FACULTY OF ENGINEERING

UNIVERSITY OF JAFFNA

EC6020

EMBEDDED SYSTEMS

GROUP PROJECT

FINAL PROJECT REPORT

DESIGN AND DEVELOPMENT OF E-HEALTH MONITORING SYSTEM USING IOT AND SENSORS

SUBMITTED BY: GROUP CG-07

2020/E/069 KARUNASENA N.W.S.

2020/E/070 KEERTHIKAN F.J.

2020/E/074 KOKILARAJ A.

Table of Contents

1.0 INTRODUCTION

The rapid advancement of wireless technology, particularly the Internet of Things (IoT), has revolutionized various sectors, notably industrial automation, healthcare, and biomedicine. IoT has enabled remote monitoring, allowing medical professionals to monitor patients using devices that provide comparable health outcomes to traditional in-person visits, increase patient satisfaction and reduce costs.

Problem Statement

Traditional health check-ups require patients to visit healthcare facilities regularly, which can be time-consuming and costly. There is a need for a system that can monitor patients' vital health parameters in real-time, providing continuous updates to both medical professionals and patients.

Solution

To address these inefficiencies, a prototype for real-time health monitoring using IoT has been developed. This system utilizes non-intrusive sensors to record vital health parameters like body temperature, heart rate, ECG signals, and SPO2 levels. The gathered data is transmitted to the IoT cloud, accessible to physicians for remote diagnosis and treatment. Patients and their close family members can also access their medical records through this cloud service, receiving regular updates as well. Overall, this project demonstrates the potential of IoT in revolutionizing healthcare delivery by enabling remote monitoring and diagnosis.

2.0 PROJECT DESIGN AND IMPLEMENTATION

2.1 Hardware Design

The hardware design consists of various sensors interfaced with a NodeMCU-ESP32 microcontroller. The sensors include:

- **DS18B20**: Temperature sensor
- **MAX30100**: Heart rate and SPO2 sensor
- **AD8232**: ECG sensor

The data collected from these sensors are sent to the NodeMCU, which processes and uploads the data to the cloud.

Block Diagram

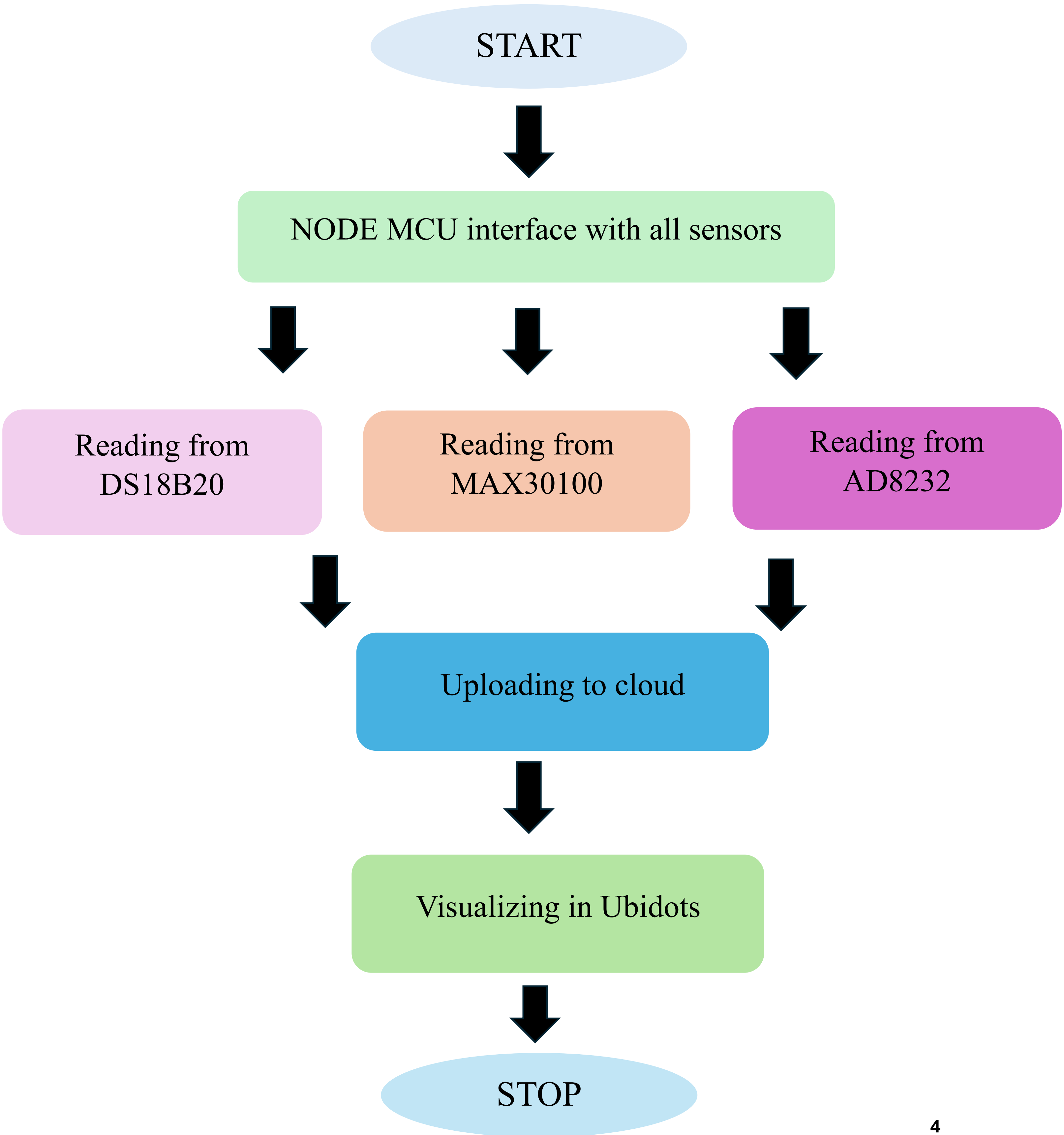


Figure 01: Block Diagram

Hardware Model

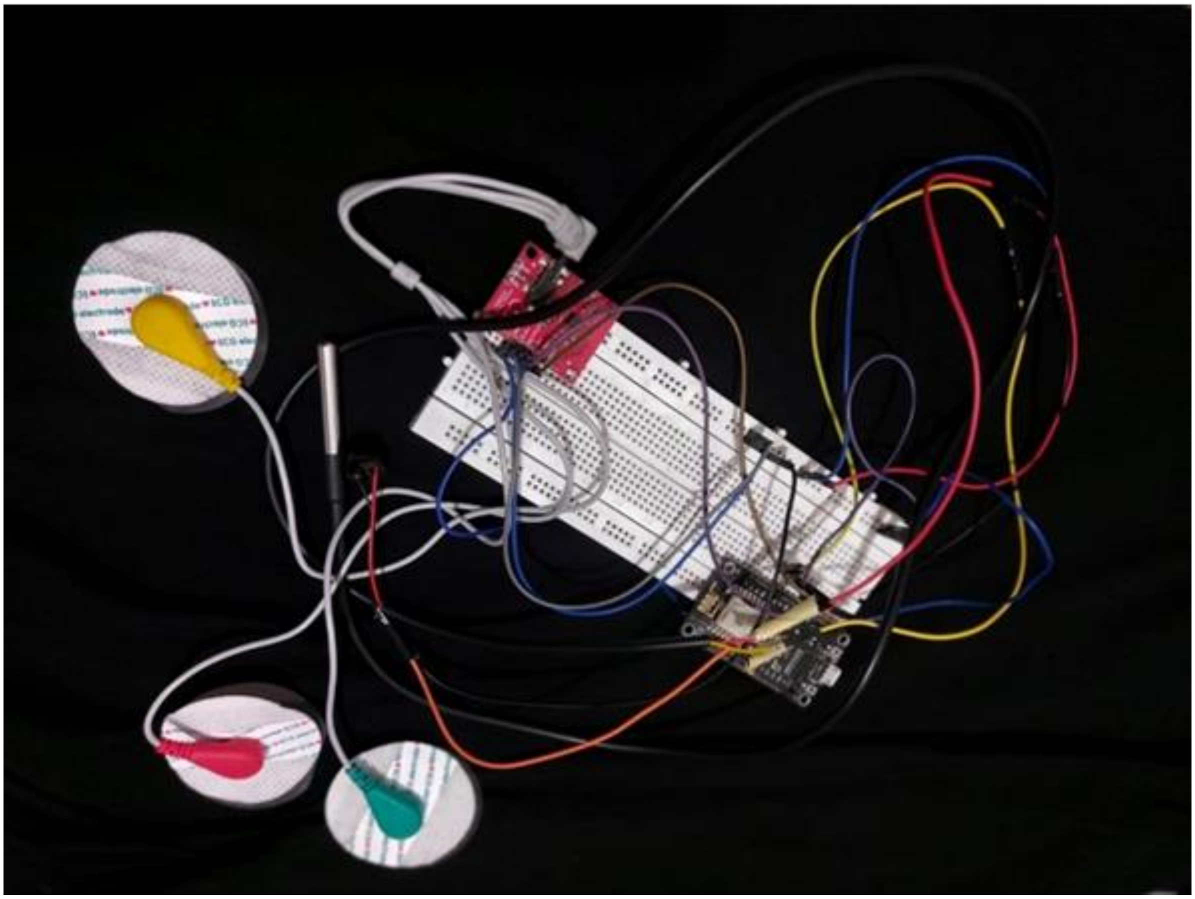


Figure 02: Hardware Model

Schematic Diagram

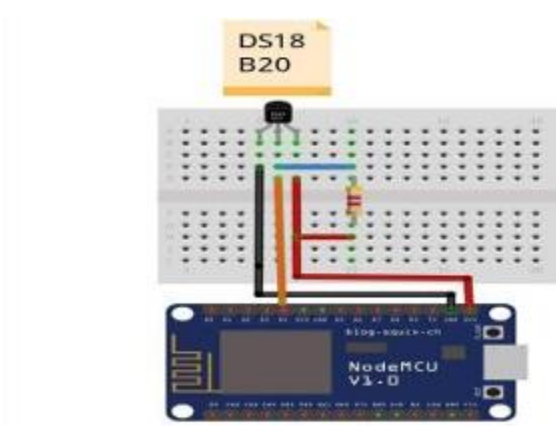


Figure 03: Schematic Diagram

2.2 Interfacing

Each sensor is connected to the NodeMCU-ESP8266 using appropriate pins and protocols. The DS18B20 uses a single-wire protocol, the MAX30100 uses I2C, and the AD8232 uses analog input pins for ECG signal acquisition.

Sensor Connections

- **DS18B20**: Connected to digital pin with a pull-up resistor
- **MAX30100**: Connected to I2C pins (SDA, SCL)
- **AD8232**: Connected to analog input pin

2.3 Software Design

The software design involves developing firmware for the NodeMCU to read data from the sensors and transmit it to the cloud. Additionally, a web-based server IS developed for data visualization.

Firmware Development

The firmware for the NodeMCU is developed using the Arduino IDE. The code includes libraries for each sensor, functions to read data, and code to upload data to the IoT cloud.

Ubidots

Ubidots is an IoT platform that enables users to visualize and analyze data collected from various devices and sensors. It provides a user-friendly interface for creating dashboards, widgets, and alerts. In this project, Ubidots is used to visualize the health data collected from the sensors. The NodeMCU uploads the data to Ubidots, where it is processed and displayed in real-time, allowing both medical professionals and patients to monitor the health parameters effectively.

Web Server

A web-based server is created to visualize the sensor data in real-time. This server can be accessed by medical professionals and patients.

2.4 Implementation

The implementation involves assembling the hardware components, developing the necessary software, and integrating the system. The system is tested to ensure it functions correctly and reliably.

Detailed Sensor Information

Temperature Sensor DS18B20

- **One Wire Communication**: The DS18B20 sensor uses a one-wire communication protocol. This means it requires only one data line (and ground) to communicate with the microcontroller. A resistor is placed between the V_{CC} (power) and the signal pin to pull the line high by default.
- **Digital Sensor**: It is a digital sensor with a 12-bit Analog-to-Digital Converter (ADC) embedded inside, providing precise temperature readings.
- **Temperature Range**: The sensor can measure temperatures ranging from -55°C to 125°C.
- **Temperature Alarm**: It has an in-built temperature limit alarm system, which can alert when the temperature crosses 37 ^{0}C limit.

ECG Sensor Module AD8232

- **Electrode Connection**: The AD8232 ECG sensor module connects to the body via electrodes. These electrodes are typically placed on the chest to detect electrical signals.
- **Signal Detection**: When the heart pumps oxygenated and deoxygenated blood, it generates electrical signals. These signals are of very low power and not noticeable by humans.
- **Conductive Gel**: The gel inside the electrodes enhances conductivity, allowing these low-power electrical signals to be detected.
- **Signal Amplification**: The sensor amplifies these weak signals to provide analog values representing the ECG, which can then be plotted or analyzed.

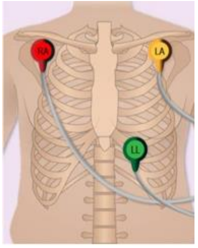


Figure 04: ECG Electrodes

Heart Rate and SPO2 Sensor MAX30100

- **Integrated Pulse Oximetry and Heart Rate Monitoring**: The MAX30100 sensor is an integrated pulse oximetry and heart rate monitor module. It uses photodetectors, optical elements, and low-noise electronics with ambient light rejection to accurately detect pulse oximetry (SPO2) and heart rate signals.
- **Dual LED Operation**: The sensor operates using two LEDs (Red and Infrared) to measure the oxygen saturation in the blood. The ratio of absorption of red and infrared light changes with the amount of oxygenated and deoxygenated blood.
- **I2C Communication**: It communicates with the microcontroller using the I2C protocol, making it easy to interface with the NodeMCU-ESP8266.
- **Low Power Operation**: Designed for low power operation, it is suitable for battery-operated devices, making it ideal for wearable health monitoring systems.

Buzzer

- **Temperature Alert**: The system includes a buzzer that will sound an alarm when the detected temperature exceeds 37°C, providing an immediate alert for potential fever conditions.

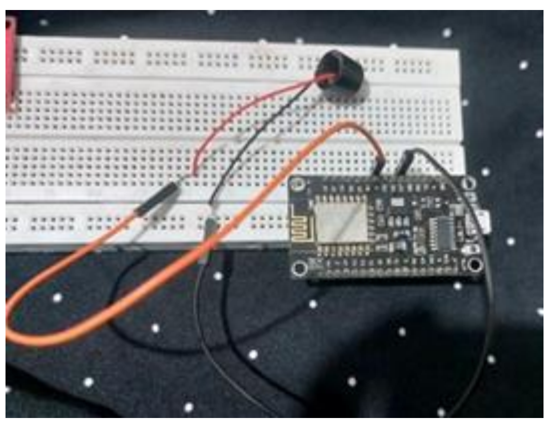


Figure 05: Circuit With Buzzer

Simulation

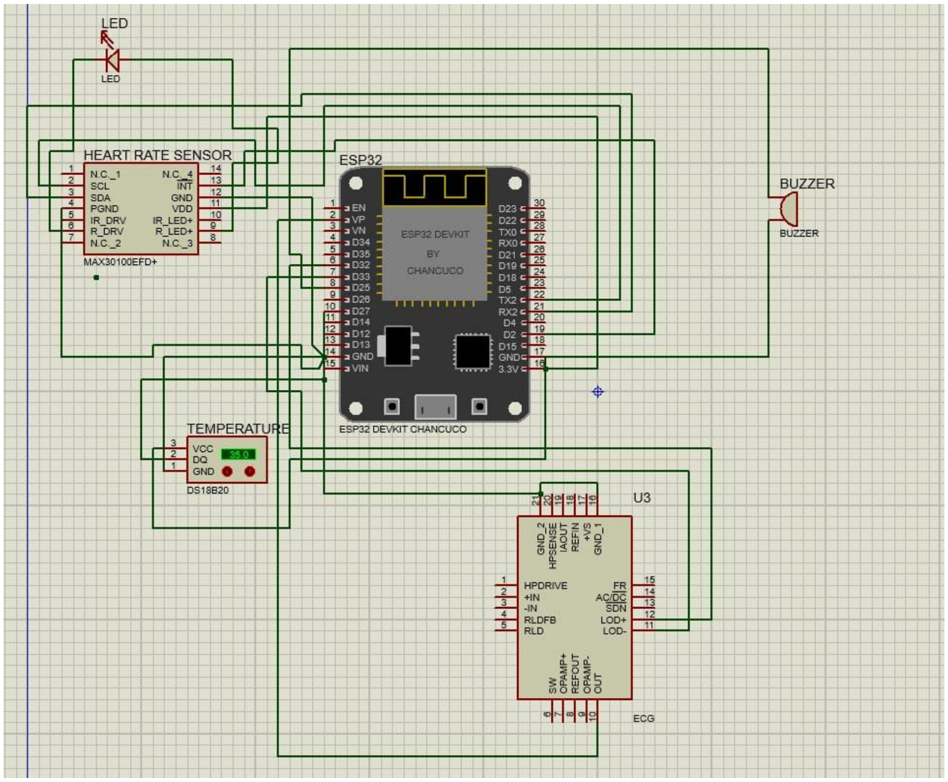


Figure 06: Simulation

Ubidots Outputs

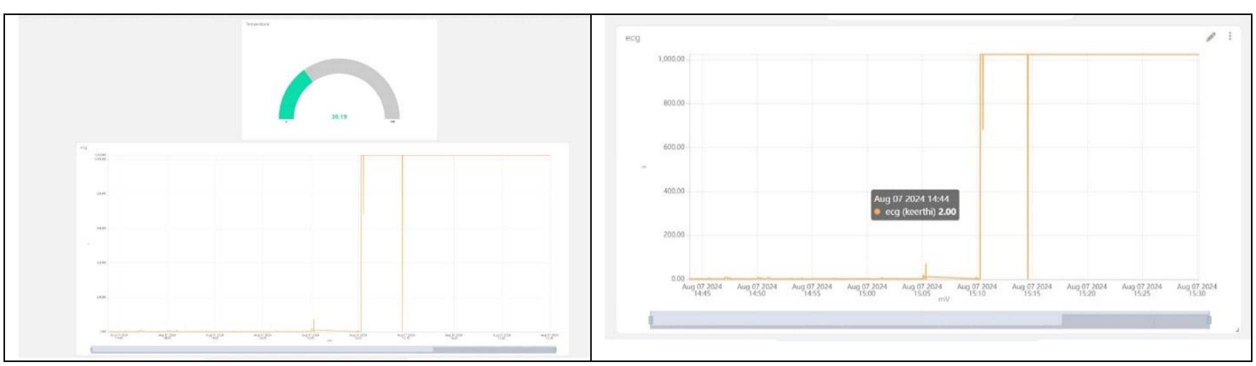


Figure 07: Ubidots Outputs

3.0 CHALLENGES AND SOLUTIONS

3.1 Challenges

- **Sensor Accuracy**: Ensuring the sensors provide accurate and reliable data.
- **Data Transmission**: Maintaining reliable and secure data transmission to the cloud.
- **User Interface**: Developing an intuitive and user-friendly interface for the web server and mobile application.
- **Design flaw with MAX30100 sensor**: This sensor had a design flaw where the voltage output by the nodemcu was not enough to run the Sensor. So we had to remove a resistor and short circuit it.

3.2 Solutions

- **Calibration**: Regular calibration of sensors to maintain accuracy.
- **Encryption**: Using encryption techniques to secure data transmission.
- **User Testing**: Conducting user testing to improve the interface based on feedback.

4.0 TIMELINE

Task Name	7th week	8th week	9th week	10th week	11th week	12th week	13th week	14th week
Project proposal	✓							
Literature Review		✓						
Hardware Research			✓					
Sensor Selection			✓					
Firmware Development				✓				
Web Server Development				✓				
Integration					✓			
Testing						✓		
Final Report and Poster Preparation							✓	✓

5.0 COMPONENTS AND COST

Hardware Requirement	
Component	**Cost (Rs.)**
NodeMCU – ESP32	1800
Temperature sensor – DS18B20	200
Heart rate and SPO2 sensor- MAX30100	1000
ECG sensor- AD8232 sensor	1900
Three lead electrodes	500
Breadboard	400
Jumper wires	480(1 pcs 12/=)
Power source(9v battery)	300
Software Requirement	
Arduino IDE	-
Total	6580

6.0 REFLECTION

Embedded Systems

The knowledge of embedded systems was crucial in the development of this project. Understanding how microcontrollers function, their architecture, and how to program them formed the foundation of this project. The NodeMCU-ESP8266 was used as the core microcontroller, which required:

- **Programming Skills**: Writing firmware in the Arduino IDE using C++ was essential. This involved implementing libraries for different sensors and managing memory efficiently.
- **Interfacing Sensors**: Each sensor (DS18B20, MAX30100, AD8232) had specific interfacing requirements. For example, the DS18B20 required a one-wire protocol, while the MAX30100 used I2C. Understanding these protocols was vital to ensure proper communication and data acquisition.

Internet of Things (IoT)

The IoT course provided a deep understanding of how to connect devices to the internet and manage data in the cloud. This knowledge was applied in:

- **Data Transmission**: The project involved sending sensor data to the cloud using the NodeMCU-ESP8266's Wi-Fi capabilities. Learning how to establish and maintain a stable connection was crucial.
- **Cloud Platforms**: Utilizing Ubidots for data visualization required understanding how to send data to the cloud, create dashboards, and configure widgets to display real-time health data effectively.
- **Security**: Ensuring secure data transmission was a key aspect. Applying encryption techniques learned in the course helped protect sensitive health information from potential breaches.

Software Development

Skills in software development were applied extensively throughout the project:

- **Firmware Development**: Writing efficient, bug-free code for the NodeMCU to read data from sensors and upload it to the cloud was a primary task. This included error handling and ensuring the system could recover from connectivity issues.

- **Web Development**: Creating a web-based server for visualizing sensor data required knowledge of web technologies such as HTML, CSS, JavaScript, and server-side programming. The course provided the necessary skills to develop a responsive and user-friendly interface.

Biomedical Sensors

The project required specific knowledge about biomedical sensors, which was covered in the course:

- **Sensor Characteristics**: Understanding the operational principles of sensors like the DS18B20, MAX30100, and AD8232 allowed for proper integration and data interpretation.
- **Signal Processing**: For the ECG sensor (AD8232), it was important to know how to process the analog signals to extract meaningful information, such as heart rate and ECG patterns.

Project Management

Managing the project efficiently involved applying project management techniques learned in the course:

- **Work Breakdown Structure (WBS)**: Breaking down the project into manageable tasks and assigning them to team members helped streamline the workflow.
- **Timeline Management**: Creating and adhering to a timeline ensured that the project stayed on track and deadlines were met. This involved scheduling tasks, setting milestones, and adjusting plans as needed.
- **Collaboration Tools**: Using tools like Git for version control and Trello for task management facilitated smooth collaboration among team members.

Ethical and Legal Considerations

The course also emphasized the importance of ethical and legal considerations in healthcare projects:

- **Data Privacy**: Ensuring patient data privacy was paramount. The project implemented measures to comply with regulations like HIPAA, ensuring that sensitive health data was protected.
- **User Consent**: Gaining informed consent from users before collecting and using their health data was a critical ethical consideration, ensuring transparency and trust.

7.0 CONCLUSION

The e-health monitoring system using IoT and sensors successfully demonstrates the transformative potential of the Internet of Things (IoT) in the healthcare industry. This project integrates multiple sensors and IoT technologies to enable real-time monitoring of vital health parameters, offering significant benefits for patients, healthcare providers, and the healthcare system as a whole.

The system effectively monitors vital health parameters such as body temperature, heart rate, ECG signals, and SPO2 levels in real-time. By using sensors like DS18B20 for temperature, MAX30100 for heart rate and SPO2, and AD8232 for ECG, the system provides accurate and reliable data. With the data being uploaded to the cloud, healthcare providers can remotely access and monitor patient health, enabling timely and informed medical interventions. Patients benefit from continuous health monitoring without the need for frequent hospital visits, enhancing convenience and reducing healthcare costs. The integration with Ubidots allows for the visualization of data in an accessible and user-friendly manner, aiding both patients and healthcare professionals in understanding health trends and making data-driven decisions.

The project proves the feasibility of using affordable, off-the-shelf components to create a functional and reliable e-health monitoring system. NodeMCU-ESP8266, a cost-effective microcontroller with Wi-Fi capabilities, serves as the core of the system, demonstrating how low-cost hardware can be utilized for advanced healthcare applications. The successful implementation of secure data transmission protocols ensures patient data privacy and meets legal and ethical standards.

Throughout the project, several challenges were encountered and addressed. Sensor interfacing and calibration required specific procedures to ensure accurate data collection, achieved through careful study of datasheets and application of appropriate protocols. Ensuring reliable wireless communication and data integrity involved dealing with connectivity issues and implementing robust error handling. Despite these challenges, the project highlights the feasibility and practicality of integrating IoT into healthcare systems, paving the way for future advancements in remote health monitoring and smart healthcare solutions.

8.0 REFERENCES

- **Internet of Things (IoT)**: Raj Kamal, "Internet of Things: Architecture and Design Principles," McGraw Hill Education, 2017.
- **Embedded Systems**: Muhammad Ali Mazidi, Sarmad Naimi, and Sepehr Naimi, "The AVR Microcontroller and Embedded Systems Using Assembly and C," Pearson Education, 2014.
- **Arduino IDE**: Official website and documentation. Available at: https://www.arduino.cc/en/Guide/Environment
- **Ubidots**: Official documentation and tutorials. Available at: https://help.ubidots.com/en/
- **Instructables**: "IoT Based Patient Health Monitoring System using ESP8266 and Arduino." Available at: https://www.instructables.com/IoT-Based-Patient-Health-Monitoring-System-using-E/

9.0 APPENDIX

9.1 Code

Arduino Code

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include "UbidotsESPMQTT.h"

#define WIFISSID "Keerthiii" // WiFi SSID
#define PASSWORD "qwerty12345678" // WiFi password
#define TOKEN "BBUS-t3FPlKXprOMiaoTgH0RKeqao7hS7vp" // Ubidots TOKEN
#define MQTT_CLIENT_NAME "1234a5d6798" // MQTT client name
#define VARIABLE_LABEL "ecg" // Variable label
#define DEVICE_LABEL "keerthi" // Device label
#define SENSOR A0 // Analog sensor connected to A0
#define LO_MINUS D5 // LO- connected to pin D5
#define LO_PLUS D6 // LO+ connected to pin D6
#define DATA_PIN D4 // Pin where the DS18B20 data line is connected
#define BUZZER_PIN D7 // Pin connected to the buzzer

char mqttBroker[]  = "industrial.api.ubidots.com";
char payload[100];
```

```
char topic[150];
char str_sensor[10];
float temp;

OneWire ourWire(DATA_PIN);
DallasTemperature sensors(&ourWire);
WiFiClient ubidots;
Ubidots ubidotsClient(TOKEN, MQTT_CLIENT_NAME);
PulseOximeter pox;
bool max30100_initialized = false;

void onBeatDetected() {
  Serial.println("Beat!");
}

void callback(char* topic, byte* payload, unsigned int length) {
  // Handle callback
}

void reconnect() {
  while (!ubidotsClient.connected()) {
    Serial.println("Attempting MQTT connection...");
    if (ubidotsClient.connect()) {
      Serial.println("Connected");
    } else {
```

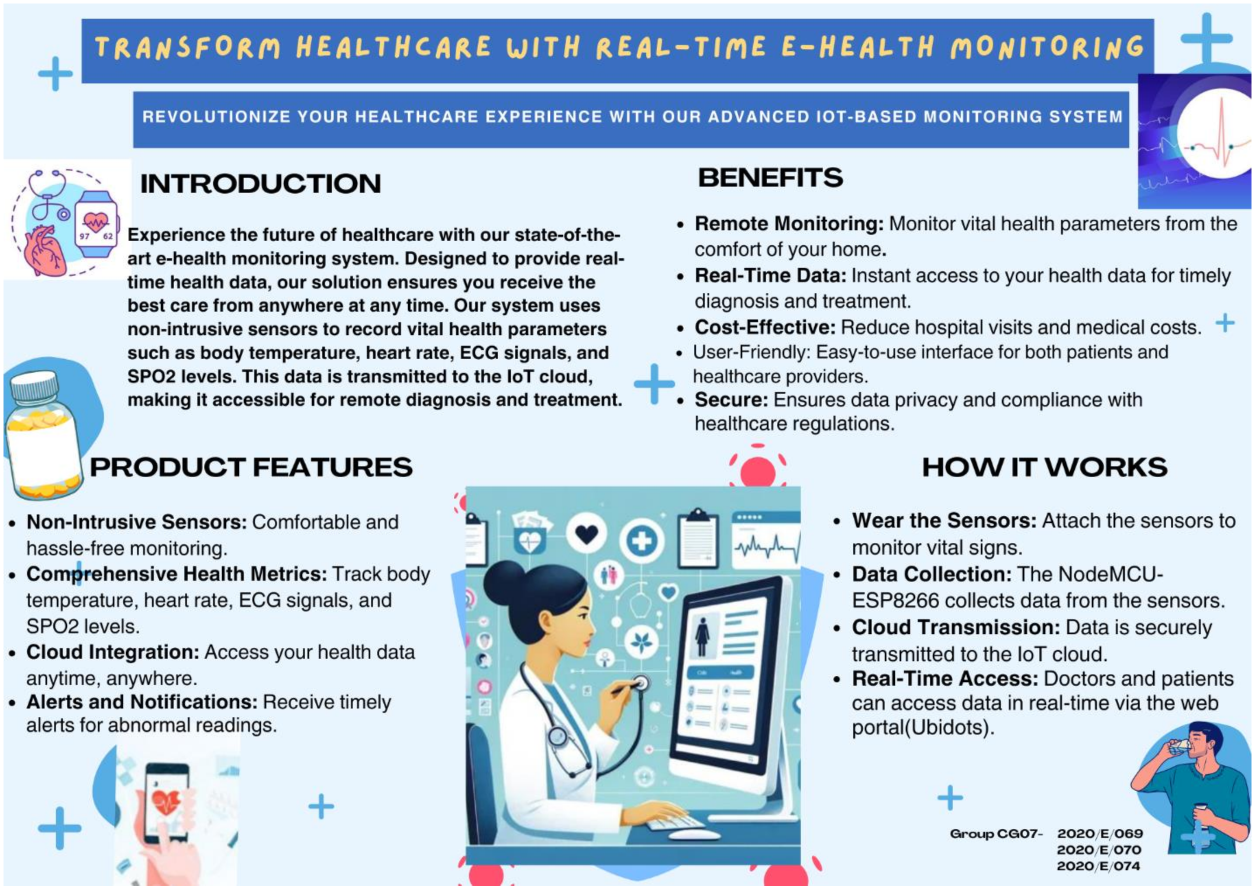
```
      Serial.print("Failed to connect to Ubidots, try again in 2 seconds");
      delay(2000);
    }
  }
}

void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFISSID, PASSWORD);
  pinMode(SENSOR, INPUT);
  pinMode(LO_MINUS, INPUT);
  pinMode(LO_PLUS, INPUT);
  pinMode(BUZZER_PIN, OUTPUT); // Set buzzer pin as output
  sensors.begin();

  Serial.println();
  Serial.print("Waiting for WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi Connected");
  Serial.println("IP address: ");
```

```
  Serial.println(WiFi.localIP());

  ubidotsClient.setDebug(true); // Enable debug prints to serial monitor

  ubidotsClient.wifiConnection(WIFISSID, PASSWORD);
  ubidotsClient.begin(callback);

  // Initialize the MAX30100 sensor
  if (pox.begin()) {
    max30100_initialized = true;
    pox.setOnBeatDetectedCallback(onBeatDetected);
    Serial.println("MAX30100 initialized successfully.");
  } else {
    Serial.println("MAX30100 failed to initialize. Continuing without it.");
  }
}

void loop() {
  if (!ubidotsClient.connected()) {
    reconnect();
  }

  sprintf(topic, "%s%s", "/v1.6/devices/", DEVICE_LABEL);
  sprintf(payload, "%s", ""); // Clean the payload
  sprintf(payload, "{\"%s\":", VARIABLE_LABEL); // Add the variable label
```

```
  float sensor = analogRead(SENSOR);
  dtostrf(sensor, 4, 2, str_sensor);

  sprintf(payload, "%s {\"value\": %s}}", payload, str_sensor); // Add the value
  Serial.println("Publishing data to Ubidots Cloud");
  ubidotsClient.add(VARIABLE_LABEL, sensor); // Insert your variable label and the
value to be sent

  sensors.requestTemperatures();
  temp = sensors.getTempCByIndex(0);
  ubidotsClient.add("temp", temp); // Insert your temperature variable label and value

  if (temp > 37.0) {
    digitalWrite(BUZZER_PIN, HIGH); // Activate the buzzer
  } else {
    digitalWrite(BUZZER_PIN, LOW); // Deactivate the buzzer
  }

  // Read data from the MAX30100 sensor if initialized
  if (max30100_initialized) {
    pox.update();
    float BPM = pox.getHeartRate();
    float SPO2 = pox.getSpO2();
```

```
    ubidotsClient.add("heart_rate", BPM);
    ubidotsClient.add("spo2", SPO2);
  } else {
    Serial.println("Skipping MAX30100 data as it is not initialized.");
  }

  ubidotsClient.ubidotsPublish(DEVICE_LABEL); // Publish the data to Ubidots under device name 'keerthi'

  delay(500);
  Serial.println("Loop end");
}
```

9.2 Poster


TRANSFORM HEALTHCARE WITH REAL-TIME E-HEALTH MONITORING
REVOLUTIONIZE YOUR HEALTHCARE EXPERIENCE WITH OUR ADVANCED IOT-BASED MONITORING SYSTEM
INTRODUCTION
Experience the future of healthcare with our state-of-the-art e-health monitoring system. Designed to provide real-time health data, our solution ensures you receive the best care from anywhere at any time. Our system uses non-intrusive sensors to record vital health parameters such as body temperature, heart rate, ECG signals, and SPO2 levels. This data is transmitted to the IoT cloud, making it accessible for remote diagnosis and treatment.
BENEFITS
Remote Monitoring: Monitor vital health parameters from the comfort of your home.
Real-Time Data: Instant access to your health data for timely diagnosis and treatment.
Cost-Effective: Reduce hospital visits and medical costs.
User-Friendly: Easy-to-use interface for both patients and healthcare providers.
Secure: Ensures data privacy and compliance with healthcare regulations.
PRODUCT FEATURES
Non-Intrusive Sensors: Comfortable and hassle-free monitoring.
Comprehensive Health Metrics: Track body temperature, heart rate, ECG signals, and SPO2 levels.
Cloud Integration: Access your health data anytime, anywhere.
Alerts and Notifications: Receive timely alerts for abnormal readings.
HOW IT WORKS
Wear the Sensors: Attach the sensors to monitor vital signs.
Data Collection: The NodeMCU-ESP8266 collects data from the sensors.
Cloud Transmission: Data is securely transmitted to the IoT cloud.
Real-Time Access: Doctors and patients can access data in real-time via the web portal(Ubidots).
Group CG07- 2020/E/069
2020/E/070
2020/E/074