UNIVERSITY OF
KWAZULU-NATAL

ELECTRONIC/COMPUTER ENGINEERING DESIGN 3

GROUP 7-PHASE 3 REPORT

SMART EXAM HALL/LECTURE ROOM

06/05/2019

NAME	STUDENT NUMBER	UNIT	SIGNATURE
Devashnee Bhagawandin	214502393	Database and Software Integration	
Rishay Ramcharan	214504863	RF ID Authentication	
Keshav Jeewanlall	213508238	Biometric Identification and Video Surveillance	
Sashin Ramdhani	214513737	Energy Efficiency, Safety System and RF Detection	

Abstract

This report entails the research, product specifications and feasibility study and the complete design and implementation for both hardware and software aspects of the Smart Exam/Lecture Room. This report also entails an explanation and discussion into the different subsystems in which the system was broken down into.

This project design aims to turn the education and training environment into an eco-friendlier& safer zone using technology to make the room more energy efficient and more secure.

The system is to be designed and demonstrated by a group of fourth year students belonging to the Computer and Electronic Engineering disciplines at the University of Kwa-Zulu Natal, Howard Campus.Some basic levels of prototyping, testing and debugging were conducted and documented in this report.

Contents

1. Introduction

The following project design aims to take the training and educational industry into the future by automating the taking of registers using two-way authentication. The project also aims to turn this environment into an eco-friendlier zone using technology to make the room more energy efficient. This project also helps to prevent cheating and foul play which could take place in an examination room.

2. Problem Statement

The client requested a technological solution which will eliminate the paper work of taking a physical register. The technology should help eliminate cheating and imposters entering the examination hall.

The client requested that the hall should be as eco-friendly as possible. This is to reduce their carbon footprint. The examination hall should have safety features and make the writing conditions comfortable.

3. System specifications

3.1. Functional Specifications

- Controlled access to room
- Two-way authentication
- Student cards and facial recognition
- Automated register control
- No student is to be greater than 20 minutes late for an exam/lecture
- Detect cell phones being used
- Camera present in case of enquiry
- Motion sensors for optimal lighting
- Temperature sensors for optimal temperature control
- Fire alarm over-rides the system and opens the service doors for faster exit
- Audio for announcements
- Over-ride function
- Camera surveillance is controlled via the access control and motion sensors

3.2. Non-functional specifications

- Reliability.
- Usability.
- Data integrity.
- Performance.
- Maintainability.

4. System and Design Constrains

- Time management, the project to be completed within three months
- Budget constrains for component costing
- Access for required design material and components due to lack of availability

5. Problem Solution

The proposed solution to the above-mentioned problem is a smart exam hall/lecture venue and surveillance system. The two-way authentication will require a RFID card which will hold the unique ID of the student will be linked to a data-base. The database and required software will have to perform several tasks which will determine if the student will enter the venue using two-way authentication.

The optimal energy efficiency for lighting and air conditioning will be achieved using motion sensors for lighting and temperature control for air-conditioning. The hall is divided into grids and depending on the motion detection in that grid the area will light up. Temperature sensors are used for controlling the temperature of the hall. A required predetermined temperature is set and depending on this the air conditioners are operated. Smoke detectors will be used in case of possible fires. An automatic override will occur if a fire is detected.

The service entrance is password access. The service personal will require a password which will allow them access to the service door. These passwords will be updated periodically for security reasons.

RF detectors will be used to detect radio frequency signals emitted by cellular phones or other similar electronic devices. This will eliminate the chance of cheating during an exam.

6. System Block Diagram

Figure 1: Complete system block diagram

7. Identification of Different Project Economic Solutions

By using RFID cards as the only method of authentication, unwanted people can easily enter the venue if they possess someone else's ID card. Hence a biometric authentication will be required to reduce the chance of this situation.

Possible biometric authentication solutions

- Finger print scanning (R600)
- Facial recognition (R300)

Surveillance of the venue with fixed cameras is required. Coverage area might be limited if a larger venue is to be used as more cameras are required to monitor the entire room. Hence a bigger budget would be needed.

8. Database and Computer application (Devashnee Bhagawandin 214502393)

8.1. Computer Application

The system requires a computer application which integrates the sub-systems to complete the system and interact as a completed unit. The computer application will follow the flow diagram of the in figure 2. The system will be taking signals from the sub-systems.

Referring to figure 2 the system will perform a two-way authentication which will perform a bio-metric test (refer to bio-metric testing) and will perform authentication with database analysis to give access to the entering students. The access granted signal needs to be processed and update the automated register.

The software application needs to access the database to update the record of exams and students. The applications need to update passwords and add and deleted student records if needed.

The software application takes signals from the fire-alarm and over-rides the system. The service door application will access the database to check if the over-ride password is correct and send a signal to the service door sub-system allowing the door to open.

The RF detector will send signals to the main application indicating there is a presence of a communications signal.

8.1.1. Functional requirements

- Update student records
- Delete student records
- Receive and send signals to the RF ID sub-system. This is required for authentication process.
- Receive and send signals to facial recognition sub-system required for updating the register
- Receive and send signals to the service door sub-system for opening the door.
- Receive signals from RF detector unit
- Receive signal from fire-alarm
- Send signal to service door sub-system to over-ride and open

8.1.2. Non-functional requirements

- Reliability.
- Usability.

- Data integrity.
- Performance.
- Maintainability.

8.2. Database

The database is required for the updating and storage of the records of registered students for the clients. The database contains dates of examinations. Students registered for the modules. Students duly performance recorded. An automated register keeping records of students.

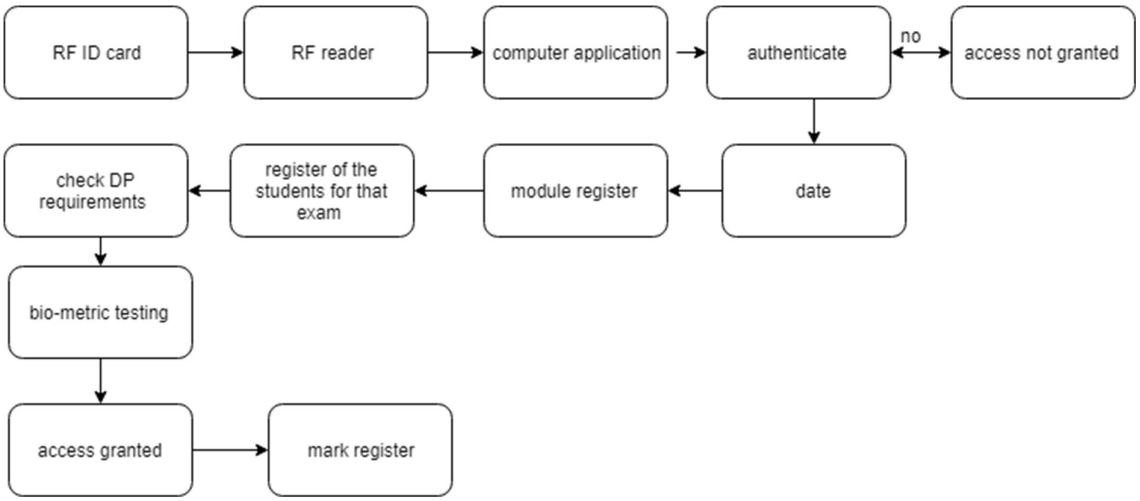


Figure 2: Database

The follow diagram above indicates the flow diagram of the two-way authentication process. The database will access the records for the current date. The date will display the examinations scheduled for the day and access the students registered for those modules. If the student is registered, then the duly performance requirements are checked. If the student meets all the above requirements, then the bio-metric testing will occur. Once the two testing methods are passed then the student is granted access to the venue.

8.2.1. Functional requirements

- Keep record of registered students
- Record of modules
- Dates of examinations
- Register of attendance
- Duly performance requirements

8.2.2. Non-functional requirements

- Reliability.
- Usability.
- Data integrity.
- Performance.

Maintainability.

8.3. System Design

8.3.1. Use case diagram

The basic operation of the system can be defined and modelled using a Use Case diagram; in accordance to the Unified Modelling Language (UML). The actors are the systems or entities that are directly involved in operations with the system. For the PC Application and Server subsystem, the actors are:

- The computer application itself
- The student trying to access the examination venue
- The lectures involved in the modules
- The service personal that maintained of the venue

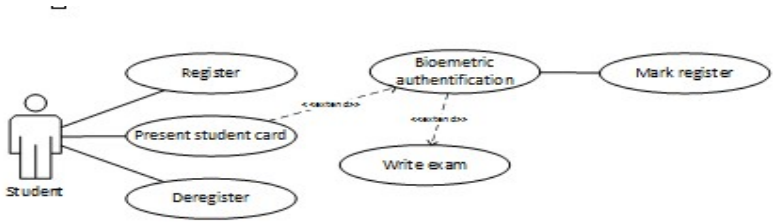


Figure 3: use case diagram of student class

8.3.1.1. Register Student

The above diagram displays the interaction of the student and the system. The student is required to register for modules. The modules which the student registers for is added to the database. The student will then be presented in a register of the module.

8.3.1.2. Enter Venue

The student is required to present an access card which will authenticate the student's identity. This card is to be tapped on the RF scanner provided at the controlled access door. If the ID card information is verified correctly, then the system will continue to biometric testing in the form of facial recognition. If the biometric testing is successful, the student can proceed to write the exam and an automated register can be taken.

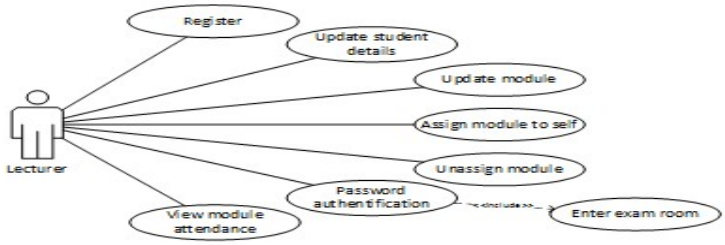


Figure 4: lecturer use case diagram

8.3.1.3. Lecturer Use case diagram

The lecture has to register himself as a lecturer and then will be given rights to the system.

The lecturer can check the register of the students attending the examination. The lecturer can also update the student's details if required and update the module details when required. The lecturer

can assign module to himself that he/she is responsible of. The lecturer can unassign modules if needed.

8.3.1.4. Lecturer Entering Examination Hall

The lecturer is required to get a password and identification to enter the examination hall. This information is generated by the system and given to the lecturer to enter via the service door entrance.

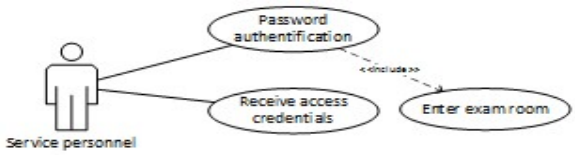


Figure 5: service personal

8.3.1.5. The service personal

The service personal is required to obtain a password to enter the examination hall. If the credentials are correct they can enter.

8.3.2. Classes

With analysis from the Use Cases, the PC Application and Server subsystem can be implemented using the following classes:

- The student class – record of student's personal information
- Modules class – module codes and modules offering
- Lecturer class – lecturers' personal information and modules responsible for
- Module register class – record of students registered for module and marks attendance
- Service door class - passwords generated for service personal

Table 1: student class table

Student class	
Description	The student class is needed to keep track of the students personal information and to keep track of identification characteristics
Attributes	ID : int ; the students ID number Full_name: string ; the students full names Surname: string ;students surnames Dob: date ; students date of birth Physical_add: string ; students physical address Email:string ; students email address Phone_number: string ; students contact numbers
Associations	Add_student() ; this function is used when a student is registering for modules . Delete_student(); function used to deregister students.
Frequency	This function is called everytime a student try to register and deregister the function is also called when the lecturer wants to update the students information.

Table 2: modules class

Modules	
Description	This class keeps a record of the modules the institution offers
Attributes	Module_name: string ; the name of the module Module_code: string ; the code of the module Credits:int ; the number of credits the module holds
Associations	Add_module(); adds the modules the institution offers Update_module(); updates the module details. Like the credits and type of module Deleted_module(); deleted the modules not offered
Frequency	Required when adding and deleting modules for student class to register

Table 3: module register

Module register	
Description	The module register keeps record of the students registered for the module and keeps the attendance of the students.
Attributes	Module_code: string ; keeps record of the module code and gets the module details from the module class. Date_of_exam: date ; this keeps record of the date of the examination to be written Students[]: array of students going to write the exam Dp_requirement : Boolean ; checks the students dp status Attendance: int ; marks the attendance of the student
Associations	Update_attendance(); updates the attendance of the students Check_status(); checks dp requirements
Frequency	The class is called every time a student's tries to enter the venue as the student's information needs to be verified. The student needs to be present on the class register of the examination of the day and pass all the tests to enter.

Table 4: lecturer class

Lecturer	
Description	The lecturer class holds the personal information of the lecturer. The class also gives admin rights to the lecturer.
Attributes	ID:int; stores the lecturer ID number Full_names:string ; stores the full names Surname:string ; stores surname Modules[] : array of modules
Associations	Add_lecturer(); registers the lecturer Update_student() ; updates the students details and dp requirement Assign_module() ; assign modules for lecturer Remove_module() ; removes modules assigned to lecturer previously View_attendence() ; view attendance of student
Frequency	Used every time a lecturer wants to view attendance ,add and deleted modules and update student details.

Table 5: Service Door Class

Service door	
Description	Keeps track of the service door passwords and access details for service personal and lecturers
Attributes	ID:int; the id of the person the password is assigned to Password: string ; the actual password assigned Date: date; the duration the password is valid
Associations	Add_access() ; adds the access by assigning ID and password
Frequency	Used everytime someone enters via service door

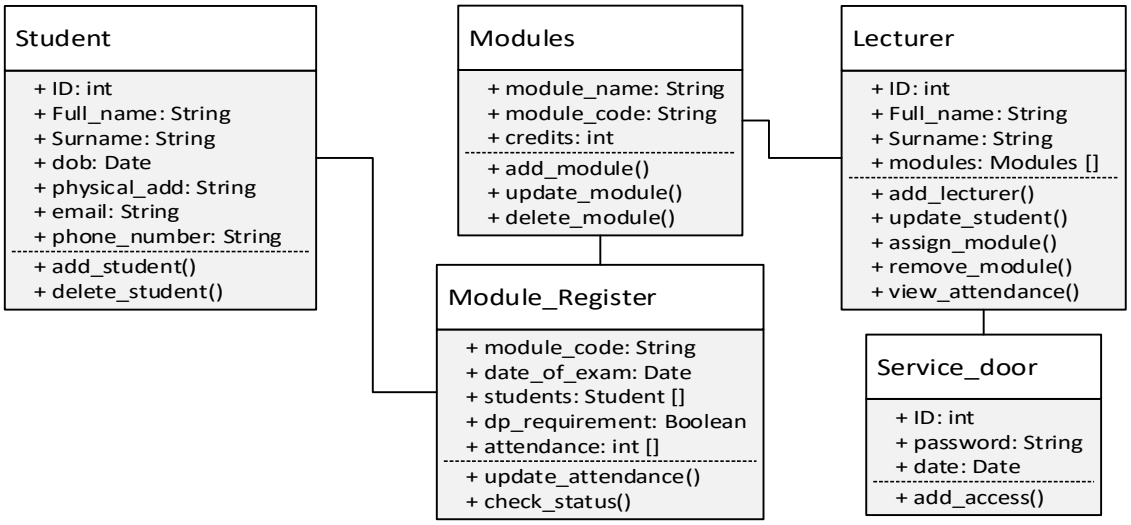


Figure 6: class diagram

8.3.3. Sequence Diagram

The sequence diagrams are used to show the flow and control of information between components and external stimuli. [1] The sequence diagrams are based on the use case diagrams and the class diagrams.

Student Sequence Diagram

The student sequence diagram shows the interaction of the student with the system. The student class shows two actors. The student and the computer application are both actors in the system. The computer application forms the link with each sub-system and integrates them.

The student needs to register for modules. When the student tries to register the system user interface interacts with the computer application. The computer application accesses the list of modules and the details of that module. If the student decides to register for that module the computer application will access the module register and add the student to that register. The student request also updates the database.

If a student decides to request entry to a venue the RF ID unit interface and sub-system interacts with the computer application. The computer application asks the authentication unit to perform a series of test to see if the student is on the list of students who can enter. The system doing the

authentication then does the bio-metric testing and returns a signal. The computer application sends a signal to mark the automated register and sub-systems open controlled access for the student.

The student can deregister from modules. When this occurs the user interface interacts with the computer application, the module register and database and removes the student.

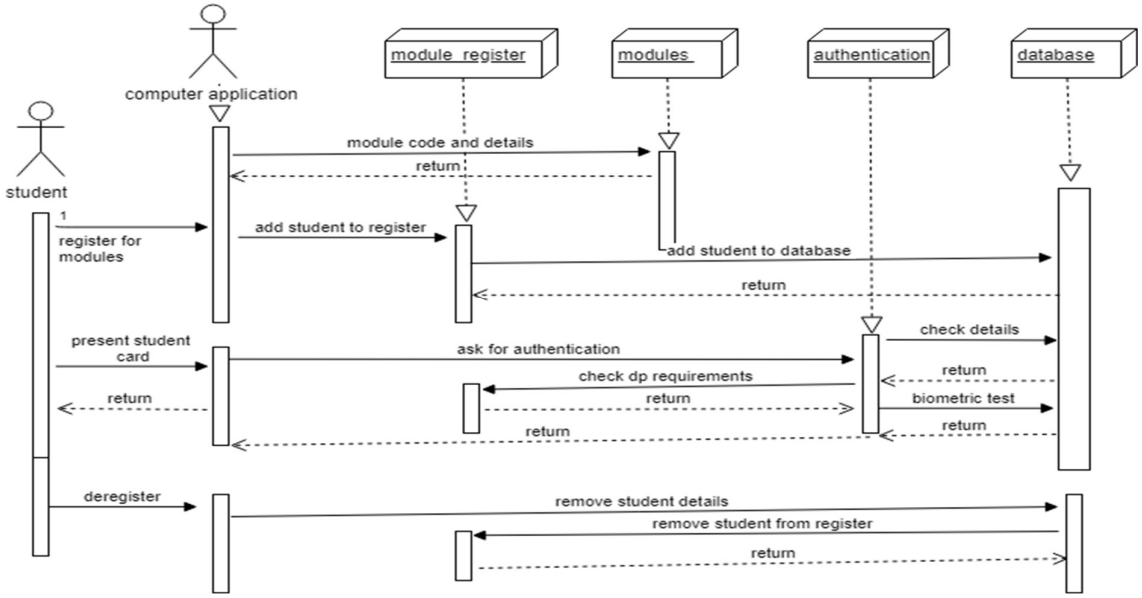


Figure 7: student sequence diagram

The Lecturer Sequence Diagram

The lecturer sequence diagram shows the interaction of the lecturer with the system. The lecturer class shows two actors. The student and the computer application are both actors in the system. The computer application forms the link with each sub-system and integrates them.

The lecturer must register himself/herself as a lecturer. The computer application will update the database with the lecturer details as shown below in the database design phase. The lecturer must then add modules to himself/herself. These modules are the modules the lecturer is responsible for. The system will return once the registration is completed.

The lecturer is allowed admin rights to the system. Such rights include the ability to update the student records. To view the attendance of the examination and edit duly performance requirements. The update process requests the computer application to update the student records via the student class. The student class update will update the database.

The lecturer requires to obtain a user ID and password to access the venue. The service door class generates the password and ID for the lecturer to enter via the service entry. When generation is completed the service returns and the computer application will delay the details.

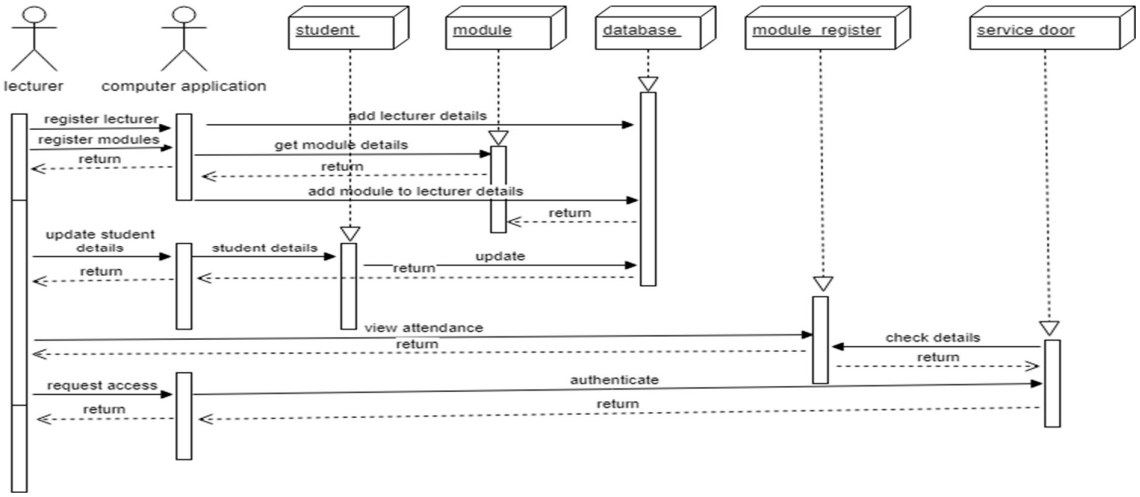


Figure 8: lecturer sequence diagram

The Service Door

The service door sequence diagram shows the interaction of the service personal with the system. The service door class shows two actors. The student and the computer application are both actors in the system. The computer application forms the link with each sub-system and integrates them.

The service personal and lecturers requires to request the system to give them a duration limited password and ID. The password and ID can be used to enter the service entry point which has a keypad and LCD .

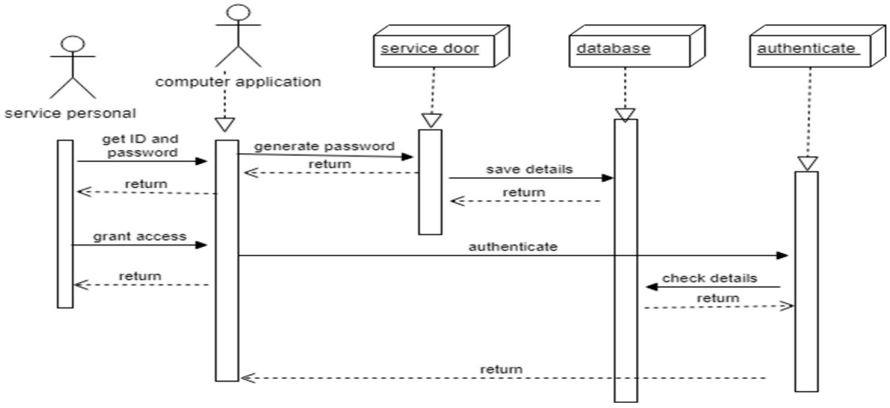


Figure 9: service door sequence diagram

8.3.4. Database design

The database is used to store the records of students, lecturers and modules the institution offers. The database has a record of student register for modules and attendance register. Each diagram below shows a table in the database.

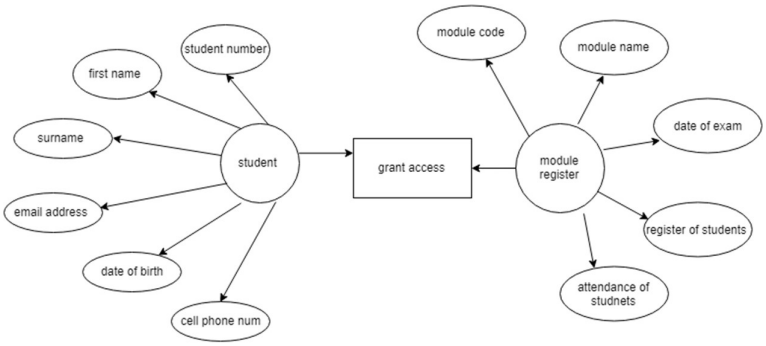


Figure 10: student class and module register class association

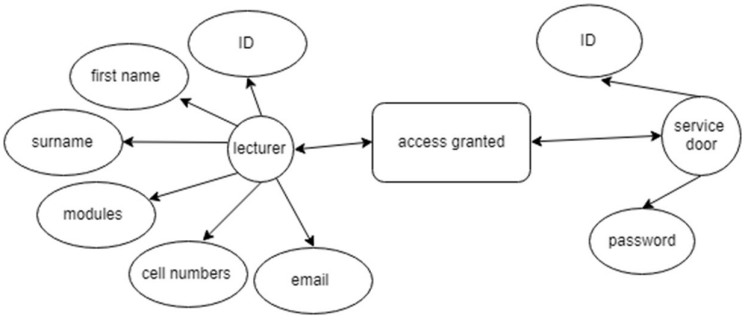


Figure 11: lecturer and service door database association

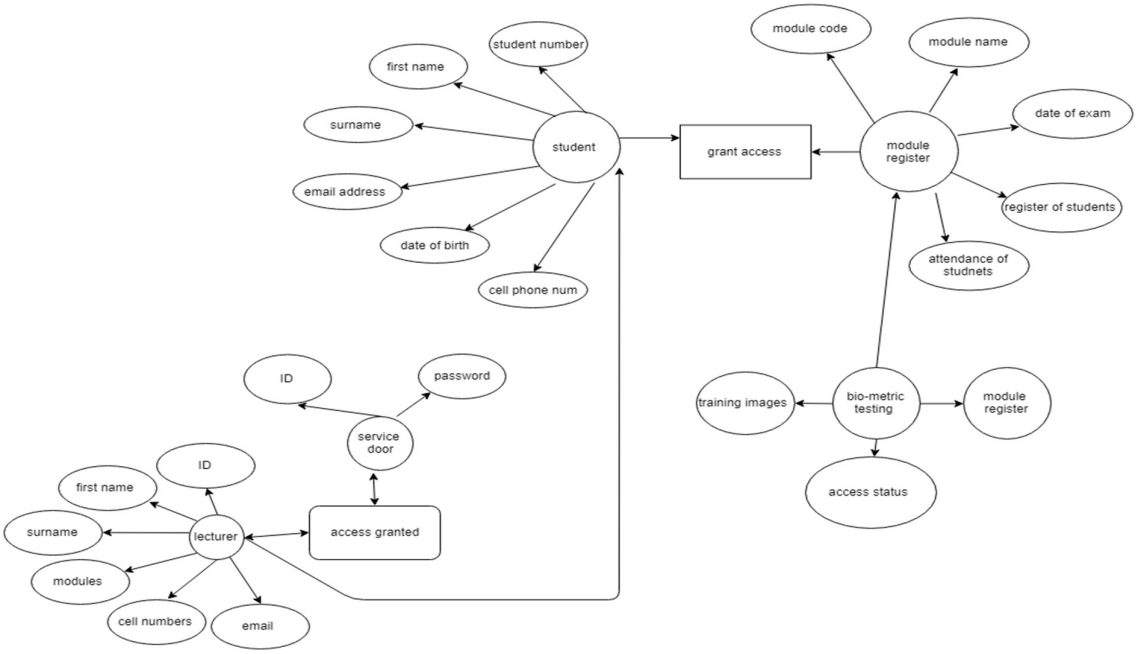


Figure 12: complete database design

8.3.5. Architectural Design

The architectural design consists of three layers. The layers are the user interface , the computer application layer and the communications link layer with the different sub-systems.

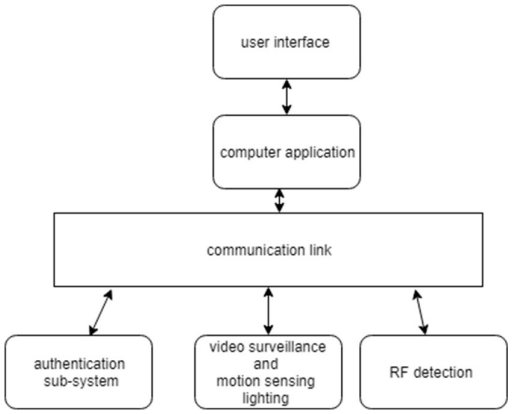


Figure 13: architectural design

8.3.6. Design choices

8.3.6.1. Computer Application

The IDE chosen for the Application development is Microsoft Visual Studio 2013. The reasons for this choice is as follows:

- Highly featured for C++ Development.
- Code syntax highlighting.
- Refactoring.
- IntelliSense makes debugging easier.
- Error highlighting.
- User friendly processes for integrating external libraries to projects.

8.3.6.2. Database

The MySQL Database software is chosen to create and maintain the database for this system. The reasons for this choice is as follows:

- Cross platform including C++ support.
- Designed for multi-threaded processing.
- Implements SQL functions using a highly optimized class library.
- Secure password system

8.3.6.3. Communication Between the subsystems

A wired connect using USART module and the hyper—terminal

8.4. Feasibility Study

The software application and database can either be designed to operate on a Single-Board Computer (SBC); i.e. Raspberry Pi; or a conventional desktop Personal Computer (PC).

Table 6: computer application alternatives

Single-Board Computer(SBC)	Desktop Personal Computer
PROS	PROS
• Uses less power. • Makes less noise. • Cheaper.	• Uses an internal Hard disk, which is faster and cheaper than an external hard disk used by the SBC. • Upgradable. • Faster data transfer rate between peripherals. • More general purpose rather than application specific. • More RAM.

Cost estimation for SBC:

Single-Board Chip (Raspberry Pi)	R750
External Hard Disk (1TB)	R900
Keyboard, mouse and monitor	R1500
TOTAL:	**R3650**

Cost estimation for Desktop PC:

Desktop PC with keyboard and mouse (1TB Hard Drive, 4 GB RAM)	R4000
Monitor	R1000
TOTAL:	**R5000**

Although the SBC costs much less than a desktop PC, there are more benefits to using a desktop PC as tabulated above. If the organization already possesses a desktop PC, the cost factor becomes irrelevant. If they do not possess one, it be an investment to the organization as the desktop PC can be used for many other applications. With this information.

8.5. Project Management

Development Phases due for Phase 2 with estimation on number of days:

- Design
    - System and Object Modelling (4 days)
    - Architectural Design (2 days)
    - Interface Design (1 day)
    - Component Design (1 day)
    - Database Design (2 days)
- Component Implementation (2 days)
- Testing and Debugging (2 days)
- Cost estimation study (2 days)
- Report (3 days)

8.6. Design and implementation

This section documents highs the vital implementation aspects of the design; which includes pieces of programming and pseudo code. The PC application was designed using Microsoft Visual Studio 2013;

which incorporates the Visual C++ language to create a Windows Form Applications that utilize Windows API and the .NET framework [2]. MySQL is used to host and manage the database.

8.6.1. Computer Application

The computer application consists of two c++ and two header files. When the project is run the main tread executes first. The main tread in this project was created to open the windows for application which was initialized in the header file. There after one other c++ and header file was created to handle the serial port communication between the sub-systems.

The main screen (*MyFrom.cpp*) gives access to the windows form application serial port header and cpp file.

```
Application::EnableVisualStyles();
Application::SetCompatibleTextRenderingDefault(false);

finalDesign::MyForm form;
Application::Run(%form);
```

Figure 14: code to open the windows form in main thread

8.6.2. Connect to Database

The connection to the database is created to allow the windows form user interface to communicate with the database. The database is also required to show the authentication and display stored data. The connect to the database is created by providing the host name, port number, username and password.

```
String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
MySqlConnection^ conDataBase = gcnew MySqlConnection(constring);
```

Figure 15: connection to database

8.6.3. Register a Student and Register a Lecturer

The registering a student and lecturer works using the same logic. The connected to the database needs to be created when the student clicks on a register button on the home screen. This same method is used for a lecturer to register him/herself. Once the connected is established then the details entered by the student and lecturer will be stored into the respective fields in the database.

```
String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
MySqlConnection^ conDataBase = gcnew MySqlConnection(constring);
MySqlCommand^ cmdDataBase = gcnew MySqlCommand("insert into design3.student (idstudent,name,surname,email,date_of_birth,cell_num,password,module1,mo
```

Figure 16: save into database

8.6.4. Login

During the registration phase of the process the lecturer obtains a userID and a he/she creates a password. This userID and password is used when login onto the system.

```
String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
MySqlConnection^ conDataBase = gcnew MySqlConnection(constring);
MySqlCommand^ cmdDataBase = gcnew MySqlCommand("select* from design3.lecturer where idlecturer = '" + this->user_name->Text + "' and password = '" +
MySqlDataReader^myReader;
try{
    conDataBase->Open();
    myReader = cmdDataBase->ExecuteReader();
    int count = 0;
    while (myReader->Read())
    {
        count = count + 1;
        }
    if (count == 1){
        MessageBox::Show("password and username correct");
        InitPortKeshav();
        //readK();
        //authentication();
        //MessageBox::Show("work");
    }
    else if (count>1)
        MessageBox::Show("password and username dup");
    else
        MessageBox::Show("password and username incorrect");
```

The connection to the database is established and the userID and password stored in the database is checked against the userID and password entered by the lecturer. Thereafter a message box will pop up indicating the status of the password. If the status of the password is correct then the lecturer will be allowed the right to pick from a series of events. One event is starting the controlled access and the other could view the attendance of the students.

8.6.5. Serial Port Communication (UART)

The serial port communication was used to communicate with the sub-systems. The specific ports needed to be initialised and there needed to be a receive and transmit code to receive and transmit data between the sub-systems. When the port is initialised there progress bar will become green to indicate this.

```
const char* port = "\\\\.\\COM6";
SerialPort picMicro(port);
if (picMicro.isConnected()) {
    this->progressBar1->Value = 100;
    }
else {
    MessageBox::Show("error");
}
```

Figure 17: initialise the port

```
const char* port = "\\\\.\\COM6";

SerialPort picMicro1(port);
 while (picMicro1.isConnected()) {

     picMicro1.readSerialPort(output, MAX_DATA_LENGTH);
```

Figure 18 :read data from sub-systems

8.6.6. Controlled access

The controlled access is two-way authentication as explained earlier. There will be a message sent via the UART module to the computer and the RFID sub-system. The Reader of the sub-system transmits the userID linked to the card. This userID was entered into the database when the student was registering. The connection to the database needs to be established as shown above. The received

message with the ID is compared with the module register for students who made Duly performance requirements.

```
MySqlCommand^ cmdDataBase = gcnew MySqlCommand("select* from design3.lecturer where idlecturer = '" + output[11] + "';", conDataBase);
```

Figure 19: statement to compare the data

8.6.7. Automated register

When the student passes the two-way authentication then the system marks the register for that specific module.

```
MySqlCommand^ cmdDataBase = gcnew MySqlCommand("UPDATE `design3`.`enel4ee` SET `attendence`='"+output[11]+"' WHERE `student_register`='
```

Figure 20:mark the register

8.6.8. View attendance

The lecturer has the right to view the attendance of the student. The lecturer has to pick the module which he/she wants to view and the connection to the database is established as explained earlier. The attendance roaster is displayed in the list box of the user interface.

```
while (myReader->Read())
{
    String^ Modules;
    Modules = myReader->GetString("attendence");
    String^ Module;
    Module = myReader->GetString("student_register");
    this->listBox1->Items->Add(Module +"                "+ Modules);
```

Figure 21: read data register from database

8.7. Testing

The home page of the user interface allowing the following actions

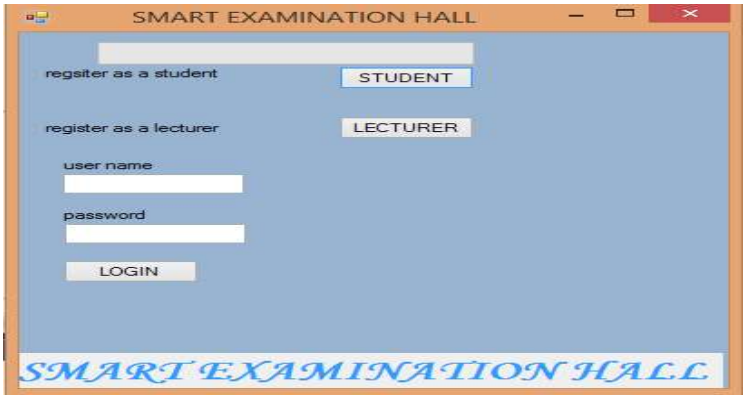


Figure 22: Home Page

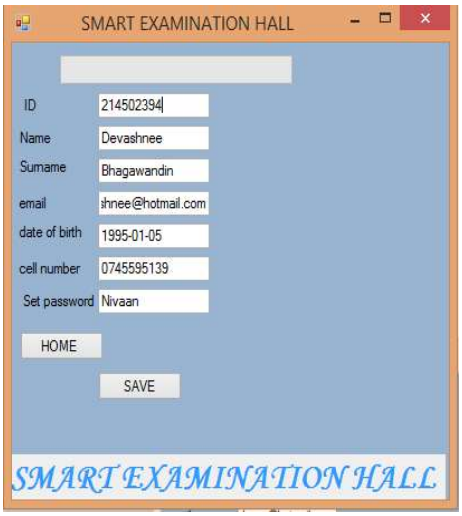


Figure 24: Register as a Lecturer page

idlecturer	name	surname	modules	cell_number	email	password
214502393	Nivaan	Krishundutt	krishunduttn@gmail.com	1996-04-09	0743391690	password
214502394	Devashnee	Bhagawandin	devashnee@hotmail.com	1995-01-05	0745595139	Nivaan
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 23: Data entered in database

Figure 23: Data entered in database and Figure 24: Register as a Lecturer page varied that the data entered in the register lecturer page is the actual data stored in the database

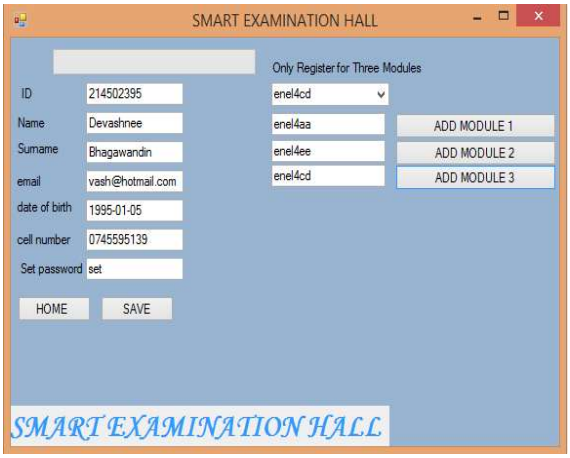


Figure 26: Student Register Page

```
1 • SELECT * FROM design3.student;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

idstudent	name	surname	email	date_of_birth	cell_num	password	module1	module2	module3
214502395	Devashnee	Bhagawandin	vash@hotmail...	1995-01-05	0745595139	set	enel4aa	enel4ee	enel4cd
50	Devashne	Bhagawandin	devashnee@...	1995-01-05	0745595139	password	enel4cd	enel4df	enel4ee
51	Sashin	Ramdhani	sashinramdha...	1995-10-03	0842798116	1234	enel4aa	enel4co	enel4ee

Figure 25:details stored in data base

Figure 26: Student Register Page and Figure 25:details stored in data base shows that the data entered by the student is stored into the database correctly.

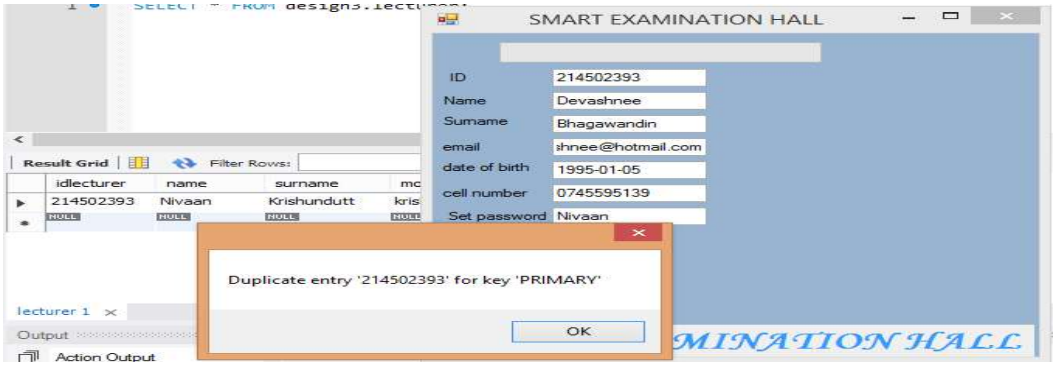


Figure 27: Duplicate Primary key

Figure 27: Duplicate Primary key shows that a lecturer tried to register but had the same ID as an already registered lecturer and this is not allowed.

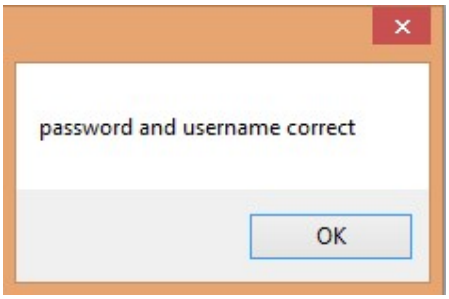


Figure 28: Login pop up box

Figure 28: Login pop up box shows the pop-up message box which is active to display the status of the login.

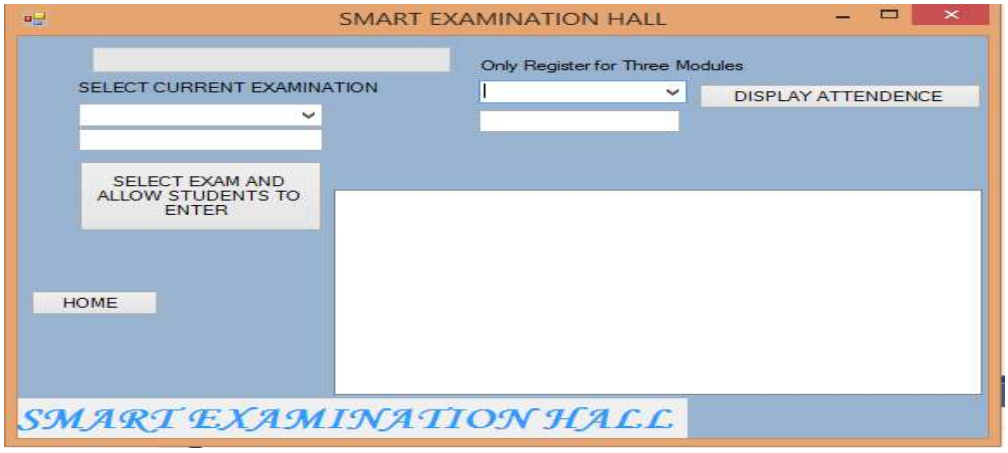


Figure 29: lecturer operation page if login correct

Figure 29: lecturer operation page if login correct the lecturer must pick a module and the controlled access will start.

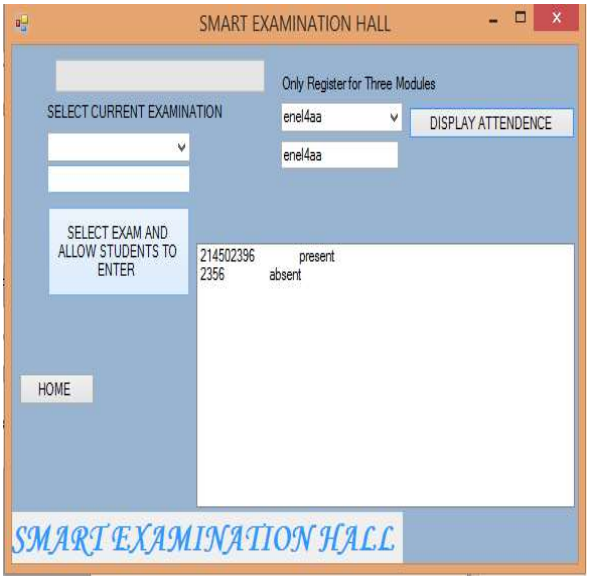


Figure 31: display register

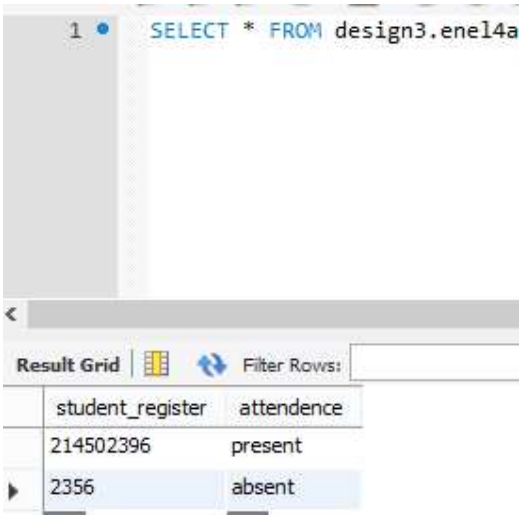


Figure 32: data in database for register

9. RFID Authentication (Rishay Ramcharan 214504863)

9.1. System overview

The RFID authentication will be used for controlled access to the exam venue. The subsystem will only allow students who are eligible to write the exam to proceed. Each student will be given a RFID student card which will contain their unique student number. A RFID reader will be installed at the entrance of the exam venue. When a student card is brought within range of the RFID reader, the subsystem will be able to detect its student number. The subsystem will then send the student to the database, where the *duly performed* status of that student will checked. If the status is satisfactory, the student will be allowed to proceed to biometric authentication.

A manual override will be implemented as well. The manual override will open the emergency exit. The override will active upon receiving a correct password from the user. A keypad and LCD screen will be used to implement the manual override. In case of a fire the manual override will be bypassed and the emergency exit will open automatically.

9.2. Specifications

9.2.1. Functional requirements

- Read RFID cards to obtain students' unique student number
- Send the student number to the database
- Allow a student to proceed to biometric authentication if RFID authentication is successful
- Unlock the emergency exit when the fire alarm is triggered
- Unlock the emergency exit when the manual override is triggered via password

9.2.2. Non-functional requirements

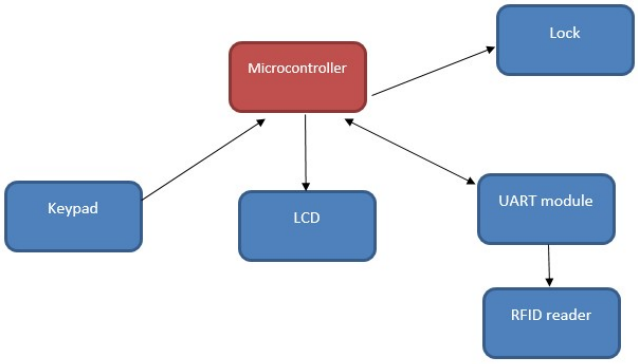
- User – friendly
- Efficiency
- Robustness
- Economically feasible to develop and maintain

- Well packed yet safe and easy/safe to open and demonstrate

9.2.3. Constraints

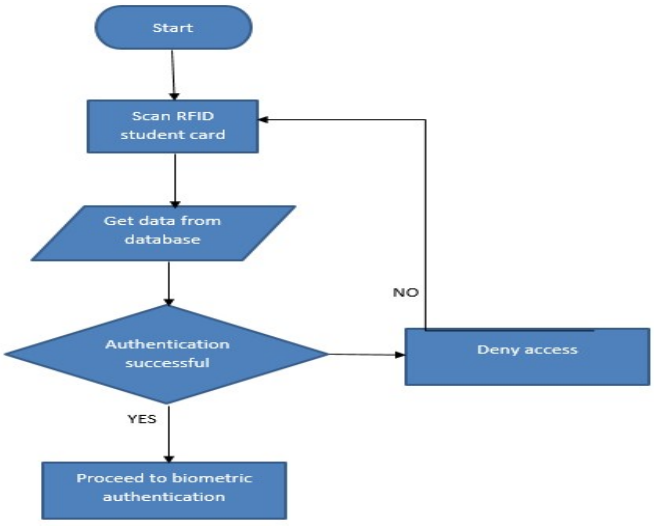
- Time
- Budget
- Availability of components

9.3. Block diagram of subsystem



9.4. Solution approach

9.4.1. Flow diagram of RFID authentication



9.4.2. RFID implementation

The smart exam hall will require a RFID system to be used as part of a two way authentication system to grant student access into the exam venue. The RFID reader will be installed at the entrance of the exam venue.

Since the RFID reader is being used for controlled access, a reader with a low range will be ideal for this application. When a student brings their RFID student card within range of the RFID reader, a unique student number will be detected and used to search the database for the student's information. The database will then send that students duly performed status back to the RFID Authentication subsystem.

9.4.3. RFID Reader choice

The EM-18 RFID reader will be implemented for this project. The EM-18 has a reading range of 10cm – 15cm and an operating frequency of 125 kHz. RFID readers are designed to read RFID tags with the same frequency, therefore the student cards used will have to be 125 kHz RFID cards.

9.4.4. The EM – 18 RFID reader and Passive RFID cards

The reader module comes with an on-chip antenna and can be powered using 5V power supply. It generates and radiates Radio Frequency Carrier Signals of 125 kHz. Through its coils.

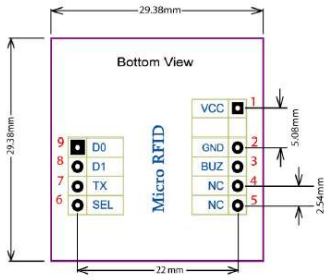


Figure 9.1 : EM - 18 diagram [3]

The following table explains the function of each pin on the reader.

Pin number	Name	Function
1	VCC	5V
2	GND	Ground
3	BUZ	Becomes low on valid card detection
4	NC	No connection
5	NC	No connection
6	SEL	High selects TTL serial o/p, Low selects WIEGAND 26 o/p
7	TX	UART TX, when TTL serial is selected
8	D1	WIEGAND Data 1
9	D0	WIEGAND Data 0

Table 7 : Functions of the pins of the EM-18 [2]

The student cards used will be Passive RFID cards of frequency 125 kHz. Passive RFID tags do not have a power supply. Passive RFID tags are usually made up of a coiled antenna alongside with an IC.

The Passive RFID tags get energized when it is brought into the field that is generated by the RFID reader through its coils. [2]

The figure below shows the internal structure and operation of the RFID System, where the Transceiver is the RFID Reader, and the Transponder is the RFID Tag.

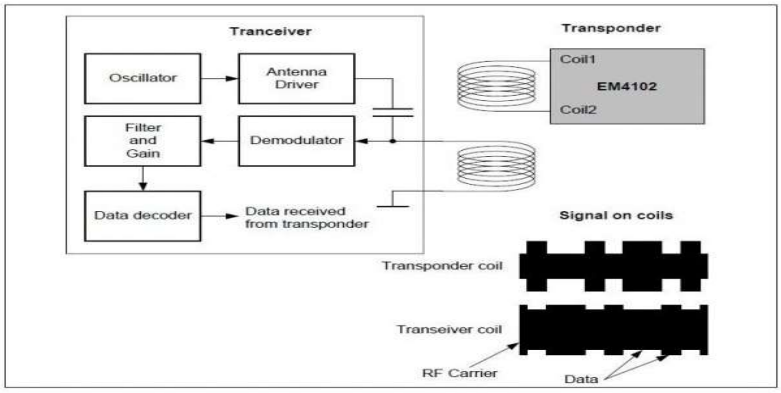


Figure 9.2 : RFID system principle [2]

By changing the modulation current through the coils, the tag will send back the information contained in the factory programmed memory array.

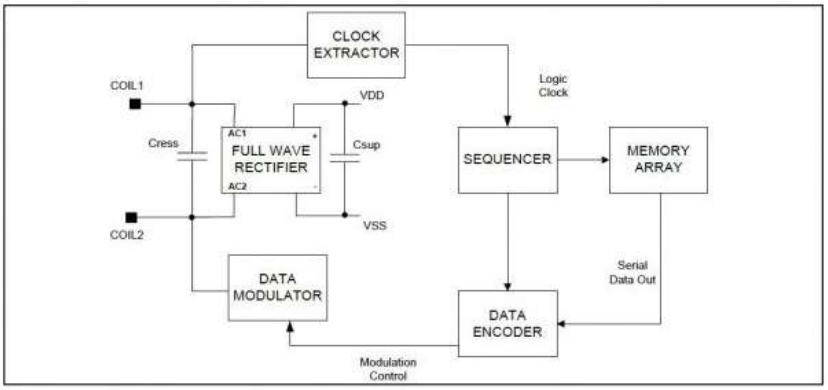


Figure 9.3 : Passive RFID tag block diagram [2]

9.4.5. PIC16F690 microcontroller

A microcontroller is required to interface our system. We have opted for the PIC16F690 microcontroller, due to it fulfilling all of our computing requirements, as well as being relatively cheap.

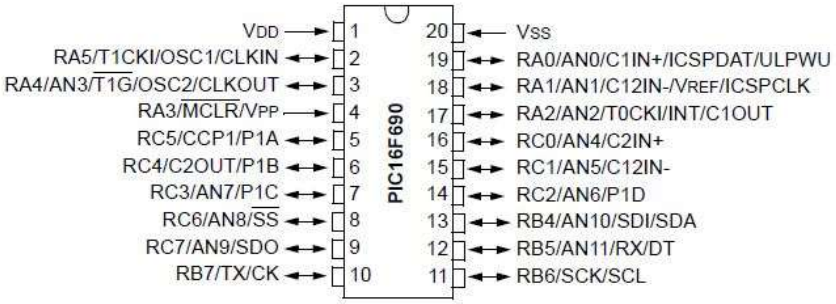
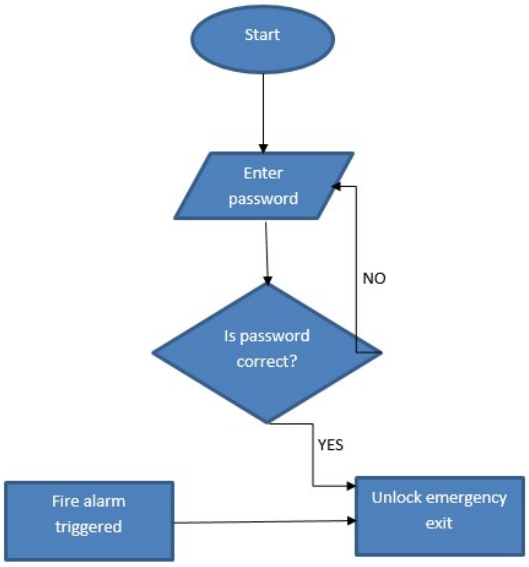


Figure 4 : Pin layout of microcontroller

9.4.6. Emergency exit flow diagram



9.4.7. 4x3 Matrix Keypad

A 4x3 matrix keypad will be used to enter a password to active the manual override that will unlock the emergency exit. A 4x3 matrix keypad has 4 rows and 3 columns of buttons.

Each column of the matrix keypad will be connected to an output pin of the PIC Microcontroller, whilst each row of the matrix keypad will be connected to an input pin of the PIC Microcontroller.
The polling method will be used to scan for button presses. The scanning takes place by keeping a specific row low at a time and read the status of the column pins as input at that instant. This chain goes on through all the rows and by this way input is read by Microcontroller.

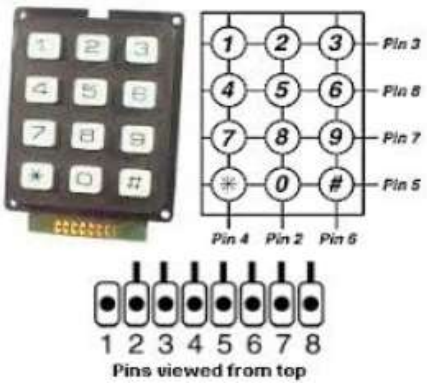


Figure 9.5 : 4x3 matrix with pin layout

9.4.8. 16x2 LCD screen

A 16x2 LCD screen will be used to display the password. A 16x2 LCD has 2 rows that can contain 16 characters each.

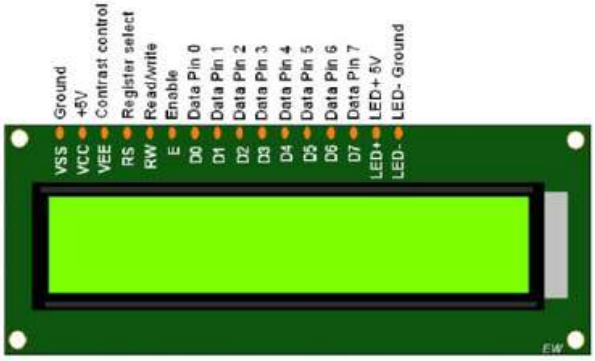


Figure 9.6 : 16x2 LCD screen [4]

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5
13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

Table 8 : Functions of the pins of the LCD [4]

The LCD can work in either 4 bit mode or 8 bit mode. In 4 bit data is sent nibble by nibble. The upper nibble is sent first then the lower nibble. In 8 bit mode data is sent directly at once. 8 bit mode is faster, however it will not implemented because it would require all 8 data pins to be connected to the microcontroller and there only 20 I/O pins on the microcontroller. [4]

9.5. Design process

9.5.1. Simulation

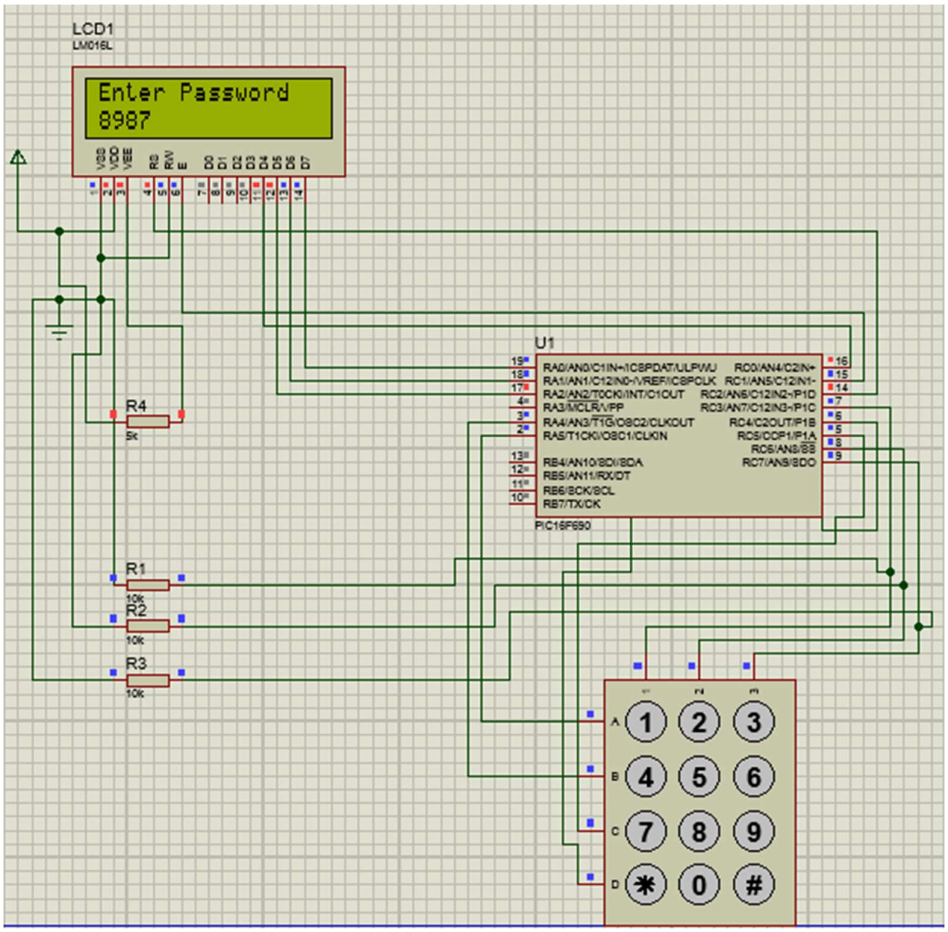


Figure 7 : Circuit simulation

Computer-Aided Design (CAD) circuit simulation tools are very expensive; hence it was not possible to simulate the entire circuit using software. There are free CAD circuit simulations available, however the EM-18 RFID reader was not supported on those platforms, therefore only part of the subsystem could be simulated on software. However, source code that will be used to program the microcontroller is included in the Appendix.

9.6. Testing and results

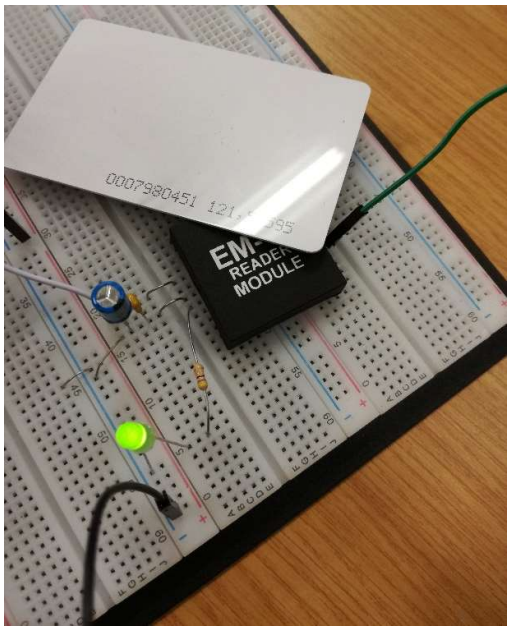


Figure 8 : Testing the RFID reader

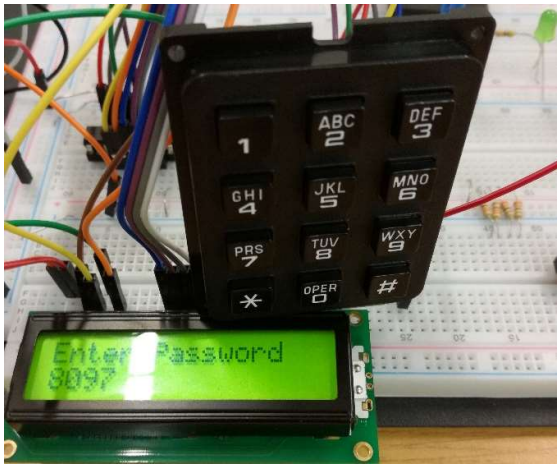


Figure 9 : Testing the manual override

9.7. Application of knowledge

- Experience of the PIC microcontroller, from Computer Engineering Design 2 was applied to this project. It became easier to use a familiar microcontroller to run our system.
- Knowledge gained from Digital Systems of using Serial Communication(UART) was applied in the integration.

9.8. Project costs

9.8.1. Component cost

- EM-18 RFID Reader – R250

- 5*RFID Tags – R100
- PIC16F690 Microcontroller – R70
- PL2303 UART Module – R30
- Keypad – R60
- LCD – R80
- 2*LEDs – R2
- Buzzer – R6
- 3*10KΩ Resistors – R1
- 470Ω Resistor – R0.50
- 2.2KΩ Resistor – R0.50
- 10KΩ Potentiometer – R4.50
- 100uF Electrolytic Capacitor – R1
- 2*0.1uF Ceramic Capacitors – R0.50
- 2*22pF Ceramic Capacitors – R1
- BC557 PNP BJT – R1

Total component cost = R608

9.8.2. Additional cost

- Labour = R28810 (category D ECSA hourly rates were applied)
- Transport = R250 (to pick up components)
- Pickit 3 programmer = R400
- Laptop = R7000

The cost of the Pickit 3 programmer and the laptop will not be included in additional cost since these are investments.

Additional cost = R36460

Total cost of subsystem = R37068

9.8.3. Budget analysis

The total of designing and implementing the project amounts to R37088. This hefty amount is due to the high hourly rate charged. To greatly reduce the cost of labour and hence, the total cost of the project, a more reasonable amount of time should be allocated in implementing the project.

10. Facial Recognition and Video Surveillance (Keshav Jeewanlall 213508238)

10.1. Subsystem Overview

A more secure way of controlling the door access is by using biometrics together with the RFID cards. It is more secure because you are identifying a person based on their physical characteristics rather than the card that the person is carrying [6]. This will prevent a person from entering with a stolen credential.

Facial recognition is one of the most flexible method of biometric identification [7] therefore it has been considered as a possible solution for a biometric identification. It works by systematically analyzing facial features that are common to all faces, such as; the distance between the eyes, width of the nose, position of cheekbones, jaw line, chin etc. These numerical value that is obtained from this analysis is then combined and used to uniquely identify the person [7].

This system will be implemented using a desktop computer, camera and a microcontroller. A camera will be at the entrance of the venue, and will detect a person's face once their RFID card has been verified. Only when the facial recognition and RFID cards have both been verified successfully, will the person be allowed to enter the venue.

The video surveillance will be implemented using a computer and a camera. This system will ensure the safety of the venue and monitor students during exams, cheating is prevented in this way, and any attempt to cheat will result in the student being identified and caught. The camera will be activated once a person enters the venue through controlled access and will be continuously recording whilst the exam or lecture session is taking place. In the event of forced entry into the room such as a break in, the motion detectors will activate the camera.

10.1.1. Functional requirements

- System must be able to detect faces of people entering the venue.
- System must be able to access the database in order to match faces detected.
- System must allow entrance to venue only when both the facial and RFID authentication is verified.
- System must prevent entrance to venue if facial authentication is not verified.
- System must video surveillance once a person enters through controlled access or when motion is detected in the venue.

10.1.2. Non-functional requirements

- System must be reliable.
- System must be efficient.
- System must have data integrity.
- System must be easy to maintain.

10.1.3. Constraints

- Budget.
- Time.
- Availability of components.

10.2. Solution Approach [ELO1]

10.2.1. Flow diagram of system

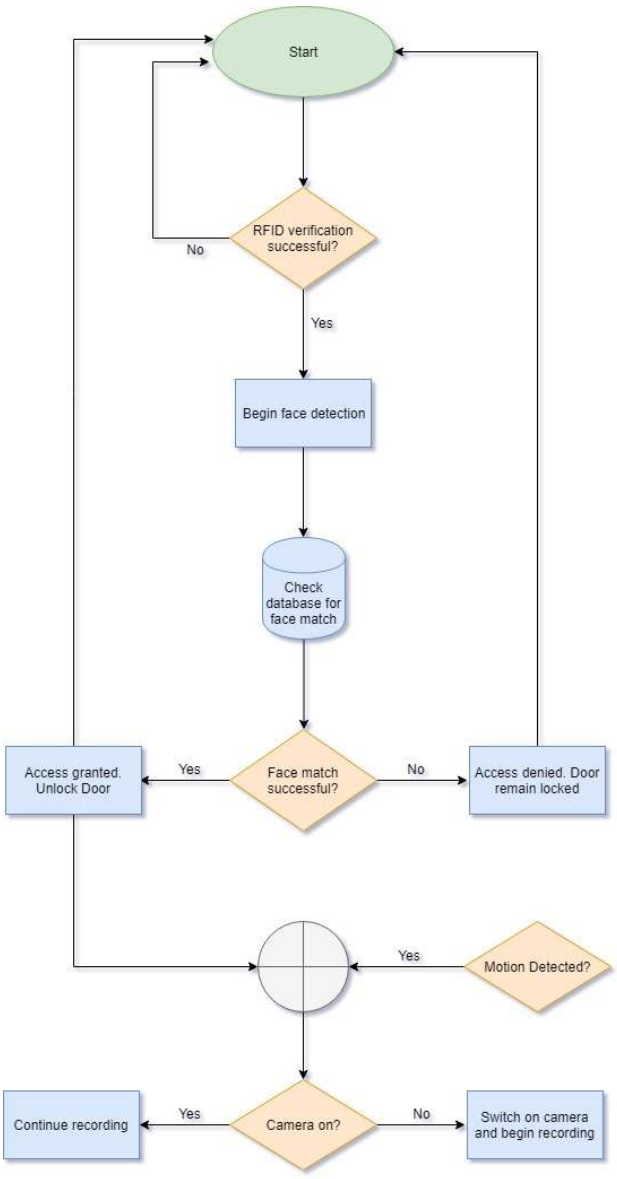


Figure 10: Flow diagram of biometric authentication and video surveillance

10.2.2. Process

- The images of new student faces will need to be added to the database.
- Once RFID has been verified, the facial recognition software will begin to detect the face of the student.
- The software will check the database of images to find a corresponding match.

- If a match is found and the face detected corresponds with the ID number on the RFID card, access will be granted.
- Once access is granted, the system will then check if the camera has been turned on, if not then it will turn on the camera and begin recording.
- In the event of a forced entry such as a break in, the motion detectors will activate the camera.
- Once the exam session is over and no motion is being detected, the camera will stop recording and turn off.
- Recorded footage will be stored on the hard drive of the computer.

10.3. Subsystem Design [ELO2, ELO3]

There are two parts in this implementation step. The first is the implementation of the facial recognition system by using software. And the second is the implementation of PIC16F877A program for the door access system.

The smart exam hall will use a facial recognition system as the second part of a two-way authentication system to grant student access into the exam venue. The camera will be installed at the entrance of the exam venue.

When the relevant data from the RFID is sent over to PC and a match occurs, a user image corresponding to the received data is extracted from the database. The database will contain images of the respective user. A snapshot of the incoming users face is taken by the camera and comparison between the extracted image and snapshot is carried out. If the images match, then the user is allowed to access the venue.

Once the first person is granted access into the venue, the video camera will start recording footage until the end of the session. Motion detectors in the venue will also activate the camera if motion is detected in the venue.

In this system, face recognition and video surveillance is implemented by using C++ programming language using the Visual Studio 2017 IDE installed on PC, and a microcontroller. USB to RS232 converter is used as the interface between the PC, camera and PIC16F877A microcontroller.

10.3.1. Hardware Design

10.3.1.1. Implementation Choices

The hardware components chosen for this subsystem are

- PIC16F690 microcontroller
- USB to RS232 Converter

The reasons for these choices are mentioned below.

10.3.1.2. PIC16F690 Microcontroller

The PIC16F690 has a built in universal synchronous asynchronous receiver transmitter (USART) hardware that allows direct communication with personal computer. The USART module has two modes of operations: synchronous (requires a synchronized clock between the transmitter and receiver) and asynchronous (no synchronization clock required) [8].

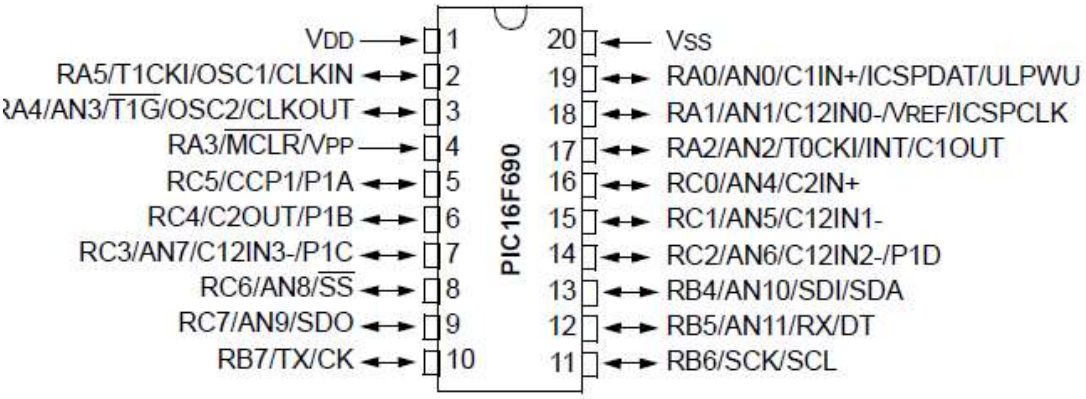


Figure 11: PIC16F690 Pin Out Diagram [9]

10.3.1.3. USB to RS232 Converter

Interfacing the camera directly with PIC Microcontroller for facial recognition and video surveillance is not possible because the size of internal ROM and RAM in PIC16F690 is insufficient. Therefore, the best way is to interface the PIC16F690with the PC via UART communication [10]. This is done using a USB to RS232 Converter. Figure 28 shows USB to RS232 converter.

MAX232 is a serial RS232 to TTL/CMOS level converter which converts +/-10 V serial RS232 signal to TTL/CMOS level 0 to 5V. This connection is important to allow the communication between PC and PIC microcontroller. Since most PCs nowadays do not have serial communication ports therefore, USB to DB9 converter is used for this design [8]. Driver software that converts the USB connection into a Virtual Communications Port is to be installed on the PC, this makes the USB connection look like a serial port on the PC.

Figure 12: USB to RS232 Converter [11]

10.3.2. Software Design

10.3.2.1. Implementation Choices

The C++ programming language and OpenCV (Open Computer Vision) libraries were chosen for the development of the software in this subsystem. The IDE used to write the code is Microsoft Visual Studio 2017. The reasons for these choices are as follows:

- Visual Studio is highly featured for C++ Development.
- OpenCV has built in classifier classes to detect faces in a video stream.
- The classifiers in OpenCV have facial features trained in them, hence no training is required.

10.3.2.1. Use Case Description

The software is programmed to run facial recognition and video surveillance. The facial recognition section is programmed take an image of the student and detect the face present in the image.

The detected face is run through facial recognition software. The facial recognition software extracts images from a database and does its comparison. When facial recognition software successfully recognises the face in the image captured, the system should give that student access to the venue.

The video surveillance section is programmed to run a live video feed and save this recorded footage on the PC.

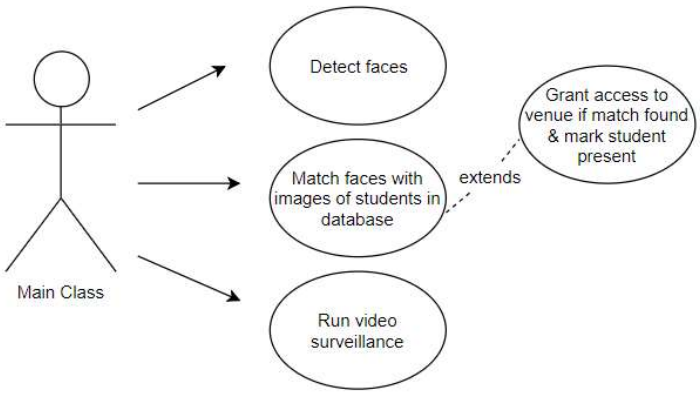


Figure 13: Use Case Diagram

10.3.2.2. Classes

With analysis from the Use Cases, the Facial Recognition and Video Surveillance subsystem can be implemented using the following classes:

- Main Class – Calls all other methods and runs the software
- SerialPort Class – Creates serial communication between the microcontroller and PC
- FacialRecognition Class – Detects and recognises faces
- VideoRecorder Class – Runs live video surveillance

Table 9: Main Class

Main Class		
Description	Calls all other methods and runs the software	
Attributes	output[]: char incomingData[]: char port: const char* microOutput1: int microOutput2: int	Stores data to be sent via serial port Stores data received via serial port Stores serial port name Stores comparison value of the microcontroller output Stores comparison value of the microcontroller output
Associations	main(): int	
Frequency	The class is to be called whenever the software is ran.	

Table 10: SerialPort Class

SerialPort Class		
Description	Creates serial communication between the microcontroller and PC and has functions for sending and receiving data	
Attributes	Handler: HANDLE Connected: bool Status: COMSTAT Errors: DWORD	Pointer Stores value to check if serial port is connected Stores com port status Stores error value
Associations	readSerialPort(char, unsigned int): int writeSerialPort(char, unsigned int): bool isConnected(): bool	
Frequency	The class is to be called whenever serial communication is required	

Table 11: FacialRecognition Class

FacialRecognition Class		
Description	Detects and recognises student's face	
Attributes	capturedImage: Mat faces: vector<Mat> ids: vector<int> im_width: int im_height: int original: Mat greyImage: Mat facePositions: vector<Rect_<int>> face_i: Rect getFace: Mat face_resized: Mat foundID: int	Stores the captured image Stores the images used for comparisons Stores the IDs of the images Stores width of the images used for comparisons Stores height of the images used for comparisons Stores a copy of the original image Stores a greyscale version of the image Position of the ith face Converts to greyscale Resizes the detected face Gets prediction from face recognizer
Associations	read_csv(const string, vector<Mat>, vector<int>): void detectFace(): int	
Frequency	The class is to be called whenever facial recognition is required	

Table 12: VideoRecorder Class

VideoRecorder Class		
Description	Records video footage	
Attributes	frame_width: int frame_height: int frames_per_second: int fcc: int frame: Mat	Stores the video frame width Stores the video frame height Stores the video frames per second Stores the video codec Stores the video frame
Associations	videoRecord(): int	
Frequency	The class is to be called whenever video surveillance is required	

10.3.2.3. Libraries

OpenCV 2.4.13.6 libraries and Serial Port libraries were used in the development of this software.

The OpenCV source files were downloaded from the official website [12]. This is a self-extracting file, which the user will have to extract to the C drive of the computer. The library files needed to be linked to Visual Studio 2017, following the necessary procedures [13], [14], as this was the IDE used to design and develop this software.

The Serial Port library used in the project is an open source library and the source files can be found on GitHub [15]

10.4. Subsystem Simulation

10.4.1. Hardware Simulation

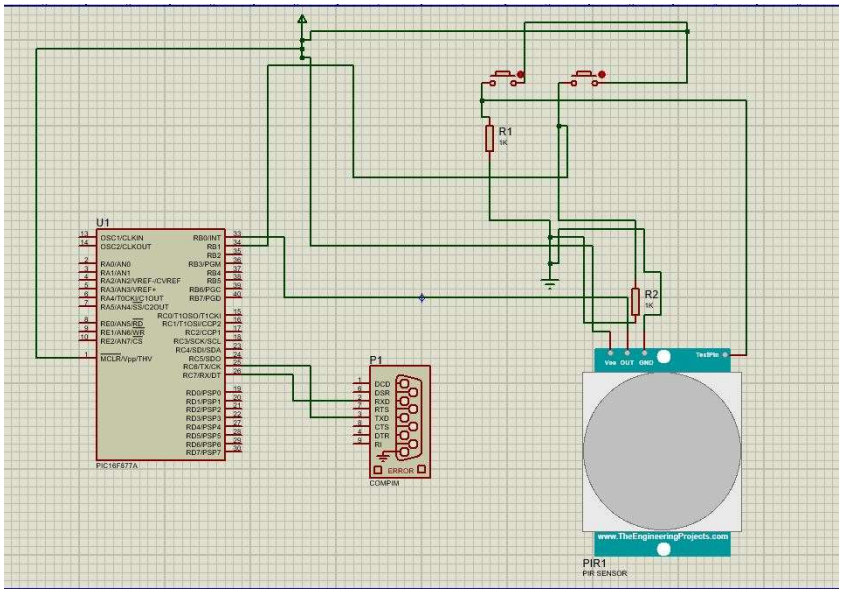


Figure 14: Subsystem Circuit Schematic

For the simulation circuit shown in Figure 30, a PIC16F877A microcontroller, 2 push buttons, a virtual communication port and a PIR motion sensor is used. The push button connected to pin RB1 of the microcontroller represents the signal that will be sent once the RFID authentication has been passed. The push button connected to the PIR motion sensor passes a high to the PIR test pin to simulate motion detection, the signal from the PIR sensor is then sent to the microcontroller to activate the video surveillance. The RX and TX pins of the microcontroller is connected to the RX and TX pins of the virtual communication port this simulates a connection between the PC and microcontroller.

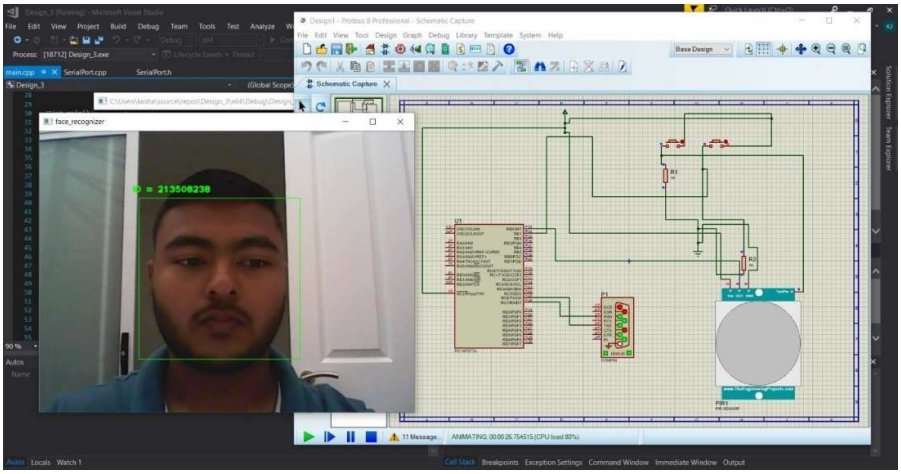


Figure 15: Simulation Showing Facial Recognition

In Figure 31, we can see that when a high signal is passed to pin RB1, the facial recognition software is started and it begins to identify the face. The information displayed on the screen is the student identity number of the face detected.

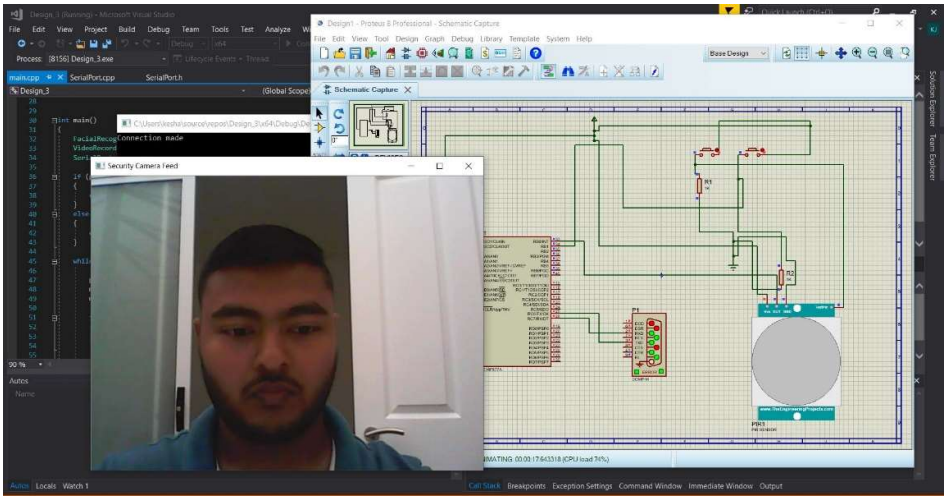


Figure 16: Simulation Showing Live Video Surveillance

In Figure 32, we can see the live video feed being displayed when video surveillance is activated. This is activated when the PIR sensor detects motion.

10.4.2. Software Simulation

10.4.2.1. Microcontroller Code

The code for simulating the microcontroller was written in the MikroC IDE. So as per the code in appendix D, when character “a” is sent to the PC via the UART, the code for the facial recognition is run. When character “b” is sent to the PC via UART, the code for video surveillance is run.

10.4.2.2. Calibrating the Camera

When capturing an image using OpenCV, a camera ID number is sent as a parameter in order for that camera to be used. This works well if multiple cameras are on a computer system. This software was deigned to use a two cameras therefore by sending a 0 in place of the first camera’s ID and 1 in place of the second camera’s ID.

```
if (cap.isOpened() == false)
{
	cout <<"ERROR: Could not open camera."<< endl;
	return -1;
}
```

The above code segment opens the default webcam connected to your computer. If it fails to open the camera, the program will exit.

10.4.2.3. Facial Recognition Software

In OpenCV, Haar-Cascades are used to detect faces in an image. In this software, the “haarcascade_frontalface_default.xml” file was used to detect faces in an image.

```
vector<Rect_<int>> facePositions;
haar_cascade.detectMultiScale(greyImage, facePositions)
```

The code above shows the function that is used to detect faces in an image. The image is passed as a parameter in the variable *greyImage*. The function detects the faces in the image using the haar-cascade defined earlier and then stores the positions of the detected faces in a vector of integers called *facePositions.* Vectors are used instead of arrays due to the fact that the number of faces detected in an image cannot be previously determined.

Open source computer version (OpenCV) currently has the following available algorithms to perform facial recognition; Eigenfaces, Fisherfaces and Local Binary Patterns Histograms (LBPH). The algorithm chosen for this project was Fisherfaces.

The fisherfaces method requires multiple images of a person as these images are compared to a face from the image captured and a prediction is made. A larger sample of images results in greater accuracy of the prediction. A CSV file is created in order to group together all the images of one person. Each image of a person is assigned the unique ID of that person.

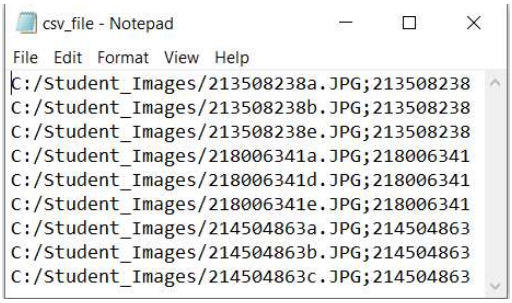


Figure 17: Example of CSV File

Figure 32 shows an example of the CSV file that is created. Each line contains an image path then followed by a delimiter, in this case a semicolon is used, which is then followed by the ID of the person in that image. When reading the CSV file, for each line that is read, the image is stored in a vector of images and the ID of that image is stored in a vector of integers. Theses vectors are going to be used to train the Face Recognizer. In order for the fisherfaces prediction to be accurate, the images used for comparisons and the captured image both have to be in greyscale. OpenCV contains a function *cvtColor(Mat, Mat, int)* which can convert an image to greyscale.

```
Ptr<FaceRecognizer> model = createFisherFaceRecognizer();
model->train(faces, ids);
```

In the code above, the first line creates a Fisherface recognizer that is used to perform Face Recognition. The second line of code utilizes a function called train which accepts two parameters. The first parameter is a vector of images. The second parameter is a vector of integers. These vectors were both created when the CSV file was read therefore these vectors contain the images that are used for comparisons and their respective IDs. By sending these parameters, the Face Recogniser is being 'trained' to recognise the faces that were in the images of the vector.

```
Rect face_i = facePositions[i];
Mat getFace = greyImage(face_i);
Mat face_resized;
cv::resize(getFace, face_resized, cv::Size(im_width, im_height), 1.0, 1.0,
INTER_CUBIC);
int foundID = model->predict(face_resized);
```

In the code above, since the positions of the detected faces in the captured image are known, a face is extracted from the image, turned to greyscale and resized to match the size of the images used for comparisons. This is then stored in the variable face_resized. Using the trained Fisherface Recognizer, the image face_resized is then sent as a parameter to the predict function. This function compares face_resized with the images that it was trained with. If a match is found, it returns the ID of the matching image thus completing the face recognition process.

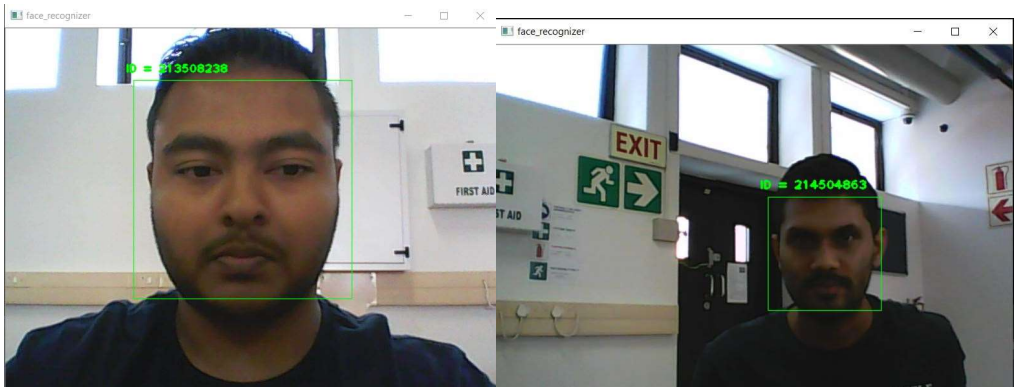


Figure 18: Simulation results of Facial Recognition software

In figure 34, it can be seen that the software does indeed identify the faces of students.

10.4.2.4. Video Surveillance Software

```
frame_width = static_cast<int>(cap.get(CV_CAP_PROP_FRAME_WIDTH));
frame_height = static_cast<int>(cap.get(CV_CAP_PROP_FRAME_HEIGHT));

Size frame_size(frame_width, frame_height);
frames_per_second = 10;
fcc = CV_FOURCC('M', 'J', 'P', 'G');


VideoWriter writeVideo("D:/MyVideo.avi", fcc,
       frames_per_second, frame_size, true);

if (writeVideo.isOpened() == false)
{
       cout <<"Cannot save the video to a file"<< endl;
       return -1;
}
```

This code segment obtains the width and height of the video frame of the camera. Using this information, the VideoWriter object is created and initialized.

VideoWriter(const String& filename, int fourcc, double fps, Size frameSize, bool isColor = true)

This is one of the available overloaded constructors of the VideoWriter object. It creates and initializes the VideoWriter object in order to write video frames to a given file. The parameters passed are as follows:

filename - Name of the file to write video frames.

fourcc - 4-character code of the codec which is used to compress the video.

- VideoWriter::fourcc('M', 'J', 'P', 'G') for Motion JPEG

- VideoWriter::fourcc('M', 'P', '4', '2') for MPEG-4 variation of Microsoft

fps - Frames per second of the written video stream.

frameSize - Size of the video frames written to this video stream

isColor - Always pass true to this argument.

```
while (true)
	{
	frame;
	bool isSuccess = cap.read(frame);
		if (isSuccess == false)
		{
			cout <<"Video camera is disconnected"<< endl;
			cin.get();			break;
		}

	writeVideo.write(frame);

	imshow(window_name, frame);

	if (waitKey(10) == 27)
	{
	cout <<"Esc key is pressed by the user. Stopping the video"<<
endl;
	break;
	}
}
```

In each iteration of the above while loop, the program performs following tasks.

- Read a frame from the camera.
- Write the frame to the file.
- Display the frame in a window.

10.5. Implementation and Testing

This section of the report deals with the testing and integration with the different subsystems. Communication between subsystems was done by wired communication. The main application will contain a database with the ID numbers required by this subsystem in order to relate the pictures with the ID number scanned by the RFID subsystem. Also signals from the PIR sensors used in the safety subsystem will activate the camera for video surveillance in this subsystem.

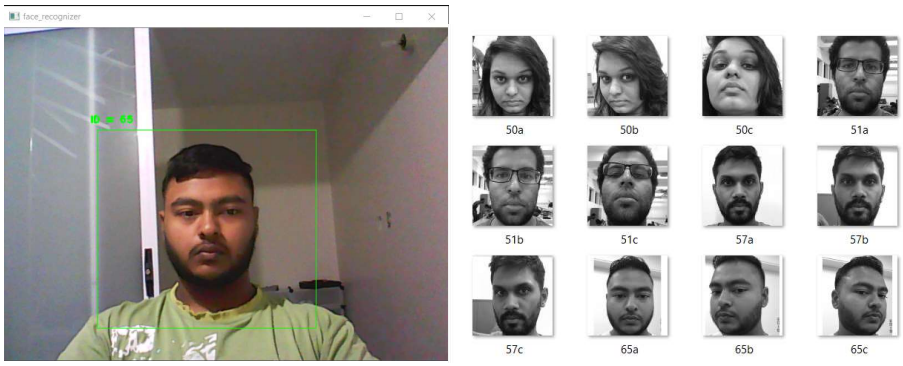


Figure 19: Face detected corresponding with face in the database

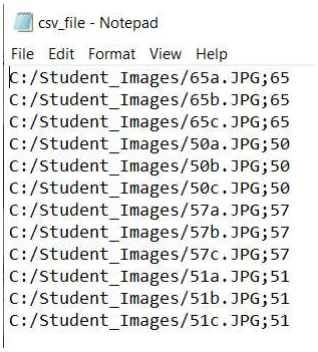


Figure 20: CSV file used

Figure 51 shows that the face detected by the software does correspond to the face stored in the database of students. In the case the student with student number was 65 which was identified.

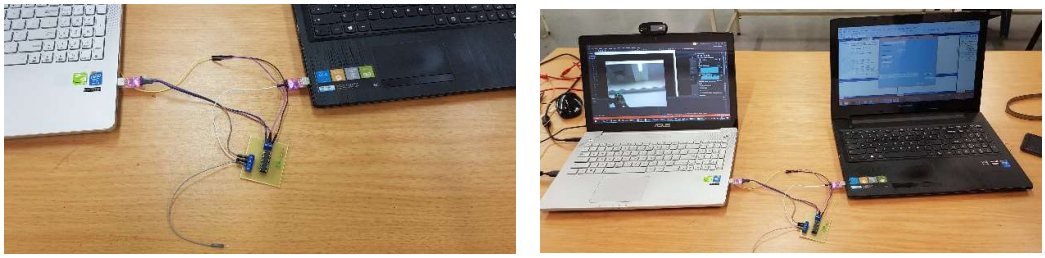


Figure 21: Integration between face recognition and main application

From figure 53 it can be seen that when a student number is sent from the main application, the face recognition camera is activated and the student number identified by the camera is compared to the student number sent by the main application. If these numbers match, then access to the venue is granted.

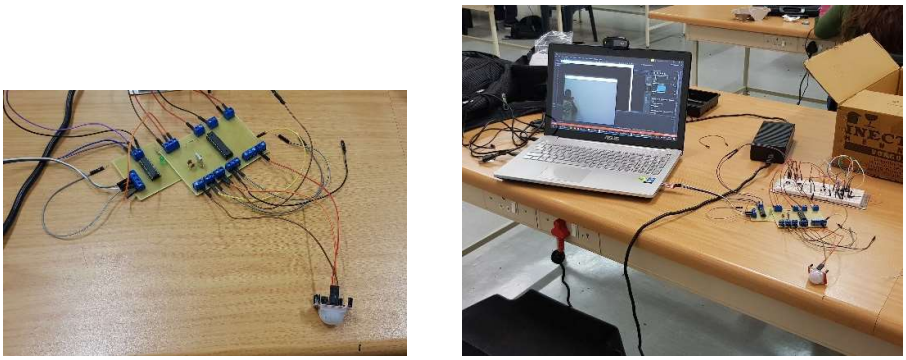


Figure 22: Integration between Video surveillance and motion detection

From figure 54 it can be seen that the video surveillance camera is activated when motion is detected by the PIR sensor. When the PIR sensor detects motion, the microcontroller in the video surveillance subsystem receives a high (5V) which then sends a signal via UART to the computer to trigger the camera.

10.6. Subsystem Feasibility Study

10.6.1. Different Economic Solutions

As stated in section 8.4, this system can also be designed to operate on either a Raspberry Pi or a Personal Computer (PC) with similar pros and cons for each design.

Cost estimation using a Raspberry Pi:

• Raspberry Pi 3 Model B+	R700
• 2x Raspberry Pi Cameras	R960
• External Hard Disk (1TB)	R800
• Monitor	R1 300
• Keyboard and Mouse	R300
TOTAL:	**R4 060**

Cost estimation for Desktop PC:

• Desktop PC	R4 000
• 2x Cameras	R600
• Monitor	R1 300
• Keyboard and Mouse	R300

TOTAL:	**R6 200**

Taking these factors mentioned in section 8.4 into account and the fact that the main application will require the use of a PC, it is decided that this subsystem will be designed for a PC running the Windows 10 operating system.

10.6.2. Component Costs

Cost of Components	
1x PIC16F877A	R85
2x Cameras	R600
1x RS-232 to USB adapter	R47
Total Components Cost	**R732**

10.6.3. Additional Costs

Referring to appendix K, the labour costs per hour for an engineer leading this type of project is R670. Working an average of 2 hours per day, for 24 days this works out to R32 160.

- Labour – R32 160
- Transport – R200
- Pickit 3 Programmer – R400 (not included in total, as this is considered an investment)
- Laptop – R6000 (not included in total, as this is considered an investment)

Applicable Additional Costs Total = R32 360

10.6.4. Total Subsystem Cost

Total Applicable Cost = R6 200 + R732 + R32 360 = R39 292

10.7. Application of Knowledge on Engineering Design [ELO2]

- My experience of using and programming the PIC microcontroller (from Computer Engineering Design 2) was applied to this project's subsystem. It became much easier to use a familiar microcontroller to run this subsystem.
- My knowledge of using Serial Communication (UART) from Digital Systems (ENEL3DS) was instrumental in the integration of the PIC microcontroller with the PC.
- My knowledge of C++ programming from Computer Methods 3 (ENEL3CC) was used to write the software for this subsystem
- My knowledge of using openCV and facial recognition was obtained from Software Engineering 2 (ENEL3SF)

11. Energy Saving, Safety and Isolation system (Sashin Ramdhani – 214513737)

11.1. Introduction

The full system consists of various subsystems that contribute towards energy saving, safety and isolation of students. These include a fire alarm system, a Cell phone detection system, a temperature control system and a lighting control system. The cell phone detection system utilizes radio frequency detection circuitry to determine if a cellular phone is being used. Both the light control and temperature control system operate using sensors to control the switching on of lights and air conditioners respectively. The fire alarm system utilizes a smoke detector sensor to trigger an alarm circuit.

11.2. Specifications:

Functional:

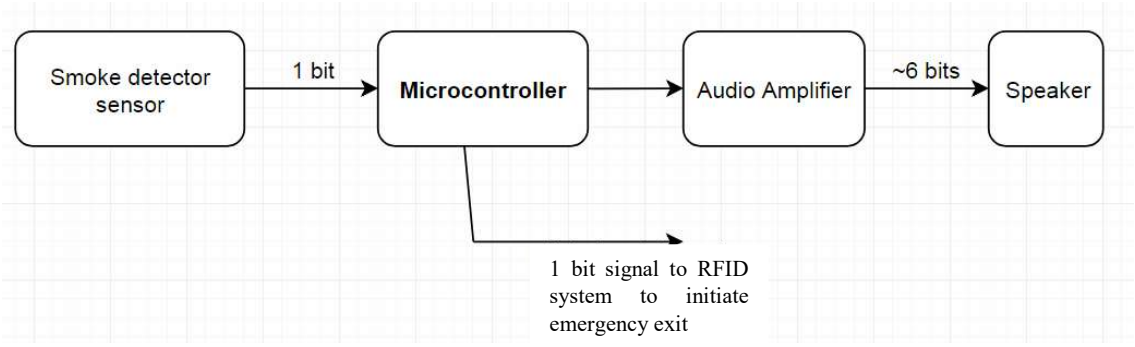
- System must measure and display an accurate reading of the ambient temperature in the exam hall.
- System must allow users to set a threshold temperature.
- If the ambient temperature is greater than the threshold temperature, air conditioners (relays used to simulate) are switched on.
- System must switch on lights in the event of students or teacher being present in the hall.
- In the event of a fire, system must issue an alarm.
- In the event of a fire, system must bypass the RFID authentication system.
- System must detect the usage of internet or wireless networks within the exam hall.

Non-Functional:

- System must be reliable.
- System must be maintainable
- System must offer high performance.

11.3. Fire alarm subsystem

11.3.1. Subsystem Block Diagram



11.3.2. Subsystem overview

The subsystem will utilize a sensor module to detect if smoke is present in the venue. The sensor will be used to trigger an alarm via the microcontroller. The controller utilizes pulse width modulation techniques to generate an alarm tone. A power amplifier is necessary if speaker used has high power characteristics (such as needed in an exam venue), as microcontroller power is far from sufficient.

Power amplifiers are specially designed amplifiers often used in audio applications, as they provide large gain, capability of carrying high current and compensations for non-linearities associated with high power transfer.

11.3.3. Components List

• 1 x ATMEGA328P-PU microcontroller	R57.06
• 1 x MD0203 gas sensor	R67.77
• 1 x 2N7000 MOSFET	R 2.70
• 1 x M40IE060008LF-SM *A* Speaker	R29.24
• 1 x power resistor	
Total:	R156.77

11.3.4. Circuit Design

The subsystem will utilize a sensor module to detect if smoke is present in the venue. The sensor will be used to trigger an alarm via the microcontroller. The controller utilizes pulse width modulation techniques to generate an alarm tone. A power amplifier is necessary if speaker used has high power characteristics (such as needed in an exam venue), as microcontroller power is far from sufficient.

Power amplifiers are specially designed amplifiers often used in audio applications, as they provide large gain, capability of carrying high current and compensations for non-linearities associated with high power transfer.

11.3.4.1. Smoke Detector Sensor:

The smoke detector to be used is the MD0203 gas sensor. This module features adjustable sensitivity (via a built-in potentiometer) as well as a wide detecting scope. It has high maximum sensitivity and a fast response time. There are only 3 pins for the sensor, HIGH and LOW, corresponding to Vcc (5V) and ground. The other pin is the output pin which is an analogue value relating to the volume of gas/smoke detected. This is to be connected to the ADC of the microcontroller and compared to a preset threshold value. The outputThe module is pictured to the right:

11.3.4.2. Microcontroller

The microcontroller used is the ATmega328. This microcontroller is fairly cost effective, with large variety of built in features and good performance specifications with respect to its cost. The more

affordable ATtiny models were considered but either featured less I/O pins than is needed to interface with the LCD, etc, or it did not have sufficient capabilities such as PWM generation and a built in ADC. The ATmega328 allows for more capable affordability by allowing for combining two systems onto one chip.

The generation of sound involves the passing of some waveform of an arbitrary frequency through a transducer that is able to convert the varying voltage to sound waves. The transmission of data through sound often provides a cost effective route as most microcontrollers possess the capability to generate voltage varying waveforms. Information can be transmitted through sounds though a vast amount of ways. These include simple combinations of ON and OFF sequences, producing something akin to the Morse code. Amplitude (via a DAC) or frequency modulation are other another possibilities, which in effect controls the volume or pitch of the sound, allowing for creation of vastly different sounds [16].

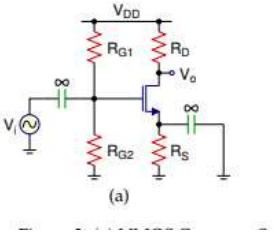
PWM can be used as a branch of amplitude modulation. For a PWM signal, the duty cycle is the ratio of ON (conduction) time over total switching period. This is hence proportional to the desired magnitude of the signal at that specific instant in time. Most microcontrollers available are capable of producing a PWM signal, either via built-in processes or via utilizing the built-in timer to generate a PWM signal [18].

The built in timer of the atmega328 will be configured as a counter and the output compare mode will be utilized to generate a PWM signal. The counter counts from 0 to 255 (maximum - total time), and the compare match can the be set at any value in that range (corresponds to ON time). For prescaler (and hence frequency) selection The Nyquist Theorem (fm<= 0.5fs) is to be used to select a sampling rate. The desired output frequency must be obtained to then calculate the sampling frequency. For an alarm tone a frequency of approximately 100 to 1000 Hz would be fairly acceptable [16].

The timer control registers (TCCROA/B), and output compare registers (OCR0A/B) are to be primarily used to set up the PWM generation [4]. The registers are displayed in Appendix G.

11.3.4.3. Power Amplifier

The power amplifier that can be used would just be a MOSFET (2N7000) driver configured a basic amplifier, as a full large signal power amplifier is not feasible. The basic MOSFET common source amplifier configuration is shown below:

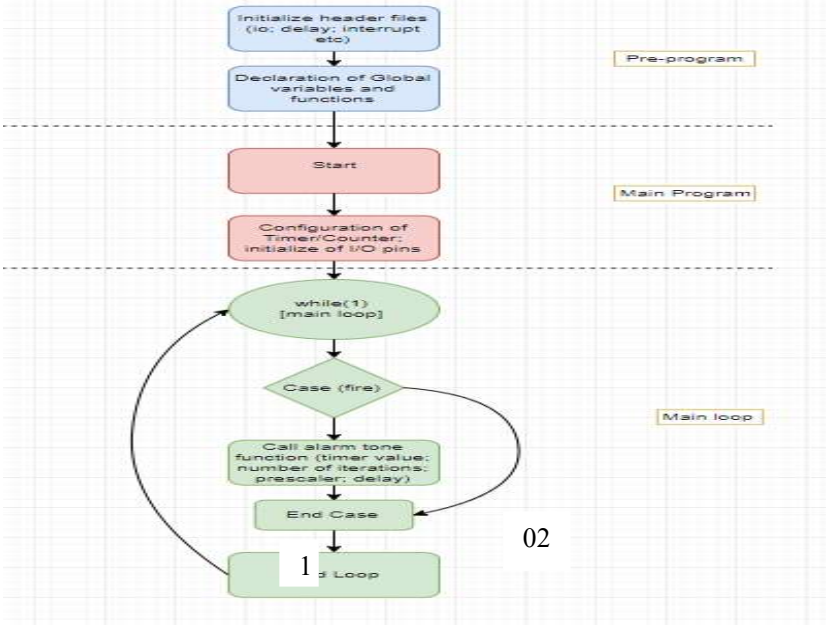


The large signal audio amplifier requires a dual power supply, at high voltage, to provide sufficiently high power to drive large speakers. The design of a large signal amplifier is considered but not implemented due to budget constraints. The high power amplifier design is separated into several stages to accommodate for the necessary complications of handling high power. These stages are the output stage, the driver stage, and the input stage[11]. A full design of a power amplifier is detailed in Appendix H, including theoretical background, component list and calculations as wells circuit diagram and simulation.

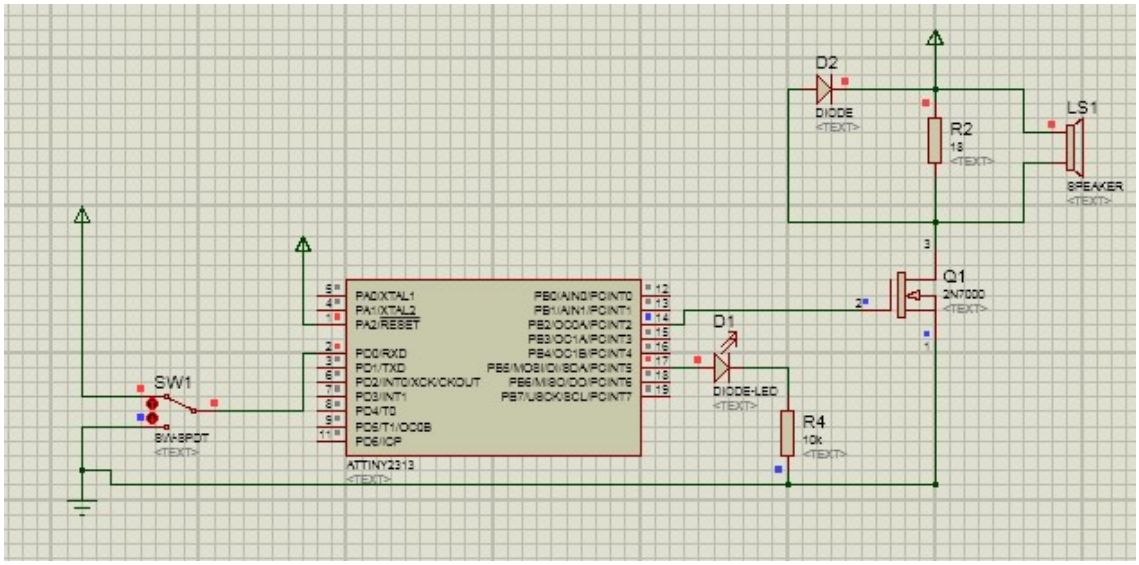
11.3.4. Speaker

For the high power amplifier designed, multiple 5 to 8W speakers may be powered. Instead, for the basic MOSFET driver, a 0.5W 8E speaker (M40IE060008) is used [9]:

11.3.5. Code Flow Diagram



11.3.6. Full circuit diagram and simulations

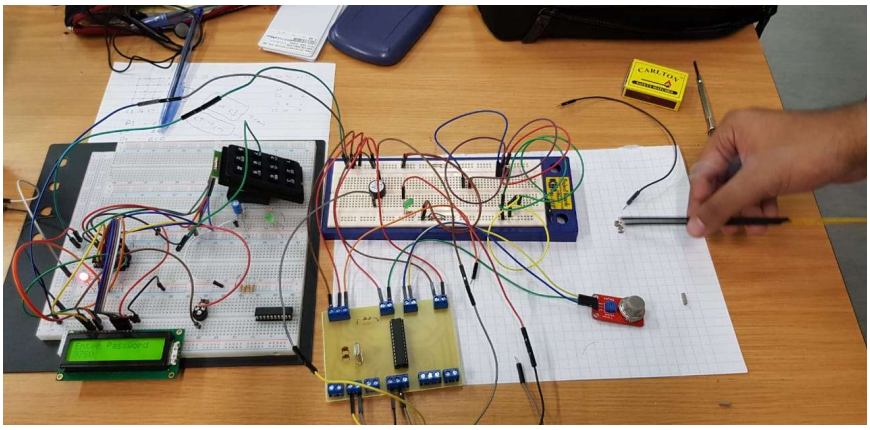


The above picture depicts the circuitry and simulation of the Fire Alarm circuit using an atmega2313. While no ADC pin is available, the basic logic conversion (~> 2.3V is HIGH) can be utilized. The microcontroller was not available, and the atmega328 was used as it allowed for minimization of the total system. The SPDT switch SW1 simulates the input from the smoke detector sensor as the Proteus platform does not have one on its database. The output at PINB5 is sent through an LED diode. This is

to simulate the signal going to the RF Identification system if the smoke detector sends a positive input as shown above. The alarm tones are generated by utilising pulse width modulation and generating various signals of different amplitude and rates of changing amplitudes (by varying the compare register value and therefore the duty cycle of the PWM signal). The MOSFET is set in the basic configuration designed to send drive a load. There is no gate resistance as the internal impedance of the MOSFET is very high. In order to properly support the speaker, the resistor needs to be a power resistor capable of handling high wattage. With the speaker resistance at 8 Ohms and the resistor at 47 Ohms, the power delivered is approximately 0.46 W. This is enough to power the speaker as it is rated at 0.5W [36]. The alarm tone was not simulated; the code will be debugged and simulated physically.

11.3.7. Implementation

The subsystem was implemented first on breadboard and then on PCB. The PCB design can be found in Appendix I. As mentioned previously, the ATmega328p chip was used to implement the system. Atmel studio and USBasp was used to program the chip. The system was implemented successfully, as shown below:



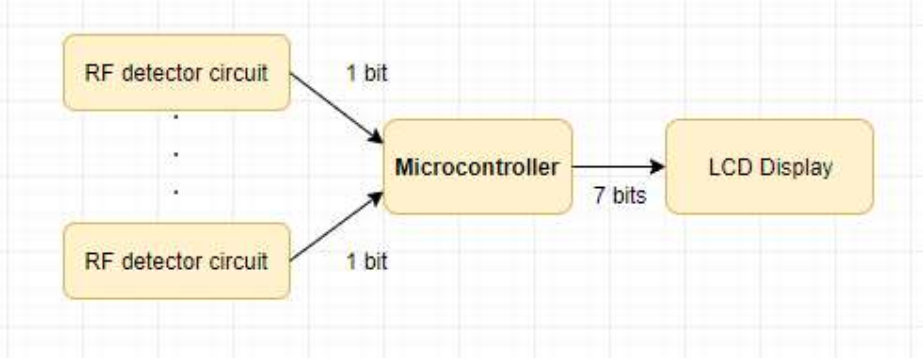
The above picture depicts the Fire alarm subsystem working and interacting with the RFID system. The system operates within expectations garnered from simulations and theoretical predictions.

11.3.8. Possible Expansion

Possible expansions naturally involve utilizing the high power amplifier designed, or adjusting it to provide even more power. Larger speakers would also coincide. A DC power supply providing split rail is required to purchased or designed for the power amplifier. Designing the power amplifier would require a transformer and expensive filtering circuits so as to minimize distortion in the power amplifier.

11.4. Cell phone detector subsystem

11.4.1. Block diagram:



11.4.2. Subsystem overview:

The system will utilize an RF detector circuit which samples RF signals and outputs the power that is proportional to the power at that point. This is the primary means of measurement for wireless signals. These detectors are the most frequently used to measure transmitter output power as well the measure and control of the Voltage Standing Wave Ratio (VSWR) [32].

The RF detector may be used in examinations as a means to detect cell phone usage. There are a few limitations to this however. These include the fact that the RF detector only detects if transmission within the specified range that is provided by the circuit schematic. Also, the RF detector would not triangulate the exact device that is transmitting, i.e. if two people are within its range and one of them has a cell phone on, the device will be triggered. However, no information would be provided as to which of the two people has the cell phone. As such, in an examination venue, it is proposed that each RF detector be assigned to a specific table number which corresponds to a certain student. Another limitation is that as the RF detectors are naturally designed to detect signal transmission, it cannot detect a cell phone that is switch off or is in 'airplane' mode, as no transmissions are being made [32].

A parallel input digital-to-analogue converter (DAC) was considered to reduce the microcontroller pins used and increase the inputs available. It achieves this by taking multiple digital inputs and converting that to a single analogue value. The available parallel input DAC's are highly expensive and hence not considered. As such, the microcontroller is connected directly to the digital inputs. The number of possible inputs is therefore limited by the number of available pins. The corresponding seat number is found and passed to the LCD display.

11.4.3. Components List

- 1 x ATMEGA328P-PU microcontroller R57.06
- 1 x LMB162NFC LCD R194.76
- 2 x CA3130 operational amplifier R 44.54
- 2 x 1.2MΩ resistor
- 2 x 2.2MΩ resistor
- 6 x 1kΩ resistor
- 2 x 100kΩ resistor
- 2 x 22pF ceramic capacitor
- 6 x 22pF ceramic capacitor
- 2 x 0.1µF ceramic capacitor
- 2 x 100µF/25V electrolytic capacitor
- 2 x BC548 NPN BJT

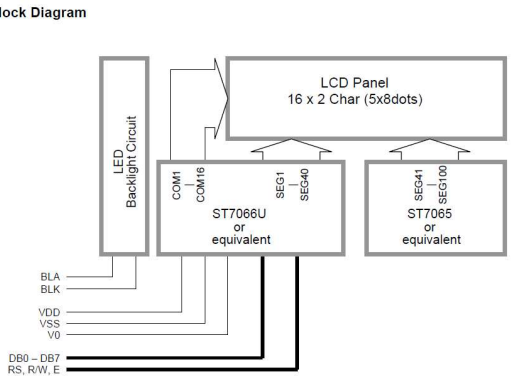
- 2 x antenna
- 3 x LEDs

Total:

R296.26

11.4.4. Circuit design

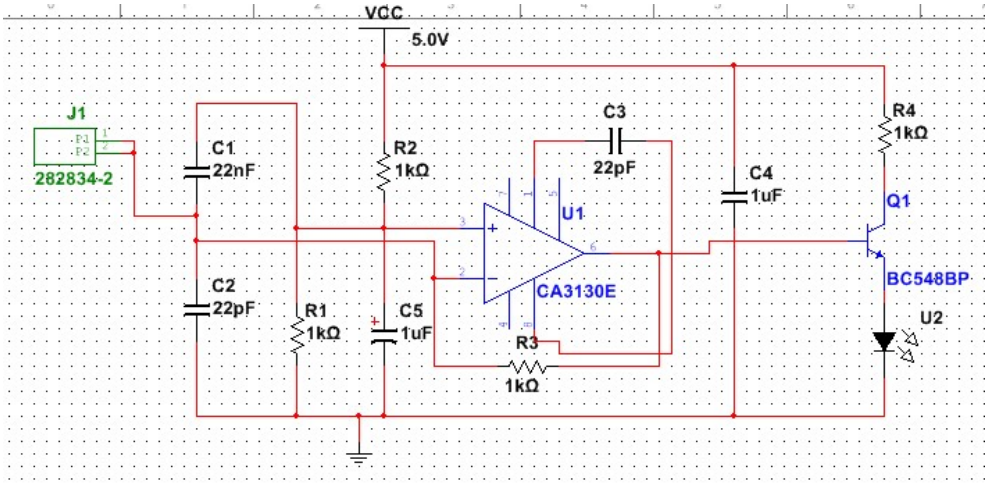
11.4.4.1. RF detector



The 22nF capacitor C1 and the antenna act as a gigahertz loop that captures the RF signal. That capacitor is also connected to both of the inputs of the CA3130 operation amplifier[41]. The operational amplifier is used to convert the current to a voltage with the capacitor C1, hence why C1 is connected to both the inverting and non-inverting loops. This op amp uses CMOS and MOSFET technology and very high input impedance and gain-bandwidth product.

The relatively large values of resistor 2 and capacitor C3 keeps the non-inverting input stable while giving an output to high state. This is therefore the feedback of the amplifier circuit. The resistor 3 matches the non-inverting input with the output when it is high. Capacitor C4 is connected between the pin 8 (strobe) and pin 1(null), to allow for the phase compensation and gain control to optimize the frequency response, allowing for better and more accurate output results.

When a signal is detected at C1 the output (pin 6) of the CA3130 becomes high. This signal is detected and displayed on the LED which is also passed to the microcontroller. The detector range is between 0-1.5m in which a signal can get detected. The signal strength and probability of detection are naturally directly proportional to each other. The signal strength also affects the magnitude of output voltage. The circuit is shown below:



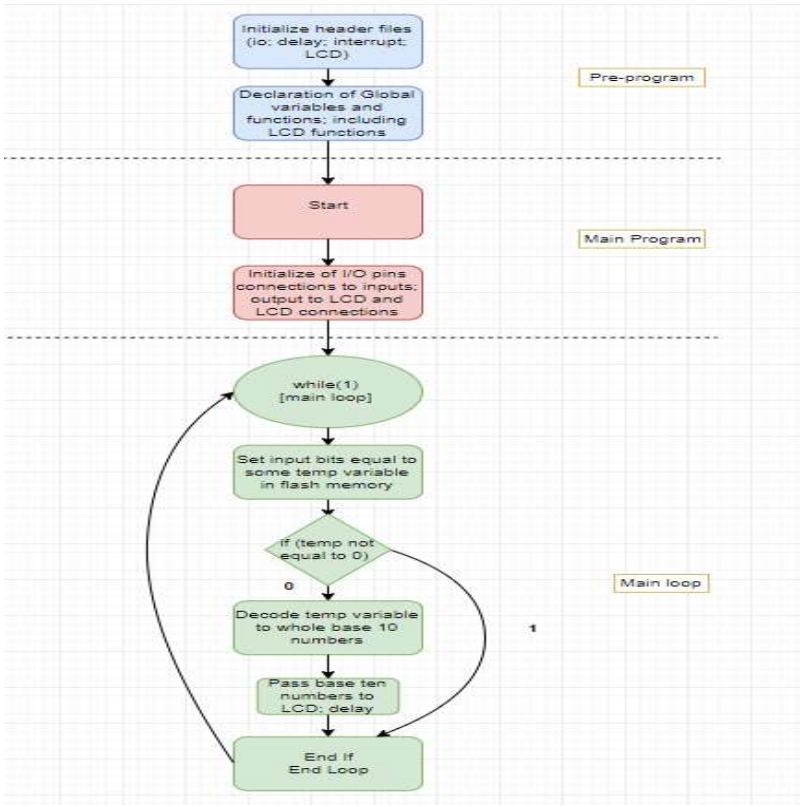
11.4.4.2. LCD

The LMB162NFC LCD was chosen to be used. This LCD can operate in one of two modes, 4 bit mode or 8 bit mode, relating to the number of bytes required for data transfer. The LCD also requires inputs to the Register Select (R), Read/Write control (R/W) and Data Enable (E). It also allows for the contrast of the display to be adjustable [42]. Almost all 16x2 LCDs operate the same way and are hence interchangeable. The figure below is the block diagram as taken from the datasheet [42]:

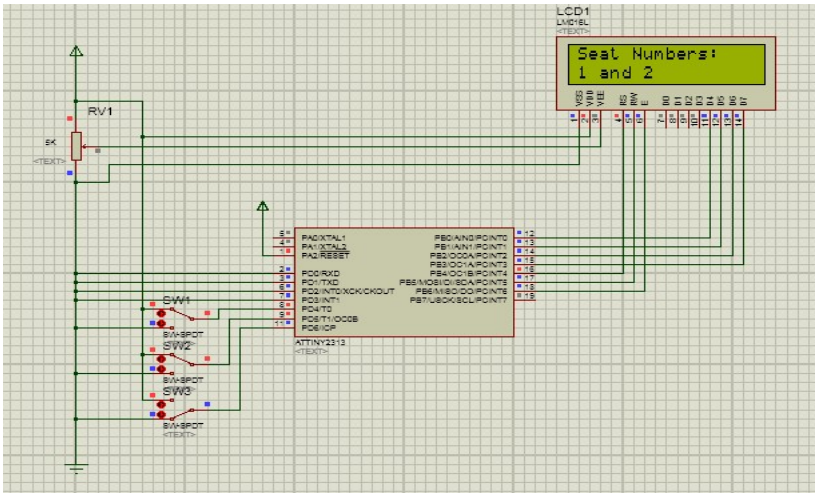
11.4.4.3. Microcontroller

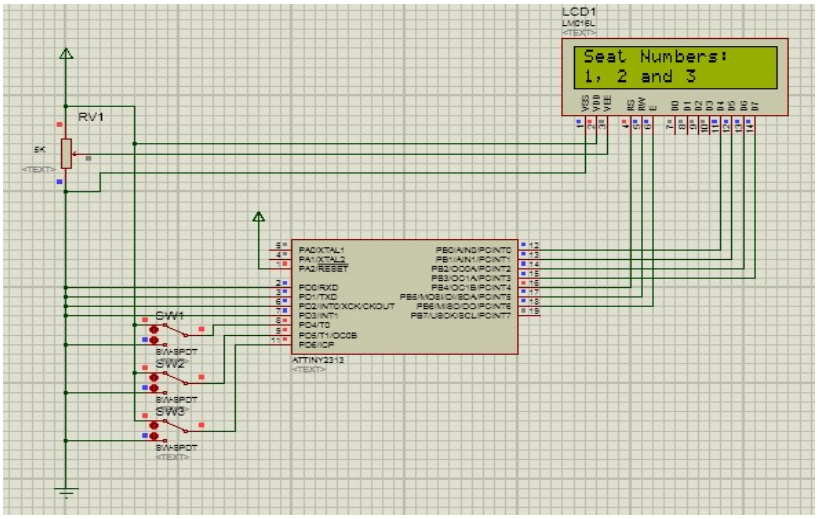
The microcontroller used is the ATmega328. This microcontroller was chosen for the purpose of uniformity as well cost compensation and circuit simplification. The ATtiny2313 was initially considered and tested but was both unprocurable as well having insufficient performance to run more than one subsystem. The atmega328 has a maximum of 23 I/O pins, with 5 of them being multiplexed as ADC channels. This makes the ATmega328 more than capable of handling both the RF detection subsystem and the Temperature control subsystem.

11.4.5. Code Flow diagram



11.4.6. Full Circuit diagram and simulations

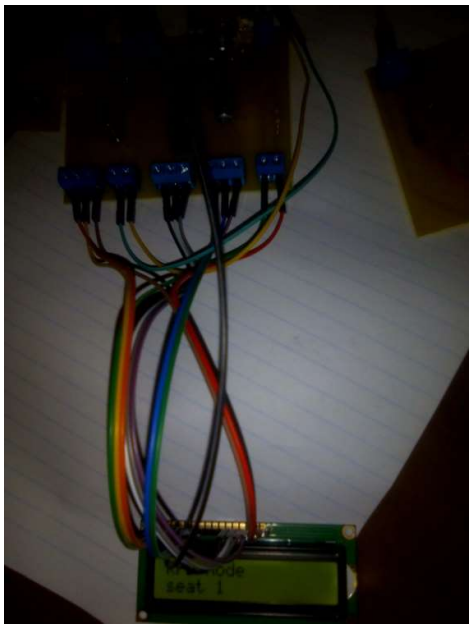




The above two pictures depict the simulations of the cell phone detector system on the platform Proteus. SPDT switches are used to simulate the inputs of the RF detector circuits. The inputs are processed using various bitwise manipulations to generate the seat numbers and pass them to the LCD display. It achieves this by analysing the integer values of the input port. The exact LCD display that will be used is not on the Proteus platform, therefore another similar LCD display was selected, the same 14 inputs and 2x16 display.

11.4.7. Implementation

The system was implemented first on breadboard and then on PCB. The PCB design can be found in appendix H. The system was implemented in conjunction with the Temperature Control system, allowing for one chip instead of two and also multiplexing the LCD, allowing for one instead of two. This drastically reduced the system costs.



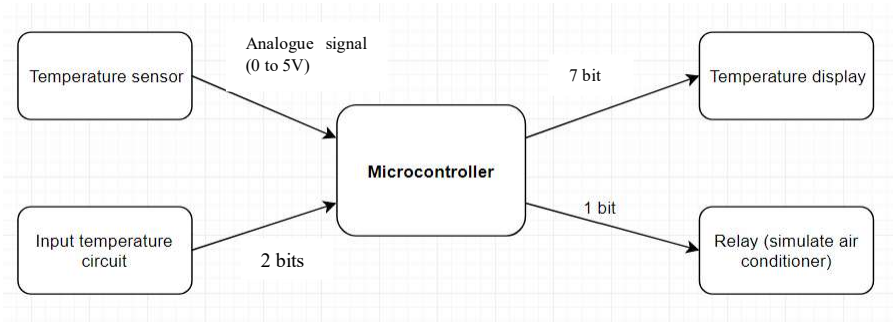
The picture on the left depicts the PCB implementation of the Cell Phone detector subsystem. The same circuit is used for the temperature control subsystem.

11.4.8. Possible expansion

To expand this subsystem such that it may operate on a larger scale would simply require incorporating parallel to serial conversion to allow for a much larger number of inputs to the microcontroller. This can be achieved either by using a synchronous multiplexer based design, or a parallel input digital-to-analogue converter such as the AD7537JNZ, a 2 channel 12 input DAC [43]. This DAC is available at Mantech at R1423.02.

11.5. Temperature control subsystem

11.5.1. Subsystem Block Diagram



1.5.2. Subsystem Overview

The above subsystem is applied towards the purpose of energy saving, by ensuring the air conditioners are on when it is required. A temperature sensor is utilized to measure the ambient temperature within the exam venue. A large variety of temperature sensors are available, varying in the type of output of the sensor module. These include voltage output, SPI output, etc. This value is then passed to the microcontroller, which is required to be able to receive the temperature value in whatever form it may be in. The microcontroller then uses either oversampling or averaging, or both, to improve the precision of the measurement. A preset room temperature is stored on the microcontroller as a constant (pre-start), this value can be toggled by the user via push-buttons. Both the measured temperature and the set temperature are displayed on a LCD. The microcontroller also compares the measured and set temperatures and outputs a HIGH is the measured value is higher, and a LOW if it is not. This output of the microcontroller goes to a relay which can be connected to an air conditioner.

11.5.3. Components List

• Potentiometer	~R12
• 1 x ATMEGA328P-PU microcontroller	R57.06
• LM35 Precision Temperature sensor	
• 1 x LMB162NFC LCD	R194.76
Total:	R267.02

11.5.4. Circuit Design

11.5.4.1. Temperature Sensor

The sensor to be used is the LM35 Precision Temperature sensor. The sensor has three pins, VCC, Ground and Output pin. The output values is calculated via the ratio 0.01V/ degrees Celcius. The sensor is found to be fairly accurate, however the low voltages and low rate of change may lead to increased error when using the ADC. The TC1047 Precision Temperature to Voltage Converter was previously considered and was subsequently purchased. However, as it being available only in Surface Mount Device (SMD) package, it generated far too many complications with regards to testing (breadboard implementation).

11.5.4.2. Microcontroller

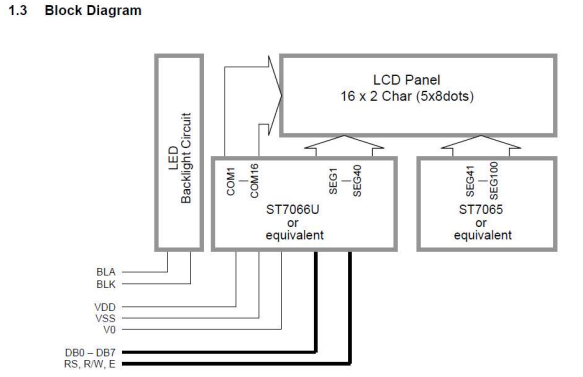
The microcontroller to be used is the ATMEGA328P-PU. This microcontroller has a 23 I/O pins, easily capable of interfacing the LCD in either 4 bit or 8 bit modes. The ATMEGA328P-PU also has a built-in 10 bit 6 channel analogue to digital converter [17]. One of these pins will be connected to the output of the temperature sensor. The microcontroller will use averaging and oversampling to improve the precision of the measured value. The theory of oversampling requires taking multiple samples of the same input i.e. taking 16 samples of a 10-bit value will give a 14-bit value, decimating this by two bits reduces multiplicative errors; leaving a 12-bit value that offers more precision. This can be extended to generate greater bit increases, at the cost of speed. The ADC will also be used to take in the input temperature value via a potentiometer. The upper and lower bounds of the input are set via the conversion equation in the code. The microcontroller will, via the built in ALU (Arithmetic Logic Unit), perform a check between the measured and set temperature values, and output a single bit as HIGH if the measured value is greater, and vice versa.

11.5.4.3. Potentiometer (input)

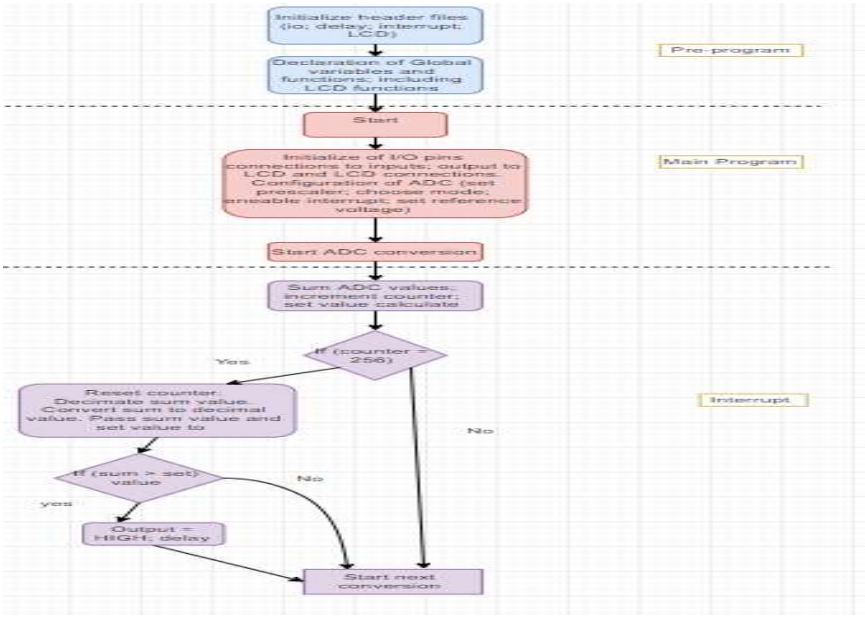
A basic potentiometer with shaft is used to provide an input voltage to an ADC pin varying from 0 to 5V. This is then converted to a temperature value by the microcontroller and then passed to the display.0

11.5.4.4. LCD

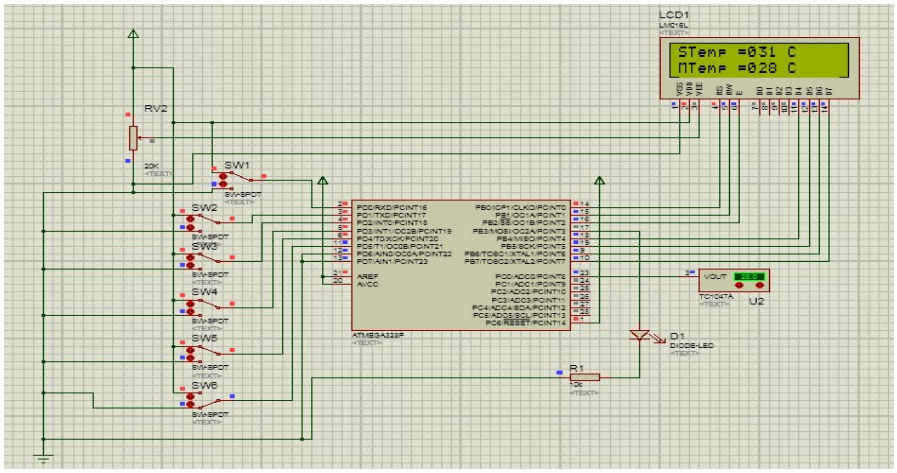
The LMB162NFC LCD was chosen to be used. This LCD can operate in one of two modes, 4-bit mode or 8-bit mode, relating to the number of bytes required for data transfer. The LCD also requires inputs to the Register Select (R), Read/Write control (R/W) and Data Enable (E). It also allows for the contrast of the display to be adjustable [42]. The figure to the side is the block diagram as taken from the datasheet:

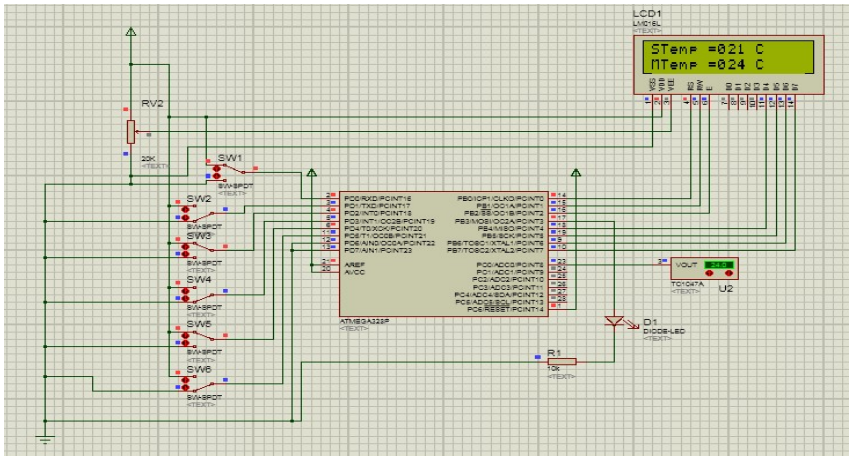


11.5.5. Code Flow diagram



11.5.6. Full circuit diagram and simulations

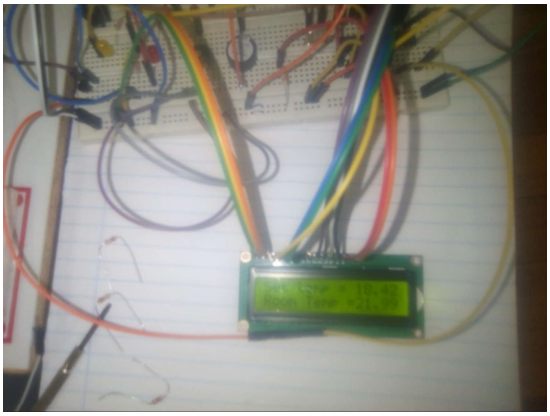


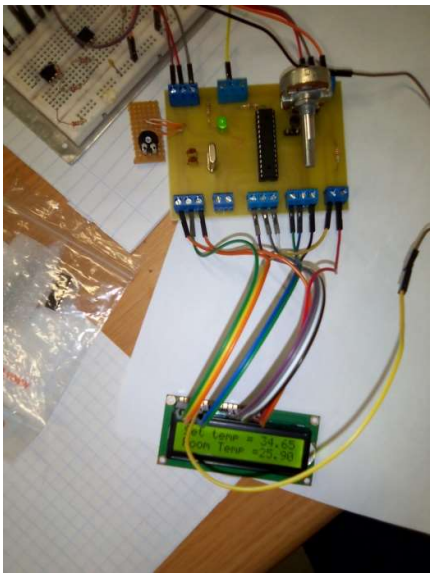


The full circuit simulations are shown above for both cases, i.e. set temperature less than, and set temperature greater than the measured temperature. The ADC was configured in single conversion mode, with a prescaler of 128, setting the ADC clock as 125 kHz. This lies within the optimum range of 50 kHz to 200kHz, in order to get the full 10-bit output. The system was tested with oversampling and decimation, via a summation loop and shifting, and the results were found to be negligible, mostly because a minimum noise level is required for it be effective. Therefore, for the purpose of the simulation, the oversampling and decimation were not used, but would probably be used when fully implementing the circuit, as noise is sure to be present then. The output pin PB3 going to LED simulates the pin going to the air conditioner. The outputs are satisfactory for this. Note, in the above circuit, SPDT switches are used to generate the input via digital pins. This is not efficient and was later replaced with a Potentiometer as input connected to an ADC pin, mirroring the temperature sensor. Only one ADC channel may be used at a time, hence a looping switch construct is to be used.

11.5.7. Implementation

The subsystem was implemented in conjunction with the cell phone detection system with both using the same LCD and microcontroller. The circuits were implemented successfully on both breadboard and PCB. The PCB designs can be found in Appendix H. The response of the system with regards to the generation of a control signal for A/C was largely successful however the temperature was found to display some reasonable errors (~max of 2 degrees celcius).

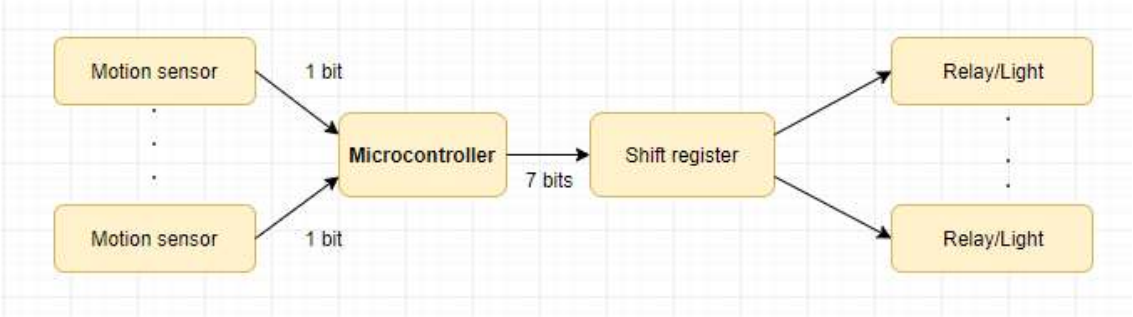




The above pictures depict the temperature control system tested on breadboard and then later implemented on PCB. It was found, as to be expected, that the sensor accuracy and sensitivity as well as possible ADC noise improved when implemented on PCB.

11.6. Lighting control subsystem

11.6. 1. Subsystem Block Diagram



11.6.2. Subsystem Overview

The subsystem attempts to conserve energy via controlling the lighting within the venue. The lights are only switched on if at least a single person is in the vicinity of the light. There are various types of motion sensors available depending on the type frequency spectrum the device senses. These types are passive infrared (PIR) sensors; ultrasonic sensors, and IR sensors. The infrared sensors are small, low power and therefore reasonably cheap. They operate on detecting heat emitted from a human in the proximity of the sensor. The ultrasonic and IR sensors detect sound and light respectively. A PIR sensor is to be utilised as it provides the most cost effective and system specific solution [33]. The motion sensors are to be directly connected to the microcontroller (possible as their outputs are digital based). The microcontroller receives these digital inputs and processes them, sending out a digital stream and clock output corresponding to the period of the digital stream. The clock output needs to stop once the bitstreams is done. In the event of a change with one of the motion sensors, this is detected by the microcontroller, and a new bitstream is sent. The clock and bitstream outputs sent to a serial to parallel register, which serves the purpose of decreasing the number outputs the microcontroller needs to support. It also allows the system to be more spaced out.The outputs of the shift register are sent to relays corresponding the lights for each motion sensor.

11.6.3. Component List

- 1 x HC-SR501 PIR motion detector R62.25
- 1 x ATMEGA328P-PU microcontroller R57.06
- 1 x 74HC595 8-bit serial in parallel out shift and latch register R6.90

Total: R189.46

11.6.4. Circuit Design

11.6.4.1. Motion sensor

The sensor to be used is the HC-SR501 PIR motion detector. The motion detector has output HIGH value of 3.33V and LOW of 0V (TTL logic). The sensor operates in direct relation to input (non-

repeatable trigger), i.e. if all people leave range of the of the sensor, after a small delay, the output goes from HIGH to LOW. The sensor has a manually adjustable range function, with a minimum range of 3m and a maximum of 7m [46]. The motion sensor is depicted to the side:

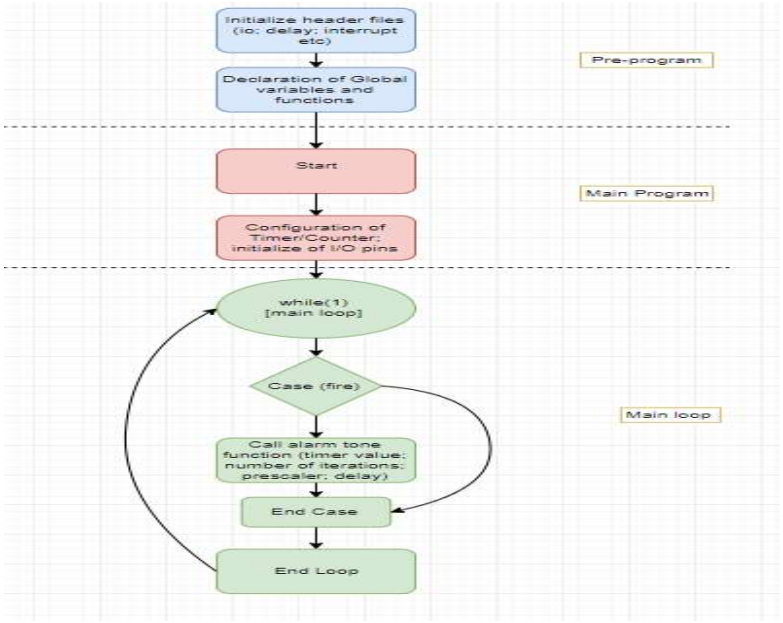
11.6.4.2. Microcontroller

The microcontroller to be used is the ATMEGA328P-PU. The cost effective ATtiny2313 was considered at first but it had insufficient flash memory (2k) [34]. This microcontroller has 23 available I/O pins [44]. Taking out 5 to the shift register, this leaves 18 pins for digital inputs. The ATtiny85 is not used as it has too few I/O bits for digital inputs, even though it would have enough flash memory.

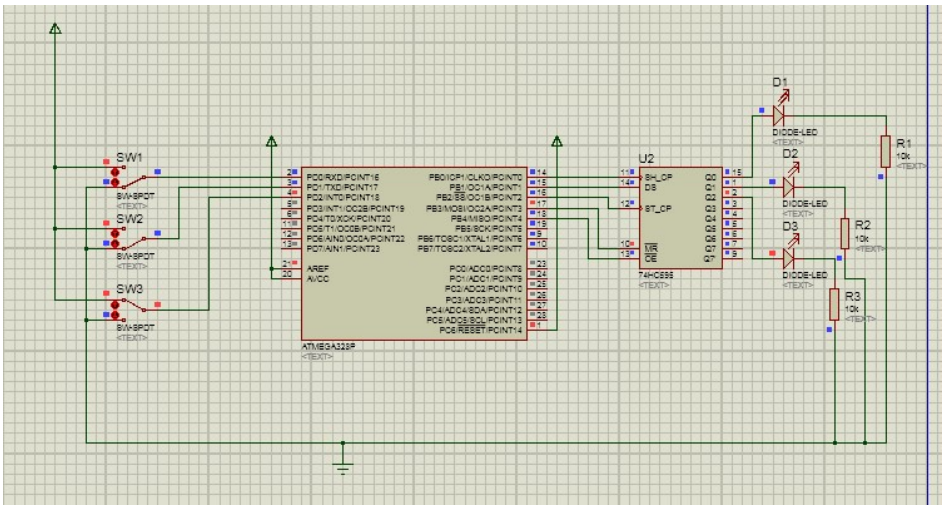
11.6.4.3. Shift register

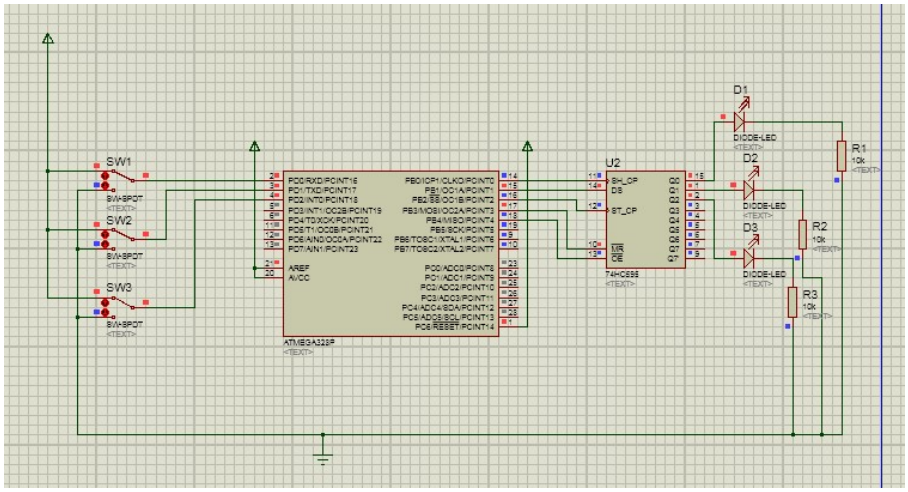
The 74HC595 8 bit serial in parallel out shift and latch register is to be used to increase the output capabilities of the system. This is an upgrade to the previous considered shift register as it also offers the latch capability. The shift register is used when considering the fact that the lights and motion sensors should a reasonable distance apart, as such, instead of passing a large number of signals (wires), only one will suffice. The shift register is an 8-bit output hence it can take an 8 bit wide bitstream as an input, and in the system, can connect to a maximum of 8 lights/relays. The latch function allows the register to store the input variables fully before passing them to the outputs. This synchronizes the outputs, passing values to them simultaneously.

11.6.5. Code Flow diagram



11.6.6. Full circuit diagram and simulations

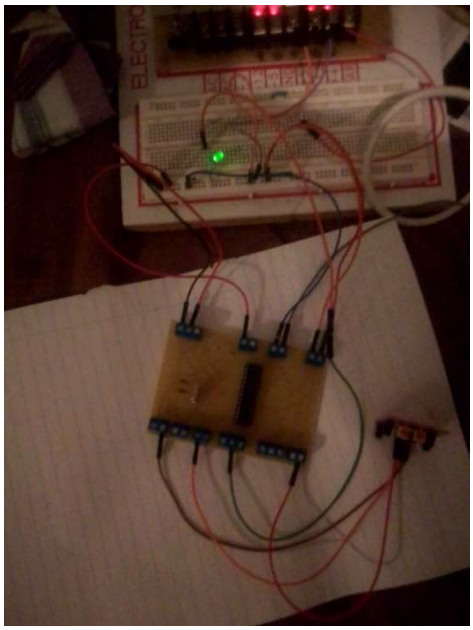


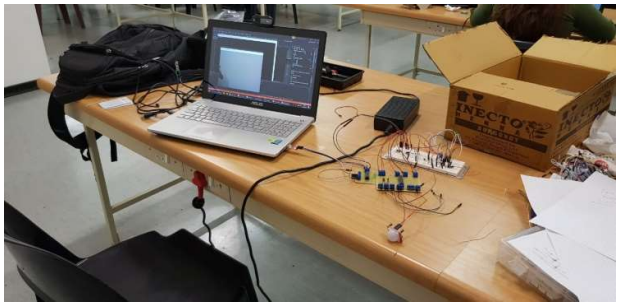


The above diagrams depict the working simulations on the Proteus platform. SPDT switches are used to simulate the outputs of the motion sensors which are not on the Proteus database.

11.6.7. Implementation

The subsystem was successfully implemented on both breadboard and PCB. The PCB designs can be found in Appendix H. The subsystem was implemented in conjunction with the Fire Alarm subsystem in an effort to decrease costs and provide some simplification. In implementation the shift register purchased was not operating as was expected, as such the outputs are passed directly from the microcontroller. As the motion sensor detects movement and not occupation of space, a time period of monitoring a low would be required to ensure the lighting does not switch off due to lack of motion and not lack of presence.





The above pictures depict the PCB implementation of the lighr control system. The picture on the right shows the integration of the light control subsystem and the Video surveillance subsystem.

11.6.8. Possible expansion

The system can be expanded via using a parallel input DAC with I2C protocol to allow for more inputs (motion sensors) to connect to a single microcontroller. Albeit it is cheaper to simply replicate the same circuit, without the expensive DAC, and arranging them in order. Just adding another shift register can increase the number of outputs a single microcontroller can drive, this is not necessary is the circuit is just replicated.

11.7. Feasibility Studies:

1.7.1. Alternative Economic Solutions

For the temperature control system could possibly use either air conditioners or fans for the purpose of cooling.

	Fans	**Air Conditioners**
Cheapest found price	R997	R3 499 [1]
Average range provided	8m (possibly)	~40m [1]

As can be seen from the table above mounted fans are far cheaper in comparison to air conditioners, while operating on a far smaller area. For an exam venue which can be considered reasonably large, possibly 100m x 50m, with two floors, a much higher number of fans would be required compared to air conditioners. Another factor to consider is that air conditioners are far quieter than fans, which a very desirable characteristic for an exam venue. For these reasons, air conditioners are chosen. For the project to be built, considered a prototype, a relay would be used to simulate the switching on/off of the air conditioner.

The purpose of the RF detection system is to prevent cheating via the usage of cellular phones in the examination hall. An alternative that achieves the same purpose, albeit via a very different method,

would a metal detection system. Various limiting constraints were encountered unfortunately, such limitations to the range of application; design issues would regard to size and cost, etc.

The most affordable microcontrollers were considered first when developed the design solutions, these constitute the ATtiny range. These microcontrollers are highly cost effective, with most of them under R30. However, the flash memory for these microcontrollers is usually very limited, for instance, the ATtiny2313A has only 2k flash. This does not allow for applications that involve more complex codes. In these cases, the ATmega328 was selected as it has very rounded specifications (10-bit ADC, 20k flash).

1.7.2. Component Costs:

Subsections:

Fire Alarm Subsystem:	R99.67
Cell Phone Detector subsystem:	R296.26
Temperature Control subsystem:	R12.00
Light Control subsystem:	R127.21
Sub-Total:	R535.14

Note: Subsystem costs decreased due components ATmega328 and LCD being used across subsystems

1.7.3. Services and additional costs

Labour: (50 x R670)(Category D of ECSA hourly rates)	R33500
USBAsp programmer:	R200
Sub-Total:	R33700
Total (services and raw materials)	**R33793.47**

Other than the labour costs, the other costs are reasonably within estimates. Taking into consideration the components already in possession further lowers the total cost.

1.8. Application of Knowledge using Engineering Design [ELO2]

- The experienced of working with the AVR ATmega32 in the Electronic Design 2 course as well as the Digital Systems course allows me to easily familiarize myself with the various other AVR microcontrollers.
- The knowledge I have of timers and counters from Digital Systems provides a knowledge base for working with pulse width modulation.
- The knowledge acquired from Analogue Electronics with regards to amplifier theory and design were found to be useful.
- The knowledge with regards to bit logic and manipulation studied in Digital Electronics and Digital Systems briefly, provide a foundation for accessing the I/O ports of registers.
- The work done with the ADC in the Electronic Design 2 course with regards to a voltage measurement system allowed the Temperature Control system design to progress smoothly as much of the configurations are already known.

12.Conclusion

This report outlines the research and designing of the Smart Examination hall and energy efficiency. The paper design of the four sub-systems have being completed successfully. Different alternatives have been explored in achieving the final design for all sub-systems. Each subsystem has undergone a feasibility study and budget analysis, implementation and testing. The system was implemented within the time and budget constraints. Most specifications have been met.

The domain expert has been given regular feedback on the development of the system.
A substantial amount of information was provided in the forms of block diagrams, calculations, theoretical performance, schematic circuit diagrams, simulations and observations and testing. Overall, the designing and implementation was a successful and the results obtained were expected and met the required specifications. We can conclude that the problem identified can be solved using this system.

13.References

[1] [Online]. Available: http://www.mantech.co.za/datasheets/products/EM-18-E34.pdf. [Accessed 22 03 2019].

[2] [Online]. Available: https://electrosome.com/em-18-rfid-module-pic/. [Accessed 22 03 2019].

[3] [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/30010038c.pdf. [Accessed 20 03 2019].

[4] [Online]. Available: http://www.electronicwings.com/pic/interfacing-lcd-16x2-in-4-bit-mode-with-pic18f4550-. [Accessed 20 3 2019].

[5] [Online]. Available: https://www.instructables.com/id/MICRO-CONTROLLER-to-PC-Communication-Via-PL2303-US/. [Accessed 28 3 2019].

[6] B. Mesnik, "Fingerprint and Facial Recognition Access Control," 12 August 2016. [Online]. Available: https://www.linkedin.com/pulse/fingerprint-facial-recognition-access-control-jensen-gooduuyn-lion-. [Accessed 3 March 2019].

[7] "Biometric authentication: what method works best?," Technovelgy LLC, [Online]. Available: http://www.technovelgy.com/ct/Technology-Article.asp?ArtNum=16. [Accessed 3 March 2019].

[8] H. Hassan, R. A. Bakar and A. T. F. Mokhtar, "Face recognition based on auto-switching magnetic door lock system using microcontroller," ICSEngT, 2012.

[9] Microchip, "16F87XA Data Sheet," 2003.

[10] S. Dutta, "Interfacing Camera with PIC Microcontroller via Matlab GUI," 24 March 2014. [Online]. Available: http://www.nbcafe.in/interfacing-camera-with-pic-microcontroller-via-matlab-gui/. [Accessed 6 March 2019].

[11] D. Electronics, "USB - TTL SERIAL USART CP2102 MODULE," [Online]. Available: https://www.diyelectronics.co.za/store/serial/40-usb-ttl-serial-usart-cp2102-module.html?search_query=usart&results=7. [Accessed 16 March 2019].

[12] "Releases," 2019. [Online]. Available: https://opencv.org/releases.html. [Accessed 10 March 2019].

[13] "Installation in Windows," 2019. [Online]. Available: https://docs.opencv.org/2.4/doc/tutorials/introduction/windows_install/windows_install.html#windows-installation. [Accessed 10 March 2019]

[14] "How to build applications with OpenCV inside the Microsoft Visual Studio," 2018. [Online]. Available: https://docs.opencv.org/2.4.13.7/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html. [Accessed 10 March 2019].

[15] M. K. Mandal, 2018. [Online]. Available: https://github.com/manashmndl/SerialPort. [Accessed 22 March 2019].

[16] "Low-cost techniques for sound generation", *Embedded*, 2019. [Online]. Available: https://www.embedded.com/design/real-time-and-performance/4006419/Low-cost-techniques-for-sound-generation. [Accessed: 26- Mar- 2019]

[17] "How to Generate Sound Using PWM With PIC Microcontroller", *Engineersgarage.com*, 2019. [Online]. Available: https://www.engineersgarage.com/embedded/pic-microcontroller-projects/How-to-Generate-Sound-Using-PWM-With-PIC-Microcontroller. [Accessed: 27- Mar- 2019]

[18] "Two-tone Alarms", *Fedsig.com*, 2019. [Online]. Available: https://www.fedsig.com/product/two-tone-alarms. [Accessed: 27- Mar- 2019]

[19] "Motion Detector Circuit with Working Description and Its Applications", *ElProCus - Electronic Projects for Engineering Students*, 2019. [Online]. Available: https://www.elprocus.com/motion-detector-circuit-with-working-description-and-its-applications/. [Accessed: 27- Mar- 2019]

[20] [Online]. Available: https://www.instructables.com/id/MICRO-CONTROLLER-to-PC-Communication-Via-PL2303-US/. [Accessed 28 3 2019].

[21] "ACTIVE VS. PASSIVE RFID," Atlas RFID Solutions, [Online]. Available: http://www.atlasrfid.com/jovix-education/auto-id-basics/active-rfid-vs-passive-rfid/. [Accessed 24 February 2018].

[22] "Microchip PIC24HJ128GP202," RS Components, [Online]. Available: https://za.rs-online.com/web/p/microcontrollers/0549413/. [Accessed 24 February 2018].

[23] "Microchip PIC16F877A," RS Components, [Online]. Available: https://za.rs-online.com/web/p/microcontrollers/4671690/. [Accessed 24 February 2018].

[24] Unifore, "DIFFERENCE BETWEEN PIR MOTION SENSOR AND INFRARED BEAM MOTION SENSOR," the innovative security world, 2017. [Online]. Available: https://www.hkvstar.com/technology-news/difference-between-pir-motion-sensor-and-infrared-beam-motion-sensor.html. [Accessed 21 February 2018].

[25] "Long range IR transmitter," 2016. [Online]. Available: https://electronicsforu.com/electronics-projects/hardware-diy/long-range-ir-transmitter. [Accessed 21 February 2018].

[26] A. HeydarNoori, "Cheriton School of Computer Science, University of Waterloo," 31 October 2003. [Online]. Available: https://cs.uwaterloo.ca/~apidduck/CS846/Seminars/abbas.pdf. [Accessed 22 February 2018].

[27] [Online]. Available: https://www.microchip.com/wwwproducts/en/PIC16F877A. [Accessed 25 Feb 2019].

[28] [Online]. Available: https://www.microchip.com/wwwproducts/en/PIC24HJ128GP202. [Accessed 25 Feb 2019].

[29] [Online]. Available: https://www.edaboard.com/showthread.php?41198-What-is-the-difference-between-13-56-MHz-and-125kHz-RFID-tag. [Accessed 26 Feb 2019].

[30] [Online]. Available: https://docs-emea.rs-online.com/webdocs/161b/0900766b8161beae.pdf. [Accessed 22 03 2019].

[31] [Online]. Available: https://electrosome.com/pl2303-usb-to-uart-converter/. [Accessed 26 3 2019].

[32] ["RF Detectors For Wireless Devices", Electronic Design, 2019. [Online]. Available: https://www.electronicdesign.com/test-amp-measurement/rf-detectors-wireless-devices.[Accessed: 14- Mar- 2019]

[33] "How do Motion Sensors Work? A Beginners Guide by SafeWise", *SafeWise*, 2019. [Online]. Available: https://www.safewise.com/resources/motion-sensor-guide/.[Accessed: 23- Mar- 2019]

[34] [Online].Available:http://www.mantech.co.za/datasheets/products/ATTINYXXX-190116A.pdf. [Accessed 26 03 2019].

[35] [Online].Available:http://www.mantech.co.za/datasheets/products/MD0203_KEYES.pdf. [Accessed 26 03 2019].

[36] [Online].Available:http://www.mantech.co.za/datasheets/products/M40IE060008.pdf. [Accessed 26 03 2019].

[37] [Online].Available:http://www.mantech.co.za/datasheets/products/ATMEGA32X.pdf. [Accessed 26 03 2019].

[38][Online].Available:https://moodle.ukzn.ac.za/pluginfile.php/157706/mod_resource/content/3/Notes/4EC%20PwrAmpNotes.pdf [Accessed 22 03 2019].

[39] "RF detector circuit electronic project using transistors", *Electroniccircuitsdesign.com*, 2019. [Online]. Available: http://www.electroniccircuitsdesign.com/radio-circuits/rf-detector-circuit-diagram.html[Accessed: 18- Mar- 2019]

[40] "RF signal detector circuit", *theoryCIRCUIT - Do It Yourself Electronics Projects*, 2019. [Online]. Available: http://www.theorycircuit.com/rf-signal-detector-circuit/.[Accessed: 19- Mar- 2019]

[41] J. Cubetech, "SIMPLE MOBILE DETECTOR CIRCUIT," instructables, 01 2016. [Online]. Available: https://www.instructables.com/id/simple-mobile-detector-circuit/.[Accessed 17 03 2019].

[42] [Online].Available:http://www.mantech.co.za/datasheets/products/LMB162NFC.pdf. [Accessed 22 03 2019].

[43] [Online].Available:http://www.mantech.co.za/datasheets/products/ AD7537JNZ-181214A.pdf. [Accessed 22 03 2019].

[44] [Online].Available:http://www.mantech.co.za/datasheets/products/ATMEGA328P-PU.pdf. [Accessed 22 03 2019].

[45] [Online].Available:http://www.mantech.co.za/datasheets/products/TC1047.pdf. [Accessed 22 03 2019].

[46] [Online].Available:http://www.mantech.co.za/datasheets/products/MD0357_KEYES.pdf. [Accessed 22 03 2019]

[47] L. ATMega328, "LCD Interfacing 4 and 8 bit mode library for AVR ATMega328", *Programming-electronics-diy.xyz*, 2019. [Online]. Available: http://www.programming-electronics-diy.xyz/2016/08/lcd-interfacing-library-4-and-8-bit-mode.html.[Accessed: 23- Mar- 2019]

Appendices

Appendix A –List of Contributions [ELO8]

Database and software integration (Devashnee Bhagawandin 214502393)

- Create the database which will store the necessary information required for the interconnected components to access.
- Software for project integration.
- Complete design of the computer application unit and the database
- Sequence diagram simulation of software
- Involved in composure of the Phase 3 report.
- Involved in computer application and database slides for presentation

Rishay Ramcharan 214504863

- Complete design of the RF ID authentication subsystem.
- Creation of basic prototypes and testing of the RF ID authentication subsystem.
- Complete budget analysis for RF ID authentication subsystem.
- Involved in composure of the Phase 3 report.
- Involved in composure of the Phase 3 presentation slides.

Keshav Jeewanlall (213508238)

- Composed the Abstract in the Phase 3 report.
- Complete design of the facial recognition and video surveillance subsystem.
- Creation of basic prototypes for testing functions to be used for facial recognition and video surveillance subsystem.
- Complete budget analysis for the facial recognition and video surveillance subsystem.
- Involved in composure of the Phase 3 report.
- Involved in composure of the Phase 3 presentation slides.

Energy Saving, Safety and Isolation Subsystem (Sashin Ramdhani 214513737)

- Complete design of the Energy Saving, Safety and Isolation Subsystems.
- Design of Cell Phone detector subsystem,
- Design of Fire Alarm subsystem,
- Design of Temperature Control subsystem,
- Design of Light Control subsystem,.
- Budget analysis for Energy Saving, Safety and Isolation Subsystems.
- Involved in composure of the Phase 3 report.
- Involved in composure of the Phase 3 presentation slides.

Appendix B – Computer Application and Database Code

```
#pragmaonce
#include"SerialPort.h"
#include<iostream>
#include<windows.h>
#include<fstream>
#include<sstream>
#include<string>
#include<stdlib.h>
int microOutput1;

namespace finalDesign {

	usingnamespace System;
	usingnamespace System::ComponentModel;
	usingnamespace System::Collections;
	usingnamespace System::Windows::Forms;
	usingnamespace System::Data;
	usingnamespace System::Drawing;
	usingnamespace MySql::Data::MySqlClient;

	char output[MAX_DATA_LENGTH];
	char incomingData[MAX_DATA_LENGTH];


	/// <summary>
	/// Summary for MyForm
	/// </summary>
	publicrefclassMyForm : public System::Windows::Forms::Form
	{
	public:
		MyForm(void)
		{
			InitializeComponent();

		}

	protected:
		/// <summary>
		/// Clean up any resources being used.
		/// </summary>
		~MyForm()
		{
			if (components)
			{
				delete components;
			}
		}
	private: System::Windows::Forms::Button^  student;
	protected:

	private: System::Windows::Forms::Button^  lecturer;
	private: System::Windows::Forms::Button^  login_student;
	protected:


	private: System::Windows::Forms::Label^  Reg_student;
	private: System::Windows::Forms::Label^  Reg_lecturer;



	private: System::Windows::Forms::TextBox^  user_name;
	private: System::Windows::Forms::Label^  user_name_label;

	private: System::Windows::Forms::TextBox^  password;
	private: System::Windows::Forms::Label^  password_label;

	private: System::Windows::Forms::Label^  student_num;

	private: System::Windows::Forms::TextBox^  student_number;
```

```
        private: System::Windows::Forms::Label^  cell_num_label;

        private: System::Windows::Forms::TextBox^  cell_number;
        private: System::Windows::Forms::Label^  Dob_label;

        private: System::Windows::Forms::Label^  email_label;

        private: System::Windows::Forms::Label^  surname_label;

        private: System::Windows::Forms::Label^  name_label;

        private: System::Windows::Forms::TextBox^  dob;
        private: System::Windows::Forms::TextBox^  email;
        private: System::Windows::Forms::TextBox^  surname;
        private: System::Windows::Forms::TextBox^  name;
        private: System::Windows::Forms::Button^  student_save;

        private: System::Windows::Forms::TextBox^  Set_password_box;

        private: System::Windows::Forms::Label^  Set_password;
        private: System::Windows::Forms::Button^  save_lecturer;
        private: System::Windows::Forms::Label^  module_label;
        private: System::Windows::Forms::Label^  cell_num_label_lec;
        private: System::Windows::Forms::Label^  email_label_lec;
        private: System::Windows::Forms::Button^  login_lecturer;
        private: System::Windows::Forms::Button^  back_home;


        private: System::Windows::Forms::ProgressBar^  progressBar1;
        private: System::Windows::Forms::ComboBox^  comboBox1;
        private: System::Windows::Forms::TextBox^  textBox1;
        private: System::Windows::Forms::TextBox^  textBox2;
        private: System::Windows::Forms::TextBox^  textBox3;
        private: System::Windows::Forms::Label^  label1;
        private: System::Windows::Forms::Button^  button1;
        private: System::Windows::Forms::Button^  button2;
        private: System::Windows::Forms::Button^  button3;
        private: System::Windows::Forms::ListBox^  listBox1;
        private: System::Windows::Forms::Button^  button4;
        private: System::Windows::Forms::TextBox^  textBox4;
        private: System::Windows::Forms::Button^  button5;
        private: System::Windows::Forms::Label^  label2;
        private: System::Windows::Forms::ComboBox^  comboBox2;
        private: System::Windows::Forms::Label^  label3;



        private: System::ComponentModel::IContainer^  components;


        protected:

        private:
                /// <summary>
                /// Required designer variable.
                /// </summary>


#pragmaregion Windows Form Designer generated code
                /// <summary>
                /// Required method for Designer support - do not modify
                /// the contents of this method with the code editor.
                /// </summary>
                void InitializeComponent(void)
                {
                        this->student = (gcnew System::Windows::Forms::Button());
                        this->lecturer = (gcnew System::Windows::Forms::Button());
                        this->login_student = (gcnew System::Windows::Forms::Button());
                        this->Reg_student = (gcnew System::Windows::Forms::Label());
                        this->Reg_lecturer = (gcnew System::Windows::Forms::Label());
                        this->user_name = (gcnew System::Windows::Forms::TextBox());
                        this->user_name_label = (gcnew System::Windows::Forms::Label());
                        this->password = (gcnew System::Windows::Forms::TextBox());
```

```
                        this->password_label = (gcnew System::Windows::Forms::Label());
                        this->student_num = (gcnew System::Windows::Forms::Label());
                        this->student_number = (gcnew System::Windows::Forms::TextBox());
                        this->cell_num_label = (gcnew System::Windows::Forms::Label());
                        this->cell_number = (gcnew System::Windows::Forms::TextBox());
                        this->Dob_label = (gcnew System::Windows::Forms::Label());
                        this->email_label = (gcnew System::Windows::Forms::Label());
                        this->surname_label = (gcnew System::Windows::Forms::Label());
                        this->name_label = (gcnew System::Windows::Forms::Label());
                        this->dob = (gcnew System::Windows::Forms::TextBox());
                        this->email = (gcnew System::Windows::Forms::TextBox());
                        this->surname = (gcnew System::Windows::Forms::TextBox());
                        this->name = (gcnew System::Windows::Forms::TextBox());
                        this->student_save = (gcnew System::Windows::Forms::Button());
                        this->Set_password_box = (gcnew System::Windows::Forms::TextBox());
                        this->Set_password = (gcnew System::Windows::Forms::Label());
                        this->save_lecturer = (gcnew System::Windows::Forms::Button());
                        this->module_label = (gcnew System::Windows::Forms::Label());
                        this->cell_num_label_lec = (gcnew System::Windows::Forms::Label());
                        this->email_label_lec = (gcnew System::Windows::Forms::Label());
                        this->login_lecturer = (gcnew System::Windows::Forms::Button());
                        this->back_home = (gcnew System::Windows::Forms::Button());
                        this->progressBar1 = (gcnew System::Windows::Forms::ProgressBar());
                        this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());
                        this->textBox1 = (gcnew System::Windows::Forms::TextBox());
                        this->textBox2 = (gcnew System::Windows::Forms::TextBox());
                        this->textBox3 = (gcnew System::Windows::Forms::TextBox());
                        this->label1 = (gcnew System::Windows::Forms::Label());
                        this->button1 = (gcnew System::Windows::Forms::Button());
                        this->button2 = (gcnew System::Windows::Forms::Button());
                        this->button3 = (gcnew System::Windows::Forms::Button());
                        this->listBox1 = (gcnew System::Windows::Forms::ListBox());
                        this->button4 = (gcnew System::Windows::Forms::Button());
                        this->textBox4 = (gcnew System::Windows::Forms::TextBox());
                        this->button5 = (gcnew System::Windows::Forms::Button());
                        this->label2 = (gcnew System::Windows::Forms::Label());
                        this->comboBox2 = (gcnew System::Windows::Forms::ComboBox());
                        this->label3 = (gcnew System::Windows::Forms::Label());
                        this->SuspendLayout();
                        //
                        // student
                        //
                        this->student->Location = System::Drawing::Point(176, 35);
                        this->student->Name = L"student";
                        this->student->Size = System::Drawing::Size(75, 23);
                        this->student->TabIndex = 0;
                        this->student->Text = L"STUDENT";
                        this->student->UseVisualStyleBackColor = true;
                        this->student->Click += gcnew System::EventHandler(this,
&MyForm::student_Click);
                        //
                        // lecturer
                        //
                        this->lecturer->Location = System::Drawing::Point(176, 86);
                        this->lecturer->Name = L"lecturer";
                        this->lecturer->Size = System::Drawing::Size(75, 23);
                        this->lecturer->TabIndex = 1;
                        this->lecturer->Text = L"LECTURER";
                        this->lecturer->UseVisualStyleBackColor = true;
                        this->lecturer->Click += gcnew System::EventHandler(this,
&MyForm::lecturer_Click);
                        //
                        // login_student
                        //
                        this->login_student->Location = System::Drawing::Point(25, 233);
                        this->login_student->Name = L"login_student";
                        this->login_student->Size = System::Drawing::Size(75, 23);
                        this->login_student->TabIndex = 2;
                        this->login_student->Text = L"LOGIN";
                        this->login_student->UseVisualStyleBackColor = true;
                        this->login_student->Click += gcnew System::EventHandler(this,
&MyForm::button3_Click);
                        //
```

```
// Reg_student
//
this->Reg_student->AutoSize = true;
this->Reg_student->Location = System::Drawing::Point(12, 35);
this->Reg_student->Name = L"Reg_student";
this->Reg_student->Size = System::Drawing::Size(102, 13);
this->Reg_student->TabIndex = 3;
this->Reg_student->Text = L"regsiter as a student";
//
// Reg_lecturer
//
this->Reg_lecturer->AutoSize = true;
this->Reg_lecturer->Location = System::Drawing::Point(12, 91);
this->Reg_lecturer->Name = L"Reg_lecturer";
this->Reg_lecturer->Size = System::Drawing::Size(102, 13);
this->Reg_lecturer->TabIndex = 4;
this->Reg_lecturer->Text = L"register as a lecturer";
//
// user_name
//
this->user_name->Location = System::Drawing::Point(24, 145);
this->user_name->Name = L"user_name";
this->user_name->Size = System::Drawing::Size(100, 20);
this->user_name->TabIndex = 5;
//
// user_name_label
//
this->user_name_label->AutoSize = true;
this->user_name_label->Location = System::Drawing::Point(22, 129);
this->user_name_label->Name = L"user_name_label";
this->user_name_label->Size = System::Drawing::Size(59, 13);
this->user_name_label->TabIndex = 6;
this->user_name_label->Text = L"user name ";
//
// password
//
this->password->Location = System::Drawing::Point(25, 196);
this->password->Name = L"password";
this->password->Size = System::Drawing::Size(100, 20);
this->password->TabIndex = 7;
//
// password_label
//
this->password_label->AutoSize = true;
this->password_label->Location = System::Drawing::Point(22, 180);
this->password_label->Name = L"password_label";
this->password_label->Size = System::Drawing::Size(52, 13);
this->password_label->TabIndex = 8;
this->password_label->Text = L"password";
//
// student_num
//
this->student_num->AutoSize = true;
this->student_num->Location = System::Drawing::Point(8, 43);
this->student_num->Name = L"student_num";
this->student_num->Size = System::Drawing::Size(21, 13);
this->student_num->TabIndex = 38;
this->student_num->Text = L"ID ";
this->student_num->Visible = false;
//
// student_number
//
this->student_number->Location = System::Drawing::Point(77, 40);
this->student_number->Name = L"student_number";
this->student_number->Size = System::Drawing::Size(100, 20);
this->student_number->TabIndex = 37;
this->student_number->Visible = false;
//
// cell_num_label
//
this->cell_num_label->AutoSize = true;
this->cell_num_label->Location = System::Drawing::Point(4, 173);
this->cell_num_label->Name = L"cell_num_label";
```

```
                        this->cell_num_label->Size = System::Drawing::Size(64, 13);
                        this->cell_num_label->TabIndex = 36;
                        this->cell_num_label->Text = L"cell number ";
                        this->cell_num_label->Visible = false;
                        //
                        // cell_number
                        //
                        this->cell_number->Location = System::Drawing::Point(77, 170);
                        this->cell_number->Name = L"cell_number";
                        this->cell_number->Size = System::Drawing::Size(100, 20);
                        this->cell_number->TabIndex = 35;
                        this->cell_number->Visible = false;
                        //
                        // Dob_label
                        //
                        this->Dob_label->AutoSize = true;
                        this->Dob_label->Location = System::Drawing::Point(4, 144);
                        this->Dob_label->Name = L"Dob_label";
                        this->Dob_label->Size = System::Drawing::Size(63, 13);
                        this->Dob_label->TabIndex = 34;
                        this->Dob_label->Text = L"date of birth";
                        this->Dob_label->Visible = false;
                        //
                        // email_label
                        //
                        this->email_label->AutoSize = true;
                        this->email_label->Location = System::Drawing::Point(4, 121);
                        this->email_label->Name = L"email_label";
                        this->email_label->Size = System::Drawing::Size(31, 13);
                        this->email_label->TabIndex = 33;
                        this->email_label->Text = L"email";
                        this->email_label->Visible = false;
                        //
                        // surname_label
                        //
                        this->surname_label->AutoSize = true;
                        this->surname_label->Location = System::Drawing::Point(4, 92);
                        this->surname_label->Name = L"surname_label";
                        this->surname_label->Size = System::Drawing::Size(49, 13);
                        this->surname_label->TabIndex = 32;
                        this->surname_label->Text = L"Surname";
                        this->surname_label->Visible = false;
                        //
                        // name_label
                        //
                        this->name_label->AutoSize = true;
                        this->name_label->Location = System::Drawing::Point(4, 69);
                        this->name_label->Name = L"name_label";
                        this->name_label->Size = System::Drawing::Size(38, 13);
                        this->name_label->TabIndex = 31;
                        this->name_label->Text = L"Name ";
                        this->name_label->Visible = false;
                        //
                        // dob
                        //
                        this->dob->Location = System::Drawing::Point(77, 144);
                        this->dob->Name = L"dob";
                        this->dob->Size = System::Drawing::Size(100, 20);
                        this->dob->TabIndex = 30;
                        this->dob->Visible = false;
                        //
                        // email
                        //
                        this->email->Location = System::Drawing::Point(77, 118);
                        this->email->Name = L"email";
                        this->email->Size = System::Drawing::Size(100, 20);
                        this->email->TabIndex = 29;
                        this->email->Visible = false;
                        this->email->TextChanged += gcnew System::EventHandler(this,
&MyForm::email_TextChanged);
                        //
                        // surname
                        //
```

```
                        this->surname->Location = System::Drawing::Point(77, 92);
                        this->surname->Name = L"surname";
                        this->surname->Size = System::Drawing::Size(100, 20);
                        this->surname->TabIndex = 28;
                        this->surname->Visible = false;
                        //
                        // name
                        //
                        this->name->Location = System::Drawing::Point(77, 66);
                        this->name->Name = L"name";
                        this->name->Size = System::Drawing::Size(100, 20);
                        this->name->TabIndex = 27;
                        this->name->Visible = false;
                        //
                        // student_save
                        //
                        this->student_save->Location = System::Drawing::Point(92, 230);
                        this->student_save->Name = L"student_save";
                        this->student_save->Size = System::Drawing::Size(75, 23);
                        this->student_save->TabIndex = 26;
                        this->student_save->Text = L"SAVE";
                        this->student_save->UseVisualStyleBackColor = true;
                        this->student_save->Visible = false;
                        this->student_save->Click += gcnew System::EventHandler(this,
&MyForm::button1_Click_1);
                        //
                        // Set_password_box
                        //
                        this->Set_password_box->Location = System::Drawing::Point(77, 196);
                        this->Set_password_box->Name = L"Set_password_box";
                        this->Set_password_box->Size = System::Drawing::Size(100, 20);
                        this->Set_password_box->TabIndex = 39;
                        this->Set_password_box->Visible = false;
                        //
                        // Set_password
                        //
                        this->Set_password->AutoSize = true;
                        this->Set_password->Location = System::Drawing::Point(7, 199);
                        this->Set_password->Name = L"Set_password";
                        this->Set_password->Size = System::Drawing::Size(71, 13);
                        this->Set_password->TabIndex = 40;
                        this->Set_password->Text = L"Set password";
                        this->Set_password->Visible = false;
                        this->Set_password->Click += gcnew System::EventHandler(this,
&MyForm::Set_password_Click);
                        //
                        // save_lecturer
                        //
                        this->save_lecturer->Location = System::Drawing::Point(77, 262);
                        this->save_lecturer->Name = L"save_lecturer";
                        this->save_lecturer->Size = System::Drawing::Size(75, 23);
                        this->save_lecturer->TabIndex = 41;
                        this->save_lecturer->Text = L"SAVE";
                        this->save_lecturer->UseVisualStyleBackColor = true;
                        this->save_lecturer->Visible = false;
                        this->save_lecturer->Click += gcnew System::EventHandler(this,
&MyForm::save_lecturer_Click);
                        //
                        // module_label
                        //
                        this->module_label->AutoSize = true;
                        this->module_label->Location = System::Drawing::Point(7, 118);
                        this->module_label->Name = L"module_label";
                        this->module_label->Size = System::Drawing::Size(49, 13);
                        this->module_label->TabIndex = 42;
                        this->module_label->Text = L"modules ";
                        this->module_label->TextAlign =
System::Drawing::ContentAlignment::MiddleCenter;
                        this->module_label->Visible = false;
                        //
                        // cell_num_label_lec
                        //
                        this->cell_num_label_lec->AutoSize = true;
```

```
                        this->cell_num_label_lec->Location = System::Drawing::Point(7, 147);
                        this->cell_num_label_lec->Name = L"cell_num_label_lec";
                        this->cell_num_label_lec->Size = System::Drawing::Size(64, 13);
                        this->cell_num_label_lec->TabIndex = 43;
                        this->cell_num_label_lec->Text = L"cell number ";
                        this->cell_num_label_lec->Visible = false;
                        //
                        // email_label_lec
                        //
                        this->email_label_lec->AutoSize = true;
                        this->email_label_lec->Location = System::Drawing::Point(8, 174);
                        this->email_label_lec->Name = L"email_label_lec";
                        this->email_label_lec->Size = System::Drawing::Size(31, 13);
                        this->email_label_lec->TabIndex = 44;
                        this->email_label_lec->Text = L"email";
                        this->email_label_lec->Visible = false;
                        //
                        // login_lecturer
                        //
                        this->login_lecturer->Location = System::Drawing::Point(24, 233);
                        this->login_lecturer->Name = L"login_lecturer";
                        this->login_lecturer->Size = System::Drawing::Size(75, 23);
                        this->login_lecturer->TabIndex = 45;
                        this->login_lecturer->Text = L"LOGIN";
                        this->login_lecturer->UseVisualStyleBackColor = true;
                        this->login_lecturer->Click += gcnew System::EventHandler(this,
&MyForm::login_lecturer_Click);
                        //
                        // back_home
                        //
                        this->back_home->Location = System::Drawing::Point(7, 230);
                        this->back_home->Name = L"back_home";
                        this->back_home->Size = System::Drawing::Size(75, 23);
                        this->back_home->TabIndex = 46;
                        this->back_home->Text = L"HOME ";
                        this->back_home->UseVisualStyleBackColor = true;
                        this->back_home->Visible = false;
                        this->back_home->Click += gcnew System::EventHandler(this,
&MyForm::back_home_Click);
                        //
                        // progressBar1
                        //
                        this->progressBar1->Location = System::Drawing::Point(43, 10);
                        this->progressBar1->Name = L"progressBar1";
                        this->progressBar1->Size = System::Drawing::Size(208, 23);
                        this->progressBar1->TabIndex = 50;
                        //
                        // comboBox1
                        //
                        this->comboBox1->FormattingEnabled = true;
                        this->comboBox1->Location = System::Drawing::Point(267, 40);
                        this->comboBox1->Name = L"comboBox1";
                        this->comboBox1->Size = System::Drawing::Size(121, 21);
                        this->comboBox1->TabIndex = 51;
                        this->comboBox1->Visible = false;
                        //
                        // textBox1
                        //
                        this->textBox1->Location = System::Drawing::Point(267, 67);
                        this->textBox1->Name = L"textBox1";
                        this->textBox1->Size = System::Drawing::Size(117, 20);
                        this->textBox1->TabIndex = 52;
                        this->textBox1->Visible = false;
                        //
                        // textBox2
                        //
                        this->textBox2->Location = System::Drawing::Point(267, 91);
                        this->textBox2->Name = L"textBox2";
                        this->textBox2->Size = System::Drawing::Size(117, 20);
                        this->textBox2->TabIndex = 53;
                        this->textBox2->Visible = false;
                        //
                        // textBox3
```

```
                        //
                        this->textBox3->Location = System::Drawing::Point(267, 113);
                        this->textBox3->Name = L"textBox3";
                        this->textBox3->Size = System::Drawing::Size(117, 20);
                        this->textBox3->TabIndex = 54;
                        this->textBox3->Visible = false;
                        //
                        // label1
                        //
                        this->label1->AutoSize = true;
                        this->label1->Location = System::Drawing::Point(266, 20);
                        this->label1->Name = L"label1";
                        this->label1->Size = System::Drawing::Size(162, 13);
                        this->label1->TabIndex = 55;
                        this->label1->Text = L"Only Register for Three Modules ";
                        this->label1->Visible = false;
                        //
                        // button1
                        //
                        this->button1->Location = System::Drawing::Point(394, 67);
                        this->button1->Name = L"button1";
                        this->button1->Size = System::Drawing::Size(165, 23);
                        this->button1->TabIndex = 56;
                        this->button1->Text = L"ADD MODULE 1 ";
                        this->button1->UseVisualStyleBackColor = true;
                        this->button1->Visible = false;
                        this->button1->Click += gcnew System::EventHandler(this,
&MyForm::button1_Click_4);
                        //
                        // button2
                        //
                        this->button2->Location = System::Drawing::Point(394, 91);
                        this->button2->Name = L"button2";
                        this->button2->Size = System::Drawing::Size(165, 23);
                        this->button2->TabIndex = 57;
                        this->button2->Text = L"ADD MODULE 2";
                        this->button2->UseVisualStyleBackColor = true;
                        this->button2->Visible = false;
                        this->button2->Click += gcnew System::EventHandler(this,
&MyForm::button2_Click);
                        //
                        // button3
                        //
                        this->button3->Location = System::Drawing::Point(394, 113);
                        this->button3->Name = L"button3";
                        this->button3->Size = System::Drawing::Size(165, 23);
                        this->button3->TabIndex = 58;
                        this->button3->Text = L"ADD MODULE 3";
                        this->button3->UseVisualStyleBackColor = true;
                        this->button3->Visible = false;
                        this->button3->Click += gcnew System::EventHandler(this,
&MyForm::button3_Click_1);
                        //
                        // listBox1
                        //
                        this->listBox1->FormattingEnabled = true;
                        this->listBox1->Location = System::Drawing::Point(183, 139);
                        this->listBox1->Name = L"listBox1";
                        this->listBox1->Size = System::Drawing::Size(376, 186);
                        this->listBox1->TabIndex = 59;
                        this->listBox1->Visible = false;
                        //
                        // button4
                        //
                        this->button4->Location = System::Drawing::Point(394, 43);
                        this->button4->Name = L"button4";
                        this->button4->Size = System::Drawing::Size(165, 23);
                        this->button4->TabIndex = 60;
                        this->button4->Text = L"DISPLAY ATTENDENCE";
                        this->button4->UseVisualStyleBackColor = true;
                        this->button4->Visible = false;
                        this->button4->Click += gcnew System::EventHandler(this,
&MyForm::button4_Click);
```

```
                        //
                        // textBox4
                        //
                        this->textBox4->Location = System::Drawing::Point(35, 84);
                        this->textBox4->Name = L"textBox4";
                        this->textBox4->Size = System::Drawing::Size(142, 20);
                        this->textBox4->TabIndex = 61;
                        this->textBox4->Visible = false;
                        //
                        // button5
                        //
                        this->button5->Location = System::Drawing::Point(35, 113);
                        this->button5->Name = L"button5";
                        this->button5->Size = System::Drawing::Size(142, 64);
                        this->button5->TabIndex = 62;
                        this->button5->Text = L"SELECT EXAM AND ALLOW STUDENTS TO ENTER ";
                        this->button5->UseVisualStyleBackColor = true;
                        this->button5->Visible = false;
                        this->button5->Click += gcnew System::EventHandler(this,
&MyForm::button5_Click);
                        //
                        // label2
                        //
                        this->label2->AutoSize = true;
                        this->label2->Location = System::Drawing::Point(32, 40);
                        this->label2->Name = L"label2";
                        this->label2->Size = System::Drawing::Size(181, 13);
                        this->label2->TabIndex = 63;
                        this->label2->Text = L"SELECT CURRENT EXAMINATION";
                        this->label2->Visible = false;
                        //
                        // comboBox2
                        //
                        this->comboBox2->FormattingEnabled = true;
                        this->comboBox2->Location = System::Drawing::Point(35, 61);
                        this->comboBox2->Name = L"comboBox2";
                        this->comboBox2->Size = System::Drawing::Size(142, 21);
                        this->comboBox2->TabIndex = 64;
                        this->comboBox2->Visible = false;
                        //
                        // label3
                        //
                        this->label3->AutoSize = true;
                        this->label3->BackColor = System::Drawing::SystemColors::Menu;
                        this->label3->Font = (gcnew System::Drawing::Font(L"Monotype Corsiva", 21.75F,
static_cast<System::Drawing::FontStyle>((System::Drawing::FontStyle::Bold |
System::Drawing::FontStyle::Italic)),
                                System::Drawing::GraphicsUnit::Point, static_cast<System::Byte>(0)));
                        this->label3->ForeColor = System::Drawing::SystemColors::MenuHighlight;
                        this->label3->Location = System::Drawing::Point(-8, 328);
                        this->label3->Name = L"label3";
                        this->label3->Size = System::Drawing::Size(396, 36);
                        this->label3->TabIndex = 65;
                        this->label3->Text = L"SMART EXAMINATION HALL";
                        //
                        // MyForm
                        //
                        this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);
                        this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
                        this->AutoSize = true;
                        this->AutoSizeMode = System::Windows::Forms::AutoSizeMode::GrowAndShrink;
                        this->BackColor = System::Drawing::SystemColors::ActiveCaption;
                        this->ClientSize = System::Drawing::Size(675, 493);
                        this->Controls->Add(this->label3);
                        this->Controls->Add(this->comboBox2);
                        this->Controls->Add(this->label2);
                        this->Controls->Add(this->button5);
                        this->Controls->Add(this->textBox4);
                        this->Controls->Add(this->button4);
                        this->Controls->Add(this->listBox1);
                        this->Controls->Add(this->button3);
                        this->Controls->Add(this->button2);
                        this->Controls->Add(this->button1);
```

```
                        this->Controls->Add(this->label1);
                        this->Controls->Add(this->textBox3);
                        this->Controls->Add(this->textBox2);
                        this->Controls->Add(this->textBox1);
                        this->Controls->Add(this->comboBox1);
                        this->Controls->Add(this->progressBar1);
                        this->Controls->Add(this->back_home);
                        this->Controls->Add(this->login_lecturer);
                        this->Controls->Add(this->email_label_lec);
                        this->Controls->Add(this->cell_num_label_lec);
                        this->Controls->Add(this->module_label);
                        this->Controls->Add(this->save_lecturer);
                        this->Controls->Add(this->Set_password);
                        this->Controls->Add(this->Set_password_box);
                        this->Controls->Add(this->student_num);
                        this->Controls->Add(this->student_number);
                        this->Controls->Add(this->cell_num_label);
                        this->Controls->Add(this->cell_number);
                        this->Controls->Add(this->Dob_label);
                        this->Controls->Add(this->email_label);
                        this->Controls->Add(this->surname_label);
                        this->Controls->Add(this->name_label);
                        this->Controls->Add(this->dob);
                        this->Controls->Add(this->email);
                        this->Controls->Add(this->surname);
                        this->Controls->Add(this->name);
                        this->Controls->Add(this->student_save);
                        this->Controls->Add(this->password_label);
                        this->Controls->Add(this->password);
                        this->Controls->Add(this->user_name_label);
                        this->Controls->Add(this->user_name);
                        this->Controls->Add(this->Reg_lecturer);
                        this->Controls->Add(this->Reg_student);
                        this->Controls->Add(this->login_student);
                        this->Controls->Add(this->lecturer);
                        this->Controls->Add(this->student);
                        this->Name = L"MyForm";
                        this->StartPosition = System::Windows::Forms::FormStartPosition::CenterScreen;
                        this->Text = L"SMART EXAMINATION HALL";
                        this->ResumeLayout(false);
                        this->PerformLayout();

                }
#pragmaendregion
        private: System::Void button1_Click(System::Object^  sender, System::EventArgs^  e) {
                //MessageBox::Show("hellos");


        }
        private: System::Void label1_Click(System::Object^  sender, System::EventArgs^  e) {
        }
private: System::Void button3_Click(System::Object^  sender, System::EventArgs^  e) {

        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select* from design3.student where idstudent =
'" + this->user_name->Text + "' and password = '" + this->password->Text + "';", conDataBase);
        MySqlDataReader^myReader;
        try{
                conDataBase->Open();
                myReader = cmdDataBase->ExecuteReader();
                int count = 0;
                while (myReader->Read())
                {
                        count = count + 1;
                        }
                if (count == 1){
                        MessageBox::Show("password and username correct");
                }
                elseif (count>1)
                        MessageBox::Show("password and username dup");
                else
                        MessageBox::Show("password and username incorrect");
```

```
        }
        catch (Exception^ex){
                MessageBox::Show(ex->Message);
        }
}
private: System::Void student_Click(System::Object^  sender, System::EventArgs^  e) {
        /*show the following fields*/
        this->student_num->Visible = true;
        this->student_number->Visible = true;
        this->email->Visible = true;
        this->email_label->Visible = true;
        this->name_label->Visible = true;
        this->name->Visible = true;
        this->Dob_label->Visible = true;
        this->dob->Visible = true;
        this->surname->Visible = true;
        this->surname_label->Visible = true;
        this->cell_number->Visible = true;
        this->cell_num_label->Visible = true;
        this->student_save->Visible = true;
        this->Set_password->Visible = true;
        this->Set_password_box->Visible = true;
        this->back_home->Visible = true;
        this->comboBox1->Visible = true;
        this->textBox1->Visible = true;
        this->textBox2->Visible = true;
        this->textBox3->Visible = true;
        this->button1->Visible = true;
        this->button2->Visible = true;
        this->button3->Visible = true;
        this->label1->Visible = true;

        hide();
    load_modules();
}
private: System::Void lecturer_Click(System::Object^  sender, System::EventArgs^  e) {
        //login.ShowDialog();
        /*show the following fields*/
        this->student_num->Visible = true;
        this->student_number->Visible = true;
        this->email->Visible = true;
        this->email_label->Visible = true;
        this->name_label->Visible = true;
        this->name->Visible = true;
        this->Dob_label->Visible = true;
        this->dob->Visible = true;
        this->surname->Visible = true;
        this->surname_label->Visible = true;
        this->cell_number->Visible = true;
        this->cell_num_label->Visible = true;
        this->save_lecturer->Visible = true;
        this->Set_password->Visible = true;
        this->Set_password_box->Visible = true;
        this->back_home->Visible = true;


        /*hide the following fields*/
        this->Reg_lecturer->Visible = false;
        this->lecturer->Visible = false;
        this->Reg_student->Visible = false;
        this->student->Visible = false;
        this->user_name->Visible = false;
        this->user_name_label->Visible = false;
        this->password->Visible = false;
        this->password_label->Visible = false;
        this->login_student->Visible = false;
        this->login_lecturer->Visible = false;

}
private: System::Void button1_Click_1(System::Object^  sender, System::EventArgs^  e) {

        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
```

```
        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into design3.student
(idstudent,name,surname,email,date_of_birth,cell_num,password,module1,module2,module3) values('" +
this->student_number->Text + "','" + this->name->Text + "','" + this->surname->Text + "','" + this-
>email->Text + "','" + this->dob->Text + "','" + this->cell_number->Text + "','" + this-
>Set_password_box->Text + "','" + this->textBox1->Text + "','" + this->textBox2->Text + "','" + this-
>textBox3->Text + "')  ;", conDataBase);
        //MySqlCommand^ cmdDataBase1 = gcnew MySqlCommand("insert into design3.student
(idstudent,name,surname,email,date_of_birth,cell_num,password) values('" + this->student_number->Text
+ "','" + this->name->Text + "','" + this->surname->Text + "','" + this->email->Text + "','" + this-
>dob->Text + "','" + this->cell_number->Text + "','" + this->Set_password_box->Text + "');",
conDataBase1);

        MySqlDataReader^myReader;
        try{
                conDataBase->Open();
                myReader = cmdDataBase->ExecuteReader();
                while (myReader->Read())
                {
                }
        }
        catch (Exception^ex){
                MessageBox::Show(ex->Message);
        }
        //load_modules();
}
private: System::Void email_TextChanged(System::Object^  sender, System::EventArgs^  e) {
}
private: System::Void Set_password_Click(System::Object^  sender, System::EventArgs^  e) {
}
private: System::Void save_lecturer_Click(System::Object^  sender, System::EventArgs^  e) {
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into design3.lecturer
(idlecturer,name,surname,modules,cell_number,email,password) values('" + this->student_number->Text +
"','" + this->name->Text + "','" + this->surname->Text + "','" + this->email->Text + "','" + this-
>dob->Text + "','" + this->cell_number->Text + "','" + this->Set_password_box->Text + "');",
conDataBase);
        MySqlDataReader^myReader;
        try{
                conDataBase->Open();
                myReader = cmdDataBase->ExecuteReader();
                while (myReader->Read())
                {
                        }
        }
        catch (Exception^ex){
                MessageBox::Show(ex->Message);
        }
}
private: System::Void button1_Click_2(System::Object^  sender, System::EventArgs^  e) {

}
private: System::Void login_lecturer_Click(System::Object^  sender, System::EventArgs^  e) {
        hide();
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select* from design3.lecturer where idlecturer
= '" + this->user_name->Text + "' and password = '" + this->password->Text + "';", conDataBase);
        MySqlDataReader^myReader;
        try{
                conDataBase->Open();
                myReader = cmdDataBase->ExecuteReader();
                int count = 0;
                while (myReader->Read())
                {
                        count = count + 1;

                }
                if (count == 1){
                        MessageBox::Show("password and username correct");
                        InitPortKeshav();
                                }
                elseif (count>1)
```

```
                        MessageBox::Show("password and username dup");
                else
                        MessageBox::Show("password and username incorrect");
        }
        catch (Exception^ex){
                MessageBox::Show(ex->Message);
        }
        this->comboBox1->Visible = true;
        this->textBox1->Visible = true;
        this->label1->Visible = true;
        this->button4->Visible = true;
        this->listBox1->Visible = true;
        this->button5->Visible = true;
        this->comboBox2->Visible = true;
        this->textBox4->Visible = true;
        this->label2->Visible = true;
        this->back_home->Visible = true;
        load_modules();


}
private: System::Void back_home_Click(System::Object^  sender, System::EventArgs^  e) {
        this->student_num->Visible = false;
        this->student_number->Visible = false;
        this->cell_num_label->Visible = false;
        this->cell_number->Visible = false;
        this->Dob_label->Visible = false;
        this->email_label->Visible = false;
        this->surname_label->Visible = false;
        this->name_label->Visible = false;
        this->dob->Visible = false;
        this->email->Visible = false;
        this->surname->Visible = false;
        this->name->Visible = false;
        this->student_save->Visible = false;
        this->Set_password_box->Visible = false;
        this->Set_password->Visible = false;
        this->save_lecturer->Visible = false;
        this->module_label->Visible = false;
        this->cell_num_label_lec->Visible = false;
        this->email_label_lec->Visible = false;
        this->back_home->Visible = false;
        this->comboBox1->Visible = false;
        this->textBox1->Visible = false;
        this->textBox2->Visible = false;
        this->textBox3->Visible = false;
        this->textBox4->Visible = false;
        this->button1->Visible = false;
        this->button2->Visible = false;
        this->button3->Visible = false;
        this->label1->Visible = false;
        this->comboBox2->Visible = false;
        this->listBox1->Visible = false;
        this->button4->Visible = false;
        this->label2->Visible = false;
        this->button5->Visible = false;


        InitializeComponent();
}

private: System::Void button1_Click_3(System::Object^  sender, System::EventArgs^  e) {
                readK();
        }
                private: void           authentication(void)
                  {
                        String^ constring = L" datasource =
localhost;port=3306;username=root;password=password@0105";
                        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
                        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select* from design3.lecturer
where idlecturer = '" + output[11] + "';", conDataBase);
                        MySqlDataReader^myReader;
                        int count = 0;
```

```
                        try{
                                  conDataBase->Open();
                                  myReader = cmdDataBase->ExecuteReader();
                                int count = 0;
                                while (myReader->Read())
                                  {
                                          count = count + 1;
                                                                        }
                                if (count == 1)
                                  {
                                                                                mark_register();

                                  }
                          }
                        catch (Exception^ex){
                                MessageBox::Show(ex->Message);
                          }
                 }
                                        private: void           InitPortKeshav(void)
                                        {
                                                constchar* port = "\\\\.\\COM6";
                                                SerialPort picMicro(port);
                                                if (picMicro.isConnected()) {
                                                        this->progressBar1->Value = 100;

        }
                                                else {

                                                        MessageBox::Show("error");
                                                }
                                        }
                                                        private: void readK(void)
                                                         {
                                                                constchar* port = "\\\\.\\COM6";

                                                                SerialPort picMicro1(port);
                                                                //output[MAX_DATA_LENGTH] = '\0';
                                                                while (picMicro1.isConnected()) {


picMicro1.readSerialPort(output, MAX_DATA_LENGTH);
                                                                                        microOutput1 =
strcmp(output, "49534848555754555550557");

        MessageBox::Show(Convert::ToString(output[11]));


                                                                                if (microOutput1 == 0)
                                                                                 {

        MessageBox::Show("show");
                                                                                                        }
                                                                        }

                 }
                private: void load_modules(void){
                        String^ constring = L" datasource =
localhost;port=3306;username=root;password=password@0105";
                        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
                        MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select* from design3.modules",
conDataBase);
                        MySqlDataReader^myReader;
                        try{
                                conDataBase->Open();
                                myReader = cmdDataBase->ExecuteReader();

                                while (myReader->Read())
                                {
                                        String^ Modules;
                                        Modules = myReader->GetString("module_code");
                                        this->comboBox1->Items->Add(Modules);
                                        this->comboBox2->Items->Add(Modules);
```

```
                                }

                        }
                        catch (Exception^ex){
                                MessageBox::Show(ex->Message);
                        }
                }

private: System::Void button1_Click_4(System::Object^  sender, System::EventArgs^  e) {

        String^ temp = comboBox1->Text;
        textBox1->Text = temp;
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);

        if (temp == "enel4ee")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4ee(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4aa")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4aa(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4co")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4co(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4df")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4df(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
```

```
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4gj")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4gj(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4cd")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4cd(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}
                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
                        }

private: System::Void button2_Click(System::Object^  sender, System::EventArgs^  e) {
        String^ temp = comboBox1->Text;
        textBox2->Text = temp;
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
        //MySqlCommand^ cmdDataBase = gcnew MySqlCommand("insert into design3.module_register
('"+temp+"') values('" + this->student_number->Text + "')  ;", conDataBase);
        //MySqlCommand^ cmdDataBase1 = gcnew MySqlCommand("insert into design3.student
(idstudent,name,surname,email,date_of_birth,cell_num,password) values('" + this->student_number->Text
+ "','" + this->name->Text + "','" + this->surname->Text + "','" + this->email->Text + "','" + this-
>dob->Text + "','" + this->cell_number->Text + "','" + this->Set_password_box->Text + "');",
conDataBase1);

        if (temp == "enel4ee")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4ee(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
```

```
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4aa")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4aa(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4co")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4co(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4df")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4df(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4gj")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4gj(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
```

```
                }
        }
        if (temp == "enel4cd")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4cd(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}
                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
}
private: System::Void button3_Click_1(System::Object^  sender, System::EventArgs^  e) {
        String^ temp = comboBox1->Text;
        textBox3->Text = temp;
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);

        if (temp == "enel4ee")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4ee(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4aa")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4aa(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4co")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4co(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
```

```
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4df")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4df(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {

                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4gj")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4gj(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4cd")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4cd(student_register) values('" + this->student_number->Text + "'); ", conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read()){}
                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
}
                private:void transmit(void){


                 }

private: System::Void button4_Click(System::Object^  sender, System::EventArgs^  e) {
        //load_modules();
        String^ temp = comboBox1->Text;
        textBox1->Text = temp;
        String^ constring = L" datasource = localhost;port=3306;username=root;password=password@0105";
        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
        /*MySqlCommand^ cmdDataBase = gcnew MySqlCommand("select* from design3.module_attendence ",
conDataBase);

        MySqlDataReader^myReader;
        try{
                conDataBase->Open();
                myReader = cmdDataBase->ExecuteReader();
```

```
                while (myReader->Read())
                {
                        String^ Modules;
                        Modules = myReader->GetString(temp);
                        this->listBox1->Items->Add(Modules);
                }

        }
        catch (Exception^ex){
                MessageBox::Show(ex->Message);
        }*/
        if (temp == "enel4ee")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4ee ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                                String^ Modules;
                                Modules = myReader->GetString("attendence");
                                String^ Module;
                                Module = myReader->GetString("student_register");
                                this->listBox1->Items->Add(Module +"               "+ Modules);
                                //this->listBox1->Items->Add(Modules);
                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4aa")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4aa ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                                String^ Modules;
                                Modules = myReader->GetString("attendence");
                                String^ Module;
                                Module = myReader->GetString("student_register");
                                this->listBox1->Items->Add(Module + "               " + Modules);
                        }

                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
        if (temp == "enel4co")
        {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4co ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                        conDataBase->Open();
                        myReader = cmdDataBase->ExecuteReader();

                        while (myReader->Read())
                        {
                                String^ Modules;
                                Modules = myReader->GetString("attendence");
                                String^ Module;
```

```
                        Module = myReader->GetString("student_register");
                        this->listBox1->Items->Add(Module + "               " + Modules);
                    }

                }
                catch (Exception^ex){
                    MessageBox::Show(ex->Message);
                }
            }
            if (temp == "enel4cd")
            {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4cd ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                    conDataBase->Open();
                    myReader = cmdDataBase->ExecuteReader();

                    while (myReader->Read())
                    {
                        String^ Modules;
                        Modules = myReader->GetString("attendence");
                        String^ Module;
                        Module = myReader->GetString("student_register");
                        this->listBox1->Items->Add(Module + "               " + Modules);
                    }

                }
                catch (Exception^ex){
                    MessageBox::Show(ex->Message);
                }
            }
            if (temp == "enel4gj")
            {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4gj ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                    conDataBase->Open();
                    myReader = cmdDataBase->ExecuteReader();

                    while (myReader->Read())
                    {
                        String^ Modules;
                        Modules = myReader->GetString("attendence");
                        String^ Module;
                        Module = myReader->GetString("student_register");
                        this->listBox1->Items->Add(Module + "               " + Modules);
                    }

                }
                catch (Exception^ex){
                    MessageBox::Show(ex->Message);
                }
            }
            if (temp == "enel4df")
            {
                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("select * from design3.enel4df ;",
conDataBase);
                MySqlDataReader^myReader;
                try{
                    conDataBase->Open();
                    myReader = cmdDataBase->ExecuteReader();

                    while (myReader->Read())
                    {
                        String^ Modules;
                        Modules = myReader->GetString("attendence");
                        String^ Module;
                        Module = myReader->GetString("student_register");
                        this->listBox1->Items->Add(Module + "               " + Modules);
                    }
```

```
                }
                catch (Exception^ex){
                        MessageBox::Show(ex->Message);
                }
        }
}
                /* private:void enel4co(void){
                         String^ constring = L" datasource =
localhost;port=3306;username=root;password=password@0105";
                         MySqlConnection^ conDataBase = gcnew MySqlConnection(constring);
                         MySqlCommand^ cmdDataBase = gcnew MySqlCommand("select* from design3.enel4co
", conDataBase);
                         MySqlDataReader^myReader;
                         try{
                                 conDataBase->Open();
                                 myReader = cmdDataBase->ExecuteReader();

                                 while (myReader->Read())
                                 {
                                         String^ Modules;
                                         Modules = myReader-
>GetString("attendence"||"student_register");
                                         this->listBox1->Items->Add(Modules);
                                 }

                         }
                         catch (Exception^ex){
                                 MessageBox::Show(ex->Message);
                         }
                 }*/
                private:void mark_register(void){
                        String^ temp;
                         temp = this->textBox4->Text;
                        String^ constring = L" datasource =
localhost;port=3306;username=root;password=password@0105";
                        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
                        if (temp == "enel4ee")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='"+output[11]+"' WHERE `student_register`='"+output[11]+"';;",
conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                         conDataBase->Open();
                                         myReader = cmdDataBase->ExecuteReader();
                                        while (myReader->Read())
                                         {
                                                //textBox1->Text += (myReader->GetInt32(0));
                                         }
                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4aa")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='" + output[11] + "' WHERE `student_register`='" + output[11] +
"';;", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                         conDataBase->Open();
                                         myReader = cmdDataBase->ExecuteReader();
                                        while (myReader->Read())
                                         {
                                                //textBox1->Text += (myReader->GetInt32(0));
                                         }
                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4co")
```

```
                        {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='"+output[11]+"' WHERE `student_register`='"+output[11]+"';; ",
conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();
                                       while (myReader->Read())
                                        {
                                               //textBox1->Text += (myReader->GetInt32(0));
                                        }
                                 }
                                catch (Exception^ex){
                                       MessageBox::Show(ex->Message);
                                 }
                        }
                       if (temp == "enel4df")
                        {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='" + output[11] + "' WHERE `student_register`='" + output[11] +
"';;", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();
                                       while (myReader->Read())
                                        {
                                               //textBox1->Text += (myReader->GetInt32(0));
                                        }
                                 }
                                catch (Exception^ex){
                                       MessageBox::Show(ex->Message);
                                 }
                        }
                       if (temp == "enel4gj")
                        {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='" + output[11] + "' WHERE `student_register`='" + output[11] +
"';;", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();
                                       while (myReader->Read())
                                        {
                                               //textBox1->Text += (myReader->GetInt32(0));
                                        }
                                 }
                                catch (Exception^ex){
                                       MessageBox::Show(ex->Message);
                                 }
                        }
                       if (temp == "enel4cd")
                        {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("UPDATE
`design3`.`enel4ee` SET `attendence`='"+output[11]+"' WHERE `student_register`='"+output[11]+"';; ",
conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();
                                       while (myReader->Read())
                                        {
                                               //textBox1->Text += (myReader->GetInt32(0));
                                        }
                                 }
                                catch (Exception^ex){
                                       MessageBox::Show(ex->Message);
                                 }
                        }

                }
```

```
                                private: void hide(void){
                                        this->Reg_lecturer->Visible = false;
                                        this->lecturer->Visible = false;
                                        this->Reg_student->Visible = false;
                                        this->student->Visible = false;
                                        this->user_name->Visible = false;
                                        this->user_name_label->Visible = false;
                                        this->password->Visible = false;
                                        this->password_label->Visible = false;
                                        this->login_student->Visible = false;
                                        this->login_lecturer->Visible = false;
                                 }

private: System::Void button5_Click(System::Object^  sender, System::EventArgs^  e) {
        String^ temp = comboBox2->Text;
        textBox4->Text = temp;
        readK();
        authentication();

}
                private:void check_storage(void){
                        String^ temp = comboBox1->Text;
                         textBox1->Text = temp;
                        String^ constring = L" datasource =
localhost;port=3306;username=root;password=password@0105";
                        MySqlConnection^ conDataBase = gcnewMySqlConnection(constring);
                        //MySqlCommand^ cmdDataBase = gcnew MySqlCommand("insert into
design3.module_register ('"+temp+"') values('" + this->student_number->Text + "')  ;", conDataBase);
                        //MySqlCommand^ cmdDataBase1 = gcnew MySqlCommand("insert into design3.student
(idstudent,name,surname,email,date_of_birth,cell_num,password) values('" + this->student_number->Text
+ "','" + this->name->Text + "','" + this->surname->Text + "','" + this->email->Text + "','" + this-
>dob->Text + "','" + this->cell_number->Text + "','" + this->Set_password_box->Text + "');",
conDataBase1);

                        if (temp == "enel4ee")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4ee('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                         conDataBase->Open();
                                         myReader = cmdDataBase->ExecuteReader();

                                        while (myReader->Read()){}

                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4aa")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4aa('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                         conDataBase->Open();
                                         myReader = cmdDataBase->ExecuteReader();

                                        while (myReader->Read())
                                         {
                                         }

                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4co")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4co('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
```

```
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();

                                       while (myReader->Read())
                                        {

                                        }

                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4df")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4df('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();

                                       while (myReader->Read())
                                        {

                                        }

                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4gj")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4gj('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();

                                       while (myReader->Read()){}

                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                        if (temp == "enel4cd")
                         {
                                MySqlCommand^ cmdDataBase = gcnewMySqlCommand("insert into
design3.enel4cd('" + temp + "') values('" + this->student_number->Text + "'); ", conDataBase);
                                MySqlDataReader^myReader;
                                try{
                                        conDataBase->Open();
                                        myReader = cmdDataBase->ExecuteReader();

                                       while (myReader->Read()){}
                                 }
                                catch (Exception^ex){
                                        MessageBox::Show(ex->Message);
                                 }
                         }
                 }
private: System::Void button6_Click(System::Object^  sender, System::EventArgs^  e) {
        MessageBox::Show(".............");
        InitPortKeshav();
        MessageBox::Show(".............");
        readK();

}
```

```
private: System::Void button7_Click(System::Object^  sender, System::EventArgs^  e) {
        InitPortKeshav();
}
};
}

#include"SerialPort.h"

SerialPort::SerialPort(constchar *portName)
{
        this->connected = false;

        this->handler = CreateFileA(static_cast<LPCSTR>(portName),
                GENERIC_READ | GENERIC_WRITE,
                0,
                NULL,
                OPEN_EXISTING,
                FILE_ATTRIBUTE_NORMAL,
                NULL);
        if (this->handler == INVALID_HANDLE_VALUE) {
                if (GetLastError() == ERROR_FILE_NOT_FOUND) {
                        printf("ERROR: Handle was not attached. Reason: %s not available\n",
portName);
                }
                else
                {
                        printf("ERROR!!!");
                }
        }
        else {
                DCB dcbSerialParameters = { 0 };

                if (!GetCommState(this->handler, &dcbSerialParameters)) {
                        printf("failed to get current serial parameters");
                }
                else {
                        dcbSerialParameters.BaudRate = CBR_9600;
                        dcbSerialParameters.ByteSize = 8;
                        dcbSerialParameters.StopBits = ONESTOPBIT;
                        dcbSerialParameters.Parity = NOPARITY;
                        dcbSerialParameters.fDtrControl = DTR_CONTROL_ENABLE;

                        if (!SetCommState(handler, &dcbSerialParameters))
                        {
                                printf("ALERT: could not set Serial port parameters\n");
                        }
                        else {
                                this->connected = true;
                                PurgeComm(this->handler, PURGE_RXCLEAR | PURGE_TXCLEAR);
                                Sleep(ARDUINO_WAIT_TIME);
                        }
                }
        }
}

SerialPort::~SerialPort()
{
        if (this->connected) {
                this->connected = false;
                CloseHandle(this->handler);
        }
}

intSerialPort::readSerialPort(char *buffer, unsignedintbuf_size)
{
        DWORD bytesRead;
        unsignedint toRead = 0;

        ClearCommError(this->handler, &this->errors, &this->status);

        if (this->status.cbInQue > 0) {
                if (this->status.cbInQue >buf_size) {
                        toRead = buf_size;
```

```
                }
                else toRead = this->status.cbInQue;
        }

        memset(buffer, 0, buf_size);

        if (ReadFile(this->handler, buffer, toRead, &bytesRead, NULL)) return bytesRead;

        return 0;
}

boolSerialPort::writeSerialPort(char *buffer, unsignedintbuf_size)
{
        DWORD bytesSend;

        if (!WriteFile(this->handler, (void*)buffer, buf_size, &bytesSend, 0)) {
                ClearCommError(this->handler, &this->errors, &this->status);
                returnfalse;
        }
        elsereturntrue;
}

boolSerialPort::isConnected()
{
        if (!ClearCommError(this->handler, &this->errors, &this->status))
                this->connected = false;

        returnthis->connected;
}

#ifndef SERIALPORT_H
#defineSERIALPORT_H

#defineARDUINO_WAIT_TIME 2000
#defineMAX_DATA_LENGTH 255

#include<windows.h>
#include<stdio.h>
#include<stdlib.h>

classSerialPort
{
private:
        HANDLE handler;
        bool connected;
        COMSTAT status;
        DWORD errors;
public:
        SerialPort(constchar *portName);
        ~SerialPort();

        int readSerialPort(char *buffer, unsignedint buf_size);
        bool writeSerialPort(char *buffer, unsignedint buf_size);
        bool isConnected();
};

#endif// SERIALPORT_H
```

Appendix C – Code for RFID Authentication Subsystem

```
//Keypad header file
#define row1port LATAbits.LATA1
#define row2port LATBbits.LATB15
#define row3port LATBbits.LATB14
#define row4port LATBbits.LATB1
#define col1port PORTBbits.RB0
#define col2port PORTAbits.RA0
#define col3port PORTAbits.RA2
```

```
#define Length 4
char const keyPadMatrix[] =
{
    '1','2','3',
    '4','5','6',
    '7','8','9',
    '*','0','#',
    ' '
};
char old_key;
void delay_Keypad()
{
    int dx,dy;
    for(dx=0;dx<40;dx++)
    {
        for(dy=0;dy<40;dy++)
        {
            Nop();
        }
    }
}
char Get_Keypad_Digit(){
    // This routine returns the first key found to be pressed during the scan.
    char key = 0, row=0;

    for( row = 0b00000001; row < 0b00010000; row <<= 1 )
    {
        {   // Turn ON Row Output
            row1port = (row & 0x0001)>>0;
            row2port = (row & 0x0002)>>1;
            row3port = (row & 0x0004)>>2;
            row4port = (row & 0x0008)>>3;
delay_Keypad();
        }
        // read colums - break when key press detected
        if( col1port )break;  key++;
        if( col2port )break;  key++;
        if( col3port )break;  key++;
        //if( col4port )break;  key++;
     }
    row1port = 0;
    row2port = 0;
    row3port = 0;
    row4port = 0;
    /*Ensures that 1 button press results in only 1 input*/
    if (key!=old_key){
      old_key=key;
      return keyPadMatrix[key];
    }
    else
    return keyPadMatrix[0x0C];
}

    }
    return;
}

void sendPassword(char password[5])
{
    char password[5]="";
    char key;
    LCDInit();
 while(1)
    {
        LCD_Clr();
        LCD_Print_String("Enter Password");
        LCD_L2();
        key=Get_Keypad_Digit();
        if (key!= ' ' && key!='#')
        {
```

```
            sprintf(password,"%s%c", password, key);
            LCD_L2();
            LCD_Print_String(password);
        }
         if(key=='*')
        {
        password[0]=0;
          LCDInit();
            LCD_Clr();
        }
         if(key=='#' && strlen(password)!=6)
        {
             password[0]=0;
          LCDInit();
            LCD_Clr();
        }
        if(key=='#' && strlen(password)==6)
        {
            LCDInit();
            LCD_Clr();
            return;
        }
    }
}
//LCD header file
#define En  LATBbits.LATB12                // LCD_Enable Pin
#define RS  LATBbits.LATB13                // LCD_RS
#define D4  LATBbits.LATB11                // LCD_D4
#define D5  LATBbits.LATB10                // LCD_D5
#define D6  LATBbits.LATB9                // LCD_D6
#define D7  LATBbits.LATB8              // LCD_D7

void LCD_L2()
{
 int var2=0x0C ; //move to 2nd row, first column
 LCDIns(var2) ;
}
void LCD_Clr()
{
 int var2= 0x08 ;
 LCDIns(var2) ;
}
void Pulse_e()
{
       En  = 1;         //EN
       En  = 0;         //EN
}
void Lcd_SetBit(char data_bit) //Based on the Hex value Set the Bits of the Data Lines
{
       if(data_bit& 1)
              D4 = 1;     //D4
       else
              D4 = 0;
       if(data_bit& 2)
              D5 = 1;     //D5
       else
              D5 = 0;

       if(data_bit& 4)
              D6 = 1;     //D6
       else
              D6 = 0;
       if(data_bit& 8)
              D7 = 1;         //D7
       else
              D7 = 0;
}
void delay()
{
    int dx,dy;
```

```
    for(dx=0;dx<40;dx++)
    {
        for(dy=0;dy<40;dy++)
        {
            Nop();
        }
    }
}
void LCDIns(int data_bits)
{
 int LCDTemp = data_bits;
 int temp;
 temp =  data_bits & 0x000F; // Most Significant Nibble first
Lcd_SetBit(temp);
       En  = 0;         //EN
       RS  = 0;         //RS
 Pulse_e() ;
 temp = LCDTemp&0x00F0; // Least Significant Nibble Second
 temp=temp/16;
 Lcd_SetBit(temp);
       En = 0;
       RS  = 0;
Pulse_e();
   delay();
}
void LCDInit()
{
  //Send the Reset Instruction
    D5 = 1;
    D4 = 1;
    D7 = 0;
    D6 = 0;
    En = 0;
    RS = 0;
    delay();
    Pulse_e();
    delay();
    Pulse_e();
    delay();
    Pulse_e();
    delay();

       D5  = 1;
       D7 = 0;
       D6 = 0;
    D4 = 0;
       En  = 0;
       RS = 0;

 Pulse_e() ;
   delay();
   int var2 = 0x082;
 LCDIns(var2) ;

 var2= 0x080 ; //Turn Off Display
 LCDIns(var2) ;
  var2=  0x010 ; //Clear Display RAM
 LCDIns(var2) ;
  var2=  0x060 ; //Set Cursor Movement
 LCDIns(var2) ;
  var2=  0x0C0; //Turn on Display/Cursor
 LCDIns(var2) ;
  LCD_L2() ; //Clear the LCD
}
void LCD_Print_Char(char data_bit)
{
 int LCDTemp=data_bit;
 int temp = LCDTemp&0x00F0;
 temp=temp/16;
Lcd_SetBit(temp);
```

```
       En  = 0;         //EN
       RS  = 1;

 Pulse_e();
 temp=LCDTemp&0x000F;
Lcd_SetBit(temp);
       En  = 0;         //EN
       RS  = 1;
 Pulse_e();
   delay();
   }

void LCD_Print_String(char *a)
{
       int i;
       for(i=0;a[i]!='\0';i++)
          LCD_Print_Char(a[i]);  //Split the string using pointers and call the Char function
}

void LCD_Print_Number(int number)
{
    if(number==0)
    {
        LCD_Print_Char('0');
        return;
    }
       int Digits[10];
       int length=0;
while(number)
{
        Digits[length]=number;
        length++;
    number /= 10;
}
int i;
for(i=length-1;i>=0;i--)
{
    LCD_Print_Char((Digits[i] % 10)+48);
}
}
// CONFIG
#pragma config FOSC = EXTRCCLK  // Oscillator Selection bits (RC oscillator: CLKOUT function on
RA4/OSC2/CLKOUT pin, RC on RA5/OSC1/CLKIN)
#pragma config WDTE = OFF       // Watchdog Timer Enable bit (WDT disabled and can be enabled
by SWDTEN bit of the WDTCON register)
#pragma config PWRTE = OFF      // Power-up Timer Enable bit (PWRT disabled)
#pragma config MCLRE = ON       // MCLR Pin Function Select bit (MCLR pin function is MCLR)
#pragma config CP = OFF           // Code Protection bit (Program memory code protection is
disabled)
#pragma config CPD = OFF         // Data Code Protection bit (Data memory code protection is
disabled)
#pragma config BOREN = OFF      // Brown-out Reset Selection bits (BOR disabled)
#pragma config IESO = OFF      // Internal External Switchover bit (Internal External Switchover
mode is disabled)
#pragma config FCMEN = ON       // Fail-Safe Clock Monitor Enabled bit (Fail-Safe Clock Monitor
is enabled)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

#include <xc.h>

void con()
{
    ANSEL = 0b00000000;       //AN7 and AN0 enabled
    ANSELH = 0b00000000;
    TRISC = 0b11001000;
    TRISA = 0b00000000;
    TRISB = 0b00010000;
 /*   TRISAbits.TRISA2 = 0;   //RC0 is a digital output
```

```
    TRISCbits.TRISC3 = 1;
    //LATBbits.LATB3 =1 ;
    //TRISA = 0;*/
}
```

Appendix D – Facial Recognition and Video Surveillance Source Code

```
/*
* FacialRecognition.h
* Author: Keshav Jeewanlall 213508238
*/

/*
* DISCLAMER: The Serial library used in this project is not created by me (Keshav Jeewanlall
2135082238), however,
* some modifications to the library code and the code written to use this library is written
by me.
*/

#ifndef FACIALRECOGNITIONHEAD_H
#defineFACIALRECOGNITIONHEAD_H

#pragmaonce
#include<opencv\cv.h>

usingnamespace cv;
usingnamespace std;

classFacialRecognition
{

private:
        Mat capturedImage;                                      //Stores the captured image
        vector<Mat> faces;                                      //Stores the images used for
comparisons
        vector<int> ids;                                        //Stores the IDs of the images

        int im_width;          //Gets width and height of the images used for comparisons
        int im_height;

        Mat original;          //Keeps a copy of the original image
        Mat greyImage;

        vector<Rect_<int>> facePositions;

        Rect face_i;                                            //Position of the ith face
        Mat getFace;                                            //Converts to greyscale
        Mat face_resized;
        //Resizes the detected face
        int foundID;                  //Gets prediction from face recognizer

public:
        FacialRecognition();
        ~FacialRecognition();

        void read_csv(conststring&filepath, vector<Mat>&images, vector<int>&labels);
        int detectFace();
};

#endif


/*
* FacialRecognition.cpp
* Author: Keshav Jeewanlall 213508238
*/

/*
* DISCLAMER: The Serial library used in this project is not created by me (Keshav Jeewanlall
2135082238), however,
* some modifications to the library code and the code written to use this library is written
by me.
*/
```

```
#include<opencv\cv.h>
#include<opencv\highgui.h>
#include<opencv2\core.hpp>
#include<opencv2\contrib\contrib.hpp>
#include<opencv2\highgui.hpp>
#include<opencv2\imgproc.hpp>
#include<opencv2\objdetect.hpp>
#include<iostream>
#include<fstream>
#include<sstream>
#include<string>
#include<stdlib.h>
#include"FacialRecognition.h"


intFacialRecognition::detectFace()
{

	read_csv("C:/Student_Images/csv_file.csv", faces, ids);		//Fills the faces
and ids vectors

	im_width = faces[0].cols;		//Gets width and height of the images used for
comparisons
	im_height = faces[0].rows;

	Ptr<FaceRecognizer> model = createFisherFaceRecognizer();
	model->train(faces, ids);			//Creates a face recognizer, trains it
with the images used for comparisons

	CascadeClassifier haar_cascade;		//Chooses haar-cascade that is used for
face detection
	haar_cascade.load("haarcascade_frontalface_default.xml");
	VideoCapture cap;						//Opens a video feed
	cap.open(0);

	if (!cap.isOpened())
	{
		cout <<"ERROR: Could not open camera.";
		return -1;
	}
	while (true)
	{
		cap >> capturedImage;					//Captures an image
and stores in the captruedImage

		original = capturedImage.clone();		//Keeps a copy of the original
image
		greyImage;
		cvtColor(capturedImage, greyImage, CV_BGR2GRAY);		//Converts the
image to greyscale

		haar_cascade.detectMultiScale(greyImage, facePositions);		//Gets the
positions of the deteced faces from the image

		for (int i = 0; i < facePositions.size(); i++)
		{
			face_i = facePositions[i];
	//Position of the ith face
			getFace = greyImage(face_i);
	//Converts to greyscale
			face_resized;
		//Resizes the detected face
			cv::resize(getFace, face_resized, cv::Size(im_width, im_height), 1.0,
1.0, INTER_CUBIC);
			foundID = model->predict(face_resized);			//Gets
predicition from face recognizer


			rectangle(original, face_i, CV_RGB(0, 255, 0), 1);		//Places
rectangle over detected face and ID of the recognized faces
			string box_text = format("ID = %d", foundID);
			int pos_x = max(face_i.tl().x - 10, 0);
			int pos_y = max(face_i.tl().y - 10, 0);
			putText(original, box_text, cv::Point(pos_x, pos_y),
FONT_HERSHEY_PLAIN, 1.0, CV_RGB(0, 255, 0), 2.0);
```

```
		}


		imshow("face_recognizer", original);
	//Shows original image with recognized faces and their IDs

		//Wait for for 30 milliseconds until any key is pressed.
		//If the 'Esc' key is pressed (ASCII value of "Esc" is 27), break the while
loop.
		//If any other key is pressed, continue the loop
		//If any key is not pressed within 30 milliseconds, continue the loop
		if (waitKey(30) == 27)
		{
			return 0;
		}
	}

}


FacialRecognition::FacialRecognition()
{
}

FacialRecognition::~FacialRecognition()
{
}

voidFacialRecognition::read_csv(conststring&filepath, vector<Mat>&images, vector<int>&labels)
{
	string line, path, classlabel;
	std::ifstream file(filepath.c_str(), ifstream::in);
	if (!file)
	{
		cout <<"ERROR: Could not read CSV file.";
	}

	while (getline(file, line))                                    //Loop runs for every line
of the CSV file
	{
		stringstream liness(line);
		getline(liness, path, ';');                            //Delimiter seperates data
		getline(liness, classlabel);
		if (!path.empty() && !classlabel.empty())
		{
			images.push_back(imread(path, 0));                              //Stores
image in vector
			labels.push_back(atoi(classlabel.c_str()));  //Stores ID in vector
		}
	}
}



/*
* VideoRecorder.h
* Author: Keshav Jeewanlall 213508238
*/

/*
* DISCLAMER: The Serial library used in this project is not created by me (Keshav Jeewanlall
2135082238), however,
* some modifications to the library code and the code written to use this library is written
by me.
*/
#ifndef VIDEORECORDERHEAD_H
#defineVIDEORECORDERHEAD_H

#pragmaonce
#include<windows.h>
#include<opencv\cv.h>

usingnamespace cv;
usingnamespace std;

classVideoRecorder
```

```
{
private:

	int frame_width;
	int frame_height;
	int frames_per_second;
	int fcc;

	Mat frame;

public:
	VideoRecorder();
	~VideoRecorder();

	int videoRecord();
};

#endif

/*
* VideoRecorder.cpp
* Author: Keshav Jeewanlall 213508238
*/

/*
* DISCLAMER: The Serial library used in this project is not created by me (Keshav Jeewanlall
2135082238), however,
* some modifications to the library code and the code written to use this library is written
by me.
*/

#include<opencv\cv.h>
#include<opencv\highgui.h>
#include<opencv2\core.hpp>
#include<opencv2\contrib\contrib.hpp>
#include<opencv2\highgui.hpp>
#include<opencv2\imgproc.hpp>
#include<opencv2\objdetect.hpp>
#include<iostream>
#include<fstream>
#include<sstream>
#include<string>
#include<stdlib.h>
#include"VideoRecorder.h"

VideoRecorder::VideoRecorder()
{
}

VideoRecorder::~VideoRecorder()
{
}

intVideoRecorder::videoRecord()
{
	//Open the default video camera
	VideoCapture cap(0);

	// if not success, exit program
	if (cap.isOpened() == false)
	{
		cout <<"Cannot open the video camera"<< endl;
		cin.get(); //wait for any key press
		return -1;
	}

	frame_width = static_cast<int>(cap.get(CV_CAP_PROP_FRAME_WIDTH)); //get the width of
frames of the video
	frame_height = static_cast<int>(cap.get(CV_CAP_PROP_FRAME_HEIGHT)); //get the height
of frames of the video

	Size frame_size(frame_width, frame_height);
	frames_per_second = 10;
	fcc = CV_FOURCC('M', 'J', 'P', 'G');

	//Create and initialize the VideoWriter object
```

```
	VideoWriter writeVideo("D:/MyVideo.avi", fcc,
		frames_per_second, frame_size, true);

	//If the VideoWriter object is not initialized successfully, exit the program
	if (writeVideo.isOpened() == false)
	{
		cout <<"Cannot save the video to a file"<< endl;
		cin.get(); //wait for any key press
		return -1;
	}

	string window_name = "Security Camera Feed";
	namedWindow(window_name); //create a window called "My Camera Feed"

	while (true)
	{
		frame;
		bool isSuccess = cap.read(frame); // read a new frame from the video camera

		//Breaking the while loop if frames cannot be read from the camera
		if (isSuccess == false)
		{
			cout <<"Video camera is disconnected"<< endl;
			cin.get(); //Wait for any key press
			break;
		}

		//write the video frame to the file
		writeVideo.write(frame);

		//show the frame in the created window
		imshow(window_name, frame);

		//Wait for for 30 milliseconds until any key is pressed.
		//If the 'Esc' key is pressed (ASCII value of "Esc" is 27), break the while
loop.
		//If any other key is pressed, continue the loop
		//If any key is not pressed within 30 milliseconds, continue the loop
		if (waitKey(10) == 27)
		{
			cout <<"Esc key is pressed by the user. Stopping the video"<< endl;
			break;
		}
	}

	//Flush and close the video file
	writeVideo.release();

	return 0;
}


/*
* SerialPort.h
* Author: Manash Kumar Mandal
* Modified Library introduced in Arduino Playground which does not work
* This works perfectly
* LICENSE: MIT
*/

/*
* DISCLAMER: This Serial library is not created by me (Keshav Jeewanlall 2135082238), however,
* some modifications to this library code and the code written to use this library is written
by me.
* This is an open-source library and code.
* The unmodified library and code can be found on at https://github.com/manashmndl/SerialPort
*/


#ifndef SERIALPORT_H
#defineSERIALPORT_H

#defineARDUINO_WAIT_TIME 2000
#defineMAX_DATA_LENGTH 255

#include<windows.h>
```

```
#include<stdio.h>
#include<stdlib.h>

classSerialPort
{
private:
	HANDLE handler;
	bool connected;
	COMSTAT status;
	DWORD errors;
public:
	SerialPort(constchar *portName);
	~SerialPort();

	int readSerialPort(char *buffer, unsignedintbuf_size);
	bool writeSerialPort(char *buffer, unsignedintbuf_size);
	bool isConnected();
};

#endif// SERIALPORT_H

/*
* SerialPort.cpp
* Author: Manash Kumar Mandal
* Modified Library introduced in Arduino Playground which does not work
* This works perfectly
* LICENSE: MIT
*/

/*
* DISCLAMER: This Serial library is not created by me (Keshav Jeewanlall 2135082238), however,
* some modifications to this library code and the code written to use this library is written
by me.
* This is an open-source library and code.
* The unmodified library and code can be found on at https://github.com/manashmndl/SerialPort
*/

#include"SerialPort.h"

SerialPort::SerialPort(constchar *portName)
{
	this->connected = false;

	this->handler = CreateFileA(static_cast<LPCSTR>(portName),
		GENERIC_READ | GENERIC_WRITE,
		0,
		NULL,
		OPEN_EXISTING,
		FILE_ATTRIBUTE_NORMAL,
		NULL);
	if (this->handler == INVALID_HANDLE_VALUE) {
		if (GetLastError() == ERROR_FILE_NOT_FOUND) {
			printf("ERROR: Handle was not attached. Reason: %s not available\n",
portName);
		}
		else
		{
			printf("ERROR!!!");
		}
	}
	else {
		DCB dcbSerialParameters = { 0 };

		if (!GetCommState(this->handler, &dcbSerialParameters)) {
			printf("failed to get current serial parameters");
		}
		else {
			dcbSerialParameters.BaudRate = CBR_9600;
			dcbSerialParameters.ByteSize = 8;
			dcbSerialParameters.StopBits = ONESTOPBIT;
			dcbSerialParameters.Parity = NOPARITY;
			dcbSerialParameters.fDtrControl = DTR_CONTROL_ENABLE;

			if (!SetCommState(handler, &dcbSerialParameters))
			{
				printf("ALERT: could not set Serial port parameters\n");
```

```
                        }
                        else {
                                this->connected = true;
                                PurgeComm(this->handler, PURGE_RXCLEAR | PURGE_TXCLEAR);
                                Sleep(ARDUINO_WAIT_TIME);
                        }
                }
        }
}

SerialPort::~SerialPort()
{
        if (this->connected) {
                this->connected = false;
                CloseHandle(this->handler);
        }
}

intSerialPort::readSerialPort(char *buffer, unsignedintbuf_size)
{
        DWORD bytesRead;
        unsignedint toRead = 0;

        ClearCommError(this->handler, &this->errors, &this->status);

        if (this->status.cbInQue > 0) {
                if (this->status.cbInQue >buf_size) {
                        toRead = buf_size;
                }
                else toRead = this->status.cbInQue;
        }

        memset(buffer, 0, buf_size);

        if (ReadFile(this->handler, buffer, toRead, &bytesRead, NULL)) return bytesRead;

        return 0;
}

boolSerialPort::writeSerialPort(char *buffer, unsignedintbuf_size)
{
        DWORD bytesSend;

        if (!WriteFile(this->handler, (void*)buffer, buf_size, &bytesSend, 0)) {
                ClearCommError(this->handler, &this->errors, &this->status);
                returnfalse;
        }
        elsereturntrue;
}

boolSerialPort::isConnected()
{
        if (!ClearCommError(this->handler, &this->errors, &this->status))
                this->connected = false;

        returnthis->connected;
}


/*
* main.cpp
* Author: Keshav Jeewanlall 213508238
*/

/*
* DISCLAMER: The Serial library used in this project is not created by me (Keshav Jeewanlall
2135082238), however,
* some modifications to the library code and the code written to use this library is written
by me.
*/

#include<stdlib.h>
#include"SerialPort.h"
#include"FacialRecognition.h"
#include"VideoRecorder.h"
```

```
char output[MAX_DATA_LENGTH];
char incomingData[MAX_DATA_LENGTH];

// change the name of the port with the port name of your computer
// must remember that the backslashes are essential so do not remove the

constchar* port = "\\\\.\\COM1";
int microOutput1;
int microOutput2;



int main()
{
	FacialRecognition recognise;
	VideoRecorder record;
	SerialPort picMicro(port);

	if (picMicro.isConnected())
	{
		cout <<"Connection made"<< endl << endl;
	}
	else
	{
		cout <<"Error in port name"<< endl << endl;
	}

	while (picMicro.isConnected()) {

		picMicro.readSerialPort(output, MAX_DATA_LENGTH);
		microOutput1 = strcmp(output, "a");
		microOutput2 = strcmp(output, "b");

		if (microOutput1 == 0)
		{
			recognise.detectFace();
		}

		if (microOutput2 == 0)
		{
			record.videoRecord();
		}

	}

	return 0;
}
```

Appendix E – Facial Recognition and Video Surveillance Microcontroller Code

```
void main() {
ANSEL = 0;
ANSELH = 0;
CM1CON0 = 0;
CM2CON0 = 0;
trisc=0b00000110;
portc=0;
while(1)
{

if(PORTC.F1 = 1)
{
UART1_Init(9600); // Initialize UART module at 4800 bps
UART1_Write_Text("b");
Delay_ms(1000);

}
else
{
```

```
}
if(PORTC.F2 = 1)
{

UART1_Init(9600); // Initialize UART module at 4800 bps
UART1_Write_Text("b");
Delay_ms(1000);
}
if (UART1_Data_Ready()) {   // If data is received,
            uart_rd = UART1_Read(); // read the received data,
            UART1_Write(uart_rd);
}
}
```

Appendix F: Codes for Energy Saving, Security and Isolation system

A] Fire Alarm subsystem code:

```
#include <inttypes.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include <util/delay.h>

int main()
 {
   // Write your code here
   DDRB |= (1<<2)| (1<<5);
   DDRD &= ~(1<<0); //sensor input
   PORTD &= ~(1<<0);

   TCCR0A |= (1<<WGM01)|(1<<WGM00)|(1<<COM0A0); // clear OC0 on compare match, set at BOTTOM
(non- inverting);

   uint8_t OCRvar1, OCRvar2, OCRvar3;
   uint8_t i=0;
   uint8_t j=0;
   //uint8_t SmokeSense;

   while (1)
   {
    //SmokeSense = PIND;
    //uint8_t SDbit = SmokeSense & 0xFE;

    if (PIND == 0x01)
    {
      PORTB |= (1<<5);
      OCRvar1 = 150;
      OCRvar2 = 42;
      OCRvar3 = 97;

      //Tone 1
      for (i = 0; i < 7; i++)
      {
         TCCR0B |= (1<<CS01);
         OCR0A = OCRvar1;
         OCRvar1 += 1; // increments duty cycle, hence output. Simulates a fade in, i.e.
increasing volume over time
         _delay_ms(10);
      }
      _delay_ms(10);

   //Tone 2
      for (i = 0; i < 7; i++)
      {
         TCCR0B |= (1<<CS01)|(1<<CS00); // alters frequency
         OCR0A = OCRvar2;
```

```
        OCRvar2 += -2; // fade out- deccreasing volume over time
        _delay_ms(10);
      }
      _delay_ms(10);

      //Tone 3
      for (i = 0; i < 9; i++)
      {
        TCCR0B |= (1<<CS01);
        OCR0A = OCRvar3;
        OCRvar3 += 3; // OCRvar2 += -2; // fade out- deccreasing volume over time
        _delay_ms(10);
      }
      _delay_ms(10);
   }
   else
   {
      _delay_ms(10);
   }
 }
   return 0;
 }
```

B] Cell Phone Detector Subsystem code:

```
//Disclaimer: The LCD header is used from the website found at reference [47]. It was created
by //Istrate Liveu

#include <inttypes.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include "OnLCDLib.h"
#include <util/delay.h>

int main()
 {
   // Write your code here
   LCDSetup(LCD_CURSOR_BLINK);

   DDRD = 0x00;        //sets port d as input
   PORTD = 0x00; // sets port d with high Z. Sinks current

   uint8_t temp1;
   //uint8_t temp2;
   //uint8_t temp3;

   LCDWriteString("Seat Numbers:");

   while (1)
   {
      temp1 = PIND;
      //LCDWriteIntXY(8, 1, temp1, 3);
      /*
      Letting pin4,5,6 correspond to seat 1, 2, 3
      Adjusting this to involve all input pins would require simply adding more case statements
      */

      switch (temp1)
      {
        case 16:
        {
           LCDWriteStringXY(1,2, "               ");
           LCDWriteStringXY(1,2, "1");
           break;
        }
        case 32:
        {
           LCDWriteStringXY(1,2, "               ");
```

```
            LCDWriteStringXY(1,2, "2");
            break;
         }
         case 48:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "1 and 2");
            break;
         }
         case 64:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "3");
            break;
         }
         case 80:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "1 and 3");
            break;
         }
         case 96:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "2 and 3");
            break;
         }
         case 112:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "1, 2 and 3");
            break;
         }
         default:
         {
            LCDWriteStringXY(1,2, "               ");
            LCDWriteStringXY(1,2, "NONE");
         }
      }
   _delay_ms(500);
   }

   return 0;
 }
```

C] Temperature Control subsystem design

```
//Disclaimer: The LCD header is used from the website found at reference [47]. It was created
by //Istrate Liveu

#include <inttypes.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include "OnLCDLib.h"
#include <util/delay.h>

uint16_t adcVal;
int rTemp;


int main()
 {
   DDRD = 0xFF;
   LCDSetup(LCD_CURSOR_BLINK);
   LCDWriteStringXY(1,1,"STemp =");
   LCDWriteStringXY(1,2,"MTemp =");

   DDRD = 0x00;
```

```
   PORTD = 0x00;
   DDRC = 0x00;
   PORTC = 0x00;
   sei();

   ADCSRA = 0x8F;
   ADMUX = 0x40;

   uint8_t temp1;
   uint8_t temp2 = 0;
   uint32_t counter=0;

   while (1)
   {
      temp1 = PIND;
      counter++;
      if ((temp2 != temp1) || (counter == 1024))
      {
         ADCSRA |= (1<<ADSC);
         LCDWriteStringXY(1, 2, "                ");
         LCDWriteStringXY(1,1,"STemp =");
         LCDWriteIntXY(8, 1, temp1, 3);
         LCDWriteStringXY(12, 1, "C");

         if (temp1 < rTemp)
         {
            PORTB |= (1<<3);
         }
         else if (temp1 > rTemp)
         {
            PORTB &= ~(1<<3);
         }
         temp2 = temp1;
         counter = 0;
      }
      else
      {
         _delay_ms(20);
      }
};
return 0;
}

ISR(ADC_vect)
{
   uint8_t theLow = ADCL;
   adcVal = ((ADCH<<8)|theLow);
   /*counter++;
   if (counter == 256)
   {
      adcVal = adcVal >> 4;
      LCDWriteIntXY(1, 2, adcVal, 6);
   }
   else
   {
      ADCSRA |= (1<<ADSC);
   } Oversampling and decimaation code               */
   LCDWriteStringXY(1, 2, "                ");
   rTemp = (adcVal - 104)/2;
   LCDWriteStringXY(1,2,"MTemp =");
   LCDWriteIntXY(8, 2, rTemp, 3);
   LCDWriteStringXY(12, 2, "C");
}
```

D] Light Control subsystem code:

```
#include <inttypes.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
```

```
#include <util/delay.h>
#include <math.h>

void shiftReg(void);
void latchReg(void);
void resetReg(void);

int main()
 {
   // Write your code here

   DDRB = 0xFF;
   DDRD = 0x00;
   PORTD = 0x00;

   uint8_t temp1;
   uint8_t temp2=0;
   uint8_t temp3=0;

   resetReg();
   PORTB &= ~(1<<4);
   int i;
   int exp;

   while (1)
   {
      temp1 = PIND;
      if (temp2 != temp1)
      {
         temp3 = temp1;
         for (i=0; i<8; i++)
         {
            exp = pow(2, (7-i));
            if (temp3 > exp)
            {
               PORTB |= (1<<1);
               temp3 -= exp;
               shiftReg();
            }
            else
            {
               PORTB &= ~(1<<1);
               shiftReg();
            }
         }
         temp2 = temp1;
         latchReg();
      }
      else
      {
         _delay_ms(20);
      }
   }


      ;
   return 0;
 }

void shiftReg(void)
{
PORTB|=(1<<0);
_delay_ms(20);
PORTB&=~(1<<0);
_delay_ms(20);
}

void latchReg(void)
{
PORTB|=(1<<2);
```

```
_delay_ms(3);
PORTB &=~(1<<2);
_delay_ms(3);
}

void resetReg(void)
{
PORTB&=~(1<<3);
_delay_ms(10);
PORTB|=(1<<3);
_delay_ms(10);
}
```

LCD header file

Disclaimer: The LCD header is used from the website found at reference [20]. It was created by Istrate Liveu

Appendix G: Microcontroller registers

PWM generation:

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x2A	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x33	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x29	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x28	OCR0B[7:0]								OCR0B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

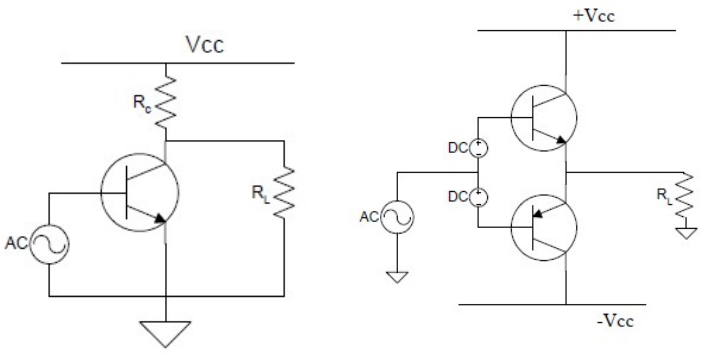
Appendix H – Power Amplifier design:

H.1. Theoretical background

1. Output Stage

For small signal amplifier design, a resistive pull-up is used at the output. This found to be inefficient when handling large bi-directional currents at the output. The basic amplifier design has very high power dissipation across its resistors when attempting to handle high current. This power loss can be

found to be far greater than the actual power being delivered. A basic amplifier circuit is shown below next to a push pull amplifier containing two BJT's [38]:



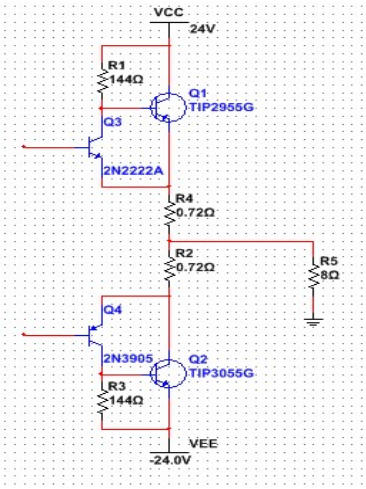
The push-pull amplifier is capable of having very low output impedances as well allowing a vast range of quiescent currents. These characteristics largely depend on how the transistors in the amplifier are biased. There are various configurations of the output stages of an audio amplifier, these are classes A, B, AB, C and D. The classes are separated based on the value of the quiescent current and the presence of switching circuitry.

Class A amplifiers characterises maintaining a current flow in both of the output transistors constantly. This characteristic is independent of signal level and is typical of all small signal amplifiers. This results in low distortion values. The setbacks to class A amplifiers are that maximum efficiency is limited to at most 50 %, and the amplifier requires a constant power supplied to it, which implies at low voltage operation, respectively large amounts of power is dissipated via the resistors as heat loss. For class B operation, the quiescent current is biased as just above zero i.e. right at cut-off. Any perceptible change in the input voltage would result in just one of the two devices passing a current to the output. By analysing the predicted output current signals it can be shown that class B amplifiers show considerably higher efficiency when compared to class A amplifiers. However, due to the proximity to cut-off voltage, there is considerable distortion in class B amplifiers at low voltages [38].

Class AB amplifiers is not a distinct class, as it's identifier suggests, but a hybrid of classes A and B. The operating principles of classes A and B represent limiting cases, thus finding middle should allow for a trade-off between the desirable characteristics of either class. A class AB amplifier has both transistors conducting for small signals, but only one conducts for large signals. The output stage is biased to carry a quiescent current a fair amount less than half the maximum output current, which is required by class A amplifiers, but still enough to ensure both transistors conducting. At higher output signal levels, only one device conducts mirroring a class B amplifier. The setup effectively displays a efficiency closer to that class B amplifiers but with significantly less distortion, particularly at low signal levels [38].

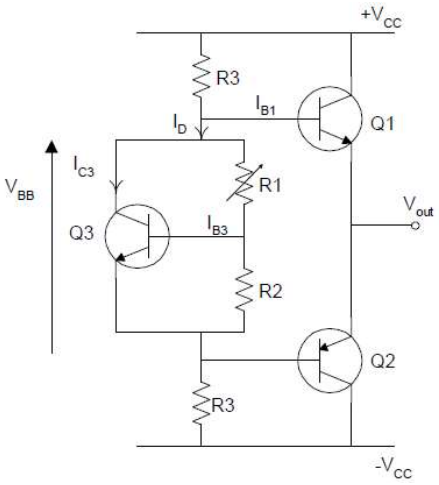
Classes C and D utilizes high frequency switching circuitry. This generates very high efficiencies but also have high quantities of switching transients that generate Electromagnetic Interference. This requires highly rigid filters to counteract its effects. The larges costs and lack of availability of high frequency components leads to these classes being largely unfeasible.

The following is a Sziklai pair class AB output stage designed to produce 25W power for an 8Ω load. The components for this circuit are listed in appendix A, while the calculations are in appendix B:

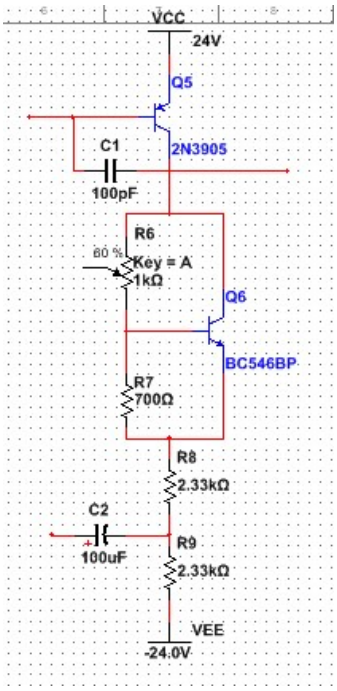


2. Driver Stage

The driver stage provides biasing for the output stage transistors as well as allowing for compensation of thermal changes with the circuitry. There exists three main methods of providing biasing to the output stage devices, namely resistive; diode; and transistor based methods. The resistive method allows for easily adjusting the bias voltage level but is not suitable for compensation of heat changes in the circuit. The converse applies to the diode biasing method as it allows for compensation of thermal changes via having similar thermal characteristics. The setback of this schema is that it cannot easily adjust the bias level due it being intrinsically discrete in quantity. The circuit depicted below is a basic example of a transistor based method, known as Vbe multiplier [38]:



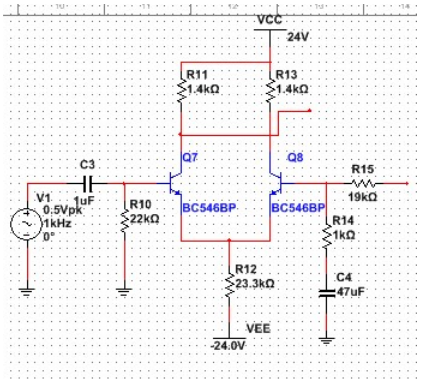
The active component helps compensate for thermal changes while the variable resistor allows adjusting of the biasing levels [38]. A driver circuit was designed using the Vbe multiplier to provide biasing to a Sziklai pair output stage. The components for this circuit are listed in appendix A, while the calculations are in appendix B:



Capacitor C1 and the transistor Q5 allow for a constant current to be supplied to transistor Q3 of the output stage. The basic Vbe multiplier and bootstrap resistors (R8 and R9) provides the biasing for the amplifier.

3. Input Stage

The input stage entails a basic amplifier stage, either single or differential, and passing the amplified signals to the output and driver stages. The following is a differential input amplifier design, the components for this circuit are listed in appendix A, while the calculations are in appendix B:



The full circuit diagram and simulation be found in Appendix C.

H.2. Components

1. Output Stage:

- 5x resistors (1 x 10 Ohm power resistor)
- 1 x TIP2955 power transistors
- 1 x TIP3055 power transistors

- 1 x 2N2222 transistor (Q3)($h_{FE\,(min)}$ value of 100)
- 1 x 2N2905 transistor (Q5) ($h_{FE\,(min)}$ value of 100)

2. Driver Stage:

- 3 x resistors
- 1 x variable resistor
- 1 x 2n2905 transistor ($h_{FE\,(min)}$ value of 100)
- 1 x BP546 transistor
- 1 x Electrolytic capacitor
- 1 x ceramic capacitor

3. Input Stage:

- 6 x resistors
- 2 x ceramic capactors
- 2 x BC546 transistor

H.3. Power Amplifier Calculations

1. Output stage:

Output Voltage:

$$P_o = \frac{V_o^2}{2R_L} \quad \Rightarrow \quad V_o = \sqrt{P_o \times 2R_L} = \sqrt{25 \times 2(8)} = 20V \quad \Rightarrow \quad V_{cc} > V_o = 20V$$

Losses: $$\text{Losses} = V_{BE}(Q6) + V_{CE\,(sat)}(Q5) = 1.8 + 1.1 = 2.9V$$

Therefore: $V_{cc} = V_o + \text{Losses} = 20 + 2.9 = 22.9V \quad \Rightarrow \quad V_{cc} = 24V$

Output Current: $I_o = \frac{V_o}{R_L} = \frac{20}{8} = 2.5A$ (below maximum current of 15A)

The resistors R_{11} and R_{13} provide a discharge path for the base currents of transistors Q6,8 respectively, when switching off. This voltage is lost, therefore we choose to take 10% of I_B (Q6,8):

$$I_{R1,R3} = 10\%\, I_{B\,(Q1,2)} = \frac{0.1 * I_o}{h_{FE\,(\min)}} = \frac{0.1 * 2.5}{20} = 0.1 * 125x10^{-3} = 12.5mA$$

Therefore: $R_{1,3} = \frac{v_{be\,(Q1,2)}}{I_{R1,R3}} = \frac{1.8}{12.5\times10^{-3}} = 144\Omega\,(150\Omega\ \text{E12series})$

The resistors R_{12} and R_{14} allow for negative feedback to occur. The negative feedback avoids thermal runaway in the Sziklai pair. The resistances necessary to drop approximately voltage approximately one v_{BE} at maximum current:

$$R_{2,4} = \frac{V_{BE}(Q6)}{I_o} = \frac{1.8}{2.5} = 0.72\Omega\,(1\Omega\ \text{E12series})$$

Power Dissipation: $I_{o\,rms}{}^2 \times R_2 = \left(\frac{2.5}{\sqrt{2}}\right)^2 \times 0.72 = 2.25W$

$$I_{C\,(Q3,4)} = I_{B\,(Q1,2)} + I_{R1,3} = 125\text{mA} + 12.5\text{mA} = 137.5\ \text{mA}$$

Hence:

$$I_{B\,(Q3,4)} = \frac{I_{C\,(Q3,4)}}{h_{FE\,(\min)}} = \frac{137.5 \times 10^{-3}}{100} = 1.375\text{mA}$$

2**. Driver Stage:**

As found in the output stage calculations, a constant current of 1.375mA is required to drive the transistor 2N2222 Q3. A transistor will be used to provide the constant current source, this is achieved by operating this transistor at a dc biasing current far exceeding the base current of Q3, maybe 3 times more. The 2N2905 transistor is selected to be used, operating at I_C = 5mA.

It can be shown the Sziklai pair at the output will require $V_{BB} = 2xV_{BE}$ for a class AB amplifier. Provided I_B (Vbb multiplier transistor) << I_{R6} (variable resistor), then it can be assumed that $I_{R6} \approx I_{R7}$, as the base current is negligible in comparison to these currents.

$$V_{BB} = v_{BE}\left(1 + \frac{R_6}{R_7}\right) \Rightarrow \left(1 + \frac{R_6}{R_7}\right) = \frac{V_{BB}}{v_{BE}} = 2 \quad \Rightarrow \quad R_6 = R_7$$

Using Kirchoff's law (assuming $I_{R6} = 1\text{mA}$):

$$R_7 = \frac{v_{BE\,(Q6)}}{I_{R6}} = \frac{0.7}{1\times10^{-3}} = 700\Omega = R_6 \ \ (680\Omega\ \text{E12series})$$

Considering quiescent current of 5mA is passing through the bootstrap resistors, R_8 and R_9. Also, the bootstrap capacitance(C_2) has a minimum where the bootstrap resistances are equal, i.e. R_8 = R_9. Therefore:

$$V_{(R8+R9)} = V_{cc} - \frac{V_{BB}}{2} = V_{cc} - V_{BE(Q6)} = 24 - 0.7 = 23.3\text{V} \quad \Rightarrow R_8 = R_9 = \frac{1}{2}\frac{23.3}{5\times10^{-3}} = 2.33\text{k}\Omega$$

(2.2kΩ E12series)

Assume minimum frequency of 30Hz that the power amplifier will be capable of handling. The effective resistance when looking through C_2 is the parallel combination of the bootstrap resistances.

$$C_2 \gg \frac{1}{2\pi\times f_L\times(R_8||R_9)} = \frac{1}{2\pi*30*(0.5*2200)} = 5.78\mu\text{F} \Rightarrow \text{take } C_2 = 100\mu\text{F}$$

3. Input Stage:

Regarding Q5: $$I_{B\,(Q5)} = \frac{I_{C\,(Q5)}}{h_{fe\,(min)}} = \frac{5\times10^{-3}}{100} = 50\mu A$$

This current is considerably low and thus contributes very little demand on the input stage transistor Q7 (BC546). However, there would most likely demands on transistor Q7 due to capacitor C1 of the driver stage. This capacitor is used for high frequency stabilization, hence the charging current may make greater demands at relatively higher frequencies. Assuming a frequency of 25 kHz and a capacitance of 100pF:

$$I_{max\ (C1)} = (C_1)(V_o)(\omega) = (100 \times 10^{-12})(20)(2\pi.20 \times 10^3) = 0.31mA$$

Setting $I_{C\,(Q7,8)}$= 0.5mA: $R_{11,13} = \frac{V_{BE\,(Q3)}}{I_{C(Q7,8)}} = 1.4k\Omega\ (E12 - 1.5k\Omega)$

$I_{R12} \approx I_{C(Q7)} + I_{C(Q8)} = 2 * I_{C(Q7,8)} = 1mA$; $V_{B\,(Q1)} \approx 0$; => $v_{E(Q7)} = v_{B\,(Q7)} - v_{BE\,(Q7)} \approx -0.7V$

Therefore: $R_{12} = \frac{V_{E\,(Q7)} - (-V_{cc})}{I_{R12}} = \frac{(-0.7) - (-24)}{1 \times 10^{-3}} = 23.3k\Omega\ (E12 - 22k\Omega)$

The differential input resistance is: $r_{id} = 2r_\pi = 2 * \frac{h_{FE\,(Q7)} \times V_T}{I_{C\,(Q7)}} = 2.\frac{100 * 25 \times 10^{-3}}{0.5 \times 10^{-3}} = 10k\Omega$

The negative feedback is applied in series, therefore the input resistance including feedback should be the differential input resistance raised by the return difference (factor D). This makes the input resistance far greater than the differential input resistance, effectively making R_{10} the dominant resistance. Choose R_{10} = 22kΩ(E12).

At Q7, the capacitor (C_1) and resistance (R_{10}) form a high pass filter:

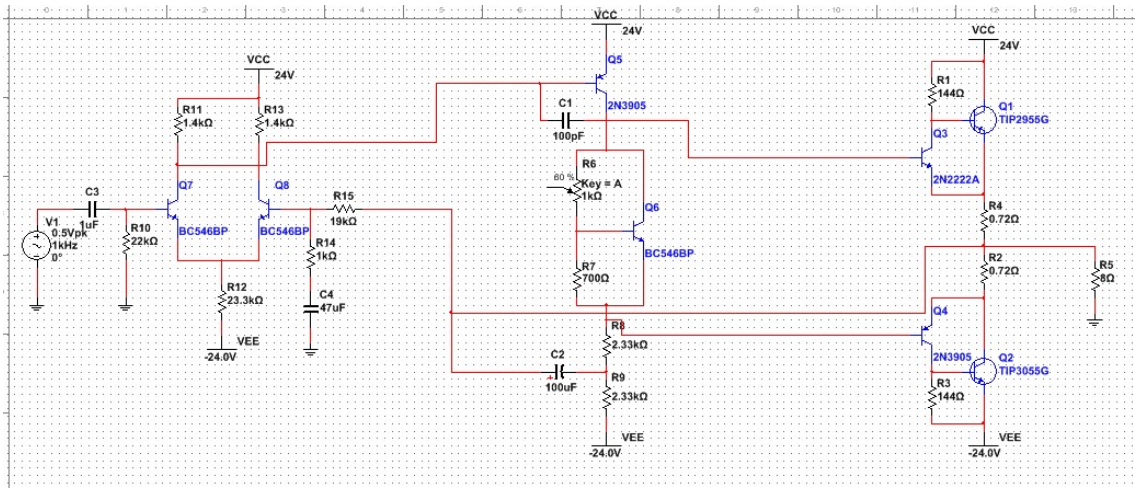
$C_3 \gg \frac{1}{(2\pi)(f_L)(R_{10})} = \frac{1}{(2\pi)(30)(22 \times 10^3)} = 241nF$; let $C_1 = 1uF$

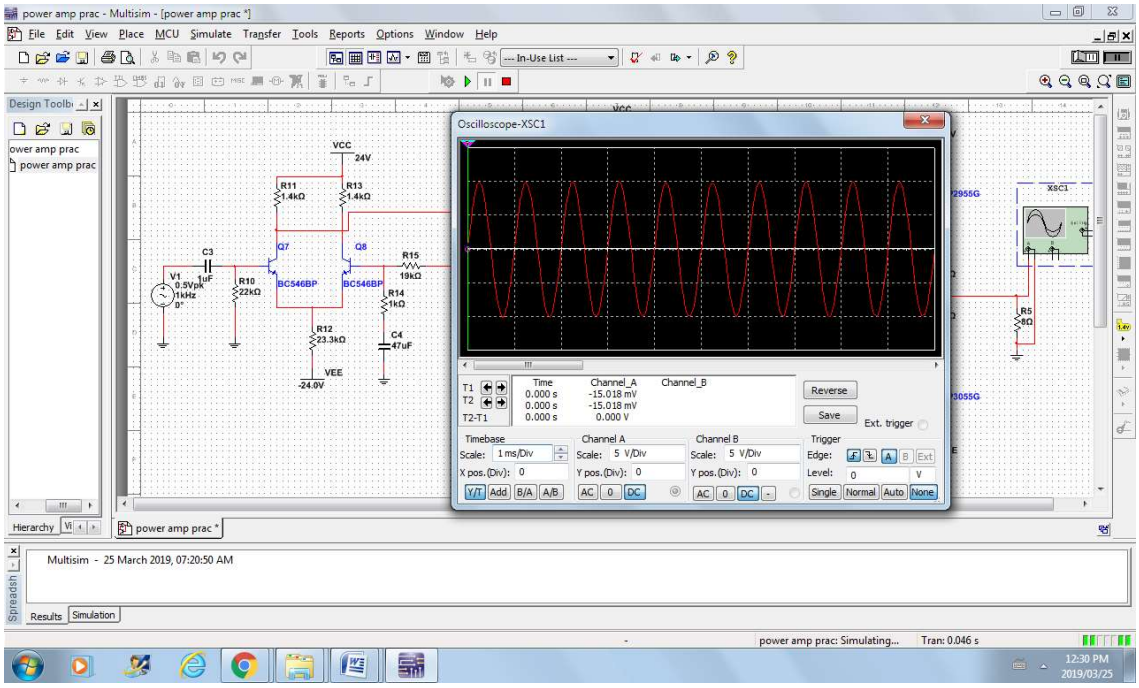
At Q8: $A_{Vdc} = \left(1 + \frac{R_{15}}{R_{14}}\right)$; let $V_{in} = 0.5V, A_{Vdc} = \frac{V_o}{V_{in}} = \frac{20}{0.5} = 40\frac{V}{V}$

Set $R_{14} = 1k\Omega$ =>$R_{15} = R_{14}(A_{Vdc} - 1) = (1000)(20 - 1) = 19k\Omega(E12 - 18k\Omega)$

For C_4: $C_4 \gg \frac{1}{2\pi f_L R_{14}} = \frac{1}{2\pi)(30)(1 \times 10^3)} = 5.3uF$; let C_2 = 47uF.

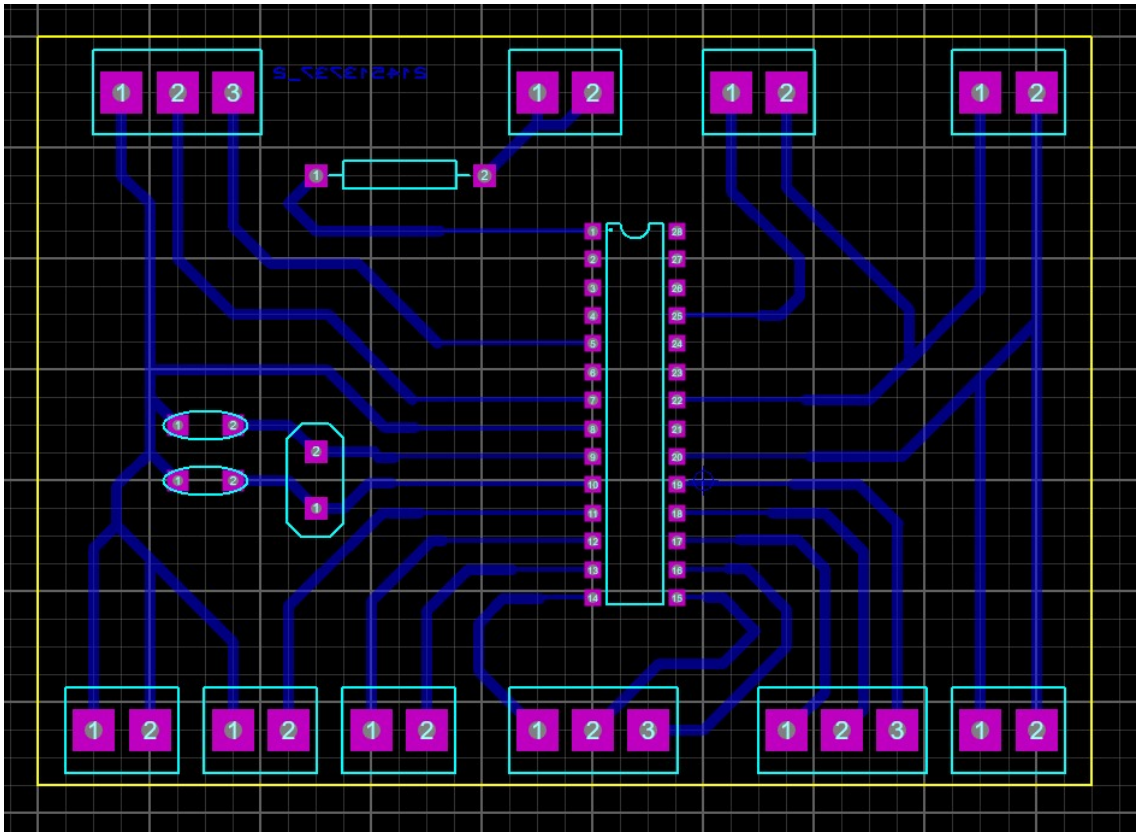
H.4. Power amplifier full circuit and simulation

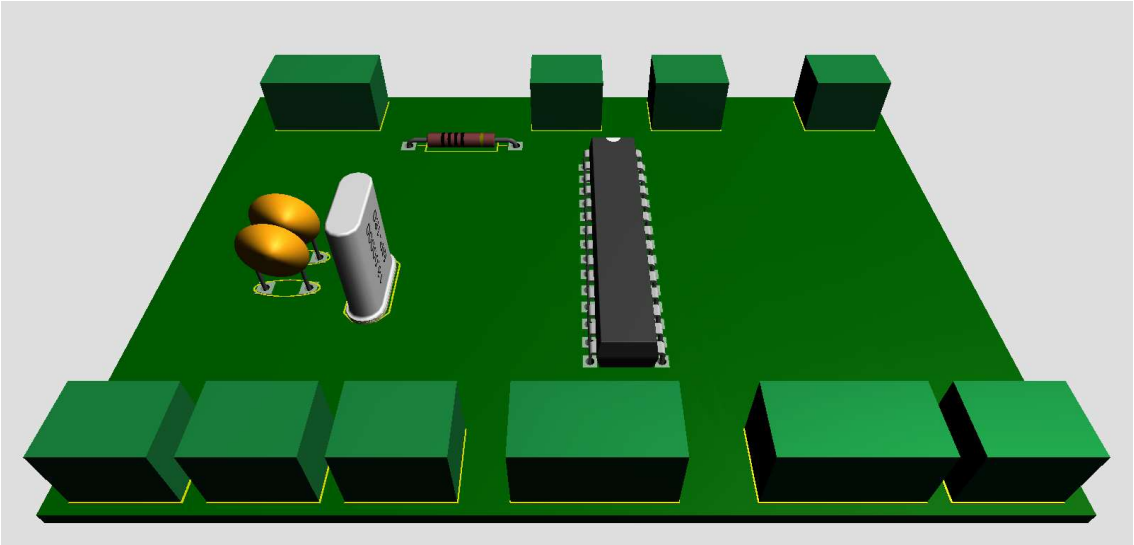




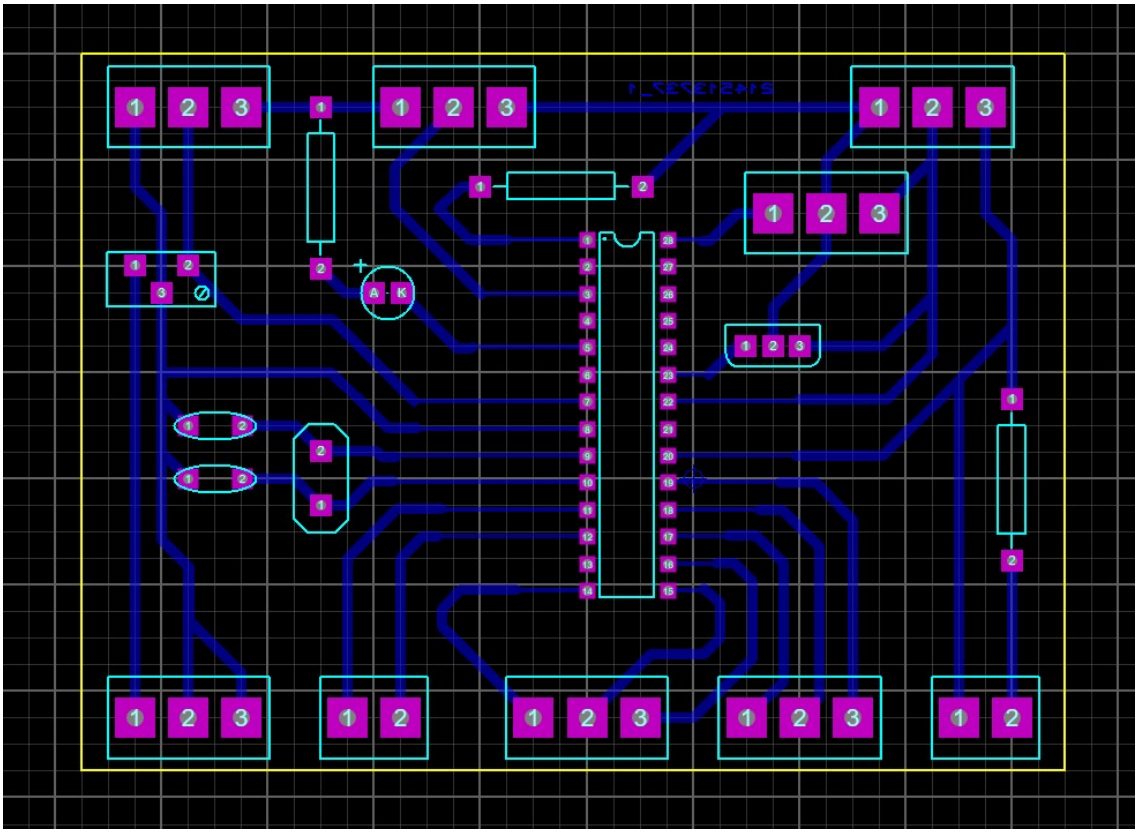
Appendix I:

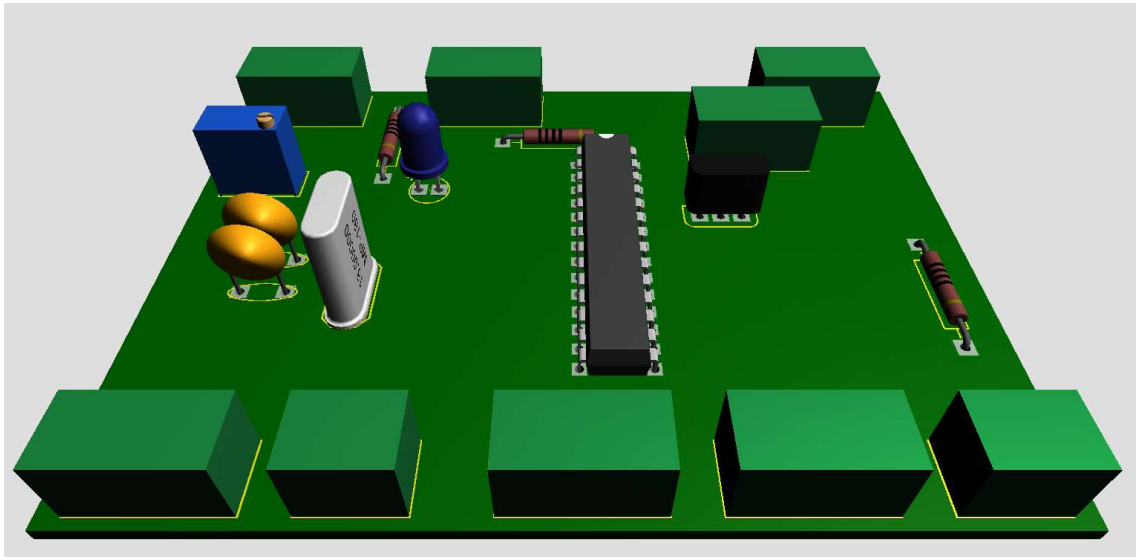
A: Fire Alarm and Light Control subsytem PCB design:



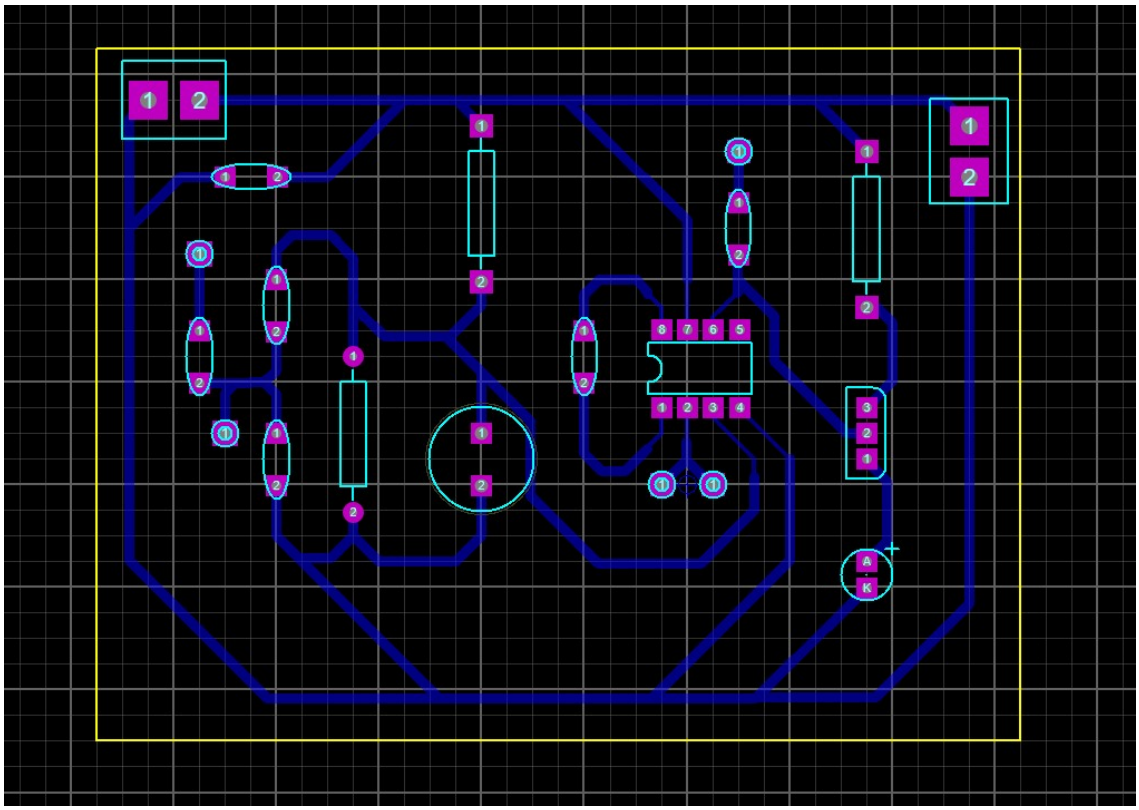


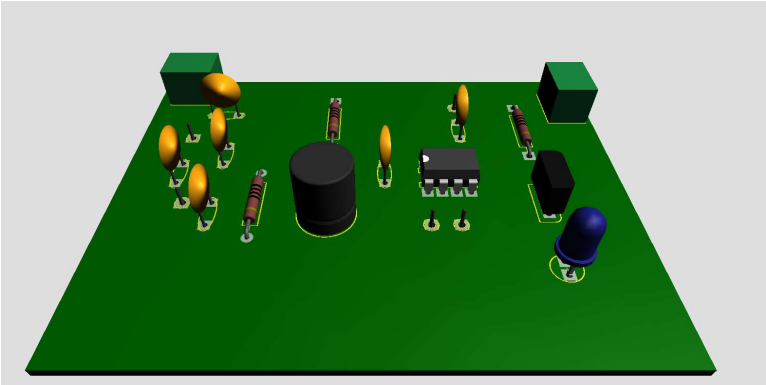
B: Temperature Control and RF detector PCB design:



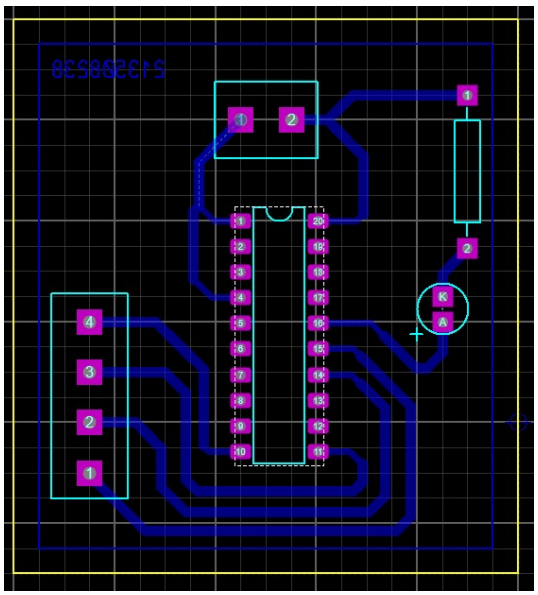


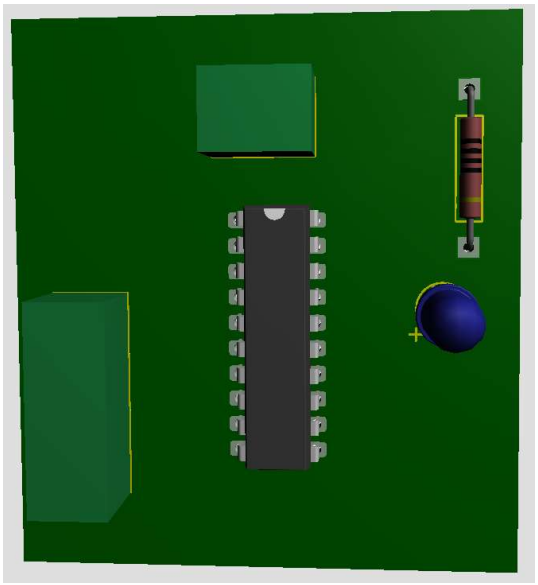
C: RF detector circuit:

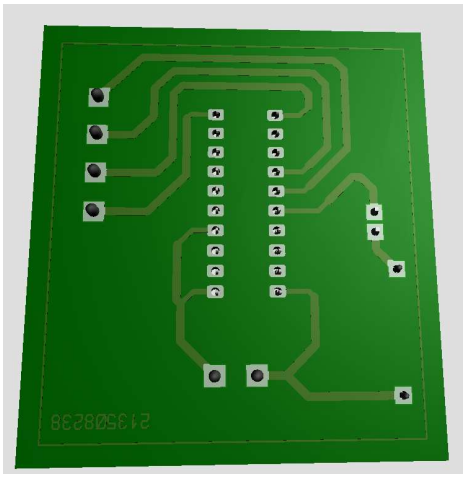




Appendix J – Face Recongintion and Video surveillance PCB design







Appendix K – Meeting Minutes

Smart Exam Hall Meeting Minutes

Date: 09 February 2019

Time: 16:00

Location: Enpower Training Services & Vital Skills (Pty) Ltd

43 Sea Cow Lake Road, Springfield, Durban

(Office of Domain Expert)

Meeting called by:	Vikash Narrainan (Domain Expert)	**Meeting purpose**	Project discussion and brain storming
Facilitator:	Devashnee Bhagawandin	**Minutes issued by:**	Devashnee Bhagawandin
Domain expert job title:	National Sales and Co-ordinating Manager		
Domain expert qualification:	Diploma in Business administration and management		
Attendees:	Vikash Narrainan (Domain Expert) Devashnee Bhagawandin Rishay Ramcharan Keshav Jeewanlall Sashin Ramdhani		

Minutes

Agenda item: Smart exam hall

Discussion:

The specifications of the exam hall:

- controlled access to room
- two-way authentication
- automated register control
- no student is to be greater than 20 minutes late for an exam/lecture
- detect cell phones being used
- camera present in case of enquiry

Parking lot:

- motion sensors for optimal lighting
- temperature sensors for optimal temperature control
- fire alarm
- audio
- over-ride function

Conclusion:

The decision was made to implement a smart exam hall.

Other Information

Domain expert:

Vikash Narrainan

Date 14/02/2019 Sign _____

Tel: 083 691 0555

Attendees:

1. Devashnee Bhagawandin

   Date 09-02-2019 Sign _____

2. Keshav Jeewanlall

   Date 09/02/19 Sign _____

3. Rishay Ramcharan

   Date 09/02/19 Sign _____

4. Sashin Ramdhani

   Date 09/02/19 Sign _____

Smart Exam Hall Meeting Minutes

Date: 21 February 2019

Time: 16:00

Location: Empower Training Services & Vital Skills (Pty) Ltd

43 Sea Cow Lake Road, Springfield, Durban

(Office of Domain Expert)

Meeting called by:	Vikash Narrainan (Domain Expert)	**Meeting purpose**	Detailed projected specifications discussion
Facilitator:	Devashnee Bhagawandin	**Minutes issued by:**	Devashnee Bhagawandin
Domain expert job title:	National Sales and Co-ordinating Manager		
Domain expert qualification:	Diploma in Business administration and management		

Attendees:

Vikash Narrainan (Domain Expert)
Devashnee Bhagawandin
Rishay Ramcharan
Keshav Jeewanlall
Sashin Ramdhani

Minutes

Agenda item: Smart exam hall detailed specifications

Discussion:

The specifications of the exam hall:

- controlled access to room
- two-way authentication
- student cards and facial recognition
- automated register control
- no student is to be greater than 20 minutes late for an exam/lecture
- detect cell phones being used
- camera present in case of enquiry
- motion sensors for optimal lighting
- temperature sensors for optimal temperature control
- fire alarm over-rides the system and opens the service doors for faster exit
- audio for announcements
- over-ride function
- camera surveillance is controlled via the access control and motion sensors
- feasibility of the system

Conclusion:

The finer details of the examination hall have been finalized. The specifications and paper design is due to commence following the next meeting.

Other Information

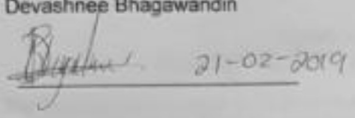
Domain expert:

Vikash Narrainan

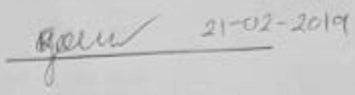
Tel: 083 691 0555

Attendees:

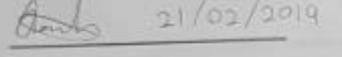
1. Devashnee Bhagawandin



2. Keshav Jeewanlall



3. Rishay Ramcharan



4. Sashin Ramdhani

Smart Exam Hall
Meeting Minutes 3

Date: 26 April 2019

Time: 13:00

Location: Enpower Training Services & Vital Skills (Pty) Ltd

43 Sea Cow Lake Road, Springfield, Durban

(Office of Domain Expert)

Meeting called by:	Vikash Narrainan (Domain Expert)	**Meeting purpose**	Integrated system demonstration
Facilitator:	Devashnee Bhagawandin	**Minutes issued by:**	Devashnee Bhagawandin
Domain expert job title:	National Sales and Co-ordinating Manager		
Domain expert qualification:	Diploma in Business administration and management		

Attendees:

Vikash Narrainan (Domain Expert)
Devashnee Bhagawandin
Rishay Ramcharan
Keshav Jeewanlall
Sashin Ramdhani

Minutes

Agenda Item: Demonstration of sub system functionalities

Discussion:

Demonstrated the functionality of the different sub systems:

- Database and computer application
- RF ID authentication and service door over-ride
- Face recognition and camera surveillance
- Energy and safety

Comments

Specifications + requirements have been met. Extremely satisfied with the outcomes.

Other Information

Domain expert:

Vikash Narrainan

Date 26/04/2019 Sign

Tel: 083 691 0555

Attendees:

1. Devashnee Bhagawandin

   Date 26/04/2019 Sign

2. Keshav Jeewanlall

   Date 26/04/2019 Sign

3. Rishay Ramcharan

   Date 26/04/2019 Sign

4. Sashin Ramdhani

   Date 26/04/2019 Sign

Smart Exam Hall
Meeting Minutes 4

Date: 3 May 2019

Time: 16:00

Location: Enpower Training Services & Vital Skills (Pty) Ltd

43 Sea Cow Lake Road, Springfield, Durban

(Office of Domain Expert)

Meeting called by:	Vikash Narrainan (Domain Expert)	**Meeting purpose**	Integrated system demonstration
Facilitator:	Devashnee Bhagawandin	**Minutes issued by:**	Devashnee Bhagawandin
Domain expert job title:	National Sales and Co-ordinating Manager		
Domain expert qualification:	Diploma in Business administration and management		

Attendees:

Vikash Narrainan (Domain Expert)
Devashnee Bhagawandin
Rishay Ramcharan
Keshav Jeewanlall
Sashin Ramdhani

Minutes

Agenda item: Demonstration of integrated systems

Discussion:

Demonstrated the functionality of the integrated system

Comments

Fully satisfied with the integration of the sub-systems & the use friendliness.

Other Information

Domain expert:

Vikash Narrainan

Date 03/05/2019 Sign 

Tel: 083 691 0555

Attendees:

1. Devashnee Bhagawandin

   Date 03/05/2019 Sign 

2. Keshav Jeewanlall

   Date 03/05/2019 Sign 

3. Rishay Ramcharan

   Date 03/05/2019 Sign 

4. Sashin Ramdhani

   Date 03/05/2019 Sign

Appendix M

BOARD NOTICE 192 OF 2010

Engineering Council of South Africa

Notification of INDICATIVE TIME BASED FEE RATES

The Engineering Council of South Africa hereby makes it known that the Rates set out in the table below are the indicative time based fee rates referred to in Clause 3.4(3)(b) of the Guideline Scope of Services and Tariff of Fees Rules published under Notice 190, Government Gazette No. 33892 of 23 December 2010.

Category of Staff	Indicative Rate
A	R 1 830 per hour
B	R 1 550 per hour
C	R 920 per hour
D	R 670 per hour

For ease of reference the definitions of Categories A to D, are quoted below:

Category A

In respect of a private consulting practice in engineering, shall mean a top practitioner whose expertise and relevant experience is nationally or internationally recognized and who provides advice at a level of specialization where such advice is recognized as that of an expert.

Category B

In respect of a private consulting practice in engineering, shall mean a partner, a sole proprietor, a director or a member who, jointly or severally with other partners, co-directors or co-members, bears the risks of the business, takes full responsibility for the liabilities of such practice and/or bears project related responsibilities on behalf of the practice and, where the level of expertise and relevant experience is commensurate with the position, performs work of a

conceptual nature in engineering design and development, provides strategic guidance in planning and executing a project and/or carries responsibilities for quality management pertaining to a project.

Category C

In respect of a private consulting practice in engineering, shall mean all salaried professional staff with adequate expertise and relevant experience performing work of an engineering nature and who carry the direct technical responsibility for one or more specific activities related to a project. A person referred to in Category B may also fall in this category if such person performs work of an engineering nature at this level.

Category D

In respect of a private consulting practice in engineering, shall mean all other salaried technical staff with adequate expertise and relevant experience performing work of an engineering nature with direction and control provided by any person contemplated in categories A, B or C.

University of KwaZulu-Natal
School of Engineering, Howard College Campus

Student's Report on ECSA Outcomes	
Programme	BSc Engineering – computer engineering (hons)
Student Name	Devashnee Bhagawandin
Student Number	214502393
Date of Submission	2019-05-06

Instructions for students

1. All sections must be completed
2. For each assessment criteria, evidence of competence must be summarized in the clear blocks, or reference to available evidence must be provided.
3. Where detailed evidence is provided in appendices, these must be both clearly marked and cross-referenced.
4. The same evidence may be used to show competency with more than one outcome.

Instructions for examiners

1. Examiners must assess if competence is demonstrated as stated by the student.
2. If requested, the examiners may interview the student.
3. Comments may be added for each outcome.
4. Remedial actions/s must be specified by the examiners in cases where outcomes are not adequately demonstrated.

1 Exit Level Outcome 1: Problem Solving

Learning outcome: Identify, formulate, analyse and solve complex engineering problems creatively and innovatively	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Application of systematic approach to problem solving including:
1.1. The problem is defined and the criterion for an acceptable solution is identified

Evidence of competence:
The problem was identified after meeting the domain expert and a solution was achieved. There were many solutions presented and evaluated and a problem statement developed and specification set out. This is shown in introduction

Exit level assessment: formulate the problem statement, draft solutions and pick the best solution which is feasible.

1.2 Relevant information and engineering knowledge and skills are identified for solving the problem

Evidence of competence:
The computer application and database application used knowledge from computer methods 1,2 and 3. The Cpp language was used and this was learnt in computer methods 3. Software engineering one and two was used for the design process and the UML language learnt in software engineering two. The project of software engineering two knowledge was used in creating and implementing the

database. Digital systems was used and the practical knowledge of UART was used for the serial communication. Design 2 was used in the simulation testing with the PIC16f690.

Exit level assessment:
Previous knowledge obtained over the years was used in the design and implementation

1.3 Various approaches are considered and formulated that would lead to workable solutions

Evidence of competence:
Wireless communication was investigated
Wired communication via UART and microcontrollers were used
Different types of PC to PC communication was investigated
The PIC and PC communication worked the best
this is shown in 8.6.3 of the report

Exit level assessment:
Feasibility study was done and the best solution was obtained

1.4 Solutions are identified in terms of strengths and weaknesses for the overall solution

Evidence of competence:
the strengths and weaknesses of different solutions was examined. This was done in sections 8.6.3 and 8.4

Exit level assessment:
feasibility analysis where different approaches were compared in terms of strengths and weaknesses.

1.5 Solutions are prioritised in order of suitability

Evidence of competence:
This was done with the domain expert in meeting 2 and in the report in section 8.1 and 8.2

Exit level assessment:
Feasibility study and best solution obtained

1.6 The preferred solution is formulated and presented in an appropriate form

Evidence of competence:
The best solution was obtain and carried out by the group by breaking down the project into sub-systems. These sub-systems were agreed on with the mentor and domain expert. The different sub-sections of the report displays this. Section 8,9;10 and 11.

Exit level assessment:
Specifications and implementation. Working a group

Range Statement:
Problems require identification and analysis. Some cases occur in unfamiliar contexts. Problems are both concrete and abstract and may involve uncertainty. Solutions are based on theory and evidence, together with judgement where necessary.

Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 1

2 Exit Level Outcome 2: Application of Scientific and Engineering Knowledge

Learning outcome: Apply Knowledge of mathematics, natural sciences, engineering fundamentals, and an engineering speciality to solve complex engineering problems. **Assessment Criteria:** Demonstrate competence to: **2.1 An appropriate mix of knowledge of mathematics, natural and engineering science at a fundamental level and in a specialised area is brought to bear on the solution of narrowly-defined engineering problems**	Competence Demonstrated	
	Yes	No

Evidence of competence:
Mathematic and engineering science at a fundamental level was used to calculate the ascii value which was transmitted via the UART modules. Fundamentals such as Object Orientated Programming appendix B was used in programming the computer application this knowledge was obtained in computer methods 3. The design process was obtained using fundamentals of the software design methodology. This is shown is section 8.3
Exit level assessment:
engineering knowledge gained from previous years to design and implement the solution.

2.2 Applicable principles and laws are applied

Evidence of competence:
The Object Orientated Programming principle is used to manage data in terms of encapsulation, efficient memory management and data integrity. Section 8.6.
Exit level assessment:
applied basic principles studied from previous years to the solution.

2.3 Appropriate engineering materials, components or processes are selected

Evidence of competence:
applied basic principles studied from previous years to the solution.
Visual studio and mysql workbenchwas used for development. UART module was used for communication with sub-systems. Section of 8.7
Exit level assessment:
Knowledge from previous years applied

2.4 Concepts and ideas are communicated effectively

Evidence of competence:
UML language used to communicate the sequence of statements and the follow of the code. Algorithms was used to display the followed algorithms for creating the solution. Section 8.3
Exit level assessment:
Using engineering knowledge and applied skills

2.5 Reasoning about engineering materials, components, systems or processes is performed

Evidence of competence:
Programming language used and IDE selected for computer application design and database design. UART used for communication
Exit level assessment:
Using programming language to develop products and provide solutions

2.6 Work is performed within the boundaries of the practice area
Evidence of competence: The chosen solution adhered to engineering practices defined by IEEE. The solution included tasks that were applicable by an engineer hence conforming to the boundaries of the practice area. Exit level assessment: performed all tasks which abide by engineering principles and professional ethics.
Range Statement: Problems require applications of mathematics, basic sciences and engineering sciences for solving them. There should be appropriate applications of the relevant mathematics, basic sciences and engineering sciences with proper judgement.
Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 2

3 Exit Level Outcome 3: Engineering Design		
Learning outcome: Perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **3.1 The design problem is formulated to satisfy user needs, applicable standards, codes of practice and legislation**		
Evidence of competence: the compute application and database design conformed to all the specifications of section 8.1 and section 8.2. the user interface is user friendly and easy to use. Exit level assessment: User friendly system developed		
3.2 The design process is planned and managed to focus on important issues and recognises and deals with constraints		
Evidence of competence: the design process followed the incremental design process with incremental testing done. Section 8.7 shows this. Exit level assessment: Design process and debugging		
3.3 Knowledge, information and resources are acquired and evaluated in order to apply appropriate principles and design tools to provide a workable solution		
Evidence of competence: The Compute application used libraries and a local host database Exit level assessment: utilized knowledge gained from previous years as well as materials produced by external sources to produce the solution		
3.4 Design tasks are performed that include component testing to relevant premises, assumptions and constraints		
Evidence of competence: Section 8.7 shows the testing		

Exit level assessment: performed individual and integration testing to success
3.5 Alternatives are evaluated for implementation and a preferred solution is selected on an elementary, technical and cost basis
Evidence of competence: Feasibility studies done in section 8.4 Exit level assessment: feasibility analysis to find alternatives and identify a viable solution
3.6 The design logic and relevant information is communicated in a report
Evidence of competence: Section 8.6 show implementation and code for relative sections Exit level assessment: Showed problems solved and logic
3.7 Occupational health and safety and environmentally related risks are identified and appropriate measures considered
Evidence of competence: Occupational health and safety and environmentally safe measures were undertaken according to the knowledge obtained in Environmental Engineering (ENEL2ENH2), Engineering Business (ENEL4EB) and Engineering Management & Labour Relations (ENCH4ML). This included finding a solution which is environmentally safe and does not put the user of the system in any form of danger. Exit level assessment: adhered to Occupational health and safety and environmentally related risks when designing and implementing the system
Range statement: A major design problem should be used to provide evidence. The problem would be typical of that which the graduate would participate in a typical employment situation shortly after graduation. The selection of components, systems, engineering works, products or processes to be designed is dependent on the discipline.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 3

8 Exit Level Outcome 8: Individual, Team and Multidisciplinary Working		
Learning outcome: Demonstrate competence to work effectively as an individual, in teams and in multidisciplinary environments.	Competence Demonstrated	
	Yes	No
Assessment Criteria: **8.1 Demonstrates effective individual work by** (i) identification and focus on objectives (ii) working strategically (iii) executing tasks effectively and (iv) delivery of completed work on time		
Evidence of competence:		

1. Identify functional and non-functional specification with the group and the domain expert . section 8.1 and section 8.2.
2. Create sub-sections which allows students time to design and implement and work within the time management objectives.
3. Create a paper design and simulations then implement.
4. Stick to time management

Exit level assessment:
perform tasks individually and tackle any issues faced individually

Assessment Criteria:
8.2 Demonstration of effective team work by
(i) making individual contribution to team activity
(ii) performing critical functions
(iii) enhancing work of fellow team members
(iv) benefitting from the support of team members
(v) communicating effectively with team members and
(vi) delivery of completed work on time

Evidence of competence:

1. Integration and compiling of report
2. setting meetings with domain expert and supervisor
3. explaining the way sub-systems will integrate with computer application and database
4. great motivation from positive team members
5. having weekly meetings with group member. Creating whatapp group for easy communication. Meeting with domain expert and supervisor.
6. Phase reports handed in on time

View the minutes of meeting

Exit level assessment:
Good time management and communication with domain expert and group members

Assessment Criteria:
8.3 Demonstration of multidisciplinary work by
(i) acquiring a working knowledge of co-workers' discipline
(ii) using a systems approach and
(iii) communication across discipline boundaries

Evidence of competence:

1. Understanding the domain experts' field and the requirements of his job minute of meeting 1.
2. Email for setting up meeting and keeping minutes of meeting with members
3. Cross over with hardware and software. UART communication section 8.6.5

Exit level assessment:
Working as a team. Being a team player. Respect for different discipline boundaries

Range Statement:
Multidisciplinary tasks require co-operation across at least one disciplinary boundary. Co-operating disciplines may be engineering disciplines with different fundamental bases other than that of the programme or may be outside engineering.

Examiners comments and recommended remedial action if competence not demonstrated for outcome 8

11 Exit Level Outcome 10: Engineering Professionalism

Learning outcome: Demonstrate knowledge and understanding of engineering management principles and economic decision-making (focus on economic decision-making only)	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Demonstrate competence in:

11.1 Detailed budget from the component level (including labour, space, etc)

Evidence of competence:
Section 8.6.3 sub-section showing cost. Feasibility study section 8.4
Exit level assessment:
detailed budget analysis for the subsystem and the entire system.

11.2 Feasibility analysis from a cost perspective from system to component level

Evidence of competence:
detailed budget analysis for the subsystem and the entire system.
section 8.4
Exit level assessment:
Feasibility study

11.3 Activity-based budgeting and time-phased budgets for cost estimation and control considered

Evidence of competence:
Section 7
Exit level assessment:
Economical solution

11.4 Review cost of the Prototype

Evidence of competence:
Section 8.4
Exit level assessment:
Feasibility study

Range Statement:
Basic techniques from economics, business management; project management applied to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

Examiners comments and recommended remedial action if competence not demonstrated for outcome 11

University of KwaZulu-Natal
School of Engineering, Howard College Campus

Student's Report on ECSA Outcomes	
Programme	BSc Computer Engineering
Student Name	Rishay Ramcharan
Student Number	214504863
Date of Submission	6 May 2019

Instructions for students

5. All sections must be completed
6. For each assessment criteria, evidence of competence must be summarized in the clear blocks, or reference to available evidence must be provided.
7. Where detailed evidence is provided in appendices, these must be both clearly marked and cross-referenced.
8. The same evidence may be used to show competency with more than one outcome.

Instructions for examiners

5. Examiners must assess if competence is demonstrated as stated by the student.
6. If requested, the examiners may interview the student.
7. Comments may be added for each outcome.
8. Remedial actions/s must be specified by the examiners in cases where outcomes are not adequately demonstrated.

1 Exit Level Outcome 1: Problem Solving

Learning outcome: Identify, formulate, analyse and solve complex engineering problems creatively and innovatively	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Application of systematic approach to problem solving including:
1.2. The problem is defined and the criterion for an acceptable solution is identified

Evidence of competence:
A meeting was held with a domain expert and a problem was identified. The problem identified was cheating and unauthorised individuals entering an exam venue. The solution is to create a Smart Exam Venue that implement controlled access and contain safety features.

Exit level assessment:

1.2 Relevant information and engineering knowledge and skills are identified for solving the problem

Evidence of competence:
To solve this problem a good understanding of microcontrollers is required as well as a means of communication between a database and microcontroller. Our experience with the PIC microcontroller (from Computer Engineering Design 2 - ENEL3CB) was applied to this project. Experience gained from computer

Exit level assessment:

1.3 Various approaches are considered and formulated that would lead to workable solutions

Evidence of competence:
In Phase 1 a feasibility study was conducted, with different possible solutions. A RFID reader was implemented for controlled access. The EM-18 RFID reader chosen since it had the required specifications. Passive RFID cards were also chosen over Active RFID cards due to its lower price, and its specifications satisfying our "proposed" solution. The PIC16F690 microcontroller was used because it contained all the required specifications. A keypad and lcd screen was used to input a password.

Exit level assessment:

1.4 Solutions are identified in terms of strengths and weaknesses for the overall solution

Evidence of competence:
Specifications were tabulated for the different solutions in the feasibility study. The reasons for our choice of components were briefly explained in this subsection.

Exit level assessment:

1.5 Solutions are prioritised in order of suitability

Evidence of competence:
The EM-18 RFID reader was used because of its low range. Passive RFID cards were chosen over Active RFID cards, due to it not requiring a battery. This makes the tags more durable and allows for a much longer lifespan. The RFID Readers were implemented via UART.

Exit level assessment:

1.6 The preferred solution is formulated and presented in an appropriate form

Evidence of competence:
The best solution was obtained and carried out by the group by breaking down the project into sub-systems. These sub-systems were agreed on with the mentor and domain expert. The different sub-sections of the report displays this. Section 8,9;10 and 11.

Exit level assessment:

Range Statement:
Problems require identification and analysis. Some cases occur in unfamiliar contexts. Problems are both concrete and abstract and may involve uncertainty. Solutions are based on theory and evidence, together with judgement where necessary.

Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 1

2 Exit Level Outcome 2: Application of Scientific and Engineering Knowledge

Learning outcome: Apply Knowledge of mathematics, natural sciences, engineering fundamentals, and an engineering speciality to solve complex engineering problems.	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence to: **2.1 An appropriate mix of knowledge of mathematics, natural and engineering science at a fundamental level and in a specialised area is brought to bear on the solution of narrowly-defined engineering problems**		

Evidence of competence:
Knowledge of mathematics and coding were used to develop the source code for the microcontroller. Analogue electronics were applied implement the circuit. Knowledge gained from Computer Engineering Design 2 and Computer Methods 1, 2 and 3 were directly applied in designing and implementing the subsystem.
Exit level assessment:

2.2 Applicable principles and laws are applied

Evidence of competence:
The principles of Passive Radio Frequency Identification (RFID) was applied for this solution.
The principles of Object Orientated Program were applied for this solution.
The principles of communication via UART were applied for this solution.

Exit level assessment:

2.3 Appropriate engineering materials, components or processes are selected

Evidence of competence:
The UART module was chosen as for communication. The 125KHz RFID tagging system was chosen due to its low cost and reliability. The PIC16F690 was chosen and programmed in C.

Exit level assessment:

2.4 Concepts and ideas are communicated effectively

Evidence of competence:
Block and Flow diagrams were used to explain the logic behind the implementation of the system.

Exit level assessment:

2.5 Reasoning about engineering materials, components, systems or processes is performed

Evidence of competence:
A feasibility study was performed to determine which components to use. The reasoning for the components are listed in the report as well as previous ELO questions.

Exit level assessment:

2.6 Work is performed within the boundaries of the practice area

Evidence of competence:
The problem for this project was a general problem that most universities/schools have. Our solution for this problem was achieved by integrating Computer Engineering with Electronic Engineering.
Exit level assessment:

Range Statement: Problems require applications of mathematics, basic sciences and engineering sciences for solving them. There should be appropriate applications of the relevant mathematics, basic sciences and engineering sciences with proper judgement.
Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 2

3 Exit Level Outcome 3: Engineering Design

Learning outcome: Perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Demonstrate competence in:

3.1 The design problem is formulated to satisfy user needs, applicable standards, codes of practice and legislation

Evidence of competence:
The subsystem meets its functional requirements. The overall project satisfies the needs of the domain expert. The Engineering Design process was followed. The following steps were taken: Problem Identification, Solution Approach, Feasibility study, Design and simulation, Testing and Debugging.

Exit level assessment:

3.2 The design process is planned and managed to focus on important issues and recognises and deals with constraints

Evidence of competence:
A feasibility study was done to ensure that we could afford to buy the components required for the solution using the budget to which our group was allocated. For each phase in our project, we developed a Time Schedule, to which we were supposed to reach our milestones. Our primary goals were to solve the domain expert's problem and to solve the problem within our budget and time constraints.

Exit level assessment:

3.3 Knowledge, information and resources are acquired and evaluated in order to apply appropriate principles and design tools to provide a workable solution

Evidence of competence:
A list of possible solutions was constructed. Thereafter a feasibility study was performed to choose the system that was not only financially feasible, but also safe and efficient.
Exit level assessment:

3.4 Design tasks are performed that include component testing to relevant premises, assumptions and constraints

Evidence of competence:
The subsystem was designed and simulated before being physically implemented. The circuit was then tested and debugged.

Exit level assessment:

3.5 Alternatives are evaluated for implementation and a preferred solution is selected on an elementary, technical and cost basis
Evidence of competence: The feasibility study conducted in the report. Exit level assessment:
3.6 The design logic and relevant information is communicated in a report
Evidence of competence: The design process conducted in the report. Exit level assessment:
3.7 Occupational health and safety and environmentally related risks are identified and appropriate measures considered
Evidence of competence: Occupational health and safety and environmentally safe measures were undertaken according to the knowledge obtained in Environmental Engineering (ENEL2ENH2), Engineering Business (ENEL4EB) and Engineering Management & Labour Relations (ENCH4ML). This included finding a solution which is environmentally safe and does not put the user of the system in any form of danger. Exit level assessment:
Range statement: A major design problem should be used to provide evidence. The problem would be typical of that which the graduate would participate in a typical employment situation shortly after graduation. The selection of components, systems, engineering works, products or processes to be designed is dependent on the discipline.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 3

8 Exit Level Outcome 8: Individual, Team and Multidisciplinary Working		
Learning outcome: Demonstrate competence to work effectively as an individual, in teams and in multidisciplinary environments.	Competence Demonstrated	
	Yes	No
Assessment Criteria: **8.1 Demonstrates effective individual work by** (i) identification and focus on objectives (ii) working strategically (iii) executing tasks effectively and (iv) delivery of completed work on time		
Evidence of competence: All tasks were completed by the deadline and was submitted on the due date. Exit level assessment:		
Assessment Criteria: **8.2 Demonstration of effective team work by**		

(i) making individual contribution to team activity (ii) performing critical functions (iii) enhancing work of fellow team members (iv) benefitting from the support of team members (v) communicating effectively with team members and (vi) delivery of completed work on time
Evidence of competence: All members of the team worked together to complete the project, and delivered work that could be successfully integrated. All work was completed on time and submitted on the due date. Exit level assessment:
Assessment Criteria: **8.3 Demonstration of multidisciplinary work by** (i) acquiring a working knowledge of co-workers' discipline (ii) using a systems approach and (iii) communication across discipline boundaries
Evidence of competence: Due to integration with other team members, knowledge of co-workers was acquired, a systems approach was used and the different disciplines were able to communicate effectively, understanding each one's role. Exit level assessment:
Range Statement: Multidisciplinary tasks require co-operation across at least one disciplinary boundary. Co-operating disciplines may be engineering disciplines with different fundamental bases other than that of the programme or may be outside engineering.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 8

11 Exit Level Outcome 10: Engineering Professionalism		
Learning outcome: Demonstrate knowledge and understanding of engineering management principles and economic decision-making (focus on economic decision-making only)	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **11.1 Detailed budget from the component level (including labour, space, etc)**		
Evidence of competence: Budget analysis Exit level assessment:		
11.2 Feasibility analysis from a cost perspective from system to component level		
Evidence of competence:		

Feasibility study Budget and analysis Exit level assessment:
11.3 Activity-based budgeting and time-phased budgets for cost estimation and control considered
Evidence of competence: Budget analysis Exit level assessment:
11.4 Review cost of the Prototype
Evidence of competence: Budget and analysis Exit level assessment:
Range Statement: Basic techniques from economics, business management; project management applied to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 11

University of KwaZulu-Natal
School of Engineering, Howard College Campus

Student's Report on ECSA Outcomes	
Programme	BSc Computer Engineering
Student Name	Keshav Jeewanlall
Student Number	213508238
Date of Submission	06 May 2019

Instructions for students

9. All sections must be completed
10. For each assessment criteria, evidence of competence must be summarized in the clear blocks, or reference to available evidence must be provided.
11. Where detailed evidence is provided in appendices, these must be both clearly marked and cross-referenced.
12. The same evidence may be used to show competency with more than one outcome.

Instructions for examiners

9. Examiners must assess if competence is demonstrated as stated by the student.
10. If requested, the examiners may interview the student.
11. Comments may be added for each outcome.
12. Remedial actions/s must be specified by the examiners in cases where outcomes are not adequately demonstrated.

1 Exit Level Outcome 1: Problem Solving		
Learning outcome: Identify, formulate, analyse and solve complex engineering problems creatively and innovatively	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Application of systematic approach to problem solving including:
1.3. The problem is defined and the criterion for an acceptable solution is identified

Evidence of competence:

We discovered a real-life problem that our Domain Expert was dealing with. The problem was that a manual register was time consuming and there was a lack of security. The client requested a technological solution which will eliminate the paper work of taking a physical register. The technology should help eliminate cheating and imposters entering the examination hall.
We proposed a solution, and did a feasibility study. The results showed that our solution was achievable and solved the issue that the domain expert was experiencing.

Exit level assessment:

1.2 Relevant information and engineering knowledge and skills are identified for solving the problem

Evidence of competence:

A good understanding of microcontrollers as well as communication methods between the microcontroller and a PC was needed. Our experience with the PIC microcontroller from Computer Engineering Design 2 - ENEL3CB was used in this project and my knowledge of Serial Communication (UART) from Digital Systems (ENEL3DS) was used to implement communication between the PIC microcontroller and the PC.

The face recognition and video surveillance application used knowledge from Computer Methods 1,2 and 3. C++ knowledge from Computer Methods 3 was used to write the code and knowledge of OpenCV from Software Engineering 2 was used to make the application possible.

Exit level assessment:

1.3 Various approaches are considered and formulated that would lead to workable solutions

Evidence of competence:

Wireless and wired communication was investigated and it was found that wired communication gave the best feasibility. Different types of PIC to PC communication was investigated and it was found that UART communication worked the best.

Exit level assessment:

1.4 Solutions are identified in terms of strengths and weaknesses for the overall solution

Evidence of competence:

In Section 10, the feasibility analysis of the face recognition and video surveillance defined various approaches to the design and implementation; which include the strengths and weaknesses of each approach.

Exit level assessment:

1.5 Solutions are prioritised in order of suitability

Evidence of competence:

Solutions were drafted out and defined in Section 10. The Desktop PC approach was chosen as the base for the face recognition and video Surveillance subsystem, reasons discussed in section 10.

Exit level assessment:

1.6 The preferred solution is formulated and presented in an appropriate form

Evidence of competence:

The most appropriate solution was obtained and implemented by breaking down the project into sub-systems. These sub-systems were agreed on with the mentor and domain expert. The different sub-systems are discussed in sections 8, 9, 10 and 11.

Exit level assessment:

Range Statement:
Problems require identification and analysis. Some cases occur in unfamiliar contexts. Problems are both concrete and abstract and may involve uncertainty. Solutions are based on theory and evidence, together with judgement where necessary.

Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 1

2 Exit Level Outcome 2: Application of Scientific and Engineering Knowledge

Learning outcome: Apply Knowledge of mathematics, natural sciences, engineering fundamentals, and an engineering speciality to solve complex engineering problems.	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence to: **2.1 An appropriate mix of knowledge of mathematics, natural and engineering science at a fundamental level and in a specialised area is brought to bear on the solution of narrowly-defined engineering problems**		

Evidence of competence:

Mathematics is used in the face recognition code in order to obtain the vectors for face identification. Fundamentals such as Object Orientated programming, and OpenCV Knowledge was

gained in Computer Methods 3 (ENEL3CC). General Programming Knowledge was gained in Computer Methods 2 (ENEL2CB) and Computer Methods 1 (ENEL2CCA). My experience with the PIC microcontroller from Computer Engineering Design 2 (ENEL3CB) was used in this project and my knowledge of Serial Communication (UART) from Digital Systems (ENEL3DS) was used to implement communication between the PIC microcontroller and the PC.

Exit level assessment:

2.2 Applicable principles and laws are applied

Evidence of competence:

The Object Orientated Programming principle is used to manage data in terms of encapsulation, efficient memory management and data integrity in the face recognition and video surveillance application. Basic principles of UART communication such as setting up the microcontroller to send and receive data, setting the corrected baud rate etc. was applied in this subsystem.

Exit level assessment:

2.3 Appropriate engineering materials, components or processes are selected

Evidence of competence:

The software engineering processes studied in Software Engineering 1 (ENEL2SE) are implemented in the design of this software application. Basic principles studied in previous years was applied to the solution. Visual studio was used for development and a UART module was used for communication with sub-systems

Exit level assessment:

2.4 Concepts and ideas are communicated effectively

Evidence of competence:

Design concepts and ideas are illustrated using flow charts, pseudocode and the Unified Modelling Language (UML) diagrams through Use Cases.

Exit level assessment:

2.5 Reasoning about engineering materials, components, systems or processes is performed

Evidence of competence:

The hardware choices used for development are discussed in section 10.3.1.1 and the programming language and IDE used for development is discussed in Section 10.3.2.1.

Exit level assessment:

2.6 Work is performed within the boundaries of the practice area

Evidence of competence:

The chosen solution adhered to engineering practices defined by IEEE. The manner in which the solution was implemented was feasible and suited the skill level of an engineering student

Exit level assessment:

Range Statement: Problems require applications of mathematics, basic sciences and engineering sciences for solving them. There should be appropriate applications of the relevant mathematics, basic sciences and engineering sciences with proper judgement.
Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 2

3 Exit Level Outcome 3: Engineering Design		
Learning outcome: Perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **3.1 The design problem is formulated to satisfy user needs, applicable standards, codes of practice and legislation**		
Evidence of competence: The face recognition and video surveillance application conformed to all system specifications defined in Section 10.1.1. Exit level assessment:		
3.2 The design process is planned and managed to focus on important issues and recognises and deals with constraints		
Evidence of competence: The face recognition and video surveillance application was developed using the Waterfall methodology. The solution to the problem was executed in a way that effectively eliminated the issue at hand while still keeping within the limit of constraints. Exit level assessment:		
3.3 Knowledge, information and resources are acquired and evaluated in order to apply appropriate principles and design tools to provide a workable solution		
Evidence of competence: The face recognition and video surveillance application utilized resources such as external libraries to perform desired tasks. This is evident as OpenCV and Serial Port libraries were used for Video recording, face detection and microcontroller communication. Exit level assessment:		
3.4 Design tasks are performed that include component testing to relevant premises, assumptions and constraints		
Evidence of competence: Software and hardware testing was conducted on the entire subsystem. Section 10.4		

documents the simulation of the face recognition and video surveillance application. Section 10.5 documents the testing and integration with the other subsystems. Exit level assessment:
3.5 Alternatives are evaluated for implementation and a preferred solution is selected on an elementary, technical and cost basis
Evidence of competence: The Feasibility study in Section 10.6 states the different solutions considered and how the chosen solution is justified. Exit level assessment:
3.6 The design logic and relevant information is communicated in a report
Evidence of competence: The design logic is easily understandable by the use of Use Case and Class diagrams depicted in Section 10.3.2. Exit level assessment:
3.7 Occupational health and safety and environmentally related risks are identified and appropriate measures considered
Evidence of competence: The safety hazards were taken into consideration, electrical wiring of circuits and positioning of systems should not be in harm's way of the user. Exit level assessment:
Range statement: A major design problem should be used to provide evidence. The problem would be typical of that which the graduate would participate in a typical employment situation shortly after graduation. The selection of components, systems, engineering works, products or processes to be designed is dependent on the discipline.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 3

8 Exit Level Outcome 8: Individual, Team and Multidisciplinary Working		
Learning outcome: Demonstrate competence to work effectively as an individual, in teams and in multidisciplinary environments.	Competence Demonstrated	
	Yes	No
Assessment Criteria: **8.1 Demonstrates effective individual work by** (i) identification and focus on objectives		

(ii) working strategically
(iii) executing tasks effectively and
(iv) delivery of completed work on time

Evidence of competence:

The face recognition and video surveillance subsystem was developed by myself this is evidence of the ability to perform individual work. All deadlines were met and time estimations were closely followed.

Exit level assessment:

Assessment Criteria:
8.2 Demonstration of effective team work by
(i) making individual contribution to team activity
(ii) performing critical functions
(iii) enhancing work of fellow team members
(iv) benefitting from the support of team members
(v) communicating effectively with team members and
(vi) delivery of completed work on time

Evidence of competence:

The four subsystems had to be integrated into a whole system in order to produce the whole subsystem. The face recognition and video surveillance subsystem was successfully integrated with other subsystems as evident in Section 10.5. Individual contributions were made to the whole team such as participation in meetings, involvement in composure of reports and providing ideas and help to other team members.

Exit level assessment:

Assessment Criteria:
8.3 Demonstration of multidisciplinary work by
(i) acquiring a working knowledge of co-workers' discipline
(ii) using a systems approach and
(iii) communication across discipline boundaries

Evidence of competence:

The project included Students from Electronic and Computer Engineering and a domain expert from a non-engineering discipline. This provided the opportunity to engage with an Electronic Engineer and a person from outside the engineering domain. Due to integration with other team members, knowledge of co-workers was acquired, a systems approach was used and the different disciplines were able to communicate effectively, understanding each one's role

Exit level assessment:

Range Statement:
Multidisciplinary tasks require co-operation across at least one disciplinary boundary. Co-operating disciplines may be engineering disciplines with different fundamental bases other than that of the programme or may be outside engineering.

Examiners comments and recommended remedial action if competence not demonstrated for outcome 8

11 Exit Level Outcome 10: Engineering Professionalism		
Learning outcome: Demonstrate knowledge and understanding of engineering management principles and economic decision-making (focus on economic decision-making only)	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **11.1 Detailed budget from the component level (including labour, space, etc)**		
Evidence of competence: Budget and analysis different Economic Solutions are discussed in section 10.6 Exit level assessment:		
11.2 Feasibility analysis from a cost perspective from system to component level		
Evidence of competence: Feasibility analysis is discussed in section 10.6 Exit level assessment:		
11.3 Activity-based budgeting and time-phased budgets for cost estimation and control considered		
Evidence of competence: Discussed in Section 7 Exit level assessment:		
11.4 Review cost of the Prototype		
Evidence of competence: Budget analysis discussed in section 10.6.1 and 10.6.2 Exit level assessment:		
Range Statement: Basic techniques from economics, business management; project management applied to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.		
Examiners comments and recommended remedial action if competence not demonstrated for outcome 11		

University of KwaZulu-Natal
School of Engineering, Howard College Campus

Student's Report on ECSA Outcomes	
Programme	BSc Electronic Engineering
Student Name	Sashin Ramdhani
Student Number	214513737
Date of Submission	06/05/2019

Instructions for students

13. All sections must be completed
14. For each assessment criteria, evidence of competence must be summarized in the clear blocks, or reference to available evidence must be provided.
15. Where detailed evidence is provided in appendices, these must be both clearly marked and cross-referenced.
16. The same evidence may be used to show competency with more than one outcome.

Instructions for examiners

13. Examiners must assess if competence is demonstrated as stated by the student.
14. If requested, the examiners may interview the student.
15. Comments may be added for each outcome.
16. Remedial actions/s must be specified by the examiners in cases where outcomes are not adequately demonstrated.

1 Exit Level Outcome 1: Problem Solving

Learning outcome: Identify, formulate, analyse and solve complex engineering problems creatively and innovatively	Competence Demonstrated	
	Yes	No

Assessment Criteria:
Application of systematic approach to problem solving including:
1.4. The problem is defined and the criterion for an acceptable solution is identified

Evidence of competence:
A meeting was scheduled with the domain expert and during said meeting the problem was identified. A range of possible solutions were considered and evaluated with regards to their feasibility and an appropriate solution was selected. A problem statement was thus developed and overall system specifications were established.

Exit level assessment: Create a problem statement, generate solutions and analyse the feasibility of said solutions, selecting the most appropriate one.

1.2 Relevant information and engineering knowledge and skills are identified for solving the problem

Evidence of competence:
The solutions were microcontroller based which were covered in Digital Systems and practically applied in Electronic Design 2. The programming language used is Embedded C, C programming and programming related problem solving were covered extensively in Computer Methods 1 and 2. Embedded C was previously researched and applied in Electronic Design 2. The analysis and design of amplifier based circuits is knowledge obtained from Analogue Electronics 1,2 and 3. The design and implementation of breadboard and PCB based circuitry was covered in Electrical Design 2 and Electronic Design 1. The working and application of the built-in Analogue to Digital Converter on a

microcontroller, which is used for interact with the various sensors, was researched and applied in Electronic Design 3.
This was listed in in the report under 11.2.

Exit level assessment:
The application of knowledge acquired in previous courses towards the design and implementation of the problem solution.

1.3 Various approaches are considered and formulated that would lead to workable solutions

Evidence of competence:
Metal Detectors were investigated, an RF detector was used instead.
A large number of microcontrollers were investigated, some simulated, and the ATmega328 was selected as the most viable solution to various trade-offs such as price vs performance and the minimization of the system.
An external alarm circuit was considered but deemed unnecessary as the microcontroller was found to be capable of generating varying waveforms that can be converted to sound waves.
This was depicted in the report under 11.7. and the circuit design sections (11.3.4; 11.4.4; 11.5.4; 11.6.4.) of each subsection.

Exit level assessment:
A feasible study was undertaken and various solutions were considered and compared to each other.

1.4 Solutions are identified in terms of strengths and weaknesses for the overall solution

Evidence of competence:
The individual possible sub solutions were compared to each other and the more appropriate options were selected. The ATmega328 was found to have the best physical specifications with respect to its cost and allowed for multiple systems to operate simultaneous on it.
This was depicted in the report under 11.7. and the circuit design sections (11.3.4; 11.4.4; 11.5.4; 11.6.4.) of each subsection.

Exit level assessment:
Feasibility study involving a direct comparison of strengths and weaknesses of various possible solutions.

1.5 Solutions are prioritised in order of suitability

Evidence of competence:
This was covered in the second meeting with the domain expert and in the report in sections 11.1.; 11.2 and sub-systems as well (11.3.2; 11.4.2; 11.5.2; 11.6.2)

Exit level assessment:
Feasibility studies undertaken and most sustainable solution selected

1.6 The preferred solution is formulated and presented in an appropriate form

Evidence of competence:
The most appropriate solution was selected and separated into separate sub-systems by the group. A sub-system was assigned to each individual member of the group and they were agreed upon with both the mentor and the domain expert. Sections 8, 9, 10 and 11 display the subsections.

Exit level assessment:
Teamwork. Analysis of solution and separation in to parts. Implementation of individual sub-tasks.

Range Statement:

Problems require identification and analysis. Some cases occur in unfamiliar contexts. Problems are both concrete and abstract and may involve uncertainty. Solutions are based on theory and evidence, together with judgement where necessary.
Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 1

2 Exit Level Outcome 2: Application of Scientific and Engineering Knowledge

Learning outcome: Apply Knowledge of mathematics, natural sciences, engineering fundamentals, and an engineering speciality to solve complex engineering problems. **Assessment Criteria:** Demonstrate competence to: **2.1 An appropriate mix of knowledge of mathematics, natural and engineering science at a fundamental level and in a specialised area is brought to bear on the solution of narrowly-defined engineering problems**	Competence Demonstrated	
	Yes	No
Evidence of competence: Mathematics was utilised in calculated the component values for various analogue circuits particularly in the design of the power amplifier in Appendix H. Mathematics was also used for the configuration and working of the ADC, for setting the appropriate sampling rate, and conversions from the bit numbers to the sensor values. Fundamentals of binary manipulation, conditional and loop constructs are used to generate the codes which is found in Appendix F. The design of each programming solution was generated using algorithmic methods covered in Computer Methods 1. Exit level assessment: Applied engineering knowledge of previous courses to generate the solutions.		
2.2 Applicable principles and laws are applied		
Evidence of competence: Fundamental principles on the analysis of Bipolar Junction Transistors and transistor based amplifier are used for the design of the Power Amplifier in Appendix H. The fundamental problem solving approach of converting the algorithmic flow diagrams to programming code is utilised to generate working codes found in Appendix F. The principles of pulse width modulation (PWM) covered briefly in Analogue Electronics and Power Electronics, as well as PWM generation covered in digital systems is utilized. Exit level assessment: Applied fundamental principles and concepts encountered in previous courses.		
2.3 Appropriate engineering materials, components or processes are selected		
Evidence of competence: Proteus, Atmel and Multisim software used for development of solutions. These softwares were all utilized in previous courses. Sections 11.3.6., 11.4.6., 11.5.6., 11.6.6., Appendix F and Appendix H Exit level assessment: Application of skills and knowledge from previous years.		

2.4 Concepts and ideas are communicated effectively
Evidence of competence: Sensors used to interact with environment communicate effectively and accurately to microcontroller. C programming language used to effectively generate the working programs. Appendix F Exit level assessment: Use of applied engineering knowledge and skills.
2.5 Reasoning about engineering materials, components, systems or processes is performed
Evidence of competence: Programming language used for assimilation with various software and experience. IDE's used for purpose of simulation of microcontroller based designs and PCB implementations (Proteus) and non-microcontroller based designs (Multisim) involving signal analysis. Sections 11.3.6., 11.4.6., 11.5.6., 11.6.6 Exit level assessment: Using appropriate tools and programming language to generate solutions
2.6 Work is performed within the boundaries of the practice area
Evidence of competence: The selected solution adhered to the guidelines of engineering practices defined by IEEE. The solutions contained sub-tasks and processes that are applicable to an engineer and hence conform to the boundaries of the practice area. Exit level assessment: Accomplished all set which simultaneously abiding by engineering principles and professional ethics.
Range Statement: Problems require applications of mathematics, basic sciences and engineering sciences for solving them. There should be appropriate applications of the relevant mathematics, basic sciences and engineering sciences with proper judgement.
Examiners comments and recommended remedial actions if any competence not demonstrated for outcome 2

3 Exit Level Outcome 3: Engineering Design		
Learning outcome: Perform creative, procedural and non-procedural design and synthesis of components, systems, engineering works, products or processes	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **3.1 The design problem is formulated to satisfy user needs, applicable standards, codes of practice and legislation**		
Evidence of competence: The Power Saving, Safety and Isolation system satisfied the varying specifications seen in sections 11.1 and 11.2. The interface is user friendly and easy to use. Exit level assessment: User friendly solution developed.		

3.2 The design process is planned and managed to focus on important issues and recognises and deals with constraints
Evidence of competence: The design process follows all incremental steps to completions, shown in sections 11.3.7., 11.4.7., 11.5.7., 11.7. Exit level assessment: Incremental design process and debugging
3.3 Knowledge, information and resources are acquired and evaluated in order to apply appropriate principles and design tools to provide a workable solution
Evidence of competence: The programs used libraries for mathematic functions and interfacing with the LCD. The system used software such as Proteus and Multisim to assist in testing and debugging. Exit level assessment: Use previously acquired knowledge as well as external tool to generate the solution.
3.4 Design tasks are performed that include component testing to relevant premises, assumptions and constraints
Evidence of competence: Sections 11.3.7., 11.4.7., 11.5.7., 11.6.7. depict implementation of the various sub-systems in isolation and when connected to other sub-systems, as well as display evidence of testing. Exit level assessment: Perform successful individual and intergrated testing
3.5 Alternatives are evaluated for implementation and a preferred solution is selected on an elementary, technical and cost basis
Evidence of competence: Feasibility study undertaken in section 11.7. Evaluations and design decisions also present in sections 11.3.4., 11.4.4., 11.5.4., 11.6.4. Exit level assessment: Feasibility study done to check viability of alternative solutions.
3.6 The design logic and relevant information is communicated in a report
Evidence of competence: Sections 11.3.5., 11.4.5., 11.5.5., 11.6.5. show relevant programming design logic while sections 11.3.4., 11.4.4., 11.5.4., 11.6.4. show relevant physical component choices and outline fundamental working principals in conjunction with sections 11.3.2., 11.4.2., 11.5.2., 11.6.2. Exit level assessment: Showed ability for problem solving and logic
3.7 Occupational health and safety and environmentally related risks are identified and appropriate measures considered
Evidence of competence: Occupational health and safety and environmentally related risks were considered, according to the information obtained in Engineering Business, Engineering Entrepreneurship, Environmental Engineering and Engineering Management & Labour Relations. This concurred with choosing a solution that both satisfied specification and satisfied all health, safety and environmental regulations necessary. Exit level assessment: Adhered to Occupational health and safety and environmental guidelines throughout the design and implementation process.
Range statement:

A major design problem should be used to provide evidence. The problem would be typical of that which the graduate would participate in a typical employment situation shortly after graduation. The selection of components, systems, engineering works, products or processes to be designed is dependent on the discipline.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 3

8 Exit Level Outcome 8: Individual, Team and Multidisciplinary Working

Learning outcome: Demonstrate competence to work effectively as an individual, in teams and in multidisciplinary environments.	Competence Demonstrated	
	Yes	No

Assessment Criteria:
8.1 Demonstrates effective individual work by
(i) identification and focus on objectives
(ii) working strategically
(iii) executing tasks effectively and
(iv) delivery of completed work on time

Evidence of competence:

(i) Demonstrated in sections 11.2.; 11.3.2; 11.4.2; 11.5.2.; 11.6.2; shows identification of functional and non-functional specifications of sub-system as well as separate sub-tasks.
(ii) The generation of sub-sections and the sequential and structured design process allowed strategic completion of solution.
(iii) A paper design was fully presented and
(iv) A time management schedule was set up (Gantt Chart) and followed

Exit level assessment:
Perform individual tasks and adapt to be able to face difficulties

Assessment Criteria:
8.2 Demonstration of effective team work by
(i) making individual contribution to team activity
(ii) performing critical functions
(iii) enhancing work of fellow team members
(iv) benefitting from the support of team members
(v) communicating effectively with team members and
(vi) delivery of completed work on time

Evidence of competence:

(i) Compiling of report and integration of subsystems
(ii) Attending and contributing to meetings with supervisor and domain expert
(iii) Explain integration aspects of various subsystems and contributing to data acquisition for group members
(iv) Mutual motivation present amongst team members
(v) Meeting deadlines for phase reports

Exit level assessment:
Studious time management and always meeting with supervisor or domain expert

Assessment Criteria:
8.3 Demonstration of multidisciplinary work by
(i) acquiring a working knowledge of co-workers' discipline

(ii) using a systems approach and (iii) communication across discipline boundaries
Evidence of competence: (i) Possess an understanding of the domain experts field of choice and his availability. (ii) Keep good communication channels with fellow group members, keeping record of important points in meetings and minutes. (iii) Combination of hardware and software for all microcontroller based circuits. Also digital and analogue combinational circuitry. Exit level assessment: Contributing as a member of the team, function satisfactorily as a team and regarding interdisciplinary fields with high regard.
Range Statement: Multidisciplinary tasks require co-operation across at least one disciplinary boundary. Co-operating disciplines may be engineering disciplines with different fundamental bases other than that of the programme or may be outside engineering.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 8

11 Exit Level Outcome 10: Engineering Professionalism		
Learning outcome: Demonstrate knowledge and understanding of engineering management principles and economic decision-making (focus on economic decision-making only)	Competence Demonstrated	
	Yes	No
Assessment Criteria: Demonstrate competence in: **11.1 Detailed budget from the component level (including labour, space, etc)**		
Evidence of competence: Sections 11.3.3; 11.4.3; 11.5.3.; 11.6.3 and 11.7 showing component pricing and cost related analysis. Exit level assessment: Budget analysis is detailed, concise, and provided for all subsystems		
11.2 Feasibility analysis from a cost perspective from system to component level		
Evidence of competence: Sections 11.3.3; 11.4.3; 11.5.3.; 11.6.3 and 11.7 show detailed costing analysis Exit level assessment: Budget analysis complete		
11.3 Activity-based budgeting and time-phased budgets for cost estimation and control considered		
Evidence of competence: Covered in section 7 Exit level assessment: Acquisition of an economical solution		
11.4 Review cost of the Prototype		

Evidence of competence: Section 11.7 show detailed costing of prototype Exit level assessment: Feasibility studies complete
Range Statement: Basic techniques from economics, business management; project management applied to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
Examiners comments and recommended remedial action if competence not demonstrated for outcome 11