Introduction to Database Systems

Term Project: Report

0716214 江岳勳

0716222 黃偉傑

1. Data
   1. Introduction

　　我們的資料以公司資訊、職缺資訊、地域、職缺類別、使用者資訊和最愛組成。其中公司資訊、職缺資訊、地域和職缺類別，是我們從[1111人力銀行](https://www.1111.com.tw/)網站爬蟲下來所得。而使用者資訊和最愛，是在application運行時，由使用者執行相對應的操作之後，執行insert指令所得。

* 1. Description of the tables

　　首先以一個表格來呈現資料之於table的構成。

|  |  |  |  |
| --- | --- | --- | --- |
| Table | Columns | Primary Key | Description |
| jobinfo | **job\_id**, job\_name, degree, low\_salary, high\_salary, exp\_year, job\_type, worktime, is\_night, needed\_num | job\_id | 職缺資訊 |
| company | **com\_id**, com\_name, capital, emp\_number, addr | com\_id | 公司資訊 |
| position | **pos\_id**, pos\_field, pos\_name | pos\_id | 職缺的職務與  領域類別 |
| localarea | **area\_id**, area\_cc\_name, area\_td\_name, area\_cctd\_name | area\_id | 臺灣各行政區 |
| job | **job\_id**, com\_id, pos\_id, area\_id | job\_id | 職缺與公司、職缺類別、地域之間的關係 |
| user | **username**, passwd | username | 使用者資訊 |
| favorite | username, job\_id | N/A | 使用者加入最愛 |

　　以下每個表格分別代表著一個table，包含了column在MySQL內儲存的資訊與功能描述。

* **jobinfo**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| job\_id | int | NO | PRI | NULL | 職缺id |
| job\_name | varchar(100) | NO |  | NULL | 職缺名稱 |
| degree | tinyint | NO |  | NULL | 最低學歷要求，見下方說明 |
| low\_salary | int | YES |  | NULL | 薪資下限，面議時為4萬 |
| high\_salary | int | YES |  | NULL | 薪資上限，面議時為NULL |
| exp\_year | tinyint | NO |  | NULL | 最低年資要求 |
| job\_type | tinyint(1) | NO |  | NULL | 工作型態，見下方說明 |
| worktime | float | NO |  | NULL | 工作時間（單位為小時） |
| is\_night | tinyint | NO |  | NULL | 日／夜班，見下方說明 |
| needed\_num | tinyint | YES |  | NULL | 職缺需求人數 |

\*說明：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field | Explanation | | | | | |
| degree | 0（不拘） | 1（國中以下） | | 2（高中／職） | | 3（專科） |
| 4（大學） | | 5（碩士） | | 6（博士） | |
| job\_type | 0（正職） | | | 1（兼職） | | |
| is\_night | 1（日班） | | | 2（夜班） | | 3（皆有） |

* **company**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| com\_id | int | NO | PRI | NULL | 公司id |
| com\_name | varchar(100) | NO |  | NULL | 公司名稱 |
| capital | varchar(10) | YES |  | NULL | 資本額 |
| emp\_number | int | YES |  | NULL | 員工人數 |
| addr | varchar(100) | YES |  | NULL | 公司地址 |

* **position**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| pos\_id | int | NO | PRI | NULL | 職務id |
| pos\_field | varchar(8) | NO |  | NULL | 職務領域 |
| pos\_name | varchar(20) | NO |  | NULL | 職務名稱 |

\*說明：此處的資訊是求職網站為職缺所立的類別，例如「業務／業務助理」或「光電半導體／半導體工程師」等。

* **localarea**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| area\_id | smallint | NO | PRI | NULL | 地域id |
| area\_cc\_name | varchar(10) | NO |  | NULL | 縣市名稱 |
| area\_td\_name | varchar(10) | NO |  | NULL | 鄉鎮市區名稱 |
| area\_cctd\_name | varchar(10) | NO |  | NULL | 前兩者之連接 |

* **job**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| job\_id | int | NO | PRI | NULL | 職缺id |
| com\_id | int | NO |  | NULL | 公司id |
| pos\_id | int | NO |  | NULL | 職務id |
| area\_id | smallint | NO |  | NULL | 地域id |

* **user**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| username | varchar(15) | NO | PRI | NULL | 使用者名稱 |
| passwd | char(64) | NO |  | NULL | 雜湊過的使用者密碼 |

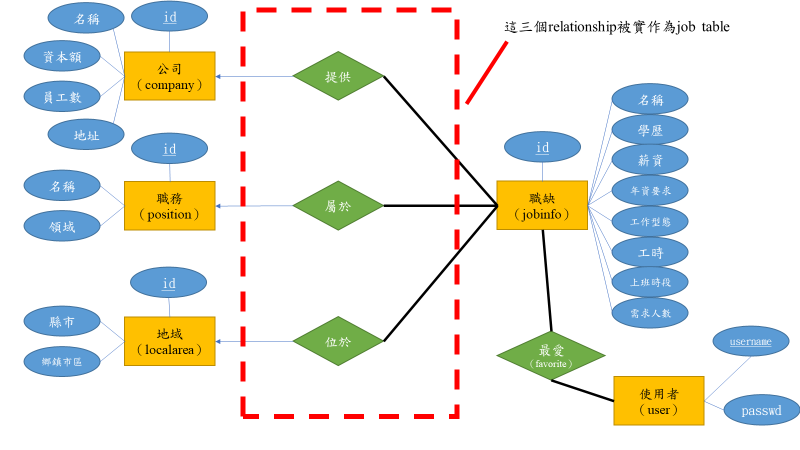
* **favorite**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Description |
| username | varchar(15) | NO |  | NULL | 使用者名稱 |
| job\_id | int | NO |  | NULL | 職缺id |

\*說明：此功能為使用者將職缺加入最愛，我們也會在功能說明的部分描述這項功能。

* 1. ER Models

　　下圖為我們的ER Model。其中，職缺（jobinfo）和公司（company）、職務（position）、地域（localarea）三者之間的關係是多對一，而職缺（jobinfo）和使用者（user）的「最愛（favorite）」關係是多對多。



1. Database
   1. Introduction / Maintenance

我們使用的資料庫是MySQL。

當我們要更新資料庫時，例如加入新的職缺資訊、公司資訊（陸續爬蟲下來），我們會利用update.sql（/src/sql/update.sql）來更新，其中的步驟包含：

1. 將job, jobinfo, company三個table truncate掉。
2. 從爬蟲程式的輸出資料夾(/src/fetch/1111/result/<timestamp>)載入新取得的資料。
3. 檢查favorite table，若其中包含著不存在於job的job\_id，代表這次的更新移除了部分使用者儲存過的職缺，這時會將這些資料從favorite table中移除。

除此之外，在運行階段時也會有動態的資料被加進資料庫中/從資料庫移除。這主要分為兩類：

1. 有新使用者註冊

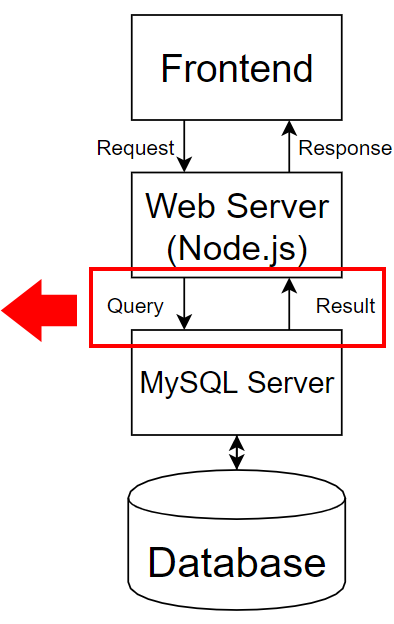
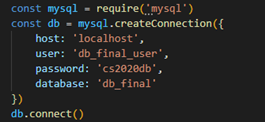
當有新使用者要註冊時，若使用者資料皆符合要求(皆為英數字元，且username不存在於user table中)，則會在user table上新增一筆(username, passwd)的資料，其中passwd為經過SHA-256加密過的64位的十六進位字串。

1. 使用者新增/移除了一筆最愛職缺

當使用者新增一筆最愛職缺時，會向favorite table增加一筆(username, job\_id)的資料；反之當使用者移除某筆最愛職缺時，符合(username, job\_id)的資料列則會從favorite table被移除。

* 1. Connection / Interaction

我們連接資料庫的方式是透過Node.js架設的後台在啟動時會require npm 中的mysql模組，建立一個資料庫的connection實例，之後的query便都經由此實例來與資料庫互動。



在前端呼叫後端存取資料庫時，前端會經由HTTP POST和Web Socket發送一條serialized成字串的JSON，包含著必要的資料(通常是job\_id等資訊)，後端收到前端的資料後會視情況(像是存取使用者最愛列表時)加入使用者目前session的資料，將資料渲染成相對應的SQL指令，並向MySQL的connection發出query請求，在結果回傳後會再將結果渲染成相應的HTML或JSON，並回傳給前端顯示。

至於Database exception handling的部分，當資料庫的query出現錯誤時，後端會停止這個頁面的渲染，在HTTP response中寫入error.html的內容並回傳給前端。

前端有很多操作會需要連結資料庫，以下會詳細說明 **(為了方便說明，從此以下到此小節結束前的檔案路徑都表示為斜體底線，並且檔案中的${}皆是執行時段會被替換成正確資料的標籤)**：

* 註冊

註冊時前端會將帳號及密碼POST到後端，後端會檢查資料庫中的user table有沒有包含username相同的資料列([*user/userExist.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/userExist.sql))，若沒有則向user table添加一條(username, passwd)的資料列([*user/newUser.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/newUser.sql))。

* 登入

登入時會向資料庫中檢查user table有沒有存在一條(username, passwd)皆符合的資料([*user/validateUserPassword.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/validateUserPassword.sql))，若有則向前端返回登入成功，並將username寫入session中。

* 首頁最愛/首頁推薦

首頁最愛：使用者載入首頁時，後端會將session中的username渲染進([*favorite/getUserFavInfo.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/getUserFavInfo.sql))中，讀取出所有在favorite table中username相同的資料列。

首頁推薦：和最愛一樣，載入首頁時，後端會將session中的username渲染進([*favorite/suggestionMain.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/suggestionMain.sql))中，將user儲存在favorite table的資料讀出來做出推薦列表。

* 簡易搜尋/進階搜尋

網頁會將搜尋內容(若是進階搜尋則會包含其他限制條件)透過Web Socket傳給後端，後端會將資校渲染進([*search/simpleSearch.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/simpleSearch.sql) **/** [*search/advanceSearch.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/advanceSearch.sql))中來對資料庫做搜尋。

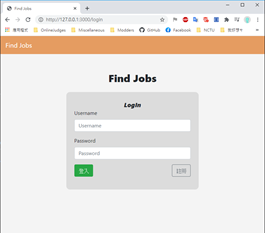
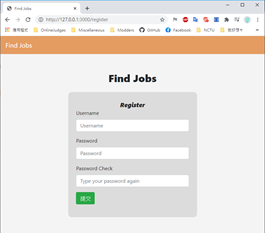
* 職缺頁面/公司頁面

頁面的job\_id/com\_id會被替換進([*job/singleJob.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/singleJob.sql) **/** [*company/compInfo.sql*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/company/compInfo.sql))中，取得資料庫中相對應的資料。

1. Application
   1. Interface

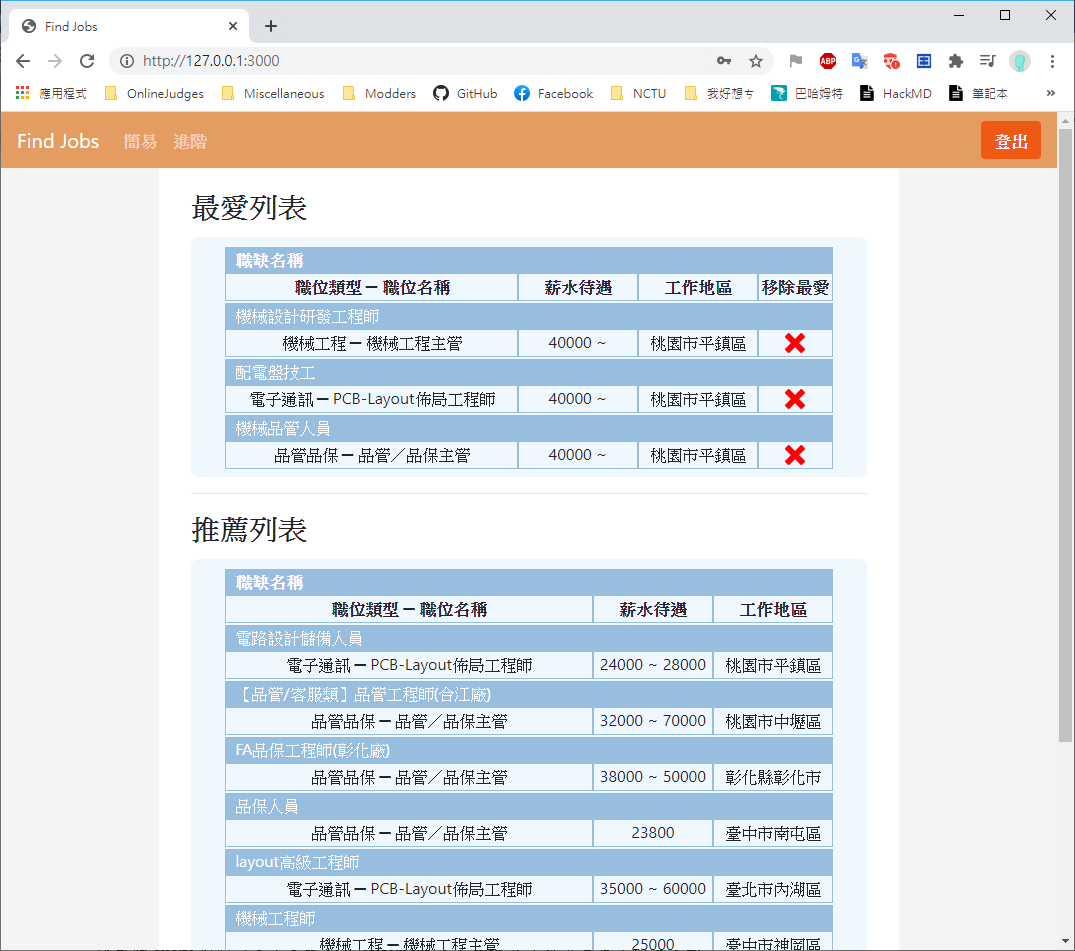
我們的專題使用Web Server架設了一個網站來呈現，我們的網站主要有幾個頁面：

* 登入/註冊頁面

上左圖為登入畫面，右圖為註冊畫面，使用時需要先登入帳號。

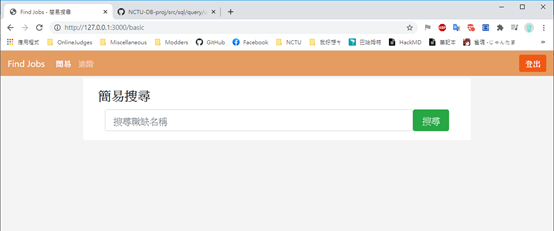
* 首頁

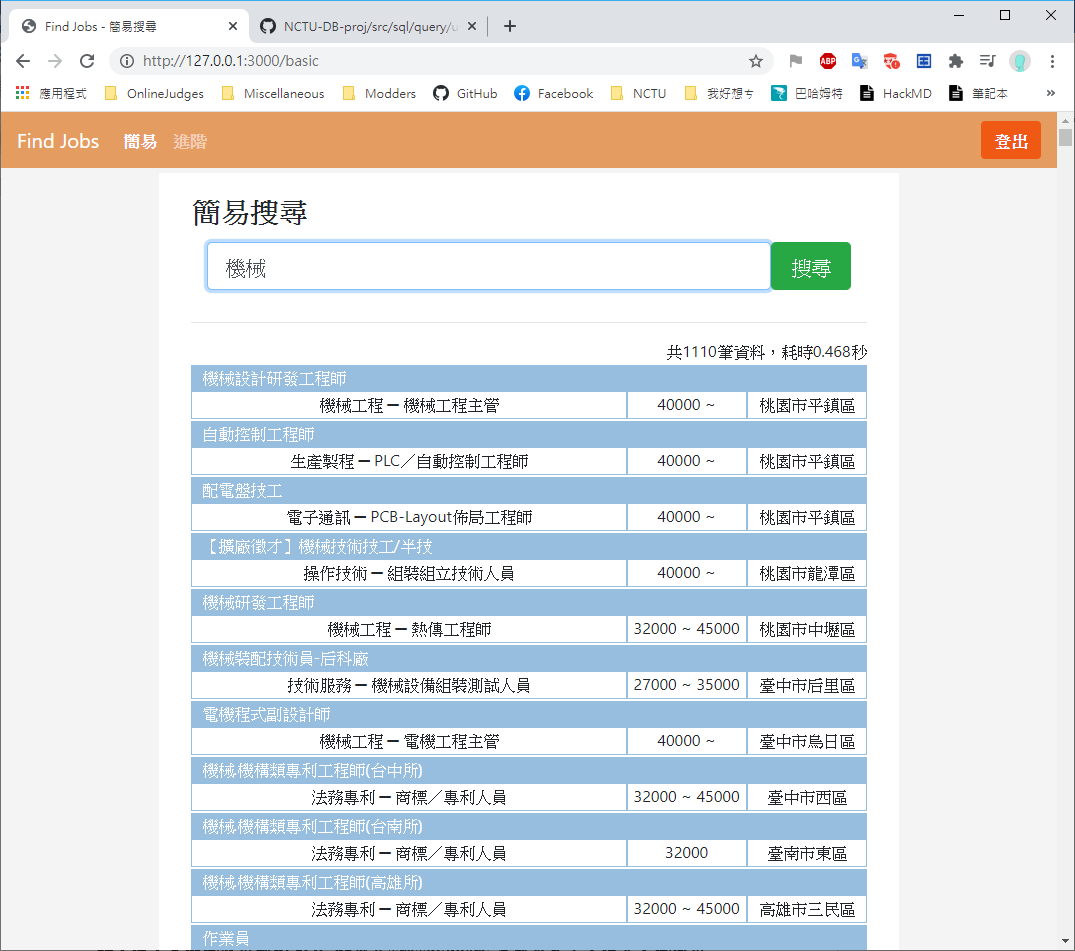


首頁會顯示使用者加入過最愛清單中的職缺，並會依據使用者的最愛清單在下方顯示一個推薦列表，推薦使用者也可能會感興趣的職缺。

* 簡易搜尋

在這個頁面中使用者可以透過搜尋關鍵字找到符合的職缺，對應的職缺資訊列表會在使用者按下搜尋後被顯示在網頁搜尋欄的下方。



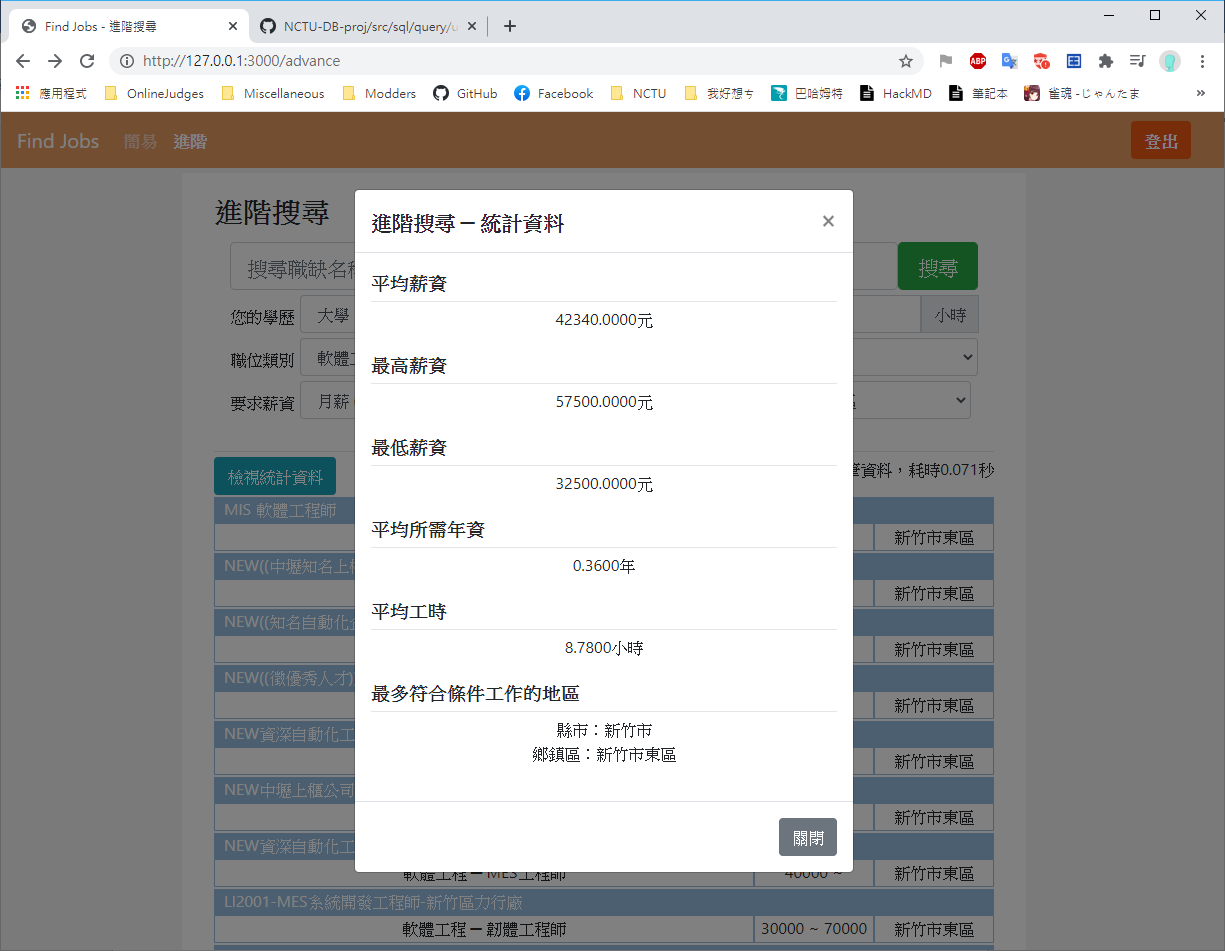


* 進階搜尋

在這個畫面中使用者可以設定除了關鍵字以外的限制，利用這些限制進一步的篩選符合自己條件及需求的工作



使用者也可以點選上圖搜尋結果左上方的「顯示統計資料」按鈕，會跳出一個小畫面顯示符合使用者設定條件的所有職缺的統計資料，如下圖所示。



* 職缺頁面

點選任何職缺資料的標題會開啟對應職缺的詳細資料畫面，七株包含了其工作時段即要求年資、學歷等資料，在下方還會推薦同個公司所徵求的職缺、同個職位的職缺及同地區的職缺讓使用者參考。



* 公司頁面

點選職缺頁面中的公司名稱則會開啟該公司的資訊頁面，除了顯示該公司的詳細資訊，還會顯示該公司所徵求中的所有職缺。



* 1. Functions & Implementations

我們的application的主要功能是查詢職缺及對職缺作統計分析，同時結合會員功能，使用者可以將中意的職缺加入自己的最愛中。以下我們將會條列各項功能，並解釋其運作方式及所用query的設計。我們使用的query，是將SQL statement直接寫在後台中作為字串使用([見此](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/web/htmlhelper.js#L400))，使用時其中的標籤(“${...}”符號)皆會被替換為對應的資料。

**為了解釋用途，我們將用到的SQL儲存在/src/sql/query資料夾底下，但並不會使用它們**（而且因為其中包含著等待被替換的標籤，該SQL無法被直接執行）。

1. **登入和註冊**

在使用任何功能之前，使用者必須先擁有一個帳號並且登入。我們的網頁伺服器會在使用者成功登入後在使用者的request session中寫入該使用者的username，以便之後query資料時會用到。在連線到除了登入和註冊以外的任何頁面時，使用者session中的username都不應為undefined，若是undefined，代表此連線尚未登入，則將其重導向登入頁面。

我們的登入功能會在使用者點擊登入按鈕時向後台POST表單的內容，後台會先檢查資料是否過長或包含非法字元(username長度限制15字元，username和passwd皆只能有英數字元)，若通過則會將passwd經過SHA-256加密後執行([*user/validateUserPassword*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/validateUserPassword.sql))，若回傳長度為1代表有對應的使用者，此時將username寫入session中並導向首頁，若長度為0則回傳包含錯誤訊息的登入畫面，要求重試。

註冊功能則是在使用者發送註冊資料時先檢查字元合法性(同上述)，檢查使用者名稱是否存在([*user/userExist*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/userExist.sql))，若不存在再使用雜湊過的密碼執行([*user/newUser*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/newUser.sql))，將使用者名稱和加密過的密碼存入user table中，再將username寫入session中並導回首頁。

**使用的SQL：**

[*user/validateUserPassword*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/validateUserPassword.sql)：SELECT出user table中username和passwd皆符合條件的資料列，若列數大於0代表有相對應的使用者。

使用標籤：${username}：使用者名稱, ${passwd}：加密過的密碼

[*user/userExist*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/userExist.sql)：SELECT出user table中username跟標籤相等的資料列，若列數大於0代表使用者名稱已被使用。

使用標籤：${username}：使用者名稱

[*user/newUser*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/user/newUser.sql)：將一列新的(username, passwd) INSERT INTO user table。

使用標籤：${username}：使用者名稱, ${passwd}：加密過的密碼



1. **首頁最愛/推薦**

使用者在首頁可以看到他們曾經加入過最愛的職缺列表，這個功能是透過在使用者載入首頁時，伺服器會使用session中儲存的username執行([*favorite/getUserFavInfo*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/getUserFavInfo.sql))，取得使用者的最愛列表，並對每一筆最愛生成一個HTML元素顯示在首頁的最愛列表位置中。

首頁的推薦列表則是透過執行([*favorite/suggestionMain*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/suggestionMain.sql))，統計使用者最愛清單中的前三多職位，並從所有職缺中選出職位在前三名中，且本身不在最愛清單中，隨機排序後選出前十個，在如上述生成HTML元素插入相對應位置。

在最愛清單中的職缺右下角會有一個刪除按鈕，按下後會生成一個包含該job\_id的HTTP POST至/delfav，伺服器收到請求後會執行([*favorite/favDel*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/favDel.sql))將favorite table中相對應的資料刪除。

**使用的SQL：**

[*favorite/getUserFavInfo*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/getUserFavInfo.sql)：SELECT出favorite table中username和標籤相符的資料列，這些便是此使用者加入最愛的職缺。

使用標籤：${username}：使用者名稱

[*favorite/suggestionMain*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/suggestionMain.sql)：先SELECT出favorite table中username和標籤相符的資料列，和job table join起來後選擇favorite.job\_id=job.job\_id的資料列，並GROUP BY job.pos\_id，用COUNT反向排序即可得到前三名的職位。再從所有job中SELECT出pos\_id在前述query中且job\_id不在該使用者最愛中的職缺，隨機排序後取前10列。和localarea, position, jobinfo等table join後可瞿德其他資訊。

使用標籤：${username}：使用者名稱

[*favorite/favDel*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/favDel.sql)：將favorite table中username和job\_id皆符合的資料列刪除。

使用標籤：${username}：使用者名稱, ${job\_id}：欲刪除的職缺id

1. **簡易查詢功能**

我們的查詢功能可以區分成簡易查詢與進階查詢。簡易查詢為了追求方便性，只有關鍵字搜尋，在使用者輸入關鍵字之後，網頁會透過Web Socket發送搜尋資料給後端，後端會用關鍵字執行([*search/simpleSearch*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/simpleSearch.sql))，再對每一列搜尋結果生成相對應的HTML，全部concat在一起後透過Web Socket送回前端。前端會在將搜尋結果放在正確的位置上，並顯示資料比數和消耗時間。

**使用的SQL：**

[*search/simpleSearch*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/simpleSearch.sql)：分別從jobinfo和company中SELECT出job\_name和com\_name LIKE “%${search}%”的資料，跟job join在一起後再和其他的table join取得詳細資料。

使用標籤：${search}：搜尋關鍵字

1. **進階查詢功能**

在進階查詢下，使用者可以針對職缺資訊，例如前述的關鍵字、學歷、年資、工作型態、工時、職位類別、薪資、地區做設定。

在按下搜尋時，前端會將所有的設定值透過Web Socket傳送給後端，後端則會用這些值來執行([*search/advanceSearch*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/advanceSearch.sql))，再對每一列搜尋結果生成相對應的HTML，全部concat在一起後透過Web Socket送回前端。前端會在將搜尋結果放在正確的位置上，並顯示資料比數和消耗時間。

在使用者未給條件賦值的情況下，我們也會給定合理的預設值。以下我們會針對各項資訊提供說明。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 條件名稱 | 預設值 | 輸入 | 描述 | 條件標籤 |
| 關鍵字 | 不拘 | 文字 | 包含職缺或公司名稱之**子字串** | search |
| 學歷 | 國中 | 選單 | 學歷需**高於或等於**職缺所需學歷 | degree |
| 年資 | 0 | 數字 | 年資需**高於或等於**職缺所需年資 | exp\_year |
| 日／夜班 | 日班 | 選單 | 輸入需包含職缺工作時段 | is\_night |
| 工時 | 0 | 數字 | 職缺工時須**小於或等於**輸入工時 | worktime |
| 職位類別 | 不拘 | 選單 | 輸入需與職缺職位類別相符 | pos\_field, pos\_name |
| 工作型態 | 全職 | 選單 | 輸入需與職缺工作型態相符 | job\_type |
| 薪資 | 0 | 數字 | 薪資上限**高於或等於**預期薪資 | salary |
| 地區 | 不拘 | 選單 | 輸入需與職缺所在地區相符 | area\_cc\_name, area\_cctd\_name |

上面的query會先被buffer起來，統計分析會使用到這條query。

在進階搜尋的清單回傳給前端後，後端會繼續執行統計資料的分析，前面buffer起來的query string會被當作統計query的subquery來使用。在統計分析完成後，會發送一個標記著統計資料類別的json給前端，其中包含著統計出的數值。前端會再將這些數值顯示在統計畫面上。

**使用的SQL：**

[*search/advanceSearch*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/search/advanceSearch.sql)：分別從jobinfo和company中SELECT出job\_name和com\_name LIKE “%${search}%”的資料，跟job join在一起後再和其他的table join取得詳細資料，join時會在加上所設定的其他constraint，做到進階搜尋的效果。

使用標籤：上列表格所列的所有條件標籤

[*statistics/statisticAreaCC*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/statistics/statisticAreaCC.sql)：將搜尋結果GROUP BY area\_cc\_name後排序選出現最多的縣市。

使用標籤：${base}：進階搜尋時buffer起來的query string

[*statistics/statisticAreaCCTD*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/statistics/statisticAreaCCTD.sql)：將搜尋結果GROUP BY area\_cctd\_name後排序選出現最多的縣市鄉鎮區。

使用標籤：${base}：進階搜尋時buffer起來的query string

[*statistics/statisticSalary*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/statistics/statisticSalary.sql)：從搜尋結果中算出MAX(\_salary), MIN(\_salary), AVG(\_salary)。

使用標籤：${base}：進階搜尋時buffer起來的query string

[*statistics/statisticsExpyear*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/statistics/statisticsExpyear.sql)：從搜尋結果中算出AVG(exp\_year)。

使用標籤：${base}：進階搜尋時buffer起來的query string

1. **職缺頁面**

在使用者載入一個職缺頁面(/job/job\_id)時，伺服器會將job\_id放進([*job/singleJob*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/singleJob.sql))中向資料庫query該職缺的詳細資料，在得到資料後會將結果放進job.html中對應的欄位裡。職缺資訊的右下角會根據用戶有沒有將這筆職缺加入最愛來決定要顯示「加入最愛」還是「移除最愛」，這個是透過執行([*favorite/isInUserFav*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/isInUserFav.sql))來檢查的。

當使用者按下加入最愛時，會生成一個包含該job\_id的HTTP POST至/addfav，伺服器收到請求後會執行([*favorite/favAdd*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/favAdd.sql))將(username, job\_id)加入favorite table中。按下移除最愛時的行為和上面介紹首頁時按下最愛刪除的行為一樣。

在職缺頁面的下方則會顯示一些和此職缺有關的其他職缺，包含：

* 同公司的其他職缺

執行([*job/suggestionComp*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionComp.sql))，用回傳結果生成HTML元素插入網頁中。

* 同職位的其他職缺

執行([*job/suggestionPos*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionPos.sql))，用回傳結果生成HTML元素插入網頁中。

* 同地區的其他職缺

執行([*job/suggestionArea*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionArea.sql))，用回傳結果生成HTML元素插入網頁中。

**使用的SQL：**

[*job/singleJob*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/singleJob.sql)：從job中SELECT 出對應的資料列，再跟jobinfo, company, localarea, position join後取得其他詳細資料。

使用標籤：${job\_id}：職缺id

[*favorite/isInUserFav*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/isInUserFav.sql)：SELECT favorite table中username和job\_id皆等於標籤的資料列，若不為空代表此職缺已被此使用者加入最愛。

使用標籤：${username}：使用者名稱, ${job\_id}：職缺id

[*favorite/favAdd*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/favAdd.sql)：將(username, job\_id) INSERT INTO favorite table。

使用標籤：${username}：使用者名稱, ${job\_id}：欲新增的職缺id

[*favorite/favDel*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/favorite/favDel.sql)：將favorite table中username和job\_id皆符合的資料列刪除。

使用標籤：${username}：使用者名稱, ${job\_id}：欲刪除的職缺id

[*job/suggestionComp*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionComp.sql)：從job中SELECT出com\_id等於標籤的資料，再和其他table join後取得其他資訊。

使用標籤：${com\_id}：公司id

[*job/suggestionPos*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionPos.sql)：從job中SELECT出pos\_id等於標籤的資料，再和其他table join後取得其他資訊。

使用標籤：${pos\_id}：職位id

[*job/suggestionArea*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/job/suggestionArea.sql)：從job中SELECT出area\_id等於標籤的資料，再和其他table join後取得其他資訊。

使用標籤：${area\_id}：地區id

1. **公司頁面**

在使用者載入一個職缺頁面(/comp/com\_id)時，伺服器會將com\_id放進([*company/compInfo*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/company/compInfo.sql))中向資料庫query該公司的詳細資料，在得到資料後會將結果放進comp.html中對應的欄位裡。

在公司頁面的下方會透過執行([*company/compJobs*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/company/compJobs.sql))取得這間公司徵求中的所有職缺並顯示在對應位置。

**使用的SQL：**

[*company/compInfo*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/company/compInfo.sql)：從company中SELECT com\_id等於標籤的資料

使用標籤：${com\_id}：公司id

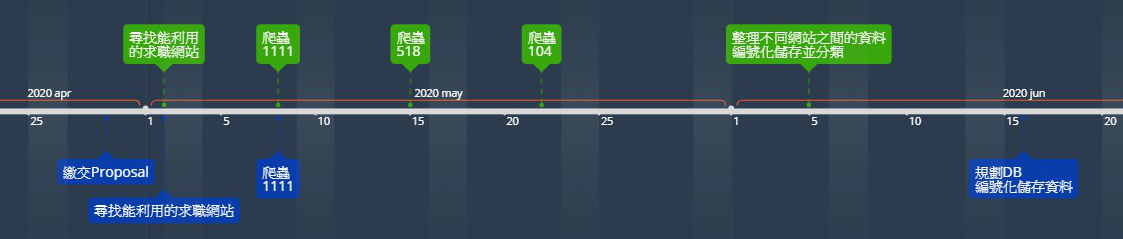
[*company/compJobs*](https://github.com/hm-ysjiang/NCTU-DB-proj/blob/master/src/sql/query/company/compJobs.sql)：從job中SELECT出com\_id等於標籤的資料，再和其他table join後取得其他資訊。

使用標籤：${com\_id}：公司id

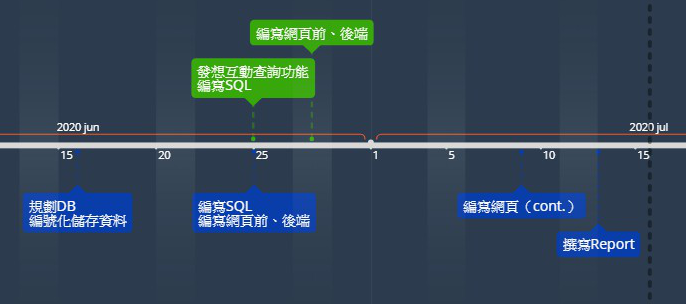
1. Exception Handling

在例外處理的方面，所有從前端傳回後端的資料都會經過檢查，確認資料合法性，像是數字欄位的內容必須要為數字(match “^[0-9]+$”)，使用者名稱和密碼都只能為英數等。我們也有幫無輸入的欄位設立預設值(如前3.2.4節表格內容所示)，因此不會有application造成的例外發生。

1. Others
   1. Progress

 上方為預期進度，下方為實際進度

(接續下頁)



* 1. Problems Encountered
* 爬蟲時發現目標網站會根據使用者行為來決定要更新什麼資訊，並不是一個靜態的網頁，所以我們改使用selenium套件來執行爬蟲，但用selenium就出現了執行速度慢的問題，因此我們不得不限縮資料的大小，我們拍攝demo影片時資料庫的資料量大約13萬筆，在執行爬蟲時便花了近6小時才將資料下載完成。未來使用時可能要使用分散式的爬蟲並讓爬蟲常駐在後台來避免更新耗時過久以及同一個IP位置向網站發出過多請求而被判定為惡意行為的問題。
* 從4.1節的進度圖表跟前面提到的爬蟲問題可以發現，我們在爬蟲上碰到的問題比當初想像的還要多很多，加上爬蟲耗時過久，發現問題到修正後重新執行並檢查問題是否解決，一個循環通常要花超過半天的時間，造成我們成功載下1111人力銀行的資料後發現進度比預期的落後非常多，我們判斷時間上並不允許我們再使用時間載下另一個求職網站的資料，必須趕快開始實作資料庫功能和使用者介面才行，於是我們放棄了爬取其他求職網站上的資料，轉為只分析1111人力銀行的資料，並利用最後多出來的一點時間實作了原本計畫外的功能作為彌補 (最愛功能、推薦職缺功能)。
* 我們在設計SQL指令時，因為在設計網頁時全部的連線都會共用同一個資料庫，所以不能有temporary table等暫存出現，不然在執行時會沒有辦法分別這個temporary table是哪一個連線在何時CREATE的。因此我們在設計進階搜尋時煩惱了許久，因為進階搜尋的統計功能需要重複的從一份職缺清單中做query，考量到大部分的痛季都需要使用到GROUP BY等指令，會無可避免的遺失掉不少原本清單的資訊，所以沒辦法在一個query就算出所有的統計量。但若是一個一個算又會花太久時間，影響使用者瀏覽網頁的體驗，所以後來我們將其改成query出清單後會先回傳結果給前端，之後在使用原本的query string當作統計query的subquery，算出統計量後分批回傳。這樣子唯一的影響只有在搜尋結果剛顯示時統計視窗內會顯示「統計中…」的字樣，大約過1~2後便會顯示出分批回傳的數值，不會影響到使用者操作。

|  |  |
| --- | --- |
| 0716214 江岳勳 | 0716222 黃偉傑 |
| 爬蟲設計  架設伺服器、撰寫網頁  錄製demo影片  設計SQL─登入註冊  設計SQL─最愛列表  設計SQL─職缺推薦  設計SQL─職缺、公司資訊 | 專案進度管理  設計資料庫模型  為demo影片上字卡  設計SQL─簡易搜尋  設計SQL─進階搜尋  設計SQL─統計分析 |

* 1. Contributions
  2. Link 2 Repo

<https://github.com/hm-ysjiang/NCTU-DB-proj>

* 1. Link 2 Discussion Channel

<https://trello.com/b/1isqFHkH/db-proj-discussion>