VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

“J” – COMPONENT REPORT

COURSE CODE / NAME	ECE4003– EMBEDDED SYSTEM DESIGN
PROGRAM / YEAR / SEM	B.Tech (ECE & ECM)/III Year /Fall 2018-19
DATE OF SUBMISSION	17.10.18
PROJECT TITLE	Faculty Availability System

TEAM MEMBERS DETAILS	**REGISTER NO.**	**NAME**
	16BEC1003	Divya Agarwal
	16BEC1074	Meghna Roy Chowdhury
	16BLC1127	Mayuri Patil

EVALUATION ITEMS	**Yes (√) / No (x)**
The project has achieved the objective?	
Level of Knowledge Gained While Completing the Project was satisfactory?	
Are the students having clear idea on their proposed idea and have they acquired to carry forward it?	
Are the contribution made by the individuals towards attaining objective of the project was satisfactory?	
Are the submitted report and presentation made by each team member was satisfactory?	

COURSE INCHARGE NAME	**Prof. V. PRAKASH**	**MARKS**	
REVIEWER’S NAME & SIGN			

Objective of the Project:

As a student, one seldom wastes time to check if a faculty is available in his/her cabin. The objective of our project is to solve this issue by making an app which senses if the cabin of a faculty is locked and updates it in an app.

Hardware Requirements:

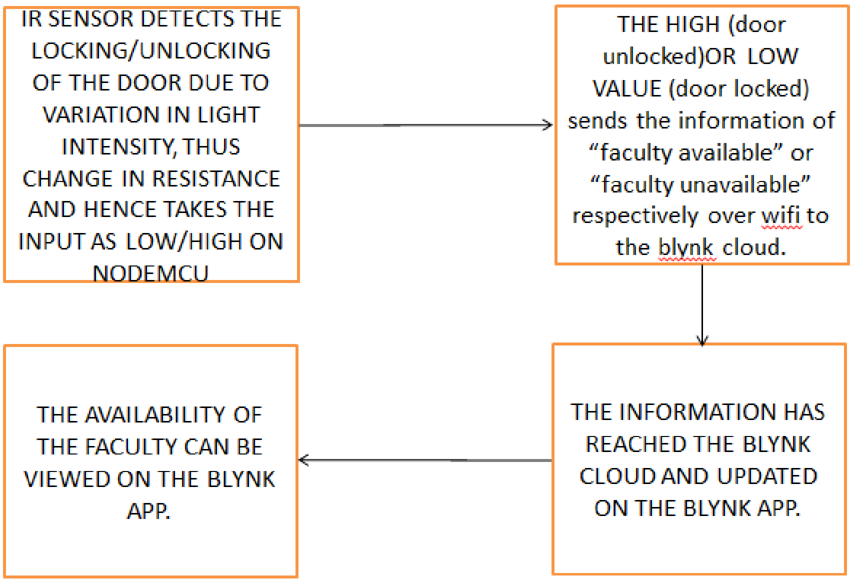
S. No	Component Name	Specification	Quantity	Cost (in Rs.)
1.	NODE MCU		1	290
2.	IR Sensor		3	60*3=180
3.	Jumper Wire	Female to Female	10	25
4.	USB Cable	Type A port	1	150
5.	Portable Charger		1	300

Software Requirements:

S. No	Component Name	Specification
1.	Arduino IDE	Version 1.9.5
2.	Blynk	

Overall cost of the Project: …945…… (in Rupees)

Block Diagram:



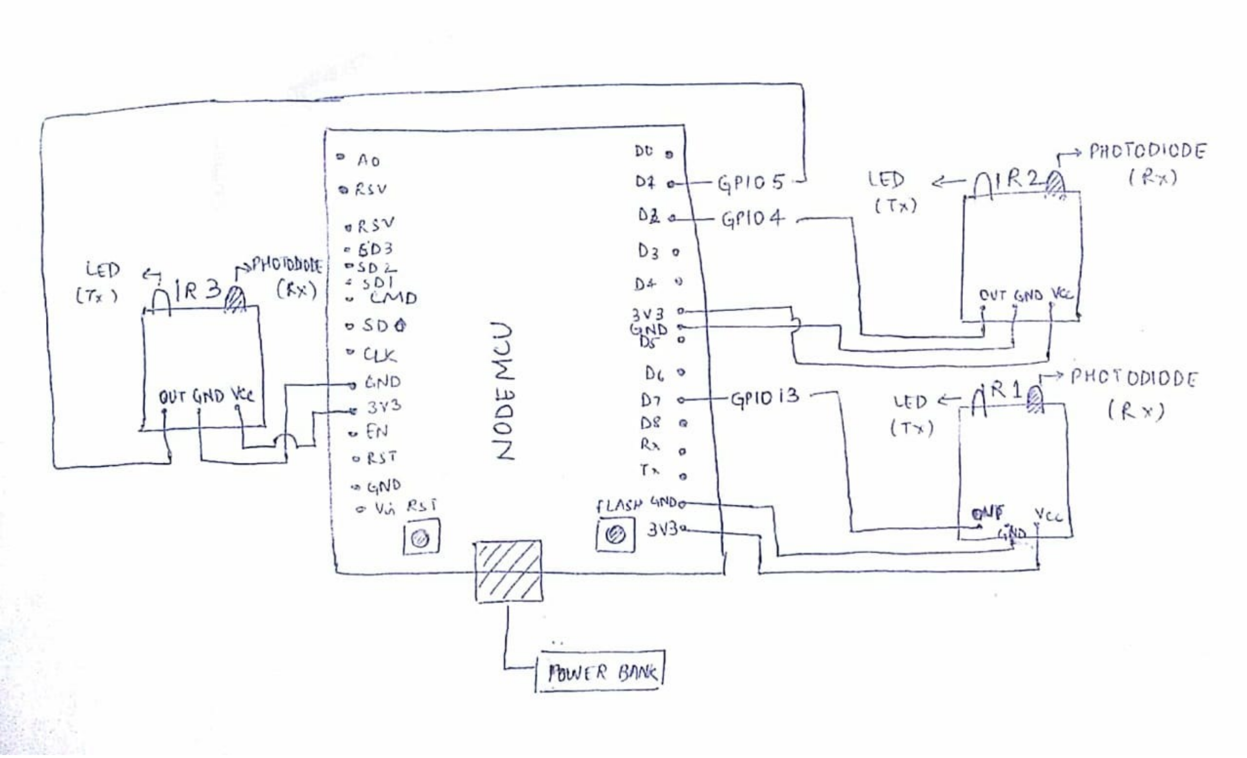
IR SENSOR DETECTS THE LOCKING/UNLOCKING OF THE DOOR DUE TO VARIATION IN LIGHT INTENSITY, THUS CHANGE IN RESISTANCE AND HENCE TAKES THE INPUT AS LOW/HIGH ON NODEMCU

THE HIGH (door unlocked)OR LOW VALUE (door locked) sends the information of “faculty available” or “faculty unavailable” respectively over wifi to the blynk cloud.

THE INFORMATION HAS REACHED THE BLYNK CLOUD AND UPDATED ON THE BLYNK APP.

THE AVAILABILITY OF THE FACULTY CAN BE VIEWED ON THE BLYNK APP.

Schematic Diagram:

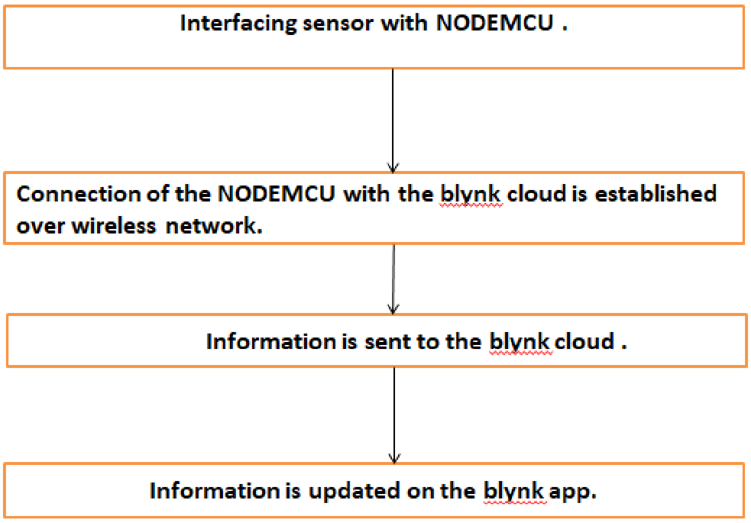


Project Description:

This project basically involves making a system which senses if the cabin of a faculty is locked and updates it in an app. We have used an infrared sensor , NODEMCU and blynk (displays information and also provides platform to build and publish separate apps). The project involves interfacing sensor with NODEMCU followed by connection of the NODEMCU with the blynk cloud over wireless network. The information thus collected due to the high or low input taken by the sensor due to its change in resistance caused by the locking or unlocking of the door is sent to the blynk cloud . The information on the cloud is then updated on the blynk app.

Further if an app is created and published using the blynk app formats , this information can then be separately viewed on an app by students , thus telling them if the faculty is available or not.

Flow chart:


Interfacing sensor with NODEMCU .
Connection of the NODEMCU with the blynk cloud is established over wireless network.
Information is sent to the blynk cloud .
Information is updated on the blynk app.

Program:

Part1: Interfacing IR Sensor and NODEMCU

```
int inputPin = 13; // Connecting D7 to IR sensor Vout

int val = 0; // variable for reading the pin status

void setup()

{  Serial.begin(115200);

  pinMode(inputPin, INPUT); // declare Infrared sensor as input

}
```

```
void loop()

{

  val = digitalRead(inputPin); // read input value

  if (val == HIGH)

  { // check if the input is HIGH

    Serial.println("HIGH");

  }

  else

  {

    Serial.println("LOW");

  }

}
```

Part 2: Connecting to the wireless Network

```
#include <ESP8266WiFi.h>

char ssid[]="Em";

char pass[]="meghna123";


void setup() {

 Serial.begin(115200);

 delay(500);

 Serial.print("Connecting to...");

 Serial.print(ssid);

 delay(500);


 //initate Wifi

 WiFi.disconnect();

 WiFi.begin(ssid,pass);

 //wait till connection is established


while(WiFi.status()!=WL_CONNECTED)
delay(500);

   Serial.println(",");

 }

 Serial.print("SSID.... ,");

 Serial.println(WiFi.SSID());

 Serial.println("Connected successfully!");


 Serial.print("IP address of NodeMCU:
");

 Serial.println(WiFi.localIP());
```

```
  Serial.print("MAC address of NodeMCU:  ");

  Serial.println(WiFi.macAddress());

  WiFi.printDiag(Serial); //prints credentials

  }

void loop() {

  // put your main code here, to run repeatedly:

}
```

Part 3: Interfacing with Blynk

```
 #include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>


#define ir1 13 //d7

#define ir2 5 //d1

#define ir3 4 //d2


char auth[] = "c160a5a3e45e4652bbc24cc76945ee91";


char ssid[] = "Em";

char pass[] = "meghna123";


WidgetLCD lcd1(V1);   //FOR BLYNK

WidgetLCD lcd2(V2);

WidgetLCD lcd3(V3);


int val1 = 0; // variable for reading IR1

int val2 = 0; // variable for reading IR2

int val3 = 0; // variable for reading IR3

void setup()

{

  // Debug console
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);

  lcd1.clear(); //Use it to clear the LCD Widget
   lcd2.clear(); //Use it to clear the LCD Widget
    lcd3.clear(); //Use it to clear the LCD Widget
}

void loop()
{
  Blynk.run();
   val1 = digitalRead(ir1); // read input value
   val2 = digitalRead(ir2); // read input value
   val3 = digitalRead(ir3); // read input value

if (val1 == HIGH) // no obstacle
   { // check if the input is HIGH
      Serial.println("HIGH");
      lcd1.clear();
      lcd1.print(0, 0, "Faculty 1  is available");
   }
   else
   {
      Serial.println("LOW");
      lcd1.clear();
      lcd1.print(0, 0, "faculty 1 is not available");
   }
```

```
     if (val2 == HIGH) // no obstacle
  { // check if the input is HIGH
    Serial.println("HIGH");
    lcd2.clear();
    lcd2.print(0, 0, "Faculty 2  is available"); // use: (position X: 0-15, position Y: 0-1, "Message you want to print")
  }
  else
  {
    Serial.println("LOW");
    lcd2.clear();
    lcd2.print(0, 0, "faculty 2 is not available"); // use: (position X: 0-15, position Y: 0-1, "Message you want to print")
  }

  if (val3 == HIGH) // no obstacle
  { // check if the input is HIGH
    Serial.println("HIGH");
    lcd3.clear();
    lcd3.print(0, 0, "Faculty 3  is available"); // use: (position X: 0-15, position Y: 0-1, "Message you want to print")
  }
  else
  {
    Serial.println("LOW");
    lcd3.clear();
    lcd3.print(0, 0, "Faculty 3 is not available"); // use: (position X: 0-15, position Y: 0-1, "Message you want to print")
  }
```

```
}
```

Part 4: App Template

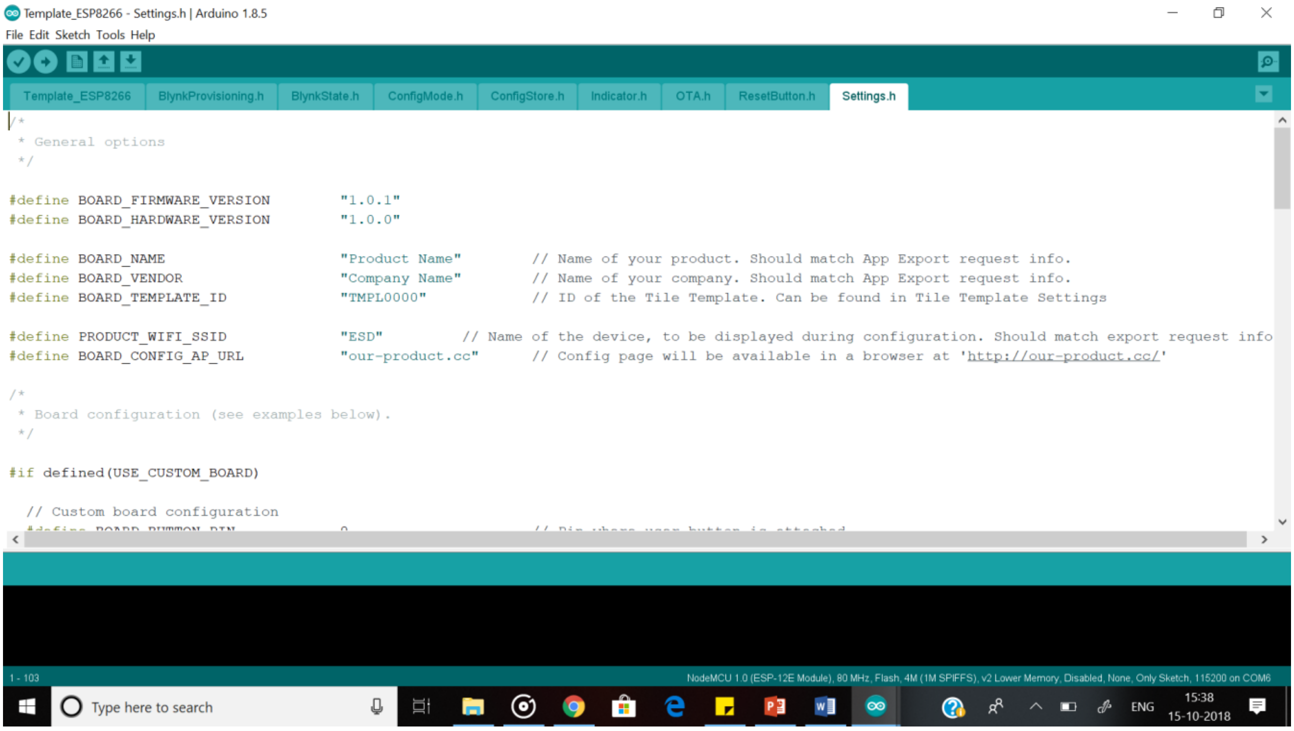
```
//#define USE_SPARKFUN_BLYNK_BOARD    // Uncomment the board you are using
#define USE_NODE_MCU_BOARD        // Comment out the boards you are not using
//#define USE_WITTY_CLOUD_BOARD
//#define USE_CUSTOM_BOARD          // For all other ESP8266-based boards -
                              // see "Custom board configuration" in Settings.h

#define APP_DEBUG        // Comment this out to disable debug prints
#define ir1 4 //d0
#define ir2 5 //d1
#define ir3 6 //d2



#define BLYNK_PRINT Serial
#include<BlynkSimpleEsp8266.h>
#include "BlynkProvisioning.h"




char auth[] = "c160a5a3e45e4652bbc24cc76945ee91";



WidgetLCD lcd1(V1);
WidgetLCD lcd2(V2);
WidgetLCD lcd3(V3);
```

```
int val1 = 0; // variable for reading IR1
int val2 = 0; // variable for reading IR4
int val3 = 0; // variable for reading IR3

void setup() {
  delay(500);
  Serial.begin(115200);

  lcd1.clear(); //Use it to clear the LCD Widget
   lcd2.clear(); //Use it to clear the LCD Widget
    lcd3.clear(); //Use it to clear the LCD Widget

    BlynkProvisioning.begin();
}

void loop()
{
  BlynkProvisioning.run();

   val1 = digitalRead(ir1); // read input value
   val2 = digitalRead(ir2); // read input value
   val3 = digitalRead(ir3); // read input value



if (val1 == HIGH) // no obstacle
   { // check if the input is HIGH
      Serial.println("HIGH");
      lcd1.clear();
    lcd1.print(0, 0, "Faculty 1  is available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")
  }
  else
  {
    Serial.println("LOW");
    lcd1.clear();
    lcd1.print(0, 0, "faculty 1 is not available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")
  }



   if (val2 == HIGH) // no obstacle
  { // check if the input is HIGH
    Serial.println("HIGH");
    lcd2.clear();
    lcd2.print(0, 0, "Faculty 2  is available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")
  }
  else
  {
    Serial.println("LOW");
    lcd2.clear();
    lcd2.print(0, 0, "faculty 2 is not available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")
  }

  if (val3 == HIGH) // no obstacle
  { // check if the input is HIGH
```

```
    Serial.println("HIGH");

    lcd3.clear();

    lcd3.print(0, 0, "Faculty 3  is available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")

  }

  else

  {

    Serial.println("LOW");

    lcd3.clear();

    lcd3.print(0, 0, "Faculty 3 is not available"); // use: (position X: 0-15, position Y: 0-1,
"Message you want to print")

  }



}
```

Simulation Output:

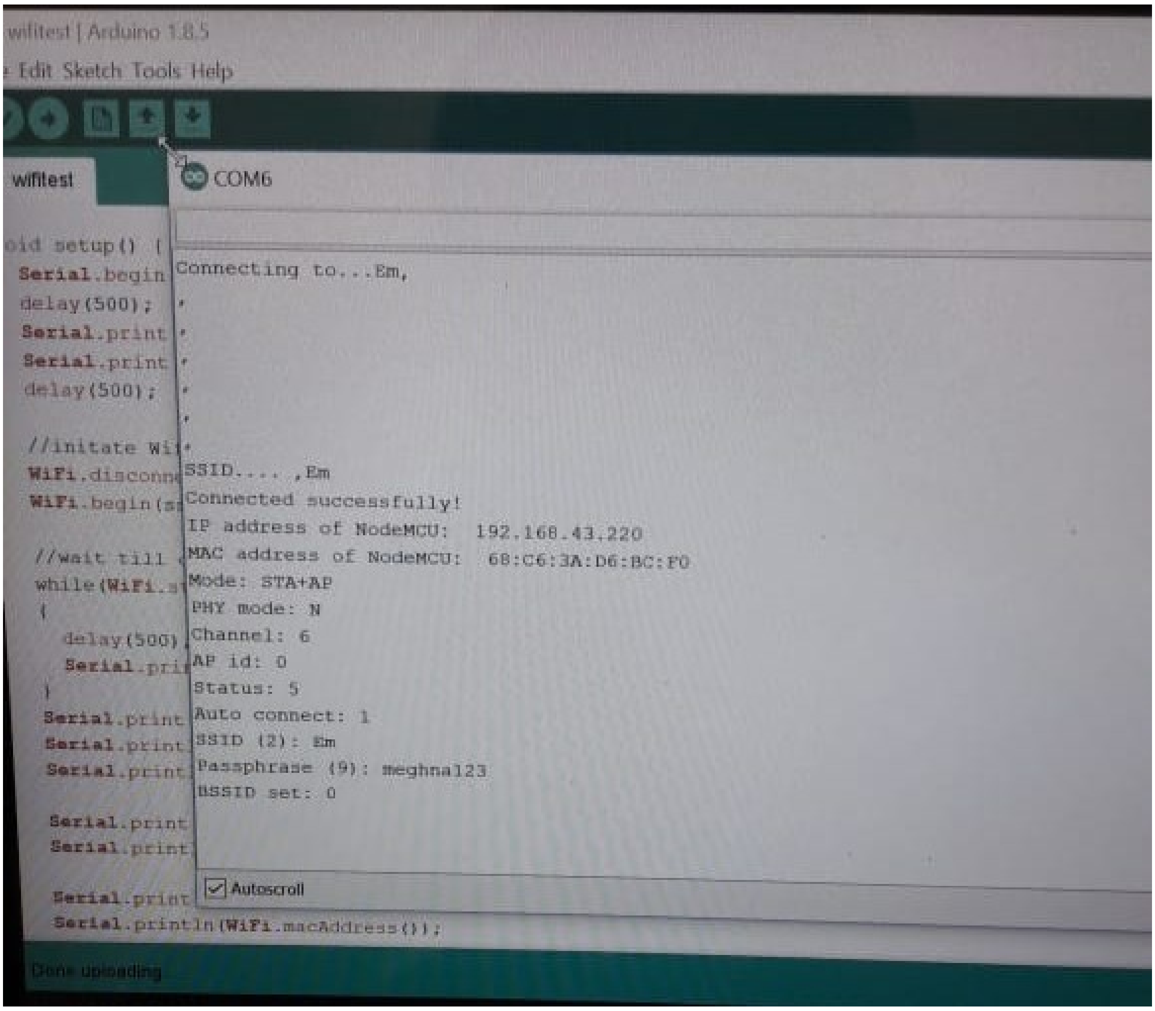
Part 1: Interfacing with the IR Sensor



```
/*
 * General options
 */

#define BOARD_FIRMWARE_VERSION        "1.0.1"
#define BOARD_HARDWARE_VERSION        "1.0.0"

#define BOARD_NAME                    "Product Name"        // Name of your product. Should match App Export request info.
#define BOARD_VENDOR                  "Company Name"        // Name of your company. Should match App Export request info.
#define BOARD_TEMPLATE_ID             "TMPL0000"            // ID of the Tile Template. Can be found in Tile Template Settings

#define PRODUCT_WIFI_SSID             "ESD"         // Name of the device, to be displayed during configuration. Should match export request info
#define BOARD_CONFIG_AP_URL           "our-product.cc"      // Config page will be available in a browser at 'http://our-product.cc/'

/*
 * Board configuration (see examples below).
 */

#if defined(USE_CUSTOM_BOARD)

  // Custom board configuration
  #define BOARD_BUTTON_PIN            0                     // Pin where user button is attached
```

```
1 - 103
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, 4M (1M SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM6
```

Part 2: Connecting with Wireless Network

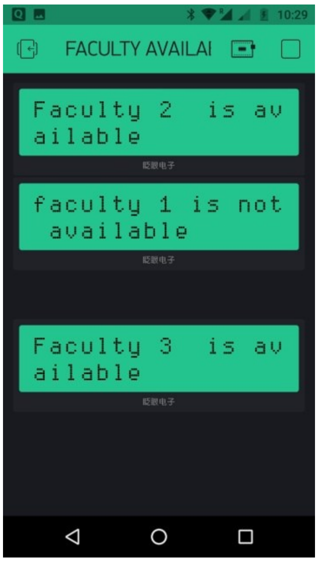


```
oid setup() {
 Serial.begin
 delay(500);
 Serial.print
 Serial.print
 delay(500);

 //initate Wi
 WiFi.disconn
 WiFi.begin(s

 //wait till
 while(WiFi.s
 {
   delay(500)
   Serial.pri
 }
 Serial.print
 Serial.print
 Serial.print

 Serial.print
 Serial.print

 Serial.print
 Serial.println(WiFi.macAddress());
```

```
Connecting to...Em,
.
.
.
.
.
.
SSID.... ,Em
Connected successfully!
IP address of NodeMCU:  192.168.43.220
MAC address of NodeMCU:  68:C6:3A:D6:BC:F0
Mode: STA+AP
PHY mode: N
Channel: 6
AP id: 0
Status: 5
Auto connect: 1
SSID (2): Em
Passphrase (9): meghna123
BSSID set: 0
```

Part 3: Interfacing with Blynk

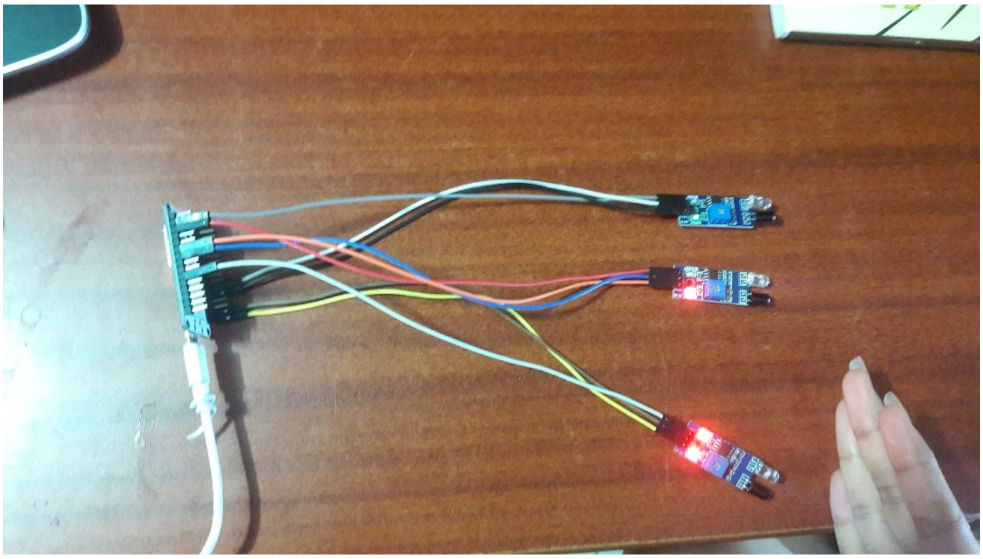
i. IR Sensor 1 is HIGH => Faculty 1 is not available



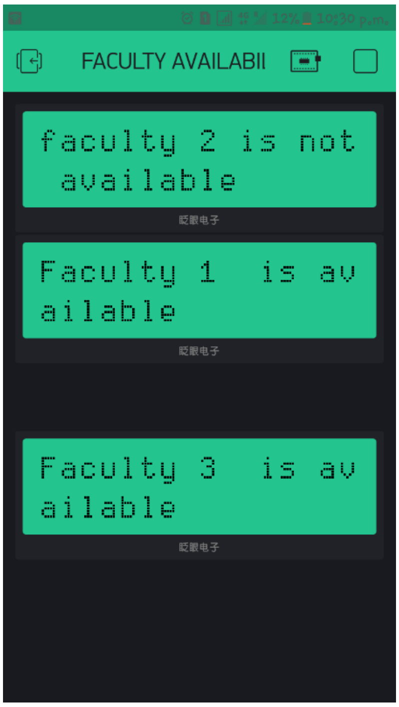
Faculty 2 is available

faculty 1 is not available

Faculty 3 is available



ii. IR Sensor 2 is HIGH => Faculty 2 is not available

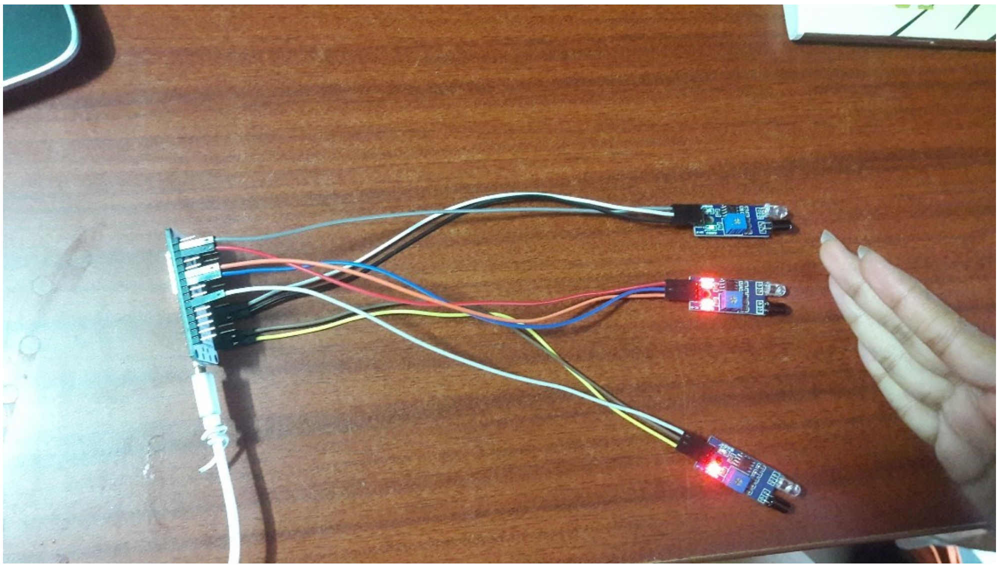


FACULTY AVAILABII

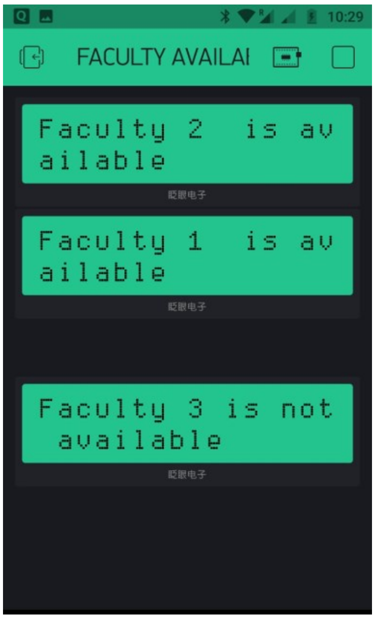
```
faculty 2 is not
 available
```

```
Faculty 1  is av
ailable
```

```
Faculty 3  is av
ailable
```



iii. IR Sensor 3 is HIGH => Faculty 3 is not available



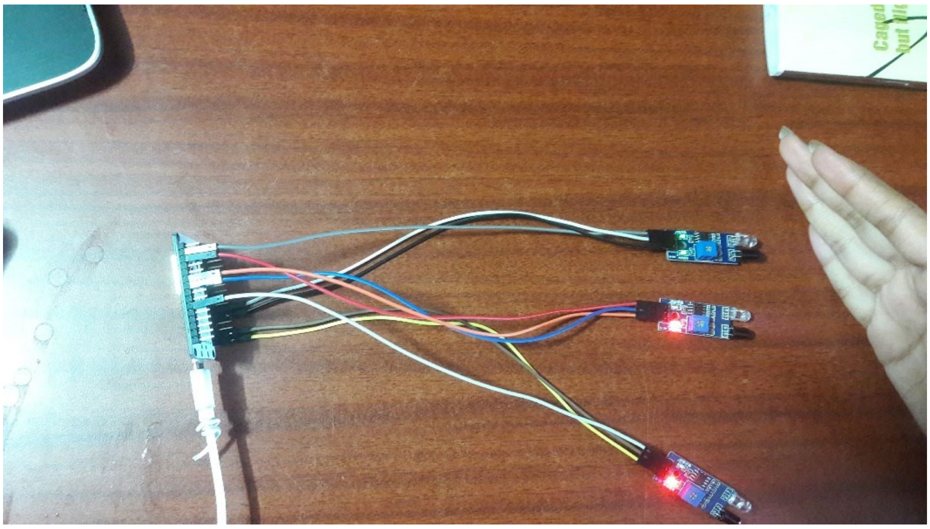
10:29

FACULTY AVAILAI

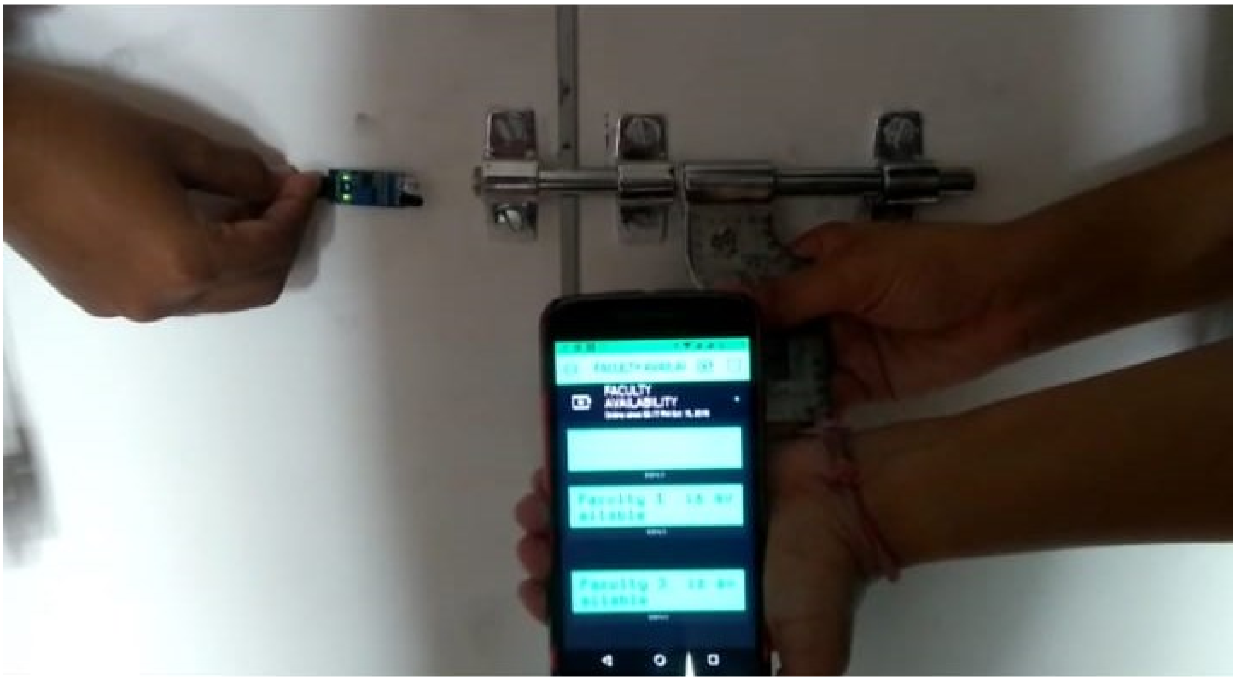
```
Faculty 2  is av
ailable
```

```
Faculty 1  is av
ailable
```

```
Faculty 3 is not
 available
```



Implementation Output(Photo):



Inference:

Following were the observations made while making the project:

1. For normal program execution GPIO0 and GPIO2 need to be pulled up to Vcc (3.3V) and GPIO15 needs to be pulled to GND.
2. Hence, GPIO0 of nodemcu should not be used for external connections. It is used for flash memory and is supposed to be low at all times.
3. IR sensor is a better detector than reed switch, as reed switch requires a magnet to detect an obstacle whereas an IR sensor detects an object in the radius of 10.2 cm (average distance calculated for 3 IR sensors).

Concepts Learned:

- Interfacing IR sensors, reed switch with nodemcu.
- Reading data from sensor to nodemcu controller, and then to a cloud and app.
- Working of IR Sensor
- Working of Reed switch
- Difference between NodeMCU and Arduino
- Using IFTTT(HTTP protocol) as cloud
- MQTT Protocol – It's a protocol used for smart lock IOT base and similar systems
- Thingspeak as a cloud application
- Blynk Cloud and App

Applications:

- Can be used for home security systems. The owner will be notified through the app if an intruder tries to open the door.
- On modifying the project, we lock/unlock a door by sending a command/message to the controller from the smartphone app.
- This can be used in educational institutes/ big organizations so that the students/employees can be notified through an app, if a particular faculty/manager is present in their cabin or not. This can save time.

Difficulties faced:

- Reading data from the cloud.
- Configuring the blynk app
- Uploading code to Arduino IDE
- Wifi Connection issues.
- Publishing the app on the app store

Timeline:

- **Review 1 –** circuit was made and code was written.
- **Review 2 –** system working with 1 IR sensor and app
- **Review 3(open house) –** complete system working with 3 IR sensors and App