REAL TIME DETECTION OF VOCS IN FRUITS AND VEGETABLES

Ayisha K M , N Charita Jain, Nikhitha K R , and Vikshitha

Department of Electronics and communication, P. A. College of Engineering, Karnataka, Mangaluru, India

E-mail:

Abstract

In recent years, Freshness and chemical safety of fruits and vegetables are critical concerns ensuring food quality and health. This project aims to design and implement real time detection of volatile organic compound (VOC's) present in fruits and vegetables using an MQ3 sensor integrated with an ESP32 microcontroller. The MQ3 sensor is calibrated to detect specific VOC's which may indicate the presence of pesticides, ripening agents, or spoilage compounds The ESP32 Programmed using Arduino IDE, serves as the central processing unit , collecting data from the sensor and analysing it to determine VOC level. The processed data is displayed on an LCD screen, offering immediate insights into the quality of the Produce.Additionally, Bluetooth communication is employed to transmit the detected data to Smartphone, providing a user friendly interface for real time monitoring . The system is designed to provide a portable, low cost and efficient solution for detecting harmful chemical residues and Ensuring the freshness of produce. This project aims to aid consumers, retailers, and quality control Units in making informed decisions about food safety and storage. By reducing the risk of consuming chemically contaminated produce, the proposed system contributes to promoting healthier dietary choices and fostering trust in food quality.

ISBN:97881-19905-39-3

1 Introduction

A balanced diet must include fresh fruits and vegetables because they are a great source of energy, fiber, and vital nutrients. Growing consumer awareness of the value of eating a healthy diet is driving a sharp increase in the demand for fresh vegetables worldwide. However, a number of factors, such as microbiological contamination, spoiling, pest infestation, and handling procedures, might jeopardize the safety and quality of fresh fruits and vegetables. Fruits and vegetables' volatile organic compounds (VOCs) can be used as markers of their safety and quality. The air around the product

contains volatile organic compounds (VOCs), which are chemical molecules that readily evaporate. The distinctive volatile organic compounds (VOCs) released by different kinds of fruits and vegetables might vary depending on ripeness, decay, and disease.

Using an ESP32 microcontroller and a MQ3 sensor, we suggest a real-time method for identifying volatile organic compounds (VOCs) in fruits and vegetables. Certain volatile organic compounds (VOCs) linked to ripening, spoiling, and chemical residues can be detected by the MQ3 sensor. In order to analyze the sensor data and wirelessly transmit it for monitoring purposes, the ESP32 functions as a processing unit. Conventional techniques for identifying volatile organic compounds (VOCs) in fruits and vegetables, such gas chromatography-mass spectrometry (GC-MS), are frequently costly, time-consuming, and necessitate specific knowledge. On the other hand, fresh food can be monitored quickly, accurately, and affordably with real-time VOC detection devices. The goal of this project is to create a real-time VOC identification system for fruits and vegetables by utilizing machine learning algorithms and cutting-edge sensor technology to produce precise and trustworthy findings.

The technology will be built to detect volatile organic compounds (VOCs) released by different kinds of fruits and vegetables as well as to spot patterns and trends that point to alterations in safety and quality.

ISBN:97881-19905-39-3

2 Objectives

This project aims to design and develop a real-time, portable, and cost-effective volatile organic compound (VOC) detection system for monitoring the freshness and quality of fruits and vegetables. The proposed system integrates an MQ3 sensor for VOC detection, an ESP32 microcontroller for data processing, and an Arduino board for sensor calibration and data acquisition. The system's primary objective is to identify specific VOCs that indicate spoilage, ripeness, or chemical contamination, enabling early detection and reducing food waste.

The system's design prioritizes accuracy, reliability, and user-friendliness. Sensor calibration

and extensive testing under various environmental conditions, such as temperature and humidity, will ensure the system's accuracy and reliability. The system will also incorporate thresholds for VOC levels, triggering alerts or notifications when spoilage or contamination is detected.

The proposed system has far-reaching implications for the agricultural industry, food suppliers, and consumers. By enabling real-time monitoring of produce quality, the system can help reduce food waste, promote sustainable consumption practices, and improve food safety. Furthermore, the system's scalability and adaptability enable future enhancements, including the detection of a broader range of VOCs, integration with IoT platforms for remote monitoring, and the use of machine learning algorithms for advanced predictive analysis and decision-making.

3 Requirement specification

Real-time system for detecting and analyzing VOC emissions from fruits and vegetables using sensors and data processing.

ISBN:97881-19905-39-3

3.1 Hardware requirement

3.1.1. ESP32 Microcontroller

- The ESP32 is a low-power, low-cost microcontroller with integrated Wi-Fi and Bluetooth capabilities.
- It has a dual-core processor, 4 MB of flash memory, and 520 KB of SRAM.
- The ESP32 is ideal for IoT projects, robotics, and automation expertise required

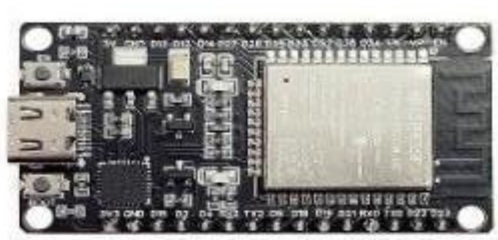


Figure 1:

.

Figure 3.1 ESP32 Microcontroller

3.1.1 MQ3 Sensor

- The MQ3 sensor is a gas sensor used to detect volatile organic compounds (VOCs) in fruits and vegetables.
- It uses a small heater inside to heat up the sensor, which helps to increase the sensitivity of the sensor.
- The MQ3 sensor is highly sensitive to VOCs and has a fast response time.

Display • The LCD display is used to show the VOC levels and other relevant information.

- It is a 16x2 LCD display, which means it can display 16 characters per line and has 2 lines

ISBN:97881-19905-39-3

Figure 2: 3.2 MQ3 Sensor

Figure 3: 3.3 LCD Display

ISBN:97881-19905-39-3

3.1.2 Power Supply

- The power supply is used to power the ESP32 microcontroller, MQ3 sensor, and LCD display.
  - It is a 5V power supply, which is the recommended voltage for the ESP32 microcontroller

3.1.3 Buzzer

- The Buzzer is an electronic device that produces a sound when an electric current is passed through it
  - It is commonly used in alarms, notifications, and feedback systems

Figure 4: 3.4 Buzzer

3.1.4 Breadboard and Jumper Wires

- The breadboard is used to connect the components and build the circuit.
  - The jumper wires are used to connect the components on the breadboard.

ISBN:97881-19905-39-3

Figure 5:

Figure 6: 3.5 Breadboard and Jumper Wires

3.2 Software Requirements

3.2.1 Blynk App

-The Blynk app is a mobile app used to display VOC levels and receive notifications when VOC levels exceed predetermined thresholds.

- It provides a user-friendly interface for users to monitor the VOC levels and receive alerts.

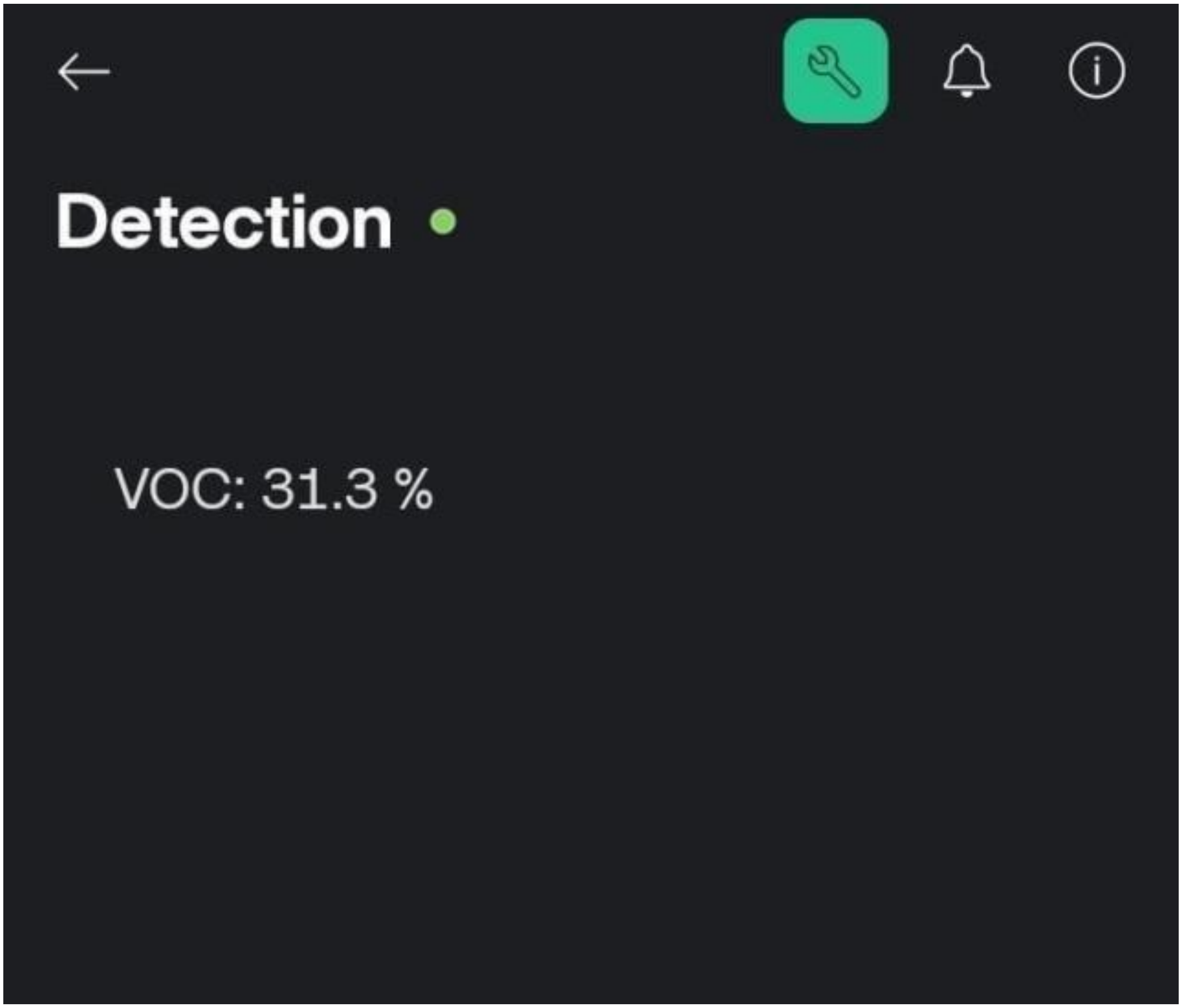


Figure 7:

3.2.2 Arduino IDE

• The Arduino IDE is an integrated development environment (IDE) used to program the ESP32 microcontroller.

• It provides a user-friendly interface for writing, compiling, and uploading code to the ESP32 microcontroller.

ISBN:97881-19905-39-3

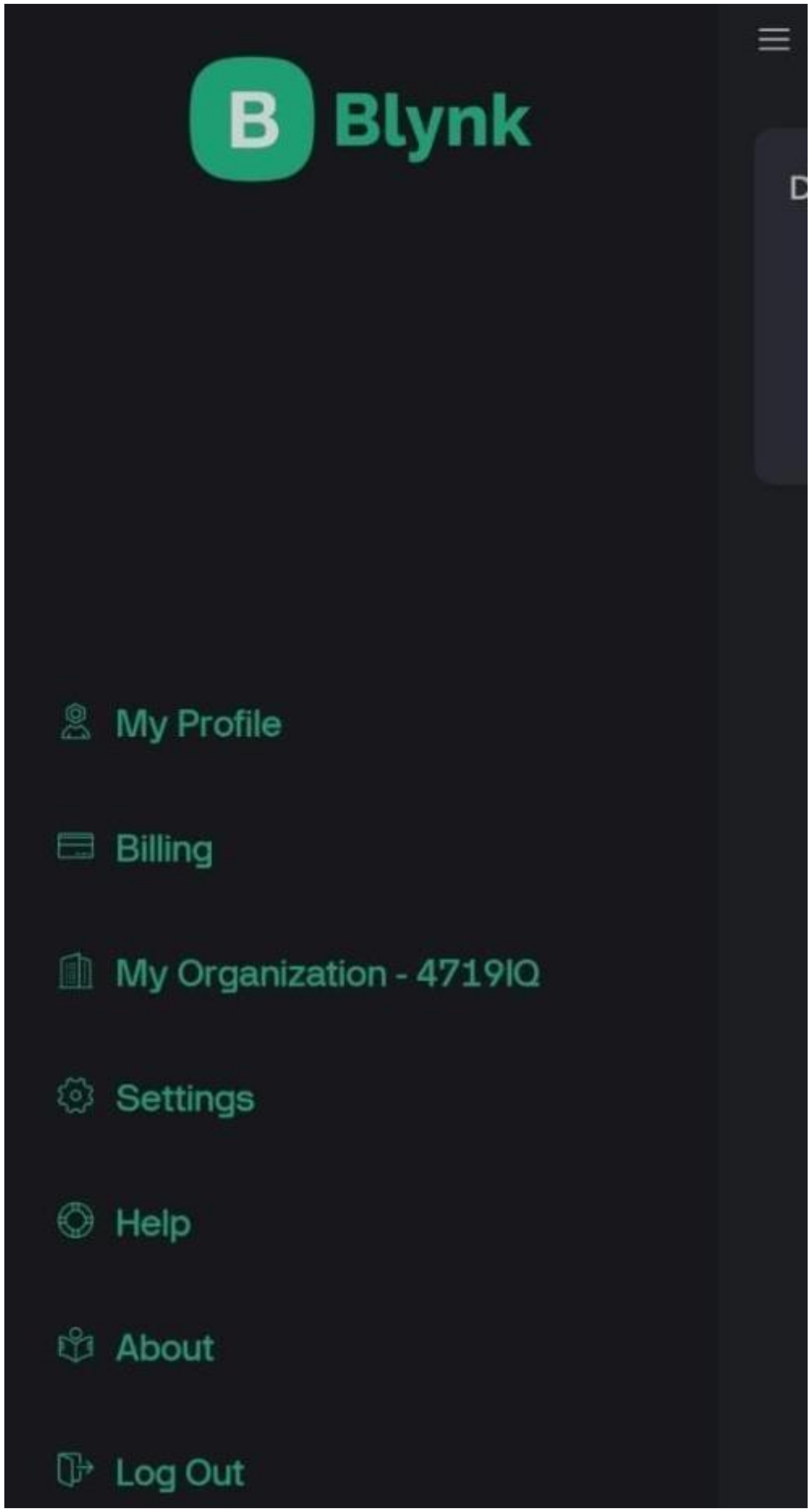


Figure 8: 3.7 Blynk home page

ISBN:97881-19905-39-3

3.2.3 ESP32 Firmware

• The ESP32 firmware is the software that runs on the ESP32 microcontroller.

• It provides the necessary drivers and libraries for the ESP32 microcontroller to function properly.

3.2.4 MQ3 Sensor Library

• The MQ3 sensor library is a software library used to interact with the MQ3 sensor.

• It provides functions for reading data from the MQ3 sensor and converting it into a usable format

3.2.5 Wi-Fi Library

• The Wi-Fi library is a software library used to establish a Wi-Fi connection and transmit data to the Blynk app.

• It provides functions for connecting to a Wi-Fi network, sending data to a server, and receiving data from a server.

Display Library • The LCD display library is a software library used to interact with the LCD display.

• It provides functions for displaying text and numbers on the LCD display.

3.3 Required Libraries

• WiFi.h: For establishing a Wi-Fi connection and transmitting data to the Blynk app.

• MQ3.h: For interacting with the MQ3 sensor.

• LiquidCrystal.h: For interacting with the LCD display.

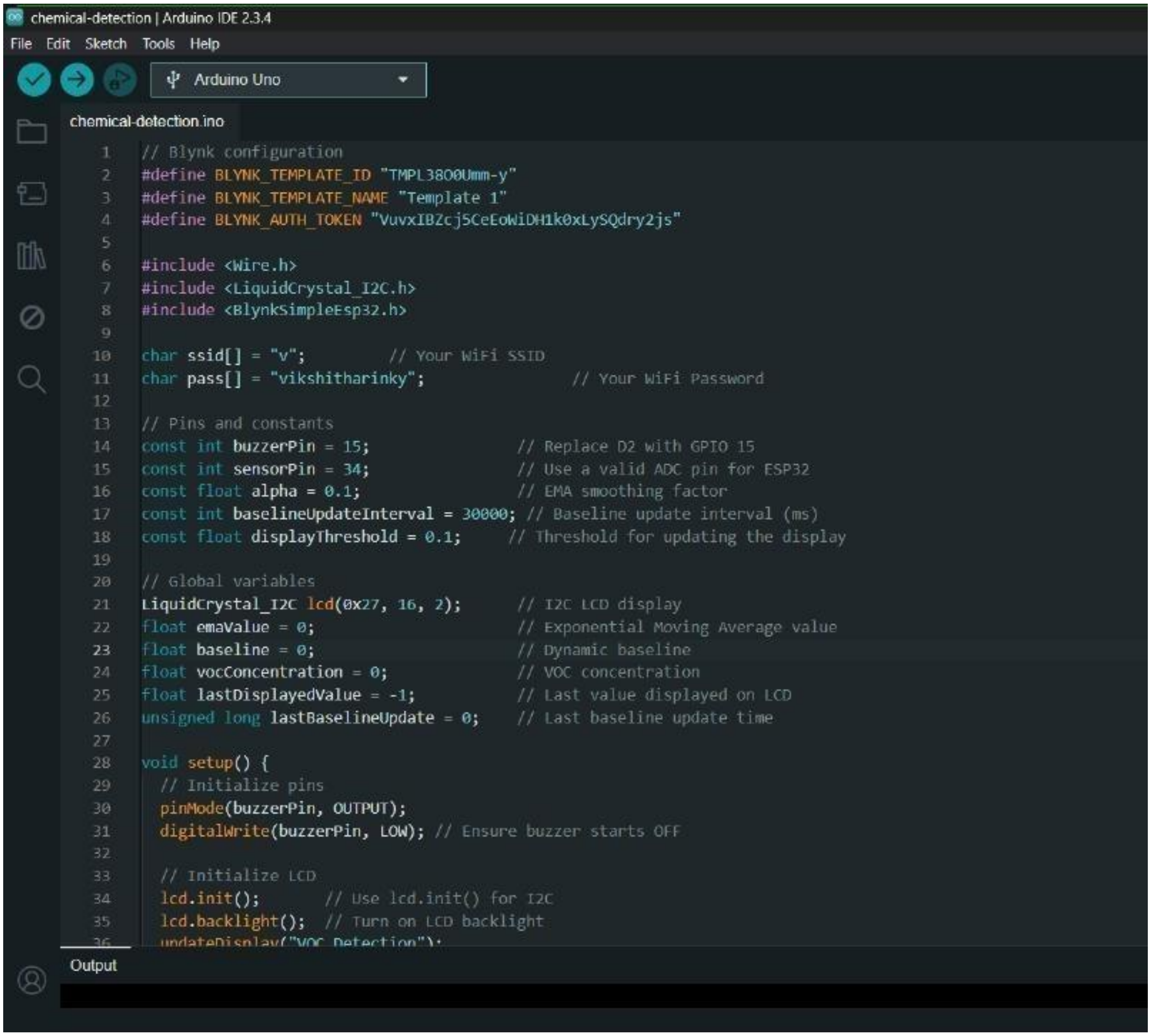
• BlynkSimpleEsp32.h: For interacting with the Blynk app.

• ESP32WiFi.h: For establishing a Wi-Fi connection and transmitting data to the Blynk app

ISBN:97881-19905-39-3

.

3.4 Program snippets



chemical-detection | Arduino IDE 2.3.4

File Edit Sketch Tools Help

Arduino Uno

chemical-detection.ino

```
// Blynk configuration
#define BLYNK_TEMPLATE_ID "TMPL38O0Umm-y"
#define BLYNK_TEMPLATE_NAME "Template 1"
#define BLYNK_AUTH_TOKEN "VuvxIBZcj5CeEoWiDH1k0xLySQdry2js"

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <BlynkSimpleEsp32.h>

char ssid[] = "v";          // Your WiFi SSID
char pass[] = "vikshitharinky";              // Your WiFi Password

// Pins and constants
const int buzzerPin = 15;                // Replace D2 with GPIO 15
const int sensorPin = 34;                // Use a valid ADC pin for ESP32
const float alpha = 0.1;                 // EMA smoothing factor
const int baselineUpdateInterval = 30000; // Baseline update interval (ms)
const float displayThreshold = 0.1;     // Threshold for updating the display

// Global variables
LiquidCrystal_I2C lcd(0x27, 16, 2);      // I2C LCD display
float emaValue = 0;                      // Exponential Moving Average value
float baseline = 0;                      // Dynamic baseline
float vocConcentration = 0;              // VOC concentration
float lastDisplayedValue = -1;           // Last value displayed on LCD
unsigned long lastBaselineUpdate = 0;    // Last baseline update time

void setup() {
  // Initialize pins
  pinMode(buzzerPin, OUTPUT);
  digitalWrite(buzzerPin, LOW); // Ensure buzzer starts OFF

  // Initialize LCD
  lcd.init();        // Use lcd.init() for I2C
  lcd.backlight();  // Turn on LCD backlight
  updateDisplay("VOC Detection");
```

Output

Figure 9:

4 Problem statement

Fruits and vegetables are highly perishable commodities, and their quality deteriorates over time due to natural ripening and spoilage processes. Volatile Organic Compounds

ISBN:97881-19905-39-3

(VOCs) emitted by these items provide critical information about their freshness and quality. However, traditional methods for detecting spoilage, such as visual inspection or manual handling, are subjective,

Time consuming, and often inefficient in identifying early signs of decay or contamination. This can result in significant food wastage, economic losses, and health risks due to the consumption of spoiled produce.

Furthermore, existing technological solutions for quality assessment, such as laboratory-based chemical testing, are expensive, require specialized expertise, and are not feasible for everyday or large-scale use in markets, homes, and storage facilities. The lack of affordable, portable, and real time detection systems creates a significant gap in ensuring the quality and safety of fruits and vegetables.

This project aims to address these challenges by developing a cost-effective, portable, and user- friendly device for detecting VOCs in real-time using an MQ3 sensor, ESP32, and Arduino. The system will provide immediate feedback on the freshness and quality of produce, enabling timely intervention to prevent food spoilage, reduce wastage, and ensure consumer safety. By integrating advanced sensor technology with real-time data processing and alert mechanisms, the proposed solution seeks to offer an innovative approach to addressing the pressing issue of food quality monitoring and sustainability.

4.1 Expected Outcomes

1. Accurate VOC Detection: Detect Volatile Organic Compounds (VOCs) in the fruits and vegetables accurately.

2. Real-time VOC Concentration Display: Display current VOC concentration in real-time on the LCD display.

3. Buzzer Alert: Trigger an alarm when VOC concentration exceeds a threshold.

4. Smartphone Notification: Send VOC concentration data to the smartphone via Wi-Fi and display it on the screen using the Blynk app.

ISBN:97881-19905-39-3

5. Improved Indoor Air Quality: Help users monitor and improve food quality by detecting high VOC concentrations and alerting them to take necessary actions.

5 System Design

This explains about block diagram, circuit connection, working and flow chart of the real time detection of vocs in fruits and vegetables.

5.1 Block Diagram

5.2 Circuit Connection:

Arduino Uno:

Acts as the primary controller to manage sensor input and output devices. Connected to a USB cable for power and programming.

ESP32:

Used for wireless communication (possibly for sending data to a smartphone or cloud). Connected to the breadboard via jumper wires, sharing ground (GND) with the Arduino.

MQ-3 Sensor:

Detects gases (like alcohol vapors). Connected to the Arduino:

VCC pin of the MQ-3 to 5V on the Arduino. GND pin of the MQ-3 to GND on the Arduino.

Analog/Digital output pin of the MQ-3 to an analog input pin on the Arduino (e.g., A0).

6 Display:

Displays the output concentration or status.

Connected to the Arduino via I2C or direct pin configuration:

ISBN:97881-19905-39-3

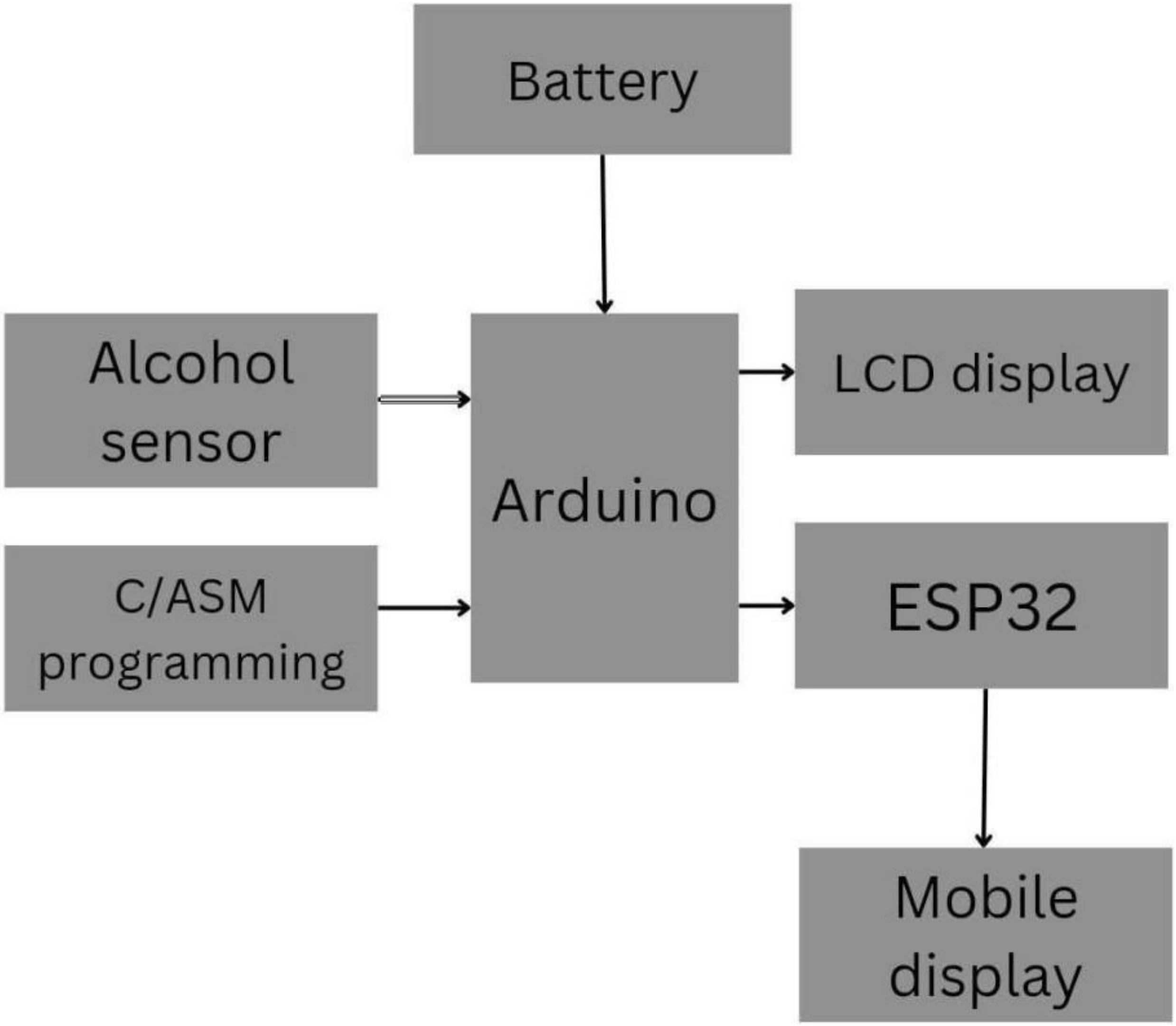


Figure 10: 5.1 block diagram

7 Pin to 5V on the Arduino GND pin to GND on the Arduino

SDA/SCL (if I2C) or RS/E/D pins (if parallel communication) to appropriate Arduino pins.

Buzzer:

Alerts the user when a certain gas concentration is detected. Connected to the Arduino: Positive terminal to a digital pin (D8) on the Arduino. Negative terminal to GND.

ISBN:97881-19905-39-3

7.1 Working

Sensing Hazardous Chemicals (Ethanol):

The MQ-3 sensor detects ethanol vapors in fruits and vegetables. The sensor's resistance changes based on the ethanol concentration, and it provides an analog output proportional to the ethanol level.

Processing with Arduino:

The analog signal from the MQ-3 sensor is fed into the Arduino, which serves as the main controller. The Arduino processes the signal, converts it into ethanol concentration (PPM), and compares it against the permissible threshold (200 PPM).

Output on LCD:

The Arduino displays the ethanol concentration in real time on a 16x2 LCD screen. If the ethanol level exceeds the permissible limit, the LCD shows a warning message.

Triggering the Buzzer:

When ethanol levels exceed 200 PPM, the Arduino triggers a buzzer to alert the user about the potential hazard.

Real-Time Monitoring on Blynk App:

The processed data is also sent to an ESP32 module, which connects to the Blynk app via Wi-Fi. The Blynk app provides real-time monitoring of ethanol levels on a smartphone, enabling remote access to the data.

Applications and Significance:

This project provides a cost-effective and efficient solution for detecting hazardous ethanol levels in fruits and vegetables. It ensures food safety by combining local alerts (LCD and buzzer) with remote monitoring (Blynk app).

7.2 Flow Chart

This flowchart outlines the process ethanol, in fruits and vegetables. Here's the step-by-step explanation:

ISBN:97881-19905-39-3

1. Start: The process begins.

2. Ethanol detection: Ethanol is detected from the fruits and vegetables using appropriate sensors.

3. Measurement: The ethanol level in the fruits and vegetables is measured.

4. Is the ethanol level above the limit?: A decision point checks if the ethanol concentration exceeds a predefined safe threshold.

If No: The ethanol value is displayed on both an LCD and a mobile device.

If Yes: The ethanol value is displayed, and a buzzer is activated to indicate that the chemical concentration is high.

8 End: The process stops after displaying the results or triggering the alert

This ensures both visual and audible alerts for detecting unsafe levels of ethanol in the samples.

9 Implementation

The implementation of a real-time VOC detection system involves installing sensors in a chamber or storage area to capture the volatile organic compounds emitted by fruits and vegetables. The VOC data is collected and sent to a processing unit, such as a microcontroller or computer, where software analyzes it against known patterns to determine the freshness, ripeness, or spoilage of the produce. The results are displayed on a screen or sent as alerts through a connected app, enabling quick and actionable insights. Regular testing with different produce and sensor calibration ensures the system remains accurate and reliable, providing an effective solution for monitoring produce quality in real-time.

ISBN:97881-19905-39-3

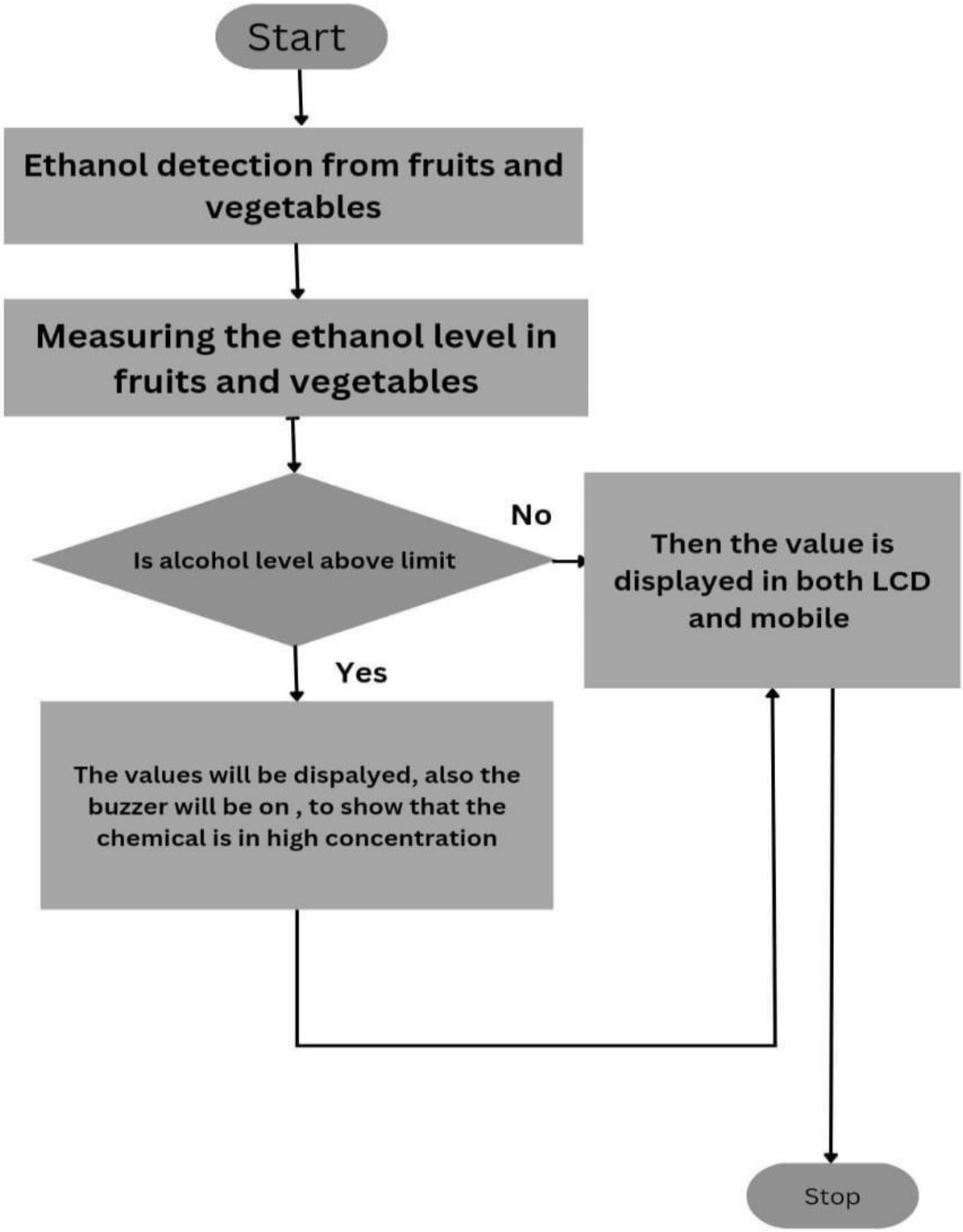


Figure 11: 5.2 flowchart

ISBN:97881-19905-39-3

9.1 Details of Language

The implementation of a real-time VOC detection system uses various programming languages for different tasks. For hardware control, languages like C or C++ are used to interact with sensors and collect data. Machine learning and data analysis are handled using tools like MATLAB or R, which are suited for advanced processing and statistical modeling. For user interfaces and IoT integration, languages like Java, Kotlin, or Swift are used to create mobile apps and enable seamless communication between devices. These languages work together to ensure efficient sensor operation, accurate data analysis, and user-friendly interaction.

9.1.1 C++

C++ is widely used in real-time VOC detection systems for its efficiency and control over hardware resources. It is commonly used to program microcontrollers or embedded systems that manage the interaction between sensors and the processing unit. C++ enables precise control of timing, memory management, and data acquisition, which is crucial for collecting VOC data in real time. Its performance allows the system to handle high-speed sensor inputs and process them with minimal delay, ensuring accurate and timely detection of volatile organic compounds. Additionally, C++ is well-suited for integrating with other system components, such as actuators, for real-time monitoring and decision-making.

10 6.1.2 C

C is often used in real-time VOC detection systems, especially for programming low-level hardware like microcontrollers and embedded systems. C's simplicity, efficiency, and ability **to** directly interact with hardware make it ideal for controlling sensors that detect volatile organic compounds (VOCs). It provides precise timing control, making it well-suited for real-time data acquisition, where sensor data needs to be captured and processed quickly.

ISBN:97881-19905-39-3

C's low-level memory management allows for optimized resource usage, ensuring the system can run efficiently even on devices with limited processing power. This makes C an essential language for implementing reliable and high-performance sensor control in such systems.

11 Results

The real-time VOC detection system accurately assesses the ripeness, freshness, and spoilage of fruits and vegetables using advanced sensing and analysis techniques. This project offers a non- invasive solution to improve quality control and reduce food waste.

-Project model

-LCD display shows '0.00%' concentration when ethanol is not detected in the fruit.

-Blynk app displays '0.00%' concentration when ethanol is not detected in the fruit.

-LCD display shows the percentage concentration of ethanol (e.g. 0.12%, 4%, etc.) when ethanol is detected in the fruit.

-Blynk app displays the percentage concentration of ethanol (e.g. 0.05%, 0.10%, etc.) when ethanol is detected in the fruit.

- Blynk app displays 'VOC Alert' when the detected ethanol concentration exceeds the set limit.

12 Conclusion & Future Work:

This section contains the outcome of the project and also what could be seen as the future of this project as whole.

12.1 Conclusion

The development of an Ethanol Detection System using MQ-3 sensor and Arduino (ESP32) has been successfully implemented. The system accurately detects ethanol concentrations

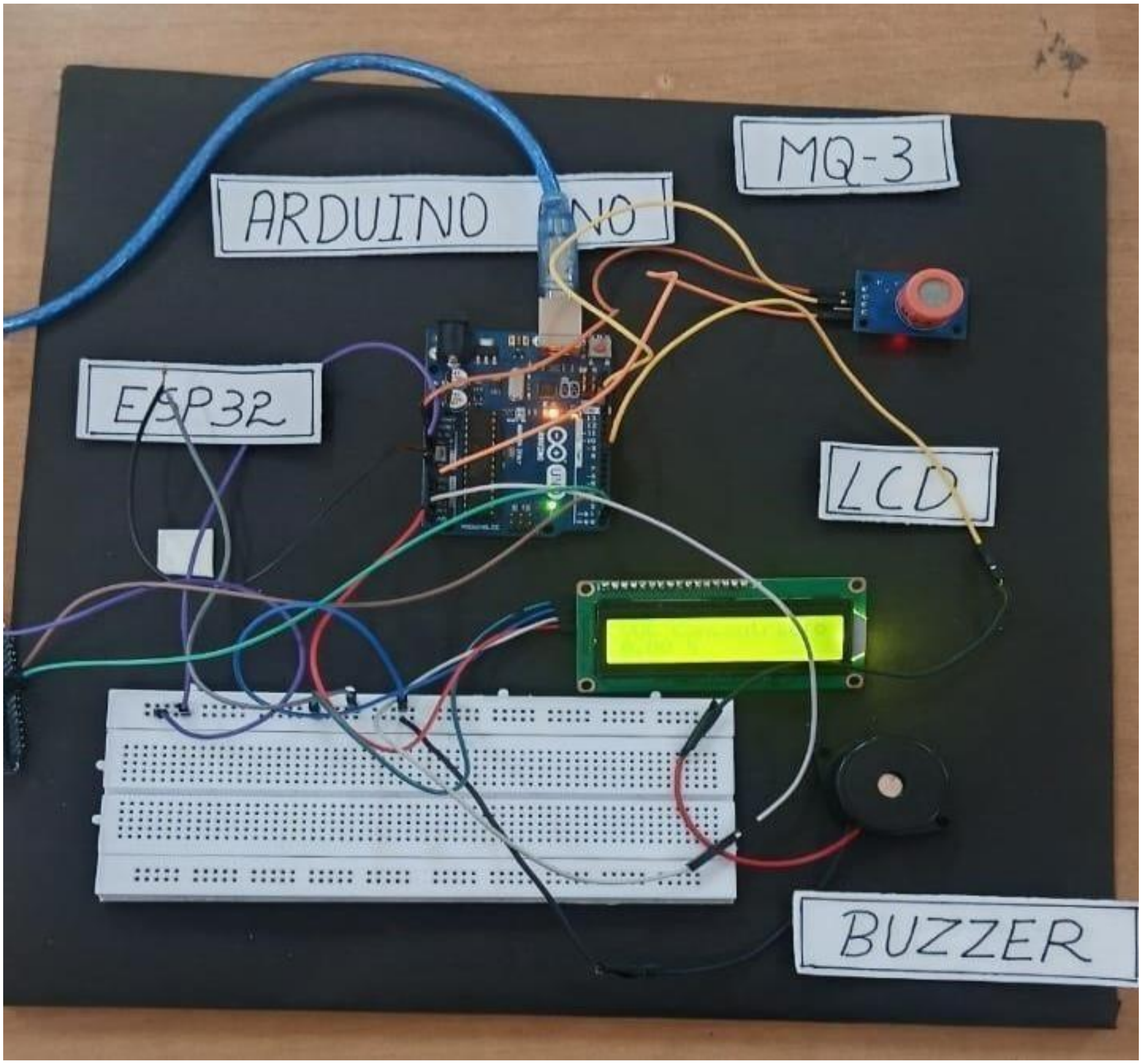


Figure 12: 7.1 Project model

in fruits and displays the percentage concentration on an LCD display and the Blynk app. Additionally, the system triggers a 'VOC Alert' on the Blynk app when the detected ethanol concentration exceeds the set limit. This project demonstrates a cost-effective and efficient solution for monitoring ethanol concentrations in fruits, which can be useful in various applications such as food quality control and safety monitoring.

ISBN:97881-19905-39-3

Figure 13: 7.2 LCD Display when ethanol is not detected

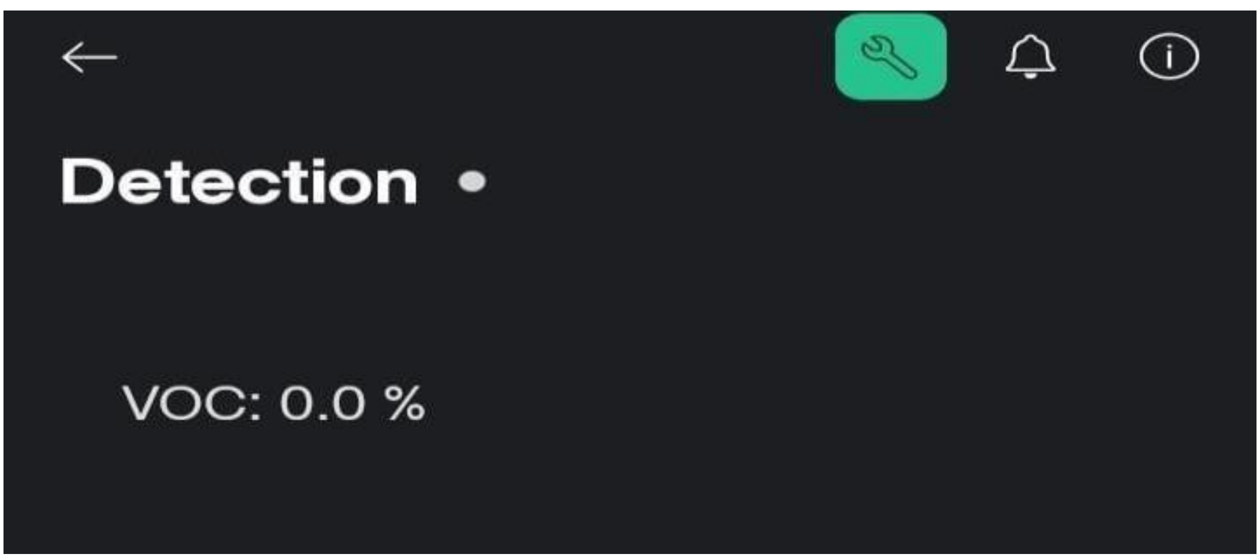


Figure 14: 7.3 Blynk display when ethanol is not detected

ISBN:97881-19905-39-3

Figure 15: 7.4 LCD Display when ethanol is detected

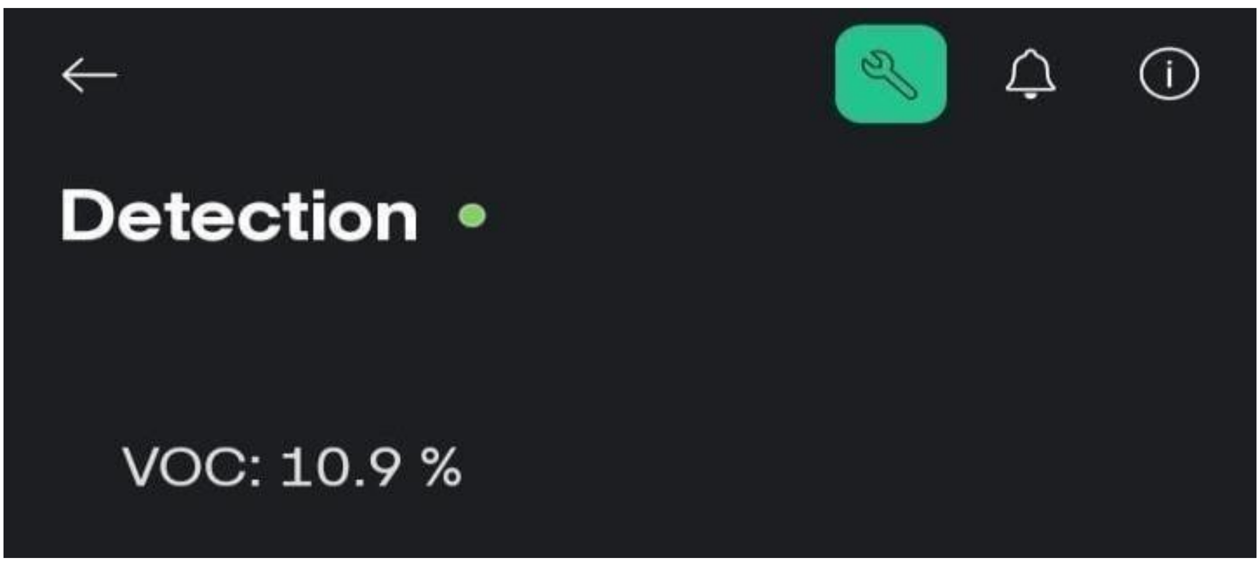


Figure 16: 7.5 Blynk display when ethanol is detected

12.2 Future work

• Integration with IoT Platforms: Integrate the Ethanol Detection System with popular IoT platforms like AWS IoT, Google Cloud IoT Core, or Microsoft Azure IoT Hub to enable real-time monitoring and data analytics.

• Machine Learning-Based Predictive Modeling: Develop machine learning-based predictive models to forecast ethanol concentrations in fruits based on environmental factors

ISBN:97881-19905-39-3

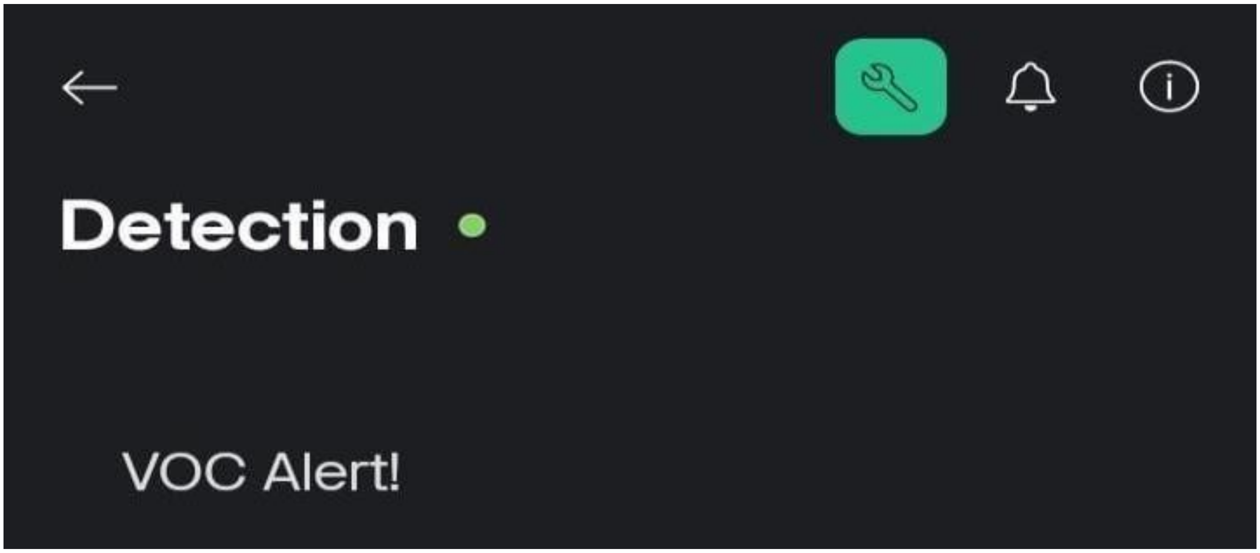


Figure 17: 7.6 Blynk display when the detected ethanol concentration exceeds the set limit

like temperature, humidity, and storage conditions.

• Multi-Gas Detection: Upgrade the system to detect multiple gases, including ethanol, to provide a more comprehensive analysis of fruit quality and safety.

• Wireless Sensor Networks: Develop a wireless sensor network (WSN) to enable multiple Ethanol Detection Systems to communicate with each other and with a central server, facilitating large-scale monitoring and data collection.

• User-Friendly Mobile App: Develop a user-friendly mobile app to provide users with a convenient and intuitive interface to monitor ethanol concentrations, receive alerts, and access historical data.

12.2.1

Acknowledgement: The corresponding author acknowledges the research facility provided by IEEE Xplore: