Stage-3 AlumniBook Group-3

郁昀彥 R00725051　郭瀚智 R02725023　鄭立民 R02725041　李奕德 B99705021
張凱涵 B00705027　施淮振 B00705047　倪嘉銘 T02705102

Contents

1 專案背景及目的

本系在發展茁壯的同時培養了許多傑出的人才，這些人散布在各個不同領域。在超級專業化的現代若能連結正確領域的人來提供系上一些建議與幫助，必能讓資管系所內的課程內容與訓練更加的紮實。所以除了學生間的相互聯絡，本系統也將會提供系友與教授及行政人員進入本系統，對其所擅長之專業領域提供協助。學生亦可藉由本系統盡早深入了解產業現況，獲得更多正確且即時的資訊並對未來發展方向有更確實精確的判斷。

為了確保使用者身分得正確性，本系統必須要求使用者經由計算機中心 (或其他認證機構) 所提供的身分認證服務來認證使用者和其所宣稱的人為同一個人以防冒用。經正確認證之使用者於進入本系統後，會來到共用之瀏覽區。在這個區塊可以看到目前系統中的所有議題，瀏覽並參與討論。在觀看議題的同時使用者也可針對特定議題發表評論，此評論將會由任何觀看此議題的人共同看見。

在討論區中的所有議題都會根據其在發文時選擇的標籤做分類，方便其他使用者閱讀。使用者可以自由的訂閱包含特定標籤的議題，系統會再此類別議題建立時發信至使用者註冊時所使用得系信箱帳戶，讓使用者可以更密切的關柱該類別的議題。若只是關注特定議題之後續發展，亦可選擇只訂閱該議題。系統將依照使用者要求在此議題及其評論更新或新增時寄信至用戶信箱。另外，留言也會自動讓系統將該議題加入訂閱列表。

若在閱讀討論區時對特定人士產生興趣，也可以在適當的隱私權相關管理下查詢特定的系友的相關資訊。能否能夠看見該使用者所有相關資料取決於資料擁有者的個人隱私條件以及搜尋者之帳號類別，或是搜尋者與被搜尋者之間有在本系統中建立特別的朋友關係。本系統中若存在朋友關係則再搜尋時會由特定對象獨立於依照帳號類別做設計的隱私規則，直接套用該使用者針對朋友設定的隱私權相關設定，讓朋友間能夠更加深入的了解彼此。在好友關係尚未建立的情況下，使用者依然可以追蹤其他使用者，系統將會在該用戶發布新的議題時告知訂閱者這項訊息。

除了瀏覽討論區外，使用者亦可編輯自身的相關資料與隱私相關選項來讓自己被更多人看見。本系統將提供預先填入之公開資訊、基礎的個人連絡資訊輸入、以及與職業生涯相關之記錄，三個主要區塊。使用者可依照自身之習慣與想法選擇性的填入資料與決定其公開性。預設的群組別為校友、學生、教授及行政人員、與院系人員及朋友幾種，未來可再依據使用者需求做調整。

本系統旨在於建立一個能夠有效凝聚資管系所內之相關人士，提供一個跨界跨部 (研究所與大學部) 之溝通平台。本系統將階段性提供完整的溝通整合服務，冀望成為連結具專業知識的人幫忙解答疑惑，也希望能夠成為校友找人、學生找事之合作橋梁。

2 專案假設與限制

為使本計畫順暢運行，其執行環境必須具有以下幾點之限制，其中缺乏任一一項將造成時程延宕或效果不如預期等等缺陷。

- [身分認證] 關於身分認證部分，台大計算機中心必須能夠提供 single sign-on 之服務，並且能夠回應本系統所有身分認證之需求。同時，回傳之 token 必須包含學號等資訊以供系統自動填入使用者帳號詳細內容使用。
- [硬體] 台大資管系必須能夠提供用於最後架站所需之伺服器的硬體設備或虛擬伺服器。
- [帳戶資料] 台大資管系必須提供預設帳戶之資料，以供建立初始使用者名單之用。或系所應能提供學號之判別方式來認證所有正常的用戶，例如其學號範圍、數字上的特徵與限制等等。
- [使用者習慣討論] 本系統需能夠對隸屬於每個不同預設分類項下之使用者做指定形式之訪談，以便了解每個職位之使用者對本系統之期望與意見。此意見將影響預設帳號類別之新增與刪去、朋友關係的用途與系統未來發展之用。
- [階段性開放] 本計畫將以階段性的方式向終點努力，期間所釋出的測試系統並不一定與完成系統之表現、內容完全相同。會依照以下順序依序開放測試:
  - 使用者帳號管理
  - 討論區測試
  - 完整上線系統
- [使用者範圍] 本系統預設之使用者範圍為校友、學生、教授及行政人員。超出本規畫範圍者將必須等到完整系統上線後發現有必要才會以更新版本的新增。
- [行動裝置] 本系統雖有預設 API 開放給未來發展之非特定行動裝置系統中的程式，但是並不包含強制性的程式開發，將再行有餘力之時完成此部分。

功能模組	功能需求	功能範圍
登入模組	輸入身分 身分認證 取得 token	讓使用者宣稱自己的身分 透過計中做身分認證 回收計中回傳的 token
個人資料管理模組	管理公開資料 管理個人聯絡資料 職業生涯資料管理 隱私權管理 關係管理 儲存資料異動	讓使用者維護帳號的公開資訊 讓使用者維護帳號的聯絡資料 讓使用者維護職業生涯相關資料 讓使用者管理帳號之公開性選項 讓使用者管理與其他帳號之間的關係 儲存任何使用者新增/修改/刪除之資料
搜尋模組	搜尋特定使用者 回傳指定使用者	依照指定學號搜尋符合帳號 依照指定學號回傳符合條件之帳號
關係建立模組	傳送使用者好友要求 記錄使用者對要求之回應 記錄使用者之友好狀況	由被要求者決定是否成為該用戶之朋友 提供被要求者回應交友要求 儲存所有與該使用者為好友關係的連結
討論區模組	新增議題 瀏覽議題 標示議題類別 管理議題	讓使用者以自己的帳號新增議題 讓使用者流覽系統中之議題 讓使用者清楚了解顯示的議題之標籤 系統管理者確認類別系統正常運作
訂閱模組	訂閱使用者 訂閱議題 訂閱議題類別	讓使用者關注其他使用者之發文 讓使用者關注特定議題的後續發展 讓使用者關注特定議題類別的新增議題
API 模組	提供應用編程介面 (API)	使其他服務可藉由 JSON 與本系統互動
行動程式	取得遠端文章資料 顯示資料	讓系統透過網路與 API 互動 將下載下來的資料呈現給使用者

表 1: 功能性需求列表

- [電子郵件服務] 台大資管系在軟硬體準備時必須開起電子郵件服務，讓本系統能順暢得依照使用者之要求處理訂閱議題以及類別之需求。

3 專案範圍

3.1 功能性需求

在功能性需求的部分本系統應包含登入模組、個人資料管理模組、搜尋模組、關係建立模組、討論區模組、訂閱模組。全部功能性需求請見表 1。

3.2 非功能性需求

非功能性需求的部分主要牽涉到關於資料未來可能的彈性、系統運作的穩定性、系統元件的重複利用可能、以及資訊安全等。

- [資料可變性] 針對例如說使用者種類或是文章類型等等資訊，設計出未來較容易擴充之資料儲存模型。
- [系統穩定性] 硬體方面，需配置不斷電系統以及資料自動備份系統以維持系統不斷線的能力。另一方面，也需要設計具有平衡流量能力的伺服器架構，以有效降平均回應時間。
- [系統安全性] 軟體方面，為防止已知的幾種攻擊方式，本系統在設計時必須確保使用者不會再不恰當的地方使用”‘“等特殊符號。包含 SQL injection 在內的幾種攻擊方式都必須要在系統設計時避免。

- [使用者期望] 為確保本系統朝向正確且易學易用的方向發展，在發展過程中將不定期的針對特定使用者族群做抽樣調查。取得之資料將在適當的分析之後，用於增進系統效能與新增功能或是修改錯誤等等面向。

4 系統設計

4.1 系統概述

本系統包含兩個主要的使用情境：「使用者個人資訊與情報的建立與管理」以及「成員之間的互動論壇」。兩者之使用情境如圖 1 與圖 2。

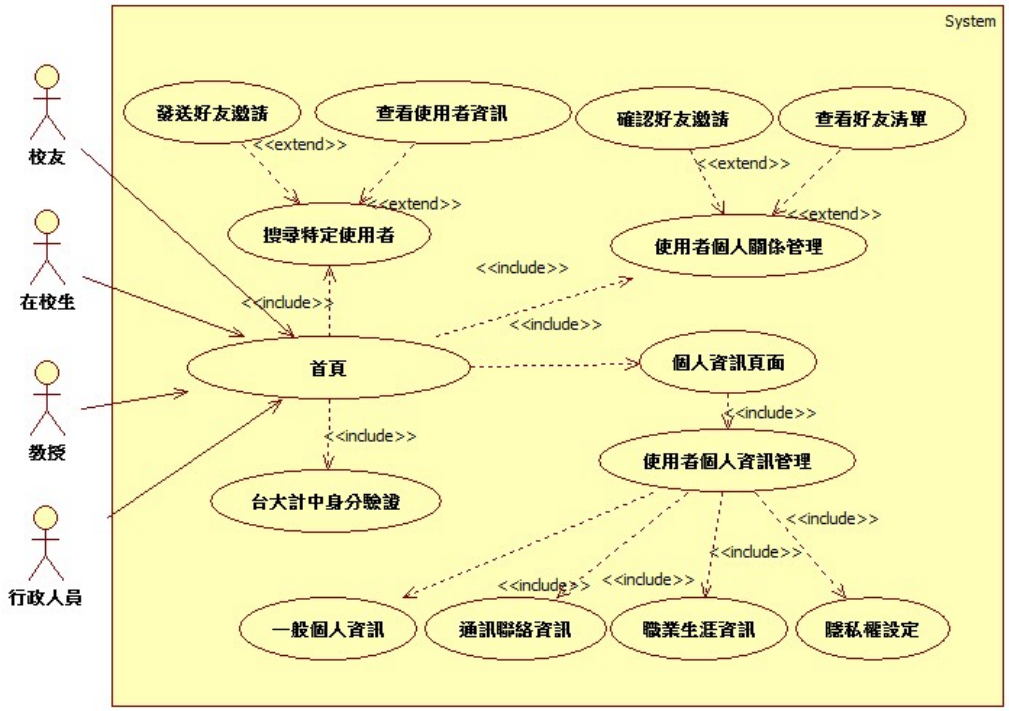

圖 1: Use Case Diagram – 使用者資訊管理

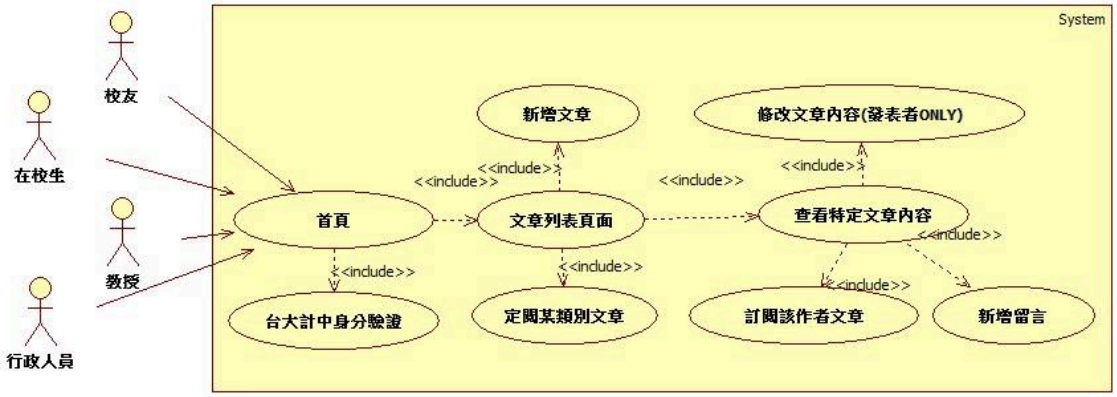

圖 2: Use Case Diagram – 文章與留言管理

以下之各小節將會逐一介紹各模組之功能及其設計概念。

4.1.1 登入模組

在登入模組中需包含以下幾點功能:

- 宣稱身分
- 身分認證
- 取得回傳之 token 並解構

在宣稱身分的地方必須由使用者宣稱自己的身分並提供相關認證資料，在經由計算機中心所提供之 single sign-on 之服務完成身分認證之後，回傳給系統一組經認證的特定格式之 Token。若使用者為初次登入本系統，則此 Token 在經系統解構之後必須能夠抽取部分必要資料填入系統之資料庫中做為使用者個基礎資訊。此行為將於前端網頁系統中完成，若有需要填入時則會交由後端資料系統負責。使用者與系統之間的互動請參考圖 3。

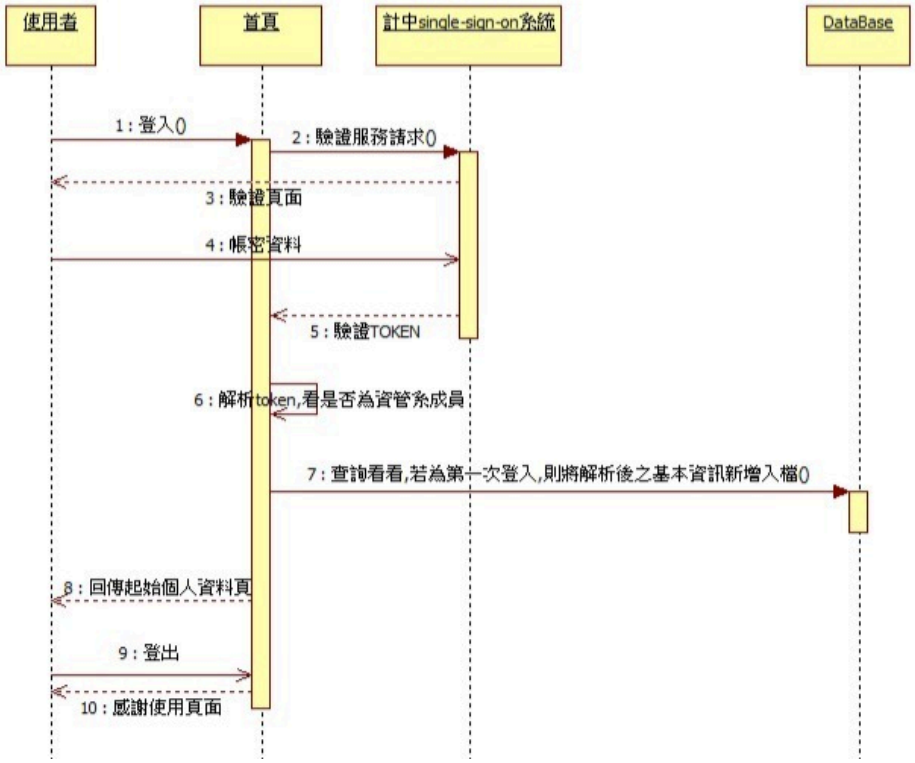

圖 3: Sequence Diagram – 首頁登入

4.1.2 個人資料管理模組

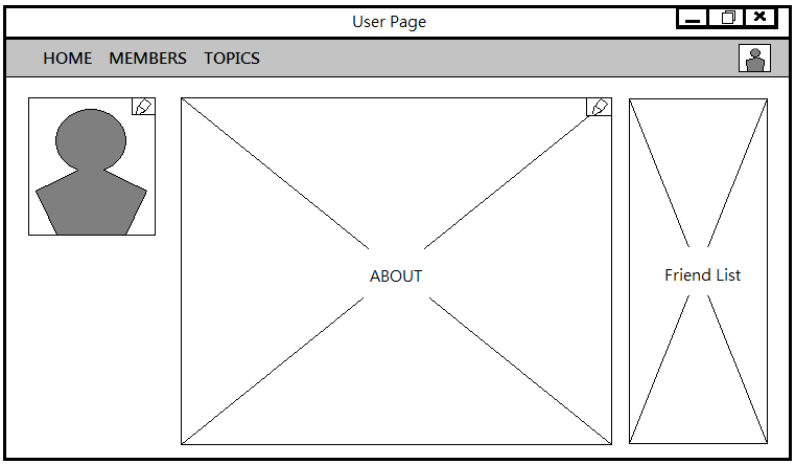

圖 4: Prototype – 用戶資訊頁面

個人資料管理模組將負責管理每個使用者各自的資料組，此資料組中包含使用者的公開資料、個人連絡資料、職業生涯相關資料與隱私權管理及關係管理。使用者必須能夠看見目前系統內的資料狀況並且能夠更動大部分的資料，其中如學號等等的則必須要經過系統管理者之認證後方可變更。此更改之行為將於前端網頁系統中完成，若有資料必須儲存則會交由後段資料系統負責。使用者與系統之間的互動請參考圖 5。此功能會產生一系列之獨立頁面以便於瀏覽與修改個人資訊，雛形如圖 4 所示。

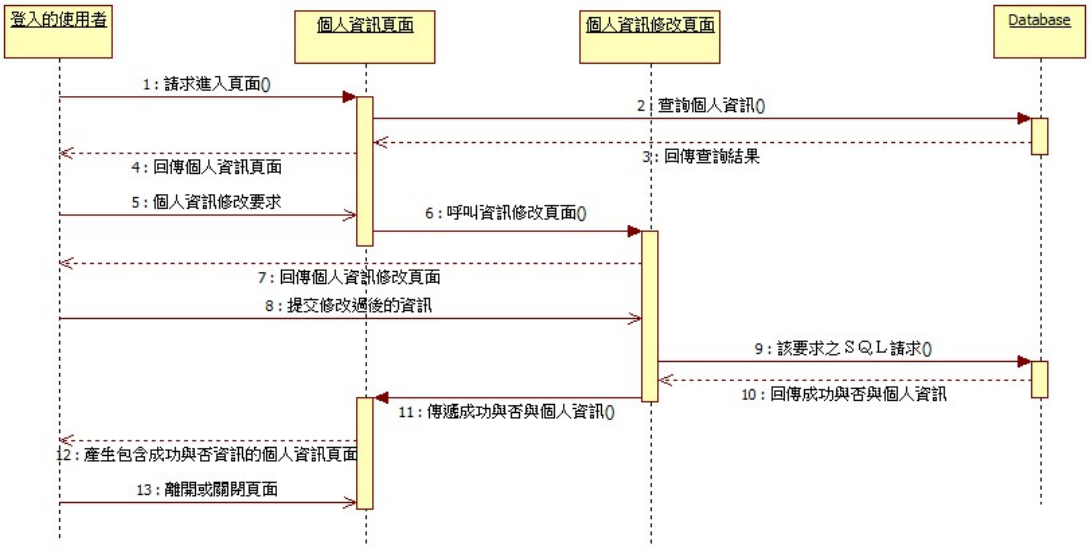

圖 5: Sequence Diagram – 個人資訊管理與個人資訊修改

4.1.3 搜尋模組

本系統為了建立一個適合使用者相互溝通之平台，必須提供使用者方便互相連結之功能，搜尋模組正是為了此需求所建立。當使用者在討論區中產生對特定用戶之好奇心，便可藉由本模組使用特定資料 (如: 學號) 做為索引找到該用戶之公開資訊。公開程度將依該用戶之隱私設定做限制，若該用戶認為預設之帳號類別方式無法達成所欲之特殊公開之組合則需透過朋友關係的建立來達成要求。互相成為朋友的兩人將以特別規則優於普通規則的方式優先採用該用戶所設定的特別規則觀看公開資訊。本搜尋功能將全部於前端網頁系

統完成。使用者與系統之間的互動模式請參考圖 6 ，可產出之相對網頁原形 (prototype) 如圖 7 。

除了使用者搜尋之外，如圖 2 中的「文章列表頁面」以及「查看特定文章內容」等部分所描述之情境，系統也允許使用者針對文章或是文章分類去進行搜尋。與此功能相對之網頁原形 (prototype) 如圖 8 。

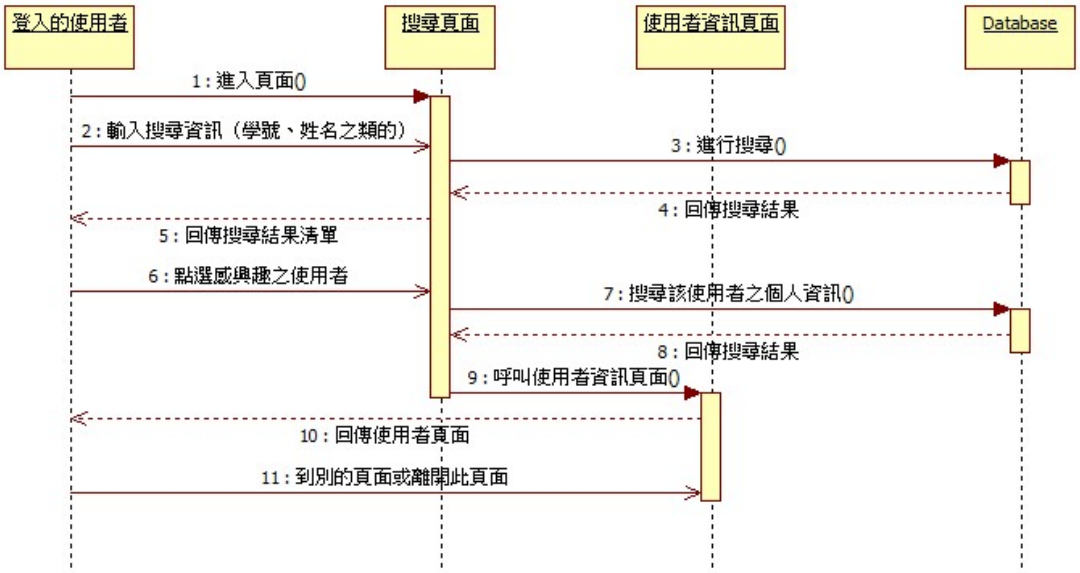

圖 6: Sequence Diagram – 搜尋查看他人使用者資訊

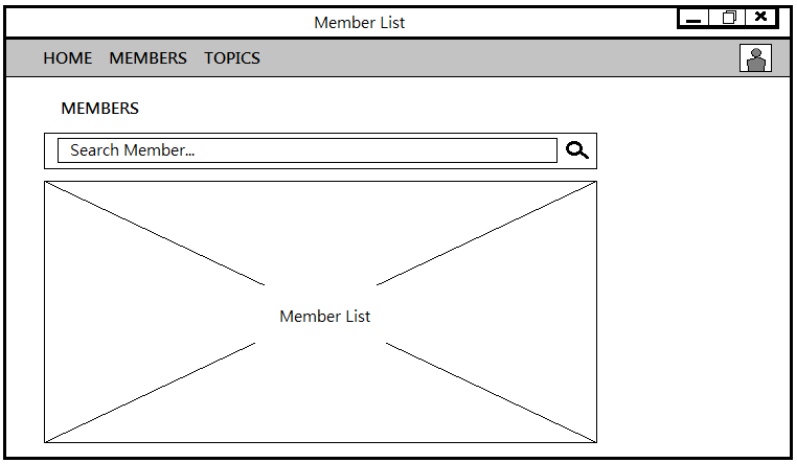

圖 7: Prototype – 使用者列表頁面 (含搜尋功能)

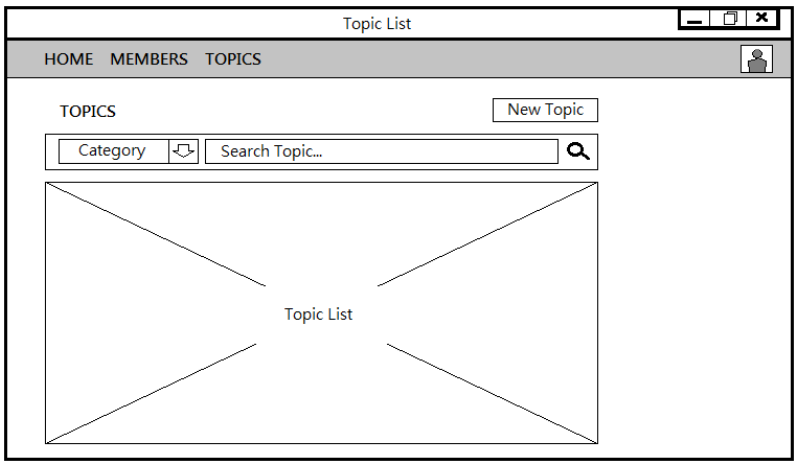

圖 8: Prototype – 文章列表頁面 (含搜尋功能)

4.1.4 關係建立模組

本系統中基於完成使用者之特定隱私權設計之理念，將在系統中提供建立好友關系的功能，也就是本模組的核心。本模組必須要能傳遞使用者的交友邀請、記錄使用者對於要求的回應與記錄使用者與其他用戶之關係。除了雙向之好友功能，本系統亦提供建立單項之關注其他使用者之功能。兩者的差異在於，好友是需要雙方合意，而可以使雙方看到對方之所有資訊；另一方面，關注關係不需要被關注者之同意，而系統僅於被關注者有發表新文章時會寄發通知信予關注者。本模組著重的重點在於資料抽象化之設計，更多細節請參考小節 4.2.2 。

4.1.5 討論區模組

討論區模組需包含以下三點功能：新增文章、瀏覽文章、標示文章類別。在新增文章時使用者將被要求選擇文章類別，以供其他使用者快速過濾想閱讀之訊息。由於系統沒有自動根據文章內容判斷其真實內容之能力，目前仍需仰賴使用者自動自發之行為來維持正確性，未來若能發展出足以將此行為自動化之方法則可降低系統在本問題之人力需求。此模組需同時提供標示文章類別與讓使用者方便瀏覽之功能，以達訊息交流之初衷。本模組將由前端網頁系統負責，只有將文章記錄至資料庫的行為需要後端資料系統輔助。

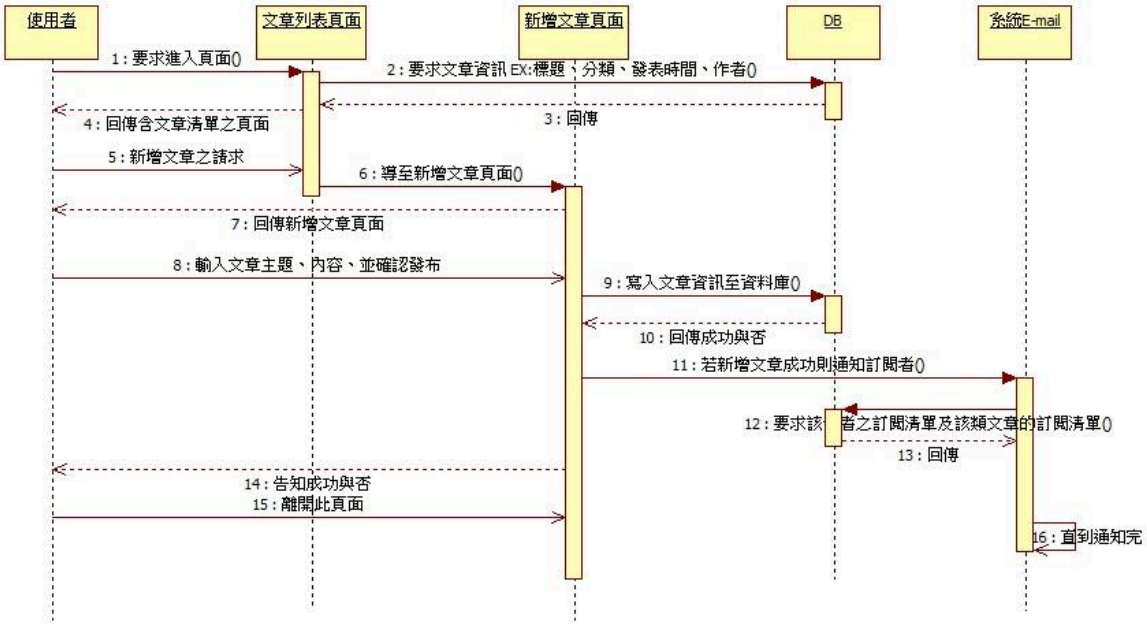

圖 9: Sequence Diagram – 新增文章

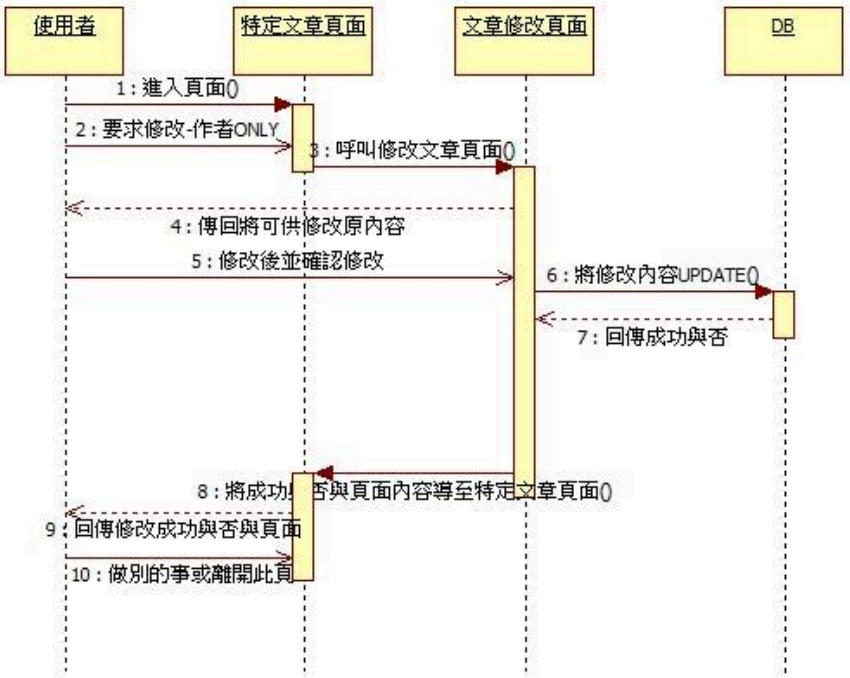

圖 10: Sequence Diagram – 修改文章

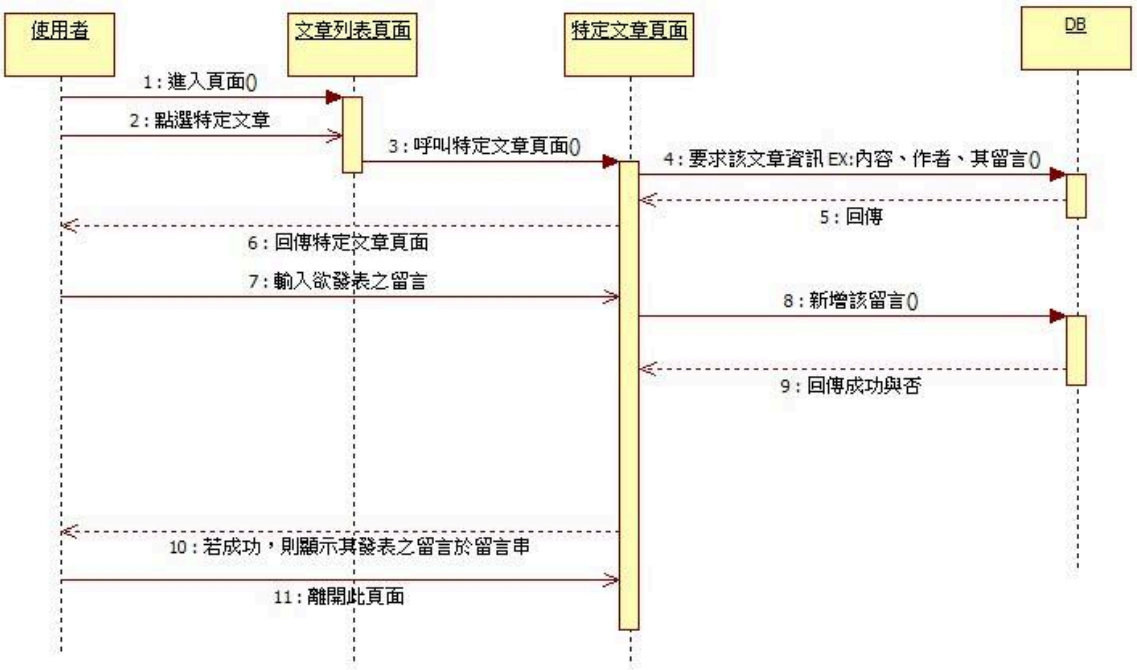

圖 11: Sequence Diagram – 新增留言

4.1.6 訂閱模組

訂閱模組中包含訂閱文章及訂閱文章類別等功能。訂閱之行為一致為當訂閱之目標有了內容之新增與改變時，將由後端信件系統發信至訂閱人所登記之信箱提醒使用者。訂閱文章的訂閱隊項為單篇文章，訂閱方法可包括在該文章留言或是點選訂閱按鈕，資料的改變為該文章被新增了評論時提醒。訂閱文章類別則將在任何一篇文章被標示為該類別時提醒使用者。本模組需由前端網頁系統展現、後端資料系統記錄與後端郵件系統共同合作完成。訂閱文章及文章分類請參考圖 12 。

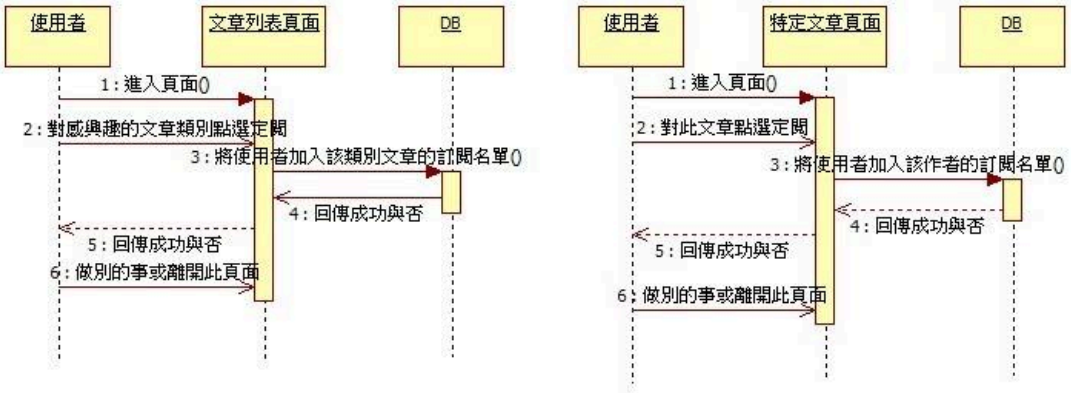

圖 12: Sequence Diagram – 訂閱文章或文章分類

4.1.7 API 模組

本系統基於 Grape，一個 REST-like API micro-framework，與 Ruby on Rails 整合，來提供系統 JSON API，並遵循 REST (Representational State Transfer) 的軟體架構風格實作，客戶端的應用通過 URI 來獲取資源的表徵，收取以 JSON 之資料格式來產生狀態的變化，對資源的操作包括獲取、創建、修改和刪除資源，這些操作對應於 HTTP 協議提供的 GET、POST、PUT 和 DELETE 方法。以新增議題為例：當使用者需要新增議題時，透過客戶端程式來以 HTTP request Server，因為新增文章代表需創建新的資料，所以使用 POST 做為 HTTP request 的方法，並對其設定的 URI（http://alumnibook.com/api/topics/）做 request，這樣 Server 就會判斷其 HTTP requset 參數來做相對應的回覆，若所夾帶參數通過檢驗即可建立新的議題，並回覆新增之議題的資料，若所夾帶參數未能通過檢驗，例如必須欄位未提供或是其資料格式錯誤，則會回傳錯誤訊息，API 所回傳的資料格式皆以 JSON 作為主要資料格式。API 最主要可以提供其他網站服務或是行動裝置端的程式使用，例如說圖 14 呈現了以手機程式為客戶端時，該服務與本系統之 API 之間可能的互動模式。

圖 13: Android Screenshot – 行動程式

4.1.8 行動程式

行動裝置端的程式極其輕便，僅帶來快速的文章流覽功能。手機程式的執行畫面雛形如圖 13 所示。在手機端程式啟動後，程式會向 API 要求資料。在接收到伺服器端所提供的 JSON 陣列後，手機端會做一連串的處理動作。從該 JSON 陣列中一項一項抽取出來的資料在成功解構後會暫時以方便取用的格式放在本機端以供未來使用，並且先將使用者列表簡單的呈現給使用者。使用者選擇有興趣的文章時則可從本機端的資料中取得該篇文章的內文、作者以及評論。手機端程式與網站 API 之間的互動細節請參考圖 14。

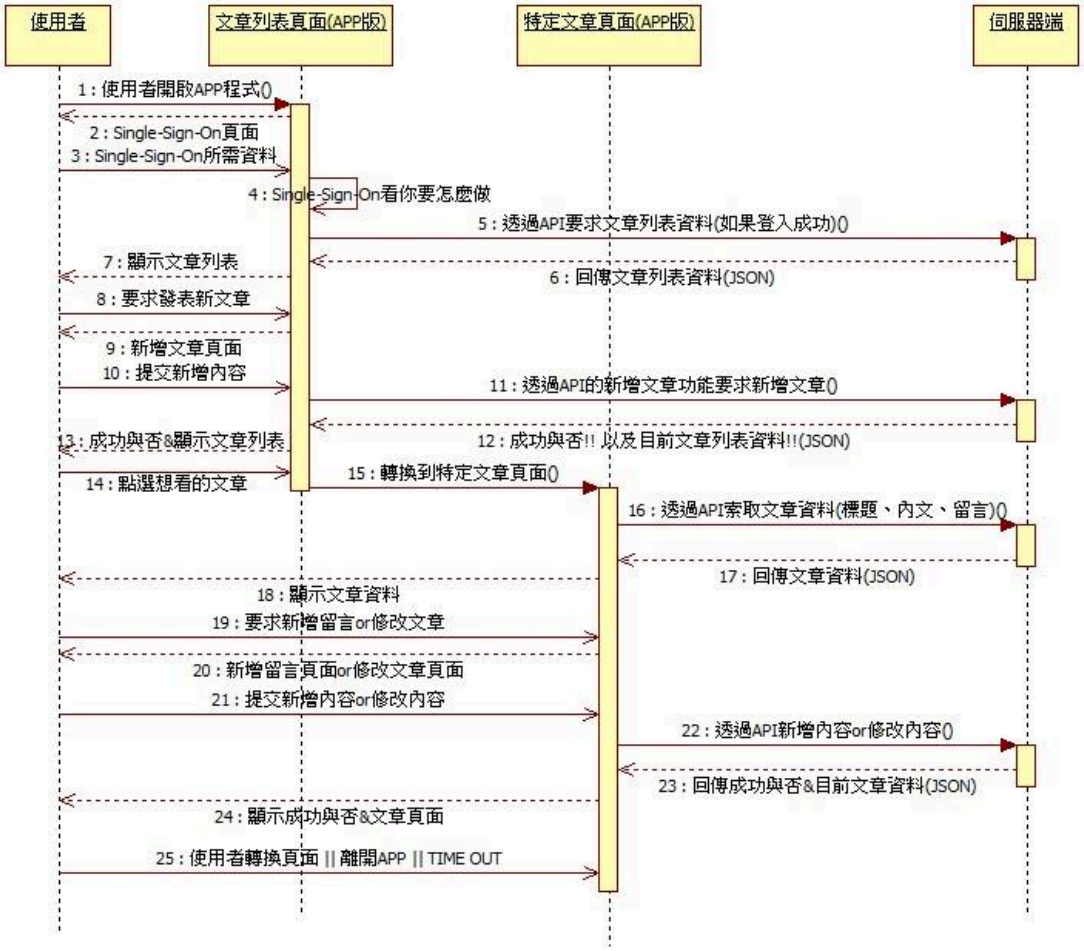

圖 14: Sequence Diagram – API and Android App

4.2 資料庫規劃

本節將會介紹本系統所需之資料模型與資料庫設計概念。本系統具有兩個相對複雜且重複性高的功能：「關注」與「留言」；更甚者，此二功能都是未來有可能因為各自牽涉之主體有所異動而需要擴充的部份，因此將此二者予以抽象化就是一項重要的工作。圖 15 與圖 16 分別對應上述兩項功能的抽象資料模型。可以看到兩圖的樣式非常接近，因此更進一步的抽象化是可能的。但是考量到可能之系統規模之後，為了避免過度設計 (over-design) 這邊就不予以進行更進一步之抽象化。另一方面兩圖中灰色的部份是未來可擴充之部分，但是同樣因為系統規模與時程考量而不予實作。

圖 15 描述了廣義的「關注」行為。此行為之概念如下：兩物件 (Object) 可以行成一組關注關係，而根據兩物件與該關注關係之種類可以決定一組實際的系統行為。此模型可以描述例如文章訂閱以及使用者關注等功能。更多「關注」相關的實際資料庫設計請參考小節 4.2.2 與 4.2.4。

圖 16 描述了概念性的「留言」。本系統目前僅有兩種留言模式：預設，以及巢狀留言。預設留言係指使用者可以針對所有非留言之文章進行留言，而巢狀留言則是特指針對現有之留言進行留言。留言行為本身的抽象化是為了未來文章種類或是留言相關功能之擴充所需。更多「留言」相關的實際資料庫設計請參考小節 4.2.3。

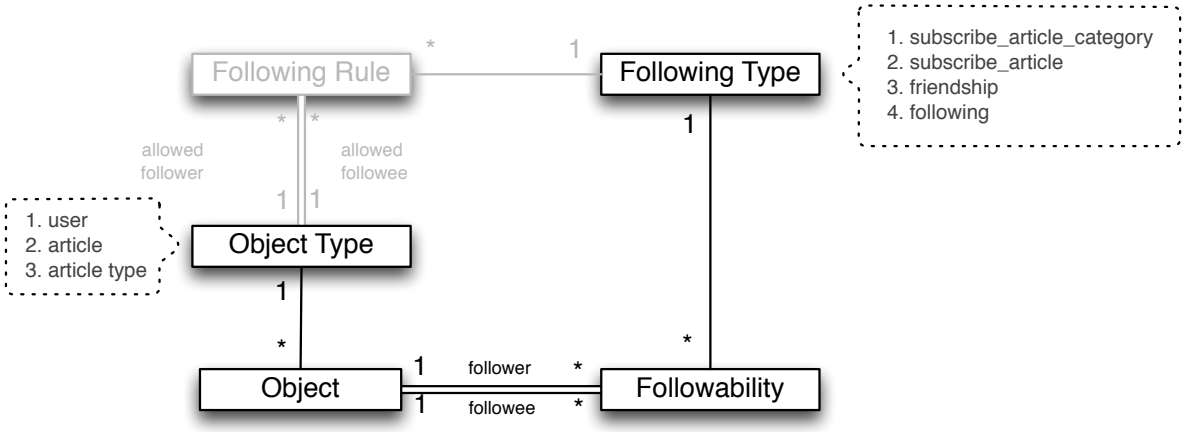

圖 15: Data Model – Followability Model

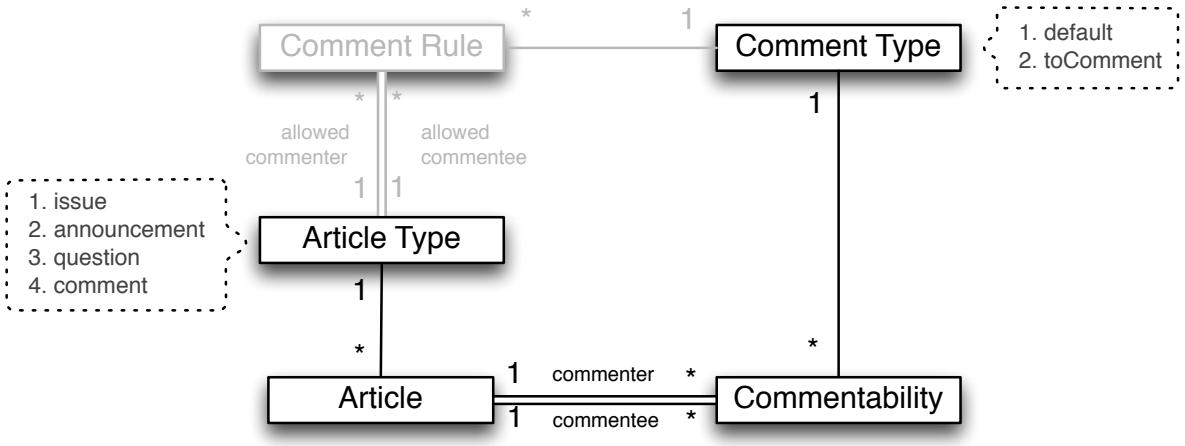

圖 16: Data Model – Commentability Model

4.2.1 個人資料管理模組

使用者資料之資料庫模型如圖 17 所示，使用者資訊根據修改的程度與層級區分成三塊：

- 不可由個人自行更動的個人系統訊息 (users_sysinfo)
- 可更動之個人系統訊息 (users_info)
- 可更動之個人公開訊息 (users_pubprofile)

一組使用者資訊會對應到某個使用者類別，目前預設的類別有：校友、學生、教員、職員。一組使用者資訊也會對應到一串工作經歷，而一項工作經歷會包含一項該工作期間的任職單位。任職單位的抽象化有助於未來履歷的建立與搜尋以及企業徵才的便利性。

4.2.2 關係建立模組

建立關係所需的資料庫模型如圖 18 所示，圖 18 中描述兩個使用者之間可以存在兩種關係：

- 好友 (friendship) – 需要對方的允許、可以閱讀對方所以非公開資訊
- 關注 (following) – 無需對方的允許、對方發文時會有通知信

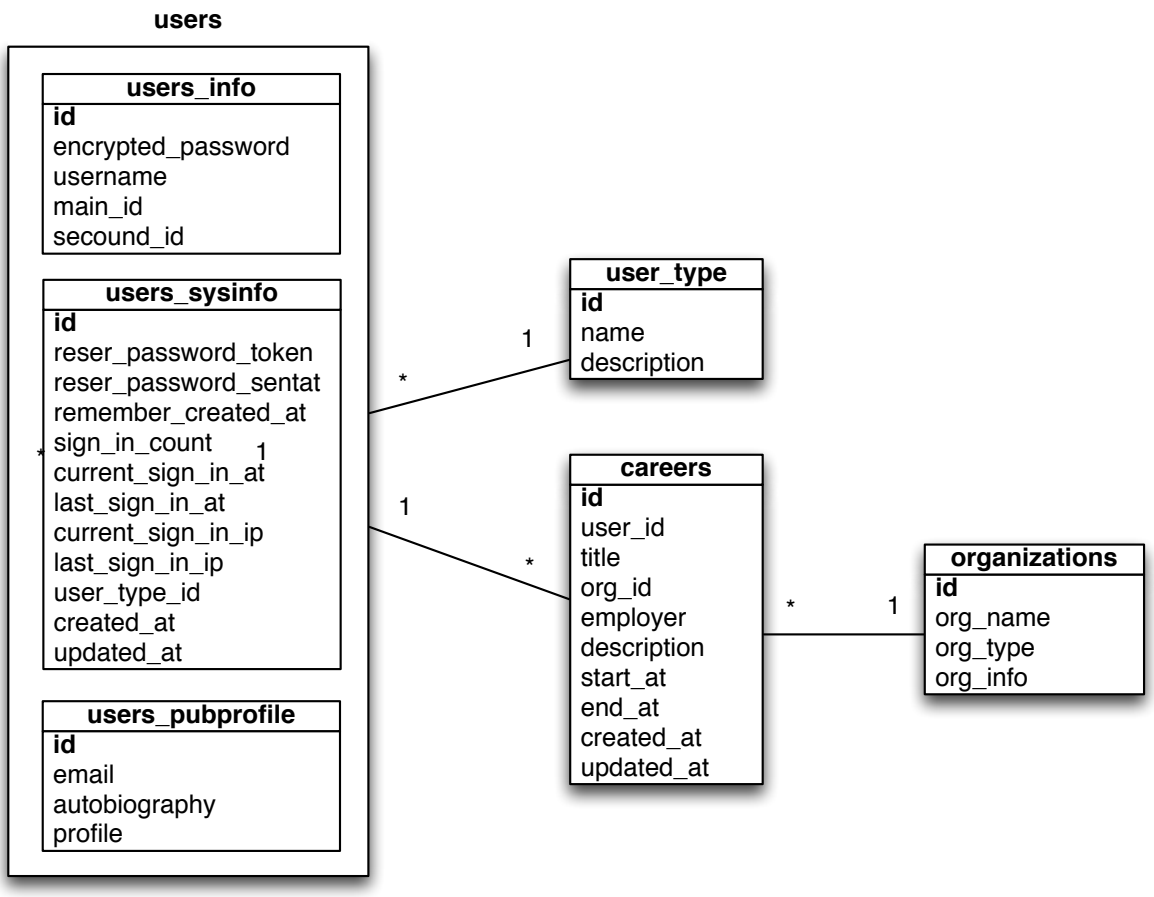

圖 17: Data Model – User Model

本模組之資料庫是實踐圖 15 所描述的抽象資料模型，如是實作之價值在於保有兩使用者間的關係之可擴充性，例如說可能將完全封鎖 (Block) 給依照此模式實作成一種 Following Type 。圖 18 中兩個使用者 (user) 的實際細節請參考圖 17。

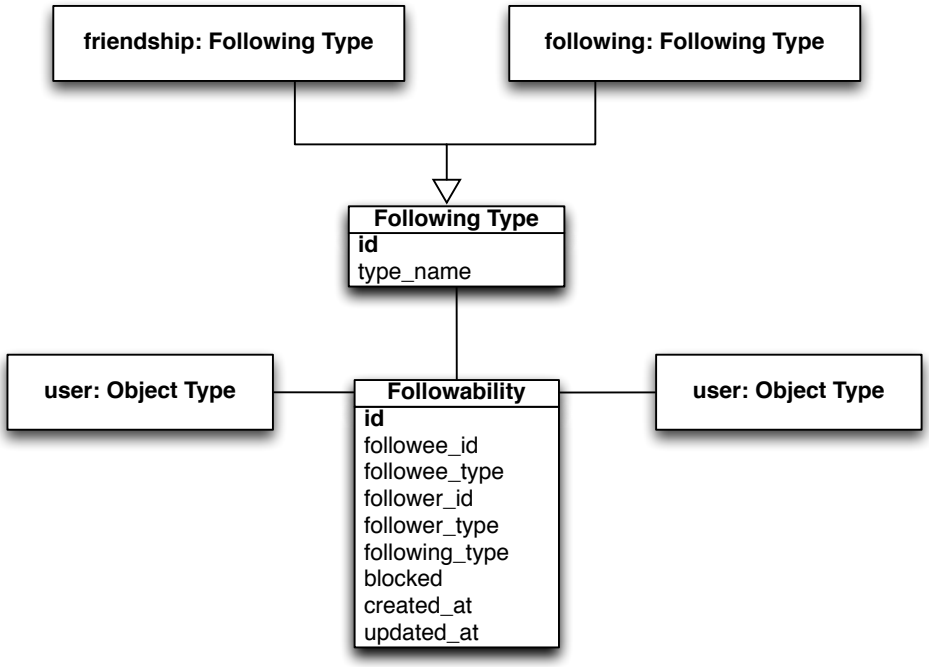

圖 18: Data Model – Following and Friendship Model

4.2.3 討論區模組

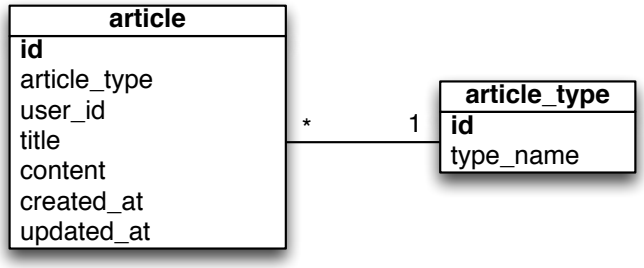

圖 19: Data Model – Article Model

討論區最基本的就是提供使用者管理文章以及發表留言等能力。文章管理，包含新增、刪除與修改等等，所需要操作之資料庫設計如下圖 19 所示。目前提供四種預設的文章型別：

- 議題 (issue) – 一般性討論議題
- 公告 (announcement) – 系所或系統公告
- 詢問 (question) – 詢問性質之文章
- 留言 (comment) – 針對上述三種文章之回應

除了文章管理以外，如圖 16 所描繪之留言機制也是討論區的一個重點功能。圖 20 意指一筆留言可以對應到一筆任意型別之文章，且根據後者個型別而有不同之行為。例如說預設的情況是：留言在非留言之文章會寄出通知信給原作者並且更新統計資訊；但若是留言給留言本身的話則僅會在寄信給原留言者。

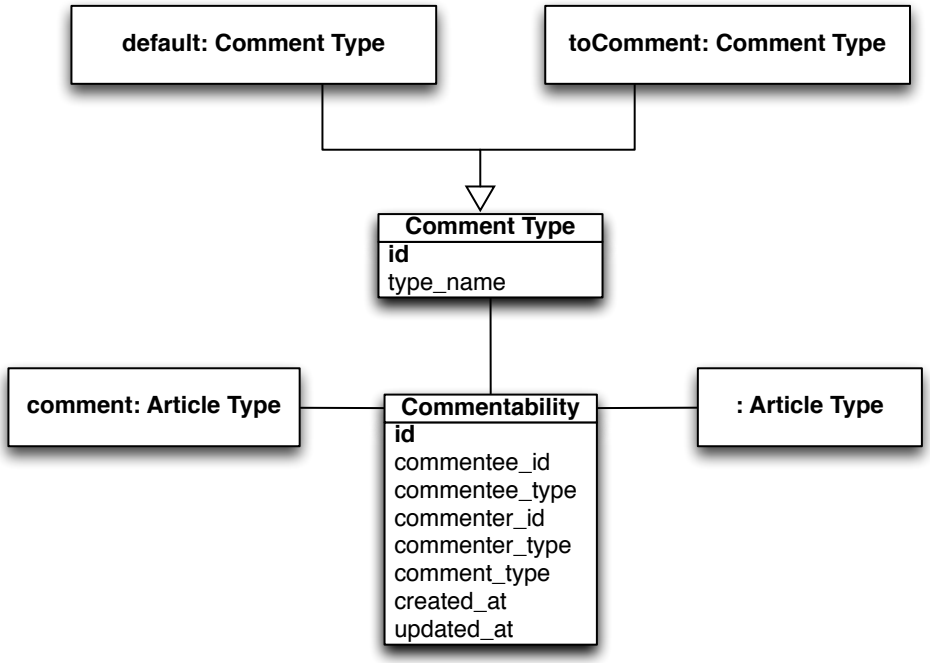

圖 20: Data Model – Comment Model

4.2.4 訂閱模組

文章訂閱模組所需的資料庫模型如圖 21 所示，使用者可以訂閱文章 (article) 或文章類別 (article type)：

- 訂閱文章類別 (subscribe_article_category) – 有新文章時會有通知；
- 訂閱單一文章 (subscribe_article) – 有新留言時會有通知；

本模組所需之資料庫如同小節 4.2.2 一般，是實踐圖 15 所描述之抽象資料模型。另外，圖 21 中組件如使用者 (user) 與文章 (article) 等之設計細節請參考小節 4.2.1 與 4.2.3。

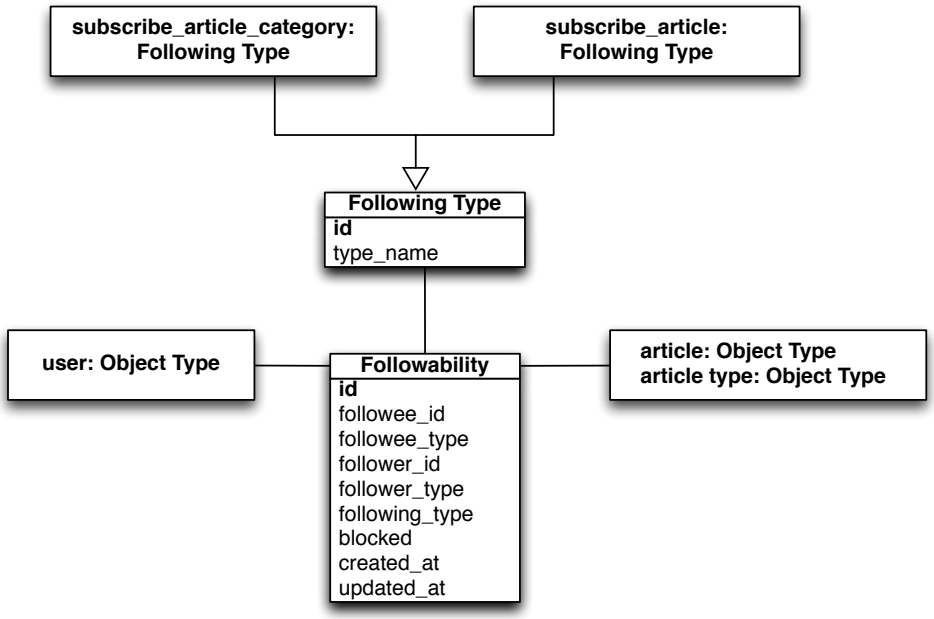

圖 21: Data Model – Subscription Model

4.3 軟工議題

4.3.1 設計模式

本系統實作時將使用下述設計模式以增加系統架構之彈性。

Model-View-Controller 本系統採用 Ruby on Rails 作為主要伺服器端開發架構，而 Ruby on Rails 是一套建基於 MVC 架構上之網站開發平台。MVC 架構的好處是可以將「回應與運算邏輯」、「資訊顯示邏輯」以及「資料操作邏輯」這三個程式邏輯所對應之程式區塊區別清楚，對於程式碼管理以及系統擴充性都有著顯著之提昇。

Adapter 本系統中具有在各種不同情境時寄發通知信之功能，而電子郵件的寄送服務會是採用外部提供之軟體工具。因此本系統採用 Adapter Pattern 將外部的郵件服務以及相關設定包裝在一自製之獨立的中介介面中，而系統中其他使用郵件服務之程式片段都是使用該自製介面而非原始的外部服務。如此一來便可以簡便地對於郵件服務予以擴充或是修改，而不需要影響到整個系統中所有與郵件服務有關之程式片段。簡略的叫用關係請參考圖 22。

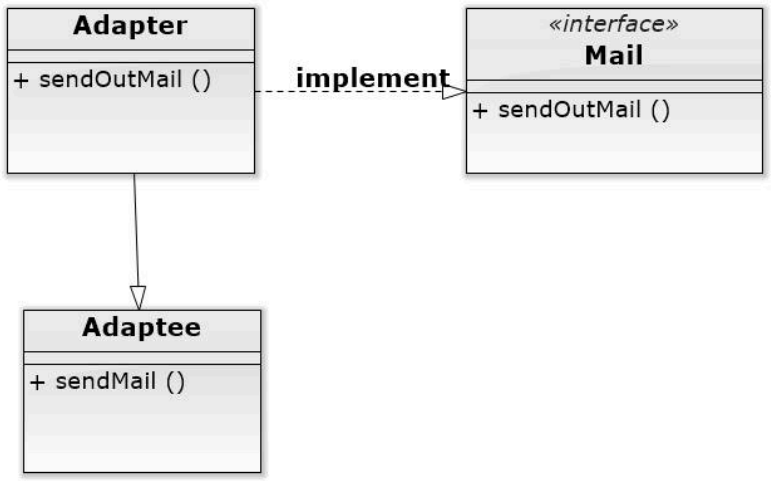

圖 22: Design Pattern – Adapter Pattern

Observer Observer 可以讓我們可以針對 Model 的生命週期中的某些階段做出對應的行為，例如 model 新增、修改、查詢、刪除時，觀察者 (observers) 可以在 model 的外部進行相關的動作，model 可以藉由註冊觀察者來讓觀察者監聽自己的行為，不同的 model 也可以註冊共用觀察者，

在此系統我們使用 Observer pattern 來設計議題產生與留言產生時的通知，當 Topic 建立時，觀察者會傳送通知信給有跟隨 Topic 建立者的使用者，在 Comment 建立時，觀察者會傳送通知信給有訂閱此 Comment 所屬的 Topic 的使用者。這樣的設計模式可以有效區別 Model 本身的用途與外部功能，例如寄送通知信、確認信等進行外部動作時，即可使用觀察者模式來設計。實作上，類別之間的關係請參考圖 23。

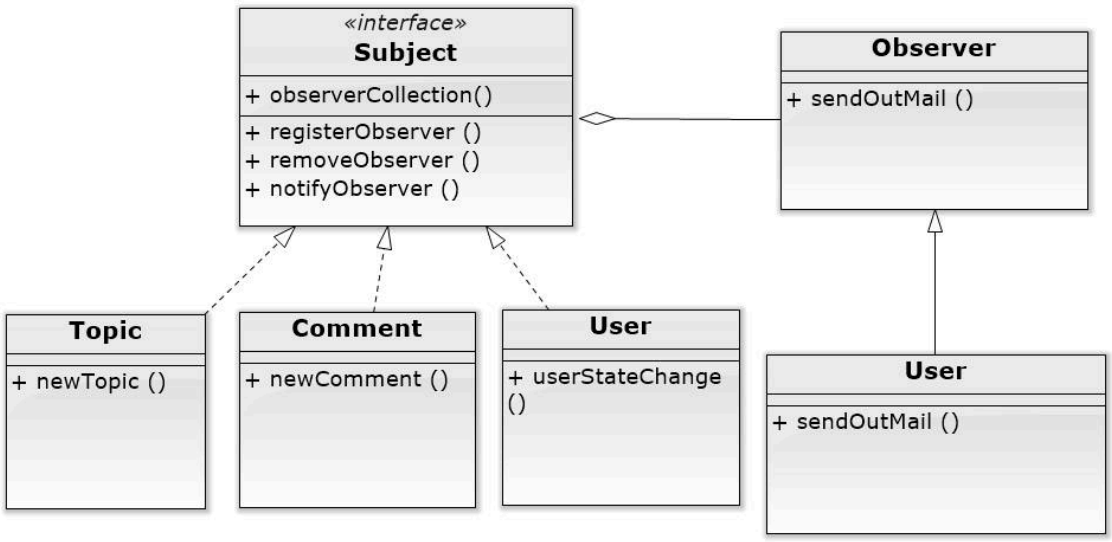

圖 23: Design Pattern – Observer Pattern

4.4 使用套裝軟體及開發工具、程式語言

本系統採用 Ruby on Rails 作為主要伺服器端開發架構，使用 Ruby 2.0.0-p247 版本，Rails 4.0.0 版本，以 Grape 這個 Ruby REST-like API micro-framework 建立起主要 API server，提供網頁端與手機 JSON API，連結 MySQL(5.5.29) 資料庫儲存索取更新並維持資料正確與一致性，前端網頁頁面以 HTML5(HTML, CSS, JavaScript) 撰寫而成，採用 Angular.js(1.2.0) 一個前端 JavaScript MVC 架構增加程式碼模組化與維護性，使用介面以 Bootstrap(3.0.0) 作為主要網頁前端外觀元件框架，加速雛形開發。

手機端的系統目前採用 Android 系統，相容於幾乎任何版本。由於資料量有限所以並沒有儲存至 Database 。若未來有儲存的需求則會使用 SQLite database 搭配名為 GreenDao 的第三方程式來管理。

4.5 系統環境需求

目前 Ruby on Rails Server 架設在 Heroku 一個雲端應用平台，提供高度延展性，以亞馬遜 AWS EC2 為基礎建構而成的伺服器叢集，使用 git 上傳即可快速部屬，伺服器主要需要運行在 Ruby 2.0.0 與 Rails4.0.0 環境下。網頁前端頁面則放置在 Http server (如 Apache, Nginx) 下，並與 PostgreSQL 資料庫放在資管系主機。前端網頁則相容於多數瀏覽器 (Chrome, Safari, Firefox…)，提供使用者多樣的使用選擇。手機端在公開之後可以請使用者下載 APK 來安裝，或是在上架後透過 googleplay 下載。相容於幾乎任何版本的 Android 作業系統，使用者應不用擔心系統版本的問題。

5 課程心得

5.1 郁昀彥 **R00725051**

5.2 郭瀚智 **R02725023**

5.3 鄭立民 **R02725041**

於本學期的課程專案中，首先在設計階段學到了許多的 UML 圖，雖然應用上也只有畫 USE CASE 跟 Sequence 而已，但其實許多其他圖的精神也都有包還在內 (像是 Sequence 感覺跟活動流程圖還蠻像的)，當然能各種圖都畫出來是最好，也最容易無死角的了解。相信在未來有做專案設計的機會時，這些經驗與知識會有所幫助。

同時在這門課也教了各式設計巧思的 Design pattern。雖然沒實際運用過的話，說實在也很難記得。其他還有像是網站資安問題，其中各式各樣的 SQL injection 讓我最為印象深刻。

本次專案最重要的是，真的體悟到一份專案要管理好實在十分困難，各式各樣的時間不好喬，各式各樣的溝通，加上組員間的能力以及積極度差異，皆會增加做好專案的難易度。所幸我們有強大的 Coding 大師們以及組長的積極溝通協調，上山下海全都包，否則產出一份專案當真沒那麼容易。

5.4 李奕德 B99705021

在這學期的課程中我學到的 Design pattern 和 Datamodeling 令我受益最多，這門課著實地讓我知道我從過去到現在所有寫過的東西都非常的雜亂且狹隘。Android 系統的開發上，雖然課堂時間有限所以很緊湊的上了非常多的東西，不過仍然對我們後來再寫手機端時仍然很有幫助。在課堂中所學將我們會用到比較複雜的部份都有帶到，讓開發起來不至於完成不了任何要求。雖然由於時間壓力沒辦法將所有設計都實作，但是仍完成了比較核心的部分。

在這次的 project 中我們選擇了 MVC 切割較為清楚的 ruby on rails 來實作，在製作 Android 端程式時也將三種不同的面相分成不同的 thread 去處理以求落實 MVC 架構。同時，我們盡量在規劃資料模型時讓他具有一定的彈性來使未來需要擴充時不至於完全沒辦法做。

5.5 張凱涵 B00705027

5.6 施淮振 B00705047

5.7 倪嘉銘 T02705102

暑假在家裡選課的時候就對軟體開發這門課程充滿期待，因為在重慶大學的資管系並沒有開設類似的寫程序的課程，然而我對軟體開發還是比較有興趣的，所以很期待這門課程。

在一個學期的學習中，課程的進度顯然要比大陸的的課程快很多，例如我們第一個作業就要求用兩個禮拜完成一個簡單的 demo，然而老師在整個過程中都沒有教任何的程式，完全讓學生自己研究。這一點，大陸很難做到，但是其實我還是覺得台灣這樣的教育是非常好的，一方面這樣的方式能夠充分鍛煉學生的自學能力，另一方面也能讓整個課程更加充實，老師能夠講授更多有意義的東西。

在整個學期的過程中，老師邀請的嘉賓從各個方面系統地全面地給我們介紹了軟體開發的各個方面。例如一開始老師就非常強調工具的使用，比如 git，這也是在大陸的課程中沒有接觸到的一部分。我能感受的區別就是，台灣的資管系與業界的聯繫要比大陸的資管系課程與業界的練習緊密許多，但是無疑台大這樣的方式能夠讓學生盡早地接觸與業界相關的東西，更能夠適應業界的規則，也能夠讓學生更快地進步。許多嘉賓的講課讓我在軟體開發方面看到了更遠的東西，不再僅僅是將老師要求的功能實現，還有更深遠的程序質量，程序後期的維護，程序的變動，數據模型的合理建構等等。

在 project 方面，老師要求我們分組做一個校友管理的系統，便是要求我們實作一個程序，我們組所應用的現在漸漸流行的 ROR，也讓我接觸到了，業界更新的，更酷的技術，雖然自己在整個開發過程中都出於學習的狀態，並沒有特別多的實作貢獻，但是還是學習到了很多心的東西，收益良多。

有一點比較遺憾的就是之前乜有學習，面向對象的 C++ 和 java，所以有些課程並沒有能夠很好地掌握，所以學習效果似乎並沒有那麼好。

總而言之，一個學期的學習還是十分充實的。