iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** III **Month of publication:** March 2024

DOI: https://doi.org/10.22214/ijraset.2024.59667

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

Rotten Fruit Detection Using IoT

Prof. Snehal Wasankar[1], Ayush Walke[2], Priyanka Thakare[3], Gauri Shirbhate[4], Yash Raypure[5], Srujan Sane[6]
[1]*Assistant Professor,* [2, 3, 4, 5, 6]*Final Year, Department of Computer Science & Engineering, Sipna College of Engineering and Technology, Amravati, Maharashtra, India.*

Abstract: ***In order to tackle the ongoing problem of keeping fruits fresh, this research presents an automated freshness detection device. The main goal is to create a device that can test the gaseous content of fruit to assess its quality. The Arduino cloud is used in a way of making and controlling Arduino projects online. You can connect your Arduino boards to the internet and send or receive data from them. You can also create web pages or apps to show and control your projects. Arduino cloud computing is easy to use and does not require much coding. Node MCU esp8266 which we use to store and transfer the data and is a small board that can connect to the internet and run programs. It has a chip called ESP8266 that can do Wi-Fi and other things. You can use it to make smart devices or internet of things projects. The device makes use of a MQ4 gas sensor as well as a DHT11 temperature and humidity sensor. Its primary goals are to monitor temperature and humidity in degrees Celsius and to detect methane concentrations in parts per million. Trace amounts of methane gas generated during the decomposition process are detected using the MQ4 sensor. The device's results are displayed on a 16x2 LCD display. Extensive fruit testing is conducted utilizing a variety of fruits, with a particular emphasis on two popular fruits: apples and bananas. The gadget is adaptable and useful for a wide range of food-related businesses, such as fruit and vegetable vendors, homes, the packaging industry, restaurants, whole chains of restaurants, and farmers looking to maximize their revenues. Interestingly, the device's design places a strong emphasis on economy and simplicity, making it useful and accessible to a broad spectrum of consumers. To sum up, our automated freshness detection gadget meets the demands of various stakeholders in the food sector by providing a creative and practical solution to the problem of fruit freshness.***

Keywords: Automated freshness detection device, Quality assessment, NodeMCU esp8266, MQ4 gas sensor, DHT11 temperature and humidity sensor, Fruit testing.

I. INTRODUCTION

The "IoT-Based Food Spoilage Detection System Using Arduino" represents a cutting-edge solution designed to address the critical challenge of timely identification and mitigation of fruit spoilage. In today's world, where food sustainability and waste reduction are paramount concerns, this innovative device leverages Internet of Things (IoT) technology and Arduino microcontrollers to provide a smart and efficient means of monitoring and detecting spoilage in fruits. The system integrates a sophisticated network of sensors capable of detecting various parameters, including but not limited to temperature, humidity, and potentially methane concentration, ensuring a comprehensive analysis of the fruit storage environment. The incorporation of Arduino, a versatile open-source electronics platform, empowers the device to collect, process, and transmit real-time data to a central server via the Internet, enabling remote monitoring and swift response to potential spoilage events. One noteworthy feature of this system is its application of methane detection technology. Methane, a natural gas produced during the decay of organic matter, serves as an early indicator of spoilage in fruits. By implementing methane sensors within the device, the system enhances its ability to detect fruit spoilage at its incipient stages, allowing for proactive measures to preserve the quality and safety of stored fruits. This will delve deeper into the intricacies of the device, exploring its hardware and software components, connectivity mechanisms, and the practical implications of its deployment. The IoT-Based Food Spoilage Detection System represents a pivotal step towards sustainable and efficient food management, contributing to the reduction of food waste and promoting a more resilient and resource-conscious approach to fruit storage and distribution. Exacerbated by faulty packaging techniques. Stringent regulations in developed countries address these hazards, leading resellers to take precautionary measures, such as disposing of food based on "best before" dates. However, it is noted that these dates may not accurately indicate spoilage, resulting in unnecessary discarding of viable and unsold food.

II. LITERATURE REVIEW

To assess the freshness of fruits, a robust method involves the integration of sensors without compromising the edibility of the fruit. Traditional approaches relied on Internet of Things (IoT) technologies to ascertain the freshness of fruits, ensuring a non-invasive evaluation that maintains the fruit's suitability for consumption.

A. Machine Learning and Big Data Analysis

Imagine a system that scans a fruit and instantly predicts its ripeness. That's the promise of machine learning algorithms trained on massive datasets of images, sensor readings, and freshness labels. Image Recognition: Deep learning algorithms analyze photographs of fruits, extracting features like color, texture, and shape. This "fruit property" helps identify freshness with remarkable accuracy.

1) *Sensor Fusion:* Beyond images, sensors can capture data like CO2 emission or electrical conductivity, further refining the freshness assessment. Combining diverse data sources with machine learning models strengthens the prediction power.
2) *Big Data Insights:* By analyzing vast databases of past fruit shipments and storage conditions, researchers can pinpoint factors impacting freshness loss. This knowledge aids in developing precise algorithms and forecasting shelf life.

B. Simulating Fruit Futures: Predicting Before Picking

Even before a fruit sees the sun, computer scientists are predicting its destiny. Computational models act as time machines, simulating fruit growth, ripening, and spoilage based on factors like weather, genetics, and storage conditions.

1) *Virtual Fruit Orchards:* Researchers experiment with different agricultural practices and storage methods in these digital fields. They tweak variables, observe virtual fruits, and optimize real-world processes for maximum freshness.
2) *Weather Whisperers:* By factoring in external forces like weather patterns and transportation delays, these models forecast potential spoilage risks before they bloom. Farmers and distributors can then take proactive measures to keep fruits happy and healthy.
3) *Disease Detectives:* Algorithmic analysis of fruit images can spot early signs of spoilage or disease, like a blemish on a

C. Internet of Things (IoT) and Smart Packaging

Fruit freshness, meet the connected world IoT sensors embedded in smart packaging monitor temperature, humidity, and even fruit-emitted gases in real-time.

1) *Smart Labels:* Imagine a label that dynamically changes color based on the fruit's ripeness, thanks to embedded sensors and displays. This empowers consumers to make informed choices.
2) *Supply Chain Monitoring:* Sensors throughout the supply chain, from farm to shelf, track environmental conditions and alert retailers or distributors of potential spoilage issues, minimizing waste and ensuring optimal freshness.
3) *Blockchain Integration:* By storing freshness data on a secure blockchain ledger, trust and transparency are established throughout the food chain.

III. METHODOLOGY

A. Hardware Requirements

The experimental setup requires NodeMCU, MQ4 sensor, DHT11 sensor, I2C sensor, LCD display

1) Node MCU

NodeMCU is an open-source firmware and hardware development kit based on the ESP8266 Wi-Fi module. It supports both Lua scripting and Arduino IDE, making it versatile for users with different programming expertise. Originally using Lua, it later incorporated Arduino IDE support, widening its user base. NodeMCU simplifies IoT development, functioning as a standalone device or part of interconnected networks. The development board includes essential components, and its active community provides ample resources and support. Widely used in IoT applications, NodeMCU is known for its cost-effectiveness, Wi-Fi connectivity, and compatibility with various projects.

Fig 1: NodeMCU

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

2) MQ4 Sensor

The MQ-4 can detect natural gas concentrations anywhere from 200 to 10000ppm. Just power the module with 5V set the threshold and you can start getting the gas concentration of the air around the sensor

Fig 2: MQ4 gas sensor

3) DHT11 Sensor

The DHT11 is a basic, ultra-low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and spits out a digital signal on the data pin (no analog input pins needed).It's simple to use but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds, so when using the Arduino library, sensor readings can be up to 2 seconds old.The DHT11 is a widely used and inexpensive digital temperature and humidity sensor. It is commonly employed in various projects and applications that require monitoring and measuring environmental conditions.

Fig 3: DHT11 sensor

4) I2C Serial Interface Module

I2C is a way of connecting devices that need to talk to each other with only two wires. One wire is for data and the other is for a clock that synchronizes the data. The devices have addresses so they know who is sending or receiving data. I2C is good for short-distance communication and low-speed devices.

Fig 4: I2C serial interface module

5) LCD Display

An LCD display with 2 rows and 16 characters typically refers to a liquid crystal display (LCD) module that can display alphanumeric characters in a 2x16 grid. These displays are commonly used in various electronic devices, such as microcontroller-based projects, Arduino projects, and other embedded systems.

Fig 5: 2x16 LCD display

B. Software Requirements

1) IoT Cloud

An IoT cloud is a massive network that supports IoT devices and applications. This includes the underlying infrastructure, servers and storage, needed for real-time operations and processing. An IoT cloud also includes the services and standards necessary for connecting, managing, and securing different IoT devices and applications[8].

2) Arduino Integrated Development Environment (IDE)

Overview: The Arduino IDE serves as a consumer-pleasant interface for writing and uploading applications to Arduino. Its move-platform nature ensures compatibility with Windows, Mac OS, and Linux, catering to a diverse user base.

Programming Languages: The Arduino IDE is primarily based on C and C++, offering customers with a acquainted programming surroundings. This segment highlights the simplicity of code organization within the IDE, assisting the languages C and C++.

Software Library from the Wiring Project: The Arduino IDE comes equipped with a software program library sourced from the Wiring challenge, e.g. {#include "thingProperties.h" }

This library consists of lots of commonplace enter and output features, streamlining the improvement method for users by supplying with ease to be had functionalities.

Upload Code to NodeMCU: Connect NodeMCU to your computer using a USB cable. Select the correct COM port in the Arduino IDE. Click the "Upload" button to upload the code to NodeMCU.

```
#include "thingProperties.h"
const int mq2Pin = A0;
void setup() {
  Serial.begin(9600);

  // Initialize IoT Cloud
  initProperties();

  // Connect to IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}
void loop() {
  // Update IoT Cloud
  ArduinoCloud.update();

  // Read MQ2 sensor value
  int mq2Value = analogRead(mq2Pin);

  // Print the sensor value to the serial monitor
  Serial.print("MQ2 Sensor Value: ");
  Serial.println(mq2Value);

  // Update the IoT Cloud property
  MQ2Sensor = mq2Value;

  // Add a delay if needed
delay(1000);
}
```

Fig 6: Arduino code to read data from MQ4 gas sensor

This Arduino code connects to the IoT Cloud, reads values from an MQ4 gas sensor connected to analog pin A0, prints the sensor value to the serial monitor, updates the corresponding IoT Cloud property (MQ4 Sensor), and repeats the process in a loop with a 1-second delay. The code leverages the Arduino IoT Cloud library for cloud connectivity and is designed to provide real-time monitoring of MQ4 sensor data through the cloud platform.

The Arduino IDE simplifies prototyping with its beginner-friendly interface, cross-platform compatibility, and C/C++ guidance. Its Wiring library expands capabilities, making it crucial for building interactive digital projects.

3) Using Arduino Cloud to Creating a Thing

The user journey always begins by creating a new Thing. In the Thing overview, we can choose what device to use, what Wi-Fi network we want to connect to, and create variables that we can monitor and control.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

Next we need to add a device by clicking on the "Select device" button on the Thing overview. Here, we choose from any board that we have already been configured, or select the Configure new device option.

Now we can add our first variable by clicking on the Add variable button. We can choose name, data type, update the setting and interaction mode for our variable. There are several data types we can choose from, such as int, float, boolean, long, char. There are also special variables, such as Temperature, Velocity, and Luminance that can be used. The variables we create are automatically generated into a sketch file.

Finally, we need to connect to a Wi-Fi network by simply clicking the Configure button in the network section. Enter your network credentials and click Save. This information will also be generated into your sketch file

4) Building the Sketch

A special sketch file can now be found in the Sketch tab, which includes all of the configurations that you have made. When the sketch has been uploaded, it will work as a regular sketch.

Additionally, each time we create a variable that has the Interaction Mode enabled, a function will also be generated. Every time this variable is triggered from the Cloud, it will execute the code within this function. This means that we can leave most of the code out of the loop() and only run code when needed.

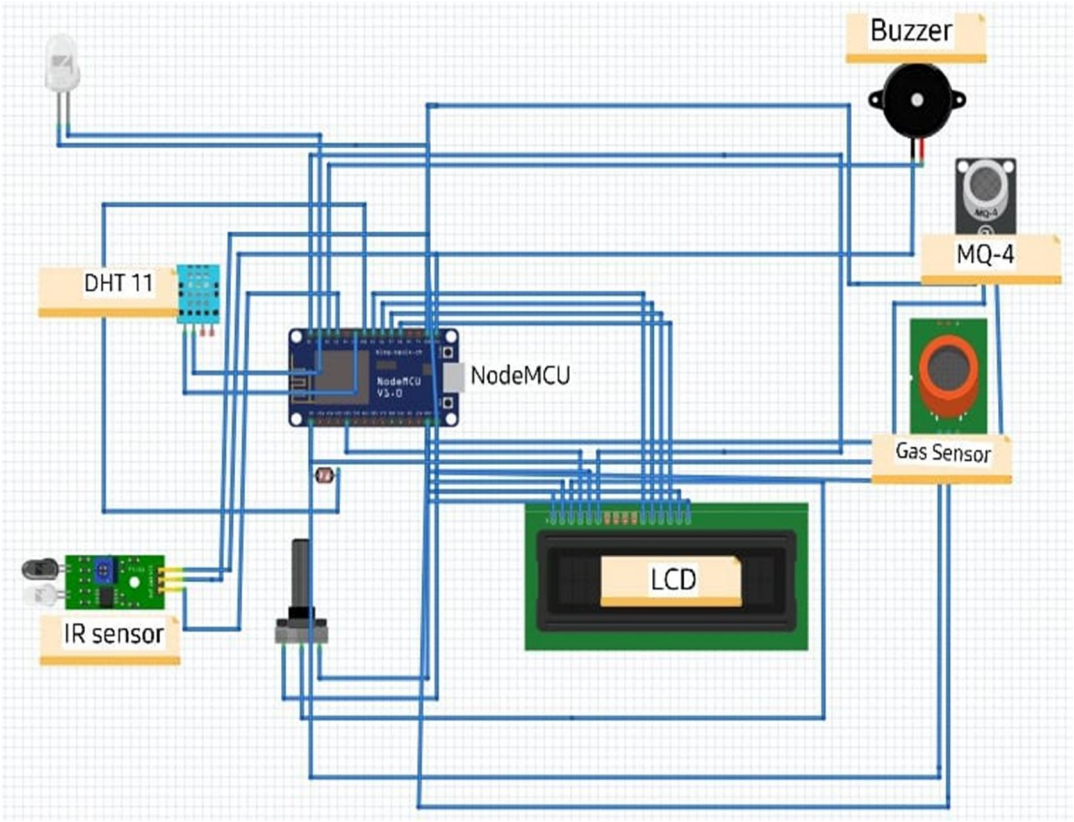
When we are done with our sketch, we can upload it to our board, by clicking the upload button.

After we have successfully uploaded the code, we can open the Serial Monitor tab to view information regarding our connection. If it is successful, it will print connected to network_name and connected to cloud. If it fails to connect, it will print the errors as well.

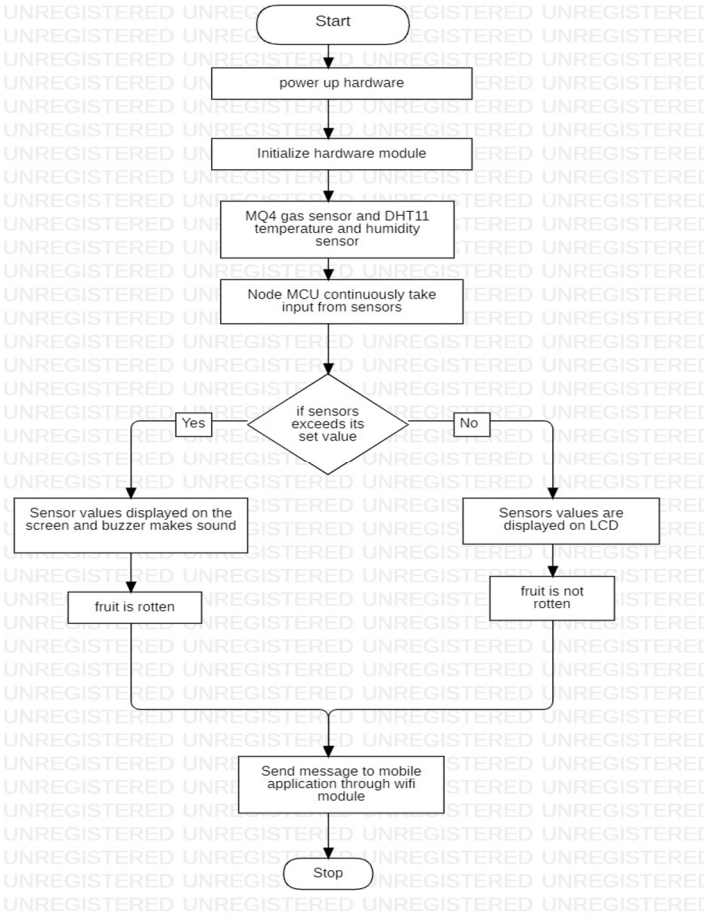
Now that we have configured the device & network, created variables, completed the sketch and successfully uploaded the code, we can move on to the dashboard.

By creating new dashboard, click on the dashboard editor and we can create the widgets for variables. That widgets displays the values received from the sensor.

IV. CIRCUIT DIAGRAM



V. FLOWCHART



VI. OVERVIEW OF WORK

When the device is made ON, initially it calculates resistance of MQ sensors to get the known concentration of gases around the sensor without the presence of other gases. MQ4 sensor calculates Methane level to know if fruit is fresh to eat or shouldn't be ate. MQ4 sensor is used to detect natural gases content in atmosphere of fruit. The atmosphere of fruit is crucial for it to be in good condition. These conditions are temperature and humidity which are calculated by DHT11 sensor. This sensor gives exact temperature and humidity so we can keep it in control with help of cold storage. We used buzzer and LCD screen to let user know what condition of fruit and its atmosphere is. If threshold value of MQ4 sensor is achieved, it will show on screen "DON'T EAT" and buzzer will buzz. Now if the MQ4 sensor is achieved i.e., it will show "GAS RELEASING" on screen. And at last, if temperature level is above a certain level which is standard temperature for fruits storage it will show "HIGH TEMPERATURE" and if humidity is high it will show "HIGH HUMIDITY".

Fig 7: LCD display displaying sensors values

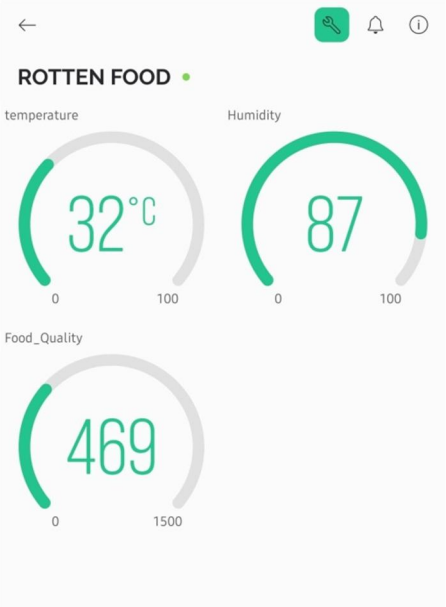


Fig 8: Sensors values on application

VII. CONCLUSION

In conclusion, the "IoT-Based Food Spoilage Detection System Using Arduino" represents a pioneering solution in the realm of fruit storage management. By harnessing IoT technology and Arduino microcontrollers, it offers a sophisticated means of monitoring fruit storage environments, detecting spoilage at its earliest stages, and facilitating swift intervention. With its ability to analyze crucial parameters and leverage methane detection technology, this system empowers users to proactively preserve fruit quality and safety while minimizing waste. There is potential for its deployment to meaningfully advance sustainable food management practices and lower the environmental costs of food waste by significantly promoting more eco-friendly alternatives. By providing accurate spoilage detection and mitigating unnecessary discards based on arbitrary expiry dates, it not only enhances efficiency in fruit distribution but also contributes to the larger global efforts toward food sustainability and resource conservation. The IoT-Based Food Spoilage Detection System stands as a beacon of innovation in the quest for more resilient, efficient, and environmentally conscious food management solutions.

REFERENCES

[1] Zohaib Hassan, Abhijeet, and Apoorva Sharma. "Internet of Life (IOL)." (2015). ISBN 978-93-5156-328-0 NXP Application note AN11075: Driving I2C-bus signals over twisted pair cables with PCA9605 (PDF), 2017-08-16, archived from the original (PDF) on 2017-08-16

[2] Yan, B.; Fan, P.; Lei, X.; Liu, Z.; Yang, F. A Real-Time Apple Targets Detection Method for Picking Robot Based on Improved YOLOv5. Remote Sens. 2021, 13, 1619.

[3] Brena RF, Aguileta AA, Trejo LA, Molino-Minero-Re E, Mayora O. Choosing the Best Sensor Fusion Method: A Machine-Learning Approach. Sensors (Basel). 2020 Apr 20;20(8):2350. doi: 10.3390/s20082350. PMID: 32326125; PMCID: PMC7219245.

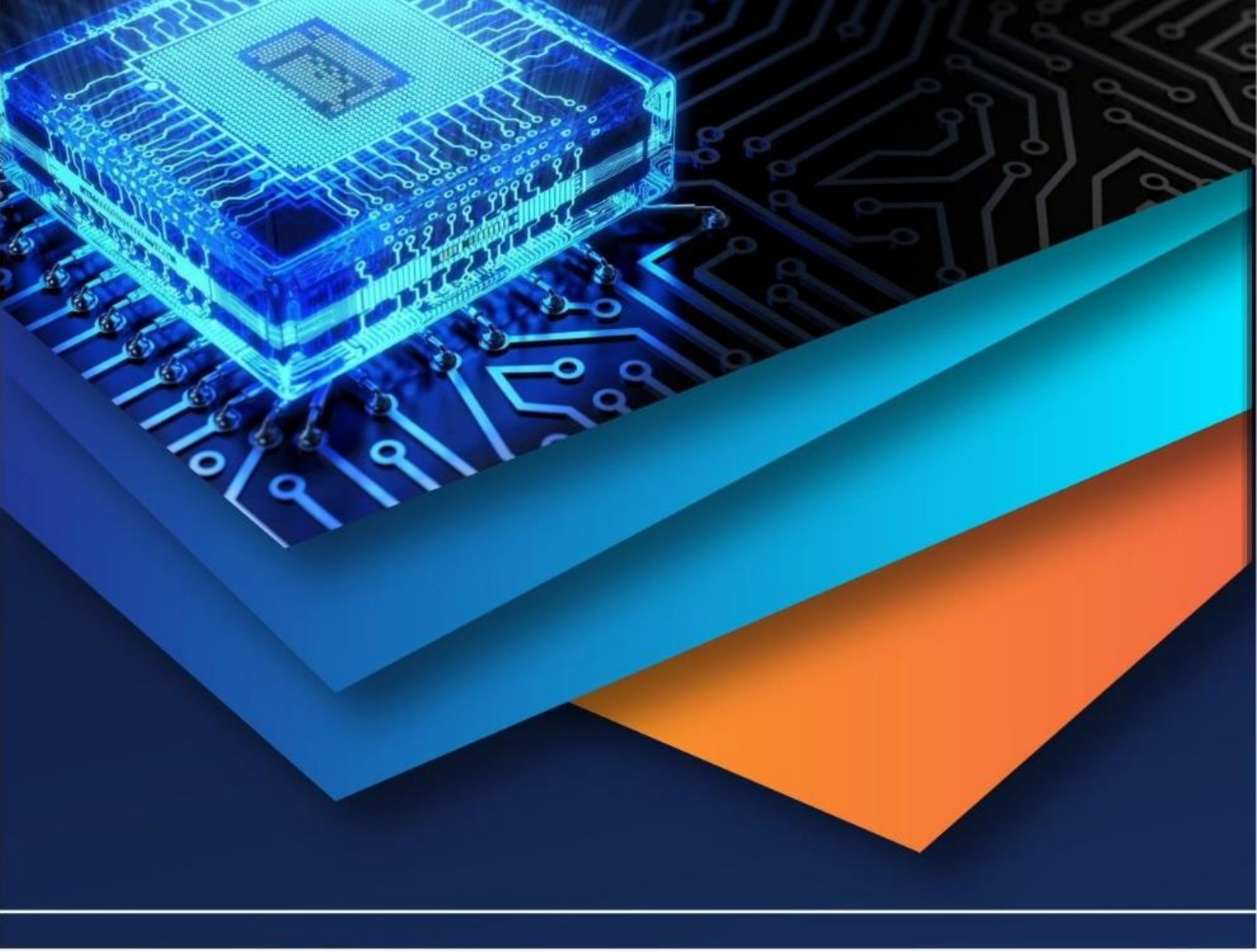
[4] Xiao, F.; Wang, H.; Xu, Y.; Zhang, R. Fruit Detection and Recognition Based on Deep Learning for Automatic Harvesting: An Overview and Review. Agronomy 2023, 13, 1625. https://doi.org/10.3390/agronomy13061625

[5] Alam AU, Rathi P, Beshai H, Sarabha GK, Deen MJ. Fruit Quality Monitoring with Smart Packaging. Sensors (Basel). 2021 Feb 22;21(4):1509. doi: 10.3390/s21041509. PMID: 33671571; PMCID: PMC7926787.

[6] Iskandar, Ade & Yuliasih, Indah & Warsiki, Endang. (2020). Performance Improvement of Fruit Ripeness Smart Label Based On Ammonium Molibdat Color Indicators. Indonesian Food Science & Technology Journal. 3. 48-57. 10.22437/ifstj.v3i2.10178.

[7] Kanagachidambaresan, G.R., N., B. (2023). Supply Chain Monitoring. In: Sensors and Protocols for Industry 4.0. Maker Innovations Series. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-9007-1_7

[8] https://www.arm.com/glossary/iot-cloud#:~:text=An%20IoT%20cloud%20is%20a,real%2Dtime%20operations%20and%20processing.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY