**Engineering Exploration – Introduction to Internet of Things (23EE001)**



**PROJECT REPORT**

**on**

**Smoke Detector**

**BATCH-2022**

**Project Mento Student Name & ID**

Dr. Monika Parmar 1.Koushik Adak (2211981201)

2.Krishna Pradhan (2211981203)

3.Krishnendu De (2211981205)

4.Jishu Adhikary (2211981185)

5.Kausik Sahoo(2211981197)

**Sustainable Development Goals**

** **

**Problem Statement:** Traditional smoke detection systems lack real-time remote monitoring capabilities, limiting their effectiveness in providing immediate alerts and visualizing gas levels for users, thereby compromising safety measures in both residential and commercial environments.

Example: Current smoke detection systems face limitations without remote monitoring and real-time data visualization, resulting in delayed responses and increased safety risks. An urgent need exists for an IoT-integrated solution providing remote monitoring and visual feedback to enable prompt and informed actions in addressing gas leaks in residential and commercial environments.

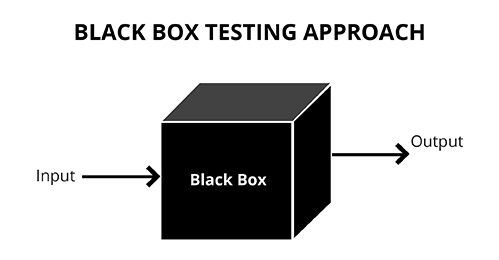
**Need Statement:** This project addresses the deficiencies in current smoke detection systems by introducing essential features like remote monitoring and real-time data visualization. By integrating IoT technology, it enables immediate awareness and prompt responses to smoke detection incidents, essential for enhancing safety in both residential and commercial settings.

Example:

This innovation enhances smoke detection systems by incorporating remote monitoring and real-time data visualization, allowing swift responses to smokes. The integration of IoT ensures timely awareness, crucial for bolstering safety in homes and workplaces.

**Objective-Function-Constraint Identification**

|  |  |  |  |
| --- | --- | --- | --- |
| **Characteristics** | **Objective** | **Constraints** | **Functions** |
| **Small size** |  | **✔** |  |
| **Low weight** |  | **✔** |  |
| **Safe** | **✔** |  | **Shock and fire resistant** |
| **Durable** |  | **✔** |  |
| **Low cost** | **✔** |  | **Mention concept here** |
| **Less power consumption** | **✔** |  | **Mention concept here** |
| **Portable** | **✔** |  | **Works on DC supply** |
| **Convenient to use** |  | **✔** |  |

** Black Box Diagram of Smoke Detector**

Output Devices: LED, Buzzer, Blynk -Notification

Input Devices: MQ2 Sensor

Output

**Name and Description of each component/device of Smoke Detector**

1. **NODE MCU ESP8266:**

**(i).Sensor Interface:** NodeMCU ESP8266 connects to smoke sensors, interpreting data received from them regarding the presence of smoke or fire through its GPIO pins or analog inputs.

**(ii).Data Processing and Alert Generation:** It processes sensor data, analyzing it based on predefined parameters or algorithms to detect smoke/fire signatures. Upon detection, it triggers an alarm and sends alerts via its Wi-Fi capabilities to connected devices.

**(iii).Remote Monitoring and Control:** Through Wi-Fi connectivity, it enables remote monitoring of the smoke detector system, allowing users to receive alerts and take necessary actions regardless of their physical location.

1. **MQ2 Sensor:**

**(i)Gas Sensing Capability:** The MQ2 sensor is designed to detect various gases like LPG, butane, propane, methane, alcohol, smoke, and other flammable gases, making it suitable for smoke detection in a project.

**(ii)Analog Output:** This sensor provides an analog output voltage proportional to the concentration of gases detected. It interfaces with microcontrollers, such as Arduino or NodeMCU, through its analog pins to transmit gas concentration data.

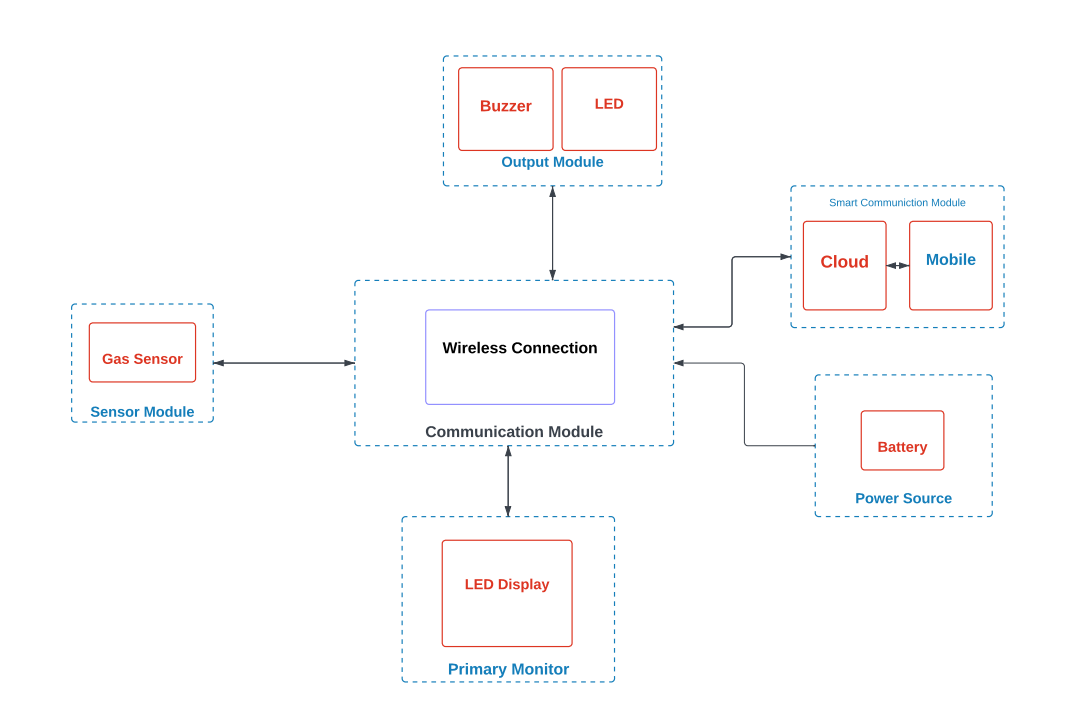
**(iii)Integration in Smoke Detection:** In a smoke detector project, the MQ2 sensor functions by detecting the presence of combustible gases or smoke particles in the air. It translates this detection into analog signals that can be processed by microcontrollers like NodeMCU ESP8266 to trigger alarms or alert systems in case of potential fire hazards.

1. **Bread Board:**

The breadboard acts as a platform for temporary circuit assembly, facilitating easy connection and testing of electronic components without requiring soldering, essential for prototyping and testing circuits in projects like a smoke detector before creating a permanent setup.

1. **LED:** Blink light after smoke detection.
2. **Buzzer:** Buzzing sound after smoke detecting**.**
3. **Conneting Wires:** Connect every component with each other.

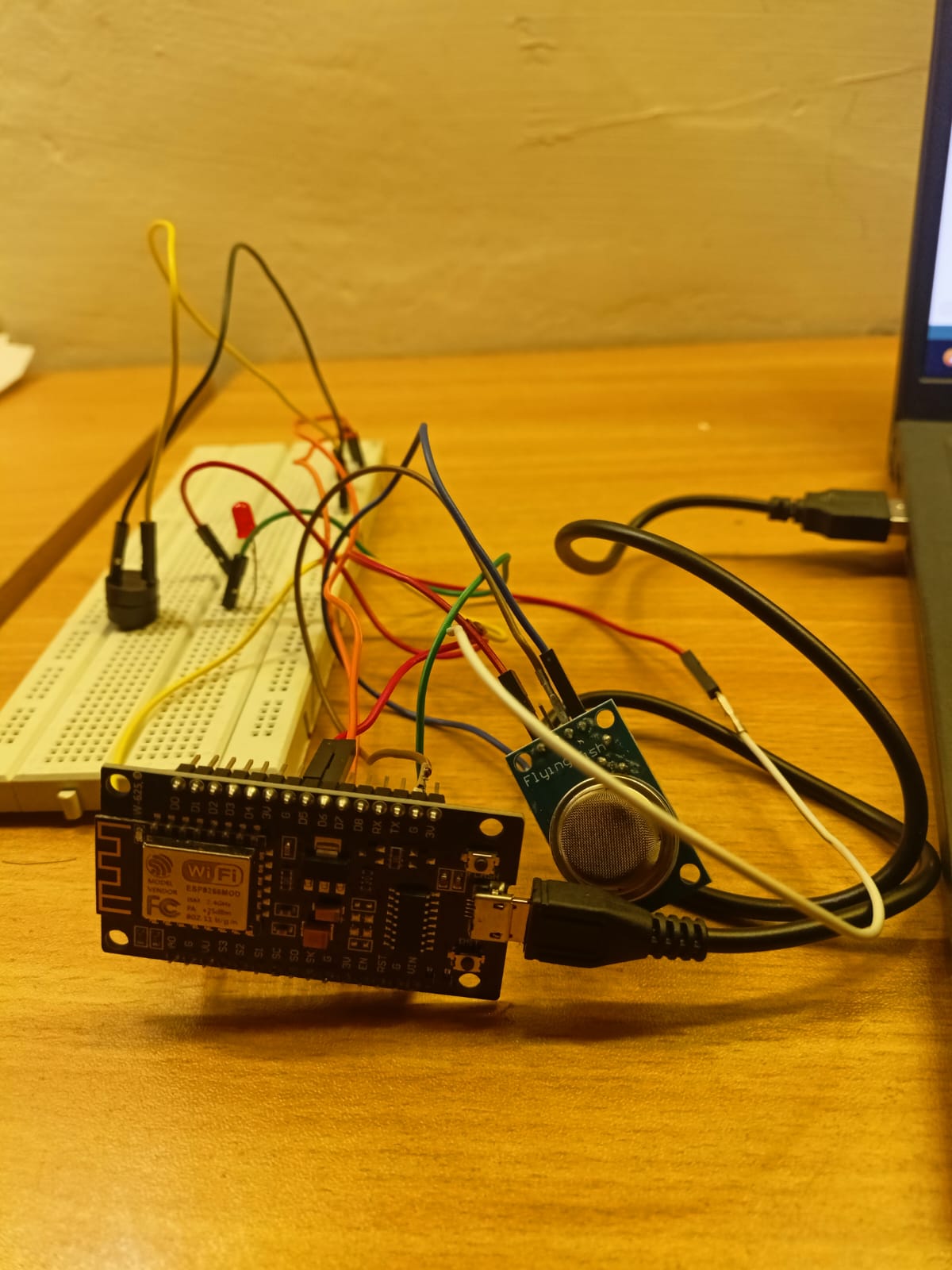
**Conceptual Design/Block diagram of Smoke Detector**

****

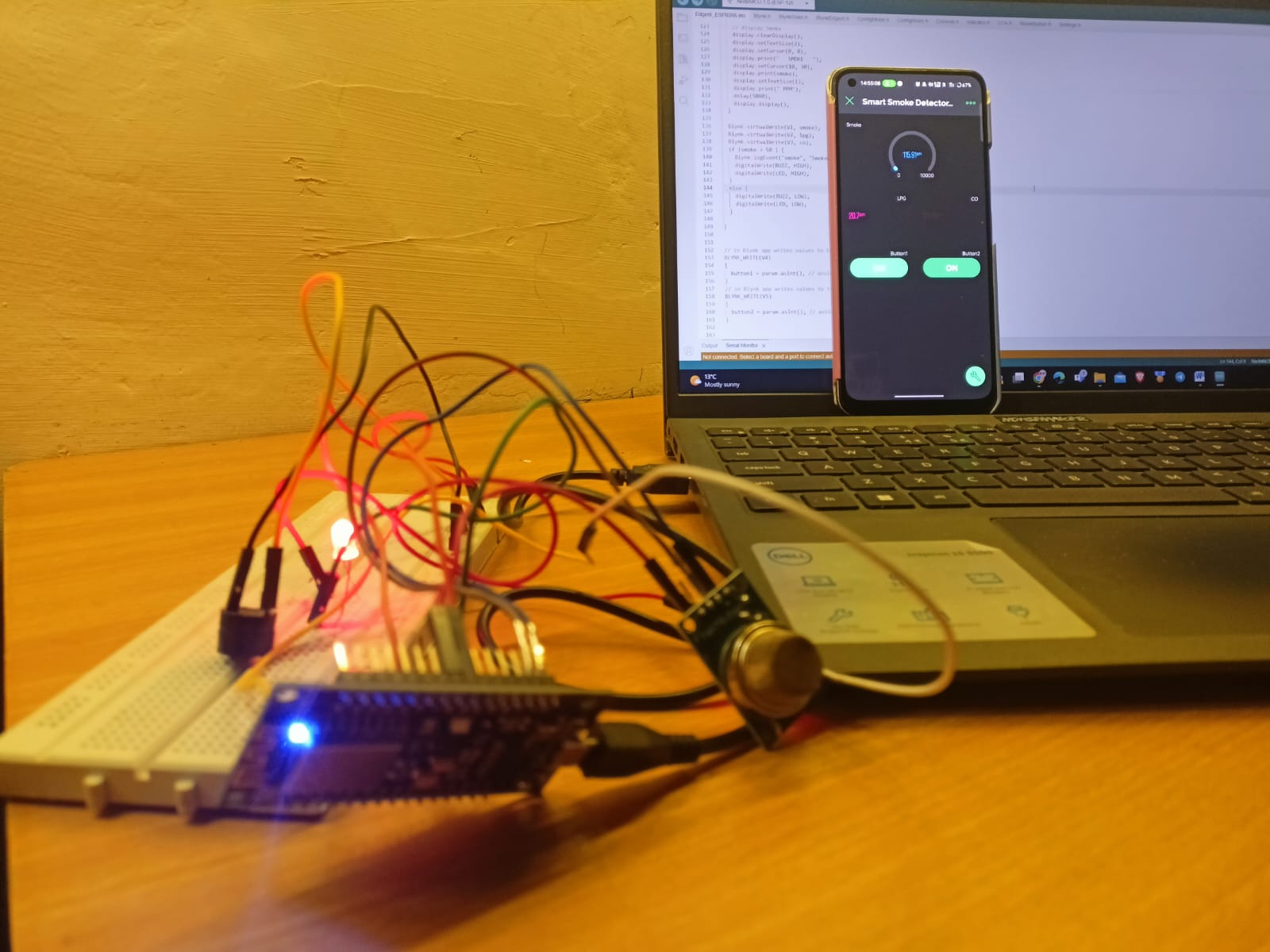
**Detailed working of Smoke Detector**

The NodeMCU ESP8266 microcontroller serves as the central control unit, orchestrating the functionalities of the MQ2 gas sensor, OLED display, buzzer, LED, and Blynk cloud integration. It manages sensor data collection, display output, and Blynk communication. The NodeMCU connects to the Blynk cloud using BlynkEdgent for seamless Wi-Fi connectivity without hardcoded credentials.It reads gas sensor data, displays it on the OLED as per user input from the Blynk app, triggers local alerts through the buzzer and LED if smoke levels surpass a threshold, and transmits sensordata (smoke, LPG, CO) to the Blynk cloud for remote real-time monitoring. This setup ensures comprehensive monitoring of gas levels both locally and remotely, providing users with alerts and access to environmental data via the Blynk app interface.

**PROTOTYPE PHOTOGRAPHS**

****

**Figure 1. Breadboard Connection diagram of proposed project**

****

**Figure 2. Breadboard Connection diagram with output Display**

**Code for execution of project:-**

#define BLYNK\_TEMPLATE\_ID "TMPL3MQiNj3Ve"

#define BLYNK\_TEMPLATE\_NAME "Smart Smoke Detector"

#define BLYNK\_FIRMWARE\_VERSION "0.1.0"

#define BLYNK\_PRINT Serial

#define BLYNK\_DEBUG

#define APP\_DEBUG

// Uncomment your board, or configure a custom board in Settings.h

#define USE\_SPARKFUN\_BLYNK\_BOARD

#define USE\_NODE\_MCU\_BOARD

#define USE\_WITTY\_CLOUD\_BOARD

#define USE\_WEMOS\_D1\_MINI

#include "BlynkEdgent.h"

#include <MQ2.h>

#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 32 // OLED display height, in pixels

#define OLED\_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)

#define SCREEN\_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for 128x64, 0x3C for 128x32

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

#define BUZZ 12 //D6

#define LED 14 //D5

//change this with the pin that you use

int pin = A0;

float lpg, co, smoke;

MQ2 mq2(pin);

int button1 = 0;

int button2 = 0;

SimpleTimer timer;

void setup()

{

Serial.begin(115200);

delay(100);

BlynkEdgent.begin();

pinMode(BUZZ, OUTPUT);

pinMode(LED, OUTPUT);

digitalWrite(BUZZ, LOW);

digitalWrite(LED, LOW);

// calibrate the device

mq2.begin();

// SSD1306\_SWITCHCAPVCC = generate display voltage from 3.3V internally

if (!display.begin(SSD1306\_SWITCHCAPVCC, SCREEN\_ADDRESS)) {

Serial.println(F("SSD1306 allocation failed"));

for (;;); // Don't proceed, loop forever

}

delay(2000);

display.clearDisplay();

display.setTextSize(2);

display.setTextColor(SSD1306\_WHITE); // Draw white text

display.setCursor(0, 0);

display.println(" IoT Smoke ");

display.setCursor(0, 20);

display.println(" Detector ");

display.display();

delay(1000);

timer.setInterval(1000L, sendSensorData);

}

void loop() {

timer.run(); // Initiates SimpleTimer

BlynkEdgent.run();

}

void sendSensorData()

{

float\* values = mq2.read(true); //set it false if you don't want to print the values to the Serial

co = mq2.readCO();

smoke = mq2.readSmoke();

lpg = mq2.readLPG();

if (button1 == 1)

{

// display LPG

display.clearDisplay();

display.setTextSize(2);

display.setCursor(0, 0);

display.print(" LPG ");

display.setCursor(10, 30);

display.print(lpg);

display.setTextSize(1);

display.print(" PPM");

delay(5000);

display.display();

}

else if (button2 == 1)

{

// display CO

display.clearDisplay();

display.setTextSize(2);

display.setCursor(0, 0);

display.print(" CO ");

display.setCursor(10, 30);

display.print(co);

display.setTextSize(1);

display.print(" PPM");

delay(5000);

display.display();

}

else {

// display Smoke

display.clearDisplay();

display.setTextSize(2);

display.setCursor(0, 0);

display.print(" SMOKE ");

display.setCursor(10, 30);

display.print(smoke);

display.setTextSize(1);

display.print(" PPM");

delay(5000);

display.display();

}

Blynk.virtualWrite(V1, smoke);

Blynk.virtualWrite(V2, lpg);

Blynk.virtualWrite(V3, co);

if (smoke > 50 ) {

Blynk.logEvent("smoke", "Smoke Detected!");

digitalWrite(BUZZ, HIGH);

digitalWrite(LED, HIGH);

}

else {

digitalWrite(BUZZ, LOW);

digitalWrite(LED, LOW);

}

}

// in Blynk app writes values to the Virtual Pin 4

BLYNK\_WRITE(V4)

{

button1 = param.asInt(); // assigning incoming value from pin V4 to a variable

}

// in Blynk app writes values to the Virtual Pin 5

BLYNK\_WRITE(V5)

{

button2 = param.asInt(); // assigning incoming value from pin V5 to a variable

}

**DIVISION OF WORK IN TEAM:-**

|  |  |  |
| --- | --- | --- |
| **Name of student** | **ID** | **Role** |
| **Koushik Adak** | **2211981201** | **Project Manager** |
| **Krishnendu De** | **2211981205** | **Project Designing** |
| **Krishna Gopal Pradhan** | **2211981203** | **Wireless connection Management** |
| **Kaushik Kumar Sahoo** | **2211981197** | **Documentation** |
| **Jishu Adhikary** | **2211981185** | **Project Testing** |

**Bill of material:-**

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.no. | Item name | No of item required | Total cost |
| 1. | NodeMCU ESP8266 | 1 | 360 |
| 2. | MQ-2 Sensor | 1 | 230 |
| 3. | Connecting wires | 15 | 70 |
| 4. | Bread board | 1 | 130 |
| 5. | LED | 1 | 20 |
| 6. | Buzzer | 1 | 30 |
|  |  | Grand total | 840 |

**Conclusion:-** The project “Fingerprint Based Voting Machine” was mainly intended to develop a fingerprint based advanced Electronic Voting Machine (EVM) which helps in free and fair way of conducting elections which are basis for democratic countries.

**Failure analysis:-**

**񯜈WIFI-CONNECTION:** During the testing of project there is some issue with the wifi-connection between the NODEMCU ESP8266 and Blynk Cloud.

**References Used:**

**You-Tube Reference:** https://www.youtube.com/watch?v=wq1bM2EpRdI&ab\_channel=TechTrendsShameer