I. Project Title

The Smart-house Project is our Intro to Computer Systems Project. It is an appropriate name describing all team members' real contribution in this course and pursues us to complete project activity and deliverables' primary object. The anticipated name is also the promise for the final product's achievement that members can take pride in remembering its values.

II. Description of the Project

1. Project Description

Smart house is our project's name which is combined with 4 components: lights controlled by voice recognition, watering plants automatically, the opened/closed door using RFID attended to Anti-theft devices. All of these functions can be implemented by MIT Application on Smartphones to give the best convenient experience for the end-user.

Controlling the light by using the voice recognition will bring the convenience of managing the light in the house as well as having the wonderful experience of the technology. Furthermore, although some people may find it easier for controlling their house, others find it struggling. Voice recognition will help disabilities to manage their home simpler without touching the switch and avoiding an unfortunate accident with the electricity.

Automatic watering based on sensors measuring soil moisture is one of the interesting and useful features when used in conjunction with a smart home system. This feature is used quite commonly nowadays but they are used as a stand-alone system and are not connected to smart homes. Therefore, our team that developed this automatic watering system aims to connect them to the smart home system and is supposed to use the voice to command watering the plants or interrupt the watering and run in parallel sensors that can self-water plants without human implement.

This automatic door is the combination of the NFC door lock which we regularly see at the hotel and the automatic sliding door that we can see at the Mall. Our automatic door work is this way, if the right ID card is read, the door will automatically open and if the right card is read again by the NFC reader, it will close the door. In anycase, if the door is opening or closing, if RFID NFC is reading the wrong ID card, the buzzer will alarm and the door will close automatically.

2. Materials Section

- **Arduino controller:**
  Arduino is commonly known as a microcontroller that is easier to you than Raspberry PI. Therefore, we decided to use Arduino UNO 3 - one of the most common and

easiest to use among the Andruino family. According to the official site of Arduino Store, this microcontroller board is based on the ATmega328. Moreover, it has a total 14 digital I/O pins (6 pins are hardware PWM), 6 Analog Input pins, USB connection, a power jack and a reset button.
The language that we use to process the project is Wiring, another platform of C/C++ language. This open source project provides some built in functions that support us to do the project for example: pinMode(), digitalWrite, delay(). Furthermore, to program nor send and receive the data from the Arduino board, we use the virtual environment called Arduino IDE to run the program.

- **Breadboard**
  Currently, the breadboard we use in the project has 400 points to connect the jumpers with the size is 5.5 cm x 17 cm for 2 power lanes on the side and double-strip in the middle. The breadboard will be the place to connect different jumpers and transfer the electricity from the Arduino to other features in the common column or row of the house.

On/Off Lights using Voice

- **LEDs**

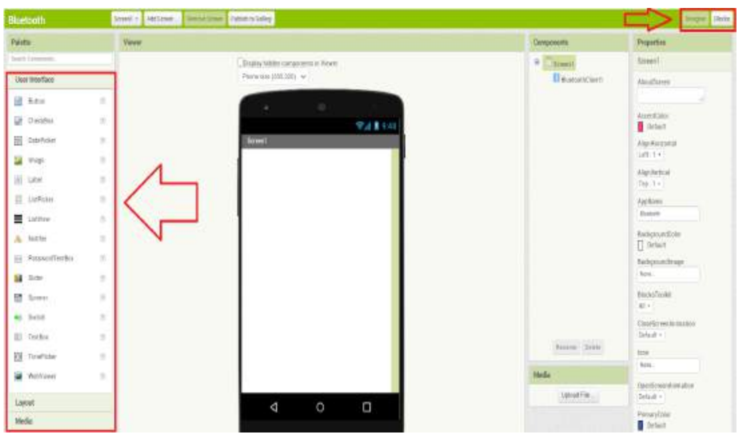
  The LED (shorted for Light Emitting Diode) is the light that can be displayed in different colours by using two legs. In detail, the LED connected to the electricity primarily via two legs with approximately 1.8 ~ 3 Voltage. The short legs will connect to the GND (ground) and others will connect to the Voltage of the Arduino.

- **Jumper Wires (Female-Male, Male-Male)**

  The jumper wire connects the feature to the Arduino directly or with the intermediaries such as breadboard. There are two types of wires namely Female-Male, Male-Male with different attributes. ‘Male’ ends have emerged and can plug into things, while the ‘female’ ends are plugged by other things. The length is 11.8 inches or 29.9 centimeters with the space when placing two jumpers to each other is 0.1 inches (2.54mm).

- **Smartphone (Android)**
- **MIT app inventor**

MIT App Inventor is the application that helps the programmer develop a mobile application without coding by manual way. It means that developers need to log in the website of the MIT app for coding by drag and drop. The MIT provides two main functions, which is the front-end and the back-end system. By the default, developers are already in the front end of the system and can drag and drop buttons on the right hand side of the website. Relating to the back-end system, the user needs to click the button “Block” in the left hand side, which is next to the button “Design”. Developers can drag and drop functions on the left hand side instead of coding by manual way..



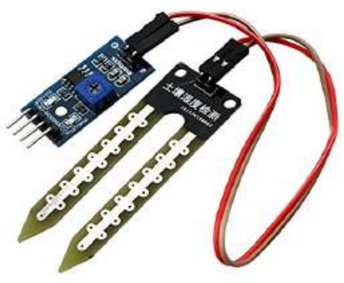
Autonomous Water Plants

- **Mini water pump 5VDC**

  Mini Water Pump 5VDC submersible motor has a very compact size including red wire (positive voltage), black wire (negative voltage). It uses a voltage of 3 ~ 5VDC, because it is a submersible pump, the motor is water resistant and works when submerged in water, used to pump water, liquid in small designs, and watering models. With the use of small voltage from 3 ~ 5VDC, the 4-pin 18650-size cell power supply can be flexibly easily used.

- **Soil moisture sensor**

  Soil moisture sensor has the function of measuring the volume of water in the soil. They operate with 4 main connector pins including VCC and GND pins, a digital pin and an analog pin to Arduino-based devices. In which 2 pins VCC and GND work to provide power for the soil moisture sensor. The soil moisture sensor will give the appropriate value for turning the relay switch on so that the pump will start pumping water and vice versa, based on the water volume data that the sensor reads in the soil.



- **Switch relay KY-019 5VDC*2**

  Relay circuit KY-019 5VDC compact size is used to switch AC or DC equipment, the circuit uses 5VDC voltage with only 3 connecting pins for easy use. Relay circuit KY-019 5VDC with switching contacts consisting of 3 contacts NC (normally closed), NO (normally open) and COM (common pins) is completely isolated from the main circuit board, in normal state without NC click will connect to COM, when the state is clicked, COM will switch to NO and lose connection with NC.

- **Battery base case and 4 Pin Cell 18650 2000m**

  A 4-size 18650 pin box with spring-loaded in series is used for attaching 4 batteries of 18650 in series, making it easy to attach to the device. The battery box has two available positive and negative power cords (red and black) very convenient. Rechargeable 18650 3.7V 2200mAh 3A is used to power the circuit. These 2 components play an important role in supplying extended power to meet the pump voltage requirement which cannot be met by Arduino circuit voltage.

Open/Close Door using RFIDs

- **RC Servo 9G**

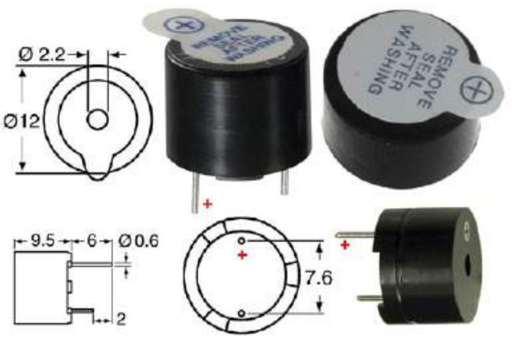
  Brief description: Our project is an automatic door. Therefore, we will use the Servo 9G to rotate to the degree that we want (maximum 180 degree) to represent the open and the close movement of the door. For example, for the opening, we rotate the door to 90 degree and for closing, the servo will rotate to 0 degree. The servo's reaction speed is up to 0.12 second for 60 degrees and the gears are made of plastic so we cannot put a lot of force since it may break the gear. The servo has a total of 3 connection wires which are Brown for Ground, Red for 5V and Orange for PWN.

- **RFID card * 2 (Different ID) and RFID tag (Different ID)**

  To make the door open and close after that, we might have to use an RFID card or tag so that the RFID NFC can read and recognize the specific ID to open the door. The cards will have the size 86 x 54 mm and the tag will be 40.5 x 32 x 4.2mm. In this case, to distinguish the things to use for opening the door, we use the blue RFID tag.

- **RFID NFC 13.56Mhz RC522**

  Besides the RFID card, this is a must item if we want it to automatically succeed. The RFID NFC module can read and write data to RFID cards or tags. Therefore, this reader module can help us to give access to specific RFID cards to open the door. The RFID NFC has two versions, one is RFID NFC 13.56Mhz PN532 and the others is RFID NFC 13.56Mhz RC522. However, because of the cost-effectiveness, we choose the RC522 version. This version of RFID NFC has blue color and the size is 40mm × 60mm. It has totals of 8 connections which are SDA, SCK, MOSI, MISO, IRQ, GND, RST, VCC (for 3.3V).
- **Buzzer 5VDC**



  The buzzer works as an alarm if the RFID card cannot be recognized since we want to develop the automatic door that will notify, by making the 2KHz "BEEEE" noise, not only the homeowners but also their neighbor that someone is trying to break into the house. The buzzer has black color and has the size 1.2cm diameter x 1cm tal. It also has two pins, the short one is for positive (+) pin and the negative is the shortened one.

3. Procedures or steps

On/Off Lights using Voice

1. **On and off the LED with the wifi**

In the first stage, connect the LED to the ESP8266 board by using at least two jumper wires type Male-Male (M-M) and supply the power from the Arduino Uno. To demonstrate, the first wire jumper head will connect to the GND of the ESP8266, the other will connect the breadboard, which will be supplied by the power of the Arduino. The power supply 3.3V from the Arduino connects to the breadboard as the intermediary wire jumper, which enables the LED to turn on the light. In the project, we are using the D4 and D2 of the ESP8266 for controlling the LED and using the transistor for having more LEDs in one pin; therefore, we have 4 LEDs using 2 pins. After testing the LED by manual on the breadboard, we weld the metal of two wire jumpers together for prolonging the wire and never breaking the wire. Because of attaching the LED into the wall, we need the glue for sticking it. Then, we let the main board Arduino as well as the ESP8266 wifi to be under the ground. The voice recognition needs programming using MIT app for the mobile and the Arduino IDE for coding the device.

Autonomous Water Plants

1. **Connect the soil moisture sensor to the Arduino**

To be able to share the space used to connect components to the arduino, using a breadboard is a useful solution. First, our team connected the arduino vs breadboard using two Male-Male (M-M) connectors. Specifically, a wire attached to the 5V pins - which serves as the power supply, with the positive voltage of the circuit board and a wire that connects the ground pins (GND) to the breadboard. Next, we connect 2 pins of 5V and GND respectively on the breadboard to power the sensor. The sensor should be connected to the arduino using 1 analog pin (in this case we use A0 pin - ESP8266 wifi), and a digital pin (in this case we use pin 15 -pin D8 of ESP8266 wifi).

2. **Connect the relay - external power - mini water pump to the Arduino**

The part connecting the three relay devices - external power - mini water pumps is a rather complicated part because they need the tools to shut off the pump (relay) and the external power to supply the pump (external power). To connect the relay to the arduino, we need to power the relay by connecting the 5V and GND pins through the breadboard which is connected to the arduino, respectively. Next, we need to connect the digital pin of the relay to the digital pin of the arduino (in this case we use digital pin 5 - pin D1 of ESP8266 wifi).

The next step, the relay and the water pump are connected via a closed circuit loop-through rule. The negative end of the power supply is in series with the negative power cord of the pump (black wire). For the positive end of the extended power supply, connect the common pin of the relay (COM) and connect the normally open (NO) pin of the relay to the positive lead of the pump (red wire) to create an electrical private circuit.

3. **Coding in the Arduino IDE**

For programming in arduino, declare pins connected to arduino including 1 analog pin (A0) and 2 digital pins (5 for relay - 15 for sensor). Next, we need to set some variable values required to set the sensor to match the data that it has measured the soil moisture level. From there, execute the relay on-off command corresponding to the water pump on and off. More specifically, we need to define a new function (wateringProcess) aind use some if statements to break up specific cases where variable values change based on data readable by the sensor. Place both this function in void loop () so it can run automatically and repeat continuously when arduino is powered on. Besides, an extra function that our team added is the voice pump opening / closing. This runs similar to automatic pump shutdowns with sensors. The only difference is to add the defined wateringProcess function to the voice programming section corresponding to the corresponding voice recognition command programmed in the section.

Open/Close Door using RFIDs

To make the automatic door, there are a total 2 steps which are connecting all the components togethers including the Arduino UNO3, the RC Servo, the RFID NFC and the buzzers, and programming on Arduino IDE to make the project work.

1. Connect all the components

a. Connect the Andruino to the RFID NFC

First, we used the Male - Female jumper wire to connect the digital pin 13 of Arduino to the SCK of the Reader NFC. Secondly, we wired the pin 12 of the Andruino to MISOI of NFC. Thirdly, we used the wires to connect the pin 11 of Arduino with MOISI of RFID NFC. The SDA of the NFC will be connected with digital pin 10 with the Male and Female jumper wire. Then we will connect the RST of the Reader NFC to the pin 9 of the microcontrollers board. And for the electricity supply, we connected the 3.3V of the NFC to the 3.3 of the Andruino.

b. Connect the Andruino to the RC Servo

After finishing connecting the Arduino with NFC, we dealt with the Servo. Firstly, we connect the orange Female wire which is PWN to the pin 6 of Arduino. Then, we connect the red one which is 5V with the 5V of Arduino for the electricity generation.

c. Connect RC Servo with Buzzer:

In the last connection between Andruino and RC Servo, we had the last wire left which is the brown wire standing for Ground. That jumper wire was connected to the negative(-) pin of the Buzzer.

d. Connect the Buzzer to Andruino

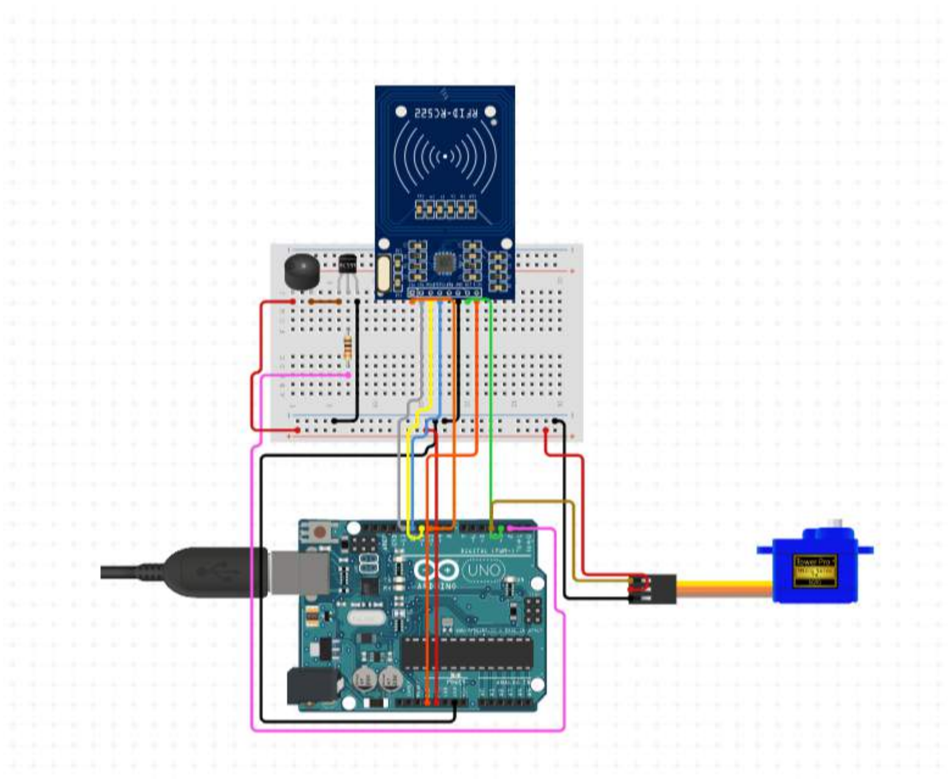
Then we connected the positive (+) pin of the buzzer to the pin 8 of the board with the Female - Male wire. Furthermore, we had to use wire Male - Male wire to use the other side of the wire to solder with the previous wired connection between the Ground of Servo and the negative (-) pin of the buzzer. The remaining side of this jumper was connected to the GND of the Andruino to later supply the electric for the RFID NFC.

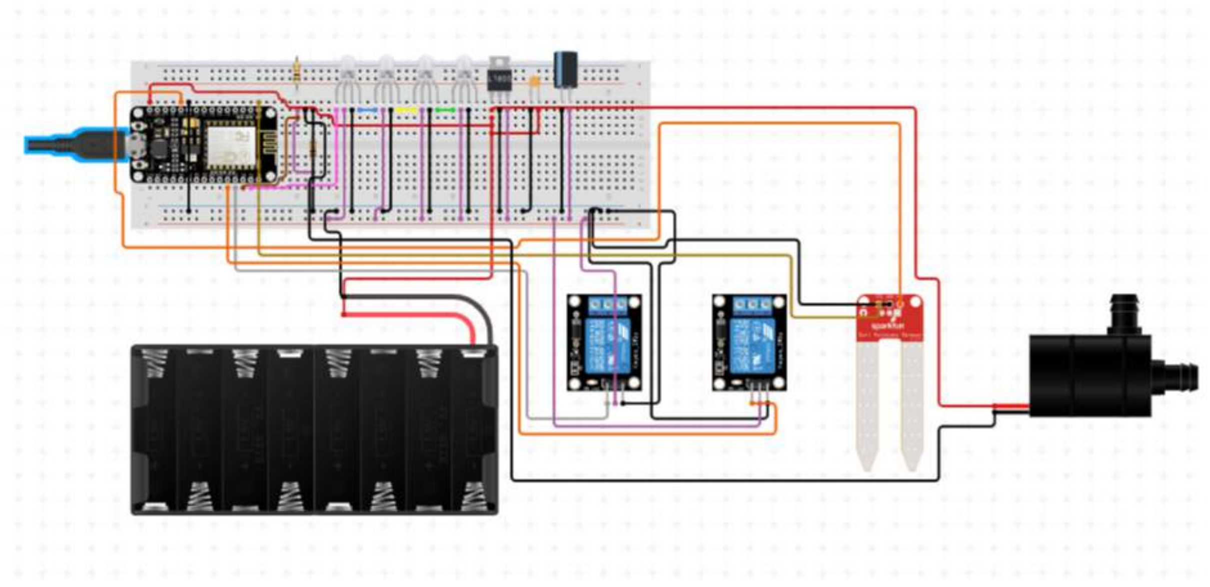
e. Buzzers connect to RFID

Finally, we connect the GND of the RFID Reader to the negative (-) pins of the buzzer. For this connection, we might have to use another Male-Male wire to solder the quadruple connection between the Ground of Servo, the negative pin of Buzzer, GND of Arduino and the GND of the RFID NFC.

Integration Section : Matching electrical circuit and Implement Application

1. Diagram for the smart house.





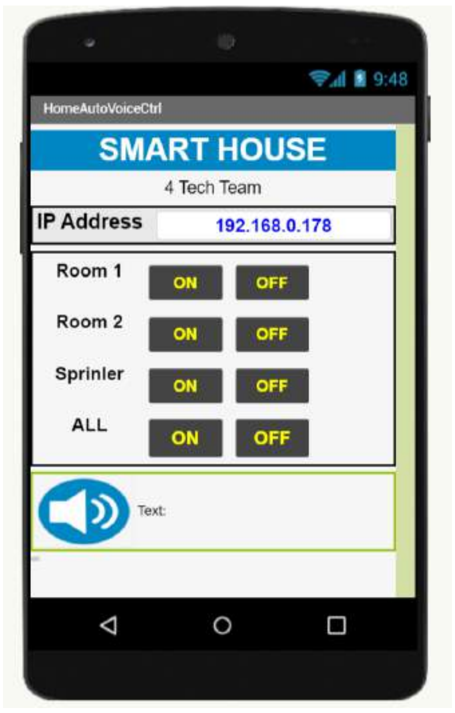
Note :

We cannot draw the whole diagram since the circuit.io just can draw one controller at one time and the diagram below was not the same as the real product. In the final product, we connect the Andruino and the NodeMCU together. The Andruino just supplies the power by one GND and one 5V and one 3.3V. 5V is connected with a breadboard power rail (+). GND will connect with the Ground Rail of the breadboard (-). The 3.3V just is used for connecting with the device that requires 3.3V power supply.

2. Mobile application

Relating to the mobile application, MIT App Inventor is the application that helps the programmer develop a mobile application without coding by manual way. To illustrate, the way of the MIT app works similar to the micro-bit editor where the developer drags and drops the function, which will develop an app with both the front-end and back-end of the system. The application we developed will use Google voice recognition to understand user commands and exchange the voice commands into texts and transmit them via wifi to the ESP8266 board for executing the feature of the smarthouse. The command will be transferred through the internet to the website and captured by the ESP8266.

After coding by blocks, developers need to use their mobile scan the QR code in order to download the software or even share the QR code of the application to the other. Developers can use the software in two different ways. First, the user can open the software by installing the MIT AI2 Companion software for scanning the QR code, which will run in that app and better for testing. Also, the user can download the application directly by generating the QR code on the website and would be better for running when users are ready to use forever. Note that the MIT app inventor is only available for the Android version.



3. **Programming**

   There are two part of coding : Coding for Andruino and Coding for NodeMCU

a. **Code for NodeMCU:**

Firstly, we include the library to run the NodeMCU which is ESP8266WiFi.h. Then we declare a WiFiClient variable called client and create the server object (WifiServer server(80)) which will respond to client in port 80. Next we declare the ssid (name of the

Wifi network) and the password variable to later set up to the Wifi.begin(ssid, passs). There is also the variable called String command that will receive the command from the mobile app.

After done setting up with the Wifi, we will declare some variable related to the water pump which are pins of the moisture sensor, pin of the relay, time_to_set_sample, sample_time, water_level_high and water_level_low, sensorMHValue (store sensor value), bumpStatus, waterStatus. Then we declare some more pins to later control for the light.

For the setup() method, as its name suggests, this function is mainly set up with the pins for the led and water pump using pinMode(), and for setup the Wifi, we call the function connectWiFi().

For the connectWifi() function, we will check if we cannot access the Wifi, then we will print out the Local IP address.

Before going to the loop() function, we had to mention the wateringProcess(). Firstly, we get the water status by using the digitalRead() and the sensor value from the function getSensorSampleValue(). We use the sensor value to turn off or turn on the water pump (-1: less water, 0: enough water, +1 : more water).

For the loop() function, we will first call out the waterProcess(). Next, we will check if the client is available. If it is available, we can receive the command from the app through Wifi. Therefore, depending on the command, we can control the light 1, light 2 and the water pump.

b. Code for Arduino:

Done all the wiring components, we moved to the coding for the project. First, we used 3 external libraries to control the components. The SPI and MFRC522 libraries are used for the RFID NFC RC522 to use some functions in the library like PICC_IsNewCardPresent(), PICC_ReadCardSerial(), etc. Another library which is Servo.h is used for the RC Servo to rotate the gears to the degree that we wanted by using servo.write(degree).

After done include all the libraries and define all the necessary variables. We first used the PICC_IsNewCardPresent() to check if there is any RFID card present. If not, it will return and start the loop again, otherwise it will continue to go to the next function PICC_ReadCardSerial() to see if it can read the ID on the card. With all the conditions met to pass through, it will check the doorStatus. If the doorStatus is open (stand for 1) and the right ID card was scanned, the doorStatus will change to close (stand for 1) and the servo will rotate to 180 degree. If the doorStatus is closed (stand for 1) with the right ID card scanned, the doorStatus will change to 0 (open) and the servo will rotate 90 degrees. If the wrong ID card is given, no matter the servo in which degrees, it will rotate to 90 degrees and the buzzer will alarm in 1000 milliseconds.

III. Conclusion

After nearly three months of commitment with intentions and projects, our members not only gain more opportunities to discover the information related to the IoT project but also implement that knowledge into the final result: The Smart House Project. The associated network contains various electrical objects represented by IoT own sensors and embedded software to transfer databases via the Internet. Acquire the knowledge of the computer's system, how it transfers data from devices to devices. Discover how to integrate multiple Arduino and power supplies into one Arduino, minimizing circuit complexity. And most importantly, the experience and knowledge in practicing with Arduino devices through the repetitive loop of bugs during product making and testing stages. Finally, an essential factor determining the project's success comes from the consciousness, effort of researching, investigating our members in learning. Build a detailed plan as well as plan implementation tactics according to the timeline, progress.

There are a few points about the project that the whole team wants to improve with the aim of optimizing the functions and making it easier for members to execute:

- Format the UI and UX of the Application.
- Draw a diagram of the whole system before beginning to connect each feature.
- Make a well-thought-out plan by detailed the task allocated, deadline, step-by-step.
- Having more consultation about the writing section with lecturers.
- Currently, the smart home system can only be controlled via Wifi using LAN, to improve we need to use the WAN network to be able to access the smart home system at any place.
- Instead of using MIT Application (inherent drag and drop tool), my team wants to program a mobile application reserved for controlling smart home products (using Java, Kotlin).

The team's final judgment is for the Project; it is an achievement after the learning process; the finished product is the proof of the members' success and efforts. Furthermore Project will be able to develop even more in the future after consolidating some more details.

IV. Code

ESP8266 NodeMCU part

```
// WATERING PLANT
int const SENSOR_MH_D_PIN = 15;
int const SENSOR_MH_A_PIN = A0;
int const T_RELAY_PIN = 5;
int const TIME_TO_GET_SAMPLE = 5000; //5s
int const SAMPLE_TIME = 500; //0.5 s
int const TREE_WATER_LEVEL_HIGH = 600;
int const TREE_WATER_LEVEL_LOW = 300;

int sensorMHValue = 0;//store sensor value
int bumpStatus = 0;
int waterStatus = 0;// -1: less water, 0: enough water,  +1 : more water


#include <ESP8266WiFi.h>
WiFiClient client;
WiFiServer server(80);
const char* ssid = "Boost Juice";
const char* password = "lovelife";
String  command =""; // Command received from Android device

// Set Pins
int pin2= 2;
int pin4 = 4;
int pin3 = 3;
int pin13 = 13;

void setup()
{
  Serial.begin(115200);
  pinMode(pin2, OUTPUT);
  pinMode(pin4, OUTPUT);
  pinMode(pin3, OUTPUT);
  pinMode(pin13, OUTPUT);

       // WATERING PLANTS
  pinMode(SENSOR_MH_D_PIN,INPUT);
  pinMode(T_RELAY_PIN, OUTPUT);

  connectWiFi();
  server.begin();
}

void loop(){
    wateringProcess();

    client = server.available();
    if (!client) return;
    command = checkClient();   // Receive commands from the Mobile
```

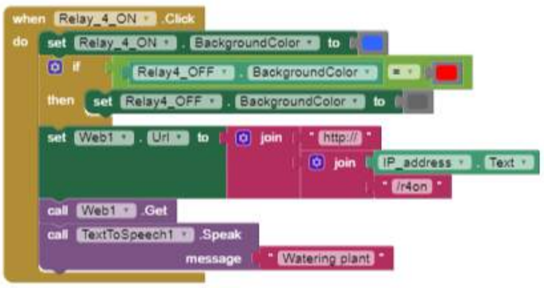
```
      if (command == "r1on"  || command == "roomoneon")      digitalWrite(pin2,HIGH);
   else if (command == "r1off" || command == "roomoneoff")     digitalWrite(pin2,LOW);
   else if (command == "r2on"  || command == "roomtwoon")      digitalWrite(pin4,HIGH);
   else if (command == "r2off" || command == "roomtwooff")     digitalWrite(pin4,LOW);
   else if (command == "r4on"  || command == "wateron")        digitalWrite(pin13,HIGH);
   else if (command == "r4off" || command == "wateroff")       digitalWrite(pin13,LOW);
   else if (command == "allon" || command == "on"){
     digitalWrite(pin2,HIGH);
     digitalWrite(pin4,HIGH);
   }
   else if (command == "alloff" || command == "off" || command == "desligar todos") {
     digitalWrite(pin2,LOW);
     digitalWrite(pin4,LOW);
   }

   sendBackEcho(command); // send command echo back to android device
   command = "";
}

/* connecting WiFi */
void connectWiFi()
{
  Serial.println("Connecting to WIFI");
  WiFi.begin(ssid, password);
  while ((!(WiFi.status() == WL_CONNECTED))){
    delay(300);
    Serial.print("..");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("NodeMCU Local IP is : ");
  Serial.print((WiFi.localIP()));
}

/* check command received from Android Device */
String checkClient (void)
{
  while(!client.available()) delay(1);
  String request = client.readStringUntil('\r');
  request.remove(0, 5);
  request.remove(request.length()-9,9);
  return request;
}

/* send command echo back to android device */
void sendBackEcho(String echo){
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println("");
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");
  client.println(echo);
  client.println("</html>");
  client.stop();
  delay(1);
}

void wateringProcess(){
  int sensorStatus = digitalRead(SENSOR_MH_D_PIN);
  int sensorValue = getSensorSampleValue();
  if(sensorStatus == 0){
    if(sensorValue > TREE_WATER_LEVEL_HIGH){
      digitalWrite(T_RELAY_PIN, HIGH);
      bumpStatus = 1;
      waterStatus = -1;
    }else if(sensorValue < TREE_WATER_LEVEL_LOW){
      digitalWrite(T_RELAY_PIN, LOW);
      bumpStatus = 0;
      waterStatus = +1;
    }else{
      digitalWrite(T_RELAY_PIN, LOW);
      bumpStatus = 0;
      waterStatus = 0;
    }
  }else{
    digitalWrite(T_RELAY_PIN, HIGH);
      bumpStatus = 1;
      waterStatus = -1;
  }
}

int getSensorSampleValue(){
  int value = 0;
  int t = TIME_TO_GET_SAMPLE/ SAMPLE_TIME;
  int total = 0;
  for(int i =0; i < t ; i++){
    total += analogRead(SENSOR_MH_A_PIN);
    delay(t);
  }
  return total / t;
}
```

```
void printToSerialPort(){
  Serial.print("Bumper enable: "); Serial.println(bumpStatus);
  Serial.print("Sensor value: "); Serial.println(sensorMHValue);
}
```

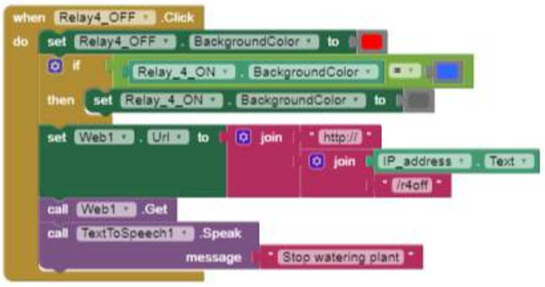
Anduino part

```
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>

#define SS_PIN 10
#define RST_PIN 9
Servo myservo;
int pos=0;
int count = 0;
MFRC522 mfrc522(SS_PIN, RST_PIN);   // Create MFRC522 instance.
int lock = 0;
int doorStatus = 0;
void setup()
{
  Serial.begin(9600);   // Initiate a serial communication
  SPI.begin();      // Initiate  SPI bus
  mfrc522.PCD_Init();   // Initiate MFRC522
  Serial.println("Approximate your card to the reader...");
  Serial.println();
  pinMode(8,OUTPUT);
  myservo.attach(6);
}
void loop()
{
  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  //Show UID on serial monitor
  Serial.print("UID tag :");
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
     Serial.print(mfrc522.uid.uidByte[i], HEX);
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  Serial.print("Message : ");
  content.toUpperCase();

  if (content.substring(1) == "17 CA 93 C9"){
    lock = 1;
    count++;
  }

  if (lock == 1){
    // Reset the lock condition
    lock = 0;

    if (doorStatus == 0) doorStatus = 1; // If the door is lock, change the status to 1
    else doorStatus = 0;   // If it were opening, change the status to 0

    //If it were locked, open the door
    if (doorStatus){
      Serial.print("Accesed\n");
      Serial.print("Open the door\n");
      myservo.write(180);
      delay(1000);

    // If it were open, close the door
    } else {
      Serial.print("Close the door\n");
      myservo.write(90);
      delay(1000);

    }
  // If it does not matched the UID from the card, close the door and also alarm
  } else {
    Serial.print("Acessed denied\n");
    myservo.write(90);
    digitalWrite(8, HIGH);
    delay(1000);
    digitalWrite(8, LOW);

  }
  Serial.print(count);
  Serial.println();
}
```

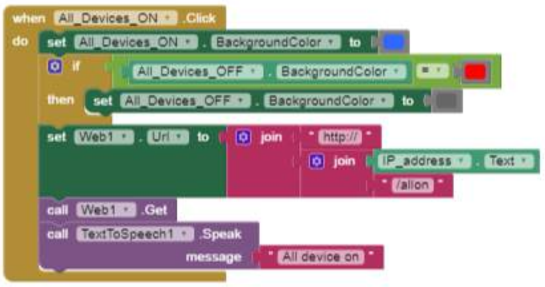
<MIT App Application>



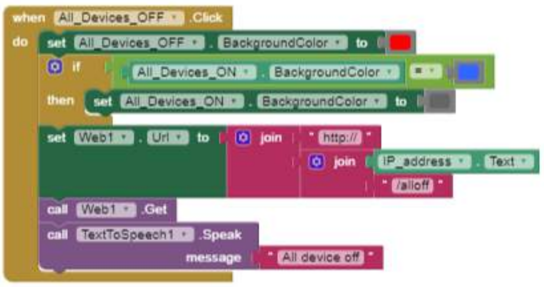
```
when Relay_4_ON .Click
do  set Relay_4_ON . BackgroundColor to 🟦
    if   Relay4_OFF . BackgroundColor = 🟥
    then set Relay4_OFF . BackgroundColor to ⬛
    set Web1 . Url to join "http://"
                           join IP_address . Text
                                "/r4on"
    call Web1 .Get
    call TextToSpeech1 .Speak
         message "Watering plant"
```



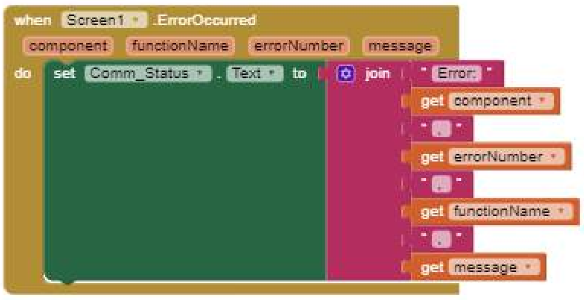
```
when Relay4_OFF .Click
do  set Relay4_OFF . BackgroundColor to 🟥
    if   Relay_4_ON . BackgroundColor = 🟦
    then set Relay_4_ON . BackgroundColor to ⬛
    set Web1 . Url to join "http://"
                           join IP_address . Text
                                "/r4off"
    call Web1 .Get
    call TextToSpeech1 .Speak
         message "Stop watering plant"
```



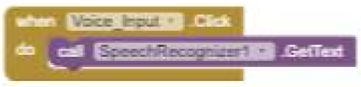
```
when All_Devices_ON .Click
do  set All_Devices_ON . BackgroundColor to 🟦
    if   All_Devices_OFF . BackgroundColor = 🟥
    then set All_Devices_OFF . BackgroundColor to ⬛
    set Web1 . Url to join "http://"
                           join IP_address . Text
                                "/allon"
    call Web1 .Get
    call TextToSpeech1 .Speak
         message "All device on"
```



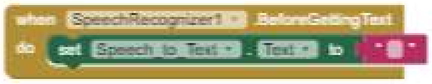
```
when All_Devices_OFF .Click
do  set All_Devices_OFF . BackgroundColor to 🟥
    if   All_Devices_ON . BackgroundColor = 🟦
    then set All_Devices_ON . BackgroundColor to ⬛
    set Web1 . Url to join "http://"
                           join IP_address . Text
                                "/alloff"
    call Web1 .Get
    call TextToSpeech1 .Speak
         message "All device off"
```



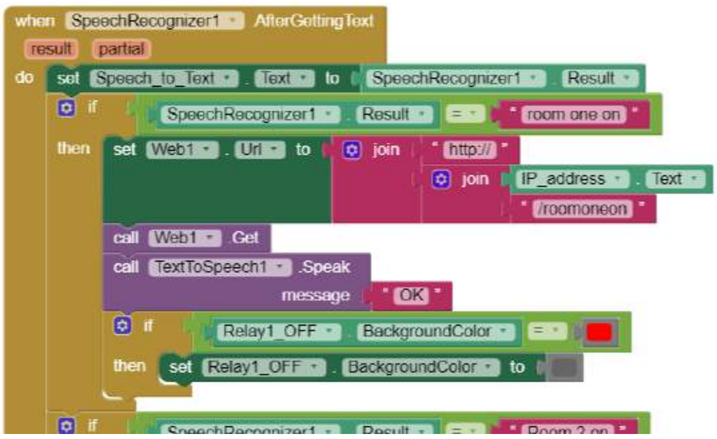
```
when Screen1 .ErrorOccurred
 component  functionName  errorNumber  message
do  set Comm_Status . Text to join "Error: "
                                   get component
                                   " "
                                   get errorNumber
                                   " "
                                   get functionName
                                   " "
                                   get message
```



```
when Voice_Input .Click
do  call SpeechRecognizer1 .GetText
```



```
when SpeechRecognizer1 .BeforeGettingText
do  set Speech_to_Text . Text to " "
```



```
when SpeechRecognizer1 .AfterGettingText
 result  partial
do  set Speech_to_Text . Text to SpeechRecognizer1 . Result
    if   SpeechRecognizer1 . Result = "room one on"
    then set Web1 . Url to join "http://"
                                join IP_address . Text
                                     "/roomoneon"
         call Web1 .Get
         call TextToSpeech1 .Speak
              message "OK"
         if   Relay1_OFF . BackgroundColor = 🟥
         then set Relay1_OFF . BackgroundColor to ⬛
    if   SpeechRecognizer1 . Result = "Room 2 on"
```

Link package: https://drive.google.com/file/d/1vp2VZQpNtIT_3VhX5Kd8s2nuZvcTHl9q/view

V. Reference

1. keebie81, 2021. *RFID Door Unlock*. [online] Instructables. Available at: <https://www.instructables.com/RFID-Door-Unlock/> [Accessed 18 January 2021].

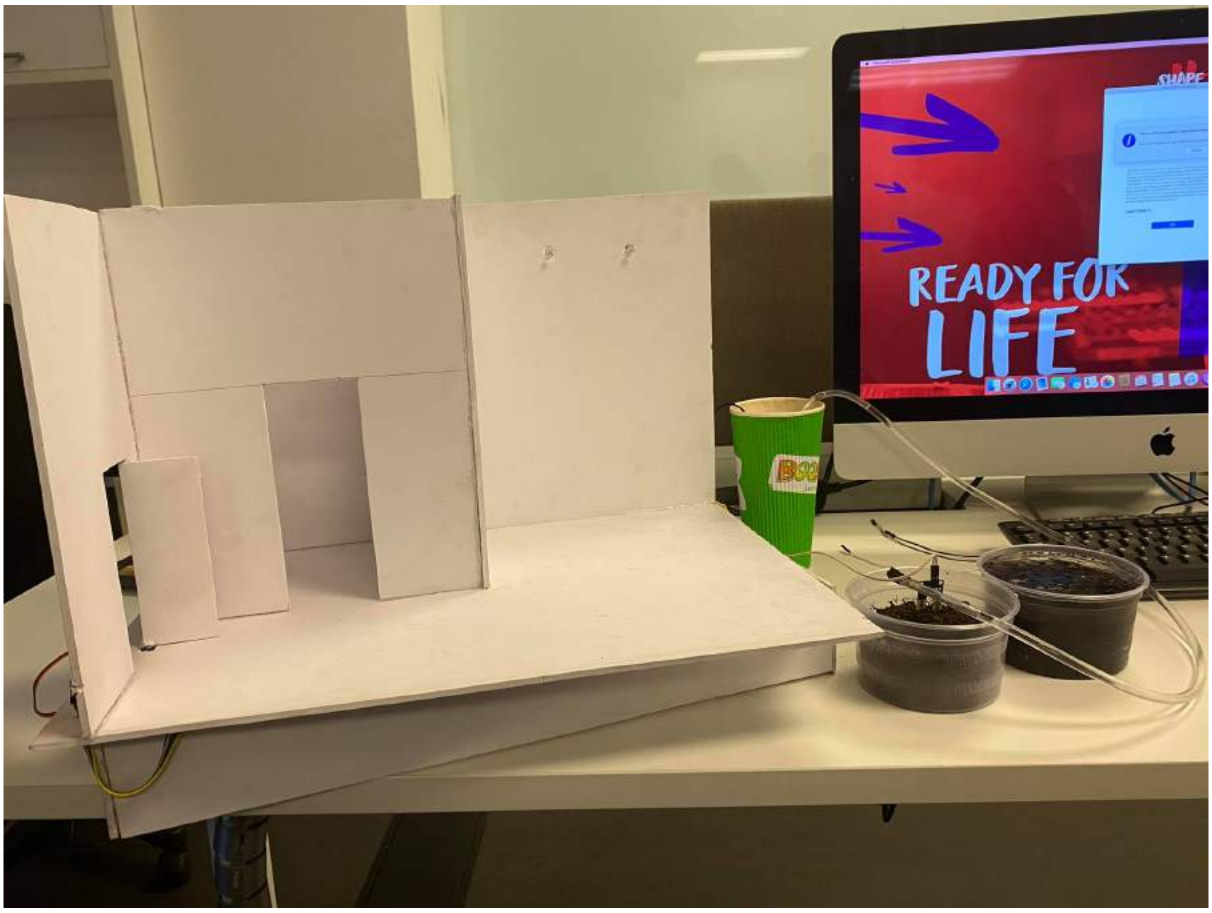
2. mjrovai, 2021. *Voice Activated Control With Android And Nodemcu*. [online] Instructables. Available at: <https://www.instructables.com/Voice-Activated-Control-With-Android-and-NodeMCU/?fbclid=IwAR2Cdvngi48zZzwtmhGZKTJevQLDRsBQIGgWYlV9IRZSB19FavkpL4UwvCo> [Accessed 18 January 2021].

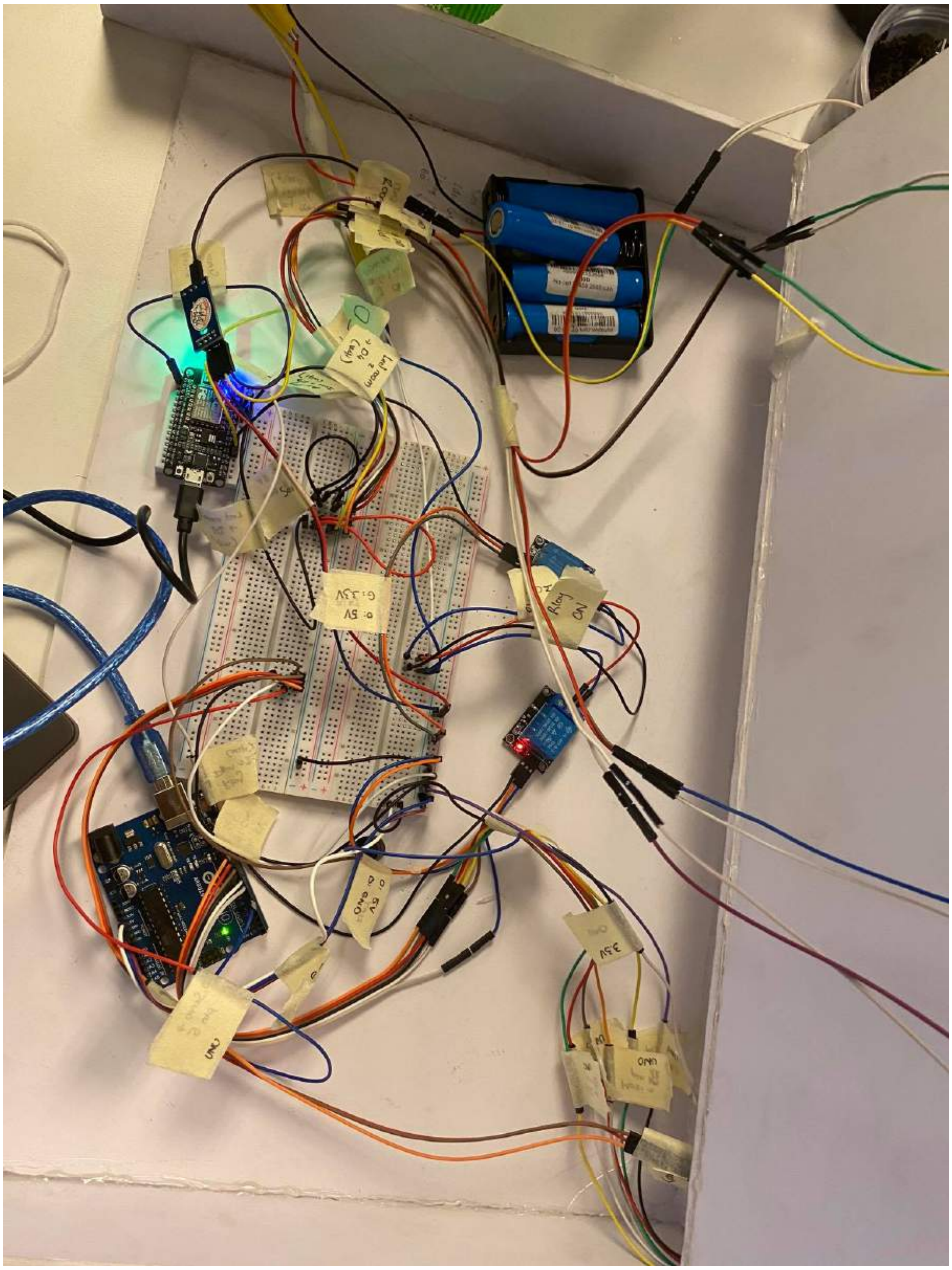
3. upGrad blog. 2021. *20 Exciting Iot Project Ideas & Topics For Beginners [2021] | Upgrad Blog*. [online] Available at: <https://www.upgrad.com/blog/iot-project-ideas-topics-for-beginners/> [Accessed 18 January 2021].

4. Electronics Hub. 2021. *How To Connect ESP8266 To Wifi | A Beginner's Guide*. [online] Available at: <https://www.electronicshub.org/connect-esp8266-to-wifi/> [Accessed 18 January 2021].

5. Sarikas, C., 2021. *7 Steps To A Successful MIT Application*. [online] Blog.prepscholar.com. Available at: <https://blog.prepscholar.com/mit-application-deadline-fee-essays> [Accessed 18 January 2021].

VI. Appendix


READY FOR
LIFE


G: 3.3V
o: 5V
Relay
ON
3.5V
UNO