



TECHNOLOGICAL PROJECT REPORT : Class attendance system

Praveen DE SILVA
Georges KHALIL
Tony THEVASURENDRAN

TABLE OF CONTENTS

INTRODUCTION	3
Conception step	4
Functional analysis	4
Hardware development	6
Software development	15
Fabrication step	26
Expectation	26
Final Product	27
Team management	31
CONCLUSION	32
ANNEXES	33
Processing Code	33
Arduino Code	49
Bibliography	52

INTRODUCTION

As the technological project course is one of the main courses from the Master 1 Embedded Systems, students are assigned to a project where they have to program a class attendance system with a microcontroller. In fact, the aim is to apply the theoretical course into practice and to go over the concepts approached during this semester. Every year, the ATMEGA328P from Microchip Technology Inc. is the microcontroller that is used during this course's projects, it is given with a kit. Our group decided to integrate as many additional features as possible because we believe that it could improve the working experience of the final product.

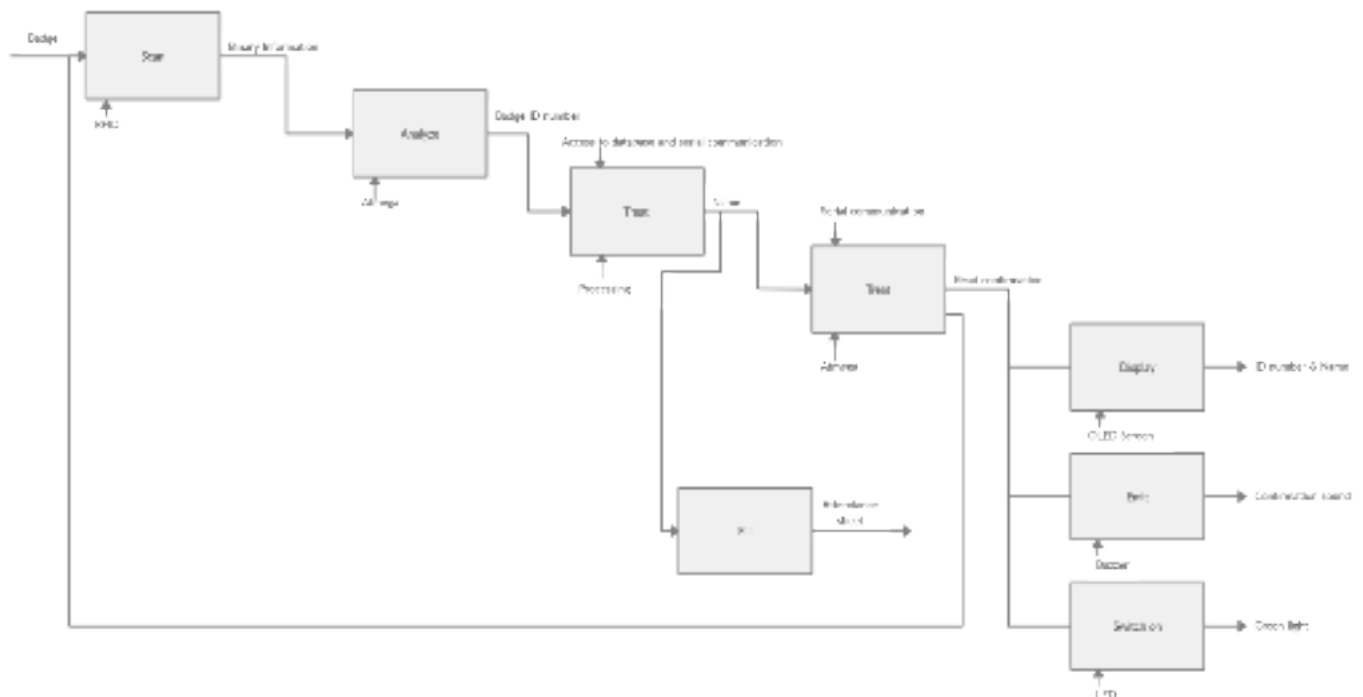
The report has been made as it covers different aspects of the project. Firstly, it is to go through the conception step of the project. Secondly, it is to explain the fabrication step. And finally, it is to see the team management.

Annexes documents are provided at the end of this report. It contains the code lines that have been used during the project and the bibliography.

1. Conception step

a. Functional analysis

Functional diagram :



Technical requirements :

Features :

- Scan a badge card
- Display the student's name and ID
- Register the name and ID on the attendance sheet by serial communication
- Allow the user to modify and/or complete the attendance sheet
- Grant and monitor absence credits to each student

Risks :

- Badge card reading unsuccessful (part reading, badge reading sometimes unsuccessful)
- Access to the database failure
- Wire disconnected/broken

- Fragile equipments

Dependencies :

- WiFi Network
- Operating Database Server
- Code compilation on Arduino IDE/Processing
- Wire and modules well connected
- User

Expected results and test process :

Our aim in this project is to build a student card recognition system. Therefore, we are using a breadboard where our circuit is built and connected to a GUI through a USB.

The first step would be to pass the student card in front of the RFID module in order to add the student into the attendance list.

While the card is passed, it's important to check whether the buffer and the LEDs are working. If the student card is valid, the green LED should turn on and the buffer should make a high pitch noise and if the card isn't valid, therefore, the buzzer should emit more of a low pitch tone and the red LED should turn on.

The second step is to check if the GLCD is working properly. If we pass a valid student card in front of the RFID, we should be able to read the student's name, surname, their card's identification number and the number of remaining absence credits in the GLCD.

We have uploaded the code from the Arduino IDE into our circuit which enables us to read the identification of the student card in the Serial port.

On the other hand, we have also created a program in Processing in order to get the Serial value received by passing the card.

Processing uses a simplified version of Java, therefore it gives us the possibility to connect our system with a database. We decided to use the database MySQL on an internal server where we have registered every student's name, surname, student card's identification, number of credits remaining and their status (present = 1 or absent = 0). The status is set to 0 at the start of each course.

So, the third step in the testing process is to check whether the status is updated to 1 when the card is passed. If it's the case, in the GUI, the "ABSENT" should change into "PRESENT" in the attendance list and the GLCD should display the student's name, surname, card's identification and the number of remaining credits.

At the end of each class, the user clicks on “Submit” and therefore it should generate a new text file where the attendance sheet of that specific class is saved in the folder “data”. We should be able to then display it on the home menu by clicking on the specific course.

So, the final test is to check whether the attendance sheet is saved and can be displayed properly.

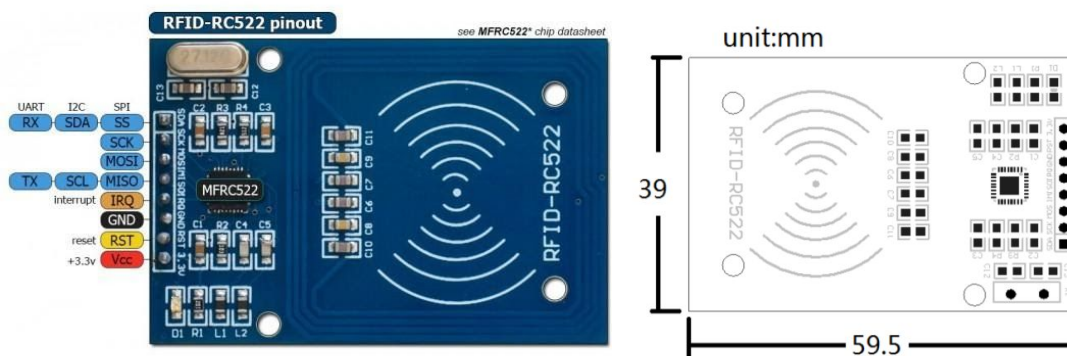
b. Hardware development

Presentation of the components :

This section describes every component used in the project. It includes:

- Component Picture
- Component Dimensions
- Most notable specifications (only those relevant to our project)
- General description of component functioning (does not include code or wiring diagrams)

❖ RFID RC522 MODULE



Notable Specs :

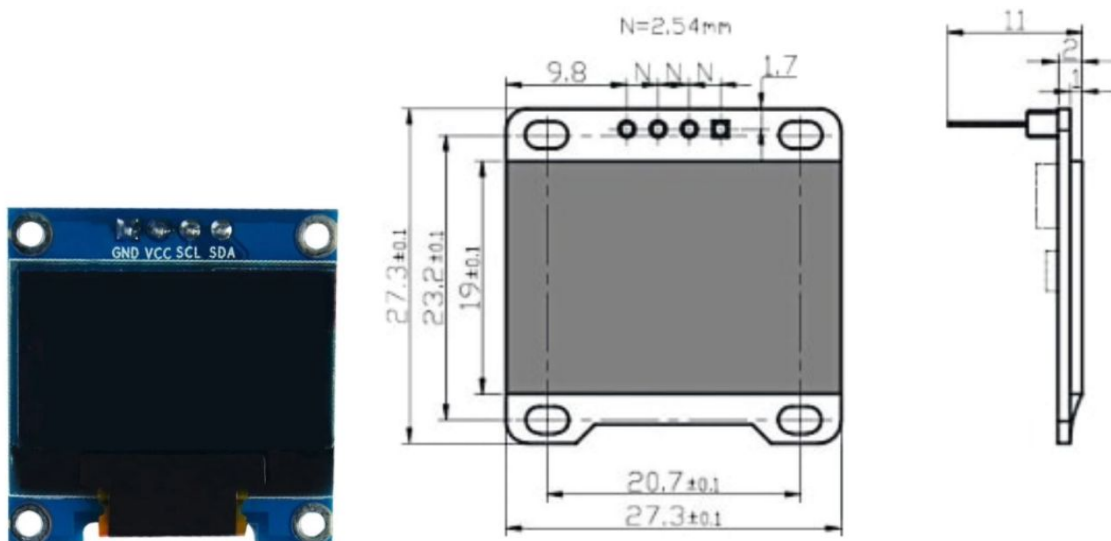
- 13.56MHz RFID module
- Operating voltage: 2.5V to 3.3V
- Communication: SPI, I2C, UART (SPI used in this project)
- Maximum Data Rate: 10Mbps
- Read Range: 5cm
- Current Consumption: 13-26mA
- Power down mode consumption: 10uA (min)

Component Functioning :

The RC522 will be supplied with a **3V3 supply voltage**. The module will read information encoded on a tag (when this tag is in proximity). The RC522 generates an electromagnetic field which causes electrons to move through the tag/card's antenna and subsequently power the chip inside it. Once powered, the chip will send information back to the module in the form of another radio (RF) signal.

This signal will be interpreted (extracting a unique ID number) and communicated to the Atmega328P using **SPI communication** and through the **MISO & MOSI pins**.

❖ Adafruit SSD1306 OLED Display



Notable Specs :

- Monochrome 4-pin SSD1306 0.96" OLED display.
- Pixel Resolution: 128×64
- Supply voltage: 3V – 5V
- Communication: I2C

Component Functioning :

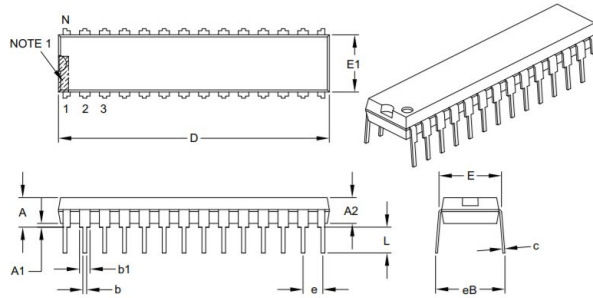
The OLED will be powered by a **5V supply voltage**. Communication with the Atmega328P will be via **I2C communication** and through the **SDA/SCL pins**.

It will be used to display student names and IDs. Display font, size, color, and other specifications are defined in the Atmega328P firmware code.

◆ ATMEGA328P

28-Lead Skinny Plastic Dual In-Line (SP) – 300 mil Body [SPDIP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Dimension Limits	Units	INCHES		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	.100 BSC		
Top to Seating Plane	A	—	—	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	—	—
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	—	—	.430

Notable Specs :

- Supply Voltage: 2.7V to 5.5V
- Speed Grade: 0 to 16MHz at 4.5 to 5.5V
- Peripheral features:
- 23 programmable I/O lines
- Programmable serial USART
- Master/slave SPI serial interface
- Byte-oriented 2-wire serial interface (I2C)

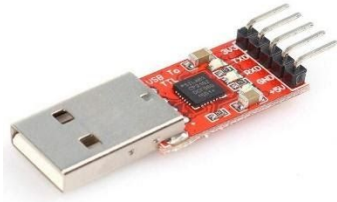
Component Functioning :

The Atmega328P will communicate with:

- RFID RC522 Module via SPI
- Adafruit SSD1306 OLED Display via I2C
- USB to TTL adapter via UART communication (TX/RX lines)
- ESP8266 via UART communication (TX/RX lines)

The Atmega328P will collect data sent from the RFID module and process it (get a clean ID number). This ID will be sent to the ESP8266 module for processing/checking. The ESP8266 will send back information (ID, student name etc.) and the Atmega328P will translate this information into commands for the OLED display and GPIO pins (LED, Buzzer etc.). A more detailed description of these processes will be given later.

❖ USB to TTL Adapter Based on CP2102 chip



Notable Specs :

- Standard USB type A male and TTL 5pin connector: TXD, RXD, GND, 3.3V, 5V.
- Baud rates: 300 bps to 1.5 Mbps

Component Functioning :

The USB to TTL adapter will be used to upload the code/firmware to the Atmega328P, and to supply the 5V, 3V3, GND power source to the components.

❖ ESP8266

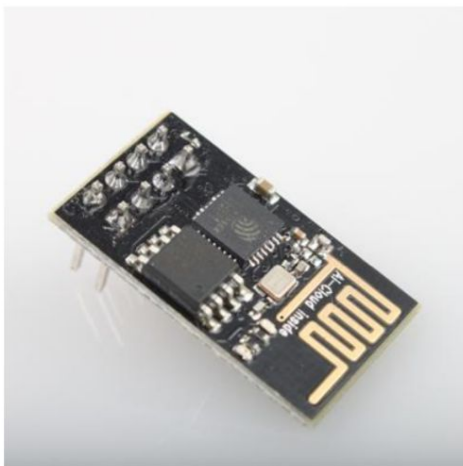


Figure 3 [Module Pin Counts, 8 pin, 14.3 mm *24.8 mm *3.0 mm]

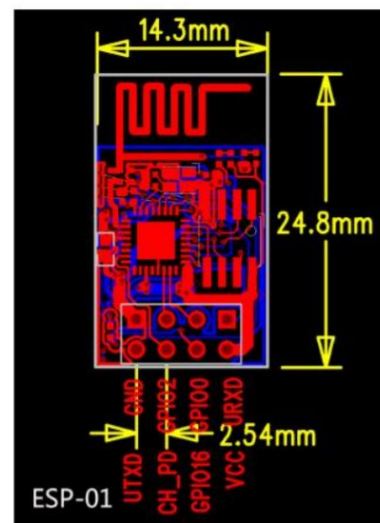


Figure 4 Top View of ESP-01 WiFi Module

Notable Specs :

- Communication : UART/TTL
- Operating Voltage: 3.3V
- Operating Current: Average value 80/100 mA / Max Value 170 mA
- Peripherals: 2 GPIO pins, 3.6V (max), 12 mA (max)

Component Functioning :

The ESP8266 will communicate with Atmega328P using UART communication and through the TX/RX lines. It is programmed to transmit this received information to our database (through Wi-Fi) and send the received data back to the Atmega328P.

- ❖ **Capacitor**
- ❖ **LED**
- ❖ **BuzzerPS1420P02CT**
- ❖ **Pushbutton**
- ❖ **Resistor**
- ❖ **XTAL**

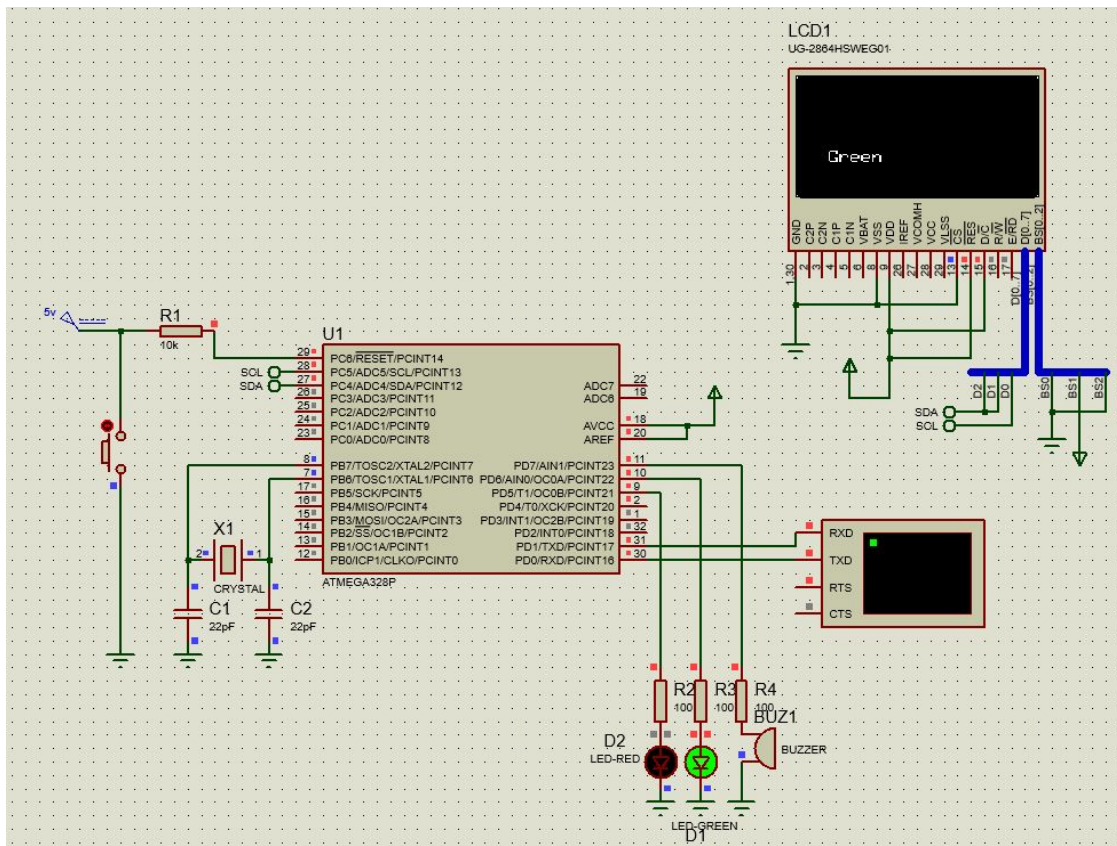
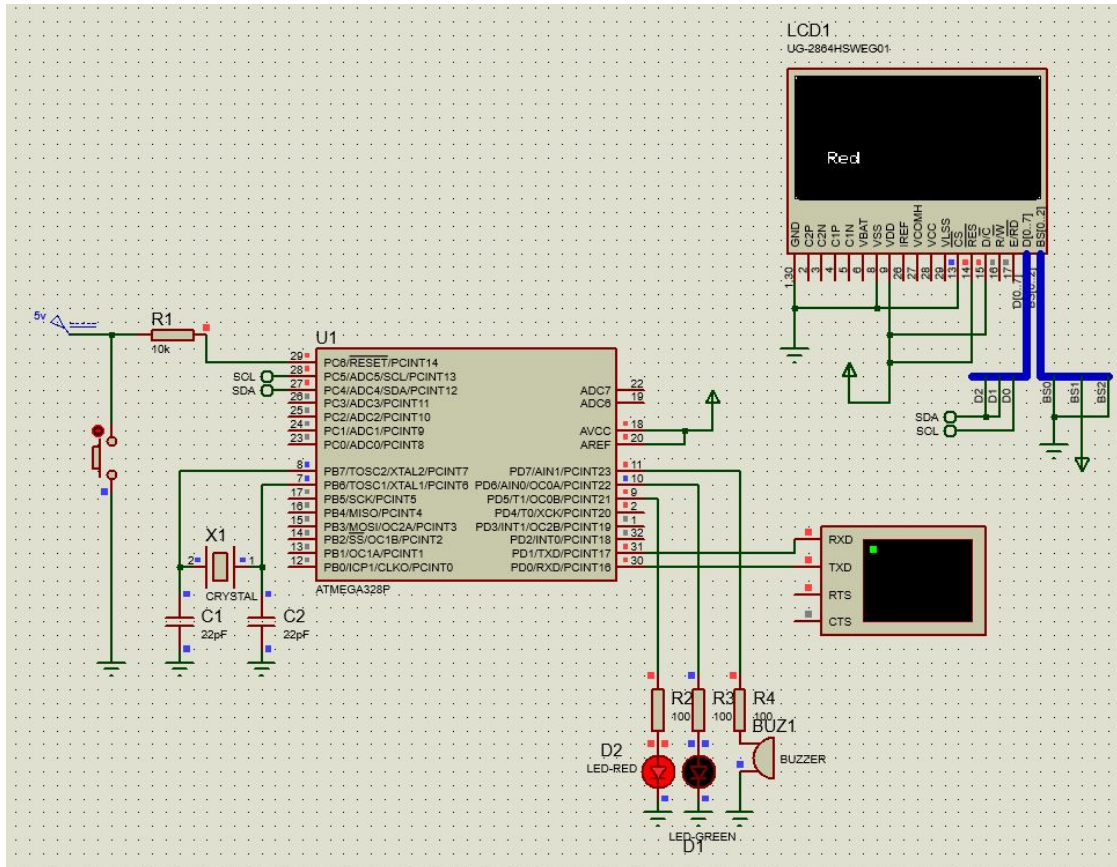
Simulation on Proteus :

Proteus simulation:

We simulated our circuit on Proteus to see if the wiring, and the code, that we decided to go with was adequate. We simulated the OLED display, and the GPIO pins with the LEDs and the Buzzer. We could not simulate the RFID because there were no similar part/libraries in Proteus. All other components are the same, as well as the libraries we used in the code.

For our simulation we decided to write a code that turns on one of the LEDs, and writes on the OLED screen the color of that LED. So when the Red LED is on, we can read on the screen the word "Red". Also we are also sending a signal to the buzzer: we can see this because the signal square of the buzzer is lighting up (red colored square above the buzzer object).

The screenshots below show the 2 different states of the LEDs and the short code we used:



```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
#define OLED_RESET      4 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);

#define LED_RED          5           // LED RED connected to ATMEGA328P
pin 5
#define LED_GREEN        6           // LED GREEN connected to ATMEGA328P
pin 6
#define buzzer            7           // Buzzer connected to ATMEGA328P
pin 7

void setup() {
  Serial.begin(9600);

  // SSD1306_SWITCHCAPVCC = generate display voltage from 3.3V
  internally
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3D)) { // Address 0x3D for
128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever
  }
}

void loop() {

  display.clearDisplay(); // Clear display buffer
  display.display();

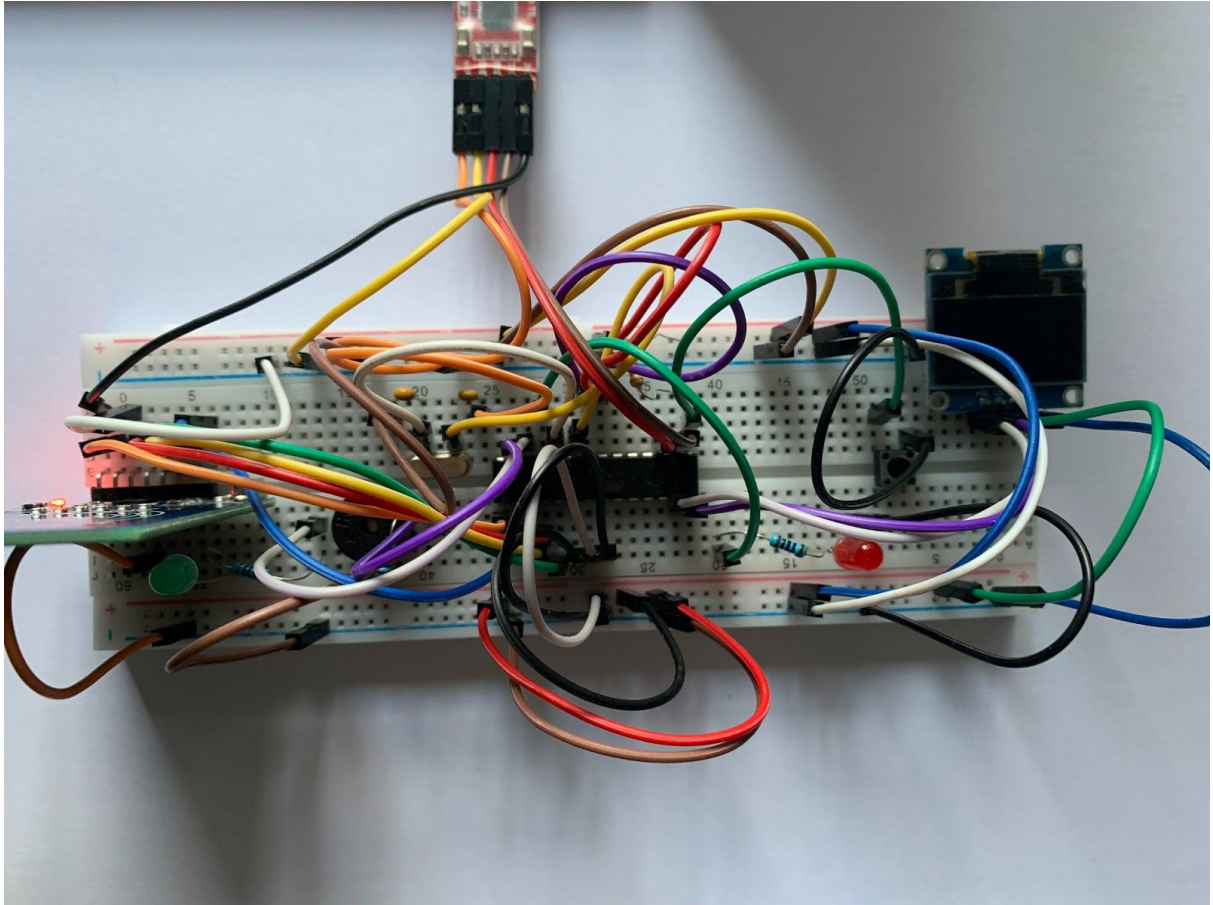
  Serial.print("hello");
  //delay(500);

  // Write "Red" an turn on Red LED
  display.setCursor(15, 40);
  display.setTextSize(1); // Normal 1:1 pixel scale
  display.setTextColor(SSD1306_WHITE); // Draw white text
```

```
display.print("Red");           // Print "Red" on OLED
display.display();
digitalWrite(LED_RED, HIGH);    // Turn the Red LED on
tone(buzzer, 2000);             // Turn on Buzzer at 2kHz
delay(50);
digitalWrite(LED_RED, LOW);     // Turn the Red LED off
noTone(buzzer);

// Write "Green" and turn on Green LED
display.clearDisplay();
display.setCursor(15, 40);
display.setTextSize(1);         // Normal 1:1 pixel scale
display.setTextColor(SSD1306_WHITE); // Draw white text
display.print("Green");         // Print "Green" on OLED
display.display();
tone(buzzer, 500);              // Send 500Hz sound
signal...
digitalWrite(LED_GREEN, HIGH);  // Turn the Green LED on
delay(50);                      //
noTone(buzzer);                 // Turn Off buzzer
digitalWrite(LED_GREEN, LOW);   // Turn the Green LED off
}
```


Labdeck prototype :



c. Software development

Microcontroller programming and program flowchart (commented code in Annexe, explanation of the code in the report) :

Libraries and Main Functions used:

For our project we download the 2 libraries: Adafruit SSD1306, which is the library of the OLED display screen we are using, and the MFRC522 library which contains functions and commands for the RFID module.

Adafruit SSD1306 by Adafruit Version 2.4.0 **INSTALLED**
 SSD1306 oled driver library for monochrome 128x64 and 128x32 displays
[More info](#)

MFRC522 by GithubCommunity Version 1.4.7 **INSTALLED**
 Arduino RFID Library for MFRC522 (SPI) Read/Write a RFID Card or Tag using the ISO/IEC 14443A/MIFARE interface.
[More info](#)
 Select version

Below is a list of all libraries used and the functions that are called:

<Adafruit_GFX.h> Library used to set up display settings (font, position etc.)

- display.setTextSize(1);
- display.setTextColor(WHITE);
- display.setCursor(15, 20);
- display.print(string); Upload the string to the screen (it does not actually print the string)
- display.display(); Used to update the display with the last settings

<Adafruit_SSD1306.h>

- if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) Used to verify the allocation to the OLED address, if the address is different from the entered values (in this case 0x3C) it returns an error message, so we know we have to change the given address.
- Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1) Declare that an SSD1306 display is connected to I2C (SDA, SCL pins)

<SPI.h>

- SPI.begin(); used to initialize the SPI bus

<MFRC522.h>

- MFRC522 mfrc522(SS_PIN, RST_PIN) Create MFRC522 instance
- mfrc522.PCD_Init Used to initialize the MFRC522 module
- mfrc522.PICC_IsNewCardPresent() Returns 1 if new card is available
- mfrc522.PICC_HaltA() Stop reading information from cards. This is used for registering the ID in a variable.

- `mfr522.uid.uidByte[i]` Get a single byte of the ID. "i" determines the position of this byte.

<Wire.h>

Code Explanation:

We will divide the code into functional blocks and explain each one of them:

- Look for a card/tag
- Read ID, Convert it to string and Send it through the Serial Port.
- Read responses from Processing, and take actions based on this response.

Looking for a card/tag:

Using the `mfr522.PICC_IsNewCardPresent()` function, we loop until we detect a new card in proximity to the RFID module. This function is available in the MFRC522 library.

Reading ID from card:

Using `mfr522.uid.uidByte[i]` we read each byte of the UID and store it in a "unsigned long array" variable. We know that all the IDs have 4 bytes. Therefore, we created a loop that goes over the 4 bytes of the ID and stores them in this array. Then, to be able to send clean data to the Processing program, we convert each element of this array to a string (with a loop that goes over the 4 elements) and store them in a new "String" variable. `Converted_UID += String(hex_num[i],HEX)` : this code segment, if used in a loop, takes each element of the `hex_num` variable (ID), converts it to a string, and concatenates it with the previous element. What we end up with is a string with the 4 bytes of the ID attached (and still in Hexadecimal format) that we can send through the serial port using the `Serial.print()` function.

Read response from Processing, and take actions based on this response:

We read the response given through Serial Port using `Serial.readString()`. Everything that happens between the last step and this one is done on Processing, the database and the user interface we created. We store the message and we get a "String" variable. We can receive 2 types of response from the Processing code: a message indicating that we did not find a match for ID in the database (or that it was not collected correctly, which is equivalent). The other message would be the name of the student. So, from a software perspective, we can define two cases: rejection message or student name. So we set up a conditional statement: if we get the rejection message, we turn on the red LED and print the rejection message on the screen (printing process detailed below). On the other hand, if we get a student name (which is equivalent to anything other than the rejection message from a software point of view), we turn on the green LED and print the student name on the screen along with the ID collected before.

The printing process is done in the following succession:

Choose text settings:


```
display.setTextSize(2);  
display.setTextColor(WHITE);  
display.setCursor(15, 20);
```

Send to the OLED the data to be printed (nothing is printed yet)

```
display.print(rejection_message);
```

Command the OLED to update and print the data given to it:

```
display.display();
```

Same process to clear the screen:

```
display.clearDisplay();  
display.display();
```

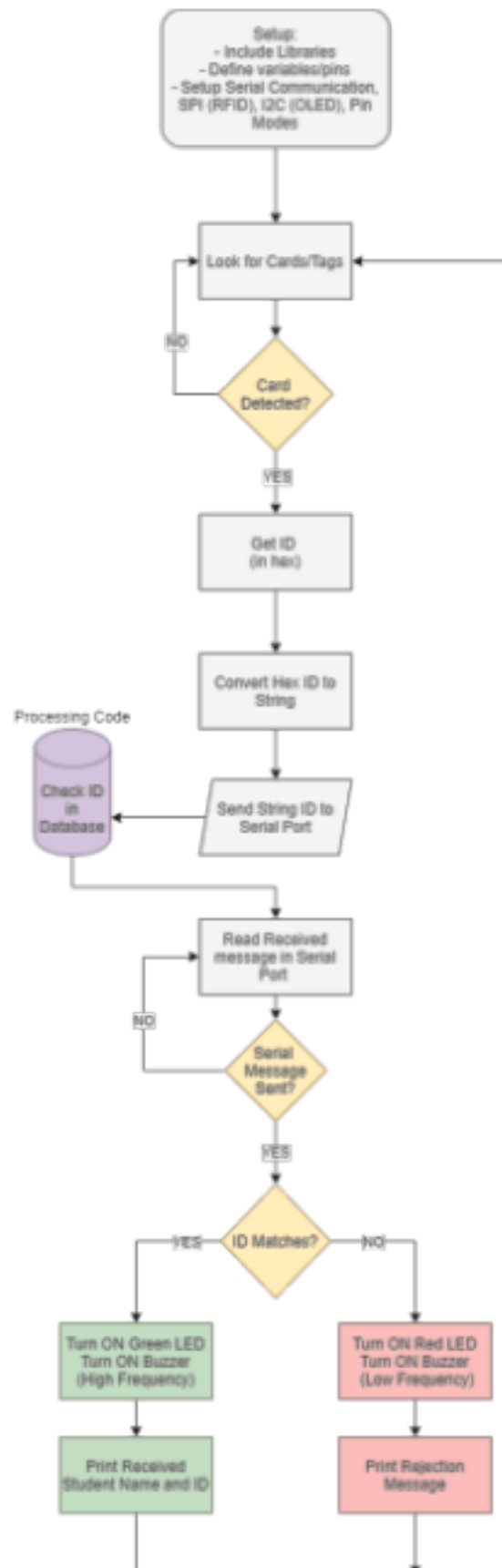
Limitations:

From a software point of view, we could have created more functions that represent code segments, this way we would have a cleaner and more readable code.

We also encountered difficulties with the serial communication between the ATMEGA328 and the Processing IDE. Sometimes the ID we send through the serial port is not collected completely and is instead broken into 2 parts that are read separately. This means that sometimes the user would get a rejection message randomly, even if his card is registered in the database. It is not a big issue because the card would still be read correctly at the next try. But this problem is still happening randomly, and we could not point out the origin of the issue. We did check the Baud Rate and other communication settings, but it was inconclusive. Maybe with more time we would have figured out a way to solve it.

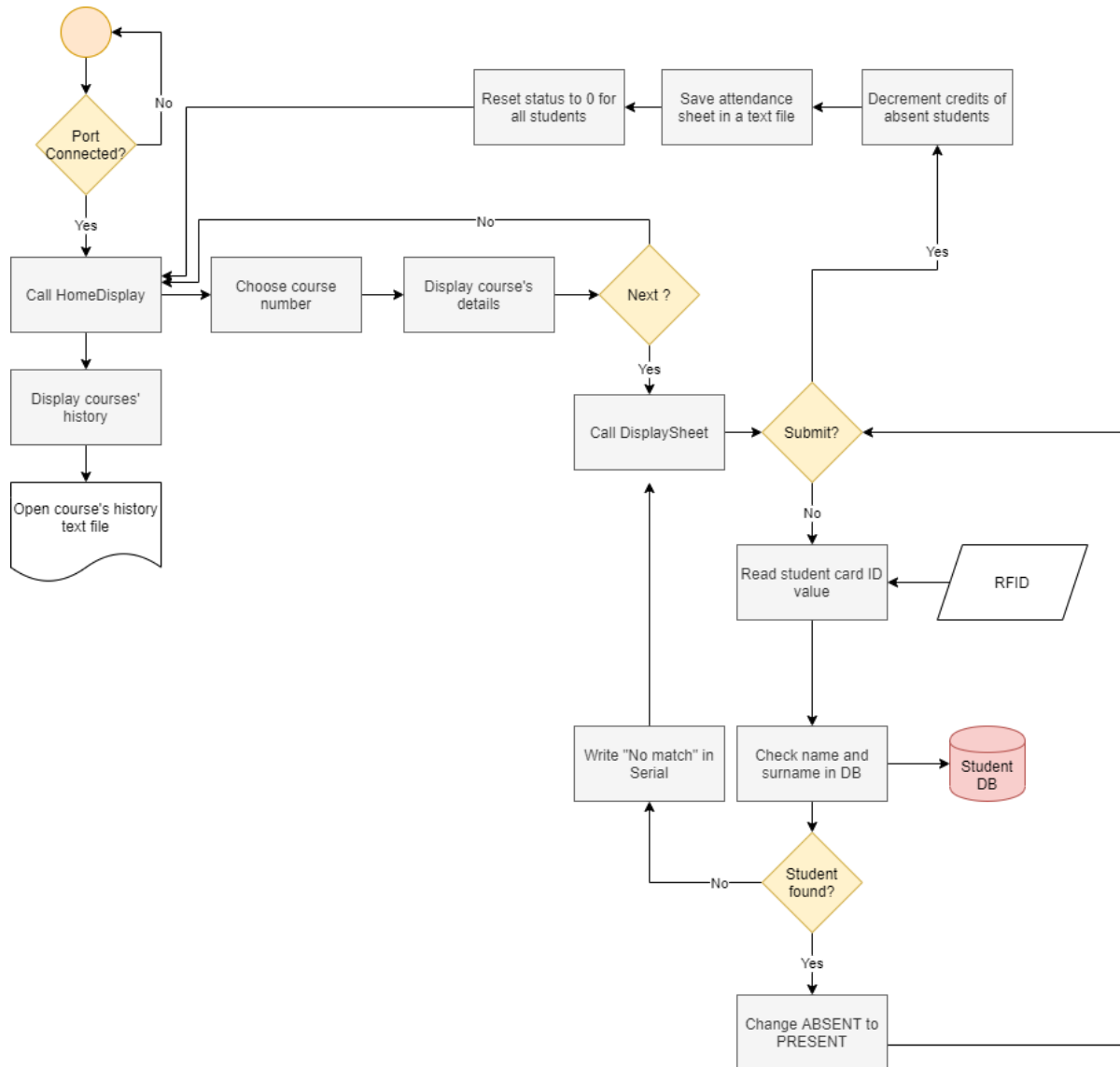
Finally, we could have added more features to our project. Notably the Wi-fi communication using the ESP8266.

Arduino code's flowchart



Processing Interface programming and program flowchart :

Processing code's flowchart



Processing code explanation

The Processing code is composed of 7 different files: Main, MainGUI, Attendance, DetailCourse, Includes, SaveTxt and Button.

File Includes

All the libraries and variables are declared in this file. We are using 4 libraries which are

processing.serial: serial connection library

de.bezier.data.sql: database library

java.io.FileWriter: writing in a text file library

ui booster: GUI library which we use to create a combo box in the program

File Main

void Settings()

It declares the size of the window display using the function `size(width, height)`.

void Setup()

Declare and establish a connection with the Serial port and then call on “homeDisplay” function which displays the home menu.

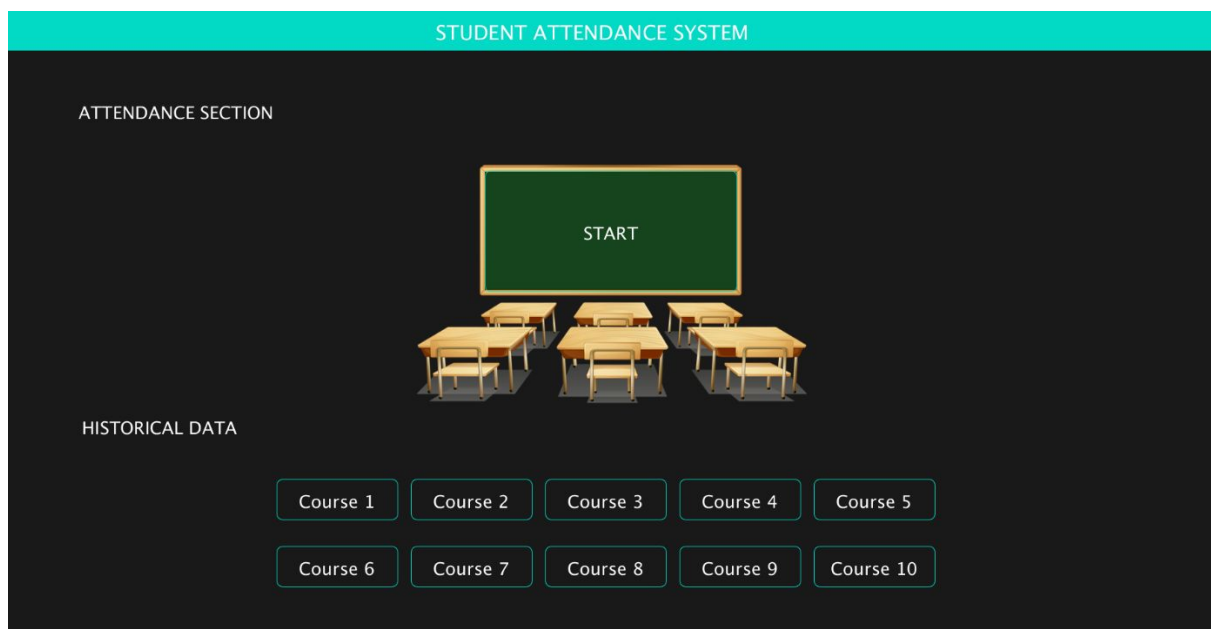
void draw()

A loop where we read the RFID value from Serial and compare it with the database.

File MainGUI

In this file, we find the code of the first page of the GUI.

It is composed of 3 parts. First, the title of the system, then in the middle, the “START” button which enables us to start the attendance sheet and then below, there are 10 buttons in which we can have access to historical data of previous courses.



File Button

Class Button

The function Button has a class with purpose to build a button from scratch without using any libraries.

The class is composed of three functions:

Button(String labelB, float xpos, float ypos, float widthB, float heightB)

The constructor in which the position x, position y, width and height are initialized.

void Draw()

It allows us to draw a rectangle with a certain background colour, a border colour and a text inside.

boolean MouseIsOver()

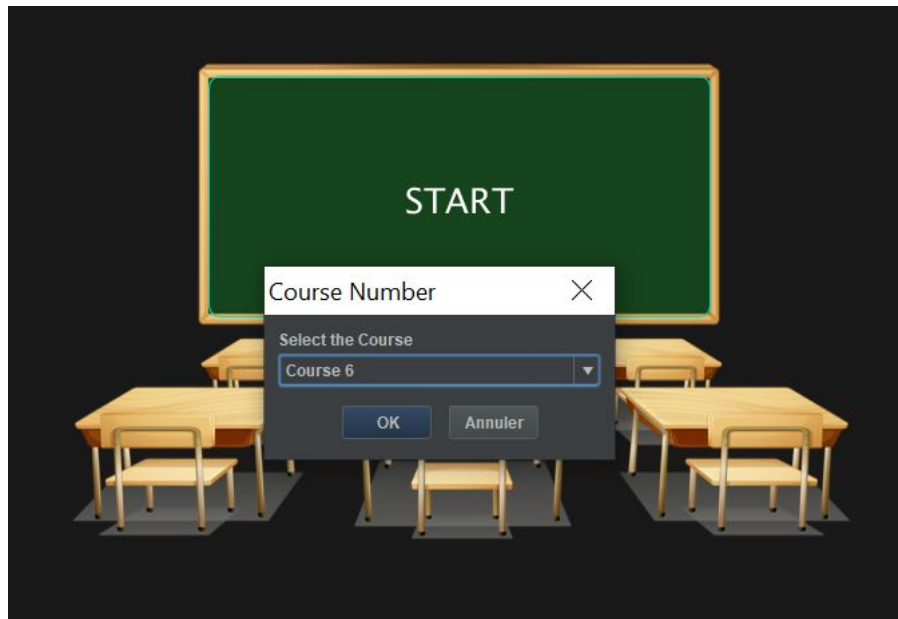
This function's use is to give instructions while a button is pressed.

We used this class to draw every button in this project.

void mousePressed()

Here, we give instructions at every time a button has been pressed.

To start the digital attendance sheet, we click on the button "START" which opens a combo box that we built using the library uibooster. Here, we can select the number of the course.

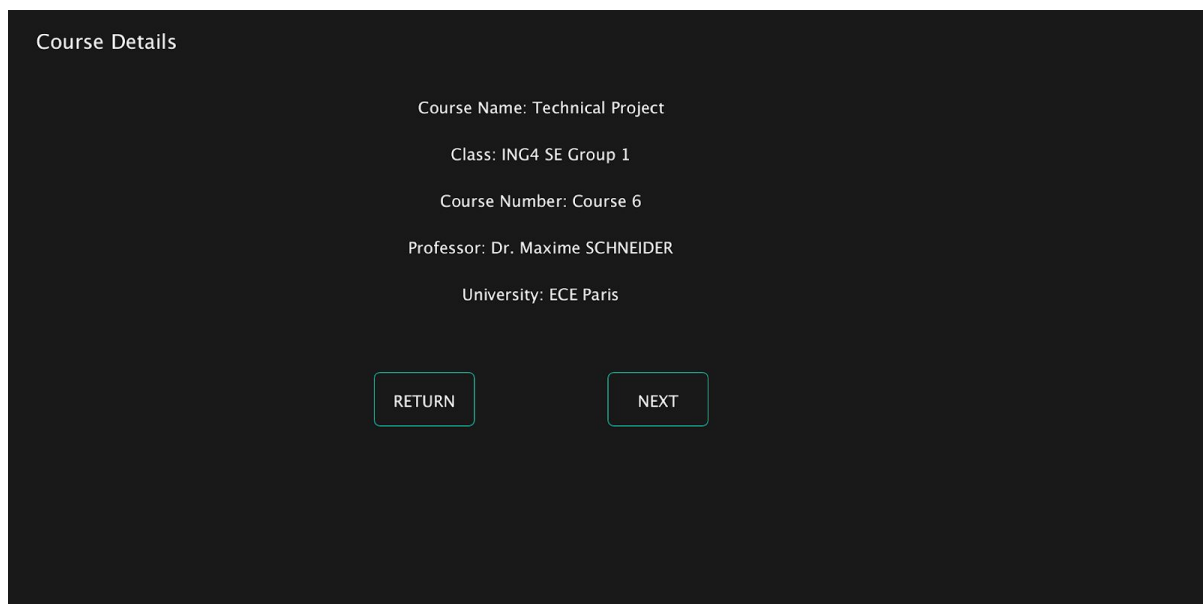


After selecting the course and pressing on “OK”, we redirect to the page “Course Details”.

File DetailCourse

void detailCourse()

In this function, we display different details of the course such as the course name, the class, the course number, the Professor’s name and then the University.



Course Details

Course Name: Technical Project

Class: ING4 SE Group 1

Course Number: Course 6

Professor: Dr. Maxime SCHNEIDER

University: ECE Paris

RETURN NEXT

By clicking on the button “RETURN”, we go back into the home page. But, by clicking on the “NEXT” button, we redirect into the digital attendance sheet.

File Attendance

In this file, we have 4 functions:

void sqlIdentification(String val)

This function allows us to compare the value read by RFID and with the identification values saved in the database. If there is a match in the identification, we select the surname, the name and the number of remaining credits of that student and write them in the Serial which is then read by the Arduino IDE and displayed in the GLCD.

void displaySheet()

It allows us to display the attendance list of the whole class with surnames, names, remaining credits and their status (Absent or Present).

ATTENDANCE SHEET				Course 6	
Surname	Name	Remaining credits	Status		
ARNAL	Melchior	9	ABSENT	RETURN	SUBMIT
BAZART	Clement	9	ABSENT		
BLOCK DE FRIBERG	Emmanuel	9	ABSENT	RETURN	SUBMIT
CASAGRANDE	Jean	9	ABSENT		
CHEN	Pascal	9	ABSENT	RETURN	SUBMIT
CRENO	Danny	9	ABSENT		
DE SILVA	Praveen	8	ABSENT	RETURN	SUBMIT
DELAHEGUE	Emilien	9	ABSENT		
KAZOUINI	Mohamed-Amine	12	ABSENT	RETURN	SUBMIT
KHALIL	Georges	4	ABSENT		
LANVERT	Olivier	9	ABSENT	RETURN	SUBMIT
LAVILLAT	Bastien	9	ABSENT		
LEBLANC	Adrien	9	ABSENT	RETURN	SUBMIT
LEFEBVRE	Faustine	9	ABSENT		
MARBEUF	Vincent	9	ABSENT	RETURN	SUBMIT
PIERRE	Louis	9	ABSENT		
PUYOU-LASCASSIES	Martin	9	ABSENT	RETURN	SUBMIT
RONDOT	Paolo	9	ABSENT		
TEMPESTA	Claudia	9	ABSENT	RETURN	SUBMIT
THEVASURENDRAN	Tony	-4	ABSENT		
VIEVILLE	Clement	9	ABSENT	RETURN	SUBMIT
WISWESSER	Miriam	9	ABSENT		

The same function is used to change the status when the student card is recognized by the database. Therefore, it changes from “ABSENT” to “PRESENT”, as you can see in the image below.

ATTENDANCE SHEET

Course 6

Surname	Name	Remaining credits	Status
ARNAL	Melchior	9	ABSENT
BAZART	Clement	9	ABSENT
BLOCK DE FRIBERG	Emmanuel	9	ABSENT
CASAGRANDE	Jean	9	ABSENT
CHEN	Pascal	9	ABSENT
CRENO	Danny	9	ABSENT
DE SILVA	Praveen	8	PRESENT
DELAHEGUE	Emilien	9	ABSENT
KAZOUINI	Mohamed-Amine	12	ABSENT
KHALIL	Georges	4	PRESENT
LANVERT	Olivier	9	ABSENT
LAVILLAT	Bastien	9	ABSENT
LEBLANC	Adrien	9	ABSENT
LEFEBVRE	Faustine	9	ABSENT
MARBEUF	Vincent	9	ABSENT
PIERRE	Louis	9	ABSENT
PUYOU-LASCASSIES	Martin	9	ABSENT
RONDOT	Paolo	9	ABSENT
TEMPESTA	Claudia	9	ABSENT
THEVASURENDRAN	Tony	-4	ABSENT
VIEVILLE	Clement	9	ABSENT
WISWESSER	Miriam	9	ABSENT

RETURN

SUBMIT

At the end of the course, the user can click on “SUBMIT” in order to finish the course. By doing so, we call the third function “decrementCredits”

```
void decrementCredits()
```

It enables us to decrement the number of remaining absence credits for each student when they are absent.

```
void resetStatus()
```

This function is used to change the status to 0 (absent) for each student at the beginning of the next course.

In the photo below, we can see that the number of remaining credits has decreased by one for the students who've been absent in the previous course.

ATTENDANCE SHEET

Course 7

Surname	Name	Remaining credits	Status
ARNAL	Melchior	8	ABSENT
BAZART	Clement	8	ABSENT
BLOCK DE FRIBERG	Emmanuel	8	ABSENT
CASAGRANDE	Jean	8	ABSENT
CHEN	Pascal	8	ABSENT
CRENO	Danny	8	ABSENT
DE SILVA	Praveen	8	ABSENT
DELAHEGUE	Emilien	8	ABSENT
KAZOUINI	Mohamed-Amine	11	ABSENT
KHALIL	Georges	4	ABSENT
LANVERT	Olivier	8	ABSENT
LAVILLAT	Bastien	8	ABSENT
LEBLANC	Adrien	8	ABSENT
LEFEBVRE	Faustine	8	ABSENT
MARBEUF	Vincent	8	ABSENT
PIERRE	Louis	8	ABSENT
PUYOU-LASCASSIES	Martin	8	ABSENT
RONDOT	Paolo	8	ABSENT
TEMPESTA	Claudia	8	ABSENT
THEVASURENDRAN	Tony	-5	ABSENT
VIEVILLE	Clement	8	ABSENT
WISWESSER	Miriam	8	ABSENT

RETURN

SUBMIT

But before calling on the function “resetStatus”, we should call on “saveAttendanceSheet” in order to save the attendance sheet in a text file with the course number.

File SaveTxt

In this file, there are 3 functions:

void saveAttendanceSheet ()

This function allows us to save the names and the surnames of the students being present and absent in each course.

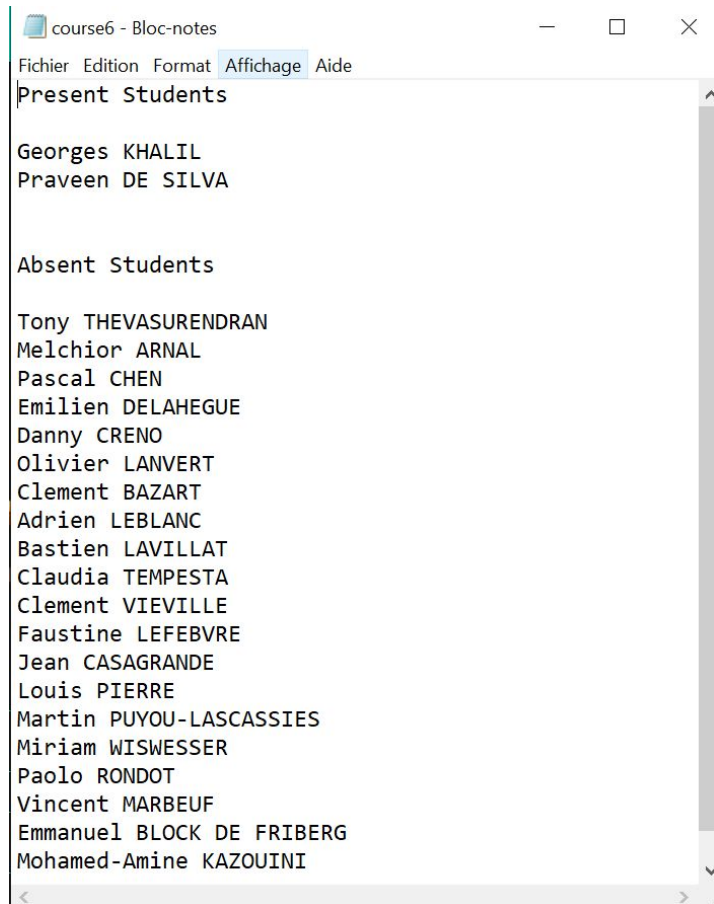
void absentStudentSql()

It's used to select the names and the surnames of absent students.

void presentStudentSql()

It's used to select the names and the surnames of present students.

On the home page, by clicking on the button with the specific course, we launch the text file with the names of present and absent students of that course.



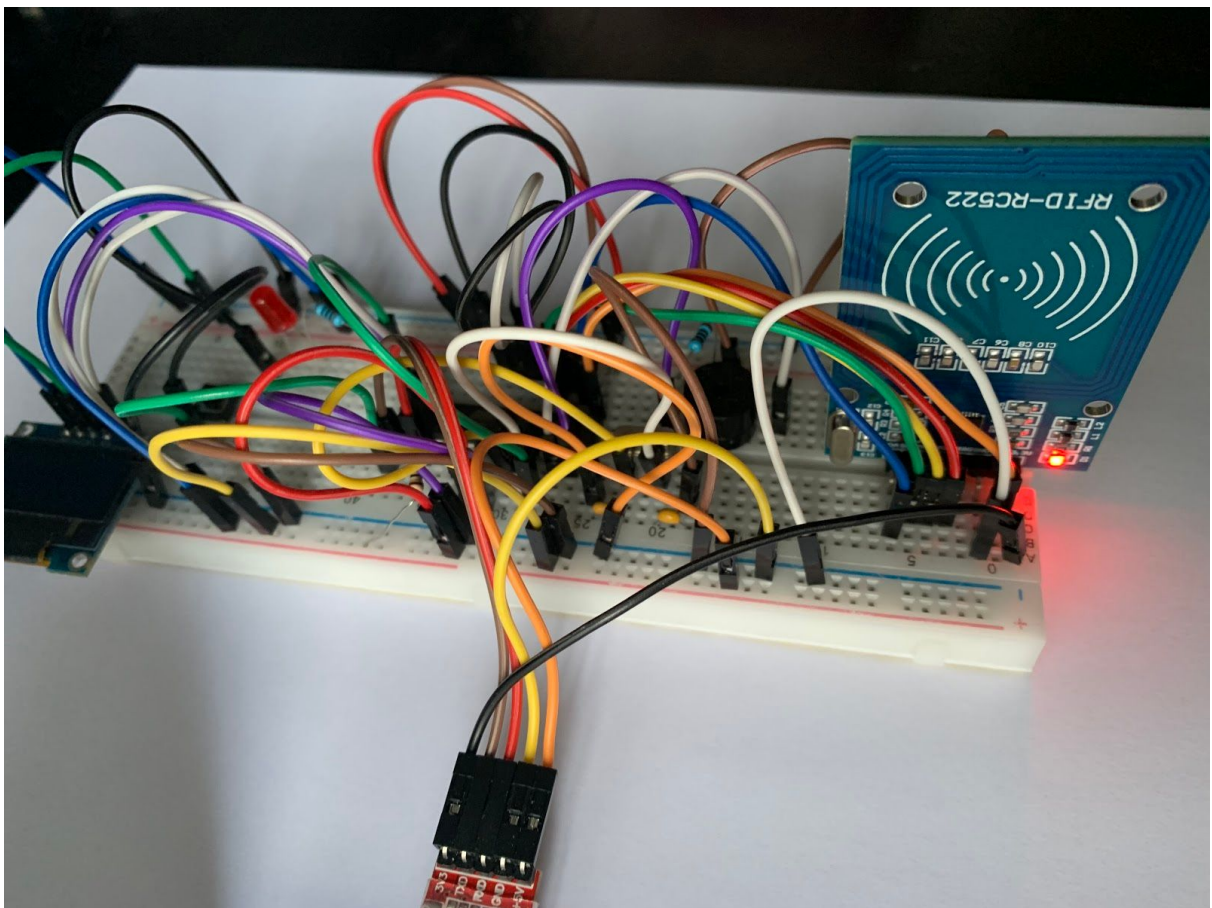
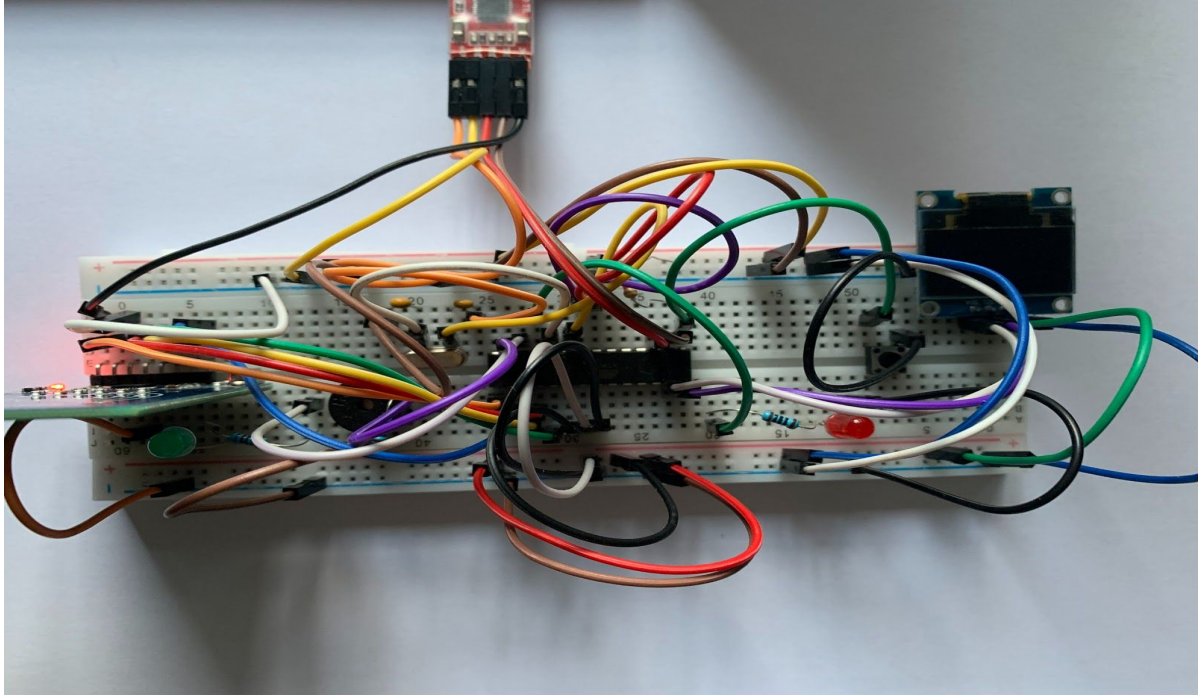
2. Fabrication step

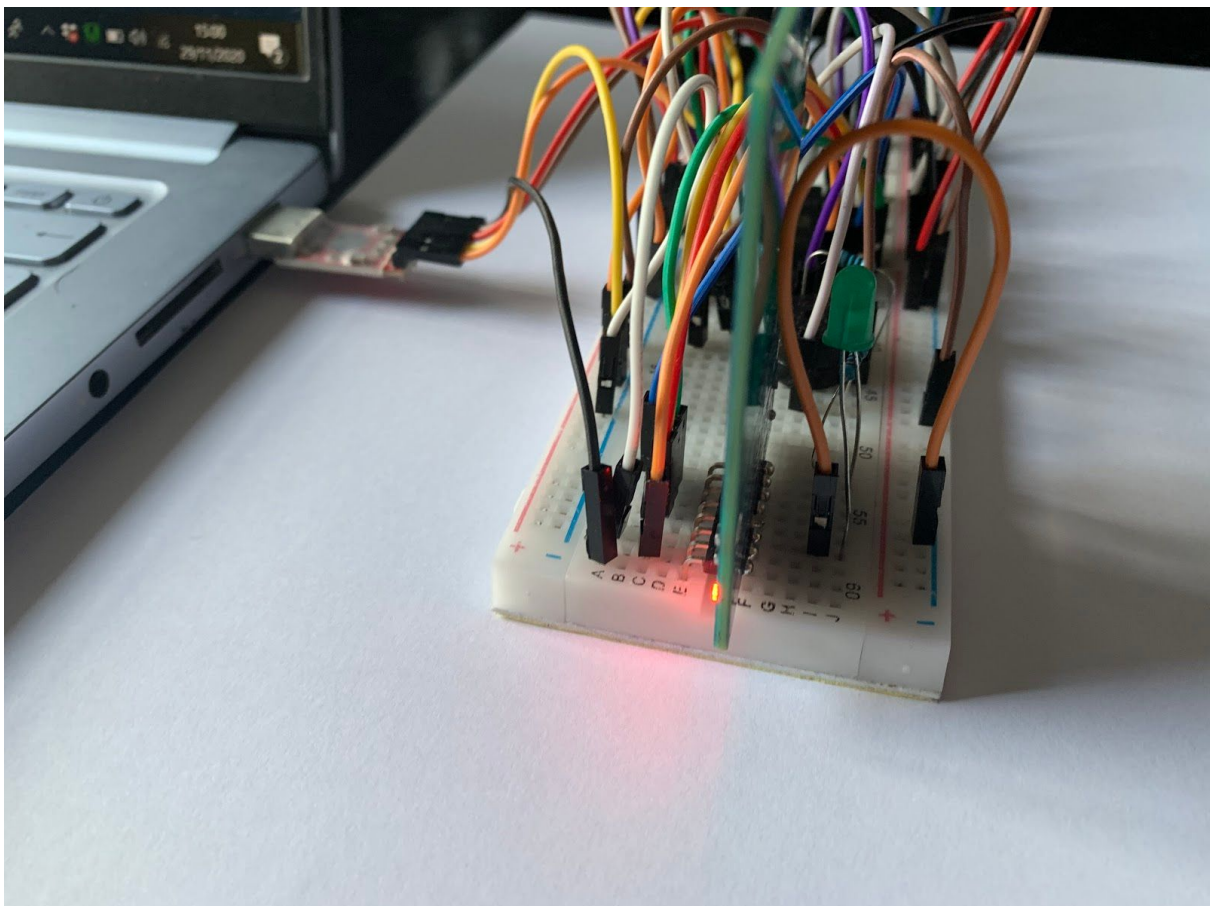
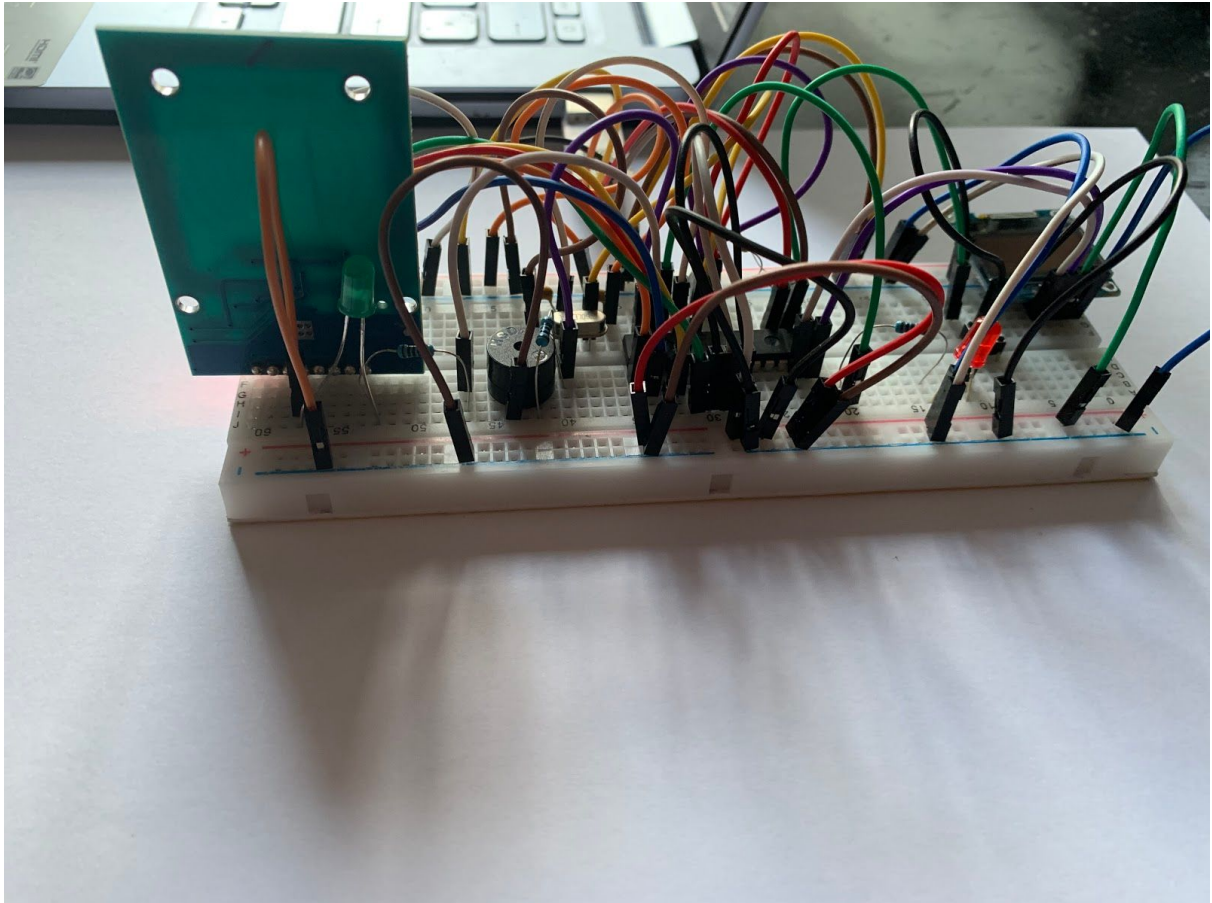
a. **Expectation**

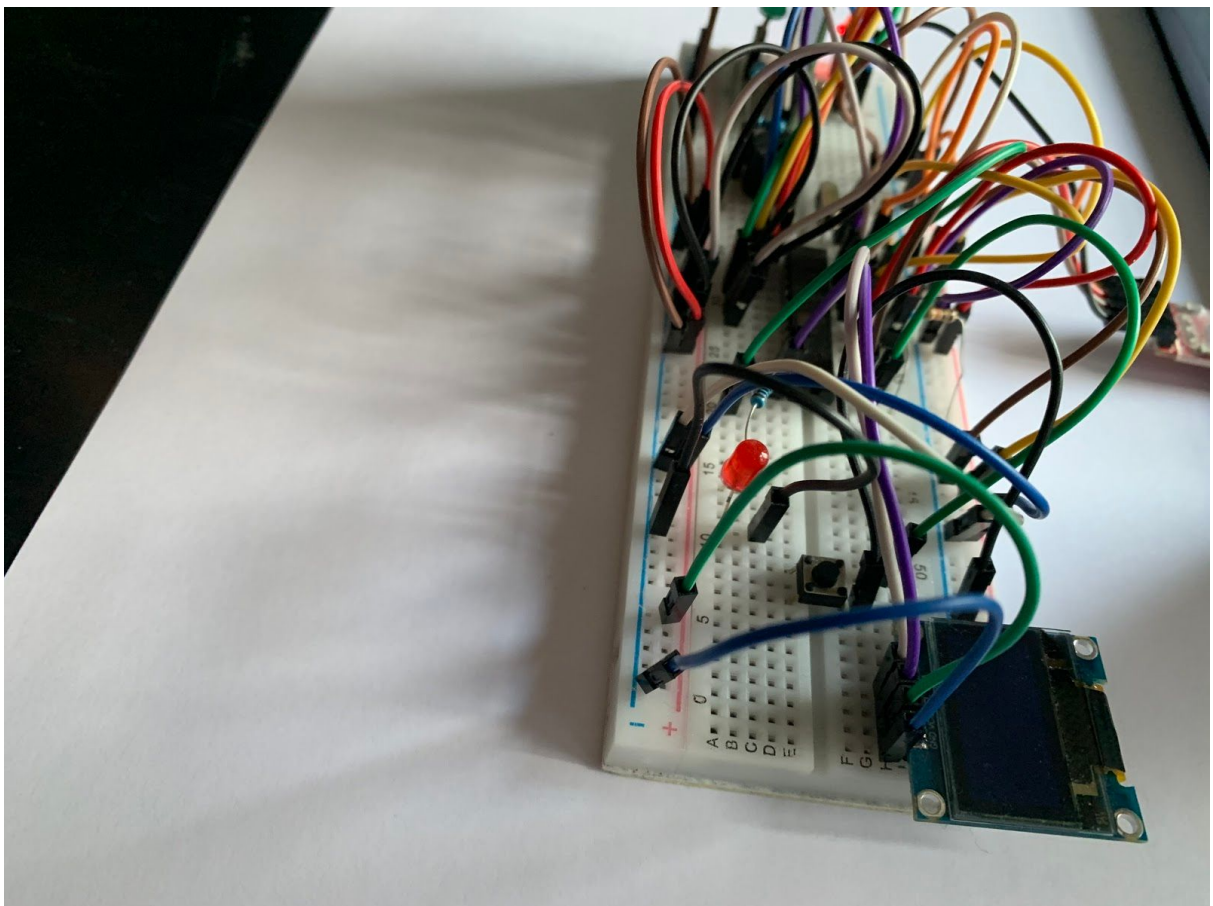
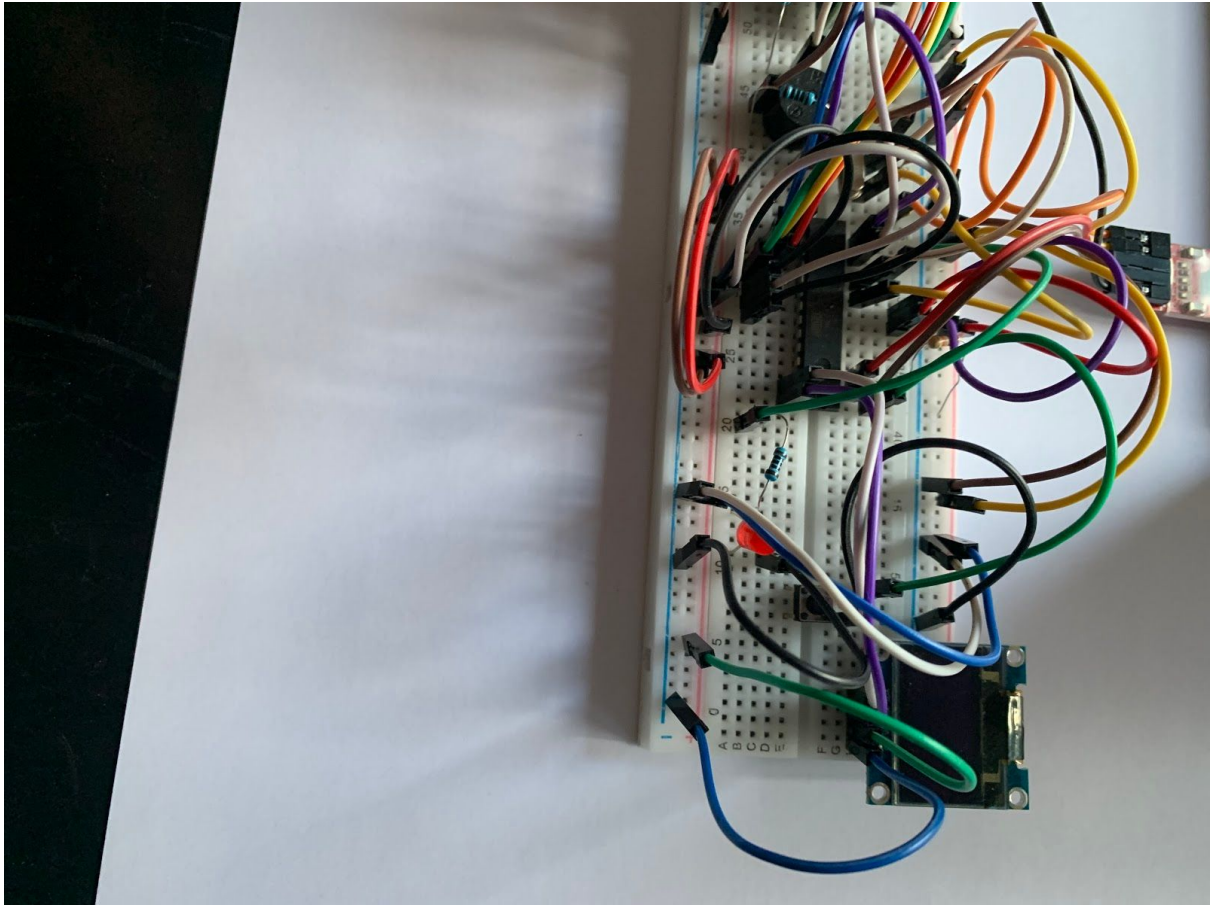
For the fabrication step, we were supposed to make our own PCB that we designed on DesignSpark. Unfortunately, the PCB lithography machine was inoperative. So we had to work only on our breadboards.

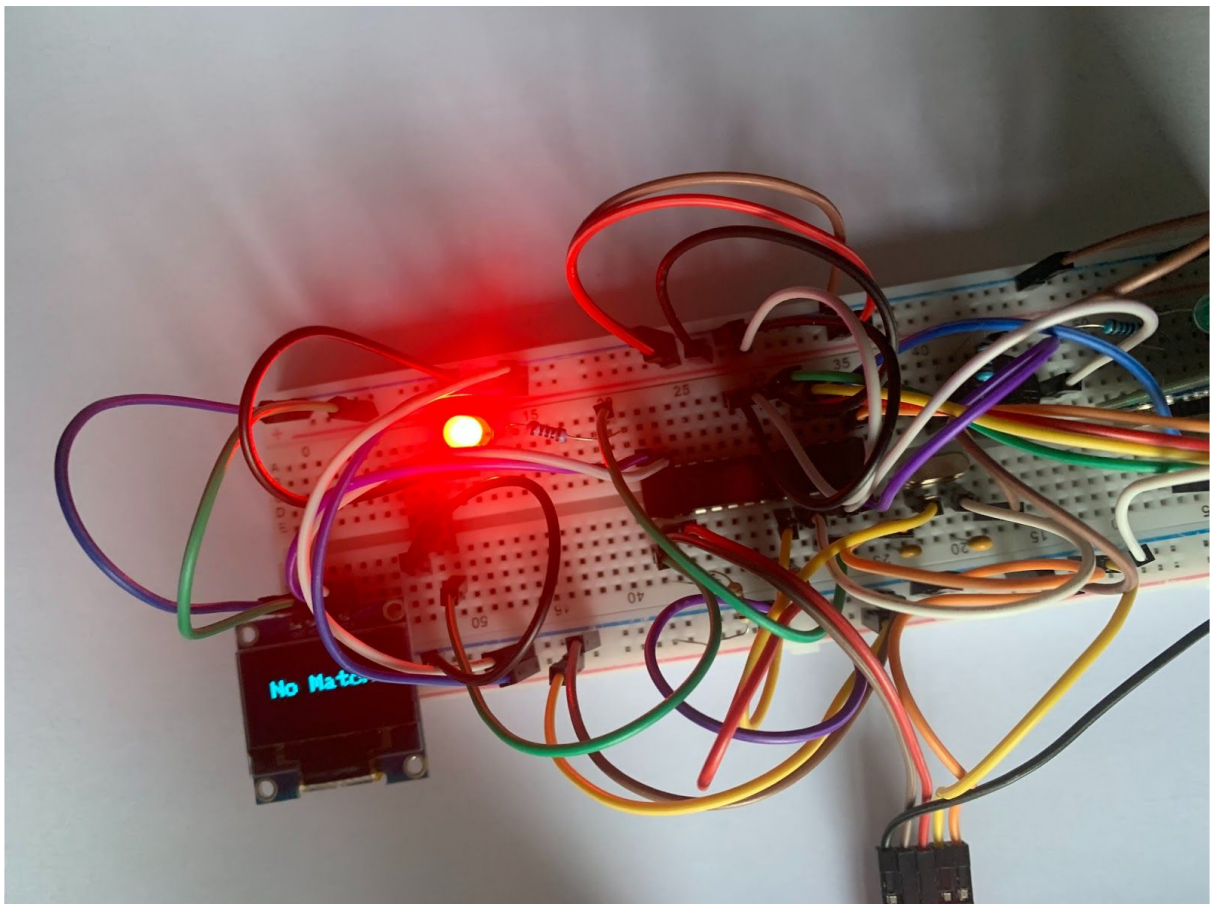
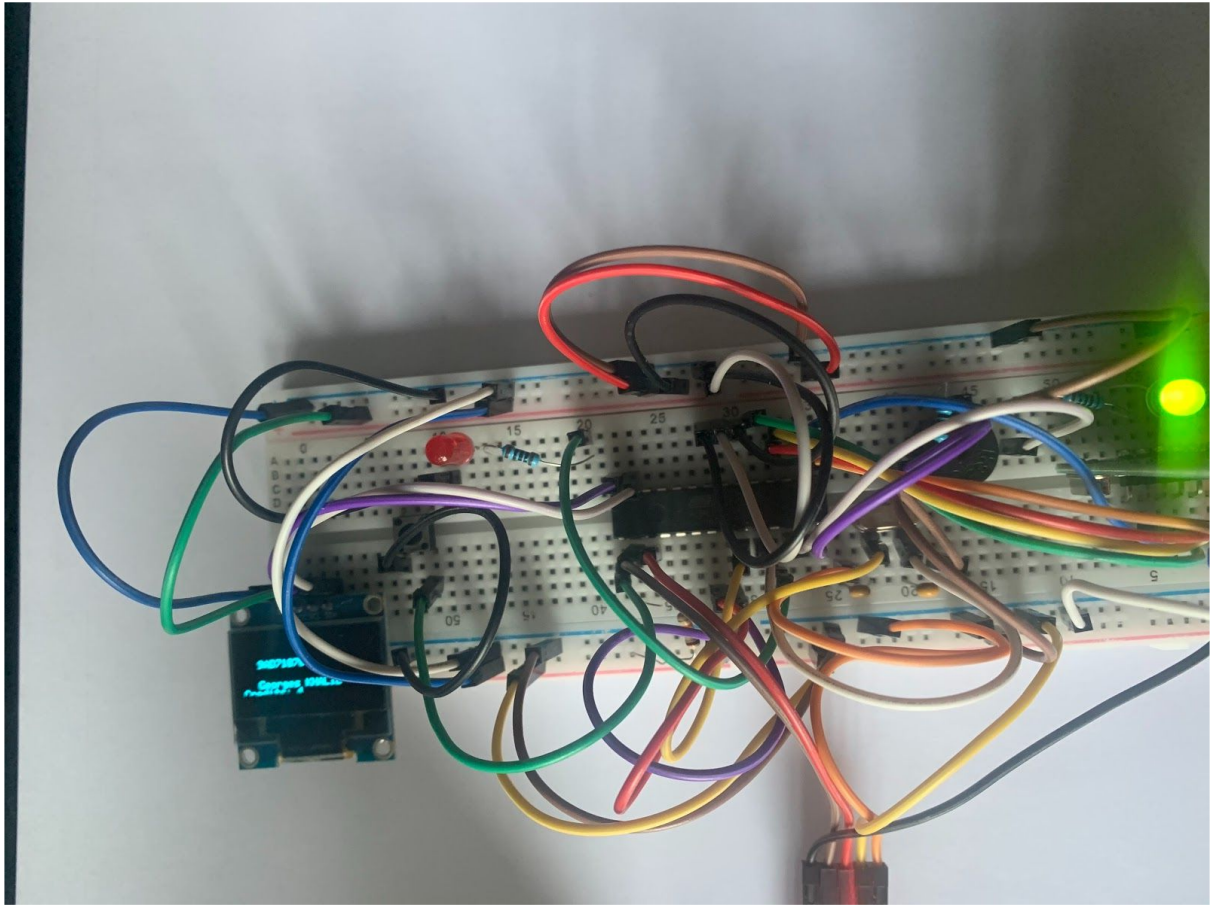
b. Final Product

The pictures below represent the final prototype of our project.









3. Team management

This part concerns the team's organization. Due to the actual COVID crisis, it was complicated to work together. Fortunately, we managed to achieve our project.

At the beginning, we tried the features that are necessary for the project before coding the whole project on the ATMEGA in order to test them all. Like it was said before, the components that we used are the RFID receptor, the buzzer, two leds (green and red), the OLED screen and obviously, the ATMEGA328P. We used to pass the board to the person who needed to test its code on the tasks he was assigned for and keep rotating the board. Praveen mainly worked on the Processing IDE and more precisely on the GUI interface. Georges was assigned to do the communication part between the RFID receptor and the ATMEGA328P, he used the Arduino IDE. Tony had to code the attendance sheet on Processing. Concerning the other technical parts of the project, we have done the work together. We helped each other when one of us had an issue with his tasks and vice versa.

CONCLUSION

The aim of the project was to apply the theory seen in class and create something that was able to gather everything we learned. The class attendance system was a good establishment because it was able to add as many features as possible. We decided to use a database (phpMyAdmin), record absence credits, develop and upgrade the GUI. Effectively, we faced a lot of challenges during the conception of the code for instance the WiFi module was not used during the project due to a lack of time. At the end, we succeeded and it was possible to demonstrate all we have learned so far on this project. This project could be developed with better components and could be sold in the market in the perspective where we could work on it and improve it on a rolling basis.

ANNEXES

Processing Code

```
Includes

import processing.serial.*; //Serial library
import de.bezier.data.sql.*; //DB library
import java.io.PrintWriter; //Library to write in text file
import uibooster.*; //GUI Library

//Serial variables
Serial myPort;
String val;

//DB variables
MySQL sql;
String id, surname, name;
int status;

//initialize buttons
Button button1, button2, button3, button4, button5, button6,
button7, button8, button9, button10, start, submitButton,
returnButton, returnDetails, nextDetails;

//initialize colors and images' variables
int col;
PImage imgCenter, imgECE;
int myColor = color(255);
int c;

//initialize variable to write in text file
PrintWriter output;
String absentStudents, presentStudents;

//initialize uibooster
UiBooster booster;

//other variables
```

```
int i, credits;
String courseNumber, course;

Main

public void settings()
{
    size(1800, 1000); //window size
}

void setup()
{
    background(25, 25, 25); //background color
    homeDisplay(); //calling homeDisplay function
    String portName = Serial.list()[0]; //Initilialize the port at
    position 0 in the list -> COM6
    myPort = new Serial(this, portName, 9600); //Establish the
    Serial connection at a frequency of 9600
}

void draw()
{
    while (myPort.available()>0) //check if the usb in connected
    {

        val = myPort.readString();
        if (val != null) //avoid lines with null
        {
            sqlIdentification(val);
            clear();
            background(200); //reset window
            displaySheet(); //attendace sheet display function
        }
    }
}

Button

// the Button class
class Button
{
```

```
String label; // button label
float x;      // top left corner x position
float y;      // top left corner y position
float w;      // width of button
float h;      // height of button

// constructor
Button(String labelB, float xpos, float ypos, float widthB,
float heightB)
{
    label = labelB;
    x = xpos;
    y = ypos;
    w = widthB;
    h = heightB;
}

void Draw()
{
    fill(014113);
    stroke(3, 218, 198); //border
    rect(x, y, w, h, 10);
    textAlign(CENTER, CENTER);
    fill(255);
    text(label, x + (w / 2), y + (h / 2));
}

//check if the button is clicked
boolean MouseIsOver()
{
    if (mouseX > x && mouseX < (x + w) && mouseY > y && mouseY <
(y + h))
    {
        return true;
    }
    return false;
}

//instructions to perform when the specific button is pressed
void mousePressed()
```

```
{
    if(button1.MouseIsOver())
    {
        //launch the text file course1.txt
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course1.txt");
    }
    else if(button2.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course2.txt");
    }
    else if(button3.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course3.txt");
    }
    else if(button4.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course4.txt");
    }
    else if(button5.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course5.txt");
    }
    else if(button6.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course6.txt");
    }
    else if(button7.MouseIsOver())
    {
        launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
```

```
Project/Project/Processing/TechProject/Main/data/course7.txt");
}
else if(button8.MouseIsOver())
{
    launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course8.txt");
}
else if(button9.MouseIsOver())
{
    launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course9.txt");
}
else if(button10.MouseIsOver())
{
    launch("C:/Users/prave/OneDrive - Groupe INSEEC
(POCE)/Bureau/Semester 1/Technical
Project/Project/Processing/TechProject/Main/data/course10.txt");
}
else if(start.MouseIsOver())
{
    //when push the start button in home screen
    booster = new UiBooster();
    //initialize combo box with uibooster library
    course = booster.showSelectionDialog(
        "Select the Course",          // Question
        "Course Number",             // window
        title
        "Course 1", "Course 2", "Course 3", "Course 4",
        "Course 5", "Course 6", "Course 7", "Course 8", "Course 9",
        "Course 10");

    if(course.equals("Course 1"))
    {
        course = "1";
        //reset window
        clear();
        background(25, 25, 25);
        //redirect to the function detailCourse
        detailCourse();
    }
}
```

```
else if(course.equals("Course 2"))
{
    course = "2";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 3"))
{
    course = "3";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 4"))
{
    course = "4";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 5"))
{
    course = "5";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 6"))
{
    course = "6";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 7"))
{
    course = "7";
    clear();
    background(25, 25, 25);
    detailCourse();
}
```

```
else if(course.equals("Course 8"))
{
    course = "8";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 9"))
{
    course = "9";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else if(course.equals("Course 10"))
{
    course = "10";
    clear();
    background(25, 25, 25);
    detailCourse();
}
else
{
    clear();
    background(25, 25, 25);
    homeDisplay();
}

}
else if(returnDetails.MouseIsOver())
{
    //return button in Course Details
    clear();
    background(25, 25, 25);
    homeDisplay();
}
else if(nextDetails.MouseIsOver())
{
    //next button in Course Details
    clear();
    background(25, 25, 25);
    displaySheet();
}
```

```
}
else if(returnButton.MouseIsOver())
{
    //return button in Attendance sheet
    clear();
    background(25, 25, 25);
    resetStatus();//reset status to 0 for every student
    detailCourse();
}
else if(submitButton.MouseIsOver())
{
    //submit button in Attendance sheet
    clear();
    background(25, 25, 25);
    decrementCredits();//decrement the number of credits of absent
students
    saveAttendanceSheet("course"+course+".txt"); //save attendance
sheet in a text file
    resetStatus();//reset status to 0 for every student
    homeDisplay();
}
}
```

DetailCourse

```
//displaying details of the course
void detailCourse()
{
    fill(255, 255, 255);
    textSize(30);
    text("Course Details", 150, 50);

    textSize(24);
    text("Course Name: Technical Project", 800, 150);
    text("Class: ING4 SE Group 1", 800, 220);
    text("Course Number: Course "+ course, 800, 290);

    text("Professor: Dr. Maxime SCHNEIDER", 800, 360);
    text("University: ECE Paris", 800, 430);

    //initialize return and next buttons
```



```
returnDetails = new Button("RETURN", 550, 550, 150, 80);
nextDetails = new Button("NEXT", 900, 550, 150, 80);

//display both buttons
returnDetails.Draw();
nextDetails.Draw();
}
```

Attendance

```
void sqlIdentification(String val)
{
    //creation of the object MySQL
    sql = new MySQL(this, "localhost", "techproject", "root", "");
    String list;
    String noMatch = "a";

    //connection to the database
    if (sql.connect())
    {
        //query where we select the student by his/her badge's id
        sql.query("SELECT * FROM rfid WHERE identification = '" + val
+ "'");

        //if there is a match in the database
        if(sql.next())
        {
            surname = sql.getString(2);
            name = sql.getString(3);
            status = sql.getInt(5);
            credits = sql.getInt(6);
            list = name+" "+surname+"\n"+"Credits: "+ credits;

            if(status == 0)
            {
                //query where we update the student's status by his/her
                badge's id (present)
                sql.query("UPDATE rfid SET status = 1 WHERE
identification = '" + val + "'");
                println("Student: "+list+"\nStatus: Added to the
attendance");
            }
        }
    }
}
```

```
        myPort.write(list);
    }
    else
    {
        //query where we update the student's status by his/her
        badge's id (absent)
        sql.query("UPDATE rfid SET status = 0 WHERE
identification = '" + val + "'");
        println("Student: "+list+"\nStatus: Left the classroom");
        myPort.write(list);
    }
}
else
{
    println("No identification in the database matches the
card!");
    myPort.write(noMatch);
}
}
else
{
    println("Database is not connected !");
}
}

//function that display the attendance
void displaySheet()
{
    //background(25,25,25);
    //creation of the object MySQL
    sql = new MySQL(this, "localhost", "techproject","root","");

    //connection to the database
    if(sql.connect())
    {
        background(25,25,25);

        //page title with its color and position
        fill(255, 255, 255);
        textSize(30);
        text("ATTENDANCE SHEET", 200, 50);
    }
}
```

```
text("Course "+ course, 1500, 50); //display course number

//display header of the attendance sheet
fill(3, 218, 198);
textSize(16);
text("Surname", 200, 100);
text("Name", 500, 100);
text("Remaining credits", 800, 100);
text("Status", 1100, 100);
fill(255, 255, 255);

//query where we select all student
sql.query("SELECT * FROM rfid ORDER BY surname ASC");

//if there is a match in the database
while(sql.next())
{
    surname = sql.getString(2);
    name = sql.getString(3);
    status = sql.getInt(5);
    credits = sql.getInt(6);

    if(status==1)
    {
        //display present when the badge is passed and recognized
        by the DB
        fill(255,255,255);
        text(surname, 200, 150+i*35);
        text(name, 500, 150+i*35);
        text(credits, 800, 150+i*35);
        fill(5,131,33);
        text("PRESENT", 1100, 150+i*35);
        i++;
    }
    else
    {
        //else display absent
        fill(255,255,255);
        text(surname, 200, 150+i*35);
        text(name, 500, 150+i*35);
        text(credits, 800, 150+i*35);
        fill(160,0,0);
    }
}
```

```
        text("ABSENT", 1100, 150+i*35);
        i++;
    }
}
i = 0;
}
else
{
    println("Database is not connected !");
}

//initialize return and submit button
returnButton = new Button("RETURN", 1300, 150, 150, 80);
submitButton = new Button("SUBMIT", 1600, 150, 150, 80);

//display both buttons
returnButton.Draw();
submitButton.Draw();

}

//function that decrement the number of credits of absent students
void decrementCredits()
{
    //creation of the object MySQL
    sql = new MySQL(this, "localhost", "techproject","root","");

    //connection to the database
    if(sql.connect())
    {
        sql.query("SELECT * FROM rfid");

        while(sql.next())
        {
            status = sql.getInt(5);
            credits = sql.getInt(6);

            if(status == 0)
            {
                //number of credits for each student decrement while
                absent
                credits--;
            }
        }
    }
}
```

```
        sql.query("UPDATE rfid SET credits = credits - 1 WHERE  
status = 0");  
    }  
}  
  
}  
}
```

//after the class, reset all students to absent

```
void resetStatus()  
{  
    sql = new MySQL(this, "localhost", "techproject", "root", "");  
    if (sql.connect())  
    {  
        sql.query("UPDATE rfid SET status = 0");  
  
        sql.close();  
    }  
    else  
    {  
        println("Database is not connected !");  
    }  
}
```

MainGUI

```
void homeDisplay()  
{  
    fill(3, 218, 198);  
    rect(0,0,1800,60);  
  
    fill(255, 255, 255);  
    textAlign(CENTER, CENTER);  
    textSize(30);  
    text("STUDENT ATTENDANCE SYSTEM", 870, 30);  
  
    historyMain();  
    attendanceSection();  
}
```

```
void attendanceSection()
{
    fill(255, 255, 255);
    textAlign(CENTER, CENTER);
    textSize(26);
    text("ATTENDANCE SECTION", 250, 150);

    imgCenter = loadImage("photos/classroom.png");
    imageMode(CENTER);
    image(imgCenter, width/2, 2*height/5, 600, 400);

    start = new Button("START", 710, 240, 375, 180);
    start.Draw();
}

void historyMain()
{
    fill(255, 255, 255);
    textAlign(CENTER, CENTER);
    textSize(26);
    text("HISTORICAL DATA", 225, 620);

    button1 = new Button("Course 1", 400, 700, 180, 60);
    button2 = new Button("Course 2", 600, 700, 180, 60);
    button3 = new Button("Course 3", 800, 700, 180, 60);
    button4 = new Button("Course 4", 1000, 700, 180, 60);
    button5 = new Button("Course 5", 1200, 700, 180, 60);
    button6 = new Button("Course 6", 400, 800, 180, 60);
    button7 = new Button("Course 7", 600, 800, 180, 60);
    button8 = new Button("Course 8", 800, 800, 180, 60);
    button9 = new Button("Course 9", 1000, 800, 180, 60);
    button10 = new Button("Course 10", 1200, 800, 180, 60);

    button1.Draw();
    button2.Draw();
    button3.Draw();
    button4.Draw();
    button5.Draw();
    button6.Draw();
    button7.Draw();
    button8.Draw();
    button9.Draw();
    button10.Draw();
}
```

```
button8.Draw();
button9.Draw();
button10.Draw();
}

SaveTxt

void saveAttendanceSheet(String fileName)
{
    output = createWriter("data/"+fileName);

    output.println("Present Students\n");
    presentStudentSql(presentStudents);
    output.println("\n");

    output.println("Absent Students\n");
    absentStudentSql(absentStudents);

    output.flush();
    output.close();
}

//retrieve the names and surnames of absent students
void absentStudentSql(String absentStudents)
{
    sql = new MySQL(this, "localhost", "techproject", "root", "");
    //DB connection
    if (sql.connect())
    {
        sql.query("SELECT * FROM rfid WHERE status = 0");

        while(sql.next())
        {
            surname = sql.getString(2);
            name = sql.getString(3);

            absentStudents = name + " " + surname;
            output.println(absentStudents); //write the names and the
            surnames of absent students in a text file
        }

        sql.close();
    }
}
```

```
}  
else  
{  
    println("Database is not connected !");  
}  
  
}  
  
//retrieve the names and surnames of present students  
void presentStudentSql(String presentStudents)  
{  
    sql = new MySQL(this, "localhost", "techproject","root","");  
    if (sql.connect())  
    {  
        sql.query("SELECT * FROM rfid WHERE status = 1");  
  
        while(sql.next())  
        {  
            surname = sql.getString(2);  
            name = sql.getString(3);  
  
            presentStudents = name + " " + surname;  
            output.println(presentStudents); //write the names and the  
surnames of present students in a text file  
        }  
  
        sql.close();  
    }  
    else  
    {  
        println("Database is not connected !");  
    }  
}
```


Arduino Code

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SPI.h>
#include <MFRC522.h>

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
#define RST_PIN 15 // RFID Reset pin connected to
ATMEGA328 pin 15
#define SS_PIN SS // RFID Reset pin connected to
ATMEGA328 pin 16
#define LED_RED 5 // LED RED connected to ATMEGA328P
pin 5
#define LED_GREEN 6 // LED GREEN connected to ATMEGA328P
pin 6
#define buzzer 7 // Buzzer connected to ATMEGA328P
pin 7

String rejection_message = "No Match";
String names;

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance

void setup() {
  Serial.begin(9600); // Initialize serial
communications with the PC
  while (!Serial); // Do nothing if no serial port
is opened (added for Arduinos based on ATMEGA32U4)
  SPI.begin(); // Init SPI bus
  mfrc522.PCD_Init(); // Init MFRC522
  //mfrc522.PCD_DumpVersionToSerial();// Show details of PCD -
MFRC522 Card Reader details
  //Serial.println(F("Scan PICC to see UID, SAK, type, and data
blocks..."));
  pinMode(buzzer, OUTPUT); // Set buzzer - pin 7 as an output
  pinMode(LED_GREEN, OUTPUT); // Declare the LED as an output
  pinMode(LED_RED, OUTPUT); // Declare the LED as an output

  //Verify allocation to the OLED address
  //Serial.begin(115200);
```

```
if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D for
128x64
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
}

// Setup OLED parameters
delay(2000);
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(30, 30);
}

void loop() {
    // Look for new cards
    if ( ! mfrc522.PICC_IsNewCardPresent()) {
        return;
    }

    // Select one of the cards
    if ( ! mfrc522.PICC_ReadCardSerial()) {
        return;
    }

    // // Dump debug info about the card; PICC_HaltA() is automatically
    // called
    // mfrc522.PICC_DumpToSerial(&(mfrc522.uid));

    //Read each byte of the UID and store it in hex_num
    unsigned long hex_num[4];
    for (int i=0; i<4; i++){
        hex_num[i] = mfrc522.uid.uidByte[i];
        //Serial.print(hex_num[i],HEX);          // Print the UID
    }
    mfrc522.PICC_HaltA(); // Stop reading

    //Convert hex_num to a String to compare it with database
    String Converted_UID;
    for (int i=0; i<4; i++){
        Converted_UID += String(hex_num[i],HEX);}
    Converted_UID.toUpperCase();
    // Serial.println("\nConverted_UID = ");
    //for (int i=0; i<100;i++){
```

```
Serial.print(Converted_UID);
//}

//String UID = "9AD71078";

names = Serial.readString();

if ( names == "aa" || names == "a" ) {

    display.setTextSize(2);
    display.setTextColor(WHITE);
    display.setCursor(15, 20);
    display.print(rejection_message);
    display.display();

    tone(buzzer, 500);          // Send 1KHz sound signal...
    digitalWrite(LED_RED, HIGH); // Turn the LED on
    delay(500);                 // ...for 1 sec
    noTone(buzzer);             // Stop sound...
    digitalWrite(LED_RED, LOW);  // Turn the LED off

    delay(2000);
    display.clearDisplay();
    display.display();
}
else {
    display.setTextSize(1); // 2
    display.setTextColor(WHITE);
    display.setCursor(15, 20); // 20,30
    display.print(Converted_UID);
    display.display();

    display.setTextSize(1);
    display.setTextColor(WHITE);
    display.setCursor(15, 40);
    display.print(names);
    display.display();

    tone(buzzer, 2000);          // Send 1KHz sound signal...
    digitalWrite(LED_GREEN, HIGH); // Turn the LED on
    delay(500);                 // ...for 1 sec
    noTone(buzzer);             // Stop sound...
    digitalWrite(LED_GREEN, LOW);  // Turn the LED off

    delay(1500);
}
```

```
display.clearDisplay();  
display.display();  
  
}  
}
```

Bibliography

- https://www.digikey.fr/en/resources/conversion-calculators/conversion-calculator-resistor-color-code?utm_adgroup=Resistors&utm_source=google&utm_medium=cpc&utm_campaign=Dynamic%20Search_EN_Product&utm_term=&productid=&gclid=Cj0KCQjwuL_8BRCXARIsAGiC51CeSQgjB2rvTe6M0mSnG3gLpmChe4gCwkl7ofikXqkEdLtxXwt_2waAoB7EALw_wcB
- <https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/>
- https://cdn.ozdisan.com/ETicaret_Dosya/536999_6091041.pdf
- <https://components101.com/wireless/rc522-rfid-module>
- <http://www.handsontec.com/dataspecs/RC522.pdf>
- <http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf>
- <https://www.sparkfun.com/datasheets/Components/SMD/ATMega328.pdf>
- <https://components101.com/wireless/esp8266-pinout-configuration-features-datasheet>
- <https://forum.processing.org/two/discussion/9982/adding-a-simple-button-to-a-class>
- <https://github.com/Milchreis/uibooster-for-processing>
- <https://processing.org/reference/libraries/>