**HEART RATE MONITORING USING ESP8266**

**Abstract**

A Real-Time Heart Rate Monitoring System using ESP8266 NodeMCU for Enhanced Health

Management.

Introduction: The Industrial Internet of Things (IIoT) revolutionizes industrial processes by

fostering interconnected devices and data collection for enhanced performance and worker

safety. Monitoring heart rate is crucial for assessing cardiovascular health and well-being.

Traditional methods often require dedicated equipment or clinical settings. This project

proposes a novel approach using the Internet of Things (IoT) for real-time heart rate

monitoring.

Objective: The objective of this project is to develop an IoT-based heart rate monitoring

system using the ESP8266 NodeMCU microcontroller. This system aims to provide a

convenient, cost-effective, and readily available solution for individuals to track their heart rate

and gain insights into their health.

Methodology:

1. Hardware Selection: The project will utilize an ESP8266 NodeMCU board as the primary

controller due to its affordability, ease of use, and Wi-Fi connectivity.

2. Sensor Integration: A pulse sensor will be integrated with the ESP8266 to detect changes in

blood volume associated with each heartbeat.

3. Signal Processing: The acquired raw signal from the pulse sensor will be processed using

appropriate algorithms (e.g., peak detection) to extract the heart rate data.

4. Data Visualization and Storage: The extracted heart rate data will be transmitted wirelessly

via Wi-Fi to a chosen platform (e.g., smartphone app, cloud-based dashboard) for real-time

visualization and potential storage for historical analysis.

Conclusion: This project demonstrates the feasibility of an ESP8266 NodeMCU-based heart

rate monitoring system. This system offers a user-friendly and cost-effective approach to

promoting health awareness and providing valuable data for individuals to manage their wellbeing.

Additionally, this project contributes to the growing exploration of IoT applications in

the realm of personal health monitoring.

INTRODUCTION

**Industrial Internet of Things (IIoT):**

The Industrial Internet of Things (IIoT) represents a paradigm shift in the way industrial

operations are managed and optimized. It fosters a network of interconnected sensors, devices,

machines, and other industrial assets, all collecting, transmitting, and analyzing data to drive

significant advancements in several key areas:

Enhanced Efficiency: Real-time monitoring and control facilitated by IIoT enable the

streamlining of production processes and a reduction in downtime. This translates to increased

output and a more efficient use of resources.

Improved Safety: The deployment of sensors within the IIoT ecosystem allows for the early

detection of potential equipment failures or safety hazards. By identifying these issues before

they escalate, operators can implement preventive measures, minimizing risks and ensuring

worker safety.

Elevated Quality Control: Continuous monitoring of product quality parameters made possible

by IIoT systems helps maintain consistency and minimize defects. This results in a higher

quality output and increased customer satisfaction.

Predictive Maintenance: By leveraging collected sensor data and advanced analytics, IIoT

facilitates the implementation of predictive maintenance strategies. This allows for the

identification and resolution of potential equipment issues before they lead to breakdowns,

significantly reducing maintenance costs and unplanned downtime.

**Heart Rate Monitoring for Worker Well-being in IIoT Applications:**

While a wide range of sensors and devices play crucial roles within IIoT, the potential

integration of heart rate monitoring using an ESP8266 NodeMCU microcontroller presents an

intriguing possibility. However, it's essential to approach this application with careful

consideration of its limitations and potential benefits.

Potential Benefits:

Stress Detection and Management: In high-pressure industrial environments, real-time heart

rate data can provide valuable insights into potential worker stress levels. Early identification

of stress allows for targeted interventions such as workload adjustments, scheduled breaks, or

even stress management training programs. This can lead to improved worker well-being,

reduced stress-related accidents, and potentially even enhanced productivity.

Heat Stress Monitoring: Workers in hot environments are at risk of heat exhaustion or stroke.

Heart rate monitoring can be a valuable tool for identifying individuals who might be

experiencing heat stress, enabling the implementation of preventive measures such as

providing hydration breaks, establishing cooldown stations, or adjusting work schedules.

Considerations and Limitations:

Privacy Concerns: Workers have a right to privacy regarding their personal health data. It's

crucial to address these concerns proactively through transparent communication, clear data

usage policies, and obtaining explicit worker consent for heart rate monitoring.

Data Interpretation: Heart rate is influenced by a variety of factors beyond workload, including

caffeine intake, medications, or individual fitness levels. When interpreting heart rate data

within an industrial context, these factors need to be carefully considered for accurate analysis.

False Positives and Negatives: Stress, heat, or exertion might not always be reflected in

elevated heart rate. Similarly, someone with a naturally high resting heart rate could be flagged

for no reason. Algorithms and data analysis frameworks need to be designed with these

potential issues in mind to minimize inaccuracies.

Suitability for Specific Roles: The applicability of heart rate monitoring may vary across

different industrial jobs. While it could be highly beneficial in physically demanding roles, the

value proposition might be less compelling for office workers with minimal physical exertion.

Conclusion:

IIoT holds immense potential for transforming industrial processes, enhancing efficiency, and

improving safety. While heart rate monitoring using an ESP8266 NodeMCU might not be a

universally applicable solution, it can be a valuable tool in specific contexts where worker wellbeing

and stress management are critical factors for worker safety and overall productivity. A

successful implementation requires careful consideration of its limitations, a focus on ethical

data management practices, and clear communication with the workforce.

STEP BY STEP PROCEDURE:

This project helps to provide high-quality support to patients & make things easier for medical

staff. Normal people can also use this as a part of their life to monitor their heart health.

Components Required:

ESP8266 NodeMCU Board

OLED Display

Heartbeat Sensor Module

Breadboard

Jumper Wires

LED Lights

**MICROCONTROLLER(ESP8266)**

The ESP8266 is a low-cost, microcontroller chip with integrated Wi-Fi capabilities, making it

a popular choice for developing Internet of Things (IoT) applications. Here's a breakdown of

its key features:

This is the ESP8266's defining feature. It enables the chip to connect to wireless

networks, facilitating communication with the internet and other Wi-Fi-enabled

devices.

Compared to other microcontrollers with Wi-Fi capabilities, the ESP8266 is

significantly cheaper, making it an attractive option for cost-sensitive IoT projects.



The ESP8266 has a well-established development ecosystem with popular

programming languages and frameworks like Arduino being readily available. This

simplifies programming and reduces devel.

**Pulse Sensor**

It is an Analog device that is used to measure the pulses/heartbeat of the human body. It

monitors the change of blood in the blood vessels by emitting & sensing the light, the process

is known as photoplethysmography.



It has three pins GND, VCC, and signal.

**VCC pin**: This pin is used to supply power to the sensor to make it work. It is typically connected

to the 3.3V pin on the NodeMCU board. Its voltage range lies between 3.3v to 5v.

**GND pin**: This pin is used to connect the sensor and the ground of the NodeMCU board.

**Signal pin**: This pin is used to transmit the pulse rate data in analog form to the NodeMCU board.

**OLED Display**

Organic Light \_Emitting Diode(OLED) Displays are used to display the output data. They are

extremely light, paper-thin, theoretically flexible, and produce a brighter, crisper image.



It has four pins VCC, GND, SCl, and SDA.

**VCC pin**: This pin is used to supply the voltage to the OLED display to turn on the

display. The voltage range of OLED lies from 3.3v to 6v.

**GND pin**: This pin is used to connect the display to the ground of the NodeMCU board.

**SDA pin**: This pin is used for data transmission in I2C communication. It is typically

connected to the D2 pin on the NodeMCU board.

**SCL pin**: This pin is used for clock signals in I2C communication. It is typically

connected to the D1 pin on the NodeMCU board.

**LED**

A green light-emitting diode (LED) aimed at the skin, and measure the amount of light

reflected back. Since red blood absorbs green light, the amount of light reflected back



changes as blood pulses through tiny blood vessels near the surface of the skin.

**JUMPING WIRES**

Jumping wires, also known as Dupont wires, are essential components for building

prototypes and experimenting with electronics in IoT projects. They provide a simple

and flexible way to connect various electronic components on a breadboard or other

prototyping platforms.



**Setting up Thingspeak for Remote Monitoring**

To monitor the output data over the cloud, we will be using the **Thingspeak** cloud platform.

There we will create a field for storing the heartbeats data. Let’s set it up quickly.

First of all, create a new account on the **Thingspeak website**.

Then login into your account and create a new channel. Channels have read and write

API keys and can be public or private.

Give a name to a field in the channel and use it for data monitoring.

Inside the channel go to API keys and copy it. API keys are different for different

channels, we need to use them in Code to provide access to the channel with channel

ID.

There are **Read & Write both API keys**. To send & Receive data to Thingspeak, use

the **write & read API keys** respectively.

**CIRCUIT DIAGRAM**



**PROGRAM**

#include <SPI.h>

#include <Wire.h>

#include <Adafruit\_SSD1306.h>

#include <Adafruit\_GFX.h>

#include <ESP8266WiFi.h>

#include <ThingSpeak.h>

Adafruit\_SSD1306 display(128,64,&Wire);

WiFiClient client;

const int sensorPin = A0; // A0 is the input pin for the heart

rate sensor

int sensorValue; // Variable to store the value coming

from the sensor

int count = 0;

unsigned long starttime = 0;

int heartrate = 0;

boolean counted = false;

long myChannelNumber = 2099478;

const char myWriteAPIKey[] = "A9JKMSZW73UXLFQ2";

void setup (void)

{ display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

pinMode (D8, OUTPUT); // D8 LED as Status Indicator

Serial.begin (115200); // Start Serial Communication @

115200

display.clearDisplay();

WiFi.begin("Semicon Media 2.4", "cdfiP29to665");

while(WiFi.status() != WL\_CONNECTED)

{delay(200);

Serial.print("..");}

Serial.println();

Serial.println("NodeMCU is connected!");

Serial.println(WiFi.localIP());

ThingSpeak.begin(client);

}

void loop ()

{ starttime = millis();

while (millis()<starttime+20000) // Reading pulse sensor for 20 seconds

{ sensorValue = analogRead(sensorPin);

//Serial.println (sensorValue);

delay(50);

if ((sensorValue >= 590 && sensorValue <=680) && counted == false) //

Threshold value is 590 (~ 2.7V)

{ count++;

digitalWrite (D8,HIGH);

delay (10);

digitalWrite (D8, LOW);

counted = true;

}

else if (sensorValue < 590)

{ counted = false;

digitalWrite (D8, LOW);

}

}

Serial.print ("Pulse ");

Serial.println (count);

heartrate = (count)\*3; // Multiply the count by 3 to get beats

per minute

Serial.println ();

Serial.print ("BPM = ");

Serial.println (heartrate); // Display BPM in the Serial Monitor

Serial.println ();

count = 0;

display.clearDisplay();

display.setTextColor(WHITE);

display.setCursor(0,0);

display.setTextSize(2);

display.println("Heart Rate");

display.setCursor(0,28);

display.print("BPM: ");

display.print(heartrate);

display.display();

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

}

**OUTPUT**

****

**CONCLUSION**

**In Fitness and Exercise:**

**Optimizing Workout Intensity:** Heart rate (HR) is a reliable indicator of exertion

level. By monitoring HR, individuals can train within target zones to maximize fitness

gains and avoid overtraining.

**Tracking Calorie Expenditure:** Heart rate data, along with other factors, helps

estimate calorie burn during exercise.

**In General Health Management:**

**Identifying Underlying Conditions:** Abnormal heart rhythms (arrhythmias) can be

detected through HR monitoring, prompting further investigation for potential heart

problems.

**Stress Management:** HR variability (HRV), the variation in time intervals between

heartbeats, reflects our nervous system activity. Monitoring HRV can help assess

stress levels and guide stress management techniques.