Bilkent University

Department of Computer Engineering

Senior Design Project

Pengout: A mobile app to find events and friends

Analysis Report

Damla Eda Bıçakcı
Babanazar Gutlygeldiyev
Sine Mete
Pegah Soltani
Hasan Selim Yağcı

Supervisor: Can Alkan
Jury Members: Varol Akman and İbrahim Körpeoğlu
Innovation Expert: Mehmet Çakır

Analysis Report
March 25, 2019

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table Of Contents

1. Introduction

The search for spending free time and getting social is now a more complicated process. Widening cities and crowding places lead to varieties of events taking place around to eliminate according to personal interest and taste. There are several ways that people find the event to go such as asking to a friend, seeing an announcement through social media or using search engines.

Researchers have showed that people have tendency to use social networking platforms to reach what they are looking for. In a survey, 41% of people respond that they are asking questions in type of recommendation, invitation and social connection and topic-wise 36% of respondents ask questions related to entertainment, places, restaurants and current events. The survey also has showed that the motivation of people to use social networking instead of search engines has three main aspect: trust, subjectivity and social connection [1].

However, the motivation of answering questions is not that high and this results in non responded questions. Thus, the idea of a social platform that works without motivation of responding but a motivation of attending to a social activity and help other people find their interested events may address such demand.

This search for attending events has another major factor besides finding it, which is to motivate or demotivate an individual to attend a social/cultural event is having or not having to attend it without a companion. In other words, an event is more enjoyable for individuals to attend if they have a friend to accompany them.

There are several portals and applications for people with social anxiety to communicate with each other. However, they are not specifically intended to encourage people to attend events and activities. Given that social activities have more potential to improve our social skills as well as cultural and educational aspects, a social platform that is specifically designed for those activities is potentially more effective. So this was our motivation for designing a project as such.

In this report, overall analysis of the system will be projected. Existing systems, our proposed system, functional, non-functional and pseudo-functional requirements will be analyzed. The system models of the system will be examined as well. Scenarios will be discussed in order to analyze the interaction situations between users and the system. Object and class model, activity and sequence diagrams will also explained briefly. At the end, navigational path and user interface mockups will be given.

2. Current system

There are many applications in the social network market to help people for their social life. In this section, we examined some of them, which are similar to our social network app.

2.1 MeetUp

MeetUp is a web-based application to socialize with people according to your interests. It has a many social groups like book clubs to nature walking groups that you can participate or you can create your own group and meet up with new friends who share your interests [2].

2.2 Circle

Circle is a mobile application, which is a location-based social network. It uses GPS to find your location and you can easily meet new people in your general area [3].

2.3 SocialRadar

SocialRadar is a location-based application that gives notifications about who is around you, when you go to an event or any social places (restaurant, bar, club etc.). This app can access your social media contacts, and gives some helpful remainders of their details [4].

3. Proposed system

3.1 Overview

PengOut is an application in which encourages users to attend more events and socialize. Many people would not attend an event because they do not have a companion to go with and they see themselves socially incapable of socializing at the event. Our application focuses more on the social aspect of the events. By using PengOut one can find the events near their locations and be notified about them.

According to the user's interests (which will be specified by the user in the very beginning of creating their accounts) and event history, the system will also suggest events. After the event is marked by the user the system will find a companion who is attending the same event. So both users will be able to communicate through the app before they attend the event. This will not only be relaxing for the ones who are not comfortable with socializing in real life, it will also encourage many people to get to know new people while attending events.

In this application, the users will be able to create their own events and set it public, in which all the other users will be able to view, or set it private in which the users will be able to share with their own friends.

3.2 Functional Requirements

- Users must be able to sign up and sign in to our system using their already existing facebook or Google accounts. Their contacts will also be synchronized accordingly. Users will also be able to create their own account in the application.

- Users will able to view the events that are planned around their location. The search system will also support filtered searches with different parameters.

- Users will also be able to create their own events and invite their friends to the event or even share it in the platform.

- After choosing an event to attend, the users (if desired) can be matched with other people that are attending the same event. So they can communicate each other and plan to attend the event together. This matching will be according to the self interests and event history of both users.

- All the users will have a calendar in which they can schedule the events that they want to attend.

- Users will have their own personal account in which they can add each other and chat with each other.

- The program will check the calendar of friendlist of each user and notify them if two or more friends have the same events to attend.

- The application will also try to rate the feedback from the users.

- By keeping the track of the category of the events that each user has attended, the application will collect the data and use it for recommending upcoming relative events.

- By user request, the application will send notifications regarding the new events near their location.

- Users will be asked to specify their areas of interests (e.g. art, music, etc) that will be used for suggesting events in the recommendation process.

- If the user will not specify his/her interests and will not login via a social media account, the events will be shown according to live location or if user does not prefer live location, just according to city.

- The user can select an event to see details such as description, attendees among friends, location, starting time etc.

- Pengout will list events feed in a way that specifies events for the user will be on top.

- The application will derive events from Google Calendar API

3.3 Nonfunctional Requirements

- **Portability:** PengOut will be an android based application. So any device that is powered by android should be able to run this program without any problems

- **Scalability:** PengOut is a social network application. This means that there will be a great amount of data to deal with. So scalability is one of the most important requirements of this application

- **Usability:** PengOut will be as user friendly as possible. So each user should not have difficulties trying to use the app without watching a tutorial or reading the manual

- **Privacy:** Users will have their own accounts in PengOut which means that they will share personal information in their accounts. Especially the users that choose to create account via their Facebook or Google accounts. So security is an important issue and all the data will be secured properly.

- **Accessibility :** People at any age greater than 18 are welcome to use this application and all of them should be able to use the application from their android devices

- **Reliability:** PengOut will accurately show the events around a specific location. Also analyzing the data regarding user's interest to find a companion or get recommendations will be done as accurate as possible.

- **Maintainability:** Customer satisfaction is the main goal of all applications provided to the public. So, our system must be design in such a way that finding the bugs and fixing them would not take long period of time.

- **Portability:** Since we want to be able to reach to a larger number of audience, portability is an important issue that will be considered in each stage of implementation of the project. So any device that is powered by android will be able to install the program and run it without any inconvenience.

3.4 Pseudo requirements

- Pengout will need internet in order to keep the updated information and match people available
- This application will be written for android based devices and will be implemented using android studio
- We will use Firebase for keeping the accounts which will also help us with the security part
- In order to determine the location we will use Google Maps API
- We will also have a server for our application

3.5 System models

3.5.1 Scenarios

Scenario 1
Use Case Name: *SignUp*
Actors: *User*
Entry Condition: *User is on Login Screen*
Exit Condition: *User is on Home Screen*
Main Flow of Events:

1. User clicks to "Sign up with email address".
2. System directs user to Create Account Screen.

3. User enters email address and password.
4. System sends confirmation email.
5. User is confirmed via link.
6. System displays Sign In Screen.
7. User enters email address and password.
8. System displays Complete Profile Screen.
9. User enters location, gender, age, profile photo.
10. Use selects his/her interests among categories of events
11. System directs user to Home Screen.

Scenario 2
Use Case Name: *ContinueWithFacebook*
Actors: *User*
Entry Condition: *User is on Login Screen*
Exit Condition: *User is on Home Screen*
Main Flow of Events:
1. User clicks on "continue with facebook" button.
2. System asks for login to Facebook account.
3. System displays Complete Profile Screen.
4. User press Continue or Skip button.
5. User enters details as location, interests, etc.
6. User press Continue or Skip button.
7. User selects his/her interests among categories of events.
8. System directs User to Home Screen.

Alternative Flow of Events:
1. Facebook authentication is incorrect.
    a. System directs User to Login Screen.
    b. System displays error information.
    c. User choose an option to sign up.
    d. System directs user to Home Screen.
2. User clicks on "continue with facebook" button.
3. System asks for login to Facebook account of User.
4. System checks whether account is logged in before.
5. Steps 3-7 are passed.
6. System directs User to Home Screen.

Scenario 3
Use Case Name: *ContinueWithGoogle*
Actors: *User*
Entry Condition: *User is on Login Screen*
Exit Condition: *User is on Home Screen*
Main Flow of Events:
1. User clicks on "continue with facebook" button.
2. System asks for login to Google account.
3. System displays Complete Profile Screen.
4. User press Continue or Skip button.
5. User enters details as location, interests, etc.
6. User press Continue or Skip button.

7. User selects his/her interests among categories of events.
8. System directs User to Home Screen.

Alternative Flow of Events:

1. Google authentication is incorrect.
    a. System directs User to Login Screen.
    b. System displays error information.
    c. User choose an option to sign up.
    d. System directs user to Home Screen.
2. User clicks on "continue with facebook" button.
3. System asks for login to Google account of User.
4. System checks whether account is logged in before
5. Steps 3-7 are passed.
6. System directs User to Home Screen.

Scenario 4

Use Case Name: *SignIn*
Actors: *User*
Entry Condition: *User is on Login Screen*
Exit Condition: *User is on Home Screen*
Main Flow of Events:

1. User types his/her email address.
2. User types his/her password and clicks "login" button.
3. System directs User to Home Screen.

Alternative Flow of Events:

1. User types email address and password incorrect.
    a. System redirects User to "login screen".
    b. System display a message indicating the incorrect email address/password.
2. User types email address and password.
    a. User clicks "remember me" check box.
    b. System save the email and password information for the next login operation.
    c. System directs User to Home Screen.

Scenario 5

Use Case Name: *ViewEvent*
Actors: *User*
Entry Condition: *User is on Home Screen or Search Screen*
Exit Condition: *User is on Event View Screen*
Main Flow of Events:

1. User clicks on an event on the Home Screen.
2. System directs User to Event View Screen and display particular event information.

Alternative Flow of Events.

1. User is on Search Screen.
2. User clicks on an event.
3. System directs User to Event View Screen and display particular event information.

Scenario 6
Use Case Name: *AttendEvent*
Actors: *User*
Entry Conditions: *User is on Event View Screen or Home Screen*
Exit Conditions: *User is on Event View Screen or Home Screen*
Main Flow of Events:

1. User is on Home Screen.
2. User clicks to "attend" icon on an event section.
3. System adds the event to the calendar of the user and updates Calendar Screen.
4. System adds the event to events of the user and updates Events Screen.

Alternative Flow of Events:

1. User is on Event View Screen and viewing an event.
    a. 2-4 steps are repeated.
2. User clicks to "attend" icon.
    a. System displays "similar events" on the bottom of Event View Screen.

Scenario 7
Use Case Name: *CommentEvent*
Actors: *User*
Entry Conditions: *User is on Event View Screen*
Exit Conditions: *User is on Event View Screen*
Main Flow of Events:

1. User clicks to "comment" icon.
2. System directs the user to Comment Screen.
3. User types his/her comment on event and clicks "send" button.
4. System updates Event View Screen of that particular event with user's new comment.
5. System directs the user to Event View Screen.

Alternative Flow of Events:

1. User clicks to comment icon.
2. System directs the user to Comment Screen.
3. User types a comment and clicks "back" button.
4. System directs the user to Event View Screen without saving the comment.

Scenario 8
Use Case Name: Search Event
Actors: User
Entry Conditions: User is on Home Screen
Exit Conditions: User is on Search Screen
Main Flow of Events:

1. User clicks on "search" icon.
2. System directs the user to Search Screen
3. User types and press "enter" or clicks "search".

4. System displays results that is relevant to the user input.

Alternative Flow of Events:

1. User clicks on "search" icon.
2. System directs the user to Search Screen.
3. User types and press "enter" or clicks "search".
4. System finds no result and display "no result" message to the user.

Scenario 9

Use Case Name: SendMessage
Actors: User 1, User 2
Entry Conditions: User 1 is on Home Screen or User 1 is on View Profile Screen
Exit Conditions: Chat Screen
Main Flow of Events:

1. User 1 clicks to "message" icon on Home Screen.
2. System directs User 1 to Messages Screen.
3. User 1 clicks on search bar at the top.
4. User 1 types "User 2".
5. System directs User 1 to Chat Screen.
6. User 1 types his/her message and clicks "send" icon.
7. System sends the message to User 2 and display the message on User 1 Screen.
8. User 2 clicks on "notification".
9. System directs User 2 to the Chat Screen.
10. System displays the message on the User 2 Chat Screen as well.

Alternative Flow of Events:

1. User 1 is on View Profile Screen and viewing the User 2's profile.
2. User 1 clicks to "message icon".
3. System directs User 1 to Chat Screen.
4. Steps 6 - 10.

Scenario 10

Use Case Name: DirectEvent
Actors: User 1, User 2
Entry Conditions: User 1 is on Event View Screen
Exit Conditions: User 1 is on Chat Screen
Main Flow of Events:

1. User 1 clicks on "direct" icon.
2. System displays list of friends and search option to the User 1.
3. User 1 clicks to "send" button next to the User 2 to direct the event.

Scenario 11

Use Case Name: ViewCalendar
Actors: User
Entry Conditions: User is on Home Screen
Exit Conditions: User is on Calendar Screen
Main Flow of Events:

1. User clicks "calendar" icon on the Home Screen.

2. System displays Calendar Screen.

Scenario 12
Use Case Name: CreateEvent
Actors: User
Entry Conditions: User is on Home Screen
Exit Conditions: User is on Home Screen
Main Flow of Events:
1. User clicks "create event" button.
2. System directs User to Create Event Screen.
3. User enters the name, place, date, description.
4. User clicks option "public".
    a. User choose categories by clicking on them.
    b. User clicks "done" icon.
    c. System creates the event and updates Calendar Screen, Events Screen and other users Home Screen accordingly.
    d. System directs User to Home Screen.
5. User clicks option "private".
    a. User clicks "done" icon.
    b. System creates the event and updates only that Calendar screen and Events screen of the User.
    c. System directs User to Home Screen.

Scenario 13
Use Case Name: *ViewProfile*
Actors: *User*
Entry Condition: *User is on Home Screen*
Exit Condition: *User is on Profile Screen*
Main Flow of Events:
1. User clicks "profile" icon.
2. System directs User to Profile Screen.
3. System displays User's profile picture, name, number of friends, number of past activities, recent activity information etc.

Scenario 14
Use Case Name: *EditProfile*
Actors: *User*
Entry Conditions: *User is on Profile Screen*
Exit Conditions: *User is on Profile Screen*
Main Flow of Events:
1. User clicks "edit profile" button.
2. System displays Edit Profile Screen.
3. User choose to change his/her profile picture.
4. System asks for an image file from User to upload or opens the camera.
5. User uploads an image file.
6. User finishes the editing and clicks on "done" icon.
7. System changes the profile picture accordingly and update Profile Screen.

8. System displays Profile Screen.

Alternative Flow of Events:

1. User clicks "edit profile" button.
2. System displays Edit Profile Screen.
3. User edits some options.
4. User clicks "back" button.
5. System does not update any changes.
6. System displays Profile Screen.

Scenario 15

Use Case Name: *AcceptMatch*
Actors: *User 1, User 2*
Entry Conditions: *User 1 and User 2 has made AttendEvent action on same event*
Exit Conditions: *User 1 is on Chat Screen*
Main Flow of Events:

1. System "notifies" User 1 and User 2 with "You might want to go together. " message.
2. User 1 clicks on notification and clicks on "accept" button.
3. User 2 also does the step 2.
4. System "notifies" User 1 and User 2 that " There is a match." message.
5. User 1 clicks to the "notification".
6. System directs User 1 to Chat Screen with User 2.

Alternative Flow of Events:

1. System "notifies" User 1 and User 2 with "You might want to go together. " message
2. User 1 accepts but User 2 does not.
3. System does not create any notification.
4. System directs User 1 to Home Screen.

Scenario 16

Use Case Name: *ViewPastEvents*
Actors: User
Entry Conditions: User is on Events Screen
Exit Conditions: User is on Past Events Screen
Main Flow of Events:

1. User clicks "Past Events" tab on the Events Screen.
2. System displays the event history of User and a search bar to search within past events.

Scenario 17

Use Case Name: *ViewSettings*
Actors: *User*
Entry Conditions: *User is on Home Screen*
Exit Conditions: *User is on Settings Screen*
Main Flow of Events:

1. User clicks "settings" icon.

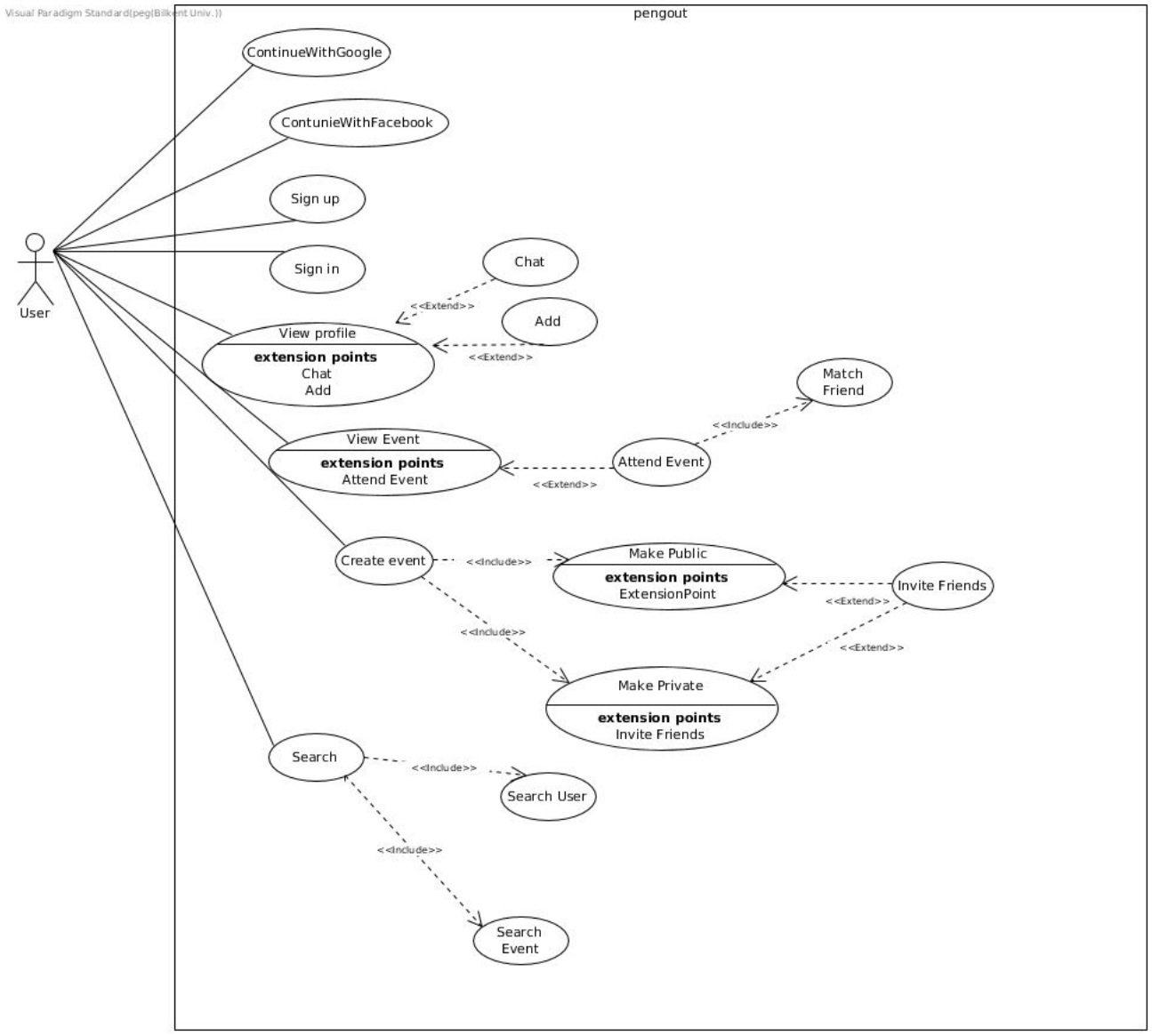
2. System directs User to Settings Screen.
3. System displays several settings such as privacy settings, account settings, application settings, notification settings and etc.

Scenario 18
Use Case Name: *ChangeSettings*
Actors: *User*
Entry Conditions: *User is on Settings Screen*
Exit Conditions: *User is on Settings Screen*
Main Flow of Events:
1. User clicks to "privacy settings" button.
2. System display the current information of the setting.
3. User changes the privacy settings from "public" to "private".
4. System updates User's profile.

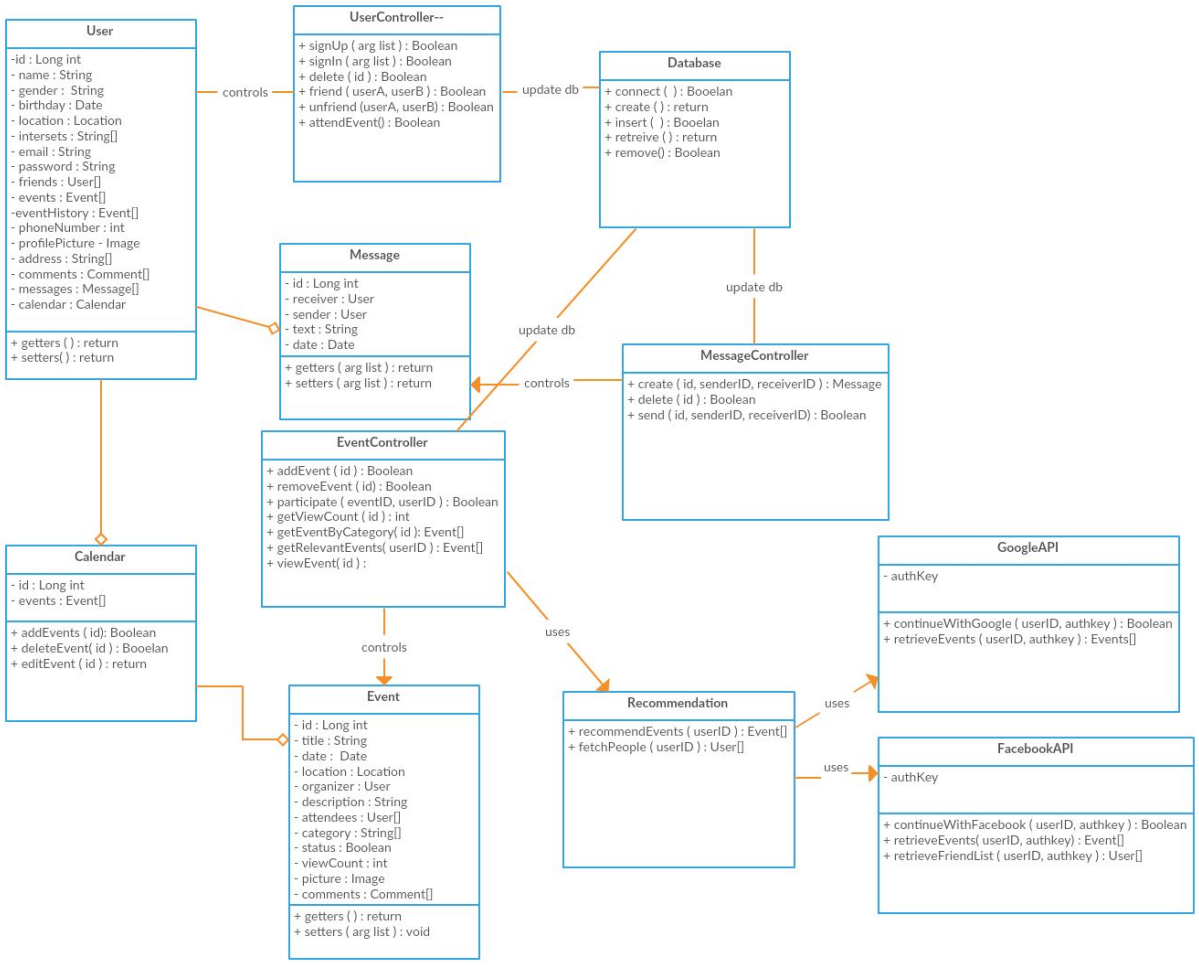
Alternative Flow of Events:
1. User clicks to "application settings".
2. System display the current information of the setting.
3. User choose "default" settings.
4. System updates the settings accordingly.

Scenario 19
Use Case Name: *Logout*
Actors: *User*
Entry Conditions: *User is on Home Screen*
Exit Conditions: *User is on Login Screen*
Main Flow of Events:
1. User clicks on "profile" icon.
2. System directs User to his/her Profile Screen.
3. User clicks to "logout" button.
4. System directs User to Login Screen.

3.5.2 Use case model



3.5.3 Object and class model



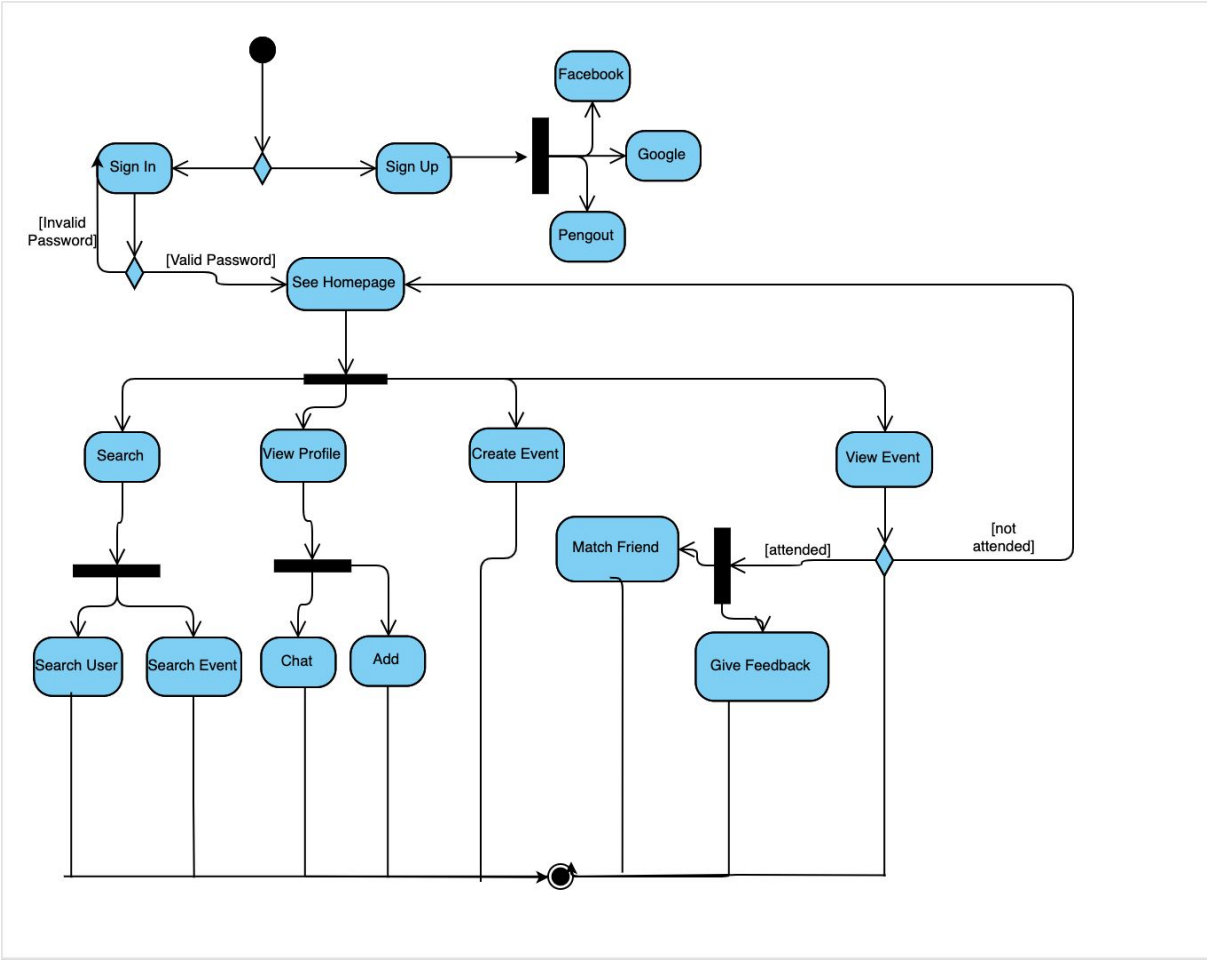
- **User**
  This class keeps information about a user and methods to modify it. For example, whenever a user wants to edit her/his personal information setter methods of these attributes will used. Additionally, this class keeps the track of user's messages and events throughout the system.
- **UserController**
  This class has methods that are executed on one or more users. Moreover, this class handles registrations and logins of users. When registration or deletion is requested, this class will call related functions of other classes e.g. Databases.
- **Event**
  This class holds details of events in the system. Again this class has methods that needs to be executed on each method separately. As an example, setters and getters of attributes are executed if a user wants to change details of his/her personal information.
- **EventController**
  This class has methods that about operations that concern multiple events. For example, getEventByCategory() method returns all events in the specified category. Moreover, this class calls related

functions of other classes, e.g. database class has methods that is executed to keep database consistent.

- **Message**
  This class is for a message. It holds text, sender and receiver of the message. Similarly, it contains setters and getters for these attributes.
- **MessageController**
  This class controls all messages by executing messages such as create(), delete() and send() on each of them. To accomplish these, if needed be, it uses other classes methods.
- **Calendar**
  Each user object contains an instance of this Calendar class. Because, each user has personalized calendar, that includes private and public events for the user. After the user adds/deletes the events s/he wants this calendar renders a calendar with visuals.
- **Recommendation**
  This class is also used by other classes such as EventController. This class is used to recommend events to the user or order them according to their relevance to the user.
- **GoogleAPI**
  GoogleAPI is used to register and login to the system using user's google account. Moreover, in case of permission and requirement, it will be used to get events of the user from his/her Google Calendar and to save events to it. Information about the events, retrieved from Google Calendar, can be used in recommendation algorithm.
- **FacebookAPI**
  This class also will be used to sign up or sign in to the system. Additionally, the user's friend list, events, places, and interests will be used in the recommendation algorithm.
- **Database**
  This is the class that keeps consistent the data of the system. Its methods are regularly executed by other classes.

3.5.4 Dynamic Models

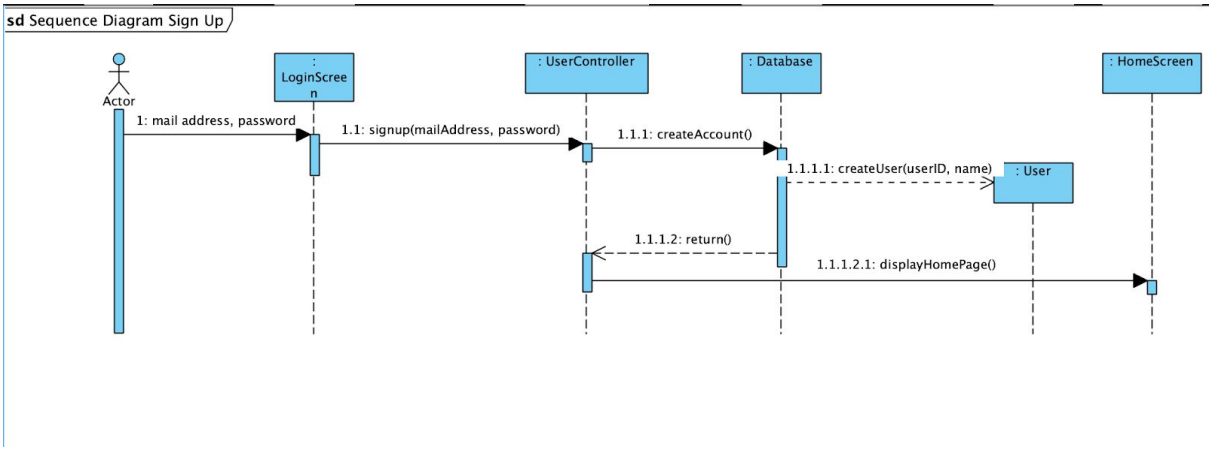
3.5.4.1 Activity Diagram



Description: User starts the process, then chooses *Sign Up* or *Sign In* actions. If user chooses *Sign Up* action system allows user 3 different sing up method. If user chooses *Sign In*, system checks whether the username and password is true or not. If given information is valid, user can continue with home page otherwise user is not allowed to use the system. When the user reach home page, there are 4 main actions, which are *Search, View Profile, Create Event, View Event* actions. If user chooses *View Event* action and attend the event, system allows user to *Match Friend or Give Feedback* actions, otherwise user is directed to *See Homepage* action. And, finally user can exist the system.

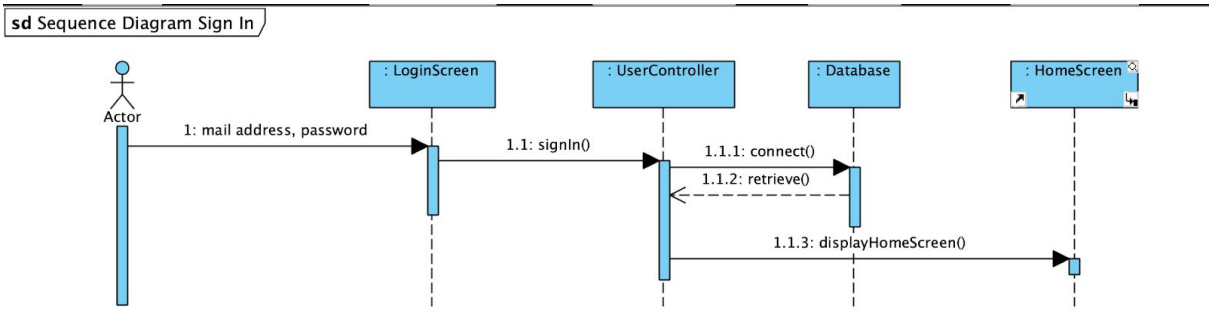
3.5.4.2 Sequence Diagrams

Sign Up:



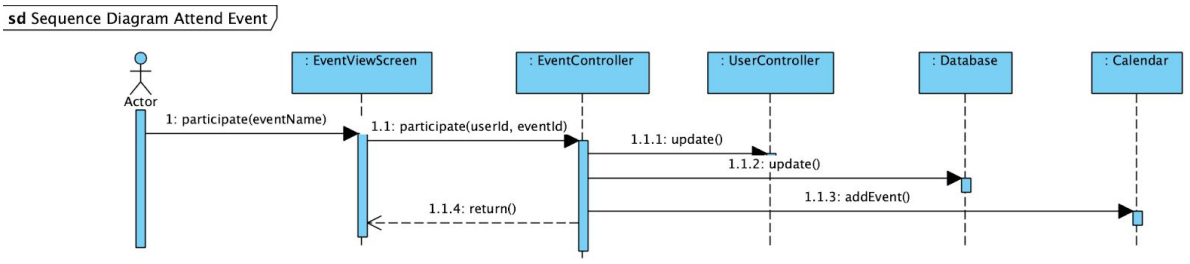
The diagram above shows the sign up process. User enters mail address and password on loginScreen, then signup action is completed by UserController, and updates the database. A new user is created, then HomeScreen is shown to user.

Sign In:



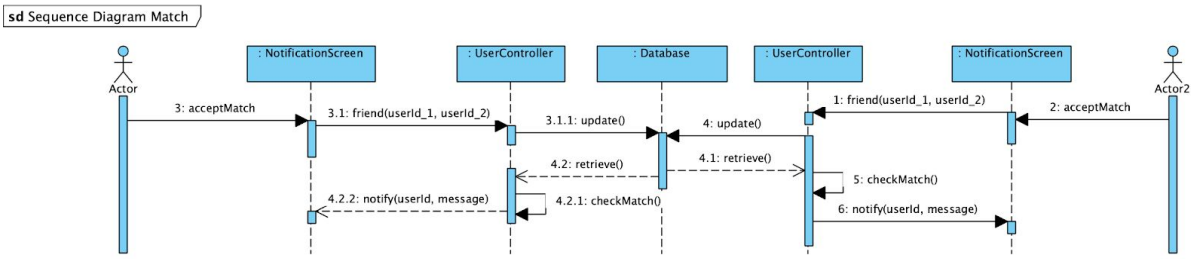
The diagram above shows the sign in process. User enters mail address, and password on LoginScreen. UserController completes signIn process, and controls the database whether the user is registered to the system or not. Then HomeScreen is shown to the user.

Attend Event:



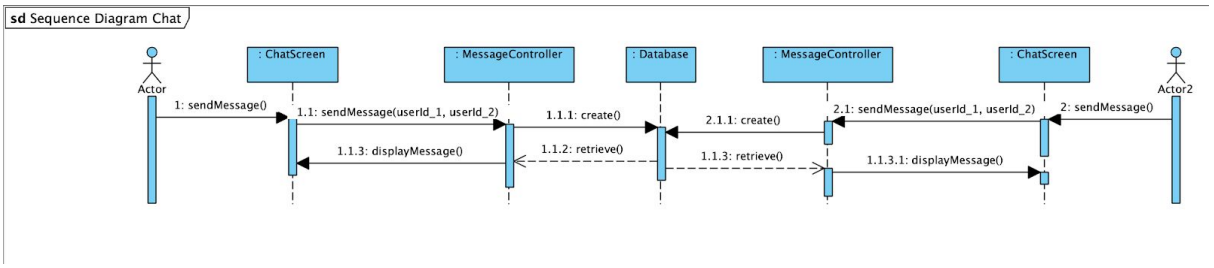
The diagram above shows the attend event process. User can participate to an event on EventViewScreen. EventContoller updates the user information, also this event is added to the calendar of the user. And EventViewScreen is shown to the user.

Match:



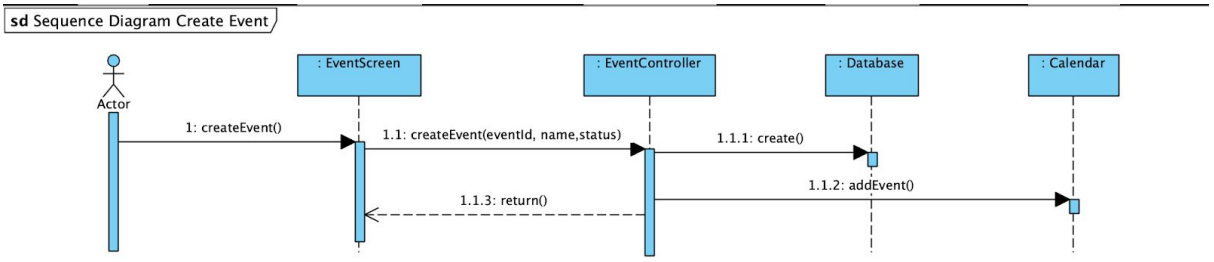
The diagram above shows the match process. User can accept the match on NotificationScreen. If the user accepts the match, UserController adds as friend that user, and also updates the database. Then, before sending notification to the user, UserController checks the match is accepted or not, and sends notification to the user.

Chat:



The diagram above shows the chat process. User can send message on ChatScreen. And message is sended by using MessageController. Then message is recorded on database, so this message is shown to the other user.

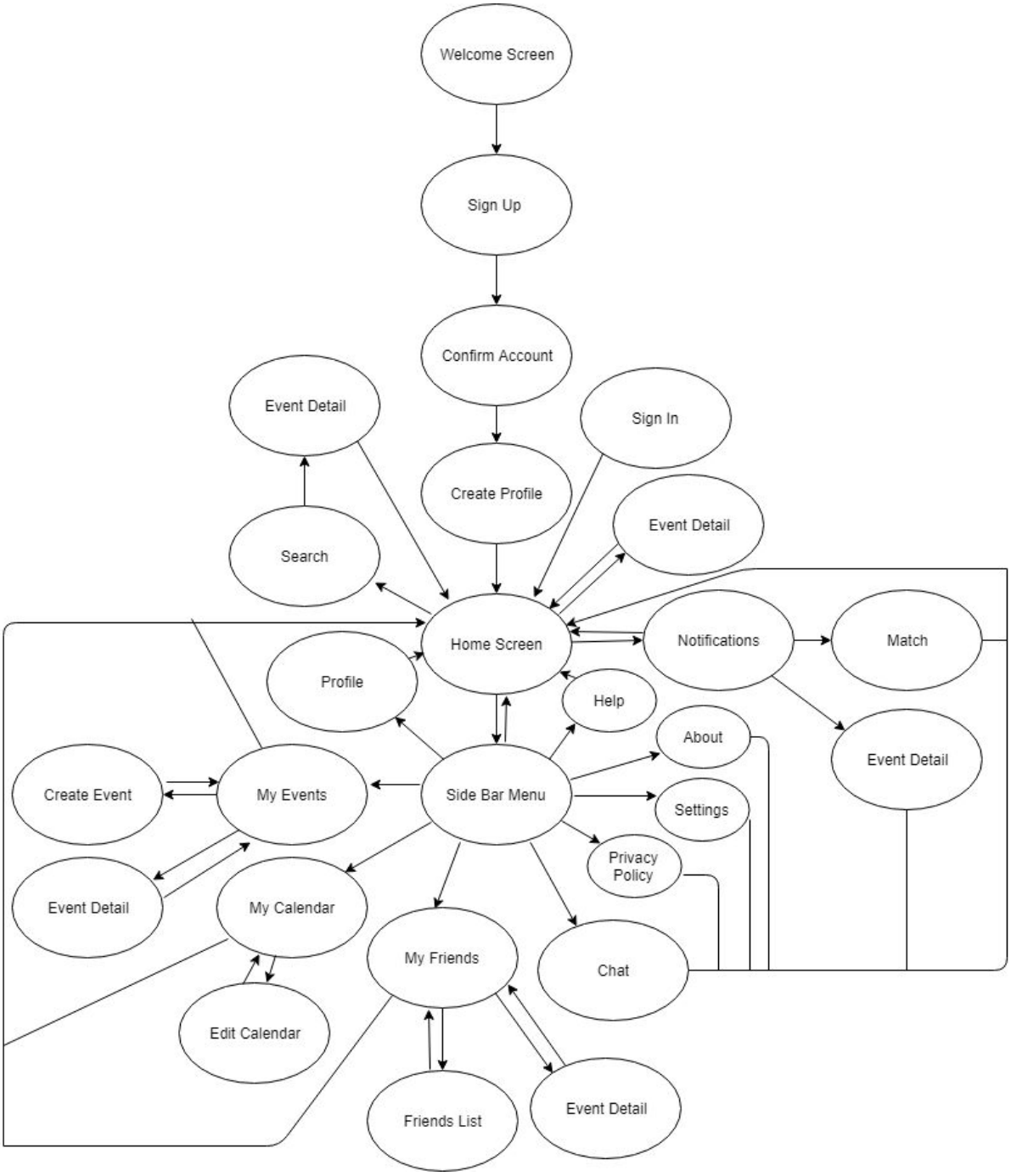
Create Event:



The diagram above shows the create event process. User can create an event on EventScreen. Then, EventController creates the event and add to database. Also, EventController adds this event to calendar of the user. Then, EventScreen is shown to the user.

3.5.5 User Interface - Navigational Path and Screen Mock-ups

3.5.5.1 Navigational Path Diagram

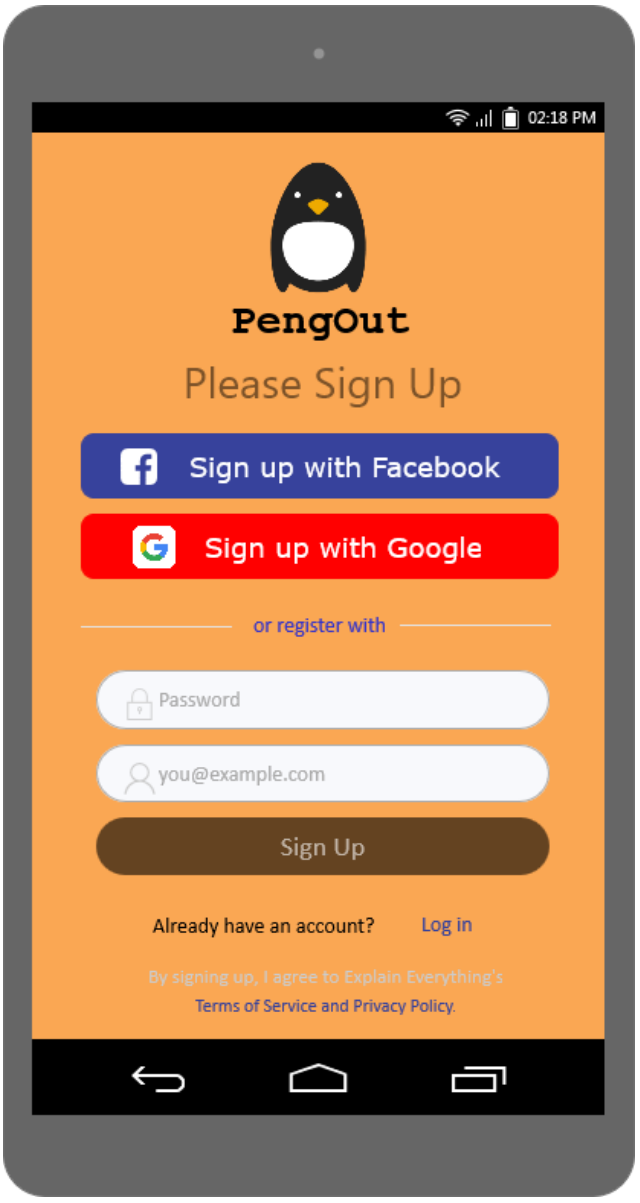


3.5.5.2 Mock-ups

3.5.5.2.1 Welcome Screen

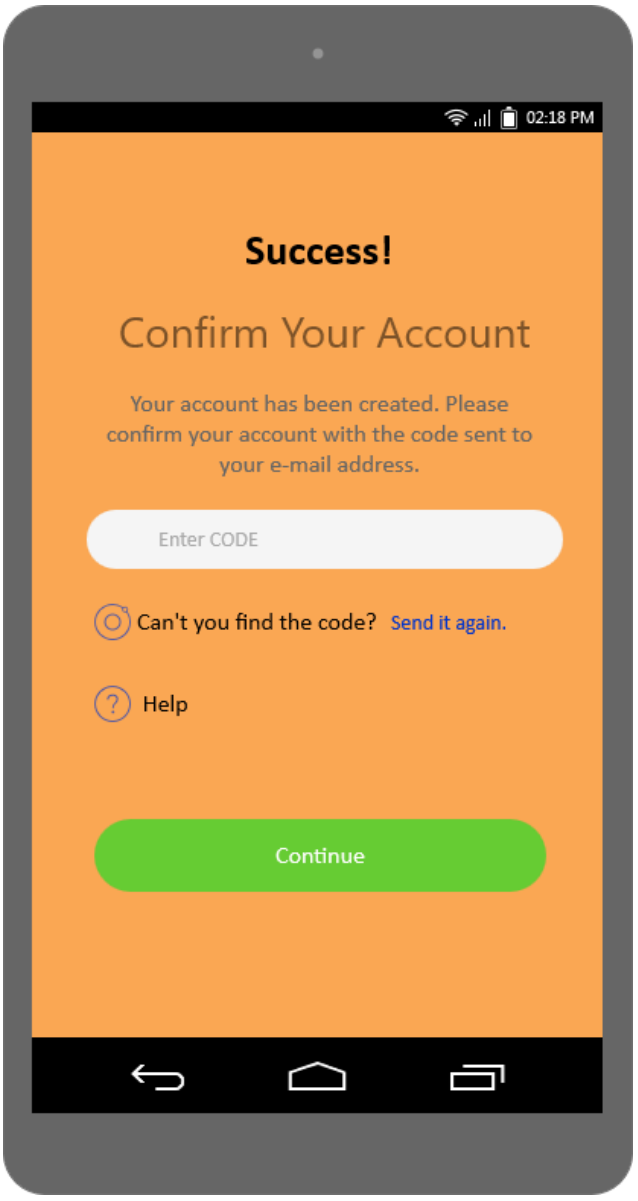
Welcome Screen includes our logo, welcomes user with a blackout effect.

3.5.5.2.2 Sign Up Screen



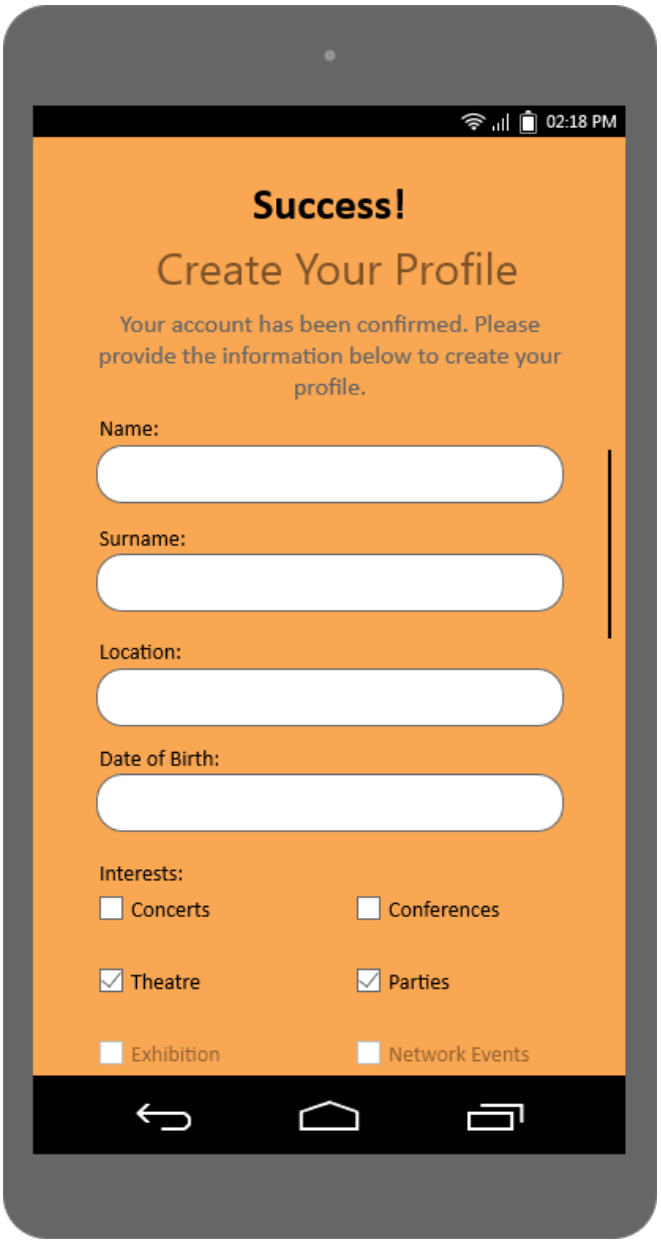
Since our platform requires an account, this page allows user to sign up with Facebook account, Google account or separate account to the system.

3.5.5.2.3 Confirmation Screen



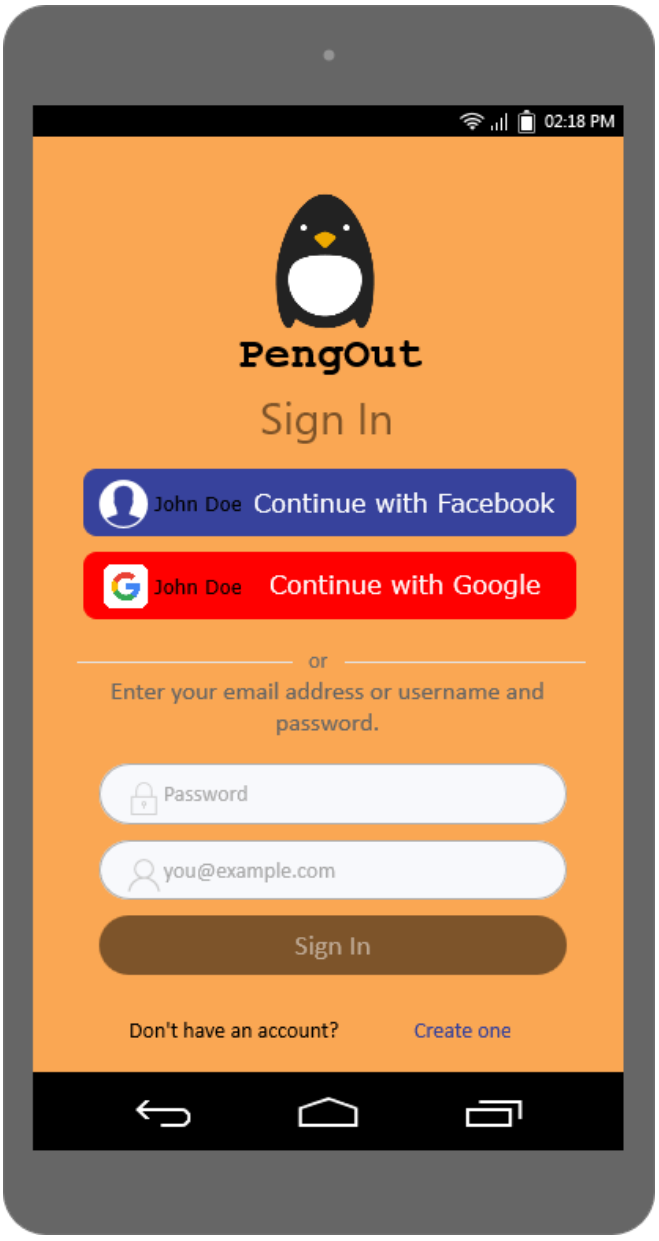
Since we care about reliability, we asked users to confirm their email address/account. After entering code sent to their email address, users will be able to continue to create their profile.

3.5.5.2.4 Create Profile Screen



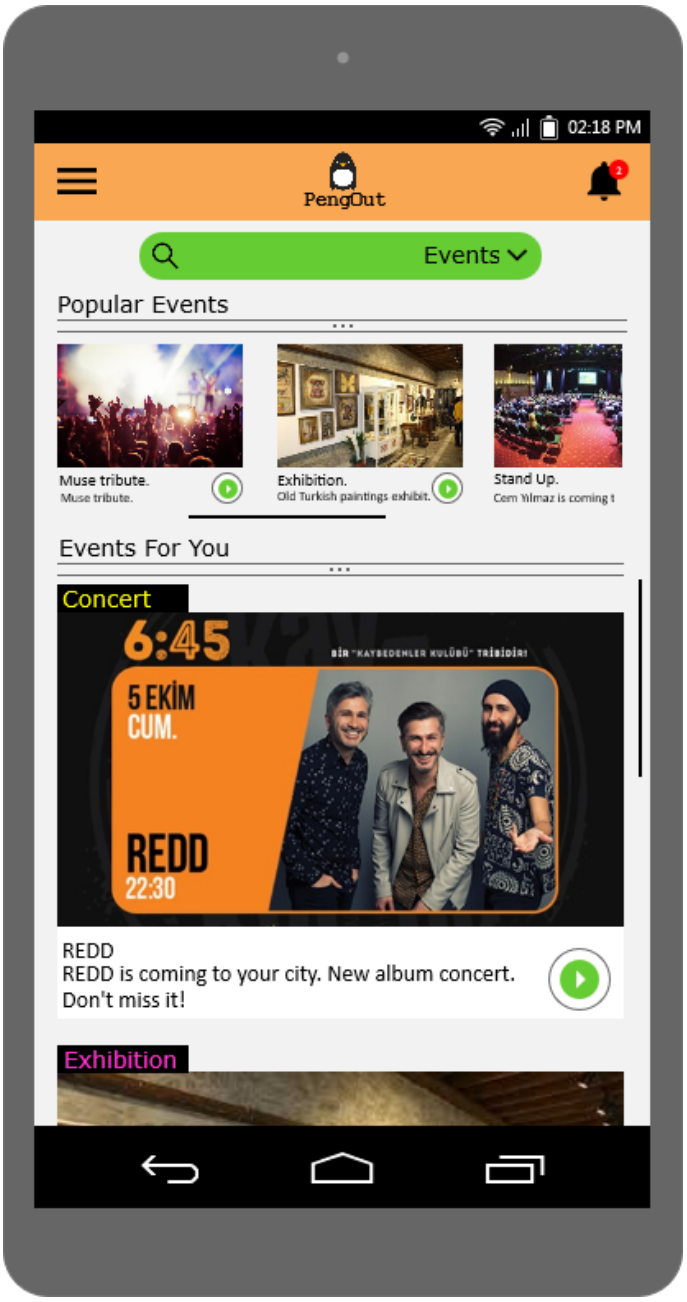
On Create Profile Screen, we ask all the information we need from user. Especially, interests and event kinds are important for our event feed.

3.5.5.2.5 Sign In Screen



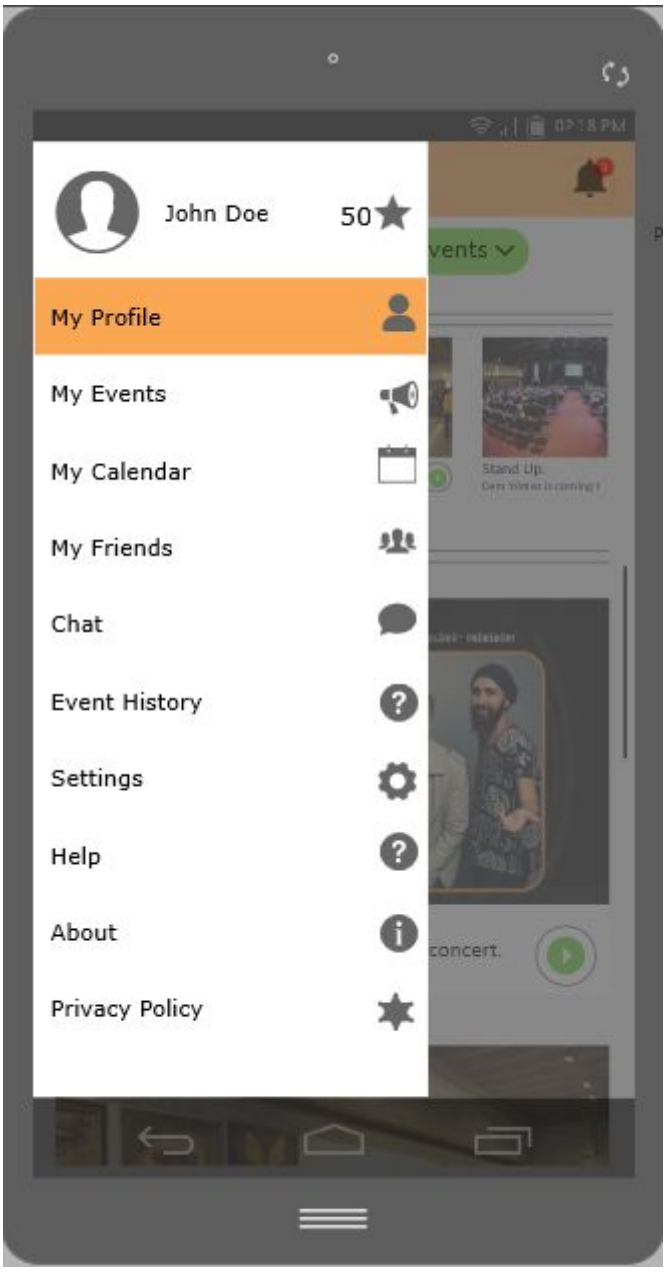
On Sign In screen, users can continue with their signed up Facebook or Google accounts, or log in to our system with their registered account. There is a markup for people do not have account or want to have another account.

3.5.5.2.6 Home Screen



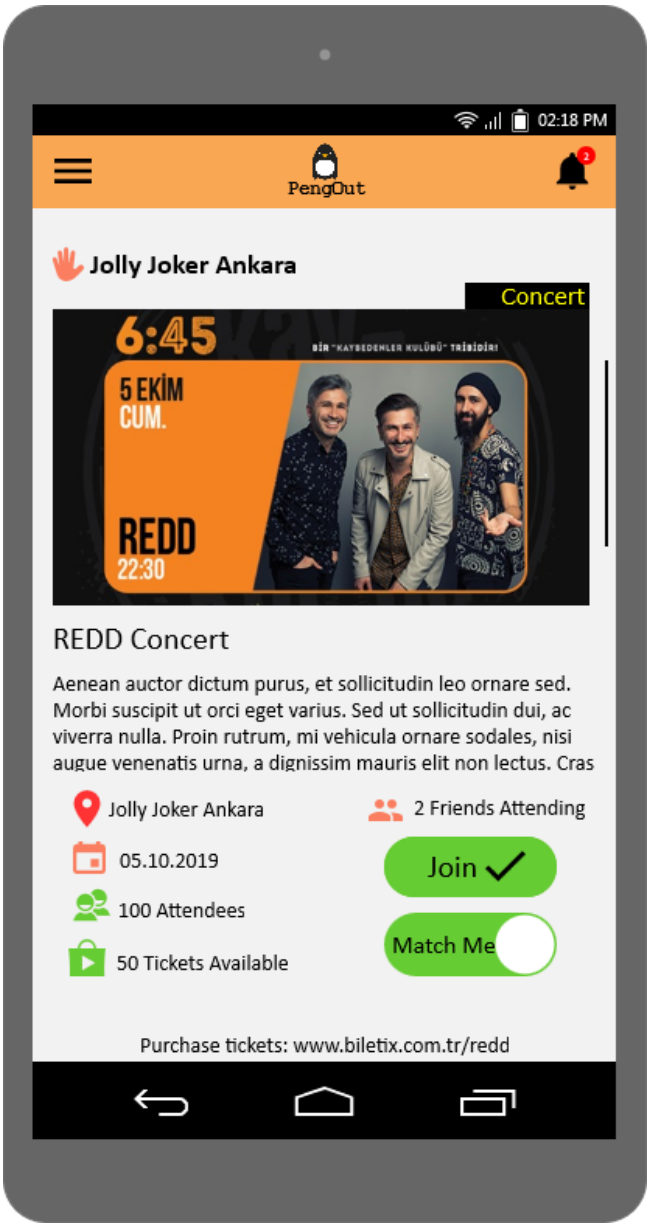
On Home screen, users see our logo, slide menu and notifications button on top. Then there is a search button for events, people, categories and organisations. Then users see popular events, followed by events according to their interests. They can click on the green button and see the event details. Users can scroll down and see all events for them.

3.5.5.2.7 Menu Screen



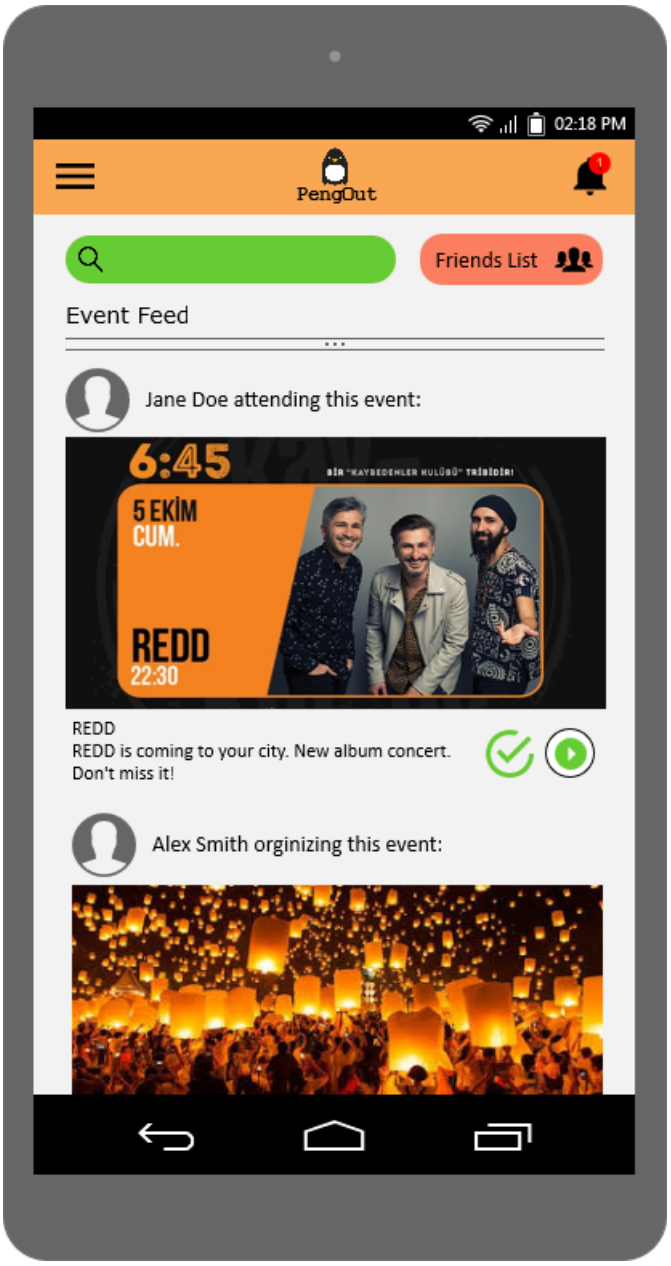
On Menu screen, users can see all the functionalities, their rate too. It is a slide bar menu type.

3.5.5.2.8 Event Detail Screen



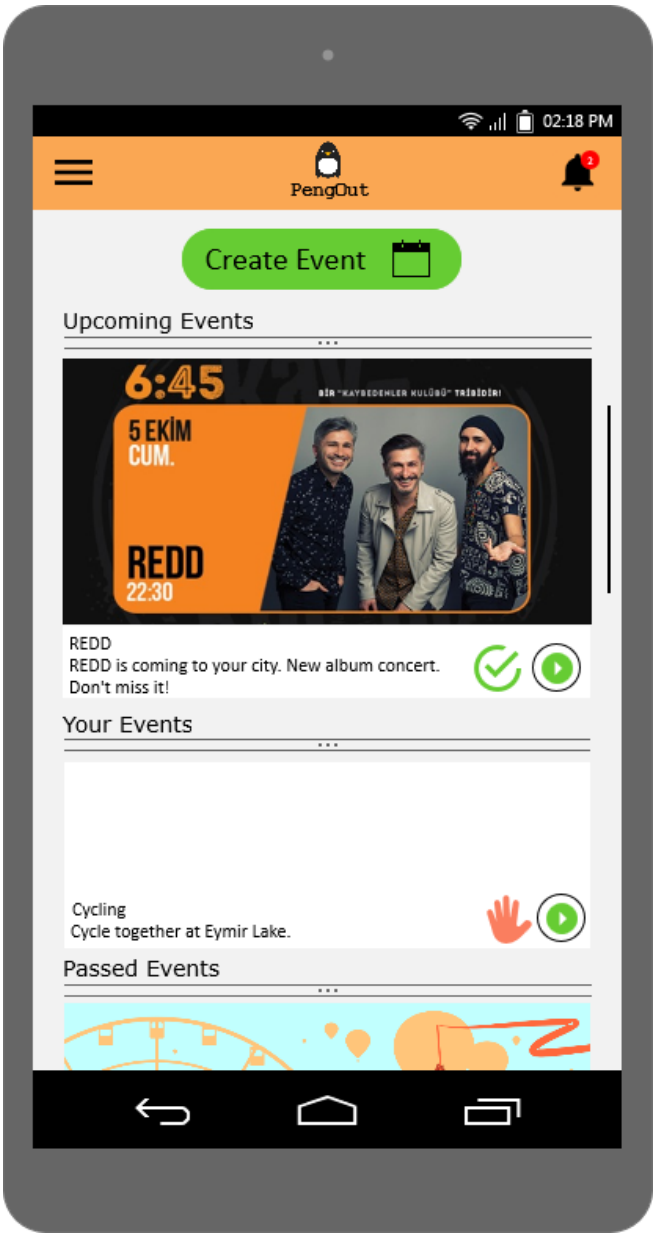
On Event Detail screen, users can see all the details for a particular event. The host, cota, location, date, description and ticket info. Via this screen, users can join the event and click the match button to find a person to go with. While the match button is on, our system will match that user to another one. By using android navigation bars, user can return the home screen.

3.5.5.2.9 My Friends Screen



On My Friends Screen, users can see the activity of their friends as event feed. Also when they clicked the search button on top, they can search for a specific friend. When they clicked the friends list button, they can see a list of their friends.

3.5.5.2.10 My Events Screen

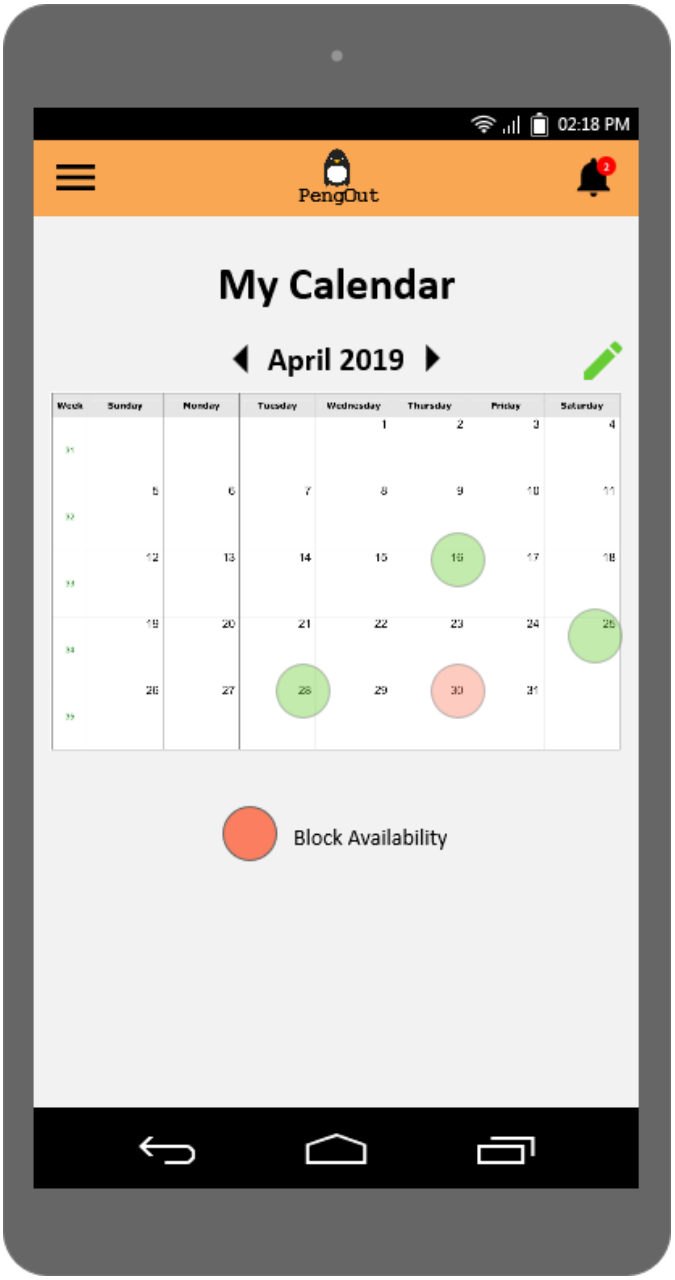


On My Events Screen, users can create an event by clicking the create event button. then they can see their upcoming events which they marked as joined, then see the events they will host. Also, users can see the passed events they joined or hosted. By clicking a particular passed event, they can rate/provide feedback for that particular event.

3.5.5.2.11 Create Event Screen

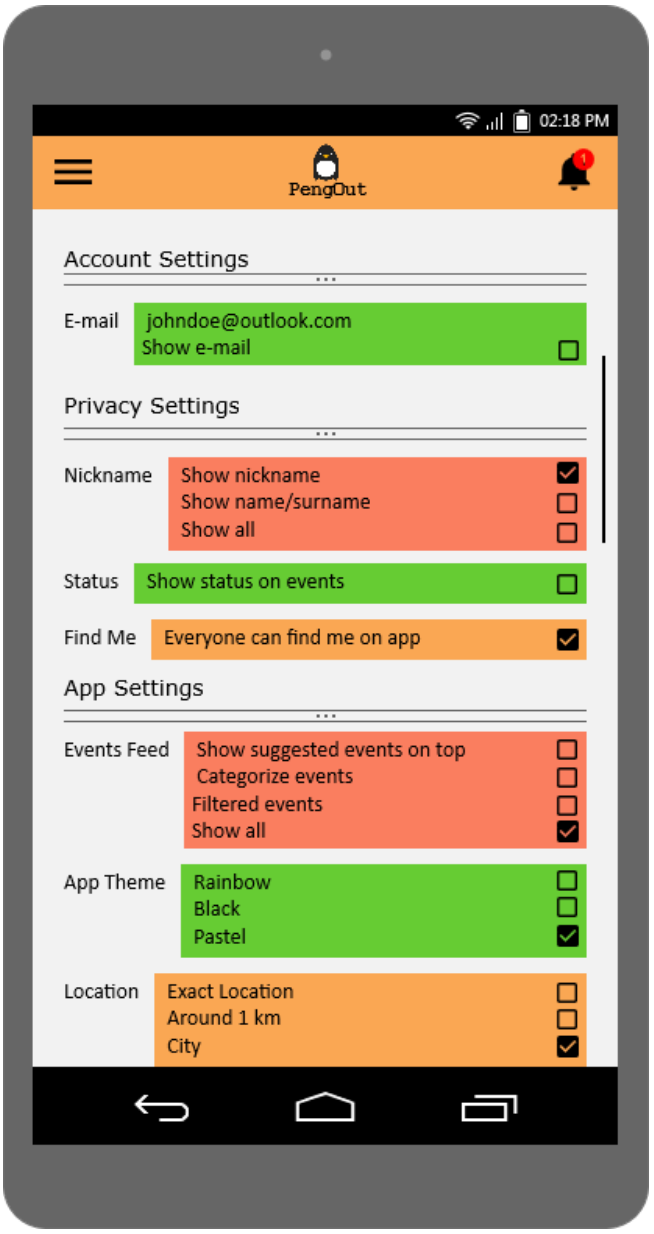
On Create Event Screen, users can enter required information like title, picture location, details, event type etc. They also adjust the privacy of event. A user can create events for just themselves like studying, reading book just for planning. When post the event button is clicked, event becomes open to other users.

3.5.5.2.12 My Calendar Screen



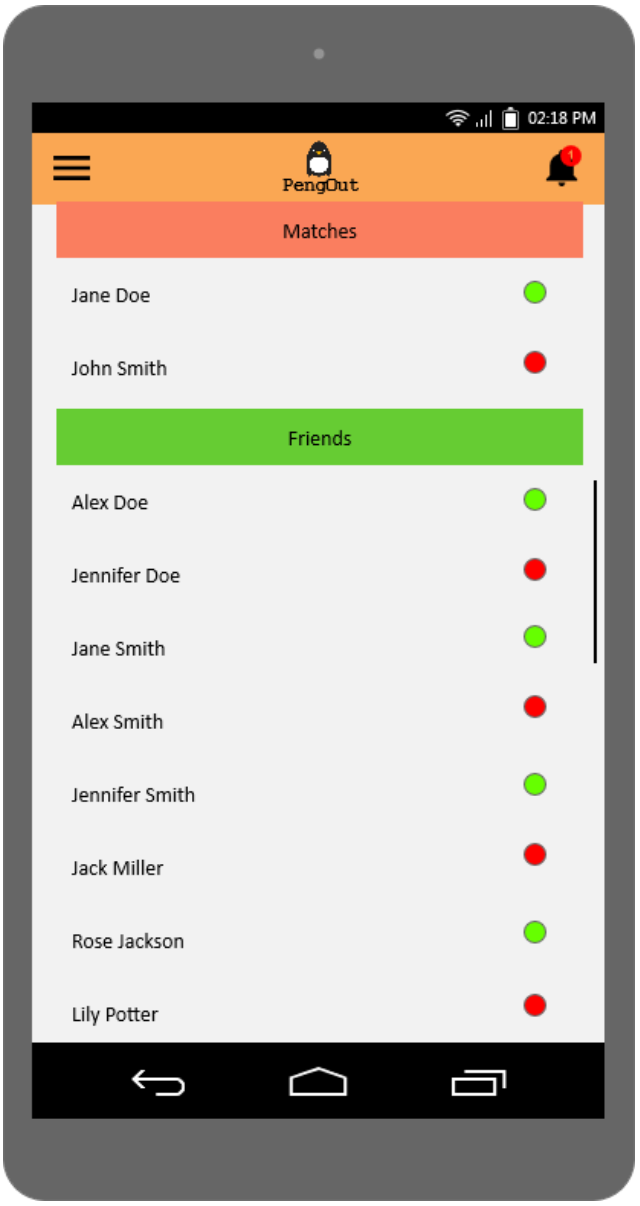
On My Calendar screen, users can see their events in a calendar. They can edit the calendar to block the times when they are not available. Our system will show events according to user's free time.

3.5.5.2.13 Settings Screen



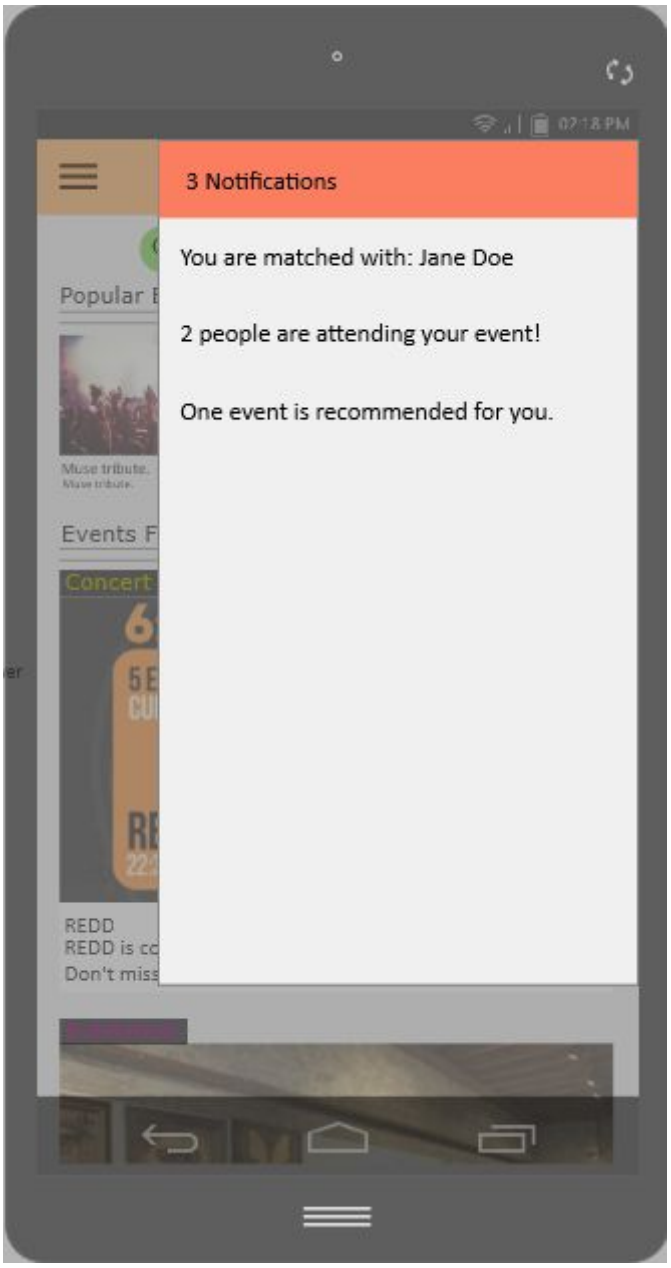
On Settings screen, users can change settings.

3.5.5.2.14 Chat Screen



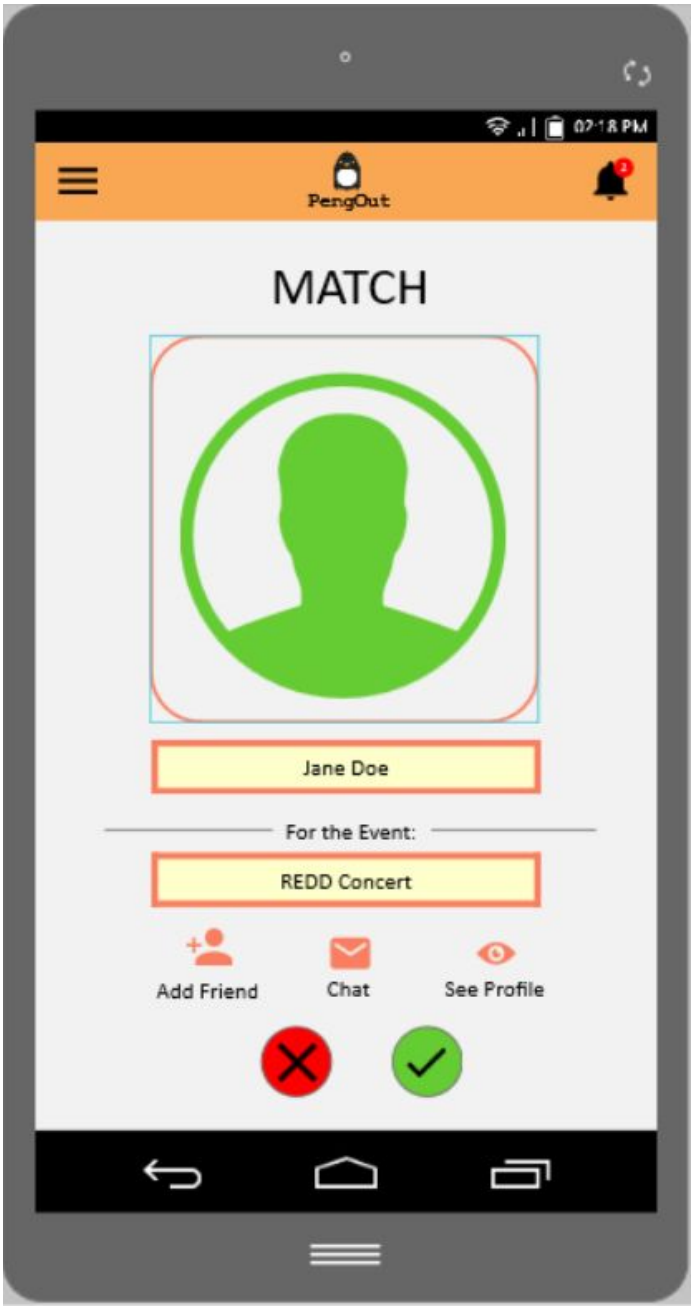
On Chat screen, users can see their friends and matched people before. They can chat with them and see whether they are online or not.

3.5.5.2.15 Notifications Pop-up Screen



Notifications pop-up screen shows the notifications for matches, for events, for friends activities.

3.5.5.2.16 Match Screen



When a match notification is clicked, Match screen becomes visible. It shows the account information, chat and add friend options. They must decline it or accept it.

4. Glossary

- **Match:** Our system will match people going to same events according to their interests. It called a match.

5. References

[1] Morris, M.R., Teevan,J. and Panovich, K., What Do People Ask Their Social Networks, and Why?. [online] Available at: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/chi10-social.pdf [Accessed 24 Feb. 2019]

[2] Meetup.com. (2019). About - Meetup. [online] Available at: https://www.meetup.com/about/ [Accessed 24 Feb. 2019].

[3] Circle-local-social-network.en.softonic.com. (2019). Circle - Local Social Network for Android - Download. [online] Available at: https://circle-local-social-network.en.softonic.com/android [Accessed 24 Feb. 2019].

[4] Socialradar.com. (2019). Welcome to socialradar.com. [online] Available at: http://www.socialradar.com/ [Accessed 24 Feb. 2019].