**ZIAUDDIN UNIVERSITY**

### FACULTY OF ENGINEERING SCIENCE TECHNOLOGY AND MANAGEMENT

**BIOMEDICAL CONTROL SYSTEM**

**PROJECT REPORT**



**Project Title: “Telehealth Monitoring System”**

**Submitted by:**

1. Gautam Sharma (4-7/2021/024)
2. Safia Tahir (4-7/2021/010)

**Submitted to:**

Engr. Tehreem khan

Engr. Muzammil Nayyar (Course In-Charge)

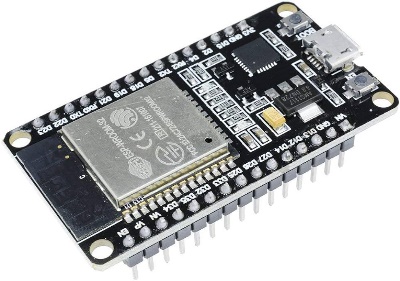
**“Telehealth Monitoring System”**

**Introduction:**

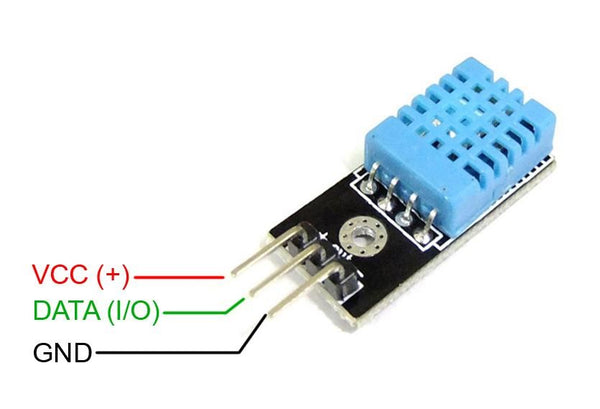
The Telehealth Monitoring System is designed to provide real-time health monitoring using a microcontroller-based system. It measures vital parameters such as heart rate, room temperature, room humidity, and body temperature. The system utilizes sensors and a Wi-Fi-enabled microcontroller to collect and transmit data to a mobile application through the Blynk platform. The integration of IoT technology ensures accessible, remote health monitoring, which is increasingly essential for patient care in telehealth.

**Components Used:**

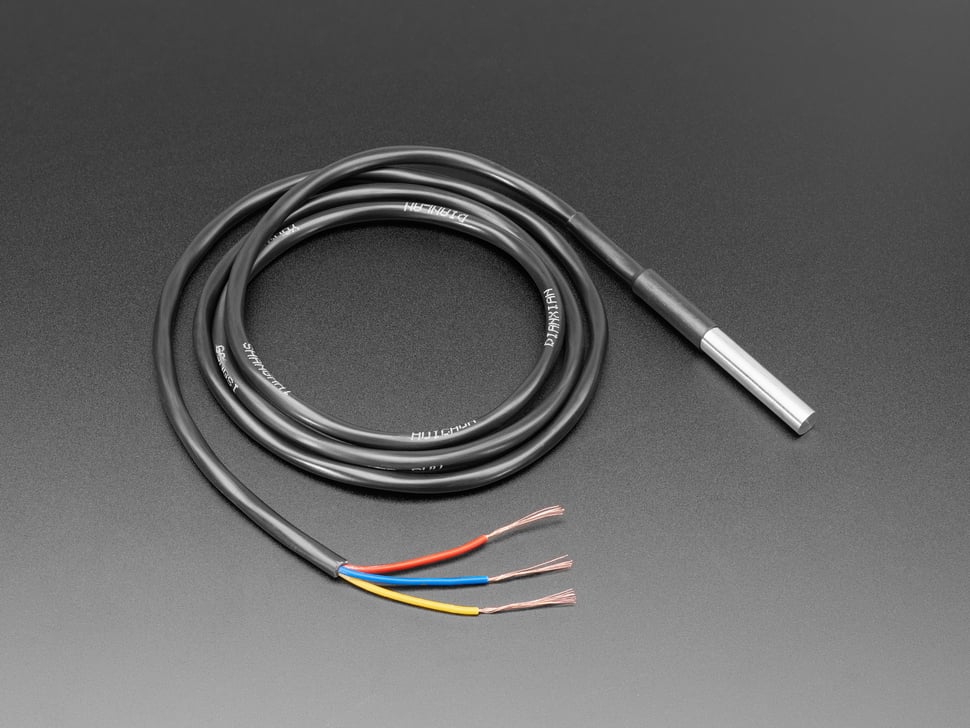
**ESP32:**



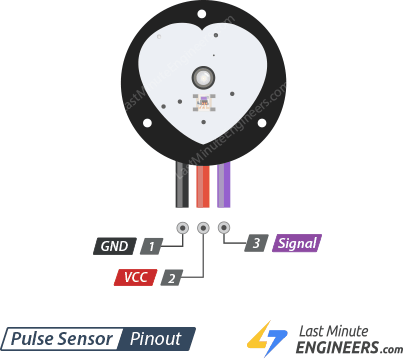
**DHT11 Sensor:**



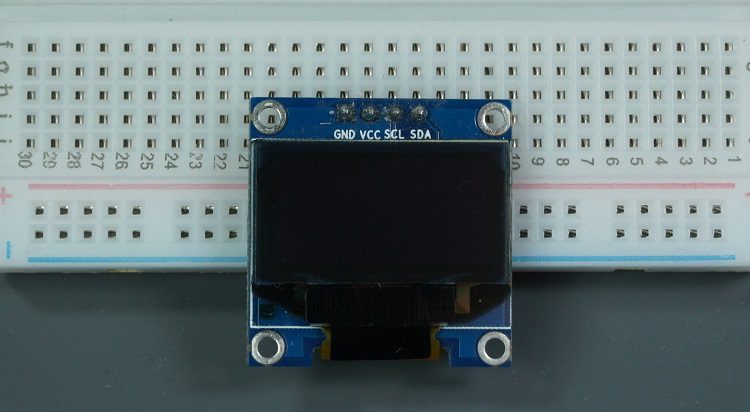
**DS18B20 Sensor:**



**HW827 Sensor:**



**Adafruit SSD1306 OLED Display:**

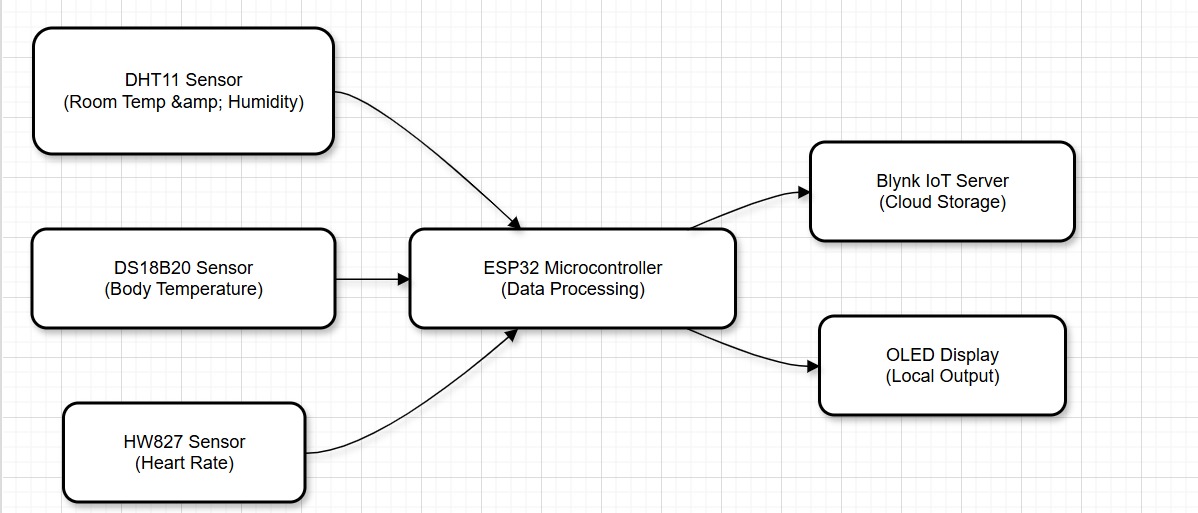


**Blynk IoT Platform:** For remote data visualization and monitoring.

**Wires and Breadboard:**



**Flow Diagram:**



**Code:**

#define BLYNK\_TEMPLATE\_ID "TMPL66kx7AtR\_"

#define BLYNK\_TEMPLATE\_NAME "Telehealth"

#include <Wire.h>

#include <Blynk.h>

#include <WiFi.h>

#include <BlynkSimpleEsp32.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include "DHT.h"

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#define BLYNK\_PRINT Serial

// Blynk and Wi-Fi credentials

char auth[] = "a\_37QCTlvSKhfAn8wo2NVKSD\_0W\_Jfv4";

char ssid[] = "Tenda";

char pass[] = "sharma123";

// Pin configuration

#define DHTTYPE DHT11

#define DHT\_PIN 18 // DHT11 sensor pin

#define DS18B20\_PIN 5 // DS18B20 temperature sensor pin

#define HW827\_PIN 34 // Analog pin for HW827 sensor

#define OLED\_RESET -1 // OLED reset pin (not used)

// Create SSD1306 OLED display object

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

// Define reporting period for serial updates (in milliseconds)

#define REPORTING\_PERIOD\_MS 1000

// Sensor objects

DHT dht(DHT\_PIN, DHTTYPE);

OneWire oneWire(DS18B20\_PIN);

DallasTemperature sensors(&oneWire);

// Variables for storing sensor data

float roomTemperature, roomHumidity;

float bodyTemperature, heartRate;

uint32\_t lastReportTime = 0;

// Constants

const int threshold = 600; // Adjust threshold based on your HW827 sensor's output (Test and tune this)

bool pulseDetected = false; // Flag for pulse detection

unsigned long lastPulseTime = 0;

unsigned long pulseInterval = 0;

unsigned long lastDebounceTime = 0; // For debouncing the pulse signal

unsigned long debounceDelay = 50; // Delay in ms for debouncing

void setup() {

Serial.begin(115200);

Blynk.begin(auth, ssid, pass);

// Initialize OLED display

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

Serial.println("SSD1306 allocation failed.");

while (1); // Halt if OLED initialization fails

}

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(SSD1306\_WHITE);

display.display();

// Initialize sensors

dht.begin();

sensors.begin();

Serial.println("Initializing sensors...");

}

void loop() {

Blynk.run();

// Read DHT11 sensor data

roomTemperature = dht.readTemperature();

roomHumidity = dht.readHumidity();

// Read DS18B20 body temperature data

sensors.requestTemperatures();

bodyTemperature = sensors.getTempCByIndex(0);

// Check for valid DS18B20 reading

if (bodyTemperature == DEVICE\_DISCONNECTED\_C) {

Serial.println("Error: DS18B20 not detected.");

bodyTemperature = 0.0;

}

// Read HW827 heart rate sensor data

int hw827Value = analogRead(HW827\_PIN);

// Implement debouncing to filter out noise

if (millis() - lastDebounceTime > debounceDelay) {

if (hw827Value > threshold && !pulseDetected) {

pulseDetected = true;

unsigned long currentTime = millis();

if (lastPulseTime > 0) {

pulseInterval = currentTime - lastPulseTime;

if (pulseInterval > 0) {

// Calculate heart rate (BPM)

heartRate = 60000.0 / pulseInterval; // Convert milliseconds to BPM

}

}

lastPulseTime = currentTime;

} else if (hw827Value < threshold) {

pulseDetected = false;

}

lastDebounceTime = millis();

}

// Ensure readings are updated at the defined interval

if (millis() - lastReportTime > REPORTING\_PERIOD\_MS) {

Serial.print("Heart Rate (HW827): ");

Serial.print(heartRate);

Serial.println(" bpm");

Serial.print("Room Temperature: ");

Serial.print(roomTemperature);

Serial.println("°C");

Serial.print("Room Humidity: ");

Serial.print(roomHumidity);

Serial.println("%");

Serial.print("Body Temperature (DS18B20): ");

Serial.print(bodyTemperature);

Serial.println("°C");

// Update OLED display

display.clearDisplay();

display.setCursor(0, 0);

display.println("Telehealth Monitoring");

display.println("--------------------");

display.print("Heart Rate: ");

display.print(heartRate);

display.println(" bpm");

display.print("Room Temp: ");

display.print(roomTemperature);

display.println(" C");

display.print("Humidity: ");

display.print(roomHumidity);

display.println(" %");

display.print("Body Temp: ");

display.print(bodyTemperature);

display.println(" C");

display.display();

// Send data to Blynk

Blynk.virtualWrite(V3, roomTemperature); // Room Temperature

Blynk.virtualWrite(V4, roomHumidity); // Room Humidity

Blynk.virtualWrite(V5, heartRate); // Heart rate (BPM)

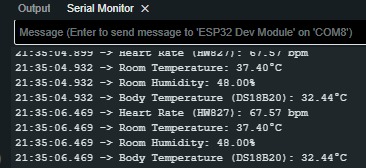
Blynk.virtualWrite(V7, bodyTemperature); // Body Temperature

lastReportTime = millis();

}

}

**Output­:**



**Blynk output:**



**Project Picture:**



**Program Algorithm:**

Initialization: Set up Wi-Fi credentials, sensors, and the OLED display.­­

**Data Collection:**

Read temperature and humidity using the DHT11 sensor.

Retrieve body temperature from the DS18B20 sensor.

Measure heart rate using the HW827 sensor, implementing debouncing to filter noise.

**Data Processing:**

Calculate heart rate in BPM based on pulse intervals.

**Data Display and Transmission:**

Show readings on the OLED display.

Send data to the Blynk app for remote monitoring.

Periodic Updates: Ensure consistent data refresh at 1-second intervals.

**Code Explanation:**

**Libraries and configuration:**  
The code imports libraries for Wi-Fi, Blynk, sensors, and the OLED display. Key parameters, such as Wi-Fi credentials and pin definitions, are defined.

**Initialization:**

Initializes the sensors (DHT11, DS18B20) and OLED display.

Connects the ESP32 to the Wi-Fi and Blynk platform.

**Sensor Reading:**

Room temperature and humidity are fetched using the dht.readTemperature() and dht.readHumidity() methods.

Body temperature is retrieved using the sensors.getTempCByIndex(0) method from the DS18B20 sensor.

**Heart Rate Calculation:**

The HW827 sensor's analog signal is processed to detect pulses.

Pulse intervals are calculated and converted to BPM.

**Display and Transmission:**

Data is displayed on the OLED using the Adafruit SSD1306 library.

It is sent to virtual pins in the Blynk app for real-time remote access.

**Construction:**

**Hardware Setup:**

Connect the DHT11 sensor to pin 18, DS18B20 sensor to pin 5, and HW827 sensor to analog pin 34 on the ESP32.

Wire the OLED display to the I2C pins of the ESP32.

Assemble all components on a breadboard for testing and debugging.

**Software Setup:**

Upload the code to the ESP32 using the Arduino IDE.

Configure the Blynk app with the provided authentication token and virtual pin mappings.

**Conclusion:**

The Telehealth Monitoring System is a cost-effective and efficient solution for real-time health parameter monitoring. By leveraging IoT and sensors, it provides vital health metrics for remote patients. This system enhances telehealth applications by improving accessibility, reducing hospital visits, and ensuring timely health interventions.