# README – IoT-Based Health Monitoring System

This guide provides everything needed to set up a real-time health monitoring system using an ESP32 microcontroller. Sensor readings including Heart Rate (BPM), SpO2 level, and Body Temperature are captured and uploaded to Firebase. These readings can be viewed dynamically on a local webpage with a connected health form.

## 🔧 Components Required

• ESP32 Dev Board  
• MAX30100 or MAX30102 (Pulse and SpO2 Sensor)  
• MLX90614 (Non-contact Infrared Temperature Sensor)  
• Jumper wires  
• Breadboard  
• Micro USB cable for ESP32  
• Wi-Fi with internet access

## 🔌 Circuit Block Diagram

• Connect MAX30100:  
 - VIN → 3.3V on ESP32  
 - GND → GND  
 - SDA → GPIO21  
 - SCL → GPIO22  
  
• Connect MLX90614:  
 - VIN → 3.3V  
 - GND → GND  
 - SDA → GPIO16  
 - SCL → GPIO17

## ☁️ Firebase Setup Instructions

1. Go to https://console.firebase.google.com/  
2. Create a new project (e.g., WellSpan-IoT)  
3. Navigate to Realtime Database → Create Database  
4. Set it to test mode and note your database URL (e.g., https://<project-id>.firebaseio.com)  
5. In the ESP32 Arduino sketch, replace `FIREBASE\_HOST` and `FIREBASE\_AUTH` with your project credentials  
6. Enable anonymous authentication (if needed) under Authentication tab

## 📟 ESP32 Arduino Code

• The code reads:  
 - Heart Rate and SpO2 from MAX30100  
 - Body Temperature from MLX90614  
• Sends data as a JSON object to Firebase every few seconds  
• Libraries required:  
 - FirebaseESP32.h  
 - Wire.h  
 - Adafruit\_MLX90614.h  
 - MAX30100\_PulseOximeter.h

## 🌐 Web Interface Integration

• A local web interface fetches Firebase values and auto-fills a health form (BPM, SpO2, Temperature)  
• JavaScript `fetch()` is used to request values from the Firebase Realtime Database  
• Users can then submit the full form to get personalized life expectancy predictions and chatbot recommendations

## 📂 Folder Contents

• iot\_code.ino – Complete Arduino sketch for ESP32  
• Firebase Database setup instructions – (see above)  
• Web form integration code – Reads Firebase values into a user form (see frontend folder)

## 📝 Notes

• Make sure your ESP32 board is connected to the same Wi-Fi as configured in the sketch  
• Check serial monitor for live debug logs during sensor initialization and upload  
• Accuracy depends on sensor placement and environmental conditions