FISH SAVVY:SMART AQUATECH POND

A PROJECT REPORT

submitted to

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY, KERALA

by

MEENAKSHI S(MUT20EC036)

in partial fulfilment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MUTHOOT INSTITUTE OF TECHNOLOGY AND SCIENCE

VARIKOLI P.O, PUTHENCRUZ - 682308

MAY 2024

ACKNOWLEDGMENTS

First of all to the Great Almighty, the author of knowledge and wisdom for his countless love.We would like to sincerely thank our guides **Prof. Anjali S V** and **Dr.Arunkant A Jose** for their support and valuable guidance. With great respect, we express our sincere thanks to our Head of The Department **Dr. Abhilash Antony** for all the proper guidance and encouragement. We extent our gratitude to the project coordinators **Dr. Prathibha Sudhakaran** , **Dr. Arunkant A Jose** and **Dr. Shajimon K John** for their timely advice, meticulous scrutiny, scholarly advice and scientific approach that helped to a very great extent throughout the project.

We express our heartfelt veneration to all who had been helpful and inspiring throughout this endeavour.

Meenakshi S

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

MEENAKSHI S
(MUT20EC036)

Place : Varikoli
Date : 07/05/2024

MUTHOOT INSTITUTE OF TECHNOLOGY & SCIENCE

Varikoli P.O, Puthencruz-682308

Muthoot Institute of Technology & Science

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the project final report entitled "**FISH SAVVY:SMART AQUATECH POND** " submitted by **Meenakshi S (MUT20EC036)** of Semester VIII is a bonafide account of the work done by her under our supervision

Ms. Anjali S V
(Project Guide)
Asst. Professor
Dept.of ECE
MITS
Varikoli

Dr. Arunkant A Jose
(Project Guide)
Asst. Professor
Dept.of ECE
MITS
Varikoli

Dr. Abhilash Antony
(Head of the Department)
Assoc. Professor
Dept. of ECE
MITS
Varikoli

Dr. Prathibha Sudhakaran
(Project coordinator)
Asst. Professor
Dept.of ECE
MITS
Varikoli

Dr. Shajimon K John
(Project coordinator)
Professor & Dean Academics
Dept.of ECE
MITS
Varikoli

External Examiner

Contents

List of Figures

Abstract

The Smart AquaTech Pond for Pisciculture employs advanced technology to create a highly automated and intelligent aquaculture ecosystem, ensuring the best possible conditions for fish growth and health while enhancing the sustainability and productivity of fish farms. The proposed system uses multiple sensors to measure in real-time water quality parameters such as temperature, turbidity, and purity (TDS) from the fish pond and then send them to a cloud database to allow fish farmers to access them in real-time with their devices (mobile phone, PC, tablets). It also focuses on early disease detection in fish using Deep learning. This system offers real-time insights and remote control through user-friendly applications, improving fish farm sustainability and productivity while reducing disease risks and operational costs. In summary, the Smart AquaTech Pond for Pisciculture leverages technology to optimize fish farming, making it more efficient, environmentally friendly, and profitable.

Chapter 1

Introduction

Our project aims to enhance the sustainability and productivity of aquaculture by developing a system for fish disease detection and real-time water quality monitoring. Leveraging image-based deep learning techniques, we will create a platform capable of early disease detection through the analysis of fish images. Additionally, our system will integrate sensors to monitor crucial water quality parameters like temperature, turbidity, and total dissolved solids (TDS) in real time. By combining these technologies, our project seeks to empower aquaculture farmers with actionable insights to improve fish health, minimize disease outbreaks, and optimize aquaculture operations.

1.1 Scope and Motivation

The Smart AquaTech Pond for Pisciculture introduces a comprehensive aquaculture ecosystem with real-time monitoring and disease detection. It encompasses cloud-based accessibility and user-friendly applications for efficient fish farm management. Driven by a commitment to revolutionize fish farming, the project aims to enhance efficiency, promote sustainability, and mitigate risks. By integrating advanced technology, it aspires to provide fish farmers with tools for proactive decision-making, reducing environmental impact and operational costs while optimizing productivity in a rapidly evolving aquaculture industry.

1. Fish Disease Detection: Using image-based deep learning techniques to detect diseases in fish. This involves training deep learning models using datasets of fish images. The motivation here is to provide a non-invasive and efficient method for detecting diseases early, which can help prevent widespread outbreaks and minimize losses in aquaculture farms.

2. Real-Time Water Quality Monitoring: Monitoring water quality parameters such as temperature, turbidity, and TDS in real-time. The motivation behind this is to ensure optimal conditions for fish growth and health. By continuously monitoring these parameters, farmers can take timely actions to maintain the water quality, which is crucial for the overall well-being of the fish.

3. IoT Implementation: Implementing IoT technologies using ESP32 and ESP8266 modules to collect and transmit data from sensors to a central system. The motivation is to create a system that is efficient,

cost-effective, and easy to deploy in aquaculture farms. This enables farmers to remotely monitor the conditions in their farms and make informed decisions without the need for manual intervention.

4. Integration with Blynk App: Connecting the ESP8266 modules to the Blynk app to visualize the data from the sensors. The motivation here is to provide farmers with a user-friendly interface to access and analyze the data collected by the sensors. This can help farmers gain insights into the conditions in their farms and take appropriate actions as needed.

Overall, the scope of your project is to develop a comprehensive system that combines deep learning for fish disease detection and IoT for real-time water quality monitoring, with the ultimate goal of improving the efficiency and sustainability of aquaculture practices.

1.2 Objectives

1. Developing Deep Learning Models: Train deep learning models using datasets of fish images to accurately detect diseases. This involves selecting an appropriate deep learning architecture (such as ResNet) and optimizing the model for high accuracy and efficiency.

2. Implementing Disease Detection on Raspberry Pi: Develop software to run on the Raspberry Pi that can capture images of fish, preprocess them if necessary, and then use the trained deep learning model to detect diseases in real-time

3. Designing and Implementing the IoT System: Design the IoT system using ESP32 and ESP8266 modules to monitor water quality parameters. This includes selecting and interfacing with the sensors, programming the microcontrollers, and establishing communication between the nodes and the central system.

4. Integrating IoT Data with Disease Detection: Integrate the data from the IoT system (water quality parameters) with the disease detection system (fish images) to provide a comprehensive view of the conditions in the aquaculture farm. This integration can help correlate water quality with disease outbreaks and provide insights for preventive measures.

5. Developing the User Interface: Create a user-friendly interface, possibly using the Blynk app, to visualize the data collected by the IoT system and the results of the disease detection. This interface should allow farmers to easily monitor the conditions in their farms and take actions as needed.

6. Testing and Validation: Conduct thorough testing and validation of the entire system to ensure its accuracy, reliability, and scalability. This includes testing the deep learning models with different datasets, evaluating the IoT system's performance under various conditions, and validating the overall system's effectiveness in real-world aquaculture environments.

1.3 Applications

The pothole detection and filling system project has various applications that can benefit both urban infrastructure management and road users. Here are some of the key applications:

Early Disease Detection: By using image-based deep learning techniques, the system can help farmers detect diseases in fish at an early stage. Early detection can lead to timely intervention, reducing the spread of diseases and minimizing losses.

Optimizing Water Quality: The real-time monitoring of water quality parameters such as temperature, turbidity, and TDS can help farmers optimize conditions for fish growth. By maintaining optimal water quality, farmers can improve the overall health and productivity of their fish.

Remote Monitoring: The IoT system allows farmers to monitor conditions in their aquaculture farms remotely. This can be particularly useful for farmers with large or multiple farms, as they can keep track of conditions without the need for frequent on-site visits

Data-Driven Decision Making: By providing farmers with real-time data on water quality and disease status, the system enables data-driven decision making. Farmers can use this information to adjust feeding schedules, water treatments, and other management practices to optimize fish health and growth.

Research and Development The data collected by the system can also be valuable for research purposes. Researchers can use this data to study the effects of different environmental factors on fish health and behavior, leading to insights that can inform future aquaculture practices.

Chapter 2

Literature Survey Report

2.1 Iot Based Biofloc Automation And Monitoring For Smart Fish Production (2023) [?]

Authors: Ahmed Shafkat, Ezharul Islam

This paper reviews some current IoT based fish farming work and proposed architecture for biofloc agriculture automation. A theoretical comparison study will demonstrate this system as new and unique. This work successfully relates IoT and biofloc technology, as well as introducing a few new features on existing drawbacks. The proposed architecture uses Raspberry pi as a microcontroller that connected to the cloud through Wi-Fi in order to store real-time data in NoSQL database. Seven types of sensors continuously monitor water parameters and four types of actuators take proper steps to main suitable conditions. Air pump, water pump, heat pump, wavemaker, and valve can be controlled by the farmer using comfortable web or mobile application. The system could be developed at low-cost for developing countries like Bangladesh and farmers could achieve higher production with less effort.

2.2 Aquatech:A Smart Fish Farming Automation And Monitoring App(2020) [1]

Authors: Epifelward Niño O. Amora , Kennery V. Romero Rennan C. Amoguis

The growth of fish, the length of time spent, effort and amount of harvest a fish farmer have every season is based on the water environment parameters that a pond has. Controlling these parameters to make the pond environment a desirable one for fish to grow normally is a great help for the fish farmers. With the use of technology, better fish farming results will be achieved. Monitoring and controlling the water environment parameters are great factors to maximize fish production in the fish farm. AquaTech filled in the gaps of the existing studies and it can give benefits to the fish farmers..

2.3 A Survey On Clustering Techniques For Wireless Sensor Network (2018)[2]

Authors: Rudranath Mitra , Diya Nandy

The paper ”A Survey on Clustering Techniques for Wireless Sensor Network” provides a comprehensive overview of clustering techniques in wireless sensor networks (WSNs). It discusses the significance of WSNs in various applications and the challenges associated with energy consumption and network scalability. The paper explores the impact of heterogeneity on WSNs and presents a comparative study of different clustering algorithms. It also delves into the concept of cluster hierarchy, the role of cluster heads, and the communication dynamics within clustered networks. Additionally, the paper highlights the importance of reliability and energy efficiency in WSN protocols and concludes with insights into the future scope of research in this field. The authors, Rudranath Mitra and Diya Nandy, have provided a valuable resource for researchers and practitioners in the field of wireless sensor networks, offering a comprehensive survey of clustering techniques and their implications for network performance and reliability.

2.4 Range Free Localization Techniques In Wireless Sensor Networks: A Review (2015)[3]

Authors: SP Singh, SC Sharma

The paper, provides a comprehensive examination of various range-free localization methods employed in Wireless Sensor Networks (WSNs). An in-depth analysis of existing techniques, outlining their principles, advantages, and limitations is conducted. The review encompasses well-established algorithms like DV-Hop, Amorphous, and APIT, shedding light on their applicability and performance in different scenarios. Through this survey, the authors aim to offer researchers and practitioners valuable insights into the state-of-the-art in range-free localization, fostering a deeper understanding of the available techniques and guiding the selection of suitable methods based on specific WSN requirements and environmental factors.

2.5 Image Processing Technique To Detect Fish Disease (2015)[4]

Authors: Hitesh C Rituraj P and Prodipto D

This paper by Hitesh C, Rituraj P, and Prodipto D focuses on employing image processing techniques for the detection of fish diseases. The authors propose a method that utilizes advanced image analysis to identify and diagnose diseases affecting fish populations. By leveraging image processing algorithms, the system aims to recognize distinct patterns, anomalies, or symptoms in fish images that may indicate the presence of diseases. The approach likely involves the extraction of relevant features from digital images of fish, followed by classification or identification using computational models. This innovative application of image processing technology holds promise for efficient and non-invasive monitoring of fish health in aquaculture settings, contributing to early disease detection and improved management practices in the fisheries industry.

2.6 Real Time Water Quality Monitoring Using Chemical Sensors (2019)[5]

Authors: Irina Yaroshenko, Dmitry Kirsanov, Andrey Legin
The paper on "Real-Time Water Quality Monitoring with Chemical Sensors" addresses the critical need for continuous and instantaneous monitoring of water quality using chemical sensors. The study presents a system that employs real-time sensor technologies to assess and track various chemical parameters in water bodies. These sensors enable the rapid detection of pollutants, such as heavy metals or contaminants, providing a timely and accurate representation of water quality. The approach likely involves deploying sensor networks that transmit data in real-time, facilitating swift responses to changes in water conditions. This innovative method holds significant implications for environmental management, allowing for proactive interventions to safeguard water resources and ensure the timely implementation of corrective measures to maintain or improve water quality in both natural and industrial settings.

Chapter 3

Problem Statement

In aquaculture, early detection of diseases in fish and maintaining optimal water quality are critical for ensuring the health and productivity of fish stocks. However, current methods for disease detection and water quality monitoring often rely on manual labor and periodic sampling, which can be time-consuming, labor-intensive, and prone to human error.To address these challenges, this project aims to develop a system for fish disease detection and real-time water quality monitoring in aquaculture using image-based deep learning techniques and IoT technologies. The system will utilize deep learning models trained on fish image datasets to detect diseases in fish non-invasively and efficiently.

Additionally, the system will include IoT nodes equipped with sensors for monitoring key water quality parameters such as temperature, turbidity, and TDS in real-time. These nodes will communicate with a central system, which will provide farmers with a user-friendly interface to visualize the data and make informed decisions.By combining deep learning for disease detection and IoT for water quality monitoring, the project seeks to revolutionize aquaculture practices by providing farmers with a comprehensive and automated system for managing fish health and farm conditions.

Chapter 4

System Methodology

4.1 Methodology

Data Collection: Gather a diverse dataset of fish images representing various species and potential diseases. Also, collect data on water quality parameters such as temperature, turbidity, and TDS from different aquaculture farms.
Data Preprocessing: Preprocess the fish images to standardize their size, orientation, and color. For the water quality data, clean and filter the data to remove noise and outliers.
Deep Learning Model Training: Train a deep learning model, such as a convolutional neural network (CNN), using the preprocessed fish image dataset. Use transfer learning if necessary to leverage pre-trained models and improve training efficiency.
Disease Detection System Development: Develop software to run on the Raspberry Pi that captures images of fish using a camera, preprocesses the images, and then uses the trained deep learning model to detect diseases in real-time. Integrate this system with the IoT network for data transmission and monitoring.
IoT System Implementation: Design and implement the IoT system using ESP32 and ESP8266 modules to monitor water quality parameters. Select and interface with the sensors, program the microcontrollers, and establish communication between the nodes and the central system.
Integration of IoT and Disease Detection Systems: Integrate the data from the IoT system (water quality parameters) with the disease detection system (fish images) to provide a comprehensive view of the conditions in the aquaculture farm. Develop algorithms to correlate water quality with disease outbreaks.
User Interface Development: Create a user-friendly interface, possibly using the Blynk app, to visualize the data collected by the IoT system and the results of the disease detection. Enable farmers to easily monitor the conditions in their farms and take actions as needed.
Testing and Validation: Conduct thorough testing and validation of the entire system to ensure its accuracy, reliability, and scalability. Test the deep learning models with different datasets and evaluate the IoT system's performance under various conditions.
Deployment and Evaluation: Deploy the system in real aquaculture farms and evaluate its effectiveness in detecting diseases early and maintaining optimal water quality. Gather feedback from farmers and make any necessary improvements to the system.

4.2 Working

The working of your project can be described in the following steps:

1. Image Capture: The system begins by capturing images of fish using a camera connected to the Raspberry Pi. These images are then processed and analyzed for disease detection.

2. Disease Detection: The captured images are preprocessed to standardize them for input to the deep learning model. The trained deep learning model is then used to analyze these images and detect any signs of disease in the fish.

3. Data Transmission: Simultaneously, the IoT system continuously monitors water quality parameters such as temperature, turbidity, and TDS using sensors connected to the ESP32 nodes. The ESP32 nodes collect this data and transmit it to the central system using the ESP8266 modules.

4. Data Integration: The data from the disease detection system (fish images) and the IoT system (water quality parameters) are integrated into a central system. Algorithms are used to correlate water quality data with disease outbreaks, providing a comprehensive view of the conditions in the aquaculture farm.

5. User Interface: The integrated data is visualized through a user-friendly interface, possibly using the Blynk app, allowing farmers to easily monitor the conditions in their farms. The interface provides real-time updates on disease detection and water quality, enabling farmers to take timely actions.

6. Alerting System: The system can be configured to send alerts to farmers in case of abnormal water quality conditions or detected diseases. This allows farmers to respond promptly and implement necessary measures to mitigate risks.

7. Continuous Monitoring: The system operates continuously, capturing images, monitoring water quality, and providing updates to the user interface. This continuous monitoring ensures that any issues are detected early, helping to maintain the health and productivity of the fish in the aquaculture farm.

The overall working of the fish disease detection and water monitoring system was done using Raspberry Pi,Deep learning models, Camera for disease detection part and ESP32, ESP8266, sensors, and cloud platform that is a Blynk App.

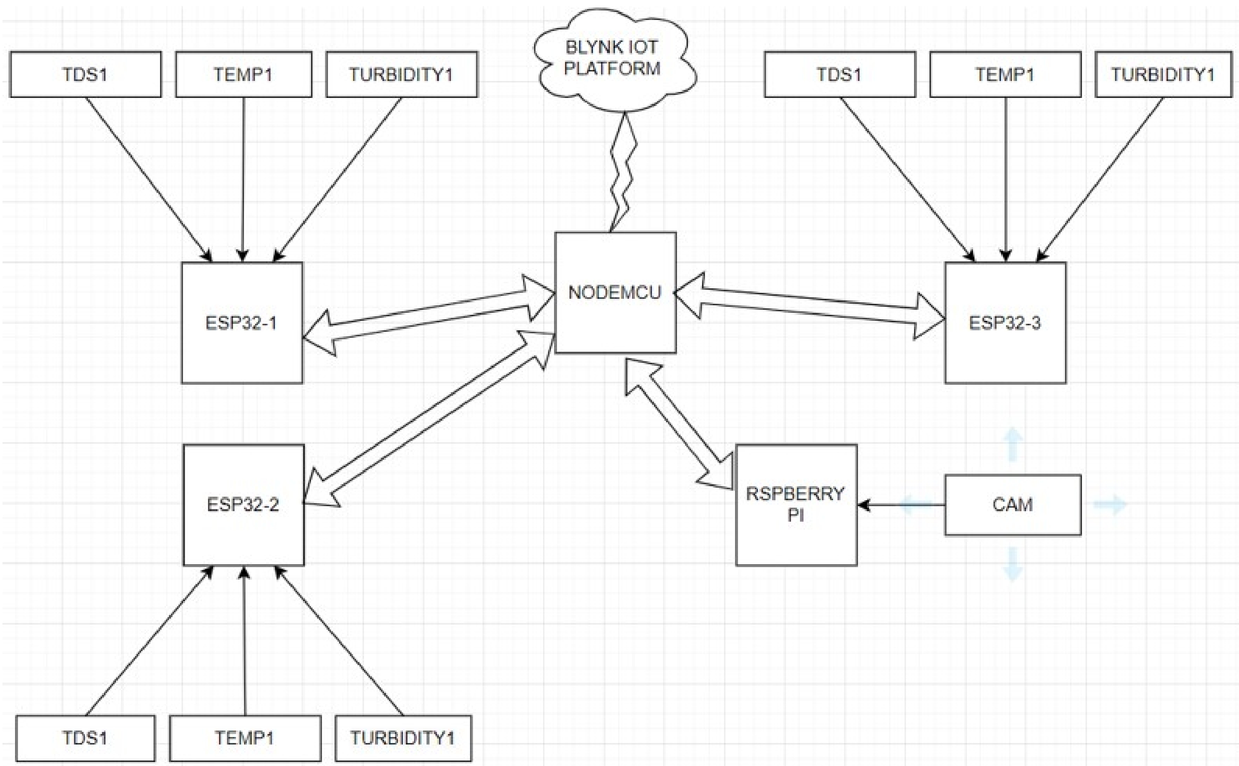

Figure 4.1: Block diagram of system architecture

Chapter 5

System Analysis

The Disease Detection and water quality monitoring System can be implemented by using the following components.

1. ESP32
2. ESP8266
3. Turbidity sensor
4. Temperature sensor
5. TDS sensor
6. Raspberry Pi
7. camera

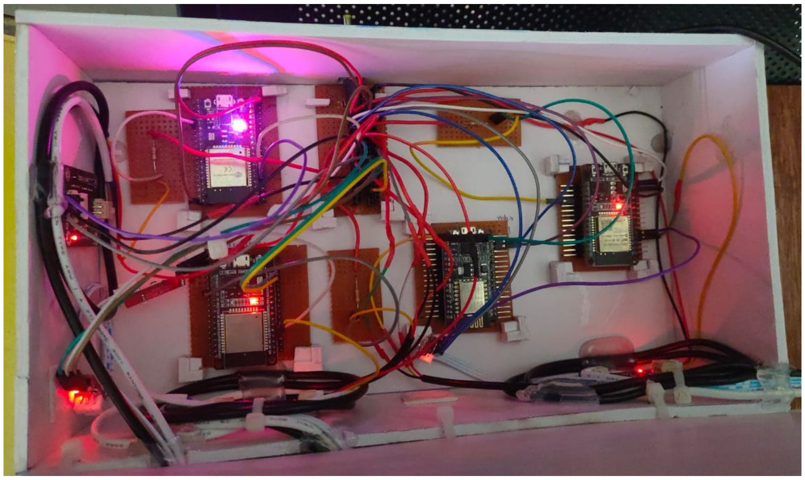

Figure 5.1: Connection

Each component of the design is described in more details below:

5.1 ESP32

ESP32 is a low-cost System on Chip (SoC) Microcontroller from Espressif Systems, the developers of the famous ESP8266 SoC. It is a successor to ESP8266 SoC and comes in both single-core and dual-core variations of the Tensilica's 32-bit Xtensa LX6 Microprocessor with integrated Wi-Fi and Bluetooth.The good thing about ESP32, like ESP8266 is its integrated RF components like Power Amplifier, Low-Noise Receive Amplifier, Antenna Switch, Filters and RF Balun. This makes designing hardware around ESP32 very easy as you require very few external components.

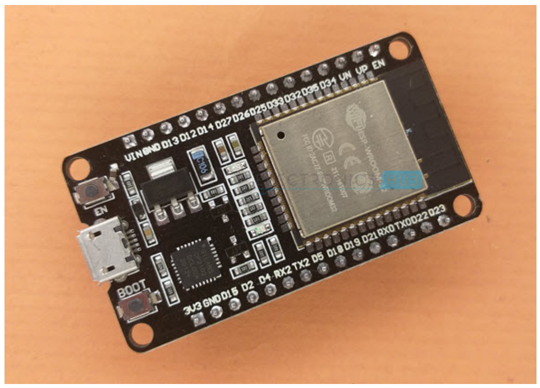

Figure 5.2: ESP32

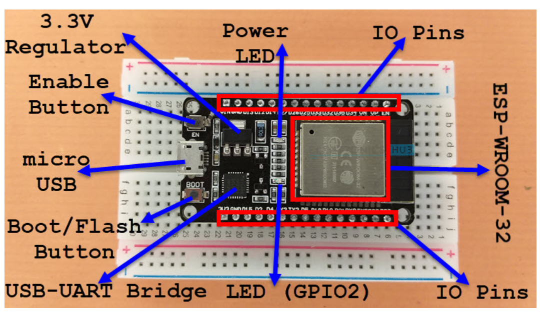

Figure 5.3: Pin diagram of ESP32

SPECIFICATIONS:

1. Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
2. 520 KB of SRAM, 448 KB of ROM and 16 KB of RTC SRAM.
3. Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
4. Support for both Classic Bluetooth v4.2 and BLE specifications.
5. 34 Programmable GPIOs.
6. Up to 18 channels of 12-bit SAR ADC and 2 channels of 8-bit DAC

7. Serial Connectivity include 4 x SPI, 2 x I2C, 2 x I2S, 3 x UART.
8. Ethernet MAC for physical LAN Communication (requires external PHY).
9. 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
10. Motor PWM and up to 16-channels of LED PWM.
11. Secure Boot and Flash Encryption.
12. Cryptographic Hardware Acceleration for AES, Hash (SHA-2), RSA, ECC and RNG.

5.2 ESP8266 (NodeMCU)

NodeMCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

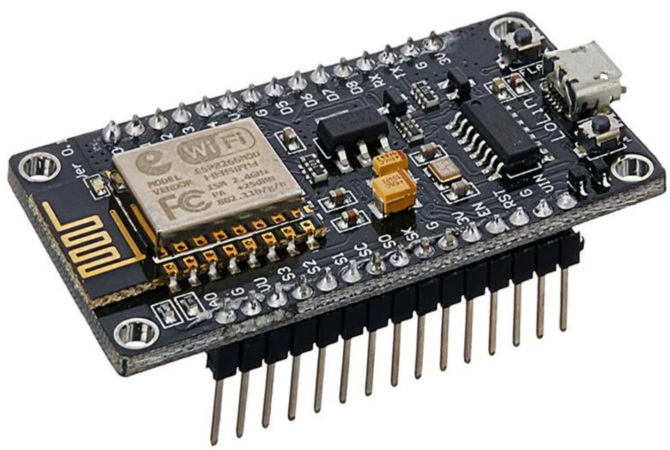

Figure 5.4: ESP8266

SPECIFICATIONS:

1. Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
2. Operating Voltage: 3.3V
3. Input Voltage: 7-12V
4. Digital I/O Pins (DIO): 16
5. Analog Input Pins (ADC): 1
6. UARTs: 1
7. SPIs: 1

8. I2Cs: 1
9. Flash Memory: 4 MB
10. SRAM: 64 KB
11. Clock Speed: 80 MHz
12. USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
13. PCB Antenna
14. Small Sized module to fit smartly inside your IoT projects

5.3 Turbidity Sensor

The Turbidity Sensor emits at its end an infrared light, imperceptible to human vision, capable of detecting particles that are suspended in water, measuring the light transmittance and the dispersion rate, which changes according to the Amount of TSS (Total Suspended Solids), increasing the turbidity of the liquid whenever levels increase.
Turbidity Sensor has an end specially prepared for direct contact, having an electronic module to amplify and send the received data to the microcontroller of the project.

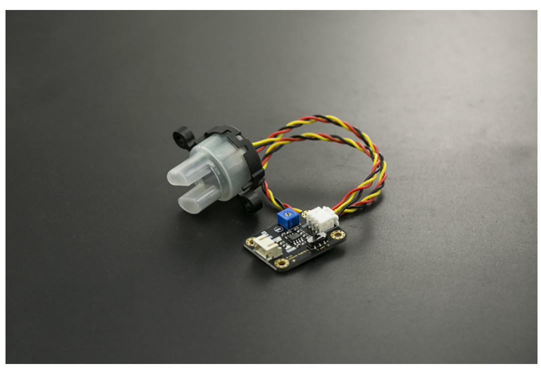

Figure 5.5: Turbidity sensor

SPECIFICATIONS:

1. Operating Voltage: 5V DC
2. Operating Current: 40mA (MAX)
3. Response Time : ¡500ms
4. Insulation Resistance: 100M (Min)
5. Output Method:
   Analog output: 0-4.5V
   Digital Output: High/Low level signal (you can adjust the threshold value by adjusting the potentiometer)

6. Operating Temperature: 5℃ 90℃

7. Weight: 30g

5.4 Temprature Sensor

They work great with any microcontroller using a single digital pin, and you can even connect multiple ones to the same pin, each one has a unique 64-bit ID burned in at the factory to differentiate them. Usable with 3.0-5.0V systems. When using with microcontroller put a 4.7k resistor to sensing pin, which is required as a pull-up from the DATA to VCC line.

Figure 5.6: Temperature sensor(DS18B20)

SPECIFICATIONS:

1. Programmable Digital Temperature Sensor

2. Communicates using 1-Wire method

3. Operating voltage: 3V to 5V

4. Temperature Range: -55°C to +125°C

5. Accuracy: ±0.5°C

6. Output Resolution: 9-bit to 12-bit (programmable)

7. Unique 64-bit address enables multiplexing

8. Conversion time: 750ms at 12-bit

9. Programmable alarm options

5.5 TDS Sensor

A Total Dissolved Solids (TDS) sensor is a device used to measure the concentration of dissolved solids in a liquid. It operates on the principle of conductivity, where the electrical conductivity of a solution is directly related to the concentration of dissolved ions. When ions such as salts, minerals, metals, and other organic compounds are dissolved in water, they increase its conductivity.TDS sensors typically consist of electrodes that are submerged into the liquid being tested. These electrodes generate an electrical current that passes through the liquid. The conductivity of the liquid affects the electrical resistance encountered by the current, which the sensor then measures. By correlating this resistance with known standards, the sensor can determine the TDS level of the liquid. TDS sensors find application in various industries including water treatment, aquaculture, hydroponics, and beverage production. Monitoring TDS levels is crucial for ensuring water quality, determining the purity of drinking water, and assessing the effectiveness of water purification processes. High TDS levels can indicate contamination or excessive mineral content, while low levels might signify inadequate mineralization. Overall, TDS sensors play a vital role in maintaining the quality and safety of water in diverse settings.

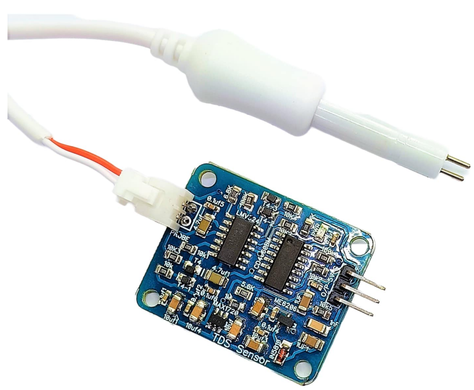

Figure 5.7: TDS Sensor

SPECIFICATIONS:

1. Operating Voltage: 3.3-5V

2. Measurement Range: 0-5000ppm

3. Operating Temperature: 0°C-50°C

4. Low power consumption

5. Requires periodic calibration to maintain accuracy
6. Weight: less than 100gm
7. Accuracy: ±0.5°C

5.6 Raspberry Pi

The Raspberry Pi is a small, affordable, single-board computer developed by the Raspberry Pi Foundation. It's designed to promote computer science education and DIY projects. The Raspberry Pi features a credit card-sized form factor and is powered by an ARM-based processor, along with RAM, storage, and various connectivity options packed onto a single board. It runs on Linux-based operating systems such as Raspbian and supports a wide range of programming languages including Python, C/C++, and Java.Despite its compact size, the Raspberry Pi is surprisingly versatile, capable of performing tasks such as web browsing, word processing, media playback, and even basic gaming. It also serves as an excellent platform for learning coding, electronics, and IoT (Internet of Things) development due to its GPIO (General Purpose Input/Output) pins, which allow it to interface with external sensors, actuators, and other hardware components.The Raspberry Pi has gained immense popularity among hobbyists, educators, and tinkerers worldwide, spawning a vibrant community of enthusiasts who share projects, tutorials, and resources. Its low cost, accessibility, and robust capabilities make it an ideal tool for experimentation, prototyping, and creating innovative solutions in various domains, from home automation and robotics to digital art and education.

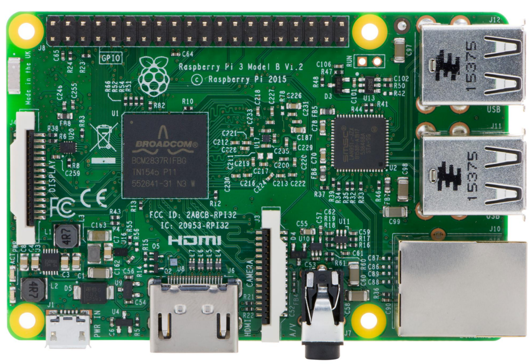

Figure 5.8: Raspberry Pi 3

SPECIFICATIONS:

1. Ethernet: 10/100 Mbps Ethernet port for wired network connectivity
2. Wi-Fi: Dual-band 802.11 b/g/n wireless LAN for wireless internet connectivity
3. Built-in Bluetooth 4.1/BLE (Bluetooth Low Energy) for wireless communication with Bluetooth-enabled devices
4. Provides 40 GPIO (General Purpose Input/Output) pins
5. Offers four USB 2.0 ports

6. Power: 5V
7. Processor: Features a 1.2 GHz quad-core ARM Cortex-A53 CPU
8. Memory: Equipped with 1GB of LPDDR2 RAM

Chapter 6

Result

The project achieved the development of an integrated system for fish disease detection and real-time water quality monitoring in aquaculture. Utilizing deep learning models trained on fish image datasets, the system can accurately detect diseases, facilitating early intervention and treatment. Concurrently, real-time monitoring of water quality parameters like temperature, turbidity, and TDS empowers farmers with critical insights into their aquaculture system's health. This information allows them to promptly optimize conditions for optimal fish growth. By combining advanced technology with practical applications, the system enhances disease management practices and promotes sustainable aquaculture practices.

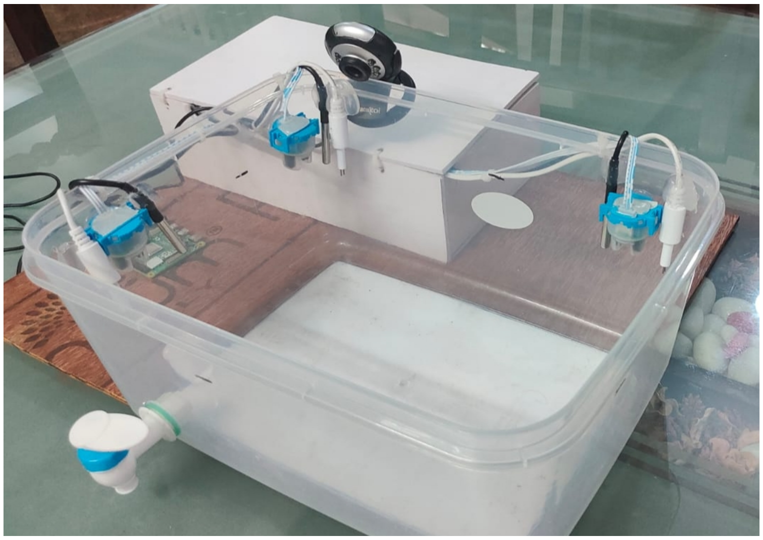

Figure 6.1: System developed for monitoring water quality parameters

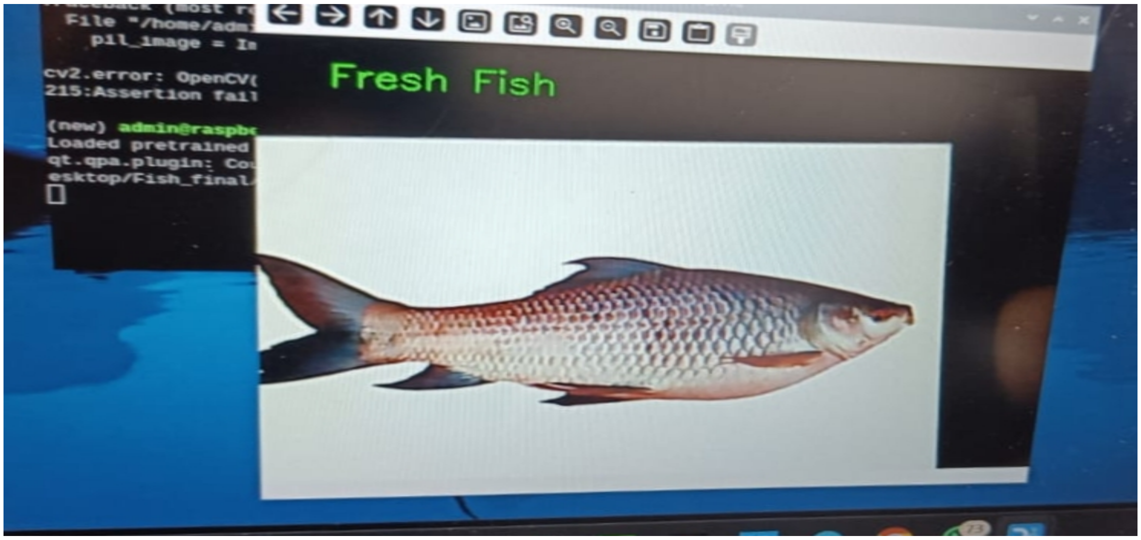

Figure 6.2: Healthy fish detected

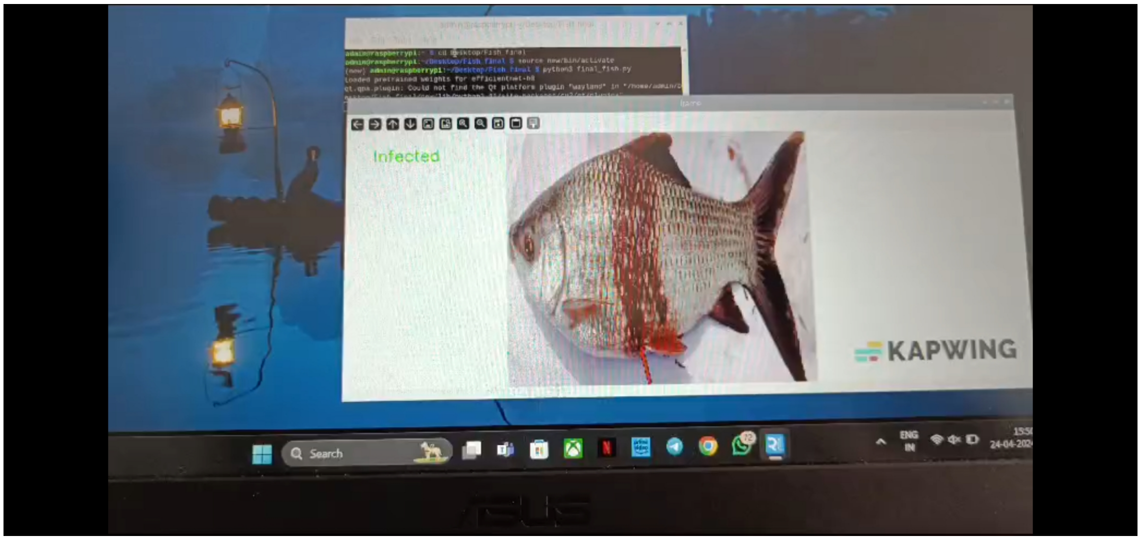

Figure 6.3: Infected fish detected

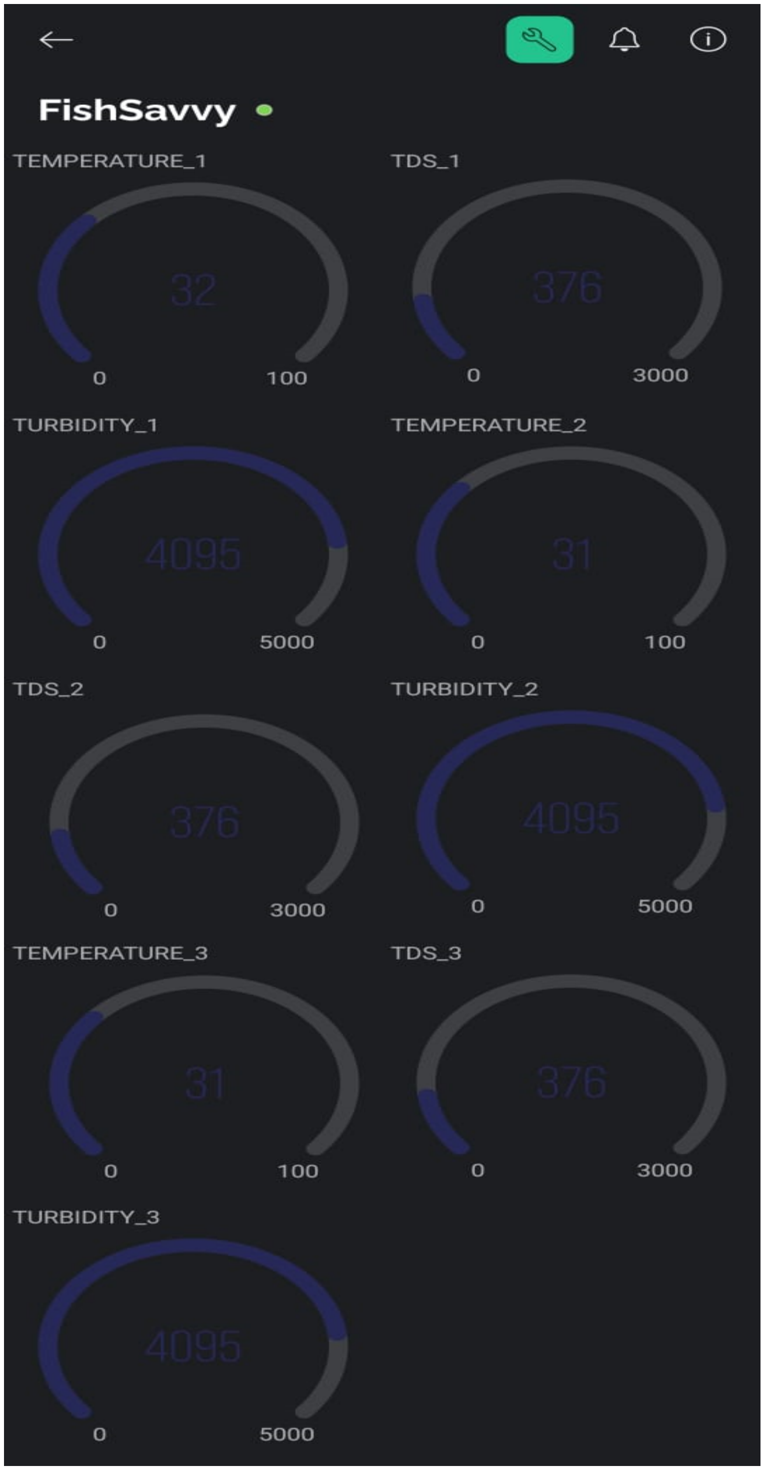

Figure 6.4: Temperature, turbidity and tds readings via blynk app

Chapter 7

Conclusion and Scope for Future Work

In conclusion,our project stands as a testament to the feasibility and effectiveness of harnessing advanced technologies, including image-based deep learning and wireless sensor networks, for the dual purpose of fish disease detection and real-time water quality monitoring in aquaculture. By seamlessly integrating these innovations, we have not only demonstrated their potential but also provided aquaculture farmers with a powerful tool for enhancing fish health and productivity.

At the core of our system lies the utilization of deep learning models trained on extensive fish image datasets. These models exhibit remarkable accuracy in identifying diseases, enabling early intervention and treatment. Concurrently, our implementation of wireless sensor networks facilitates the real-time monitoring of critical water parameters such as temperature, turbidity, and TDS. This amalgamation empowers farmers with invaluable insights into their aquaculture system's dynamics, enabling timely adjustments to optimize conditions for fish growth.

Looking forward, our vision entails several avenues for further enhancing our system's capabilities. Firstly, we aim to expand the repertoire of detectable diseases and refine our deep learning models to improve their accuracy and speed. Additionally, integrating more advanced sensors for monitoring additional parameters like pH and dissolved oxygen levels is on our agenda. Furthermore, we aspire to augment our platform by incorporating automated feeding systems and environmental controls, paving the way for a fully automated and optimized aquaculture management system.

In summary, our project not only underscores the transformative potential of advanced technologies but also embodies a commitment to the sustainable evolution of the aquaculture industry. By providing farmers with a comprehensive solution for disease detection and water quality monitoring, we aspire to contribute significantly to the sector's growth and resilience. With continuous refinement and innovation, we believe our project holds the promise of revolutionizing aquaculture practices, ensuring their long-term viability and sustainability.

7.1 Future Expansion

The following ideas can be considered for the future expansion of the model:

1. Danger signs can be shown when the levels of water quality parameters dangerously varies from the required levels.

2. Accuracy of early stage disease detection can be improved by including more images in the dataset.

3. Continuously train deep learning models to recognize a broader range of fish diseases, including both common and emerging pathogens, to further improve the system's diagnostic capabilities.

4. Continuously gather feedback from end-users, including aquaculture farmers, researchers, and industry experts, to identify pain points, user requirements, and areas for improvement, and use this feedback to iteratively refine and enhance the system.

7.2 Future Scope

The future scope of the project holds significant potential for various areas of development and application. Here are some potential future directions and opportunities for such a project:

1.Enhanced Disease Detection: Expand the range of diseases detectable by deep learning models, encompassing a broader spectrum of common fish ailments. Collaborate with domain experts to gather diverse datasets for robust model training.

2.Model Optimization: Continuously refine and optimize deep learning models to improve accuracy, speed, and efficiency. Explore advanced techniques such as transfer learning and ensemble methods to enhance model performance.

3. Integration of Additional Sensors: Incorporate advanced sensors for monitoring additional water quality parameters critical for fish health, such as pH, dissolved oxygen, ammonia, and nitrate levels. Ensure seamless integration with existing sensor networks for comprehensive monitoring.

4. Automation and Control Systems: Integrate automated feeding systems and environmental controls into the platform to create a fully automated aquaculture management system. Implement smart algorithms for adaptive feeding and environmental adjustments based on real-time sensor data.

5. Data Analytics and Insights: Develop advanced data analytics tools and algorithms to derive actionable insights from the collected data. Utilize machine learning algorithms for predictive analytics, disease outbreak forecasting, and optimization of aquaculture operations.

6. Collaborative Research and Partnerships: Foster collaborations with research institutions, industry partners, and aquaculture stakeholders to leverage expertise, resources, and funding opportunities. Engage in joint research projects to address emerging challenges and advance the state-of-the-art in aquaculture technology.

7.User Interface and Experience: Enhance the user interface and experience of the integrated system to ensure ease of use, accessibility, and user engagement. Incorporate intuitive dashboards, visualization tools, and mobile applications for seamless interaction and decision-making.

8.Regulatory Compliance and Certification: Ensure compliance with regulatory standards and certification requirements for aquaculture facilities. Collaborate with regulatory agencies and industry associations to address regulatory challenges and obtain necessary certifications for the integrated system.

Overall, the expansion points outline a strategic roadmap for advancing the project's capabilities and impact in aquaculture. By focusing on enhancing disease detection, optimizing models, integrating additional sensors, and implementing automation, the project aims to create a comprehensive and efficient system for monitoring and managing aquaculture operations. Transitioning to a cloud-based platform, fostering collaborations, and ensuring regulatory compliance further contribute to the project's scalability, reliability, and industry relevance. Additionally, the emphasis on user experience, education, and outreach underscores a commitment to empowering aquaculture stakeholders with accessible and user-friendly tools while promoting knowledge exchange and capacity building. Collectively, these expansion points reflect a holistic approach towards leveraging technology to address key challenges in aquaculture, ultimately fostering sustainable practices, improving productivity, and contributing to the long-term viability of the industry.

Bibliography

1 “Noraini Hasan, Shafaf Ibrahim, Anis Aqilah Azlan , “Fish Diseases Detection Using Convolutional Neural Network (CNN).” Int. J. Nonlinear Anal. Appl, vol. 13, 2022.

[0] [1] “Sucipto, Kusrini, Emha Luthfi Taufiq (2016). “Classification Method of Multi-class on C4.5 Algorithm for Fish Diseases”, 2nd International Conference on Science in Information Technology (ICSITech) : ”Information Science for Green Society and Environment“, 26-27 October 2016.

[2] “Nishq Poorav Desai , Mohammed Farhan Balucha, Akshara Makrariyab, Rabia MusheerAziz .View of Image processing Model with Deep Learning Approach for Fish Species Classification. Turcomat.org. Published 2023. Accessed January 3, 2023.

[3] “S. Albawi,T.A.Mohammed and S. Al-Zawi, Understandingofa convolutional neural network, 2017 Int. Conf.Engin.Technol.(2017)1–6.

[4] Lakmal, H. K. I. S., and Maheshi B. Dissanayake. ”Pothole Detection with Image Segmentation for Advanced Driver Assisted Systems.” In 2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE), pp. 308-311. IEEE, 2020.

[5] A. Qureshi, “10 Best Water Quality Testers For Professionals,” Wonderful Engineering, 2016. [Online]. Available: http://wonderfulengineering.com/10-bestwater-quality-testers-for-professionals/. [Accessed March 2017].

[6] R.A.Roseline and Dr.P.Sumathi, Energy Efficient Routing Protocol and Algorithms for Wireless Sensor Networks-A Survey. Global Journal of Computer Science and Technology, vol.11, December 2011.

[7] Heinzelman W, Chandrakasan A, Balakrishnan H. Energy Efficient Communication Protocol for Wireless Microsensor Networks. In Proceedings of the 33rd Hawaii International Conference on System Sciences. Maui: IEEE Computer Society, 2000, Vol.2: 3005-3014.

[8] T.S. Rappaport. “Wireless Communications: Principles and Practice”, Second Ed., Prentice Hall, 2002

[9] C. Perkins, E. Belding-Royer, S. Das, ”Ad-hoc On Demand Distance Vector (AODV) Routing”, IETF Network Working Group, RFC 3561, July 2003.

[10] S. Halder and A. Ghosal, “A survey on mobile anchor assisted localization techniques in wireless sensor networks,” Wireless Networks, vol. 22, no. 7, pp. 2317–2336, 2016.

[11] Xiaoying Yang* and Wanli Zhang,"An Improved DV-Hop Localization Algorithm Based on Hop Distance and Hops Correction" International Journal of Multimedia and Ubiquitous Engineering, Vol.11, pp.319-328,2016

[12] Budi Prijo Sembodo and Novendra Geofanda Pratama, "Smart Aquarium Based Microcontroller", Journal of Applied Electrical Science Technology - University of PGRI Adi Buana Surabaya, Vol. 03, No. 2, 2021.

Appendix A

Program Code

TRANSMITTER CODE

```
#include <esp_now.h>
#include <WiFi.h>
#include <OneWire.h>
#include <DallasTemperature.h>

#define TDS_PIN 34       // Analog pin to which TDS meter is connected
#define ONE_WIRE_BUS 2     // Digital pin to which DS18B20 is connected
#define TURBIDITY_PIN 35   // Analog pin to which TS-300B turbidity sensor is connected

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// Replace with the MAC address of your ESP8266 receiver
uint8_t broadcastAddress[] = {0x48, 0x55, 0x19, 0xEC, 0x8D, 0xF8};

typedef struct struct_message {
  char a[32];
  int sensorValue;
  float floatValue;
  int tdsValue;
  float temperature;
  int turbidityValue;
} struct_message;

struct_message myData;

esp_now_peer_info_t peerInfo;
```

```
// Callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t sendStatus) {
  Serial.print("Last Packet Send Status: ");
  if (sendStatus == ESP_NOW_SEND_SUCCESS){
    Serial.println("Delivery success");
  }
}


void setup() {
  Serial.begin(115200);
  sensors.begin();
  WiFi.mode(WIFI_STA);

  // Initialize ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("ESP-NOW initialization failed");
    return;
  }

  //esp_now_set_self_role(ESP_NOW_ROLE_CONTROLLER);
  esp_now_register_send_cb(OnDataSent);
  // register peer
  peerInfo.channel = 0;
  peerInfo.encrypt = false;
  // register first peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}

void loop() {
    int tdsValue2 = readTDS();          // Read TDS value from the sensor
    float temperature2 = readTemperature(); // Read temperature from DS18B20
    int turbidityValue2 = readTurbidity();  // Read turbidity value from TS-300B

    strcpy(myData.a, "Sensor data from station 1");
    myData.floatValue = 3.0;
```

```
    myData.tdsValue = tdsValue2;
    myData.temperature = temperature2;
    myData.turbidityValue = turbidityValue2;

    //myData.sensorValue = myData.tdsValue + myData.temperature + myData.turbidityValue;

    esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));
    delay(2000);
}

int readTDS() {
  // Read analog value from TDS sensor
  int analogValue = analogRead(TDS_PIN);

  // Convert analog value to TDS value (adjust calibration values as needed)
  int tdsValue = map(analogValue, 0, 1023, 0, 5000);  // Assuming the TDS sensor range is 0-5000 ppm

  return tdsValue;
}

float readTemperature() {
  sensors.requestTemperatures();  // Request temperature readings from DS18B20
  return sensors.getTempCByIndex(0);  // Assuming only one DS18B20 is connected
}

int readTurbidity() {
  // Read analog value from turbidity sensor
  int turbidityValue = analogRead(TURBIDITY_PIN);

  // Convert analog value to turbidity value (adjust calibration values as needed)
  // You may need to refer to the TS-300B turbidity sensor documentation for calibration details
  //int turbidityValue = map(analogValue, 0, 1023, 0, 100);

  return turbidityValue;
}
```

RECEIVER CODE

```
#include <ESP8266WiFi.h>
#include <espnow.h>
#include <BlynkSimpleEsp8266.h>
```

```
#define BLYNK_TEMPLATE_ID "TMPL3gagPe82x"
#define BLYNK_TEMPLATE_NAME "FishSavvy"
#define BLYNK_AUTH_TOKEN "7EftVvX7lLvfcah7g2yyjOvzTjXulkbB"
#define BLYNK_PRINT Serial


typedef struct struct_message {
    char a[32];
    int sensorValue;
    float floatValue;
    int tdsValue;
    float temperature;
    int turbidityValue;
} struct_message;

// Create a struct_message called myData
struct_message myData;

int temper1, tds1, turb1,temper2, tds2, turb2,temper3, tds3, turb3;

const char *ssid = "VERDANT PW 4G";
const char *pass = "Verdant@123456";

void OnDataRecv(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.println("Received data:- ");
  Serial.print("Node: ");
  Serial.println(myData.floatValue);

  //Check if floatValue is 1.0
  if (myData.floatValue == 1.0) {
    temper1 = myData.temperature;
    Serial.print("temperature1: ");
    Serial.println(temper1);
    Blynk.virtualWrite(V0, temper1);
    tds1 = myData.tdsValue;
    Serial.print("tdsValue1: ");
    Serial.println(tds1);
    Blynk.virtualWrite(V1, tds1);
    turb1 = myData.turbidityValue;
    Serial.print("turbidity1: ");
```

```
    Serial.println(turb1);
    Blynk.virtualWrite(V2, turb1);
    Blynk.run();
  }
  //Check if floatValue is 2.0
  if (myData.floatValue == 2.0) {
    temper2 = myData.temperature;
    Serial.print("temperature2: ");
    Serial.println(temper2);
    Blynk.virtualWrite(V3, temper2);
    tds2 = myData.tdsValue;
    Serial.print("tdsValue2: ");
    Serial.println(tds2);
    Blynk.virtualWrite(V4, tds2);
    turb2 = myData.turbidityValue;
    Serial.print("turbidity2: ");
    Serial.println(turb2);
    Blynk.virtualWrite(V5, turb2);
    Blynk.run();
  }
  //Check if floatValue is 3.0
  if (myData.floatValue == 3.0) {
    temper3 = myData.temperature;
    Serial.print("temperature3: ");
    Serial.println(temper3);
    Blynk.virtualWrite(V6, temper3);
    tds3 = myData.tdsValue;
    Serial.print("tdsValue3: ");
    Blynk.virtualWrite(V7, tds3);
    Serial.println(tds3);
    turb3 = myData.turbidityValue;
    Serial.print("turbidity3: ");
    Serial.println(turb3);
    Blynk.virtualWrite(V8, turb3);
    Blynk.run();
  }
}

void setup() {
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
```

```
  //delay(1000);
  WiFi.mode(WIFI_STA);

  // Initialize ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("ESP-NOW initialization failed");
    return;
  }

  // Set callback for received data
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);

  // Your setup code here

}
```

DISEASE DETECTION CODE

```
import cv2
from PIL import Image
import numpy as np
import torch
import torch.nn as nn
import torchvision.transforms as transforms
from efficientnet_pytorch import EfficientNet
import requests

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

class EfficientNetModel(nn.Module):
    def _init_(self, num_classes):
        super()._init_()
        self.model = EfficientNet.from_pretrained('efficientnet-b0')
        self.model._fc = nn.Linear(self.model._fc.in_features, num_classes)

    def forward(self, x):
        x = self.model(x)
        return x
```

```
model = EfficientNetModel(num_classes=2)
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
model.to(device)

# Load weights from a file
weights = torch.load('model-ckpt.pth', map_location=torch.device('cpu'))
model.load_state_dict(weights)

# Define the transformation to be applied to input images
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

# Set up Blynk
url = "https://blynk.cloud/external/api/update"
token = "6ELplz5W_bFCsc0MUNzFaULNokUb8rEV"

# Open a connection to the camera (0 is usually the default camera)
cap = cv2.VideoCapture("Untitled_Project_V1.mp4")

# Set model to evaluation mode
model.eval()

while True:
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Convert the frame to PIL Image format
    pil_image = Image.fromarray(cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))

    # Apply the transformation to the input image
    input_tensor = transform(pil_image)
    input_tensor = input_tensor.unsqueeze(0)  # Add a batch dimension

    # Perform inference
    with torch.no_grad():
        input_tensor = input_tensor.to(device)
        outputs = model(input_tensor)
        probabilities = torch.softmax(outputs, dim=1)
```

```
        predicted_class = torch.argmax(probabilities, dim=1).item()

    # Prepare text for display
    if predicted_class == 0:
        text = "Fresh Fish"
        pin_value = 50
    else:
        text = "Infected"
        pin_value = 30

    # Display the result
    cv2.putText(frame, text, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.imshow('frame', frame)

    # Update Blynk virtual pin value using Blynk API
    params = {"token": token, "v0": pin_value}
    requests.get(url, params=params)

    # Break the loop if 'q' is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release the camera and close OpenCV windows
cap.release()
cv2.destroyAllWindows()
```