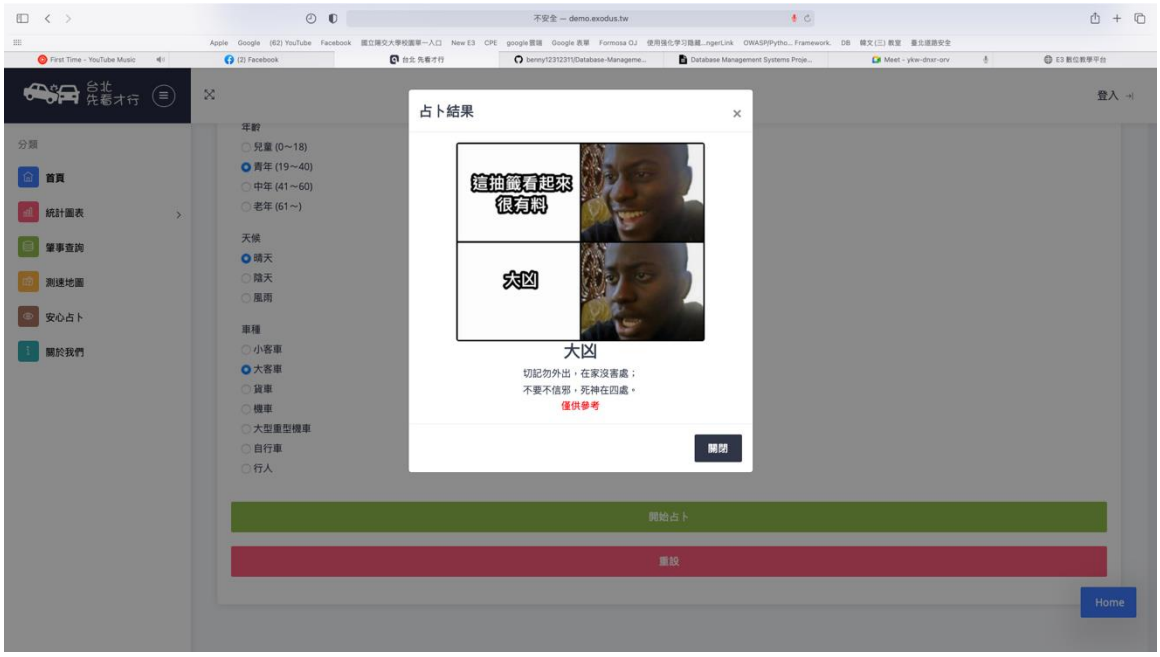


下方則是使用者新增資訊的地方以及新增過的資訊，這樣使用者使用上較為直覺。

5.7 安心占卜

這部分使用者以點選的方式進行條件的選擇，我們認為以單選按鈕（Radio Button）的方式顯示條件相較下拉選單方便且直覺，接著按下「開始占卜」後會顯示結果(結果分為五種:大吉、中吉、小吉、凶與大凶，依照事故死亡率判斷)，我們以互動視窗的方式顯示避免開新的分頁，並以迷因圖以及創意籤詩來說明占卜結果，讓使用上更有趣。





5.8 使用情境

1. 當使用者想要查詢過去十年的事故資訊以及統計資料
2. 使用者想要得知哪些地區的交通較危險
3. 使用者想要知道測速相機的位置
4. 使用者想要為出門做事故安全評估（占卜）或有行車保險需求

5.9 介面互動

使用者可以做登入登出的動作，以及查詢事故的資料，還有自行選擇選項篩選資料以進行排名，還可以看到事故的發生位置以及測速照相的位置，使用者也可以自行新增、更新以及刪除測速照相的資訊。

5.10 資料互動

所有由資料庫查詢出來的資料，我們不會直接顯示原始數據（Raw Data）在網頁上，而是經由篩選欄位、格式化並視覺化後，才以表格、圖表、地圖等方式呈現給使用者，這樣使用者就可以輕易瞭解他們想要的資訊，而不需要在令人眼花撩亂的資料中找到答案。

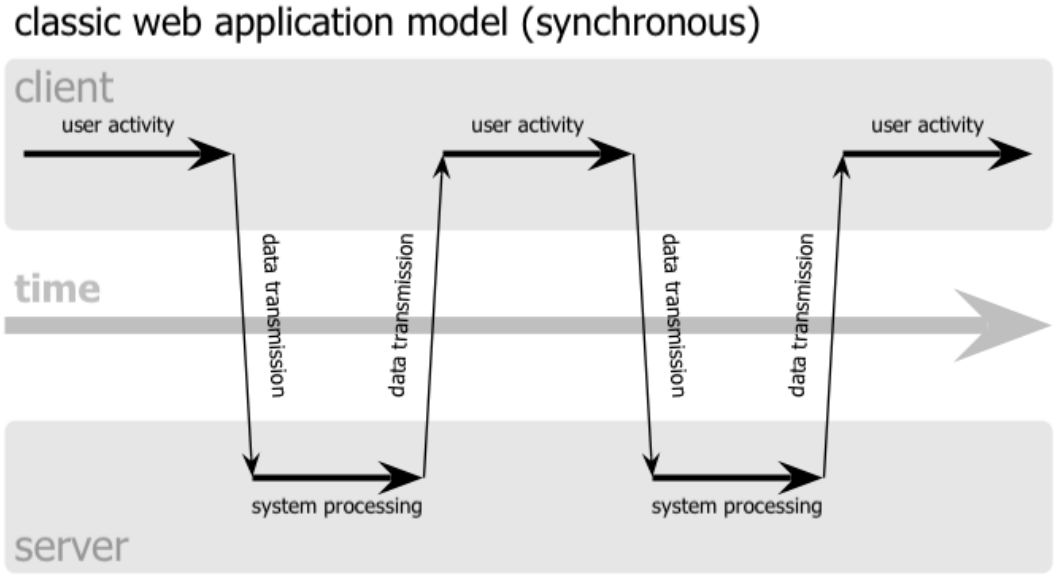
5.11 如何提升使用者體驗（UI/UX）

UI 部分：RWD 網頁設計

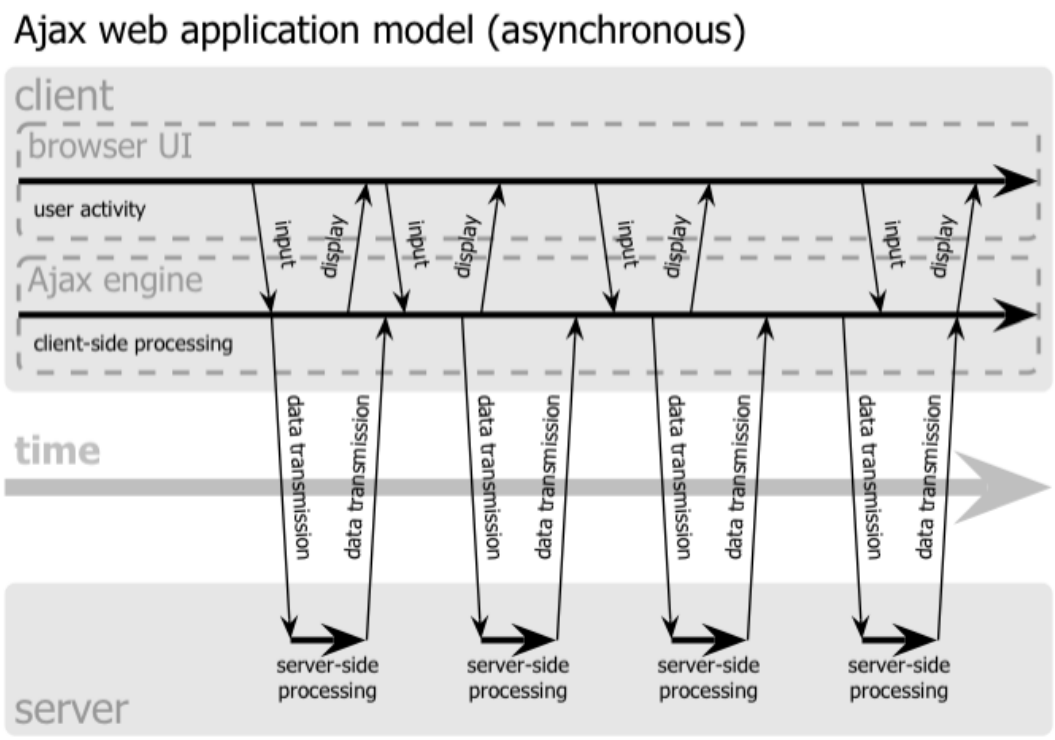
響應式網頁設計（Responsive Web Design）是一種網頁設計的技術，這種設計可使網站在不同的裝置（從桌面電腦顯示器到行動電話或其他行動裝置）上瀏覽時對應不同解析度皆有適合的呈現，減少使用者進行縮放、平移和捲動等操作行為。
我們這次採用的網頁模板即有 RWD 的功能，使使用者不論在任何裝置皆可以方便的瀏覽網頁，而不需要煩惱縮放問題。

UX 部分：AJAX 技術的應用

AJAX 即「Asynchronous JavaScript and XML」(非同步的 JavaScript 與 XML 技術)，指的是一套綜合了多項技術的瀏覽器端網頁開發技術。



上圖就是一般動態網頁客戶端(Client)與伺服器端(Server)傳輸資料的情形，可以發現資料傳輸的過程中，客戶端需要等待伺服器回傳資料才可以繼續瀏覽，這對於使用者來說，體驗一定會大打折扣。因此若我們採用 AJAX 技術，透過其非同步處理的性質，就可以實現流暢的網頁瀏覽體驗，如下圖：



可見中間加入 AJAX Engine 後，使用者可以在資料傳輸時繼續使用網頁應用程式，達到流暢的使用體驗。

我們盡可能的在網頁中實作 AJAX 技術，包含了首頁的最新消息（透過 API 即時獲得資料）、HERE Geocoder API 查詢經緯度以及安全占卜中的 SQL 查詢，都是透過 AJAX 技術達到無中斷的瀏覽體驗。

參考資料

回應式網頁設計 - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/%E5%93%8D%E5%BA%94%E5%BC%8F%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1
AJAX - 維基百科，自由的百科全書
https://zh.wikipedia.org/wiki/AJAX
JavaScript 101 快速入門教學
https://www.happycoder.org/2016/12/21/javascript101-tutorial/

6 Team Work

6.1 分工情形

姓名	貢獻	對應章節	貢獻度
李璨宇	1. PHP 後端設計 2. Here API 串接 3. 資料庫管理 4. JQuery/AJAX 設計	2.3 3.1 3.2 3.3 3.4 4.1 4.2 4.3 4.4 4.5 4.6 4.7 5.11	25%
張辰宇	1. 資料庫設計 2. SQL 查詢設計 3. ER Model 繪製	2.1 2.2 2.4 2.5 2.6	25%
黃則維	1. 前端 UI 設計 2. SQL 查詢設計 3. Logo 設計 4. Demo 影片剪輯	5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 5.10	25%
劉承遠	1. 資料庫設計 2. SQL 查詢設計 3. Google Map Geocoding API 4. 簡報設計	1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 1.10 1.11 1.12	25%

（依照姓名筆畫順序排序）

6.2 遭遇問題與解決方法

主題變更抉擇

一開始我們的主題所使用的資料，我們發現較難連結以及使用，再討論後我們決定要換一個主題，讓資料間可以更完整的互相連結以及運用。

SQL 資料型別的不一致

查詢時 cast 成相容的型別。

冗餘資料造成查詢效率低落

Table redundant data 過多，查詢資料過久，最後利用正規化來簡化 Table，提高查詢效率。

Google Geocoding API 無法查詢路口

Search API 沒辦法找 2 條路的路口，增加搜尋條件的判斷後解決。

第一次接觸網頁設計

不熟悉網站前端的設計（沒有接觸過），但藉由網路查了很多教學，就慢慢的將網站設計模板寫出來。

6.3 成果展示

Demo Site

http://demo.exodus.tw

原始碼

GitHub Repository
https://github.com/benny12312311/Database-Management-Systems-Project.git

關於重現性（Reproductivity）
網頁的內容皆已包含在原始碼中，除了內含 SQL 連接字串的 exec/sql.php 外，所有已用到的 PHP 檔皆已附上。
資料庫部分則是儲存在 resource/DATABASE_DBProject.zip，內含一個可以重建資料庫的.sql 檔，基本上只要建置好資料庫就可以重現這份 Project。

討論紀錄

HackMD
https://hackmd.io/uUZ_PujyTceiRg9qkRbrIQ?view