Bilkent University

Department of Computer Engineering

# Senior Design Project

Automated Attendance Taking System (AATS)

## Final Report

Alba Mustafaj, Argert Boja, Ndriçim Rrapi, Rubin Daija

Supervisor: Selim Aksoy
Jury Members: Ibrahim Korpeoglu and Hamdi Dibeklioglu

May 10, 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 CS492.*

# Contents

## Final Report

Automated Attendance Taking System (AATS)

# 1. Introduction

This project's aim was to automate the process of attendance taking, which leads to fewer responsibilities for instructors and more accuracy for students. By using face recognition, we are able to identify all the students present in class. Furthermore, by conducting several checks through the class hour there will be no possibility of cheating. Both students and instructors are able to check the results on the custom built client application. Even if there is any inaccuracy, the students are able to still be counted as present by letting their professor know that they were present in the taken picture. In this way, the new data is collected and the system is trained again in order to adapt to individuals changing their look. This report aims to provide the final architecture and design of the project, impact of our solution, contemporary issues, tools and technologies used during the application's lifecycle. The final architecture and design of your system as well as the final status of the project is presented in this report. The report must explain the impact of engineering solutions, developed in a project, in a global, economic, environmental, and societal context. Similarly, the report must include a section that discusses the contemporary issues related with the area of the project. The new tools and technologies used during the course must be explained in a section. Use of library resources and Internet resources to find background information including similar design, component information, and basic engineering principles must be given.

# 2. Architecture and Design

One of the goals while designing this project was to make it modular, such that it could be easily adapted to another system or case without the need to change the whole system. Hence every part of the system is designed independently. For example there is the communication module where if the communication medium is changed, then simply that module would need to be changed without affecting the other part of the system. In addition to these the system is designed to be self-sufficient in the sense that all the computations are done locally. The system can run on any hardware that is comparably equal to or better than a Raspberry pi 3 B+, however we have seen through our testing that an x86 based architecture provides better results overall. Furthermore it is noteworthy to note that the accuracy increases with the increase in the quality of the camera hardware used.

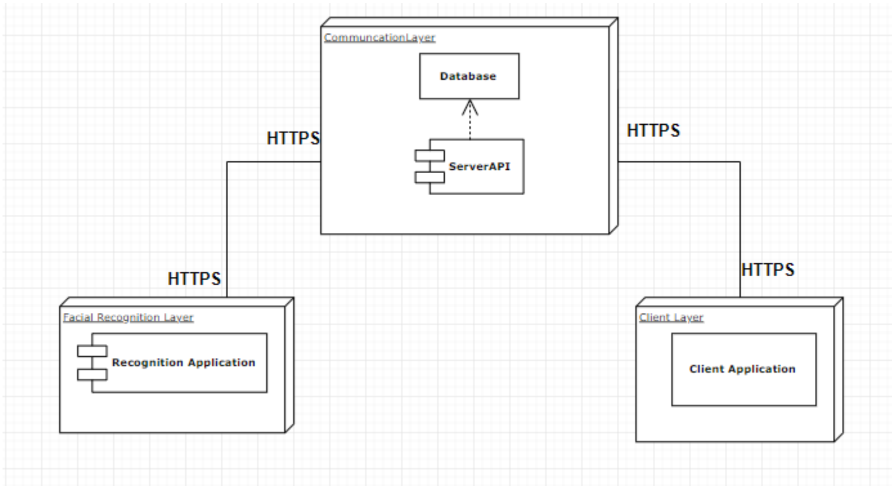



Figure 1 : Component Diagram

As it can be seen from the figure 1 above, the modularity is supported by loose coupling the components from each other. As such the Recognition devices (Raspberries) are not aware of the client devices. They communicate their actions using the ServerAPI. The client application also uses the ServerAPI as well. The ServerAPI in it's hand connects to the database and feeds the appropriate response to the incoming server requests. The ServerAPI is also loosely coupled with the database code-wise being that with only minor differences on the ServerAPI code a new Database can be easily adopted. This matches our prior requirement, scalability, being that our solution should adapt not only for our university but for universities in general. Security-wise all the data communication is done through HTTPS. An extra layer of security is added to the Database by encrypting its sensitive data. The server on top of which the API resides is optimized against other security issue such as when high number of request come from one computer within a short amount of time then the requests are instantly rejected to start with.

## 3. Impact of the Solution

### a. Environmental Context

The traditional way of taking attendance is done through collecting signatures during class hours in a piece of paper. On a worst case scenario in single class (Bilkent) there can be 8 hours. Since courses usually have 2 hour blocks we assume for a single classroom there need to be spent 4 pieces of A4 paper for collecting student signatures. Assuming there are at least 400 classrooms at the least at Bilkent University it means that for a single day 1600 pieces of paper need to be used for taking attendance. In a week the number accumulates to 4000 sheets of paper per week and 144 000 sheets of paper per year. This means that in order to complete the attendance process around 25 trees have to be cut [1]. Also, the papers that are used rarely go back into the recycling process. Therefore our solution will try to compensate for the negative effects that the current traditional method of attendance taking has on the environment by getting rid of the need of papers in the process.

### b. Societal Context

Manual attendance taking process is strenuous, shifts the student's attention during class hours and also takes up the professor's time since they have to manually mark each and every student one by one. When the number of students in one class increases the marking process becomes even more tedious. Another issue that might arise with manual attendance taking consists in the students trying to manipulate their presence by having someone else sign for them. All of these issues are addressed by our solution, where professors and students both would barely be aware of the system after it is implemented being that it is fully automated.

### c. Global Context

Taking attendance during class hours is a strenuous mechanical process that occurs in universities. The global perspective stands behind the fact that our system is aimed at being easily adapted to any university type with only a small number of configurations needed to be made.

### d. Economical Context

As calculated in part a) around 144000 sheets of papers are used per year on average, say at Bilkent. The yearly cost of that according the market prices is around 600$. On the other side the cost for a raspberry Pi and the attached camera is around 34$ when purchasing in such large quantities. Assuming we put one pair in every classroom for 400 classrooms we would need a total of one-time cost of 14000$ assuming we include the additional services costs as well. Obviously the tradeoff here means that we spend 600$ per year with the paper based system and one-time fee of 14k$ for the automated system. Economically speaking our solution might appear to be expensive, however the ethical constraints such as environmental and societal would end up having extra points and winning over the economical aspect.

## 4. Contemporary Issues

One of the main contemporary issues that this project also tackles is privacy.  This is done by being fully compliant with GDPR. Hence the system provides end to end encryption as an essential feature. Furthermore the system also does not save the pictures taken during the class hour and only processes them locally. In addition to these the student faces are actually saved as vectors hence the system does not actually have any picture saved in it at all.

## 5. New Tools and Technologies

## 5.1 OpenFace Library

In order to power our facial recognition system we have made use of the open source OpenFace library [2]. The main challenge was to fully comprehend how the library works and how we could adapt it to help solve our facial recognition problem. New technologies used for this project include the OpenFace library, which is used for face recognition. Dlib is another library that we use in tandem with OpenFace. This library is used for face detection.  Our facial recognition algorithms run on a Raspberry Pi 3+ which limits our capabilities in the computational power perspective. In that sense another challenge was to optimize the usage of the library so that we can reduce the running time being that we needed to fetch the recognition results as fast as we could.

## 5.2 Android Studio

Android Studio was used to build the android client application. The application is connected to the database through https connection.

## 5.3 InnoDB

The type of the database that we used to keep the user's data was InnoDB. The sensitive user data that resides in the database is already encrypted and in case of an attack the information stolen would be of no use without decryption

## 5.4 End-To-End Encryption

The devices have to communicate with the database. To make use of end-to-end encryption we put a small API of services in our server. The communication with the API is secured over SSL. On the other hand the communication with the database can only be done from within our servers and not in other machines. In order to offer the other end of the encryption we encrypt all of the data that is entered in the database to start with. When the devices need data the server side script will decrypt the data and send it back through SSL.

## 5.5 Server API

In order to enable a secure device-database information exchange we decided to build an intermediate layer on the server side using PHP scripts that would serve requests regarding authentication of devices and data fetching.

# 6. Project Status

So far we have managed to meet all of our initially held requirements that we specified at the very beginning. The only major changes from the previous iteration consist in small UI choices that we made to offer better user experiences.

## 6.1 Deliverables

- Android App Available on PlayStore

- Server Side API ready to serve requests

- Database Structured and Adapted

- Facial Recognition Implemented for Raspberry Pi 3+

- Final Report

# 7. User Manual

## 7.1 Hardware

### 7.1.1 Requirements

***- Raspberry Pi 3 Model B+***

The Raspberry Pi module contains the needed software to automatically run the cameras, connect to the database, and perform image recognition accordingly. It should be accompanied by the raspberry compatible camera, the details of which are given below:


***- Raspberry Pi Camera NoIR V1.3  5MP (2592x1944 px)***

Each Raspberry-Camera pair alone is able to capture and recognize a maximum of 40 students. For larger classes additional Raspberry-Camera pairs must be added. Addition of a new pair is automatically reconfigured, where one of the raspberries will work as the master and the other one as the slave device in order to reduce the server load.

## 7.2 Software

### 7.2.1 Server Side Requirements

- PHP Version 7.2

- InnoDB supporting storage engine database

- SSL Secured Server through HTTPS

### 7.2.2 Client Side Requirements

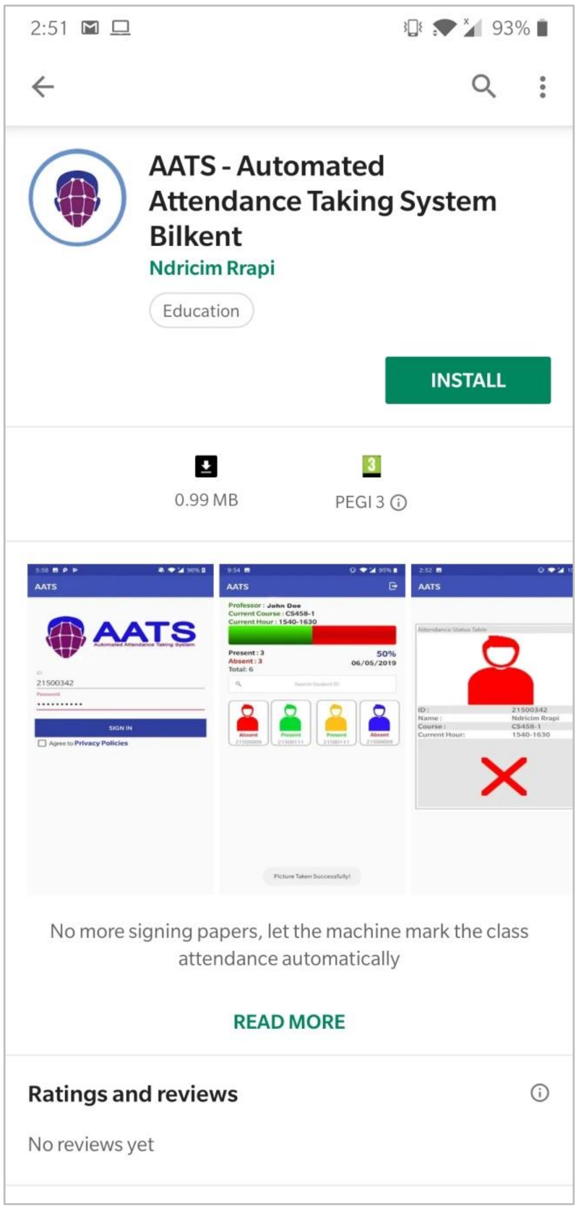
- Android 4.4 version and above

- Permission to use internet connection

- Permission to use the camera feature

- PlayStore App installed

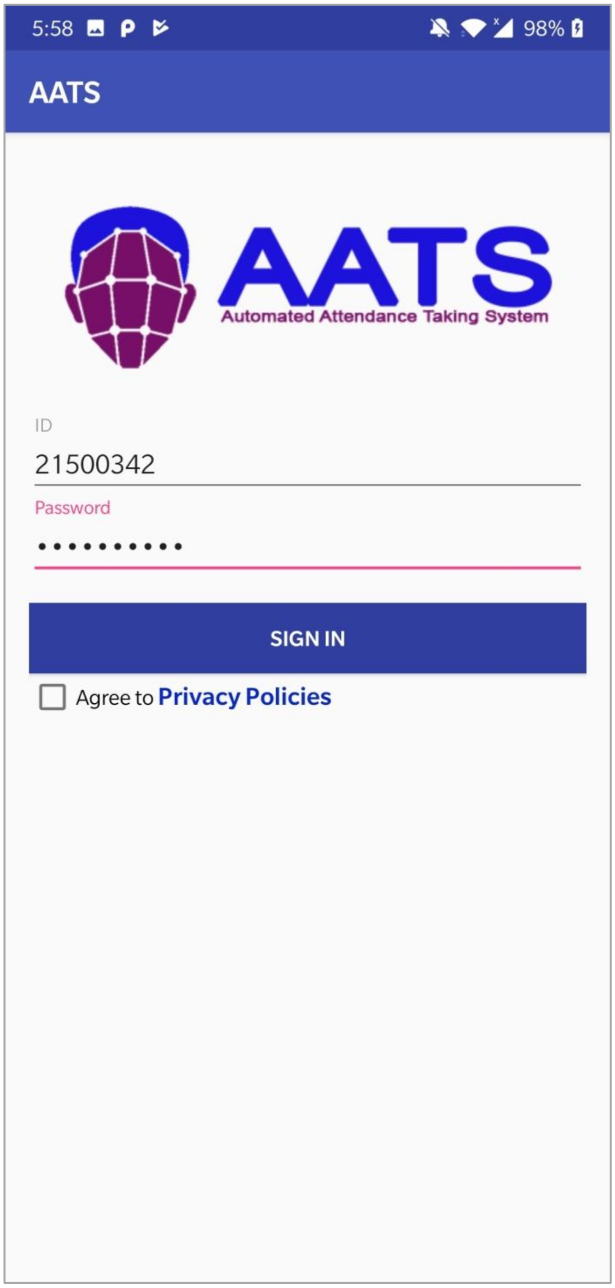
- The client app is available on PlayStore as seen in the screenshot below.



### 7.2.3 User Interface

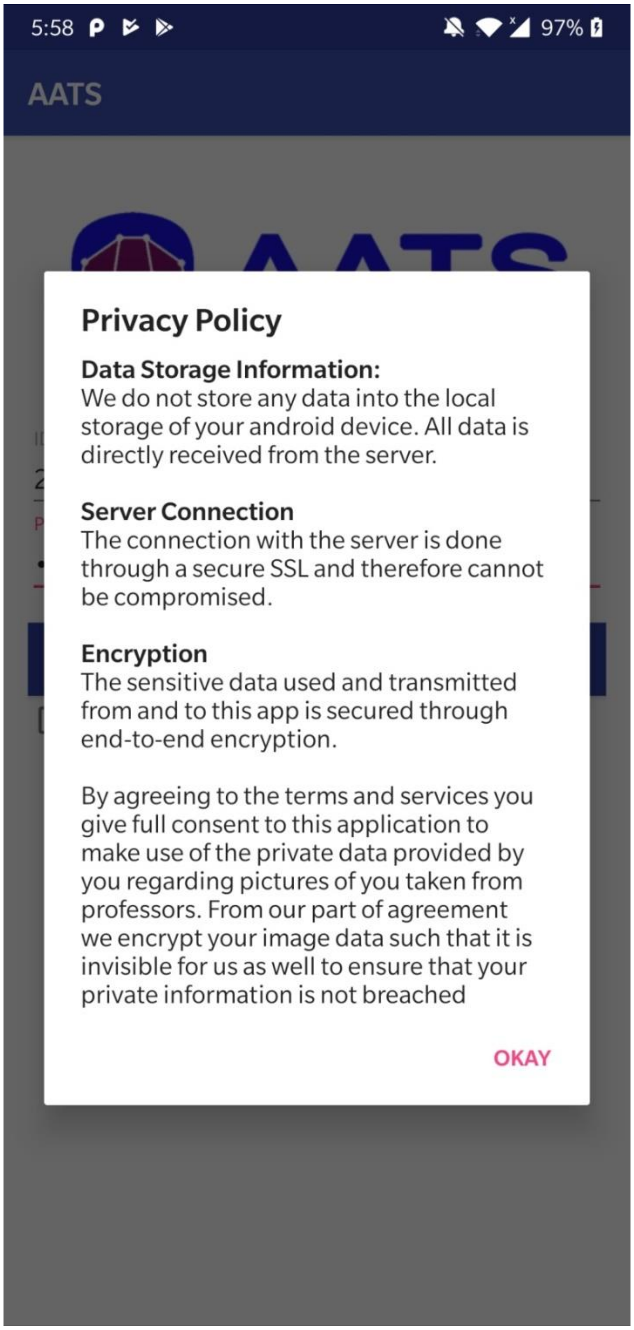
#### 7.2.3.1 Login Screen

Both professors and students can login through the same screen using their ID and password.



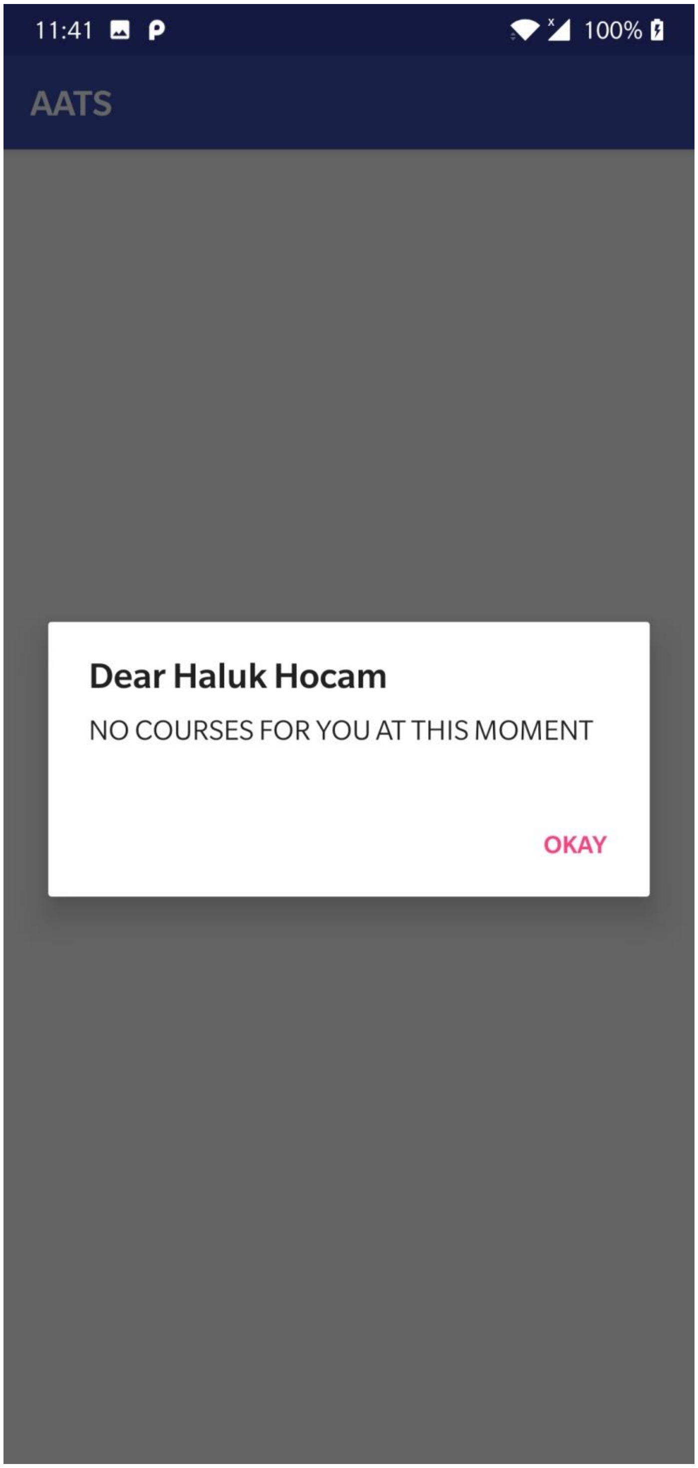
#### 7.2.3.2 Privacy Policy

Students and Professors need to agree to our Privacy Policies/Terms and Agreements. This is done to request consent from users and letting them know what data we use and how we use it.



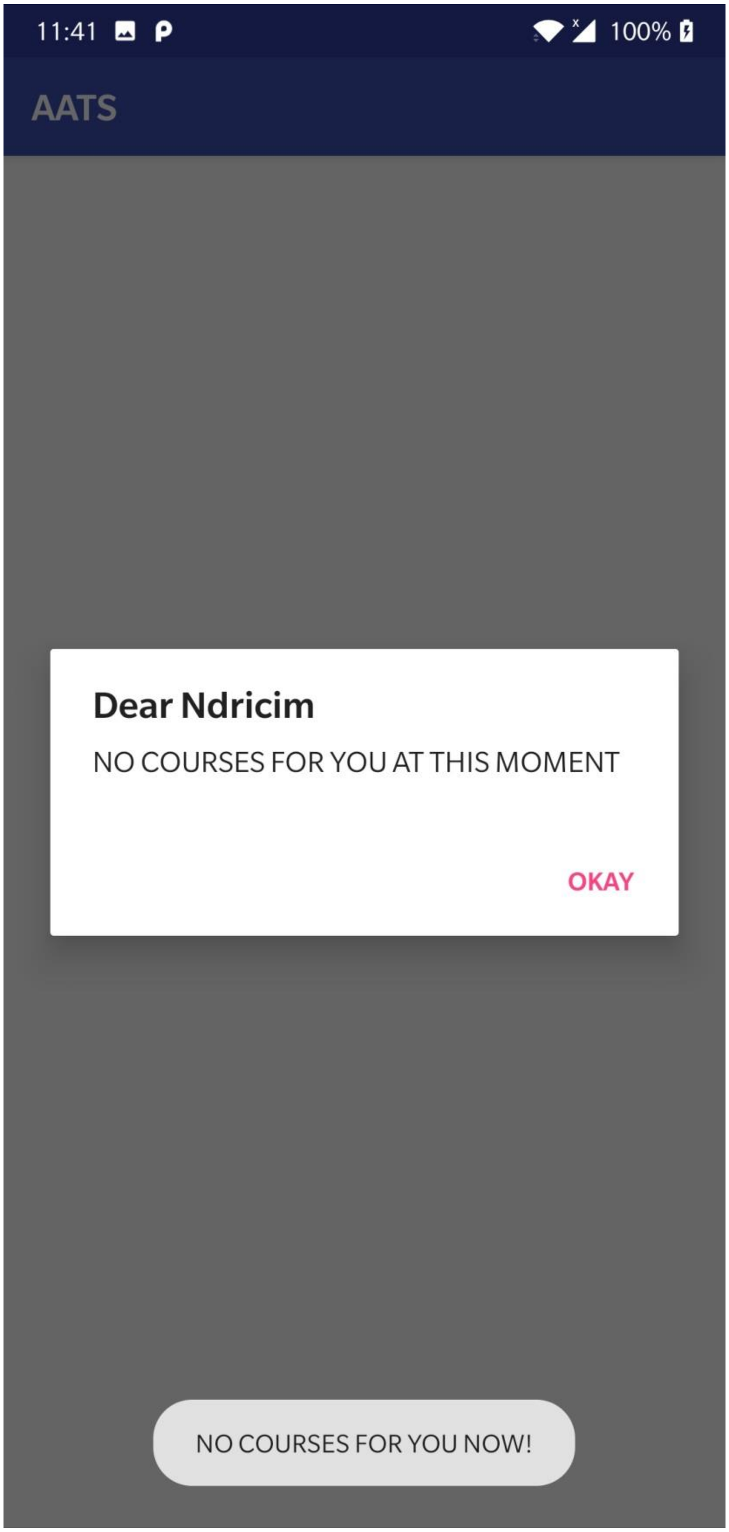
### 7.2.3.3 Professor No Course Screen

When the professor logs in outside of the class hours a small dialog box appears notifying them that there are currently no courses at that login time for the professor.



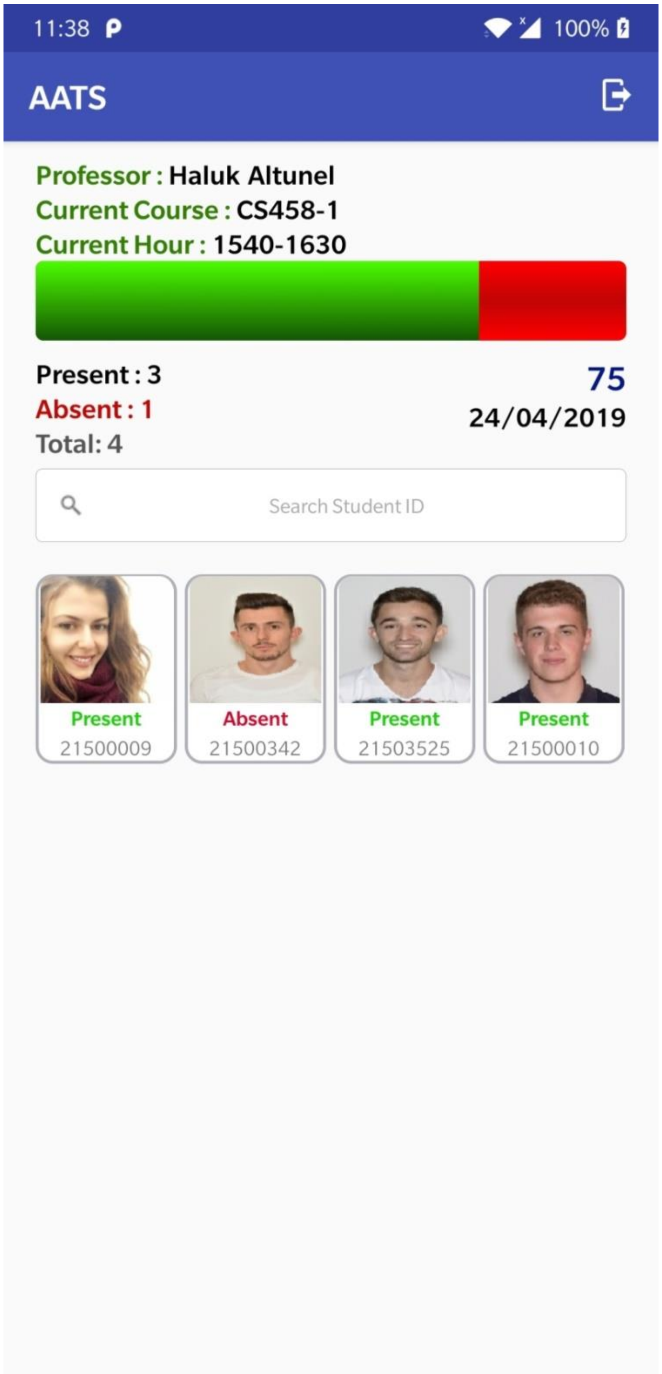
### 7.2.3.4 Student No Course Screen

When the student logs in outside of the class hours a small dialog box appears notifying them that there are currently no courses at that login time for the student.



### 7.2.3.5 Professor Main Screen

When the professor logs in successfully the following screen appears, where the live results of the raspberry pi image recognition show which students were marked present and which not after the recognition. Other relevant data regarding attendance ratio for the class can also be seen.

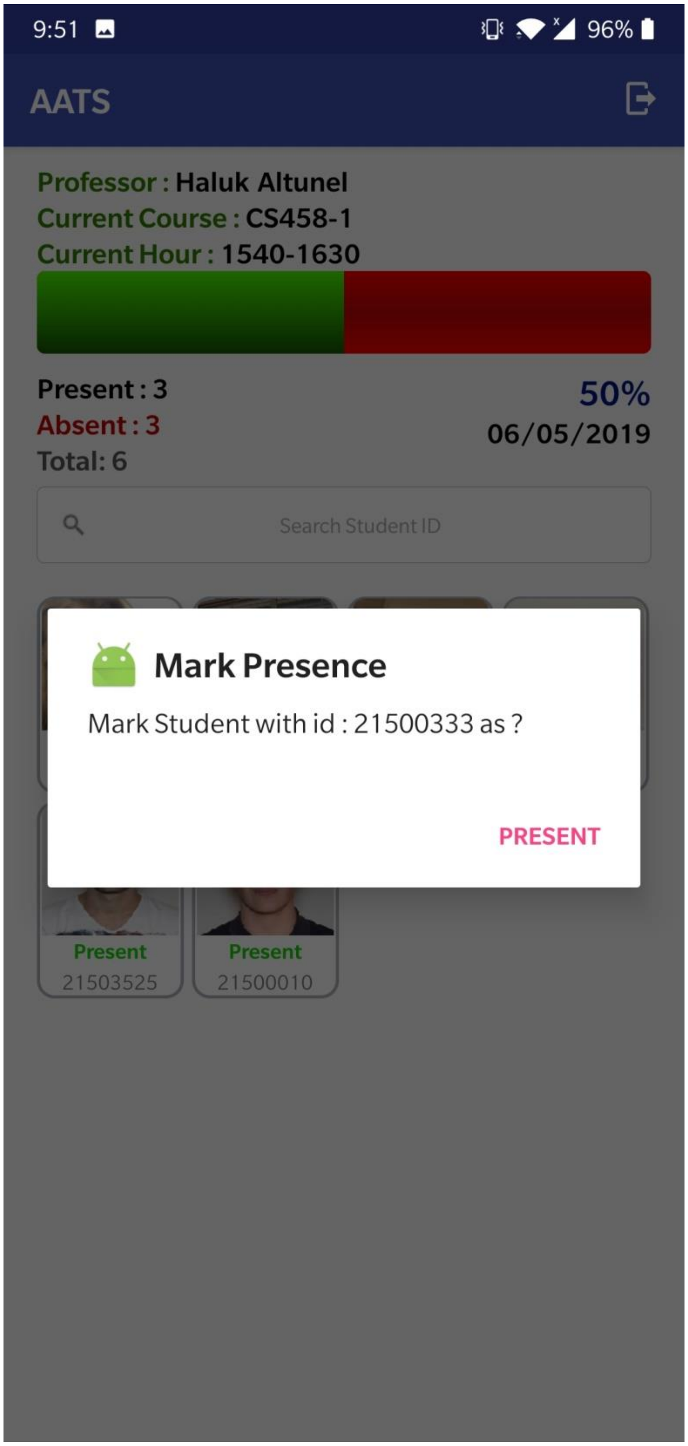


#### 7.2.3.6 Student Main Screen

When the student logs in successfully the following screen appears, where the live results of the student's attendance status appears. The following example shows that the student was not recognized for that hour.

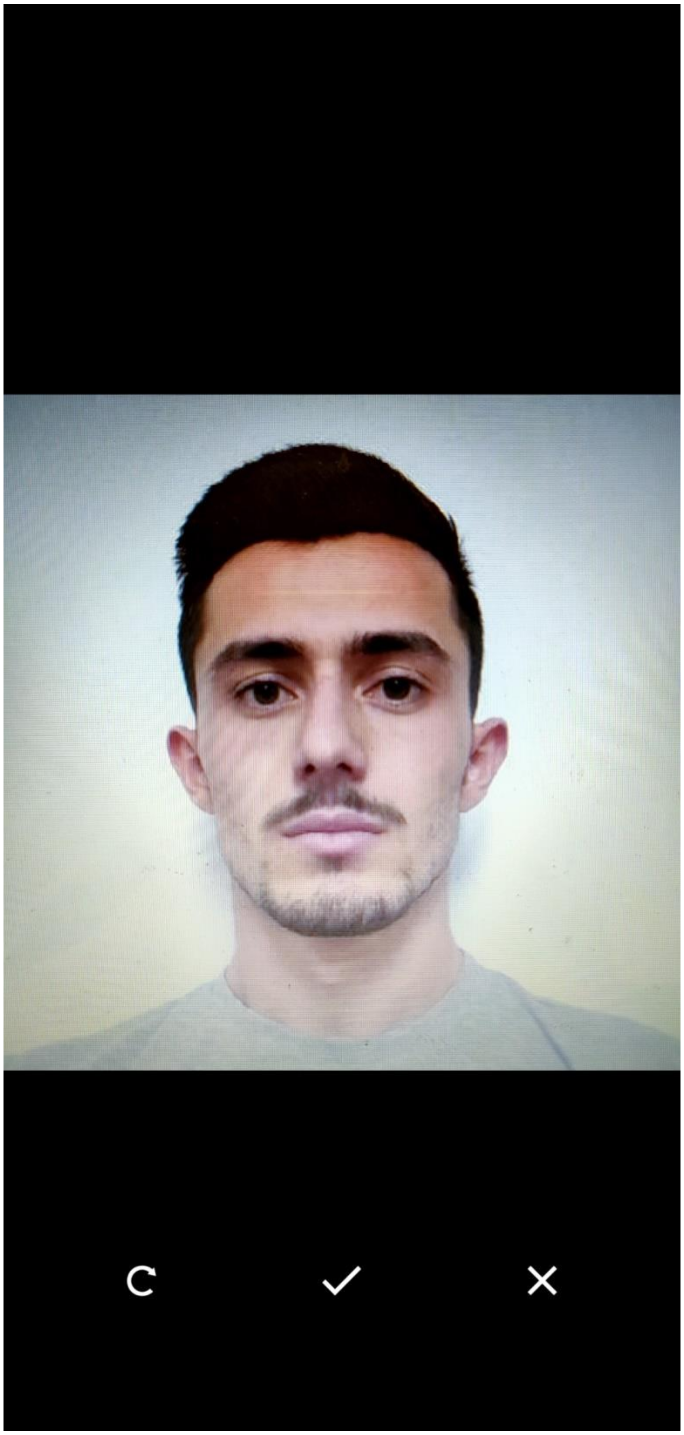
### 7.2.3.7 Professor Marks Student Present - 1

When the raspberry pi's fail to recognize a student who is already present the student may complain to the professor during course break, and the professor will be able to instantly mark them present through their main screen by first clicking on student's picture and then on present button.



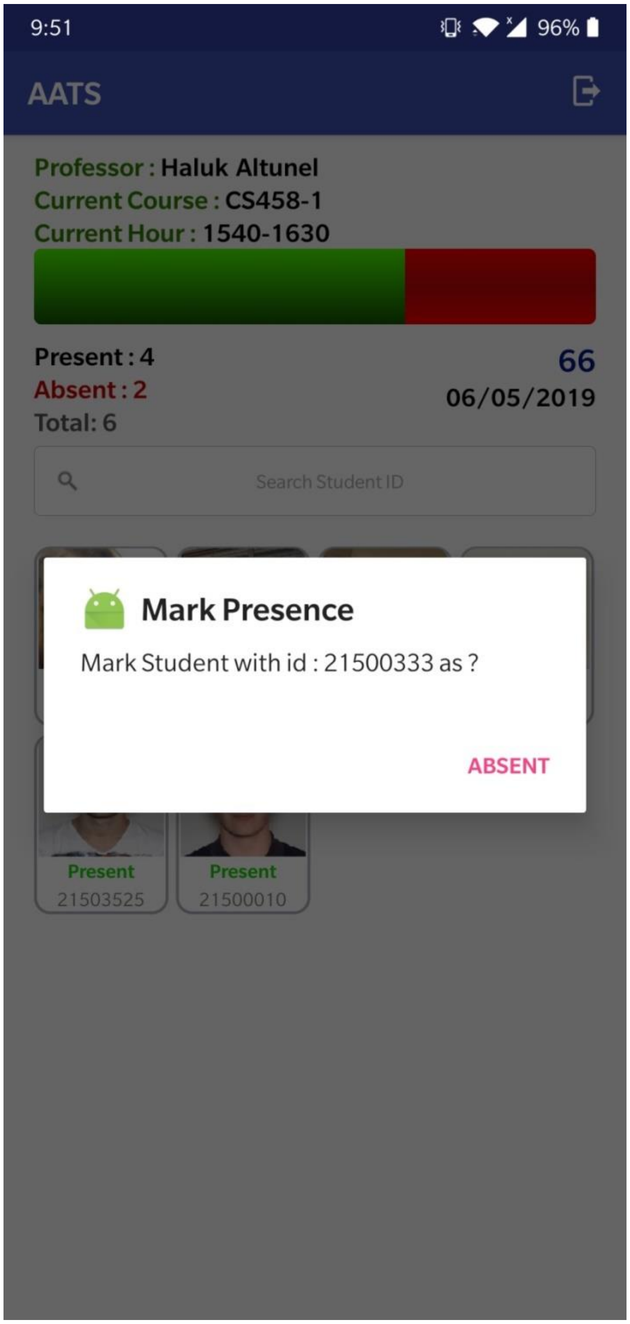
#### 7.2.3.8 Professor Marks Student Present – 2

After clicking on Present button the camera opens and the professor needs to capture a picture of the student and click on confirm. The picture will be automatically sent to the server and the student will be marked.



### 7.2.3.9 Professor Marks Student Absent

In case the professor needs to force mark a student absent that was marked by the system as present. The professor may simply click on the students picture and then click on Absent. The student will be marked absent instantly.



# 8 References

[1] How Much Paper Does One Tree Produce? [Accessed May 01, 2019]
https://www.sierraclub.org/sierra/2014-4-july-august/ask-mr-green/how-much-paper-does-one-tree-produce

[2] B. Amos, B. Ludwiczuk, M. Satyanarayanan,
"Openface: A general-purpose face recognition library with mobile applications,"
CMU-CS-16-118, CMU School of Computer Science, Tech. Rep., 2016.