FACIAL RECOGNITION

INTRODUCTION

The facial recognition implementation had a couple of major aims. The first was to avoid any possibility of impersonation or cheating by students. The second was to avoid any accidental errors in marking students' attendance. A third final aim was to make the solution available easily in terms of today's technology, which based on our survey, we decided would be an Android app. This not only makes the solution more relevant but also makes it very handy and mobile in its use. The core functionality has been implemented thanks to the Kairos facial recognition REST API.

DONE

In terms of code components three things were done. The first was taking a picture of the student, the second was calling the relevant service of the Kairos API using the image taken and the final one was parsing the response and displaying the success or failure of the action to the user. In simpler terms, enrollment and recognition(for marking attendance) of a student were implemented. The end user has an option to enter a name and select the enroll option to register himself. Or choose the recognize button to automatically detect himself in the picture taken. As stated earlier, both these actions display whether the action was successful or not to the user. An SQLite database has been created for recording attendance locally on the phone.

DOING

Currently, work is in progress for recording attendance in the database. This is in continuation of the last point from the previous section. Whenever a user successfully uses the recognize option, his/her attendance will be saved to the local database. The next step will be the final part of the next user functionality, where the end user will be able to select a date or date range and view the attendance of all students for that time period. The last possible functionality for the future is the feature of class segregation, creating separate classes (for different courses) and enrolling and recognizing students specific to a class. And of course, viewing attendance for each of those classes separately. Login will be the last functionality, if at all, because of it being a common feature in all apps, but even more importantly because of the reason mentioned in the next section.

ROADBLOCKS

A major roadblock, or should we say a learning curve, was Android development. Since we weren't familiar with any sort of mobile app development, a fair amount of time after the January phase was spent in learning to code in and familiarize ourselves with the Android framework and environment. After this was testing and checking the feasibility of a facial recognition API which was a make or break criteria for this implementation. Thankfully the Kairos API worked out perfectly for our needs. A future roadblock seems to be in the registration of users, and this is purely in terms of the possibility of the release of this app for commercial purposes. Registration on the Kairos website provides an app_id and an app_key which can be configured and used, but only for personal use. Commerical use requires a commercial account which involves a fee for the use of the API. Though for the requirements of this course, this wouldn't be an issue.

ENGAGEMENT

Studying the various problems of Attendance Management, a repeating factor in grievances was lost Classroom time. The obvious solution is to minimize as much time as possible. However, many suggestions from professors in our survey revealed an opposite approach. We can let Attendance take up class time if it's a part of class and can help with student engagement in the classroom. The key focus points of this application are to track attendance first, and measure or encourage engagement second.

To track attendance through the lens of engagement, the ways engagement could be tracked were considered in designing this application. Many professors use attendance as a participation grade. Quizzes are occasionally given out at the beginning of classes to serve either as a reminder of homework or a previous class or a test to see if students payed attention or completed homework. These could vary between multiple choice easy to grade quizzes, and open ended questions to spark discussion and interesting ideas. One professor in our survey specifically asked for a way to easily cold call students for discussion.

Considering these points, as well as those from our previous reports, the initial non-functional requirements of this application needed to be that it was portable enough that students could use it on their laptops, and must be fast and easy to use for both students and professors. Functional requirements for the final product are as follows. The interface will separate which users are professors and students. Professor abilities will include setting up quizzes and their duration, type, questions, answers, and deletion. They will also be able to see past quizzes and their results. Student attendance can be viewed by individual student or by class as well. Attendance determined by whether quiz was taken. Student key requirements however only include the ability to take quizzes open by class. An additional functionality for the student would be to help the student keep track of miss days or quizzes in this case. Because the professor would set the duration, it would encourage the student to be on time.

The application currently written in Java using Netbeans IDE, contains an object oriented inspired design for fluidity in the interface. Its backend Database is MySQL, and has ten tables holding data. Students and Professors upon loading the program need to login and will see similar interfaces. Both have a section that quickly states whether there is/was a quiz today, and options for the professor to see results of the quiz, or for the student, if its open, to take it and submit. Professors can search by date for past or future quizzes and set up new ones. Past quizzes are not editable, future ones however are. Currently, only default quizzes are set for creation. Default quizzes contain one "I'm here" button that creates the log that that student took the quiz and was in class. For a particular quiz, a professor can quickly see all who participated, and click one button to randomly call a student. A future functionality once non-Default quizzes are enabled, is to let the professor see a spreadsheet of answers, without student names, for open discussion.

Two major drawbacks to this implementation are the java interface used, and its reliance on the honor system in students. The Java GUI used while portable is not as preferable as for a web scripting language like php. Quizzes could be taken in one's room, and students could then claim to be in class. However, the engagement and its use in class discussions would deter those from cheating the system.

VOICE RECOGNITION

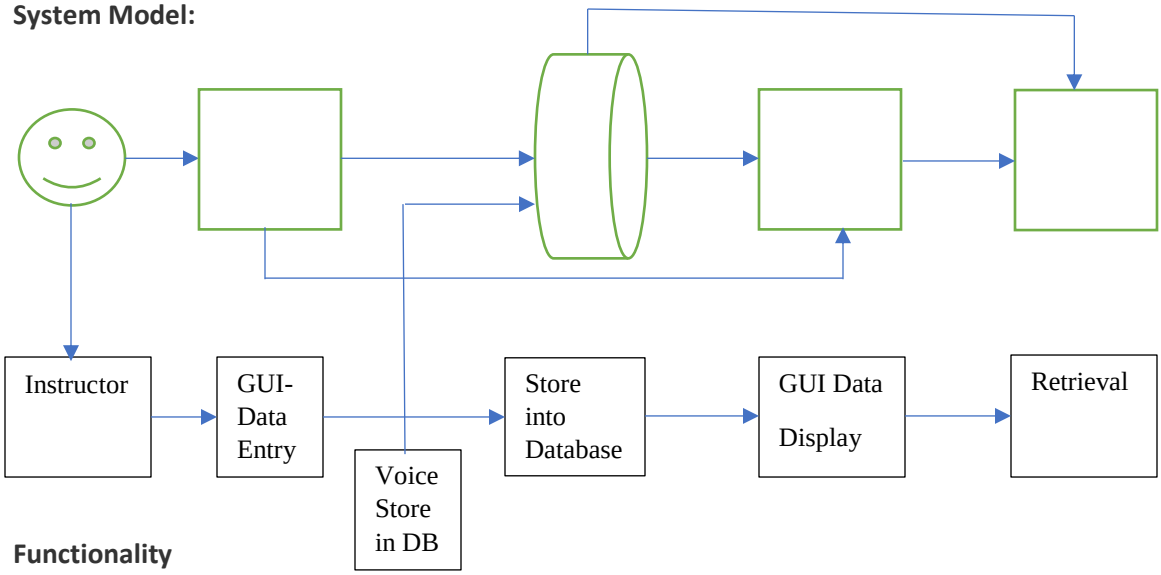
Requirement Document: Pi Attendance Management System

Introduction:

The voice recognition system is focused on enhancing the capabilities of the teachers/instructors to take effective and efficient attendance in class. Our research has shown that manual attendance is time consuming in addition to having an impact on students grades as they miss classes if a loophole can be found to skip the class without attendance. Pi-Voice is based on a simple design with a user friendly interface.

Purpose: The objective here is to give ease of taking attendance, save time, reduce error while taking attendance which will imply more focus on topics inside the class.
Stakeholders: Academic Institutes
Users: Instructors
Requirements Constraints: limitations of time and budget
Naming Conventions: DataEntry, Domain, Database
Relevant Facts: The importance of this product in developing countries, it is simple and easy to use. Can be influenced to deploy to such classrooms with minimum cost and easy usability.
Assumptions: helpful for developing world

System Model:



Functionality

The system will provide an interface with a functionality such as recording the student's records, capturing voice commands, searching and verification of voice samples, and reporting on daily or weekly basis. The graphical interface must provide efficient functionalities to record and retrieve information as per the user request.

Scope: the product is a simple graphical user interface system. The system will provide functionalities to store and retrieve data through a database system connected to the system. The system will also provide reporting tools for the instructors. The overall goal here is to provide simplicity and ease of use to the users.

Functional & Data: the product will have multiple interfaces for recording and retrieving data. The second is connection to the database. It will also provide reporting option for the user.

Non-Functional Requirements:

1. The intended appearance is focused on simplest design pattern. The purpose here is to give easy access to instructors (specially in developing world with minimum computer knowledge) to easily use the product).
2. The intended operating requirement is Microsoft Windows platform.

Open Issues:

Below are open issues in the project.

1. Database connectivity
2. Reporting

Things to do:

1. Work on the connectivity of the database that will store the voice information along with other data.
2. Provide reporting structure. User can request for report.
3. Final testing

Upcoming plan:

1. Work on the open issues and make sure that product is ready for final touch.