IJIRIS: International Journal of Innovative Research in Information Security
Volume 09, Issue 03, June 2023
E-ISSN: 2349-7017
P-ISSN: 2349-7009
https://www.ijiris.com/archives

Food Freshness Detection Using IOT and ML

R Sriram Kumar
Assistant professor, Department of CSE
Sri Sairam College of Engineering
Anekal, Bengaluru, India
sriramkumarr.cse@sairamce.edu.in
Shreya s kalal, Pujashree V, Asmath Unnisa
UG Scholar, Department of CSE,
Sri Sairam College of Engineering
Anekal, Bengaluru, India
sce20cs060@sairamtap.edu.in, sce20cs062@sairamtap.edu.in
sce20cs112@sairamtap.edu.in

Publication History
Manuscript Reference No: IJIRIS/RS/Vol.09/Issue03/JNIS10097
Research Article | Open Access | Double-blind Peer-reviewed |Article ID: IJIRIS/RS/Vol.09/Issue03/JNIS10097
Received: 02, June 2023 | Accepted: 20, June 2023 | Published: 23, June 2023
Volume 2023 | Article ID JNIS10097 http://www.ijiris.com/volumes/Vol09/iss-03/18.JNIS10097.pdf
ArticleCitation: Sriram,Shreya,Pujashree,Asmath(2023).Food Freshness Detection using IoT and ML. International Journal of Innovative Research in Information Security (IJIRIS), 10, 141-146
doi: https://doi.org/10.26562/ijiris.2023.v0903.18
BibTex key: Sriram2023Food

Copyright: ©2023 This is an open access article distributed under the terms of the Creative Commons Attribution License; which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Abstract: In today's modern world, the issue of food spoilage poses a significant challenge, as consuming spoiled meals can be hazardous to consumers' health. Our objective is to develop a solution for detecting spoiled food by employing suitable sensors that monitor gasses released by various food items. By utilizing a microcontroller equipped with Internet of Things (IoT) capabilities, an alert can be issued, prompting appropriate action to be taken. This technology has the potential for widespread application in the food industry, where manual food detection is currently employed. Our plan is to create devices that can learn and adapt to this detection process, enabling us to estimate the likelihood of food spoilage and predict the timeframe within which it may occur, especially when sourced from specific suppliers. By implementing this system, we aim to foster increased competition among retailers, encouraging them to prioritize the sale of healthier and fresher food products. Ultimately, this will contribute to the creation of a safer environment for all consumers. To achieve our goals, we propose the integration of Internet of Things (IoT) and Machine Learning (ML) technologies in food freshness detection. This system is designed to assist food suppliers and retailers in monitoring the freshness of their food products. The proposed system employs a combination of IoT devices and ML algorithms to identify changes in the freshness of food items. IoT-enabled sensors are utilized to gather data from the surrounding environment, which is then processed by ML algorithms to detect any deviations in the freshness of the food products. In instances where the freshness begins to deteriorate, the system promptly notifies the food suppliers and retailers, enabling them to take immediate action. We have conducted thorough testing of the proposed system through a case study, and the results demonstrate its effectiveness in detecting changes in the freshness of food products. The integration of a microcontroller equipped with IoT capabilities ensures timely alerts, facilitating the necessary actions to prevent the sale or consumption of spoiled food. This technology has the potential to revolutionize the food industry by automating and improving the detection of food spoilage, thereby safeguarding the health and well-being of consumers. In summary, our proposed solution utilizes IoT and ML to detect changes in food freshness. By leveraging IoT-enabled sensors and ML algorithms, we can monitor and predict the likelihood of food spoilage, enabling suppliers and retailers to take appropriate measures. This approach not only fosters competition among retailers to provide healthier and fresher food but also ensures a safer and more secure food supply chain for consumers worldwide.

Keywords: Iot, Machine Learning, ML Algorithm

I. INTRODUCTION:

In today's 21st-century landscape, the food industry plays a vital role in our economy. However, a pressing challenge it faces is the issue of food spoilage, particularly in meat and poultry products. The existing manual process of quality checking in the fruit and vegetable industries relies on individuals stationed along conveyor belts to inspect items as they pass by. Introducing an automated system has the potential to not only enhance the accuracy of detecting spoiled food but also reduce the need for extensive manual labor. To automate this process, our proposed solution involves the utilization of intelligent sensors and a microcontroller, such as the Node MCU. When a spoiled or stale food item is detected, an audible buzzer is activated to attract attention. Additionally, this information is transmitted to the cloud as an Internet of Things (IoT) application. This enables relevant authorities to track the frequency of receiving spoiled food, fostering transparency and accountability within the industry. The device incorporates various sensors to measure parameters such as temperature, humidity, and other environmental conditions surrounding the food items.

This data is then utilized to determine the freshness of the food, empowering consumers to make informed purchasing decisions. Furthermore, the system can identify and report any anomalies related to food freshness, thereby assisting food companies in better monitoring and managing their inventory. This, in turn, leads to improved customer service and satisfaction. In summary, by automating the process of quality checking in the fruit and vegetable industries, we aim to enhance the accuracy of detecting spoiled food while reducing the reliance on manual labor. By utilizing smart sensors and a microcontroller like the Node MCU, the system alerts authorities and sends data to the cloud for transparency purposes. The device's ability to measure environmental conditions and detect anomalies contributes to informed consumer choices and better management of food stocks by companies. Overall, this implementation promotes efficiency, accountability, and customer satisfaction in the food industry.

2. PROBLEM STATEMENT:

Food spoilage poses a major challenge in the food industry, specifically concerning the deterioration of meat and vegetables. However, an even more pressing concern arises when these spoiled products go unnoticed and end up in the hands of consumers. Currently, in the fruit and vegetable industries, quality control is primarily performed through manual inspection, typically involving an individual stationed alongside a conveyor belt, carefully scrutinizing each item as it moves past.

3. PROPOSED SYSTEM :

Food spoilage is a major concern, as it poses risks to consumer health. Our project aims to detect spoiled food using specialized sensors and monitor the gasses emitted by food items. A microcontroller triggers an IoT alert when spoilage is detected, enabling appropriate action. This has significant utility in food industries that currently rely on manual detection methods. By implementing machine learning, we can estimate the likelihood and duration of food spoilage, encouraging competition among retailers to provide healthier and safer meals. Gas sensors MQ2 and MQ135 measure carbon monoxide, benzene, and ammonia levels in specific food items. Our machine learning model predicts spoilage based on oxygen and ammonia content. When the Node MCU comes into contact with spoiled food, it emits a buzzer and uploads data to a cloud platform. The number of spoiled meals can be tracked, and the model predicts average shelf life. To enhance accuracy, decisions are based on cumulative readings from oxygen and ammonia sensors, reducing false negatives. By incorporating gas sensors, our smart system detects CO, ammonia, and benzene in any food object, allowing for optimized staffing during peak times to ensure safety.

4. SOFTWARE REQUIREMENTS:

A software requirements specification (SRS) is a thorough statement of the functional and nonfunctional requirements for a software system that needs to be produced. Based on the agreement between the customer and the contractors, the SRS is developed. It offers use cases for the various ways in which users will engage with the product. The software requirement specification document contains all the specifications needed to carry out the project.

5. HARDWARE REQUIREMENTS

1. Processor: i5
2. RAM: 8 GB
3. Hard Disk: 400GB
4. Processor Speed: 2.4GHZ
5. System Type: 64-bit
6. Arduino UNO Microcontroller

VI. SYSTEM DESIGN

System design is the process of defining the architecture, modules, interfaces, and data of a system to meet predetermined requirements. It applies systems theory to product creation, particularly in the context of computer systems that incorporate hardware, software, and communication components. System design service companies collaborate with clients to select suitable hardware and software solutions for a specific project, followed by system development, installation, and implementation. High-level design refers to a more comprehensive architecture document that outlines the relationships between modules and systems within the overall system structure. It provides a broader overview of the system's components and their interactions. On the other hand, low-level design focuses on more detailed aspects of the architecture, providing specific instructions on how modules and systems should be implemented. It delves into the intricacies of the system design, including algorithms, data structures, and other technical considerations. Fundamental design concepts continuously evolve and inspire software designers to create and test new ideas and foundations. These concepts encompass various aspects, such as efficiency, accuracy, modifiability, coding techniques, and data encryption, all aimed at meeting the requirements of the design. Input design involves converting user-generated inputs into a computerized format. The main objective is to simplify and minimize errors in the process of computerization. It focuses on organizing and capturing data effectively, ensuring user-friendliness, reliable organization, and intelligent modification to provide accurate and timely records. Input design requires careful consideration, as it involves collecting all necessary input elements, which can be costly. Output design focuses on meeting customer requirements and presenting output data in a clear and meaningful way. It encompasses the communication of processing results to users and other systems. Effective output design enhances the system's interaction with source and destination machines. In the proposed device, the optimized output design takes the form of a pie chart, displaying the percentage of men and women following the information article, thereby providing a concise and visually appealing representation of the data.

IJIRIS © 2014-23, AM Publications -All Rights Reserved

VII. DEVELOPMENT OF SYSTEM

Improvement of gadgets is a method in which product development is completed or all system problems are resolved. Software development includes a number of stages and techniques for expanding software programmes. A walk system is used to complete software development. The method used on this project is incremental versioning.

7.1 Different phases of model

Requirement: Gather all prerequisites for software expansion, including user needs, system specifications, and functional requirements.

Design: Convert the system specifications into software design, considering the system requirements. This phase includes defining the system's algorithm, structure, and overall design.

Coding: Develop the software by writing the necessary code, following the design specifications. The system requirements are translated into a readable and executable code structure.

Implementation: Execute the project, which involves deploying the coded software. This phase includes activities such as building libraries, creating documentation, and providing client manuals.

Testing: Coordinate and test all project modules to ensure the software meets the specified requirements. This phase focuses on verifying and validating the system to ensure it functions correctly.

Maintenance: The most crucial stage, where the software should be user-friendly, adaptable, error-free, and enhance project productivity. It involves regular updates, bug fixes, and improvements to meet evolving needs

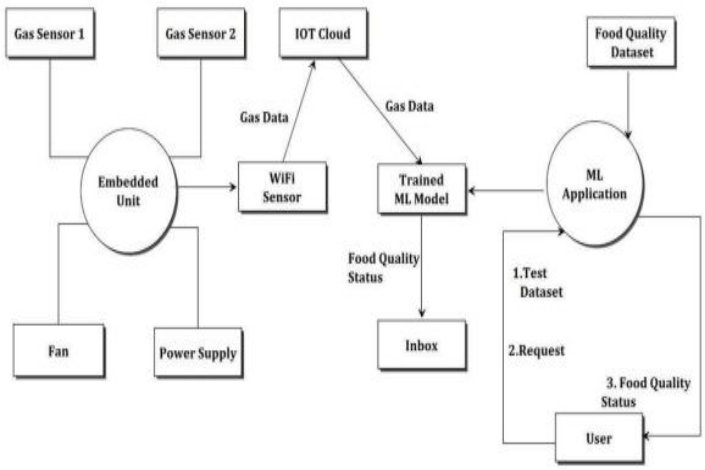
8.Purpose of choosing Increment model

The incremental model is a software development system in which necessities are broken down into multiple standalone modules of the software development cycle. Steps in incremental development include evaluation design, implementation, testing/verification, and maintenance.

- Machine development is broken down into numerous mini-development tasks.
- Partial structures are built one after the other to provide the final general machine.
- The highest precedence requirement is addressed first.
- Once developed, the requirements for that increment are frozen. When to use Incremental models?
- The device's requirements are completely understood.
- while a demand for a product's early release arises
- while the software engineering group isn't completely knowledgeable or skilled
- when it comes to high-risk functions and dreams
- This method is more difficult for internet applications and product-based complete groups.

9. SYSTEM ARCHITECTURE

The process of the architectural configuration focuses on developing a fundamental framework structure. Recognizing the components of the framework and the interactions between them are included. Construction modeling outline is the initial configuration process of identifying these subsystems and constructing a structure for subsystem control and correspondence. The output of this outline technique is a representation of the product structural planning. Below is a description of the system's suggested architecture. It demonstrates the system's design and briefly demonstrates how it operates.



9.1 Arduino Module

The Arduino Uno is a microcontroller board that utilizes the ATmega328 microcontroller, as stated in its datasheet. It is equipped with several components, including a 16 MHz ceramic resonator, 6 analogue inputs, 14 digital input/output pins (6 of which can function as PWM outputs), a USB port, a power jack, an ICSP header, and a reset button. The board is designed to provide complete support for the microcontroller, simplifying the setup process. Users can begin by connecting the Arduino Uno to a power source using a USB cable, an AC-to-DC adapter, or a battery.

9.2 Sensors Module

The MQ2 gas sensor module is used for detecting gas leaks in homes and industries. It can detect gasses such as H2, LPG, CH4, CO, Alcohol, Smoke, and Propane. However, for precise detection of specific components, specialized instruments may be necessary. The MQ135 sensor module is implemented to detect smoke, benzene, vapors, and other hazardous gasses. It can detect a range of harmful gasses and is suitable for air quality monitoring, detecting noxious gasses, home air pollution detection, industrial pollution detection, and portable air pollution detection.

9.3 LCD Module

With a 16x2 LCD, there are 2 lines that can each display 16 characters. Each character on this LCD is presented using a 5x7 pixel matrix. The 224 different characters and symbols that can be displayed on the 16 x 2 intelligent alphanumeric dot matrix display. There are two registers on this LCD: Command and Data..

9.4 Wifi Sensor Module

A self-contained SOC with an integrated TCP/IP protocol stack, the ESP8266 Wi-Fi Module allows any microcontroller to connect to your Wi-Fi network. The ESP8266 is capable of offloading all WiFi networking tasks from another application processor or hosting an application.

10.System Testing

Essential phase in the software development process. It involves examining a piece of work for any errors or weaknesses. The purpose of testing is to ensure that the software functions properly, meets user expectations, and fulfills the specified requirements. Various test types are employed to address specific testing needs and assess the performance of individual components, subassemblies, assemblies, or the final product. By conducting system testing, software developers can verify that the software operates without unacceptable failures and meets the desired criteria set by users.

Unit Testing: This type of testing focuses on verifying the correctness of individual software components. It involves designing test cases to validate the core program logic, ensuring that inputs produce expected outputs. Unit testing is conducted on each separate unit of the application before integration, examining the internal code flow and decision branches. It provides an invasive structural test that assesses how the components were built and verifies that each path of the business process adheres to specifications.

Integration Testing: Integration testing is performed to ensure that the merged software components function properly as a unified program. It tests the combined components to identify any issues that may arise from their integration. The testing is event-driven and emphasizes the overall outcome of screens or fields. Integration testing validates the accuracy and consistency of the component combination, even if the individual components passed unit testing.

Functional Testing: Functional testing aims to systematically validate that the tested functions comply with technical and business requirements, system documentation, and user manuals. It focuses on areas such as accepting valid input, rejecting identified classes of invalid input, exercising specified classes of application outputs, and calling interacting systems or processes. Functional tests are designed based on requirements, important functionality, and unique test cases. The coverage of data fields, predefined processes, business process flows, and subsequent processes is important in this type of testing. Additional tests may be added during the process, and the effectiveness of existing tests is assessed.

System Testing: System testing evaluates the integrated software system as a whole to ensure compliance with specifications. It tests the system setup to provide predictable outcomes. A configuration-oriented system integration test is an example of system testing. Process flows and descriptions form the basis of system testing, with a focus on pre-driven process connections and integration points.

White Box Testing: White box testing is a type of testing where the tester has knowledge of the internal workings, structure, and language of the software being tested. The goal of white box testing is to examine and validate specific areas of the software that are not accessible at a black box level. Testers with knowledge of the software's implementation details can design test cases that exercise different paths and conditions within the code. White box testing is typically conducted by developers or testers who have access to the source code.

Black Box Testing: Black box testing is a type of testing where the tester has no knowledge of the internal workings, architecture, or language of the module being tested. Testers perform black box testing without considering how the software functions internally. The tests are designed based on external specifications or requirements, treating the software as a "mystery" where inputs are provided, and outputs are observed without knowledge of the internal mechanisms. Black box testing is focused on validating the software's functionality and behavior from a user's perspective

Test Strategy and Approach:

The testing approach for this project includes manual field testing and detailed functional testing.

Test Objectives:

1. Ensure proper functioning of all field entries.
2. Validate activation of pages from identified links.
3. Verify timely display of entry screens, messages, and responses.

Features to be Tested:

1. Entry Format Verification: Verify correct format of entries.
2. Duplicate Entry Prevention: Ensure no duplicate entries are allowed.
3. Link Navigation: Validate that all links direct users to the correct pages.

By combining manual field testing and detailed functional testing, we aim to achieve comprehensive coverage of the system's functionality, ensuring that it operates smoothly and meets the specified requirements.

11.Integration Testing:

Integration testing is the incremental testing of multiple integrated software components on a single platform to identify interface flaws and induce failures. The main objective is to ensure seamless collaboration between software applications or components within a system or across an entire organization.

Test Results: All mentioned test cases passed successfully, and no defects were encountered during integration testing.

Test Environment: The software development process includes testing to ensure compliance with intended standards. Test cases are developed as part of the testing process to assess the product. In our study, we aim to utilize plant leaves for disease prediction. We categorize the leaves based on ailments, along with their symptoms and treatments.

Unit Testing of Modules: Designing test cases for unit testing focuses on validating the internal program logic and ensuring that inputs produce valid outputs. It involves examining the code flow and decision branches within each individual software component. Unit testing is performed after the completion of each component but before integration. These tests are based on the internal structure and require understanding of how the software was built. Unit tests verify that each route of a business process follows the specified specifications and produces expected results for defined inputs.

Module	Input	Expected output	Actual Output	Results
Data preprocessing	Food	Status: Fresh/Spoiled	Spoiled	Pass

11.1 Table Test Case Design

12.RESULT AND ANALYSIS

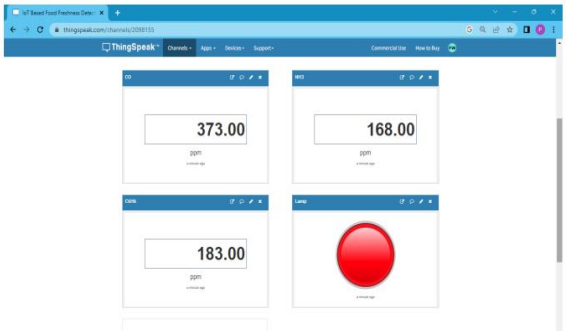


Fig 7.1 Screenshot of Thinks Speech of IOT Platform

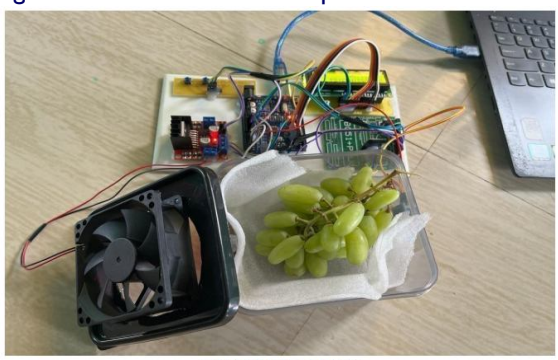


Fig 7.2: Checking of Fruits Freshness

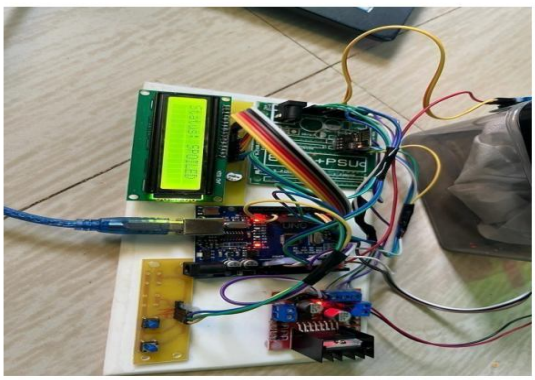


Fig 7.3: Spoiled Food

Fig 7.4: Initially Checking the food

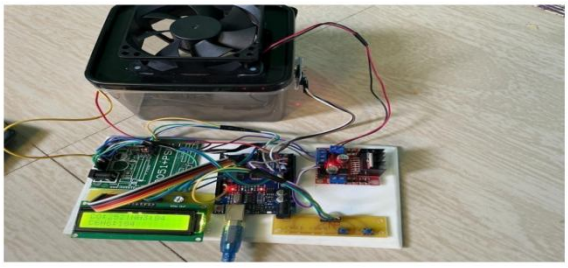


Fig 7.5: Final readings of gases

13. CONCLUSION

In conclusion, IoT and machine learning can be used together to develop a system for detecting the freshness of food. This system can analyze data from various sensors and sources to identify patterns and characteristics that indicate the freshness or spoilage of food. By training machine learning models on this data, the system can accurately classify food as fresh or spoiled in real-time, which can help prevent food waste and improve food safety. However, it is important to ensure that the system is properly calibrated and validated to ensure its accuracy and effectiveness before deployment in real world settings. One of the key advantages of this approach is the ability to collect and analyze various parameters such as temperature, humidity, gas levels, and even visual indicators. ML algorithms can learn from this data to detect patterns and anomalies that indicate potential spoilage or deterioration of food items. By leveraging this information, stakeholders can make informed decisions regarding the storage, transportation, and sale of perishable goods, ensuring that consumers receive fresh and safe products. For instance, if an abnormal temperature rise is detected in a cold storage facility, an alert can be immediately sent to personnel, enabling them to take corrective actions promptly and avoid food loss. This technology also promotes transparency and accountability in the food industry. With the ability to track and record the entire journey of a food product, from farm to table, consumers can access detailed information about its freshness and quality. This fosters trust between consumers and producers, empowering individuals to make informed choices regarding the food they consume.

REFERENCES

[1]. Stephan Leible1, Steffen Schlager2, Moritz schubotz3 " A Review on Blockchain Technology and Blockchain projects Fostering open science, fbloc.2019.00016
[2]. Zibin Zheng1, Sharon Xie2, Hongning Dai2, Xianping Chen3, Huaimin Wang4 " An overview of Blockchain Technology: Architecture, Consensus and future trends" 2017 IEEE
[3]. Amir A.Khwaja1, Adnan Nadeem2, Hafiz Farooq Ahmad3 " A survey on Blockchain Technology Evolution, Architecture and security, IEEE Access 2021,March 2 2021
[4]. J. Xie, H. Tang, T. Huang, F. R. Yu, R. Xie, J. Liu, and Y. Liu, "A survey of blockchain technology applied to smart cities: Research issues and challenges," IEEE Commun. Surveys Tuts., vol.21, no. 3, pp. 2794–2830, Feb. 2019.
[5]. Mu-Hsing kuo1 " Opportunities and challenges of cloud computing to improve health care Services, JMR publication, vol 13, 21.09.2011
[6]. Nalini Subramanian, "Recent security challenges in cloud computing" science direct, vol 71, oct 2018