AY 2024-25

IV Semester

ECL405

A PROJECT REPORT
ON

PROJECT TITLE

Bachelor of Technology

in

Electronics and Communication Engineering

By

Nitin Sharma (22234)

SCHOOL OF ELECTRONICS

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY UNA HIMACHAL PRADESH

APRIL 2024

BONAFIDE CERTIFICATE

This is to certify that the project titled *Health Monitoring System* is a bonafide record of the work done by

Nitin Sharma (22234)

in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Electronics and Communication Engineering of the INDIAN INSTITUTE OF INFORMATION TECHNOLOGY UNA, HIMACHAL PRADESH, during the year 2024 - 2025.

under the guidance of

Mr. Satish Kumar(CSE)

Project viva-voce held on: 26, April.'2024

Internal Examiner External Examiner

ORIGNALITY / NO PLAGARISM DECLARATION

We certify that this project report is our original report and no part of it is copied from any published reports, papers, books, articles, etc. We certify that all the contents in this report are based on our personal findings and research and we have cited all the relevant sources which have been required in the preparation of this project report, whether they be books, articles, reports, lecture notes, and any other kind of document. We also certify that this report has not previously been submitted partially or as whole for the award of degree in any other university in India and/or abroad.

We hereby declare that, we are fully aware of what constitutes plagiarism and understand that if it is found at a later stage to contain any instance of plagiarism, our degrees may be cancelled.

Nitin Sharma (22234)

ABSTRACT

In an era of increasing awareness about personal health, the integration of technology into healthcare systems has become imperative. This project presents the development of a Health Monitoring System (HMS) centered around a heart rate sensor and an accompanying application designed to provide real-time monitoring and analysis of vital signs.

The Health Monitoring System offers a comprehensive solution for individuals seeking to proactively manage their health. By providing continuous monitoring, insightful analysis, and personalized feedback, it empowers users to make informed decisions about their well-being and facilitates early intervention when necessary. Through the seamless integration of hardware and software components, this system represents a significant advancement in the realm of personal health monitoring.

Furthermore, the HMS employs advanced algorithms for data analysis, allowing for the detection of anomalies and trends in the user's vital signs. These algorithms facilitate early detection of potential health issues and enable proactive intervention. Additionally, the system ensures data privacy and security through robust encryption methods, safeguarding sensitive health information.

ACKNOWLEDGEMENT

We would like to thank the following people for their support and guidance without whom the completion of this project in fruition would not be possible.

We would like to express our sincere gratitude and heartfelt thanks to Mr. Satish Kumar for their unflinching support and guidance, valuable suggestions and expert advice. Their words of wisdom and expertise in subject matter were of immense help throughout the duration of this project.

We also take the opportunity to thank our Director and all the faculty of School of Computing/Electronics, IIIT Una for helping us by providing necessary knowledge base and resources.

We would also like to thank our parents and friends for their constant support.

Nitin Sharma(22234)

TABLE OF CONTENTS

Title **Page No.**

LIST OF ACRONYMS

IOT	Internet of Things
RPM	Remote Patient Monitoring
IDE	Integrated Development Environment

LIST OF FIGURES

Chapter 1
Introduction

1.1 BACKGROUND

These days, the expansion of innovations by wellbeing specialists is exploiting these electronic devices. IoT devices are profoundly utilized in the clinical area. In this paper, the research is about an IoT-based health monitoring system. In particular, for COVID-19 patients, high blood pressure patients, hypertension patients, diabetic patients, etc., in a country territory, in rural areas, the number of doctors is not exactly the same as in urban areas. Medical equipment is not readily available in rural areas, except for government medical centers. The percentage of patients in these clinics is greater than that in government medical facilities. Similarly, the equipment has, for the most part, ended. As a result, if an emergency situation arises, this hardware component will send a report to the physicians or medical professionals as soon as possible. The remaining work will be done by doctors based on their reports.

The main contribution of this paper is to highlight IoT-based healthcare-monitoring systems in detail so that future researchers, academicians, and scientists can easily find a roadmap to understand the current healthcare-monitoring systems and can easily provide solutions and enhancements for such critical applications. In this research paper, we provide a general idea of IoT-based healthcare-monitoring systems in a systematic way, along with their benefits and significance, and a literature review. Moreover, we discuss the concepts of wearable things in healthcare systems from an IoT perspective. The paper also provides a classification of healthcare-monitoring sensors, addresses security and protocols for IoT healthcare-monitoring systems, and details challenges and open issues. We also suggest solutions to overcome these challenges and issues in the future.

1.2 SIGNIFICANCE

The development of monitoring systems for healthcare is receiving a great deal of attention from researchers and leaders in the medical field. Several successful research projects have been conducted in this area, and many more are currently underway. The number of gaps in care provided by healthcare providers is increasing significantly, directly resulting from the rapidly growing number of older adults and patients with chronic illnesses. The major shortcoming is that healthcare is only provided in hospitals; therefore, it is unsuitable for seniors and people with disabilities and cannot always meet their needs. The IoT, with the help of sensor values and telecommunications, provides an effective and practical solution to the issue of real-time monitoring of the health status of the elderly.

It has been shown that the IoT, in conjunction with smart technologies, can provide various improved and enhanced services. Using sensors, researchers have developed various emergency systems using technologies that enable intelligent and remote wireless communication. These technologies have been used for various medical purposes, particularly in monitoring the health of the elderly. This way, data can be collected on general health and dangerous situations by capturing important vital signs.

1.2.1 Benefits of Using IOT in Healthcare

The IoT will reshape healthcare as we know it, with profound implications. In terms of how apps, devices, and people communicate with each other to deliver healthcare solutions, we have reached a whole new level of evolution. The IoT has given us a new perspective and tools for an integrated healthcare network, greatly improving healthcare quality.

The IoT has made it possible to automate healthcare procedures that previously required a significant amount of time and left room for error due to human involvement. For example, to control airflow and temperature in operating rooms, many hospitals now use networked devices.

There are almost endless ways the IoT can improve medical care; however, the following are some of the key benefits:

1. Reduced cost of care
2. Human errors are reduced.
3. Elimination of the limitations of distance.
4. Reduced amounts of paperwork and record keeping.
5. Chronic disease are detected early.
6. Improvement in medication management,

1.3 OBJECTIVE

The primary objectives of the project include

- The development of a scalable IoT infrastructure,
- Implementation of sensors for health parameter monitoring,
- Creation of a user-friendly interface,
- Ensuring real-time data transmission, and incorporating robust security measures to safeguard sensitive health information.

1.4 SCOPE

The scope of the Health Monitoring System using IoT encompasses the development of a robust and scalable infrastructure for real-time health parameter monitoring. The project will focus on key health metrics such as heart rate, body temperature, and activity levels. Additionally, the system will provide a secure and user-friendly interface for individuals and healthcare professionals to access and interpret health data.

Chapter 2

Review of Literature

2.1 Health Monitoring System

IoT based patient health monitoring system is a generic term given to any medical equipment that has internet capability and can measure one or more health data of a patient who is connected to the device such as heartbeat, body temperature, blood pressure, ECG, steps etc.

IoT based health monitoring system is used where the patient and heath expert(s) are at different locations. For example, a patient can stay at home and continue his/her routine life and a doctor can monitor patient's heath. Based on the received data the heath expert can prescribe a best treatment or take an immediate action in case of an emergency.

An IoT based health monitoring system using HC-05 is a project that involves integrating various sensors and devices to gather and analyze health data of an individual. The HC-05 is a module that can provide wireless functionality. We are using this module to set up communication between the Arduino and the mobile application.

The system can be designed to monitor various health parameters such as heart rate, body temperature, oxygen level, and many others. The data collected from the sensors can be transmitted through a HC-05 Bluetooth Module to Mobile Application Develop under this project which show sensor reading and store individual data records.

2.1.1 Block diagram

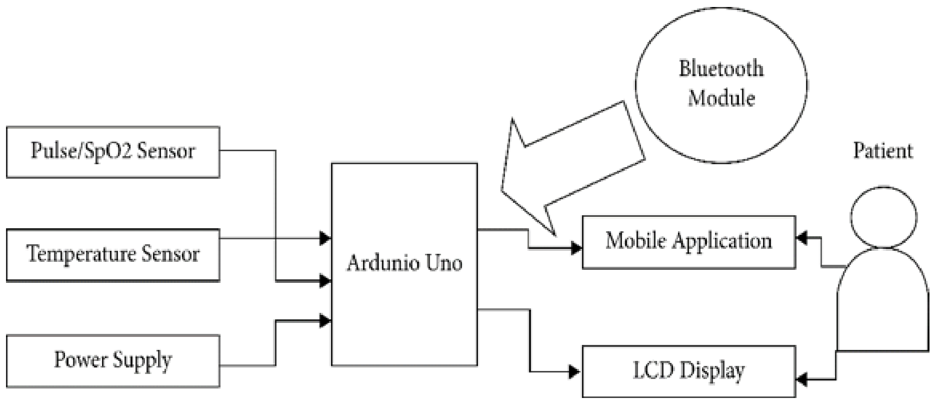

Figure 2.1: Block diagram of the system

The system's structure is depicted in Figure 2.1. Here, patients will measure their pulse rate and SpO2 using the MAX30100 sensor and body temperature using the Lm35 sensor, and patients can see measurement data in the mobile app and LCD display. The data will be shown in the mobile app with the help of a Bluetooth module that will receive data from the Arduino and save it in the cloud. From there, the data will be transferred into the mobile application, and the patients can view the measurement of the health parameters

Chapter 3

System Architecture

3.1 Hardware Design

This health monitoring system consists of sensors and a microcontroller. We used the Arduino Uno as the microcontroller, and the sensors are MAX30100, LM35 and DHT11. And there are more components we are using, such as an HC-05 (Bluetooth module), to connect the Arduino with the mobile application and LCD display. All the needed components for the health monitoring system are described in Table 2.1.

Components	Description
Arduino UNO	An open-source microcontroller based on the 8 bit ATmega238p microchip. It will work as an interface between the sensor and the mobile application .
MAX30100	The MAX30100 is a pulse oximetry and heart rate monitor sensor that can be powered by 1.8 V or 3.3 V power supply.
DHT 11	The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air
HC-05	The HC-05 is a module that can provide wireless functionality. We are using this module to set up communication between the Arduino and the mobile application.

Table 3.1: Components

Figure 2.2 is the circuit diagram for the system. An Arduino Uno microcontroller, three sensors (MAX30100, DHT11 and LM35), a Bluetooth module make up the circuit. The whole system is powered by 5V. The microcontroller (Arduino Uno) is connected to the computer using a USB.

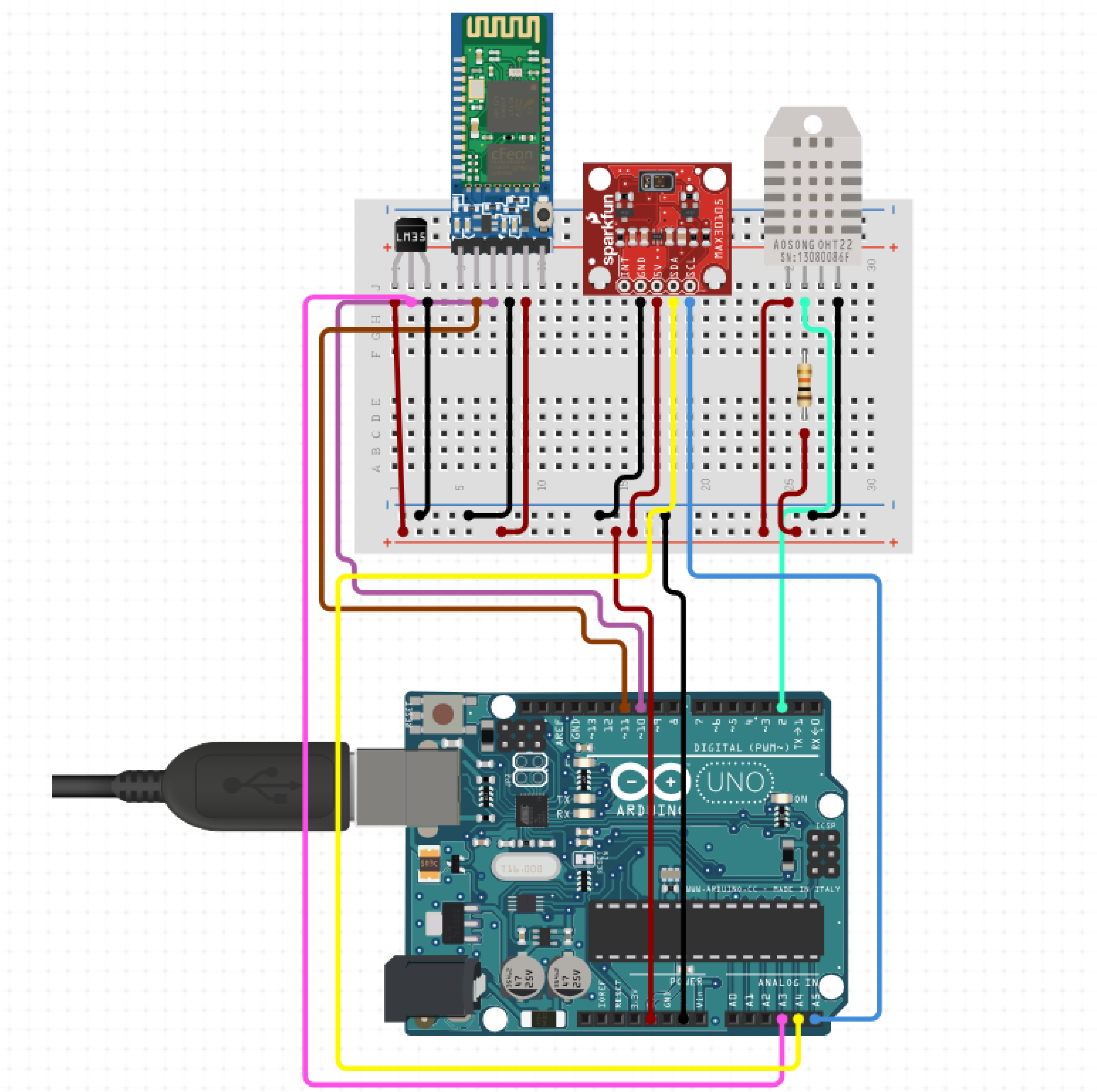

Figure 3.1: Circuit Diagram of system

3.1.1 Description of Module

HC-05: Designed to replace cable connections HC-05 uses serial communication to communicate with the electronics. Usually, it is used to connect small devices like mobile phones using a short-range wireless connection to exchange files. It uses the 2.45GHz frequency band. The transfer rate of the data can vary up to 1Mbps and is in range of 10 meters.

The HC-05 module can be operated within 4-6V of power supply. It supports baud rate of 9600, 19200, 38400, 57600, etc. Most importantly it can be operated in Master-Slave mode which means it will neither send or receive data from external sources.

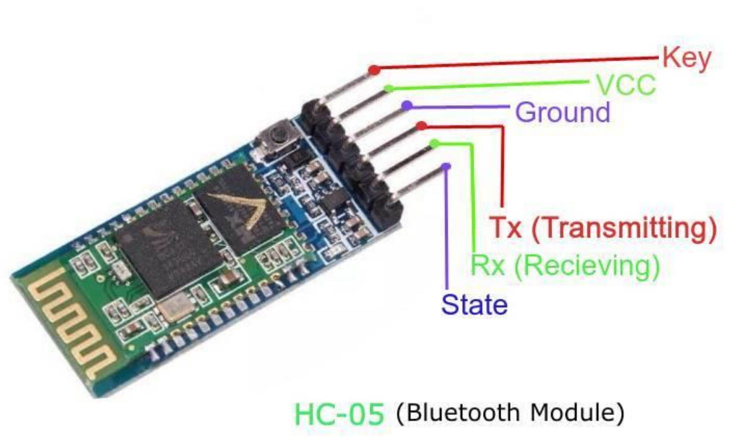

Figure 3.2: HC-05

DHT11: DHT11 is a digital temperature and humidity sensor. DHT11 humidity and temperature sensor is available as a sensor and as a module. The difference between this sensor and module is the pull-up resistor and a power-on LED. DHT11 is a relative humidity sensor. To measure the surrounding air this sensor uses a thermistor and a capacitive humidity sensor.

The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy. The sampling rate of this sensor is 1Hz.i.e. it gives one reading for every second. DHT11 is small in size with operating voltage from 3 to 5 volts. The maximum current used while measuring is 2.5mA. DHT11 sensor has four pins- VCC, GND, Data Pin and a not connected pin. A pull-up resistor of 5k to 10k ohms is provided for communication between sensor and micro-controller

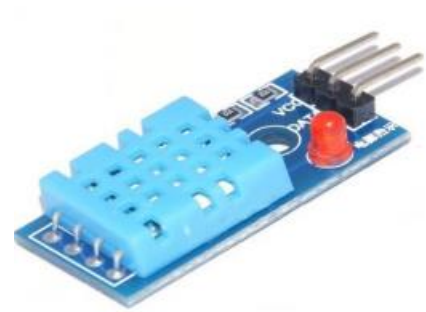

Figure 3.3: DHT11

MAX30102: MAX30102 is a multipurpose sensor used for multiple applications. It is a heart rate monitoring sensor along with a pulse oximeter. The sensor comprises two Light Emitting Diodes, a photodetector, and a series of low noise signal processing devices to detect heart rate and to perform pulse oximetry. It is a 7-pin sensor module with an enabled I2C communication protocol to interact with the microcontroller.

Pin1 (VIN): This pin provides a power supply to the sensor.
Pin2 (SCL): This pin is the I2C serial CLK pin.
Pin3 (SDA): This pin is the I2C serial data pin
Pin4 (INT): This is an interrupt pin that is pulled HIGH through the onboard resistor although when an interrupt takes place, it goes LOW until it clears.
Pin5 (IRD): It is an infrared LED cathode & also a connection point of the LED Driver.
Pin6 (RD): It is a Red LED cathode & connection point of the LED driver.
Pin7 (GND): This is a ground pin and it is connected to the source GND pin.

Figure 3.4: MAX30102

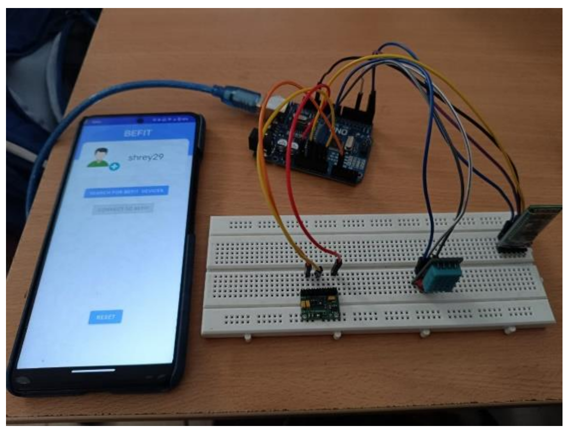

Figure 3.6: Prototype of system

3.2 Software Design

The mobile application was developed using Android studio. After developing the application, we simply load it onto the mobile device, and a link will be provided to download the application. After connecting the Bluetooth device to the application through scanning with the mobile, a connected message will be viewed. Then, after performing the required process, we can show our collected results on the screen.

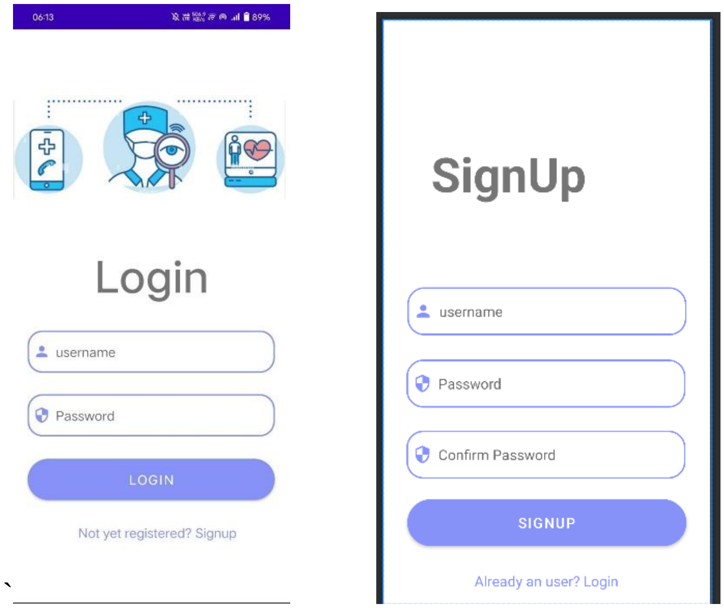

Figure 3.7: Mobile Application

Bluetooth Integration: We made a open socket for Arduino Uno which help our app to find the Arduino Bluetooth and connect HC-05 module to our app and store message inside the Bluetooth using Connect Thread class that will open the BT Socket to the Arduino BT Module, given a BT device, the UUID and a Handler to set the results. When it successfully connected to our app we have open stream which is a connected thread class that given an open BT Socket will open, manage and close the data Stream from the Arduino BT device.

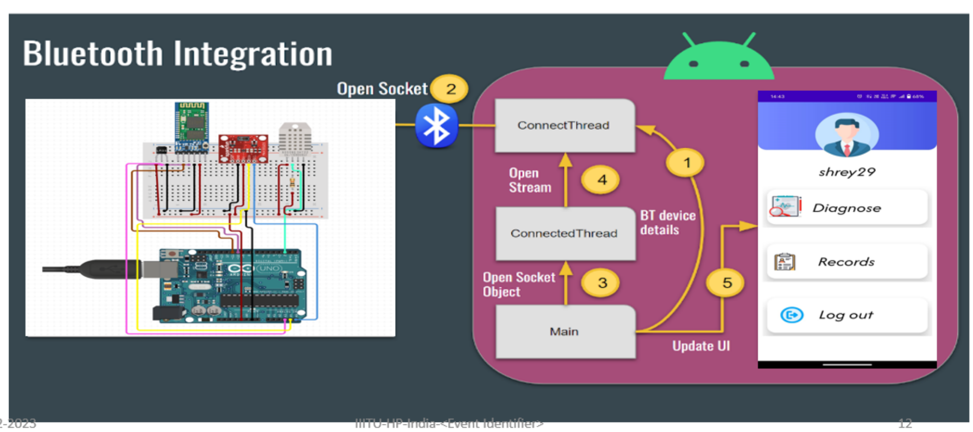

Figure 3.8: Bluetooth Integration Model

Chapter 4

Result and Analysis

4.1 Result and Analysis

The completed system consists of the pulse rate and SpO2 sensors and the body temperature sensor connected to an Arduino Uno. The Arduino is connected to a device with the help of a USB, which will help power up the system. When we upload data to the Arduino, the system starts working, and the measurement data will be shown in the serial monitor of the Arduino Integrated Development Environment (IDE) and the data will also be shown in a mobile application with the help of a Bluetooth module.

Figure show the data of the measurement of all the parameters in the serial monitor of Arduino IDE. The data value is taken from the sensors MAX30100 and DHT11.

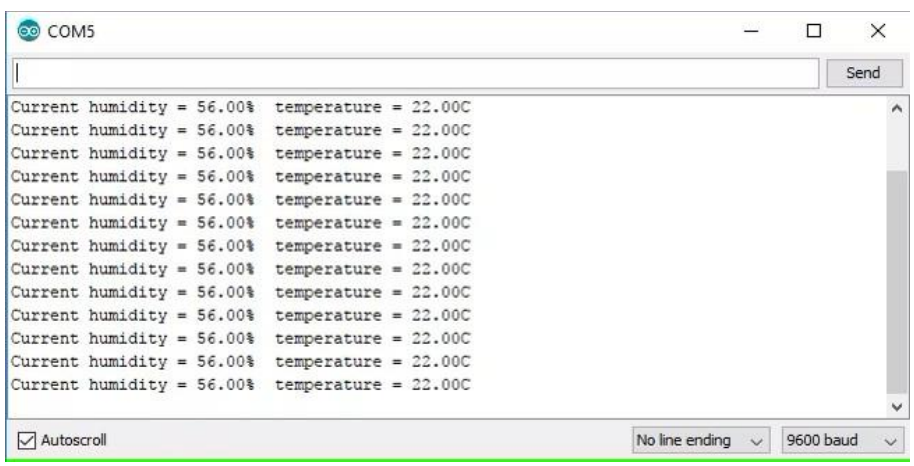

Figure 4.1: Serial Monitor readings

Figure show the data of the measurement of all the parameters in the App. The data value is taken from the sensors MAX30100 and DHT1

Chapter 5

Conclusion

In conclusion, this review of literature not only informs the development of the Health Monitoring System using IoT but also underscores the dynamic and ever-evolving nature of health monitoring technologies. By synthesizing existing knowledge, the project can navigate challenges, capitalize on opportunities, and contribute to the ongoing dialogue surrounding the intersection of IoT and healthcare.

The design and implementation of a health monitoring system using IoT are presented in this study. This IoT-based device allows users to determine their health parameters, which could help regulate their health over time. Eventually, the patients could seek medical assistance if the need arises. They could easily share their health parameter data instantly within one application with the doctor. As we know, the IoT is now considered one of the most desirable solutions in health monitoring. It makes sure that the parameter data is secured inside the cloud, and the most important thing is that any doctor can monitor the health of any patient at any distance. The paper is about an IoT-based health monitoring system using Arduino that has been developed. The system will measure a patient's body temperature, heartbeat, and the SpO2 levels in the blood and send the data to an app via Bluetooth.

5.1 Future work

The gaps in the existing system can be summarized as follows:

- The IoT has the potential to be integrated with a wide variety of devices, which is not possible with most of the systems that are currently in use.
- There is the possibility that the data that are stored will not be protected.

- Complex systems have many disconnects between the various people, stages, and procedures.
- An investigation into the circumstances surrounding an accident will typically reveal the existence of several gaps, but gaps themselves are rarely the cause of accidents.
- The ability to understand and reinforce the normal ability of practitioners in order to bridge gaps contributes to an increase in overall safety.
- The conventional viewpoint, which maintains that systems ought to be shielded from the unreliable influence of humans, is challenged by this point of view.
- We have a limited understanding of how professionals pinpoint newly formed gaps and devise solutions to close them when systems undergo transformation

References

1. J. Wan, M. A. A. H. Al-awlaqi, M. S. Li, M. O. Grady, and X. Gu, "Wearable IoT enabled real-time health monitoring system," *EURASIP Journal on Wireless Communications and Networking*, vol. 298, pp. 1–10, 2018.
2. A. Sharma, A. K. Sing, K. Saxena, and M. A. Bansal, "Smart health monitoring system using IoT," International Journal for Research in Applied Science and Engineering Technology, vol. 8, no. 5, pp. 654–658, 2020.
3. H. T. Yew, M. F. Ng, S. Z. Ping, S. K. Chung, A. Chekima, and J. A. Dargham, "IoT based real-time remote patient monitoring system," in *Proceedings of the 2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, pp. 176–179, Langkawi, Malaysia, Febraury 2020.
4. S. Sengan, O. I. Khalaf, S. Priyadarsini, D. K. Sharma, K. Amarendra, and A. A. Hamad, "Smart healthcare security device on medical IoT using Raspberry Pi," *International Journal of Reliable and Quality E-Healthcare*, vol. 11, no. 3, pp. 1–11, 2021.

Appendices

Appendix A

Code Attachments

A.1 Arduino Code

```
 #include "DHT.h"

#include <Wire.h>

#include "MAX30100_PulseOximeter.h"
#define DHTPIN 7     // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11   // DHT 11
#define REPORTING_PERIOD_MS     1000
PulseOximeter pox;

uint32_t tsLastReport = 0;

DHT dht(DHTPIN, DHTTYPE); //Initialize the DHT component
void onBeatDetected()
{
    Serial.println("Beat!");
}

void setup() {
  Serial.begin(9600);
  Serial.println("DHTxx test!");
   Serial.begin(115200);

    Serial.print("Initializing pulse oximeter..");
     if (!pox.begin()) {
        Serial.println("FAILED");
        for(;;);
    } else {
        Serial.println("SUCCESS");
    }
     pox.setOnBeatDetectedCallback(onBeatDetected);

  dht.begin();
}
```

```
void loop() {
  pox.update();
  // Wait a few seconds between measurements.

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow
sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
        Serial.print("Heart rate:");
        Serial.print(pox.getHeartRate());
        Serial.print("bpm / SpO2:");
        Serial.print(pox.getSpO2());
        Serial.println("%");

        tsLastReport = millis();
    }

  String message = (String) "Humidity: "+h+"% \n Temperature: " +t+"°C";
  Serial.println(message);
  delay(2000);
}
```

A.1 Bluetooth Interfacing Code

- **Connect Thread for Arduino**

```
package com.sarmale.arduinobtexample_v3;

import android.annotation.SuppressLint;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.os.Handler;
import android.util.Log;

import java.io.IOException;
import java.util.UUID;


public class ConnectThread extends Thread {
    private final BluetoothSocket mmSocket;
    private static final String TAG = "FrugalLogs";
    public static Handler handler;
    private final static int ERROR_READ = 0;

    @SuppressLint("MissingPermission")
    public ConnectThread(BluetoothDevice device, UUID MY_UUID,
Handler handler) {

        BluetoothSocket tmp = null;
        this.handler=handler;

        try {

            tmp =
device.createRfcommSocketToServiceRecord(MY_UUID);
        } catch (IOException e) {
            Log.e(TAG, "Socket's create() method failed", e);
        }
        mmSocket = tmp;
    }

    @SuppressLint("MissingPermission")
    public void run() {

        try {

            mmSocket.connect();
        } catch (IOException connectException) {

            handler.obtainMessage(ERROR_READ, "Unable to connect to
the BT device").sendToTarget();
            Log.e(TAG, "connectException: " + connectException);
            try {
                mmSocket.close();
```

```
            } catch (IOException closeException) {
                Log.e(TAG, "Could not close the client socket",
closeException);
            }
            return;
        }


    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) {
            Log.e(TAG, "Could not close the client socket", e);
        }
    }

    public BluetoothSocket getMmSocket(){
        return mmSocket;
    }
}
```

- **Connected Thread For app**

```
• package com.sarmale.arduinobtexample_v3;


import android.bluetooth.BluetoothSocket;
import android.os.Handler;
import android.util.Log;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;


public class ConnectedThread extends Thread {

    private static final String TAG = "Logs";
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;
    private String valueRead;

    public ConnectedThread(BluetoothSocket socket) {
        mmSocket = socket;
        InputStream tmpIn = null;
        OutputStream tmpOut = null;


        try {
            tmpIn = socket.getInputStream();
        } catch (IOException e) {
            Log.e(TAG, "Error occurred when creating input stream",
e);
```

```
        }
        try {
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
            Log.e(TAG, "Error occurred when creating output
stream", e);
        }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public String getValueRead(){
        return valueRead;
    }

    public void run() {

        byte[] buffer = new byte[1024];
        int bytes = 0;
        int numberOfReadings = 0;

        while (numberOfReadings < 1) {
            try {

                buffer[bytes] = (byte) mmInStream.read();
                String readMessage;

                if (buffer[bytes] == '\n') {
                    readMessage = new String(buffer, 0, bytes);
                    Log.e(TAG, readMessage);

                    valueRead=readMessage;
                    bytes = 0;
                    numberOfReadings++;
                } else {
                    bytes++;
                }

            } catch (IOException e) {
                Log.d(TAG, "Input stream was disconnected", e);
                break;
            }
        }

    }


    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) {
            Log.e(TAG, "Could not close the connect socket", e);
        }
    }
}
```