KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

Smart campus project

Senior Project Final Report

Sultan Alotaibi 200891300
Ali Alhussaini 200818540
5/10/2014

Prepared for:

Dr. Mohammad Elrabaa

Contents

1 Introduction

The idea behind the project is to design a smart campus that has a set of features to meet with the technical requirements provided by our advisor. The project is proposed to introduce a new solution, namely to have a smart campus that is able to do the following:

- Deliver messages to recipients
- Give ability to purchase without cash
- Automatic attendance
- Provide intelligent access control
- Environment aware
- Real time data
- More services could be implementable on top of the architecture.

The project is directed towards utilizing the technology out there into creating a network of things that helps make the system more useful and provide high quality services in cooperation with many other services.

2 Problem Statements

Motivation:

The project is proposed to eliminate the following:

- Wasted time, and thus money:

  If we can utilize technology in the workplace and or campus to save time and make people more productive, then it will subsequently save money.
- Human error:

  By automating work, errors that can happen by human miscalculations will be eliminated.
- Identity fraud:

  Implementing smart authentication systems will ultimately result in the elimination of Identity fraud.
- Inconvenience:

  In conventional ways, there are many things that are implemented in an inconvenient fashion. The project is set out to provide convenience to its users by eliminating the conventional way of doing things.

Impact:

Campus community will have more productive day having most routine jobs automated where they get to focus on jobs that are best done by people.

3 Project Specifications

Customer requirements

- Non-invasive:

  The user shouldn't find the service invasive by, for example, wearing bracelets or having to carry cards, Keys, cash … etc. As much as possible. Unless it's highly needed with no feasible alternative.
- Convenient:

  The System must provide convenience by getting things done as fast and accurate as possible when compared to conventional ways.
- To be Modular:

  Added services should work with the available services.
- Efficient:

  Doesn't burden the infrastructure un-necessarily.

Technical Specifications

- Automated

  Doesn't need human interaction
- Bandwidth efficient.

  The system must utilize the hardware provided and it shouldn't be hogging the network bandwidth. If more Bandwidth is needed then the infrastructure must accommodate for that change.
- Modularity

  The system must provide modularity , meaning that more services can be built on top of it as well as services can communicate with other services simply with a goal of making it a PnP.

4 Teamwork

Responsibilities & Contributions:

- Sultan:
  - Research SOAP & XML:

    This task was done by reading about the services and evaluating if they were needed for our project. We spent a few weeks to get familiar with them but ended up excluding them either for being old and somewhat obsolete, in the case of SOAP, or the availability of better and easier interfaces to work with, in the case of XML.
  - Decide on Hardware:

For the hardware, we had to make one decision for our implemented service. That is, to decide on what we need as an image detection node. We had evaluated different options and have decided to use Raspberry Pi and its camera module for fast and efficient processing.

  - Work on Image detection software:

The software associated with the end node was implemented in C++ and OpenCV libraries for face detection.

  - Prototype implementation:

For the prototype, we had to work with a workstation for rapid development and ease of use. This work will be deployed on the hardware chosen later.

- Ali:
  - Research SOA: do some research about Service oriented architecture which yielded that it focuses on separating the architecture into simple module to make it simple to add or modify services without affecting other services.
  - Research Image detection and recognition: after some research by both team members they found that the optimal way to implement image recognition is by using the openCV library.
  - Order needed hardware: after the hardware have been decided on. Ali ordered the parts and received the shipment from UPS.
  - Implement architecture: utilizing the decided upon factors, we had to implement the architecture that we have designed to support the deployment of the prototype.

Expertise

- Sultan:
  - Web development for front end
  - Raspberry PI operation
  - Report writing & review
  - Hardware ordering from the web
- Ali:
  - Database management
  - Architecture development
  - Project analysis
  - Application development
  - Document formatting

5 Engineering Design

Completely document the project design. Use graphical illustrations as much as you can.

5.1 Architecture

Sub-function identification

- take a still photo of subjects.
- detect people and count them.
- extract features.
- analyze features.

- mark students that attended
- update the DB
- allow unidentified students to be added manually

System architecture and components

- attendance program
- image detection
- image recognition
- Database
- web server
- camera

Hardware vs. software components

- Software:
  - attendance program
  - image detection
  - image recognition
  - Database
  - web server

- Hardware:
  - camera

Functions of each component

- attendance program:
  - get the class roster for each room from DB
  - request an image of each student in a room
  - send images to identification software
  - if an image is unidentified set alarm on alarm component
  - record attendance back to the DB
- image detection:
  - when the program receives a request take an image
  - crop faces from the image
  - send the cropped faces back to the requester
- image recognition:
  - when an image is received find the subject in the image
  - send the ID of the image owner back to the requester
  - if the image is unidentified send back an error
- DB:
  - store records of students, classes and rooms
  - room records have all rooms with ip address of their image detection module
  - class records have all sections with their time , location ,instructors and student rosters
  - Student records have all students with their IDs.
- web server:
  - authenticates instructors

- displays list of classes
- when a class is selected display roster with attendance record
- if some images were not recognized, allow the instructor to identify them manually and the images are used to further train the image recognition module.

• Interfaces between components
Image detection > Attendance server (HTTP).
Attendance server > DB (SQL)
Image recognition > Attendance server (HTTP).

5.2 Design Decisions

For the design, we have chosen to have 4 main components. Namely: Attendance server, database, image recognition server and image detection node. We have decided to split the detection from the recognition due to the load of the recognition on the hardware chosen as well as the limited storage capacity for storing the data.

Moreover, splitting the detection from the recognition adds modularity to the system, which includes that replacing any entity doesn't affect the others.

Below are the hardware and software choices that we had to make, along with the justifications.

Examined design options

Hardware:

Image detection node:
Arduino Microcontroller.
Mobile phone
Raspberry pi.
Workstation with web camera.

For prototyping, we chose the raspberry pi since it has a full OS that will take care of low-level development including the network stack. Also, it's very affordable even counting the additional camera.

Software:

Libraries:

Matlab:

Easy to use for prototyping, but relatively slow when compared to other options. It also requires licensing.

OpenCV:

This library is the choice for the project. It's Free to use and build on top of. It's also well documented and fast. However, It's relatively more complex to work with.

Self-developed:

Although, it would be highly beneficial for the team, but to develop the libraries needed would be unnecessary for the task as well as extremely hard.

Database solutions:

Active directory:

Allows the prototype to utilize established infrastructure unfortunately the server used for the prototype is on a production environment so AD can’t be used.

MySQL:

MySQL is widely used, cross platform and also secure. It can also handle huge amounts of data with ease. Therefore, we chose it for our system.

Criteria for choosing the adopted design options:

All of our choices took these points in consideration:

- Documentation for ease of development and later extendibility.
- Open source, hence extendable.
- Adds modularity
- Well-known interfaces.
- Simple and efficient.

5.3 Component Design and Implementation

For the hardware, we used the Raspberry since it provides the CPU power needed to do the image recognition and detection. It is considered a small-sized computer with a full OS that’s based on Linux. This elevates the issue of having to implement the network stack and having to deal with low level coding on the hardware interfaces assuming we chose other hardware such as Arduino or built our own.

Also, the software used is Open CV since it is suitable for our needs and is open source, hence completely free to use.

We wrote our own communication software to communicate between nodes as well as some code based on Open CV libraries for our use case.

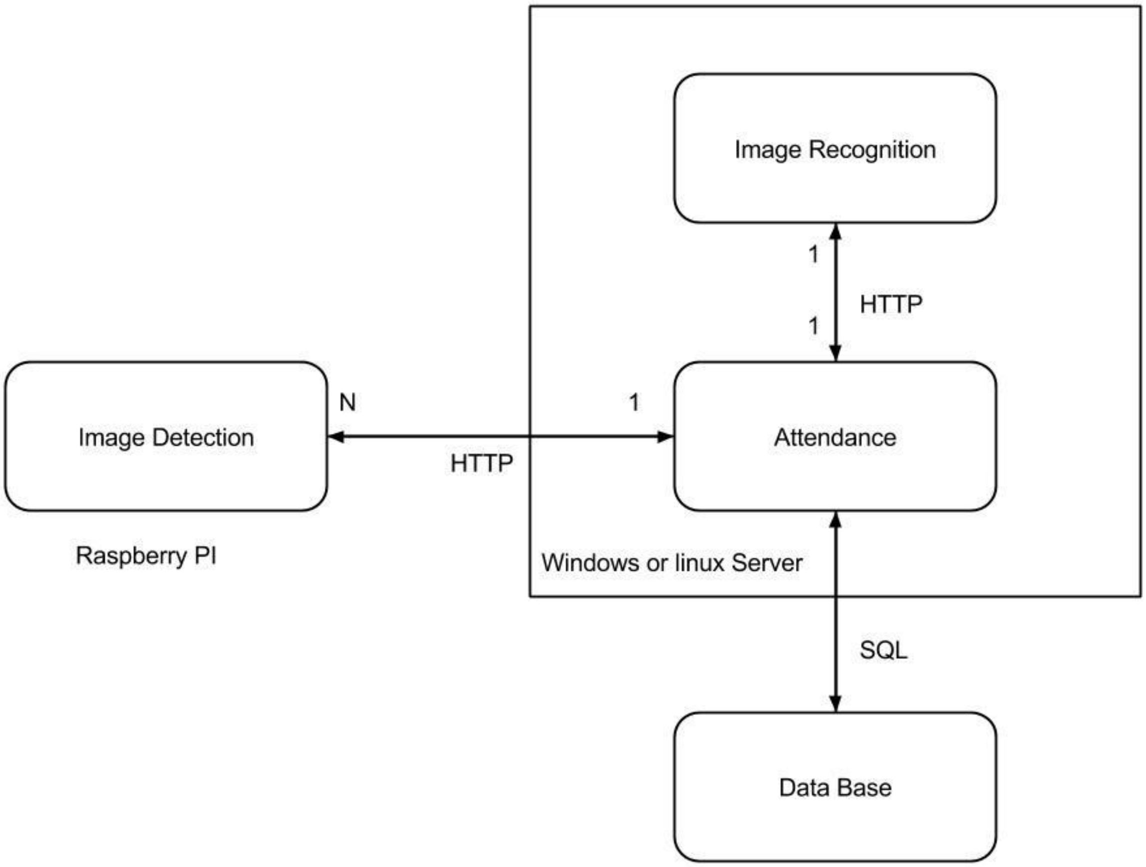

Figure1: Relation interfaces between components

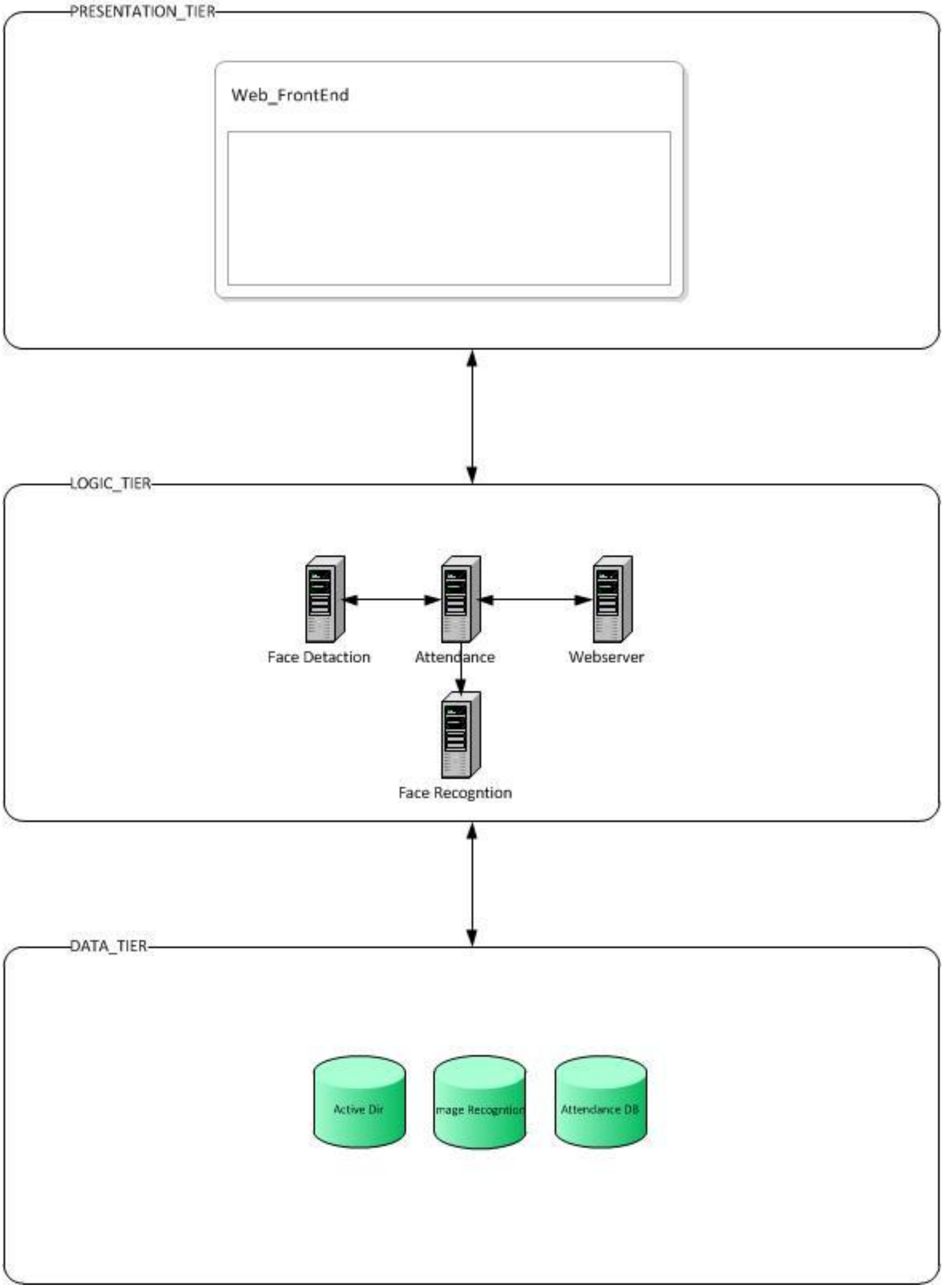

Figure 2: Three tier architecture

DB schema

Students:

String ID	String Name

Rooms:

String RoomNumber	String IPAddress	Int Port

Class:

Int CRN	int SectionNumber	String courseName	String instructorName	Time Time	
boolean Sunday	boolean Monday	boolean tuesday	boolean wednesday	boolean thursday	String RoomNumber

StudentClass:

int key	Int CRN	String ID

StudentAttendance:

int StudentClass.key	Date Date

Pseudo code

Image detection:

```
{
pic p = new pic(); // pic obj

pic[] a = new pic(); // array of type pic (dynamic , or )

int n;

main(){

        // Start camera comm.
        camera.start();

        // use camera module to take picture and assign it
to
```

```
    p = camera.takepicture();

    //Create a preview window with name "preview" and
size auto.
    window.create("preview", autosize);

    //Put that camera picture in the view.
    window.attach("preview" , p);

    //Use a method called analyze to find the people in
the picture and count them. it returns the counted
people.
    n = analyze ( p );

    //Create a socket object and start it with given IP
and port ( Server IP and port # 80 or 21)
    socekt s = new socket();
    s.opensocket(ip,port);
    s.httpSend(n);

    //use a for loop to crop the people one at a time
and store them in an array.  then send those on the
created socket through HTTP or FTP.
    for ( i=0;i<n;i++){

    a[i] = imagecrop(p);

    s.httpSend(a[i]);
    }

    return 0;
    }
}
```

Image recognition:

```
{
file f openFile(socket.cfg);
char ip[16]
int port ;
fScanf(“%s%i”,ip,port)
openSocket(ip,port);
while(1)
{
    Image img = reciveHtmlImg();
    if((string name =getName.(img)==NULL)
    //face not recognised
        sendHTMLString(error);
        sendHtmlImg(img); //send the image back;
    else
```

```
            sendHTMLString(name);
        //send the name of the person
    }
    }
```

Attendance:

```
    {
    string ip,recIp;
    int port ,recPort;


    main()
    {
        file f openFile(socket.cfg);
        fScanf(“%s%i”,ip,port,recIp,RecPort);
        //read the ip and port of the attendance image
        //recognition from config file
        openSocket(ip,port);

        while(1)
        {
            if (time.minutes()!=0)
            //time is 1:00 or 2:00 etc...
                sleepMinutes(60-time.minutes());
                //start at the start of an hour
            stringArray section sql(“select CRN from class
            where time=”+time.hour);
                //assuming classes start every hour

            for(int i=0,i<section.length(),i++)
                creatThread(Run,section[i]);
            //start a thread for each section
        }
    }




    Run(string sec)
    {
        sleep(600+(RND()%300))
        //resume after 10-15 minutes randomly to avoid
        //running all of the threads at the same time (10
        //minutes so that attendance is not taken before 10
        //minutes has passed)

        string classSocket =sql(select IPAddress ,Port from
        room right join class where SRN = sec);
        //get the socket for the image detector
        int port0 =RND();
        socket soc = openSocket(ip,port0);
        //creat a random socket


        sendHtmlRequest(soc,classSocket);
             //send request images using the socket

        int i =(int) reciveHtmlString(soc,classSocket);
        //number of images sent

        image img [i];//array to save the images

        int j = i;

        while(j>0)
        {
             j--;
             img [j]=reciveHtmlImg(soc,classSocket);
                  //get next image from image detection
        }
        j= i;
        Socket recSoc=openSocket(recIp,recPort);
        while(j>0)
        {
             j--;
             sendImage(soc,recSoc,img[j]);
             if((string name =reciveHtmlString())==”error”)
                  saveFile(imagerecived,sec +”//”+
                  (string)Date+”.”+j);
        //save image with name “date.img number” eg.”14-5-
        //5.3” in section folder
             else
                  attendStudent(name,Sec);
        }
    }
    }
```

6 Issues

Issues faced and their resolution

OpenCV installation on raspberry pi:

Installing openCV for Raspberry pi is easy, but long. We had to download the source files and compile them locally for our machine. This took a very long time to build which was ~ 6 hours. First attempt was not successful due to lack of familiarity with the process, which cost us time assuming it wouldn't work. Another attempt was made to correctly make the files needed and was successful after a very long time due to CPU limitations on the Pi.

OpenCV linking to Raspberry Pi cam:

Using raspberry pi camera module in combination with the OpenCV library was a big issue. OpenCV, by default, carries no support for the communication mode for the Raspi cam since it uses a parallel – ribbon – cable to communicate. This issue was attempted to be solved by copying the raspicam operation code from source and modifying it to link with openCV. This was an unsuccessful attempt. Another attempt was made to install UV4L package that adds the drivers necessary for the raspberry pi camera module. This attempt lead to issues with dependencies on other packages that we dealt with one at a time but with more dependencies coming up. And given the slow rate of development on the Raspberry pi due to CPU limitations, we decided we should move to a more rapid environment to develop the prototype , which is to work on an end station Installing OpenCV on Mac osx machine.

Linking OpenCV to eclipse IDE:

We have decided to use Eclipse IDE for development. However, there isn't a lot of documentation that gives a good idea of linking openCV with eclipse IDE. After reading multiple documentation, we have reached a conclusion on how to do that which we have documented for later use. This involves identifying the "include" and "lib" directories for the library and associating them with the compiler and the linker of the C++ language respectively. Namely, for each project, you have to do this step as well as naming the libraries you are going to use emitting the "lib" prefix and the ".dylib" suffix.

Access loss on VM provided by CCSE:

After logging in to the provided server, we have installed multiple software for the project. After installing Active directory, we have decided on a later time to remove since we were unable to use it for the prototype. This removed all the local accounts as well including our own account to login to the server.

This issue was resolved by contacting the CCSE system administrator who helped us regain access.

Team co-ordination:

This is not a technical challenge, but it was a challenge nonetheless. As a team of two, we had a lot of issues coordinating meetings. This problem persisted for a good part of the project time especially when more busy portions of the semester took place. However we had to deal with it by setting meeting time long ahead of those meetings as well as defining the meeting agenda so that we can discuss it and make better use of the meeting time.

Limitations and constraints of the design:

This design needed to meet the criteria set by the project advisor. Therefore, we had to choose components and services that provide convenience and not be invasive to the users of the system as well as be open and modular.

Limitations and constraints of the implementation:

As we were using a server that was in a production environment, we weren't able to use active directory as it can create some problems for the production environment as advised by the network administrator that provided the infrastructure for us. This forced us to drop the AD and have student records in the Database itself.

Since the outcome would be a prototype, it's hard to test in a big environment to assess the shortcomings of the implementation when use in a real world scenario.

7 Engineering Tools and Standards

Relevant available tools

For our use, we had to deal with making the choice for the IDE to do our C++ development as well as using the database software framework.

Some IDEs that we looked at were:

- XCODE
- Visual Studio
- Code blocks
- Eclipse CDT

For our use, we went with using Eclipse IDE CDT codenamed "juno". Reason being that we have been familiar with the tool for other projects. Moreover, the tool has no licensing issues and is cross-platform.

For the DB software framework, we had the choices of:

- MySQL workbench
- Microsoft SQL server management studio

After some research we found out that MySQL workbench is more suitable while we already have some experience in using it.

Relevant standards

For our use, we only explored two and they are the ones that will be used. Those are:

- **HTTP:**

  HTTP communication protocol will be used to send cropped images and necessary files between the image detection node and the attendance server. It will also be used between the attendance server and the image recognition server.

- **SQL:**

  SQL will be used between the attendance server and the database server.

8 Conclusion

What was learned

As part of the project, the team was exposed to many new technologies including different computer vision algorithms and the way to use them as well as different classification schemes such as Haar-based features and LBP (Local Binary Pattern).

We have also learned about the combination of different weak classifiers (Cascade classifiers) to form a strong classifier which is used to classify objects for detection and recognition use cases.

What would you do differently in a similar project

For this project, we have made the choices to try to meet the requirements as accurately as possible. This has lead to embracing computer vision to achieve the requirement of being “none – invasive “. However, this choice added a complex overhead to the project development which required the learning of a lot of new aspects that we were never exposed to, namely: Computer vision. This was a highly beneficial experience however; it would have produced a better final outcome if we had worked with something that we had a solid background with such as RFID. This wouldn’t meet the requirement in an optimal way, but would have generated a more complete outcome for the project, and left more time for testing and troubleshooting.

Conclusion

This project has been an interesting experience. It had led us to exchange our expertise as a team and learn new things like how to interact with OpenCV. This as a whole added a lot of value to our knowledge as computer engineers and had helped us in knowing where to get the right information for troubleshooting numerous problems that we faced throughout the project.