Bilkent University
Department of Computer Engineering

Senior Design Project

Project short-name: EyeContact

Project Analysis Report

Melisa ONARAN – 21301232

Nazlı ABAZ – 21400231

Sarp SAATÇIOĞLU – 21400375

Yunus ÖLEZ - 21401539

Supervisor: Hamdi DİBEKLİOĞLU

Jury Members:

Çiğdem GÜNDÜZ DEMİR

Selim AKSOY

Innovation Expert: Çağla ÇIĞ KARAMAN

Website: eyecontact.in

Nov 06, 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

Improvement of every society depends on to the contribution of individuals in it. More people contributing to it with higher rates mean faster and bigger development. Governments endeavor to create better standards to increase production of individuals. However, individuals with disabilities do not have the same living conditions since they have special needs and thus, they cannot be as productive as others.

The aim of this project is to increase the living conditions of people who are visually impaired, and thus increase the development of societies. These individuals cannot get proper visually feedback from the environment which causes them to have a limited communication with societies. This project offers a solution for visually handicapped individuals that verbalizes the environment by using image processing using computer vision. Therefore, these people will be able to communicate, create and produce more.

2. Current system

During our research to find a similar product and technologies, we have discovered a few already in-use features, products and studies. Even though these mentioned technologies are similar, their aim differs from our product. While there some applications that help visually impaired people, the major applications that are similar to ours are Orcam, Horus, Be My Eyes and Emotion as a Service. Orcam is an application that helps visually impaired people through a camera that can be attached to the glasses. Through this camera, Orcam reads texts, recognizes street signs and faces, then through a small speaker gives feedback to the user [1]. Horus is a wearable assistant that recognizes objects and faces that has not yet been released [2]. Be My Eyes uses volunteers to help visually impaired through video chat. Volunteers answer the questions and describe the environment for the visually impaired [3]. Emotion as a Service is an application that analyzes emotional response through images and videos [4]. There are several more emotion sensing technologies such as Affectiva and Emovu [5, 6].

While there are many technologies as described above to help visually impaired people visualize their environment through audial feedback and detect emotions through images and videos, there is not a specific product that helps the communication and the socialization between 2 or more people by aural feedback through video chat that combines emotion detection, facial recognition and aural feedback.

3. Proposed System

The purpose of this project is to solve the human-interaction problem of visually impaired people in terms of non-verbal communication during video chatting. This problem is encountered by almost all visually impaired people. Hence they are visually impaired, they do not have a chance to use video chatting applications such as Skype, Google Hangouts, etc. We propose a system that assists visually impaired people during video chats.

The system is able to recognize the person whose face is detected by the camera of the user's computer or external camera while video chatting and tells the identity of the person verbally. Furthermore, it also helps during the conversation by notifying whether the person is looking directly to the user while he/she is talking or not. This will provide user to be more interactive in the conversation in terms of non-verbal communication. Moreover, the system recognizes some certain emotions expressed by whom the user is making video conversation with.

Camera that the system uses will be user's platform's camera. Verbal notifications will be output from the platform with respect to user's choice of output method (e.g. headphone, speaker, bluetooth devices etc.). This system is based on computer vision technology in order to recognize faces that user sees. The visual input taken from the camera is sent to the cloud platform so as to be processed and the verbal output is sent to the user's smartphone to be presented.

This system is available for cross-platform as a web application to be used by the users. Development of the user application will be done using React. Processing part will be handled with a web service on cloud. For computer vision, Python and OpenCV will be used running on cloud. Cloud computing will be handled with REST service methodology in order to be efficient.

3.1. Overview

EyeContact will help visually impaired people to video chat with others through audial feedback which informs the visually impaired person about the other participant’s emotions and gaze.

The user will first add their contacts to their contact list. Then they will be able to video call them through the easily used interface of EyeContact. The audial feedback will relate to the user whether the other participant is looking at them or somewhere else, what their contact is emoting.

3.2. Functional Requirements

- User will be able to communicate with others through application’s own video chatting feature.
- The application will detect human faces.
- User will be notified when other participants are looking towards them.
- When an unknown face is detected, the user will be able to add this face as a contact.
- The program will be able to recognize contacts by their faces and give a verbal feedback to the user.
- The program will determine emotions and also, give verbal feedback about it to the user.
- User will able communicate via text chatting, visually impaired people will use the built-in text-to-speech feature of the platform.

3.3. Nonfunctional Requirements

- **Compatibility:** For the facial detection, the environment must be well lit for the camera to detect humans.

- **Robustness:** The camera that is used must be high resolution and sensitive to detect facial changes in order to determine emotions. Unrelated objects should be overlooked.

- **Performance:** System should run efficiently. As the application will use computer vision on cloud, the system should be fast enough to give feedback simultaneously.

- **Usability:** The system will be implemented in such a way that visually impaired people will have a chance to video chat. In other words, the visually impaired people will gain equal footing while live chatting.

- **User-friendliness:** Interface should be user-friendly. It should be easy to use for a visually impaired person, even with ones who are not good at using computers. Sighted-people will also use this application to communicate with visually impaired. Therefore, the interface should be inviting and easy-to-use for both sighted and visually impaired.

- **Adaptability:** The system should be used cross-platform. Different computers have different cameras, so the application should be adaptable to be used with different cameras.

3.4. Pseudo requirements (Constraints)

3.4.1. Implementation Constraints

- The application will be a web application.
- Application part of the system will be developed based on React and JavaScript.
- Computer Vision part of the application will be done using OpenCV (Open Source Computer Vision) and Python.
- For the version control, GitHub and Git will be used in order to collaborate within our group.
- Cloud computing will be handled by use of third-party cloud platforms, Python and node.js.

3.4.2. Economic Constraints

- The application will be a web application which enables the user to sign in with their accounts.
- The libraries and technologies may require additional purchases.
- The application will require a computer with a high resolution or a computer with an external camera.

3.4.3. Ethical Constraints

- We will not share information with any third parties.
- Users will be able to access their account by signing in.

3.4.4. Sustainability Constraints

- System will work as a web application for video chatting. Therefore, it will be a web application. Hence, in the future, system may be approved to be able to work as a cross platform application.

3.4.5. Social Constraints

- Application will be released in English language.
- The system will be designed on the basis of making an opportunity to visually impaired people to live video chat.

3.4.6. Technical Constraints

- Internet connection is required for complete functionality.

3.5. System models

3.5.1. Scenarios

Use Case Name: Login

Actors: User

Entry Conditions:

- User runs the application
- User logs off from the system.

Exit Conditions:

- User successfully logs in to the system.

Main Flow of Events:

1. User opens the application
2. User types his information to the login screen
3. Authentication is successful and users logs in to the system

Alternative Flow of Events:

1. User types wrong user information. System asks again for his information.

Use Case Name: Add Contact

Actors: User

Entry Conditions:

- User adds a contact by typing the username.

Exit Conditions:

- The contact is found and he is already in the contact list.
- The contact is not found via username.

Main Flow of Events:

1. User press "+" button to add contact and types the username.

2. System checks the username to find that contact member.

3. System matches the username with a member of EyeContact.

4. System updates the user's contact list accordingly and notifies the user.

Alternative Flow of Events:

a1. When user types the username, system cannot recognize the username.

a2. System asks to the user from a list of possible contacts to choose in case username was typed wrong.

a3. User chooses the right contact username from the list.

a4. System adds the chosen user sends a request to add and updates the status, then notifies the user when accepted as a contact.

b1. When user types the username, system cannot recognize the username.

b2. The system cannot find any similar contact.

b3. The system asks the user what the username's detail of information is and tries to match with a member.

b4. The system updates the check list accordingly and notifies the user.

Use Case Name: Making a Call

Actors: User

Entry Conditions:

- A user is about to make a call to someone from his contact list, there are some conditions to be able to make the call, such as status. System checks for the status of the user who is about to be called.

Exit Conditions:

- User is notified via application that the status of the user who is taking the call is not online at the moment.

Main Flow of Events:

1. System checks the status of the user if available to take a call.
2. System confirms the status as online.
3. System allows the user the make the call.

Alternative Flow of Events:

1. System checks the status of the user if available to take a call.
2. System confirms the status as offline.
3. System does not allow the user to make the call and notifies.

Use Case Name: Delete Contact

Actors: User

Entry Conditions:

- User deletes a contact from his list.

Exit Conditions:

- Contact is deleted and contact list is updated.

Main Flow of Events:

1. User opens his contact list.
2. User deletes a contact.
3. System recognizes that username and removes.
4. System recognizes the removed contact.
5. The system updates the contact list and notifies the user.

Use Case Name: Create Contact List

Actors: User

Entry Conditions:

- User is at create contact list screen in application.
- User has proper authentication.

Exit Conditions:

- User creates the contact list successfully.
- User switches to main menu.

Main Flow of Events:

1. User enters to contact list menu
2. User types usernames and send request to add to contact list.
3. System saves the list when accepted, and it can be viewed later.

Use Case Name: Find User in Contact List

Actors: User

Entry Conditions:

- User goes to contact list and chooses a contact from alphabetically ordered list.
- User has a proper authentication.

Exit Conditions:

- User goes back to main menu.

Main Flow of Events:

1. User opens the contact list.
2. User chooses a contact from alphabetical order.
3. User makes an operation.
4. The system updates the made operation.

Alternative Flow of Events:

1. User opens the contact list.
2. User cannot find the contact from alphabetical order because user is either deleted or does not have an account anymore.

Use Case Name: Suggest Contact to Add

Actors: System

Entry Conditions:

- User has a valid authentication
- User navigates to “suggest contact” menu

Exit Conditions:

- User leaves the “suggest contact” menu

Main Flow of Events:

1. User enters to suggest contact menu
2. System checks the list and compare it with possible contacts in the database.
3. The system makes a suggestion list.
4. User selects a contact.
5. The system shows the status or request and sends a notification to the chosen contact.

Alternative Flow of Events:

1. User enters to suggest contact menu.
2. System checks the list and compare it with possible contacts in the database.
3. The system makes a suggestion list.
4. User does not select a contact.

Use Case Name: End Call

Actors: User

Entry Conditions:

- User presses end call button to end the call.
- User has a valid authentication.

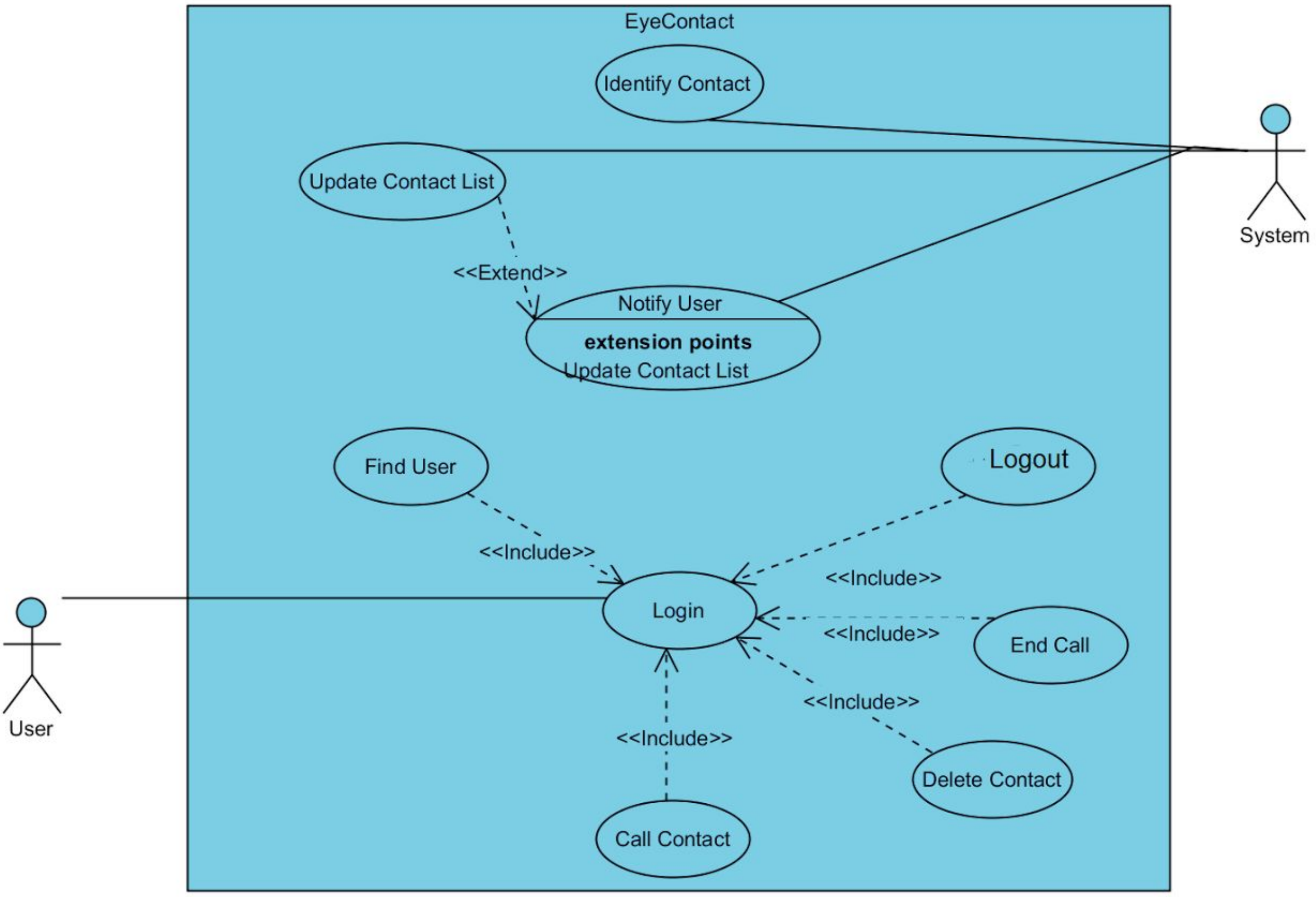
Exit Conditions:

- User leaves the call page.

Main Flow of Events:

1. User presses to the button to end the call.
2. System displays a notification to confirm that the call is ended.

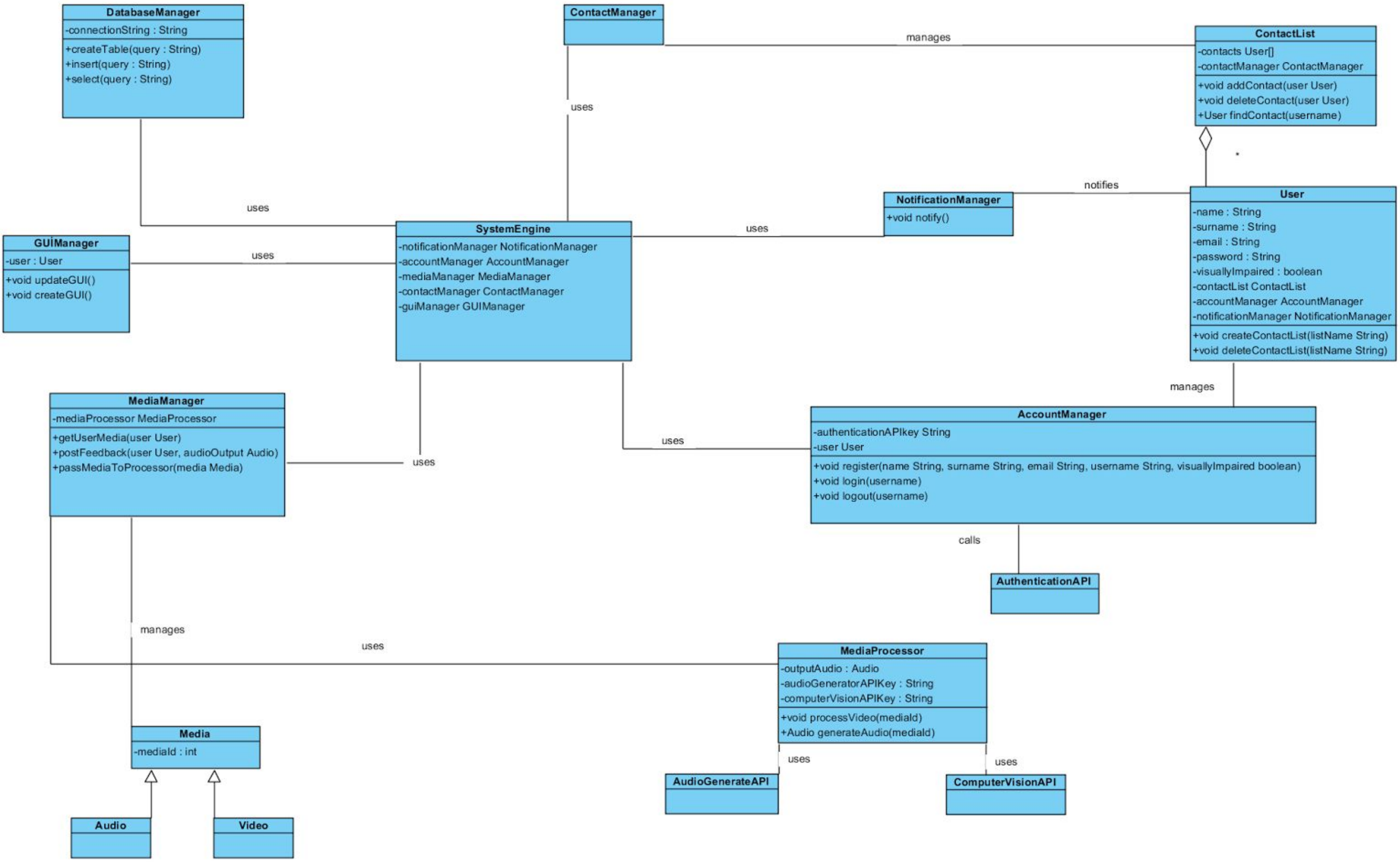
3.5.2. Use Case Model



The user at first needs to login to the system. Then the user can do the following; the user can find other users, call his/her contacts, delete the contacts, call a contact and end the call. System does the following; system identifies the contacts, update contact lists and notify the user.

3.5.3. Object and Class Model

Class Diagram


DatabaseManager
-connectionString : String
+createTable(query : String)
+insert(query : String)
+select(query : String)
ContactManager
manages
ContactList
-contacts User[]
-contactManager ContactManager
+void addContact(user User)
+void deleteContact(user User)
+User findContact(username)
uses
*
notifies
NotificationManager
+void notify()
uses
User
-name : String
-surname : String
-email : String
-password : String
-visuallyImpaired : boolean
-contactList ContactList
-accountManager AccountManager
-notificationManager NotificationManager
+void createContactList(listName String)
+void deleteContactList(listName String)
uses
GUIManager
-user : User
+void updateGUI()
+void createGUI()
uses
SystemEngine
-notificationManager NotificationManager
-accountManager AccountManager
-mediaManager MediaManager
-contactManager ContactManager
-guiManager GUIManager
manages
MediaManager
-mediaProcessor MediaProcessor
+getUserMedia(user User)
+postFeedback(user User, audioOutput Audio)
+passMediaToProcessor(media Media)
uses
uses
AccountManager
-authenticationAPIkey String
-user User
+void register(name String, surname String, email String, username String, visuallyImpaired boolean)
+void login(username)
+void logout(username)
calls
AuthenticationAPI
manages
uses
MediaProcessor
-outputAudio : Audio
-audioGeneratorAPIKey : String
-computerVisionAPIKey : String
+void processVideo(mediaId)
+Audio generateAudio(mediaId)
Media
-mediaId : int
uses
uses
AudioGenerateAPI
ComputerVisionAPI
Audio
Video

User Class

- name String
- surname String
- email String
- password String
- username String
- visuallyImpaired boolean
- contactList ContactList
- accountManager AccountManager
- notificationManager NotificationManager

+ void createContactList(listName String) : A contact list is created.
+ void deleteContactList(listName String) : Specified contact list is deleted.

AccountManager Class

- user User,
- authenticationAPIkey String

+ void register(name String, surname String, email String, username String, visuallyImpaired boolean) : User register with the specified parameters.
+ void login(username) : User can login via their username.
+ void logout(username) : User can logout.

NotificationManager Class

+ Void notify()

ContactList Class

- contacts User[]
- contactManager ContactManager

+ void addContact(user User) : A contact is added to the user’s contact list.
+ void deleteContact(user User) : Specified contact is deleted from the user’s contact list.
+ User findContact(username) : Specified contact is found through their username.

SystemEngine Class

- notificationManager NotificationManager,
- accountManager AccountManager,
- mediaManager MediaManager,
- contactManager ContactManager,
- guiManager GUIManager,
- databaseManager DatabaseManager

DatabaseManager Class

- connectionString String
+ void createTable(query String) : Database implementation in the program.
+ void insert(query String) : Database implementation in the program.
+ void delete(query String) : Database implementation in the program.
+ select(query String) : Database implementation in the program.

MediaManager Class

- mediaProcessor MediaProcessor
+ getUserMedia(user User) : Returns user’s video.
+ postFeedback(user User, audioOutput Audio) : Returns audial feedback to the user.
+ passMediaToProcessor(media Media) : Passes the above mentioned video to the media processor.

Media Class

- mediaId int

MediaProcessor Class

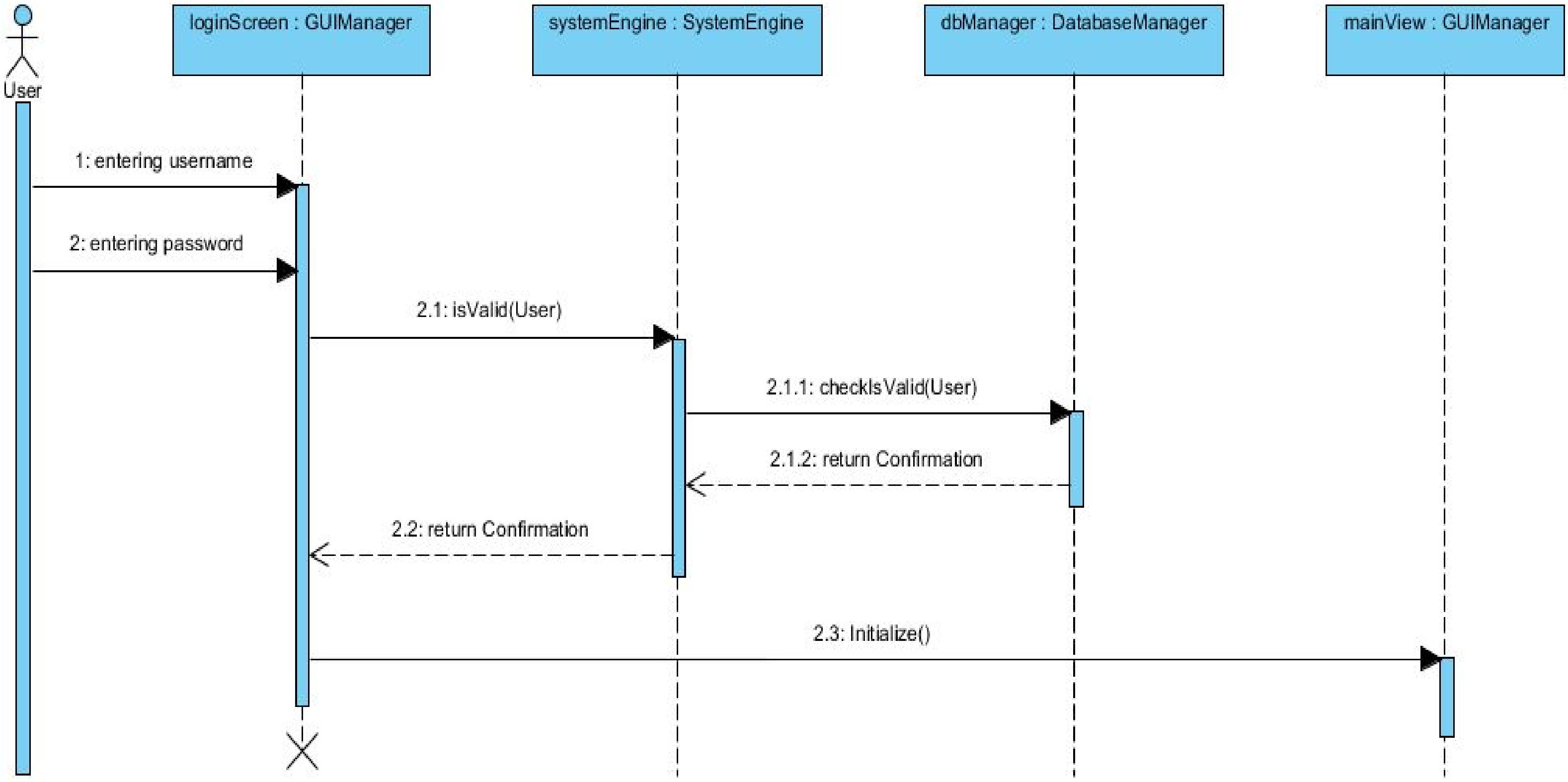
- outputAudio Audio
- audioGeneratorAPIkey String
- ComputerVisionAPIkey String
+ void processVideo(mediaId) : Process the video.
+ Audio generateAudio(mediaId) : Generates the audial feedback.

<u>GUIManager</u>

- - user User
- + void createGUI() : Creates the graphical interface.
- + void updateGUI() : Changes the graphical interface.

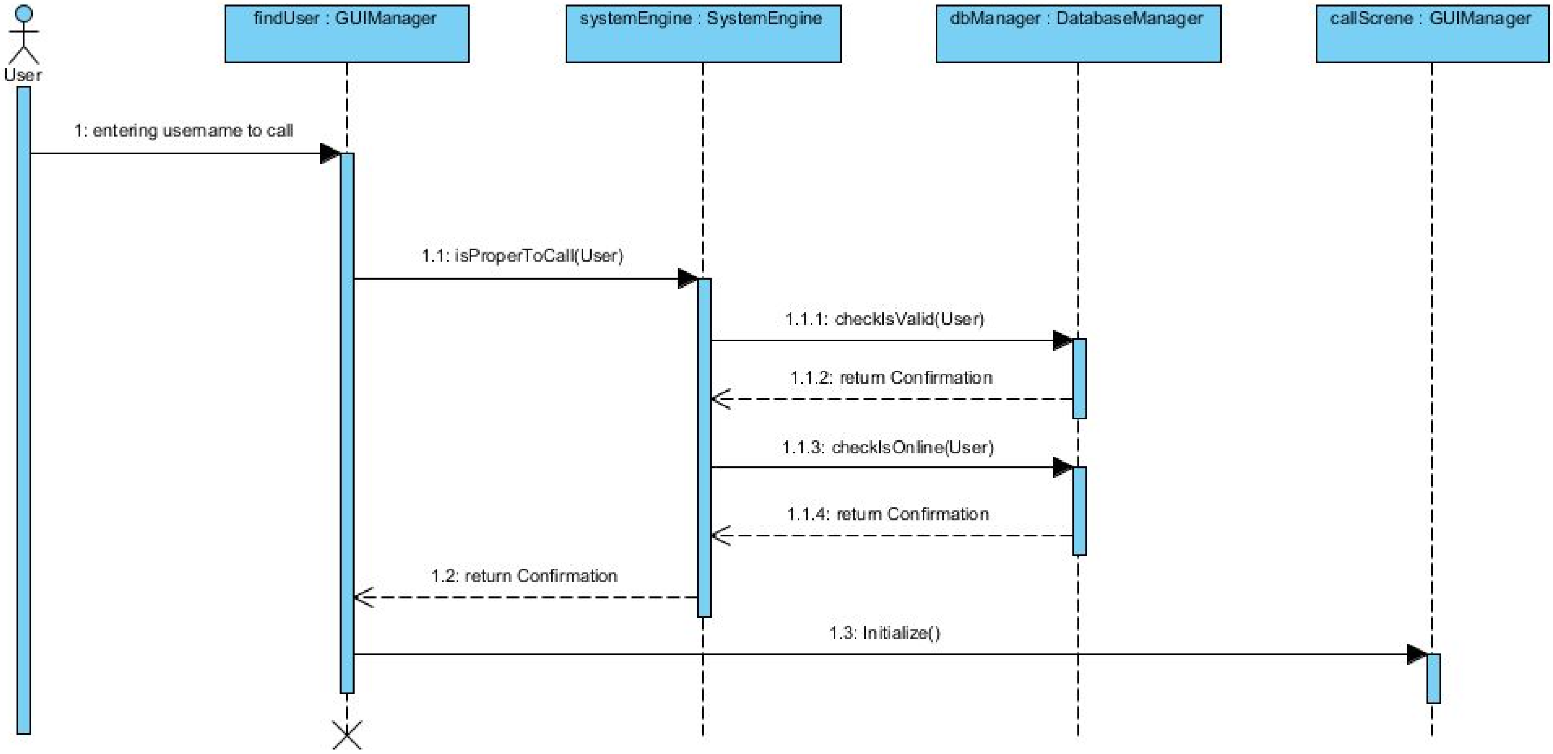
3.5.4. Dynamic Models

3.5.4.1. Login Sequence Diagram (Success)



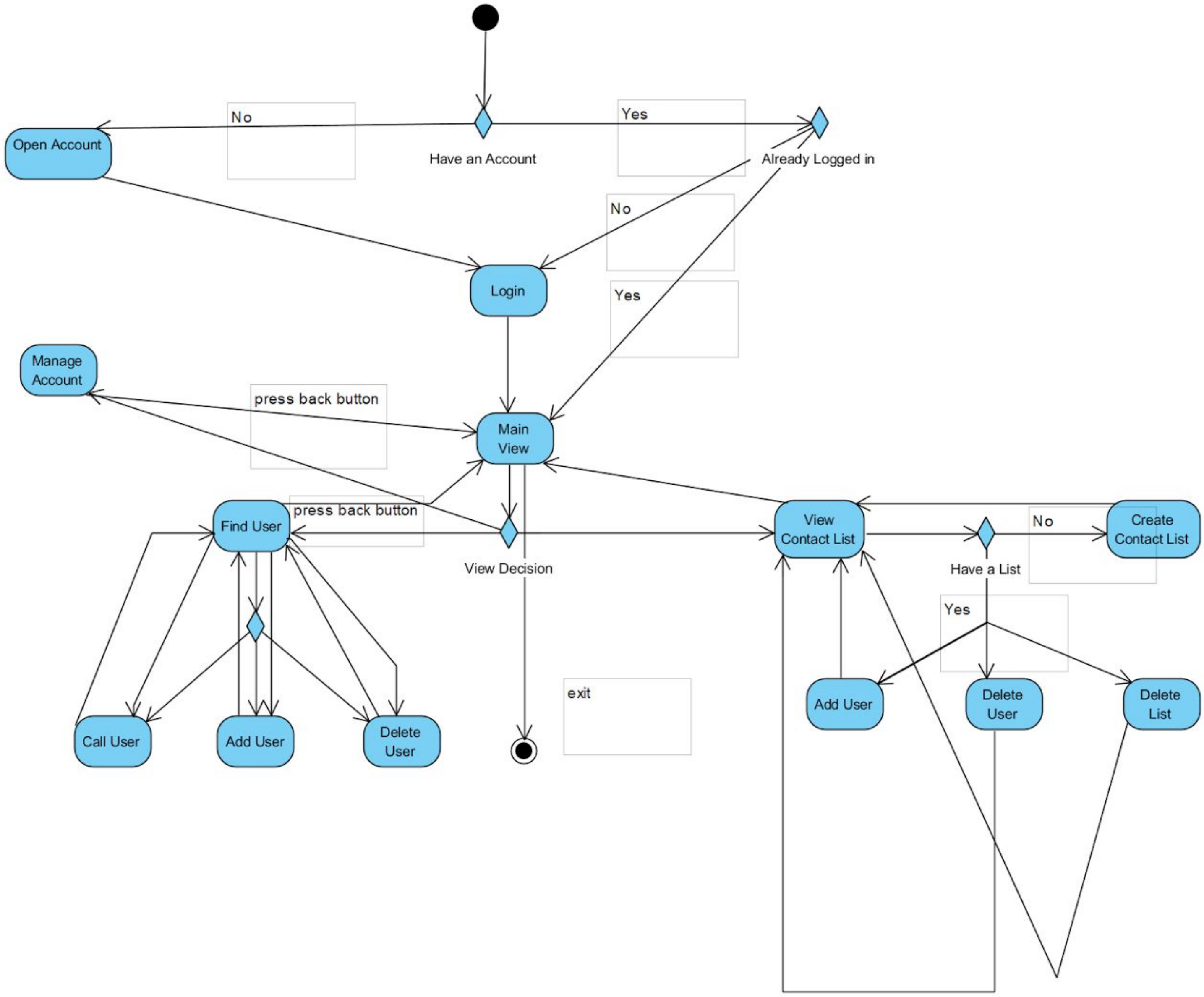
In order a user to login to the system, one must first enter his/her username and password. The Graphical User Interface Manager (GUIManager) then sends the entered data to the System Engine (SystemEngine). After that, the System Engine sends a task to the Database Manger (dbManager) to check whether there exists a user with the entered data in the system or not. Lastly, Database Manager sends returns the resulting Boolean to System Engine and it to the GUI Manager. Since this scenario results successfully, the GUI Manager creates a mainView and deletes loginScreen.

3.5.4.2. Make Call Sequence Diagram (Success)


User
findUser : GUIManager
systemEngine : SystemEngine
dbManager : DatabaseManager
callScrene : GUIManager
1: entering username to call
1.1: isProperToCall(User)
1.1.1: checkIsValid(User)
1.1.2: return Confirmation
1.1.3: checkIsOnline(User)
1.1.4: return Confirmation
1.2: return Confirmation
1.3: Initialize()

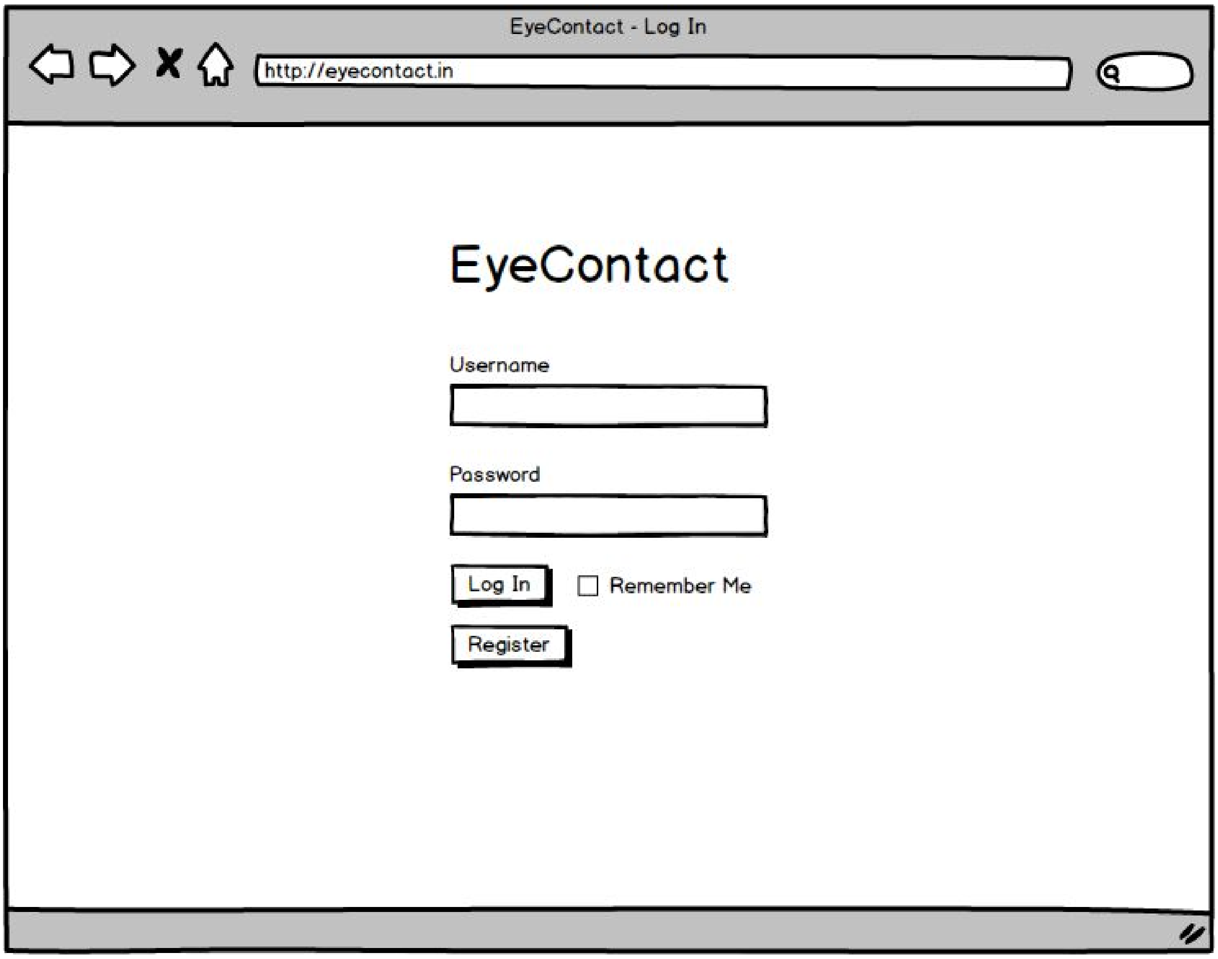
To make a proper call in the proposed system, the user must first find the one who he/she wishes to call. In order to do so the user must enter the username who he/she wishes to call in the findUser Screen. Then the GUI Manager will send the information to the Database Manager via the System Engine. The Database Manager will search and find two Booleans for the System Engine: one for whether the given username is valid or not, and one for whether the given user is online or not. If both of the Booleans return true, which is the case in this scenario, the System Engine will allow the user to make the call.

3.5.4.3. Activity Diagram



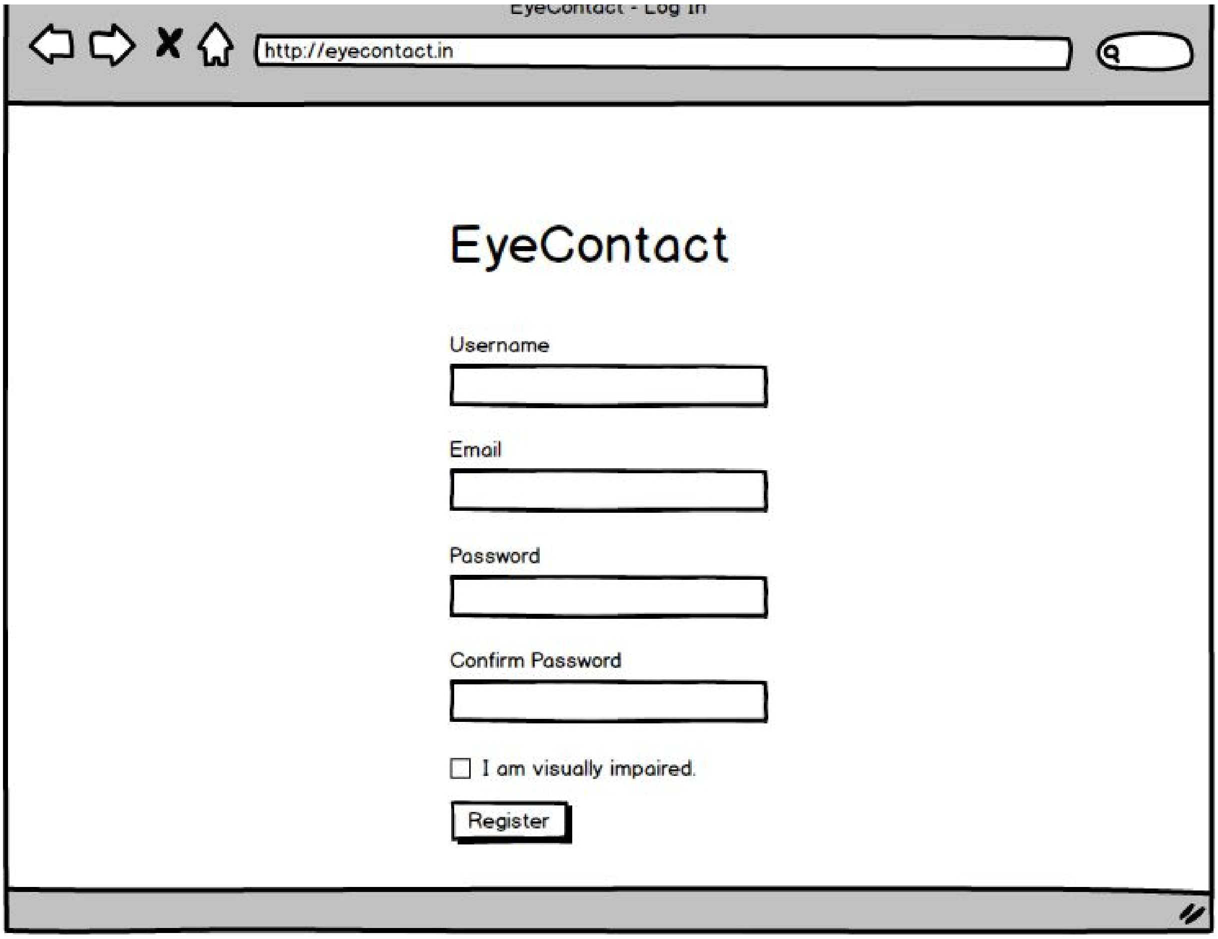
3.5.5. User Interface

3.5.5.1. Login



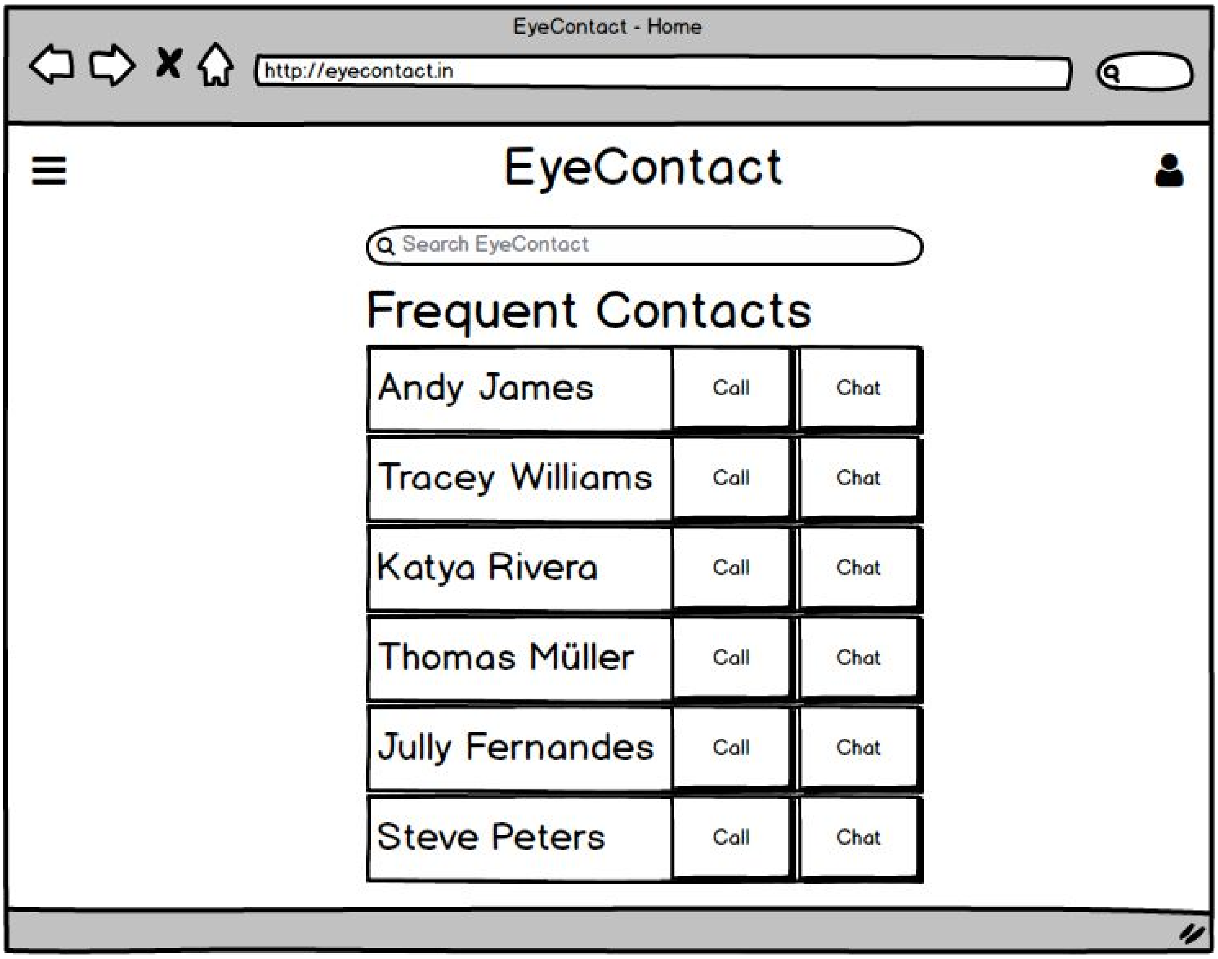
Log In screen for user to log in or register if a newcomer.

3.5.5.2. Register



Register screen for user to register to the system with indicating whether he/she is visually impaired or not.

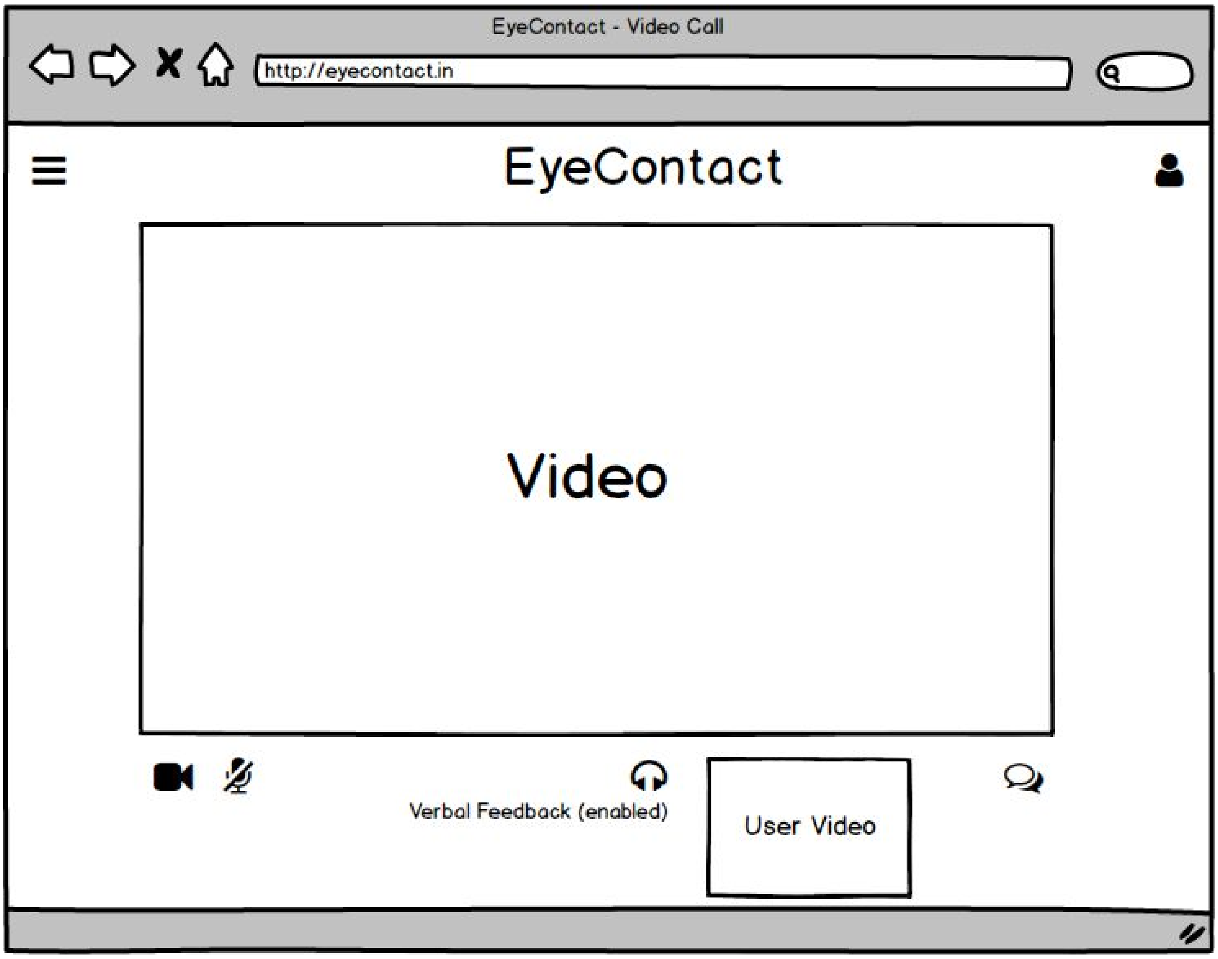
3.5.5.3. Home Screen



Andy James	Call	Chat
Tracey Williams	Call	Chat
Katya Rivera	Call	Chat
Thomas Müller	Call	Chat
Jully Fernandes	Call	Chat
Steve Peters	Call	Chat

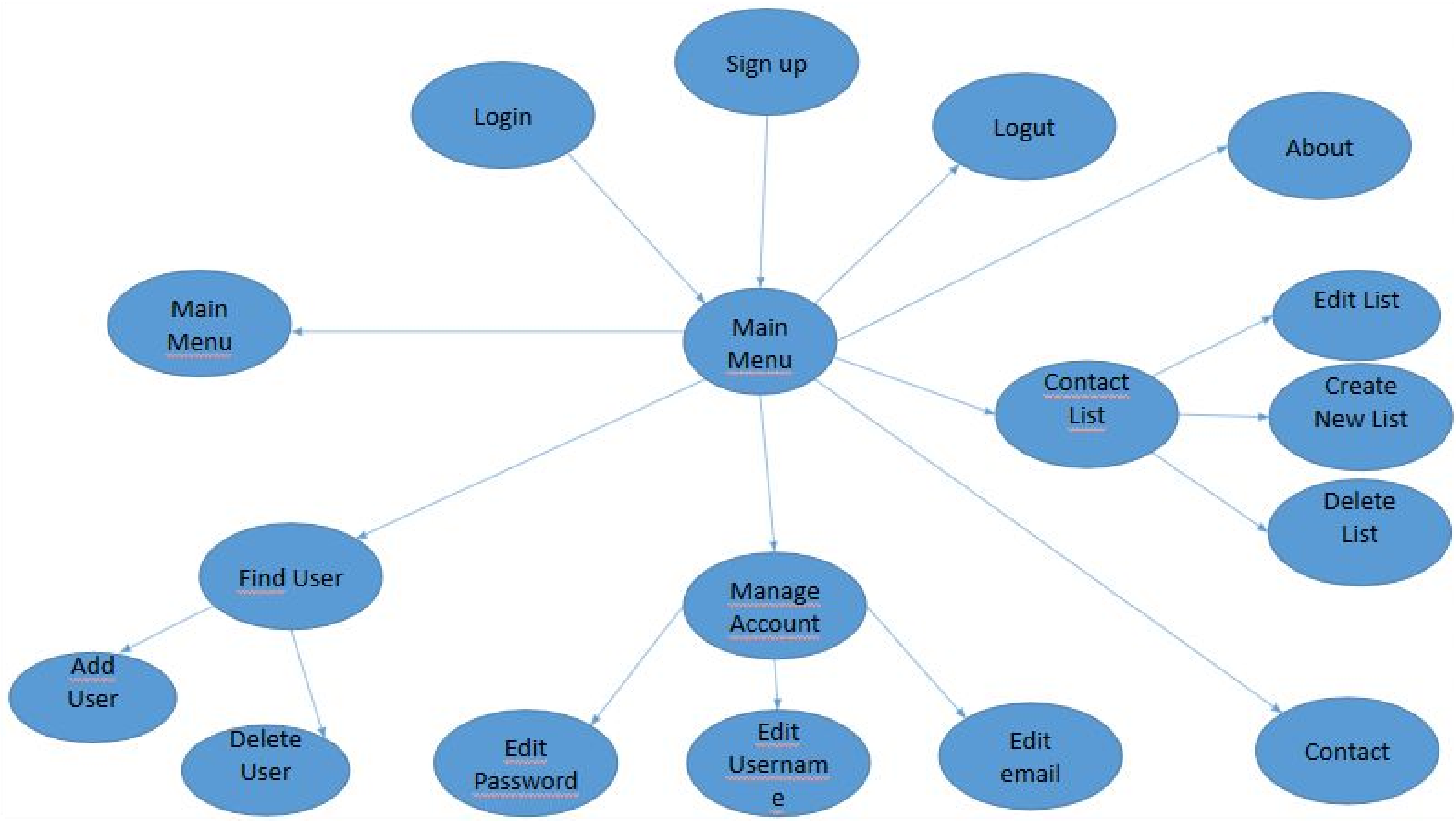
Home screen consists of frequent contacts and options to call them or chat with them immediately. On top left corner a menu icon directs user for more options to navigate. User button on top right corner navigates to more options about EyeContact and user profile.

3.5.5.4. Video Call



Video call screen consists of a video of the person you are communicating with, a video camera toggle button, a microphone toggle button, a small video of the user, a chat button to start chat with same person immediately and a verbal feedback toggle button.

3.5.6 Navigational Path



4. References

[1] “Making the World Accessible: The Latest Tech for the Blind and Visually Impaired” https://www.mytherapyapp.com/blog/tech-for-the-blind-and-visually-impaired. [Accessed: Oct 09, 2017].

[2] “Horus Tech” https://horus.tech/?l=en_us. [Accessed: Oct 09, 2017].

[3]”The App That Lets Users Lend Their Eyes, And Blind People See Things In A New Way” https://www.fastcompany.com/3041054/the-app-that-lets-users-lend-their-eyes-and-blind-people-see-things-in-a-new-way. [Accessed: Oct 09, 2017].

[4] “Emotion as a Service” https://www.affectiva.com/product/emotion-as-a-service/. [Accessed: Oct 09, 2017].

[5] “Affectiva: How It Works” https://www.affectiva.com/#how-it-works. [Accessed: Oct 09, 2017].

[6] “Emovu” http://emovu.com/e/technology/. [Accessed: Oct 09, 2017].