B-tech ICT Second Year Fourth Semester

Course Name : Database Management System

Group Number 28.

Group Members :

Nancy Radadia (AU1841070)

Suhanee Patel (AU1841113)

CHAT APPLICATION

1. Project Description

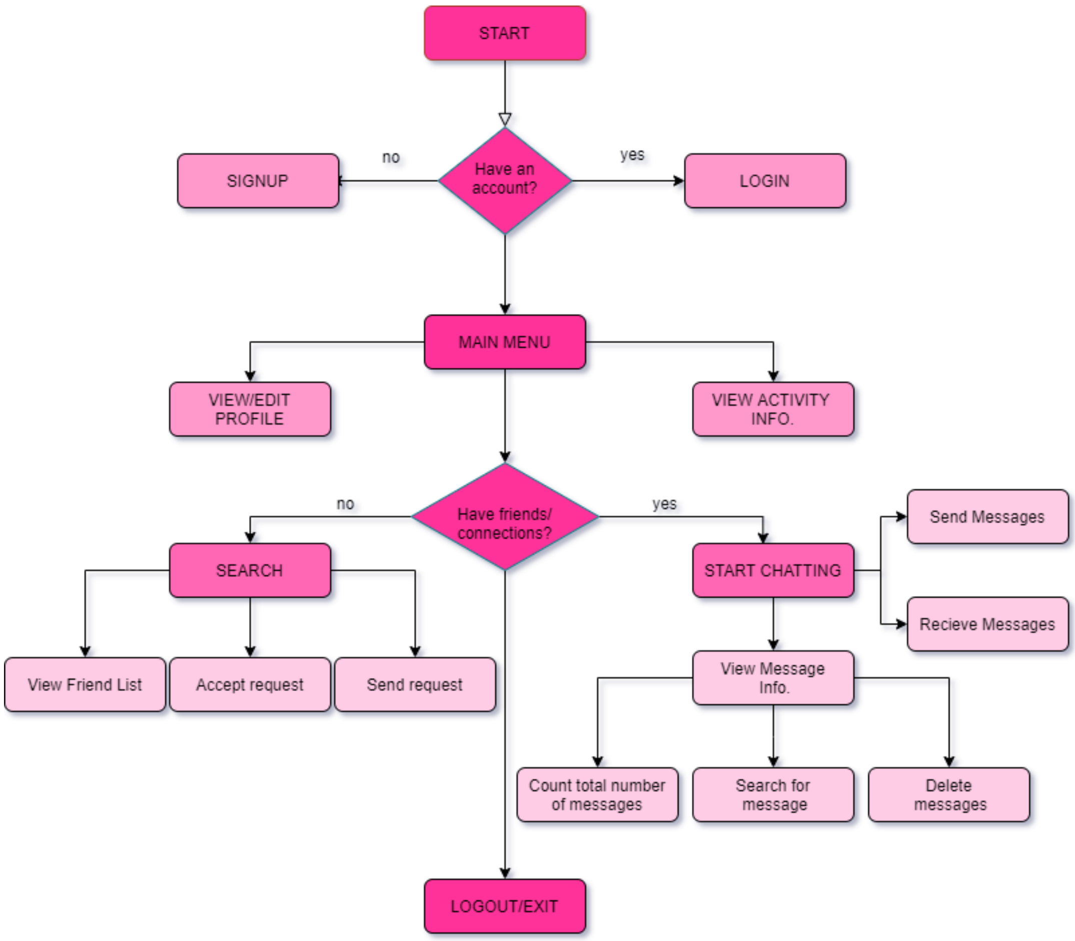
We have tried to build a social media platform which would facilitate the user to create profile, develop social network via making friends and enjoy chatting with them.

All the messaging and user related data would be stored in a database and accessed through it as per user demand. Here we have application which we have tried to develop doesn't require data transfer through internet, socket programming or anything like chat-bot. Our simple idea is to store and retrieve the messages and other data by executing appropriate queries which would then fetch the required data from tables stored in our database (in Oracle) and display the results. The description of the functionalities for both back-end and front-end is given as follows-

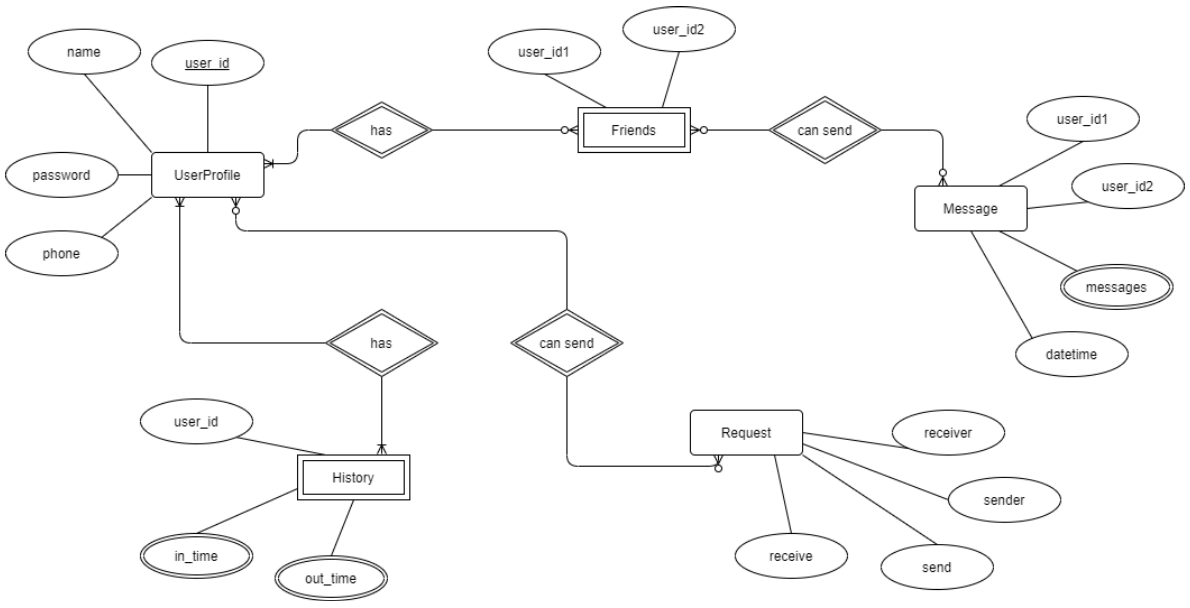
- **Login/Sign-up** : If you don't have an account then you must sign-up and if you already have an account then login by your username and password. Here, in sign-up and login there are some validations for each field and for that triggers are used in back-end database. Moreover, procedures and functions are used to execute the given functionalities.
- **View Profile** : You can view your name, password and other fields in this module. This module also provides functionality to delete your account from the application. Deleting your account will delete your friend, your stored messages and everything. To execute this we use a trigger which deletes your child record before the parent record.
- **Edit Profile** : This modules allows you to change your username, password or any other field which you want. Again for the proper validations and for updating each field, triggers and required procedures are used.
- **Sending request module** : Here, you can search for the people, send request to them and accept request of other people and become friends. This module also provides autocomplete functionality while searching for your friend and proper procedures are used to run this. Also, you can unfriend your friend if you wish, and this deletes all your child record (i.e. all messages of your friend and other history).
- **Friend List** : You can view all your friends in this module.
- **Chatting module** : You can chat with all your friends in this module. Simultaneously chatting is possible at a time if you run the project twice and login with different accounts.
- **Chatting information** : This module provides three different functionalities -
  * **Counting total number of messages** : You can view the total number of messages between you and your friend in chat till today's date and also you have access to delete all your messages that you have send.

   * **To view all the messages between given two dates** : You can view all your messages with your friend between two given dates, which you have to provide.
   * **To view the particular messages by providing the substring** : By just providing a word/substring you can view all the messages of you and your friend which contain that substring with date and time.

- **Activity module** : This module provides history and track record of number of times you entered this application and the displays the login and logout date and time.



2. Entity-Relationship Diagram



3. Table Design

User_profile

Column Name	Data Type	Constraints	Format	Description
user_id	Varchar2 (20)	Primary Key	natasha	It is a unique username of every user signing in the application
name	Varchar2 (20)	not null	Natasha Roy	Full name of the user
password	Varchar2 (8)	not null	@Natasha	Password of length 8 with validations (trigger)
phone	Varchar2 (10)	not null	9898989898	Contact number of user

Friends

Column Name	Data Type	Constraints	Format	Description
user_id1	Varchar2 (20)	Foreign key (referred from User_profile table)	naina	This is the username of the user which is referred from user profile table
user_id2	Varchar2 (20)	Foreign key (referred from User_ profile table)	suhu	This is the another username of the user which is referred from user profile table and is friend of user_id1

Message

Column Name	Data Type	Constraints	Format	Description
user_id1	Varchar2 (20)	Foreign key (referred from friends table)	naina	This is the username of the user which is referred from user profile table
user_id2	Varchar2 (20)	Foreign key (referred from friends table)	suhu	This is the another username of the user which is referred from user profile table and is friend of user_id1
messages	Varchar2(500)	not null	Hey girl, what's up ?	This column contains messages which is send by user_id1 and received by user_id2
datetime	date	not null	14/04/2020 23:11:08 (sysdate)	This column represents the time and date at which the message was send by user_id1

Request

Column Name	Data Type	Constraints	Format	Description
sender	Varchar2 (20)	Foreign key (referred from User_profile table)	naina	This is the username of the user which sends the request to receiver and is referred from user_profile table
receiver	Varchar2 (20)	Foreign key (referred from User_profile table)	suhu	This is the another username of the user which accepts or rejects the request of sender and is referred from user_profile table
sent	NUMBER	not null	1	If a request is send
receive	NUMBER	not null	0	If a request is accepted

History

Column Name	Data Type	Constraints	Format	Description
user_id	Varchar2 (20)	Foreign key (referred from User_profile table)	naina	This is the username of the user which is referred from user profile table
in_time	date	not null	14/04/2020 23:11:08 (sysdate)	This column represents the time when user login or signup the chat application.
out_time	date	not null	14/04/2020 23:20:08 (sysdate)	This column represents the time when user logout the chat application.

4. Stored Procedures

1. Signup Procedure

```
create or replace procedure signup_validation(uname IN user_profile.userID%TYPE, fullname IN user_profile.name%TYPE,pass IN user_profile.password%TYPE, ph IN user_profile.phone%TYPE,  ans OUT varchar ) as
BEGIN
        INSERT INTO user_profile VALUES (uname, fullname, pass,ph);
        ans := 'yes';
        exception
        when others then
        ans := 'no';
        rollback;
END;
/
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call signup_validation(?,?,?,?,?)}");
            ps.setString(1, jTextField3.getText());
            ps.setString(2, jTextField1.getText());
            ps.setString(4, jTextField2.getText());
            ps.setString(3, jPasswordField1.getText());
            ps.registerOutParameter(5, java.sql.Types.VARCHAR);
            ps.executeUpdate();

            String confirm = ps.getString(5);
            if (confirm.equals("no")) {
                JOptionPane.showMessageDialog(null, "Trigger Fired\n Make sure that phone no is of 10 digis\n Password length must be 8 \n Password must start from special character");
            } else {
                current_user = jTextField3.getText();
                Home y = new Home(current_user);
                y.setVisible(true);
                this.setVisible(false);
                this.dispose();

               CallableStatement ps2 = conn.prepareCall("{call login_activity(?,?)}");
                ps2.setString(1, jTextField3.getText());
                ps2.registerOutParameter(2, java.sql.Types.VARCHAR);
                ps2.executeUpdate();
```

2. View Profile

```
create or replace procedure viewprofile(username IN user_profile.userid%TYPE, uname OUT user_profile.userid%TYPE, fullname OUT user_profile.name%TYPE,ph OUT user_profile.phone%TYPE, pass OUT user_profile.password%TYPE, result OUT varchar ) as
CURSOR cu_viewprofile IS SELECT * FROM user_profile;
begin
    result := 'yes';
   FOR r_viewprofile in cu_viewprofile LOOP
   if (r_viewprofile.userid = username) then
       uname := r_viewprofile.userid;
       fullname := r_viewprofile.name;
       ph := r_viewprofile.phone;
       pass := r_viewprofile.password;
   end if;
   END LOOP;
   exception
   when others then
   result := 'no';
   rollback;

end;
/


declare
begin
   viewprofile('naina');
end
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call viewprofile(?,?,?,?,?,?)}");
            ps.setString(1,s);
            ps.registerOutParameter(2, java.sql.Types.VARCHAR);
            ps.registerOutParameter(3, java.sql.Types.VARCHAR);
            ps.registerOutParameter(4, java.sql.Types.VARCHAR);
            ps.registerOutParameter(5, java.sql.Types.VARCHAR);
            ps.registerOutParameter(6, java.sql.Types.VARCHAR);
            ps.executeUpdate();

            String confirm = ps.getString(6);
            System.out.println(confirm);
            if (confirm.equals("no")) {
                JOptionPane.showMessageDialog(null,"Something went wrong...");
            } else {
                jLabel6.setText(ps.getString(2));
                jLabel7.setText(ps.getString(3));
                jLabel8.setText(ps.getString(4));
                jLabel9.setText(ps.getString(5));
            }
```

3. Edit Profile

```
create or replace procedure editprofile(username IN user_profile.userid%TYPE,uname IN user_profile.userid%TYPE, fullname IN user_profile.name%TYPE, ph IN user_profile.phone%TYPE, pass IN user_profile.password%TYPE, ans OUT varchar) as
begin
   UPDATE user_profile
   SET userid = uname,
       name = fullname,
       phone = ph,
       password = pass
```

```
    WHERE userid = username;
    ans:='yes';
    exception
    when others then
    ans := 'no';
    rollback;
end;
/
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call editprofile(?,?,?,?,?,?)}");
            ps.setString(1, s);
            ps.setString(2, jTextField1.getText());
            ps.setString(3, jTextField2.getText());
            ps.setString(4, jTextField3.getText());
            ps.setString(5, jTextField4.getText());
            ps.registerOutParameter(6, java.sql.Types.VARCHAR);
            ps.executeUpdate();

            String confirm = ps.getString(6);
            System.out.println(confirm);
            if (confirm.equals("no")) {
                JOptionPane.showMessageDialog(null, "Triggered Fired\n Make sure that phone no is of 10 digis\n password lenth must be 8 \n It must start from special character");
            } else {
                JOptionPane.showMessageDialog(null,"Updated");
            }
```

4. Counting Messages

```
create or replace procedure countingmessage (user IN user_profile.userid%TYPE, friend IN user_profile.userid%TYPE, total OUT int, you OUT int, yourfriend OUT INT) as
CURSOR cu_message IS SELECT * FROM message;
BEGIN
    total := 0;
    you := 0;
    yourfriend := 0;
   FOR r_message in cu_message LOOP
       if(r_message.userID1 = user and r_message.userID2 = friend ) then
               total := total + 1;
               you := you + 1;
        end if;
       if (r_message.userID2 = user and r_message.userID1 = friend ) then
               total := total + 1;
               yourfriend := yourfriend + 1;
        end if;
   END LOOP;
   exception
   when others then
   total := 0;
   rollback;
END
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call countingmessage(?,?,?,?,?)}");
                ps.setString(1, s);
                ps.setString(2, friend);
                ps.registerOutParameter(3, java.sql.Types.INTEGER);
                ps.registerOutParameter(4, java.sql.Types.INTEGER);
                ps.registerOutParameter(5, java.sql.Types.INTEGER);
                ps.executeUpdate();

                String total = ps.getString(3);
                String you = ps.getString(4);
                String yourfriend = ps.getString(5);
```

5. Delete Profile

```
create or replace procedure deleteprofile (username IN user_profile.userid%TYPE, ans OUT varchar)
as
begin
ans := 'yes';
    DELETE from user_profile where userid = username;
exception
   when others then
   ans := 'no';
   rollback;
end;
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call deleteprofile(?,?)}");
            ps.setString(1, s);
            ps.registerOutParameter(2, java.sql.Types.VARCHAR);
            ps.executeUpdate();
            String confirm = ps.getString(2);
            if (confirm.equals("yes")) {
                JOptionPane.showMessageDialog(null, "Your messages, friends and account has been
deleted");
            }
```

6. Delete Messages

```
create or replace procedure deletemessages (username IN user_profile.userid%TYPE, friend IN
user_profile.userid%TYPE, ans OUT varchar) as
begin
ans := 'yes';
    DELETE from message where userid1 = username and userid2 = friend;
exception
   when others then
   ans := 'no';
   rollback;
end;
```

JAVA Query

```
String f = jComboBox1.getSelectedItem().toString();
            CallableStatement ps = conn.prepareCall("{call deletemessages(?,?,?)}");
            ps.setString(1, s);
            ps.setString(2, f);
            ps.registerOutParameter(3, java.sql.Types.VARCHAR);
            ps.executeUpdate();
            String confirm = ps.getString(3);
            if (confirm.equals("yes")) {
                JOptionPane.showMessageDialog(null, "The messages that you sended to your friend
has been deleted");
            }
```

7. History

```
create or replace procedure track_activity(username IN user_profile.userid%TYPE, time OUT varchar,
n OUT int ) as
CURSOR cu_history IS SELECT * FROM history;
BEGIN
     n := 0;
     FOR r_history IN cu_history LOOP
          if (r_history.userid = username and r_history.outtime IS NOT NULL) then
             n := n+1;
             time := CONCAT(time,TO_CHAR(r_history.intime));
             time := CONCAT(time,'%');
             time := CONCAT(time,TO_CHAR(r_history.intime, 'HH24:MI:SS'));
             time := CONCAT(time,',');
             time := CONCAT(time,TO_CHAR(r_history.outtime, 'HH24:MI:SS'));
             time := CONCAT(time,'$');
         end if;
     END LOOP;
   exception
   when others then
   time := 'no';
   rollback;
END;
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call track_activity(?,?,?)}");
            ps.setString(1, s);
            ps.registerOutParameter(2, java.sql.Types.VARCHAR);
            ps.registerOutParameter(3, java.sql.Types.INTEGER);
            ps.executeUpdate();

            String time = ps.getString(2);
            int n = ps.getInt(3);

            jLabel1.setText("Total number of times logged and logged out: "+n);

            char t[] = time.toCharArray();

            String temp = "Date: ";

            for (int i = 0; i < t.length-1; i++) {
                if(t[i] == '$'){
                   temp = temp.concat("\n \n \nDate: ");
                }else if(t[i] == ','){
                    temp = temp.concat("\t Logged out Time: ");
                }else if(t[i] == '%'){
                    temp = temp.concat("\nLogged in Time: ");
                }else{
                    temp = temp + t[i];
                }
            }
```

8. Login Activity

```
create or replace procedure login_activity(username IN user_profile.userid%TYPE, ans OUT varchar)
as
begin
        INSERT INTO history VALUES (username, sysdate, null);
        ans := 'yes';
        exception
        when others then
        ans := 'no';
        rollback;
end;
```

JAVA Query

```
CallableStatement ps2 = conn.prepareCall("{call login_activity(?,?)}");
                ps2.setString(1, jTextField3.getText());
                ps2.registerOutParameter(2, java.sql.Types.VARCHAR);
                ps2.executeUpdate();
```

9. Logout Activity

```
create or replace procedure logout_activity(username IN user_profile.userid%TYPE, ans OUT varchar)
as
begin
        UPDATE history set outtime = sysdate where userid = username;
        ans := 'yes';
        exception
        when others then
        ans := 'no';
        rollback;
end;
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call logout_activity(?,?)}");
                ps.setString(1, s);
                ps.registerOutParameter(2, java.sql.Types.VARCHAR);
                ps.executeUpdate();
```

10. Unfriend

```
create or replace procedure unfriend (username IN user_profile.userid%TYPE, friend IN
user_profile.userid%TYPE, ans OUT varchar) as
begin
ans := 'yes';
    DELETE from friends where userID1 = username and userID2 = friend;
    DELETE from friends where userID2 = username and userID1 = friend;
exception
   when others then
   ans := 'no';
   rollback;
end;
```

JAVA Query

```
CallableStatement ps = conn.prepareCall("{call unfriend(?,?,?)}");
            ps.setString(1, curn);
            ps.setString(2, f);
            ps.registerOutParameter(3, java.sql.Types.VARCHAR);
            ps.executeUpdate();
            String confirm = ps.getString(3);
```

5. Functions

1. Login Function

```
CREATE OR REPLACE FUNCTION login(name varchar,password varchar)
RETURN varchar AS ans varchar(5);
CURSOR cu_login IS SELECT * FROM user_profile;
```

```
BEGIN
     ans := 'no';
     FOR r_login IN cu_login LOOP
        IF(name = r_login.userID) THEN
            IF(password = r_login.password)THEN
               ans := 'yes';
            END IF;
        END IF;
     END LOOP;
     RETURN ans;
END;
/
```

JAVA Query

```
String query = "select login('" + jTextField1.getText() + "','" + jPasswordField1.getText() + "') from dual";
            PreparedStatement stmt = conn.prepareStatement(query);
            ResultSet rs = stmt.executeQuery();
            String check = null;
            while (rs.next()) {
                check = rs.getString(1);
            }

            if (check.equals("no")) {
                JOptionPane.showMessageDialog(null, "Invalid username/password");
            } else {
                current_user = jTextField1.getText();
                Home y = new Home(current_user);
                y.setVisible(true);
                this.setVisible(false);
                this.dispose();

           CallableStatement ps = conn.prepareCall("{call login_activity(?,?)}");
                ps.setString(1, jTextField1.getText());
                ps.registerOutParameter(2, java.sql.Types.VARCHAR);
                ps.executeUpdate();

            }
```

2. Message form Date

```
create or replace function MessagesFromDates(from_d varchar, to_d varchar, username varchar, friend varchar)
RETURN varchar AS ans varchar(2000);
CURSOR cu_message IS SELECT * FROM message ORDER BY DANDT;
flag int;
BEGIN
flag := 0;
FOR r_message in cu_message LOOP
    if( ((r_message.userid1 = username) and (r_message.userid2 = friend)) or ((r_message.userid2 = username) and (r_message.userid1 = friend)) ) then
        if ( (to_char(r_message.DandT) > from_d) and (to_char(r_message.DandT) < to_d) ) then

          flag := 1;
          ans := CONCAT(ans,to_char(r_message.DandT));
          ans := CONCAT(ans,'$');
          dbms_output.Put_line(r_message.DandT);
          if(r_message.userid1 = username) then
             ans := CONCAT(ans,'You to ');
             ans := CONCAT(ans, r_message.userid2);
             ans := CONCAT(ans,': ');
             ans := CONCAT(ans,r_message.msg);
             ans := CONCAT(ans,'%');
          else
             ans := CONCAT(ans,r_message.userid1);
             ans := CONCAT(ans, ' to you: ');
             ans := CONCAT(ans,r_message.msg);
             ans := CONCAT(ans,'%');
```

```
            end if;
          end if;
      end if;
END LOOP;
    if (flag =0)then
       ans := CONCAT(ans,'No messages found');
    end if;
RETURN ans;
END;
/
```

JAVA Query

```
String query = "select MessagesFromDates('"+ f_date +"','"+t_date +"','"+s+"','"+f+"') from dual";
            System.out.println(query);
            PreparedStatement stmt = conn.prepareStatement(query);
            ResultSet rs = stmt.executeQuery();
            String check = null;
            while (rs.next()) {
                check = rs.getString(1);
            }
            char c[] = check.toCharArray();
            String temp = "";

            for (int i = 0; i < c.length-1; i++) {
                if(c[i] == '$'){
                   temp = temp.concat("\n");
                }else if(c[i] == '%'){
                    temp = temp.concat("\n \n");
                }else{
                    temp = temp + c[i];
                }
            }
            jTextArea1.setText(temp);
```

3. Search Message from given Substring

```
create or replace function message_subs(message  varchar, username varchar, friend varchar)
RETURN varchar AS ans varchar(200);
CURSOR cu_message IS SELECT * FROM message ORDER BY DANDT;
position int;
flag int;
BEGIN
flag := 0;
FOR r_message in cu_message LOOP
    if( ((r_message.userid1 = username) and (r_message.userid2 = friend)) or ((r_message.userid2 = username) and (r_message.userid1 = friend)) ) then
      position := INSTR(LOWER(r_message.msg),LOWER(message));
      if (position != 0) then
          flag := 1;
          ans := CONCAT(ans,to_char(r_message.DandT));
          ans := CONCAT(ans,'$');
          dbms_output.Put_line(r_message.DandT);
          if(r_message.userid1 = username) then
             ans := CONCAT(ans,'You to ');
             ans := CONCAT(ans, r_message.userid2);
             ans := CONCAT(ans,': ');
             ans := CONCAT(ans,r_message.msg);
             ans := CONCAT(ans,'%');
          else
             ans := CONCAT(ans,r_message.userid1);
             ans := CONCAT(ans, ' to you: ');
             ans := CONCAT(ans,r_message.msg);
             ans := CONCAT(ans,'%');
          end if;
       end if;
    end if;
END LOOP;
    if (flag =0)then
```

```
        ans := CONCAT(ans,'No such message found');
      end if;
    RETURN ans;
    END;
    /
```

JAVA Query

```
String query = "select message_subs('" + substring + "','"+s+"','"+f+"') from dual";
            PreparedStatement stmt = conn.prepareStatement(query);
            ResultSet rs = stmt.executeQuery();
            String check = null;
            while (rs.next()) {
                check = rs.getString(1);
            }
            char c[] = check.toCharArray();
            String temp = "";

            for (int i = 0; i < c.length-1; i++) {
                if(c[i] == '$'){
                   temp = temp.concat("\n");
                }else if(c[i] == '%'){
                    temp = temp.concat("\n \n");
                }else{
                    temp = temp + c[i];
                }
            }
```

6. Triggers

1. Account Detail Validation

```
CREATE OR REPLACE TRIGGER check_signup
BEFORE INSERT ON user_profile
FOR EACH ROW
DECLARE
     prefix VARCHAR2(1);
     CURSOR cu_signup IS SELECT * FROM user_profile;
BEGIN
FOR r_signup in cu_signup LOOP
        IF(r_signup.userID = :new.userID) THEN
           dbms_output.Put_line('Username already exists....Try another one.');
           Raise_Application_Error (-20001, 'Duplicate username found');
        END IF;
END LOOP;

prefix := substr(:new.password,1,1);
if(LENGTH(:new.password) != 8) then
    dbms_output.Put_line('The length of the password must be 8 characters');
    Raise_Application_Error (-20002, 'Password Length must be 8');
end if;
if(LENGTH(:new.phone) != 10) then
    dbms_output.Put_line('The length of the phone number must be 10');
    Raise_Application_Error (-20003, 'Phone number must be size of 10');
end if;
if( (REGEXP_LIKE(prefix, '[a-z]')) or (REGEXP_LIKE(prefix, '[0-9]')) or (REGEXP_LIKE(prefix, '[A-Z]'))) then
    dbms_output.Put_line('The first letter of the password must be special character');
    Raise_Application_Error (-20004, 'first letter of the password must be special character');
end if;
END;
/
```

2. Account Edit Details

```
CREATE OR REPLACE TRIGGER check_updation
BEFORE UPDATE ON user_profile
FOR EACH ROW
DECLARE
     prefix VARCHAR2(1);
BEGIN
prefix := substr(:new.password,1,1);
if(LENGTH(:new.password) != 8) then
    dbms_output.Put_line('The length of the password must be 8 characters');
    Raise_Application_Error (-20002, 'Password Length must be 8');
end if;
if(LENGTH(:new.phone) != 10) then
    dbms_output.Put_line('The length of the phone number must be 10');
    Raise_Application_Error (-20003, 'Phone number must be size of 10');
end if;
if( (REGEXP_LIKE(prefix, '[a-z]')) or (REGEXP_LIKE(prefix, '[0-9]')) or (REGEXP_LIKE(prefix, '[A-Z]'))) then
    dbms_output.Put_line('The first letter of the password must be special character');
    Raise_Application_Error (-20004, 'first letter of the password must be special character');
end if;
END;
/
```

3. Delete Profile

```
create or replace trigger Delete_Profile before delete on user_profile
for each row
begin
       delete from message where userID1 = :old.userid or userID2 = :old.userid;
       delete from friends where userID1 = :old.userid or userID2 = :old.userid;
       delete from request where sender = :old.userid or accepter = :old.userid;
       delete from history where userID=:old.userid;
end;
```

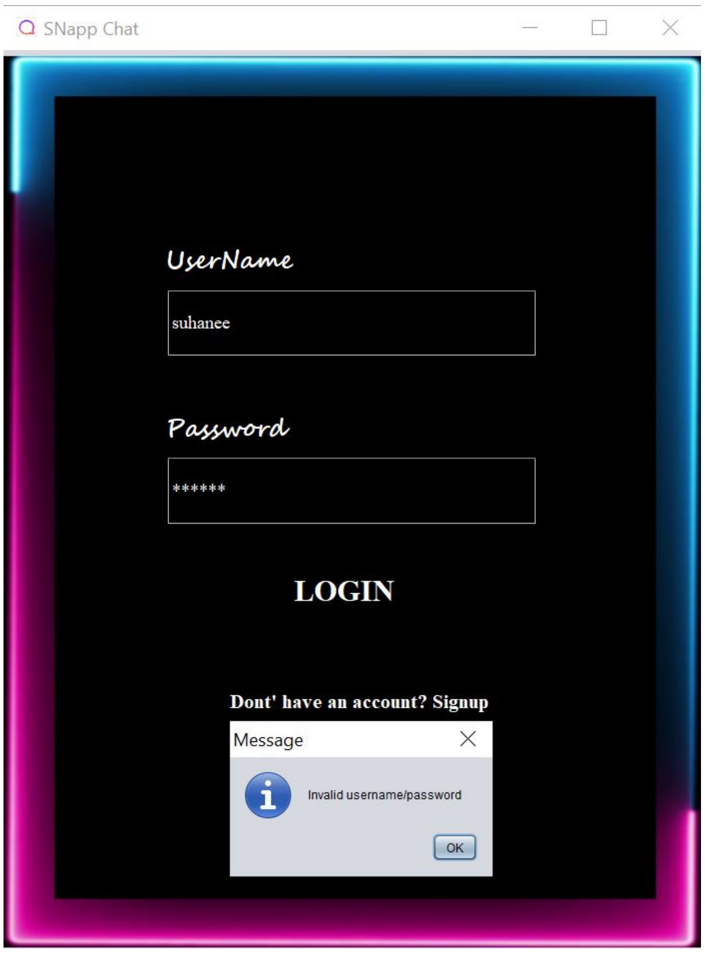
4. Unfriend

```
create or replace trigger Delete_friends before delete on friends
for each row
declare
CURSOR cu_message IS SELECT * FROM message;
CURSOR cu_request IS SELECT * FROM request;
begin
     FOR r_message in cu_message LOOP
          if ( (r_message.userid1 = :old.userID1 and r_message.userid2 = :old.userID2) ) then
             delete from message where userid1 = :old.userID1  and userid2 = :old.userID2 ;
          elsif ( (r_message.userid1 = :old.userID2 and r_message.userid2 = :old.userID1) ) then
             delete from message where userid1 = :old.userID2  and userid2 = :old.userID1 ;
          end if;
     END LOOP;

     FOR r_request in cu_request LOOP
          if ( (r_request.sender = :old.userID1 and r_request.accepter = :old.userID2) ) then
             delete from request where sender = :old.userID1  and accepter = :old.userID2 ;
          elsif ( (r_request.sender = :old.userID2 and r_request.accepter = :old.userID1) ) then
             delete from request where sender = :old.userID2  and accepter = :old.userID1 ;
          end if;
     END LOOP;
end;
```

7. Screenshots

- Login Page : It has simple validation function for matching the password, wrong username/password will deny access. Signup to create a new account.


SNapp Chat

UserName

suhanee

Password

******

LOGIN

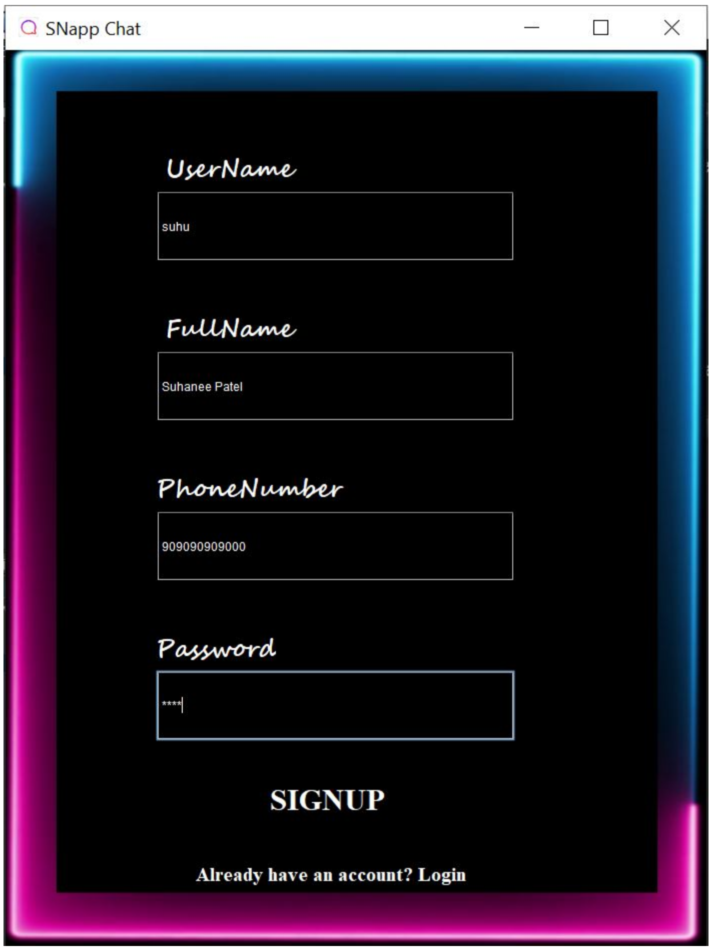
Dont' have an account? Signup

Message

Invalid username/password

OK

- Signup Page : It has a signup procedure and 3 validations, first to check the correctness of phone number (10 digits) , second to check the strength of password (minimum 1 special character and 8 characters in total) , and third will not allow duplicate user IDs.


SNapp Chat

UserName

suhu

FullName

Suhanee Patel

PhoneNumber

909090909000

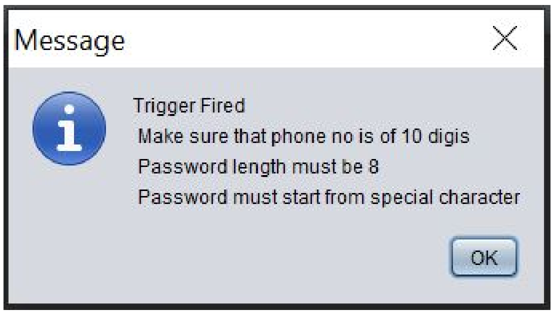
Password

****

SIGNUP

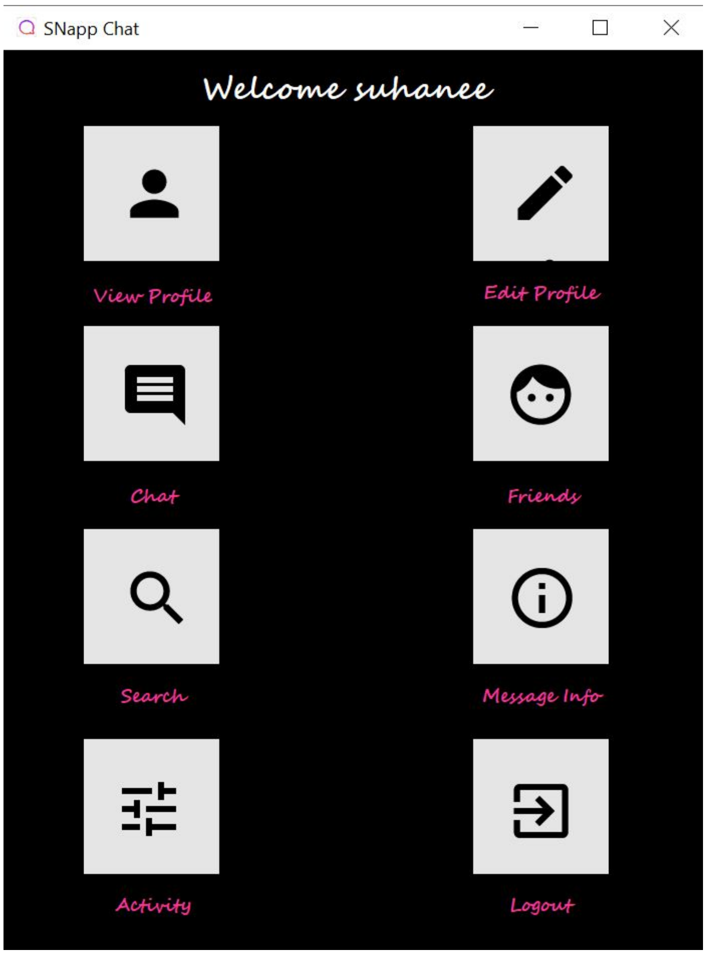
Already have an account? Login

If any of the above condition is violated the user wont be allowed to create and account and the trigger for same will raise and error.

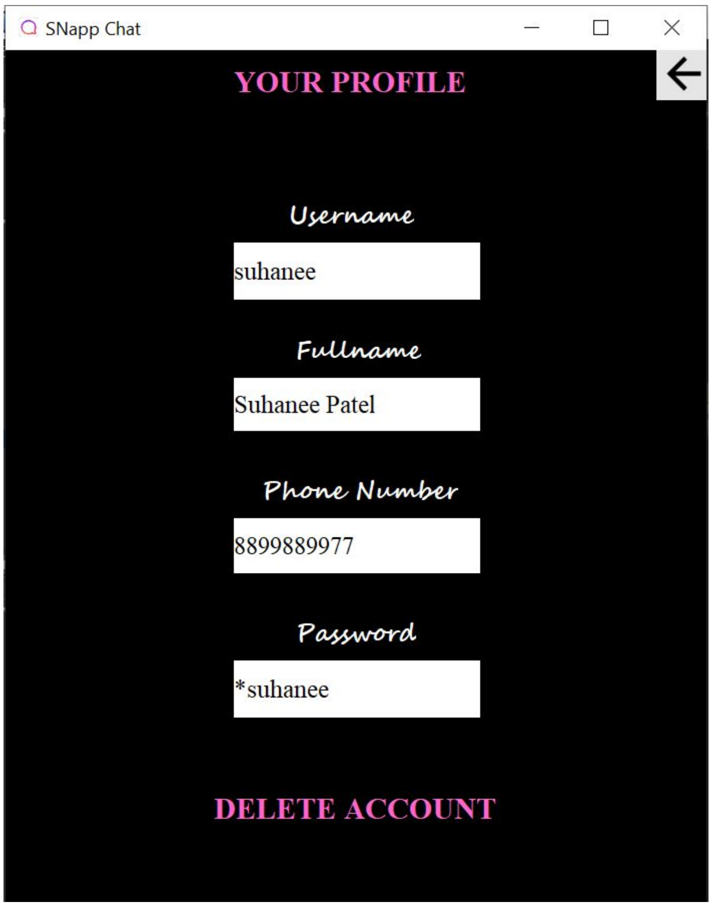


Trigger Fired
Make sure that phone no is of 10 digis
Password length must be 8
Password must start from special character

- Home Page : Once login/signup is done successfully, the user will land on the home page which has 8 main options and each option might contain sub options.



- View Profile : User can view his/her profile here. It has a "Delete Account" option at the bottom which will allow the user to delete his/her account.


SNapp Chat

YOUR PROFILE

Username: suhanee

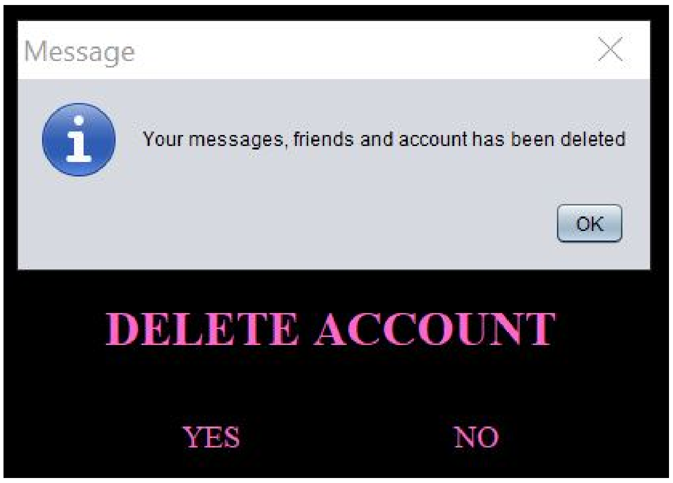
Fullname: Suhanee Patel

Phone Number: 8899889977

Password: *suhanee

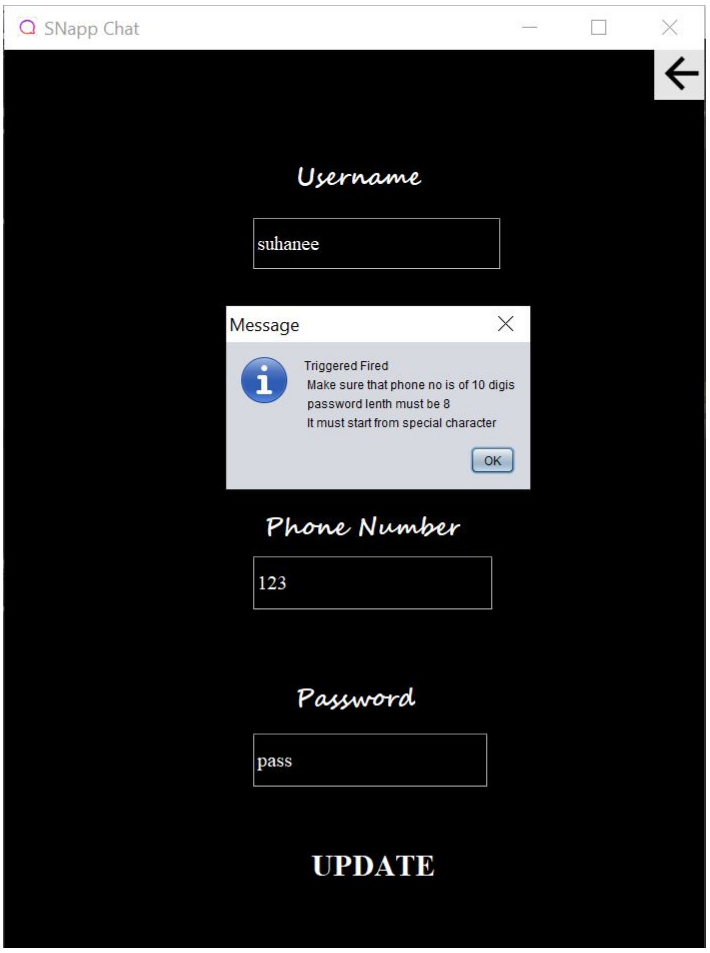
DELETE ACCOUNT

All the details related with his/her messages, friends and history will also be deleted. And a trigger for the same will be fired during this procedure.


Message

Your messages, friends and account has been deleted

- Edit Profile : User can edit his/her profile here. All the validations will be checked during updating the profile details and if any condition is violated trigger will be fired.



SNapp Chat

Username

suhanee

Message

Triggered Fired
Make sure that phone no is of 10 digis
password lenth must be 8
It must start from special character

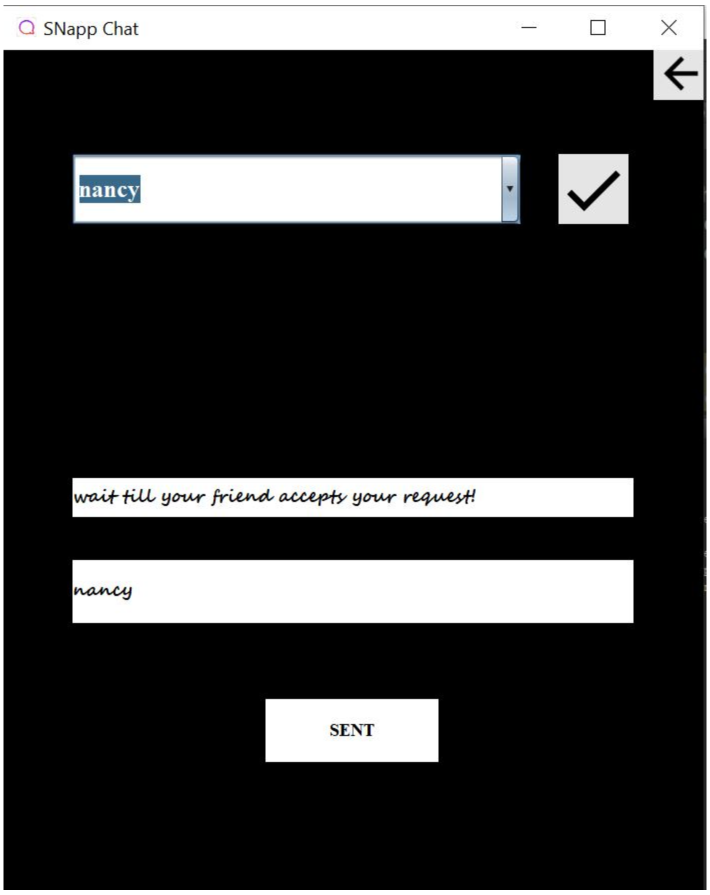
Phone Number

123

Password

pass

- Search for Friends : User can search for other users and send/accept request to make connections. Once the connections are made the user can chat with his/her friends. The drop down box along with autocomplete facility allows you to search and make new friends.



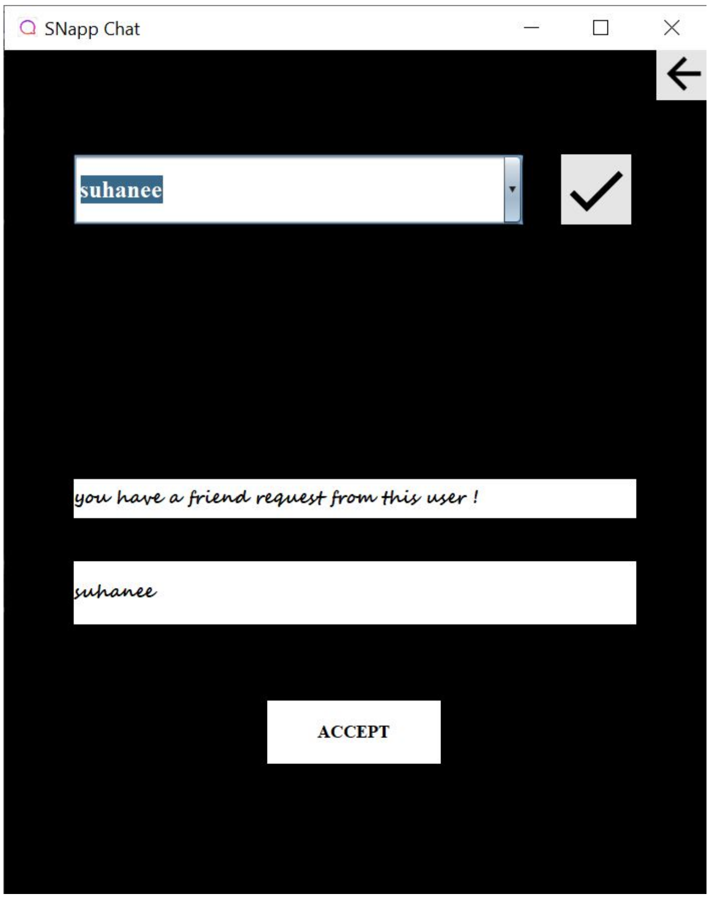
SNapp Chat

nancy

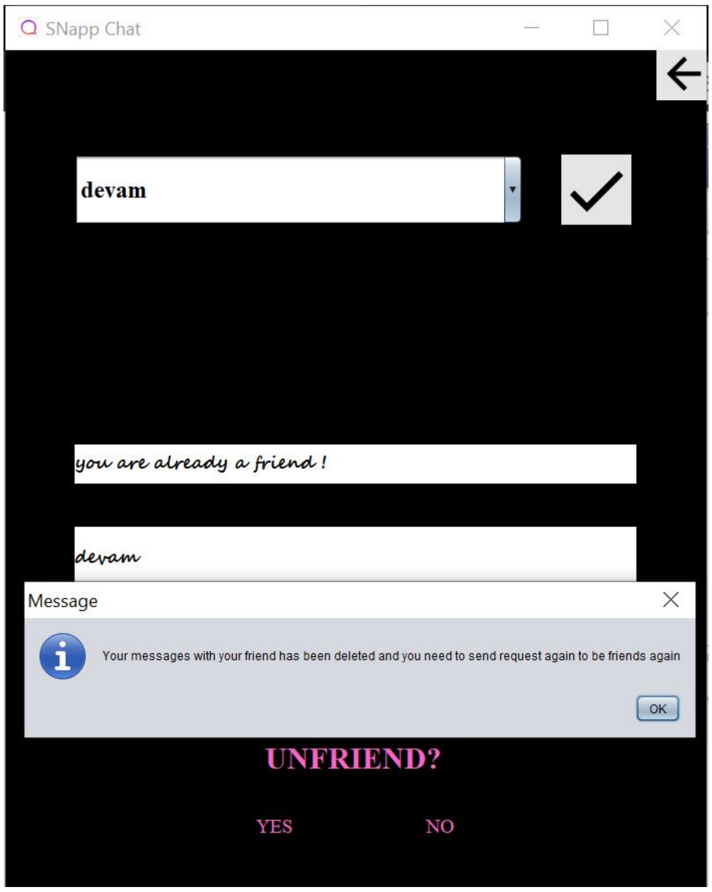
wait till your friend accepts your request!

nancy

Once the you send a friend request to anyone you can wait until he/she accepts it.



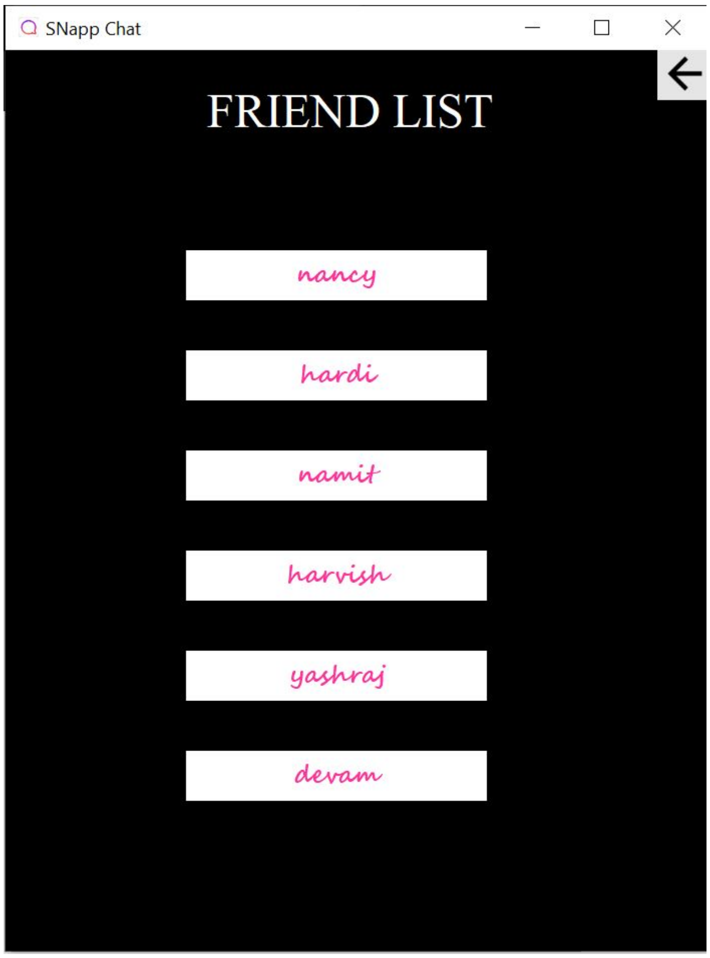
If the other person accepts your friend request then you are now friends and can now enjoy the chatting experience. You can make many friends similarly and can also "Unfriend" your friends.



Your messages with your friend has been deleted and you need to send request again to be friends again

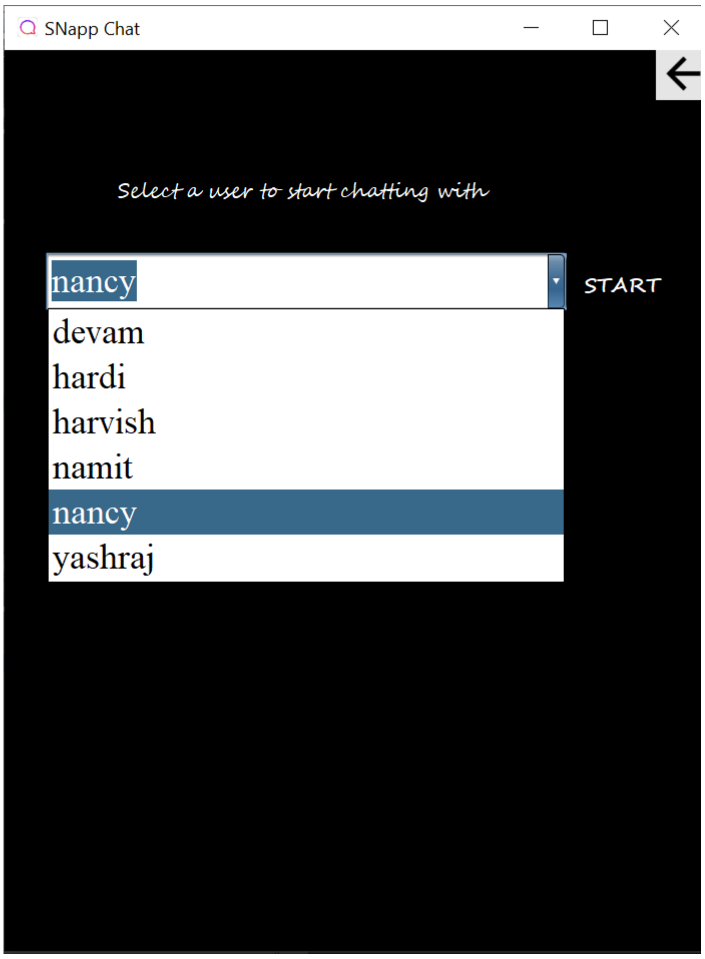
All the messages from your side to that person will be deleted when you unfriend him/her. Trigger for the same will be executed and you will receive a pop up using the same.

- Friend List : This page will display userIDs of all your friends.


SNapp Chat

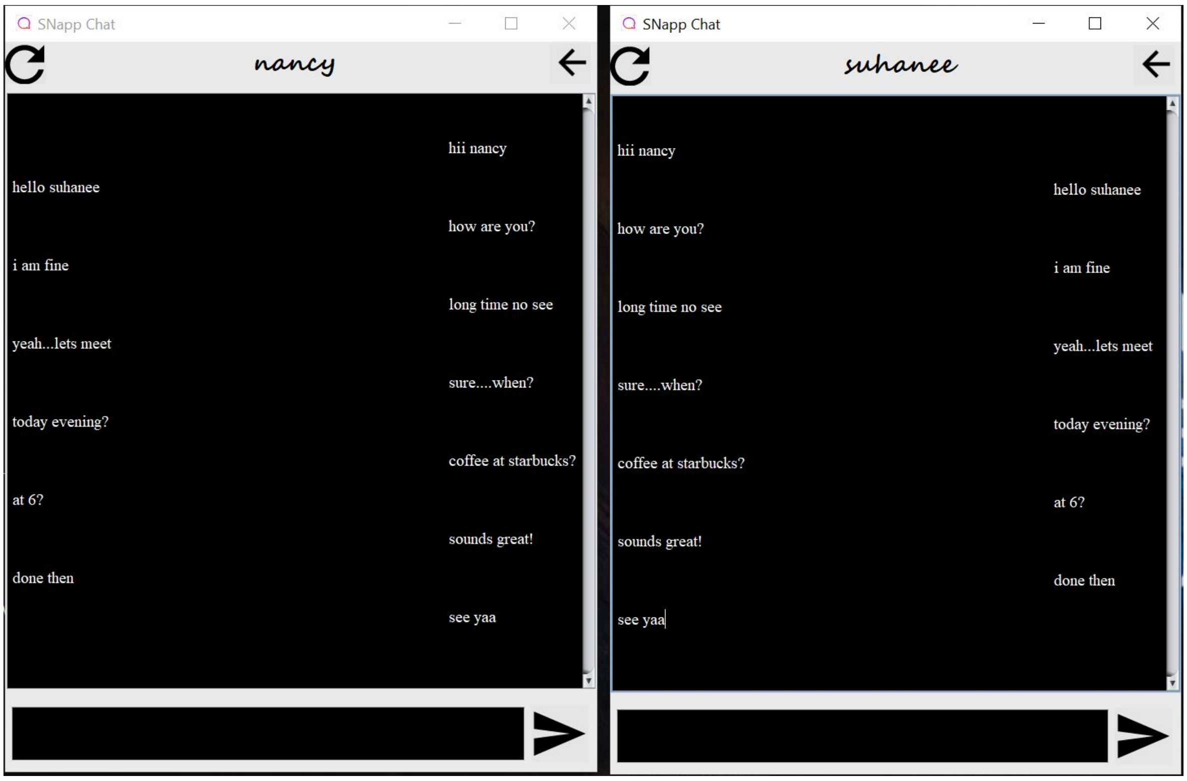
FRIEND LIST

- Message : You can select anyone from your friend list to start chatting with.


SNapp Chat

Select a user to start chatting with

- Chatting : You can send and receive messages.


SNapp Chat

nancy

hii nancy

hello suhanee

how are you?

i am fine

long time no see

yeah...lets meet

sure....when?

today evening?

coffee at starbucks?

at 6?

sounds great!

done then

see yaa

SNapp Chat

suhanee

hii nancy

hello suhanee

how are you?

i am fine

long time no see

yeah...lets meet

sure....when?

today evening?

coffee at starbucks?

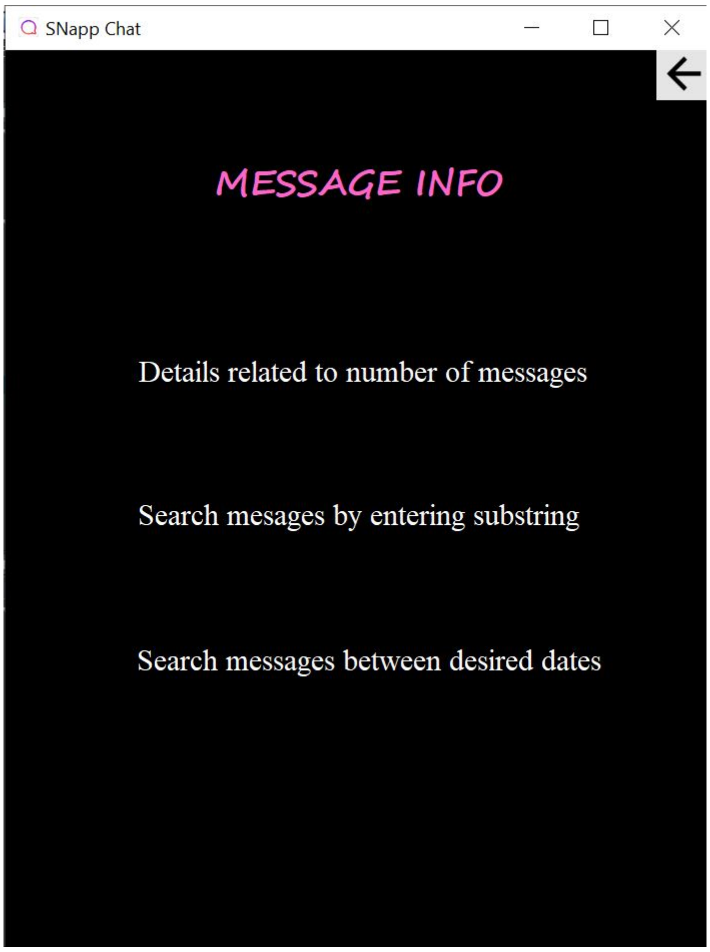
at 6?

sounds great!

done then

see yaa

- Message Info : This page will provide you with three functionalities.


SNapp Chat

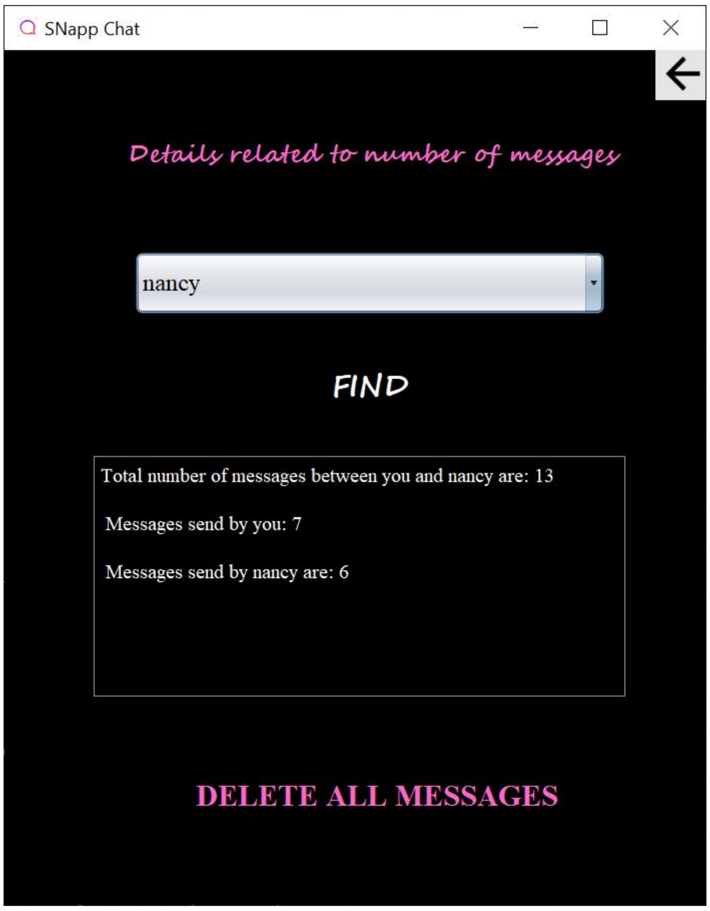
MESSAGE INFO

Details related to number of messages

Search mesages by entering substring

Search messages between desired dates

- Count Messages : First is you can count the total number of messages that is shared with any of your selected friend.



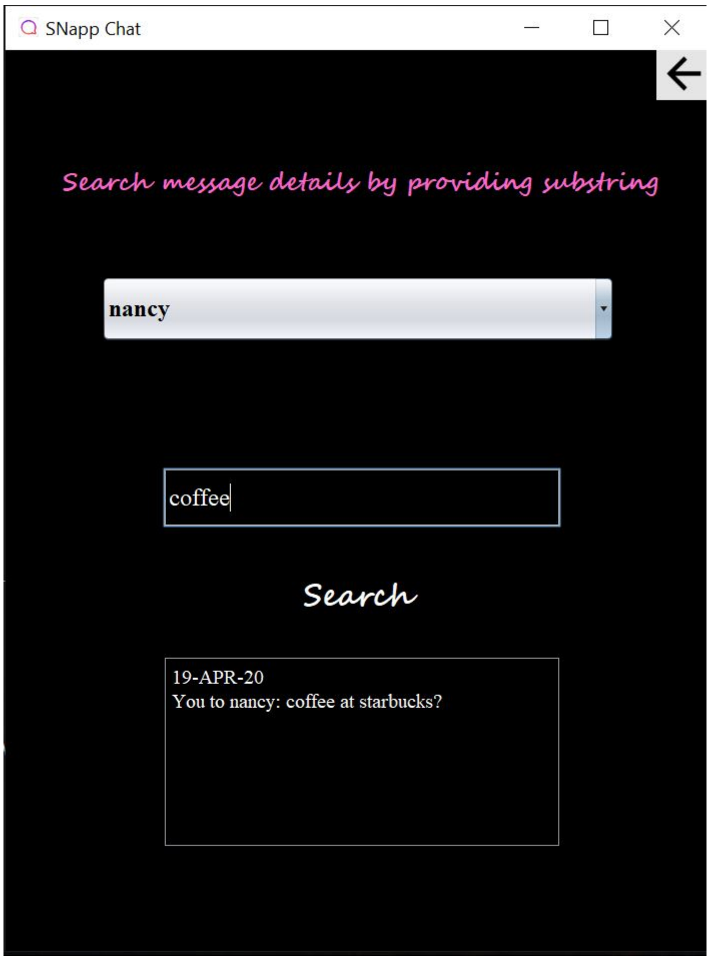
Total number of messages between you and nancy are: 13

Messages send by you: 7

Messages send by nancy are: 6

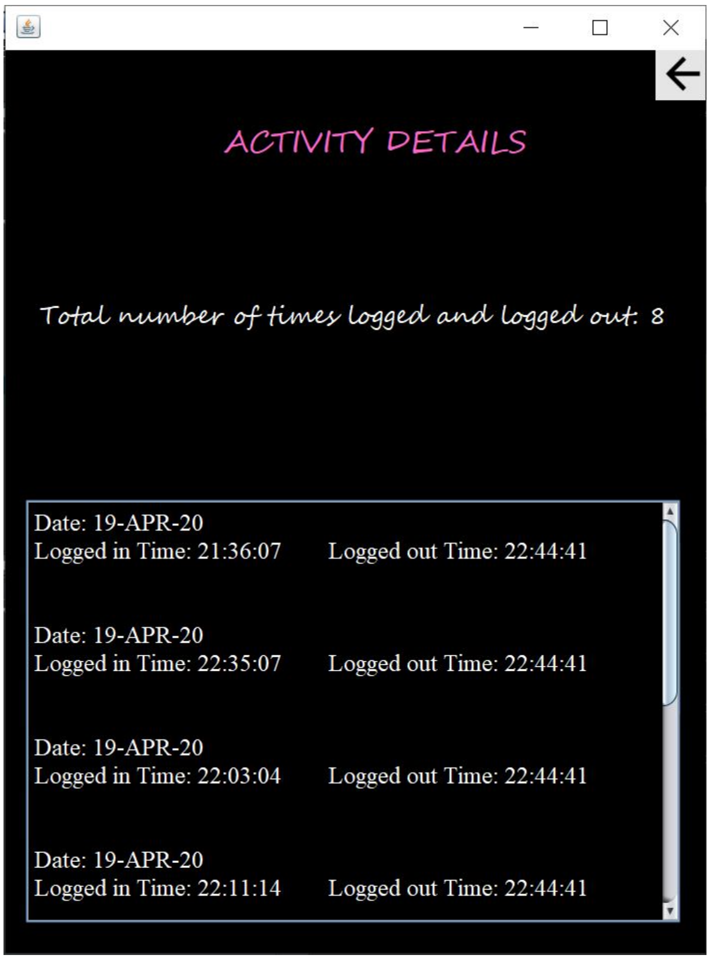
This page also has the functionality to delete all the messages from your side.

- Search for message : You can search for the message by providing a substring of it.



19-APR-20
You to nancy: coffee at starbucks?

- Activity : This page will display the total number of timed you have logged in/logged out in the application along with the date and time.



ACTIVITY DETAILS

Total number of times logged and logged out: 8

```
Date: 19-APR-20
Logged in Time: 21:36:07      Logged out Time: 22:44:41


Date: 19-APR-20
Logged in Time: 22:35:07      Logged out Time: 22:44:41


Date: 19-APR-20
Logged in Time: 22:03:04      Logged out Time: 22:44:41


Date: 19-APR-20
Logged in Time: 22:11:14      Logged out Time: 22:44:41
```