**Final Project Report**

Group 6

112550031陳炤宇、11255039李宗諺、112550040游子慶

112550044 吳宇藤、112550191許書瑋

* Main idea

庫存管理是銷售業會面臨的重要課題，無論企業規模大小，都常常遇到缺貨、庫存過剩或是各類成本驟增的問題。因此，我們希望藉由這次的期末專題，深入練習如何有效進行庫存分析。我們將建立相關的資料庫、設計便於查詢的方式，以達成更有效的庫存管理，並提供上游廠商討論區對產品進行評論，進而模擬企業的營運效率的優化過程。

* Data
  1. Source  
     此專案使用的資料集來自Kaggle，資料是來自一家酒類上游公司，內容包括酒品的庫存分布和下游廠商的訂單紀錄。庫存紀錄涵蓋了酒品的容量、品牌、價格及存放的倉庫位置；而在訂單紀錄部分，則記載了下游廠商的編號、名稱、訂單日期、提貨日期、付款日期和訂單金額等資訊。

Dataset連結: [Inventory Analysis Case Study](https://www.kaggle.com/datasets/bhanupratapbiswas/inventory-analysis-case-study/data)

* 1. Our Data  
     因為我們希望打造一個可以搜尋酒品並顯示商業系統相關資訊的網站，因此選取了我們認為較具實用性的欄位，透過酒品名稱進行查詢，並與其他表格連結，最終匯總出「剩餘庫存、進貨資訊、銷售資訊」等完整的欄位。
  2. Tables
* Database

1. Data Language  
   我們使用mysql當作後端，並用python的flask網路框架來完成前後端對接。
2. Data Maintance and Connect Database to the Application  
   **新增資料**  
   在新增資料方面，包括新增使用者與新增貼文，兩者皆使用相同的流程。應用程式透過連接資料庫，執行 MySQL 的 INSERT 查詢來完成資料新增操作，如下圖所示。

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

**更新資料**

更新資料包含更新貼文，如下圖，當應用程式收到 POST 請求，也就是當使用者要提交修改後的貼文內容，就執行sql來更新指定 id 的貼文標題與內容；否則，當應用程式收到GET請求(此函式前面限制只能使用POST和GET)，也就是網頁要載入欲編輯的貼文內容，就從資料庫查詢指定 id 的貼文資料並顯示至編輯頁面。

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

自動產生的描述

**刪除資料**

刪除包含刪除貼文，如下圖，應用程式刪除指定 id 的貼文並更新資料庫，執行後顯示成功或錯誤訊息。   
一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

**搜尋資料**

搜尋功能為搜尋酒名跳出相關查詢解果，sql和前後端方面這兩步驟比較長所以分兩段落講，先講sql方面，我們透過跑sql腳本來完成搜尋功能，由於sql腳本較長，因此這裡只用文字說明，實際code可以參考page\_2.sql檔案。

此 SQL 腳本用於從五個表中提取酒品相關的完整資訊，並篩選符合特定酒品名稱的結果（如 "Belvedere Vodka"）。首先透過 `@inputDescription` 設定搜尋的酒名，然後使用 `JOIN` 和 `LEFT JOIN` 分別將 `EndInvFINAL12312016`、`2017PurchasePricesDec`、`PurchasesFINAL12312016` 以及 `InvoicePurchases12312016` 進行關聯。查詢的欄位包括剩餘庫存、城市、售價、進貨資訊以及發票支付日期。使用 `WHERE` 條件過濾關鍵字和非空值資料，確保結果完整且精準。結果可直接用於生成匯總表或進一步分析。

在前後端方面，我們透過輸入`@inputDescription` 設定搜尋的酒名，並跑sql腳本，最後拿sql的結果來顯示，如下圖。

一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

1. Exception Handling

此專案的每個函數都具備完整的檢查機制，能在更新失敗、連線失敗等異常情況下，於前端顯示對應的錯誤訊息。同時，所有操作均確保資料庫連線會被正確關閉，避免資源浪費，並且在出現錯誤時停止後續資料庫操作，確保數據完整性；在輸入方面也有防止sql injection，透過使用下圖語法，讓使用者輸入被視為資料存放，而不是直接放進sql語法裡。

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

* Application

1. Main page

此為我們網站進入後的第一個畫面，如圖Fig. a1所示。我們將我們的搜尋系統建立在首頁以方便，上方為導覽列，方便使用者前往討論區、關於我們以及登入系統。為避免使用者以訪客身分在此網站上進行查詢，所以我們會先判斷使用者的登入狀態，如果使用者未登入 (即畫面右上角的狀態顯示Log in )，如圖Fig. a2，我們會跳出請登入的訊息，並跳轉至登入頁面。已經登入的使用者，就會利用使用者輸入的資料進行搜尋，如果有搜尋到酒名，就會顯示酒類資訊，如圖Fig. a3，如果沒有的話則跳出無結果，如圖 Fig. a4

CRUD: R:讀取使用者輸入的酒名

查詢應用: 查詢使用者的登入狀態、利用使用者輸入的酒名進行查詢

錯誤處理: 如果未登入就跳轉登入頁面

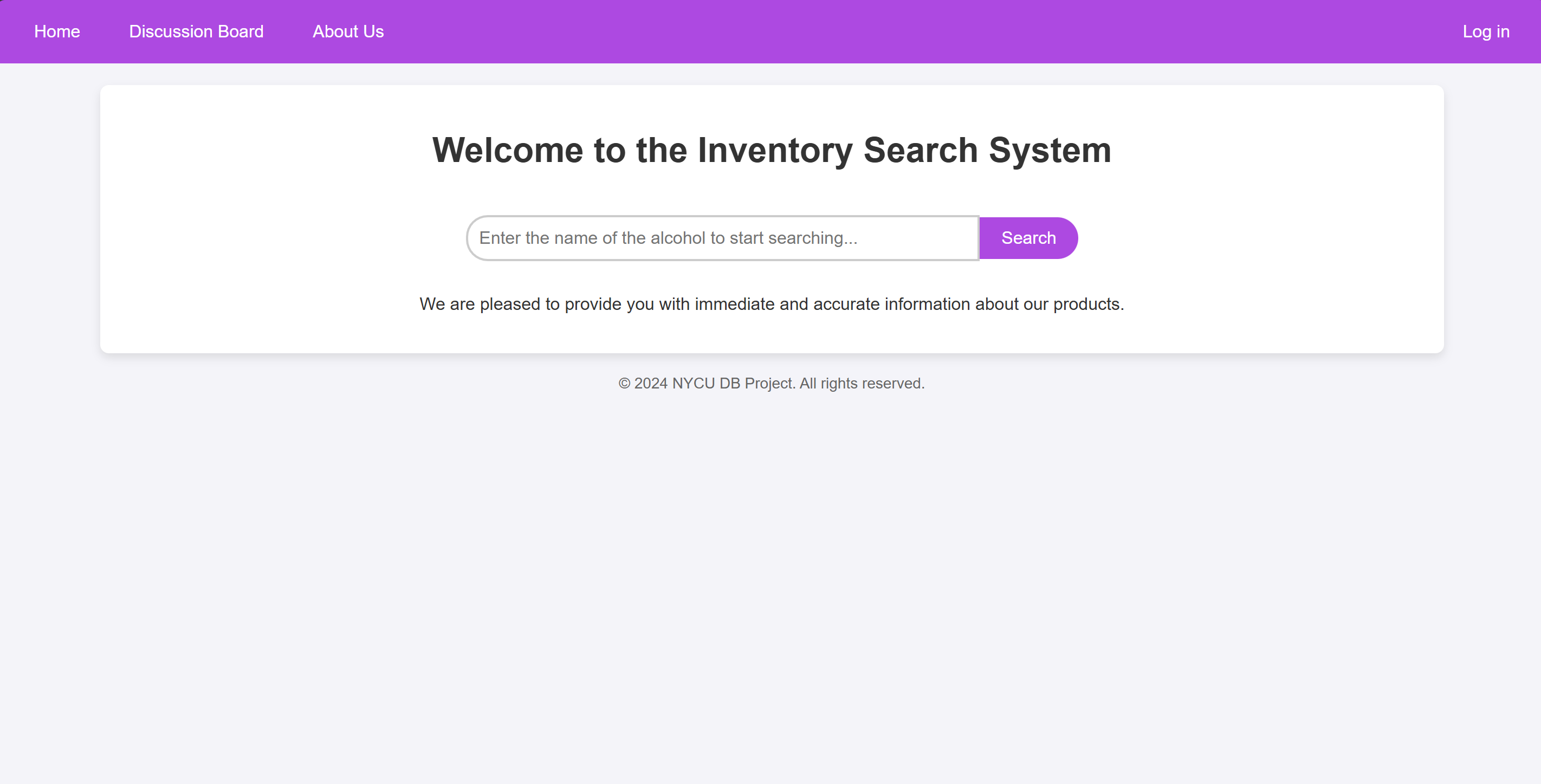


Fig. a1

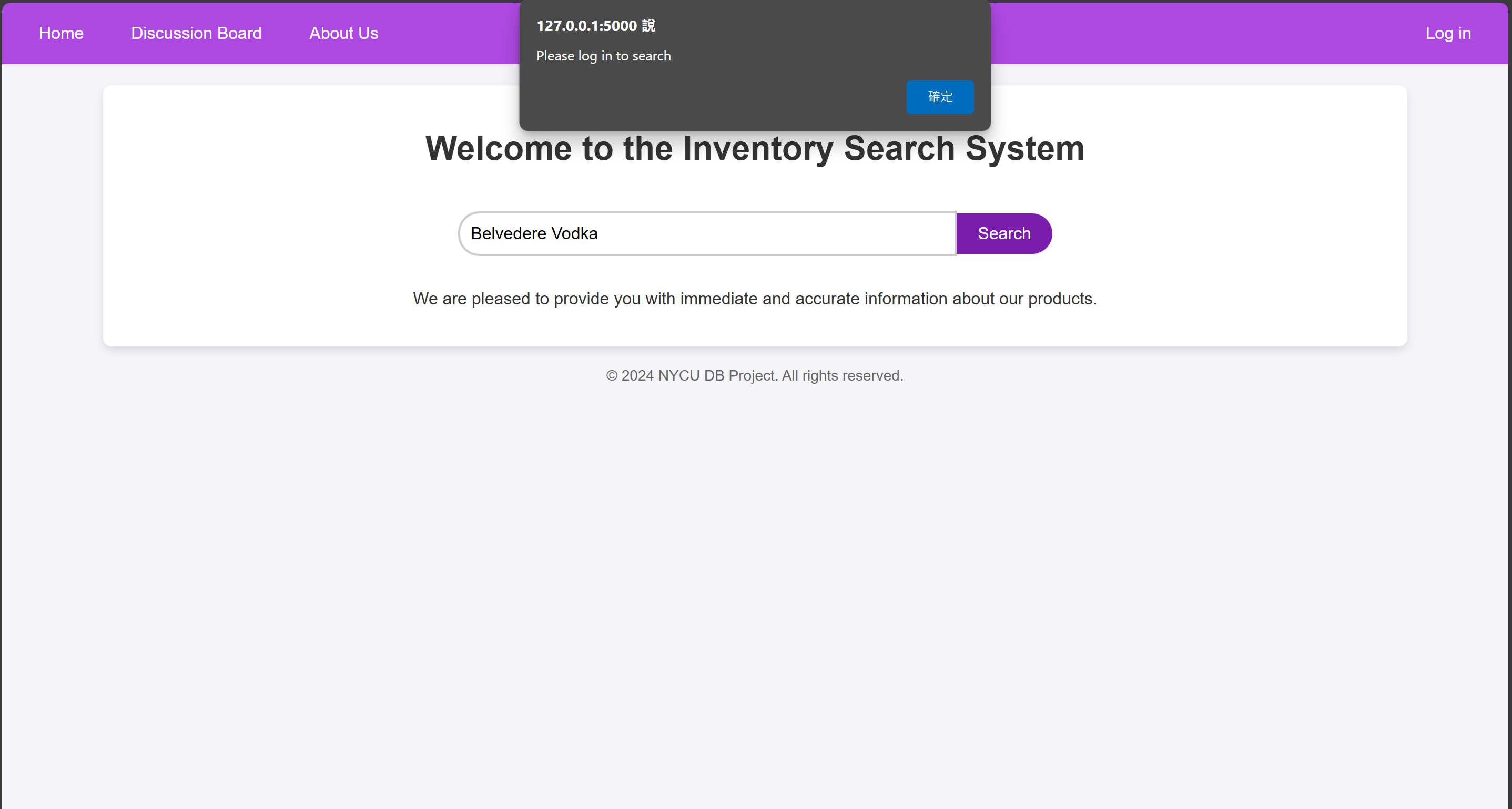


Fig. a2

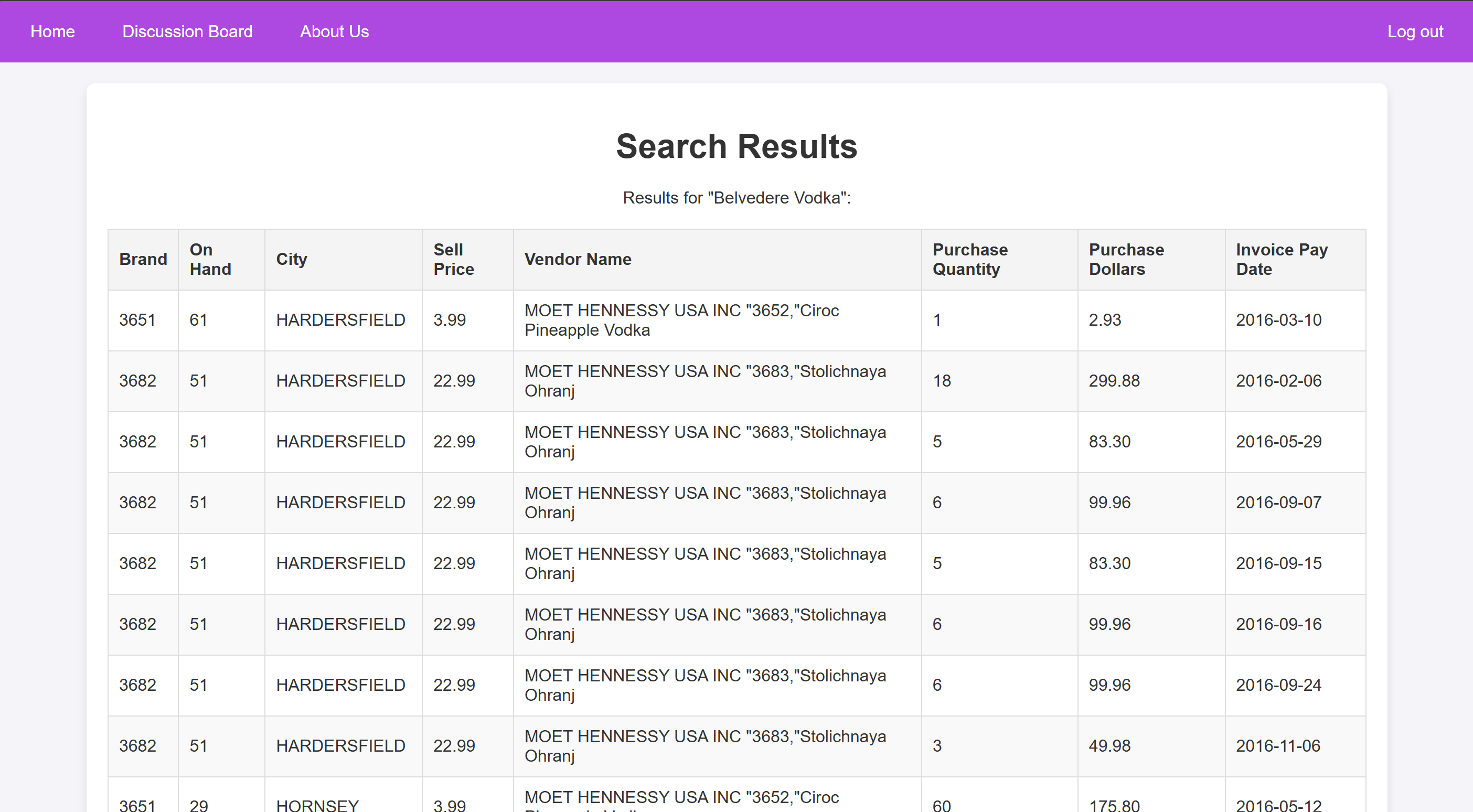


Fig. a3

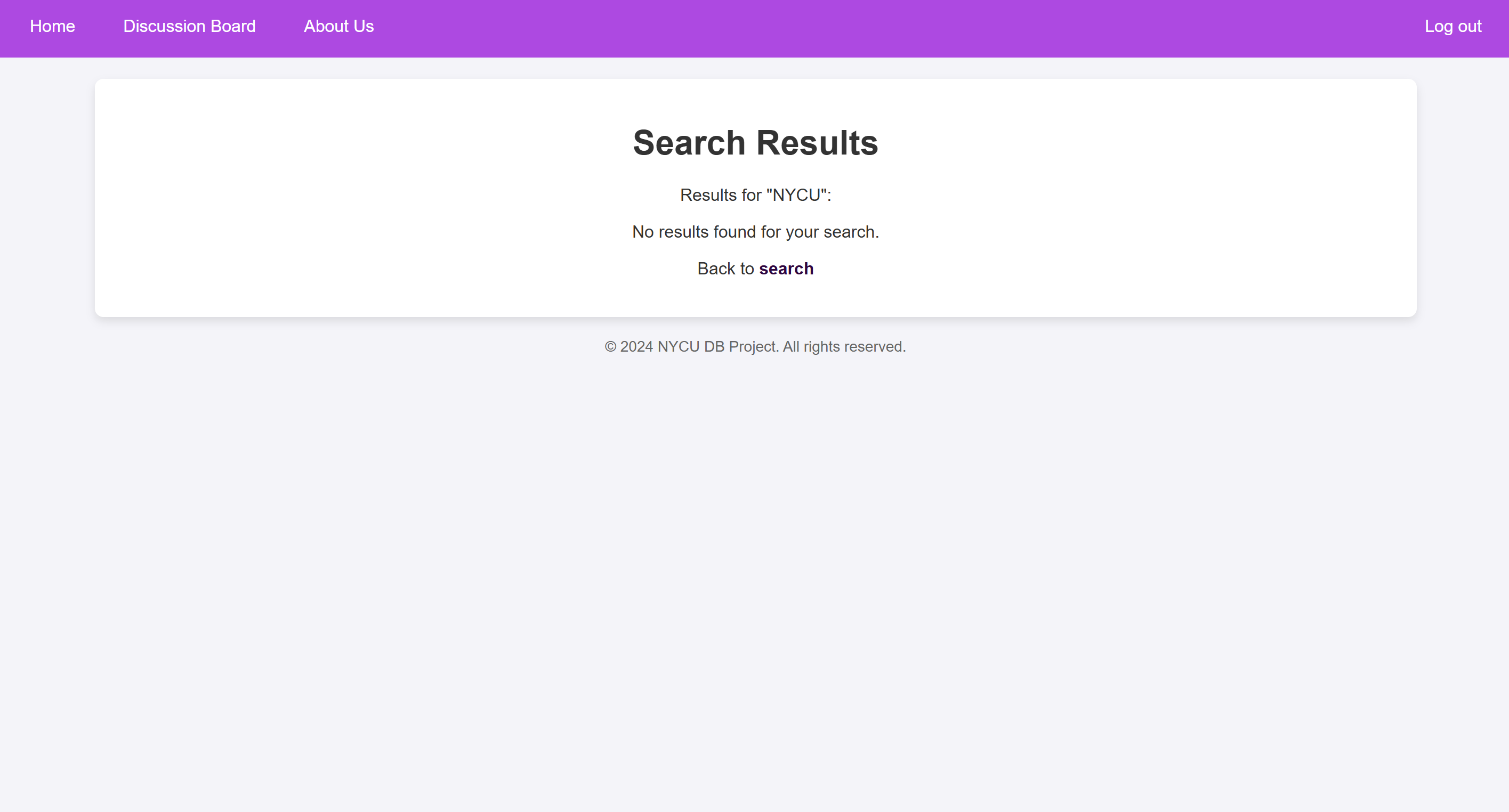


Fig. a4

1. Login page

由使用者點擊導覽列的登入或是由系統自動跳轉登入後，可進入至我們的登入系統，如圖Fig. b1。此時帳號密碼為尚未輸入的狀態，藉由使用者輸入正確帳密後，即可進行登入，並跳轉回剛剛登入前的介面(比如如果是由討論區點擊登入且登入成功的話，系統會跳轉至剛剛討論區的畫面)，此時右上角的登入狀態從原先的Log in 轉變成 Log out，如圖Fig. b2。但如果輸入的帳密有誤的話，會跳出登入失敗的訊息，如圖Fig. b3。

CRUD: R: 讀取使用者帳密U: 更新使用者的登入狀態

查詢應用: 查詢使用者帳密是否在資料庫中

錯誤處理: 當使用者輸入錯誤的帳密，會跳出登入失敗的訊息

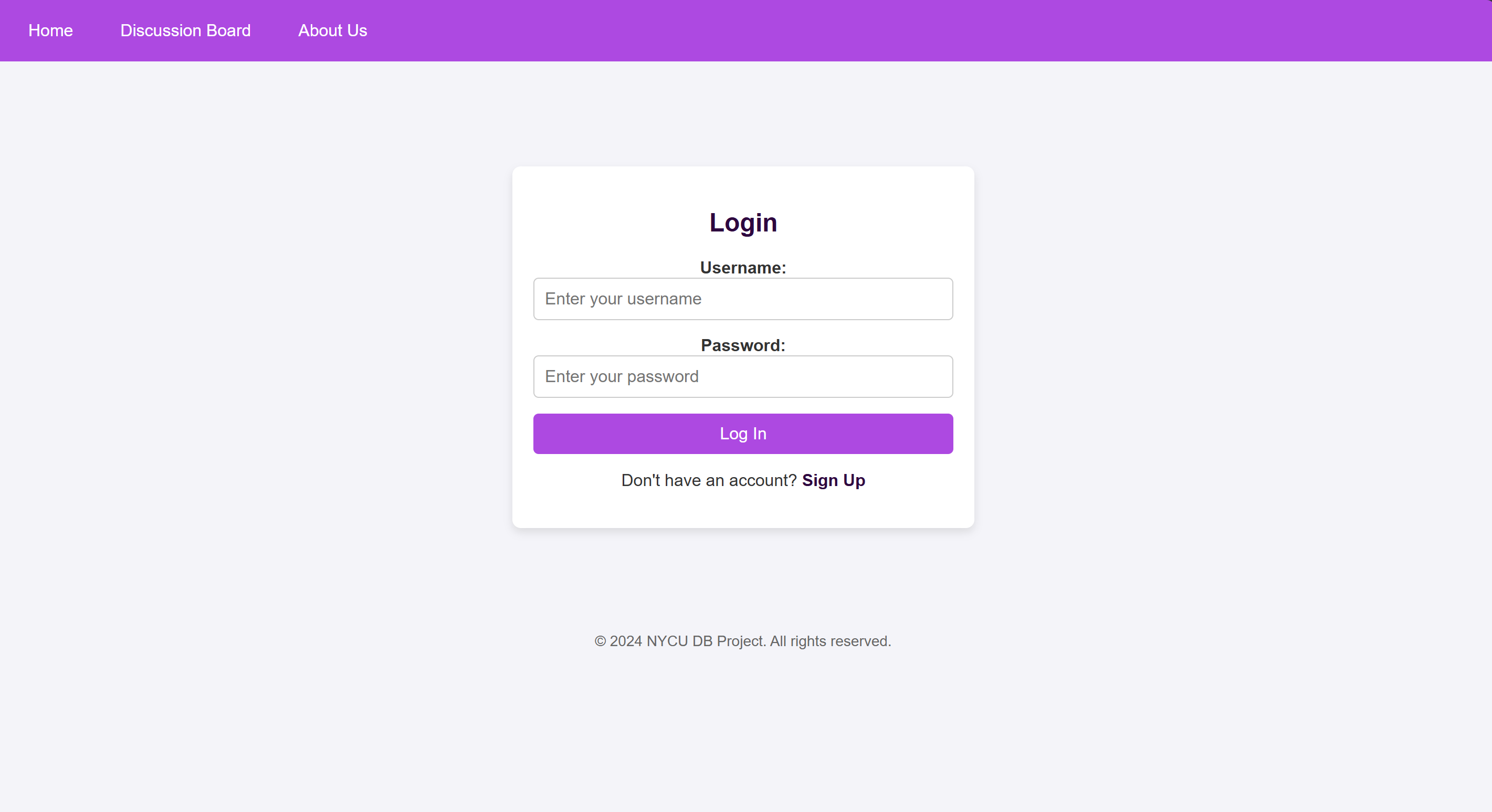


Fig. b1

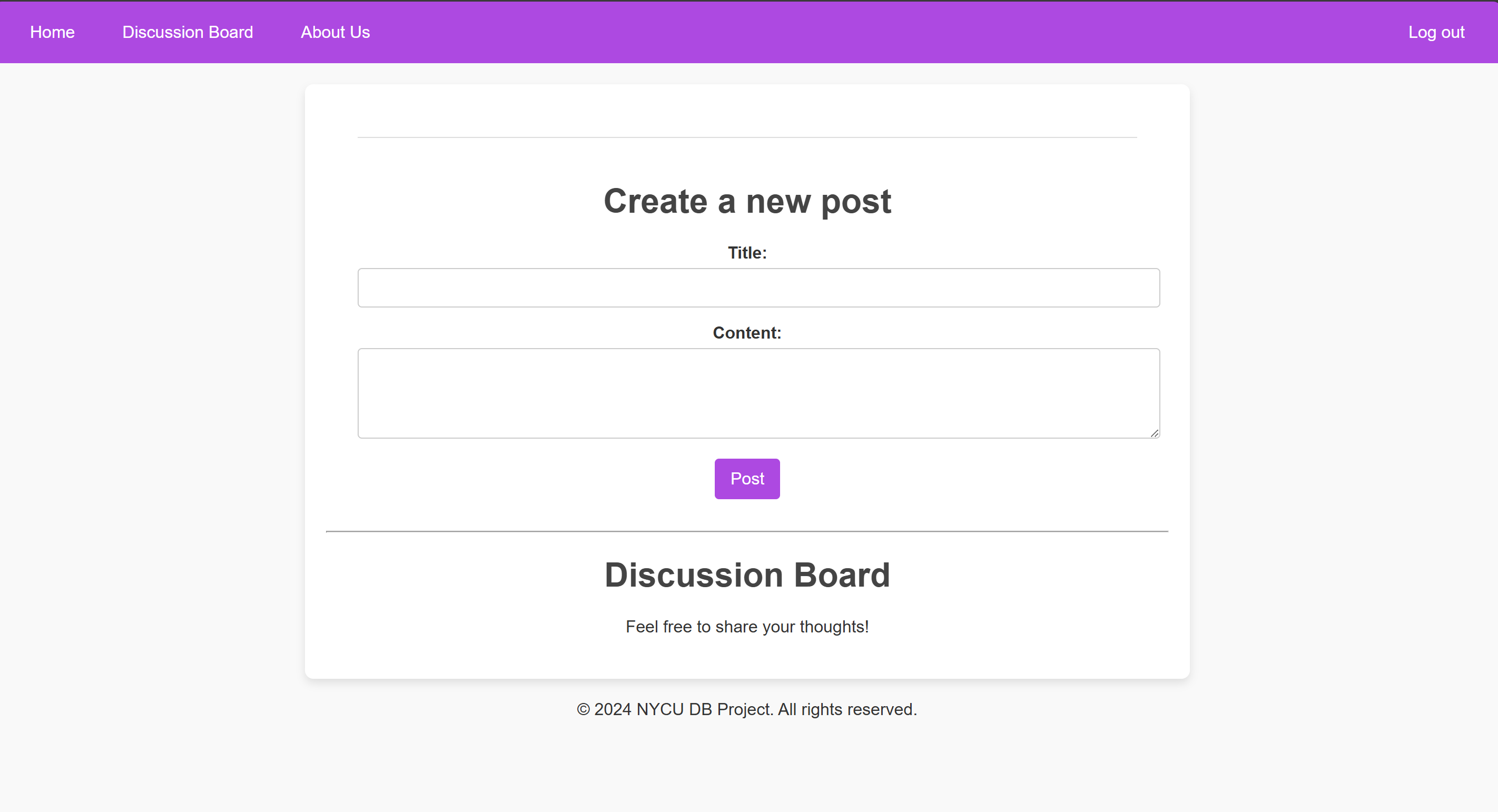


Fig. b2

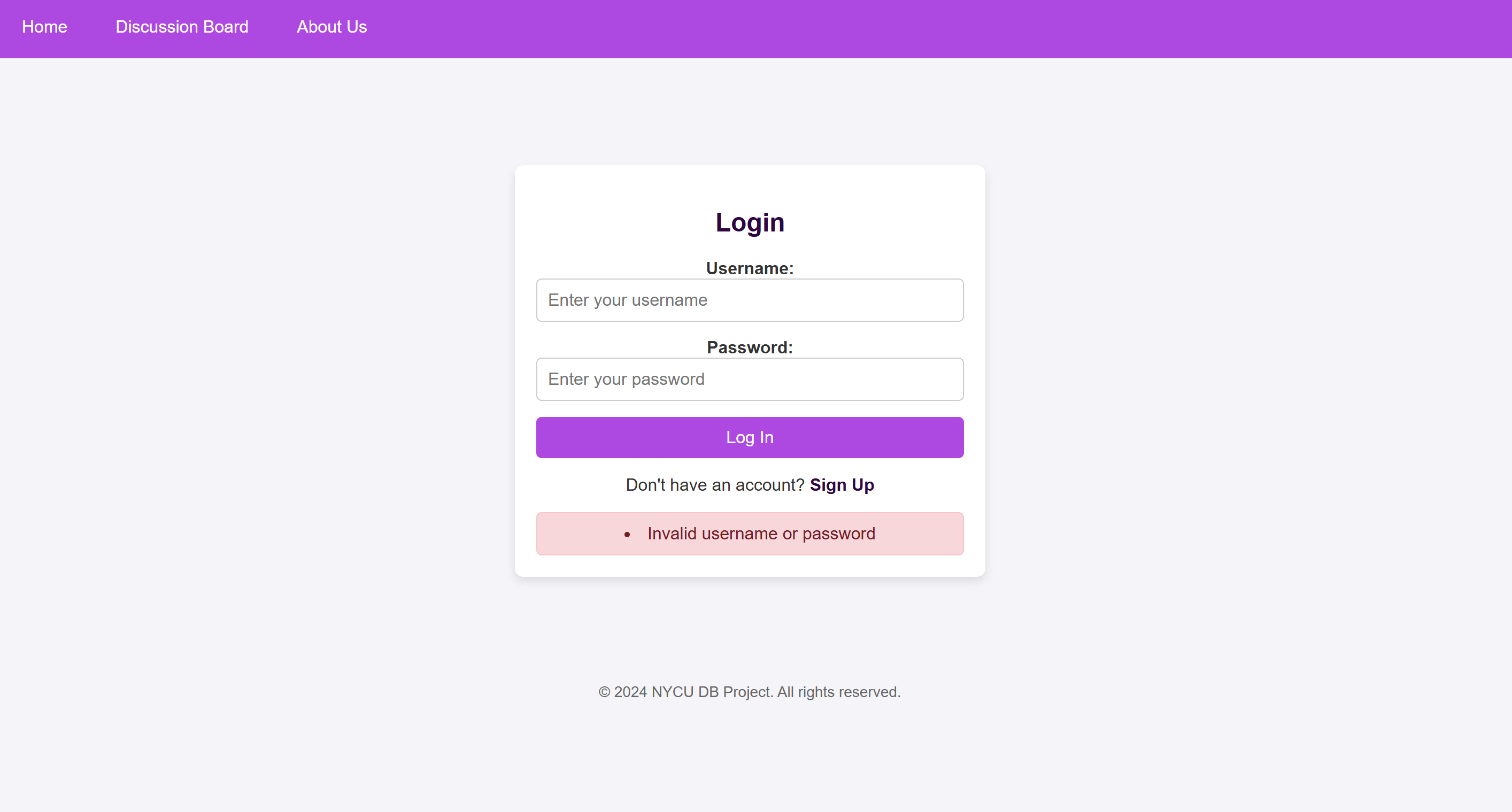


Fig. b3

1. Sign up page

如果使用者沒有帳號的話，可以從Log in page點擊下方的Sign up，並跳傳至Sign up page，如圖Fig. c1所示。在此我們可以創立我們的帳號密碼，如果創立成功，會自動跳轉至Log in page，並跳出創立成功的訊息，使用者只須輸入剛剛設立的帳密即可登入，如圖Fig. c2所示。但如果帳號已經創立了，且使用者嘗試使用同個名字在創建的話，系統會跳出創建失敗的訊息，如圖Fig. c3所示。

CRUD: C創建使用者帳密於資料庫中 R讀取使用者輸入的帳密

查詢應用: 查詢資料庫的所有帳號

錯誤處理:如果查詢到已經有重複的帳號，跳出創建失敗的訊息

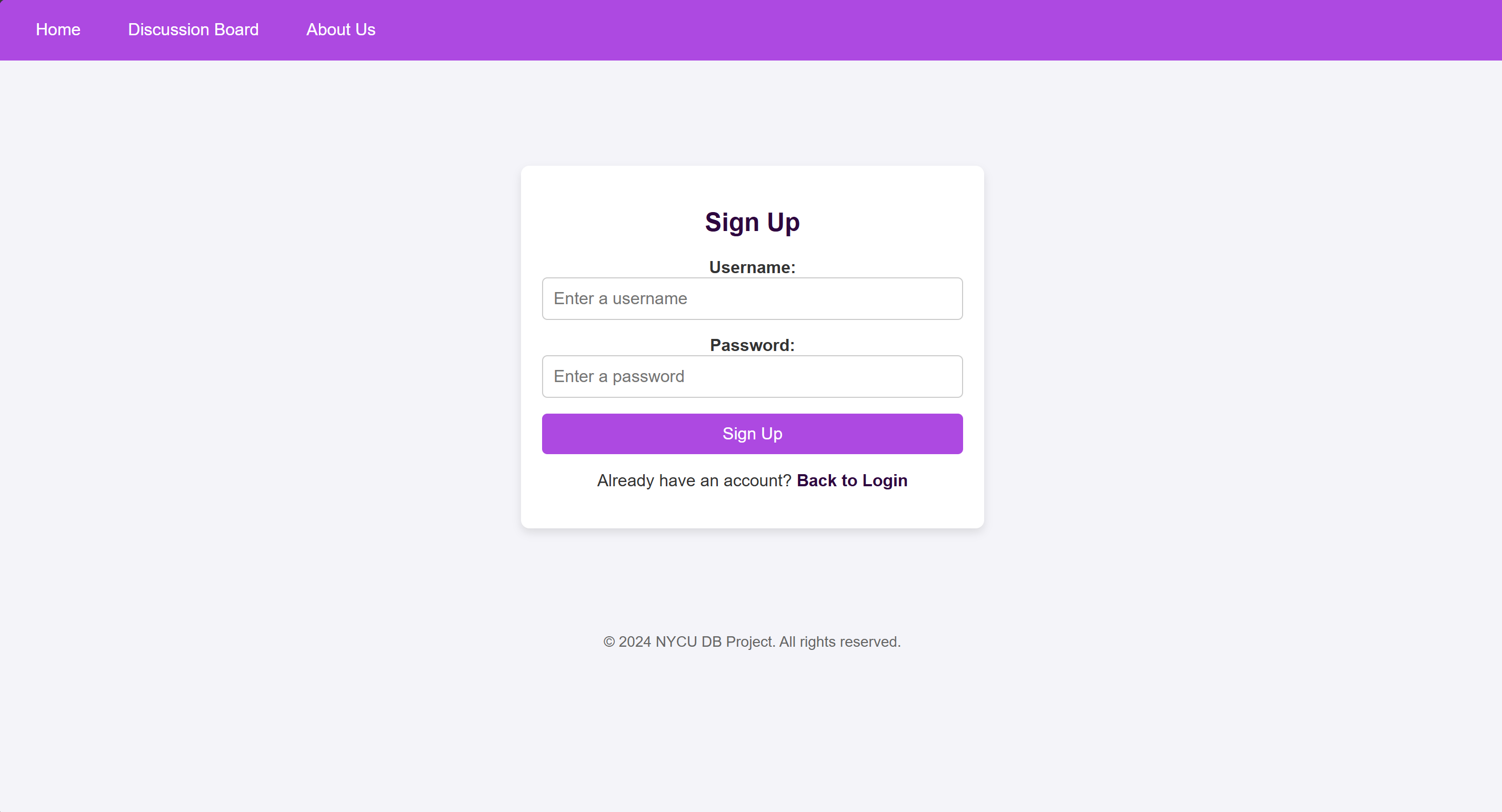


Fig. c1

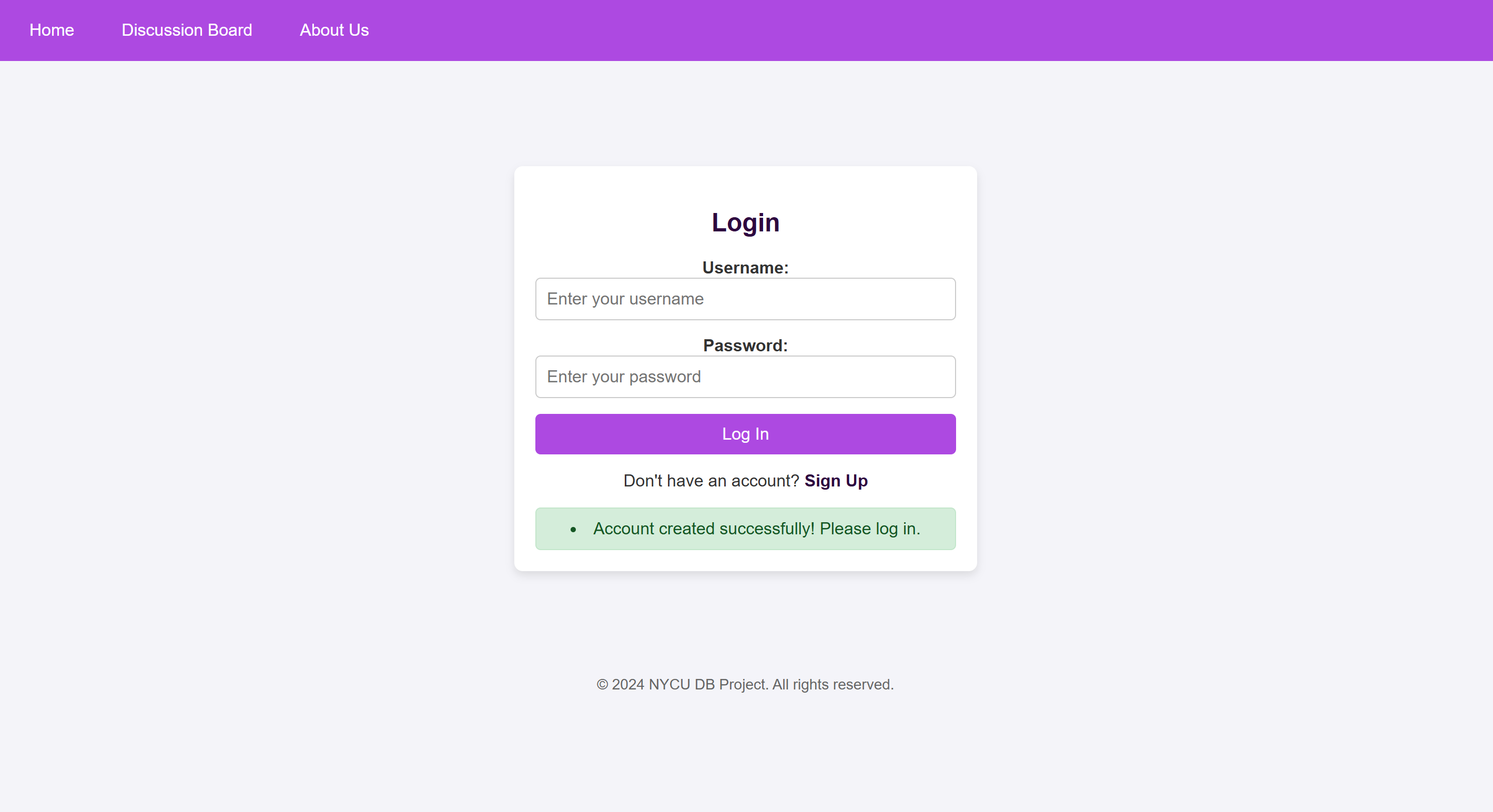


Fig. c2

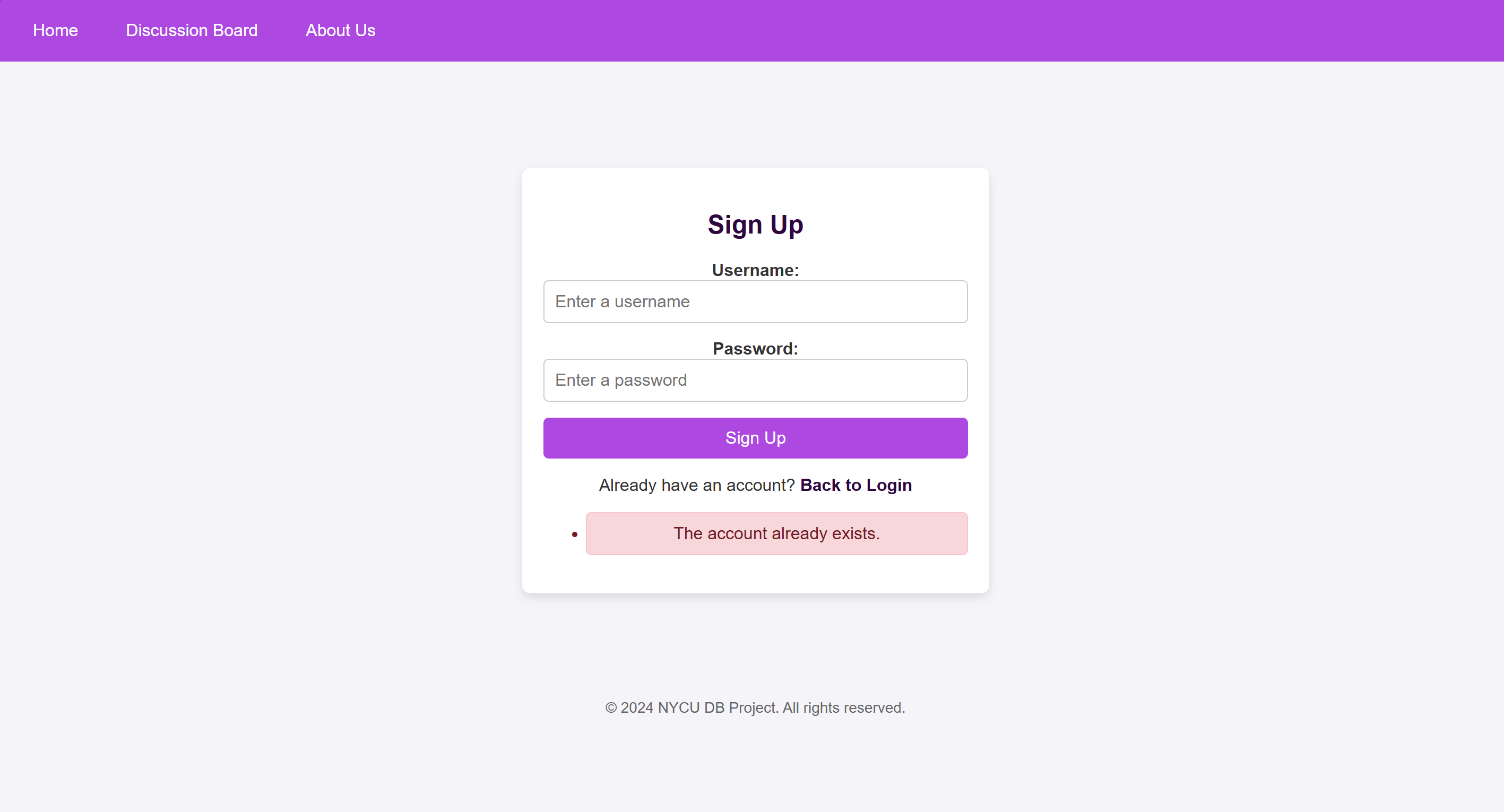


Fig. c3

1. Discussion board page & edit post page
2. 初始畫面(未登入)

使用者點選導覽列上的Discussion board 即可前往討論區。其中未登入的使用者無法對於討論區做任何的動作，僅可察看討論區上的貼文，如圖Fig. d1。

CRUD: R:讀取post的資料

查詢應用: 查詢所有post裡的貼文，並顯示在網頁上

錯誤處理: 對於未登入的使用者，限制其操作並僅允許察看貼文

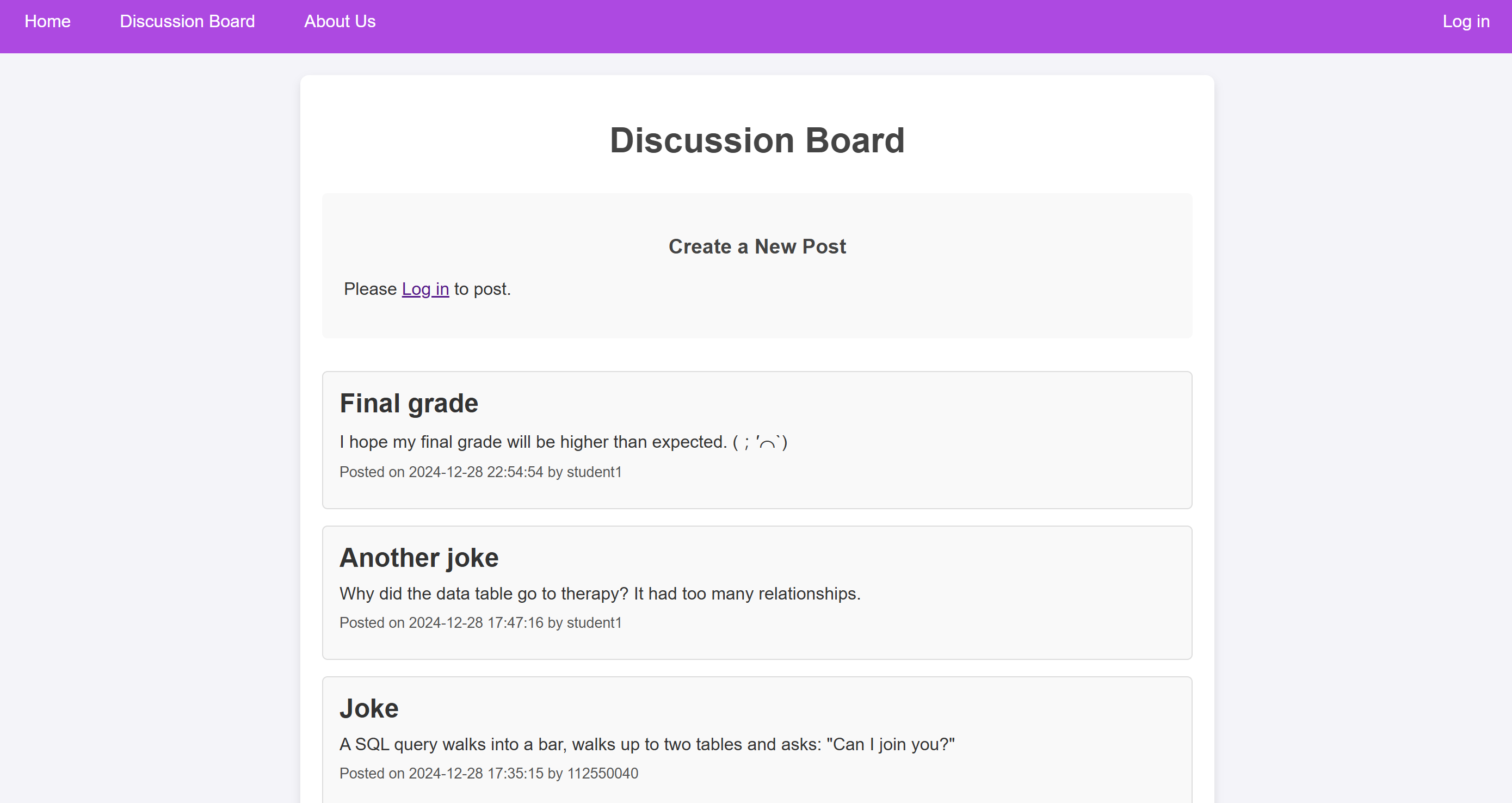


Fig. d1

1. 初始頁面(登入後)

當使用者登入後，討論區中的創建貼文就開放使用，如圖Fig. d2。使用者可以在討論區中輸入該貼文的標題、內容，輸入完成後按下送出即在資料庫創建好了貼文內容，並顯示在網頁上面，如圖Fig. d3與Fig. d4。

CRUD : C:使用者可以自行創造貼文 R: 當使用者創造完貼文後可以顯示在網頁上

查詢應用: 查詢資料庫現有的貼文

錯誤處理: 使用者不得送出空標題與空內容

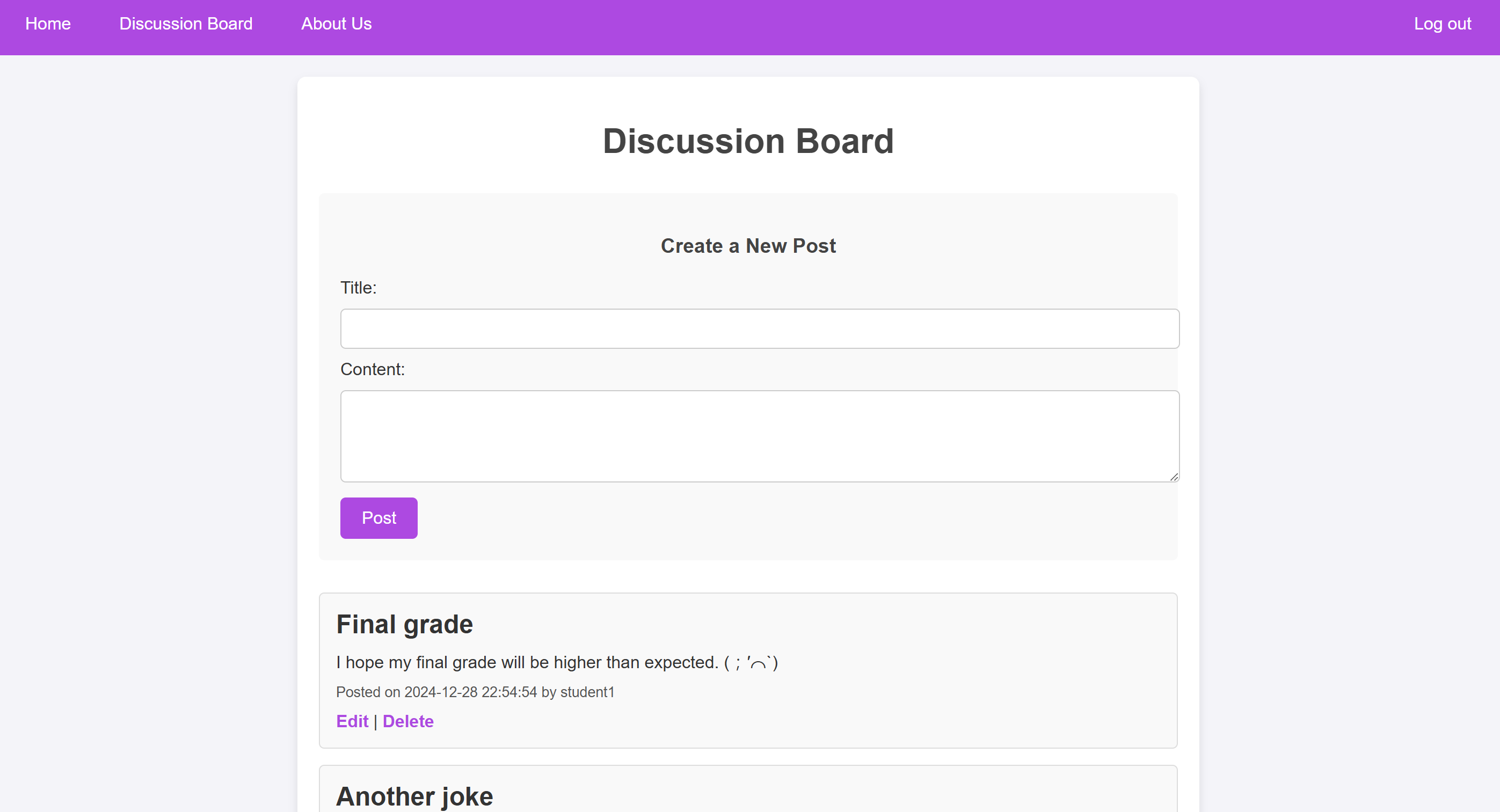


Fig. d2

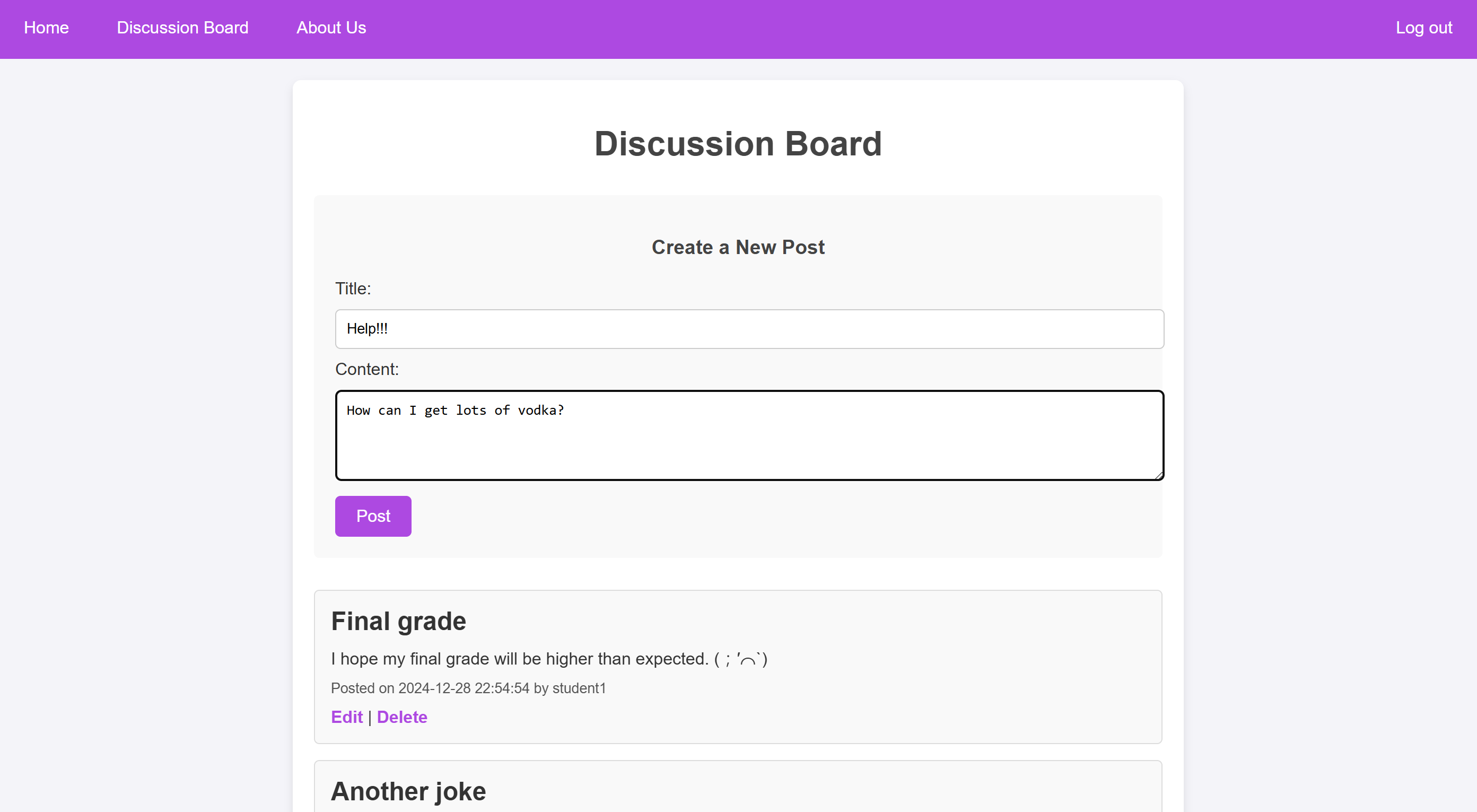


Fig. d3

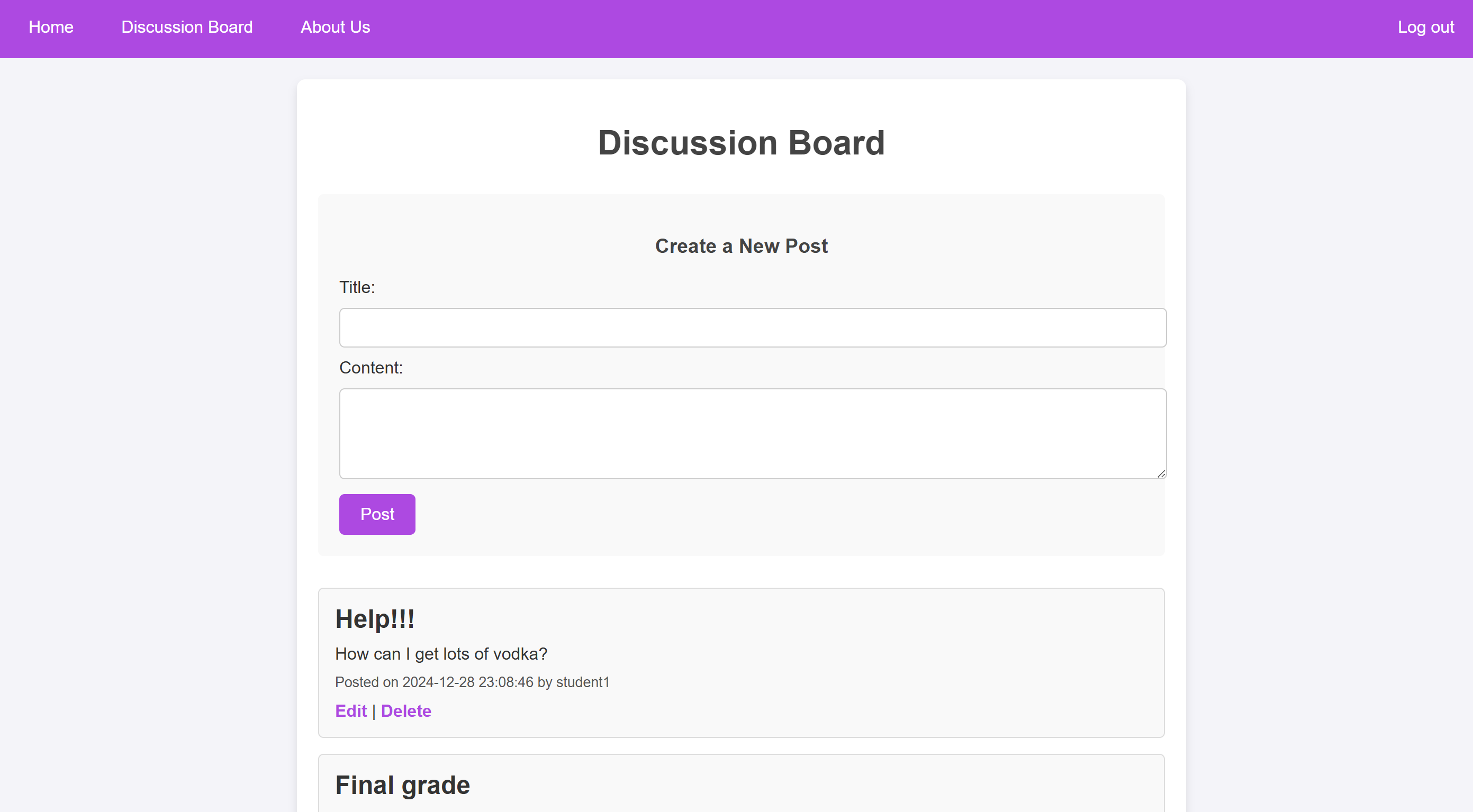


Fig. d4

1. 編輯頁面

在討論區中，使用者只能針對自己的文章進行編輯，如圖Fig. d5。如果想要編輯文章的話，僅需點擊該文章中的Edit，畫面會跳轉至編輯頁面，如圖Fig. d6。此時使用者如果編輯完標題、內容後，按下Update，畫面就會跳轉回剛剛的討論區的內容，而貼文也會隨之更新為使用者剛剛更新的內容，如圖Fig. d7。

CRUD: R:讀取文章現有的內容以及使用者更新的內容 U:將新的內容更新於資料庫中

查詢應用: 查看討論區的文章，系統須從資料庫中查詢

錯誤處理: 更新後的內容不能為空

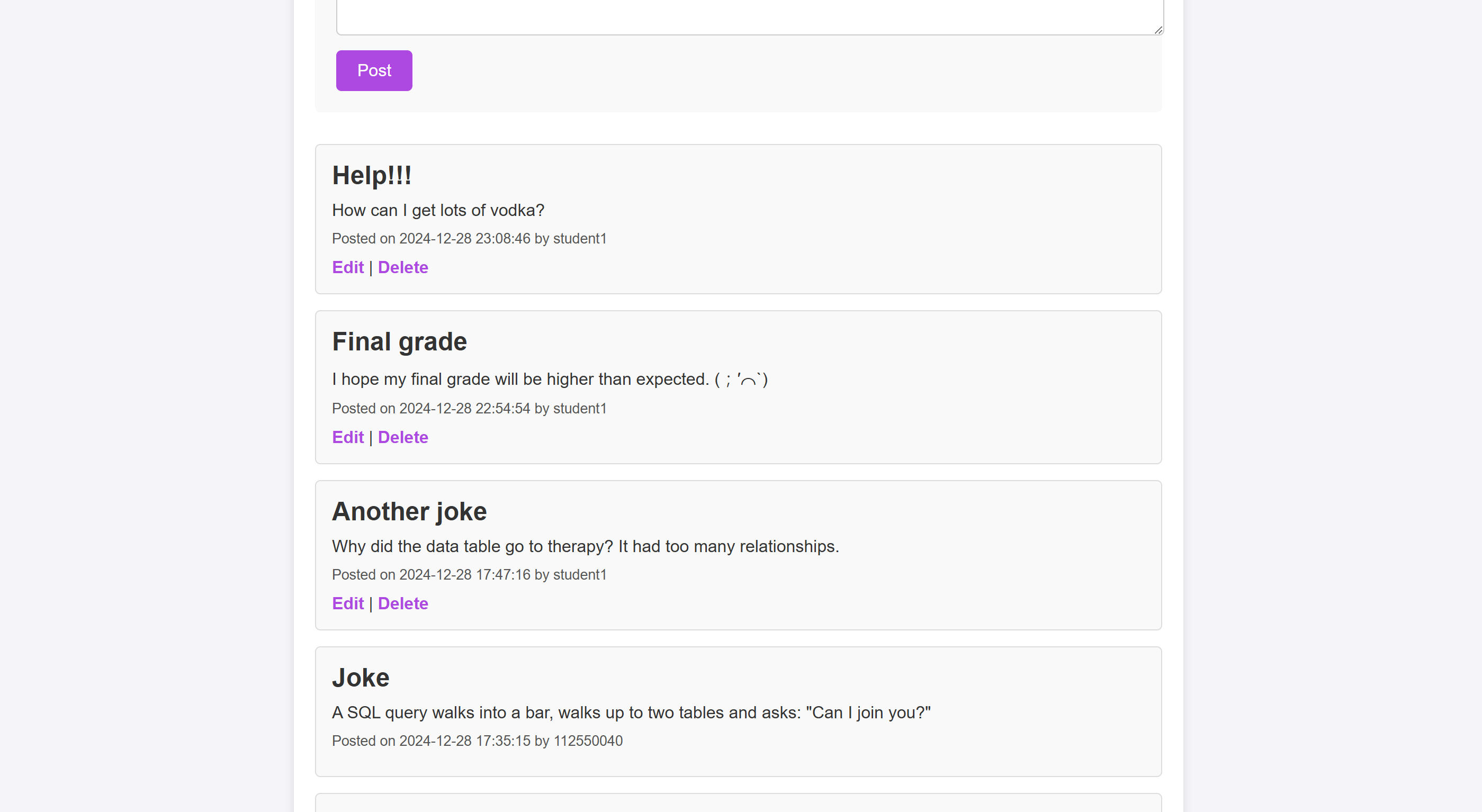


Fig. d5

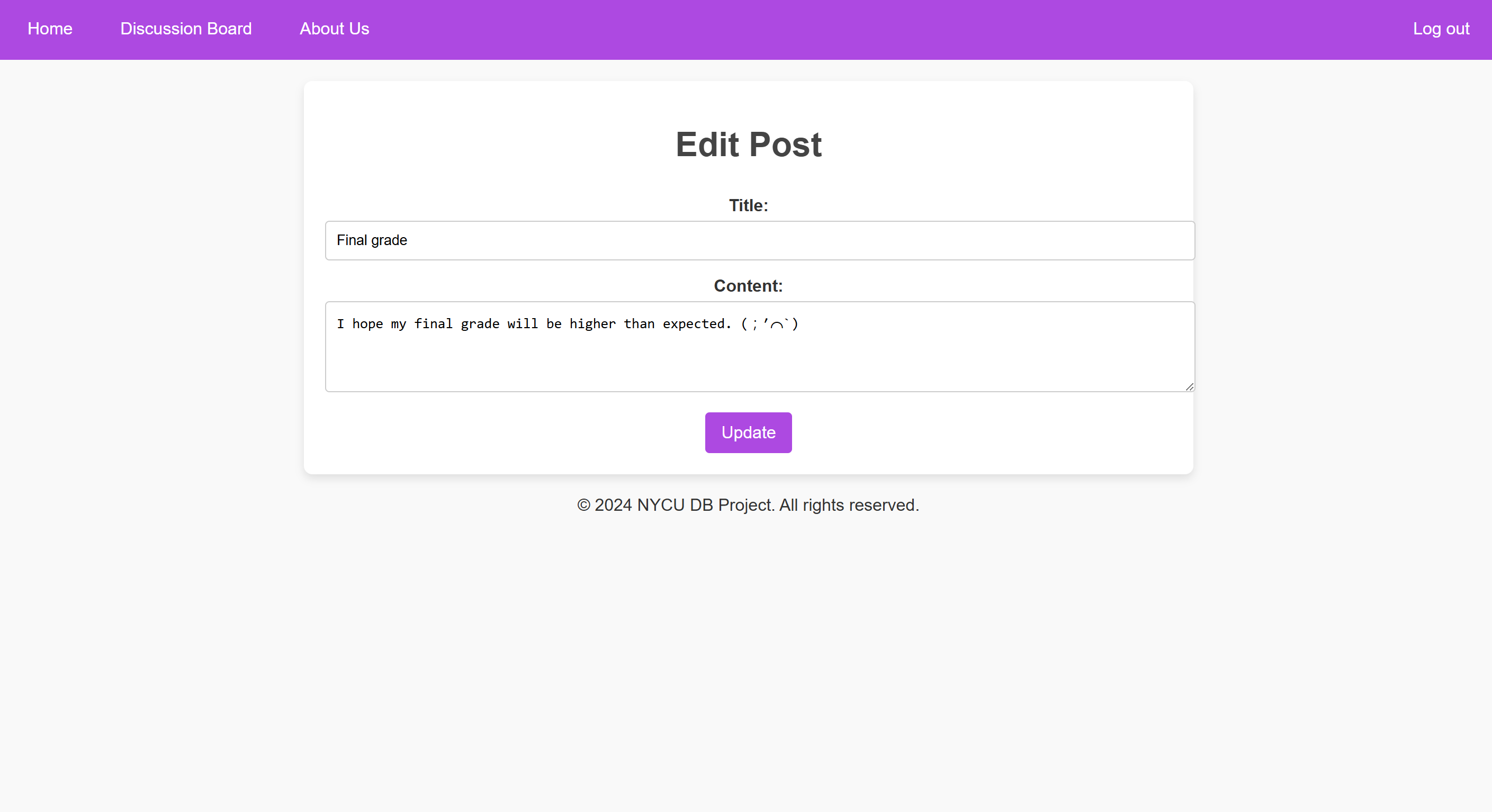


Fig. d6

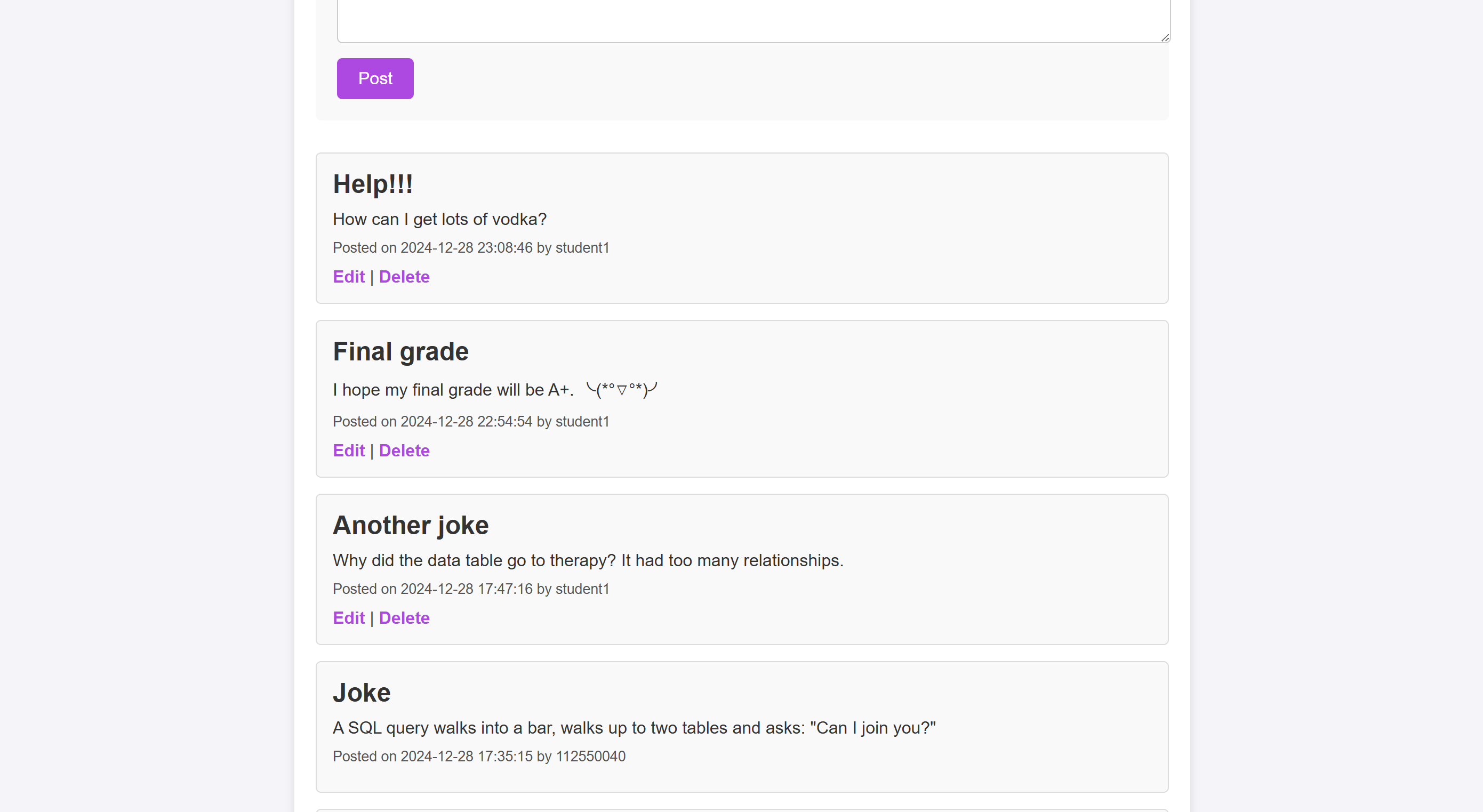


Fig. d7

1. 刪除頁面

與編輯頁面相似，使用者僅可以刪除自己的文章，故使用者可以點擊文章的delete按鍵，點擊完後資料庫裡的文章就會被刪除，網站的文章內容也不會顯示該文章，如圖Fig. d8 (刪除前)與Fig.d9 (刪除後)。

CRUD: D:使用者點擊delete就可以刪除資料庫中的對應文章 R:使用者刪除完成後，系統會更新並顯示剩餘文章

查詢應用:系統需要查詢資料庫中的現有文章

錯誤處理:無

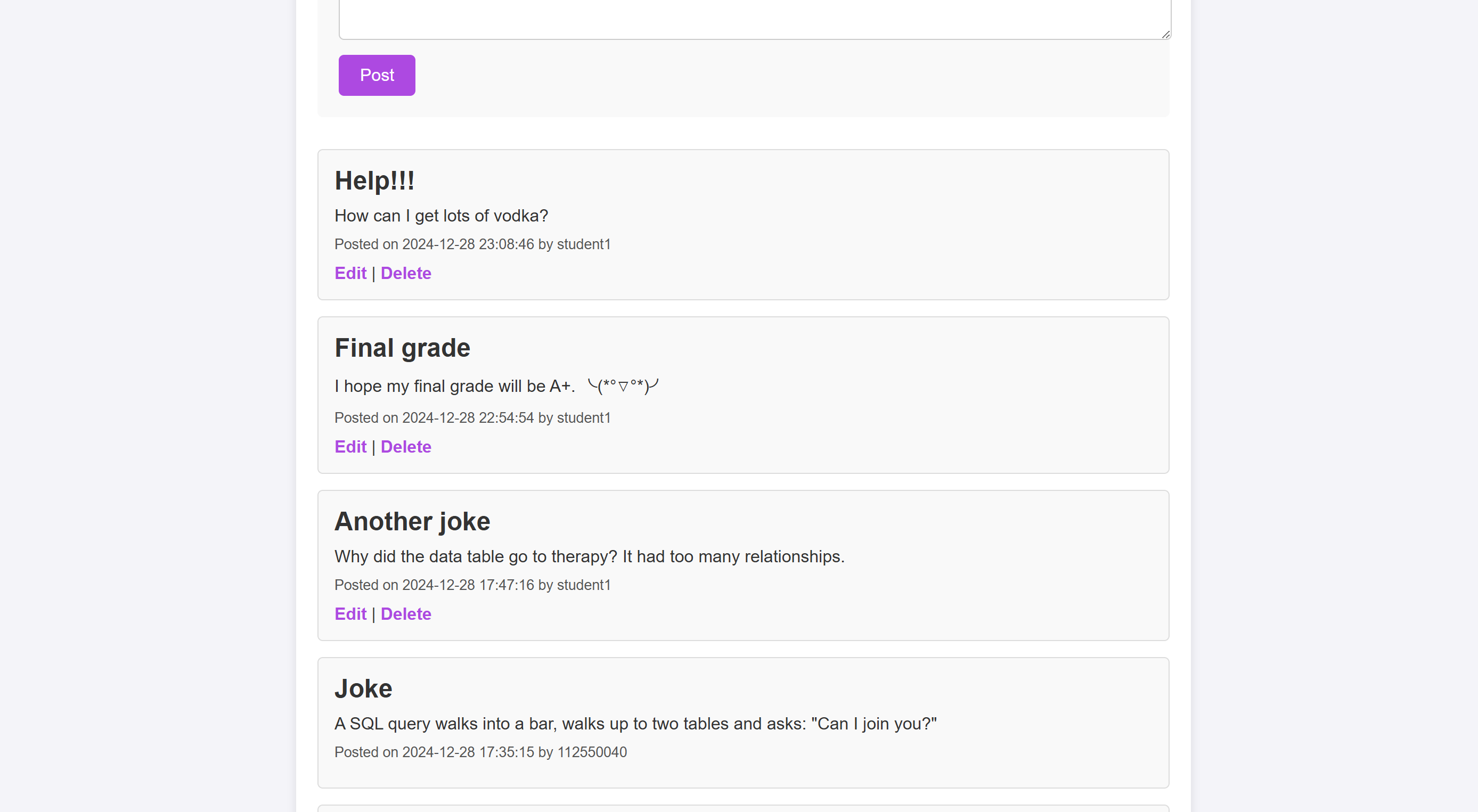


Fig. d8

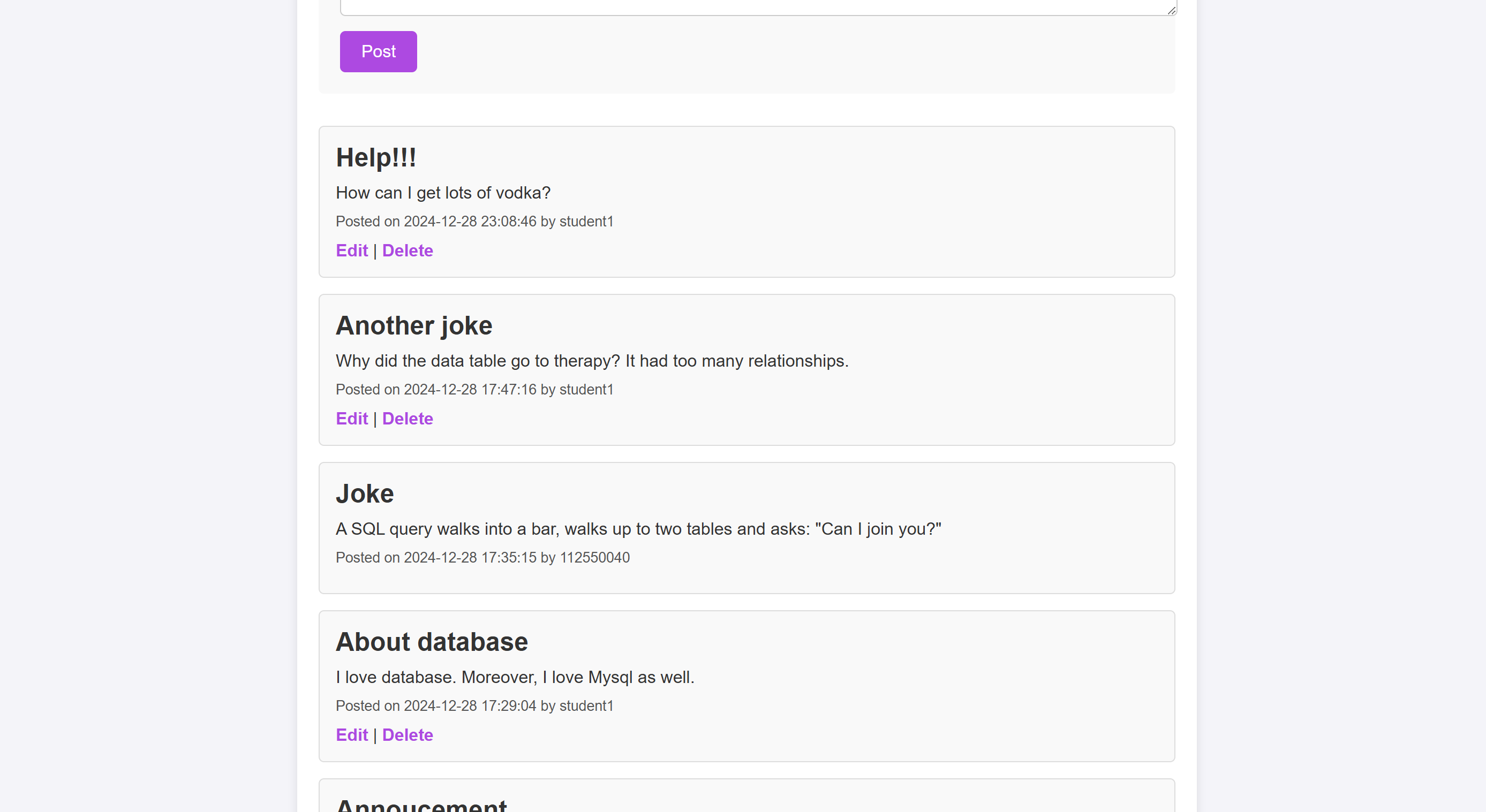


Fig. d9

1. About us page

這邊僅做簡單介紹，沒有特別的功能，如圖Fig. d10

CRUD:無

查詢應用:無

錯誤處理:無

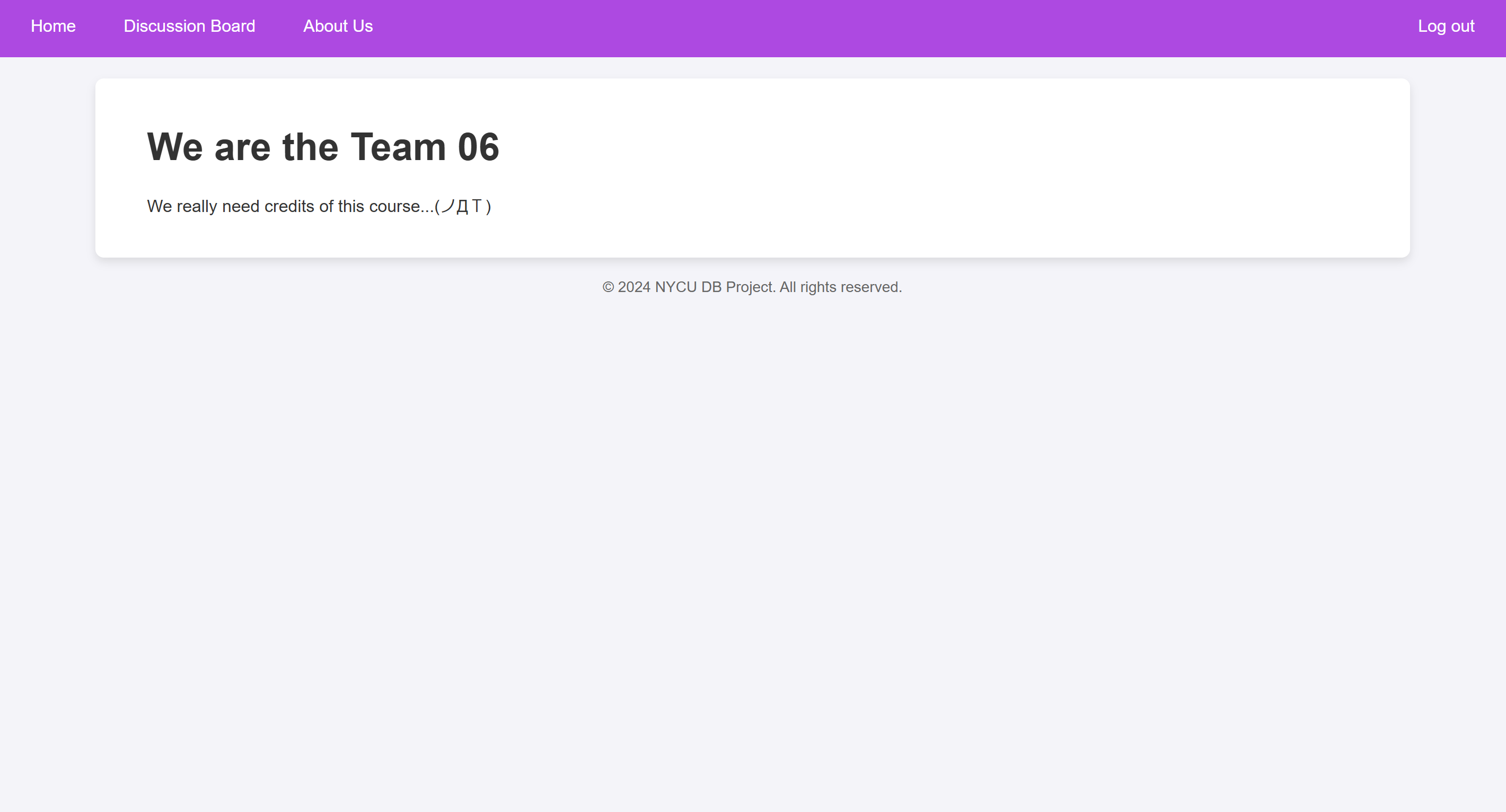


Fig. d10

* Attachment

Link to project code:  
<https://github.com/mikehsuhoodie/NYCU-Intro-to-Database>

Link to video:  
/\* to be added \*/

* Other problems

1. Draw the progress of your project
2. Expected Progress

/\* to be added \*/

1. Actual Progress

/\* to be added \*/

1. Problems we met
2. 如何結合資料庫與網頁設計出一個專案?  
   我們這組使用Mysql搭配Python+flask做前後端連結，並透過git進行共同編程，達成多人實作專案的目的。
3. 部分的csv檔的項目無法找到最有效的複合主鍵?  
   我們經過討論後決定自創一個遞增數列用來辨別項目
4. 原本的資料集並未描述各個欄位，我們並不知道每個欄位所代表意義是甚麼?  
   我們經過多方查詢、翻閱商業領域的文章，逐漸了解各欄位的意義
5. 部分資料在csv檔中的格式出現換行,空白,非可見字符等情況，導致MySQL CLI 的表格輸出對齊混亂  
   在SELECT時使用REPLACE(REPLACE(資料名, '\r', ''), '\n', '')，在查詢時清理字串
6. Contribution of each team member

|  |  |
| --- | --- |
| 陳炤宇 | 繪製schema、網站偵錯、撰寫報告 |
| 李宗諺 | 資料預處理與導入、資料查詢與整合 |
| 游子慶 | 網頁前端、後端製作與對接、撰寫報告 |
| 吳宇藤 | 簡報製作、功能偵錯 |
| 許書瑋 | 管理github、前後端對接、網頁美化、撰寫報告 |