Bilkent University

Department of Computer Engineering

Senior Design Project

Automated Attendance Taking System (AATS)

High Level Design Report

Alba Mustafaj, Argert Boja, Ndriçim Rrapi, Rubin Daija

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

December 31, 2018

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.

Contents

High Level Design Report

Automated Attendance Taking System (AATS)

1. Introduction

In universities participation in lectures is highly advised and sometimes even mandatory. As we all might have experienced, the traditional way of checking for attendance is by passing a paper which contains all student names so that they can sign for themselves. This method of taking attendance is yet another reason for irregularities in this system. This is because sometimes the professors might forget to bring the attendance paper, or the students might forget to return the attendance paper. Furthermore, some students might come a few minutes late and they forget to sign the attendance.

This project aims to improve the methods of attendance taking, which will lead to less responsibility for instructors and more accuracy for students. By using face recognition, we will be 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 will be able to check the results on an application that will be built. Even if there is any inaccuracy, the students will be 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 will be collected and the system will be trained again in order to adapt to individuals changing their look. The following sections of this report will provide a thorough description of the high level design of our project.

Initially the purpose of the system will be described, followed by the design goals. In addition, existing software architectures will be presented and after that we introduce the software architecture that we propose. The latter is later going to be used as the foundation of building the application in the later steps of the project that follow. This report will be built in coherence with the latest object oriented software engineering methodologies [1].

1.1 Purpose of the system

The purpose of our system architecture is mainly to remove decoupling and make use of reusability on the later stages of code development as we try to implement our idea. So far we have already defined our requirements and functionalities in the previous stage reports, yet the complete structure of the system is not yet defined. This means that we have to define a structure which relates all of the parts of the system by grouping the functionalities with similarities in several components. This will in turn provide the bigger picture of the system as a whole. In this way we will be able to connect the groups with each other and create services to facilitate communication between groups/components. Last but not least, having an organized structure will provide a swift management of our application features and will make the application more robust and reliable in the future.

1.2 Design goals

1.2.1 Performance

The system should be able to process at least a picture from each class in the University within the timeframe of a lecture hour. This performance goal will be reached by creating a system that makes parallel computation of the received input by making use of the computational power of the edge devices which in our case consist in the Raspberry Pi devices.

1.2.2 Capacity

The system should be able to take a picture that will encompass all of the students in a class. The upper limit is the highest number of students per class which might go up to 80 in the big seminar classes. The cameras should be capable of capturing wide angle pictures. The capacity goal though is bottlenecked by the cost of the camera modules, which means that we might not be able to reach our upper limit of 80 due to financial constraints. Thus this would mean that we would be considering a lower limit capacity level to show a proof of our concept which might consists in the system being able to capture between 30-50 students in a class. Other options to reach the capacity goal would be buying multiple cameras and putting them in different angles however that is strictly dependent on the budget of the project which per se is strictly limited. This means that with higher budget the intended capacity goal can be reached with just more time to make the necessary changes.

1.2.3 Privacy & Security

A picture cannot be taken without firstly notifying the user. This will be possible through the notification system that will allow the student to be notified when the cameras have captured and recognized them as present. Also, the pictures are considered sensitive information which can in no way be made public. So the privacy goal here is to ensure end-to-end encryption for pictures. The information that is to be carried through the online network will be mainly in vector format and encrypted as well to make it impossible for an intruder to use or alter the information within the image. Furthermore the system will not save any sensitive information, like the students faces. The vector format that they are encoded, will be used to compare them with the newly obtained pictures from the lecture hour. In addition to these the pictures taken during the class hour are also not saved, and as such if a student makes an objection they also have to physically go to the professor to prove that they were there.

1.2.4 Accessibility

The application will use university credentials for a one time log in and is accessible by all students. The ultimate goal here is to make the application accessible from any handheld device or laptop with a connection to the internet by providing interactive UIs. The personal information does not have to be entered manually since it will be fetched automatically through the student's university information data.

1.2.5 Availability

The system shall be available to the users at any time of the day provided that they have a connection to the internet. This means that both students and professors can check the attendance data of their courses through their AATS mobile applications. In order to make objections though the students can only use the time during the course, since there is no way that a student has been present in class after classes have finished

1.2.6 Reliability

In case of a server issue and inability to process the data, the board units will save the results until the server is back online. Even though these scenarios are very rare we have to make sure that the user feels safe in using a fail-proof application. The system though will automatically back up data on the edge devices and re-upload them when the system is back online.

1.2.7 Accuracy

The system should be able to identify multiple individuals with a high degree of similarity of their facial features. This is the main feature of our application thus we have to make sure that we optimize the system architecture to make this process as fast as possible. The accuracy goal can be achieved only if all the components of the systems are connected properly and the communication system distributes the job requests properly.

1.3 Definitions, acronyms, and abbreviations

AATS – Automated Attendance Taking System
UI – User Interface
HTTP - HyperText Transfer Protocol
FTP – File Transfer Protocol
JDBC – Java Database Connectivity

1.4 Overview
AATS is an application that will be built to detect and recognize the faces of the students in class. Furthermore it should be able to depict its processing results through an easy to use and friendly interface. These features should be reached according to the specified design goals. Therefore being that the number of devices in university scale will be large and the number of total students faces to be recognized is even larger. Thus, in order to achieve our overall goal of offering all these functionalities in a one-time install application we need to specify the details of our architectural structure and make sure that we make all the on-field and on-code optimizations properly so that we do not have to be dealing with overloading problems or system lagging issues.

2. Current software architecture
There are currently no specific architectures that are similar to the one we propose. This is because our structural organization of the architecture includes several services and features all-in-on. The already existing applications are not compound, which is to say they only contain one feature similar to ours instead of containing all of them. Among the existing applications, the ones that are concerned with attendance taking systems focus their architecture infrastructure online towards a UI that does not offer facial recognition and is neither automatic. Also they are not aimed at performing large scale face recognition by making use of multiple on-field edge devices.

3. Proposed software architecture
3.1 Overview
The following sections will be explaining the detailed structure of our architecture. The overall aim of our design is to make use of code reusability as well as decoupling functionality-dependent components so that they can be dealt with by a single individual that does not necessarily need to know the whole structure of the system. The latter will facilitate the development process and make it easier to connect the whole system as one at the very end of this project. Since we are also using physical on-field devices in our project, we will also be explaining the co-relation between the software and the physical devices in order to offer a clearer out of the box image of the bigger picture of our architecture.

3.2 Subsystem decomposition
The functional requirements pre-defined in the Analysis Report cannot be dealt with as is, so a new representation which groups the functional requirements into self-contained components is made. The purpose of this section therefore is to introduce the subsystem decomposition of our system. This decomposition is made in order to make

it easier to deal with each part as an individual task instead of assigning a lot of small tasks to different individuals and trying to connect the pieces later.

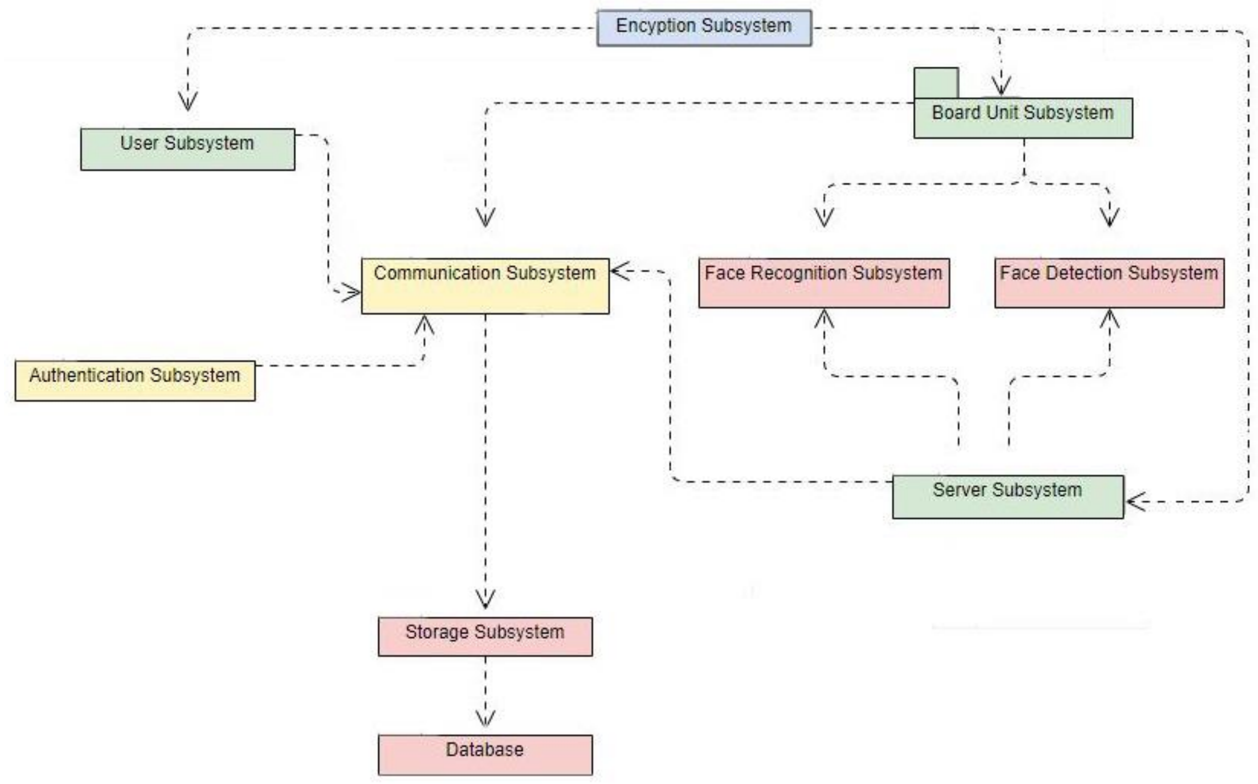

Figure 1 : Subsystem Decomposition Diagram

3.2.1 User Subsystem

This subsystem mainly contains all the user-related functionalities. More specifically this subsystem includes UI that will be provided to students and professors and other relative backend infrastructure. As depicted in the diagram above this subsystem communicates with the Database by making use of its connection to the *Communication Subsystem*

3.2.2 Authentication Subsystem

This subsystem will mainly provide the authentication service needed to secure the system from external breaches or attacks. Its structure will mainly be concerned in securing the server-client communication and close all possible leaks. It is of utmost importance that we secure all the data that is to be transmitted within the system so this subsystem must be robust and detailed as well.

3.2.3 Board Unit Subsystem

This subsystem will mainly consist in the software that will be running on the on-field devices. The major part of it consists in the coordination of the incoming input from the cameras. This subsystem is connected with three other subsystems; face recognition subsystem, face detection subsystem and lastly the communication subsystem. The first two connections are mainly attributed to the fact that the main purpose of this project is on-site image recognition. So, we expect to do both face recognition and face detection on the board unit. Lastly all the processing work that is done on this unit will need to be transmitted towards the server which means that this subsystem will also have the necessary infrastructure to delegate and receive information with the Database through the Communication System.

3.2.4 Server Subsystem

The server is one of the key structures of the whole system being that it functions as an inter-connection between on-field devices. That being said this subsystem will contain a software system with algorithms that will solve the face detection and recognition part efficiently. The server will contain the specific server-side algorithmic structure to process all the data. It will work in parallel with the board unit subsystem in cases when the board unit subsystem will be incapable of computing all of the data within the threshold. The server will also need to coordinate the communication between the on-site devices by making use of its connection with the communication system. The data transferred from and to the server subsystem is crucial in the overall performance of the whole system.

3.2.5 Face Recognition Subsystem

This subsystem will be formulated on the basis of modularity being that it will be used heavily by two other subsystems server and board unit. The main job of this subsystem is to recognize the faces and match them to the respective persons. This information is then used by the server and board unit and the results are recorded in the database through the communication system.

3.2.6 Face Detection Subsystem

This subsystem is similar in structure with the Face Recognition subsystem since it is also being used by the same two subsystems. The core of this subsystem is built upon the algorithms that will enable detecting the faces on the input image. This information will afterwards be coordinated with the respective systems and given as an input to the Face Recognition subsystem for further processing.

3.2.7 Encryption Subsystem

The images that will be captured by the on-site cameras and the information processed through them is considered as sensitive information, which is why there will be an Encryption subsystem to ensure that the sensitive data transfer will be encrypted and with fail-proof methods to ensure user data reliability at the finest level.

3.2.8 Storage Subsystem

The storage subsystem will be ensuring that data is safely stored and the incoming requests are verified before providing access to them. The storage subsystem functions as an interconnection between the *Communication subsystem* and the *Database* and is therefore connected with the two of them.

3.2.9 Database Subsystem

The database subsystem is the core of our system since it will manage the data of the application. The database subsystem will contain the necessary queries that will be used by the storage system to fetch and insert new data into the database. This subsystem should be robust and fast since its interruption of service would be fatal for the overall system. The database will be built using MySQL.

3.3 Hardware/software mapping

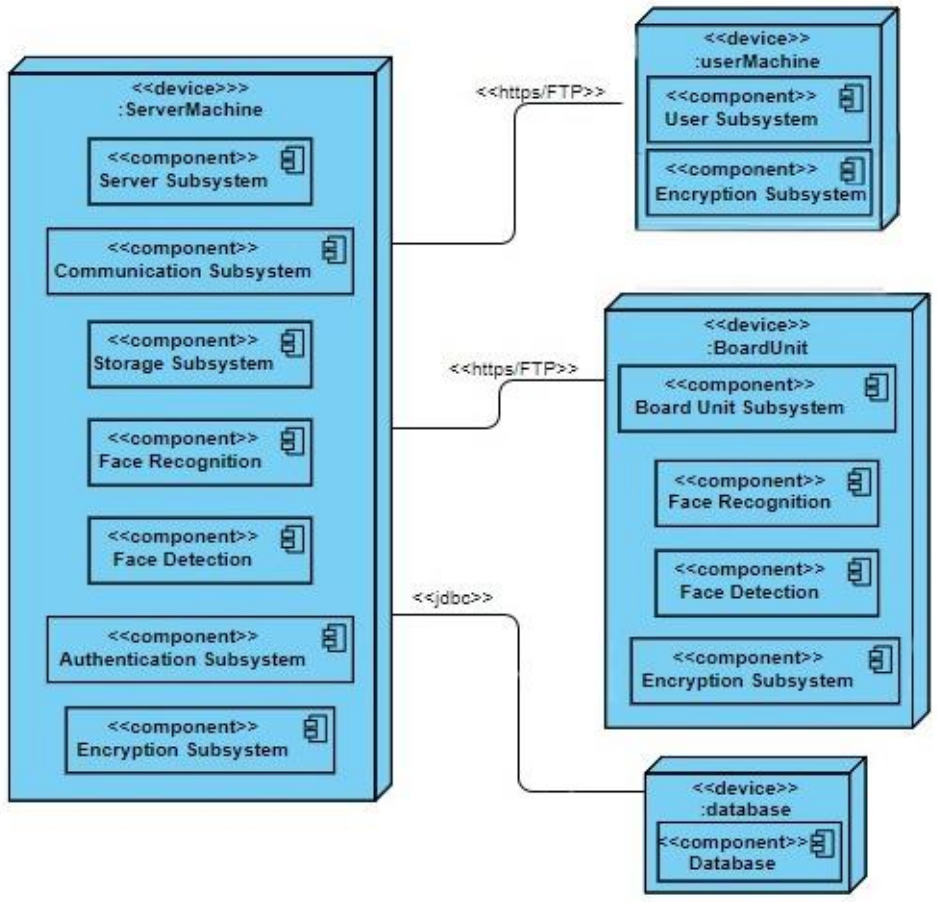

Figure 2 : Hardware/Software Mapping Diagram

The overall hardware/software mapping diagram is represented in the above figure. The components are structured under each device as shown. Face Recognition and Face Detection subsystems are found in both the Board Unit and Server since both these devices will use these subsystems. User Machine, Board Unit and Database will communicate with the server through the communication subsystem. The Authentication subsystem is responsible for authentication of Board Units and the users.

Each of the devices has an encryption subsystem as they will be communicating with one another through the communication subsystem however the messages they send need to be encrypted in order to ensure the user data safety.

The storage subsystem is a used as a connection between the complete system and the database for the persistent data storage. This way of a design allows for the versatility of easily changing the type of database used with this system.

If a device needs to communicate with another one, it will send an HTTP request to the server machine. This request will be redirected by the Communication Subsystem towards the Authentication Subsystem. The latter filters the requests and only after the request has been authenticated can a response be generated. The response that the server generates is then propagated towards the Communication Subsystem which returns the response towards the device that made the request initially. Furthermore the system will utilize the FTP to transfer files throughout itself, such as results from the lecture hour or faces required by the board unit for the next detection.

3.4 Persistent data management

3.4.1 Identifying Persistent Objects

AATS deals with three sets of objects that require storing. The first includes the student's faces, whereby each face that is encoded through the face recognition algorithm is stored. That information is then used to identify an individual without actually having any pictures or any other identity revealing information stored on the database. The second set is the student's attendance which is stored to keep track of a student's attendance. The third and last set is the information contained within the board unit during the timespan of a class hour. That information will pertain all the students recognized thus far, and the amount of times they were recognized, as multiple recognition might occur per class hour to facilitate that a student has been there throughout most of the hour.

3.4.2 Selecting a Storage Strategy

The main goal on the development of AATS is to provide a secure system, while scalability and minimizing operating costs are secondary. Thus all the information stored through AATS will be encrypted, in order to protect the data even in a case of a data breach. The first and second set will be saved in a database during all the time the system will be in use, as they will be needed to identify the different individuals. Thus for these two a database-independent API (e.g., JDBC) to store the sets in a relational database will be used. On the other hand, since the board unit will save and need the third set only for the timespan of a class hour, the storage strategy for this case will be to use flat files. These offer the opportunity to be faster in access as no remote database will be used and furthermore the files can be designed to accommodate the system's needs.

3.5 Access control and security

AATS has only two actors that are constantly active. Those two actors are the board unit and the individual who is using the application to interact with the system. The accesses they have to the server are shown in Table 1 below, in the form of an access matrix.

ACTORS\OBJECTS	SERVER
INDIVIDUAL USER	receiveLastAttendance requestObjection answerObjection sendNewPicture
BOARD UNIT	requestFaces sendResults

Table 1: Access control matrix for the AATS

Both actors are authenticated by the server when they send a request or message of whatever form; this is done with the goal of restricting access to AATS only to the appropriate parties. As it can be observed from the access matrix the access of a user is limited. The student can only request an objection through the system , while a professor can answer an objection and also send a new picture to the system. The need for a new picture arises when the system does not detect the student, and it befalls upon the professor as a higher authority to provide the correct necessary information. Furthermore the student is allowed to access the last attendance of their last class; all other attendance records can be accessed through the university's services. The board unit's access is restricted to it only being able to require the student's faces as well as sending the new results when it is done with the calculations.

3.6 Global software control

The server will be designed based on an event driven control flow, as it will respond only when a request is handed to it, and send the results when all the calculations are done. However the server will use threads to process multiple requests. Thus if more than one request is sent to the server at the same time, it will try to process both simultaneously by executing each in a separate thread. However the AATS system has more components than just the server. The user's application will be event driven as well, as it will wait for a user input in order to perform an action, such as sending an objection. The board unit is more complex. As the unit should be designed to be self-contained and work in an event driven manner, however since there are multiple boards they process the necessary information for a class in a parallel manner. Then they send their requests to the serve which uses a mix of event driven control with thread control flow to handle all the requests.

3.7 Boundary conditions

3.7.1 Start-up and shutdown cases

As depicted in the UML hardware/software mapping diagram in **Figure 2**, AATS includes four run-time components: the Server, the Board Unit, the User Machine and the Database. Except of the user machine all the others are supposed to continue to be running after they were started, whereas the user machine can be started and stopped at the need of the user.

Table 2 Start-up and shutdown cases

StartServer	The server is started manually by the administrator of the system. The application is run on the server. It checks the time and configuration file to determine the next step that it should take.
StartUserApplication	The application is started by the User. When it starts it checks for any new notifications by asking the server. If there is a new notification then it presents it to the user.
StartDatabase	The database is started normally by the system administrator.
StartBoardUnit	The board unit starts by reading the configuration file. It configures itself according to the file. Then if it was shutdown improperly, it checks the file which stores all the work that it has done before being shut down. It reports that file to the server, which takes action accordingly.
ShutDownUserApplication	The application is closed. There is no information persistent in the memory of the user's device.

3.7.2 Exception Cases

AATS can experience five major failures that may hinder the work of the AATS:

- A network failure on the server side
- A network failure on the board unit's side
- A board unit failure in which the board unit is suddenly shut down
- A server failure in which the server is suddenly shut down
- A database failure in which the database is suddenly shut down

The network failure and sudden shutdown of the server can be handled by the board unit. The unit does understand that the information is not going through to the server and as such keeps the results it has until the server comes back online. Similarly a database failure is handled by the server in a similar manner whereby the server keeps the data until the database is back online.

As discussed in the persistent storage section the unit board will use flat files. Every time it does a check of the class it will save the results. When the hour is done it will report the results to the server. If for some unexpected reason the server is not working, it will keep that file until the server is back online. Furthermore if the board unit disconnects from the internet it can continue saving files until it is back online. If the case would be of the sudden shutdown then the board unit has the file to know where it left off, what to report and the next action to be taken. The next action might depend on the amount of time the board was off. If it was within the same hour then it can simply pick up where it left, however if it is a new hour then it has to report the results and continue with the new hour.

4. Subsystem services

4.1 Unit Board- Communication Subsystem Related Services

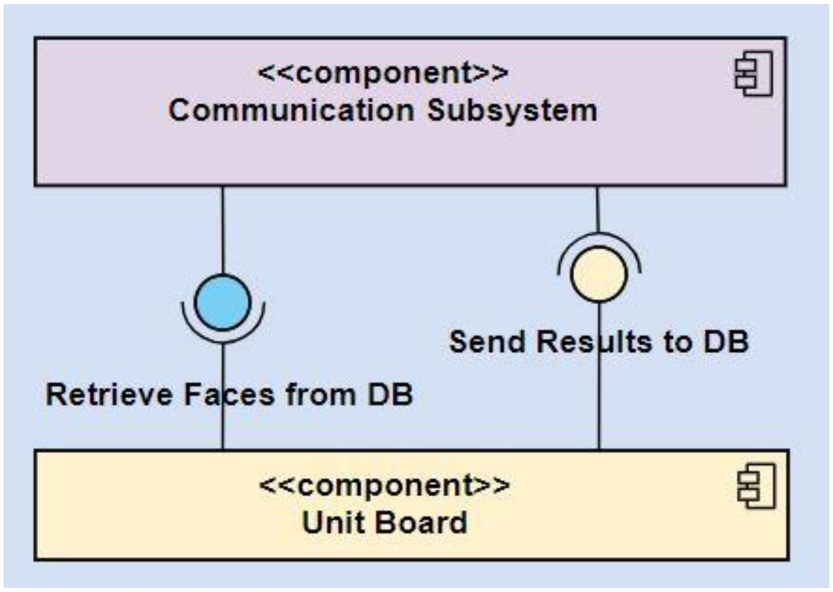

Figure 3 Communication Subsystem - Unit Board

Each unit board will be responsible for taking the pictures of the class, detecting the faces and performing face recognition. In order to do this, this subsystem needs the comparison IDs in order to recognize the faces. Hence the Communication Subsystem will provide specific *retrieve* and *send* services which will send/retrieve these data to the unit boards. The results of face recognition have to be sent to the database and this will be done through the communication subsystem again.

4.2 Communication-Encryption Subsystem Related Services

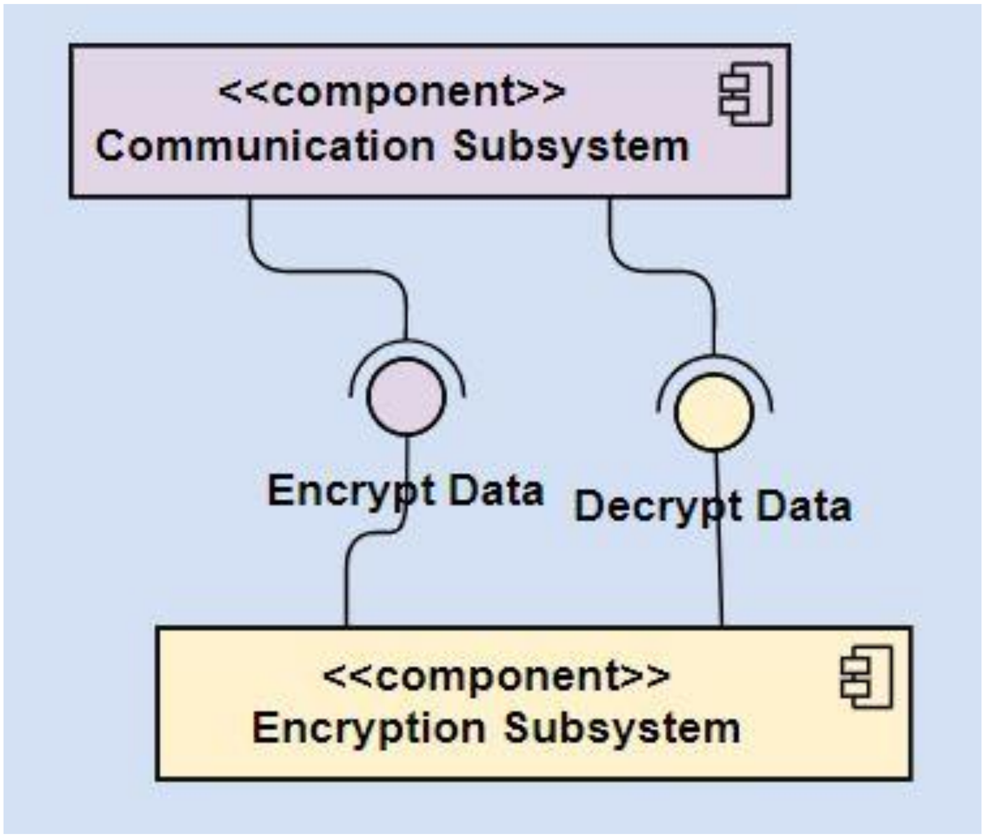

Figure 4 Communication Subsystem - Encryption Subsystem

For privacy purposes, all the data flow will be encrypted. The Communication Subsystem will be the channel of transportation, however it will be using **two services** from the Encryption subsystem; **encrypt data and decrypt data** in order to make sure that the data is initially secured on each data transaction.

4.3 Communication-Server Subsystem Related Services

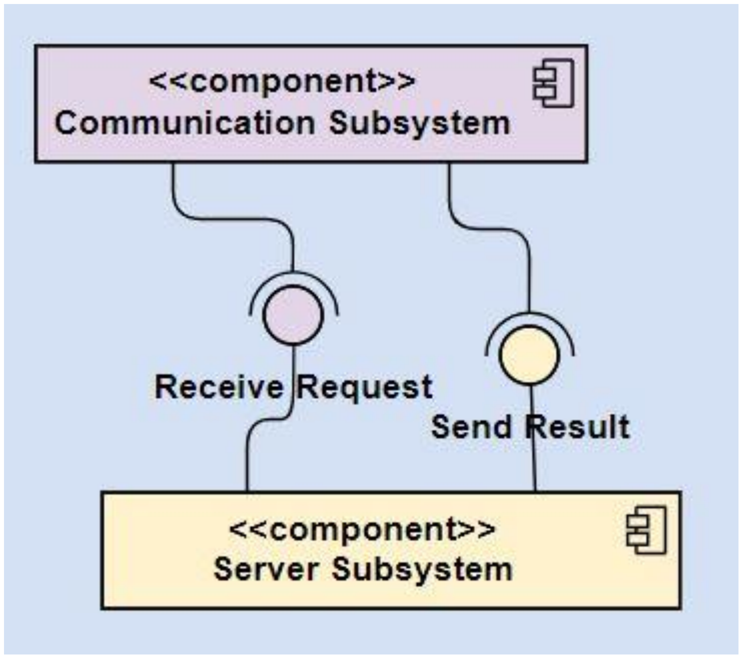

Figure 5 Communication Subsystem - Server Subsystem

The Server subsystem will need the data from the database in order to perform face recognition. Hence just as the unit boards, the server subsystem will also communicate with the Communication Subsystem by using two services: ***receive request*** and ***send result*** in order to facilitate the inter-component communication

4.4 Face Detection Subsystem Related Services

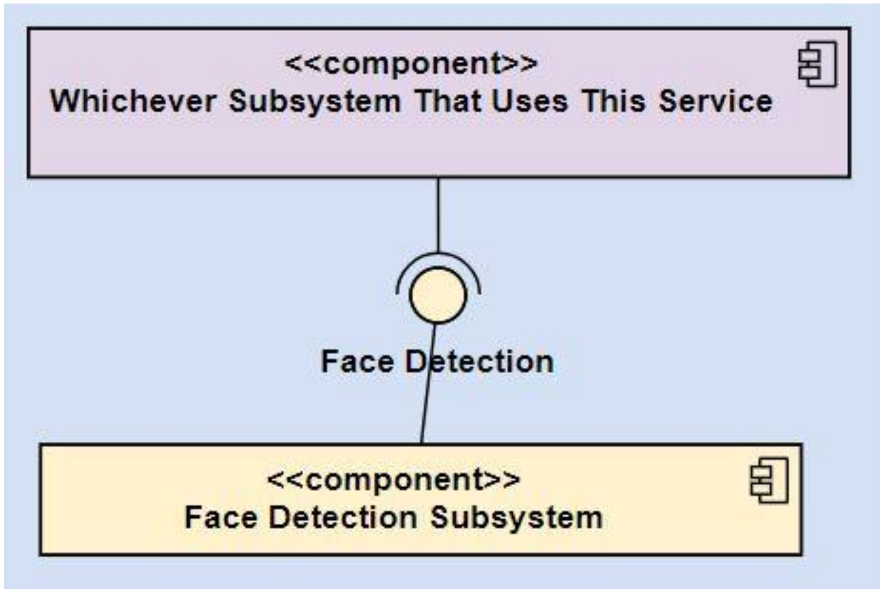

Figure 6 Face Detection - Using Subsystems

Face Detection subsystem will be responsible for detecting the faces in the picture take. However, either the Server or the Unit Board will not be aware of the inner structure of the Face Detection Subsystem. Instead they will be using a service the ***face detection* service** to get the required result. The interiors of Face Detection Subsystem are of no “interest” to the subsystems using it. This in turn will make communication much easier and less complicated.

4.5 Face Recognition Subsystem Related Services

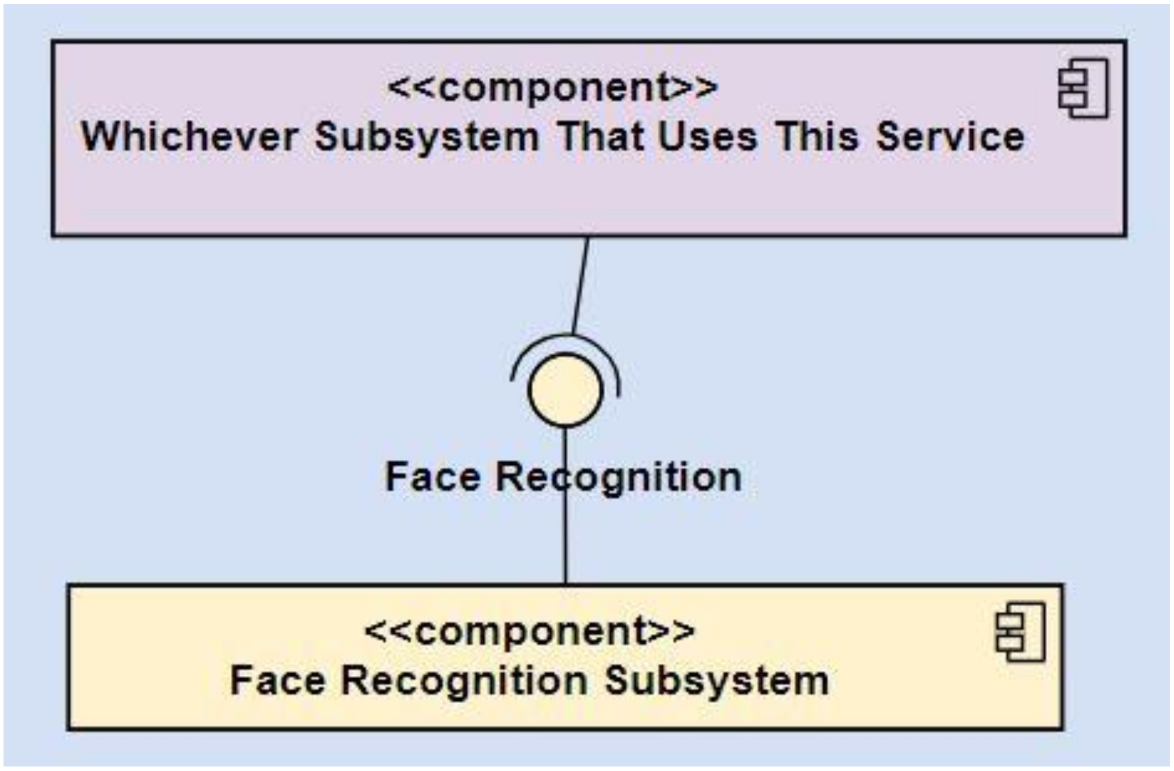
<<component>>
Whichever Subsystem That Uses This Service
Face Recognition
<<component>>
Face Recognition Subsystem

Figure 7 Face Recognition - Using Subsystems

Face Recognition subsystem will be responsible for recognizing the faces detected in the picture and its correlation to the subsystems using it is similar to the Face Detection Subsystem. However, either the Server or the Unit Board will not be aware of the inner structure of the Face Recognition Subsystem. Instead they will be using a service; the ***face recognition* service** to get the required result. The interiors of Face Recognition Subsystem are of no “interest” to the subsystems using it. This in turn will make communication much easier and less complicated.

4.6 Communication-Storage Subsystem Related Services

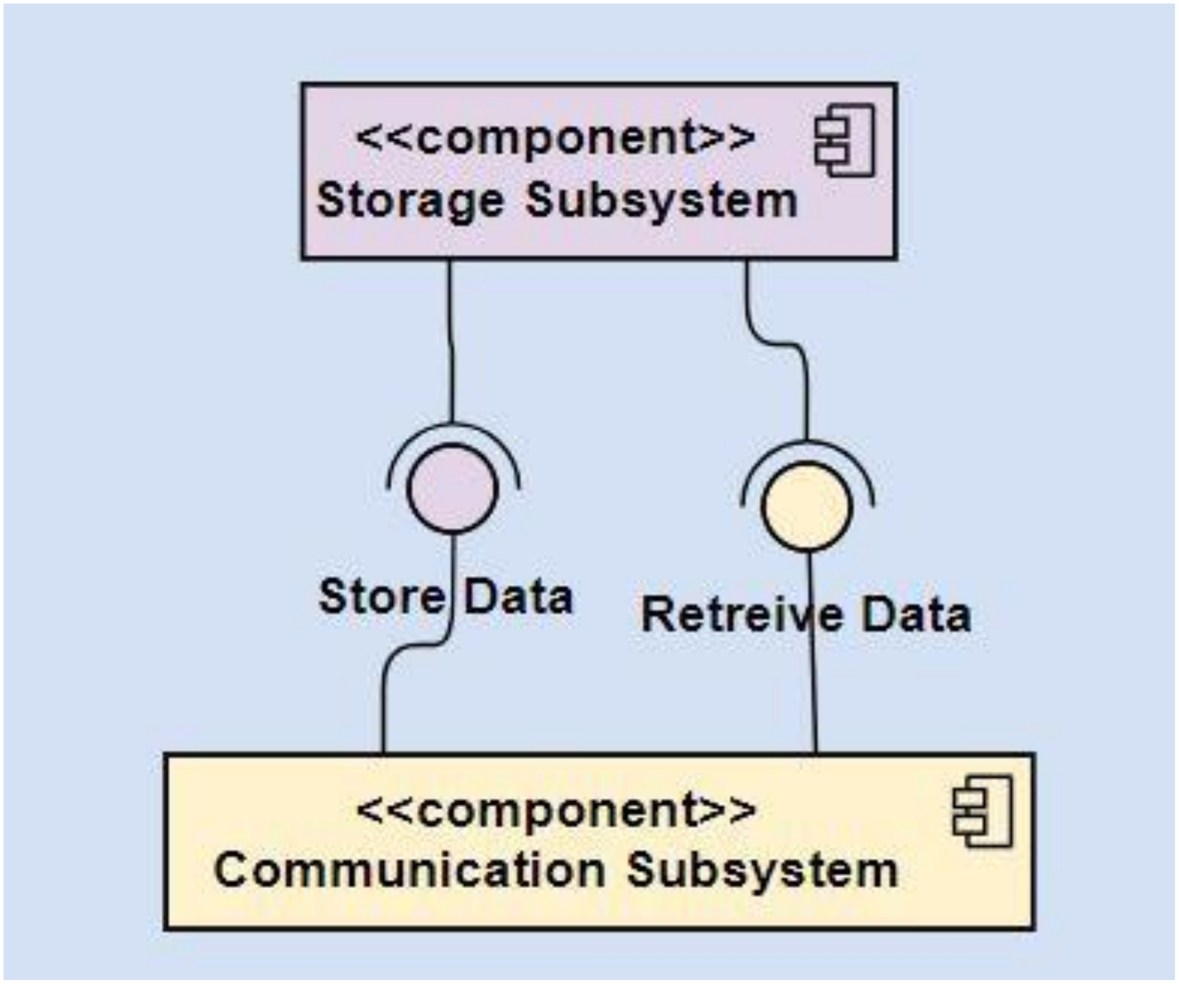

Figure 8 Storage Subsystem - Communication Subsystem

The User subsystem, the Unit Boards and the Server need to access the data stored in the database and also to send data to it. However they do not have direct contact with either the Storage Subsystem, neither the Database. This is because they use the communication subsystem as a channel of information flow. The latter is responsible to delegate the required requests (only after authenticated) towards the Storage Subsystem. Surely the Communication does not need to know how the Storage Subsystem will deal with the request. Instead the Storage Subsystem will offer two exterior services ; **store data** and **retrieve data** in order to facilitate the delegation of request between the two subsystems.

4.7 Communication-User Subsystem Related Services

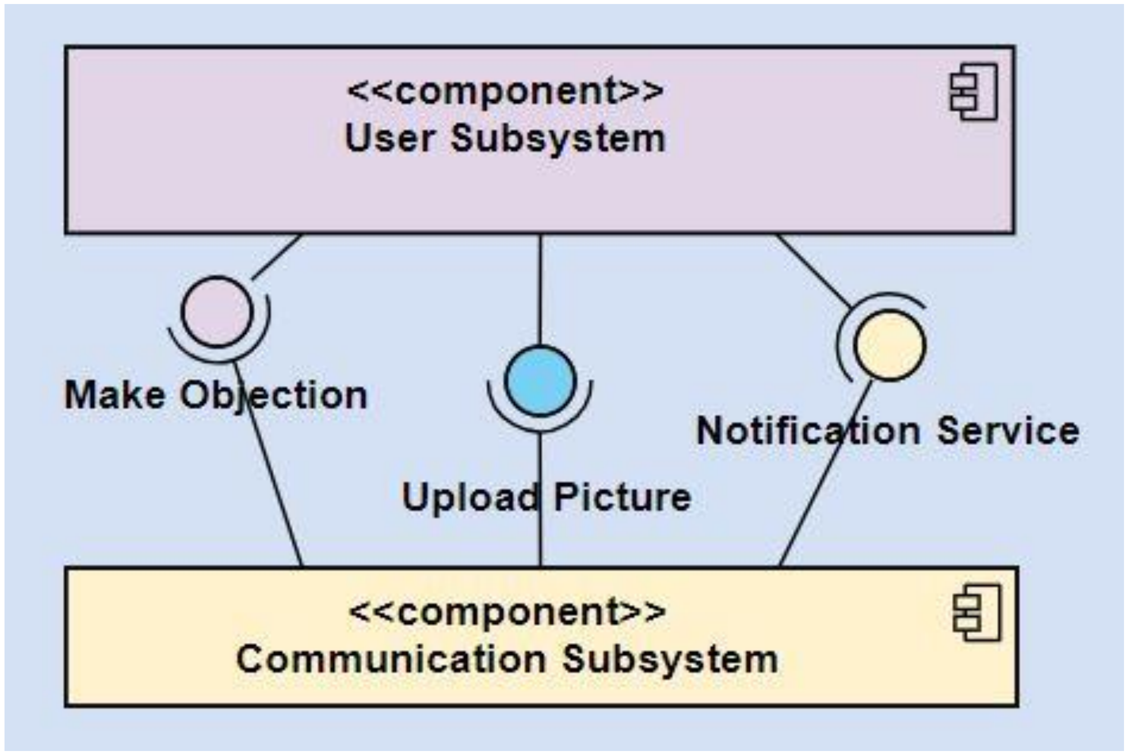

Figure 9 User Subsystem - Communication Subsystem

The User Subsystem will also communicate with the Communication Subsystem. The user subsystem has two services which are **Make Objection** and **Upload Picture**. The Make Objection Service is used to make an objection whenever the system doesn't recognize a student. Upload Picture Service is used to uload a new picture by the user in order to increase the possibility of the system to recognize the student or in order to help the system recognize the student even if they might have done any change in their apperance. The Communication Subsystem will provide the **Notification Service** to the User subsystem. Through this service, the user will get notified whether it has been recognized by the system and marked as present or not.

4.8 Communication-Authentication Subsystem Related Services

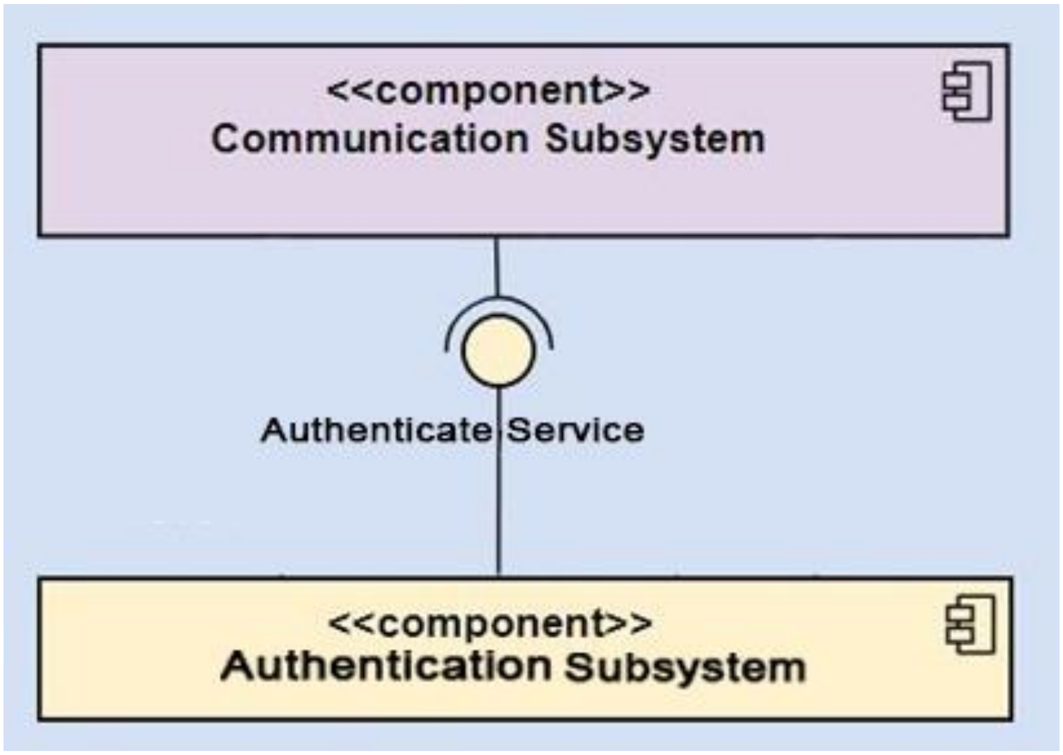

Figure 10 Communication Subsystem – Authentication Subsystem

The Communication Subsystem will be the passage way for requsts sent throughout the system, as such it needs mechanisms to only accept the requsts from parts of the AATS system. This is done also to ensure data safety as not everybody should have access to the data that the AATS system holds. The Authentication Subsystem will provide the **Authenticate Service** to the Communication Subsystem. This service in turn will be used to authenticate every incoming message or request.

5. Glossary

MySQL – A Structured Query Language
Unit Board – Will be a board computer (e.g. raspberry pi 3) which will have a camera module. This unit will take the pictures and process them.
FTP – is a file transferring protocol used to transfer files through the internet
JDBC – is a database independent API, that will provide means to storing persistent objects in a rational database

6. References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 3nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2010, ISBN: 0-13-606125-7