Team C

The Hive
Design Report For Online Project Management System

Version 1.0

Linda Wong
Niharika Alam
Alexandria Guo
Michael Mayaguari

Revision History

Date	Version	Description	Author
3/16/2020	1.0	Software Specification	Team C
4/29/2020	1.1	Design Report	Team C

Table of Contents

Design Report

1. Introduction

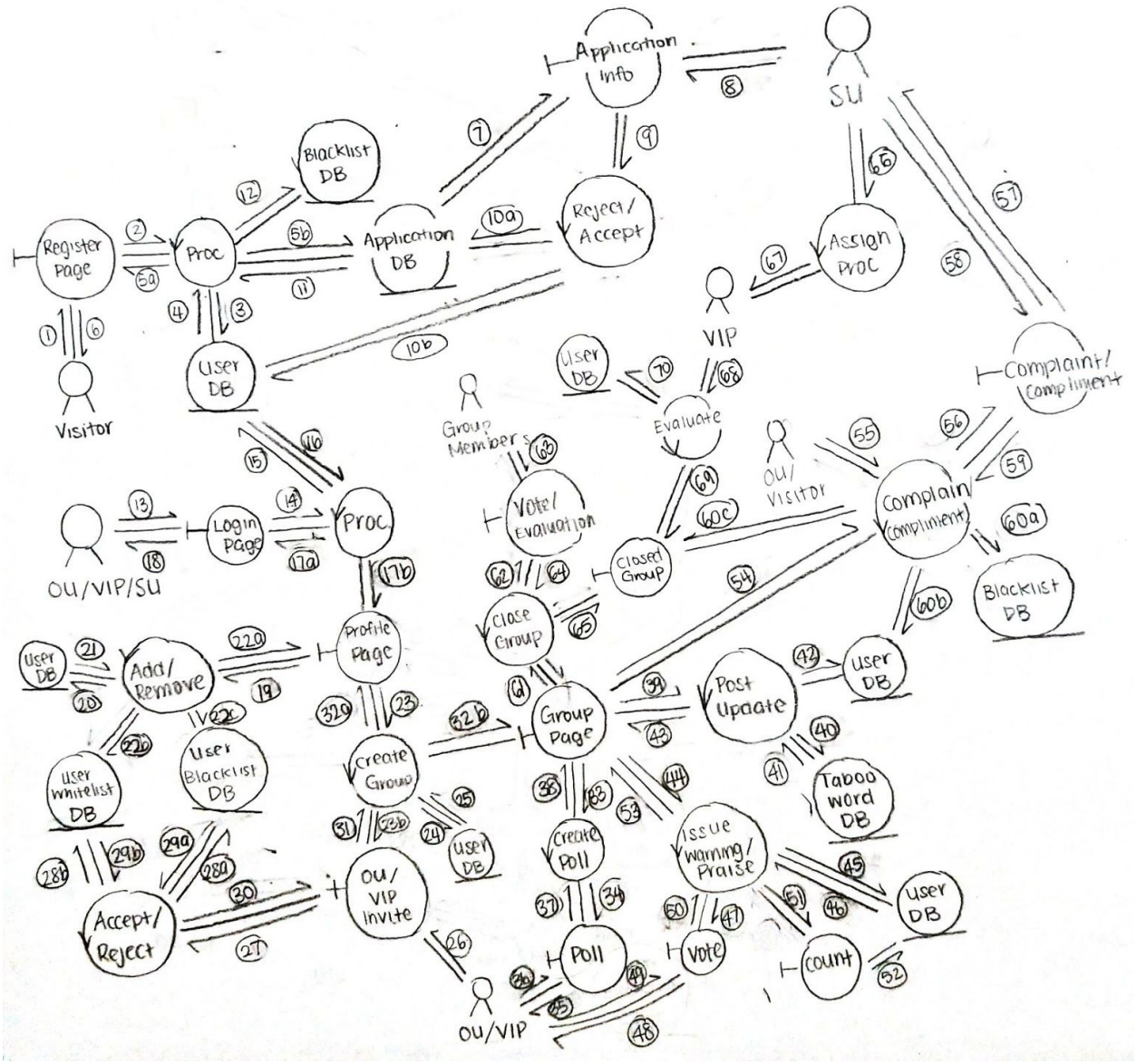
1.1 Purpose

This design report gives an overview of the design and functionality of the entire team management system - *The Hive.*

1.2 Collaboration Class Diagram

Collaboration diagrams are models that illustrate the relationships between the system's objects and how they interact with each other. The collaboration class diagram below outlines our entire team management system, outlining how various types of users interact with each other and the system, and the overall functionality of the system.

1. Enters information
2. Information
3. Check DB for existing
4. Response
5a. Not valid/ User exists
5b. Valid information
6. Try again prompt
7. Send information
8. Check information
9. Decision
10a. Reject application - can appeal
10b. Accept application - add to Users DB
11. Reject response
12. 2nd Rejection, add to blacklist DB
13. Enter username and password
14. Username and password
15. Check DB
16. Response
17a. Invalid login
17b. Valid login
18. Try again prompt
19. OU information
20. Check DB
21. Response
22a. Not valid user, try again
22b. Add to whitelist DB
22c. Add to blacklist DB
23. Group information
24. Check DB
25. Response
26. Receive invite
27. Decide
28a. Check blacklist DB
28b. Check whitelist DB
29a. Response - reject
29b. Response - accept
30. Decision
31. Responses from users
32a. Create group failed, try again
32b. Create group page
33. Enter poll information
34. Create poll
35. Send to members
36. Response
37. Responses from members
38. Meet-up time
39. Enter post information
40. Check taboo DB
41. Response, change to ***
42. Update user DB reputation score
43. Publish post
44. Enter warning or praise
45. Check DB
46. Response
47. Create vote
48. Send vote to members
49. Response
50. Member responses
51. Add to count
52. 3rd warning or praise, update DB respectively
53. Update page
54. Enter complaint or compliment information
55. OU/visitors enter information
56. Complaint/compliment information
57. SU receives information
58. SU reviews information
59. Decision
60a. Add OU to blacklist DB
60b. Update user DB reputation score
60c. Shut down group
61. Start to close group
62. Vote and Evaluation poll
63. Members vote and evaluate
64. Group Response
65. Close group
66. Assign VIP
67. VIP gets assignment
68. VIP begin evaluation
69. Evaluate group
70. Update user DB reputation scores


Application Info
SU
Blacklist DB
Reject/ Accept
Register Page
Proc
Application DB
Assign Proc
VIP
User DB
Visitor
Complaint/ Compliment
Evaluate
Group Members
Vote/ Evaluation
OU/ Visitor
Complain/ Compliment
Login Page
Proc
Closed Group
OU/VIP/SU
Blacklist DB
Profile Page
Close Group
Add/ Remove
Post Update
Group Page
User Whitelist DB
User Blacklist DB
Create Group
Taboo Word DB
Create Poll
Issue Warning/ Praise
Accept/ Reject
OU/ VIP Invite
Poll
Vote
Count
OU/VIP

2. All Use Cases

In this section, we provide a more detailed overview of each use case mentioned in the specification report. For each of the use case scenarios listed, we provide a collaboration class diagram to illustrate the specific interactions between the classes/objects in the system. We also provide petri-net class diagrams for three of the use cases to show the processes involved in the use case.

2.1 Use Case Scenarios

I. A OU/VIP/SU logins to the system

Normal Scenario:

A user is prompted to enter their username and password. Once they have successfully logged in, they will be directed to their profile/welcome page, and have access to their account.

Exceptional Scenario:

If a user enters a wrong username or password, they will be prompted to try again.

II. A visitor registers for the system

Normal Scenario:

A visitor is given an option to register for the system. They are prompted to enter their name, email, reference, interest, and credentials. The users' information is checked by a SU to either be approved or rejected. A user is allowed to appeal once if rejected.

Exceptional Scenario:

If the user left any information empty, they will be prompted to enter all the needed information. If the email that the user entered matches one already in the database, they are notified an account exists and prompted to try again.

III. A OU forms a group

Normal Scenario:

Registered users are greeted with their welcome page where they have the option to create a group. They can send invites to other registered users to collaborate in the same project.

Exceptional Scenario:

If no other OU accepts the invitation, the group cannot be formed and the OU is informed.

IV. A OU puts another OU in their white/black list

Normal Scenario:

Given that the OU is not on the intended OU's white/black list nor the intended OU already in either white/black list, the OU can access the option to put the intended OU on their white/black list.

Exceptional Scenario:

If the intended OU already exists in the black/white list respectively, they cannot be put into the other list.

V. A group member sets up a meet-up poll

Normal Scenario:

Group members will have the option to set up a meet-pull. They will be prompted to enter date, time, and topic.

Exceptional Scenario:

Group members cannot set up a meet-up poll with invalid fields such as date and times prior to the current day. The user will be prompted to re-enter valid information.

VI. A member posts an update

Normal Scenario:

Members are given the option to post updates. They will be presented with a post update preview draft. Once reviewed, they can submit the update.

Exceptional Scenario:

If the post violates any guidelines such as the use of taboo words, the post will be published with the taboo words converted to *** . The user will be given a warning and a reputation score deduction.

VII. A group votes to kick out a member

Normal Scenario:

Group members will be given the option to vote anonymously to kick a member of the group. If the votes reach a high enough threshold, then the request to remove the member will be honored.

Exceptional Scenario:

In the case where the number of votes does not reach the threshold, the vote to remove the member will not be honored.

VIII. A group votes to close the group

Normal Scenario:

Group members will be given the option to vote anonymously to close the group. If the votes reach a high enough threshold, then the request to close the group will be honored.

Exceptional Scenario:

In the case where the number of votes does not reach the threshold, the vote to close the group will not be honored.

IX. The SU assigns a VIP to evaluate the group

Normal Scenario:

The assigned VIP will evaluate the group based on a defined criteria. Given those criterias, the VIP will rate the group on a scale and assign a grade to that group. If the group violates any guidelines, the VIP can submit a request to shut down the group.

Exceptional Scenario:

In the case that the SU determines that the VIP's evaluation is not an accurate representation of the group, the SU can override the VIP's evaluation.

X. A OU compliments or complains about a group or member

Normal Scenario:

Each OU can have the option to submit a compliment or complaint on the intended group or member.

Exceptional Scenario:

If the OU is already on the blacklist then they do not have the option to compliment or complain about another OU. If an OU receives 3 unique compliments, their reputation score increases.

XI. The SU shuts down a group or OU

Normal Scenario:

The SU will have the option to shut down a group or OU based on a complaint or report.

Exceptional Scenario:

The group or OU can submit an appeal to SU regarding their status.

XII. A OU get promoted to VIP

Normal Scenario:

When an OU's reputation score surpasses 30, they will be promoted to VIP and alerted. VIP's are granted additional features.

Exceptional Scenario:

If the OU violated any guidelines or their reputation score falls below the threshold for VIP, their promotion to VIP will be revoked.

XIII. A VIP gets demoted to OU

Normal Scenario:

A VIP will be demoted if their reputation score no longer meets the threshold and will be alerted of their demotion.

Exceptional Scenario:

VIP's can retain status if their reputation score is above the threshold.

XIV. A VIP votes for a democratic SU

Normal Scenario:

VIP's will have the option to submit a vote for a democratic SU once. The VIP with the highest number of votes will be promoted to SU.

Exceptional Scenario:

If the VIP violates any of the guidelines, the SU status will be revoked and they will be given a warning. If the VIP's reputation score falls below 30, their status is revoked.

XV. A OU is removed from the system - placed on blacklist

Normal Scenario:

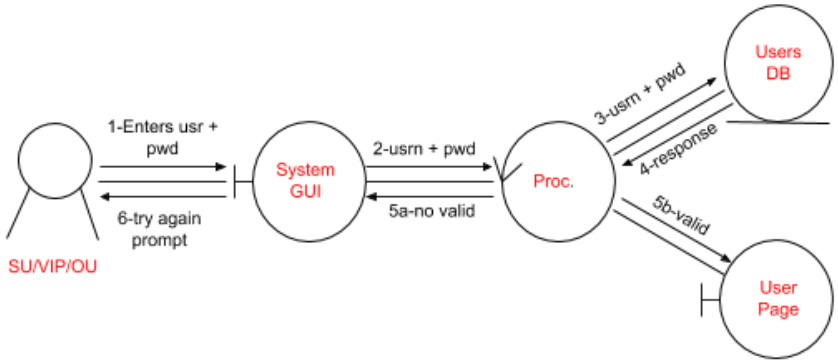
An OU will be removed from the system if they have a negative reputation score, the user is automatically removed from the system.

Exceptional Scenario:

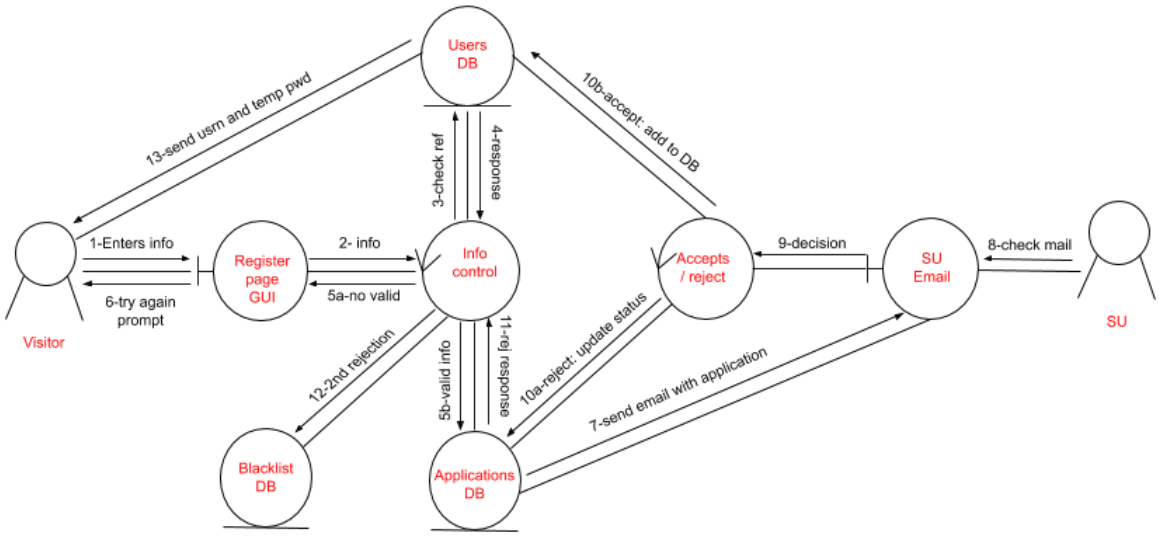
The OU can appeal to the SU for inquiries regarding the blacklist.

2.2 Collaboration/Sequence Class Diagrams

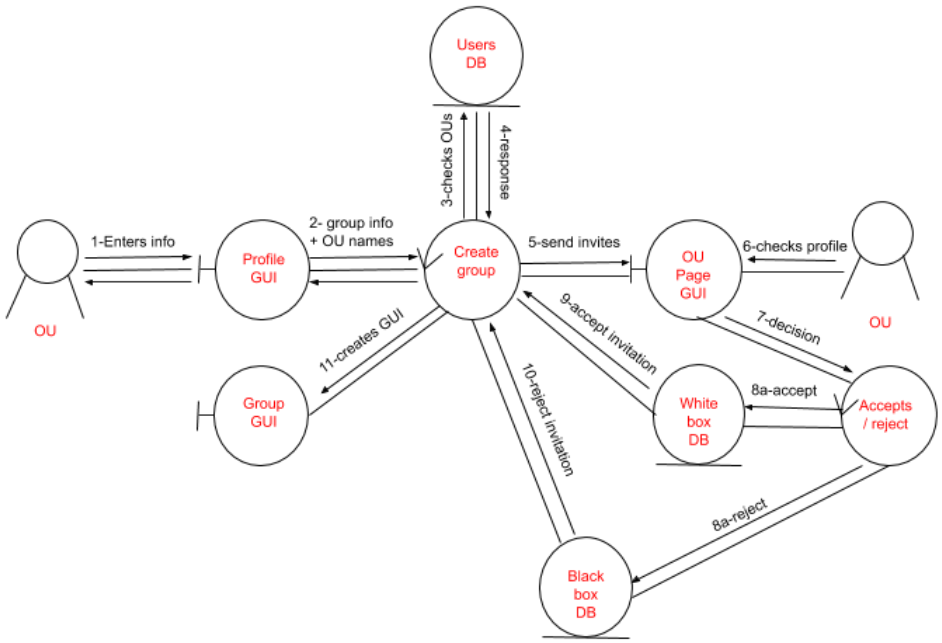
I. A OU/VIP/SU logins to the system



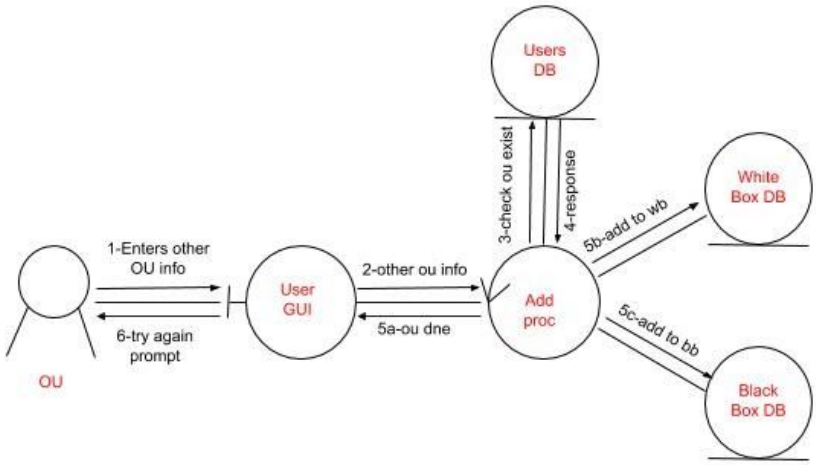
II. A visitor registers for the system



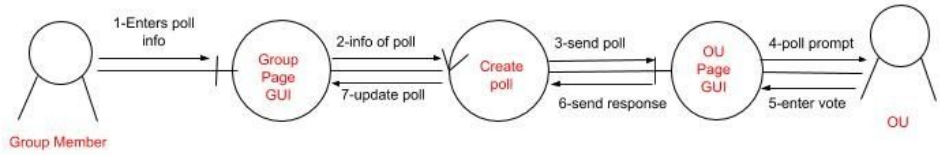
III. A OU forms a group



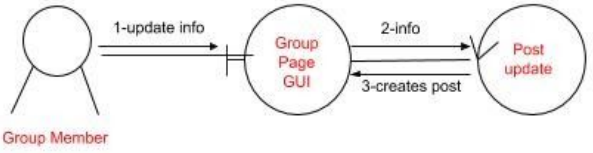
IV. A OU puts another OU in their white/black list



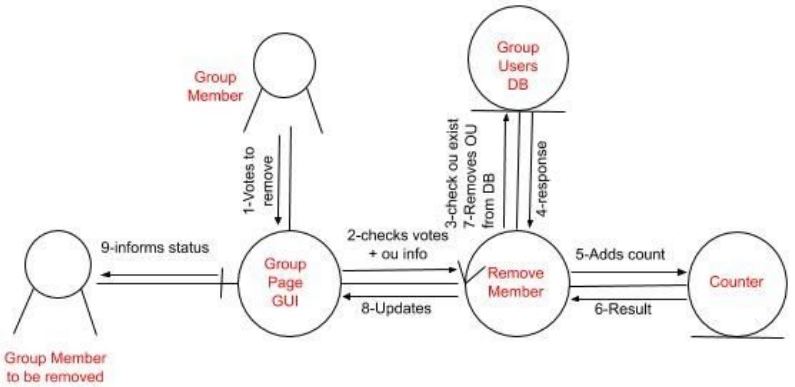
V. A group member sets up a meet-up poll



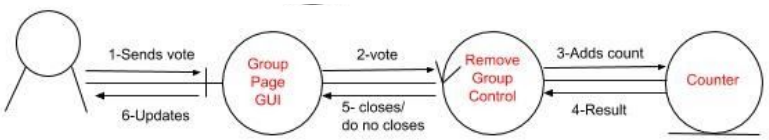
VI. A member posts an update



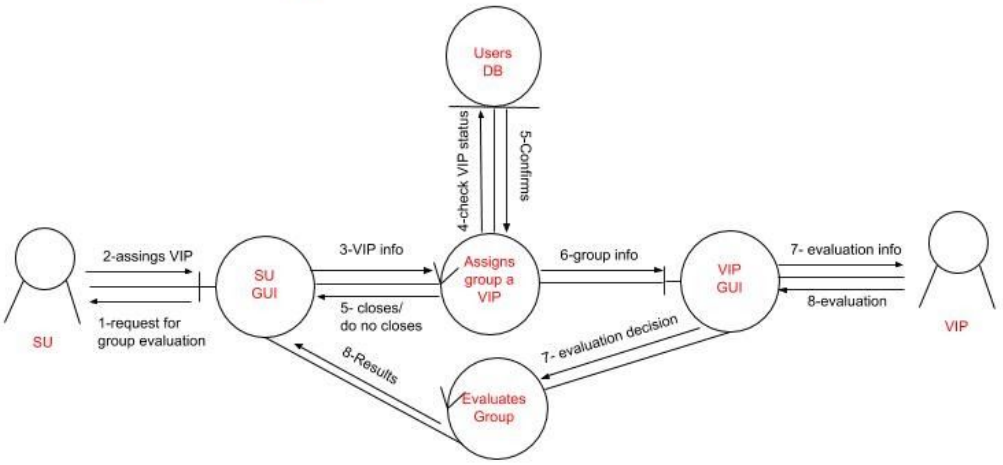
VII. A group votes to kick out a member



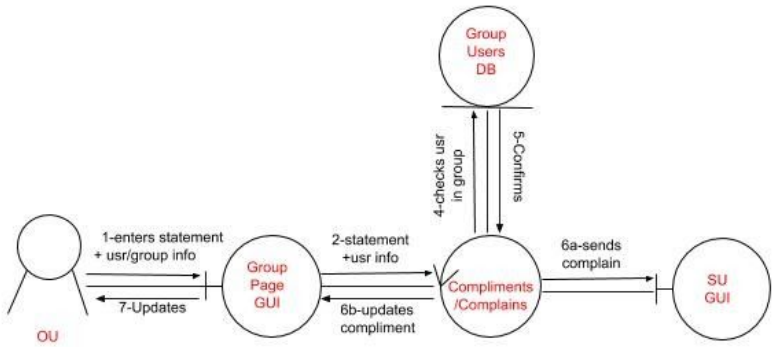
VIII. A group votes to close the group



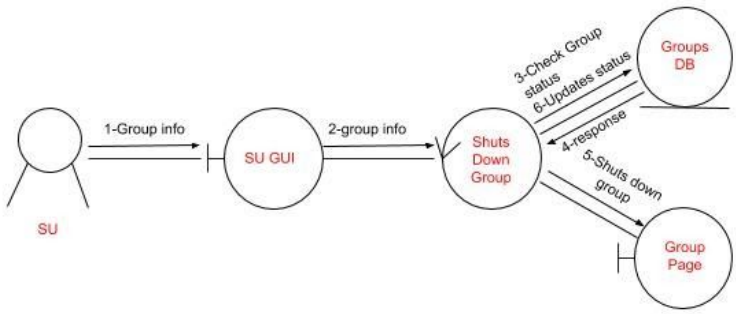
IX. The SU assigns a VIP to evaluate the group



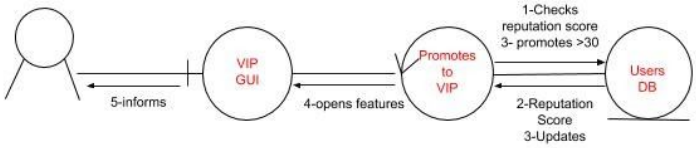
X. A OU compliments or complains about a group or member



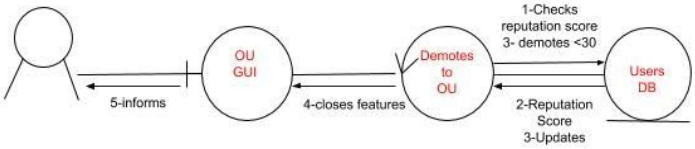
XI. The SU shuts down a group or OU



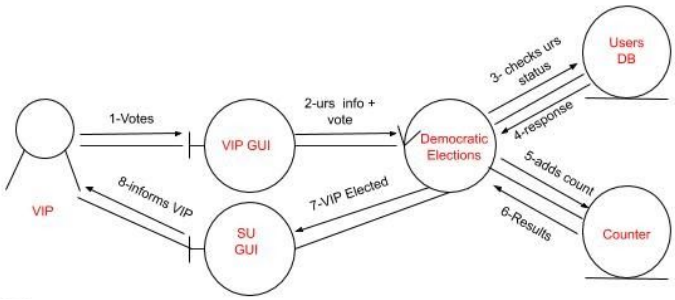
XII. A OU get promoted to VIP



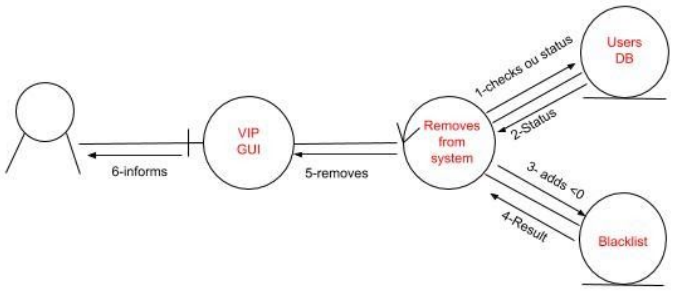
XIII. A VIP gets demoted to OU



XIV. A VIP votes for a democratic SU

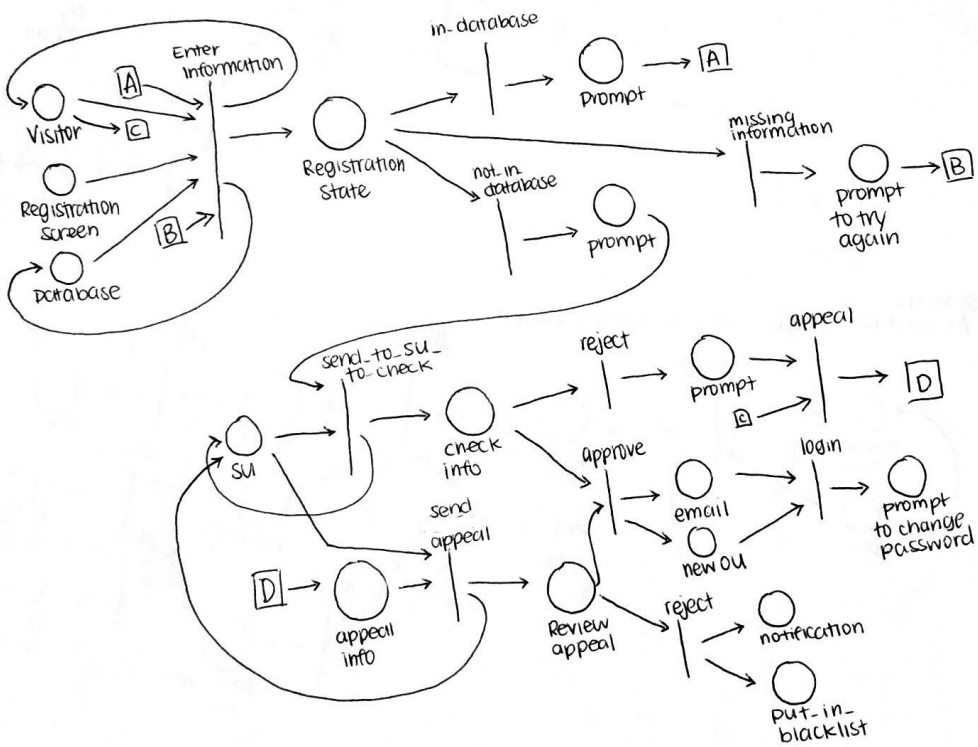


XV. A OU is removed from the system - blacklist

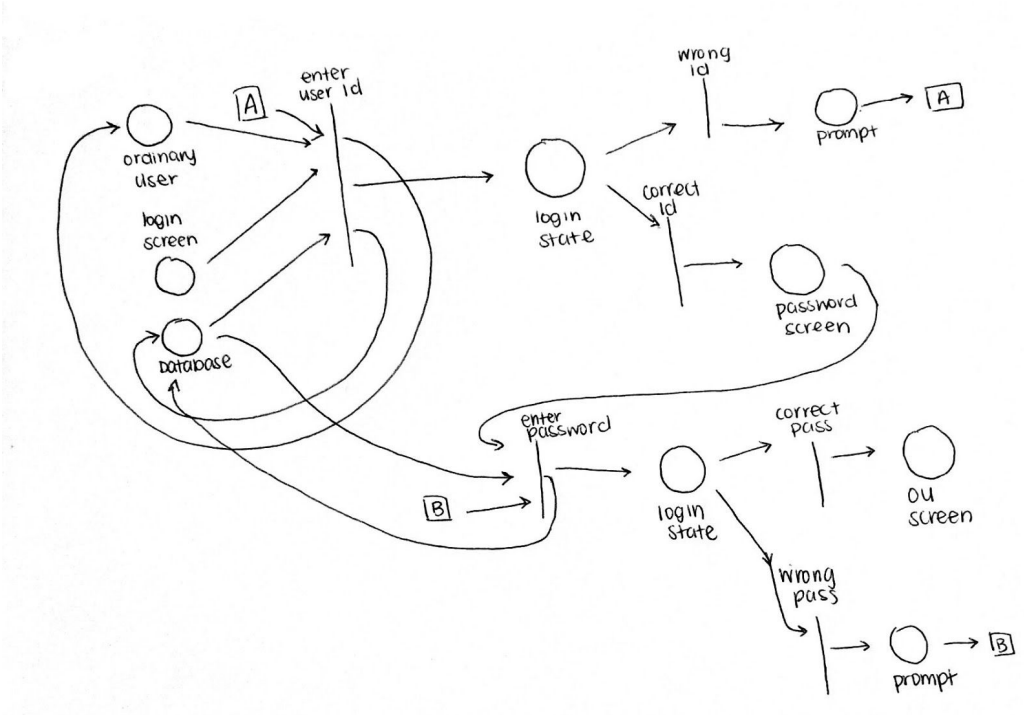


2.3 Petri-Net Class Diagrams

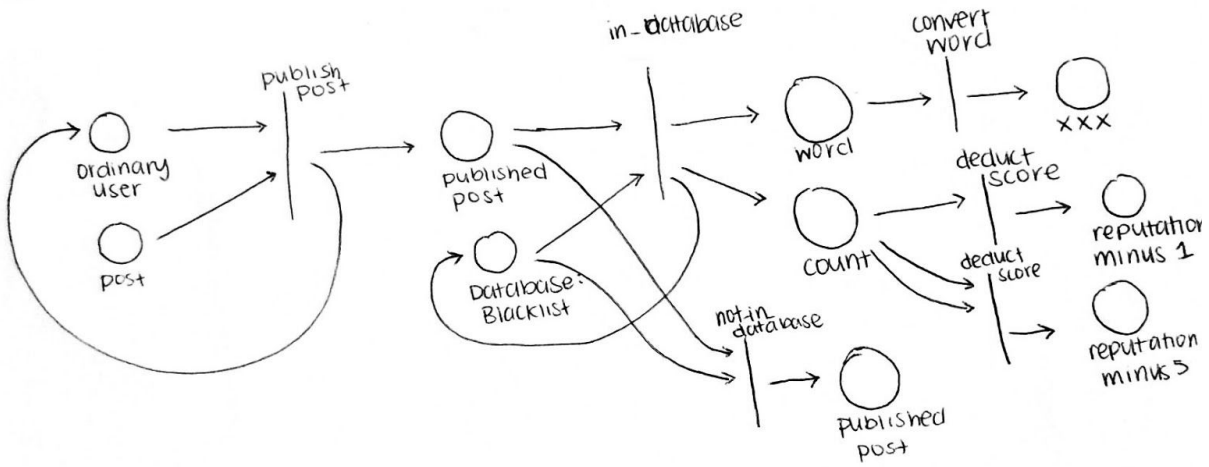
I. A OU/VIP/SU logins to the system



II. A visitor registers for the system

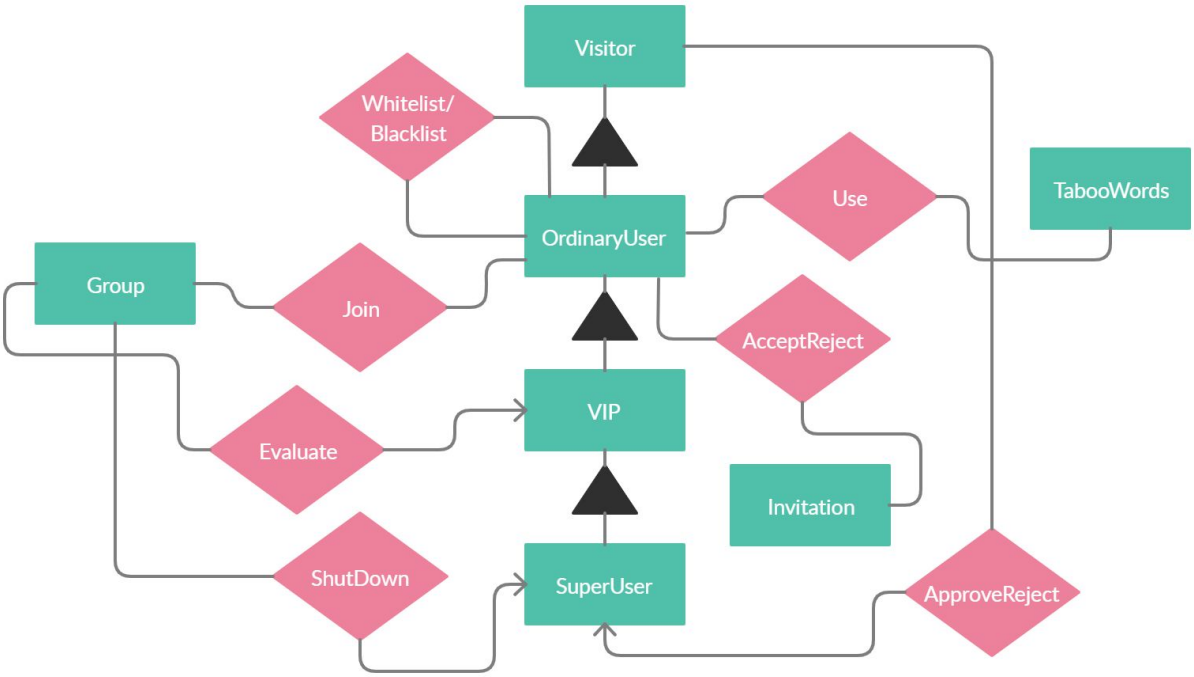


VI. A member posts an update



3. E-R diagram for the entire system

An entity relationship (E-R) diagram illustrates the relationships of entity sets stored in a database. Below is an E-R diagram describing the overall team management system.



1. Visitor is the parent class. OrdinaryUser, VIP, and SuperUser are the children classes.
2. A SuperUser can approve/reject many visitors and a visitor is approved/rejected by 0 or 1 SuperUser
3. An OrdinaryUser can blacklist/whitelist many OrdinaryUsers.
4. An OrdinaryUser can use many TabooWords and a TabooWord is used by many OrdinaryUsers.
5. An OrdinaryUser can accept/reject many group invitations and an invitation is accepted/rejected by many OrdinaryUsers.
6. An OrdinaryUser can join many groups and a group is joined by many OrdinaryUsers.
7. A VIP evaluates many groups if assigned by a SuperUser and a group is evaluated by 0 or 1 VIP.
8. A SuperUser can shut down many groups and a group is shut down by 0 or 1 SuperUser.

4. Detailed design

4.1. Register

Receive: User Information - Name; Email; Interest; Credidental; Reference

Return: Success (send to SU) or failure

```
def reg_btn(self):
    name = self.name.get()
    email = self.email.get()
    reference = self.reference.get()
    interest = self.interest.get()
    credential = self.credential.get()

    cursor.execute("SELECT * FROM users WHERE email = %s", (email,))
    account = cursor.fetchone()

    if name == "" or email == "" or reference == "" or interest == "" or credential == "":
        messagebox.showwarning("Registration Status", "All fields are required!")
    elif not account:
        messagebox.showinfo("Registration Status", "Thank You! A SuperUser will review your application "
                                                   "and if approved, an email will be sent to you with "
                                                   "your login details.")
        send_email(name, email, reference, interest, credential)
        self.welcome()
    else:
        messagebox.showerror("Registration Status", "Account with this email already exists!")
```

4.2. Login

Receive: User credentials

Return: Found or not found credentials in database

```
def log_btn(self):
    username = self.username.get()
    password = self.password.get()

    cursor.execute('SELECT * FROM users WHERE username = %s AND password = %s', (username, password))
    account = cursor.fetchone()

    if account:
        self.su()
    elif username == "" or password == "":
        messagebox.showwarning("Login Status", "All fields are required!")
    else:
        messagebox.showerror("Login Status", "Account does not exist!")
```

4.3. Create Group

Receive: group name, group creator, other invites,

Return: group status, send invitations

PseudoCode:

```
create_group(name, creator, list, type):
    For u in list:
        Send invitation to u
    new _list: response from list
Create group GUI(type,new_list)
Create group GUI(type,list):
    Set group as: type
    make available to: list
```

4.4. Upvote/Downvote User Reputation Score

Receive: Selection to upvote or downvote

Return: Update user reputation score

PseudoCode:

```
usr_rep_score(user, add_or_remove, points):
    cur_score = user.reputation_score
    if (add_or_remove == 'add'):
        cur_score = cur_score + points
    else:
        cur_score = cur_score - points
    update_score_in_db(user, cur_score)
```

4.5. Scheduling meetings

Receive: Date/Time, Invite, Users

Return: Invite Sent

PseudoCode:

```
create_meeting:
    set_date
    set_time
    set_topic
send_invite:
    for user in list:
        send_invite_to_users
```

4.6. Post updates

Receive: Post

Return: Success or Failure

PseudoCode:

```
For everyword_in_post:
    check_taboo_word_blacklist
    if word_found_in_BD:
        change_word_to_***
        taboo_word_use++
        Update database (taboo_word_use)
If taboo_word_use == 1:
    User_rep--
Update database (user_rep)
If taboo_word_user >1:
    User_rep -= 5
    Update database (user_rep)
Post_update
```

4.7. Add/Remove User from Blacklist/Whitelist

Receive: User to be removed/added to blacklist/whitelist

Return: User is removed/added

PseudoCode:

```
if user is_in blacklist/whitelistDB:
    Return "User already in DB"
else:
    Add/remove user from DB
```

4.8. Accept/Reject Invitation

Receive: Invite - project, admin_user

Return: Accept or Reject

PseudoCode:

```
if user_in_whitelist:
    Accept_invite
else if user_in_blacklist:
    Reject_invite
else:
    Accept/Reject_Invite
```

4.9. Report Group/User to Super User

Receive: User id/name or Group id/name

Return: User/Group report sent to SU

PseudoCode:

```
report_grp:
    if report_grp:
        report_sent_to_su
    else:
        no_action_taken
report_user:
    if report_user:
        report_sent_to_su
    else:
        no_action_taken
```

5. System Screens

Below are the initial GUI screens of the system which users can access. Visitors will be able to register for an account and users who have registered can login to access the main user interface

5.1: Register Function

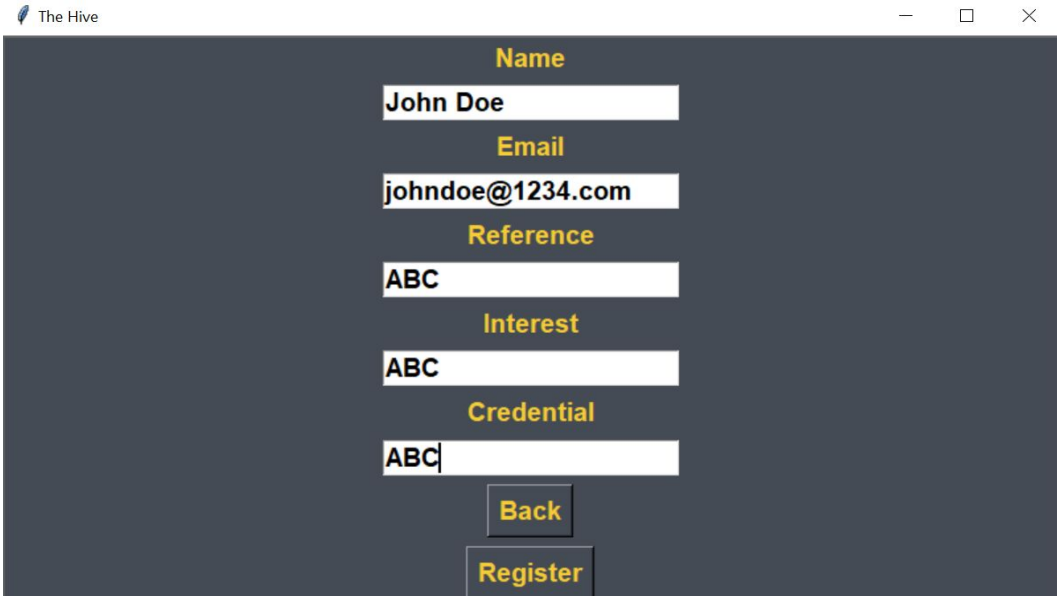


Figure 1: Users will be prompted to enter credentials to register for an account

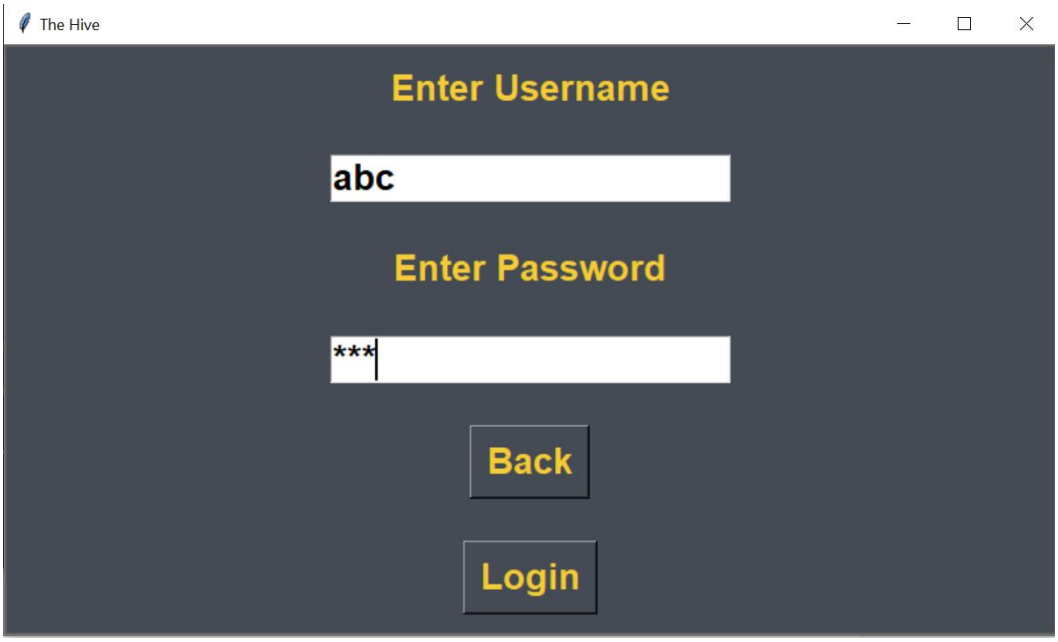


Figure 2: Users can log in after registering for an account

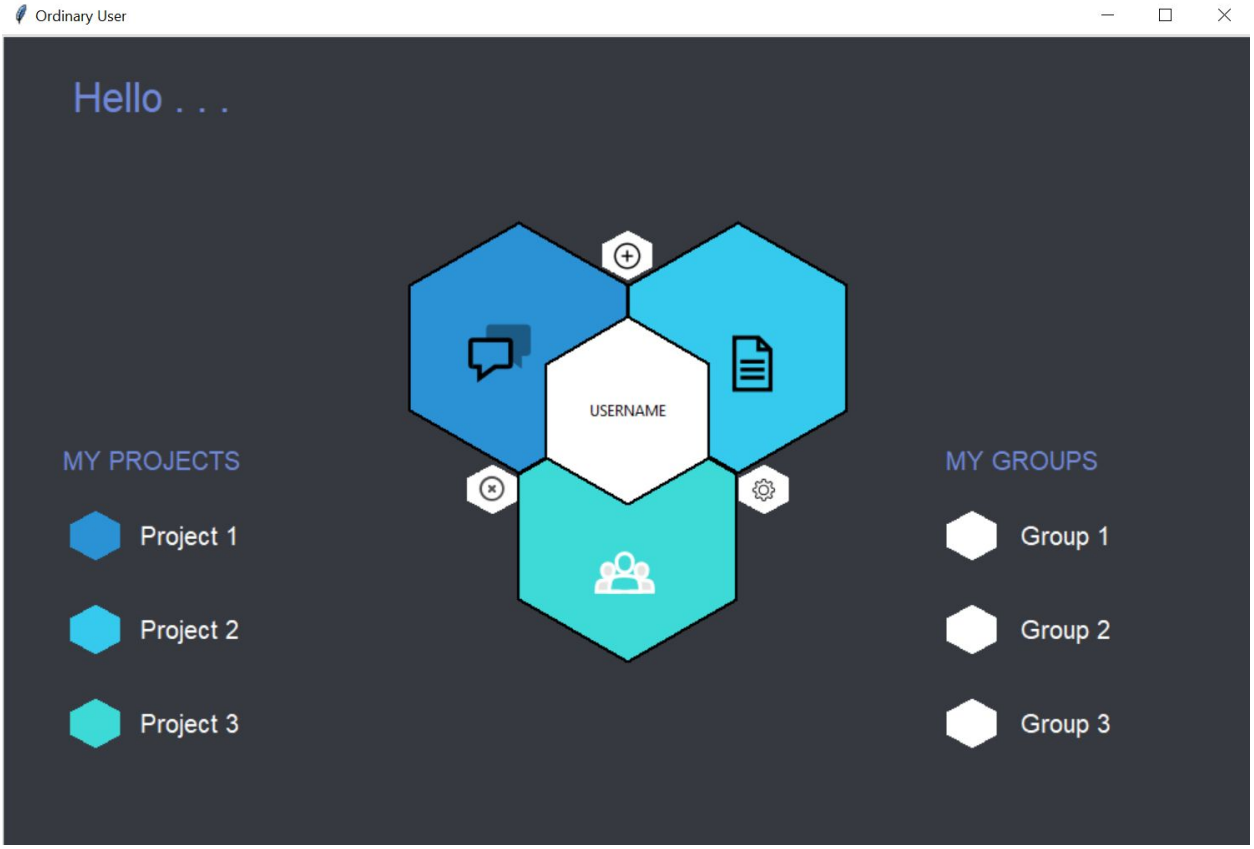


Figure 3: After registering for an account, users can log in and access the user interface with select options

6. Time

We have allocated weekly zoom meetings to discuss our progress and next steps.

Meeting #	Length	Discussion
1	1 hour	Phase I: Specification Report
2	1 hour	Laying out functions/capabilities of each user
3	1 hour	Implementing login/registration/email system
4	1 hour	Phase II: Design Report

7. Github

Find our project at: [mgmayagu/The-Hive: Software Engineering Project - Team C]