Friendify
A friend-making Application

CSC111 Final Project

Eeshan Narula and Avnish Pasari

Contents

1 Introduction **3**

2 Datasets **4**

3 Computational Planning **5**

3.1 Overview 5

3.2 Handling the Data 6

3.3 Recommending friends 7

3.4 Visualizing friend circle 9

3.5 Searching for People 10

3.6 User Interface handling 11

4 Download/Obtain Datasets **13**

5 Running the App **14**

6 Changes from Phase 1 Proposal **22**

7 Discussion **24**

8 References **26**

Chapter 1

Introduction

The onset of COVID-19 has made it extremely hard for people to socialize and make new friends.

The constant quarantine and not being able to meet people, visit restaurants, go to clubs or attend social events has led to an increased social isolation amongst people.

This increased social isolation amongst people, if not addressed, can negatively impact the mental health and development of the concerned individual. (Source 1)

Studies have shown that loneliness is often associated with negative thoughts (cognition). Moreover, anxiety and depression may cause social withdrawal which exacerbates the loneliness and isolation associated with social distancing. (Source 2)

Most humans, by nature, are social creatures. Over the past few months - repeated lock-downs, bans on public gatherings and self-imposed quarantine have severely impacted interaction amongst the people.

This has led to the development of fear, stress and anxiety amongst the masses during these uncertain and difficult times. Thus it becomes increasingly vital for us to interact with people (online) and keep ourselves engaged, both to preserve our mental health and also to create a positive environment for those around us (i.e. family).(Source 3)

Interacting and connecting with people who have shared interests and hobbies can help reduce mental stress, anxiety and isolation-trauma, thus enabling people to better cope with the COVID-19 pandemic.

Studies have shown that online social interaction in such a time, can help rejuvenate our low spirits, improve mental health and sharpness as well as enable us to develop life-long relations with others during this time, which may have otherwise been impossible without our current technology. (Source 4)

To address this issue of adverse mental health conditions due to decreased social interaction and to provide a solution in the form of an online platform where people can find and connect with each other - **We plan to design a program which would allow people connect with each other so as to find and make friends based on their shared interests and compatibility.**

Chapter 2

Datasets

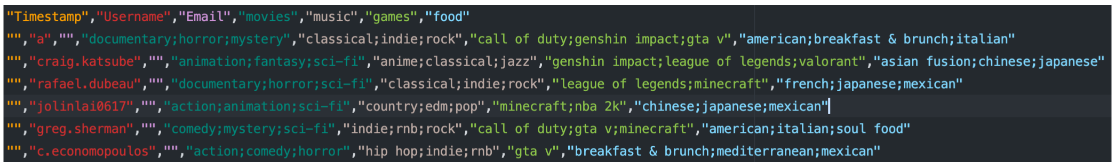
Description

Initially our app would not have any users registered with it.

Thus we conducted a survey where we asked people about their movies, music, games and food preferences.
The survey was conducted using Google forms.

We also asked the participants for their emails, but scrapped the part after @ to maintain privacy. The timestamp was also removed from the data-set, for the same reason.

The data collected has been stored in a csv file. Provided below is a picture demonstrating how the concerned csv file looks



```
"Timestamp","Username","Email","movies","music","games","food"
"","a","","documentary;horror;mystery","classical;indie;rock","call of duty;genshin impact;gta v","american;breakfast & brunch;italian"
"","craig.katsube","","animation;fantasy;sci-fi","anime;classical;jazz","genshin impact;league of legends;valorant","asian fusion;chinese;japanese"
"","rafael.dubeau","","documentary;horror;sci-fi","classical;indie;rock","league of legends;minecraft","french;japanese;mexican"
"","jolinlai0617","","action;animation;sci-fi","country;edm;pop","minecraft;nba 2k","chinese;japanese;mexican"
"","greg.sherman","","comedy;mystery;sci-fi","indie;rnb;rock","call of duty;gta v;minecraft","american;italian;soul food"
"","c.economopoulos","","action;comedy;horror","hip hop;indie;rnb","gta v","breakfast & brunch;mediterranean;mexican"
```

Columns Used:

- Username: the column ”Username” is used to identify the users.
- movies, music, games, food: Used to recommend friends to the users of the app.

Chapter 3

Computational Planning

3.1 Overview

The App uses real world data, to recommend friends, visualize network and socialize.

The data contains information of all the users of the app. The information consists of the unique user ID and the preferences of the user. The preferences represent the different movies, music, food and games that the user likes.

This data is used to construct recommendation graphs and network graphs which consists of the users and their preferences as the vertices, with edges representing a relation between two users or a users preferences.

This data is stored on cloud and is accessed by our app very time a user wants to see friend recommendations, visualise their friend circle or socialize.

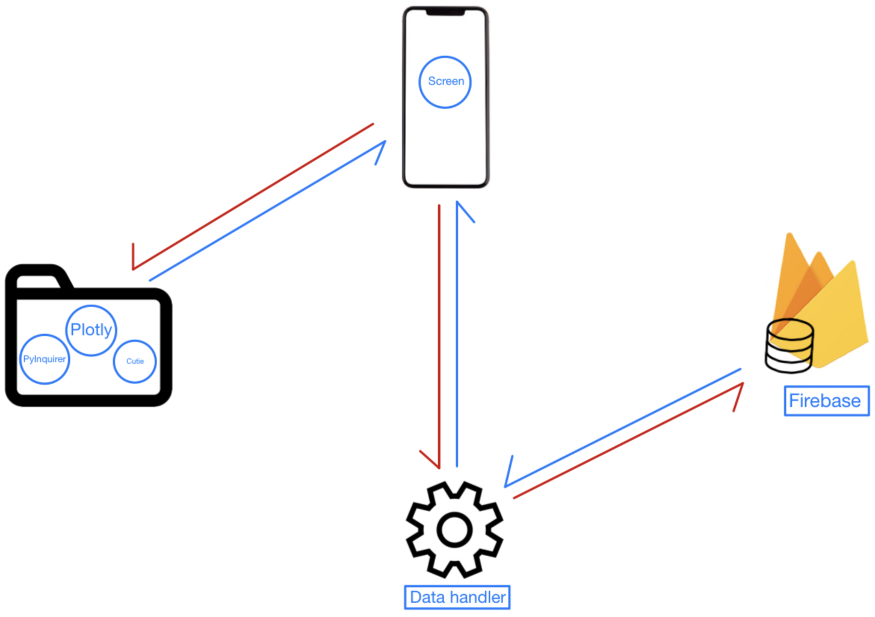
The working of the app is divided into 5 main parts:- Handling the user's data, Recommending friends, Visualizing friends circle, Searching for People, and User Interface handling.

3.2 Handling the Data

All the data is stored in Google's Firebase cloud storage, as a database. The database consists of user data, identified by the unique user ID. We are using a library called firebase-admin to communicate with the cloud storage, to retrieve and set data.

We have created a DataHandler class (authentication.py), which configures the connection between the cloud and the app. This class is responsible to handle all the requests to the cloud. It contains the functions, to create a new user, sign in the user, load a user's data by their unique user id, set a user's data, update a user's data and many other functionalities.

We have tried to use the MVC architecture to handle the data in an efficient way. The DataHandler class acts as the Model-bridge between the View controller and the Database. Both View controller and the Database interact with each other through the DataHandler.



In the above figure, the screen represents the View controller. For getting data, signing in and registering, the Screen first sends a request to the DataHandler.

The DataHandler then send a request to the Firebase cloud server to get the data. The server responds and returns the data to the DataHandler. The DataHandler then returns this data to the Screen/View Controller.

Initial Data

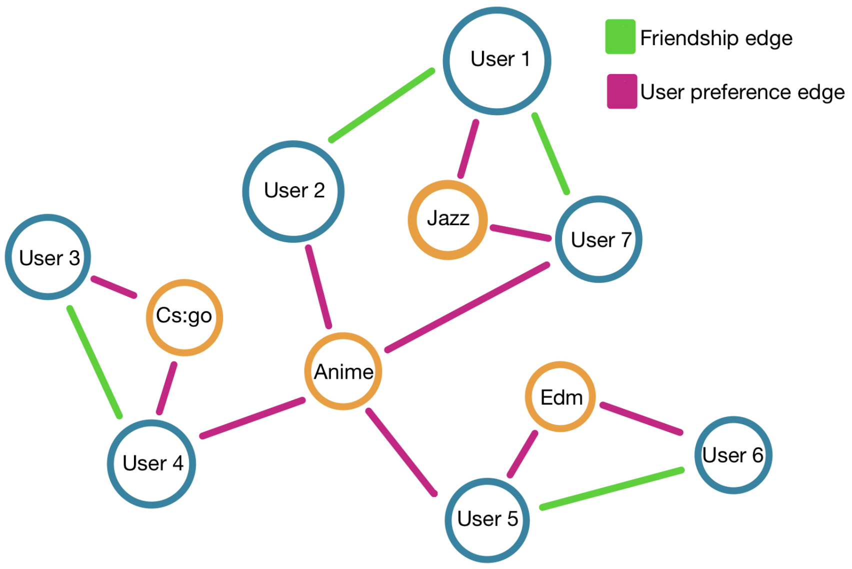
Initially there were no users, so there were no friend recommendations. Thus we conducted a survey in which approximately 50 people participated. The DataHandler class was also used to read the survey data, and register all the users to the cloud database. These 50 users were also assigned some friends according to the recommendation system.

3.3 Recommending friends

The main goal of the app is to recommend friends to the user. The app fulfills this goal by constructing recommendation and network graphs.

Constructing The Graph

- To construct the graph first we load data for all the users of the app using the DataHandler class (authentication.py).
- For each user, we create a vertex, with the unique user ID as the value of the vertex. These vertices are of type user.
- We would then add all the possible preferences a user can have to the graph, as a vertex. These vertices are labelled as preferences. For instance these preferences could be the type of music, type of food, movies or games the user likes.
- Now we would add an edge between all the users and their preferences, which would represent a user preference relation, and an edge between user and other users, which would represent friendship between the two users.
- The graph is generated by the load_friends_graph function present in Graph class in recommendation_graph.py
- This is how the graph should look like (the vertices in orange circles are the preferences, these are used to recommend friends to the user):



Use the Graph to Recommend Friends

- To recommend friends to a user, first we load data for all the users in the database.

- Then we iterate over all the users in the app, using the graph that we built. For each user we calculate a similarity score to the user we want to recommend friends to.

- The similarity score for two users is calculated by calculating the ratio of number of preferences common between the two users and the total number of preferences both users have in total. This score is calculated by _Vertex.similarity_score function in recommendation_graph.py (This part is a bit similar to Assignment 3 but has been modified for our app).

- Let $G = (V, E)$ represent this graph. Let $u_1, u_2 \in V$ represent the two users. Then the similarity score between two users is given by the below formula:

$$sim(u_1, u_2) = \begin{cases} 0, & d(u_1) = 0 \text{ and } d(u_2) = 0 \\ \frac{v|v \in V, \text{ v is a preference, } u_1 \text{ and } u_2 \text{ are adjacent to } v}{v|v \in V, \text{ v is a preference, } u_1 \text{ or } u_2 \text{ is adjacent to } v} & \text{otherwise} \end{cases}$$

- The users are then sorted according to their similarity score in reverse order, and we return the top 10 users which had the highest similarity score to the user.

- We also return the similarity score alongside the name of the returned users.

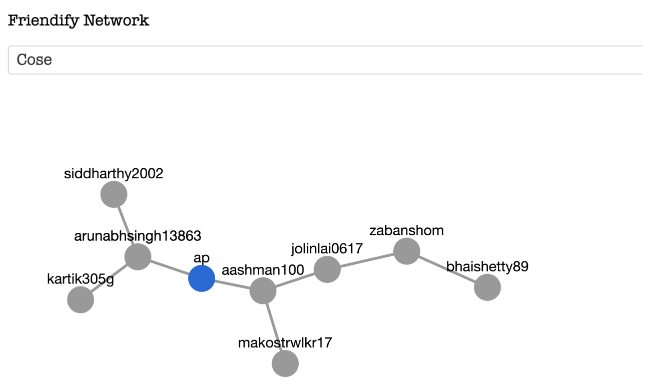
- It is highly unlikely, but a user may not get any friend recommendations with a significant match score. This could only happen when the user does not have many preferences.

- The recommendations are returned by the recommend_friends function in the Graph class in recommendation_graph.py

- After the recommendations are returned, the user can add those recommendations as friends.

3.4 Visualizing friend circle

The user also would have the feature of visualizing their friend circle. They would be able to visualizing a graph containing their friends, their friends' friends and so on. The graph would be interactive and the user can move the vertices and edges.



We are using Dash for performing this visualisation. Other inbuilt libraries we are using are logging and webbrowser.

Creating the Visualization

- To create the visualisation, we first create the graph using the recommend_friends function in the Graph class in recommendation_graph.py.

- Then we recursively create a sub graph of the graph, starting from the user for which we want to create the visualization till a depth d. The depth d represents the depth of the graph. This sub-graph is generated by _Vertex.generate_graph function in recommendation_graph.py. For instance, if the user wants to see only their friends, then the depth would be 1, else if they want to see their friends, and their friends' friends then the depth would be 2 and so on.

- After extracting the sub graph, we again use recursion to format the data for the dash library to plot it as a graph. The formatting is done to make a list of dictionaries representing a vertex or edge. This formated data is generated by _Vertex.format_for_graph function in recommendation_graph.py

- After getting the formatted data, we use the Dash library, to visualize the network. Specifically we plot a Cytoscape graph. We have also used Dash's DOM objects to add the feature of representing the graph in different ways, including 'breadthfirst', 'grid', 'random', 'circle', 'cose', and 'concentric'. The ploting is done in the plot function in Graph class in recommendation_graph.py

- The graph does not open automatically, so we used the webbrowser library, which is an in-built library, to open the site where the visualisations are happening. We have set the app to run on localhost 5050.

- All the verties and edges in the visualized graph are interactive and can be moved around.

3.5 Searching for People

The user also has the feature to search for people by their user ID. This is similar to what can be seen on social media apps.

This feature returns the top search results, which are the most similar user IDs to the search query, which the user typed in.

The user can then go into the profile of the search results, and add them as friends, or unfriend them if they are already friends.

Process of Searching a Query

- First we load up all the user IDs we have in our app's data from the firebase database.

- We then pass the list of user ID's into the merge sort algorithm, but with an extra argument that is the search queue to which the user IDs are being compared. The sorting algorithm is in the SearchPeople class in screen.py

- To get the similarity score, we are using a class from an inbuilt library called difflib, which is SequenceMatcher. The score is calculated in the function similar in the SearchPeople class in screen.py

  This class takes in an argument two sequences. We access the ratio function of the SequenceMatcher, which returns the similarity ratiop between the two sequences (in our case which are the user IDs).

- We use this similarity between the query and the user IDs as the key to sort and get the top search results.

- We are returning only the top 10 search results, which have the highest similarity to the search query.

- The user can now go into the profile of the search results, and add them as friends, or unfriend them if they are already friends

3.6 User Interface handling

Friendify is a terminal app, and we have intensely used Object Oriented Programming to create the view control of the app.

The app is divided into different screens, for instance home screen, sign in screen, registration screen and many more.

Each screen shown in the app is represented as a Screen object. Each type of Screen has its own class which inherits from the Screen class, and is responsible for keeping track of its view, and deciding what needs to be shown next, that is which screen to show next.

For instance, the HomeScreen inherits from Screen object and is responsible for representing a home screen, similarly there is DocumentationScreen which is responsible for showing any arbitrary document or string as a document.

All the Screen classes are contained in the screen.py file.

Transition between Screens

- Each Screen object inherits a show function. This function is responsible for printing the content of the screen and handling the transition to the next screen.

- The show function always returns a Screen object, which represents the next screen to show. this is how we transition from one screen to another.

- As the app starts running, the current screen is set to the HomeScreen and a loop is set to:

  ```
  current_screen = HomeScreen()

  while current_screen:
      current_screen = current_screen.show()
  ```

  This loop transitions from one screen to the another, but if the current screen is None then the app would stop. This would only happen when the user wants to exit the app.

- To decide which is the next screen to transition to, the show function uses Pyinquirer and cuti, both of which are used to inquire in the python terminal in a beautiful way. Both of them have wide variety of questions we can ask.

- Our program passes a list of questions to Pyinquirer or cuti, about where the user wants to navigate, and they intern ask the question in terminal in an interactive way, and return the answer to our program.

- These questions are either hard-coded in constants.py file or can be created from the generate_question_with_choices and generate_question_multi_choice functions in the constants.py file.

- The program the returns the appropriate screen according the returned answer.

Different Screen objects also contain functions to perform their own computations. For instance the SearchForPeople Screen, screen which searches for query, have a merge sort function which helps in giving the top search results.

Operating Systems

Our app intensively uses terminal commands to make the screen interactive. For instance every time the app transitions from one screen to another, we clear the screen.

But, the terminal apps for windows, mac and linux are different and take in different commands for the same task. For instance to clear the screen of the terminal mac and linux use clear command but windows uses cls command.

We have used python's inbuit library called sys, which has an attribute called platform which returns different values for different operating systems. For instance for mac it return darwin.

We are using this attribute to distinguish between the different systems and use the system specific commands. These are used in Screen.clear_screen function in screen.py and constants.py file.

We even have different messages for different operating systems, as windows operating system does not support emojis and many special characters. This can be seen in constants.py file.

Logo

We are also using pyfiglet library to print beautiful terminal logo for our app. pyfiglet returns a string representation of a piece of text in ASCII characters. The function to print the logo is print_logo in the constants.py file.

Chapter 4

Download/Obtain Datasets

Initially our app would not have any registered users.

Thus, we conducted a survey and registered 50 users with our app.

This data is stored in the firebase cloud.

This data set is NOT used while running the app.

We have still provided the **dataset as a zip file (data.zip)** with our submissions, although it is **not required to run the app.**

The dataset was located in a folder named ”data”. so the path to the dataset is ”data/Survey.csv”

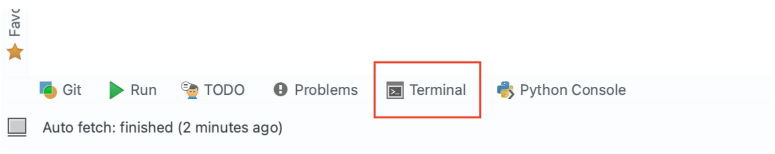
Chapter 5

Running the App

Apart from the instructions provided below, we had made two videos (one for mac and one for windows) on how to use our app.
We recommend watching the VIDEOS first and then reading the instructions (provided below).

1. First download all of the requirements from requirements.txt

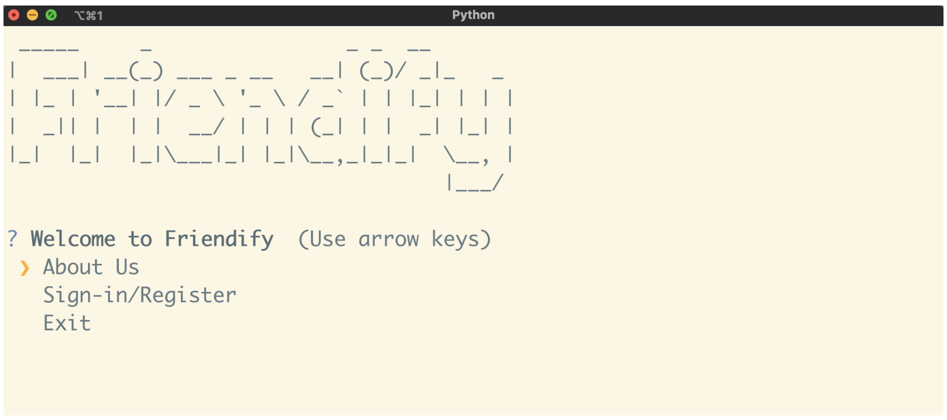
2. Note that this is a terminal app and would **ONLY work in the terminal (and not on python console)**. Open the app in Pycharm and click on the terminal icon on bottom left of the Pycharm screen.



Auto fetch: finished (2 minutes ago)

3. Type "python main.py" in the terminal. **Note that this line could change from system to system. If this does not work you can try "py main.py" or "python3 main.py".**

4. As soon as the line is run in the terminal, the app should start running and this is how the screen should look like:

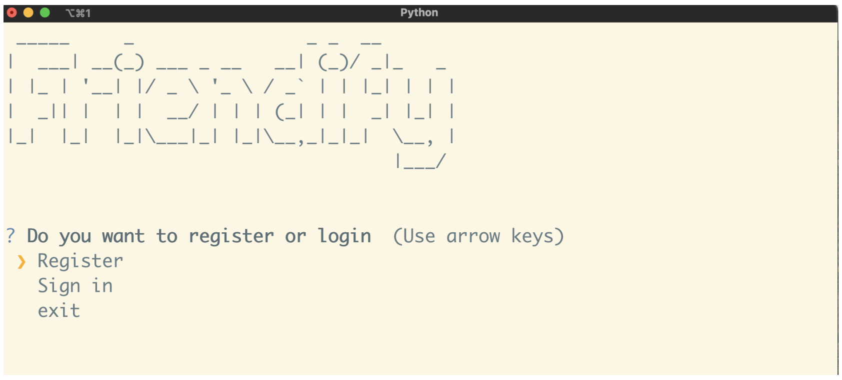


```
 _____     _                _ _  __       
|  ___| __(_) ___ _ __   __| (_)/ _|_   _ 
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/ 

? Welcome to Friendify  (Use arrow keys)
 ❯ About Us
   Sign-in/Register
   Exit
```

This is the Welcome Screen of our app. On the Welcome Screen the user will get 3 options

- About us - which consists of information about the app
- Sign in or register - which is used to sign in or register to the app
- Exit the app

5. Now using arrow keys, go to the Sign-In/Register option and press enter. Now the screen should change and should look something like this:

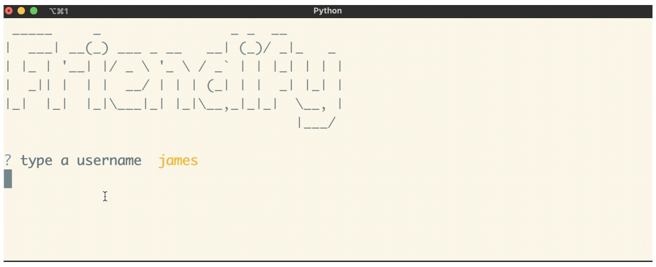


```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/


? Do you want to register or login  (Use arrow keys)
 ❯ Register
   Sign in
   exit
```

6. Now, navigate to register if you are not signed up and otherwise go to Sign in.

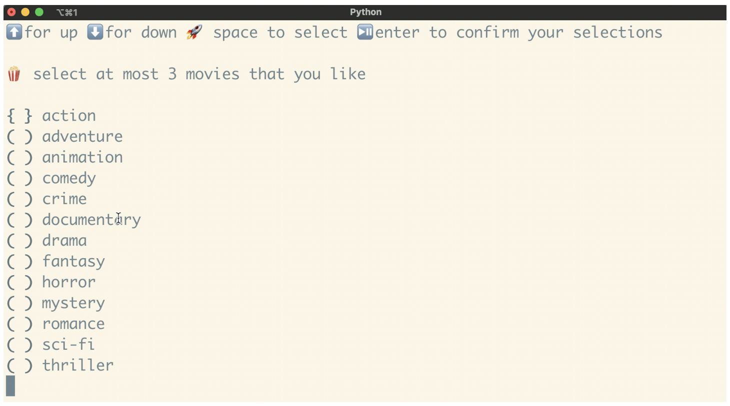
If you clicked Register

(a) If you clicked register, you will be asked to enter a username. This username would be unique to you. This is how the screen should look like:



```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

? type a username  james
```

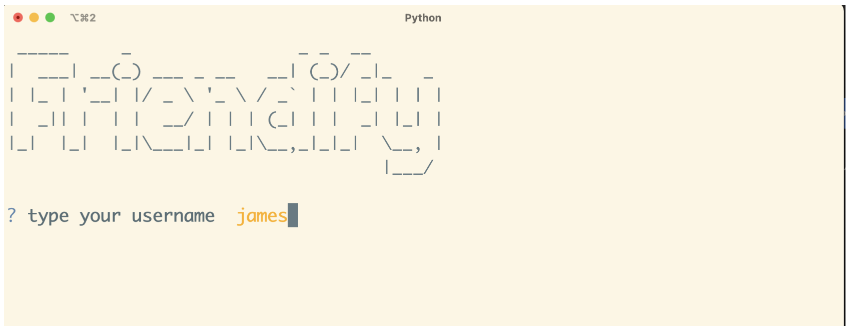
(b) After entering your username, 4 other questions would be asked about your preferences - movies, music, games and the food you like. This is how one of them should look like:- [**You can move up and down using the arrow keys and select an option using the space-bar**]



```
⬆for up ⬇for down 🚀 space to select ⏯enter to confirm your selections

🍿 select at most 3 movies that you like

{ } action
( ) adventure
( ) animation
( ) comedy
( ) crime
( ) documentary
( ) drama
( ) fantasy
( ) horror
( ) mystery
( ) romance
( ) sci-fi
( ) thriller
```

If you clicked Sign In

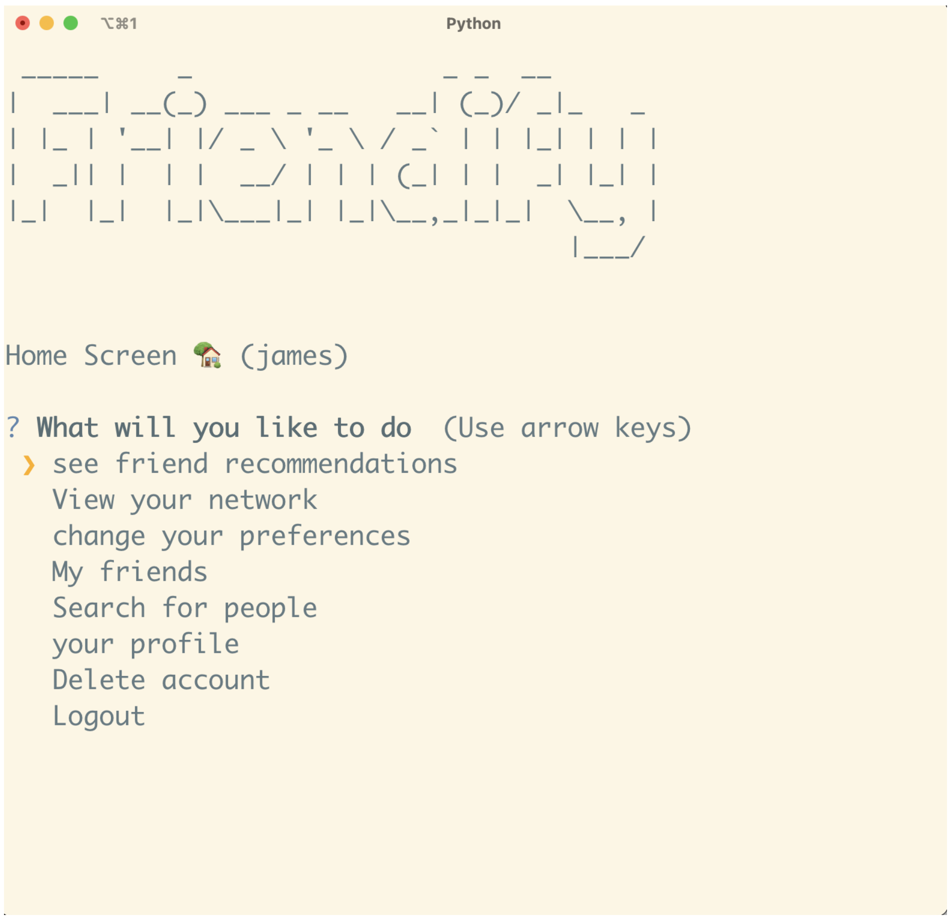
If you clicked sign in, you would only be asked to type the username with which you registered. This is similar to when you are asked to fill in a new username while registering. this is how it should look:



```
 _____     _              _ _  __
|  ___|__(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

? type your username  james
```

After Answering the Registration/Sign-In Questions you would be navigated to your home screen

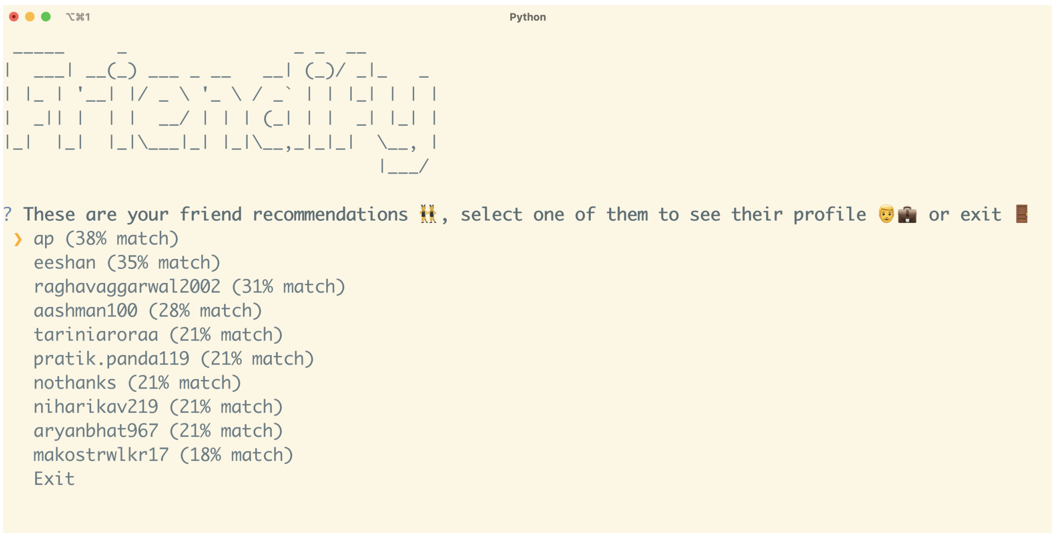
7. Now you should have entered your home screen, and this is how it should look like:



```
 _____     _              _ _  __
|  ___|__(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/


Home Screen 🏡 (james)

? What will you like to do  (Use arrow keys)
 ❯ see friend recommendations
   View your network
   change your preferences
   My friends
   Search for people
   your profile
   Delete account
   Logout
```

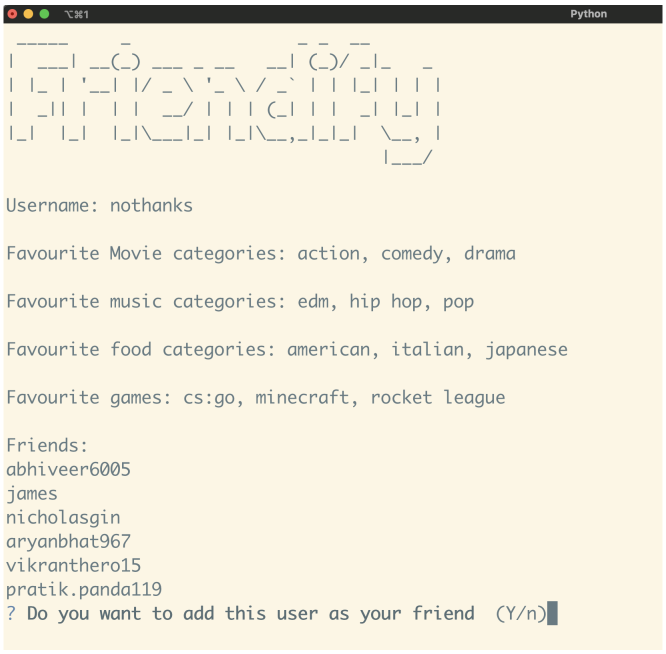
Form your home screen you can:

(a) **see friend recommendations** : navigates you to your friend recommendations. After navigating, you should see a screen which looks like this:



```
 _____     _                _ _  __       
|  ___| __(_) ___ _ __   __| (_)/ _|_   _ 
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/ 

? These are your friend recommendations 👯, select one of them to see their profile 👨‍💼 or exit 🚪
 ❯ ap (38% match)
   eeshan (35% match)
   raghavaggarwal2002 (31% match)
   aashman100 (28% match)
   tariniaroraa (21% match)
   pratik.panda119 (21% match)
   nothanks (21% match)
   niharikav219 (21% match)
   aryanbhat967 (21% match)
   makostrwlkr17 (18% match)
   Exit
```

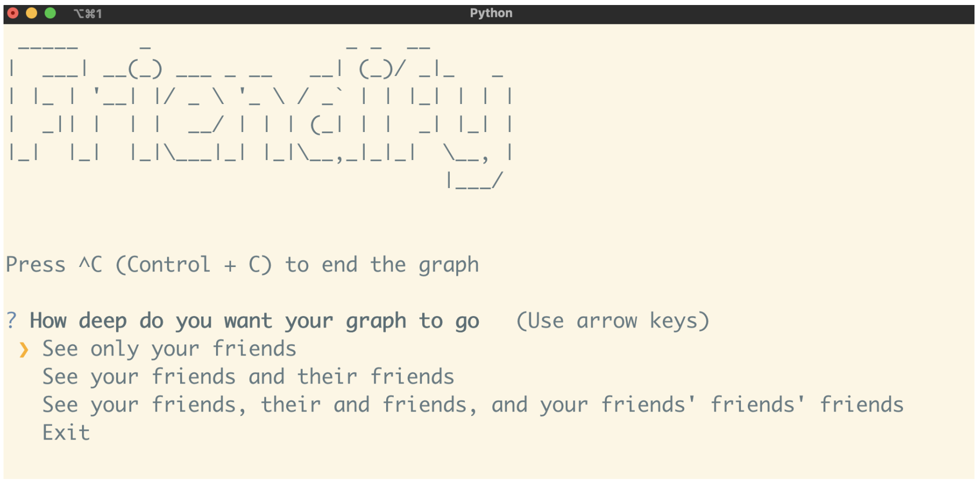
For here you can navigate to any of the users listed below, and this is what you should see:



```
 _____     _                _ _  __       
|  ___| __(_) ___ _ __   __| (_)/ _|_   _ 
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/ 

Username: nothanks

Favourite Movie categories: action, comedy, drama

Favourite music categories: edm, hip hop, pop

Favourite food categories: american, italian, japanese

Favourite games: cs:go, minecraft, rocket league

Friends:
abhiveer6005
james
nicholasgin
aryanbhat967
vikranthero15
pratik.panda119
? Do you want to add this user as your friend  (Y/n)
```

This is the user's profile, which contains all the information about the user. You have the option to add them as your friend. After choosing whether you want to friend the user, you would be navigated to the previous screen containing you recommendations.

(b) **View your network** : This option is used to visualise your friend circle. The option would navigate you to the following screen:

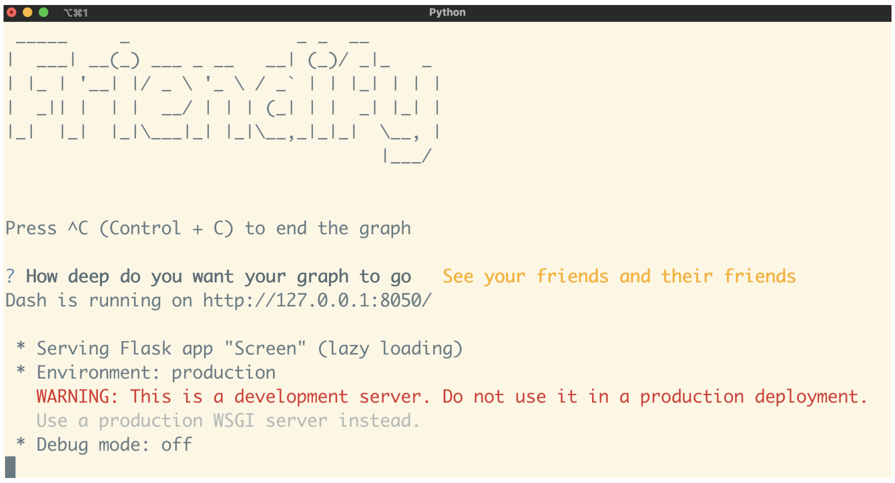


```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/


Press ^C (Control + C) to end the graph

? How deep do you want your graph to go   (Use arrow keys)
 ❯ See only your friends
   See your friends and their friends
   See your friends, their and friends, and your friends' friends' friends
   Exit
```

From here you can choose from the three options, of seeing only your friends, or your friend and their friends or or your friend, their friends, and their friends.

After selecting an option, you would automatically be navigated to a web-browser. If you are not just enter http://127.0.0.1:5050/ in your web-browser.

After viewing your visualisation in the browser, you can come back to the app. This is how it should look:

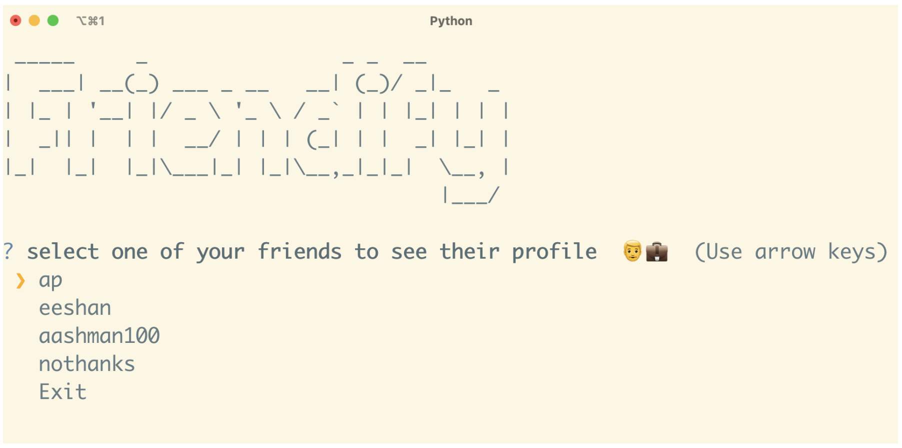


```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/


Press ^C (Control + C) to end the graph

? How deep do you want your graph to go   See your friends and their friends
Dash is running on http://127.0.0.1:8050/

 * Serving Flask app "Screen" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
```

End the visualisation by pressing Command + C (on windows) or Control + C (for mac).After you exit this screen, you would be navigated to the home screen.

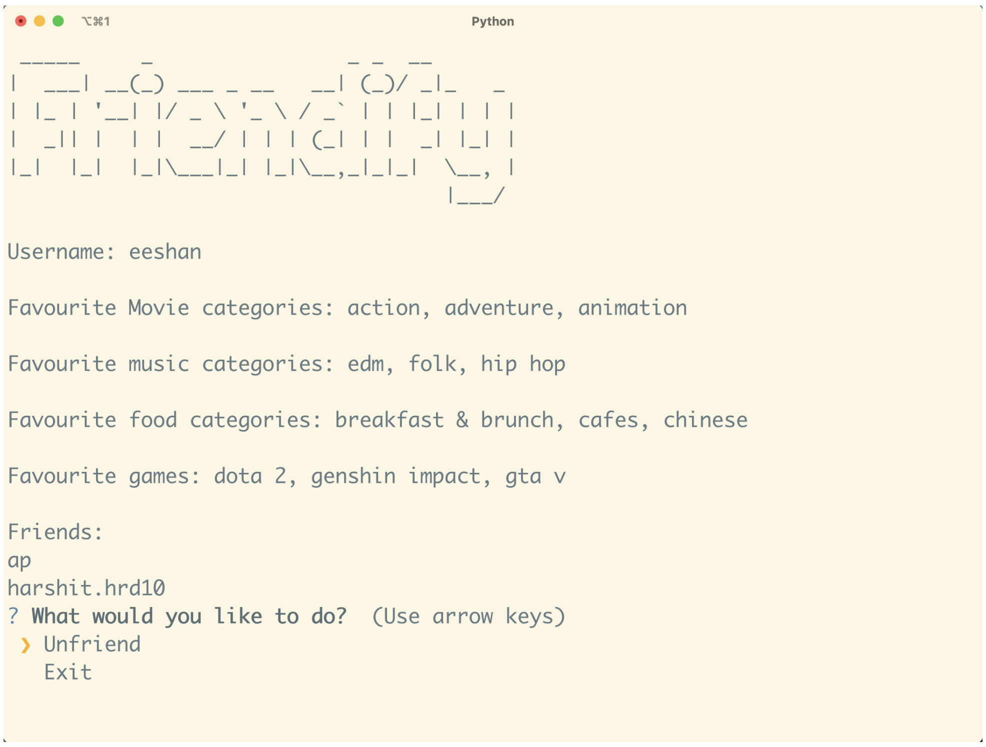
(c) **change your preferences** : this feature is used to change the preferences you choose during registration.

(d) **My Friends** : this feature is used to view all your friends, their profile and handle them. You would be navigated to a screen like this:



```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

? select one of your friends to see their profile  👨‍💼  (Use arrow keys)
 ❯ ap
   eeshan
   aashman100
   nothanks
   Exit
```

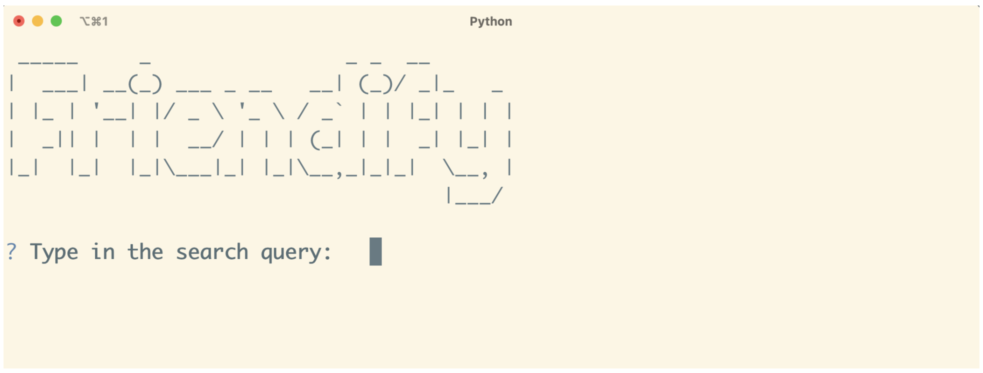
From here you can navigate to any of the friends and view their profile. This is how it should look:



```
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

Username: eeshan

Favourite Movie categories: action, adventure, animation

Favourite music categories: edm, folk, hip hop

Favourite food categories: breakfast & brunch, cafes, chinese

Favourite games: dota 2, genshin impact, gta v

Friends:
ap
harshit.hrd10
? What would you like to do?  (Use arrow keys)
 ❯ Unfriend
   Exit
```

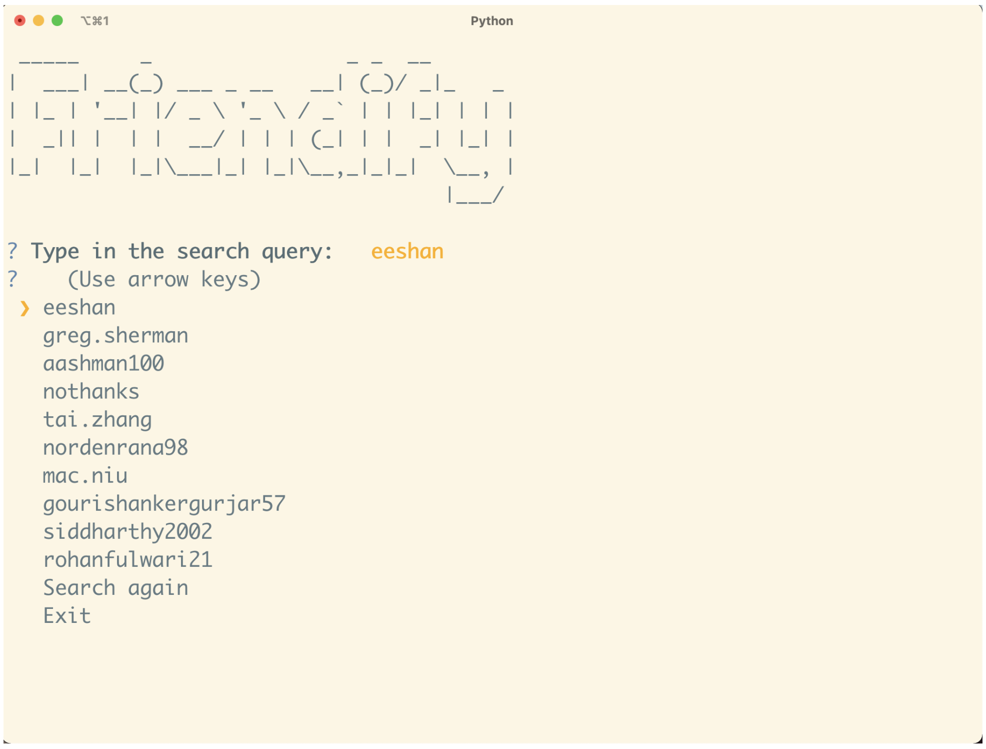
From here you would have the option to unfriend the user, or exit. If you choose to unfriend the user, the user would be removed from your friend list and you would be navigated back to your friends list.

(e) **Search for people** : This feature allows you to search for people using the app. As you select the feature you would be navigated to a screen which looks like:



```
 _____     _                _ _  __       
|  ___| __(_) ___ _ __   __| (_)/ _|_   _ 
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/ 

? Type in the search query:
```

You have to type the username of the user you are trying to search for, and you will get the top 10 results matching that query. This is how it looks:

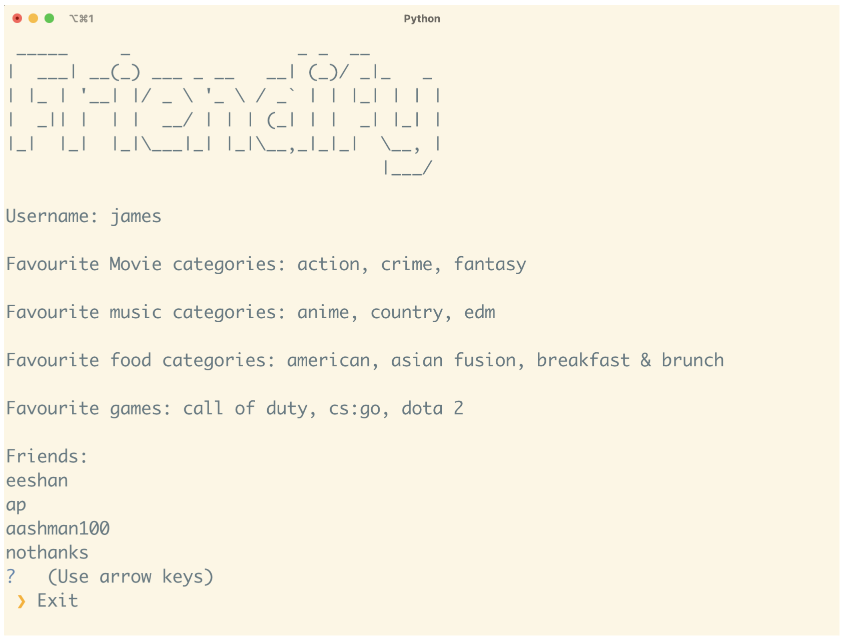


```
 _____     _                _ _  __       
|  ___| __(_) ___ _ __   __| (_)/ _|_   _ 
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/ 

? Type in the search query:   eeshan
?   (Use arrow keys)
 ❯ eeshan
   greg.sherman
   aashman100
   nothanks
   tai.zhang
   nordenrana98
   mac.niu
   gourishankergurjar57
   siddharthy2002
   rohanfulwari21
   Search again
   Exit
```

Form here you can navigate to any of the users. The screen to which you would navigate would contain the profile of the user and the option to friend and unfriend the user depending on whether they are your friend or not.

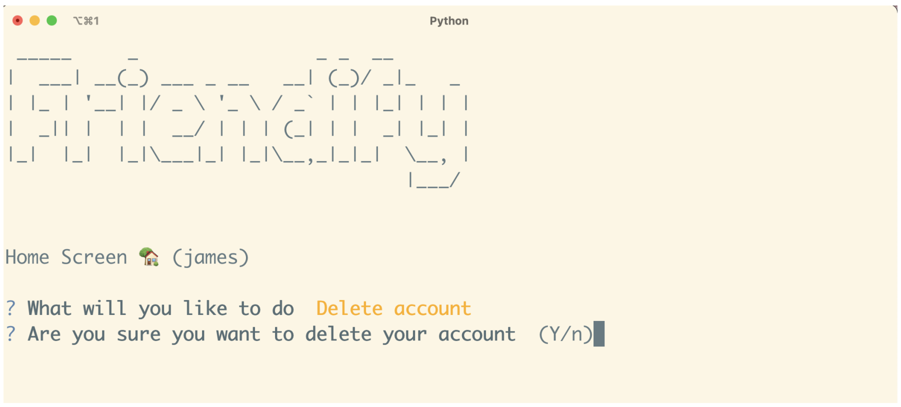
You also have the option to search again or exit the screen and go back to the home screen.

(f) **your profile** : this feature shows your profile as seen by other users. this is how it looks:

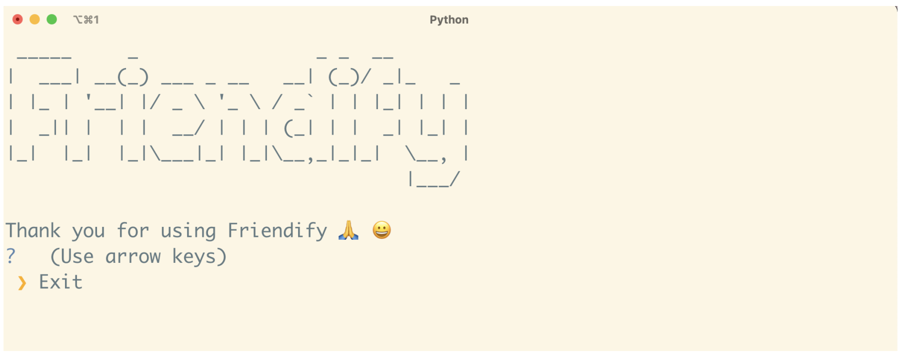


```
Python
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

Username: james

Favourite Movie categories: action, crime, fantasy

Favourite music categories: anime, country, edm

Favourite food categories: american, asian fusion, breakfast & brunch

Favourite games: call of duty, cs:go, dota 2

Friends:
eeshan
ap
aashman100
nothanks
?   (Use arrow keys)
 ❯ Exit
```

(g) **delete account** : Finally you can delete your account by choosing this option. This is how your screen should look:



```
Python
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

Home Screen 🏡 (james)

? What will you like to do  Delete account
? Are you sure you want to delete your account  (Y/n)
```



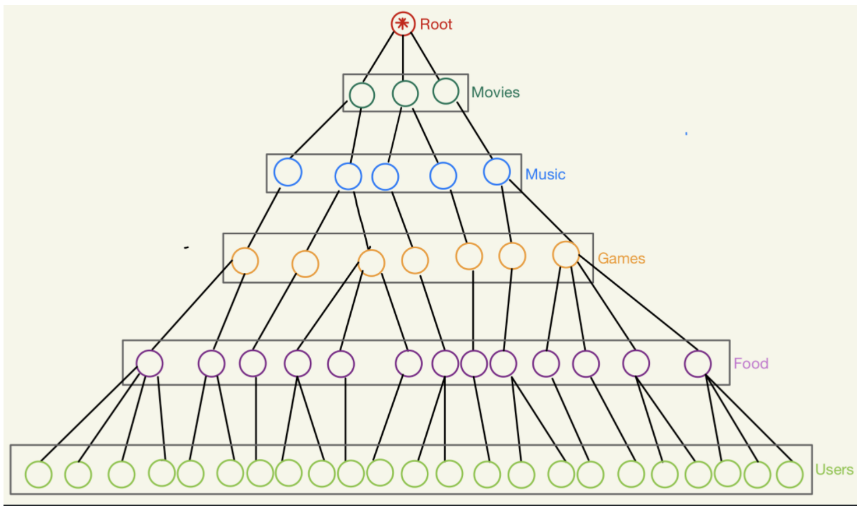
```
Python
 _____     _                _ _  __
|  ___| __(_) ___ _ __   __| (_)/ _|_   _
| |_ | '__| |/ _ \ '_ \ / _` | | |_| | | |
|  _|| |  | |  __/ | | | (_| | |  _| |_| |
|_|  |_|  |_|\___|_| |_|\__,_|_|_|  \__, |
                                    |___/

Thank you for using Friendify 🙏 😀
?   (Use arrow keys)
 ❯ Exit
```

From here you would be navigated to the sign-in/register screen.

Chapter 6

Changes from Phase 1 Proposal

- The biggest change we added is the method of recommending friends. Previously our plan was to build up a recommendation tree which looked something like this:



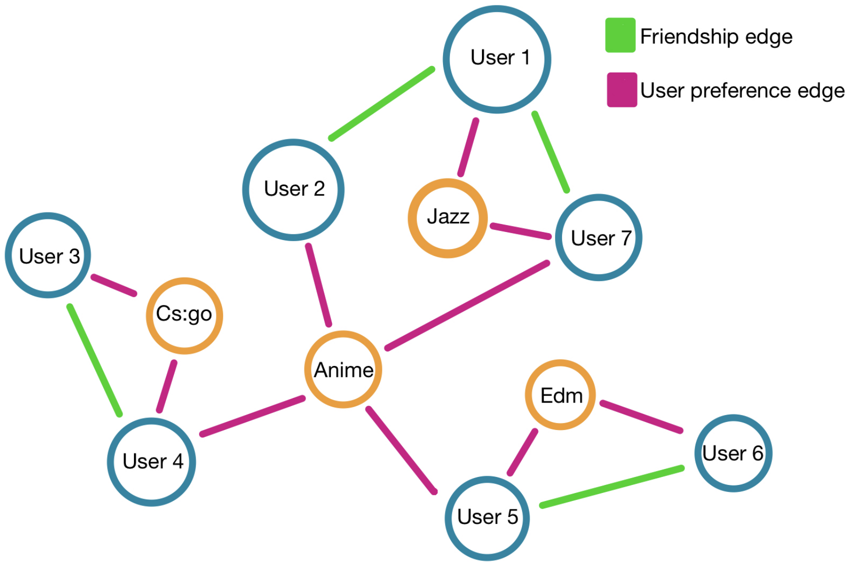
We concluded that this approach was not very useful in our case.

This is because the order of the preferences mattered.

For instance, using the tree method (as suggested in Phase 1), a user would get different recommendations based on whether his preferences of movies were placed above or below his preferences of games in the recommendation tree.

This led to some of the preferences, not being searched in the tree, which in-turn led to poor recommendations and poor matching scores.

Instead, by using graphs to recommend friends, every possible preference was taken into account while recommending friends. This is how it looks now:



- We also added the feature to search for people using the app - which was not previously present in our proposal

- Further, we added the feature of displaying the similarity score while showing/visualizing our friend recommendations.

- We also added the feature of visualizing our friend network in different architectures (of graph) - like cose, circle, bredthfirst etc.

- Additionally, we added the feature to handle different friends and their user accounts.

Chapter 7

Discussion

The main problem which people face during this pandemic is connecting with people and making friends.
We believe that the results and working of this app does help address the problem which we introduced at the beginning of our proposal.
Our app helps people connect with each other based on their interests and preferences. Using our app, people can see their friend recommendations, visualize their friend network as a graph, add/remove friends, change their personal preferences (movies, food, games etc.) and do a lot more.
Such healthy online interaction is vital for the mental health and well-being of an individual in this time of social isolation.
In view of the above points, we feel that our app helps address this issue in a meaningful way.

Some limitations/problems we encountered -

- Initially, we could not find any dataset online, hence we had to conduct a survey. The response to our survey was also not great, since approximately 50 people responded.
- To make the online app, we had to use firebase. We encountered several HTTP errors in order to make this an online application. Also, we were not able to use authentication via passwords. since firebase_admin (official firebase python lib) did not have those options for python (or we were not able to find it).
- At the beginning, we were using trees to recommend friends. This led to some of the user preferences, not being searched in the tree, which in-turn led to poor recommendations and poor matching scores. Later, by using graphs to recommend friends, we were able to account for every possible user preference while recommending friends
- While using the dash app for visualization, the visualization would not open automatically. Thus, we had to find a way open it using the web-browser library. We are still not able to shut the site down automatically, because the web-browser library can not do that. We can instead kill the localhost site using some terminal commands.
- Currently, our app is a terminal app and the UI runs on the terminal. We tried to run it on the the console, but since Pyinquiere and Cutie make use of the

terminal, this process was difficult for us to implement and thus we were unable to do this.

Future exploration -

- We plan on adding authentication via password to all user accounts to increase security.
- We also plan on introducing a feature which would allow users to make their accounts public or private, based on their choice. This feature would help users maintain their privacy.
- We plan on adding a ‘friend request’ feature, instead of directly allowing the users to add other people without their permission.
- We also plan to add a built-in chat system in the future.

Chapter 8

References

Libraries:

- Firebase admin
- PyInquirer
- Cutie
- Pyfiglet
- Dash and Plotly

Sources:

- Source 1 - https://www.healthline.com/health-news/people-with-covid-19-more-likely-to-develop-depression-anxiety-and-dementia#How-the-new-coronavirus-affects-the-mind
- Source 2 - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7306546/#: :text=Quarantine%20and%20social%20distancing%20are,and%20mental%2Dhealth%20related%20reperc
- Source 3 - https://www.kff.org/coronavirus-covid-19/issue-brief/the-implications-of-covid-19-for-mental-health-and-substance-use/
- Source 4 - https://oaksatdenville.org/blog/benefits-social-interactions/