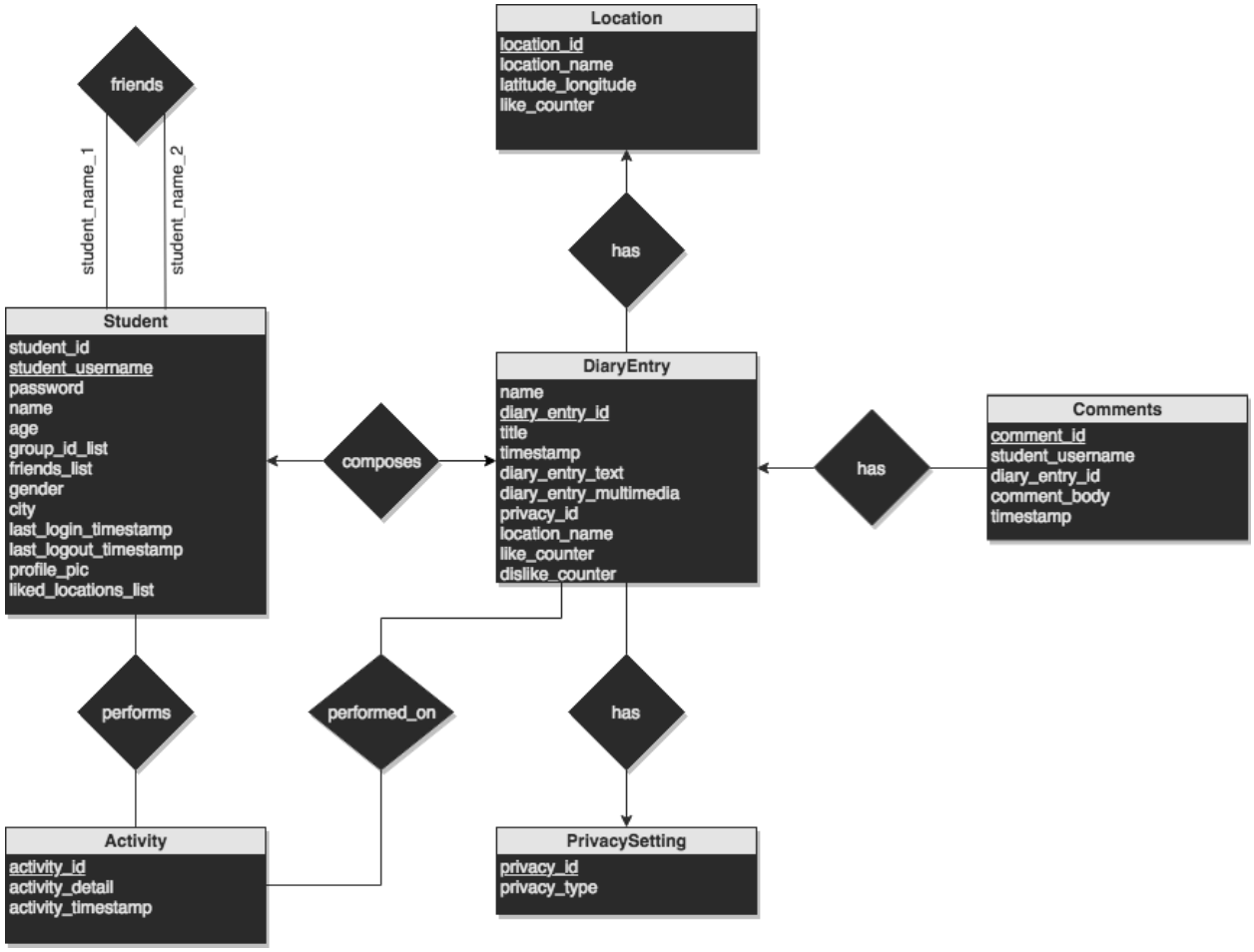
Principles of Database Systems Project

Vikram Sunil Bajaj (vsb259), Ameya Shanbhag (avs431)

We have chosen to create a web application for the **University Students Social Network** use case. Part 1 of this project deals with the designing of a database for the backend of the web application. Here, in Part 2, we will be developing the User Interface for the backend of the web application that we made in Part 1.

Table of Contents

1. Entity-Relationship Diagram



The diagram above illustrates the updated ER diagram for our database.
As we know, undirected lines denote many-to-many relationships and directed lines represent one-to-many or many-to-one relationships (with the arrow pointing to the *One* entity).

2. Relational Schemas

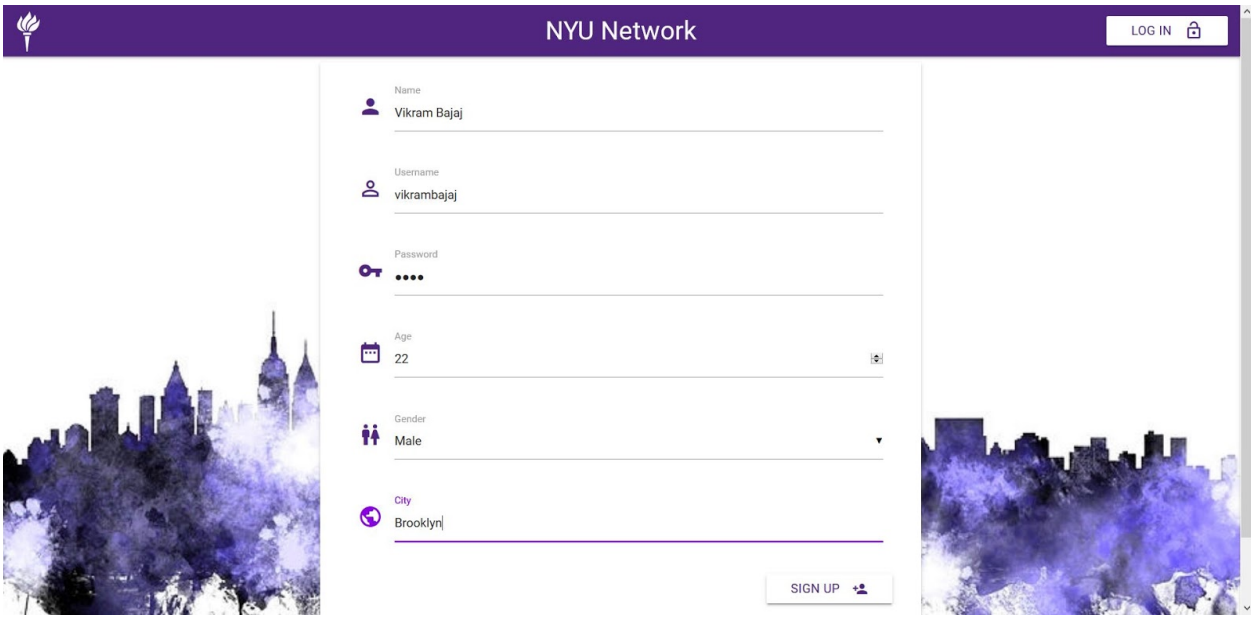
- Student(<u>student_username</u>, student_id, password, name, age, group_id_list, friends_list, gender, city, last_login_timestamp, last_logout_timestamp, profile_pic, liked_locations_list)
- Location(<u>location_id,</u> location_name, latitude_ longitude, like_counter)
- PrivacySetting(<u>privacy_id</u>, privacy_type)
- DiaryEntry(<u>diary_entry_id</u>, name, title, timestamp, diary_entry_text, diary_entry_multimedia, privacy_id, location_name, like_counter, dislike_counter)
- Comments(<u>comment_id</u>, student_username, diary_entry_id, comment_body, timestamp)

- Friends(student_name_1, student_name_2, timestamp, status)
- Activity(activity_id, activity_detail, activity_timestamp)

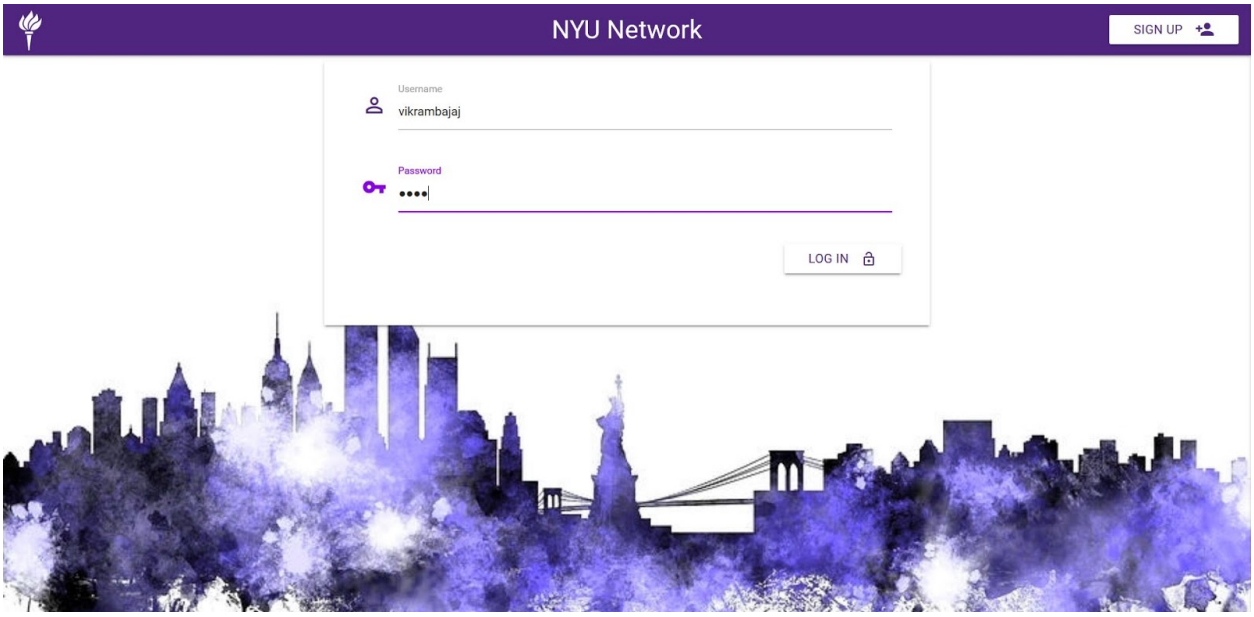
3. Web App Design

3.1. Login and Signup Pages

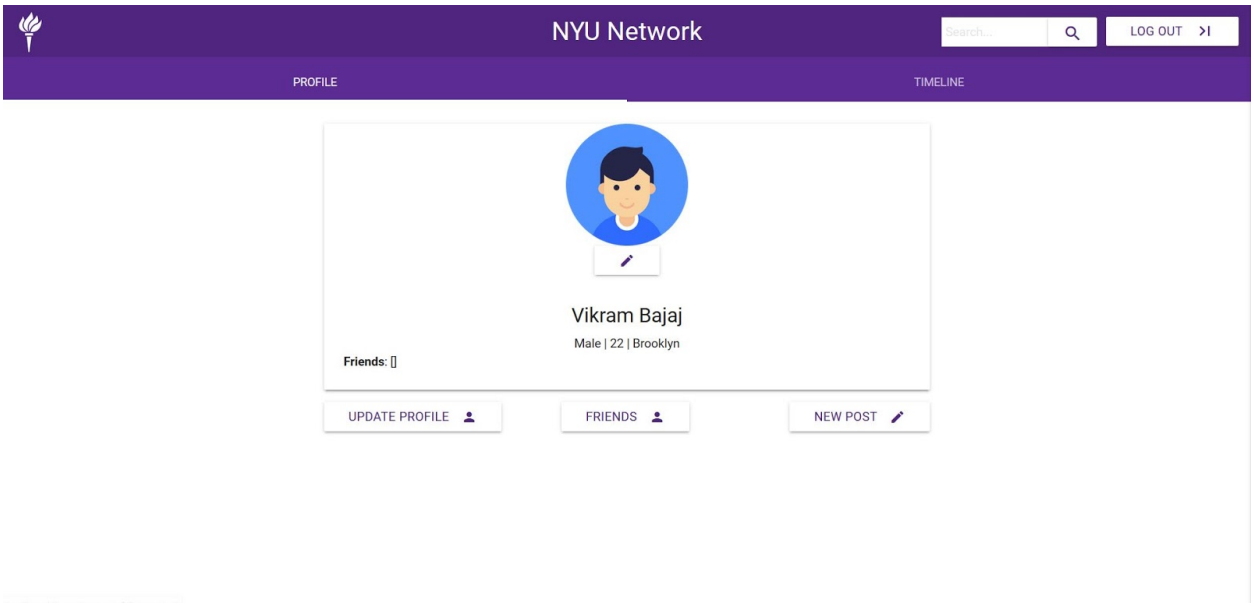
- We have two pages, one for login and one for signup, wherein we will be using the stored procedures that we had created in Project Part 1 for signing up new users and logging in the users registered with the network.
- Signup page will get the data from the input fields and insert it to our MySQL database.
- Login page will check the username and password that user has entered against the values our database. Once logged in, user session will be started and will be destroyed once logged out.
- A successful login will redirect the user to the homepage. New users are given a default profile picture based on gender.



Sign Up Page



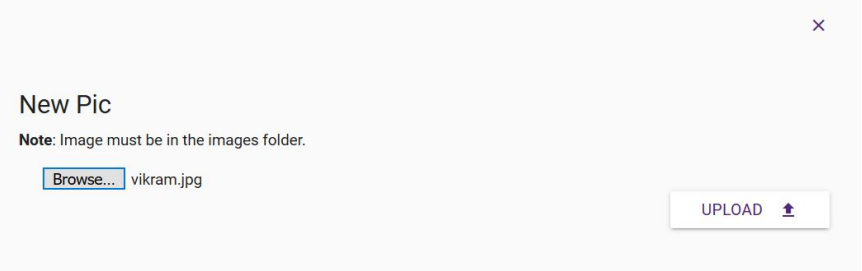
Login Page



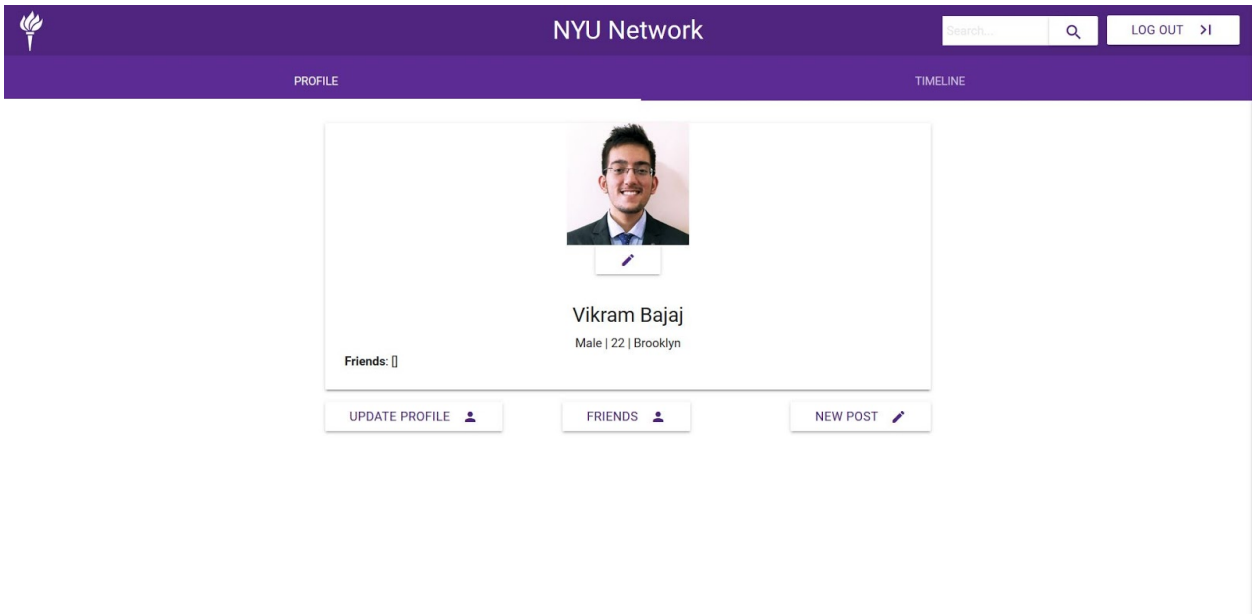
Profile Page upon successful Login

3.2. Edit Profile

- We allow the users to update their profile pictures:

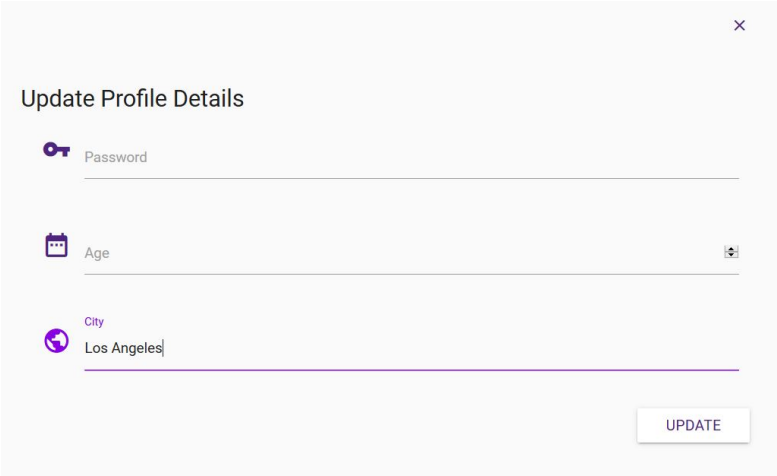


Uploading a new Profile Pic



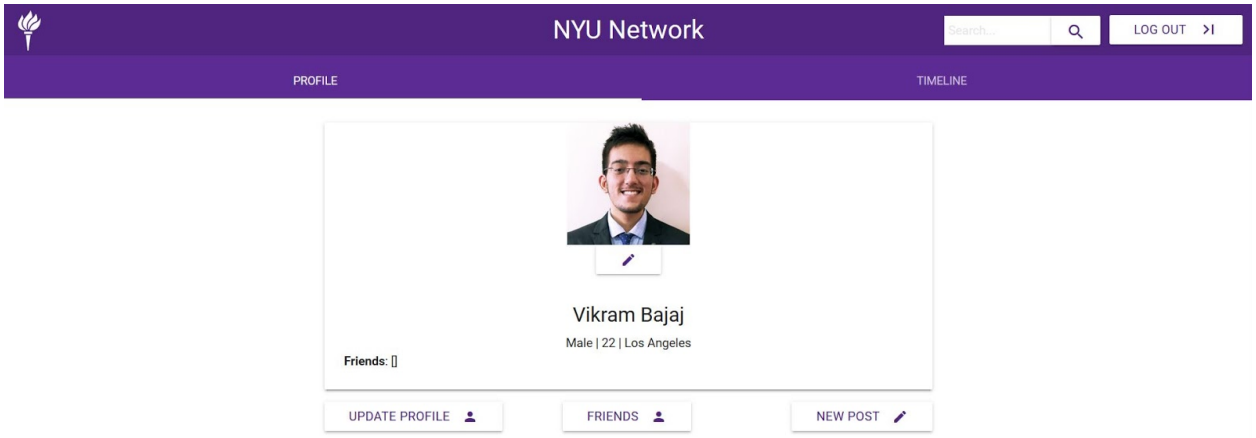
Profile Pic Updated

- By clicking on ‘Update Profile’, a user can update his/her password, age and city. As it is not necessary to someone to stay in the same city or have the same password for decades, edit profile will help them change their details.



Update Profile Details

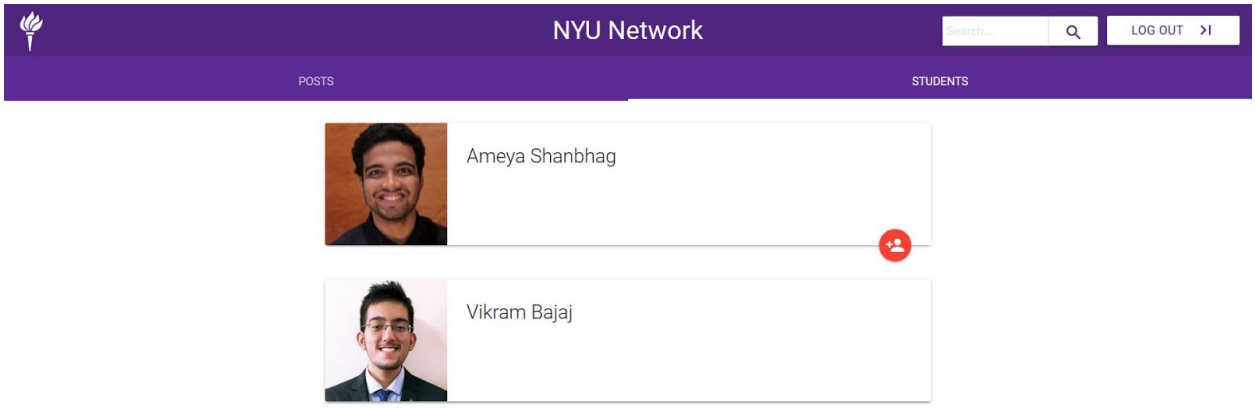
As shown below, the location gets updated:



3.3. Friend Requests

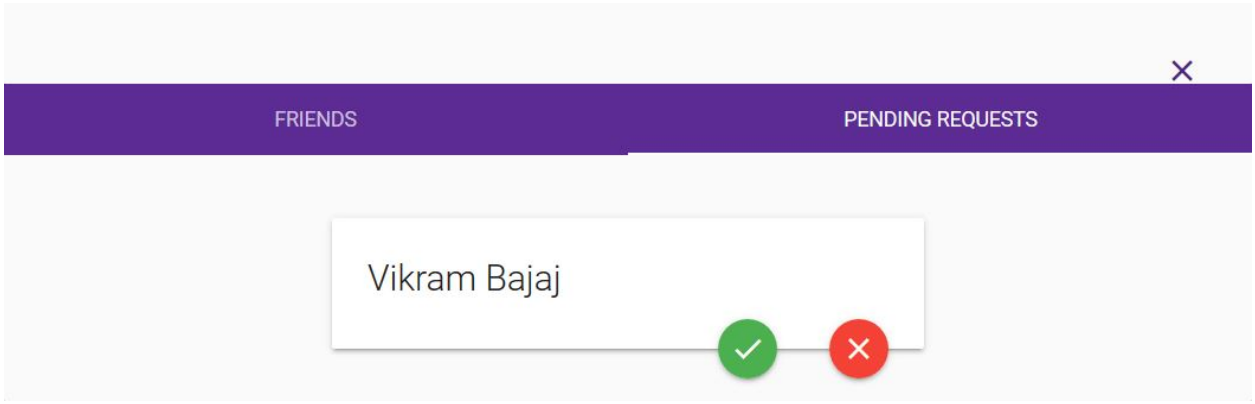
3.3.1. Pending Request

- A user can search for another user using the search bar which will then display cards from where he/she can send a friend request to someone who is not his/her friend (also, a user will not be allowed to send himself/herself a friend request). Searching for “am” in the search bar retrieves the following students:



As shown, the user (Vikram Bajaj) cannot add himself as a friend. On clicking the add button, a “Friend Request Sent” alert is displayed and the status in the Friends table is set to “Pending”

- Once a user A sends a friend request to user B, B will receive a pending request from A in his/her Friends tab. B can choose to accept/decline the request. If B declines the request, the status in the Friends table is set to “Declined” and no further action is performed.



3.3.2. Accept Request

- Once a user B accepts the request from user A, user B will be added to the friend’s list of user A and vice-a-versa and the pending request notification will be

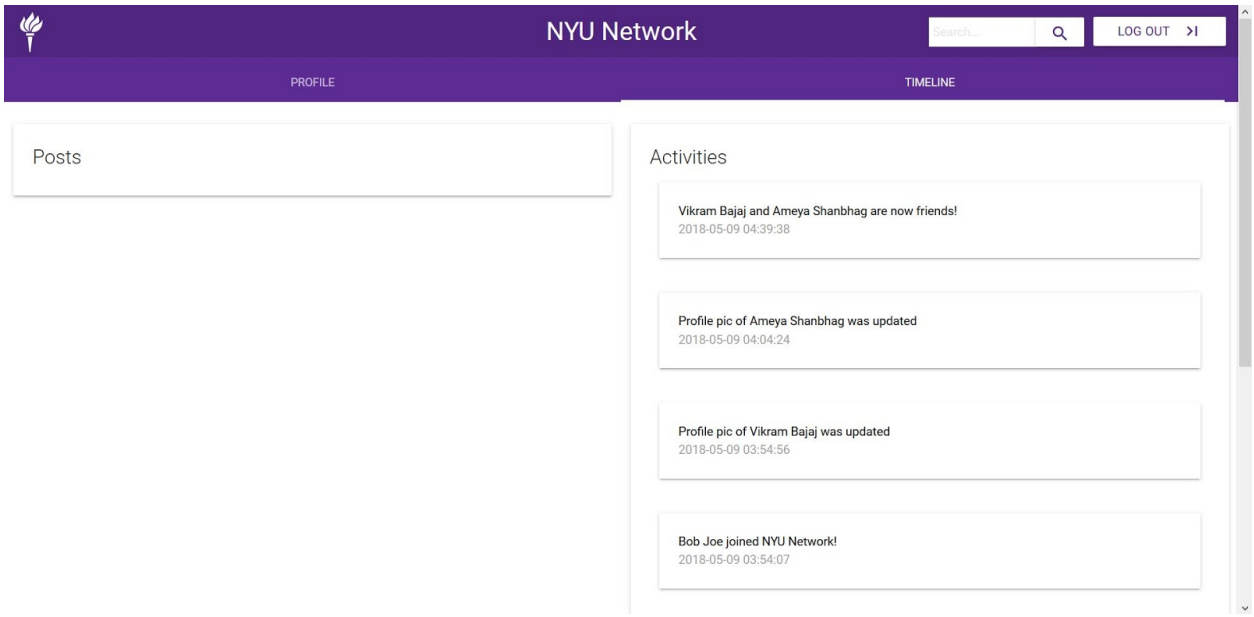
removed. Also, the status in the Friends table is updated to “Accepted”.

3.3.3. Decline Request

- If a user B declines the friend request sent by user A, neither of them will be added to each other’s friends lists and the pending request notification will no longer be shown.

3.4. Activity

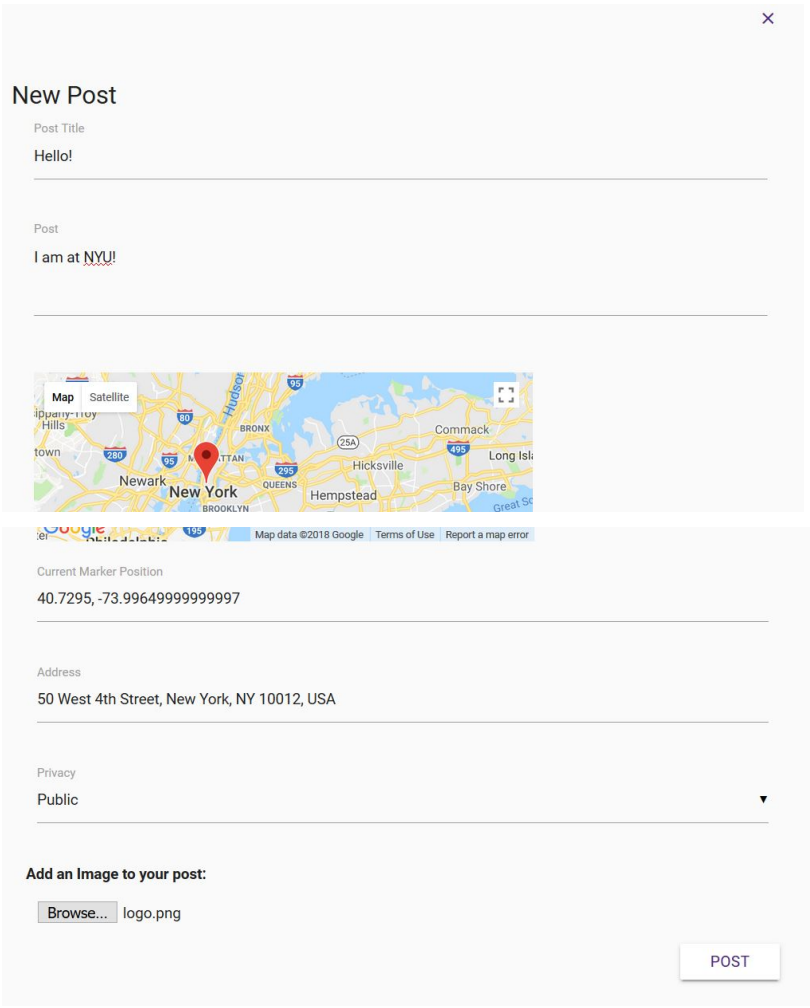
- This is a part of our timeline feature. Activity for user A will keep the user A updated about the activities that are being performed by the other users.
- Different types of activities include:
    - A user joining NYU Network
    - A user “friending” another user
    - A user updating his/her profile picture
    - A user adding a new post
    - A user liking/disliking another user’s post
    - A user checking into a location
- Activities are displayed in reverse chronological order.



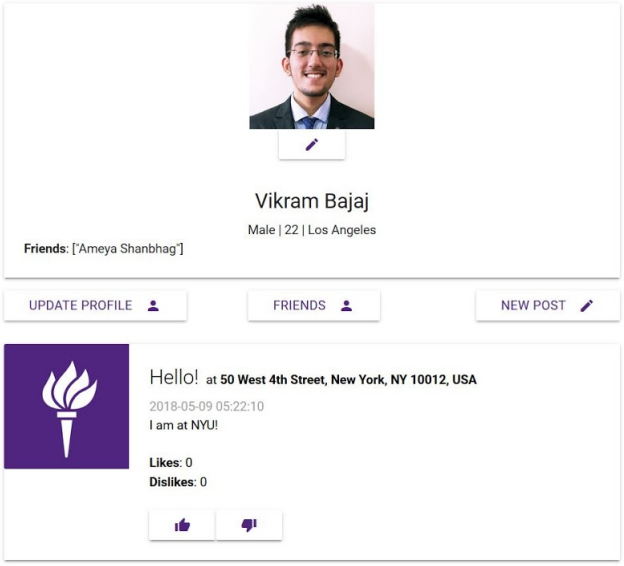
3.5. Diary Entry (Post)

- Each user can add a post wherein he/she can add the post title, post body, post multimedia (image) and location, and set the privacy level to private, friends, friends of friends or public.

  To provide a location, the user can adjust the marker on the map according to where he/she wants their post location to be, and using Google Maps API, the marker will automatically display the location of the marker.

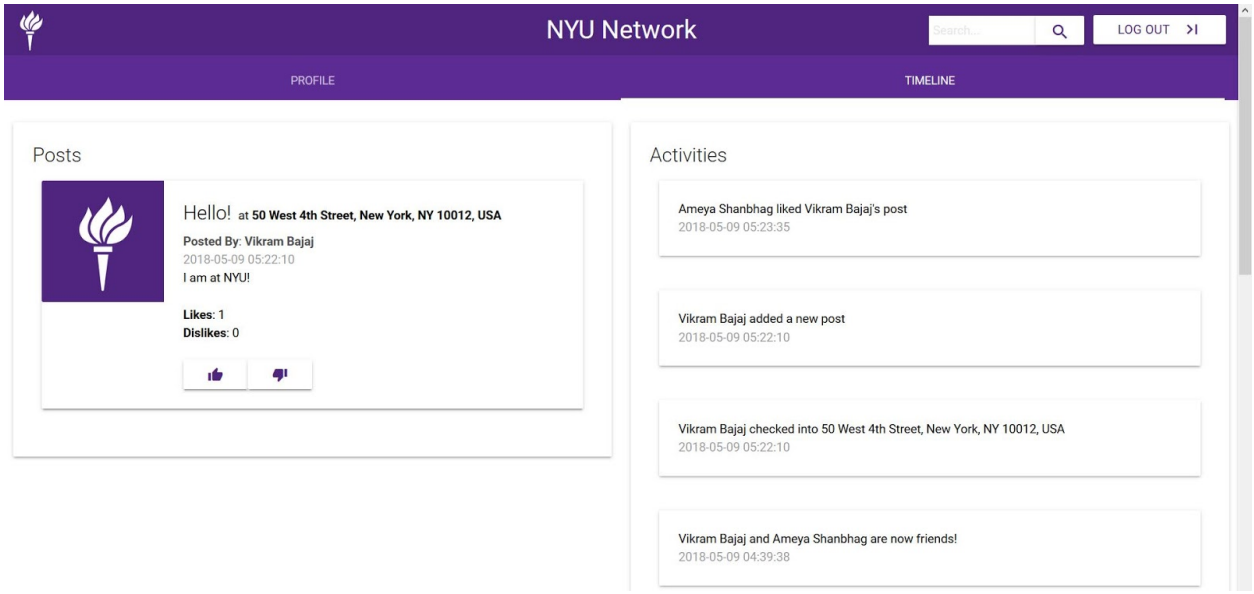


- Once posted, the diary entries will appear in the posts section of the user. Posts are shown in reverse chronological order.



3.6. Like and Dislike

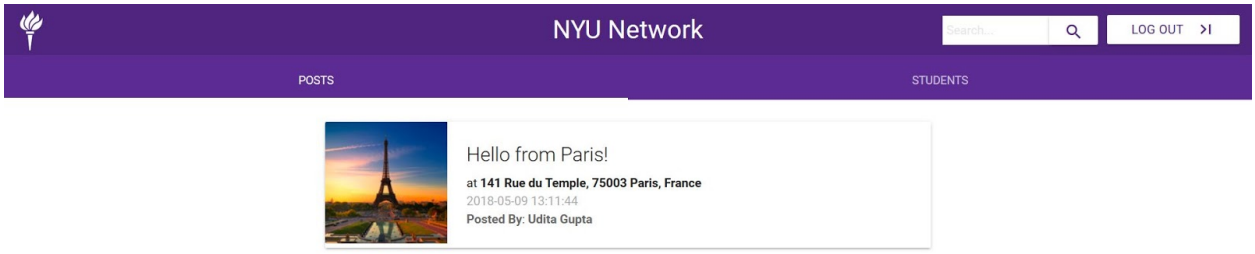
- Each post has a like and a dislike counter which will be updated based on user interaction.



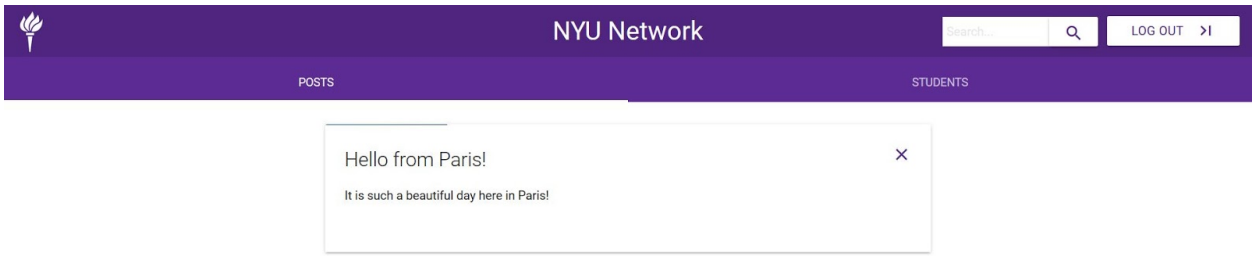
Ameya Shanbhag liked Vikram Bajaj's post

3.7. Search

- The search bar at the top right corner allows the user to search for anything that is there on the social network including friends' posts (with the title or body containing the search term), friends or users who aren't friends. Only posts that aren't private will be displayed.
- This is very useful when it comes to adding a new friend to your network.
- Searching for "paris" will retrieve the following post:

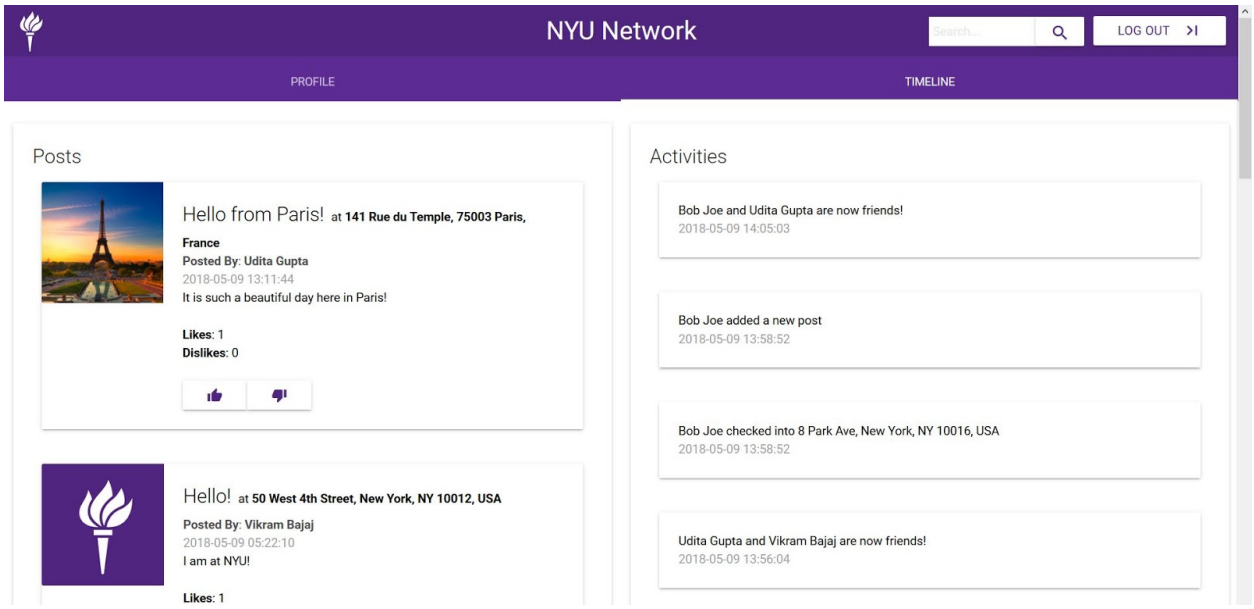


The post body is revealed upon clicking the post title.



3.8. Timeline

- This will allow the user to be updated about the activities that are being performed by students in the network and will also display posts from his/her friends.
- Our timeline works similar to Facebook's timeline. We have two columns, one of them displays the posts from the student's friends (as well as the current user's posts) and the other shows activities that are being performed by all the students in network.



4. User Flow

- We have kept the user flow very intuitive and user-friendly. A student will be presented with a login page where he/she can login with his username and password if he/she is a part of the network, else he/she will also have an option of signing up as a new student in the network.

- Once logged in, he/she will be directed to his/her personal profile page where there are lots of options like editing profile pic, changing profile details, writing a new post.
- If there are other students on the network and the user wants to be friends with them, he/she can search for them on the network and send them a friend request.
- It then depends on the other user whether to accept the friend request or to decline it.
- Upon accepting the request, the user will be added to the accepted user's friend list and vice-a-versa. Also, the pending request notification is removed.
- Upon declining the request, there will be no action taken and the pending request notification will be removed.
- Using the search tab, the user can also search for posts that contain a specific keyword.
- Each post will have a like, dislike counter which help user express his emotions toward the post. Upon clicking either of the buttons, the respective counter will be updated.
- Each user will have his own timeline, where he can view all the posts by his friends and also all the activities performed on the network

5. Changes from Project Part 1

- To start with, the ER diagram from Part 1 was modified as we realized that the attributes which we thought of to be the primary keys were not supposed to be.
- This resulted in our relational schemas being modified as well.
- Our part 1 talked about manually specifying the location of the post but in this part, with the help of Google Maps API, we just have to point the marker to the desired location and the API will display the address of the location.
- Instead of having two activities with more attributes just for storing the activities, here we have merged the two tables, and with very less attributes, we are able to store the activities performed by users on the network.
- We also didn't take into consideration the dislike counter in our previous project submission. So here, along with like, we also have a dislike counter which will keep a track of the number of users who have disliked a post.