Minor Project Report

Resolving Multi-Party Privacy Conflicts in Social Media

Submitted by

G Supriya(B121286)

S Sudharani(B121289)

Under Supervision of:

RamaKrishna Sir

Department of Computer Science & Engineering

Rajiv Gandhi University of Knowledge Technologies

Basar-504107, INDIA

April, 2017

Certificate

It is certified that the work contained in this report titled “Resolving Multi-Party Privacy Conflicts in Social Media” is the original work done by G.Supriya(B121286), S.Sudharani(B121289), and has been carried out under our supervision.

_____

Mr. Samit K. Pradhan
Assistant Professor
Department of CSE
RGUKT, Basar

_____

Mr. K. Parsuram
Lab Instructor
Department of CSE
RGUKT, Basar

Date:18-06-2017

Acknowledgement

The title of our project is Resolving Multi-Party Privacy Conflicts in Social Media. This project is done as a part of our academic curriculam during B.Tech third year of graduation course under the guidance of .

This report includes our documents from requirement specification to design level documents. The contents of this are followed according to IEEE standard format and those were written without any plagiarism.

G.Supriya (B121286)

S.Sudharani (B121289)

Abstract

Item shared through Social Media may affect more than one user's privacy e.g.,photos that depict multiple users,comments that mention multiple users,events in which multiple users are invited,etc. Current Social Infrastructures(facebook, twitter and etc.) are lack in multi-party privacy management support.These makes users unable to appropriately control to whom these items are actually shared or not.Recent computational mechanisms are able to merge the privacy preferences of multiple users into a single policy for an item. This can help to solve this problem.However, merging multiple user's privacy preferences is not an easy task,because privacy preferences may conflict, so some methods are needed to resolve those conflicts.Those methods must have in order to propose solutions that can be acceptable by all of the users affected by the item to be shared.

Current approaches are only consider fixed ways of aggregating privacy preferences.In this project we propose the first computational mechanism to resolve conflicts for multi-party privacy management in Social media. That is able to adapt to different situations.

Contents

1. Introduction

2. Software Requirement Specification

2.1 Introduction

2.1.1 Purpose.....1

2.1.2 Document Conventions.....1

2.1.3 Intended Audience and Reading Suggestions.....

2.1.4 Product Scope.....1

2.2 Overall Description.....2

2.2.1 Product Perspective.....2

2.2.2 Product Functions.....2

2.2.3 User Classes and Characteristics.....3

2.2.4 Operating Environment.....

2.2.5 Design and Implementation.....

2.2.6 User Documentation.....

2.2.7 Assumptions and Dependencies.....3

2.3.External Interface Requirements.....3

2.3.1 User Interfaces.....

2.3.2 Hardware Interfaces.....

2.3.3 Software Interfaces.....3

2.3.4 Communications Interfaces.....4

2.4 System Features.....4

2.4.1 Functional Requirements.....4

2.5 Nonfunctional Requirements.....5

2.5.1 Safety Requirements.....5

2.5.2 Security Requirements.....5

2.5.3 Software Quality Attribute.....

2.6 Other Requirements.....5
2.6.1 Hardware constraints.....5
2.6.2 Software constraints.....5

3. Detailed Design Document

3.1. Introduction.....6

3.1.1 Objective.....6

3.1.2 Scope.....6

3.2 Architectural and Component-Level Design

3.2.1 Architecture diagram.....7

3.3 Description of Components

3.3.1 Data flow diagrams.....8

3.3.2 Sequence diagrams.....8

3.3.3 Class Diagram.....8

3.4 Graphical user interface module.

3.4.1 Class #1 User.....

3.4.2 Class #2 Admin.....

3.4 Data Architecture.......13

3.4.1 Entity-Relation Diagram.....14

3.4.2 Data dictionary.....14

4. Implementation and Screen shots

4.1 Introduction.....15

4.1.1 Flow charts.....16

4.1.1.1 Flow chart of Admin.....15

4.1.1.2 Flow chart of User.....16

4.2 Modules and Screen shots.....15

5. Software test Documentation(UTP)

5.1 GUI Tests......20

5.1.1 Introduction.....20
System overview.....21

Test approach.....21

5.1.2 Test Plan.....22

Functionality.....23

Features to tested.....24

Features not to tested.....25

5.1.3 Test Cases.....26

Case GUI test1.....27
Name.....28

Purpose.....29

Inputs.....30

Expected output.....31

Test procedure.....31

Test outcome.....32
Case GUI test2.....33

Name.....33

Purpose.....34

Inputs.....34

Expected output.....35

Test procedure.....35

Test outcome.....36

Case GUI test1.....37

Name.....37

Purpose.....38

Inputs.....39

Expected output.....39

Test procedure.....40

Test outcome.....40

Case GUI test4.....41

Name.....41

Purpose.....42

Inputs.....42

Expected output.....43

Test procedure.....43

Test outcome.....43

6. Conclusion.....

List of Tables

Table 1 : User login table.....34

Table 2 : Adminstrator login table.....34

Table 3 : Messages table....….....34

Table 4 : Images table.....

Table 5 : Request table.....

Table 6 : Sign In table.....

List of figures

fig 1: Use case diagram.....14

fig 2: Architecture diagram.....20

fig 3: Level-0-DFD ….....21

fig 4: Level-1-DFD.....22

fig 5: Sequence diagram

fig 7: Class diagram.....25

fig 9: Entity-relation diagram.....33

1.Introduction

Hundreds of billions of items that are uploaded to Social Media are co-owned by multiple users,yet only the user that uploads the item is allowed to set its privacy settings. This is a serious problem as user's privacy preferences for co-owned items usually conflict, so applying the preferences of only one party risks such items being shared with undesired recipients, which can lead to privacy violations with severe consequence. Examples of items include photos that depict multiple people, comments that mention multiple users, events in which multiple users are invited and etc.

Multi-party privacy management is, therefore, of crucial importance for users to appropriately preserve their privacy in Social Media. There is recent evidence that users very often negotiate collaboratively to achieve an agreement on privacy settings for co-owned information in Social Media. In particular, users are known to be generally open to accommodate other users' preferences, and they are willing to make some concessions to reach an agreement depending on the specific situation. However, current Social Media privacy controls solve this kind of situations by only applying the sharing preferences ofthe party that uploads the item, so users are forced to negotiate manually using other means such as e-mail, SMSs, phone calls, etc.e.g., Alice and Bob may exchange some e-mails to discuss whether or not they actually share their photo with Charlie. The problem with this is that negotiating manually all the conflicts that appear in the everyday life may be time-consuming because of the high number of possible shared items and the high number of possible accessors (or targets) to be considered by users.

Computational mechanisms that can automate the ne-gotiation process have been identified as one of the biggest gaps in privacy managemen in social media.The main challenge is to propose solutions that can be accepted most of the time by all the users involved in an item. so that users are forced to negotiate manually as little as possible, thus minimising the burden on the user to resolve multi-party privacy conflicts. Very recent related literature proposed mechanisms to resolve multi-party privacy conflicts in social media. Some of them need too much human intervention during the conflict resolution process, by requiring users to solve the conflicts manually or close to manually. Other approaches to resolve multi-party privacy conflicts are more automated but they only consider one fixed way of aggregating user's privacy preferences without considering how users would actually achieve compromise and the concessions they might be willing to make to achieve it depending on the specific situation. Only considers more than one way of aggregating users' privacy preferences, but the user that uploads the item chooses the aggregation method to be applied, which becomes a unilateral decision without considering the preferences of the others.

In this project,we propose the first computational mechanism for social media. Here given the individual privacy preferences of each user involved in an item, is able to find and resolve conflicts by applying a different conflict resolution method based on the concessions users' may be willing to make in different situations.

2 : Software Requirements Specifications

2.1 Introduction

2.1 Purpose

The main purpose of our project is to reduce the mannual process of the resolving multi-party conflicts in the social media by providing a web interface for users.We are taking the individual privacy preferences of each and every user for resolve the conflict.

2.1.2 Document Convenctions

IEEE format. Size of main headings and sub headings are 26,20 respectively. The font of the remaining description is Times New Roman.

2.1.3 Intended Audience and Reading Suggestions

In this project the SRS document is developed by us. Here we have project manager , developer, coder, tester, and users of the product who are reading our document.

2.1.4 Product Scope

The main problem with the current system is the lack of multi-party management support. Currently in social media when users in group,if any user want to share the co-owned item to perticular person(s) it may not sure item is recieved by him/her or not. In this proposed system can able to find the conflicts by using privacy preferences and can resolve it.

2.2 Overall Description

2.2.1 Product perspective

The software being designed as a web based application mainly aimed for social media.This can able to share items(images,messages).While sharing the images if any conflict occurs then it will resolve and it will share as per consern.

2.2.2 Product Functions

- Users and Admin can use the web interface to login and perform the desired task. Before login they should be registered.
- Users can view and update their profiles.
- Users can send friend request and confirm the request.
- Users will share the messages and images to others.
- Users can view the friends list.
- Admin can able delete the user accounts.
- Admin can check the conflict and if conflict occurs he/she can resolve.
- User and admin log out.

2.2.3 User Classes and Characteristics

Users: They can share the images or messages by registering.

Admin: He/She can resolve the conflicts between the users and can change the database,enter, remove and update the data.

2.2.4 Operating Environment

- This web application can be deployed on window machine with NetBeans IDE 8.2 and Navicat Premium Softwares.

- Standard pc, 4GB RAM, 500GB hard disk.

- This application can be accessed by user through a machine having any web browser with html javascript support. The client devices must preferably have browsers like IE9 or above, Mozilla firefox or chrome installed in their OS.

2.2.5 Design and Implementation Constraints

- User should install any one of the above specified web browsers.
- Admin need to maintain database with security and need to follow constraints.

2.2.6 User Documentation

IEEE format.

2.2.7 Assumptions and Dependencies

None.

2.3 External Interface Requirements

2.3.1 User Interfaces

The user interface of the application will be user-friendly and good looking, intuitive and easy to use, implementing the well standards. We provide better GUI(Graphical User Interface) for easy access.

2.3.2 Hardware Interfaces

The system should have these hardware requirements.

a. Processor (Intel i3).
b. RAM (2 GB)
c. Hard Disk (500 GB).

2.3.3 Software Interfaces

Operating Systems: Windows etc.
Database : Mysql
Front End: Java,Jsp,HTML.

2.3.4 Communications Interfaces

Web browser, network servers communications protocols , HTTP Protocols.

2.4 System Features

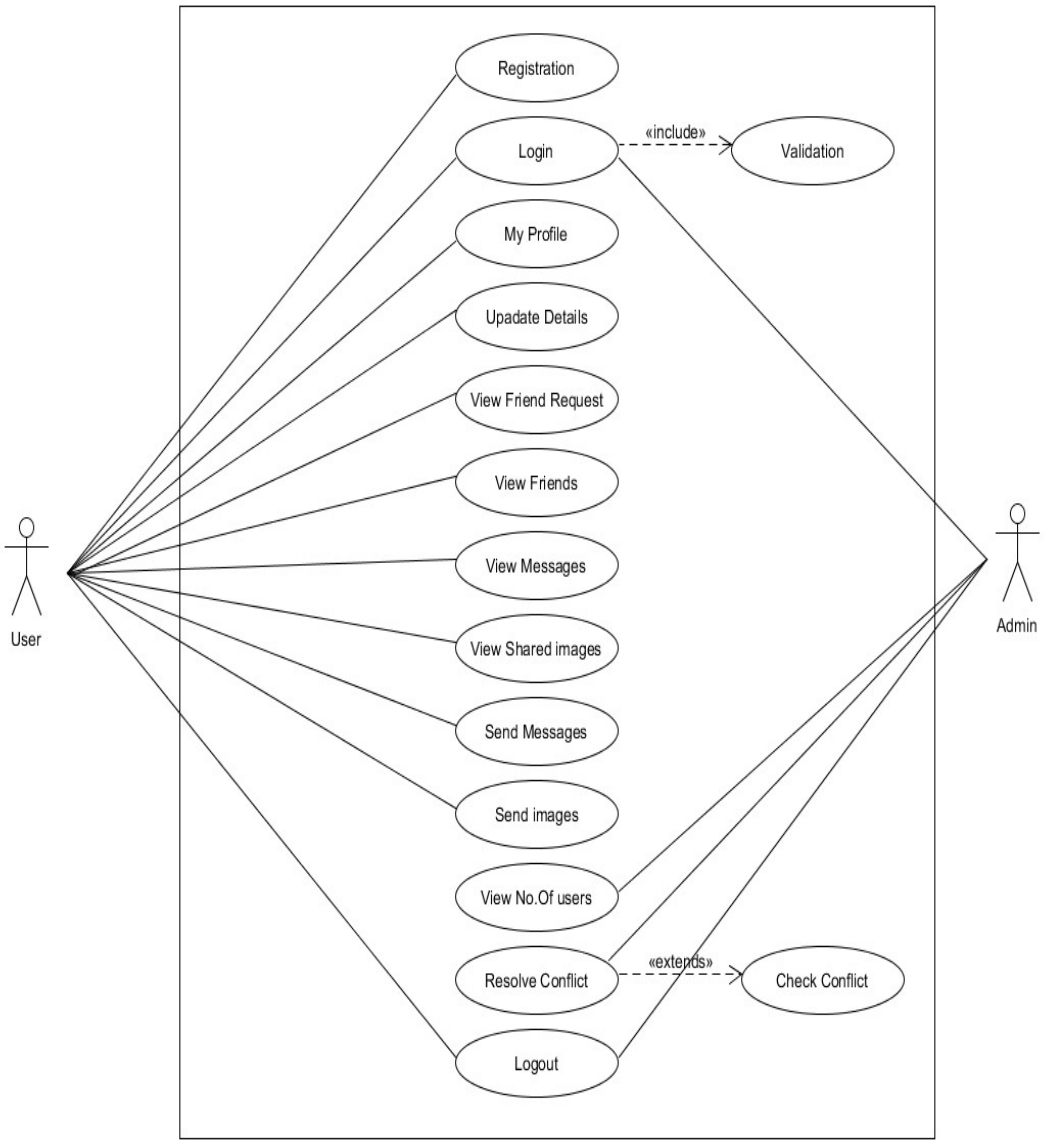


fig 1: Use case Diagram

2.4.1 System Feature : Register

Name	Register
Participating actors	Users
Goals	Link will be open to register and to get registed.
Triggers	Request to open the registration link
Pre-condition	The user should be logged with authorized access
Post-condition	Registration page will be displayed and user can register.
Basic flow	1. Click the sign Up button 2. Display the register page 3. User can register
Alternate flow	Server busy wait for some time.
Exception	Enter valid details.
Quality	After clicking the register button, registration will be done with in 10 sec

2.4.2 System Feature : Sign In

Name	Sign In
Participating actor	Users,Admin
Goals	To get into this site and to make information secure
Triggers	Click the login button
Pre-condition	Register before login and access our link
Post-condition	Displays home page
Basic flow	1. Enter username, password. 2. If user enter correct information then open the main page
Alternate flow	1. Error message as invalid login 2. Server busy wait for some time
Exception	Error message as invalid login
Quality	Main page will be displayed less than 5 sec.

2.4.3 System Feature : View Profile

Name	View Profile
Participating actors	users
Goals	Displays user profile.
Triggers	Clicks to My Profile of the user
Pre-condition	User home page
Post-condition	My profile page will be displayed.
Basic flow	1. Click the My profile. 2. Display the profile page 3. User can edit profile.
Alternate flow	Server busy wait for some time.
Quality	Action will be done with in 5 sec.

2.4.4 System Feature : Update Details

Name	Update Details
Participating actors	users
Goals	Displays user details.
Triggers	Clicks to Update details of the user.
Pre-condition	User want to update the details,user has to register before.
Post-condition	User can update the details
Basic flow	1. Click the update 2. Display the update details of the user.
Alternate flow	Server busy wait for some time.
Quality	It will be done with in 5 sec.

2.4.5 System Feature : Send Request

Name	Send Friend Request
Participating actors	Users
Goal	Users want to make friend.
Triggers	Click on send Request.
Pre-condition	It shows all the details of unfriends list.
Post-condition	User will send the friend request.
Basic flow	1. Click the Send Request button 2.It done automatically send the request.
Alternate flow	Server busy wait for some time.
Exception	If any users are not available No users existed.
Quality	After clicking the sen Button,Request will send with in 5 sec.

2.4.6 System Feature : Confirm Request

Name	Confirm the Request
Participating actors	Users
Goal	Users want to make friend.
Triggers	Click on waiting button.
Pre-condition	It shows all the details of already who are sent friend request.
Post-condition	User confirm the friend request.
Basic flow	1. Click the waiting button. 2.It will automatically confirm the request.
Alternate flow	Server busy wait for some time.
Exception	If any users are not available No users existed.
Quality	This action done with in below 5 sec.

2.4.7 System Feature : View Friends list and shared images.

Name	View Friends list and shared images.
Participating actors	Users
Goal	Displays friends list and shared images.
Triggers	Click on friends to view the friends list Click on shared images to view the images.
Pre-condition	The user should be logged with authorized access.
Post-condition	Displays friends list and shared images.
Basic flow	1. Click the friends. 2.It display the friends list. 3.Click the shared images. 4.It display the shared images.
Alternate flow	Server busy wait for some time.
Exception	If any users are not available Error message is diaplayed as No users existed. If any images not avaliable then it will display error message.
Quality	Action perform with in 5 sec.

2.4.8 System Feature :Send messages and images

Name	Send messages and images.
Participating actors	Users
Goal	Users can send messages and images.
Triggers	Click on send/share
Pre-condition	It display two catagories messages and images.
Post-condition	If user click messages or images it displays all the details and image or message will send.
Basic flow	1. Click the Message or image if click the message the follwing will be done 2.page can displayed,user can write the message. 3.If users click the image then page is diaplayed where user can share image.
Alternate flow	Server busy wait for some time.
Exception	If Image size is more than 50 MB error message display.
Quality	Action done in below 5 sec.

2.4.9 System Feature :Select Group and Rate.

Name	Send group and rate
Participating actors	Users
Goal	Users can select any one the the group and give rate .
Triggers	Click on Move To for Select Group. Click on rate for giving rate.
Pre-condition	The user should be logged with authorized access.
Post-condition	Users give the group and rate to user friends.
Basic flow	1. Click the move to option for user can keep another user in any group. 2.Click the Rate for giving the user can rate for the another user.
Alternate flow	Server busy wait for some time.
Quality	Action done in below 5 sec.

2.4.10 System Feature : View Conflict

Name	Conflict
Participating actors	users
Goals	Displays conflict users.
Triggers	Click on conflict.
Pre-condition	The user should be logged with authorized access.
Post-condition	Users can view the conflict members list.
Basic flow	1. Click the conflict. 2. Display the list of conflicted users.
Alternate flow	Server busy wait for some time.
Quality	It will be done with in 5 sec.

2.4.11 System Feature :Delete User Accounts

Name	DeleteThe admin should be logged.
Participating actors	Admin
Goal	Delete the registered user accounts.
Triggers	Click on delete button.
Pre-condition	The admin should be logged.
Post-condition	Page can be displayed and admin can delete the accounts.
Basic flow	Click the Delete button.
Alternate flow	Server busy wait for some time.
Exception	Some error occurred - try again.
Quality	Information will be deleted within 6 sec.

2.4.12 System Feature :Check the conflict

Name	Check conflict
Participating actors	Admin
Goal	Check the conflict between any two users.
Triggers	Click on view conflict
Pre-condition	The admin should be logged.Before checking conflict admin have to choose any two users.
Post-condition	Page will be displayed.
Basic flow	1.Click on view conflict. Page displays with conflicted users.
Alternate flow	Server busy wait for some time.
Exception	Some error occurred - try again.
Quality	Action perform within 5 sec.

2.4.13 System Feature :Resolve the conflict

Name	Resolve conflict
Participating actors	Admin
Goal	Resolve the conflict.
Triggers	Click on resolve.
Pre-condition	The admin should be logged and Admin should be check the conflict.
Post-condition	Page will be displayed
Basic flow	1.Click on view resolve.
Alternate flow	Server busy wait for some time.
Exception	Some error occurred - try again.
Quality	Action perform within 5 sec.

2.4.14 System Feature :Log out.

Name	Log out.
Participating actors	User,Admin
Goal	Actors will leaves from this site.
Triggers	When actor click the Logout button.
Pre-condition	User must logged in before Logout.
Post-condition	Displays login page.
Basic flow	1.click the logout button 2.display the home/login page
Alternate flow	Server busy wait for some time.
Quality	Action perform within 5 sec.

2.5 Non Functional Requirements

- The system shall allow simultaneous use by several users , without data corruption and data interruption
- The system would exhibit high performance to users because it would be well optimized.
- The system will reduce the use of manual process of resolving conflicts by admin.

2.5.1 Safety Requirements

The database may get crashed and corrupted at any certain point time because of virus or operating system failure. Hence, it is required to take the database backup.

2.5.2 Security Requirements

Admin can be able to Create, delete the records as required.
We are developing secured database for users and create a user-friendly environments.

2.5.3 Software Quality Attribute

Security : It will be secure by using user name and password.

Portability : It will be platform independent.

Maintainability : The data will be maintained long period of time.

Availability : It can be accessed through the internet and also through the off-line.

2.6 Other Requirements

Hardware constraints: The system requires a database in order to store persistent data. The database should have backup capabilities.

Software constraints: The development of the system will be constrained by the availability of required software such as web servers, database and development tools.

3 .Detailed Design Document

3.1 Introduction

3.1.1 Objectives

The main objective of is to develop software is resolve the multi-party conflicts in Social Media and maintain privacy. This document precisely describes the each of the
essential requirements of the software. It also defines and describes the operations, interfaces, performance, design constraints of the system.

3.1.2 Scope

The Resolving multi-party Conflicts would store the details of the all the users profile, friends list, messages, images. The main scope of this this project is if any conflicts occures in social media it will resolve and reduce the mannual process.

3.2 Architectural and Component-Level Design

Describe the overall system architecture and component partitioning in general terms.

3.2.1 Architecture diagram

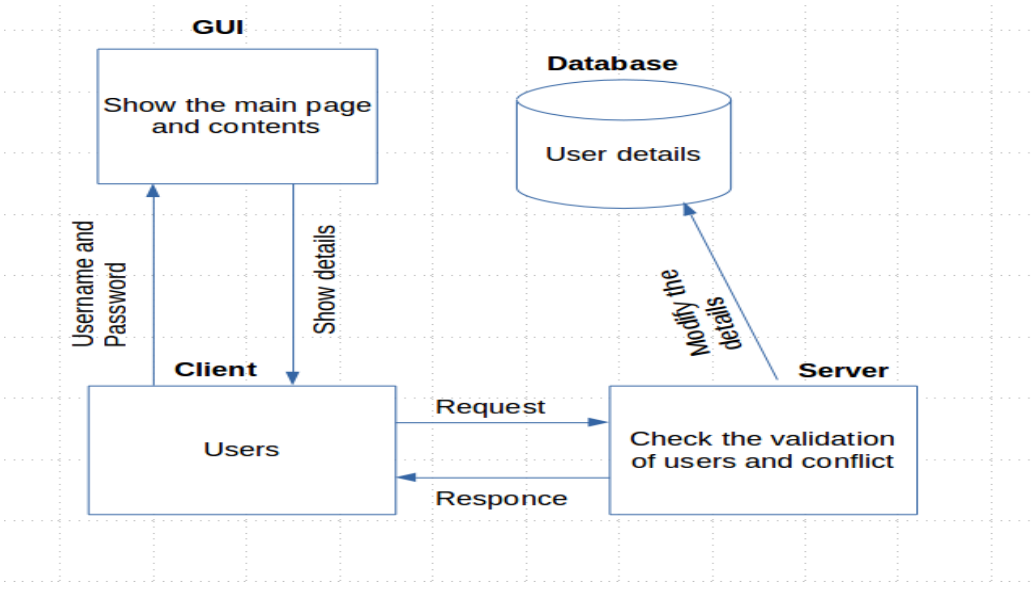


fig 2: Architecture diagram

3.3. Description of Components:

3.3.1 Data flow Diagrams

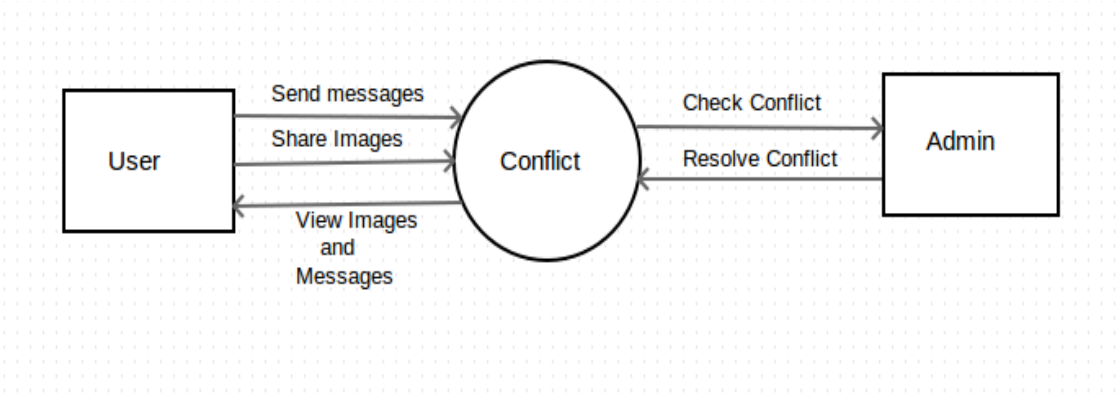


fig 3: Level-0- DFD

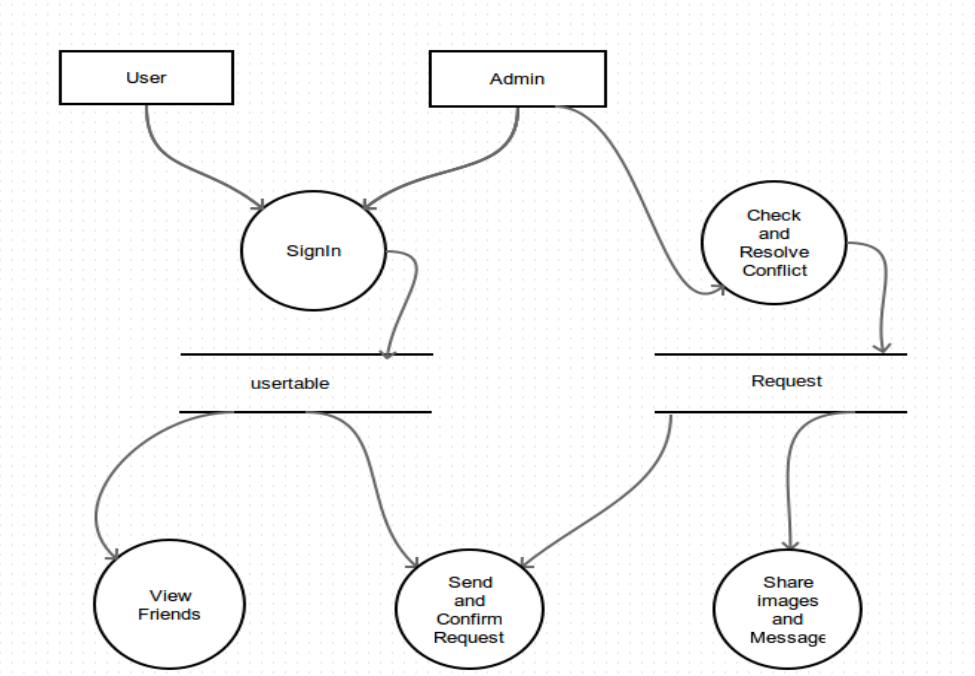


fig 3: Level-1- DFD

3.3.2 Sequence Diagrams

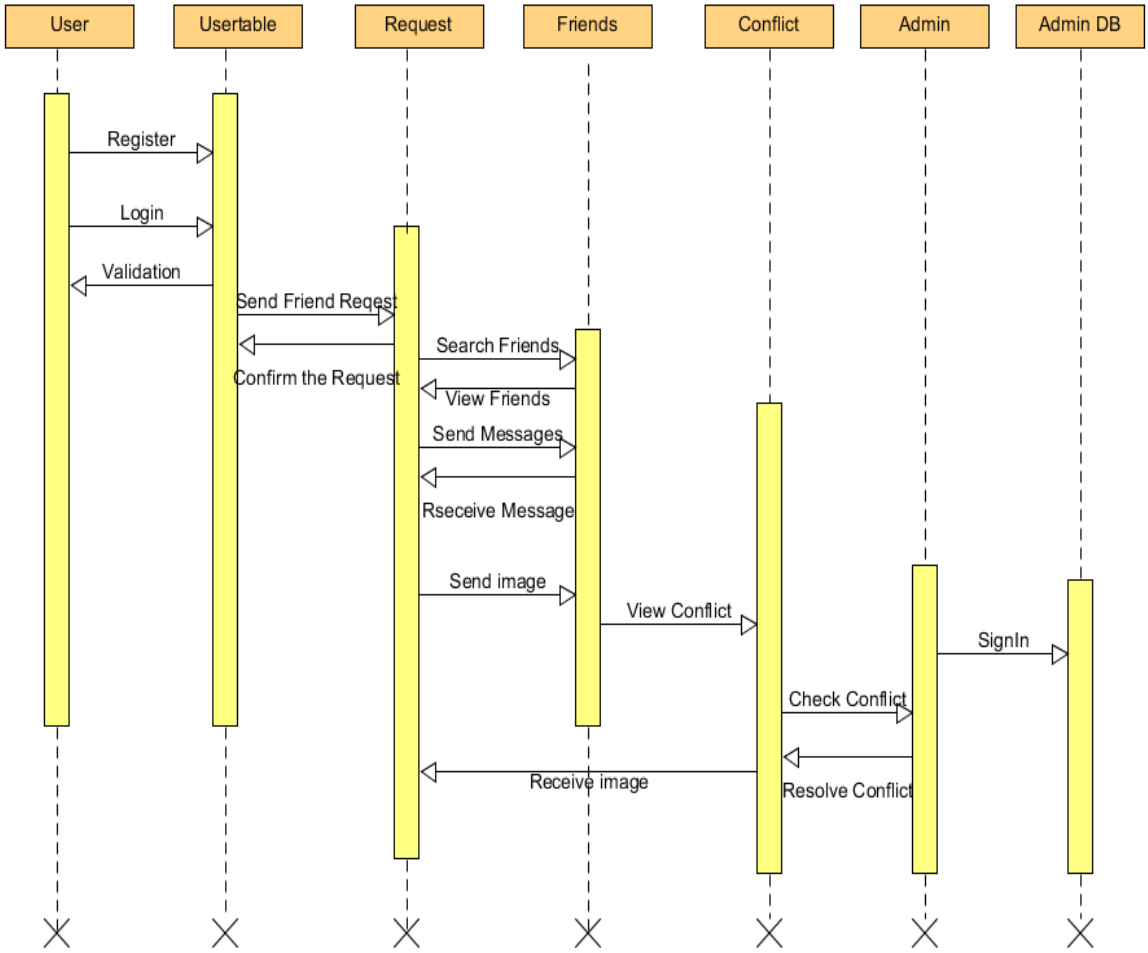


fig 4: Sequence diagram

3.3.3 Class Diagram

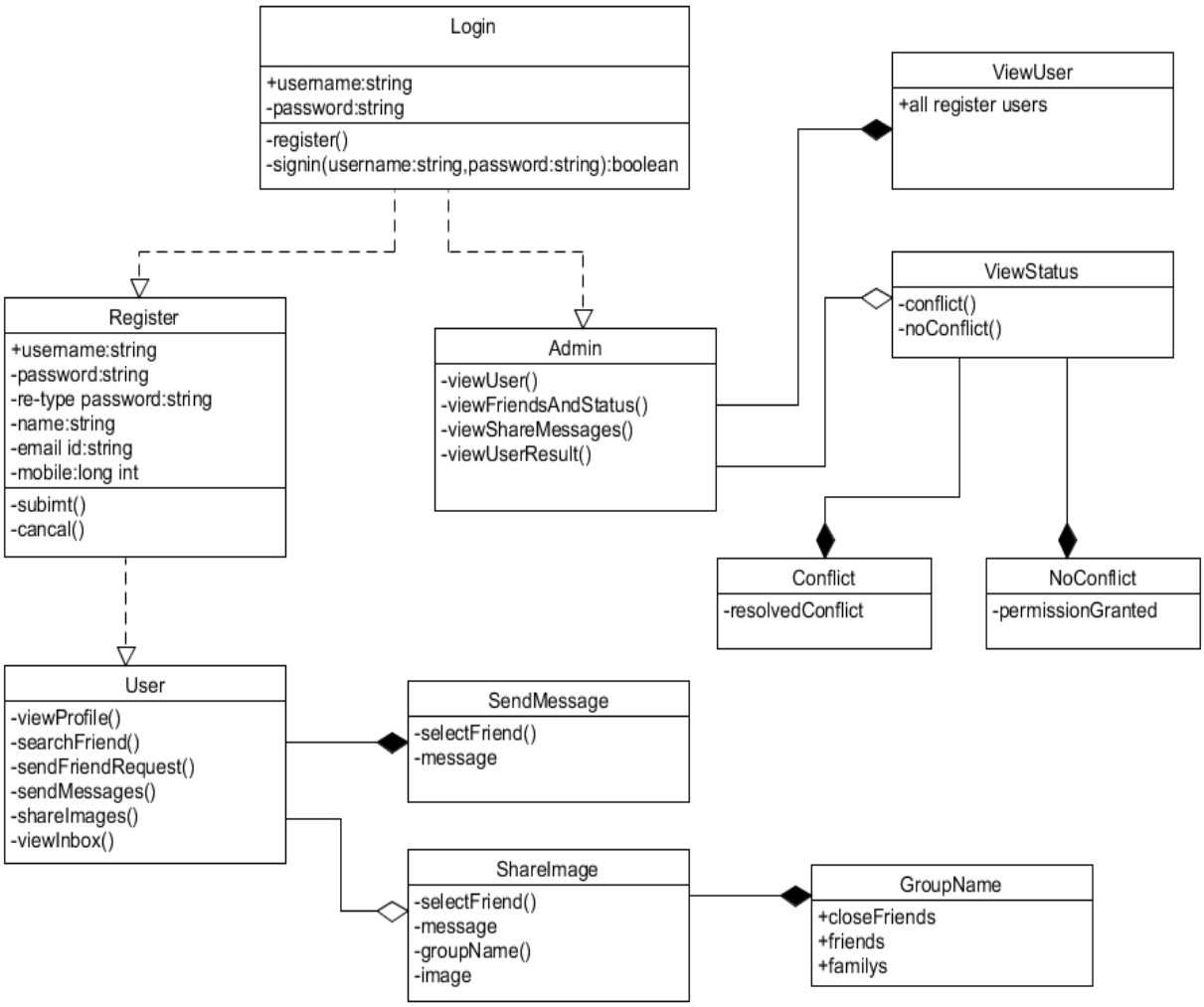


fig 5: Class diagram

3.4 Class #1 User

1.1 Login page

Precondition : Register before login and access our link.

Postcondition : Displays home page

Algorithm :

if uname!=null && password!=null

select * from usertable where username=uname;

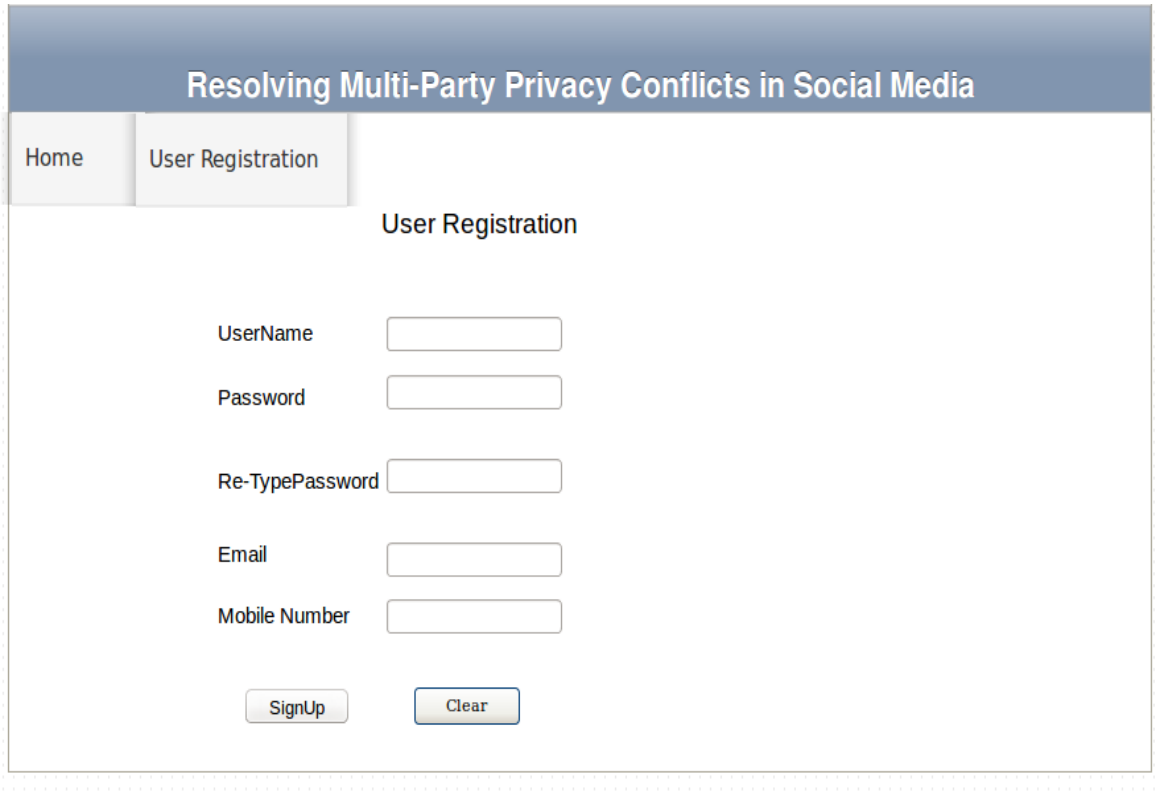
if uname==username && pwd==password

print logged in successfully

else
please enter correct details

Error Handling : If user enters invalid account details the error message will display.

1.2 Rigister Page



Precondition : The user should be logged with authorized access.

Postcondition : Registration page will be displayed and user can register.

Algorithm :

if username!=null && pwd!=null && email!=null

insertintoAdmin(username,password,retype_password,name,
email,mobile_no)
values(username,password,retype_password,name,email,
mobile_no);

print successfully registered.

else

print registration failed. Try again.

Error Handling : If user enters invalid account details the error message will be display.

1.3 View Profile

Precondition : User home page.

Postcondition : My profilepage will be displayed.

Algorithm :

if is_userloggedin(username)==true

print select * from usertable where username=”##”

else

please sign in first.

Error Handling : Some error occurred try again.

1.4 Update Details

Precondition : User home page.

Postcondition : User can update the details.

Algorithm :

if is_userloggedin(username) == true

print enter information

string a=read input

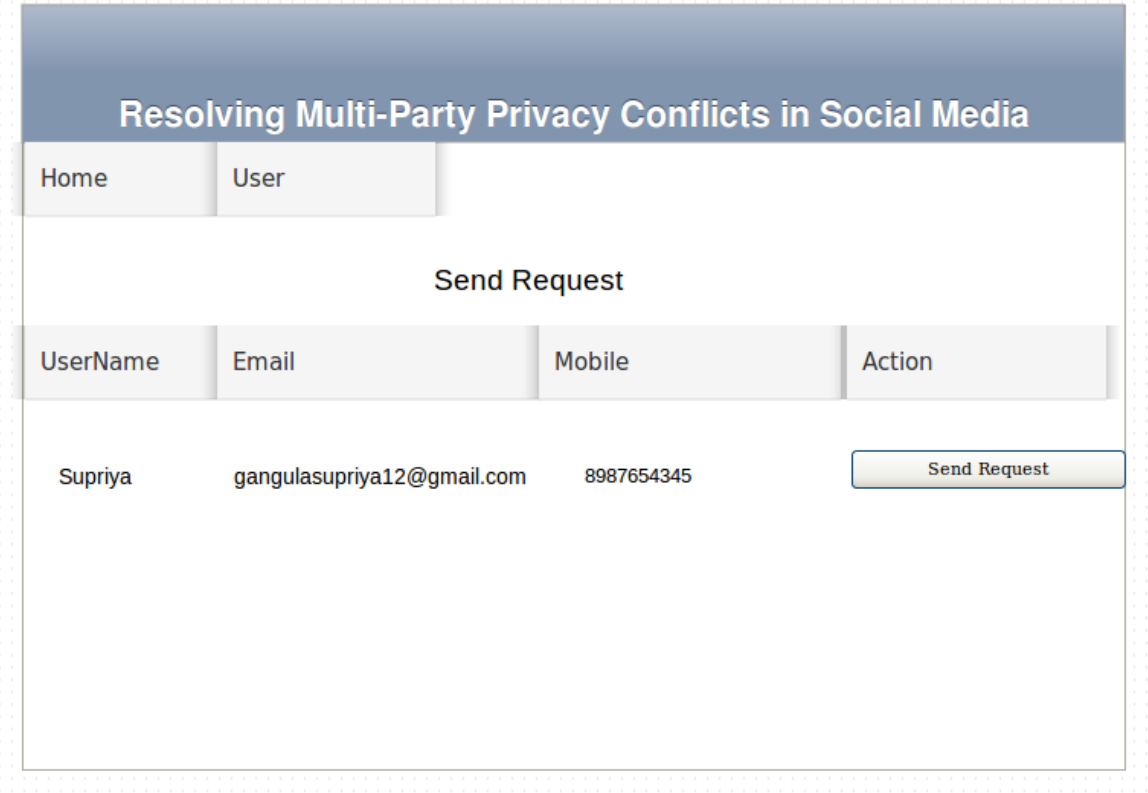
update table usertable set info=a where uname=”##”

else

please sign in first

Error Handling : so error occurred try again.

1.5 : Send Request



Precondition : It showsall the details of unfriends list.

Postcondition : User will send the friend request.

Algorithm :

if is_userloggedin(username) ==true

select * from usertable where username==uname

then

send request

else

please sign in.

Error Handling : so error occurred try again.

1.6 Confirm Request

Precondition : It showsall the details of already who are sent friend request.

Postcondition : User confirmthe friend request.

Algorithm :

if is_userloggedin(username) == true;

select * from request where to_user==username;

print confirm the request

else

please sign in.

Error Handling : so error occurred try again.

1.7 View Friends list.

Precondition : It showsall the details of unfriends list.

Postcondition : User will send the friend request.

Algorithm :

if is_userloggedin(username) == true

select * from request where status =’C’

print view friends list

else

please sign in first

Error Handling : so error occurred try again.

1.8 Send messages and images

Precondition : It display two catagories messages and images.

Postcondition : If user click messages or images it displays all the details and image or message will send.

Algorithm :

if is_userloggedin(username) == true

select * from requests where status =’C’

print send messages and images

else

please sign in first.

Error Handling : so error occurred try again.

1.9 Select Group

1.9 Give Rate

Precondition : The user should be logged with authorized access.

Postcondition : Users give the group and rate to user friends.

Algorithm :

a) Select Group

if is_userloggedin(username) == true

select * from requests where status =’C’

update requests set category=’##’ where slno=’##’

print

select group.

Error Handling : so error occurred try again.

b**) Give rate**

if is_userloggedin(username) == true

select rating from request where touser =’##’ and fromuser=’##’

update requests set score,conflict,rating where slno=”##”

print

give rating.

Else

please sign in.

1.10 View Conflict

Precondition : The user should be logged with authorized access.

Postcondition : Users can view the conflict members list.

Algorithm :

if is_adminloggedin(username) == true

select touser from requests where confilct='C'

print

view conflict condidates list

else

please sign in.

Error Handling : so error occurred try again.

3.4 Class #2 Admin

2.1 Sign In

Precondition : Register before login and access our link.

Postcondition : Displays home page

Algorithm :

if uname!=null && password!=null

select * from usertable where username=uname;

if uname==username && pwd==password

print logged in successfully

else
please enter correct details

Error Handling : If user enters invalid account details the error message will display.

2.2 Check the conflict

Precondition : The admin should be logged.Before checking conflict admin have to choose any two users.

Postcondition : Page will be displayed.

Algorithm :

if is_adminloggedin(username) == true

select touser,fromuser from requests where confilct='C'

check conflict

else

please sign in.

Error Handling : so error occurred try again.

2.3 Resolve the conflict

Precondition : The admin should be logged and Admin should be check the conflict.

Postcondition : Page will be displayed.

Algorithm :

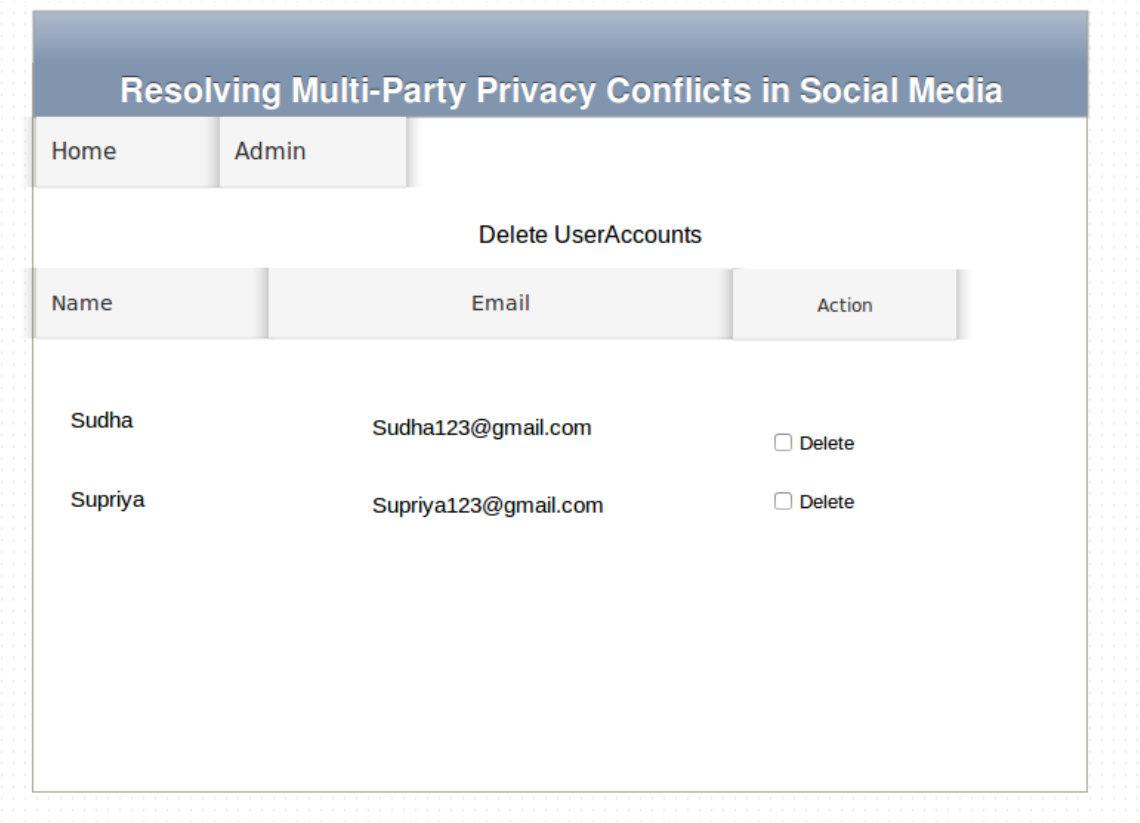
if is_adminloggedin(username) == true

select touser,fromuser from requests where confilct='C'

update requests set conflict,rating where touser="##" and
fromuser="##";
else
please sign in first

Error Handling : so error occurred try again.

2.4 Delete User Accounts



Precondition : The admin should be logged.

Postcondition : Page can be displayed and admin can delete the accounts.

Algorithm :

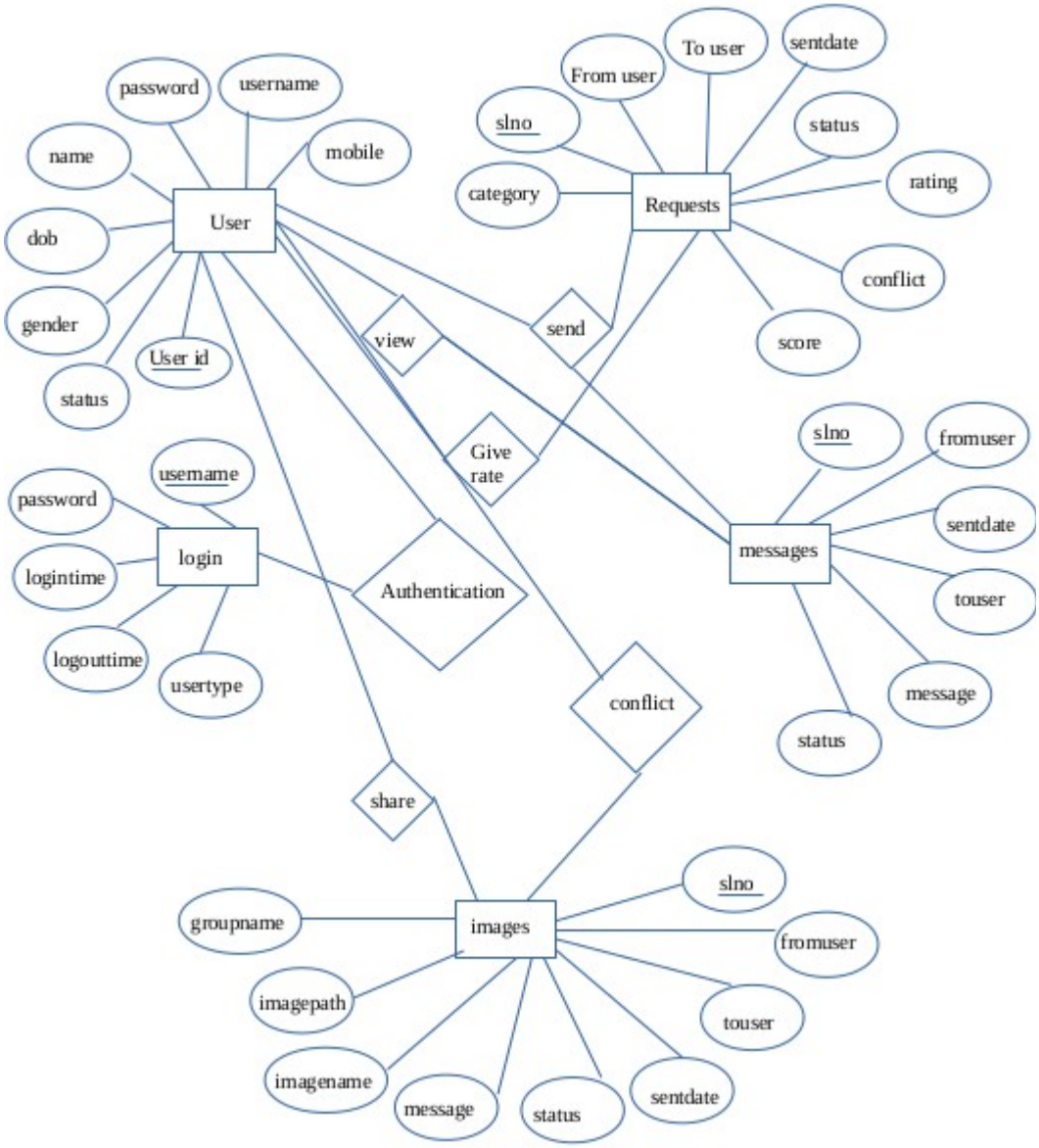
```
if is_adminloggedin(username) == true
        select username from usertable;
                del(user)
else
        please sign in first
```

Error Handling : so error occurred try again.

3.5 Data Architecture

Data collected from users are stored in the database in form of tables with which we manipulate corresponding records.

3.5.1 Entity-Relation Diagram



Data Dictionary:

Table User:

Name	Type	Length	Decimals	Allow Null	
userid	int	11	0	☐	1
username	varchar	11	0	☑	
password	varchar	11	0	☑	
name	varchar	50	0	☑	
email	varchar	20	0	☑	
mobile	varchar	25	0	☑	
dob	varchar	50	0	☑	
gender	varchar	11	0	☑	
occupation	varchar	50	0	☑	
address	varchar	1000	0	☑	
pincode	varchar	20	0	☑	
city	varchar	20	0	☑	
imageLocation	varchar	100	0	☑	
imageSize	int	11	0	☑	
imageName	varchar	100	0	☑	
imageType	varchar	50	0	☑	
Status	varchar	20	0	☑	

Table Name: requests

Name	Type	Length	Decimals	Allow Null	
slno	int	11	0	☐	1
fromuser	varchar	20	0	☑	
touser	varchar	20	0	☑	
sentdate	datetime	0	0	☑	
status	char	1	0	☑	
category	char	2	0	☑	
rating	char	1	0	☑	
conflict	char	1	0	☑	
score	char	1	0	☑	

Table Name: signIn

Name	Type	Length	Decimals	Allow Null	
id	int	11	0	☐	1
username	varchar	11	0	☑	
timein	varchar	20	0	☑	
timeout	varchar	20	0	☑	

Table Name: messages

Name	Type	Length	Decimals	Allow Null	
slno	int	11	0	☐	1
fromuser	varchar	20	0	☑	
touser	varchar	20	0	☑	
sentdate	datetime	0	0	☑	
status	char	1	0	☑	
message	varchar	200	0	☑	

Table Name: login

Name	Type	Length	Decimals	Allow Null	
username	varchar	20	0	☐	1
password	varchar	20	0	☑	
usertype	varchar	6	0	☑	
logintime	varchar	20	0	☑	
logouttime	varchar	20	0	☑	

Table Name: Images

Name	Type	Length	Decimals	Allow Null	
slno	int	11	0	☐	1
fromuser	varchar	20	0	☑	
touser	varchar	20	0	☑	
sentdate	datetime	0	0	☑	
status	char	1	0	☑	
message	varchar	200	0	☑	
imagename	varchar	200	0	☑	
imagepath	varchar	200	0	☑	
groupname	varchar	100	0	☑	

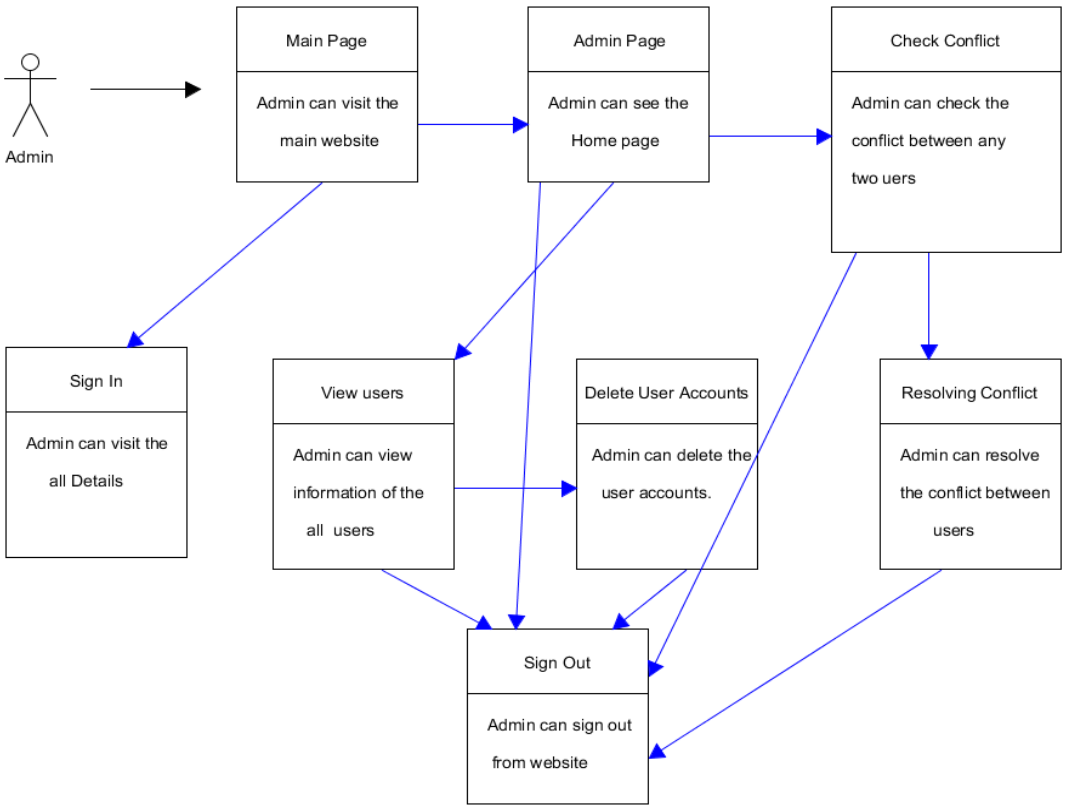
4 . Implementation and Screen shots

4.1 Introduction

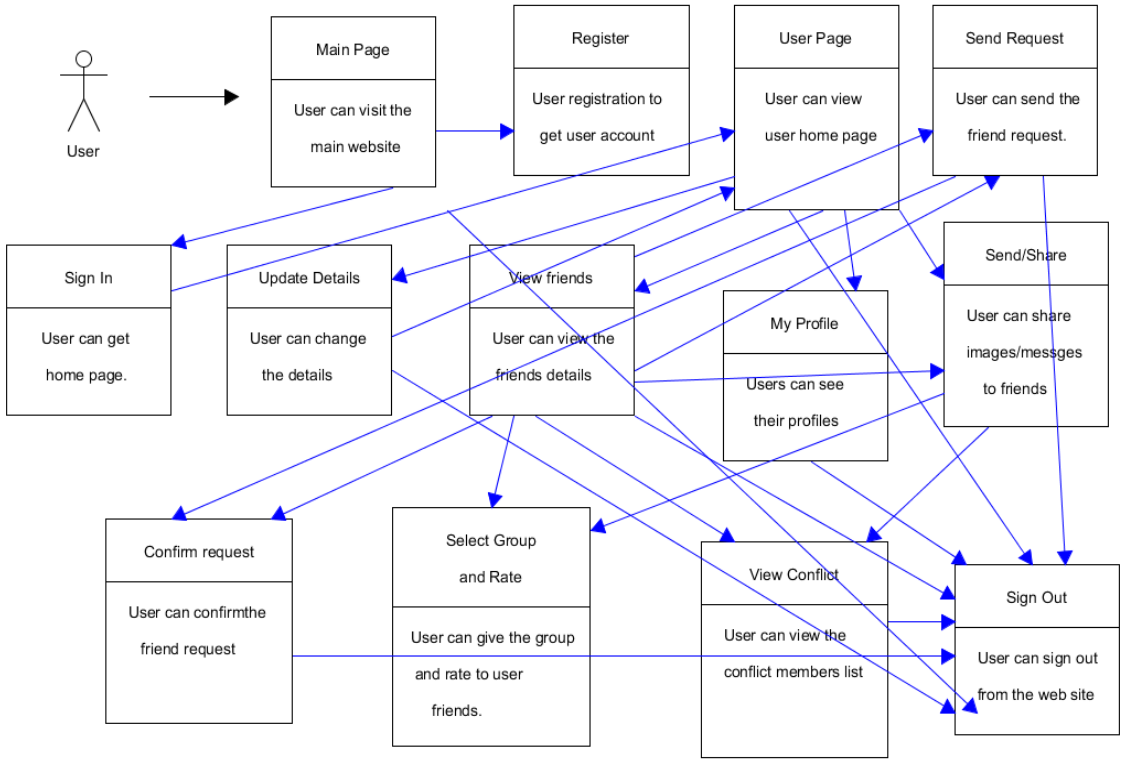
This project implemented with using JAVA, JSP ,HTML and MY SQL. It will be capable of running on standard internet web browsers, although, the project is designed primarily around mozilla Firefox and chrome.It will be useful for who are communicate in the Social Media.The main purpose of this project is resolve conflict between users in current system while sharing items.

4.1.1 Flow chart

Flow Chart of Admin



Flow chart of User



4.2 User and Admin Pages / Modules and Screen shots

4.2.1 Main Page

Introduction: The main page of the Resolving Multi-Party Conflicts in Social Media is the entry point for all other pages contained in the website. users can view two categories user and admin.

Details: The main page will be developed in HTML,Java. The page will contain links to the other pages / modules.Users box for User login,Admin box for Admin login.

Database Tables: The main page will not be linked to a database table. The page will simply point to other pages that have database connections.

Error Handling: The page will handle page errors by attempting to reload the module or report that the module is unavailable.

Implementation Home page :

4.2.2 Login

Introduction: Users can Login as a candidate or admin.To get information from this website.

Details: The users need to enter the username and password to login into the website. A prompted box with a form to provide Username , password. The entered details will be checked with the values stored in the database whether the user is valid or not. If the details are valid it will allow the user to visit the website otherwise not allow.

Database Table:

Table Name: User table.

Error Handling: The page should check the entered values be submitted valid or not.

Implementation Login page :

4.2.3 Register

Introduction: Users can register to get account for this Web Site.

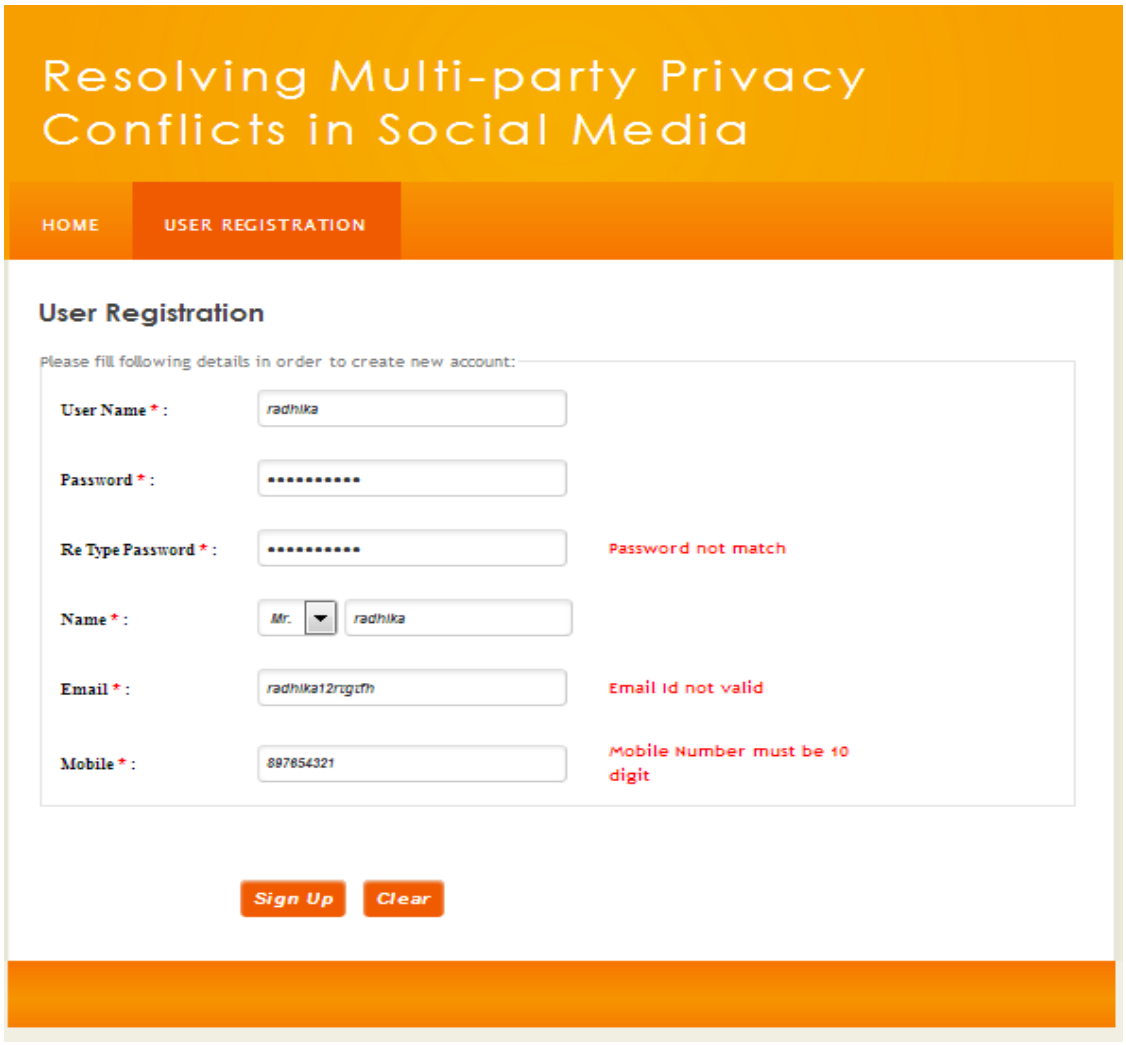
Details: The users need to provide valid information to the website in order to register. A prompted box with a form to provide User , full name, password, email, mobile number and he has to select role(student or faculty or admin). The details will stored into the database. Changes to the user account cannot occur in this website.

Database Table

Table Name: Users

Error Handling: Incomplete information will not be stored into the database. Only information that is complete will allow to be submitted to the website. The page should check the values be submitted and determine if the value is in correct format or not and check the validations.

Implementation Registration page :



4.2.4 Send Request

Introduction: User want to make a friend,when user send the request to perticular user.

Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called View then a page is displayed with send request button.Where user can send request to others.

Database Tables:

Table Name:requests.

Error Handling: The request will send to the particular user. If the sender does not choose the receiver the request will not be send.

Implementation Send Request page :

4.2.5 Confirm Request

Introduction: Users can confirm the requests only when they get request from others.

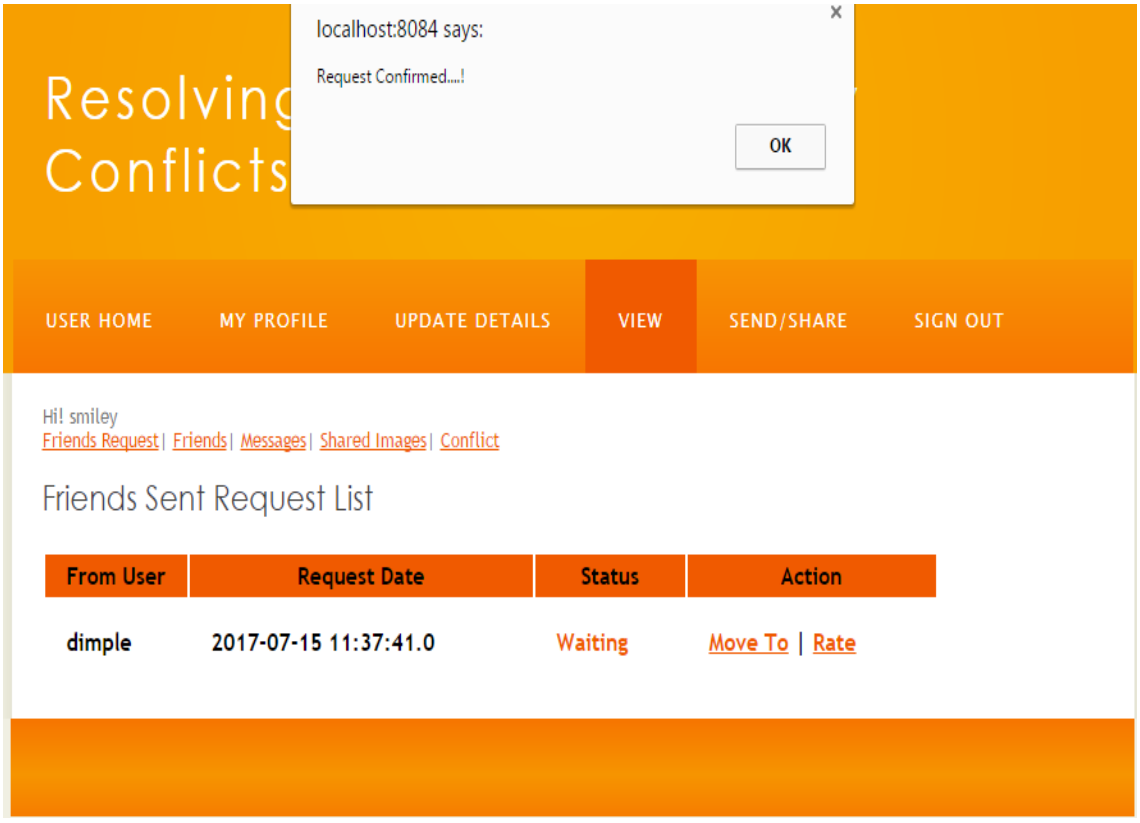
Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called Friends Request then a page is displayed with waiting.If user click the waiting button then the request will confirm.

Database Tables:

Table Name:requests.

Error Handling: The confirm request will send to the perticular user who already sent the request.

Implementation Confirm Request page :



4.2.6 Send messages.

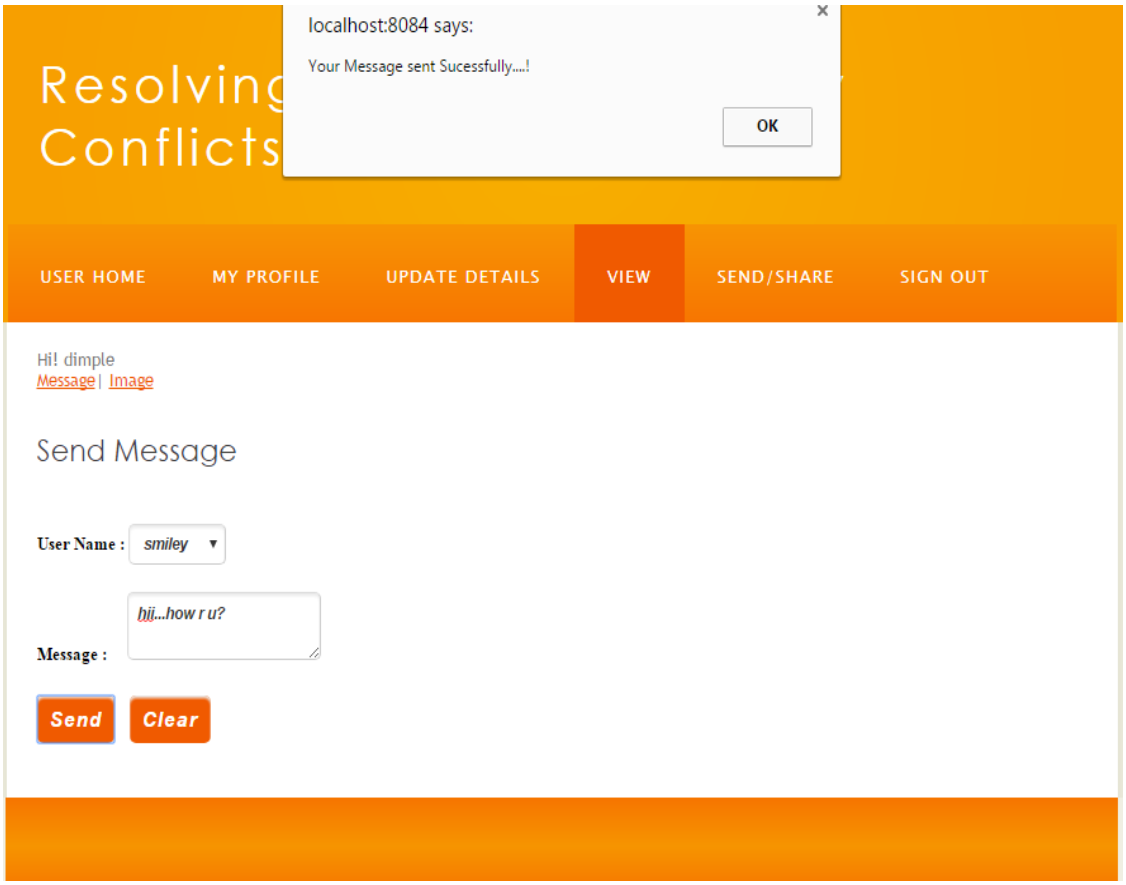
Introduction: Users can send messages to confirmed request candidates who is registered in this website.

Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called Send/Share where choose option called messages.

Database Tables:
Table Names:
messages(for messages).

Error Handling: The message will send to the particular user. If the sender does not choose the receiver the message will not be send.

Implementation Send messages page :



4.2.7 Share Images.

Introduction: Users can send image to receiver who is registered in this website and receiver must don't have conflict.

Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called Send/Share where choose option called messages.

Database Tables:
Table Names:
images(for images).

Error Handling: The image will send to the after resolving the conflict between the sender and receiver.If there was conflict then error message will displayed.

Implementation Share image page :

4.2.8 Select Group.

Introduction: Users can select group for already confirmed candidates.

Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called View where choose option friends.Here confirmed candidates details will displayed and choose Move To option.

Database Tables:
Table Names: requests

Error Handling: It can select group for confirmed friends.

Implementation Select Group page :

4.2.9 Give Rate.

Introduction: Users can give Rate for already confirmed candidates.

Details: To get this information user need to logged into the website after that home page of user will be displayed and choose the option called View where choose option friends.Here confirmed candidates details will displayed and choose Rate option.

Database Tables:
Table Names: requests

Error Handling: It can give rate for confirmed friends.

Implementation Give Rate page :

4.2.10 Check Conflict

Introduction: Admin can check the conflict between already registered users.

Details: To get this information admin need to logged into the website after that home page of admin will be displayed and choose the option called Check Conflict.

Database Tables:
Table Name:requests.

Error Handling: Admin Check the conflict between only already registered candidates.

Implementation Check Conflict page :

4.2.11 Resolve Conflict

Introduction: Admin can Resolve the conflict after checking conflict between already registered users .

Details: To get this information admin need to logged into the website after that home page of admin will be displayed and choose the option Check Conflict.Then click Resolve button.

Database Tables:
Table Name:requests.

Error Handling: Admin checks the conflict after If conflict is present the Admin will resolve the conflict.

Implementation Resolve Conflict page :

5. Software Test Documentation

5.1 Module Name: Registration

5.1.1 Introduction :

System Overview:

Whenever a new user wants to open and see about this website the user should enter his/her details correctly. That registered details will be stored into the database.

Test Approaches:

i) Black box testing: Unit Testing

5.1.2 Test Plan

Functionalities:

The functionality of this module is to get the details from the new users and store them into the database.

Features to be tested: User ID, Re-enter password, Email ID, Mobile Number.

Features not to be tested: Full name.

5.1.3 Test Cases

Case: GUI Tests

Test Case ID: ConflictResolve1

Test Case: valid User ID, Re-enter password, Email ID, Mobile Number

Purpose: To get details from the user and store them into the database

Input Data Requirements: Enter User Details

Expected Results: Details should be stored into database

Pass/Fail: Pass

5.2 Module Name: Sign In

5.2.1 Introduction

System Overview:

Whenever user enters username and password it will match with the database values. If the details are matched user will be successfully logged in.

Test Approaches:

Black box testing: Unit Testing

5.2.2 Test Plan

Functionalities:

The main aim of this module is to get the User ID and password and match with the database values.

Features to be tested: User ID and password

Features not to be tested:none

5.2.3 Test Cases

Case: GUI Tests

Test Case ID: ConflictResolve2

Test Case: valid User ID, invalid password

Purpose: to alert the user to enter correct password.

Input Data Requirements: User ID and password

Output: alert(Wrong username or password).

Pass/Fail: pass.

Test Case ID: ConflictResolve3

Test Case: Invalid User ID, valid password

Description: To alert the user to enter Wrong username..
.
Input Data Requirements: User ID and password.
Output: alert(Wrong username or password).

Pass/Fail: pass.

Test Case ID: Conflict Resolve4

Test Case: Invalid User ID , Invalid password

Description: to alert the user to enter proper username and password

Input Data Requirements: User ID and password.

Output: alert(Wrong username or password)

Pass/Fail:pass

Test Case ID: Conflict resolve5

Test Case: valid ID , valid password

Description: values are store into the database.

Input Data Requirements: User ID and password.

Expected Results: Successful login.

Pass/Fail: Pass

5.3 Module Name: View list for admin

5.3.1 Introduction

System Overview: It helps admin to view users.

Test Approaches:

Black box testing: Unit Testing

5.3.2 Test Plan

Functionalities: functionality of this feature allows the admin to see All the users.

Features to be tested: none

Features not to be tested: none

5.3.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict Resolve6

Test Case: none

Purpose: The purpose of this module is to see all the users.

Input Data Requirements: click on view users.

Output: Admin can see all the users.

Pass/Fail: pass

5.4 Module Name: View list for user

5.4.1 Introduction

System Overview: It helps user to view un friend list

Test Approaches:

i) Black box testing: Unit Testing

5.4.2 Test Plan

Functionalities: functionality of this feature allows the user to see
All the un friends.

Features to be tested: none

Features not to be tested: none

5.4.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict Resolve7

Test Case: none

Purpose: The purpose of this module is to see all the un friends list.

Input Data Requirements: click on view button..

Output: user can see all the un friends..

Pass/Fail: pass

5.5 Module Name: Check Conflict

5.5.1 Introduction:

System Overview: This helps to check the rating between two users.

Test Approaches:

i) Black box testing: Unit Testing

5.5.2 Test Plan

Functionalities: the functionality of this feature is to check the rating between two users.

Features to be tested: none

Features not to be tested: none

5.5.3 Test Cases

Case: GUI Test

Test Case ID: Conflict Resolve8

Test Case: none

Purpose: The purpose of this module is to check the conflict between two users.

Input Data Requirements: click on check button.

Output: checked rating between the two users.

Pass/Fail: pass

5.6 Module Name: Conflict Resolve

5.6.1 Introduction:

System Overview: This helps to resolving the conflict between two users.

Test Approaches:

ii) Black box testing: Unit Testing

5.6.2 Test Plan

Functionalities: the functionality of this feature is to conflict resolved.

Features to be tested: none

Features not to be tested: none

5.6.3 Test Cases

Case: GUI Test

Test Case ID: Conflict Resolve9

Test Case: none

Purpose: The purpose of this module is to resolving the conflict between two users.

Input Data Requirements: click on resolve.

Output: conflict resolved.

Pass/Fail: pass

5.7 Module Name: Send Friend Request

5.7.1 Introduction

System Overview:

It helps the user to send a friend request to the another user which are in the database.

Test Approaches:

i) Black box testing: Unit Testing

5.7.2 Test Plan

Functionalities: functionality of this feature is to send a request to another user.

Features to be tested: none.

Features not to be tested: none

5.7.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict Resolve10

Test Case: none.

Purpose: The purpose of this module is to send a request to the user.

Input Data Requirements: click on send request button.

Output: Sends the request

Pass/Fail: pass

5.8 Module Name: Confirm Request

5.8.1 Introduction

System Overview:

It helps to user confirm the friend request.

Test Approaches:

i) Black box testing: Unit Testing

5.8.2 Test Plan

Functionalities: functionality of this feature is confirm the request .

Features to be tested: none.

Features not to be tested: none

5.8.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict Resolve11

Test Case: none.

Purpose: The purpose of this module is confirm the request which we get the request from the another user.

Input Data Requirements: click on waiting button.

Output: confirm the request.

Pass/Fail: pass

5.9 Module Name: Update

5.9.1 Introduction

System Overview:

It helps to update the information of the user.

Test Approaches:

i) Black box testing: Unit Testing

5.9.2 Test Plan

Functionalities:

The functionality of this feature is to upadate information of the user.

Features to be tested: none

Features not to be tested: none

5.9.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict resolve12

Test Case: none

Purpose: It will update the information that is stored in the database.

Input Data Requirements: adding or deleting the information.

Output: update the information in the database.

Pass/Fail: pass.

5.10 Module Name: Delete

5.10.1 Introduction

System Overview:

The users will be deleted in the database.

Test Approaches:

i) Black box testing: Unit Testing

5.10.2 Test Plan

Functionalities:

The Functionality of this feature is to delete users from the database.

Features to be tested: none

Features not to be tested: none

5.10.3 Test Cases

Case: GUI Tests

Test Case ID: Conflict Resolve13

Test Case: none

Purpose: It will delete the users.

Input Data Requirements: click on delete button

Output: It will delete the users.

Pass/Fail: pass.

5.11 Module Name: Send/Share

5.11.1 Introduction

System Overview:

This helps users to send the messages and images.

Test Approaches:

i) Black box testing: Unit Testing

5.11.2 Test Plan

Functionalities: The functionality of this feature is to send the messages and images.

Features to be tested: none

Features not to be tested: none

5.11.3 Test Cases

Case: GUI Test

Test Case ID: Conflict Resolve14

Test Case: none

Purpose: The purpose of this module is to send the messages and images.

Input Data Requirements: writing messages and select the images.

Output: Send messages and images.

Pass/Fail: pass

5.12 Module Name: Sign Out

5.12.1 Introduction

System Overview:

Whenever a user wants to quit from the webpage of our website he/she can be able to click the sign out button. Then he/she can be return back to the home page.

Test Approaches:

i) Black box testing: Unit Testing

5.12.2 Test Plan

Functionalities: The functionality of this feature is to sign out from a particular web page.

Features to be tested: none

Features not to be tested: none

5.12.3 Test Cases

Case: GUI Tests

Test Case ID: ConflictResolve15

Test Case: none

Purpose: To quit from a particular web page and return back to the home page.

Input Data Requirements: click on sign out button

Output: User will be signed out from the website.

Pass/Fail: pass.

6.Conclusion

We have completed successfully Resolving Multi-party Privacy Conflicts in Social media project with good team work.The main aim of our project is to Detecting and Resolving the Conflicts in Social Media.In our project, mediator firstly inspects the individual privacy policies of all users involved looking for possible conflicts. If conflicts are found, the mediator proposes a solution for each conflict. It has the potential to reduce the amount of manual user interventions to achieve a satisfactory solution for all parties involved in multi-party privacy conflicts.