Diana Danh, Eduardo Emeregildo, Maliha Tabassum

Group D

CSc 33200: Project Design report

April 29, 2020

Table Of Contents

1. Introduction

The purpose of this document is to provide the overview of design and functionality of the system.

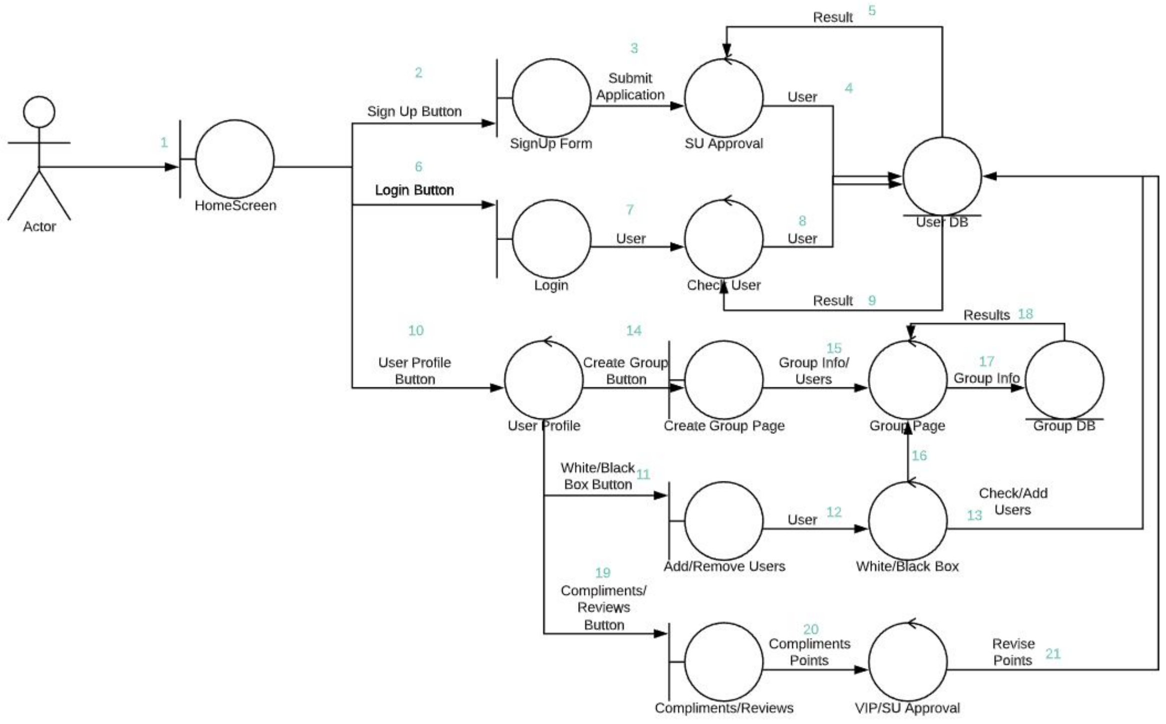


Figure 1.1: Collaboration Class Diagram of Overall System

2. All Use Cases

Homepage

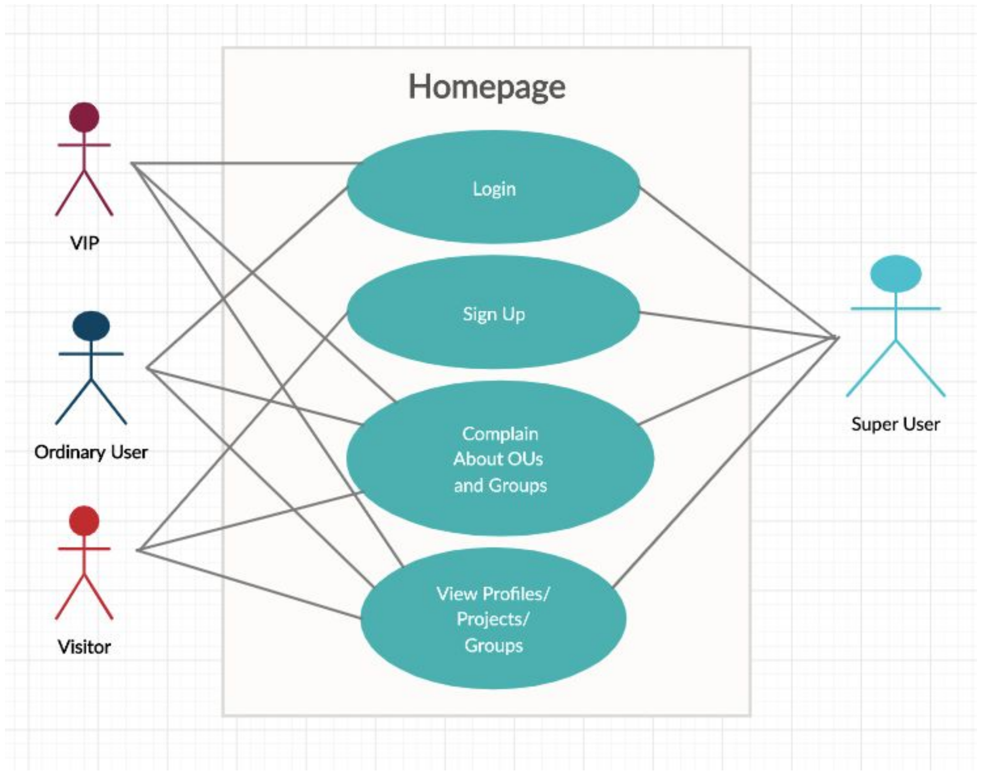

Homepage
VIP
Ordinary User
Visitor
Login
Sign Up
Complain About OUs and Groups
View Profiles/ Projects/ Groups
Super User

Figure 2.1: Homepage Use Case

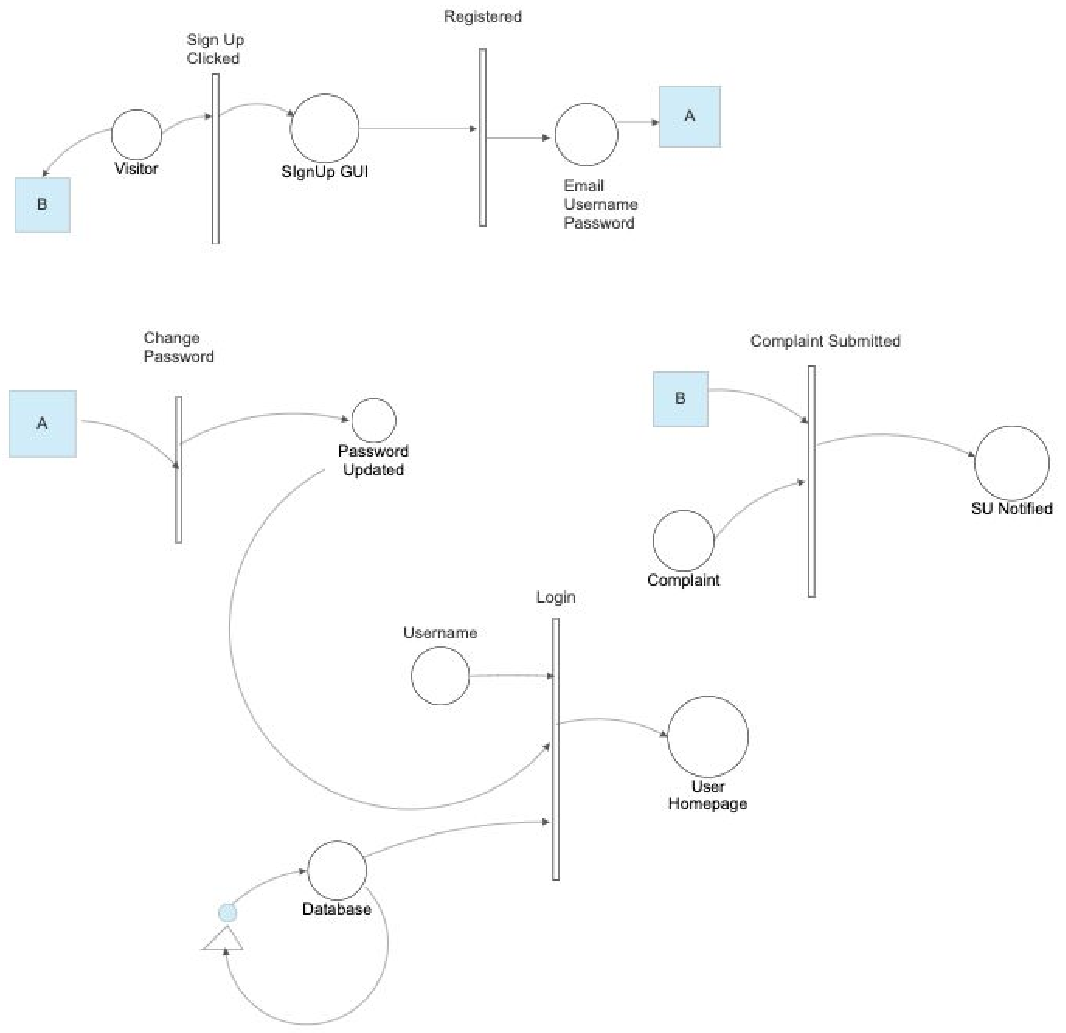


Figure 2.2: Homepage Petri Net

Signup Page

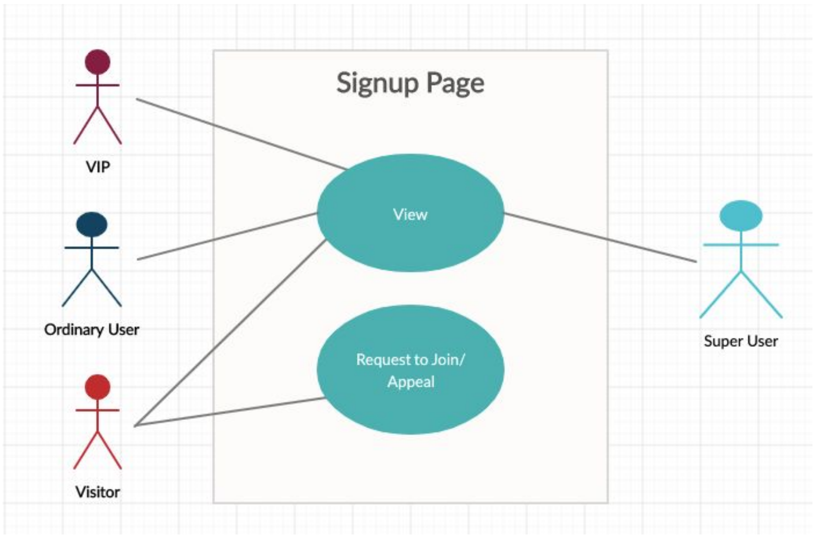


Figure 2.3: Signup Page Use Case

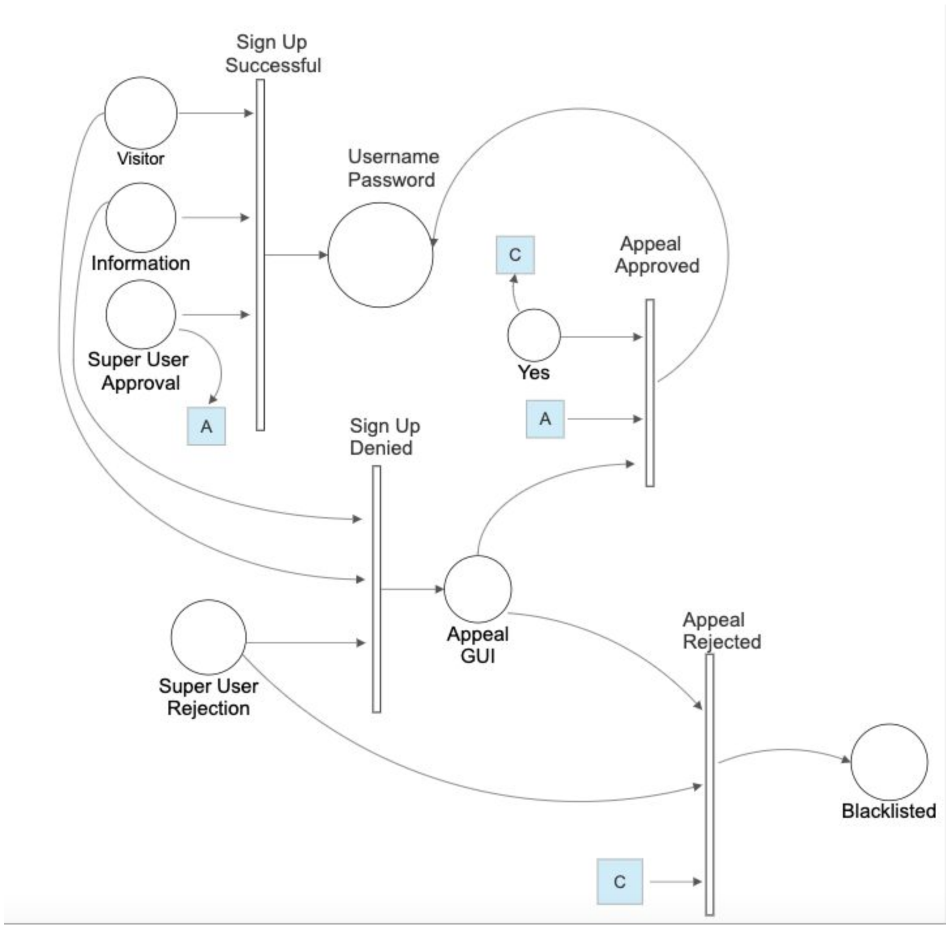


Figure 2.4: Signup Page Petri Net

OU Homepage

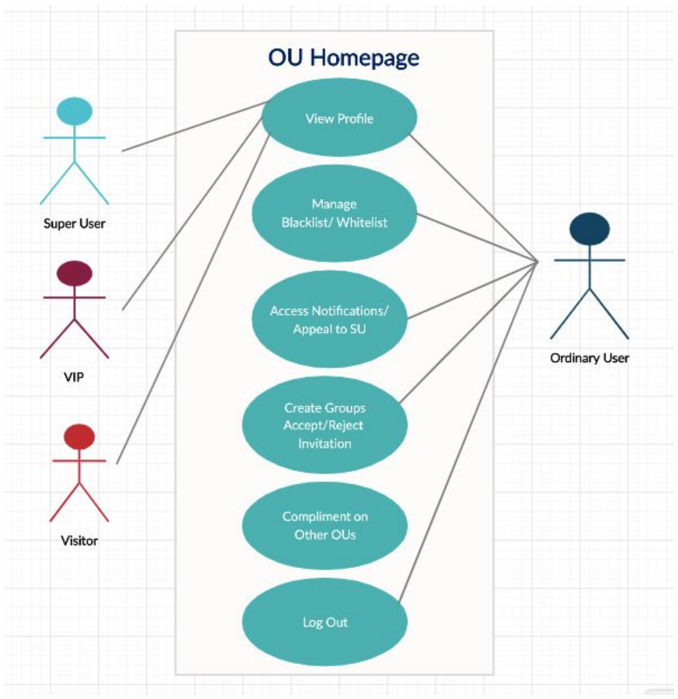

OU Homepage
View Profile
Manage Blacklist/ Whitelist
Access Notifications/ Appeal to SU
Create Groups Accept/Reject Invitation
Compliment on Other OUs
Log Out
Super User
VIP
Visitor
Ordinary User

Figure 2.5: Homepage (OU) Logged In View Use Case

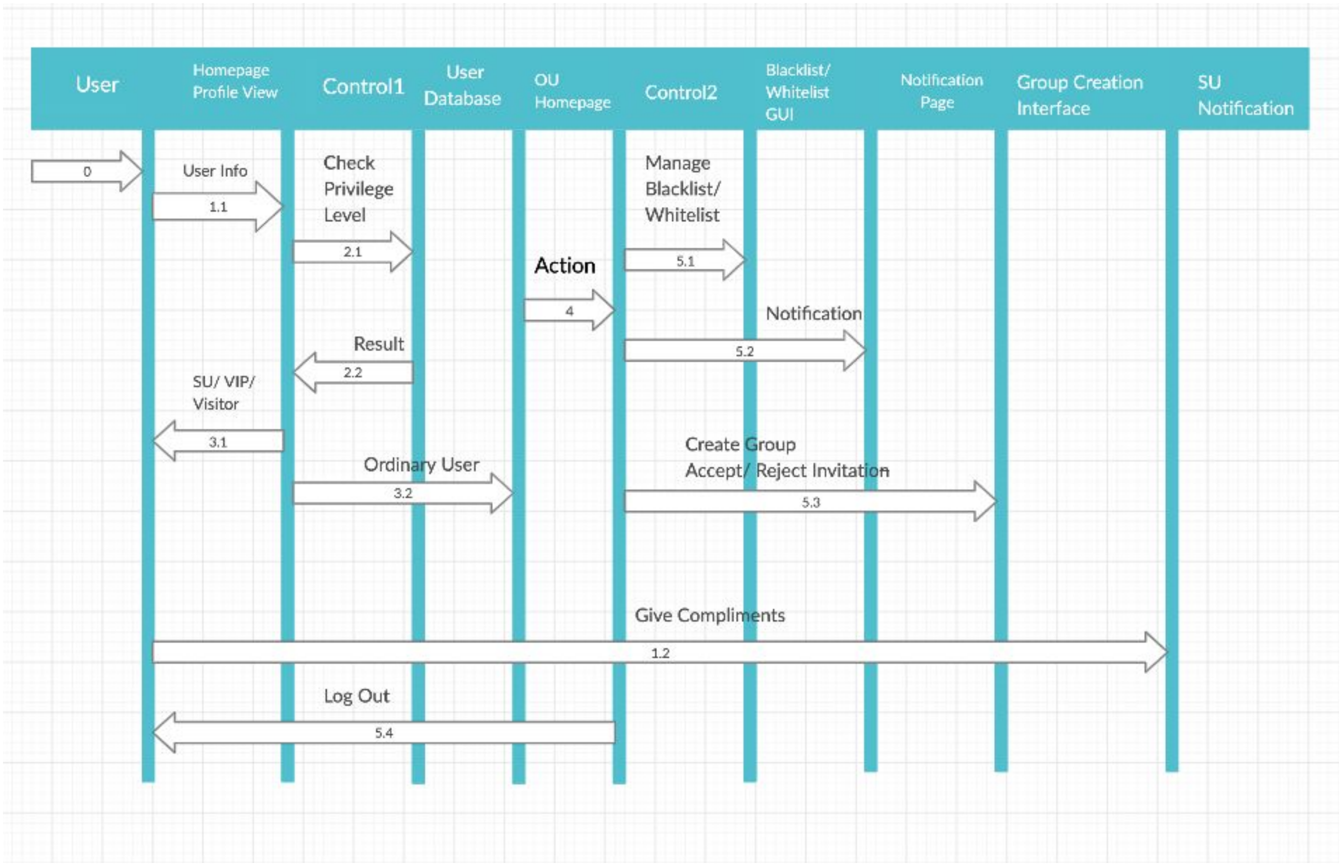


Figure 2.6: Homepage (OU) Logged In View Sequential Diagram

SU Homepage

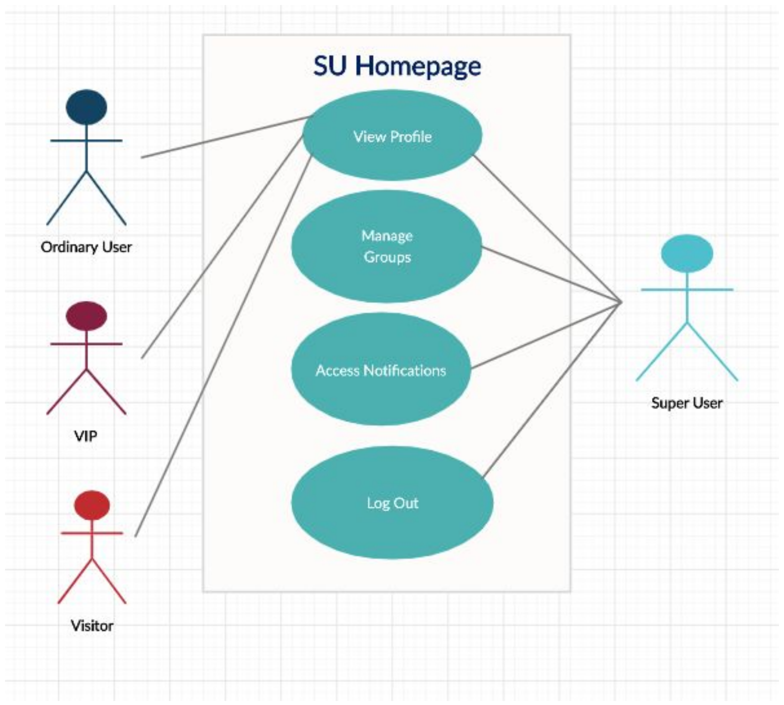


Figure 2.7: Homepage (SU) Logged In View Use Case

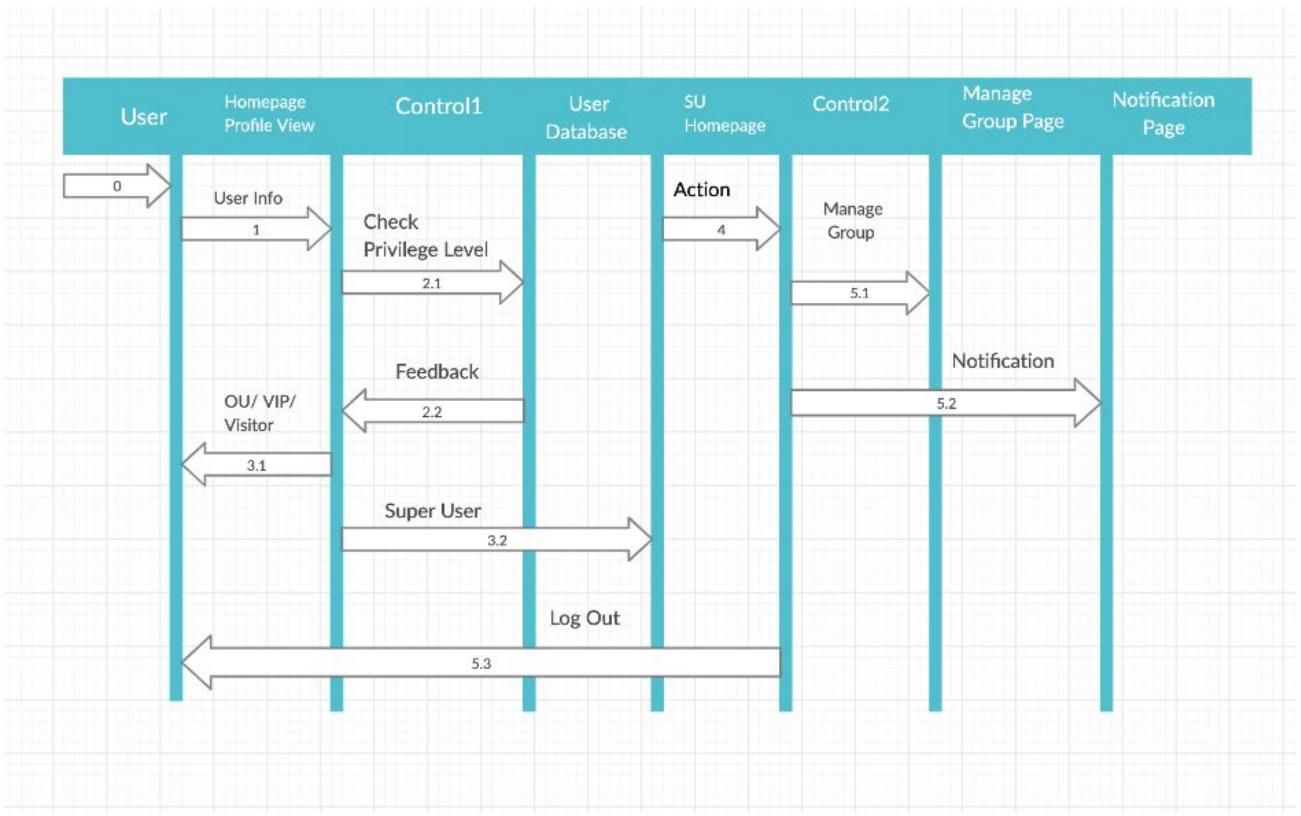


Figure 2.8: Homepage (SU) Logged In View Sequential Diagram

SU Notification Page

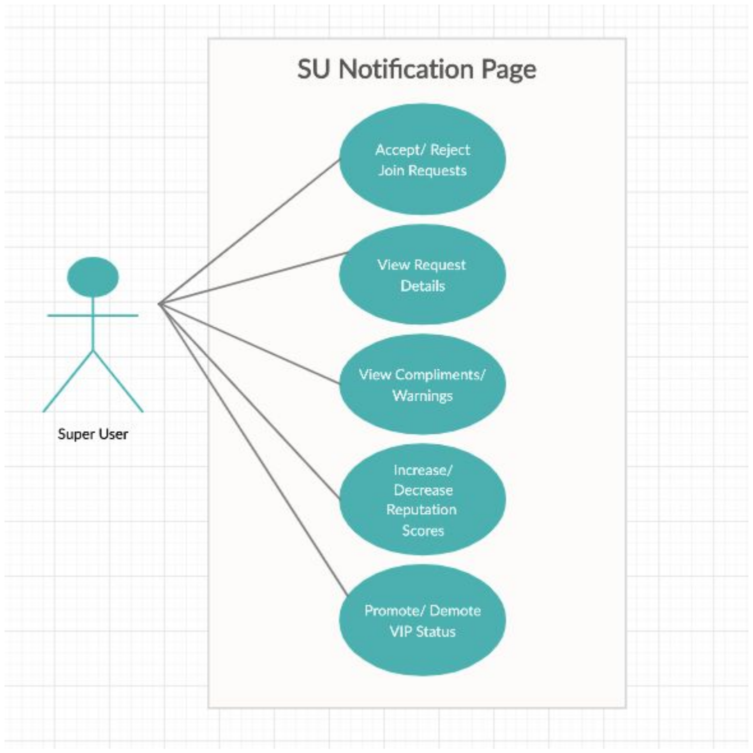

SU Notification Page
Super User
Accept/ Reject Join Requests
View Request Details
View Compliments/ Warnings
Increase/ Decrease Reputation Scores
Promote/ Demote VIP Status

Figure 2.9: Notification Page Use Case

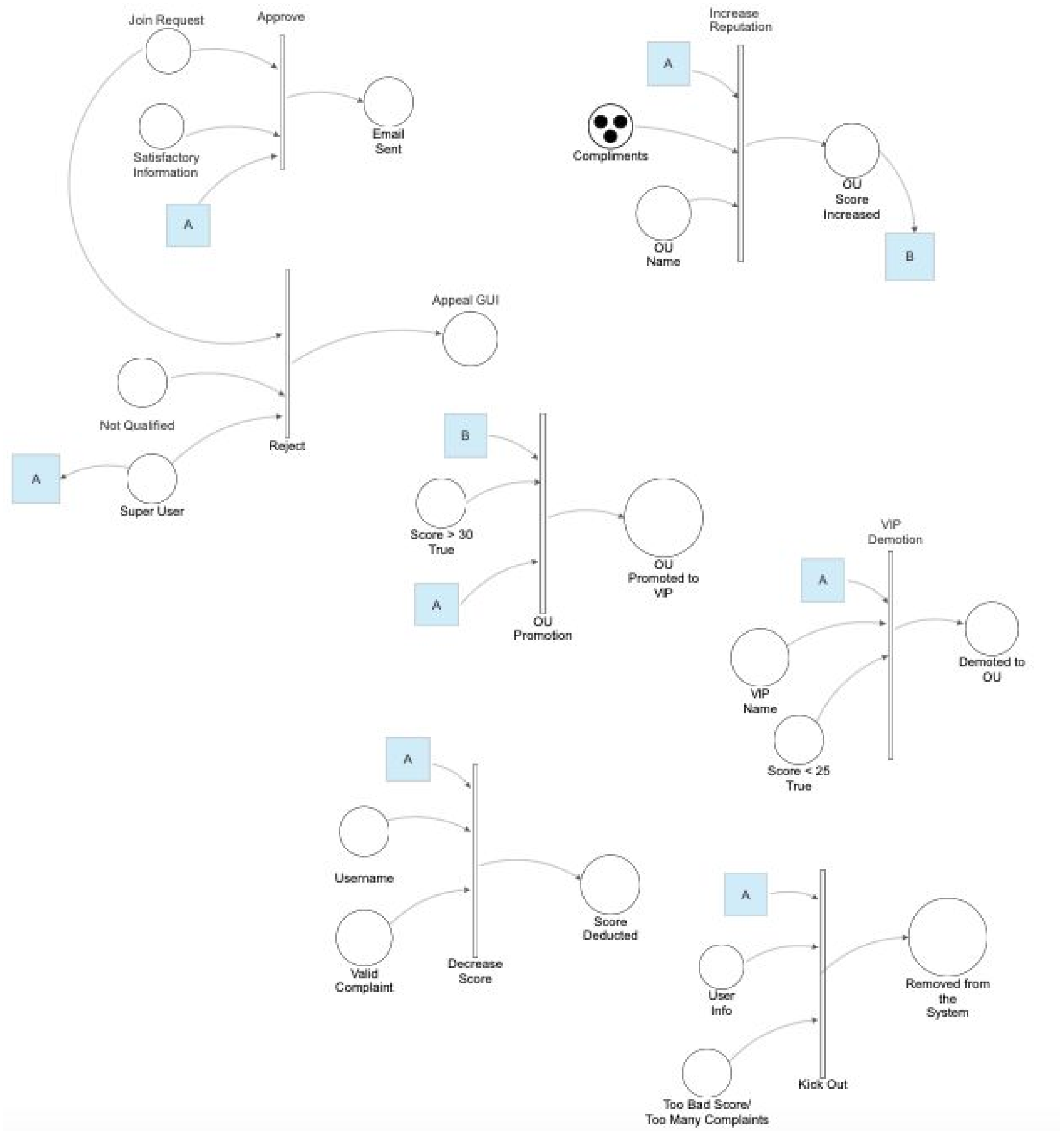


Figure 2.10: Notification Page Petri Net

Group Page

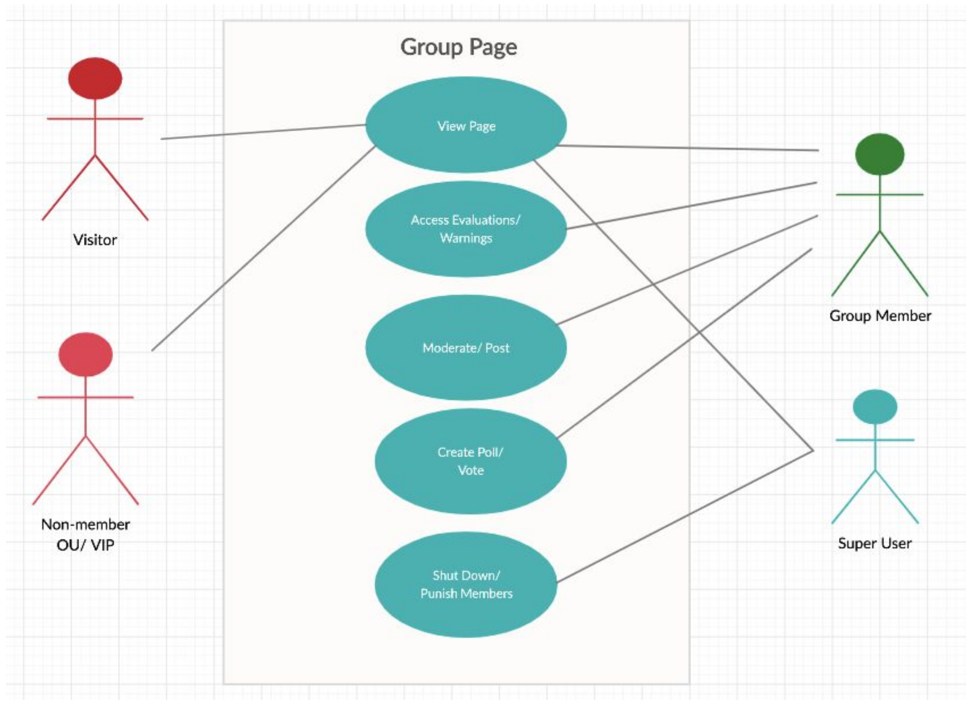


Figure 2.11: Group Page Use Case

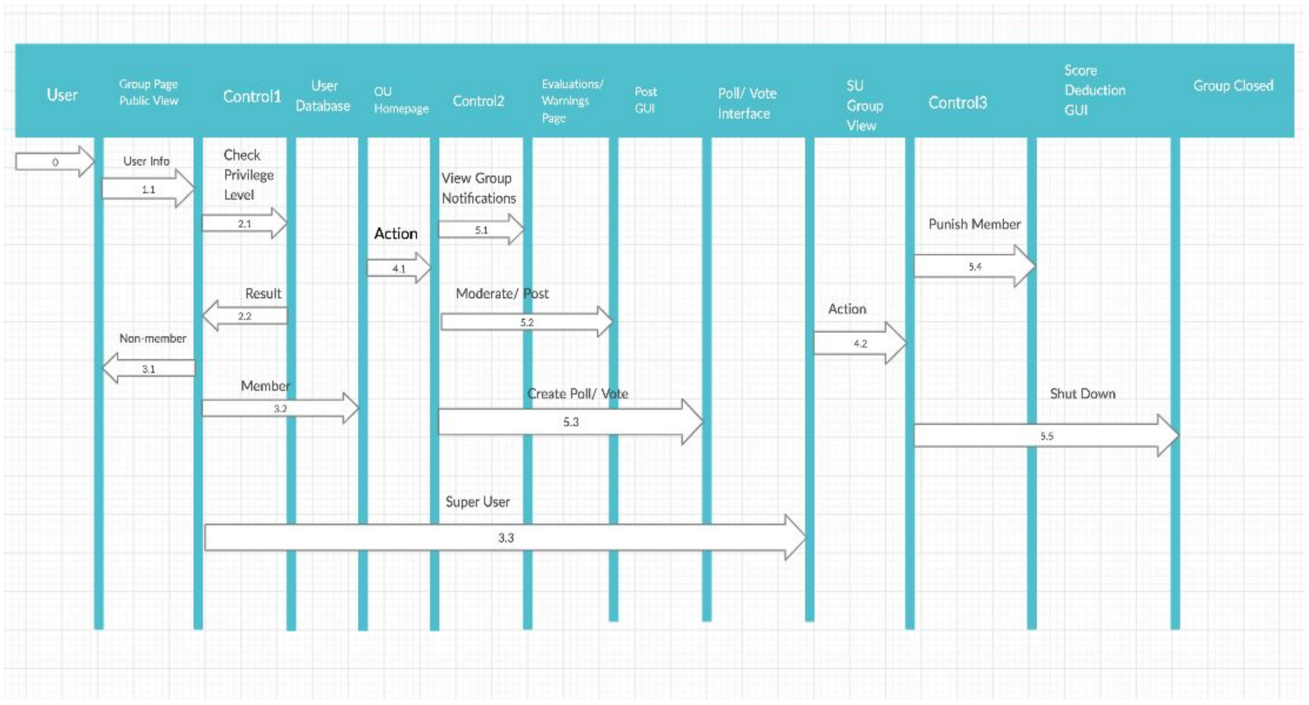


Figure 2.12: Group Page Sequential Diagram

3. E-R Diagram of System

The figure below shows the entity relationship diagram for this system. It provides all of the different relationships that all entities can possibly have. The angular arrows indicate a 0-or-1 to many relationship and the triangular arrows represent a 1 to many relationship.

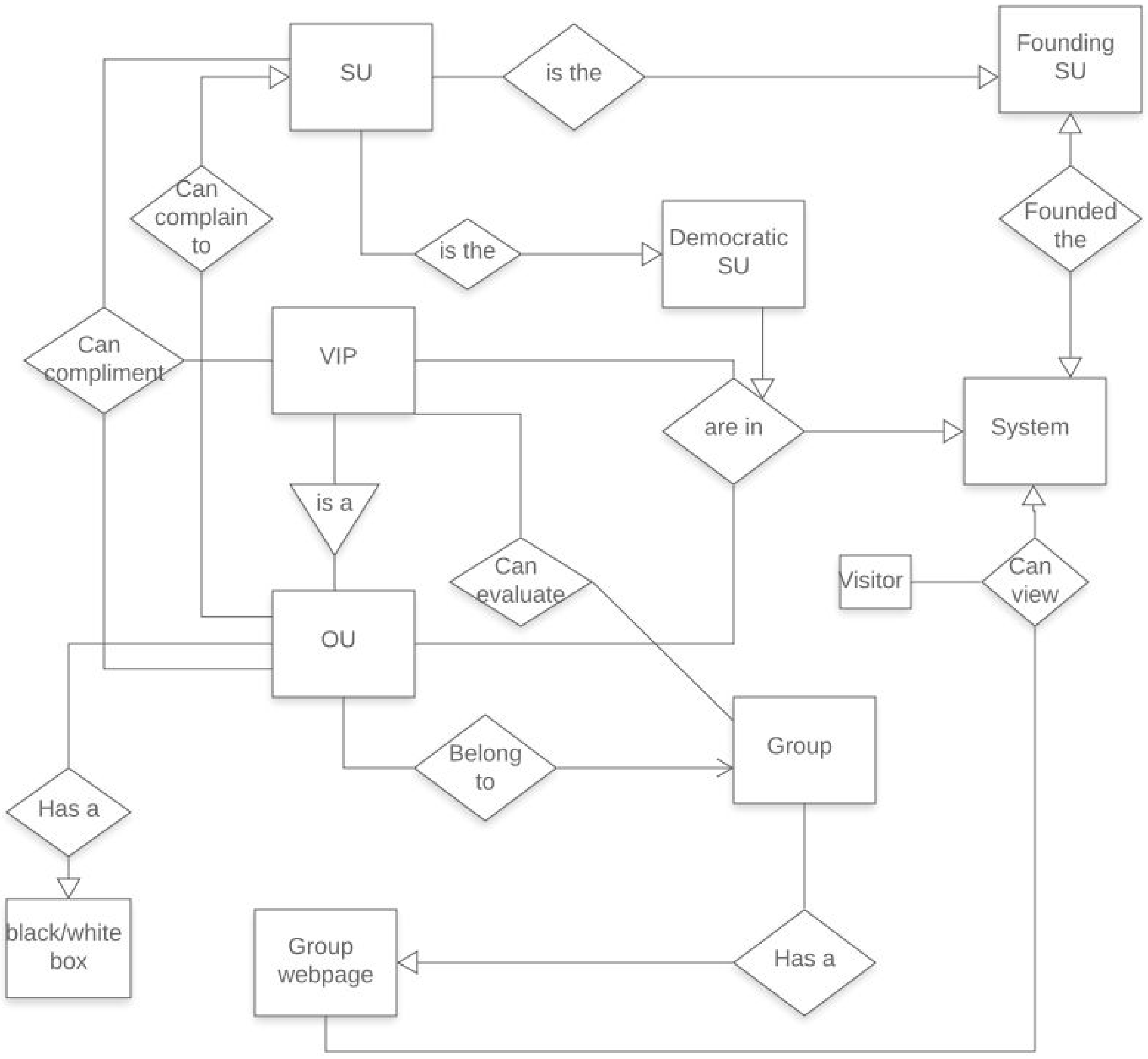


Figure 3.1: E-R Diagram of System

4. Detailed Design

In this section, we list every method and use pseudo-code to delineate the input/output and main functionalities of our system.

add_post(postType, postInfo, postDeadline, email):

- Add post to database and UI
- Check post deadline is at least 72 hours away

- Distribute warnings to users who did not vote for poll before postDeadline (Triggers addAutoWarning)

add_auto_warning(email, warningName, pointValue):

- Add warning to user's profile and database automatically (without SU approval)
- Triggers checkWarnings()

add_blacklist(email):

- Add user to blacklist

accept_request()

- if approved store the information in the database

change_points(email, pointValue):

- Deduct or add point value to user
- Triggers checkPoints()
- Triggers change_rank(email)

change_rank(email):

- Checks current user points
- If current OU:
    - Under 30: nothing
    - Over 30: change to VIP
- If current VIP
    - Over 26: nothing
    - Under 25: change to OU

check_users(email):

- Check user exists

check_warnings(email, groupId): Boolean

- Check number of warnings issued by group
- If number of warnings exceed 3, automatically kicks out user (Triggers removeGroupUser())

check_points(email): Boolean

- Looks at user's current points
- If current points are negative, adds user to blacklist (Triggers add_blacklist())

check_blacklist(email): Boolean

- Check if user is in the blacklist

check_taboo(email, text): Boolean

- If Taboo words, deduct points
- Triggers add_auto_warning()
- Triggers change_points()

delete_application()

- deletes all information from the system if the reject count is more than 1
- puts the user in the blacklist (Triggers add_blacklist())

error_message():

- Switch
- Displays message to user if any text input is null or no users were added

get_group_info()(groupInfo, users):

- Check all information is valid
- Adds group information submitted into database
- Sends invitation requests to all users listed

get_signup_info()

- Reads the user input from the text boxes
- Stores the information in the database
- invokes another function to notify the SU for approval

log_in():

- Check if email is in the system
- Check if the password entered matches password in the database
- If email and password match, go to OU,VIP or SU home page
- If email and password do not match, Show pop up displaying the error

profile():

- Switches page to the corresponding OU,VIP or SUs profile page. In this page they will be able to access their white box and black box as well as information that was displayed on the login screen(top users,top projects). For SUs, their profile page will allow them to view sign up requests they've received.

    -add_user_box(boxType,email)

    - Checks if email exists in the system
    - Checks that the user corresponding to the email is not in the blacklist (Triggers check_blacklist())
    - Check that the email is not in the opposite box. If it is, display an error.
    - Add user to the boxType specified

    -remove_user_box(boxType,email)

    - Checks if email exists in the corresponding box
    - Check that the email is not in the opposite box. If it is, display an error.
    - delete user from the boxType specified

remove_group_vote(removalType, groupId, email): True

- If removalType is to "removeUser", send request to all users of group to vote whether to remove user or not.
    - If vote is unanimous, kick out user. (Triggers removeGroupUser())
- If removalType is to "removeGroup", send request to all users of group to vote whether to close the group or not
    - If vote is majority, close group. (Triggers removeGroup())

remove_group_user(email, groupId):

- Remove user from group
- Gets point deduction (Triggers changePoints())

remove_group(groupId):

- Send all users of group requests to evaluate each group member
- Send request to SU to assign VIP to evaluate group

reject_application()

- if rejected for the first time store the name and email temporarily
- sends the user an email to appeal
- sets the reject count to be 1
- Triggers delete_application() if reject count is more than 1

show_users(groupId):

- Look up and display users of group

show_group_info(groupId):

- Look up and display group information, (such as description, group name, work history, etc) of group

show_group_posts(groupId):

- Look up and display group posts, current and previous.

show_users_group():

- When the user clicks the button, the page will change to a list of group pages that the user is currently in. Clicking on these groups will take them to the group page, where they can add posts and make polls.

show_details_request()

- shows the detailed information about the user requested to join

sign_up():

- When user presses sign up button, switch the page to the sign up screen where they can enter fields to sign up

show_compliments()

- displays the top 7 compliments about the Ous
- Checks if the user is a SU
- Invokes a function to increase the score if the SU wants
- Dismiss the notification if clicked on the dismiss button

show_warnings()

- Shows all warnings in the system
- Checks if the user is a SU
- Invokes a function to decrease the scores of the OUs if the SU wants
- Dismiss the notification

show_group_notifications()

- Shows close requests and assigned tasks
- Assigns a VIP to handle the requests if the user is an SU

task_assign(email, groupid, taskid):

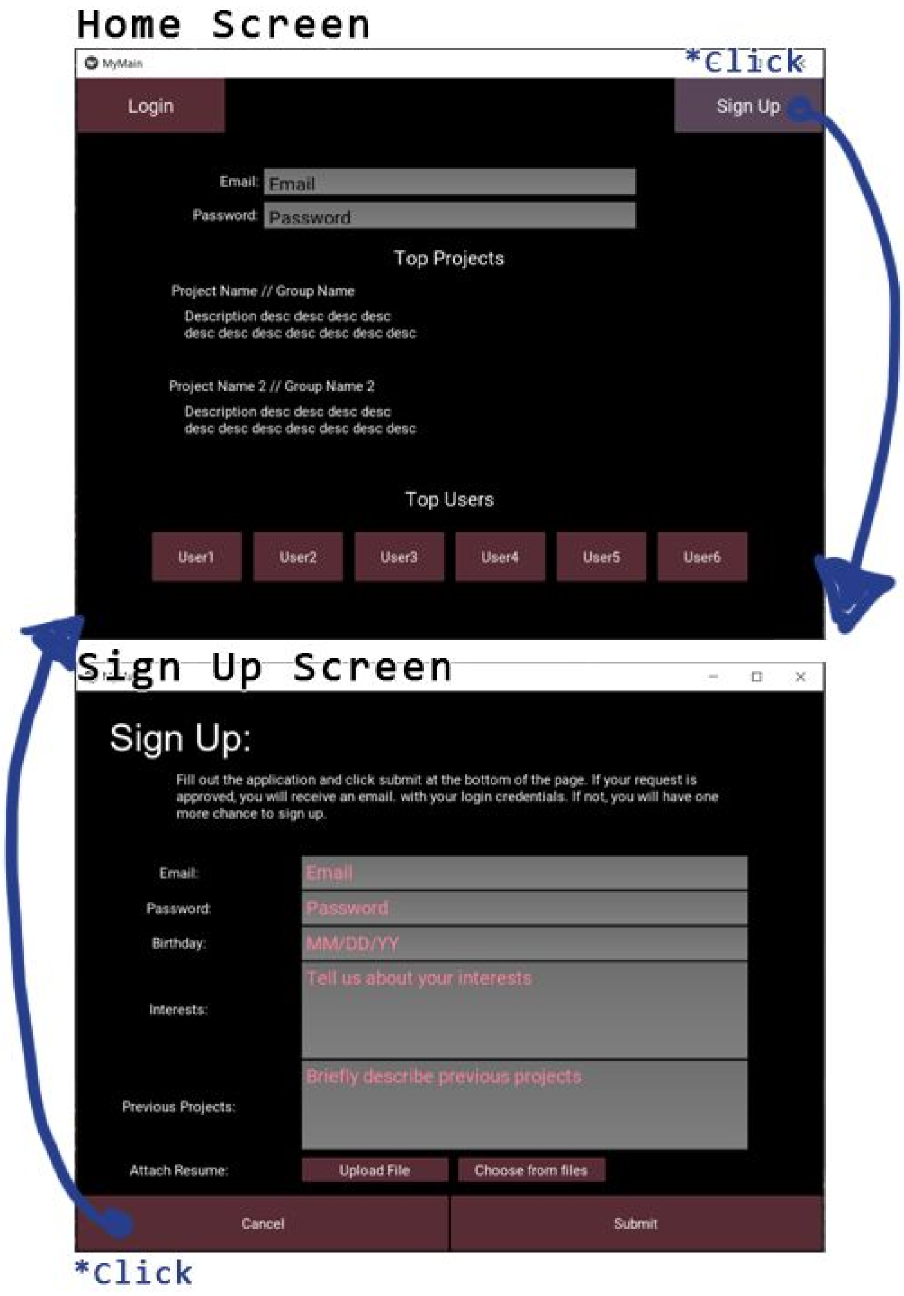
- Assigns task to user
- Task is sent to user's notification page
- Task pending count on group page is updated

task_complete(email, groupid, taskid):

- Task is removed from the user's notification page
- Task completed count on group page is updated

5. System Screens

In this section, we demonstrate major GUI screens of the system and a prototype of the Sign Up, Login, and Profile functionality in our system.



Project Name // Group Name

Description desc desc desc desc
desc desc desc desc desc desc desc

Project Name 2 // Group Name 2

Description desc desc desc desc
desc desc desc desc desc desc desc

Sign Up:

Fill out the application and click submit at the bottom of the page. If your request is approved, you will receive an email. with your login credentials. If not, you will have one more chance to sign up.

Email:	Email
Password:	Password
Birthday:	MM/DD/YY
Interests:	Tell us about your interests
Previous Projects:	Briefly describe previous projects

Figure 5.1: Home Screen (Visitor View) to Sign Up Screen

The Home Screen (Visitor View) shows the top projects and top users. It also shows a ‘Login’ button and a ‘Sign Up’ button. When the Sign Up button is clicked, it will bring the visitor to the Sign Up Screen. This screen has the application needed to be filled out by the visitor and two buttons on the bottom, ‘Cancel’ and ‘Submit’. If the form is filled out and the submit button is clicked, it will send the application to the SU and display a popup that says “Application Received.” However, if the cancel button is clicked, it will bring the visitor back to the Home Screen (Visitor View)

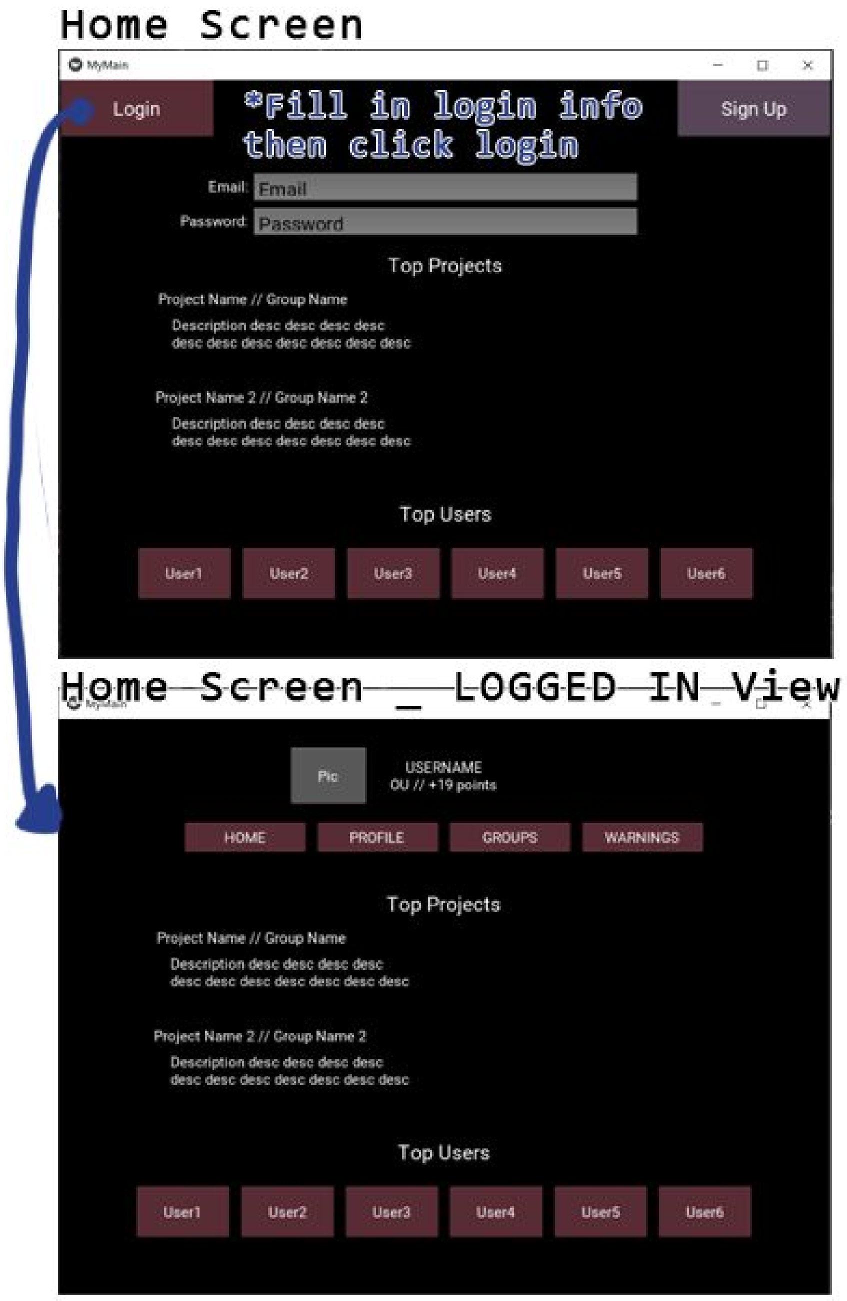
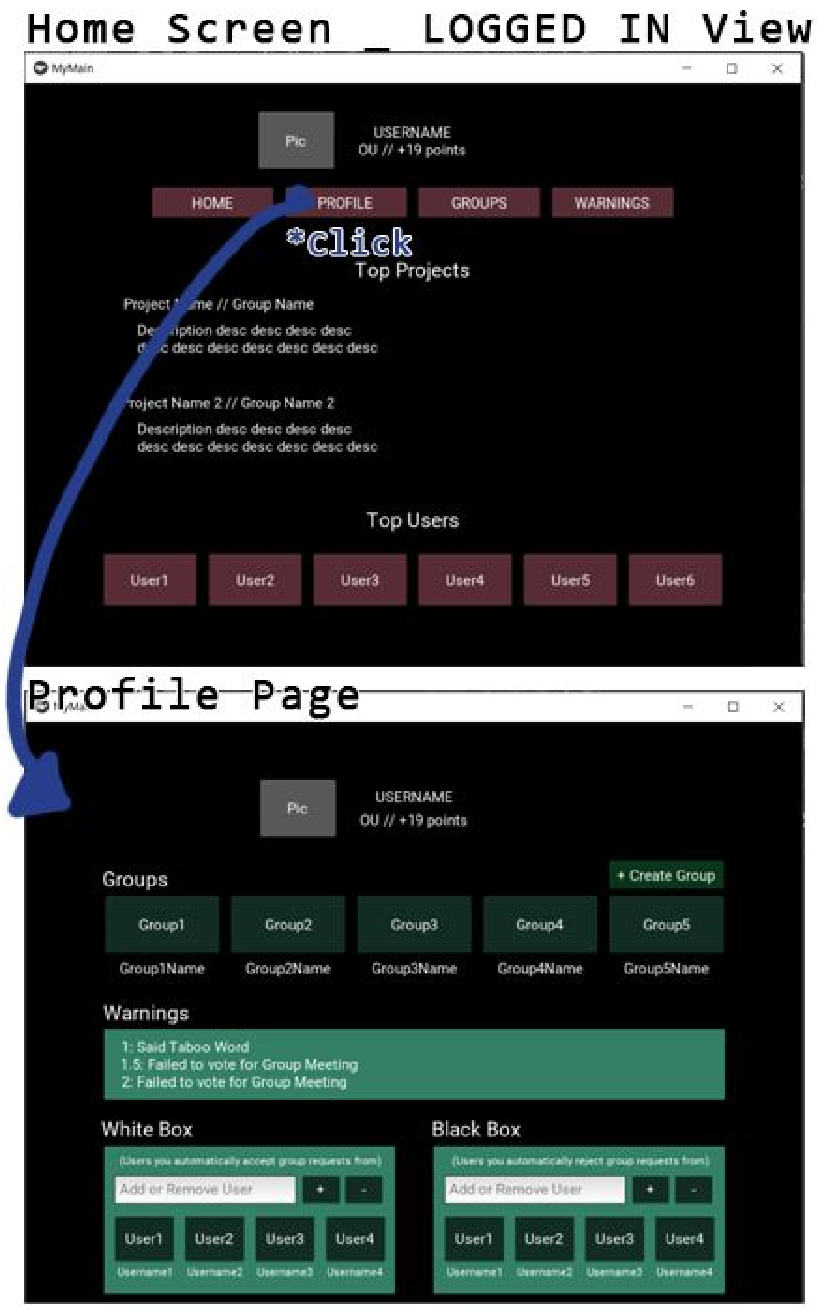

Home Screen
MyMain
Login
*Fill in login info then click login
Sign Up
Email:
Email
Password:
Password
Top Projects
Project Name // Group Name
Description desc desc desc desc desc desc desc desc desc desc desc
Project Name 2 // Group Name 2
Description desc desc desc desc desc desc desc desc desc desc desc
Top Users
User1
User2
User3
User4
User5
User6
Home Screen _ LOGGED IN View
MyMain
Pic
USERNAME
OU // +19 points
HOME
PROFILE
GROUPS
WARNINGS
Top Projects
Project Name // Group Name
Description desc desc desc desc desc desc desc desc desc desc desc
Project Name 2 // Group Name 2
Description desc desc desc desc desc desc desc desc desc desc desc
Top Users
User1
User2
User3
User4
User5
User6

Figure 5.2: Home Screen (Visitor View) to Home Screen (LOGGED IN View)

If the visitor already has an account, the visitor can log in by putting in their email and password. After, they must click Login. The database will verify the user. If the email or password is incorrect, a pop up will appear with the corresponding message. If the email or password is correct, it will bring the user to the OU View of the Home Screen which has the user information displayed on top along with links to refresh the page, the user profile, the user's groups, and the user's warnings.



Project Name // Group Name
Description desc desc desc desc
desc desc desc desc desc desc desc

Project Name 2 // Group Name 2
Description desc desc desc desc
desc desc desc desc desc desc desc

1: Said Taboo Word
1.5: Failed to vote for Group Meeting
2: Failed to vote for Group Meeting

(Users you automatically accept group requests from)

(Users you automatically reject group requests from)

Figure 5.3: Home Screen (LOGGED IN View) to User Profile Page

If the user clicks on the ‘Profile’ link above, it will bring the user to their own profile page. The profile page displays the user’s information on top such as email, rank, and points. It also shows the user’s current groups (and links to them) and an option to create a new group. It also displays the user’s current warnings, if there are any. And as well, the user’s white and black box. The white and black box display the current users already added in the box and have a text input for a email to either remove or add user to the respective boxes.

6. Group Meeting Minutes

This section is a log of group meetings, meeting duration, and meeting agendas from March 5, 2020 to April 24, 2020.

Meeting Date	Meeting Duration	Meeting Agenda
03/05/20	60 minutes	● Project specifications overview ● Brainstorm for which language/database to use ○ Narrowed down list to test over weekend ● Near-future goals in terms of project progress
03/11/20	90 minutes	● Report 1 discussion, splitting up parts ● Deciding on language and additional tools ● Low-level designing and discussing layouts for each and every screen needed for project
03/29/20	180 minutes	● Set up of GitHub, Python, Kivy, and Firebase ● Building the first screen (homepage) together
03/30/20	60 minutes	● Finishing first screen ● Splitting tasks for additional screens and integrating database
04/03/20	180 minutes	● Stand-Up Meeting: Sharing work done since last meeting, current problems ● Discussing future steps ● Splitting tasks for additional screens and building on code already written ● Coding switching screens in Python/ Kivy together
04/07/20	90 minutes	● Stand Up Meeting ● Integrating current screens ● GitHub pull requests, testing and debugging ● Splitting tasks for additional screens and started working on more functionality of the code
04/17/20	60 minutes	● Stand Up Meeting ● Report 2 discussion, splitting up parts ● Assigning additional tasks in terms of coding more screens and adding functionality to existing layouts
04/24/20	60 minutes	● Stand Up Meeting ● Report 2 discussion and review ● Integration of screens ● Additional discussions of project specifications

		● Splitting tasks to add more functionality to layouts
TOTAL: 8 meetings	**TOTAL MINs:** 780 minutes	

7. Git Repo

This section includes the URL to our group's git repo (github) of our work so far. All current code is on the github, including this report.

URL: https://github.com/eduardo-emeregildo/cs322_project