|  |
| --- |
| SCHOOL OF MEDIA STUDIES AND IT |
| **Groupon Travel Package** |
| CPAN204 - Project |
|  |
| ***Fuchun Chai - N01210879 & Henrique R Belotto - N01245990*** |
| **December 9th, 2018** |

|  |
| --- |
| This is the CPAN204-Project2 which students design a web application using PHP/Python/MySQL |

**Table of Contents**

[**Section 1: Analysis**](#_30j0zll) **2**

[**Section 2: Database Design**](#_1fob9te) **3**

[**Section 3: Design Web forms**](#_3znysh7) **4**

[**Section 4: PHP codings**](#_rwcmw2rbpidd) **9**

[**Section 5: Python codings**](#_tyjcwt) **15**

[**Section 6: Program execution/output**](#_3dy6vkm) **18**

[**Section 7: Project Feedback**](#_1t3h5sf) **22**

# Section 1: Analysis

Design the features:

a. Webpage for the customer to create an account;

b. During the account creation, the user selects an interest and date from two dynamically created drop-down list. After an account is created, the registrationID, which will be a 6-digit random number, will be displayed for the user. The user must use email/registrationID to login in the webpage next time to check the group;

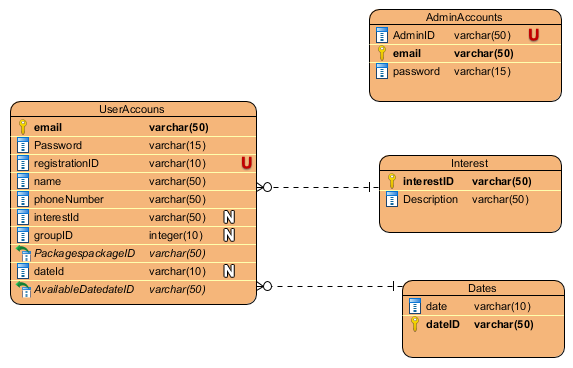
c. Webpage for the customer to login in the system using email/registrationID;

d. Webpage for an admin account to login in the system, using the hardcoded account in the database;

e. Webpage for the admin to manage the groups and import data from a file;

# Section 2: Database Design

The database will be designed as below:



The userAccounts table will hold the user’s information, including the group that the user was selected and his/her interests. The AdminAccounts, for the moment, will have one value already set, which is:

**AdminID: 01**

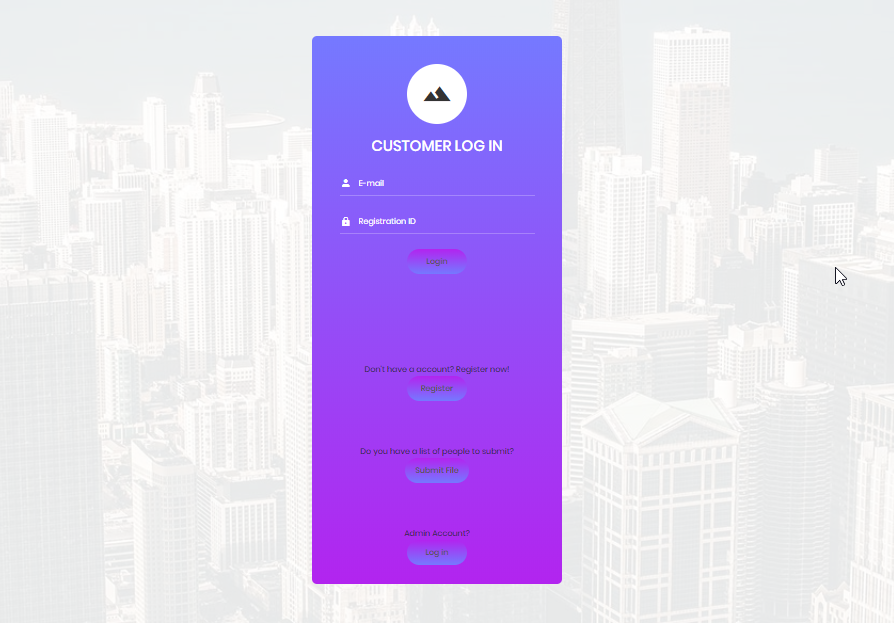
**Email: admin@gmail.com**

**Password: admin**

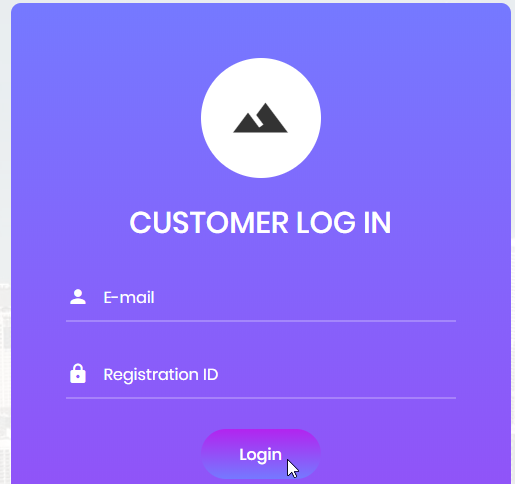
The interest table will hold, for the moment, some packages/attractions available for the customer.

# Section 3: Design Web forms

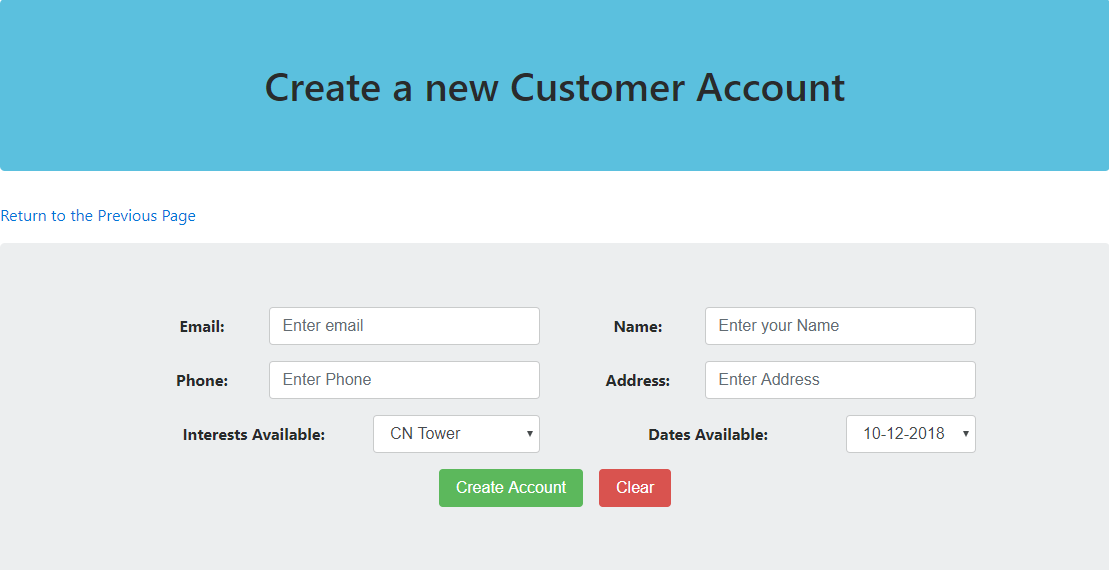
**Login Screen**



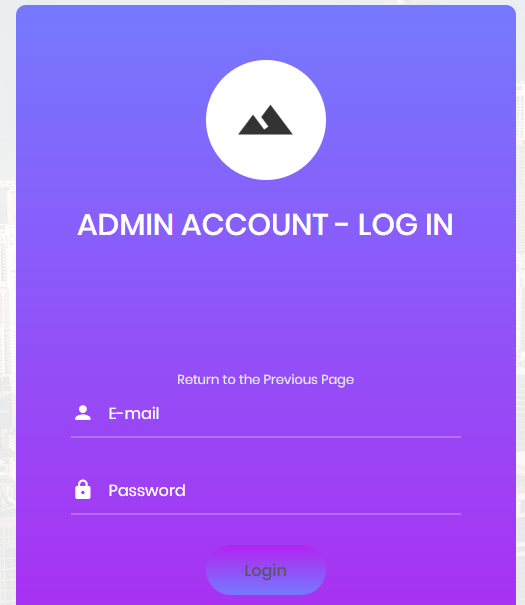
When a user hovers his/her mouse on the button, it changes the color



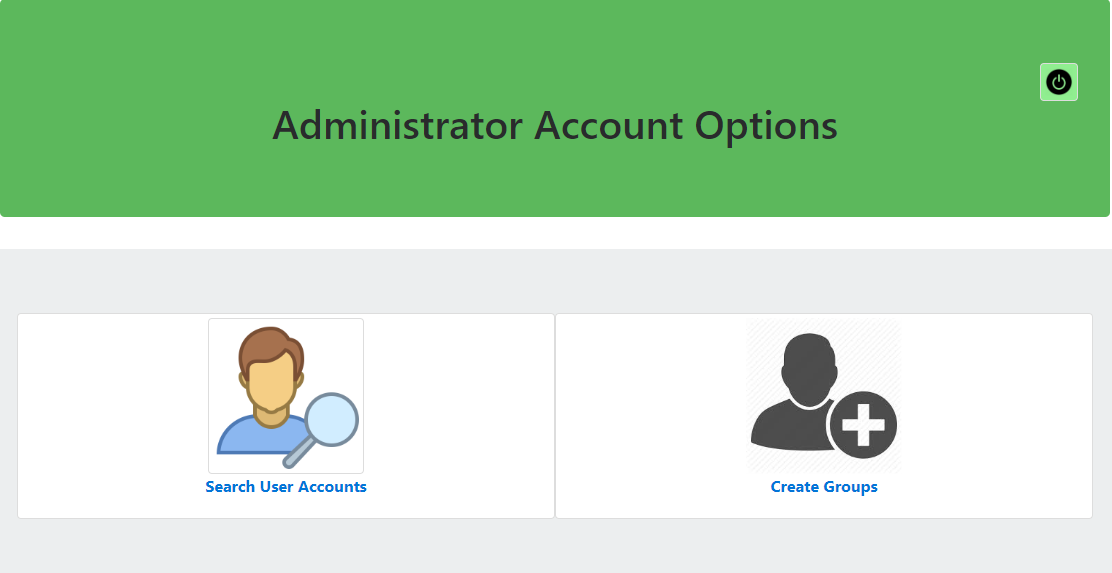
**Create a new Account Web Page**

****

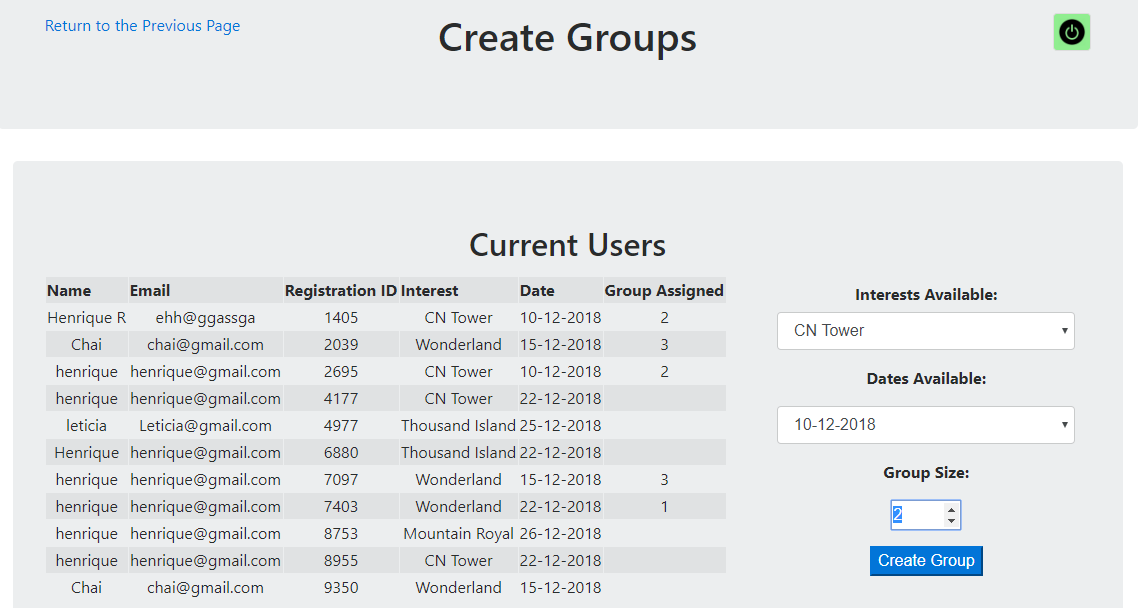
**Admin Account Login**

****

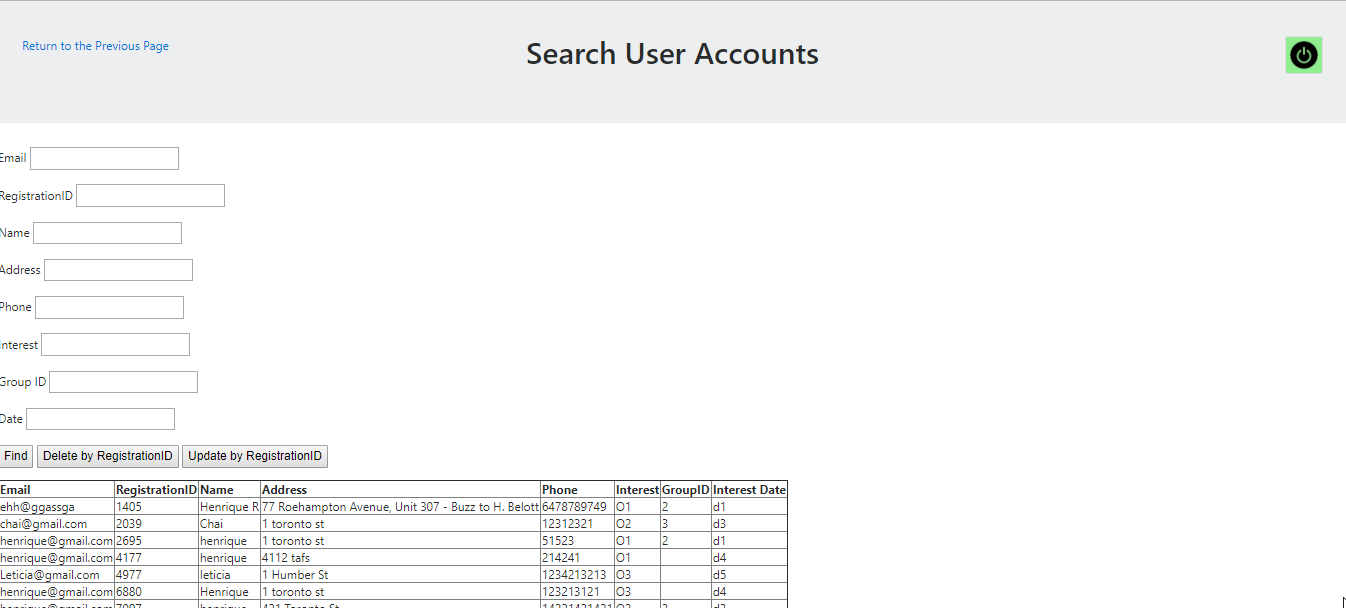
**Admin Account Options**

****

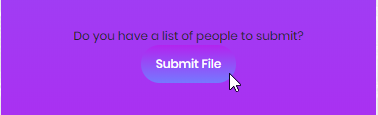
**Create Groups from the Admin Account**

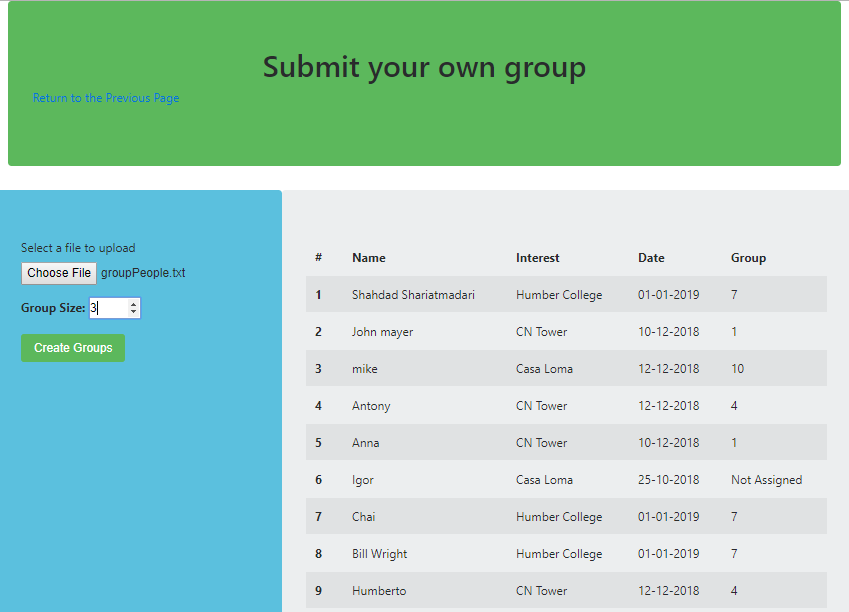
****

**Search & Edit User Account from the Admin Account**

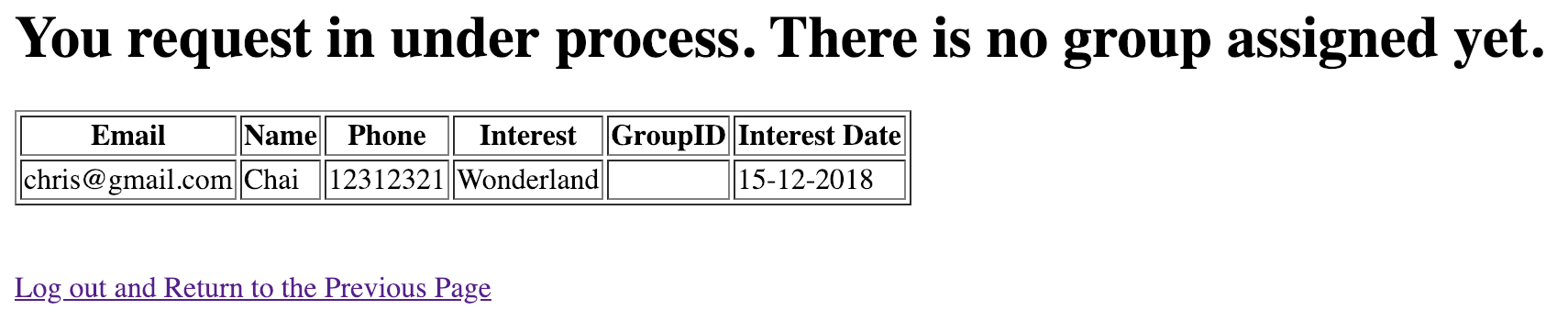
****

**Submit File (from the login screen)**

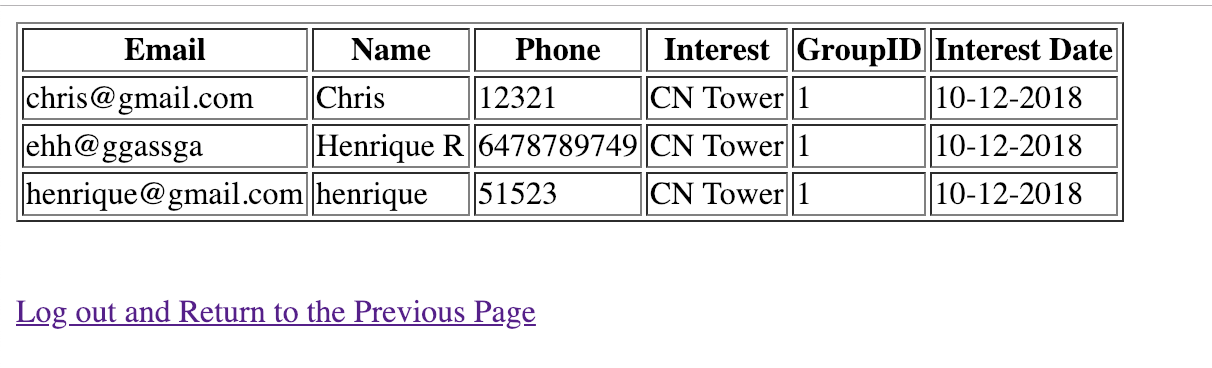
****

****

**Before Assigning Group:**

****

**After Assigning Group**

****

# 

# 

# 

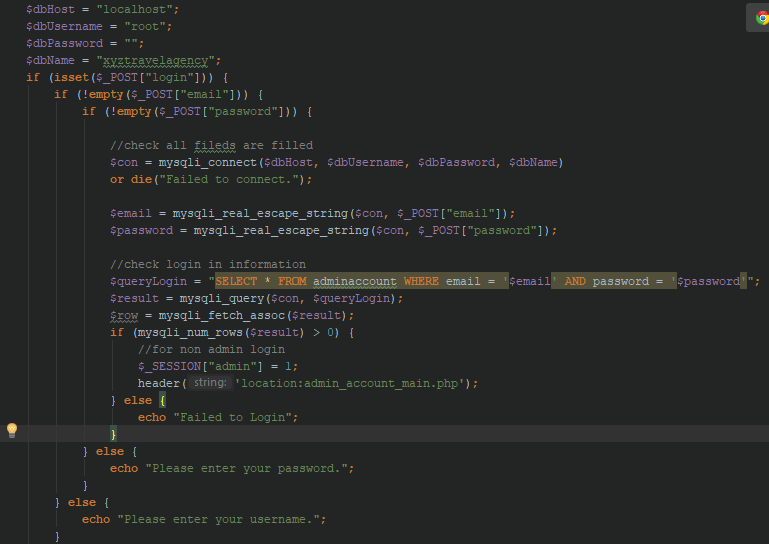
# 

# 

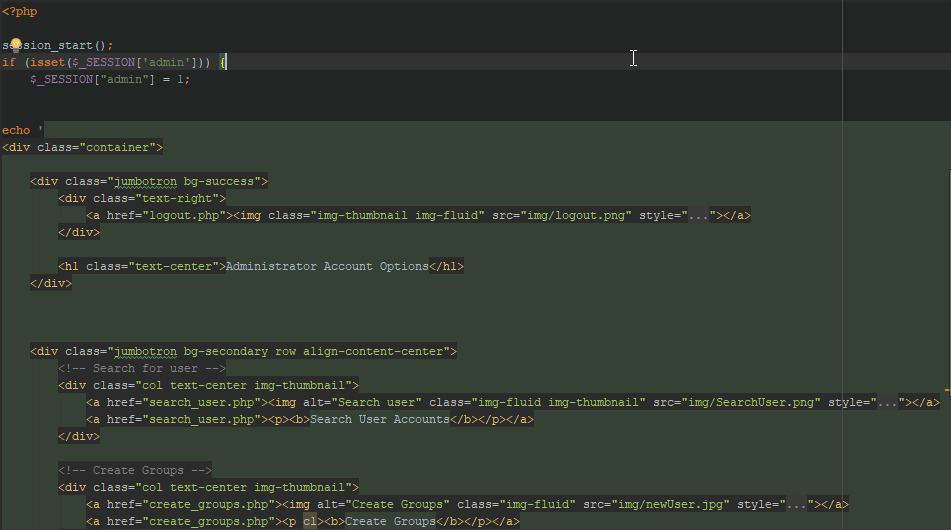
# Section 4: PHP codings

The **index.php** file contains the login screen, which allows the user to log in as a customer, using his/her registrationID and email, or to click to submit a file, or to click to log as an Admin account.

The **admin\_account\_login.php** file contains the admin account login screen. The login entered by the user is verified on the database. If it’s correct, allow the user to proceed to the next webpage. It also creates a Session which is used to guarantee that only authenticated users can visualize the admin pages.

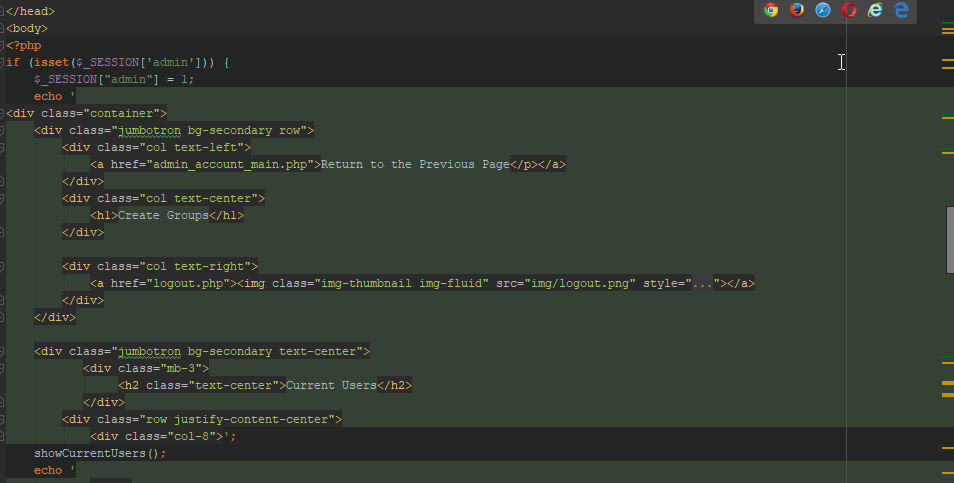


The **admin\_account\_main.php** file contains the HTML code that allows the admin user to select to search for a user or to create groups. Only authenticated users can access this webpage

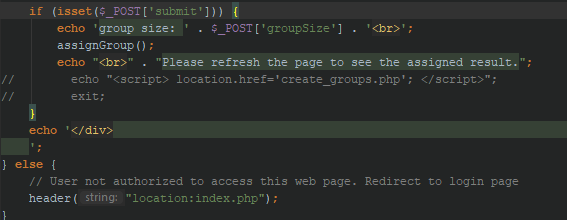


The **create\_groups.php** file contains the code used to display the HTML/CSS for the user and to generate and store the groups based on the user’s input.

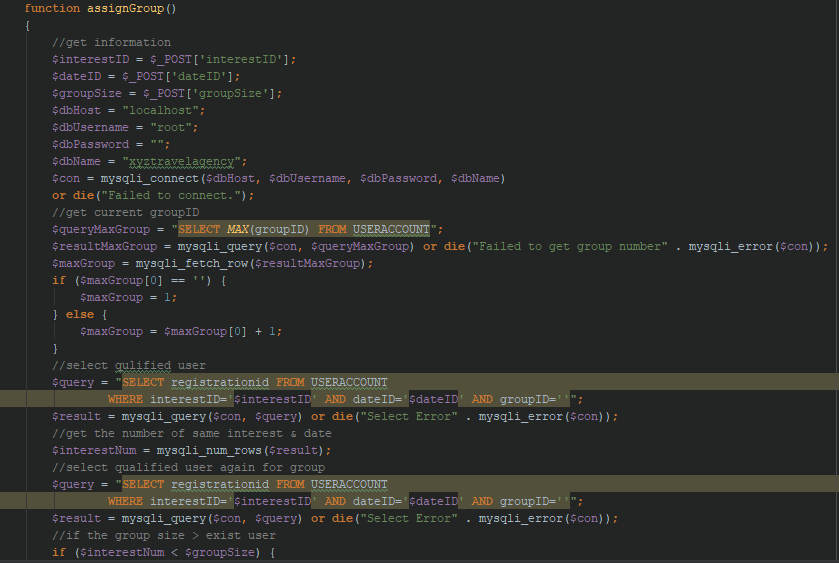
Only authenticated users can access this webpage. Once the user access this webpage, a list with the current users, including user’s details and groups, is displayed.



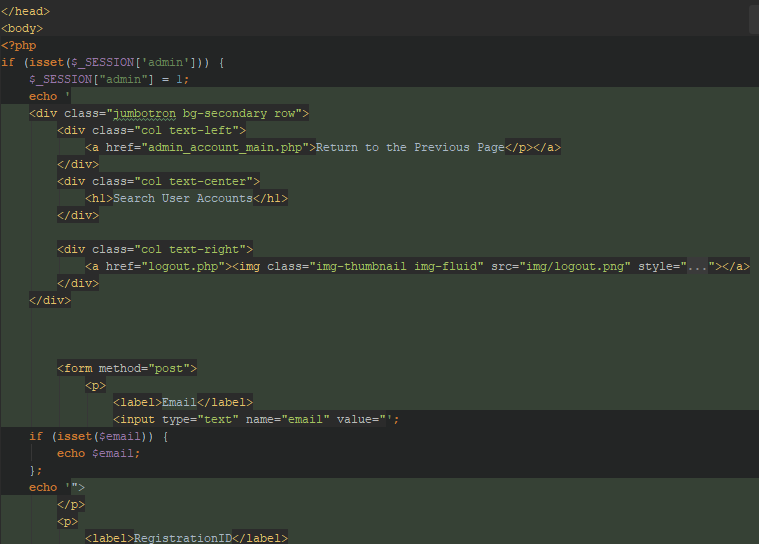
If a user clicks on submit, the following code is executed. It verifies that “group size” and “submit” button were set. If they were set, then execute the function “assignGroup()”, which is located inside the same file.

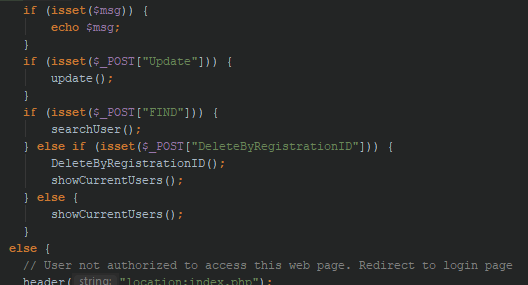


The function assignGroup execute multiples query to the database and automatically selects the groups based on the group size, interest, and date selected by the admin user.

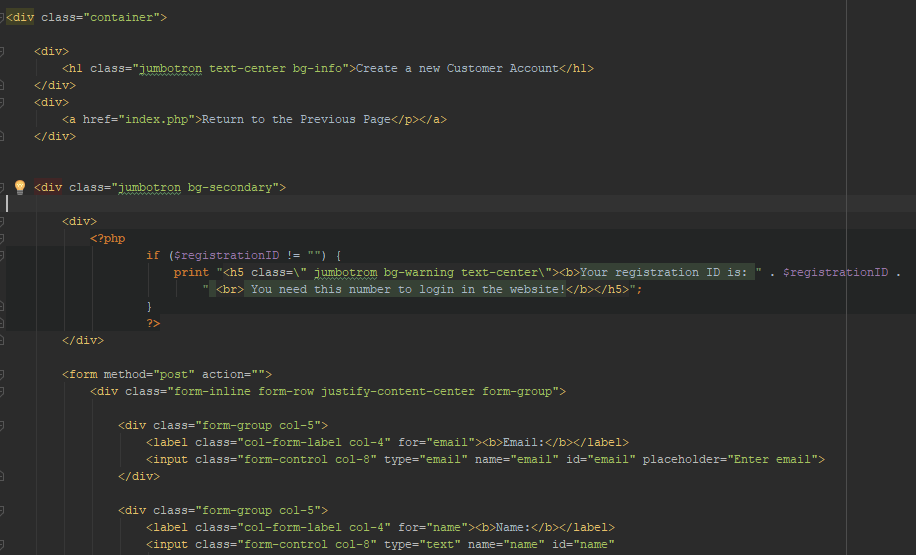


The **search\_user.php** file contains all the code necessary to generate the HMTL/CSS webpage and to search, update and delete user accounts. This webpage is only accessible using an admin account.

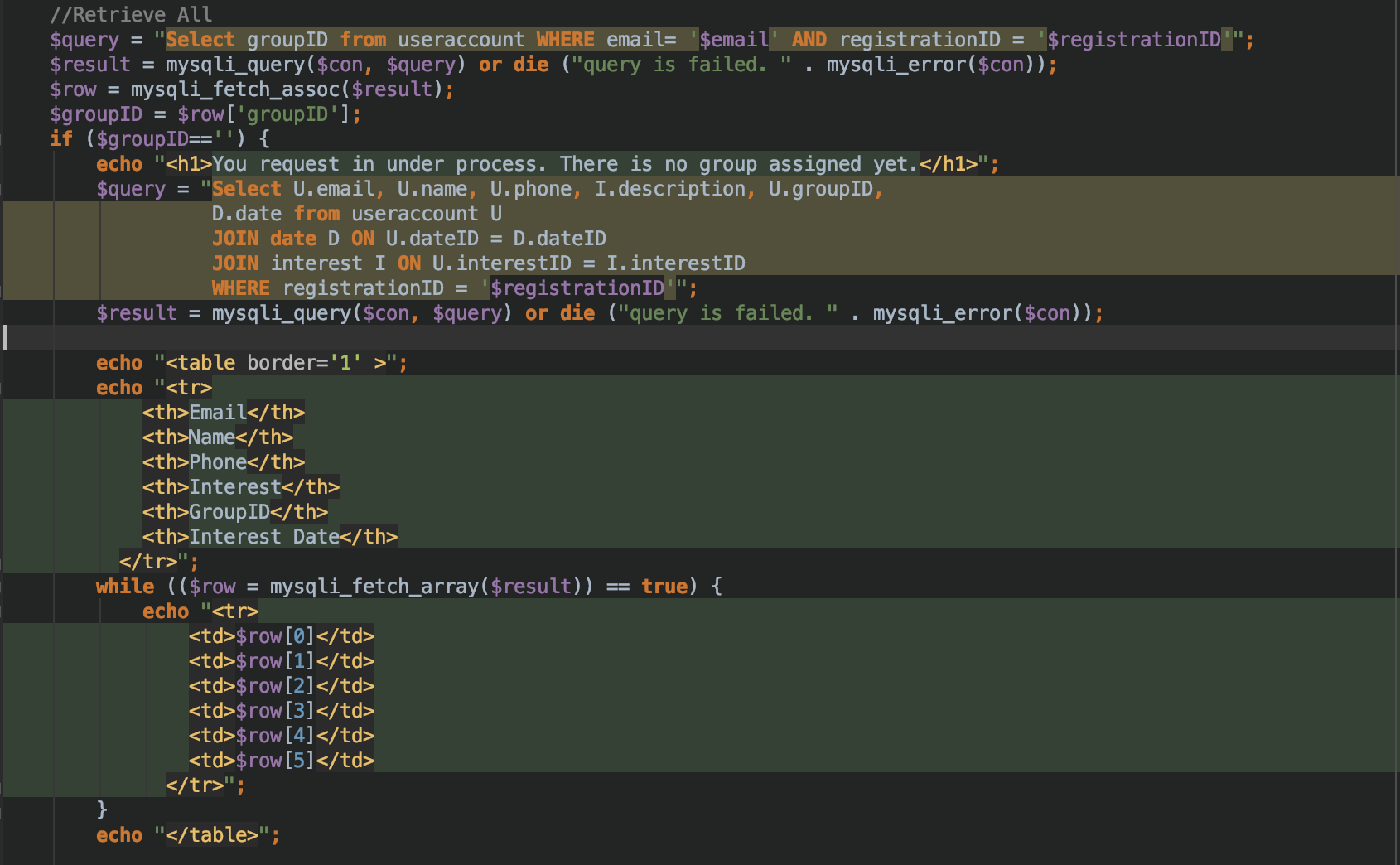


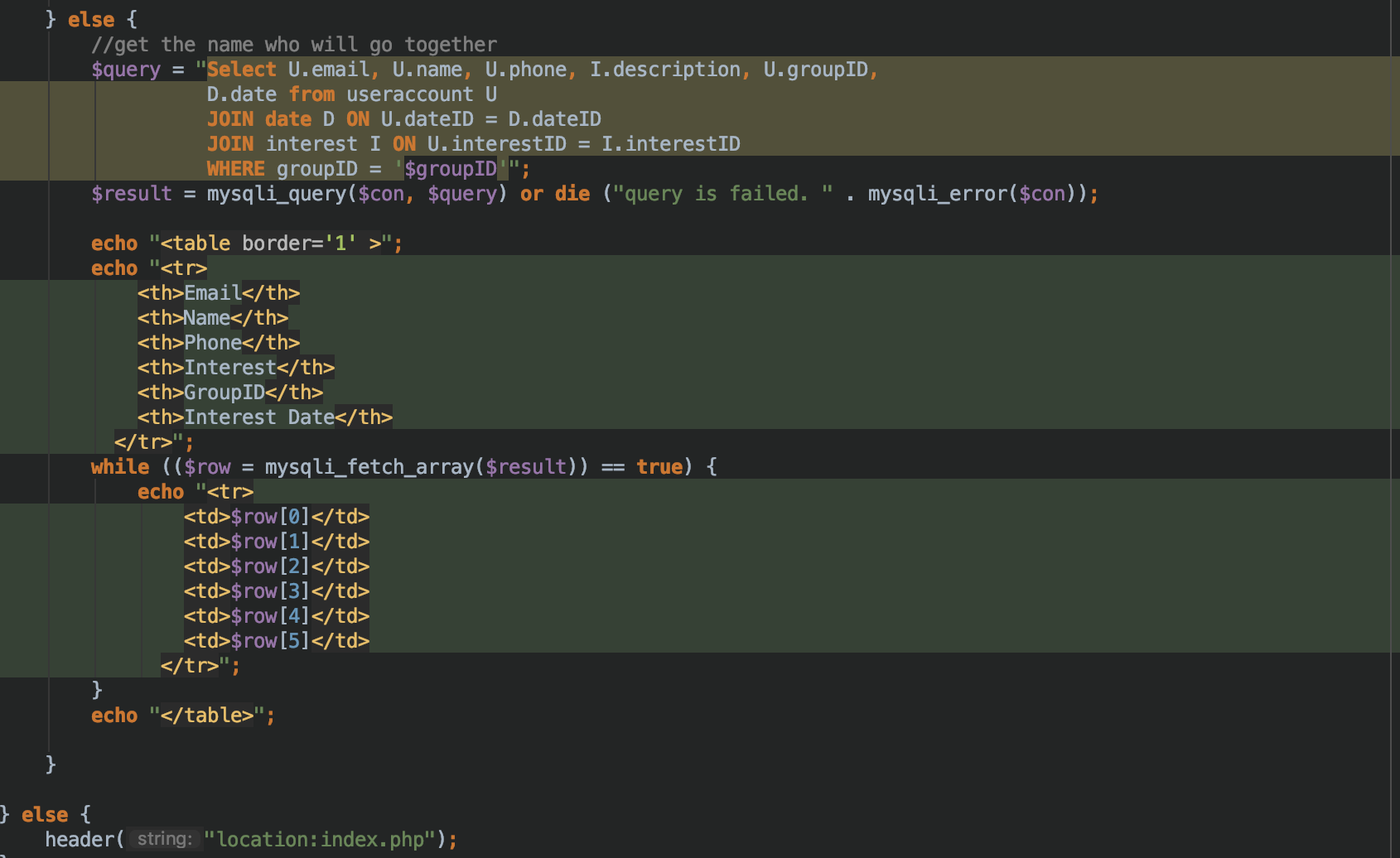


The **create\_customer\_account.php** file contains all the code necessary to generate the HTML/CSS to a new user to create an account. This webapage can be accessed from the login screen and does not require any authentication.

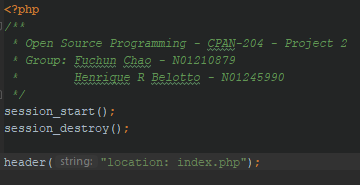


The **user\_main.php** file contains codes that will show user information. If a user is assigned a group, he will get his group ID, as well as other group members information. If he is not assigned yet, it shows not assigned.





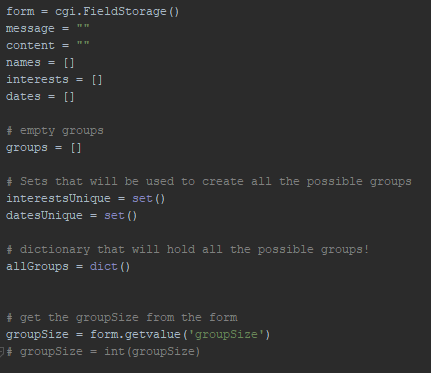
The l**ogout.php** file contains a simple code that is used to redirect a user when he/she wants to log out. This code basically kills the session, allowing the user to log out from the website.



# Section 5: Python codings

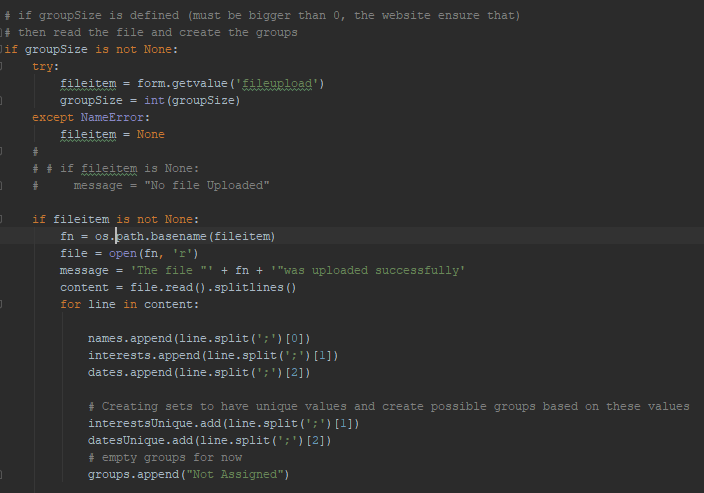
The only Python program is called **processFile.py**, and it is used to generate the basic HTML/CSS of the webpage, then receive the file from the user and, based on the group size, automatically generate all possible groups.

This part of the code checks the user’s input and generate the empty arrays that will be used to store the user’s input



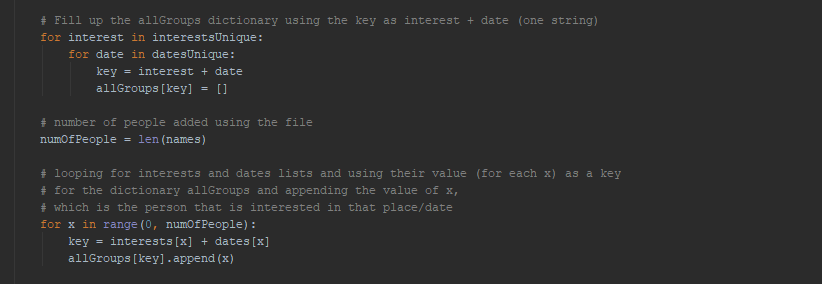
The next image shows the code that verifies if the user has entered the group size and uploaded a file. If a file was uploaded and a group size selected, then read the information from the file and store into the names, interests and dates array.

Also, datesUnique and interestUnique sets are also created during this part. They contain all the interests and dates obtained from the file, without repetition.

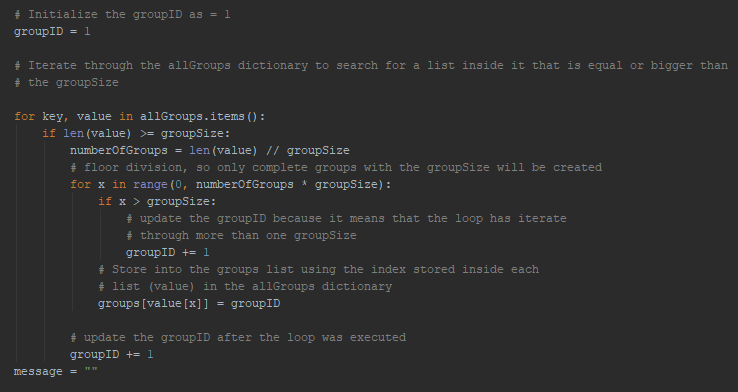


The next image shows how the keys for the dictionary called “allGroups” were created, which is basically a string containing “interestdate” all together obtained from the unique values from the interestUnique and datesUnique sets.

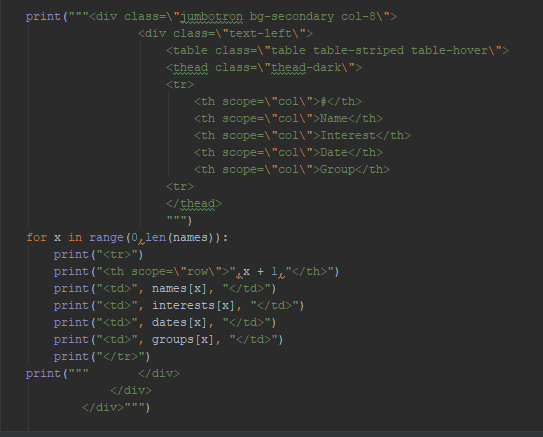
After these keys were created, iterate from the dictionary adding the index of the names that have inserted the same interest + date



The next part shows an iteration on the items stored into the allGroups dictionary to verify how many indexes were inserted on each key/value, which basically means how many people have said the same interest + date. Based on the number of people, if it’s equal or bigger than the group size, then it creates a group number for these indexes. The codes also make sure that no partial groups are created.

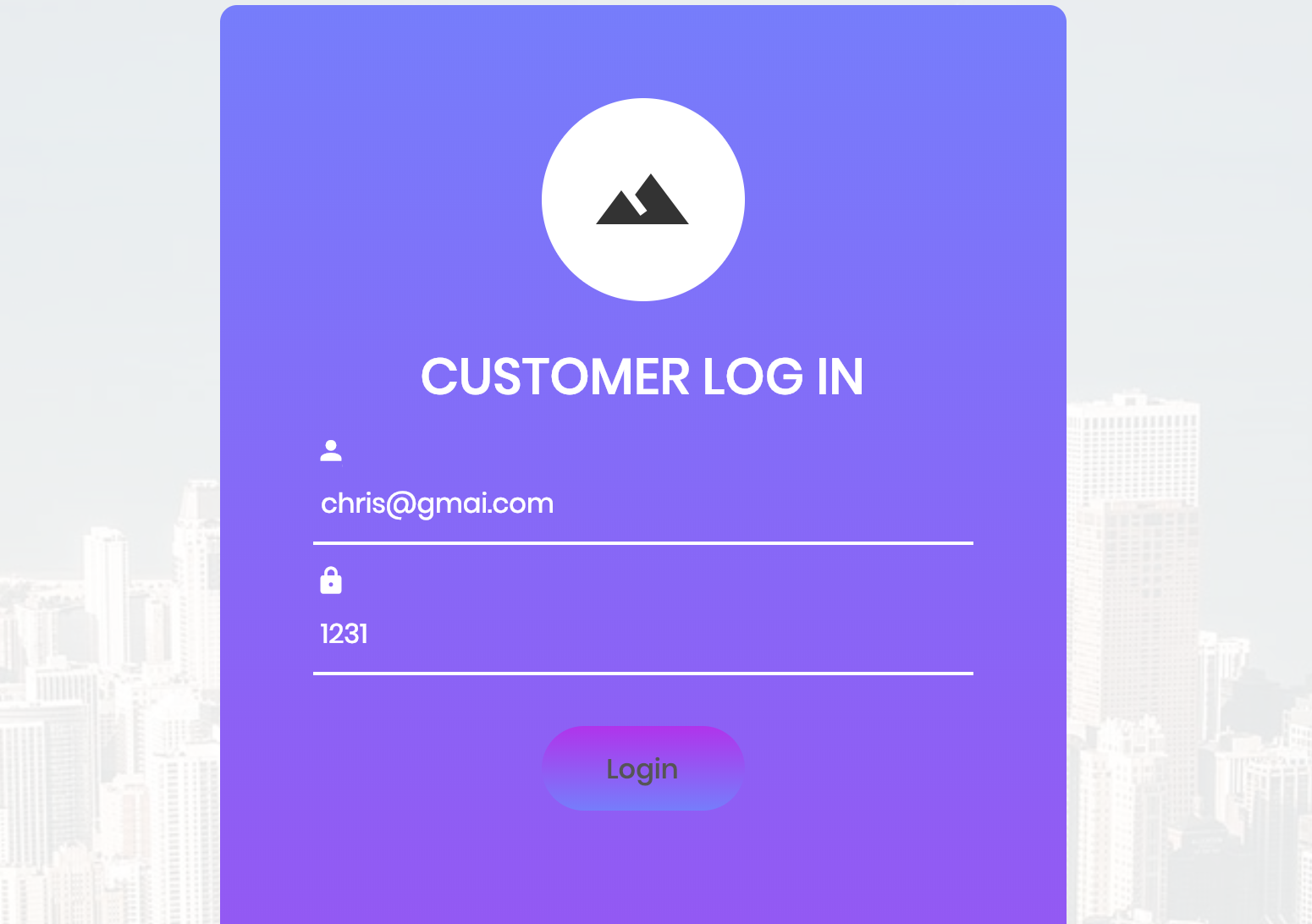


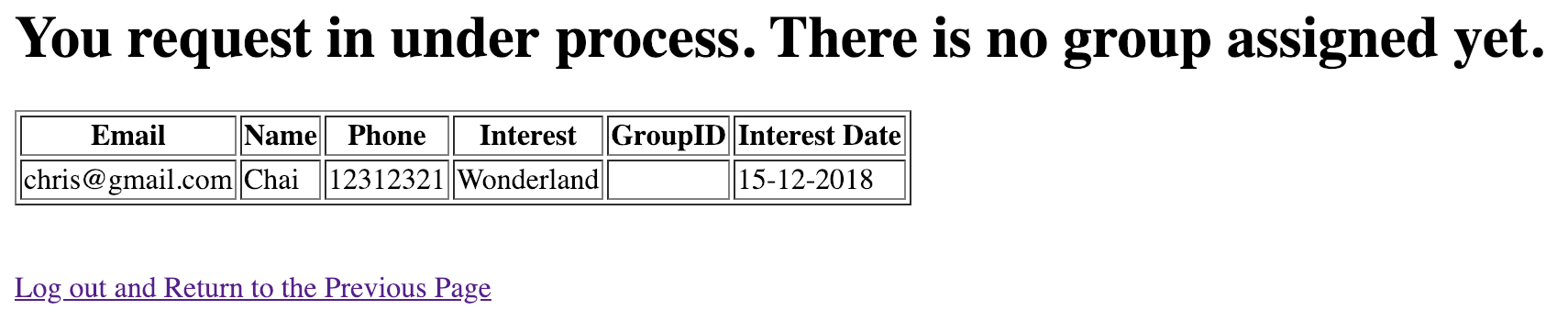
The output is displayed in a table, following the code below

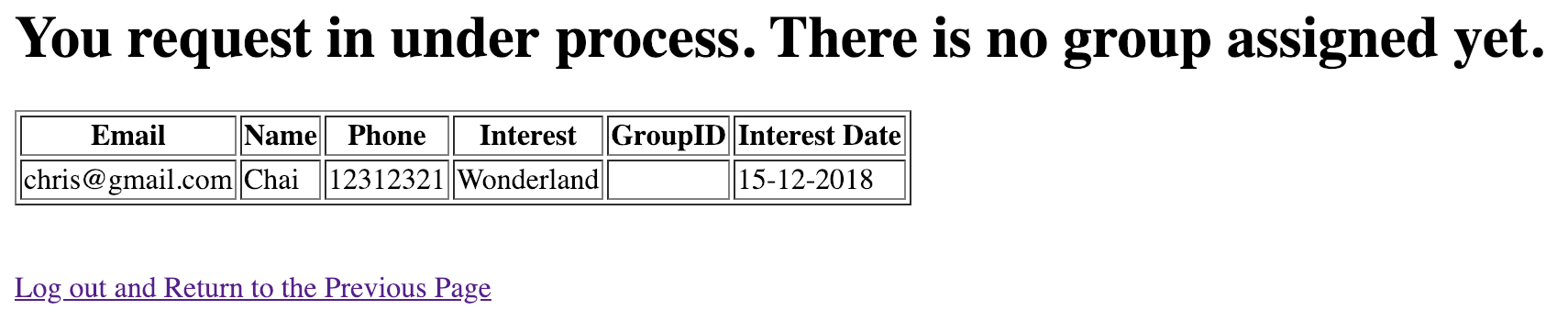


# Section 6: Program execution/output

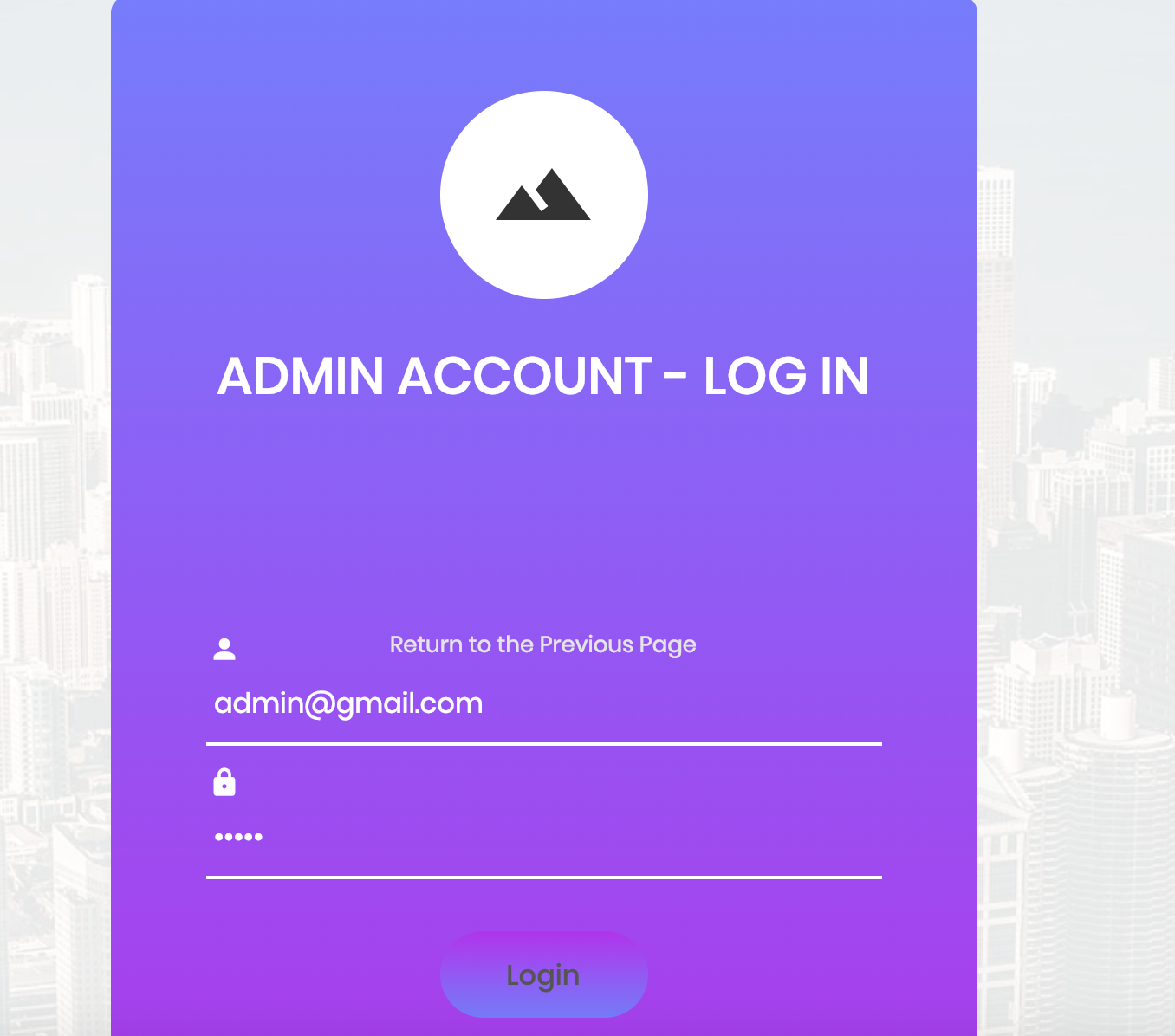
**User Login:**

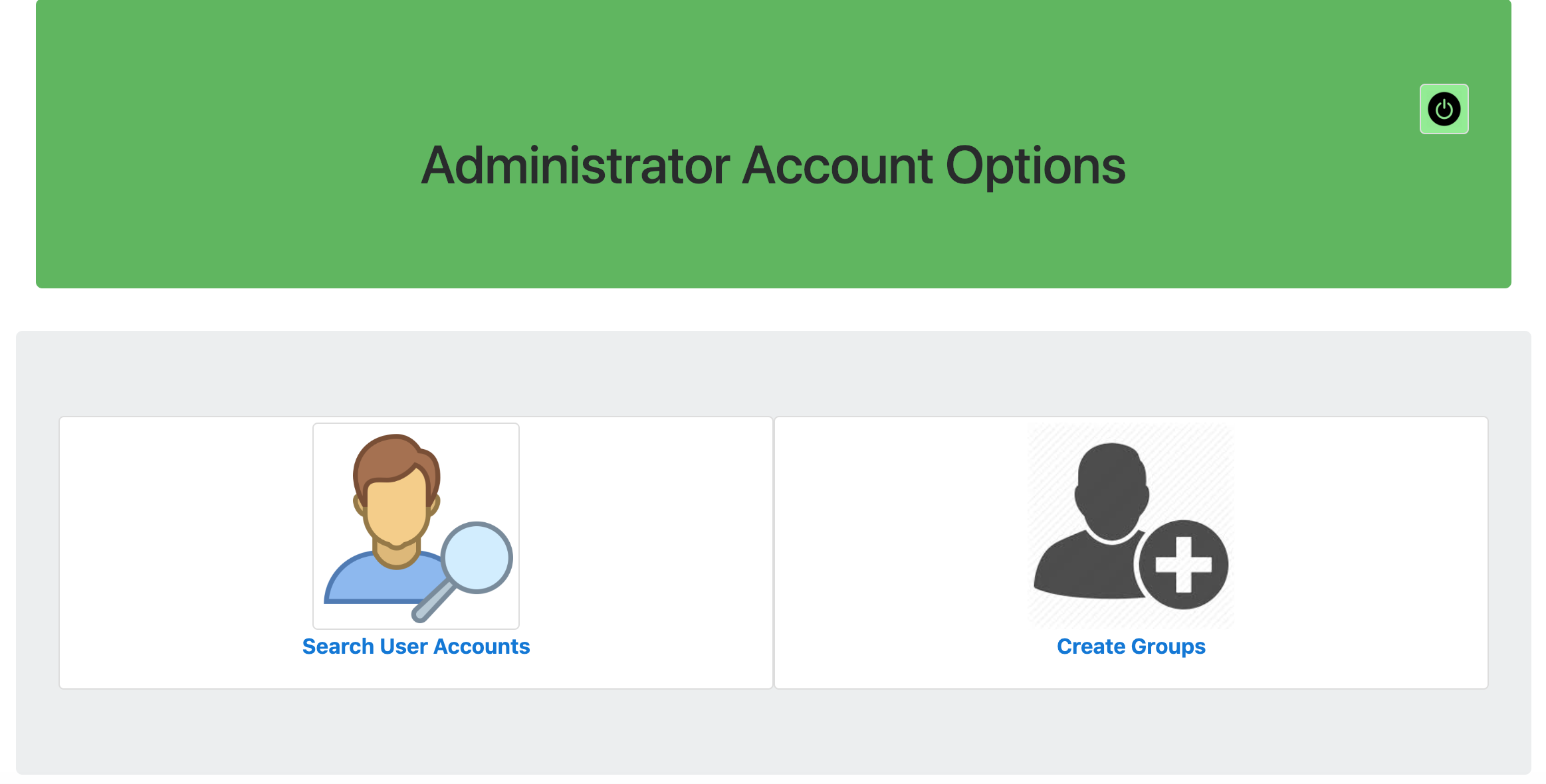
****

****

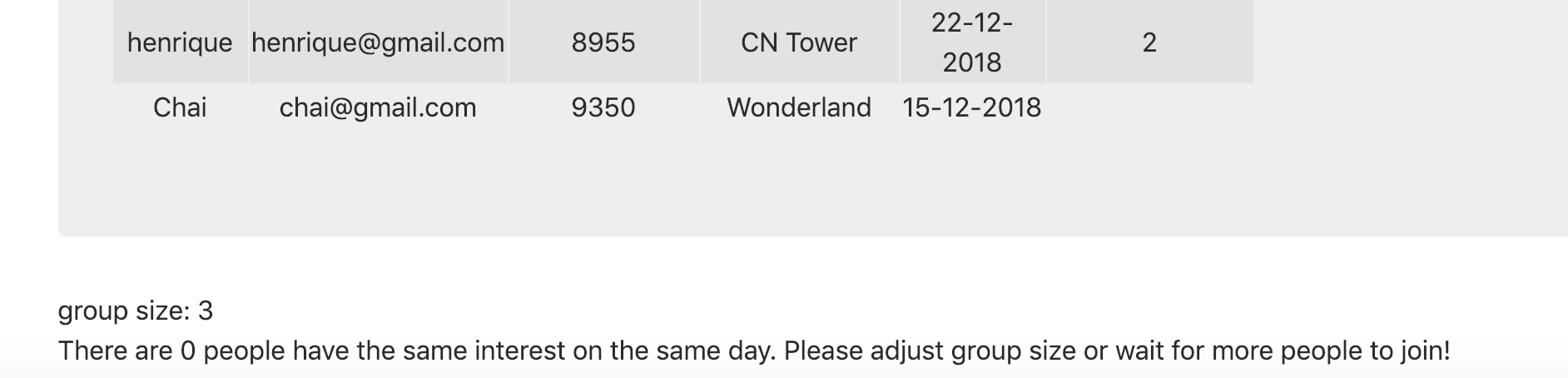
****

**Admin Login:**

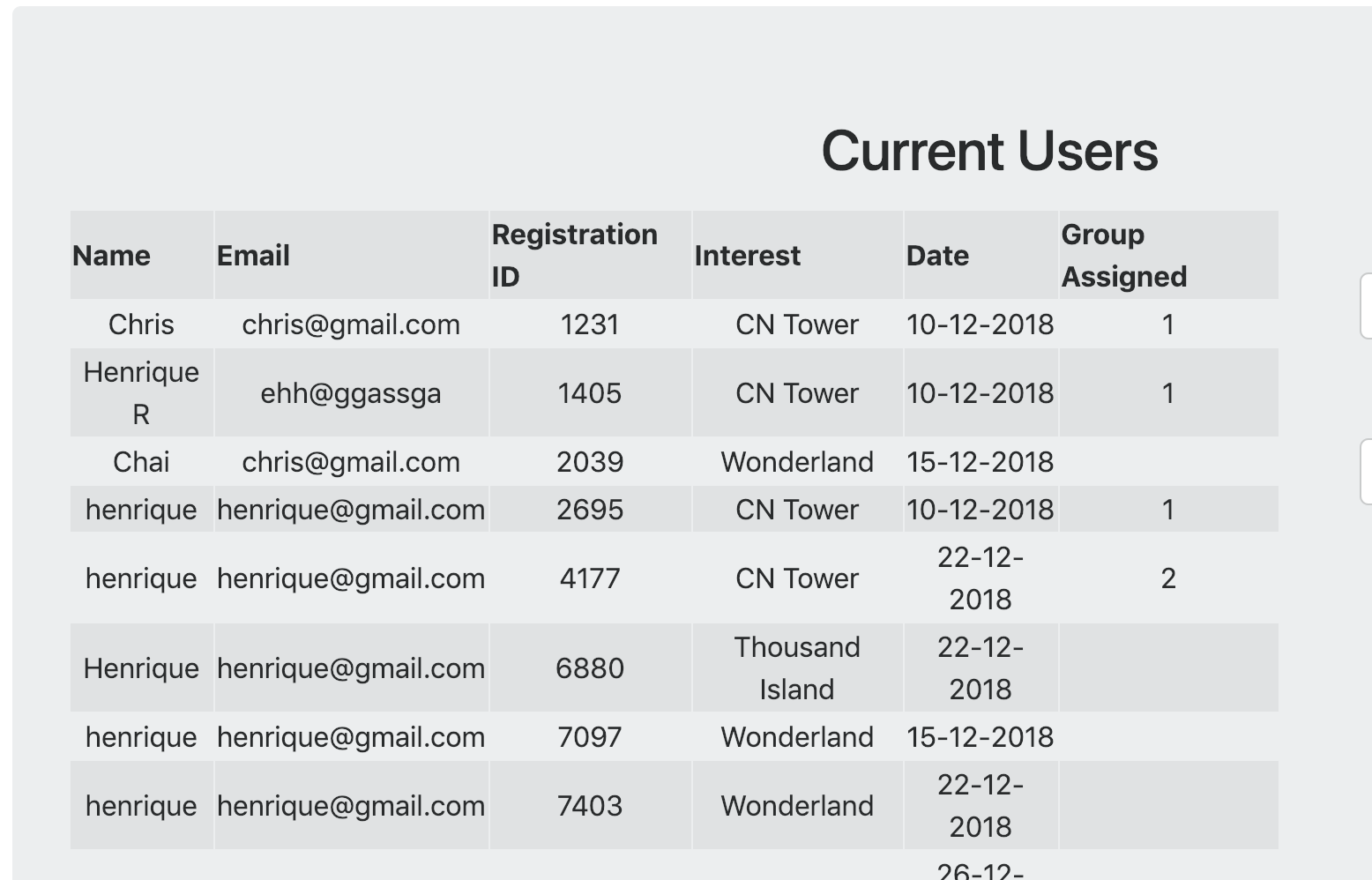
****

****

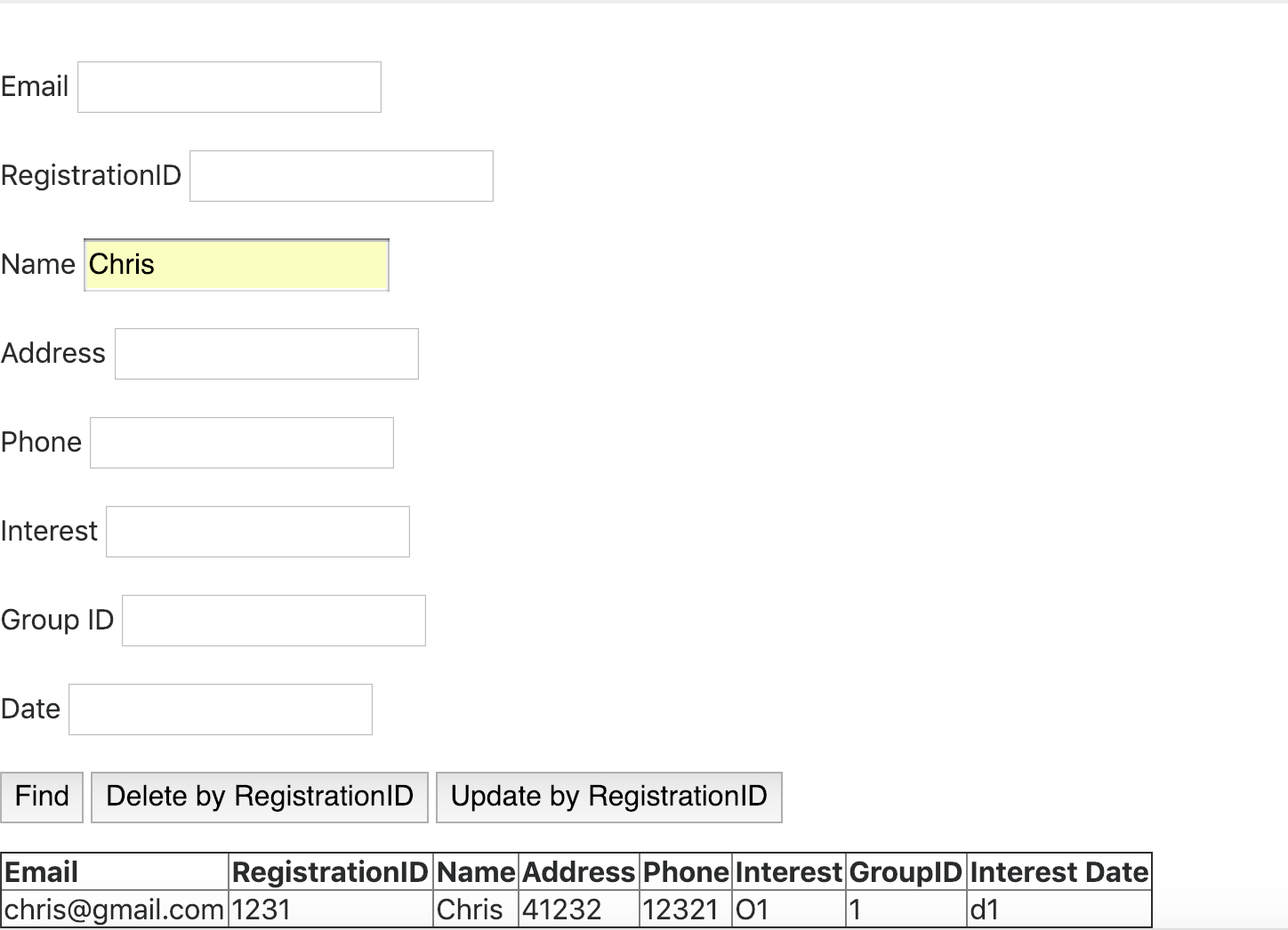
**When Assign group that contains no same interesting people:**

****

**When assignment successful:**

****

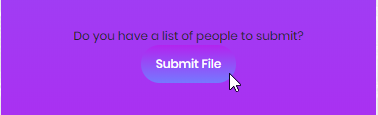
**Search User by Entering any Piece Information:**

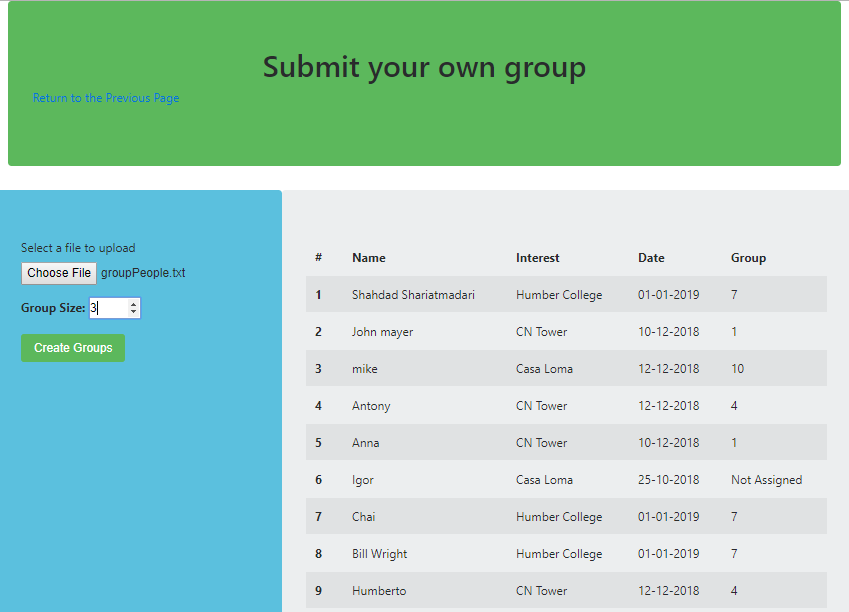
****

**Update Any Piece / Pieces Information:**

****

**For Submitting File:**

****

****

# Section 7: Project Feedback

This project allowed us to work together using a repository on Git Hub, this helped us to share our code and keep updated with the latest version of whatever the other member has just finished.

Moreover, during this project, we faced multiple challenges regarding the implementation of full web pages using PHP or Python, also to create an algorithm to read a file and automatically selects the groups only based on the group size. It was a challenge to come up with something that looks as efficient as possible and reliable.

Time management was a big issue for our group because both members work during the week and weekend, also, other assignments from other courses required a lot of our time. Despite that, I believe that we used our time efficiently to provide a well-designed and developed project.