Design Document

Team 12

Zuoyang Ding

Weichu Hu

Siyu Jiang

Wendi Zhang

Kenny Zheng

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

1. Purpose

Today more and more applications have added location-based feature functions. Also, chatting applications have become a popular way for people to communicate with each and everyone, but able to find someone with similar interest as you can be problematic. We want to develop a chatting application combined with the advantages of location-based feature which could make the chatting experience interesting and efficient. Our intent is to develop an iOS application that utilized the GPS technology and personal preferences to locate the perfect groups and people within the distance. Unlike many chatting applications, our application allows the users to appear anonymously and contains auto-remove feature for personal privacy and security.

1.1 Functional Requirements

Users can create account and edit personal information

a. As a user, I want to be able to create an account through my current Google account.
b. As a user, I want to be able to use my favorite nickname (which can be the same as others and can be changed in the future)
c. As a user, I want to be able to see groups around me.
d. As a user, I want to be able to upload and change my avatar as I want.
e. As a user, I want to be able to modify my personal profile.
f. As a user, I want to be able to see other user's profile information

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

Group chat room related

a. As a user, I want to know when will the group be disbanded.

b. As a user, I want to be able to join a group.

c. As a user, I want to be able to create a group with topic tags I like.

d. As a user, I want to be able to make a temporary group becomes permanent before it expires (when I think the topics and people in group are interesting)

e. As a user, I want to be able to be removed / leave from a group

f. As a user, I want to be able to sort groups accordingly to activeness

g. As a user, I want to know when the group will be disbanded

h. As a user, I want to check history messages in group within 24 hours

i. As a user, I want to be able to sort groups by tags

j. As a user, I want to be able to vote to decide whether a group will become permanent.

k. As a user, I want to be able to send pictures within group chat.

l. As a user, I want to be able to find roommates using this app (if time allows)

Administrator / backstage manage functions

a. As a user, I want to be able to report particular user with inappropriate activities.

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

b. As an administrator, I can delete or ban a user who have inappropriate activities or reported by other users
c. As an administrator, I can delete or disband groups that are used for discussing illegal / inappropriate information

1.2 Non-Functional Requirements

Client Requirements

a. The application would run on Android device developed using java language
b. The application would be able to get access location information of the device by using Android API.
c. The application would be able to adapter multi-devices of different size and resolution.

Server Requirements

a. The application would be able to allow the users to sign up or login by using Google API verify and local database.
b. The application would be able to interact with the server on the Firebase, and the server will send the wrapped data to database which will be built on Firebase.
c. The application would be able to access data from the backend server.

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

Design Requirements

a. The interface of the application would be simple, reasonable and user-friendly.
b. The interface of the application would be appealing and please for the user to see.

Performance Requirements

a. I want the Android application to be responsive to all user requests.
b. I want both the client and server to gracefully handle errors and the failure of other components that they interact with.
c. I want the server design to scale to support the additional load without requiring any change to the API or client code.

Security Requirements

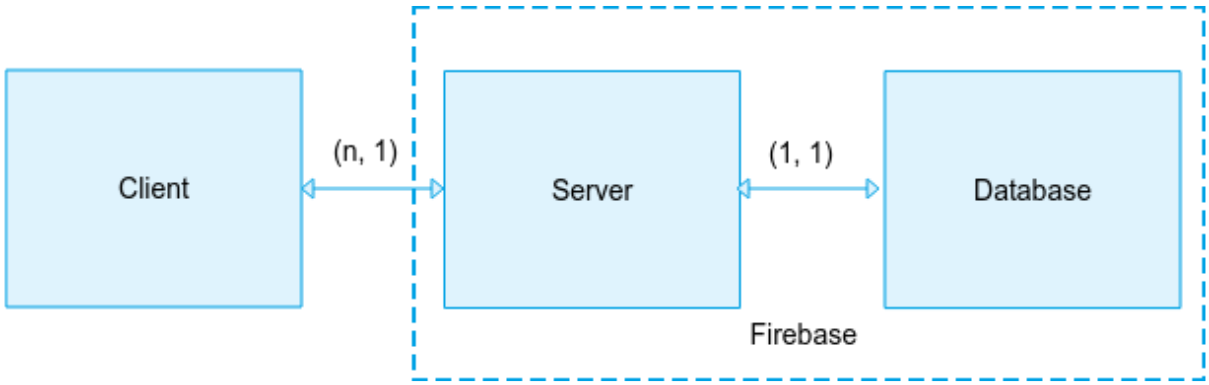
a. The application would make sure the user's personal information are secure.

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

2. Design Outline

High-Level Overview

Our project will follow the client-server model. The server will respond to the client's requests, and the client will parse and render the requested data generated by Google Firebase. The Firebase application platform allows us to work with backend more efficiently without setting up server and database separately. The figure below demonstrates a high-level overview of our system.



1. Android Client

a. The users will interact with other users through the Android client.
b. The client will send login information to the server for it to be verified.

2. API Server

a. The API server will verify login information entered by users and return corresponding actions.
b. The API server will interact with the database to retrieve group information using the Group ID.
c. The API server will send JSON data to the database once the account data sent by client had been verified.

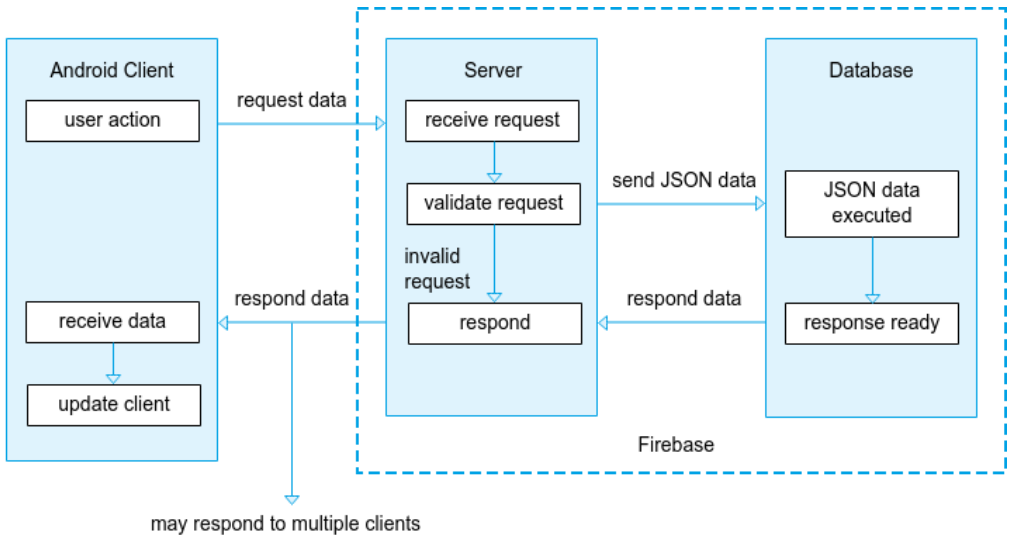
Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

3. Android Client

a. Account information will be stored into database which will be supported by Firebase.
b. Stores all group information such as messages sent as well as personal information such as nicknames.

Flow of Event

The diagram below shows a typical flow of events. The sequence begins with a user opening the Android client application, if the client requires data from the server to update the interface, it will make a request to the server. The server will first validate the request to make sure the requesting user has required permissions for such a request. If the request is valid, the server will send the JSON data to the database, the server will then receive the executed data and generate a response to the client. If the request is invalid, an error response will be sent to the client without making any further connection to the database. After the client received the response, the interface would be updated.



Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

3. Design Issues

Functional Issues

1. How will the users login to our service?

- Users are allowed to login using their Facebook, Google or other social media account
- Users will create an account through our service

Decision: By using the Google API, users are allowed to login using their Google account or create an account if they don't already have a Google account. Users are also allowed to create a personal nickname which will be used as an identifier, the nickname will be stored in our database (Firebase) and the users can manage their nickname through their personal profile page.

2. How will different groups appear on the homepage?

- The groups will be ordered based on GPS location system from groups nearest to the user to groups that are farthest to the user
- The groups will be ordered accordingly to each group's activity rate

Decision: The homepage will contain both permanent groups and temporary groups with number of members in the group, and the groups will be ordered accordingly to each group's activity rate

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

Non Functional Issues

1. How are we going to manage our server and client?

- Use Firebase which allows us to manage the backend such as the server and client
- AWS

Decision: We decided to use Firebase to support our server and client because it provides the programmers the freedom of backend management and not to worry about GET and POST.

2. How many messages will be stored in the database to ensure smooth flow of application?

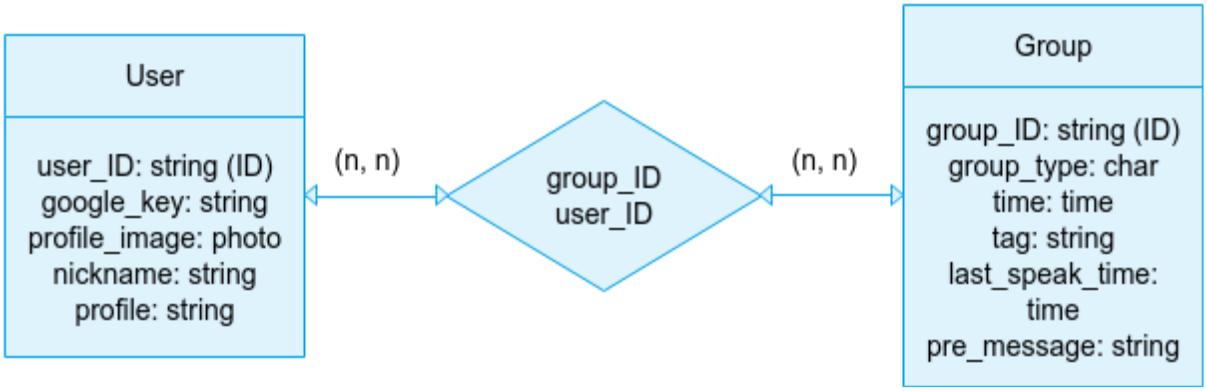
- All messages will be stored inside of database for certain period of time
- Messages will not be stored in database at all
- Only certain amount of messages will be stored

Decision: We decided not to store messages at all due to the fact that the database might overflow or the server might experience busy traffics which might cause the application to run slow. However, chat room will display descriptions of the group instead of the previous conversation.

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

4. Design Details

Database Class design



User Class

user_ID: The identity string of every user, can't be changed after the account was created.

google_key: The identity string received from google account verification.

profile_image: The image file for user's avatar.

nickname: Nickname (string) set by the user.

profile: User's personal profile (string).

location: Reocrd location information

Group Class

group_ID: The identity string for the group, can't be changed after the group was created.

group_type: A tag variable (char) identify whether the group is temporary or permanent.

time: The time that the group was created.

tag: Interest or topic of the group (set by group owner).

last_speak_time: The time for the most recent chat.

Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

pre_message: The welcome message (set by group owner).

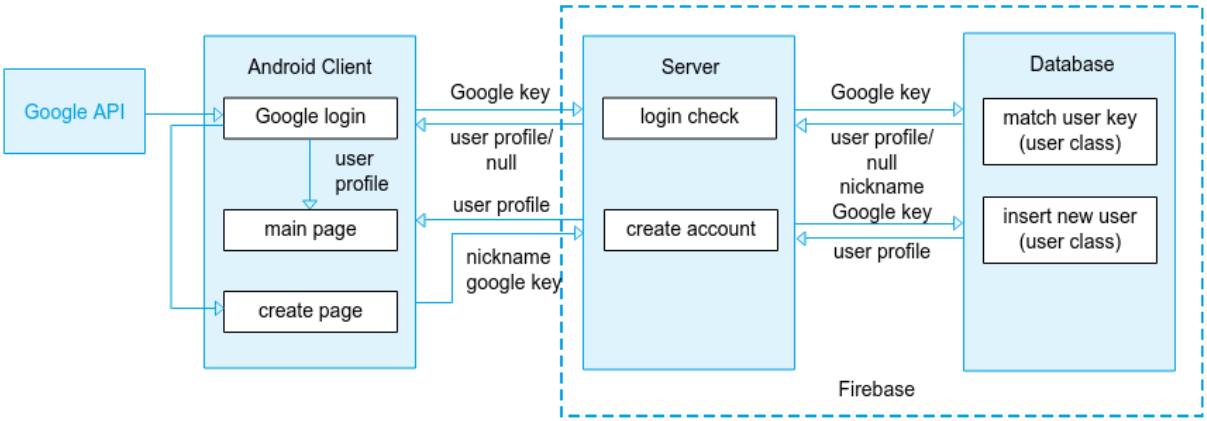
location: Reocrd location information

users: users[user] record connected user ID with this group

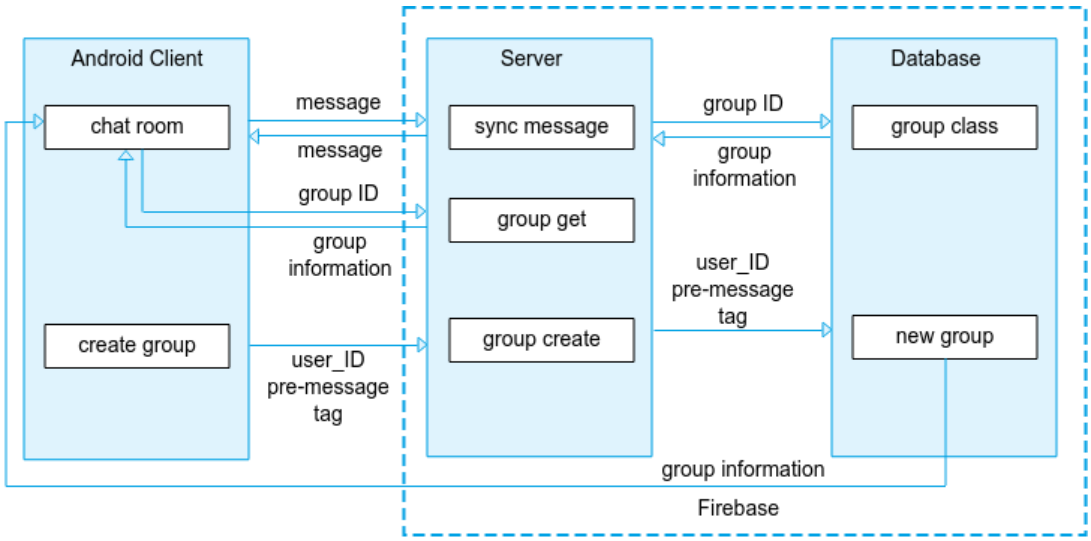
Sequence Diagrams

Below are the sequece diarams for our three major activites

1. **User login and personal profile edit**



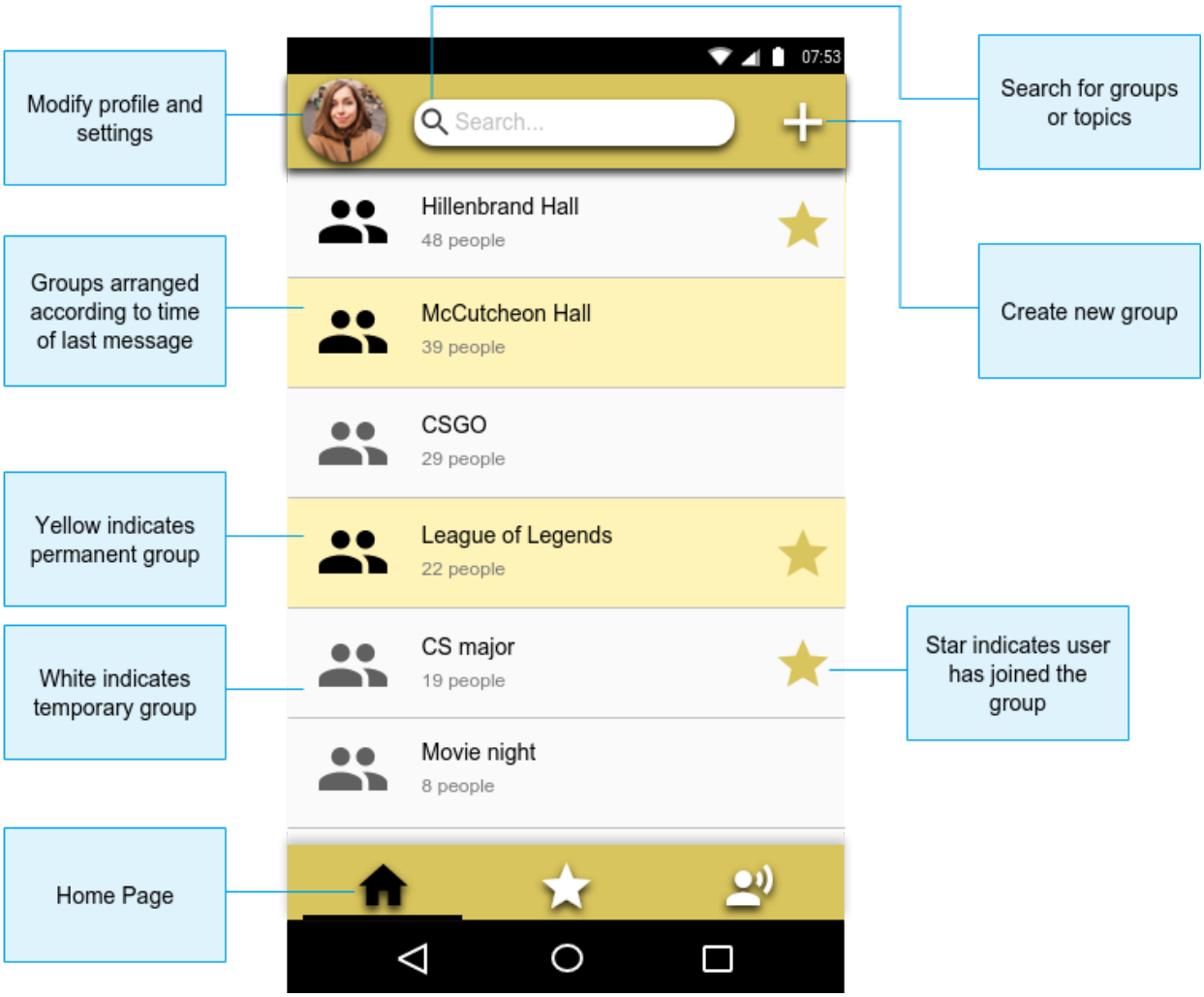
2. **Create chat room**



Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

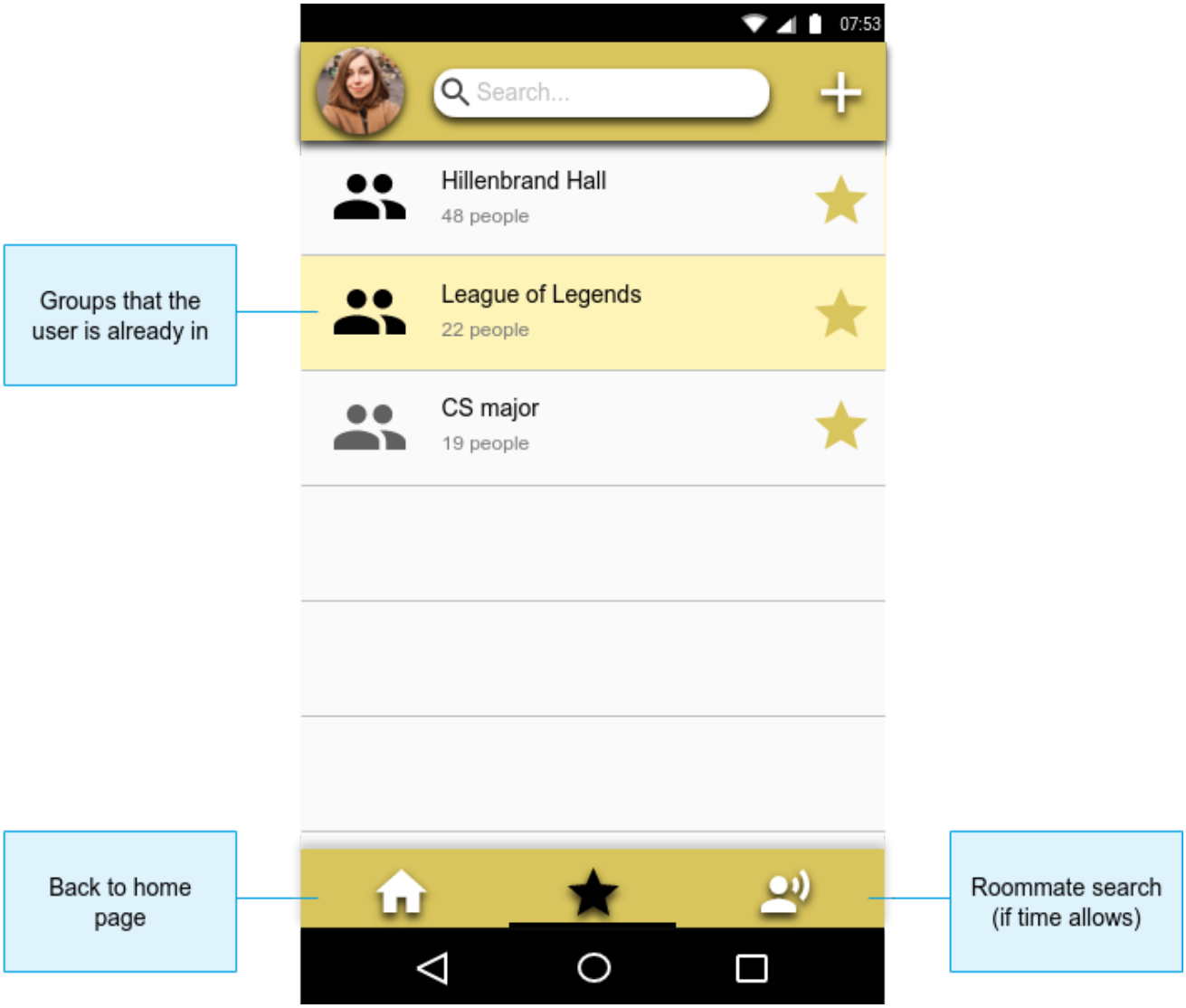
Mockup UI

1. Group chat main page



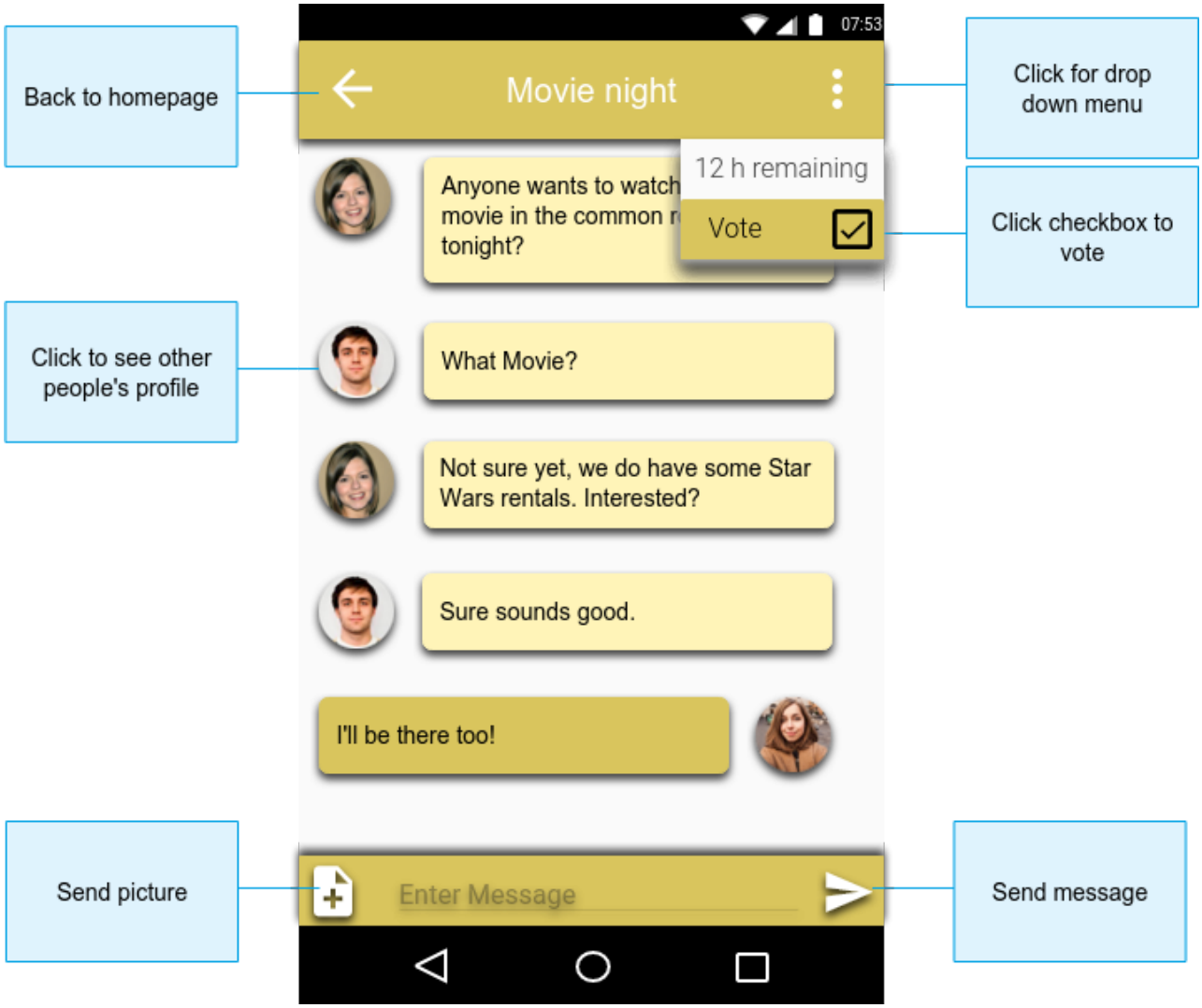
Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

2. User current enroll page



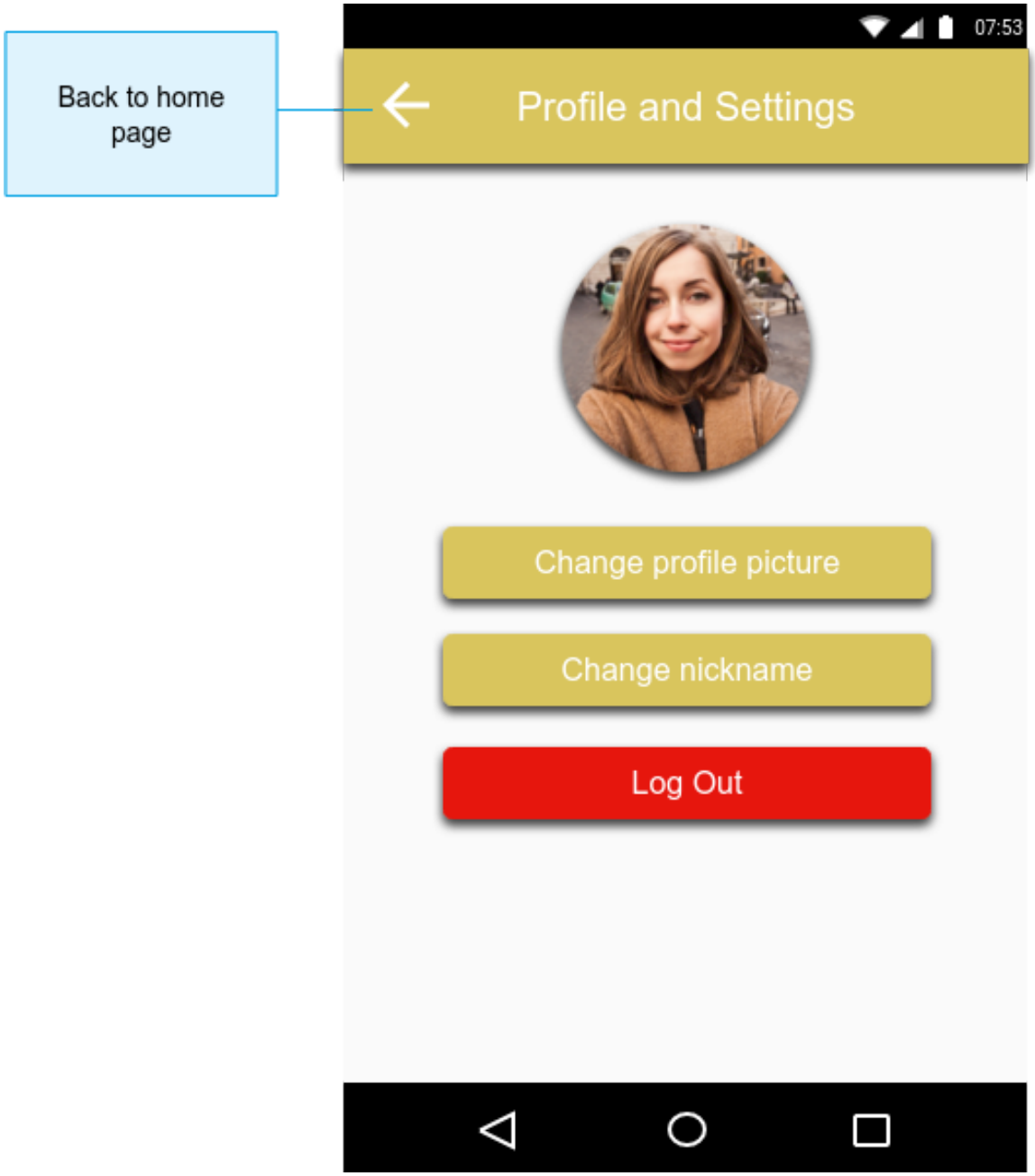
Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

3. Chat room page



Copyright © 2017 Purdue CS307 Team 12. All rights reserved.

4. **Profile page (user or others)**



Copyright © 2017 Purdue CS307 Team 12. All rights reserved.