Bilkent University

Department of Computer Engineering

Senior Design Project

QUEX

Analysis Report

Barış Ardıç 21401578

Emir Acımış 21201233

Mert Kara 21400976

Umutcan Aşutlu 21301093

Atakan Özdemir 21301134

Supervisor: Çiğdem Gündüz Demir

Jury Members: Fazlı Can, Hamdi Dibeklioğlu

Nov 6, 2017

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

Quex is a mobile social media application which aims to find its users trustworthy individuals to help them with the problems encountered in their daily routine. The program provides a platform that connects the user to another user with required skills to try and solve any problem that can occur such as computer hardware or software issues, daily decision-making, travelling or dining.

Even the simplest problems take huge amount of time when the person facing the problem is not familiar with the problem domain. In addition, although there may be lots of solutions described online to the problem, unfamiliarity with the problem domain makes it harder for the person to understand and put the solution into practice. Quex aims to solve this problem.

2 Proposed System

2.1 Overview

The basic principle of the program assumes that there exists a user with a problem, the user notifies the program of that problem and gets assigned to an another user of the program who is close by if that user is considered an expert on the topic based on user background, in app ratings and user history with similar problems. After matching the user's Quex then allows these users to start a conversation in order to come up with a solution.

We are trying to solve the problem of "expert finding," therefore, there is a noticeable amount of features regarding expertise validation. The program is designed to fuse certain methods in order to determine which user would be the most appropriate and helpful for which problem. Quex considers the feedback from users and the user's

preferences in their profiles. The program uses the GPS data of its users to consider how close the potential expert is to the user with the problem. The closest possible expert is generally the best match. However, there also exist situations where the proximity of the expert is irrelevant, so the user can ignore the generic matching method and push the notification directly to main dashboard under a certain category where the problem can be seen by users without matching. This use cases that benefit from locality can present themselves in a variety of situations.

Consider that you are in a university campus and you are having trouble with your computer. A simple notification to Quex can quickly refer you to an engineering student t whom may be able to help you instantly. In completely different setting imagine that you are standing in front of the movie theater and trying to decide which movie to see, notifying the program will potentially result in a match with another user who just come out of a movie in that theater. The locality (user GPS data) aspect of the program provides the user with exclusive information that cannot be found on the internet such as how to connect to Bilkent Vpn for the first use case example or information about the sound system of the movie theater for the second example.

2.2 Functional Requirements

2.2.1 Login and Sign Up

- Users should be able to login to the program via their Facebook or Google accounts.
- If users do not have a Facebook or a Google account, they can create their own Quex account.
- They need to decide their user name and password to login. Also, e-mail address and full name of the user is needed.
- In case a password is lost, user can recover it via his/her e-mail verification.

2.2.2 Customization of Profile

- Program will ask user to create a profile.
- Their profile Picture, city, expertise and personal information can be accessible in user profile section.
- Areas of knowledge and proficiency need to determine by user.
- User profiles are updatable, users can change and add different information about themselves. Also expertise of the user can be added or deleted later.
- It is possible to reach a user's reputation which he/she gains with his/her behaviors in the program.

2.2.3 Problem and Matching

- Users should be able to send problems to the program with or without certain fixed category.
- Quex must have fixed expertise categories in order to provide better matching between users.
- They can create various topics, problem description (if it's needed) and its category are needed to start matching process.
- Quex tries to match a problem with the closest expert. To do this, it uses the GPS data of the users. This data is taken from the Google Maps API.
- Elapsed time of the matching process is shown by the program to the user.

- When program finds a match, user have an option to accept or decline it. If the match is accepted by the user, a notification will be sent to the expert.
- If the match is accepted by expert too, the program will provide for a real-time conversation in Inbox section.

2.2.4 Dashboard

- If program cannot find any expert for a specific category and it fails to match it locally, the problem will be appeared in Dashboard section.
- Everyone who use the program and have a proficiency about the category can see the problem in their dashboard.
- Later, they can provide help to owner of the problem if they want. Dashboard is not instant, but provides a chance to a user if he/she cannot find a local expert for his/her problem.
- There is a search option to find a specific problem or a specific account in dashboard.
- If they want, users can directly send their problems to the dashboard.

2.2.5 Inbox and Real Time Conversation

- When user matches with an expert via dashboard or local matching, a real time conversation is started by the program in Inbox section.
- The message traffic between users is monitored by the program in order to block certain behavior that is considered inappropriate or irrelevant. For instance, it should not be possible to send a phone number or slang words.
- In case a need arises for users to meet, users can share their location voluntarily.
- Users can search for specific keywords in Inbox section.
- Users will be asked to rate the expert after the conversation is closed. Also experts can up vote or report users too.

2.2.6 Settings and Logout

- Program provides settings section to manage user accounts.
- Notifications, change password, blocked users, privacy will be covered by the program settings.
- Settings section will be revised more detailed later. Additional features may be added.

- Users can logout with a button in menu and program redirects them to login screen again.

2.3 Non-Functional Requirements

2.3.1 Extensibility

- Quex will work better as the user base gets larger. Extended user number means much more matching and chance to solve o problem.
- Quex's side menu system will allow new extensions, these extensions can be suggested by users or determined by the development team.

2.3.2 Supportability

- Quex will be easily adapted to the new technologies or features via new modules. Object oriented design principles will allow for these type of expansions.
- New updates will be regularly provided to make the application up to date and innovative.

2.3.3 Usability

- Quex will be easy to use and easy to understand its working mechanism.
- Quex will have user friendly interface.
- Sign in and sign up functions will not be complicated, user can login via their Gmail or Facebook accounts. The application gets minimal information from the user.
- Quex will provide helps and tips in itself to inform users about the application and its features.
- The application should not need a long time to perform operations and tasks. Time complexity of the services will be shortened as much as possible.

2.3.4 Availability

- Quex server will always be available for the users.
- Quex will be a mobile application which can be accessed very easily.
- The application will be free to use, users can download and use it without paying money.

2.3.5 Reliability

- Quex will be a bug and error free application.
- Quex will detect any undesired substrings with its black-list mechanism including bank account numbers to prevent use case of making profit by the application.
- Secured login system and reputation point system will increase the reliability of the users.
- Personal data of the users will be preserved well.

2.3.6 Portability

- Quex will be made for Android system but it can be ported to different operating systems like iOS, Windows with a little effort.
- Different ports of the program may be provided later.

2.3.7 Scalability

- Quex will support large number of users.
- Capacity of the Quex will be increased according to number of users.

2.4 Pseudo Requirements

- The program will be made to run on Android Operating System.
- Quex will have a server.
- The program will be written in java.
- The program will be implemented on Android Studio.
- Internet connection and GPS system are required for Quex.

2.5 System Model

2.5.1 Scenarios

2.5.1.1 Sign Up

Use Case Name: Sign Up

Actors: User

Entry Conditions: User is on Login screen.

Exit Conditions: User is successfully registered to the system OR user hits the back button.

Basic Flow:

1. User hits the “Sign Up” option.
2. Quex displays Sign Up screen.
3. User enters required information.
4. User hits the “Create Account” option.
5. Quex validates the information and creates the account.
6. Quex displays the “Profile” screen for expertise selection.

Alternative Flow:

1.
    a. User hits “Sign Up with Facebook” option.
    b. User hits “Sign Up with Google” options.
2.
    a. Quex displays “Facebook Login” screen.
    b. Quex displays “Google Login” screen.
3. User enters e-mail and password for the chosen platform.
4. User hits “Sign Up” option.
5. Quex assigns expertise by fetching profile information from the chosen platform.
6. Quex displays the “Profile” screen for expertise selection or removal.

2.5.1.2 Login

Use Case Name: Login

Actors: User

Entry Conditions: User is on Login screen.

Exit Conditions: The user is logged in to Quex.

Basic Flow:

1. User enters his email address.
2. User enters password.
3. User hits “Login.”

4. Quex validates the information.
5. Quex displays the Dashboard screen.

2.5.1.3 Create Topic

Use Case Name: Create Topic

Actors: Questioner, Expert

Entry Conditions: Questioner and Expert are both logged in. Expert gives permission to Quex for notifications.

Exit Conditions: Expert is found and accepted by the questioner, or matching is not successful.

Basic Flow:

1. Questioner hits the Create Topic icon.
2. Questioner is displayed Create Topic screen.
3. Questioner chooses the category which the topic is about.
4. Questioner writes a description of the topic to the text bar.
5. Questioner presses “Match” button.
6. Quex displays the Matching screen.
7. Quex searches for the closest experts on the topic.
8. Quex sends notifications to the closest experts.
9. Expert accepts the notification.
10. Experts rank is displayed to Questioner.
11. Questioner accepts the Expert’s help.
12. Chat screen is displayed to both.

Alternative Flow:

6. Quex finds no close experts who accept to help.
7. Quex puts the question into the dashboard.
8. Quex displays inbox to the Questioner.

2.5.1.4 Search in Dashboard

Use Case Name: Search in Dashboard

Actors: User

Entry Conditions: User is on the Dashboard screen.

Exit Conditions: User opens a topic, or User hits the Dashboard.

Basic Flow:

1. User hits the search icon.
2. Quex displays the search bar.
3. User enters the topic he wants to search.
4. User hits search.
5. Quex displays relevant topics on the dashboard.
6. User chooses and presses on a topic from the search results.
7. Quex displays the chosen topic's page.

Alternative Flow:

5. Quex cannot find any relevant search result.
6. Quex displays a warning that no search result was found.
7. User enters new keywords to search bar.
8. User hits search.
9. Quex displays relevant topics on the dashboard.
10. User chooses and presses on a topic from the search results.
11. Quex displays the chosen topic's page.

2.5.1.5 Search in Inbox

Use Case Name: Search in Inbox

Actors: User

Entry Conditions: User is on the Inbox screen.

Exit Conditions: User opens a chat, or User opens the Dashboard.

Basic Flow:

1. User hits the search icon.
2. Quex displays the search bar.

3. User enters the keywords he wants to search.
4. User hits search.
5. Quex searches the keywords in the chats existing on User's inbox.
6. Quex displays relevant chats.
7. User chooses and presses on a chat from the search results.
8. Quex displays the chosen chat.

Alternative Flow:

6. Quex cannot find any relevant search result.
7. Quex displays a warning that no result was found.
8. User hits the Inbox icon on top.
9. Quex cancels search and displays the Inbox.

2.5.1.6 Chat with Match

Use Case Name: Chat with Match

Actors: Questioner, Expert

Entry Conditions: Questioner and Expert are successfully matched.

Exit Conditions: User closes the topic as resolved or unresolved.

Basic Flow:

1. Questioner and Expert both accept the match.
2. Questioner and Expert are displayed the chat screen.
3. Expert and/or Questioner enter messages to the text bar.
    a.Expert and/or Questioner enter "Send Location" button.
    b.Quex displays a confirmation popup.
    c.Expert and/or Questioner hits "Confirm" button.
4. Expert and/or Questioner press "Send" button.
5. Quex displays the sent message on the Chat screen.
6. Questioner presses the "Resolved" button.

7. Quex marks the topic as resolved and closes the chat for further communication.
8. Quex increases the rank of the Expert.
9. Quex displays the dashboard.

Alternative Flow:

6. Questioner presses the “Unresolved” button.
7. Quex marks the topic as unresolved.
8. Quex displays the “Evaluate Help” screen to the Questioner.
9. Questioner hits a value between -5 to 5.
10. Questioner hits the “Done” button.
11. Quex puts the topic to the dashboard.
12. Quex changes the rank of the Expert accordingly.
13. Quex displays the dashboard.

Alternative Flow 2:

6. Questioner presses the “Unresolved” button.
7. Quex marks the topic as unresolved.
8. Quex displays the “Evaluate Help” screen to the Questioner.
9. Questioner hits a value between -5 to 5.
10. Questioner hits the “Rematch” button.
11. Quex displays the Matching screen.

2.5.1.7 Reply to Dashboard

Use Case Name: Reply to Dashboard

Actors: User, Questioner

Entry Conditions: Topic is in the dashboard as unresolved.

Exit Conditions: The chat is put on the User’s inbox.

Basic Flow:

1. User presses on the unresolved topic on Dashboard.
2. Quex displays the Chat screen of the topic.
3. User enters the message to the text field.

4. User presses “Send” button.
5. Quex puts the message to the topic’s chat screen.
6. Quex sends notification to the questioner of the topic.
7. Quex puts the topic chat to the inbox of the user.

2.5.1.8 View Profile

Use Case Name: View Profile

Actors: User

Entry Conditions: User is logged in.

Exit Conditions: User presses the “Back” icon.

Basic Flow:

1. User presses the “menu” icon.
2. Quex displays menu options on the current screen.
3. User presses “Profile” option.
4. Quex displays the Profile screen.
5. User presses back.

2.5.1.9 Edit Profile

Use Case Name: Edit Profile

Actors: User

Entry Conditions: User is on the Profile screen.

Exit Conditions: User presses the “Change” button.

Basic Flow:

1. User presses on the expertise he wants to remove.
2. User presses on the expertise he wants to add.
3. User presses “Change” button.
4. Quex assigns the changes to the profile.
5. Quex displays the Profile screen again.
6. User presses back.

2.5.1.10 Adjust Settings

Use Case Name: Adjust Settings

Actors: User

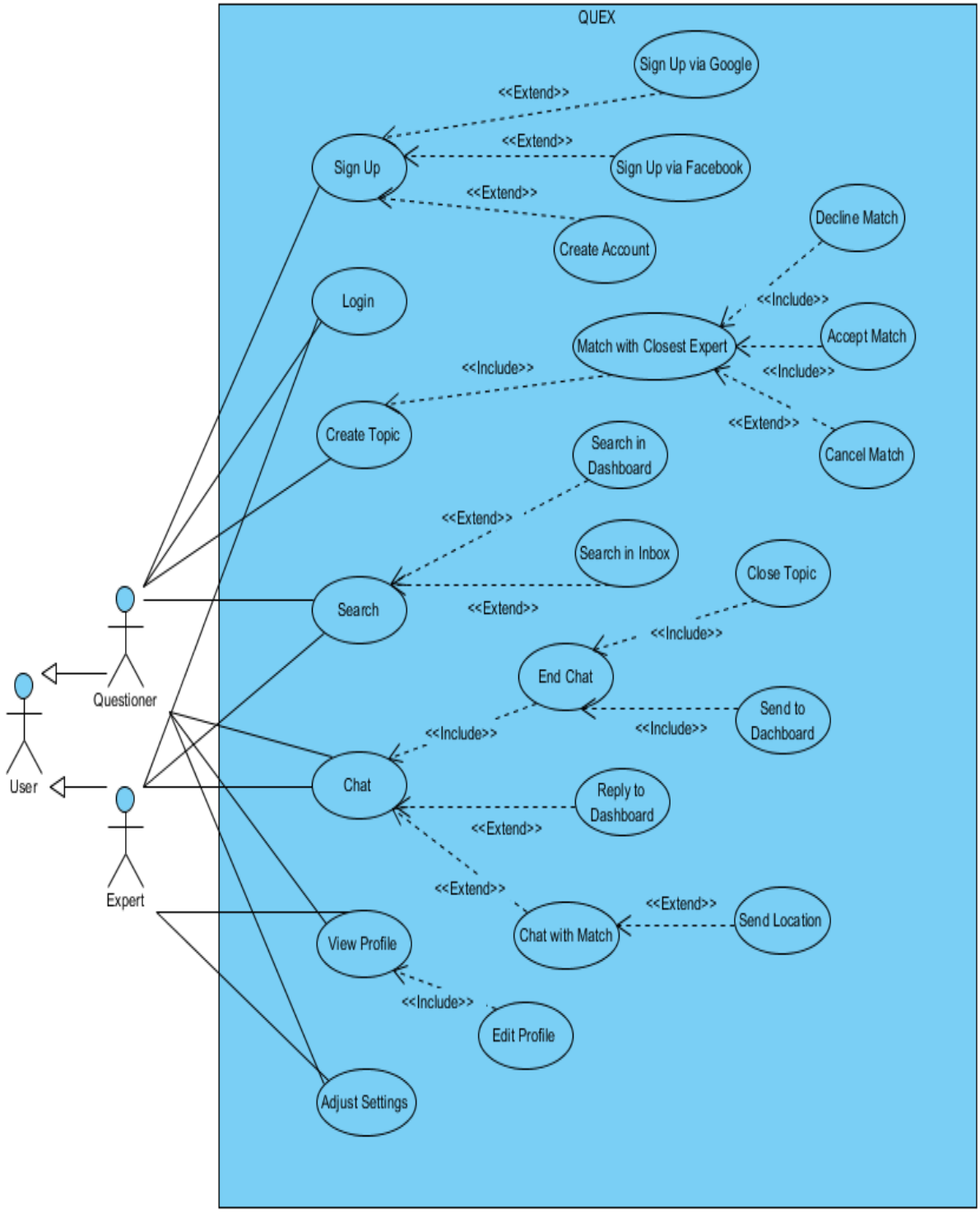
Entry Conditions: User is logged in.

Exit Conditions: User presses the “Done” button.

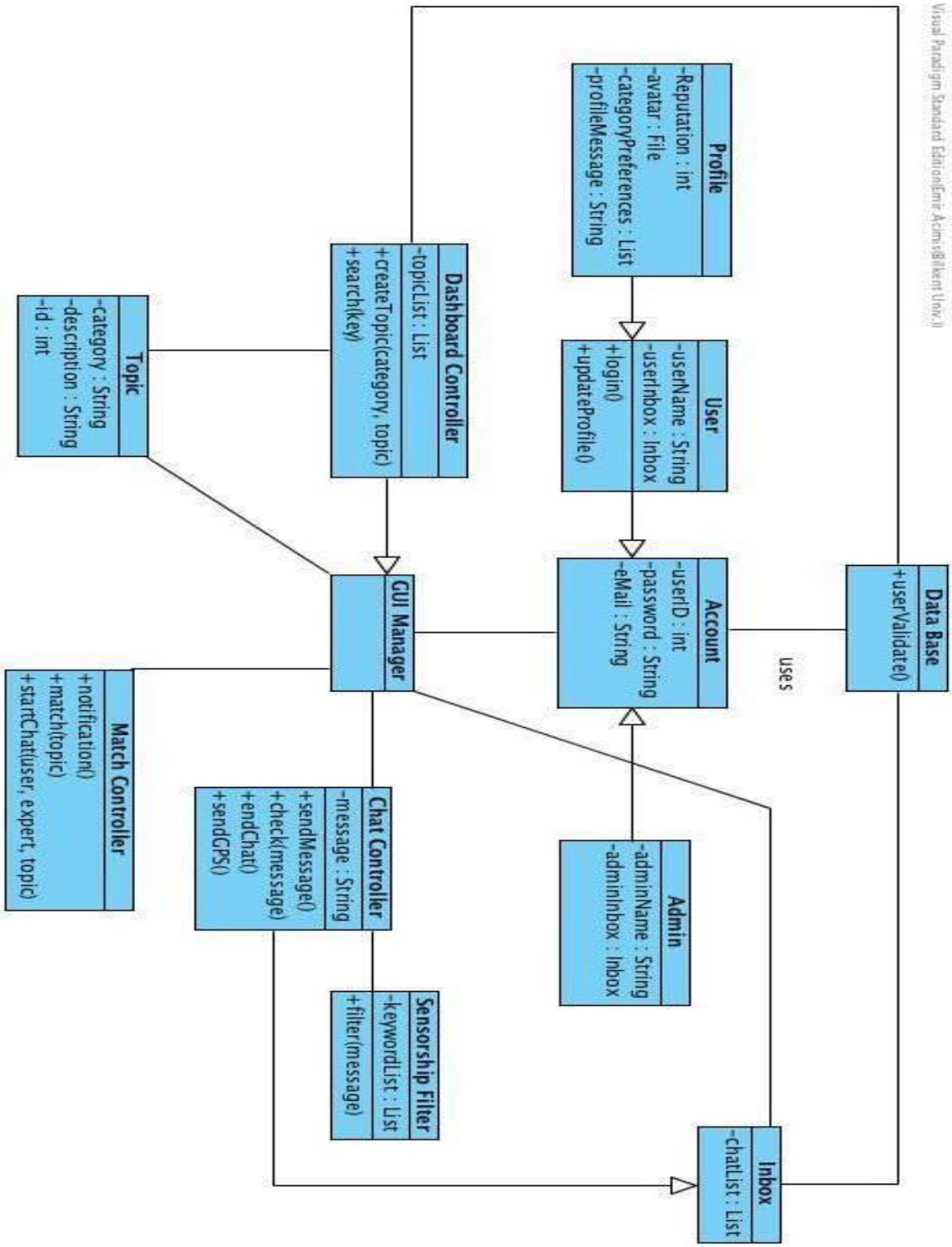
Basic Flow:

1. User presses the “menu” icon.
2. Quex displays menu options on the current screen.
3. User presses “Settings” option.
4. Quex displays the Settings screen.
5. User checks/unchecks the checkbox for the setting he wants to change.
6. User presses the “Done” button.

2.5.2 Use Case Model



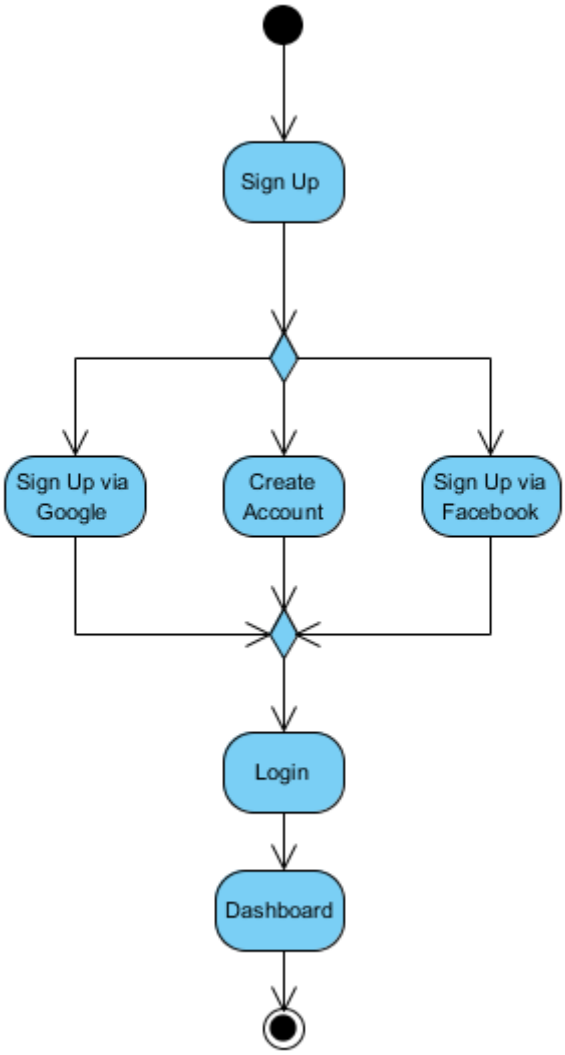
2.5.3 Object and Class Model



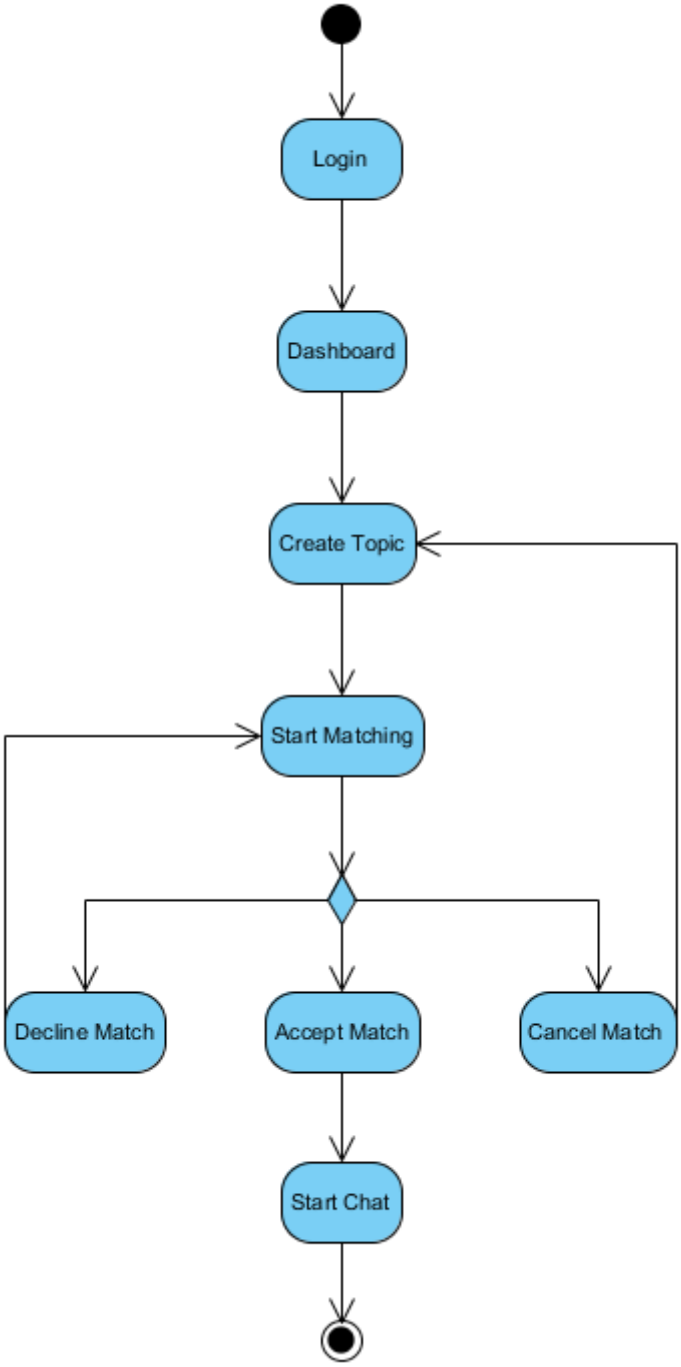
2.5.4 Dynamic Models

2.5.4.1 Activity Diagrams

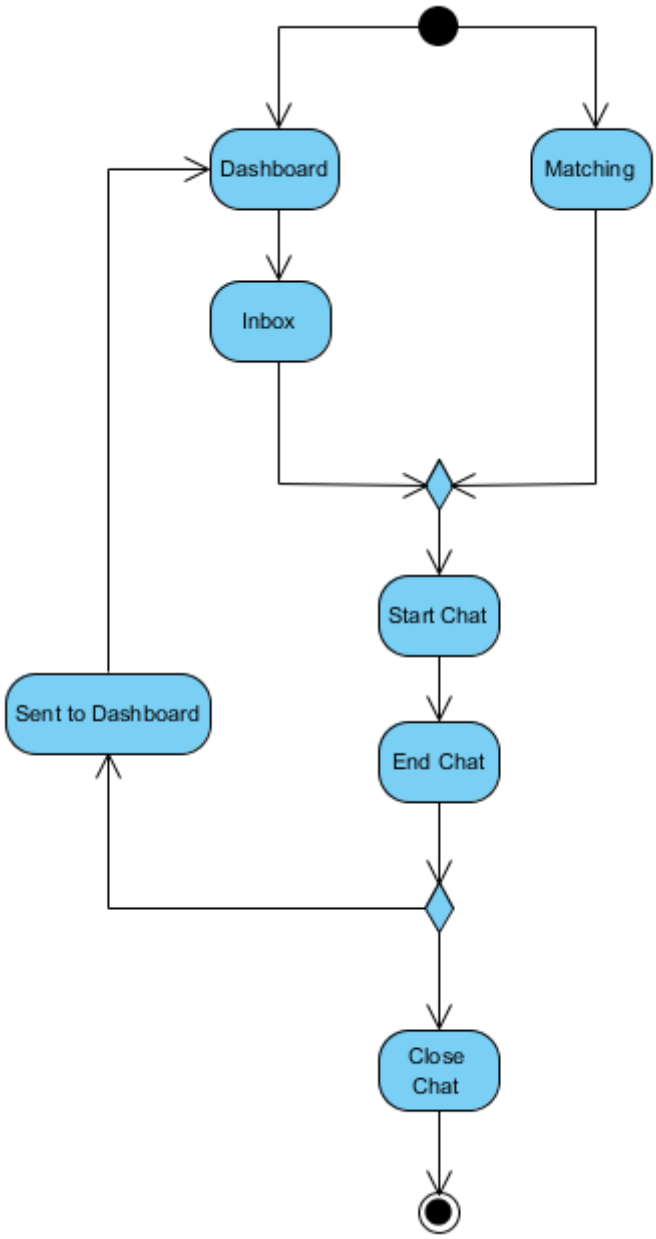
Activity Diagram for signing up to QUEX :



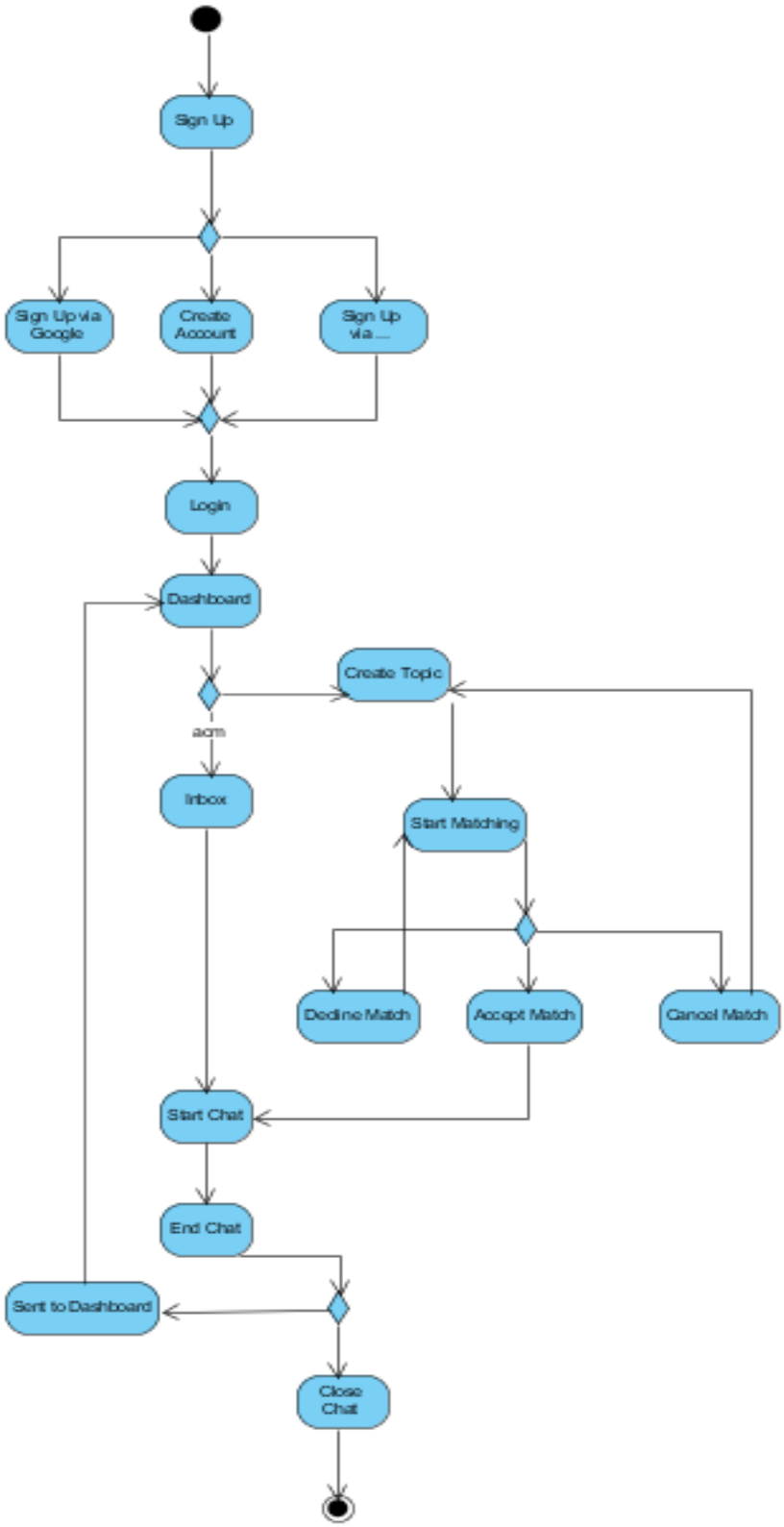
Activity Diagram for creating a topic in QUEX :



Activity Diagram for chatting in QUEX :

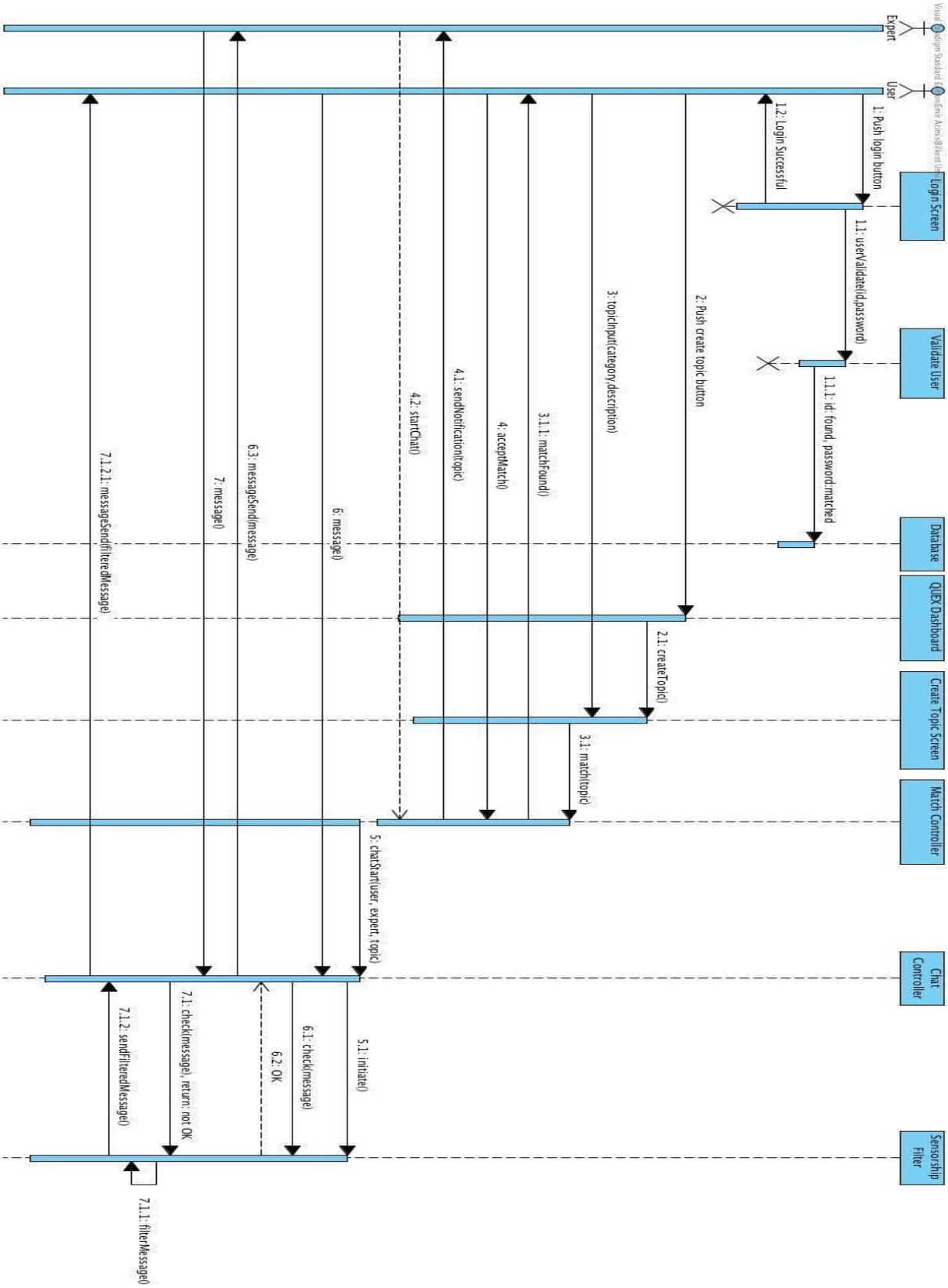


Activity Diagram for the general flow of QUEX :

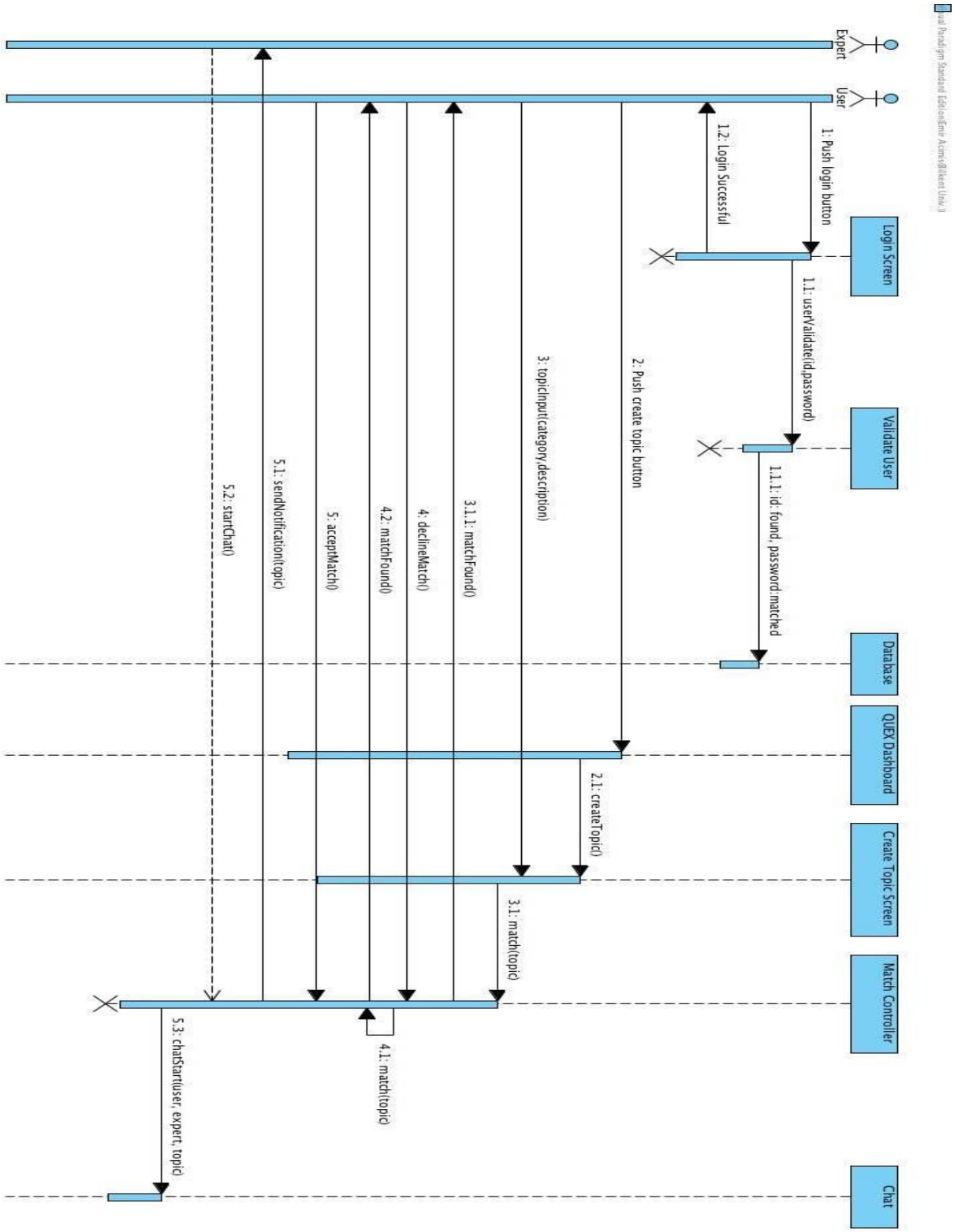


2.5.4.2 Sequence Diagrams

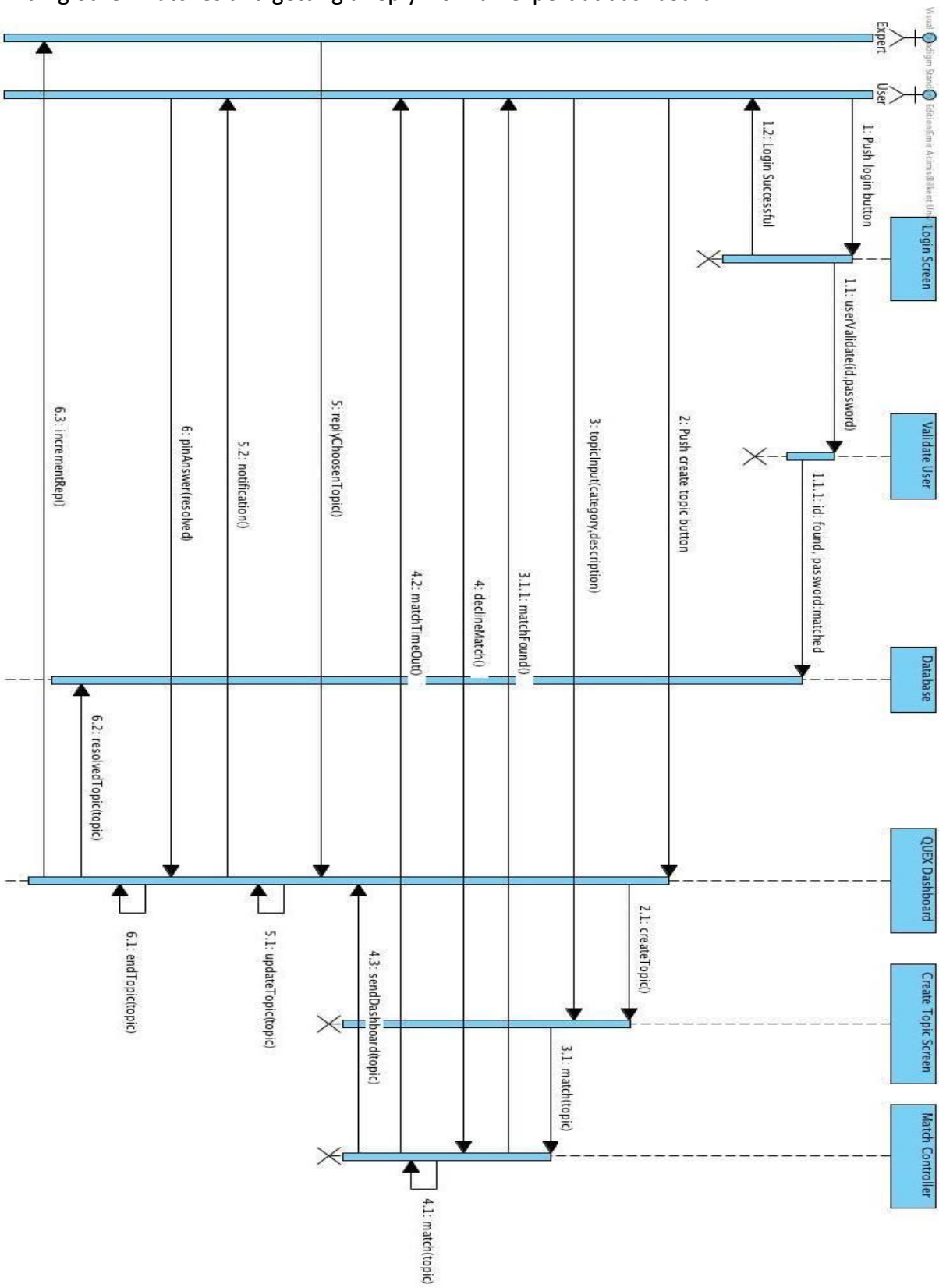
Sequence Diagram for a user logging in, creating a topic and chatting with a matched expert.



Sequence Diagram for a user logging in, creating a topic, declining the first and accepting the second match.

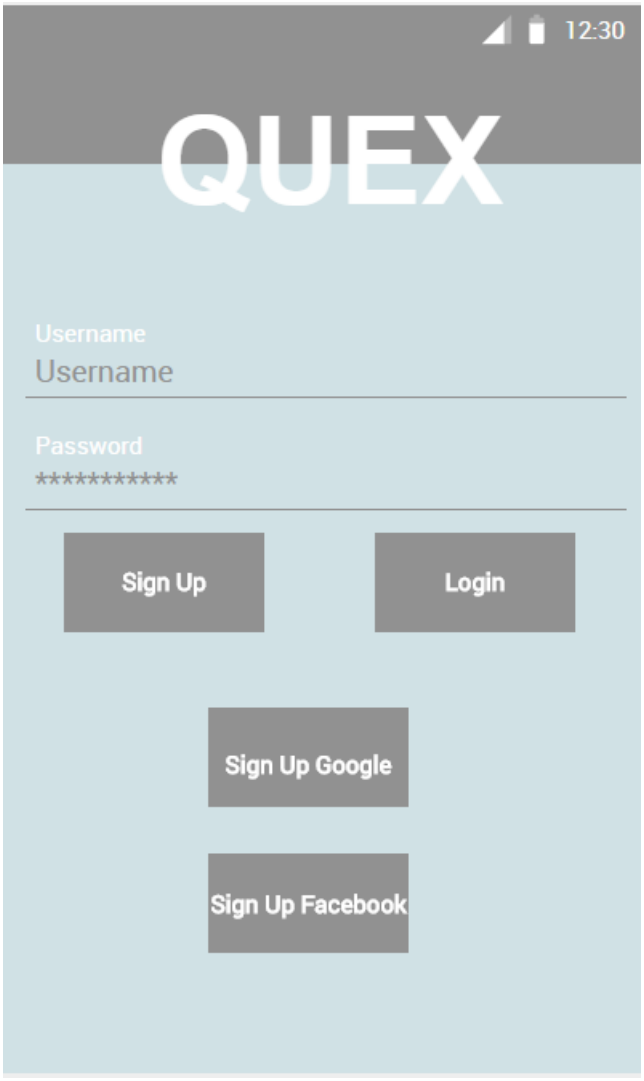


Sequence Diagram for a user logging in, creating a topic, declining the first match without finding other matches and getting a reply from an expert at dashboard.

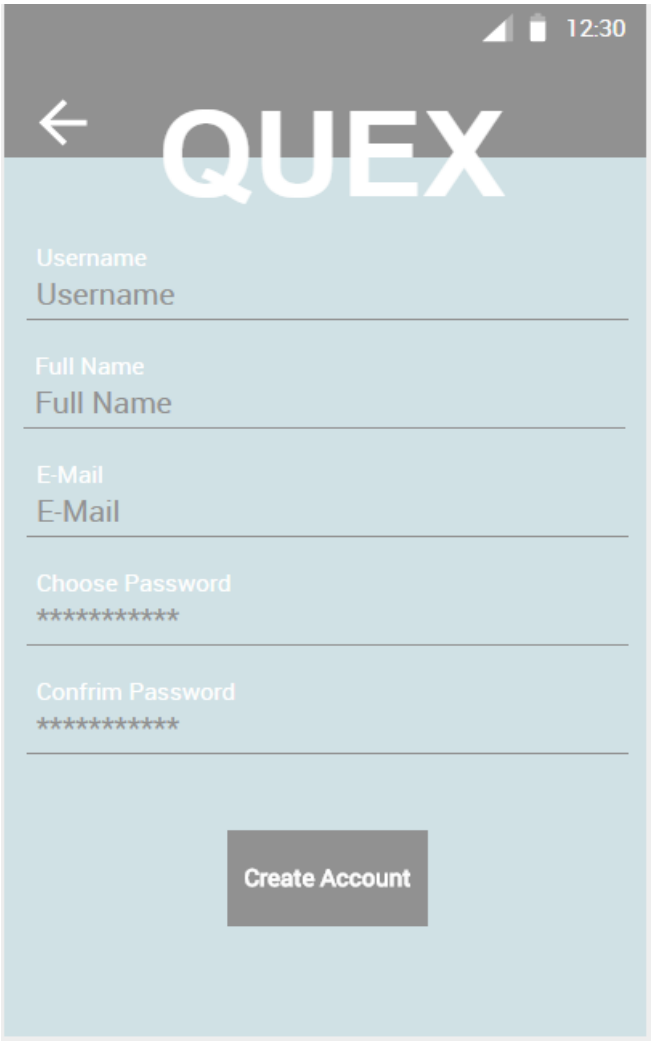


2.5.5 User Interface-Navigational Paths and Screen Mock-Ups

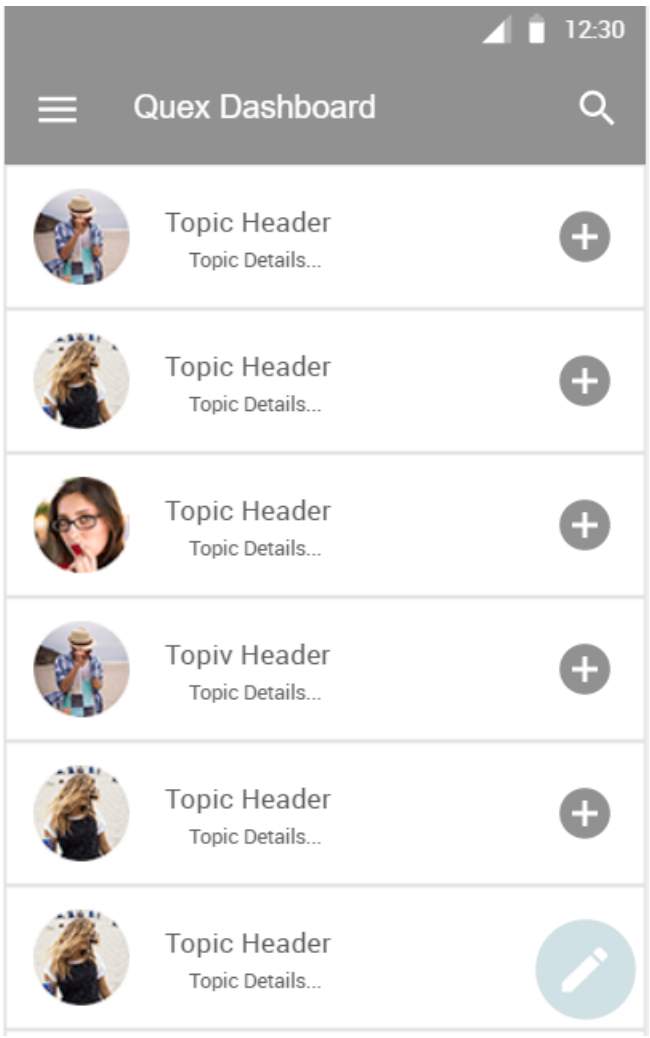
2.5.5.1 Screen Mock-Ups



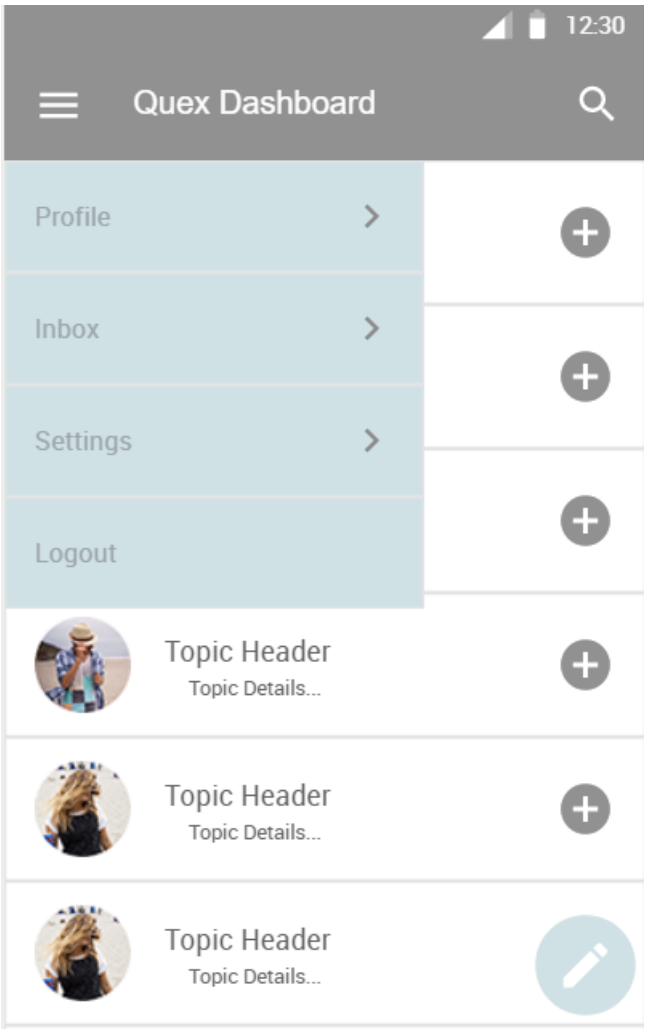
Quex provides a login screen. User can directly touch login with your existing username and password to access the program's features. If user do not have an account, Sign Up button should be used to create an account. Also this screen provides Sign Up via Google and Facebook account buttons.



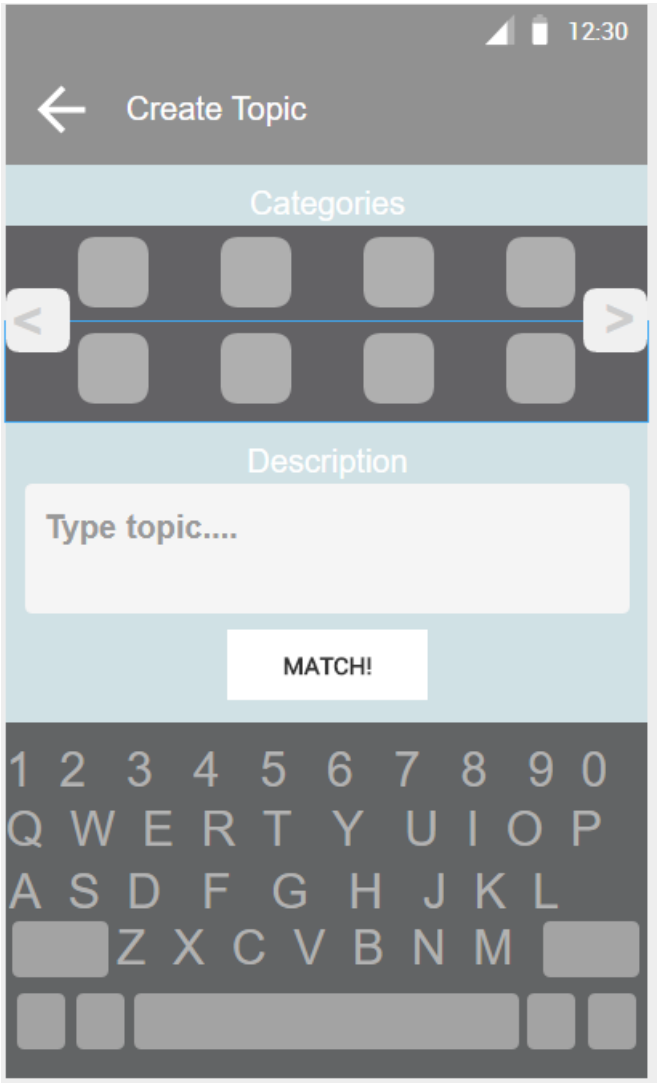
In create account section, user needs to write their username, full name, e-mail address, password and confirmation of the password in the corresponding text fields. Finally, user should touch on Create Account button. User can turn back with the back button which is placed in the left top of the screen.



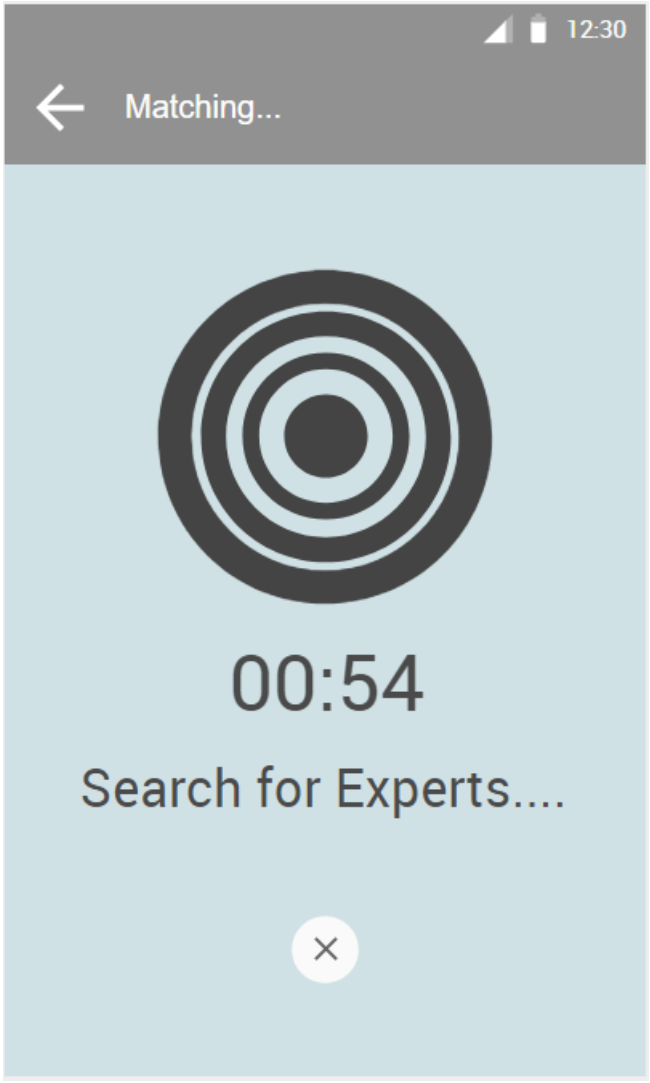
Dashboard screen has search, main menu and create topic buttons in it. User can search a specific word with use of search button on the right top, and user can reach main menu with the button on the left top. Also user can touch on topic headers to reach information about different problems of the people. Create Topic button is located at the right bottom of the screen.



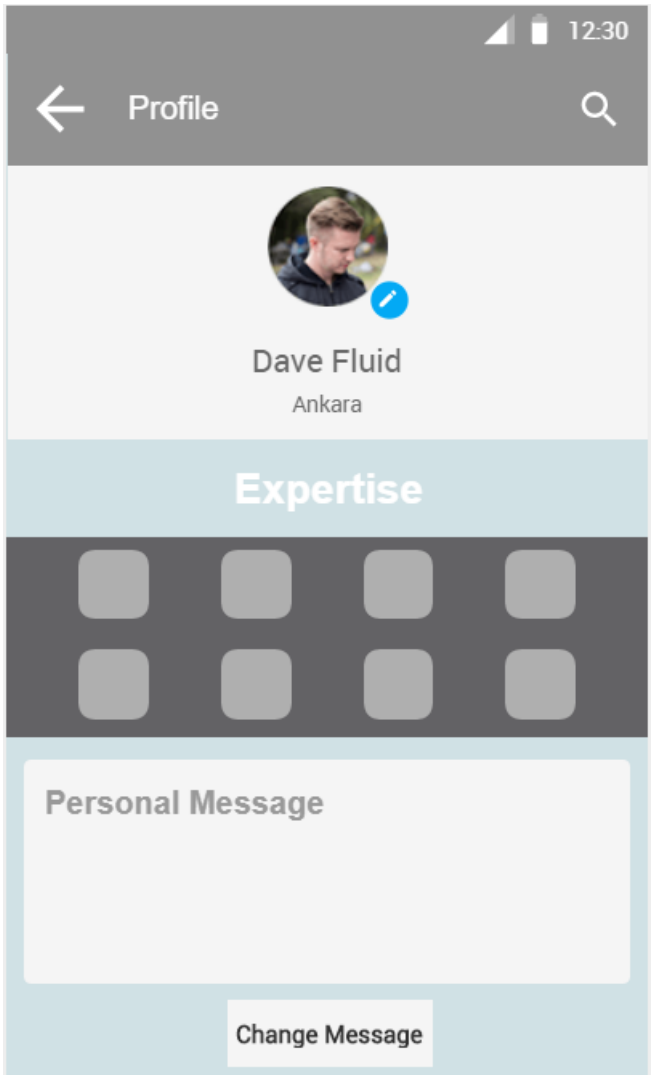
Profile button redirects user to the user profile screen. User also can touch Inbox button to reach chats with the other users. Settings and logout buttons are included in the main menu too. User should touch the main menu button again to close it.



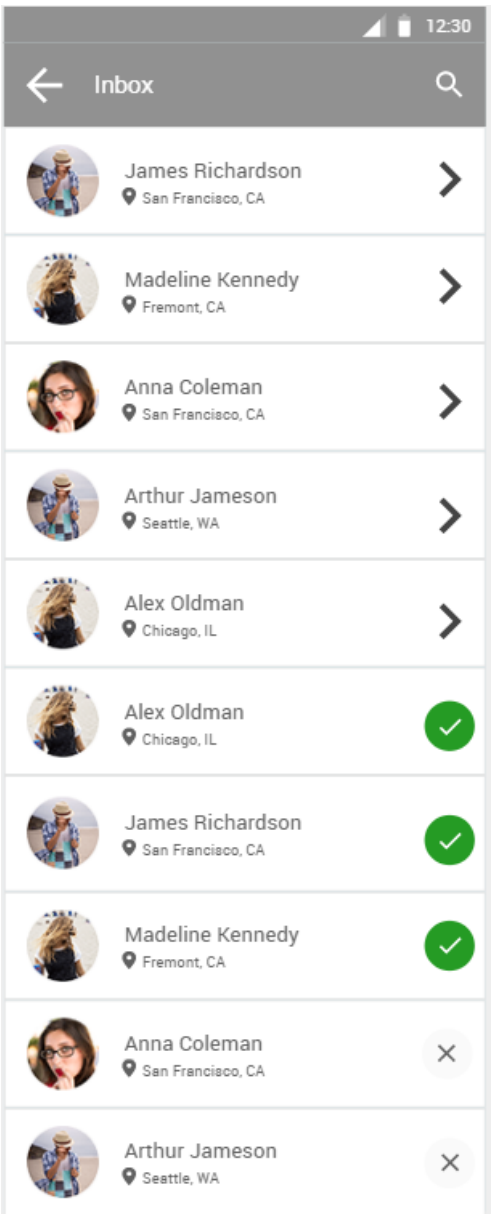
Create Topic screen have a category tab to determine the problem's detailed categorization. In the description text field, user can write detailed content of the problem. Match button redirect program to the matching screen. This screen has a back button on the left top too.



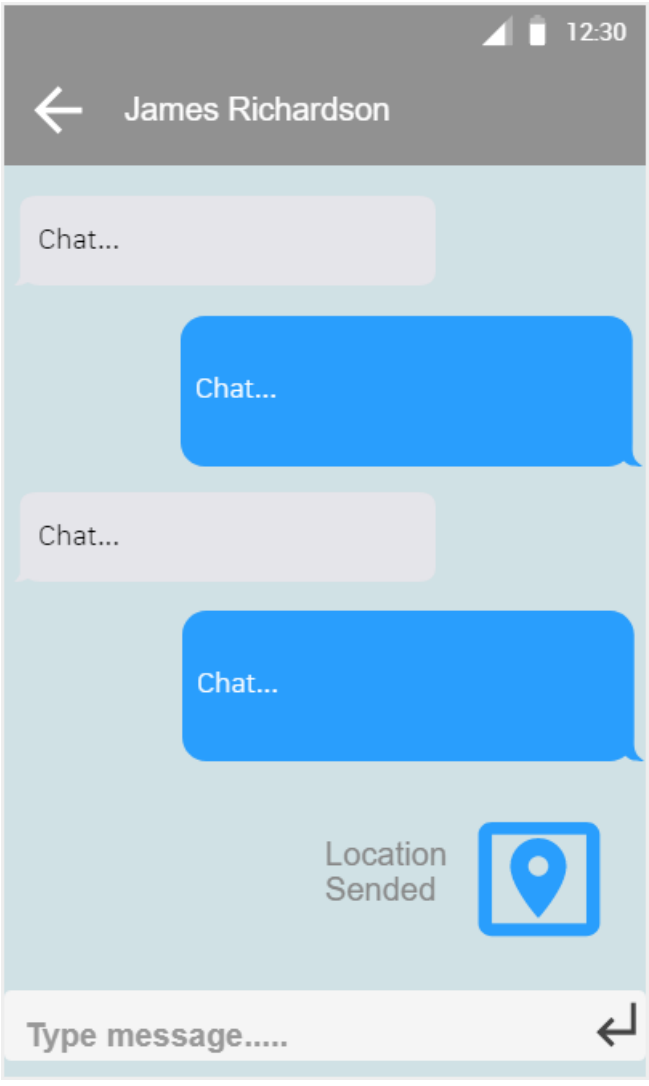
In matching screen there is a time counter which shows the elapsed time during the search process. With the “X” button, user can cancel the matching process and turn back to the create topic screen. Also back button can be used to turn back to the main dashboard screen while the matching process is continuing.



Profile screen has a change message button to change personal message of the user account. It can be seen at the bottom of the screen. User can touch on the profile picture to change it with an another picture in the phone. Blue pen icon is for changing password and email. Also, expertise buttons can be used to add or delete proficiencies to the user account.

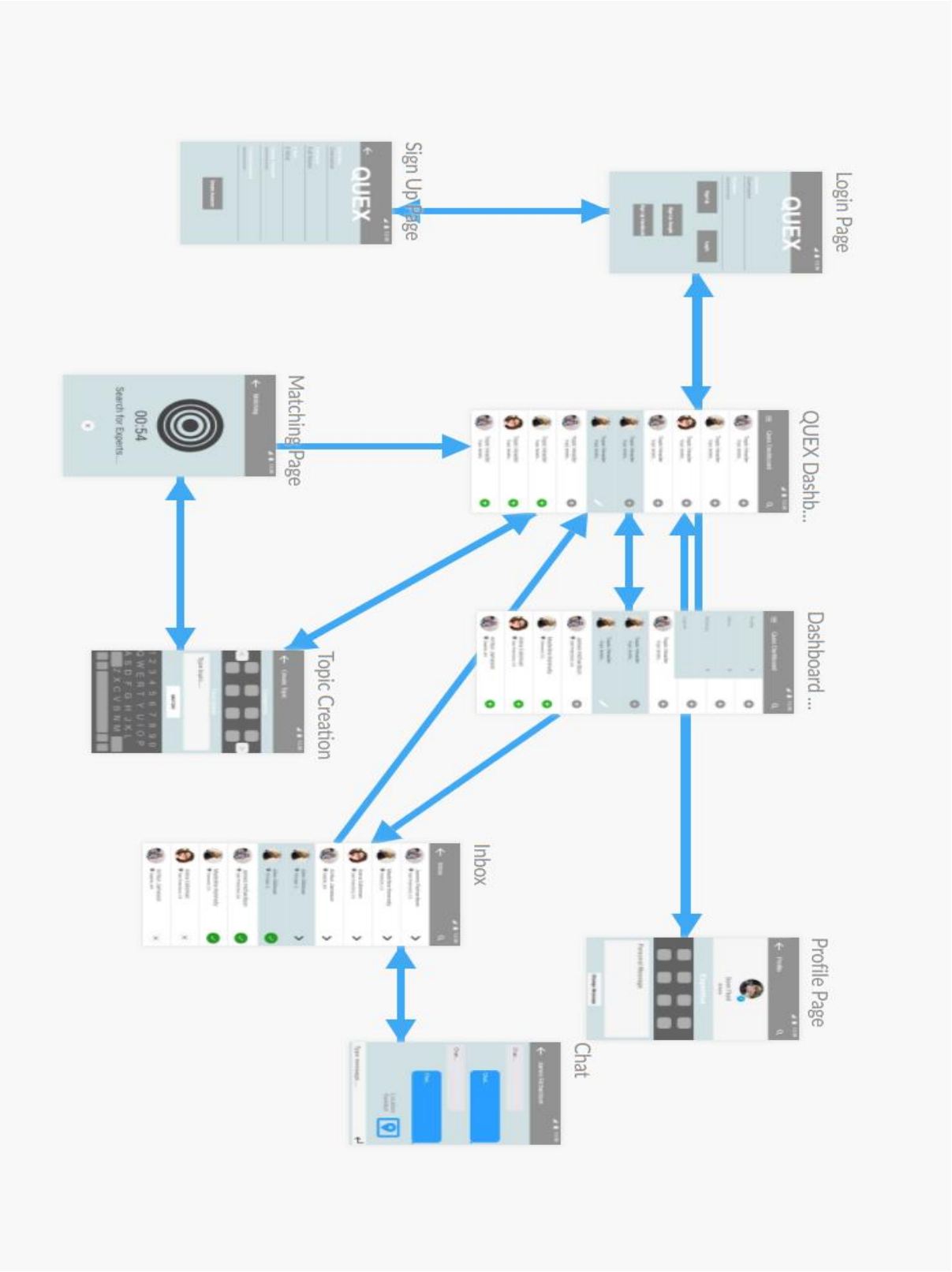


In the Inbox screen, there is a list of the existing chats with the other users. User can touch on them to write something to the others, or he/she may just look at the old chats. Search button can be used to search a keyword in the existing chats. Chats can be deleted or managed by switching them to the leftwards.



Chat screen provides a send location button next to the type message text field to send current location to another user if any need arises. Name of the user can be seen on the top of the screen. Messages of the original user can be seen on the right side of the chat screen, and messages of the other user can be seen on the left side. User can go back to the Inbox with the back button.

2.5.5.2 User Interface-Navigational Paths



3 References

[1] Android Studio, User Guide. Access: 05.11.2017
https://developer.android.com/studio/intro/index.html

[2] Visual Paradigm, User Guide. Access: 05.11.2017
https://www.visual-paradigm.com/support/documents/vpuserguide.jsp

[3] Fluid Ui, Demos. Access 05.11.2017
https://www.fluidui.com/demos/