CS-223 SOFTWARE REQUIREMENTS SPECIFICATION

for

Project 7 Classroom Visualisation App-1

Prepared by:
Group-12
Mitansh Jain - 160101042
Sujoy Ghosh - 160101073
Akul Agrawal - 160101085

February 13, 2018

Contents

1 Introduction

1.1 Purpose

In this Document, we describe the software requirements for an android application, named as "Classroom Visualiser". The purpose of this document is to give a detailed description of the requirements for the application. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications. This SRS covers how the entire application, from front end graphical user interface to the back end of the product, recognizes the student and maps them to their assigned state.

1.2 Document Conventions

The format specified by IEEE was followed while creating this document. The headings of each topic are in bold. Bullet points are used wherever required.

1.3 Intended Audience and Reading Suggestions

This Software Requirements Document is intended for:

- Developers who can review project's capabilities and more easily understand where their efforts should be targeted to improve or add more features to it (design and code the application it sets the guidelines for future development).
- Project testers can use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. This way testing becomes more methodically organized .
- End users of this application who wish to read about what this application can do.

In the next section, we discuss about Product perspective, Product functions, User Classes and Characteristics and Operating environment. Then the external interface requirements highlighting the logical characteristics of each interface between the software product and the users are discussed, followed by system features with their functional requirements are presented to highlight the major services provided by the intended product.Finally, this specification is concluded with the Nonfunctional Requirements and reference documents on which this document is based on.

1.4 Project Scope

The purpose of this project is to create convenient and easy-to-use android app for users. This app will enable professors to recognise the students present in the class and map them to their assigned states and augment a bounding box of particular color. The architecture of the app will allow it to be compatible to additional functionality for dynamic state allocation.

The system is based on computer vision. Above all, we hope to provide a comfortable user experience along with best results.

1.5 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998

2 Overall Description

2.1 Product Perspective

This android application is mainly intended to serve the lecturers. The main goal behind developing this application was to develop a mechanism to make the lecturer aware of attentiveness of the students. Due to short span of development-time, the states of the students are assigned randomly. The app will recognise the students present in class at any instant of time and map them to their assigned states by augmenting a bounding box based on state value. Though the target audience includes lecturers, it would also be open to use by other presenters. This application will help organisers to recognise people at some event and determine their states.

2.2 Product Functions

Functions included in the final product will be as follows:

- Image recognition and state allocation
- Student addition
- Student deletion
- Editing student data
- Sign-up
- Login
- Logout

2.3 User Classes and Characteristics

The intended users for the product should have the following characteristics:

- Intended users are mainly lecturers, but the application is suitable for any presenters with a known audience.
- The users should be able to afford and use an Android device with specified specifications.

- The users should understand the functioning and operation of the software on a basic level.
- The users should be able to understand the English Language to operate the application.

2.4 Operating Environment

The application will work in any Android based smart-phone with version 6.0 and above. It will require at least 5MP camera.

2.5 Assumptions and Dependencies

The main functioning of the application depends on accuracy of OpenCV libraries in facial recognition which is the main building block of result evaluation. Android and OpenCV are open source leading our application to be free of cost. Performance of application will depend on android version and hardware architecture.

3 External Interface Requirements

3.1 User Interfaces

The application would have a user friendly interface and users of every age group can operate the app easily. App would start with a login screen where user will have to provide credentials to access portal.

Portal would contain dedicated buttons to following:

- Add new student
- Edit existing students:

  For adding new students and editing existing students on screen keyboard.
- Display list of existing students
- Opening camera view and recognizing students.

  The app will recognize students in real time and augment a bounding box to each of their faces of a specific color according to assigned states.
- A logout button to exit portal.

3.2 Hardware Interfaces

A typical android smart-phone with the basic peripherals (Touch Screen, Rear camera) is needed to run and have full control of the product. Further access to storage is required to store training images for recognizing the individuals.

This app will require 400MB of RAM and 100MB free space and above as number of students increases.

The final developed product will be usable by any Android with :

- Android 6.0 (Marshmallow) or API level 23 with least recommended resolution of 1280x720.
- Minimum RAM size : 400 MB
- Least version of CPU: Dual Core 1.2 GHz
- Minimum required Space : 300 MB
- Video : 720p@30fps
- Rear Camera: Least specs -> 5.0 MP

3.3 Software Interfaces

This system can operate on Android versions 6.0 and above. We shall use OpenCV, namely Open Source Computer Vision, to recognise faces using training dataset of atleast 20 images of a person. Android Studion will be our IDE(Integrated Development Environment), which includes comprehensive set of development tools. Its libraries will help us augment bounding box around faces, according to its state.

4 Functional Requirements

This section will describe in detail all the features in the previous section.

4.1 Image recognition and state allocation

- **Input:** Real time video
- **Output:** Bounding boxes augmented in real time according to allocated states
- **Description:** This function will recognize the faces of the students and assign random states to each student every 5 minutes. Bounding boxes of the color corresponding to the allocated states will be augmented to the recognized faces in real time.

4.1.1 Face detection and recognition

- **Input:** Real time video
- **Output:** Coordinates of faces in the video and unique student ID corresponding to each face
- **Description:** This function will process the real-time video and detect the faces. Also, it will recognize the student corresponding to the detected faces.

4.1.1.1 Video capture

- **Input:** View from rear camera
- **Output:** Captured video in processable OpenCV format
- **Description:** In this function, video will be taken from rear camera and stored in such a format that it can be further used for processing and producing desired results.

4.1.1.2 Video to image frames conversion

- **Input:** Video in a processable OpenCV format
- **Output:** Image frames corresponding to the input video
- **Description:** This function will convert the input video into a number of image frames in sequential order which will act as raw data for further processing.

4.1.1.3 Face recognition

- **Input:** Image frames corresponding to the real-time video
- **Output:** Recognized faces and their coordinates, along with their corresponding student ID
- **Description:** This function will detect the location of faces from the image frames and find the student corresponding to each face.

4.1.2 Random state generator

- **Input:** Time of last random state generation
- **Output:** Updated random states along with updated time of the last random state generation as current time, if last random state was generated more than 5 minutes before.
- **Description:** This function will generate random states corresponding to each student and update the current states of the students if the time of the last update is found to be more than 5 minutes ago. Also, the time of the last generation of random states would be updated if the random states are updated.

4.1.3 Bounding box augmentation with given states

- **Input:** Student IDs of the students whose face has been recognized along with the coordinates of their corresponding faces
- **Output:** Augmented bounding boxes in real time
- **Description:** This function finds the color of the bounding box corresponding to the assigned states of the students whose face has been recognized, and augments a bounding box of that color around the recognized faces.

4.1.3.1 State-color mapper

- **Input:** Student IDs of the students whose face has been recognized
- **Output:** Allocated colors of the bounding box corresponding to each recognized student
- **Description:** This function finds the colors of the bounding boxes corresponding to the assigned states of the recognized students.

4.1.3.2 Bounding box augmentation with given colors

- **Input:** Details of recognized students along with coordinates of the recognized faces and corresponding color of augmentation boxes
- **Output:** Augmented bounding box in real time.
- **Description:** This function augments the bounding box around the recognized faces with the corresponding color, along with the details of the student

4.1.4 Flash light toggle

- **Input:** Instance trigger.
- **Output:** Toggles flash light.
- **Description:** This turns on flash light in times of bad lighting in room or turns it off if it was on.

4.2 Student addition

- **Input:** Student Data
- **Output:** Popup confirmation of addition of student details
- **Description:** Function to insert details of the student into database . Details include name Student ID and 20 different images of the student. The details and corresponding images are stored in the database.

4.2.1 Student details addition

- **Input:** Student details
- **Output:** Popup confirmation of adding of new student in database with given specifications
- **Description:** Function to take the name and student ID of the student to be added. New entry in database is created with the entered details.

4.2.2 Student images addition

- **Input:** Student ID and images
- **Output:** Popup confirmation of updating the new student database with the 20 images
- **Description:** Function to take 20 different images of the student to be added. The images are stored in the database corresponding to the student ID.

4.2.2.1 Student images addition from gallery

- **Input:** 20 different images of the student to be added from the gallery and student ID of the new student to be added
- **Output:** Popup confirmation of updating the new student database with the 20 images
- **Description:** Function to take 20 different images of the student corresponding with the given roll number from the gallery. The images are stored in the database corresponding to the unique roll number.

4.2.2.2 Student images addition from camera

- **Input:** Instance trigger
- **Output:** Popup confirmation of updating the new student database if number of images is 20 and popup of error message if number of images is less than 20.
- **Description:** Function to take the roll number of the new student and open the camera to take his 20 different images. If the number of images is less than 20, no database is formed and user is prompted to add more images with the corresponding student ID. Otherwise, The images are stored in the database corresponding to the unique student ID.

4.3 Student deletion

- **Input:** Student ID of the student to be removed from the database
- **Output:** Popup confirmation of deletion of student details with the updated database
- **Description:** Function to take unique college id of the student to be deleted. If a student with the corresponding Student ID is found in the database, it is removed from the database.

4.4 Student list display

- **Input:** Instance trigger
- **Output:** Details of the students displayed on the screen
- **Description:** Function that takes the database of the added students and displays the name and student ID of all the added students along with an image of each student.

4.5 Edit student

- **Input:** Student ID along with data to updated
- **Output:** Popup with confirmation of updated student data
- **Description:** This function will update student data in the database with specified input.

4.5.1 Editing student details

- **Input:** New Details of the specified student
- **Output:** Popup with confirmation of updated student details in the database
- **Description:** This function will update student details such as some correction in name or other details.

4.5.2 Retraining face recognition model for a student

- **Input:** New images of specified student.
- **Output:** Updated trained model of face recognition, trained according to new images.
- **Description:** This function can be used if image recognition is not working properly for a student due to bad photos being used for training. Face recognition model can be retrained according to new better

4.6 Sign-up

- **Input:** User credentials and other details
- **Output:** Sign-up Confirmation
- **Description:** This function will enable user to receive authorization to use the app so that his database is safe from others' interference.

4.7 Login

- **Input:** User login credentials
- **Output:**Two Possible outputs:
    - Access to portal if credentials are correct.
    - Wrong credentials message

- **Description:** This function will take user credentials as input and will give access to the portal if credentials are correct, otherwise display the wrong credentials message.

4.8 Logout

- **Input:** Instance trigger
- **Output:** Logging out from the users' account and appearance of the login screen
- **Description:** This function will close the portal and bring app to the login screen.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

The performance of the product shall depend on the quality of the video captured. The product shall take initial load time depending on OpenCV libraries to load. The performance of the product shall depend on the hardware components of the client's device.

Video bitrate maybe high and OpenCV might take high CPU usage and hence app would run smoothly on high end phones.

5.2 Security

The system will use a login system for authentication and thus will be highly secure and will prevent any type of unauthorised access to private contents.

5.3 Software Quality Attributes

This section includes additional quality characteristics of the product :

5.3.1 Reliability

The software is supposed to work properly while running any light-weight background processes.

It is supposed to recognise person when video is captured in proper lighting conditions. The face of the user should be in proper orientation and only limited no. of faces should be there in focus, determined according to camera quality.

5.3.2 Availability

The system will be available for use whenever the user deems necessary 24/7. The system shall allow users to restart the application after failure with the loss of at most the last image matrix captured while operation.

5.3.3 Maintainability

The system will be updatable from software patches available through the Google Play Store. Any discrepancies will be addressable by any developer as the coding will be done according to the coding standards of IEEE.

5.3.4 Portability

The software will be easily transferrable to any Android device satisfying the minimum software dependency requirements as specified in this SRS Document. The software can be installed on an Android using the same method as any other Android App via the Android App Manager.

5.4 Usability

The system has been envisioned according to the User Center Design model(UCD). The usability of the system has been described based on the factors given below :

5.4.1 Contextual Inquiry

Contextual Inquiry is a user centred design research method which is usually structured as an approximately two hour one on one interaction in which the researcher watches the user in the course of users normal activities and discusses those activities with the user. It can be performed in following ways:

- **Active CI :** In this type of CI, the user is actively included in development team.
- **Passive CI :** In this type of CI, the user merely provides his observations and experiences to be recorded by the administrators.

In our development project we have adopted passive type of contextual inquiry. Following the above convention we had proper contextual inquiry with 4 target users.

- **Interview :** In this part of contextual inquiry, the users were provided with general idea of the application and then asked same set of questions to all of them. Their responses to the questions are as follows:

  Question 1 : Would you prefer to use this app for attendance purposes over signature based attendance ?

  - **User 1 :** Yes, I would prefer to use this app for attendance purposes because signature based attendance takes a lot of class-time. Passing the attendance sheets distracts the students concentration.
  - **User 2 :** Yes, I would prefer to use this app for attendance purposes. I generally take attendance myself, so it will save my lecture time and help me to focus more on my lecture.
  - **User 3 :** I would surely try this app. But it may happen that some students, specially backbenchers, may not be recognised during lecture, which will hamper thier attendance. So, I would like to see the accuracy level of the app.

Question 2 : How much frequent would you like to run the application so that students are notified of their loosing attentiveness?

- **User 1 :** Every 2-3 minutes so that they remain focussed during the lecture.
- **User 2 :** Every 5 minutes, so that they can enjoy the class and there is active participation from all students.
- **User 3 :** Every 7-8 minutes, I like the students to relax a bit during lectures so that they dont get bored.

Question 3 : Would you like to cross-check possibility of proxies using this application?

- **User 1 :** Yes, I would like to use this app for cross-checking possibility of proxies. This will help me concentrate more on lecture rather than wasting my time on manual checking for proxies.
- **User 2 :** I shall surely try it once, but it may happen that it is not able to recognise some students, so it may be problematic for them.
- **User 3 :** I have full confidence on signature-based attendance. But I would use this app to confirm its accuracy.

Question 4 : Do you want to give any suggestions?

- **User 1 :** If possible, please include flash light feature so that faces can be recognised during low lighting conditions.
- **User 2 :** I am not able to think of any suggestion for the moment. I'm sorry.
- **User 3 :** You may add login feature that will keep the data secured.

6 Other Requirements

A general knowledge of using smartphones is required to use the product. The primary requirement for this application is Rear camera. Failing above requirements, the product won't compute correct results.

6.1 Appendix A: Glossary

- GNU-GPL: GNU General Public License is a widely used free software license which guarantees end users the freedom to run, study, share and modify their software.
- Android (stylized as android) is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets.
- Machine learning is the subfield of computer science that gives computers the ability to learn without being explicitly programmed.
- OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision.
- OpenGL (Open Graphics Library) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics.
- Android Studio(SDK) includes comprehensive set of development tools. It includes a debugger, libraries, a handset emulator based on QEMU, documentation, sample code and tutorials.