REAL TIME SYSTEM AND INTERNET OF THINGS FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA

IOT-BASED PATIENT HEALTH MONITORING SYSTEM

GROUP IP-19/P-19/R-19

RYAN SAFA TJENDANA	**2206826835**
DRASSETA ALIYYU DARMANSYAH	**STUDENT ID 2**
DARREN ADAM DEWANTORO	**22068616600**
KEVIN ARIONO	**2206059603**

PREFACE

We express our gratitude to God Almighty for His blessings and grace, which have enabled us to complete this final project as part of the practical session for the Real-Time Systems and Internet of Things course. This project represents a tangible implementation of the concepts and theories we have studied during the lectures and practical sessions.

This project is the result of the hard work and collaboration of our team, under the guidance of Laboratory Assistant Ahmad Rifqi Fadhlurrahman. Additionally, we have received direction and support from the course instructor, Mr. F. Astha Ekadiyanto, S.T., M.Sc., who has provided valuable insights and in-depth understanding of real-time system concepts and Internet of Things (IoT) technology.

Through this project, we aim to apply the knowledge we have acquired to develop an IoT-based solution for real-time patient health monitoring. We also hope that this project can serve as a small contribution toward advancing technological innovations that benefit society.

We acknowledge that this project still has many limitations. Therefore, we are open to suggestions and feedback for improvements in the future. Finally, we extend our heartfelt thanks to everyone who has helped and supported us in completing this project.

.

Depok, December 10, 2024

Group IP-19/P-19/R-19

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Real-time monitoring of patients' health conditions is an increasingly urgent need, especially in situations where patients require intensive attention or remote care. Traditional monitoring systems that rely on the direct supervision of medical personnel are often inefficient, time-consuming, and do not allow for continuous monitoring, especially in the case of patients who are in remote locations or have limited mobility. This can increase the risk of delays in detecting potentially dangerous changes in health conditions.

Furthermore, many modern health monitoring devices are expensive, difficult to access, and lack integration with IoT technologies that enable cloud-based surveillance. This creates a gap in providing an affordable, accessible, and efficient solution to monitor health parameters such as heart rate, oxygen saturation, body temperature, and environmental temperature and humidity of patients simultaneously.

Therefore, there is a need for an IoT-based health monitoring system that can not only provide real-time data but also integrate various patient health parameters into one easy-to-use platform, such as the Blynk app. With this solution, it is expected that the problems of limited access, high cost, and monitoring efficiency can be effectively overcome.

1.2 PROPOSED SOLUTION

To address the challenge of real-time patient health monitoring, an IoT-based health monitoring system is proposed as an innovative and flexible solution. The system uses an ESP32 module as the controlling centre to integrate various sensors, including DHT11 to measure room temperature and humidity, DS18B20 to monitor the patient's body temperature, and MAX30100 to detect heart rate (BPM) and oxygen saturation level (SpO2). Data from these sensors is transmitted in real-time to the Blynk IoT platform, allowing users to monitor health conditions through an app with a user-friendly interface.

One of the key features of the system is the flexibility in choosing the health parameters to monitor. Users can decide whether to monitor temperature or humidity from the

DHT11, as well as heart rate or SpO2 from the MAX30100, according to the patient's specific needs. In addition, the system is also equipped with an alert mechanism that will send a notification via the Blynk app if the temperature or heart rate exceeds a predetermined limit.

The system not only provides real-time data, but also facilitates local monitoring via serial output. With these capabilities, the system offers a reliable, flexible, and easy-to-implement health monitoring solution, which is expected to improve the efficiency of patient health surveillance, both in hospitals and at home.

1.3 ACCEPTANCE CRITERIA

The following are the acceptance criteria to ensure the successful implementation of the IoT-based Patient Health Monitoring System:

1. Sensor Data Collection:
    - The system can accurately read the temperature and humidity data from the DHT11 sensor.
    - The system can accurately read body temperature data from the DS18B20 sensor.
    - The system can accurately read heart rate (BPM) and oxygen saturation (SpO2) data from the MAX30100 sensor.
2. Real-Time Monitoring:
    - Data from all sensors can be displayed in real-time on the Blynk app.
    - Users can choose the type of data they want to monitor, such as temperature or humidity from DHT11 and heart rate or SpO2 from MAX30100.
3. Notifications and Alerts:
    - The system can send notifications or alerts through the Blynk app if the room temperature, body temperature, or heart rate exceeds a predetermined limit.
4. Hardware and Software Integration:
    - ESP32 is able to integrate data from all sensors without interruption.
    - The system can transmit data to the Blynk platform via WiFi connectivity with high stability.
5. Ease of Monitoring:
    - Data from sensors can be displayed via serial output for local monitoring.
    - Data on the Blynk app is easy to understand with a user-friendly interface.

6. System Stability:
    - The system must work continuously without failure.
    - The system can recover automatically after WiFi disconnection.
7. Energy Efficiency:
    - The system is able to run with minimal power consumption for long-term use.
8. Flexibility and Scalability:
    - The system allows customisation of temperature or heart rate thresholds as per user requirements.
    - The system can be expanded to add other sensors or features in the future without disrupting basic functionality.

1.4 ROLES AND RESPONSIBILITIES

The roles and responsibilities assigned to the group members are as follows:

Roles	Responsibilities	Person
Project Team Member	Develop the software and manages the integration and testing with the hardware	Ryan Safa Tjendana
Project Team Member		Drasseta Aliyyu Darmansyah
Project Manager	Oversees the execution of the project and manages the team.	Darren Adam Dewantoro
Project Team Member	Documenting the project	Kevin Ariono

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES

Tasks		Hardware Design completion	Software Development	Integration and Testing of Hardware and Software	Final Product Assembly and Testing
Week 1	27				
	28				
	29				
	30				
	1				
	2				
	3				
Week 2	4				
	5				
	6				
	7				
	8				
	9				
	10				

CHAPTER 2

IMPLEMENTATION

2.1 HARDWARE DESIGN AND SCHEMATIC

The hardware schematic describes the interconnections and components used in this project, focusing on the ESP32 module as the main controller. Below is a textual explanation to describe the setup:

- **Main Components**
  - ESP32 Microcontroller
    - Acts as the central processing unit, collecting data from various sensors and sending it to the Blynk IoT platform via Wi-Fi.
  - DHT11 (Room Temperature and Humidity Sensor)
    - Pin DATA: Connected to GPIO 2 on the ESP32.
    - Pin VCC: Connected to 3.3V from the ESP32.
    - Pin GND: Connected to GND on the ESP32.
  - DS18B20 (Body Temperature Sensor)
    - Pin DATA: Connected to GPIO 4 on the ESP32, with a 4.7k ohm pull-up resistor between the DATA pin and VCC.
    - Pin VCC: Connected to 3.3V from the ESP32.
    - Pin GND: Connected to GND on the ESP32.
  - MAX30100 (Heart Rate and SpO2 Sensor)
    - Pin SDA: Connected to GPIO 21 (I2C SDA) on the ESP32.
    - Pin SCL: Connected to GPIO 22 (I2C SCL) on the ESP32.
    - Pin VCC: Connected to 3.3V on the ESP32.
    - Pin GND: Connected to GND on the ESP32.
  - LED Indicator
    - One terminal connected to GPIO 15 for alarms or notifications.
    - The other terminal connected to GND, with a resistor in series to limit current.
  - Power Supply
    - Use a stable 3.3V supply for ESP32 and sensors. Ensure that the current rating is sufficient for all components.

- **Connections Overview**
    - The DHT11 monitors room conditions and feeds data to the ESP32 via GPIO **2**.
    - The DS18B20 provides body temperature data via GPIO **4**, with a pull-up resistor ensuring a stable One-Wire communication.
    - The MAX30100 communicates with the ESP32 using the I2C protocol, connecting through GPIOs **D21** (SDA) and **D22** (SCL).
    - The LED is used for real-time alerts when thresholds are exceeded, connecting through GPIO **15**.

The schematic:

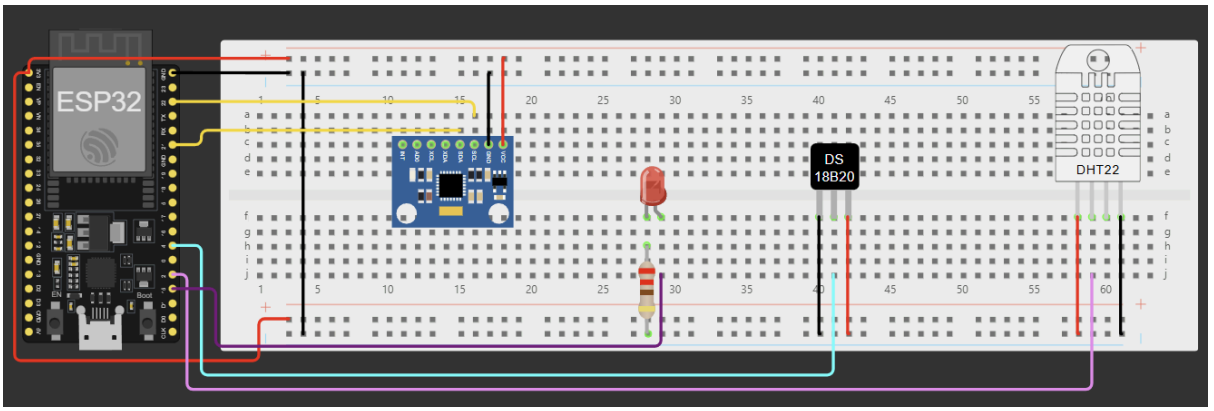

Fig 1. Hardware Schematic

2.2 SOFTWARE DEVELOPMENT

The software for the IoT-Based Patient Health Monitoring System is developed to ensure seamless interaction between the ESP32 microcontroller, sensors (DHT11, DS18B20, and MAX30100), and the Blynk IoT platform. This section outlines the development process, including the setup of required libraries, implementation of data transmission, and integration with the Blynk app for real-time monitoring.

1. **Setup and Required Libraries**

    The software is implemented using the Arduino IDE, which is a user-friendly platform for programming the ESP32. The following libraries are essential for the project:

- **`DHT` Library**: For handling the DHT11 temperature and humidity sensor.
- **`OneWire` and `DallasTemperature` Libraries**: For the DS18B20 waterproof temperature sensor.
- **`MAX30100_PulseOximeter` Library**: To read heart rate and SpO2 from the MAX30100 sensor.
- **`BlynkSimpleEsp32` and `WiFi` Libraries**: To enable cloud connectivity and interaction with the Blynk app.

These libraries can be installed via the Arduino IDE Library Manager.

2. **Implementing Data Transmissions**

The main software consists of several key components:

1. **Initialization**:
   - Sensors are initialized in the `setup()` function using their respective libraries.
   - LED is initialized as output for alerts.
   - Wi-Fi is configured to connect the ESP32 to a local network.
   - Run the `TaskBlynk()` task to start communicating with the Blynk application
   - Run the `TaskSensor()` task to start the reading from sensors.
2. **Sensor Data Acquisition**:
   - In the `TaskSensor()` task, data is periodically fetched from the sensors, including:
     - **Room Temperature and Humidity** from the DHT11.
     - **Body Temperature** from the DS18B20.
     - **Heart Rate (BPM)** and **SpO2** from the MAX30100.
3. **Data Transmission to Blynk**:
   - In the `TaskSensor()` task, after each data acquisition it is sent to the Blynk app via virtual pins using `Blynk.virtualWrite()`.
4. **Alerts**:

- Thresholds for temperature and heart rate can be set by the user from Bylnk to trigger alerts with LED.

3. **Integration with Blynk**

The Blynk app acts as the user interface for monitoring health parameters. Virtual pins are configured to display data in real-time:

- **V1**: Room Temperature
- **V2**: Room Humidity
- **V3**: Body Temperature
- **V4**: Heart Rate (BPM)
- **V5**: SpO2 Levels

These values are updated in the app and logged for historical analysis.

Below are snippets of the key parts of the implementation:

1. **Wi-Fi Setup**

```
#include <WiFi.h>

// Wi-Fi Credentials

const char* ssid = "GJ";

const char* password = "sumbawa8";


void setup() {

  WiFi.begin(ssid, password);

  Serial.println("Menghubungkan ke Wi-Fi...");

  while (WiFi.status() != WL_CONNECTED) {

    delay(1000);

    Serial.print(".");

  }

  Serial.println("\nTerhubung ke Wi-Fi!");

  Serial.print("Alamat IP: ");

  Serial.println(WiFi.localIP());
```

```
}
```

2. Sensor and Alert Initialization

```
#include <DHT.h>
#include <OneWire.h>
#include <DallasTemperature.h>

// Konfigurasi DHT11
#define DHTPIN 2
#define DHTTYPE DHT11
#define LED_PIN 15

// Inisialisasi DHT11 & DS18B20
DHT dht(DHTPIN, DHTTYPE);
#define ONE_WIRE_BUS 4
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// Variabel global
float temperatureThreshold = 20.0;
float dhtTemperature, dhtHumidity, temperatureC;

// Task untuk membaca sensor dan mengontrol LED
void TaskSensor(void *pvParameters) {
  while (true) {
    // Baca suhu dan kelembapan DHT11
    dhtTemperature = dht.readTemperature();
    dhtHumidity = dht.readHumidity();
```

```
    if (!isnan(dhtTemperature) && !isnan(dhtHumidity)) {
      Serial.printf("Suhu     DHT11(Task):     %.2f     °C\n",
dhtTemperature);
      Serial.printf("Kelembaban     DHT11:     %.2f     %%\n",
dhtHumidity);

    } else {
      Serial.println("Sensor DHT11 gagal dibaca!");
    }

    // Baca suhu DS18B20
    sensors.requestTemperatures();
    temperatureC = sensors.getTempCByIndex(0);

    if (temperatureC != DEVICE_DISCONNECTED_C) {
                Serial.printf("Suhu   DS18B20:   %.2f   °C\n",
temperatureC);

      if (temperatureC > temperatureThreshold) {
        digitalWrite(LED_PIN, HIGH);
        Serial.println("LED NYALA (Suhu tinggi)!");
      } else {
        digitalWrite(LED_PIN, LOW);
        Serial.println("LED MATI (Suhu normal)!");
      }
    }

     vTaskDelay(2000 / portTICK_PERIOD_MS);   // Delay baca
setiap 2 detik
```

```
  }
}


void setup() {
  dht.begin();
  sensors.begin();
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, LOW);
  xTaskCreatePinnedToCore(TaskSensor, "Sensor Task", 4096,
NULL, 1, NULL, 1);
}
```

3. Blynk Data Transmission

```
// Blynk Credentials
#define BLYNK_TEMPLATE_ID "TMPL6n86zxBll"
#define BLYNK_TEMPLATE_NAME "Proyek Akhir"
#define BLYNK_AUTH_TOKEN "2whbOcCAyDZ7ZyHsmjXalL9lcGamiZwg"
#include <BlynkSimpleEsp32.h>


// Callback untuk Blynk slider
BLYNK_WRITE(V4) {
  temperatureThreshold = param.asFloat();
  Serial.print("Threshold suhu diperbarui: ");
  Serial.println(temperatureThreshold);
}


// Task untuk mengelola Blynk
void TaskBlynk(void *pvParameters) {
  while (true) {
```

```
    Blynk.run();
    vTaskDelay(10 / portTICK_PERIOD_MS);  // Task delay agar tidak memblokir
  }
}

// Task untuk membaca sensor dan mengontrol LED
void TaskSensor(void *pvParameters) {
  while (true) {
    // Baca suhu dan kelembapan DHT11
    if (!isnan(dhtTemperature) && !isnan(dhtHumidity)) {
      Blynk.virtualWrite(V1, dhtTemperature);
      Blynk.virtualWrite(V2, dhtHumidity);
    }

    // Baca suhu DS18B20
    if (temperatureC != DEVICE_DISCONNECTED_C) {
      Blynk.virtualWrite(V3, temperatureC);
    }
     vTaskDelay(2000 / portTICK_PERIOD_MS);   // Delay baca setiap 2 detik
  }
void setup() {
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);
  xTaskCreatePinnedToCore(TaskBlynk, "Blynk Task", 4096, NULL, 1, NULL, 1);
}
```

2.3 HARDWARE AND SOFTWARE INTEGRATION

The hardware and software integration in the IoT-Based Patient Health Monitoring System ensures seamless communication and functionality between the sensors, the ESP32 microcontroller, and the Blynk IoT platform. This process merges the physical hardware components with the software framework to create a unified system capable of real-time health monitoring.

The hardware setup involves connecting the DHT11 sensor to measure room temperature and humidity, the DS18B20 sensor for precise body temperature readings, and the MAX30100 sensor for heart rate (BPM) and SpO2 measurements. These sensors are interfaced with the ESP32 microcontroller through designated GPIO pins, enabling the collection of health data.

On the software side, the ESP32 is programmed using Arduino libraries to initialize sensors, process data, and communicate with the Blynk IoT platform over Wi-Fi. Key libraries like DHT, DallasTemperature, and MAX30100_PulseOximeter are used to handle sensor-specific data acquisition, while the Blynk library facilitates data transmission and visualization. The ESP32 processes raw sensor data into readable metrics and sends these values to the Blynk app via virtual pins for real-time display.

To ensure reliable performance, the system is tested for data accuracy, communication stability, and responsiveness to alerts when thresholds are exceeded. Integration challenges, such as synchronization issues and network connectivity, are addressed through optimized coding practices and error-handling mechanisms.

By integrating hardware and software effectively, this system achieves real-time, accurate monitoring of health parameters, offering a scalable solution for remote patient care.

CHAPTER 3

TESTING AND EVALUATION

3.1 TESTING

Testing is an essential stage in the development of the IoT-Based Patient Health Monitoring System. This phase ensures that all components, both hardware and software, function reliably and meet the project's objectives. The testing process focuses on verifying the accuracy of sensor readings, evaluating system communication, and ensuring overall stability during operation.

1. **Sensor Functionality Testing**

   Each sensor was tested individually to confirm its accuracy and reliability:

   - **DHT11 Sensor**: The room temperature and humidity readings were cross-checked with a standard thermometer and hygrometer to validate measurement accuracy.
   - **DS18B20 Sensor**: This sensor was submerged in water at known temperatures to assess its precision in detecting body temperature.
   - **MAX30100 Sensor**: Heart rate (BPM) and SpO2 measurements were compared with a commercial pulse oximeter to ensure correctness.

2. **Communication and Data Transmission**

   The communication between the ESP32 microcontroller and the Blynk IoT platform was thoroughly evaluated:

   - **Wi-Fi Connection Stability**: The system's ability to maintain a stable connection under both optimal and degraded network conditions was assessed.
   - **Data Integrity**: Sensor readings transmitted to the Blynk platform were compared with serial monitor outputs to verify consistency.

3. **Alert Functionality**

Threshold limits were set for parameters like body temperature and heart rate to test the alert mechanism.

- Alerts were triggered when values exceeded these limits and were displayed in the Blynk app.
- This functionality was tested repeatedly to ensure reliable notifications.

4. **Environmental Testing**

The system was exposed to varying environmental conditions to test adaptability:

- The DHT11 sensor's responsiveness to changes in room temperature and humidity was monitored.
- The MAX30100 and DS18B20 sensors were tested under diverse light and ambient conditions.

3.2 RESULT

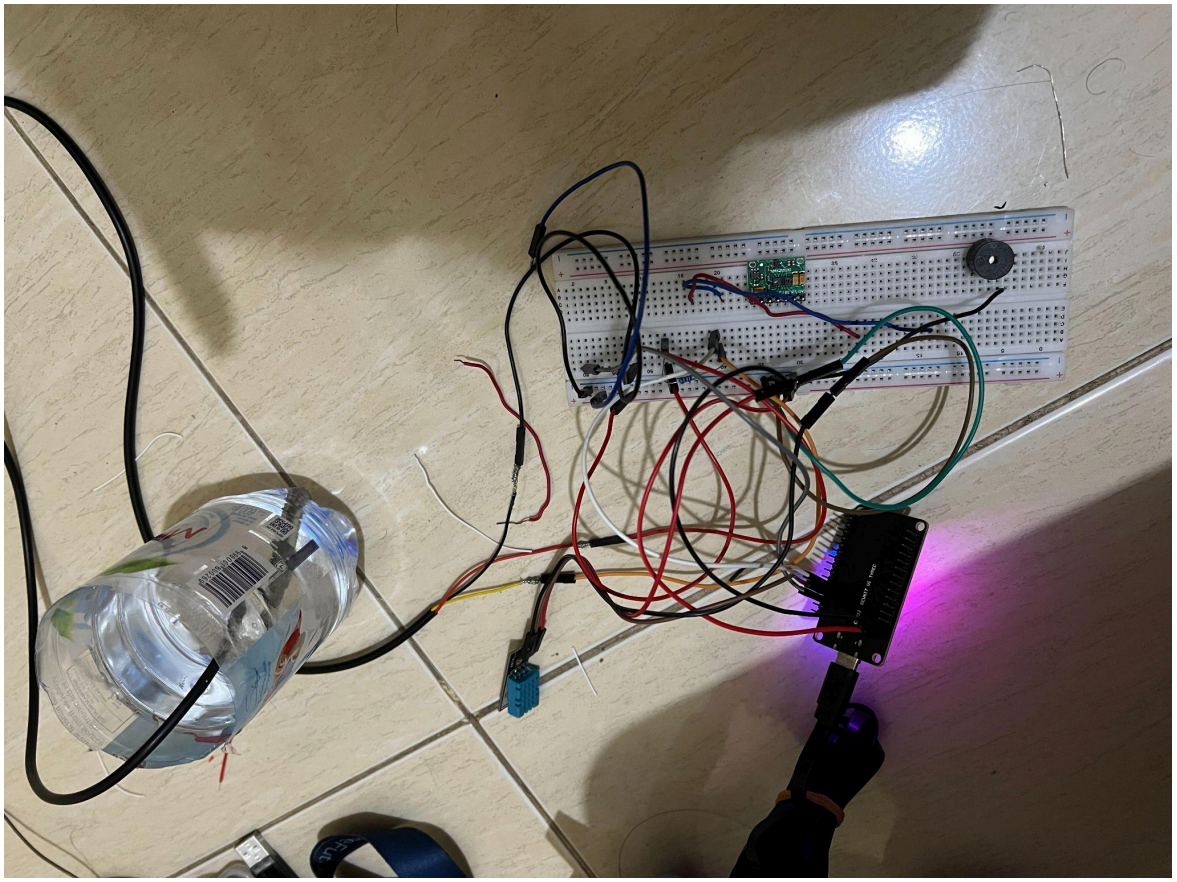

Fig 4. Testing Result

The results of the IoT-Based Patient Health Monitoring System testing demonstrate that the system effectively fulfills its objectives of providing real-time health monitoring. Each component performed as expected, and the integration of hardware and software was seamless, ensuring reliable data acquisition, processing, and visualization.

The DHT11 sensor successfully measured room temperature and humidity, with readings that aligned closely with standard reference devices used for comparison. This confirms its suitability for monitoring the patient's surrounding environment. Similarly, the DS18B20 sensor accurately captured body temperature data, providing consistent results with minimal deviation from reference thermometers. The MAX30100 sensor reliably measured heart rate (BPM) and blood oxygen saturation (SpO2) levels, with results comparable to those of a commercial-grade pulse oximeter.

Data collected from all sensors were transmitted to the Blynk IoT platform in real-time using the ESP32 microcontroller. The Blynk app displayed this data clearly, allowing for easy interpretation by users. Furthermore, the system's alert mechanism functioned as intended, generating notifications promptly when any parameter exceeded the predefined thresholds, ensuring that users could respond quickly to potential health risks.

3.3 EVALUATION

The IoT-Based Patient Health Monitoring System was thoroughly evaluated to assess its effectiveness, reliability, and potential for real-world application. The evaluation focused on key aspects such as system accuracy, usability, and areas for improvement.

The system demonstrated high accuracy during testing, with all sensors providing reliable data that closely matched reference instruments. This confirms that the hardware components and their integration were suitable for the intended monitoring tasks. The ESP32 microcontroller efficiently handled data acquisition and transmission, ensuring seamless communication with the Blynk IoT platform. The real-time visualization in the Blynk app was user-friendly, making the system accessible even to users with minimal technical expertise. Alerts were triggered promptly when thresholds were exceeded, showcasing the system's potential to assist in proactive health management.

Despite these strengths, a few limitations were identified. The system's dependence on a stable internet connection limits its usability in areas with poor connectivity. Additionally,

while the sensors functioned effectively during testing, their long-term performance under varying environmental conditions remains to be evaluated. The current design is tailored for single-patient monitoring; future iterations could benefit from enhancements that allow for multi-patient support, improving its scalability for broader healthcare applications.

Overall, the evaluation highlights the system as a functional and practical tool for real-time health monitoring. While some areas need refinement, the project successfully demonstrates how IoT technology can address critical healthcare challenges, offering a strong foundation for further development and innovation.

CHAPTER 4

CONCLUSION

This project successfully delivered a functional IoT-Based Patient Health Monitoring System, addressing the need for efficient, real-time monitoring of vital health parameters. By utilizing sensors like the DHT11 for room temperature and humidity, the DS18B20 for body temperature, and the MAX30100 for heart rate and SpO2, the system provides comprehensive health data to users through a single, integrated platform. The ESP32 microcontroller efficiently managed sensor data and transmitted it to the Blynk IoT platform, where users could monitor patient conditions with ease and clarity.

The system met its objectives, performing reliably during testing and effectively alerting users when any parameter exceeded the predefined thresholds. This ensures timely awareness of potential health risks, demonstrating the practicality and relevance of IoT solutions in healthcare.

Despite its success, the project revealed areas for improvement. The reliance on a stable internet connection limits its usability in remote or underserved regions, and the sensors' long-term durability under variable conditions requires further evaluation. Future iterations could enhance scalability to support monitoring for multiple patients, extending the system's utility for broader healthcare applications.

In conclusion, this project showcases the potential of IoT technology in addressing healthcare challenges. It provides an affordable, accessible, and effective solution for real-time health monitoring, serving as a foundation for further innovation. With continued development, this system could significantly contribute to improving patient care and remote health management.

REFERENCES

[1] Microcontrollers Lab "Control ESP32 Outputs with Blynk App and Arduino IDE" [Online] Available: https://microcontrollerslab.com/control-esp32-outputs-blynk-app-arduino-ide Accessed: 10 December 2024

[2] Ardutech "Kontrol Lampu WiFi dengan Blynk dan Arduino" [Online] Available: https://www.ardutech.com/kontrol-lampu-wifi-dengan-blynk-dan-arduino/ Accessed: 10 December 2024

[3] Arduino Docs "WiFi Library for Arduino" [Online] Available: https://docs.arduino.cc/libraries/wifi/ Accessed: 10 December 2024

[4] Arduino Project Hub "IoT Project for Controlling Devices" [Online] Available: https://projecthub.arduino.cc/arcaegecengiz/12f621d5-055f-41fe-965d-a596fcc594f6 Accessed: 10 December 2024

[5] Microcontrollers Lab "MAX30100 Pulse Oximeter Heart Rate Sensor Arduino Tutorial" [Online] Available: https://microcontrollerslab.com/max30100-pulse-oximeter-heart-rate-sensor-arduino-tutorial/ Accessed: 10 December 2024

[6] Maker Guides "DS18B20 Arduino Temperature Sensor Tutorial" [Online] Available: https://www.makerguides.com/ds18b20-arduino-tutorial/ Accessed: 10 December 2024

[7] FreeRTOS Documentation "Implementing a Task in FreeRTOS" [Online] Available: https://www.freertos.org/Documentation/02-Kernel/02-Kernel-features/01-Tasks-and-co-routines/05-Implementing-a-task Accessed: 10 December 2024

APPENDICES

Appendix A: Project Schematic

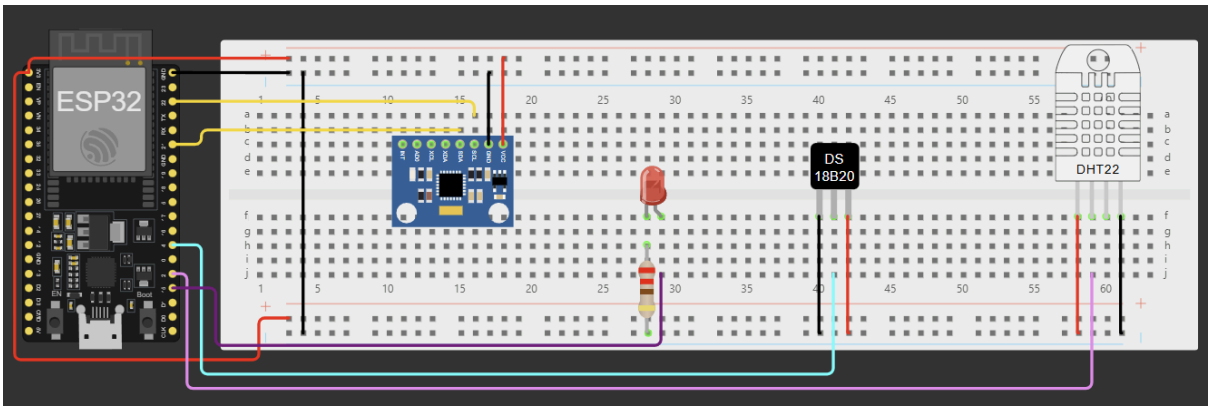

Fig 1. Hardware Schematic

Appendix B: Documentation

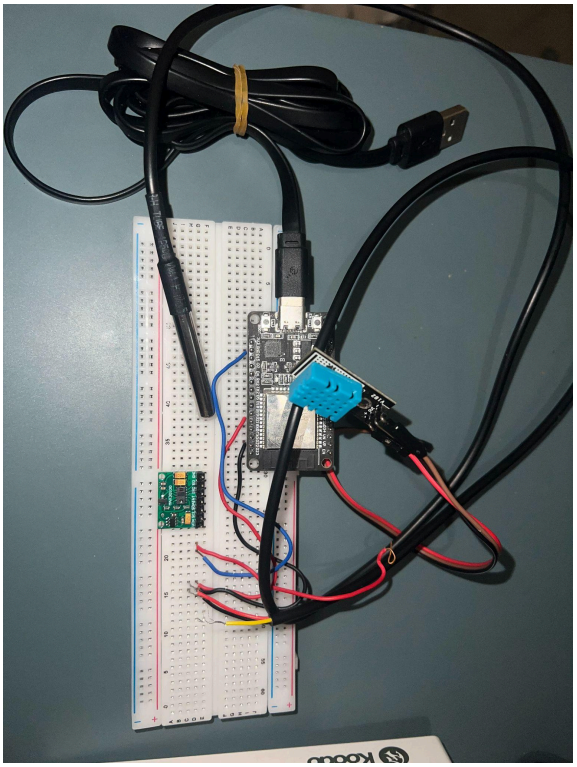

Fig 2. Testing Pertama

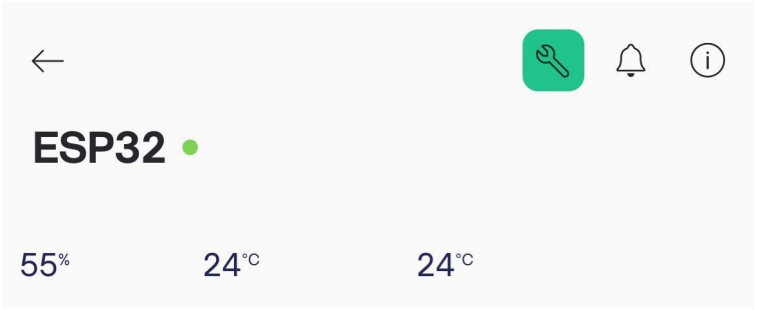

Fig 3. Blynk View Pertama

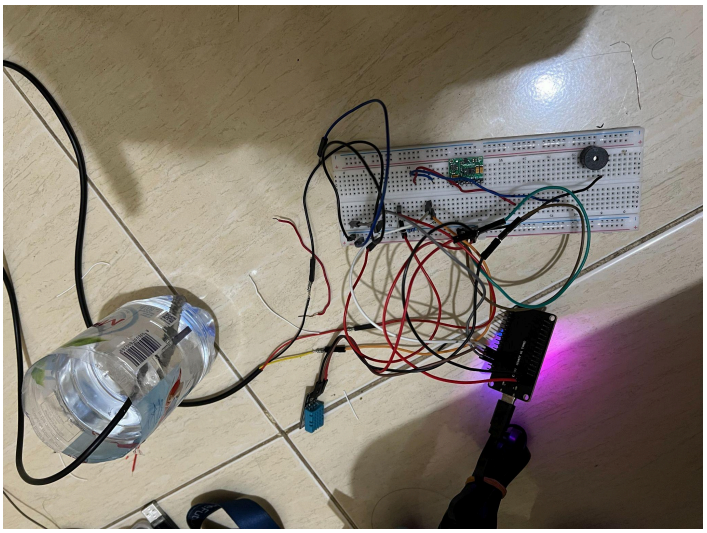

Fig 4. Testing Kedua

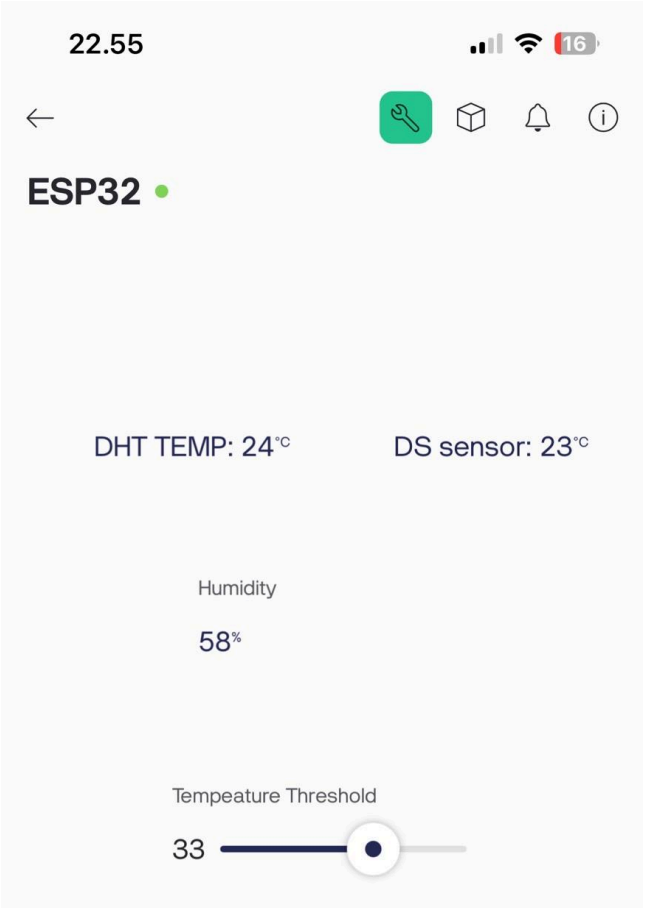

Fig 5. Blynk View Kedua