Embedded System Experiment HW1 – Chat Room

電機四 b02901014 王崇勳 / 電機四 b02901123 林哲佑

◎Web app 功能

1. 新增使用者與登入

每位使用者都會有一組帳號或密碼。如果是第一次登入，便會註冊這組帳號密碼。若是輸入的帳號密碼已經被登入過，則會被要求重新輸入帳號密碼。

2. 公開留言板

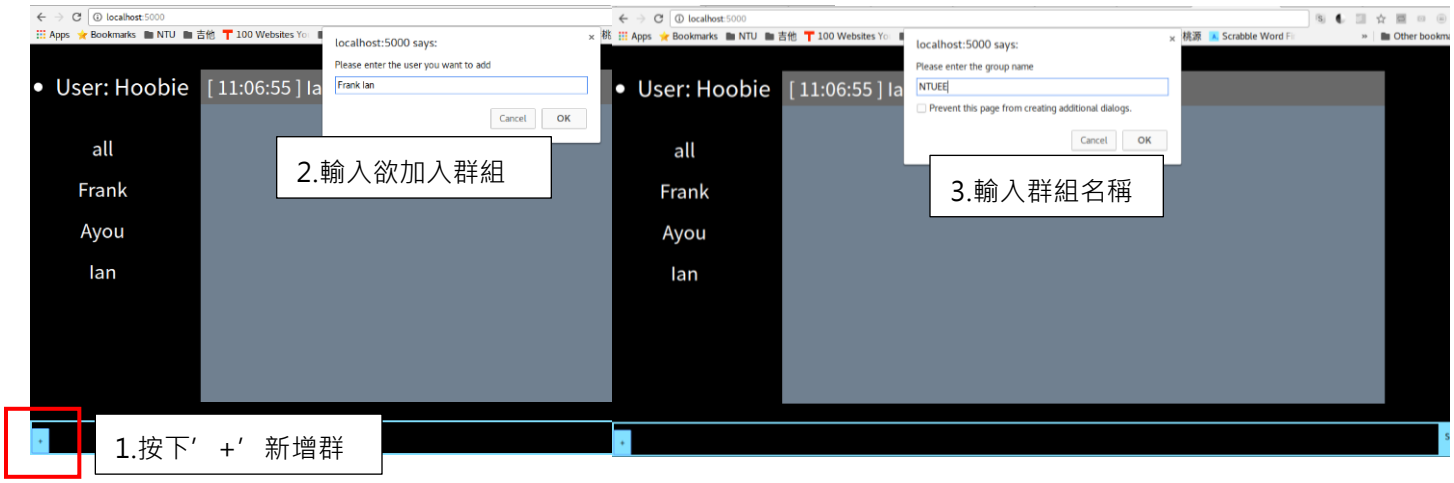
在留言板上的訊息是公開的，所有使用者都能夠看到，也都能回復訊息。

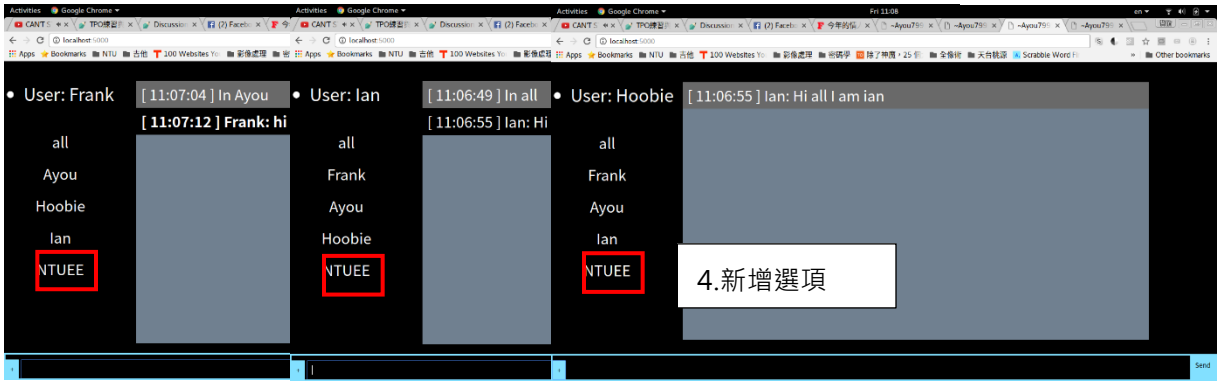
3. 與個別成員聊天

除了公開留言板外，還可以選擇與其他聊天室的成員對話。

4. 建立群組

點選左下的「＋」按鈕，可以建立群組，開啟群聊。





◎系統架構

1. 帳號密碼驗證

(1.) 使用者輸入帳號密碼後，Client 端會經由 socket.emit(‘auth’) 請求 Server 端驗證帳號密碼是否合法。Sever 端由 socket.on(‘auth’)接收此事件。此時 Server 端會檢查資料庫中是否有該帳號密碼。

a.) 如果帳號密碼不存在，便將此組帳號密碼加入資料庫中。同時資料庫會傳送訊息告知 Server 端此組帳號密碼合法。

b.) 若是帳號密碼存在，*資料庫會判斷*是否有使用者使用此組帳號密碼登入。如果沒有，資料庫告知 Server 端為此組帳號密碼合法；反之亦然。

(2.) Server 端接收到資料庫的訊息後會判定是否驗證成功，由 socket.emit(‘pass auth’)將結果告知 Client 端。同時 Server 會用 io.emit(‘user join’)告知所有使用者有一個使用者登入。

(3.) 如果 Server 端判定驗證成功，會在 Server 端存取該使用者的帳號名稱以及 Socket，以便於未來使用者間對話的傳輸使用。

(4.) Figure 1 為登入帳號密碼的系統架構及流程圖。

※socket 的部分都是使用 socket.io 實作

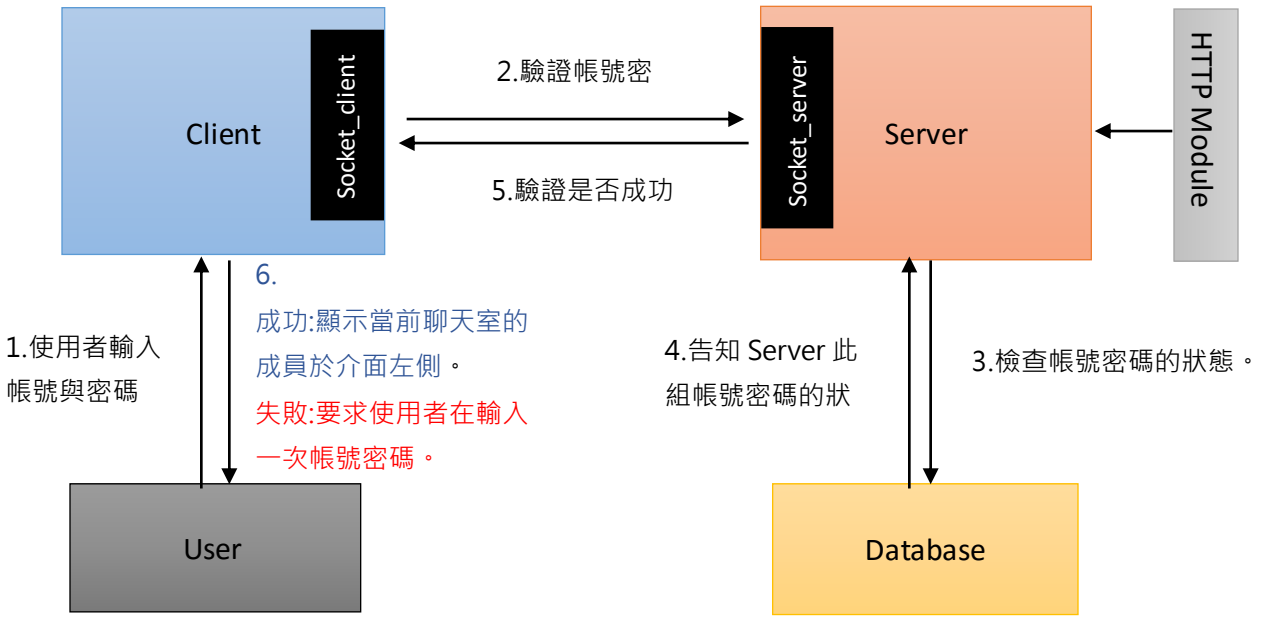


Figure 1

2. 選擇訊息傳送對象 & 要求歷史紀錄

(1.) 當使用者點選欲傳送訊息對象時，Client 端會先把原本顯示在使用者介面的訊息移

除。同時 Client 端會利用 socket.emit(‘old message’)向 Server 端請求從資料庫搜尋是否有歷史對話紀錄。如果有，資料庫會將歷史訊息經由 Server 端利用 socket.emit(‘append old message’)傳給 Client 端。最後呈現在使用者介面上。

(2.) Figure 2 為選擇訊息傳送對象，以及要求歷史紀錄的系統架構及流程圖。

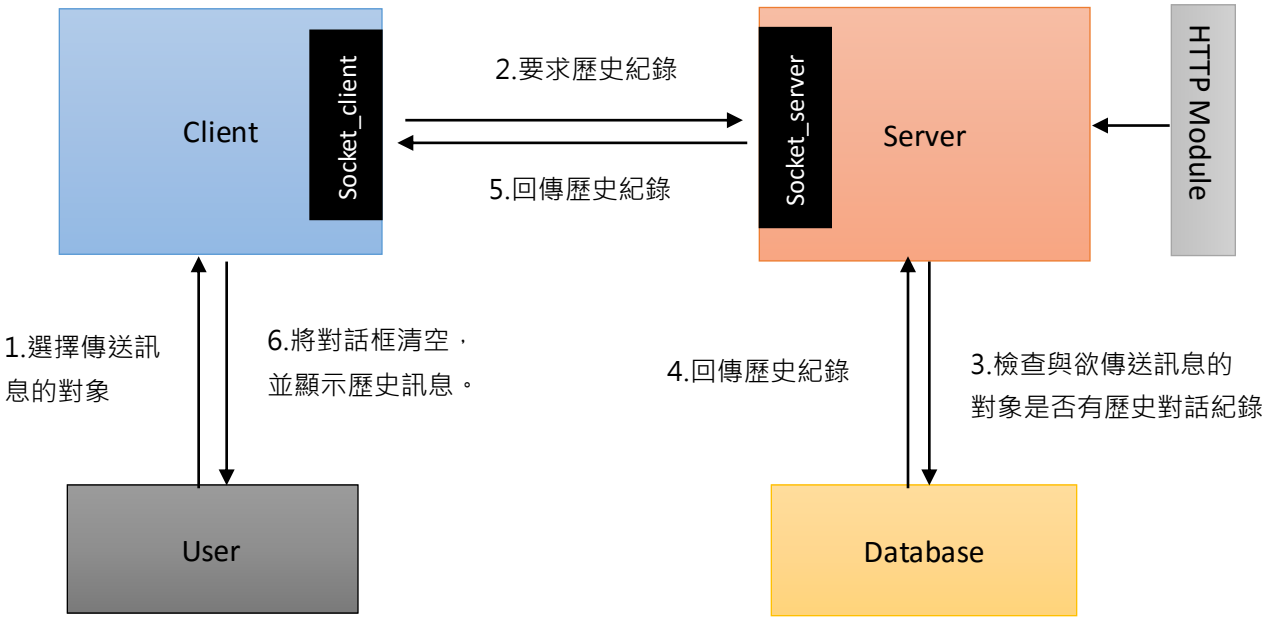


Figure 2

3. 訊息傳送 & 儲存訊息

(1.) 當使用者傳出訊息時，此訊息會被包成一個物件，此物件包含：傳送者、接收者、傳送時間、訊息。此訊息除了會被顯示在傳送者的對話框中，同時此物件會經由 Client 端的 socket.emit(‘chat message’)傳送至 Server 端。傳送至 Server 端後，此物件會儲存到資料庫中作為歷史紀錄。

(2.) Server 端接受到 Client 端的物件後會判斷接收者是誰，並在 Server 端存取的所有 Socket 中找出接收者的 Socket。不過 Server 端要先檢查接收者當前的對話對象是不是傳訊者。

a.) 如果是，Server 端便可以透過 Socket 傳送物件給接收者的 Client 端，最後呈現在接收者的使用者介面上。

b.) 如果不是，此訊息不會透過 Server 端傳送至 Client 端。此則訊息會在下次接收者點選該使用者時，以歷史訊息的方式傳給 Client 端。

(3.) Figure 3 為訊息傳送與儲存訊息的系統架構及流程圖。

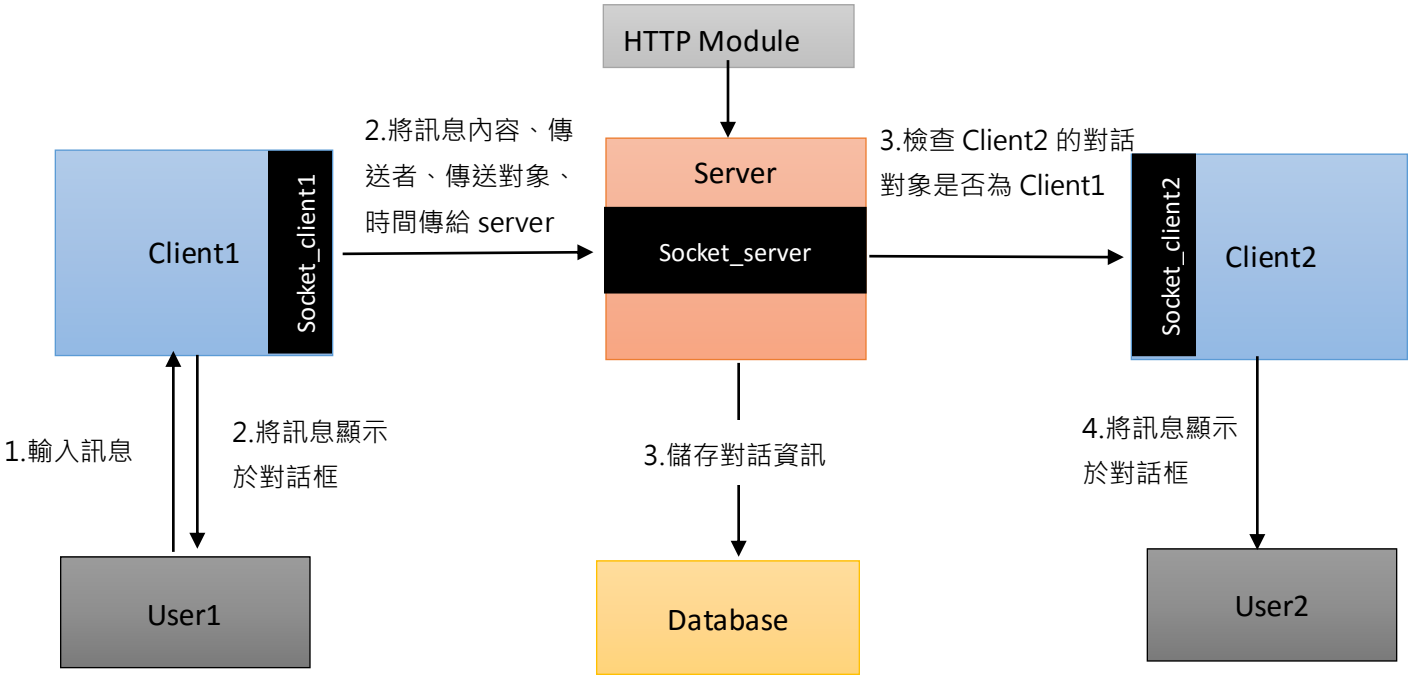


Figure 3

4. 建立群組

(1.) 當使用者輸入群組名稱以及欲加入群組的成員時，Client 端會觸發 socket.emit('add group')將建立群組的訊息傳遞給 Server 端。Server 端會根據群組內的成員，找出對應於這些成員的 Socekts，利用這些 Sockets 將此訊息傳給群組成員的 Client 端。最後在使用者介面上的使用者列表中，新增一個群組名稱的選項。

(2.) Figure 4 為建立群組的系統架構及流程圖。

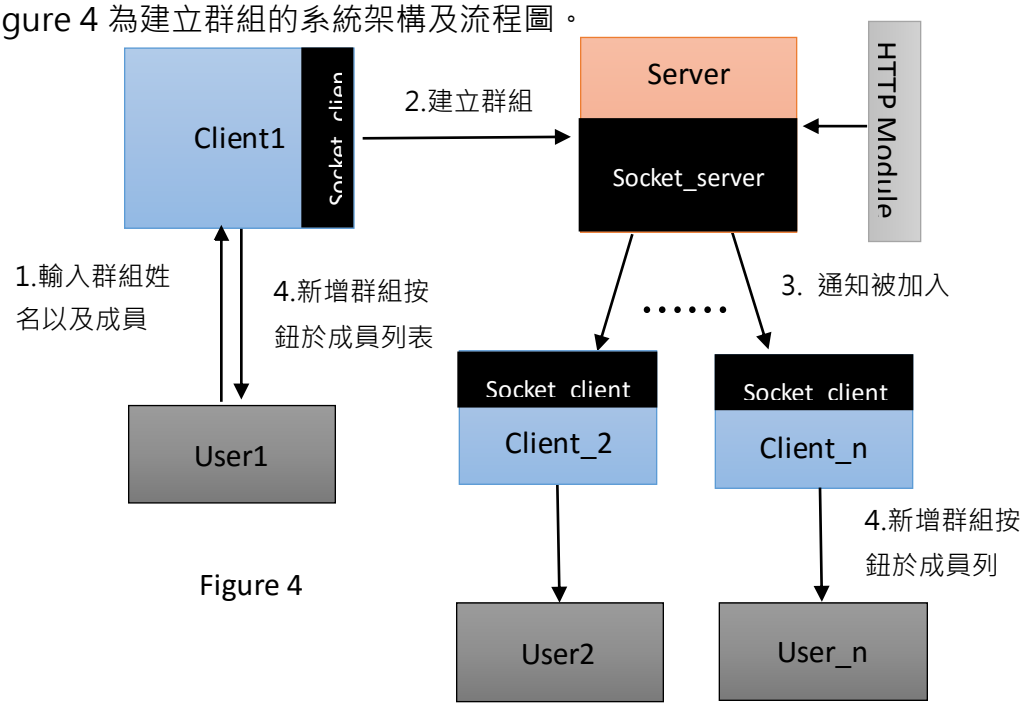


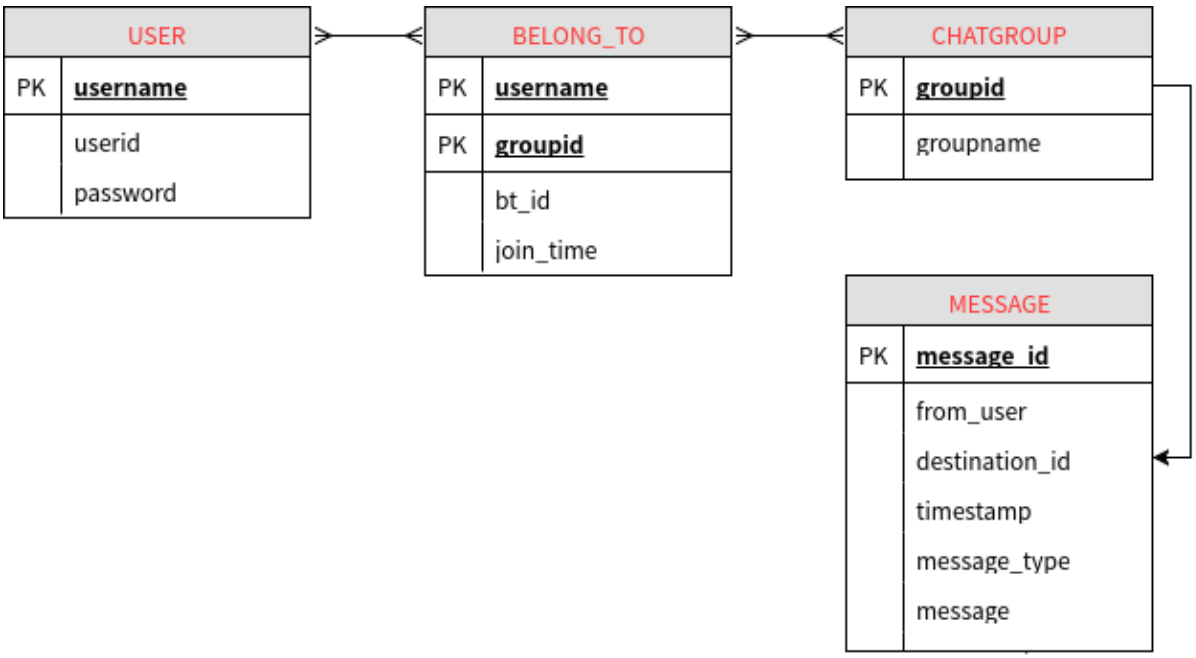
Figure 4

◎資料庫

ayou7995_chatroom 資料庫（MARIADB）

架構

資料庫最初設計的目標是作出羣組聊天的功能，其次爲支援文字傳輸之外的圖片以及音檔傳輸，但這次進度只有做到群組聊天，圖片以及音檔傳輸還沒來的及完成。底下的關係圖是資料庫的儲存架構，基本的想法是把所有聊天室假想成羣組(GROUP)，不論是兩人還是多人聊天，全部都是羣組的概念，而每一個羣組都有屬於自己唯一用來區別的值：



(1) USER 表：儲存使用者的名稱(username)和密碼(password)，其中，這裏把 username 當做是唯一的值（Primary Key），所以重複的使用者名稱是不允許被重複註冊的。userid 是一個會自動增加的 key，這裏沒有實質的功能。

(2) CHATGROUP 表：儲存所有羣組的 groupid 以及其名稱(groupname)。這裏的 groupid 是把某一羣組的所有使用者的名成接起來後，通過一個 hash function 得到的一個值，下面有些函式會用這個部分。Default 所有人都屬於 gorupid=0(布告欄：all)的這個羣組。

(3) BELONG_TO 表：USER 表以及 CHATGROUP 表的關聯(Relation)，用以儲存任意使用者屬於哪個羣組，也可以說是聊天的對象(包含一對一和一對多)。bt_id 是一個會自動增加的 key，這裏沒有實質的功能；join_time 則是用以記錄使用者以及羣組是在什麼時候被綁在一起。

(4) MESSAGE 表：所有聊天記錄儲存的表。每個 record 包含某一使用者(from_user)在某一時間(timestamp 傳了某一條聊天記錄(message)到某一羣組(destination_id)；message_type 是原本要做到圖片和音檔傳輸而事先設計的，不過因爲這次還沒做出來，所以這一欄全部都是 null。

函式

把 server 會需要 request 資料庫的部分包成函式給 server 使用。

login()：server 端傳入使用者輸入的 username 和 password。如果傳入的 username 不在 USER 表中，則呼叫 add_new_user()處理註冊；如果傳入的 username 已經存在，則判斷 password 是否正確，如果相同則回傳 True，反之則回傳 False。
>>>> add_new_user()：把 username 以及 password 註冊進 USER 表。

add_message()：server 端傳入傳送者的 username、某個羣組的所有人的 username、時間以及訊息。如果是傳入布告欄(all)，則 groupid=0，並把資料傳入 MESSAGE 表；如果不是，則透過上面說過的方法找出對應的 groupid，然後一樣把資料存入 MESSAGE 表。

update_groupname()：更新一個羣組的名稱。
begin_chat()：使用者開始與某一羣組聊天。server 端傳入某個羣組中所有的 username，如果這個羣組曾經被創建過，則呼叫 retrieve_prev_message()，把所屬 groupid 的訊息按照舊到新的順序傳給 server；相反的，如果這個羣組不存在，則呼叫 add_new_group()和 bind_user_to_group()。
>>>> retrieve_prev_message()：按照舊到新的排序把訊息回傳給 server。
>>>> add_new_chatgroup()：把一羣 username 對應的 groupid 以及羣組名稱存入 CHATGROUP 表。
>>>> bind_user_to_group()：把這個羣組的所有人和 groupid 綁在一起，存入 BELONG_TO 表，方便後續搜尋哪個 username 存在於哪個羣組中。

query_belong_group()：使用者初始 login 後，server 端會傳入 username，資料庫會把所有羣組人數大於兩人，同時 username 存在在這個群組的 groupname 以及羣組的 usernames 回傳給 server。