Project Title

IoT Based Patient Health Monitoring

Ishita Roy - 21E51A0543

Farooq Hassian - 21E51A6250

Hyderabad Institute of Technology & Management

Date of Submission: 15th February, 2024

Abstract:

This project aims to develop a robust system for monitoring the health parameters of patients with the help of Internet of Things (IoT) technology, which is rapidly evolving in today’s world. There is a need to enhance healthcare services as traditional healthcare monitoring often faces challenges in providing real-time data on a patient's vitals and health.

With the help of IoT, we aim to create a system that provides remote and up-to-date monitoring of patients, and a user-friendly interface for data visualization. This will help in tracking the patient’s overall health.

The methodology involves the usage of various sensors such as the Pulse Oximeter Sensor, DS18B20 Sensor and DHT11 Sensor into a centralized IoT platform. The patient’s vitals readings that will be collected as data will be stored into platforms such as Thingspeak for finding trends and abnormalities in the patient’s data, and analyzing the overall health of the patient.

Key findings from the project include the ability of the system to provide real-time readings of the health status of the patient, that can ensure timely care through continuous monitoring and data-driven information. This would also lead to enhanced and better quality of healthcare delivery. Potential future improvements could include the usage of additional sensors for detailed patient’s health readings, and scalability.

List of Figures and Tables:

Introduction:

In today's scenario, modern healthcare and technology go hand-in-hand. With the help of Internet of Things (IoT) technology, healthcare experts are able to monitor patients in real-time, which has been very beneficial.

The objective of this building a patient health monitoring system is to enable continuous and remote monitoring of vital health parameters of patients. Using sensors and data visualization platforms, this system can provide healthcare professionals with timely insights into patient health status, facilitating early detection of abnormal vitals of the patient.

Our goal includes developing an architecture and implementing an algorithm that can easily detect irregularity or deviation in the patient's vital readings, and take precautionary steps if necessary. It also provides a user-friendly and easy-to-understand interface to visualize the data efficiently.

Literature Review:

IoT-based patient health monitoring systems make use of several technologies in order to collect and analyze patient data. Using modern sensor technologies, wireless communication and sensor miniaturization have provided advances in technologies such as the development of wearable devices that monitor heart rate, blood pressure, and temperature in real-time efficiently. Apart from this, integrating cloud computing technology also facilitates data storage, processing, and analysis.

There are various communication protocols used in IoT-based healthcare systems. The most commonly employed protocols are Wi-Fi, Zigbee, MQTT (Message Queuing Telemetry Transport) and Bluetooth Low Energy (BLE) among others. Wi-Fi enables high-speed data transfer and Zigbee is ideal for home healthcare monitoring systems. MQTT ensures efficient data transmission and BLE is widely used in wearable devices, making it suitable for applications requiring real-time communication.

Standardization is essential for any system to provide reliability, security and interoperability. The Institute of Electrical and Electronics Engineers (IEEE) and the International Organization for Standardization (ISO) have developed specific standards for healthcare IoT applications.

Motivation of the Project:

The project's motivation lies in the pursuit of a more efficient and accessible healthcare approach that prioritizes patient well-being and ensures a higher quality of care through continuous monitoring and data-driven information.

In today's scenario, it is essential that we build and implement modern technologies to meet the rising healthcare needs of the world, including the increasing prevalence of chronic diseases and rising healthcare costs. Having a remote health monitoring system would be beneficial for individuals to detect health issues and enable timely intervention to prevent complications.

With the help of a health monitoring system that provides real-time insights, people will be aware of their health status, which will lead to improved and healthier lifestyles. A remote monitoring system will also reduce the need for frequent hospital visits, making healthcare more accessible.

These factors have motivated us to address rising healthcare demands and shift towards preventive healthcare. It also seeks to bridge healthcare disparities and transform healthcare delivery through real-time and remote health monitoring systems.

List of Used Components:

SNo	Name of the Components
1	ESP32 Board
2	MAX30100 Pulse Oximeter Sensor
3	DS18B20 Sensor
4	DHT11 Sensor
5	Resistor 4.7K
6	Connecting Wires
7	Breadboard

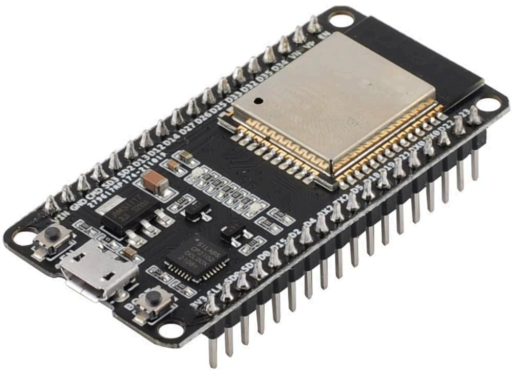


Fig 1.1

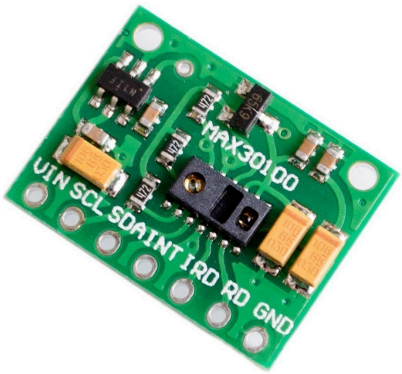


Fig 1.2

Fig 1.3

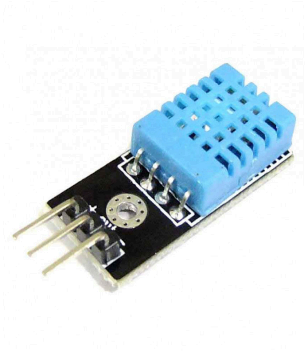


Fig 1.4

Fig 1.5

Fig 1.6

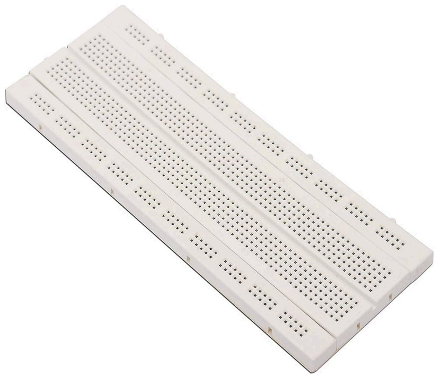


Fig 1.7

Methodology:

The **approach** and **implementation** for IoT-based Patient Monitoring System involves:

1. Setup the hardware by connecting all the components including the sensors and the breadboard together.
2. Assemble each sensor according to the given specifications and attach it to the ESP32 board, and use relevant libraries to interact with every sensor and gather data precisely.
3. Use Arduino IDE to write code to collect sensor data, and send it via Wi-Fi to the ThingSpeak cloud platform.
4. Gather sensor data continuously to spot abnormalities, and extract insightful information from the gathered data using ThingSpeak's data visualization feature.

The **hardware** and **software specifications** include:

1. ESP32 Board: A microcontroller that has wireless communication capabilities integrated into it.
2. MAX30100 Pulse Oximeter sensor: It is used to measure blood oxygen levels and pulse rate.
3. DS18B20 Temperature sensor: Provides accurate digital temperature readings.
4. DHT11 Humidity sensor: It is used to measure the environment's relative humidity levels.
5. 4.7K resistor: Used with the DS18B20 sensor to condition signals.
6. Using a breadboard and connecting wires makes it easier to make physical connections between parts for testing and prototyping.

7. The ThingSpeak Platform acts as the main source for sensor data. It gives customers access to channels for data visualization and storage, enabling them to track and examine both historical and real-time data.

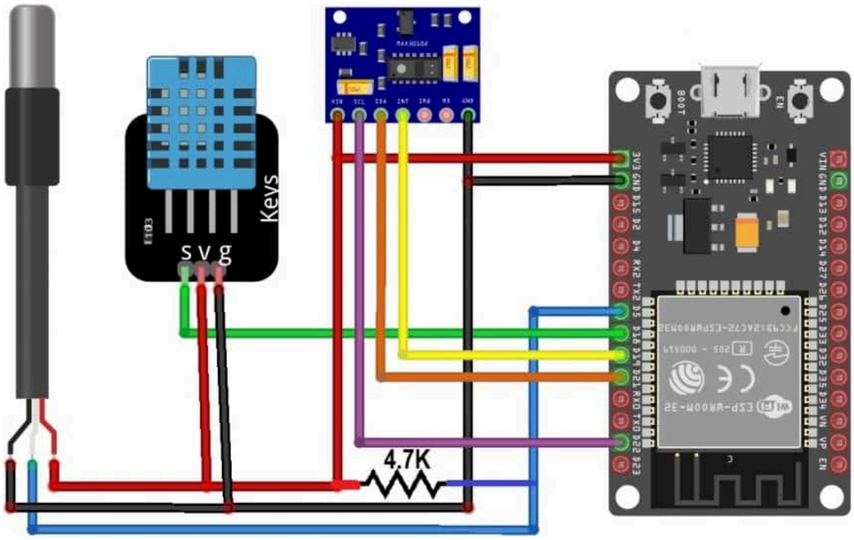


Fig 1.8

Implementation:

The hardware setup involves connecting the parts together. After that, each sensor is wired correctly for data communication and connected to the ESP32 board. Using the Arduino IDE, libraries are integrated to interface with sensors and send data wirelessly to the ThingSpeak platform, all while providing error handling to ensure data integrity. Setting up ThingSpeak channels, acquiring API keys for secure connection, preparing the ESP32 board for Wi-Fi connectivity, and putting data logging into place to monitor sensor readings are all part of cloud integration. With the help of ThingSpeak's tools, data visualization professionals may design unique charts and graphs. They can also set up dashboards to monitor data in real-time and the past to monitor abnormal conditions and inform healthcare practitioners.

The **algorithm** for the IoT-based Patient Monitoring System is as follows:

1. Include necessary libraries such as WiFi, WebServer, Wire, and sensor libraries. Define pins and constants for sensors and communication protocols.

2. Set up Wi-Fi credentials (SSID and password) for connecting to the local network. Attempt connection to the Wi-Fi network and wait until successfully connected. Print Wi-Fi connection status and assigned IP address upon successful connection.

3. Initialize the sensors and a WebServer object for handling HTTP requests.

4. Continuously handle incoming client requests and sensor updates within the loop. Update sensor readings (temperature, humidity, heart rate, blood oxygen level) periodically.

5. Utilize ThingSpeak's visualization tools to create custom charts and graphs for representing sensor data. Configure dashboard layouts to display

real-time and historical data trends, allowing users to monitor patient health parameters at a glance.

6. Define functions to handle HTTP requests, including the main page ("/") and 404 errors (Not Found). When a client requests the main page, send HTML content containing sensor readings (temperature, humidity, heart rate, blood oxygen level, and body temperature). When a client requests a non-existent page, send a 404 error response indicating that the requested resource was not found.

Arduino Code:

```
#include <WiFi.h>

#include <WebServer.h>

#include <Wire.h>

#include "MAX30100_PulseOximeter.h"

#include <OneWire.h>

#include <DallasTemperature.h>

#include <dht.h>

#include "ThingSpeak.h"

#define DHT11_PIN 4

#define DS18B20 5

#define REPORTING_PERIOD_MS     1000

float temperature, humidity, BPM, SpO2, bodytemperature;

/SSID & Password/
```

```
const char* ssid = "HITAM_G09";  // Enter SSID here

const char* password = "hitam@2023";  //Enter Password here

unsigned long myChannelNumber =2431378;

const char * myWriteAPIKey = "5ZJVNQ5SEZUTXO44";

WiFiClient  client;

dht DHT;

PulseOximeter pox;

uint32_t tsLastReport = 0;

OneWire oneWire(DS18B20);

DallasTemperature sensors(&oneWire);

WebServer server(80);


void onBeatDetected()

{

  Serial.println("Beat!");

}


void setup() {

  Serial.begin(115200);

  pinMode(19, OUTPUT);

  delay(100);
```

```
Serial.println("Connecting to ");
Serial.println(ssid);

//connect to your local wi-fi network
WiFi.begin(ssid, password);
ThingSpeak.begin(client);

//check wi-fi is connected to wi-fi network
while (WiFi.status() != WL_CONNECTED) {
delay(1000);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected..!");
Serial.print("Got IP: ");  Serial.println(WiFi.localIP());
server.on("/", handle_OnConnect);
server.onNotFound(handle_NotFound);
server.begin();
Serial.println("HTTP server started");
Serial.print("Initializing pulse oximeter..");
```

```
  if (!pox.begin()) {
    Serial.println("FAILED");
    for (;;);
  } else {
    Serial.println("SUCCESS");
    pox.setOnBeatDetectedCallback(onBeatDetected);
  }
   pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
  // Register a callback for the beat detection
}
void loop() {
  server.handleClient();
  pox.update();
  sensors.requestTemperatures();
  int chk = DHT.read11(DHT11_PIN);
  temperature = DHT.temperature;
  humidity = DHT.humidity;
  BPM = pox.getHeartRate();
  SpO2 = pox.getSpO2();
  bodytemperature = sensors.getTempCByIndex(0);
```

```
sendToThingSpeak(temperature, humidity, BPM, SpO2, bodytemperature);
if (millis() - tsLastReport > REPORTING_PERIOD_MS)
{
  Serial.print("Room Temperature: ");
  Serial.print(DHT.temperature);
  Serial.println("°C");
  Serial.print("Room Humidity: ");
  Serial.print(DHT.humidity);
  Serial.println("%");
  Serial.print("BPM: ");
  Serial.println(BPM);
  Serial.print("SpO2: ");
  Serial.print(SpO2);
  Serial.println("%");
  Serial.print("Body Temperature: ");
  Serial.print(bodytemperature);
  Serial.println("°C");
  Serial.println("***********");
  Serial.println();
  tsLastReport = millis();
}
```

```
}

void handle_OnConnect() {

  server.send(200, "text/html", SendHTML(temperature, humidity, BPM, SpO2,
bodytemperature));

}

void handle_NotFound(){

  server.send(404, "text/plain", "Not found");

}

void sendToThingSpeak(float temperature, float humidity, float BPM, float
SpO2, float bodytemperature) {

  ThingSpeak.writeField(myChannelNumber, 1, temperature, myWriteAPIKey);

  ThingSpeak.writeField(myChannelNumber, 2, humidity, myWriteAPIKey);

  ThingSpeak.writeField(myChannelNumber, 3, BPM, myWriteAPIKey);

  ThingSpeak.writeField(myChannelNumber, 4, SpO2, myWriteAPIKey);

ThingSpeak.writeField(myChannelNumber, 5, bodytemperature,
myWriteAPIKey);

}

  String SendHTML(float temperature,float humidity,float BPM,float SpO2,
float bodytemperature){

  String ptr = "<!DOCTYPE html>";

  ptr +="<html>";

  ptr +="<head>";
```

```
  ptr +="<title>ESP32 Patient Health Monitoring</title>";

ptr +="<meta name='viewport' content='width=device-width,
initial-scale=1.0'>";

  ptr +="<link
href='https://fonts.googleapis.com/css?family=Open+Sans:300,400,600'
rel='stylesheet'>";

  ptr +="<style>";

  ptr +="html { font-family: 'Open Sans', sans-serif; display: block; margin: 0px
auto; text-align: center;color: #444444;}";

  ptr +="body{margin: 0px;} ";

  ptr +="h1 {margin: 50px auto 30px;} ";

  ptr +=".side-by-side{display: table-cell;vertical-align: middle;position:
relative;}";

  ptr +=".text{font-weight: 600;font-size: 19px;width: 200px;}";

  ptr +=".reading{font-weight: 300;font-size: 50px;padding-right: 25px;}";

  ptr +=".temperature .reading{color: #F29C1F;}";

  ptr +=".humidity .reading{color: #3B97D3;}";

  ptr +=".BPM .reading{color: #FF0000;}";

  ptr +=".SpO2 .reading{color: #955BA5;}";

  ptr +=".bodytemperature .reading{color: #F29C1F;}";

  ptr +=".superscript{font-size: 17px;font-weight: 600;position: absolute;top:
10px;}";

  ptr +=".data{padding: 10px;}";
```

```
ptr +=".container{display: table;margin: 0 auto;}";

ptr +=".icon{width:65px}";

ptr +="</style>";

ptr +="</head>";

ptr +="<body>";

ptr +="<h1>ESP32 Patient Health Monitoring</h1>";

ptr +="<h3>Ishita Farooq HITAM</h3>";

ptr +="<div class='container'>";


ptr +="<div class='data temperature'>";

ptr +="<div class='side-by-side icon'>";

ptr +="<svg enable-background='new 0 0 19.438 54.003'height=54.003px
id=Layer_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px
xml:space=preserve xmlns=http://www.w3.org/2000/svg
xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path
d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.96
5,0,3.252,2.715,3.252,6.063v30.982";

ptr
+="C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9
.719-4.351,9.719-9.718";

ptr
+="c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084
v-2h4.084V8.82H11.976z M15.302,44.833";
```

```
  ptr
+="c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.
236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

  ptr
+="s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.83
3z'fill=#F29C21 /></g></svg>";

  ptr +="</div>";

  ptr +="<div class='side-by-side text'>Room Temperature</div>";

  ptr +="<div class='side-by-side reading'>";

  ptr +=(int)temperature;

  ptr +="<span class='superscript'>&deg;C</span></div>";

  ptr +="</div>";



  ptr +="<div class='data humidity'>";

  ptr +="<div class='side-by-side icon'>";

  ptr +="<svg enable-background='new 0 0 29.235 40.64'height=40.64px
id=Layer_1 version=1.1 viewBox='0 0 29.235 40.64'width=29.235px x=0px
xml:space=preserve xmlns=http://www.w3.org/2000/svg
xmlns:xlink=http://www.w3.org/1999/xlink y=0px><path
d='M14.618,0C14.618,0,0,17.95,0,26.022C0,34.096,6.544,40.64,14.618,40.64s1
4.617-6.544,14.617-14.617";

  ptr +="C29.235,17.95,14.618,0,14.618,0z
M13.667,37.135c-5.604,0-10.162-4.56-10.162-10.162c0-0.787,0.638-1.426,1.426-
1.426";
```

```
  ptr
+="c0.787,0,1.425,0.639,1.425,1.426c0,4.031,3.28,7.312,7.311,7.312c0.787,0,1.425
,0.638,1.425,1.425";

  ptr +="C15.093,36.497,14.455,37.135,13.667,37.135z'fill=#3C97D3 /></svg>";

  ptr +="</div>";

  ptr +="<div class='side-by-side text'>Room Humidity</div>";

  ptr +="<div class='side-by-side reading'>";

  ptr +=(int)humidity;

  ptr +="<span class='superscript'>%</span></div>";

  ptr +="</div>";


  ptr +="<div class='data Heart Rate'>";

  ptr +="<div class='side-by-side icon'>";

  ptr +="<svg enable-background='new 0 0 40.542 40.541'height=40.541px
id=Layer_1 version=1.1 viewBox='0 0 40.542 40.541'width=40.542px x=0px
xml:space=preserve xmlns=http://www.w3.org/2000/svg
xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path
d='M34.313,20.271c0-0.552,0.447-1,1-1h5.178c-0.236-4.841-2.163-9.228-5.214-
12.593l-3.425,3.424";

  ptr
+="c-0.195,0.195-0.451,0.293-0.707,0.293s-0.512-0.098-0.707-0.293c-0.391-0.
391-0.391-1.023,0-1.414l3.425-3.424";
```

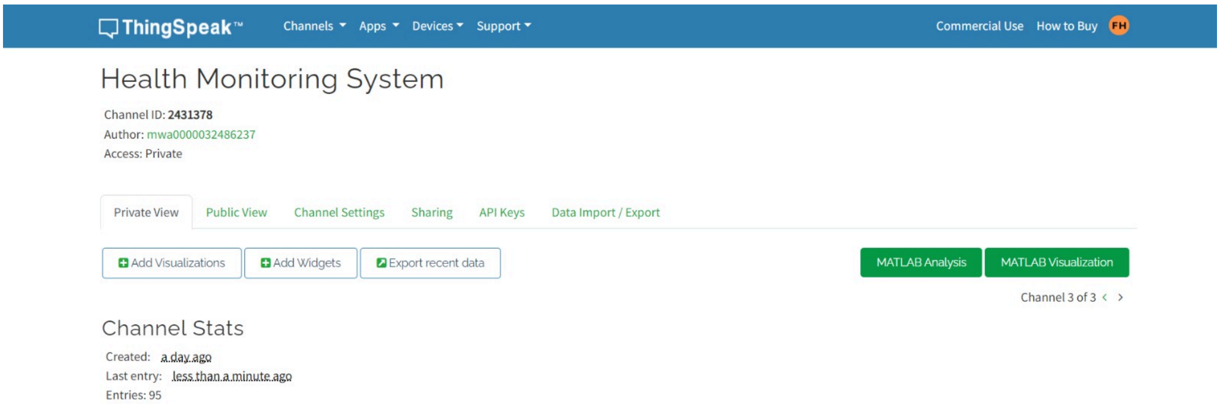
```
  ptr
+="c-3.375-3.059-7.776-4.987-12.634-5.215c0.015,0.067,0.041,0.13,0.041,0.20
2v4.687c0,0.552-0.447,1-1,1s-1-0.448-1-1V0.25";

  ptr
+="c0-0.071,0.026-0.134,0.041-0.202C14.39,0.279,9.936,2.256,6.544,5.385l3.
576,3.577c0.391,0.391,0.391,1.024,0,1.414";

  ptr
+="c-0.195,0.195-0.451,0.293-0.707,0.293s-0.512-0.098-0.707-0.293L5.142,6.8
12c-2.98,3.348-4.858,7.682-5.092,12.459h4.804";

  ptr
+="c0.552,0,1,0.448,1,1s-0.448,1-1,1H0.05c0.525,10.728,9.362,19.271,20.22,19.2
71c10.857,0,19.696-8.543,20.22-19.271h-5.178";

  ptr +="C34.76,21.271,34.313,20.823,34.313,20.271z
M23.084,22.037c-0.559,1.561-2.274,2.372-3.833,1.814";

  ptr
+="c-1.561-0.557-2.373-2.272-1.815-3.833c0.372-1.041,1.263-1.737,2.277-1.928
L25.2,7.202L22.497,19.05";

  ptr +="C23.196,19.843,23.464,20.973,23.084,22.037z'fill=#26B999
/></g></svg>";

  ptr +="</div>";

  ptr +="<div class='side-by-side text'>Heart Rate</div>";

  ptr +="<div class='side-by-side reading'>";

  ptr +=(int)BPM;

  ptr +="<span class='superscript'>BPM</span></div>";
```

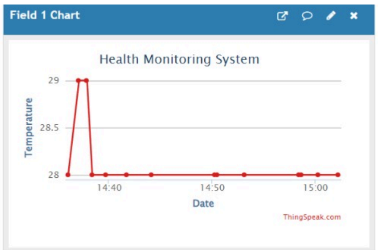
```
ptr +="</div>";


ptr +="<div class='data Blood Oxygen'>";

ptr +="<div class='side-by-side icon'>";

ptr +="<svg enable-background='new 0 0 58.422 40.639'height=40.639px
id=Layer_1 version=1.1 viewBox='0 0 58.422 40.639'width=58.422px x=0px
xml:space=preserve xmlns=http://www.w3.org/2000/svg
xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path
d='M58.203,37.754l0.007-0.004L42.09,9.935l-0.001,0.001c-0.356-0.543-0.9
69-0.902-1.667-0.902";

ptr
+="c-0.655,0-1.231,0.32-1.595,0.808l-0.011-0.007l-0.039,0.067c-0.021,0.03-0.
035,0.063-0.054,0.094L22.78,37.692l0.008,0.004";

ptr
+="c-0.149,0.28-0.242,0.594-0.242,0.934c0,1.102,0.894,1.995,1.994,1.995v0.01
5h31.888c1.101,0,1.994-0.893,1.994-1.994";

ptr +="C58.422,38.323,58.339,38.024,58.203,37.754z'fill=#955BA5 /><path
d='M19.704,38.674l-0.013-0.004l13.544-23.522L25.13,1.156l-0.002,0.001C24.6
71,0.459,23.885,0,22.985,0";

ptr
+="c-0.84,0-1.582,0.41-2.051,1.038l-0.016-0.01L20.87,1.114c-0.025,0.039-0.046,
0.082-0.068,0.124L0.299,36.851l0.013,0.004";

ptr
+="C0.117,37.215,0,37.62,0,38.059c0,1.412,1.147,2.565,2.565,2.565v0.015h16.9
89c-0.091-0.256-0.149-0.526-0.149-0.813";
```

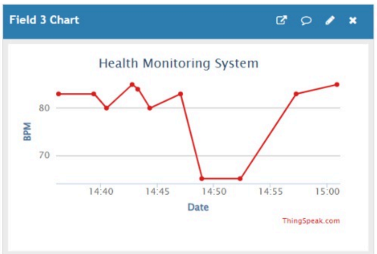
```
  ptr +="C19.405,39.407,19.518,39.019,19.704,38.674z'fill=#955BA5
/></g></svg>";

  ptr +="</div>";

  ptr +="<div class='side-by-side text'>Blood Oxygen</div>";

  ptr +="<div class='side-by-side reading'>";

  ptr +=(int)SpO2;

  ptr +="<span class='superscript'>%</span></div>";

  ptr +="</div>";


  ptr +="<div class='data Body Temperature'>";

  ptr +="<div class='side-by-side icon'>";

  ptr +="<svg enable-background='new 0 0 19.438 54.003'height=54.003px
id=Layer_1 version=1.1 viewBox='0 0 19.438 54.003'width=19.438px x=0px
xml:space=preserve xmlns=http://www.w3.org/2000/svg
xmlns:xlink=http://www.w3.org/1999/xlink y=0px><g><path
d='M11.976,8.82v-2h4.084V6.063C16.06,2.715,13.345,0,9.996,0H9.313C5.96
5,0,3.252,2.715,3.252,6.063v30.982";

  ptr
+="C1.261,38.825,0,41.403,0,44.286c0,5.367,4.351,9.718,9.719,9.718c5.368,0,9
.719-4.351,9.719-9.718";

  ptr
+="c0-2.943-1.312-5.574-3.378-7.355V18.436h-3.914v-2h3.914v-2.808h-4.084
v-2h4.084V8.82H11.976z M15.302,44.833";
```

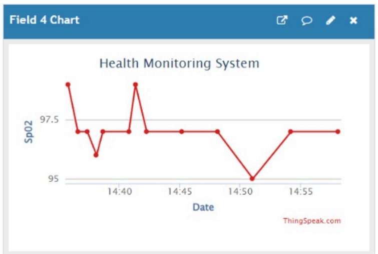
```
  ptr
+="c0,3.083-2.5,5.583-5.583,5.583s-5.583-2.5-5.583-5.583c0-2.279,1.368-4.
236,3.326-5.104V24.257C7.462,23.01,8.472,22,9.719,22";

  ptr
+="s2.257,1.01,2.257,2.257V39.73C13.934,40.597,15.302,42.554,15.302,44.83
3z'fill=#F29C21 /></g></svg>";

  ptr +="</div>";

  ptr +="<div class='side-by-side text'>Body Temperature</div>";

  ptr +="<div class='side-by-side reading'>";

  ptr +=(int)bodytemperature;

  ptr +="<span class='superscript'>&deg;C</span></div>";

  ptr +="</div>";


  ptr +="</div>";

  ptr +="</body>";

  ptr +="</html>";

  return ptr;

}
```

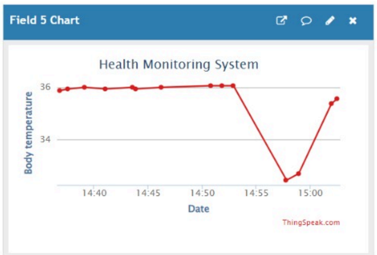
Demonstration of Live Data:











Flow Diagram:

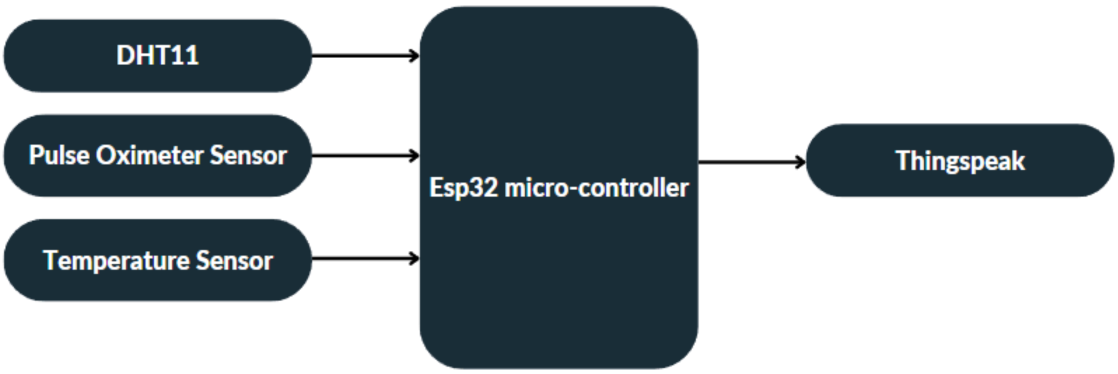


Fig 1.9

Data Validation and Inferences:

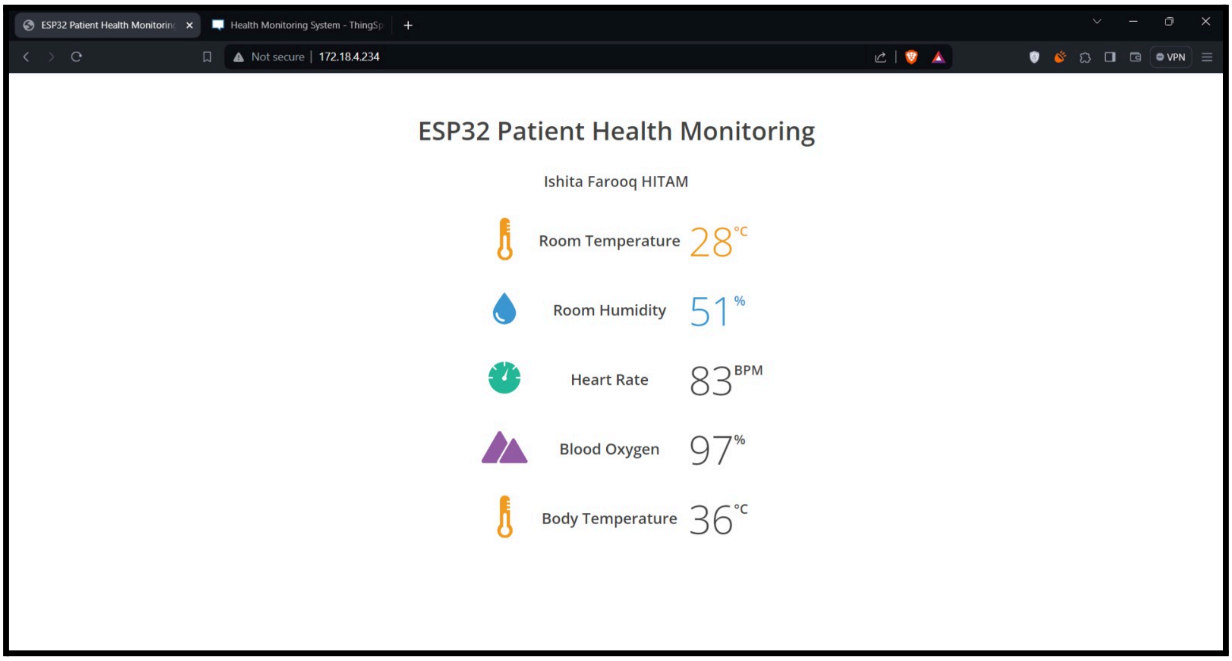
In order to ensure the accuracy and reliability of the information gathered in IoT-based patient health monitoring systems, data validation is essential. Conclusions from the information collected may be made and data validation can be managed.

1. Evaluate each sensor's calibration and verify sensor values using a normal patient's vital readings.
2. Examine verified data to determine trends or anomalies and to determine the health status of the patient.
3. Apply machine learning algorithms to identify patterns suggestive of particular disease states or health disorders.
4. Give healthcare professionals access to decision support technologies that deliver insights derived from verified data.

Extra Implementations:

Data can be displayed in real-time on a **web-based dashboard** in the following manner.

The ESP32 will attempt to establish a network connection. The IP Address will be shown once connected. The IP address can be pasted onto any web browser, and real-time readings like body temperature, heart rate, blood oxygen level, and room temperature and humidity will show up on the screen. The patient's health status can also be checked using a mobile device, by pasting the IP address on the mobile device's web browser.


ESP32 Patient Health Monitoring
Ishita Farooq HITAM
Room Temperature 28 °C
Room Humidity 51 %
Heart Rate 83 BPM
Blood Oxygen 97 %
Body Temperature 36 °C

Result:

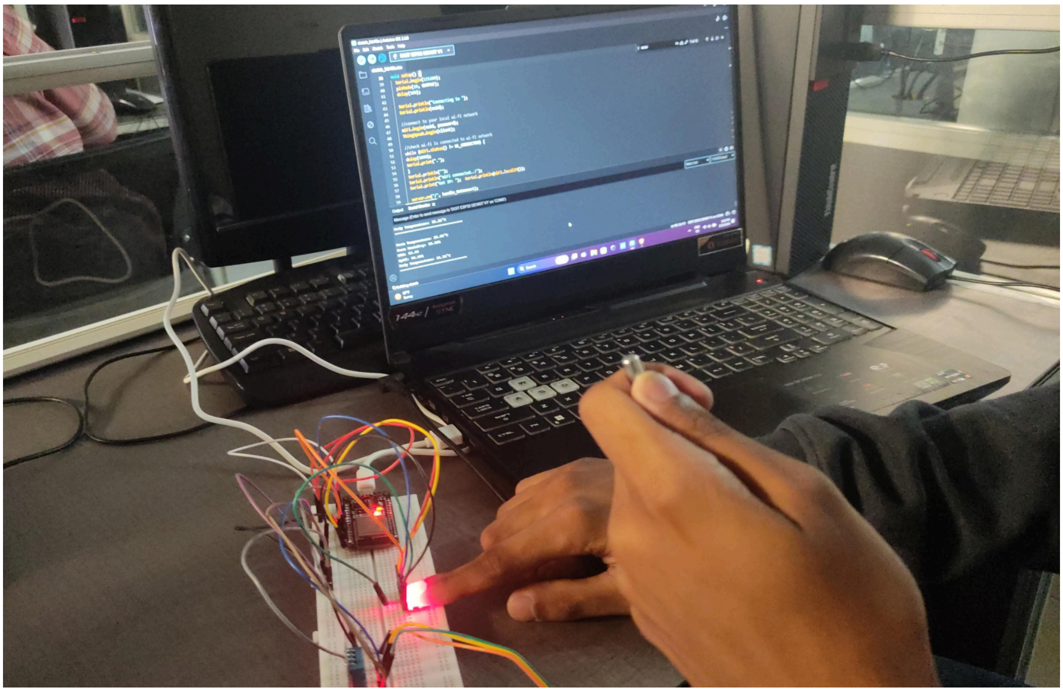
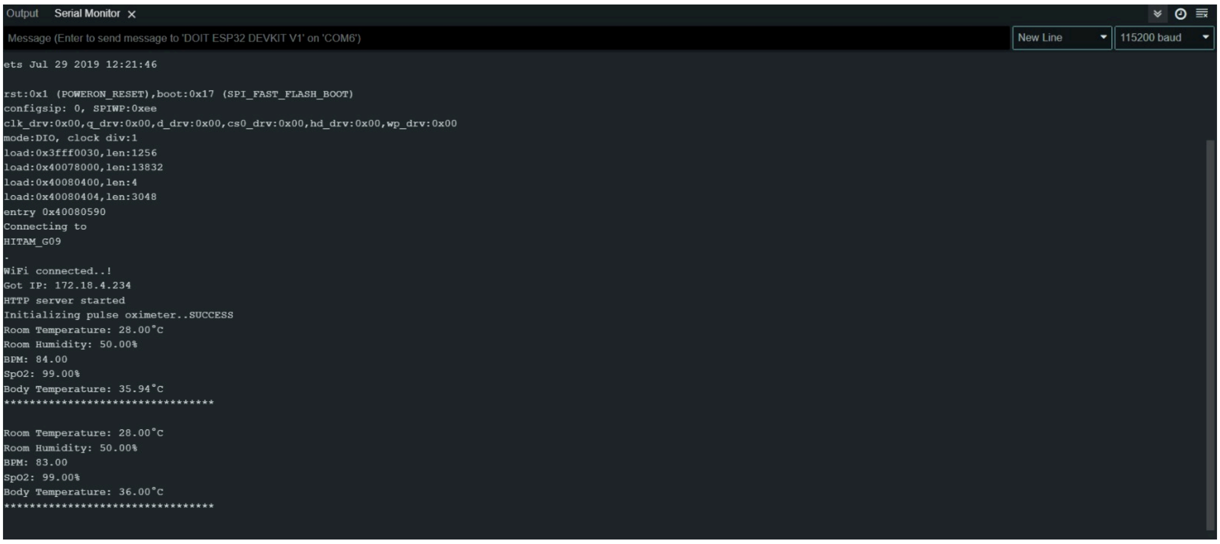


Fig 1.10



Conclusion:

With the combination of IoT technology, the implementation of the Patient Health Monitoring System, which provides **real-time tracking** of health metrics and vital signs, is a major step towards transforming **healthcare technology**. The system demonstrated the feasibility and effectiveness of using IoT devices for continual patient monitoring, giving medical professionals the ability to evaluate patients' health state from a distance with precision and efficiency.

Extensive health parameter monitoring was made possible by the integration of sensors, which generated information about health conditions and general well-being of individuals. Moreover, **smooth transmission of information**, **storage, and visualization** were made possible by the adoption of a platform for data storage and visualization, like ThingSpeak. This made it accessible to medical staff to view and analyze patient data instantly, enabling prompt interventions and individualized treatment.

Future research could concentrate on **advancement** and growth of the system. This involves incorporating more sensors or wearable devices for more thorough health monitoring, improving sensor accuracy through calibration and validation procedures, and improving data processing algorithms to identify subtle health patterns and anomalies. Concerns about security should also be addressed in order to protect patient information and privacy.

The IoT-based Patient Monitoring system, which provides **patient-centered, economical**, and efficient **remote monitoring capabilities**, is a big step in revolutionizing healthcare delivery for individual’s better quality of life.

References:

[1] Ms. Prachi Patil "IoT based Patient Health Monitoring System", International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181

[2] Sulaiman Abdulmalek "IoT-Based Healthcare-Monitoring System towards Improving Quality of Life: A Review", PMCID: PMC9601552, PMID: 36292441

[3] Kelly J, Campbell K, Gong E, Scuffham P "The Internet of Things: Impact and Implications for Health Care Delivery" DOI: 10.2196/20135

[4] Susanna Spinsante and Biswanath Samanta "Application of Internet of Things and Sensors in Healthcare", PMCID: PMC9371210, PMID: 35957294