UNIVERSITY OF NAIROBI

FACULTY OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

FINAL YEAR PROJECT REPORT

PROJECT INDEX: PRJ 141

IOT BASED HEART MONITORING SYSTEM USING ECG

STUDENT NAME**: SHARNIZE PASQOL OWINO**

REGISTRATION NO.**: F17/1776/2019**

SUPERVISOR**: DR. A. NYETE**

EXAMINER**: DR. G. OMONDI**

This Project Report is submitted in partial fulfilment of the requirement for the award of the degree of Bachelor of Science in Electrical and Information Engineering at the University of Nairobi.

SUMBITTED ON 25TH JUNE 2024

DECLARATION OF ORIGINALITY

Name of Student	Sharnize Pasqol Owino
Registration Number	F17/1776/2019
College	College of Architecture and Engineering
Faculty/School/Institute	Engineering
Department	Electrical and Information Engineering
Course Name	Engineering Projects
Course Code	FEE 560
Title of the Work	IOT Based Heart Monitoring System Using ECG

Declaration

1. I am aware of what plagiarism is and I know that University's policy regarding plagiarism.
2. I do declare that this project is the original work I did and has not been submitted elsewherefor examination, award of a degree or publication. I have acknowledged and referenced the other people's work that has been used to bolster my understanding of the subject in accordance with the University of Nairobi requirements.
3. I declare that I have not sought or used any services of professional agencies in the implementation of the work done.
4. I have not allowed, and will never give anyone a copy of the work that I have done with the intent of them passing it off as their own work.
5. I do understand that any false claim in respect to this work shall result in disciplinary actionin accordance with the University of Nairobi Plagiarism Policy.

Signature: ..... **Date** ……...

Approved by:

Supervisor: DR A. NYETE

Signature: ..... **Date**……...

ACKNOWLEDGEMENT

The completion of this real-time heart monitoring project would not have been possible without the invaluable support and guidance of many individuals. First and foremost, I would like to express my deepest gratitude to my supervisor, Dr A. Nyete, whose encouragement and constructive feedback were instrumental in shaping this project.

My heartfelt thanks go to my colleagues and peers, who offered their time and expertise to review my work and provided valuable feedback that, improved the quality and accuracy of the project. I am also grateful to my family and friends for their unwavering support and encouragement throughout this journey, which motivated me to persevere through challenges and achieve my goals.

ABSTRACT

This project presents the development of a real-time ECG (electrocardiogram) monitoring system utilizing an ESP32 microcontroller, which interfaces with an AD8232 ECG sensor to capture heart activity. The system uses Wi-Fi connectivity to transmit data to Ubidots, an IoT cloud platform, enabling remote monitoring and analysis. The ESP32 reads the analog ECG signals, processes these readings, and timestamps each data point using Network Time Protocol (NTP) for precision. These data points are then sent via MQTT (Message Queuing Telemetry Transport) to Ubidots, where they are stored and visualized in real-time dashboards. The implementation phase employs the Arduino IDE's serial plotter for immediate, graphical feedback of the sensor data, facilitating debugging and calibration. Ubidots enhances the system's functionality by providing robust data visualization, historical analysis, alert notifications, and remote access, which are crucial for effective healthcare monitoring. This project demonstrates the integration of hardware and cloud-based solutions to create a comprehensive, real-time ECG monitoring system that can aid in early detection of cardiac anomalies and support continuous health tracking. The modular design ensures adaptability for various medical applications, offering a scalable solution for remote patient monitoring and telemedicine.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AC: Alternating Current

AFE: Analogue Front End

BPM: Beats Per Minute

CVDs: Cardiovascular Diseases

DC: Direct Current

ECG: Electrocardiogram

GND: Ground

GSM: Global System for Mobile Communication

HTTP: Hyper Text Transfer

MQTT: Message Queuing Telemetry Transport

NTP: Network Time Protocol

IDE: Integrated Development Environment

IOT: Internet of Things

IP: Internet Protocol

LED: Light Emitting Diode

OLED: Organic Light Emitting Diode

PCB: Printed Circuit Board

PD: Photo Diode

PPG: Photoplethysmography

PWM: Pulse Width Modulation

RLD: Right-leg Driver

RSSI: Received Signal Strength

SCL: Serial Clock

SDA: Serial Data

SSID: Service Set Identifier

WBASNs: Wireless Body Area Sensor Network

1. CHAPTER 1: BACKGROUND.

Over the past few decades, heart diseases have become a major concern, and many people pass away from related illnesses. Heart disease should therefore not be taken lightly. Over the past ten years, chronic and cardiovascular diseases (CVDs) have been the leading cause of death in every nation. Ailments that impact the heart and blood vessels are known as cardiovascular diseases. Cardiovascular diseases (CVDs) include illnesses of the blood vessels, such as coronary artery diseases. Heart-related conditions include arrhythmias, heart failure, cardiomyopathy, rheumatic heart disorders, stroke, and heart attack. According to the World Health Organization (WHO), cardiovascular diseases account for 17.9 million deaths annually, making them the primary cause of death worldwide.

In the present-day healthcare, immediate heartbeat detection and continuous heart rate monitoring are crucial priorities. The purpose of this research is to develop a new, comprehensive paradigm for the assessment of cardiovascular diseases and to support disease control and prevention through the use of Internet of Things (IoT) monitoring of electrocardiogram (ECG) signals. The process of creating an electrocardiogram, also known as an EKG or ECG, or electrocardiography, involves repeatedly recording the electrical activity of the heart through cardiac cycles. This is a heart electrogram, which uses electrodes applied to the skin to create a graph of voltage versus time for the electrical activity of the heart. These electrodes pick up the minute electrical alterations brought about by the depolarization and repolarization of the heart muscle during each cardiac cycle (heartbeat). Numerous cardiac abnormalities can cause changes in the normal ECG pattern, such as electrolyte imbalances (hypo- or hyperkalemia), cardiac rhythm disturbances (atrial fibrillation, ventricular tachycardia), and inadequate coronary artery blood flow (myocardial ischemia, myocardial infarction).

It is possible to wear an ECG monitoring device that sends data wirelessly and directly, without the need for a mobile terminal, to the Internet of Things cloud. Broader coverage and faster data speeds are two advantages of Wi-Fi over Bluetooth or Zigbee technology. The IoT cloud provides data services that physicians and patients can easily access with smartphones running different operating systems thanks to a web-based graphical user interface. Electrocardiogram Internet of Things (IoT) works are numerous. Now more than ever, the risk of impairments can

be effectively managed thanks to advancements in sensor technology, communication infrastructure, data processing, modelling, and analytics algorithms. Because the traditional healthcare mode is passive, by which patients call the healthcare service by themselves. Consequently, they usually fail to call the service if they are unconscious when the heart disease attacks.

The care of patients with heart disease is a problem that the Internet of Things (IoT) techniques vastly outperform because they can alter the mode of service in a way that is widespread and start providing healthcare based on the physical state of the patient rather than their emotional state. An essential component of the ubiquitous healthcare service is a remote monitoring system. The information gathered can be sent to the doctors who work remotely at a minimal cost, guaranteeing that these professionals are always and instantly informed about their patients' physical conditions. Over the past few decades, ECG monitoring systems have been developed and extensively utilized in the healthcare industry. As smart enabling technologies have emerged, these systems have undergone significant evolution.

1.1 OBJECTICVES.

1.1.1. Overall objective:

• To design, develop, and implement an Internet of Things (IoT)-based cardiac monitoring system that uses electrocardiogram (ECG) technology to monitor cardiac activity in real-time and provide timely alerts and information for improved healthcare.

1.1.2. Specific objectives:

Chapter1 To design system architecture for a wearable IoT-based heart monitoring system, specifying the hardware components, communication protocols, and data flow.

Chapter2 To integrate ECG sensors with the IoT platform to capture real-time electrocardiographic data accurately.

Chapter3 To implement a robust wireless communication system for seamless transmission of ECG data from the monitoring device to the cloud-based server.

Chapter4 To set up a secure and scalable cloud-based storage system to store and manage the ECG data, ensuring accessibility and reliability.

Chapter5 To implement algorithms for real-time processing of ECG data to extract relevant features and detect anomalies or irregularities in cardiac activities.

Chapter6 To develop an alert system that triggers notifications in case of abnormal ECG patterns, ensuring timely communication to healthcare providers or emergency contacts.

Chapter7 To optimize the power consumption of the monitoring device to ensure long battery life, especially for continuous and prolonged usage.

1.2 PROBLEM STATEMENT.

The need of creating an advanced and effective heart monitoring system is highlighted by the rising incidence of cardiovascular illnesses and the requirement for ongoing cardiac monitoring. Many of the monitoring techniques used now are not real-time and are not easily incorporated into healthcare systems. Conventional Electrocardiogram (ECG) equipment are usually limited to use in clinical settings, which makes it difficult for them to continuously monitor people with cardiac issues in daily life. Furthermore, the lack of a dependable, user-friendly, Internet of Things-based cardiac monitoring system makes it difficult to detect cardiac abnormalities in a timely manner and presents obstacles for preventive healthcare intervention.

The conventional systems are often characterized by:

• Limited Mobility: Real-world cardiac activity capture is hampered by the immobility of current ECG monitoring devices, resulting in it becoming difficult to do continuous, discrete monitoring in a range of contexts.

• Delayed Anomaly Detection: The incomplete processing and analysis of ECG data causes a delay in the identification of important cardiac events, which might compromise patient outcomes by delaying medical treatments.

• Lack of Seamless Integration: Insufficient integration with electronic health records and healthcare systems prevents thorough patient data analysis and hinders healthcare practitioners' ability to work together to manage cardiac problems.

• Privacy and Security Concerns: Patient confidentiality may be compromised by current systems' inadequate handling of privacy and security issues related to the transmission and storage of sensitive ECG data.

The creation of a cutting-edge IoT-based heart monitoring system that combines revolutionary ECG technology with a safe, scalable, and user-friendly platform is of vital importance to address these issues. A system like this would make it easier to monitor patients in real time, identify cardiac abnormalities in a timely manner, and integrate with healthcare ecosystems seamlessly. This would improve the overall management of cardiovascular health for people in a variety of settings.

1.3 SCOPE OF THE PROJECT.

The project will mainly focus on designing and developing a portable and wearable ECG monitoring device equipped with sensors for accurate data capture, implementing a reliable and secure wireless communication to facilitate transmission of real-time ECG data from the monitoring device to a cloud-based server, developing algorithms for real-time processing of ECG data to extract meaningful features and detect anomalies or irregularities in cardiac activities and optimization of the power consumption of the monitoring device to ensure extended battery life, enabling prolonged and continuous usage without frequent recharging.

1.4 ORGANIZATION OF THE PROJECT.

Chapter 2: Outlines the background and relevant literature.

Chapter 3: Describes the process of problem solving of the system. It focuses on the components needed and the techniques for combining them into a single system.

Chapter 4: Explains project implementation and data collection methods.

Chapter 5: Shows the results of the evaluation of the collected data of the based prototype system and contrasts it with other systems that are currently in use.

Chapter 6: Conclusion of the paper and gives recommendations for future works.

2. CHAPTER 2: LITERATURE REVIEW

This chapter shows an extensive review of the body of research on Internet of Things (IoT)-based cardiac monitoring systems that utilize the use of electrocardiogram (ECG) technology. The review encompasses key concepts, technological advancements, and relevant research studies that form the foundation for the development and implementation of the proposed IoT-based heart monitoring system.

There has been a lot of improvement in the development of cardiac monitoring systems over time. Early work concentrated on creating simple electrocardiograph devices, which were later improved with digital features. A revolutionary change has occurred with the integration of IoT into cardiac monitoring systems, enabling real-time data collection, analysis, and remote patient monitoring.

2.1 Key Concepts

1. Internet of Things (IoT) - The Internet of things (IoT) describes devices with sensors, processing ability, software and other technologies that connect and exchange data with other devices and systems over the Internet or other communications networks.

In the proposed heart monitoring system, IoT enables the seamless integration of ECG sensors, allowing for real-time data transmission, analysis, and remote monitoring.

2. Electrocardiogram (ECG) - An electrocardiogram records the electrical signals in the heart. It's a common and painless test used to quickly detect heart problems and monitor the heart's health.

ECG technology is fundamental to the project, serving as the primary source of data for monitoring the cardiac activity of individuals.

3. Heart - This is a muscular organ in most animals. This organ pumps blood through the blood vessels of the circulatory system. The pumped blood carries oxygen and nutrients to the body, while carrying metabolic waste such as carbon dioxide to the lungs. In humans, the heart is approximately the size of a closed fist and is located between the lungs, in the middle compartment of the chest, called the mediastinum.

2.2 Historical Technological Advancements.

Significant developments in clinical practices as well as major technological advancements have led to the evolution of heart monitoring systems. The following summarises the major phases in the development of cardiac monitoring systems:

1. **Early Electrocardiography (late 19th to early 20th century):** The electrocardiograph, also known as the EKG or ECG, was created in 1903 by Willem Einthoven, and it is this invention that laid the groundwork for modern heart monitoring systems. The simple recording of the heart's electrical activity made possible by Einthoven's invention provided the foundation for the diagnosis of a number of cardiac conditions.
2. **Analog Electrocardiographs (mid-20th century):** Electrocardiographs changed over the course of the mid-20th century from large, analogue machines to smaller, more portable devices. These improvements enabled easier access to ECG testing in clinical settings, which sped up the diagnosis and treatment of cardiac conditions.
3. **Digitization and Computerization (late 20th century):** ECG data could now be stored, transmitted, and analyzed via computers thanks to the digitization of analogue signals. ECG interpretation became more accurate and efficient as a result of the digitization that made it easier to develop computer-aided diagnosis (CAD) systems.
4. **Holter Monitoring (1970s onwards):** Ambulatory ECG monitoring was revolutionized with the introduction of Holter monitoring in the 1970s. Holter monitors allowed for continuous ECG recording over an extended period of time, usually 24 to 48 hours, in contrast to traditional ECGs performed in clinical settings. The periodic cardiac abnormalities that might not be picked up on short-term ECG tests can now be detected with the help of technology.
5. **Telemetry and Wireless monitoring (late 20th century to present):** With the development of telemetry devices in the late 20th century, patients could now wirelessly transmit their ECG data to medical professionals. This advancement made it easier to monitor a patient's cardiac status in real time, especially in critical care environments like emergency rooms and intensive care units (ICUs).
6. **Wearable and Remote Monitoring (21st century):** Wearable electronics with ECG sensors have become increasingly common in the twenty-first century, including fitness trackers and smartwatches. These gadgets gave people the ability to continuously monitor

their heart health in daily life and provided information on heart rate, rhythm, and activity levels. The remote transmission and interpretation of ECG data has also been made possible by developments in telemedicine and remote monitoring technologies, which facilitate online consultations and remote patient management.

7. **Integration with IoT and Artificial Intelligence (present and future):** The most recent development in cardiac care is the integration of artificial intelligence (AI) and the Internet of Things (IoT) with heart monitoring systems. Artificial intelligence (AI) algorithms improve the accuracy of ECG interpretation and enable predictive analytics for the early detection of cardiac abnormalities, while IoT-enabled devices facilitate seamless connectivity, data exchange, and remote monitoring.

2.3 Reviews on Related Works

The works that have already been done on Internet of Things (IoT)-based cardiac monitoring systems are reviewed in this section. Important ideas, scientific developments, and pertinent studies are covered in this section, which serves as a basis for the creation and application of the suggested Internet of Things (IoT) heart monitoring system.

The article [1] presents the design and implementation of an IoT-based heart rate monitoring system using photoplethysmography (PPG) sensors. Photoplethysmography (PPG) is a low cost optical technique that is able to detect volumetric changes in blood flowing through capillaries from the skin surface. The technology uses optoelectronic components, such as a red or near infrared light source, to illuminate skin and a photo detector to observe the variations in light intensity within the observed area. The PPG sensors optically observe changes in blood flow volume by detecting changes in light intensity. In this system we use the pulse sensor with Arduino Uno and Bluetooth HC-05 module, the pulse sensor is placed on the finger and it measures the heart rate and then sends the heart rate to android mobile application via Bluetooth.

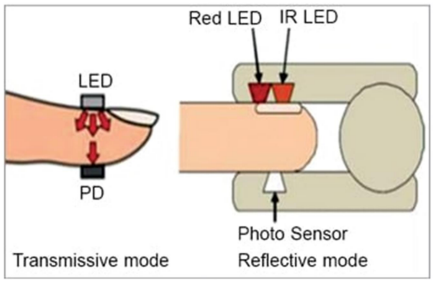

Figure 2.1: Photoplethysmography

This research paper [2] describes the development of a smart healthcare system for heart monitoring using IoT and smartphone accelerometers. Present healthcare systems use a large number of environments and platforms including Wireless Body Area Networks (WBANs), mobile devices and wireless grid/cloud/web services to make healthcare services available, observable, transparent, seamless, reliable and sustainable. The accelerometer is worn around the patient's waist, either using a waist belt or modified mobile-phone carry-case. The hardware components of the prototype comprises of WBAN nodes (hereafter referred to as BS nodes or Body Sensor nodes), implemented on Arduino Fio platforms to collect physiological data and transmit it wirelessly to the Central Intelligent Node (CIN). From the CIN, physiological data of patient is then uploaded to the Medical Health Server (MHS). Physiological data received on the MHS would be equivalent to that obtained at the medical facility, if the patient were to go there for a medical check-up.

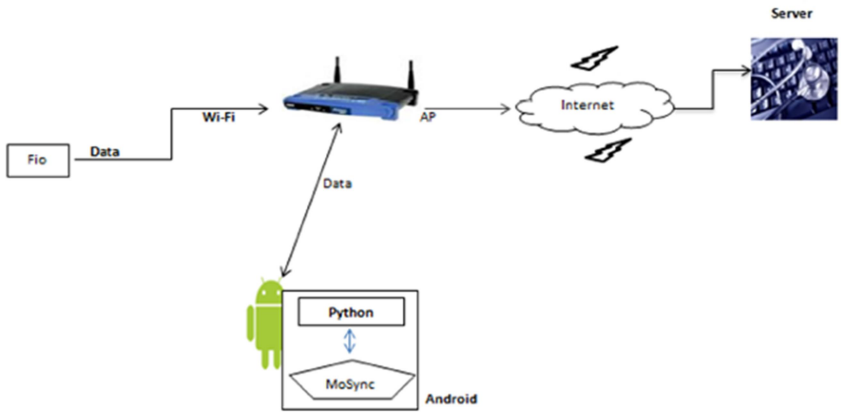

Figure 2.2: WBAN Connection

This study [3] presents the development of a wireless heart rate monitoring system using IoT-based textile sensors. In the proposed system, a single sensor is placed in the clothes' front side fabric and senses the cardiorespiratory activity. The input units are the user's fingertips capturing unit; heart pulse sensor unit, the power supply unit and the user interface unit. The output units are the liquid crystal display (LCD) and the WiFi Module unit. The system is controlled by the ATmega32p microcontroller and programmed using embedded C programming language. The sensed data from the sensor is transmitted to the analog to digital converter (ADC) for

conversion to digital signal. The converted digital signal is then transmitted to the microcontroller. The microcontroller acts on the signal based on the instructions coded in the embedded C language. The processed data are transmitted to the LCD screen for user information. Furthermore, the data are sent in real-time to the WiFi module and transmitted to the webserver for further analytics and visualization.

This study [4] presents the development of an IoT-based heart monitoring system using pulse oximetry sensors. The pulse oximeter instrument can be used to measure the Heartbeat and SPO2 (oxygen saturation) of the person by applying IoT based Wi-Fi technology, which can help to monitor oxygen saturation level and also regular check-up to avoid any the critical situation of health. In the present paper [4], a proposed system, which consists of MAX30100 heart rate sensor, ESP32 development board with Wi-Fi has been utilized. Here heart rate and SpO2 are measured in real time and displayed in Android smartphones.

2.4 Capturing and Collection of Data

In the context of heart monitoring systems, recording techniques refer to the procedures used to record and gather information about the electrical activity of the heart. In order to precisely evaluate cardiac function and identify any anomalies, these methods are essential. The following are some common methods of recording used in cardiac monitoring:

1. **Electrocardiography (ECG or EKG):** This is the most common technique for capturing the electrical activity of the heart is the electrocardiogram (ECG). To detect and record the electrical signals produced by the heart's contractions, electrodes are placed on the skin's surface, usually on the arms, legs, and chest. ECG recordings are a useful source of information regarding irregularities in conduction, heart rate, and rhythm.
2. **Holter Monitoring:** This method called "holter monitoring" is used to continuously record an ECG for an extensive time—usually 24 to 48 hours. A Holter monitor is a portable device that patients wear to continuously record ECG signals as they go about their daily lives. Holter monitoring can be used to assess symptoms like palpitations or vertigo, as well as to identify intermittent cardiac arrhythmias.
3. **Ambulatory ECG Monitoring:** Ambulatory ECG monitoring uses wearable technology to record ECG signals for a prolonged amount of time—typically 24 hours or longer. These devices could be patch-based monitors that are attached to the patient's chest or

loop recorders. Continuous monitoring of cardiac activity outside of clinical settings is made possible by ambulatory ECG monitoring, which can identify irregular heart rhythms and offer important insights into long-term cardiac rhythm patterns.

These recording methods are essential for the diagnosis of cardiac disorders, tracking the advancement of the disease, and evaluating the efficacy of therapeutic measures. They offer insightful information about the electrical activity of the heart, empowering medical professionals to decide wisely and treat patients with cardiac conditions promptly.

2.4.1 Electrocardiography.

The electrocardiogram (abbreviated as ECG or EKG) represents an electrical tracing of the heart and is recorded non-invasively from the surface of the body. The word ECG derives from the German language. In German, it is elektro-kardiographie. In 1902, the Dutch physician Einthovan invented ECG, and his tremendous input in clinical studies for about ten years led to full recognition of the clinical potential of the technique.[5]

Many arrhythmias and ECG changes associated with angina and atherosclerosis were identified by 1910. William Einthoven was named the "father of electrocardiography" and was awarded Nobel Prize in Medicine in 1924 for his hard work that laid the foundation of the most fundamental technique for investigating heart disorders. ECG was soon recognized as a robust screening and clinical diagnostic tool, and today it is used globally in almost every healthcare setting. [6]

ECG is a non-invasive diagnostic modality that has a substantial clinical impact on investigating the severity of cardiovascular diseases.[7] ECG is increasingly being used for monitoring patients on antiarrhythmics and other drugs, as an integral part of preoperative assessment of patients undergoing non-cardiac surgery, and for screening individuals in high-risk occupations and those participating in sports. Also, ECG serves as a research tool for surveillance and experimental trials of drugs with recognized cardiac effects.

Cardiovascular disease, as the number one cause of death, puts a great emphasis on healthcare providers developing skills and knowledge in interpreting ECGs to provide the best care promptly. Many healthcare providers find the advanced interpretation of ECG findings a

complicated task. Errors in the analysis can lead to misdiagnosis, delaying the appropriate treatment.

The heart is a vital body organ and occupies space in the central chest between the lungs. Together with the blood vessels and blood, it constitutes the body's circulatory system. The heart is a muscular organ comprised of four chambers with two atria (right and left) opening into right and left ventricles via tricuspid and mitral valves, respectively. A wall of muscle called the septum separates all four chambers. The heart receives deoxygenated blood from the whole body via superior and inferior vena cava, which first enters the right atrium. From here, it transits through the right ventricle and then passes into the lungs via the right and left pulmonary arteries, where it is oxygenated. The oxygenated blood from the lungs pours into the left atrium through the right and left pulmonary veins, and from here, it is pumped by the left ventricle into the aorta to the rest of the body. The heart derives its blood supply from the coronary arteries that branch off the aorta. [8]

There are three main components to an ECG:

- The P wave, which represents depolarization of the atria.
- The QRS complex, which represents depolarization of the ventricles.
- The T wave, which represents repolarization of the ventricles.

P-wave - It represents atrial depolarization on the ECG. As atrial depolarization initiates by the sinoatrial node located in the right atrium, the right atrium gets depolarized first, followed by left atrial depolarization. So the first half of the P wave represents right atrial depolarization and the second half shows left atrial depolarization.

QRS Complex- It represents ventricular depolarization as current passes down the atrioventricular node. A standard QRS complex has duration of less than three small squares (fewer than 120 ms, usually 60 to 100 ms). A prolonged QRS may indicate hyperkalemia or bundle branch block.

T-Wave - It represents ventricular repolarization. Its morphology is highly susceptible to cardiac and non-cardiac influences like (hormonal and neurological).

ST-Segment - It depicts the end of ventricular depolarization and the beginning of ventricular repolarization.

QT Interval - It represents the start of depolarization to the end of the repolarization of ventricles.

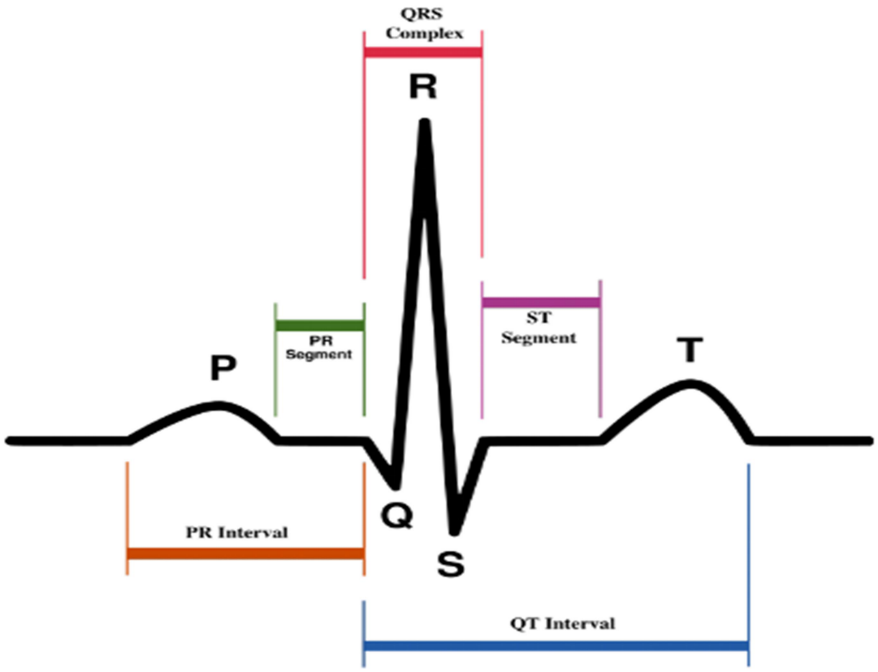

Figure 2.3: An Electrocardiogram

2.5 IoT in Healthcare

IoT in healthcare refers to a combination of medical tools and software communicating with healthcare IT systems over internet computer networks. IoT enables machine-to-machine connection through Wi-Fi-enabled medical devices, establishing the groundwork for a seamless data-sharing and analysis system.

Healthcare IoT launched its first appearance in the 1990s by introducing basic telehealth and remote patient monitoring technology. Wearable health devices and intelligent medical equipment result from early 2000s miniaturization and data analytics advancements.

Personalized treatment, real-time data-driven decision-making, and remote patient monitoring are all rendered possible by IoT, which is currently revolutionizing the healthcare industry. [9]

The Internet of Things (IoT) is becoming increasingly popular in the healthcare industry every year, which shows that this solution has many significant benefits:

a) Monitoring and Alerts - IoT offers ongoing tracking of patients' vital signs and health parameters with connected smart devices. Intelligent sensors can monitor various parameters, including blood pressure and glucose levels.
b) Remote Assistance - With the help of online consultations and communications with connected equipment, IoT makes remote medical support possible.
c) End-to-end Connectivity - The healthcare system interacts seamlessly thanks to IoT. Hospitals, doctors' offices, and intelligent medical gadgets automatically communicate patient data.
d) Data Analysis - Healthcare IoT gadgets' automated data collection provides helpful information. By analyzing data, medical experts can find trends and connections between various health markers.
e) Preventive Healthcare - The IoT system for healthcare can spot early warning symptoms of an illness and issue prompt notifications.

2.6 The Challenges of IoT in Healthcare.

Although IoT in healthcare brings numerous benefits, it also encounters challenges along its development path.

a) Data Overload - One of the significant challenges of implementing IoT in healthcare is the sheer volume of data generated by connected devices.
b) Data Security - Data security becomes crucial with the exponential increase in data collection through IoT devices.
c) Integration of Protocols - IoT devices often operate on different communication protocols and standards, making seamless integration challenging.
d) Cost Constraints - Initial installation expenses can run high. Integrating IoT devices, upgrading infrastructure, and hiring new people costs money.

IoT has quickly changed the healthcare industry, offering essential advantages, including remote patient monitoring and data-driven decision-making. The potential for IoT in healthcare is exciting, despite issues like data overload and security worries. Collaborative efforts are essential to solve problems, fully take advantage of IoT's advantages and build a patient-centered, effective, and data-driven healthcare system. IoT will change patient care and healthcare services with continual innovation, leading to better results and an improved health industry.

2.7 Arduino.

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. [10]

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

3. CHAPTER 3: MATERIALS AND METHODS.

This study's methodology includes a number of important phases, such as software development, hardware implementation, system design, and experimental validation. We describe in detail the components, tools, and methods used at each level to guarantee the IoT-based heart monitoring system's successful implementation.

3.1 Analogue Front End.

Analogue front ends (AFEs) are a set of analogue signal conditioning circuitry that is used to interface sensors to an ADC or a microcontroller. In applications like heart rate monitors, AFEs bear tremendous importance. There will be low amplitude in the received ECG signal. This very weak signal, which ranges from 0.5mv to 5.0mv, must thus be handled by the front end. We will therefore need amplification. This also calls for the elimination of noise produced by multiple sources, resulting in the need for the use of a strong attenuator. The amplified signal must be filtered in order to remove any noise that may have been captured during the ECG capture process. AC noise from mains power (50 Hz in Kenya) is usually reduced using a band-pass or notch filter. In addition to respiration, if the patient is not still during recording, it can produce noise akin to an electromyogram and result in baseline drift. Electrode offset noise is another possibility. Furthermore, electrode offset noise may arise from poor link between the electrodes and the patient's skin. All of these noise components must be eliminated or greatly attenuated in order to guarantee that the ECG signal is suitable for analysis before the process of digitizing.

While there are many different and fully integrated ECG front ends, the Gravity-Analogue Heart Rate Sensor ECG SN0213 which is manufacture by DFROBOT has been used. Its IC/Module is the AD8232. The AD8232 has been chosen on the basis that it has low-power consumption and low cost.

Parameter	***AD8232***	***HM301***	***ADS1191***
Manufacturer	Analogue Devices	ST Microelectronics	Texas Instruments
Dimensions(mm)	4 x 4	6 x 6	5 x 5
Operating Voltage	2 – 3.5V	1.62 -3.62V	1.7 – 3.6V

Operating Current (Operating Power)	170µA	1.3mA	(170µW/channel)
Output Impedance	10GΩ	50MΩ	100MΩ
Gain	100 V/V	64 V/V	12 V/V
Low Power Mode	✓	✓	✓
ECG Channel	1	3	2
Chip Cost	KSH 350	KSH 700	KSH 700

Table 3-1: Comparing ECG Sensors

The voltage range of the AD8232 is 2-3.5 V, and its low supply current is 170µA. Using this make with an external microcontroller, like the Arduino, is easy. It can also have a small PCB design thanks to the 4mm x 4mm packaging (as seen to be used in the SN0213). The AD8232 has a number of features, such as an integrated fast restore circuit, leads-off detection, and a right-leg driver (RLD). Utilizing a three-electrode design with RA, LA, and RL(RLD) electrodes, a high signal-to-noise (SNR) ratio is required for safety. The RLD/ground electrode is used to drive an opposing signal into the patient and extract the common-mode voltage, thereby reducing the effects of common noise.

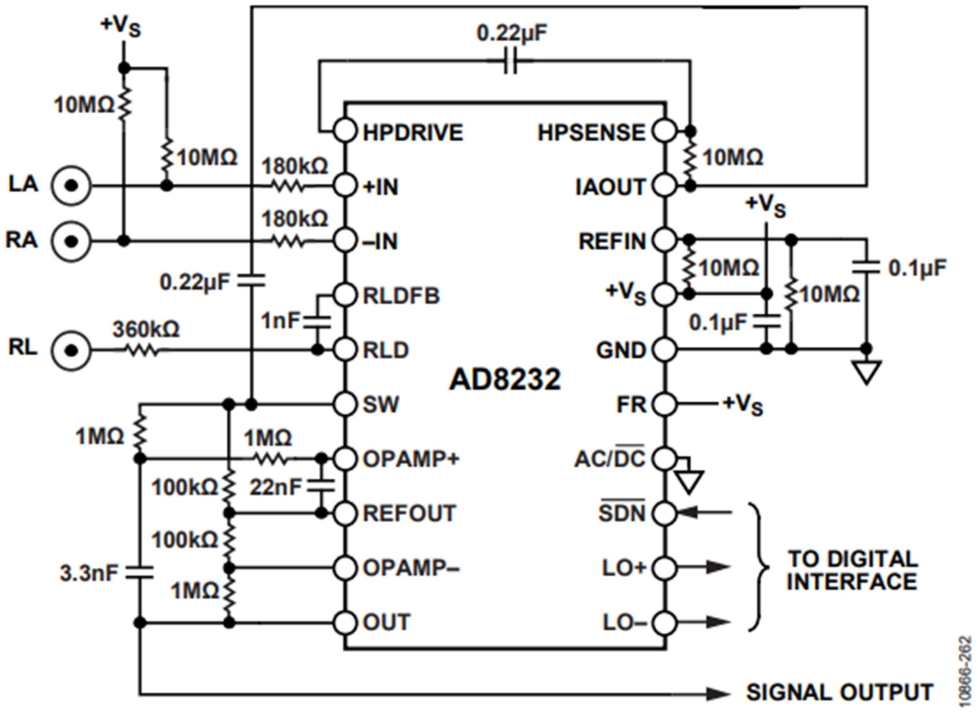

Figure 3-1: AD8232 Internal connections

- LA = Left Arm Electrode, conventionally placed on the Left Arm or Chest (closest to heart therefore strongest signal)
- RA = Right Arm Electrode, conventionally placed on the Right Arm or Chest
- RLD = Right Leg Drive Electrode, conventionally placed on Lower abdominal or Leg (allows for common-mode rejection, reducing noise)

The AD8232 typically operates from a single power supply voltage, typically ranging from 2.0V to 3.5V. It is important to provide a stable and noise-free power supply to ensure proper operation of the device. The AD8232 features an instrumentation amplifier (INA) as its primary signal conditioning stage.

The gain of the instrumentation amplifier is determined by the ratio $\frac{RG}{RF}$

The AD8232 includes a dedicated right leg drive (RLD) amplifier to minimize common-mode noise. The value of the right leg drive resistor affects the amplitude of the feedback signal applied to the patient's right leg. Following amplification, the AD8232 incorporates a low-pass filter (LPF) to attenuate high-frequency noise. The resistor and capacitor values for the LPF are typically chosen to set the cutoff frequency of the filter, providing adequate attenuation of noise while preserving the desired signal components:

- RLPF: Resistor in the low-pass filter, typically in the range of 10 kΩ to 100 kΩ.
- CLPF: Capacitor in the low-pass filter, typically in the range of 100 pF to 1 nF.
- The cutoff frequency of the LPF is determined by the equation: f_cutoff = 1 / (2π * RLPF * CLPF).

The AD8232 provides a buffered output signal suitable for further processing or transmission. The resistor values for the output buffer are typically set internally by the device, and external resistor values are not required. It also includes integrated lead-off detection circuitry to detect electrode contact with the patient's skin. The resistor and capacitor values for lead-off detection are typically set internally by the device.

The primary tool for determining the heart's electrical activity is the DFRobot Heart Rate Monitor sensor. It has the AD8232, which processes the PR and QT intervals to generate a clear signal. The Arduino IDE "Serial Plotter" feature, which allows users to view the ECG output in a graphical format on a PC, is also compatible with the device.

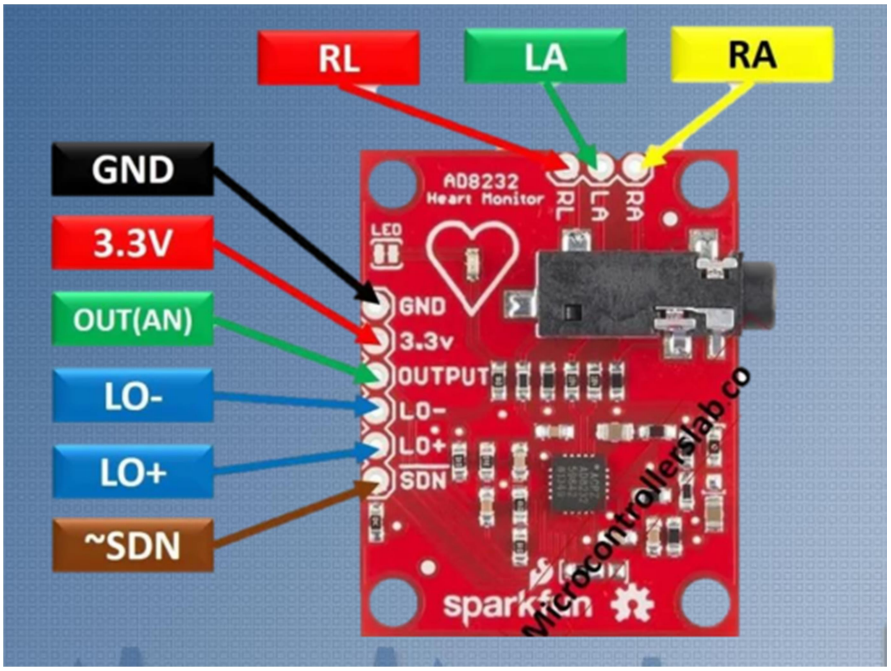

Figure 3-2 : AD8232 Pin Out

3.2 Choice of Microcontroller

ESP32 is the SoC (System on Chip) microcontroller which has gained massive popularity recently. Whether the popularity of ESP32 grew because of the growth of IoT or whether IoT grew because of the introduction of ESP32 is debatable. The following are some of the important specifications of the ESP32 WROOM 32 variant.−

- Integrated Crystal− 40 MHz
- Module Interfaces− UART, SPI, I2C, PWM, ADC, DAC, GPIO, pulse counter, capacitive touch sensor
- Integrated SPI flash− 4 MB
- ROM− 448 KB (for booting and core functions)
- SRAM− 520 KB
- Integrated Connectivity Protocols− WiFi, Bluetooth, BLE
- On−chip sensor− Hall sensor
- Operating temperature range− −40 − 85 degrees Celsius
- Operating Voltage− 3.3V

- Operating Current− 80 mA (average)

It's quite simple to figure out why ESP32 is so popular when you have the specs listed above in front of you. Thinking about the specifications a microcontroller (μC) in an Internet of Things device would need to meet, the major operational blocks of any IoT device are sensing, processing, storage, and transmitting. As a result, the μC should initially be able to communicate with a range of sensors. All common communication protocols needed for sensor interfaces, such as UART, I2C, and SPI, should be supported. It ought to be capable of pulse counting and ADC. ESP32 satisfies each of these specifications. Second, the μC needs to have enough memory to store the data and be able to process incoming sensor data, sometimes at very high speeds. The maximum operating frequency of the ESP32 is 40 MHz, which is a respectable amount. Its two cores enable parallel processing, which is an additional bonus. Ultimately, its 520 KB SRAM is adequate for handling a sizable amount of data while onboard. You can not only store data, but also text files, images, HTML and CSS files, and a lot more within SPIFFS.

Lastly, the ESP32's integrated Wi-Fi and Bluetooth stacks for data transmission have shifted the game. It is not necessary to connect an additional module (such as an LTE or GSM module) in order to test cloud communication. All you need to get started is the ESP32 board and an active Wi-Fi network. You can use Wi-Fi in both access point and station mode with the ESP32. It supports HTTPS in addition to HTTP, MQTT, TCP/IP, and other conventional communication protocols. It is equipped with a crypto-accelerator, also known as a crypto-core, which is a specialized piece of hardware designed to speed up the encryption process. As a result, you can securely communicate with your web server in addition to doing so. Support for BLE is also essential for a number of applications. Of course, the ESP32 can be interfaced with LTE, GSM, or LoRa modules. Consequently, ESP32 surpasses expectations in terms of "transmitting data" as well. ESP32 can be programmed using the Arduino IDE, making the learning curve much less steep.

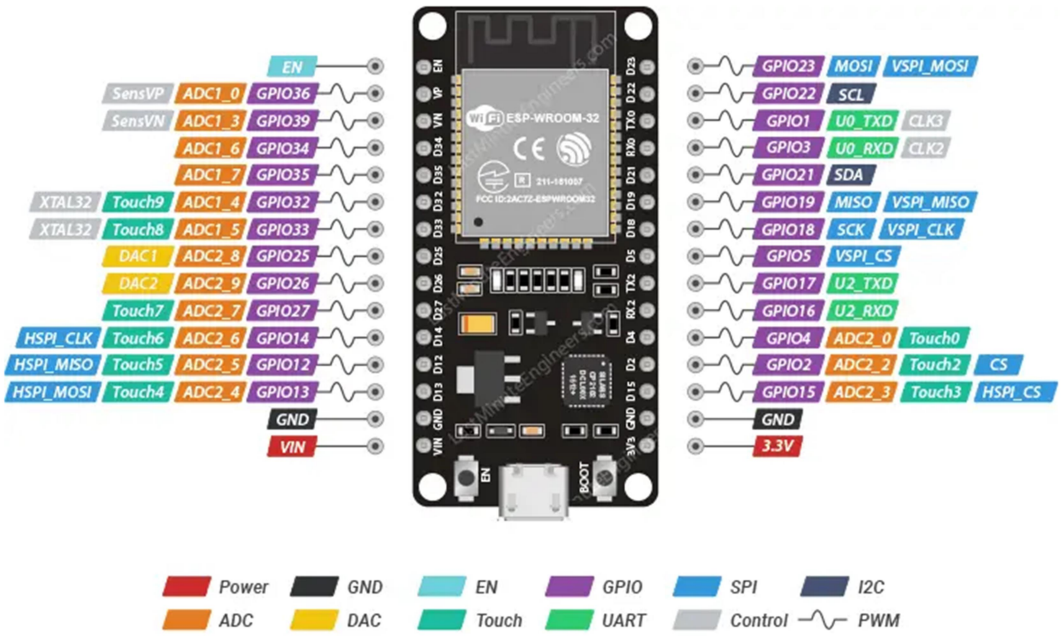

Figure 3-3: ESP32 Pin Out

Although the ESP32 has 48 GPIO pins in total, only 25 of them are broken out to the pin headers on both sides of the development board. Thanks to the ESP32's pin multiplexing feature, which allows multiple peripherals to share a single GPIO pin. For example, a single GPIO pin can act as an ADC input, DAC output, or touch pad. The ESP32 development board has 25 GPIO pins that can be assigned different functions by programming the appropriate registers. There are several kinds of GPIOs: digital-only, analog-enabled, capacitive-touch-enabled, etc. Analog-enabled GPIOs and Capacitive-touch-enabled GPIOs can be configured as digital GPIOs. Most of these digital GPIOs can be configured with internal pull-up or pull-down, or set to high impedance. Pins GPIO34, GPIO35, GPIO36 (VP) and GPIO39 (VN) cannot be configured as outputs. They can be used as digital or analog inputs, or for other purposes. They also lack internal pull-up and pull-down resistors, unlike the other GPIO pins.

The ESP32's ADC is a 12-bit ADC, which means it can detect 4096 (2^12) discrete analog levels. In other words, it will convert input voltages ranging from 0 to 3.3V (operating voltage) into integer values ranging from 0 to 4095. This results in a resolution of 3.3 volts / 4096 units, or 0.0008 volts (0.8 mV) per unit. The ADC2 pins cannot be used when Wi-Fi is enabled. Since there is a good chance of using Wi-Fi on a microcontroller designed to use it, only the ADC1 can

be used. Ideally, you would expect a linear behavior when using the ADC, but this is not the case. The ADC converters on the ESP32 are non-linear in nature. In the graph below, the non-linearities at the lower and upper ends of the input voltage are clearly visible.

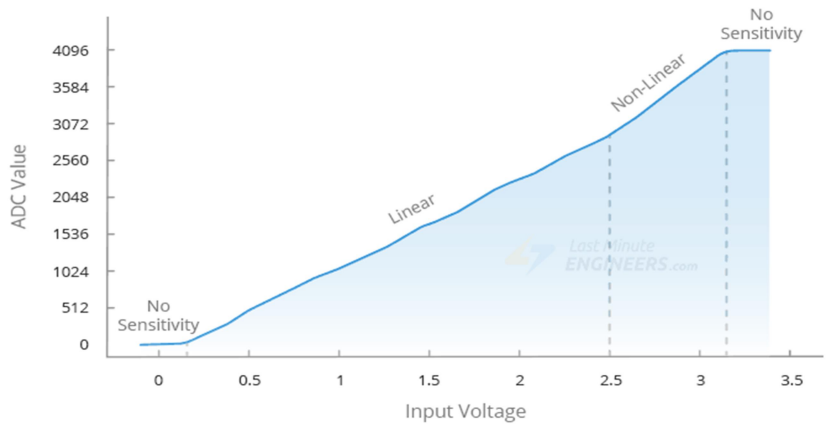

Figure 3-4: ESP32 Non-Linear characteristics

This basically means that the ESP32 cannot distinguish 3.2V from 3.3V; the measured value will be the same (4095). Similarly, it cannot distinguish between 0V and 0.13V signals; the measured value will be the same (0). The electrical noise of the ADC also implies a slight fluctuation of the measurements. However, this can be corrected by adding a capacitor at the output and by oversampling.

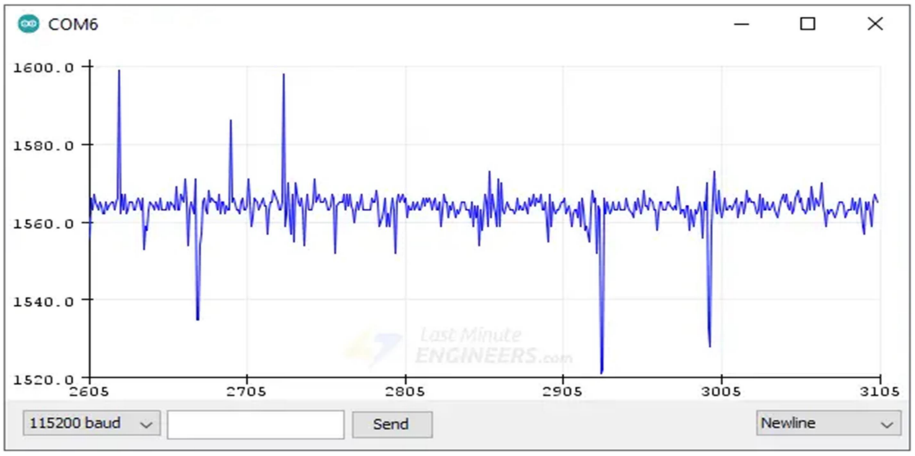

Figure 3-5: Electrical Noise in ESP32

The ESP32 includes two 8-bit DAC channels for converting digital signals to true analog voltages. It can be used as a “digital potentiometer” to control analog devices. These DACs have an 8-bit resolution, which means that values ranging from 0 to 256 will be converted to an analog voltage ranging from 0 to 3.3V. The ESP32 has 9 capacitive touch-sensing GPIOs. When a capacitive load (such as a human finger) is in close proximity to the GPIO, the ESP32 detects the change in capacitance. You can make a touch pad by attaching any conductive object to these pins, such as aluminum foil, conductive cloth, conductive paint, and so on. Because of the low-noise design and high sensitivity of the circuit, relatively small pads can be made. Additionally, these capacitive touch pins can be used to wake the ESP32 from deep sleep.

The ESP32 has two I2C bus interfaces, but no dedicated I2C pins. Instead, it allows for flexible pin assignment, meaning any GPIO pin can be configured as I2C SDA (data line) and SCL (clock line). However, GPIO21 (SDA) and GPIO22 (SCL) are commonly used as the default I2C pins to make it easier for people using existing Arduino code, libraries, and sketches. ESP32 features three SPIs (SPI, HSPI, and VSPI) in slave and master modes. The ESP32 dev. board has three UART interfaces, UART0, UART1, and UART2 that support asynchronous communication (RS232 and RS485) and IrDA at up to 5 Mbps. In addition, UART provides hardware management of the CTS and RTS signals and software flow control (XON and XOFF) as well. The board has 21 channels (all GPIOs except input-only GPIOs) of PWM pins controlled by a PWM controller. The PWM output can be used for driving digital motors and LEDs. The PWM controller consists of PWM timers, the PWM operator and a dedicated capture sub-module. Each timer provides timing in synchronous or independent form, and each PWM operator generates a waveform for one PWM channel. The dedicated capture sub-module can accurately capture events with external timing.

Some GPIOs are routed to the RTC low-power subsystem and are referred to as RTC GPIOs. These pins are used to wake the ESP32 from deep sleep when the Ultra Low Power (ULP) co-processor is running. The GPIOs highlighted below can be used as external wake up sources. There are five strapping pins: GPIO0, GPIO2, GPIO5, GPIO12, and GPIO15. These pins are used to put the ESP32 into BOOT mode (to run the program stored in the flash memory) or FLASH mode (to upload the program to the flash memory). Depending on the state of these pins, the ESP32 will enter BOOT mode or FLASH mode at power on. On most development boards

with built-in USB/Serial, you don't need to worry about the state of these pins, as the board puts them in the correct state for flashing or boot mode. However, if peripherals are connected to these pins, you may encounter issues when attempting to upload new code or flash the ESP32 with new firmware, as these peripherals prevent the ESP32 from entering the correct mode. The strapping pins function normally after reset release, but they should still be used with caution.

There are two power pins: the VIN pin and the 3V3 pin. The VIN pin can be used to directly power the ESP32 and its peripherals, if you have a regulated 5V power supply. The 3V3 pin is the output from the on-board voltage regulator; you can get up to 600mA from it. GND is the ground pin. The EN pin is the enable pin for the ESP32, pulled high by default. When pulled HIGH, the chip is enabled; when pulled LOW, the chip is disabled. The EN pin is also connected to a pushbutton switch that can pull the pin LOW and trigger a reset.

3.3 Data Communication

MQTT stands for Message Queuing Telemetry Transport. It is an extremely simple and lightweight messaging protocol (subscribe and publish) designed for limited devices and networks with high latency, low bandwidth or unreliable networks. Its design principles are intended to guarantee supply security while lowering network bandwidth and device resource requirements. These principles also benefit Internet of Things (IoT) or machine-to-machine (M2M) devices, as battery life and bandwidth are critical factors.

Resource-constrained Internet of Things (IoT) devices can publish or send information on a particular topic to a server that serves as a MQTT message broker by using MQ Telemetry Transport. The information is then sent by the broker to the clients who have already purchased the client's topic. A topic appears to a human as a hierarchical file path. Consumers have the option to subscribe to a single topic hierarchy level or to several levels by using a wildcard character. For wireless networks with fluctuating latency brought on by inconsistent bandwidth restrictions or unpredictable connections, the MQTT protocol works well. The broker buffers the messages and sends them to the subscriber when the subscriber is back online in the event that the connection between the subscribing client and the broker is lost. The Broker may disconnect and send a cached message containing publisher instructions to the Subscriber if the Publishing Client disconnects from the Broker without notifying the Subscriber.

The heart of any publish/subscribe protocol is the **MQTT broker**. A broker can handle up to thousands of concurrently connected MQTT clients, depending on how it is implemented. All messages must be received by the broker, who will then sort them, ascertain who subscribed to each one, and deliver the messages to the clients who have subscribed. All persistent clients' sessions, including missed messages and subscriptions, are also kept by the Broker. The Broker's responsibility also includes client authorization and authentication. Custom authentication, authorization, and backend system integration are made easier by the broker's extensibility. Because the Broker serves numerous clients, is frequently the component that is directly exposed on the Internet, and must forward messages to systems for downstream analysis and processing, integration is particularly crucial.

Using a broker, MQTT allows messages to be shared with other hardware or software. Each message has a topic that determines how the Broker can proceed with processing it. Every message also includes message content, also known as the **payload**. The MQTT payload can be created in any way and is not restricted to any particular format. To make the message content readable by other software or devices, it is useful to define a specific structure. Clients have the ability to publish and subscribe to messages sent to and received from the broker. A message can be processed further by using the specified MQTT topic, which is required when sending a message to the broker.

In HTTP, your messages are directly sent to the intended server and you even get an acknowledgment in the form of status codes. In MQTT, you just send messages to the broker in the hope that your intended server(s) will take it from there.

3.4 Data Visual Application

With the help of UBIDOTS, industries and innovators can prototype and scale IoT projects to production. This makes it possible to create IoT applications without having much experience with databases or programming. Many internet-connected projects in the fields of manufacturing, energy, and healthcare have successfully implemented the numerous tiny IoT and cloud features that facilitate digital transformation. It offers real-time basic information analysis on the measured variable along with cloud and storage services. UBIDOTS is used in this project as a

cloud service to store and process real-time sensor data. By entering the required credentials, which include a password, email address, and username, users can access it.

When transferring data from the cloud to your devices, this pro MQTT is very helpful. Envision a cloud-based gadget that allows you to remotely open and close doors. When using HTTP, the device would need to keep sending GET requests to the Ubidots server in order to check if a variable has changed and, if so, to act based on the most recent reading. As it depends on the frequency of polling, this requires a lot of requests and isn't always a real-time exchange. The device can "listen" to the cloud via MQTT, only receiving notifications in the event that a variable changes.

In this manner, battery life, network capacity, and real-time performance are all improved while maintaining an open connection between the device and the cloud and only sending data when required. In this instance, the ESP32 transmits and receives data via the MQTT protocol over the internet using the Ubidots platform.

3.5 Electrodes.

Ionic potentials are transformed into electronic potential by electrodes. Typically, they consist of a metal electrode, electrode-conducting paste or gel for the surface electrode, and lead (for electrical current conduction). The electrical changes are detected by electrodes.

The right, left, and right legs are where the three electrodes are positioned. Here, the instrumentation amplifier of the AD8232 receives positive and negative input from electrodes placed on the left and right arms, respectively, with the electrodes on the right arm serving as a ground for the EC. Lead cables and silver chloride electrodes, which are used in the ECG, are used to receive and transmit data when they are firmly placed on the chest wall's flat ventral surface. Upon accurate reading and interpretation of the ECG data, a variety of cardiac conditions, including arrhythmias and cardiac ischemia, can be identified and monitored.

3.6 System Overview

3.6.1 Component Block Diagram

A typical ECG monitoring system component block diagram includes a power module (PC), an ECG sensor (AD8232), a processing unit (ESP32), Wi-Fi connectivity, and an IoT cloud server (Ubidots). The power module, which is typically a computer (PC), supplies a steady power supply to all system components, ensuring that the processing unit and the ECG sensor run reliably and efficiently without power outages—an essential component for the precision and dependability of medical data collection.

The purpose of the ECG sensor, specifically the AD8232, is to evaluate the heart's electrical activity. Electrodes are applied to the skin to record the patient's raw ECG signals. With its high sensitivity and ability to filter out noise, this sensor can provide an accurate and clear picture of the heart's activity. In order to get the analogue signals ready for digital processing, the AD8232 preprocesses them by boosting them and removing unwanted frequencies.

Subsequently, the ESP32 processing unit receives the processed analogue signals from the AD8232. The ESP32 is a powerful microcontroller that has Bluetooth and Wi-Fi built right in. With the aid of its integrated analog-to-digital converter (ADC), it digitises the incoming analogue ECG signals. The ESP32 can process these signals further once they've been digitalized, such as by doing real-time analysis or getting the data ready for transmission. Because of its efficient data handling programming, the ESP32 can run algorithms that instantly identify irregularities in the ECG data, such as arrhythmias.

The ESP32 uses its integrated Wi-Fi module to connect wirelessly to the internet for connectivity. It transmits the processed ECG data to an IoT cloud server, like Ubidots, via this connection. Ubidots is a cloud-based platform that facilitates the collection, storing, and analysis of data from a variety of sensors. Using HTTP or MQTT protocols, the ESP32 transmits the data to Ubidots in a secure and dependable manner.

When the ECG data gets to Ubidots, it is saved and the patient or medical professionals can access it remotely. Ubidots makes it simpler to track heart health over time by offering a variety

of tools for data visualization and analysis. If it finds any anomalies in the ECG data, it can send out alerts, allowing doctors to intervene in time. In order to create a complete ecosystem for health monitoring, the cloud server can also integrate with other devices or systems.

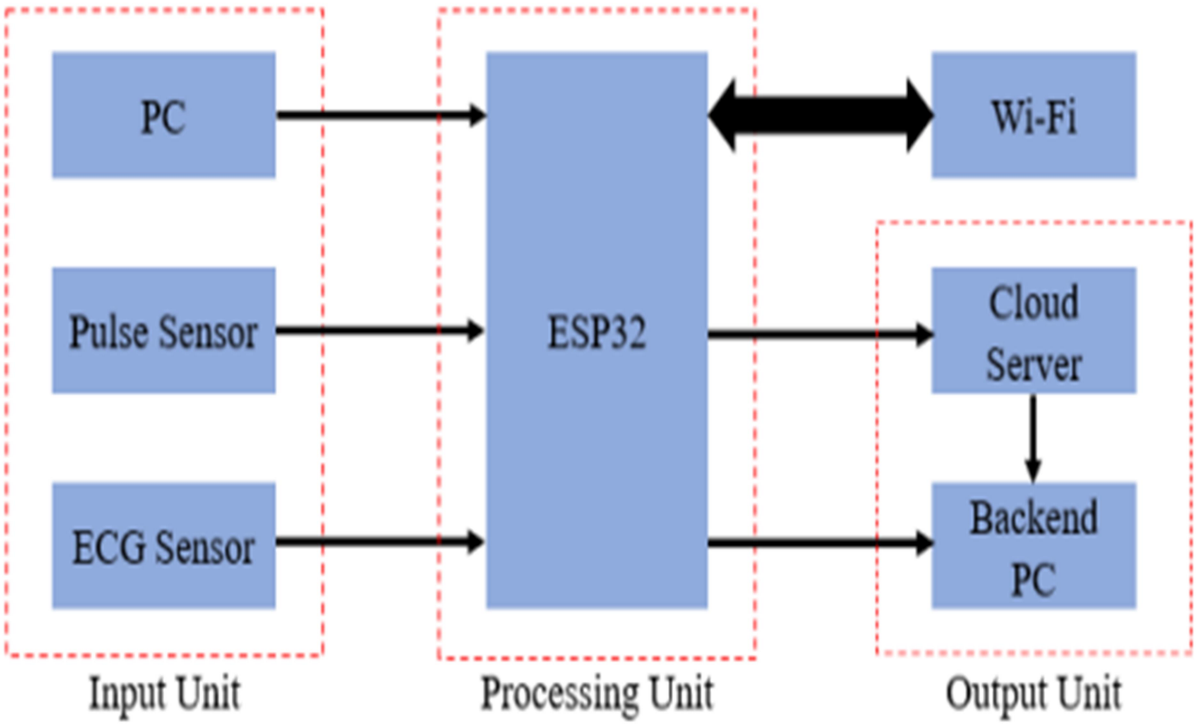

Figure 3-6: Components Block Diagram

In conclusion, this ECG monitoring system's component block diagram presents a solid and comprehensive strategy for real-time heart monitoring. The ESP32 processes and transmits the data, the power module (PC) guarantees continuous operation, the AD8232 ECG sensor records and preprocesses heart signals, and Ubidots acts as the cloud platform for remote access, storage, and analysis. By enabling precise, timely, and remote monitoring of cardiac health, this networked system has the potential to save lives by identifying heart conditions early.

3.6.2 System Flow Design

System flow is crucial when choosing hardware because it explains how things should happen in a chronological order. This stage aids in the creation of a method that illustrates how the system

operates.

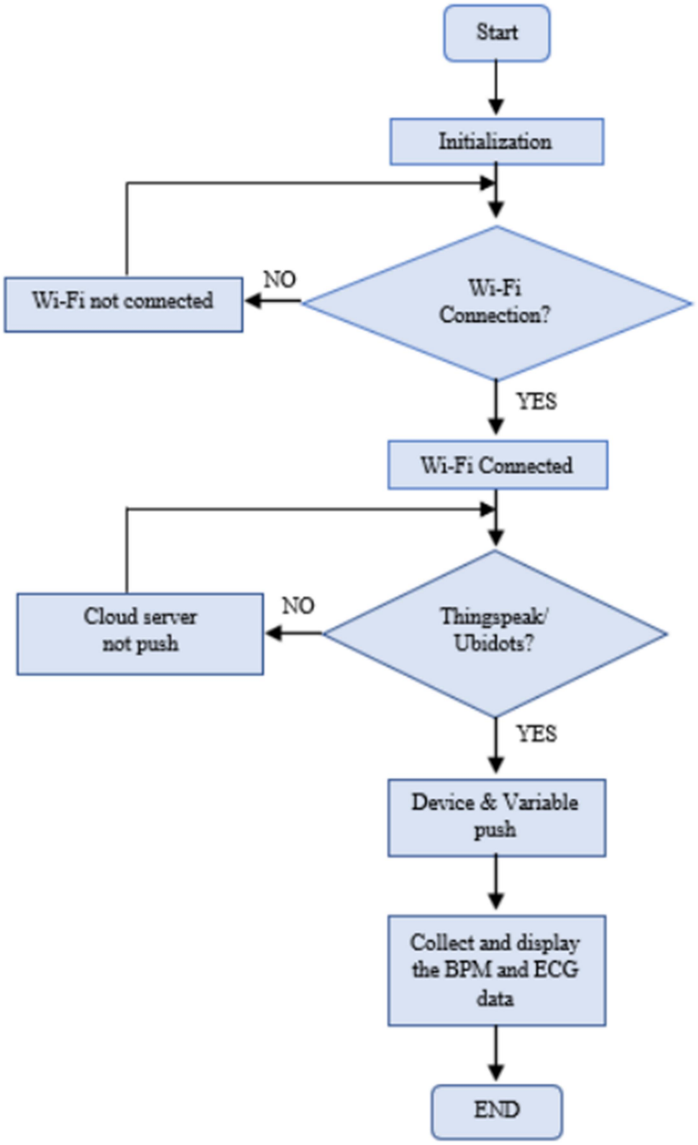

Figure 3-7: System Flow Chart

The system is initialized at the beginning of the flowchart, which entails turning on the microcontroller (ESP32) and the ECG sensor (AD8232), as well as setting up initial conditions. The patient's heartbeat is detected by the ECG sensor, which then records the electrical activity using electrodes and converts it into analogue data. After preprocessing, this raw data is cleaned up and amplified to prepare it for additional processing.

The ESP32 microcontroller is used in the following step, and it uses its integrated analog-to-digital converter (ADC) to digitize the preprocessed ECG signals. The continuous analogue signals are converted into discrete digital values through digitization so that the microcontroller can process them. Subsequent to that, the system reaches the first decision block, which is connecting to a Wi-Fi network. In this instance, the ESP32 tries to join a pre-established Wi-Fi network. The system may enter an error state or attempt to reconnect after a predetermined number of attempts, informing the user to check the Wi-Fi module or network settings.

Data transmission is the next step the system takes after a successful Wi-Fi connection is made. The digitalized ECG data is prepared for transmission, usually in JSON or another structured format, so that the receiving server can easily parse and process it. This data is then sent by the ESP32 to an IoT cloud server called Ubidots. This step adds another decision block: confirming that the data upload to Ubidots was successful. Should the data transmission fail—possibly because of a server outage or network problem—the system has the ability to log the error, try again, or notify the user of a connectivity issue.

The data is processed and stored in the cloud after it is successfully transmitted to Ubidots. With the help of several tools from Ubidots, patients and healthcare professionals can view real-time ECG data visualization in the form of graphs and charts. Ubidots examines the incoming data for anomalies in a decision block included in the system flow chart.

3.7 The Code Design

The code is an implementation that uses an ESP32 microcontroller to read data from an ECG sensor. It then processes the data and sends it via the MQTT protocol to Ubidots, an IoT cloud platform. This system ensures accurate and real-time monitoring and visualization of heart activity data by utilizing NTP (Network Time Protocol) for precise timestamping and Wi-Fi for connectivity.

The code design is an example of a complete IoT-based solution for real-time ECG monitoring. Correct data collection, processing, and transmission to a cloud platform are guaranteed by the system's integration of Wi-Fi connectivity, NTP time synchronization, and MQTT

communication. Continuous monitoring, data visualization, and prompt intervention are made possible by this, which may improve patient care and health outcomes. The code's robustness and modularity allow it to be easily customized for a wide range of environments and applications, showcasing the versatility and power of IoT solutions in the healthcare industry.

4. CHAPTER 4: IMPLEMENTATION

The system's actual implementation, where it is used and its results are tallied, is the focus of this chapter. The measurement and subsequent processing of the heart rate are the primary goals. This essential physiological parameter is thoroughly examined and goes through the required calculations in the ESP32 to extract relevant data. The data is sent to a cloud server for observation and analysis after it has been sufficiently processed.

Wider accessibility and the smooth integration of the heart rate data with other devices and applications are made possible by the transmission of this vital information. Because of this interconnection, a variety of stakeholders can better understand a person's cardiovascular health by monitoring and analyzing the data that has been collected. The data can be interpreted and analyzed more thoroughly by using a wider range of platforms. This could lead to medical professionals or other interested parties taking action.

This chapter, taken as an entire unit, represents the system's successful implementation. The measured heart rate is carefully processed, and the resultant data is transmitted to various devices and applications for observation and analysis.

4.1 Setting up the Electrodes

Prior to applying the sensor pads to the body, it is advised to snap them onto the leads. The measurement is better the closer the pads are to the heart. To assist in determining correct placement, the cables are color-coded.

Red: RA (Right Arm)

Yellow: LA (Left Arm)

Green: RL (Right Leg)

Follow any of these methods to place electrodes on the human body, but I followed the second one because of my electrodes shorter size. The closer to the heart, the better the measurement would be. The cables are color-coded to help you identify a proper placement.

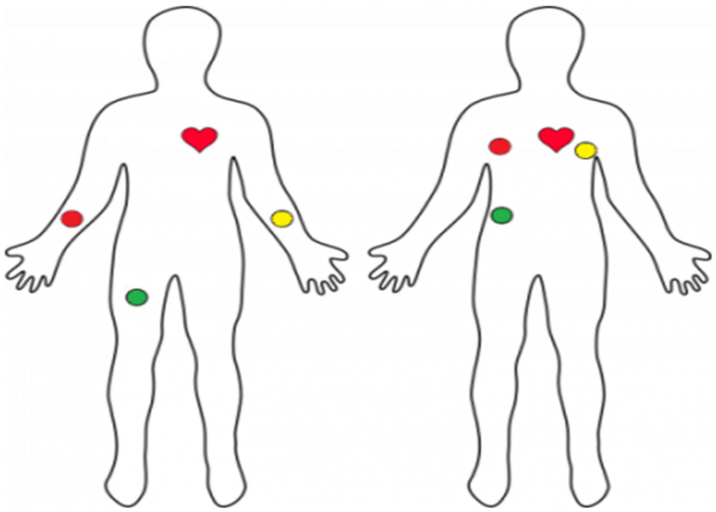

Figure 4-1: Electrode positioning on Human body

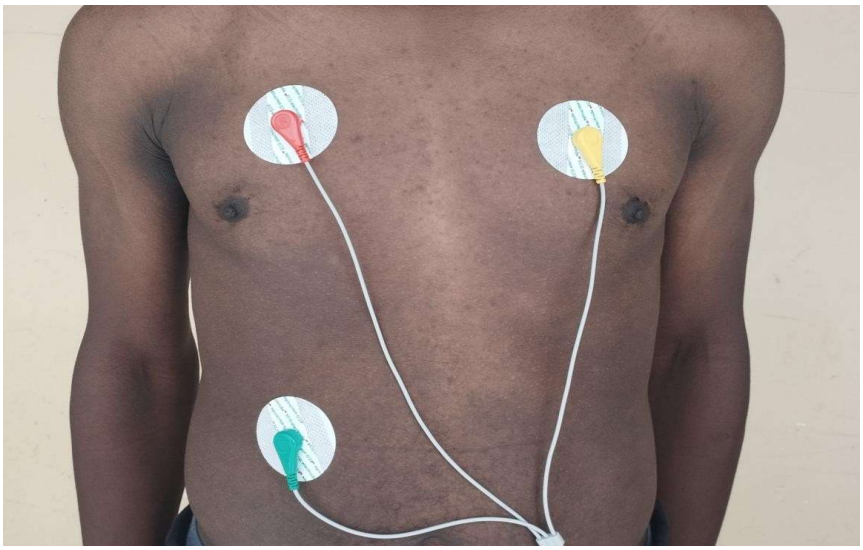

Figure 4-2: Physical placement of Electodes

LA (Left ARM) - The instrumentation amplifier of the IC AD8232 has a positive input (+IN) known as LA. This is where a signal from the electrode attached to the human left arm is received.

RA (Right ARM) - The instrumentation amplifier of IC AD8232 has a negative input (-IN), denoted as RA. This is where a signal is received from the electrode attached to the human body's right arm.

RL (Right LEG) - The green biomedical electrode known as RL is attached to the human body's right leg and serves as an electrode input.

3.5 mm Female Biomedical Electrode Connector Jack

This connector serves as a backup for the RA, LA, and RL pins. Using a 3.5mm male jack, we can connect three electrodes to this connector in place of the RA, LA, and RL pins. The two-electrode method and the three-electrode method are the two ways to find the heartbeat. Whereas the Three Electrodes method makes use of a DC signal, the Two Electrodes method uses an AC signal. This module uses LEAD OFF DETECTION to identify the method that is being used.

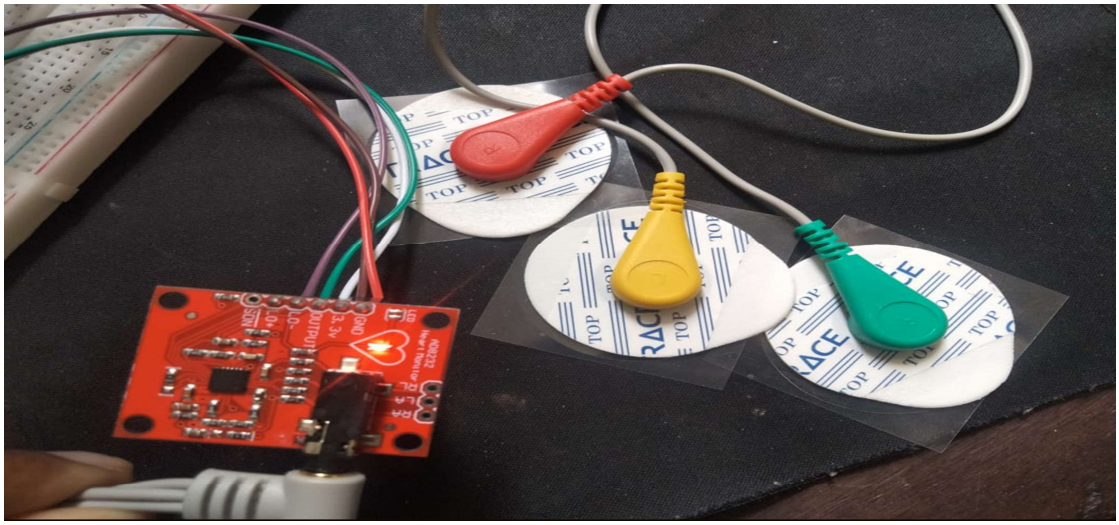

Figure 4-3: Connection between the AD8232 Sensor and the Electrodes

4.2 Programming the ESP32

The ESP32, a flexible microcontroller with built-in Wi-Fi capabilities that makes it perfect for Internet of Things applications, is at the center of this implementation.

First, the code consists of the necessary libraries: PubSubClient.h for MQTT communication, WiFiUDP.h for UDP communication required by NTP, WiFi.h for managing Wi-Fi connectivity, and NTPClient.h for time synchronization. The functions and methods required to manage the system's data processing and communication protocols are provided by these libraries.

Wi-Fi credentials (WIFISSID and PASSWORD), the Ubidots API token (TOKEN), the MQTT client name (MQTT_CLIENT_NAME), and labels for the Ubidots variable and device (VARIABLE_LABEL and DEVICE_LABEL) are among the macros defined in the configuration section. These macros make it simple to customize the code and guarantee that it can be adjusted to work in various setups and environments. The ESP32 pin that is connected to the ECG sensor is designated by the SENSORPIN macro, and it is analogue pin A0.

To store data required for time management, payload construction, and MQTT communication, global variables are declared. These include variables for handling time (epoch seconds, epoch milliseconds, current_millis, current_millis_at_sensordata, and timestamps), buffers for building MQTT messages (payload and topic), and buffers for string conversion of sensor readings and timestamps (str_sensor and str_millis). To control data batching and loop iterations, an integer j is employed as a counter.

The auxiliary functions improve the readability and modularity of the code. In order to facilitate debugging and message reception verification, the callback () function handles incoming MQTT messages by printing the payload and topic to the serial monitor. The ESP32 keeps a steady connection to the MQTT broker thanks to the reconnect () function. This feature gives the system resilience and dependability by attempting to reconnect every two seconds if the connection is lost.

In order to enable serial communication for debugging and Wi-Fi connection configuration, the setup() function initializes the system. A series of dots is printed to the serial monitor until a connection is made, signifying the ESP32's attempt to connect to the designated Wi-Fi network. After connecting, the NTP client is updated and initialized to retrieve the current epoch time in seconds. For accurate timestamping, the seconds are converted to milliseconds. The callback function and the Ubidots broker address are set up in the MQTT client.

The loop() function is the primary code execution cycle, running continuously to keep the system functional. It starts by determining whether the MQTT client is connected, and if not, it tries to reconnect. The batching of sensor data is managed by the j counter, which increases with each loop iteration. The device label is used to create the MQTT topic, and the ECG sensor is repeatedly read to prepare the payload. After each reading is converted to a string, the timestamp

associated with it is determined by adding the milliseconds of elapsed time since the first time sync to the epoch. The precise time stampeding of every data point is guaranteed by this method, which is essential for real-time monitoring.

The payload is formatted as a JSON object, containing sensor values and their timestamps. This structured format allows for easy parsing and processing by the Ubidots cloud platform. The completed payload is then published to the specified MQTT topic on Ubidots, where it can be visualized and analyzed. The system prints the payload to the serial monitor, providing a way to verify the data being sent.

All things considered, this code design represents a complete IoT-based solution for real-time ECG monitoring. Correct data collection, processing, and transmission to a cloud platform are guaranteed by the system's integration of Wi-Fi connectivity, NTP time synchronization, and MQTT communication. Continuous monitoring, data visualization, and prompt intervention are made possible by this, which may improve patient care and health outcomes. The code's robustness and modularity allow it to be easily customized for a wide range of environments and applications, showcasing the versatility and power of IoT solutions in the healthcare industry.

4.3 Monitoring data on Serial Monitor

The ESP32 was successfully connected to the programmed Wi-Fi and assigned an IP address when I opened the serial monitor following a successful upload. Following that, it created a MQTT connection to the Ubidots cloud and established a connection. Subsequently, it began sharing ECG data with a timestamp. Variable Label and four data points with an exact timestamp are included in every data packet.

The code captures three intermediate readings plus one final reading of the ECG sensor data within each iteration of the loop(). These readings are separated by a delay of 150 milliseconds, which ensures that the data points are spaced out over time and can capture the dynamic changes in the ECG waveform. Each sensor reading is timestamped using the current time derived from the NTP synchronized epoch time. The timestamps are calculated by adding the elapsed time since the initial synchronization to the epoch time in milliseconds. This ensures that each data point is accurately time-stamped, allowing for precise reconstruction of the ECG waveform.

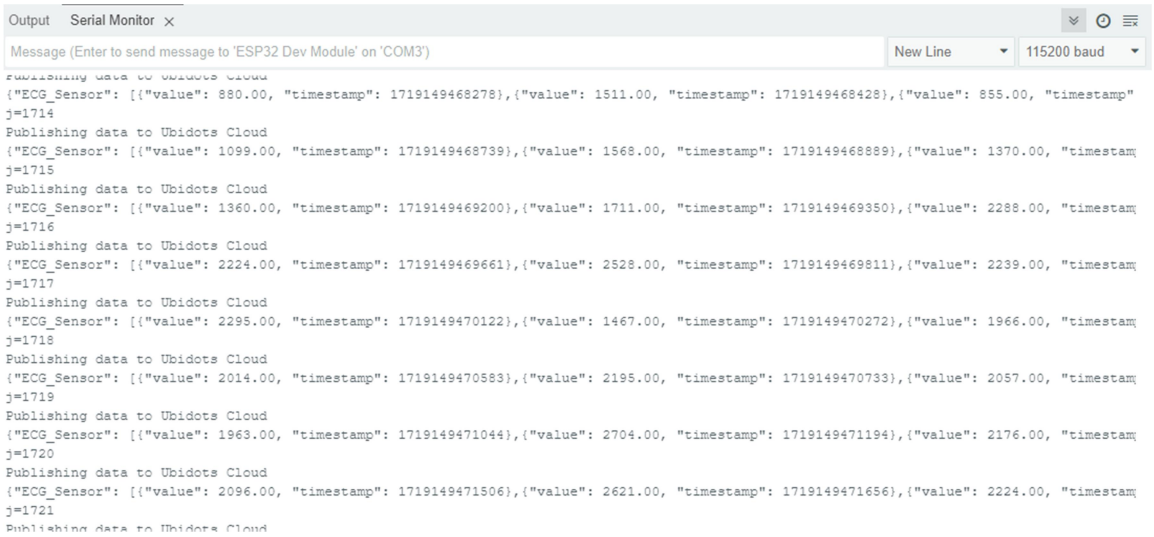

Figure 4-4: Iterations showing data points and timestamps on Serial monitor in Arduino IDE

Given that a typical ECG cycle (heartbeat) occurs roughly every 0.6 to 1.2 seconds (for a heart rate of 50 to 100 beats per minute), the 150-millisecond interval between readings means that the four data points can span a significant portion of the cardiac cycle. This temporal spacing is sufficient to capture the critical features of the ECG waveform but may not be granular enough to capture very fine details. For more detailed analysis, the sampling rate could be increased. Each data point represents the voltage at a specific moment in time, and the timestamps allow for these points to be placed accurately along the time axis. When plotted, these points form a piecewise approximation of the continuous ECG waveform. While not as smooth as a high-frequency sampling would provide, this method still allows for the visualization of the general shape and features of the ECG signal.

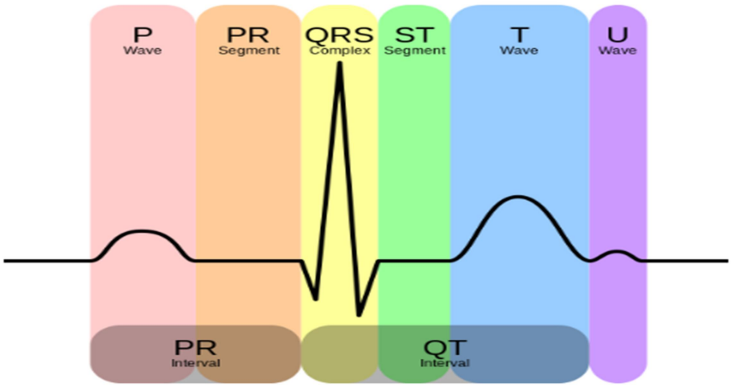

Figure 4-5: PQRSTU wave

4.4 Displaying data on Serial Plotter and Ubidots.

During the implementation stage of the ECG monitoring system, displaying the data points on both the serial plotter and Ubidots plays a crucial role in ensuring accurate data collection, real-time monitoring, and effective analysis. The serial plotter, typically used during the development and debugging phase, provides immediate visual feedback on the raw sensor data, while Ubidots, an IoT cloud platform, offers robust tools for long-term data storage, visualization, and remote access.

The **serial plotter** is an integrated tool within the Arduino IDE that graphically displays data sent from the microcontroller via the serial port. When the ESP32 reads the ECG sensor data and prints these values to the serial monitor, the serial plotter can interpret this data and plot it in real time. As the sensor captures the electrical activity of the heart, the corresponding voltage levels are plotted against time on the serial plotter. This real-time feedback is essential for verifying the functionality of the sensor and ensuring that it accurately captures the ECG waveform. Developers can observe the characteristic features of the ECG signal, such as the P wave, QRS complex, and T wave, and ensure that the signal is free from significant noise or interference.

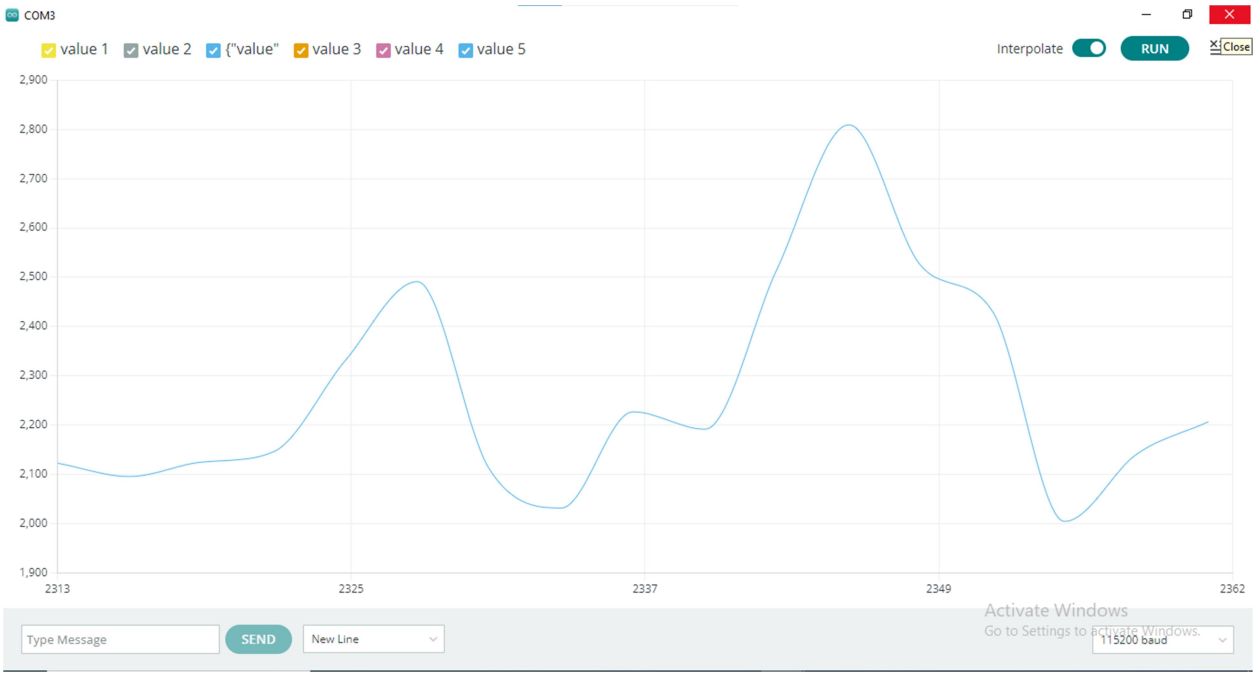

Figure 4-6: ECG Wave on Serial Plotter

Ubidots is an IoT platform that provides extensive tools for data visualization, analysis, and remote monitoring. Once the ESP32 successfully connects to Wi-Fi and transmits the ECG data via MQTT, Ubidots receives this data and stores it in the cloud. The platform's capabilities for displaying and analyzing data are critical during the implementation stage.

The line chart widget, in particular, is well-suited for plotting ECG data. It can display the waveform with high temporal resolution, capturing the nuances of each heartbeat. This helps in identifying patterns and anomalies in the ECG signal. Unlike the serial plotter, which is primarily used for real-time monitoring, Ubidots enables long-term storage and analysis of historical data. Developers and healthcare providers can review past ECG data to identify trends, track the progression of cardiac conditions, and correlate events with specific timestamps.

During implementation, Ubidots' alerting capabilities can be configured to notify developers or healthcare providers of specific events, such as abnormal heart rhythms or sensor malfunctions. These alerts can be sent via email, SMS, or other communication channels. Ubidots allows multiple users to access the data remotely, facilitating collaboration among team members, healthcare providers, and researchers. This is particularly useful during the implementation stage when input from various stakeholders is needed to refine the system.

Figure 4-7: Setting up user interface on Ubidots cloud

5. CHAPTER 5: RESULTS AND TESTING.

This chapter is dedicated to recording the results obtained from the developed heart monitoring system. These outcomes are then contrasted with the information gathered from the cardiac monitoring equipment used in the hospital. The chapter includes a comparative analysis as well as an alert system that is incorporated into the developed system and is meant to notify the user in case of abnormal pulse behavior.

The results obtained from the testing phase of the heart monitoring system project provide comprehensive insights into its performance, reliability, and accuracy. During this phase, the ECG monitoring system, consisting of an ESP32 microcontroller connected to an ECG sensor, is rigorously tested under various conditions to ensure it meets the required standards for medical applications. The primary objective is to verify that the system can accurately capture, process, and transmit ECG data, which can then be visualized and analyzed in real-time via Ubidots.

5.1 Initial Testing and Calibration

The initial testing phase involves setting up the hardware and software components and ensuring their proper functionality. The ECG sensor is connected to the ESP32, which is programmed to read analog signals from the sensor, convert them to digital values, and transmit these values to the Ubidots cloud platform. The system is first tested using the Arduino IDE's serial plotter. This allows developers to visualize the raw ECG data in real time, providing immediate feedback on the sensor's performance. During this stage, the sensor is calibrated to ensure it accurately captures the heart's electrical activity. Calibration involves adjusting the sensor’s sensitivity and filtering out noise to obtain a clean ECG signal that clearly shows the P wave, QRS complex, and T wave.

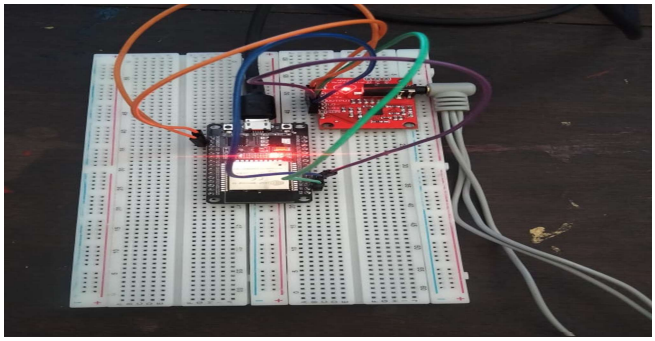

Figure 5-1: Connection between ESP32 and AD8232

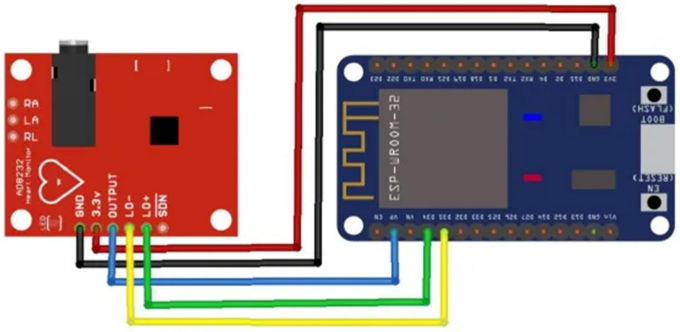

Figure 5-2: Circuit diagram of the heart monitoring system

5.2 Data Transmission and Visualization

Once the initial setup and calibration are complete, the focus shifts to data transmission and visualization. The ESP32, connected to a Wi-Fi network, uses MQTT to send the ECG data to Ubidots. The reliability of this transmission is tested by continuously running the system over extended periods. This ensures that the data packets are consistently delivered without loss or significant delay. In Ubidots, the data is visualized using real-time dashboards. Line charts display the ECG waveform, while other widgets such as gauges and tables provide additional insights into the heart's activity. The system's ability to maintain a stable connection and accurately reflect real-time data in Ubidots is a critical aspect of the testing phase.

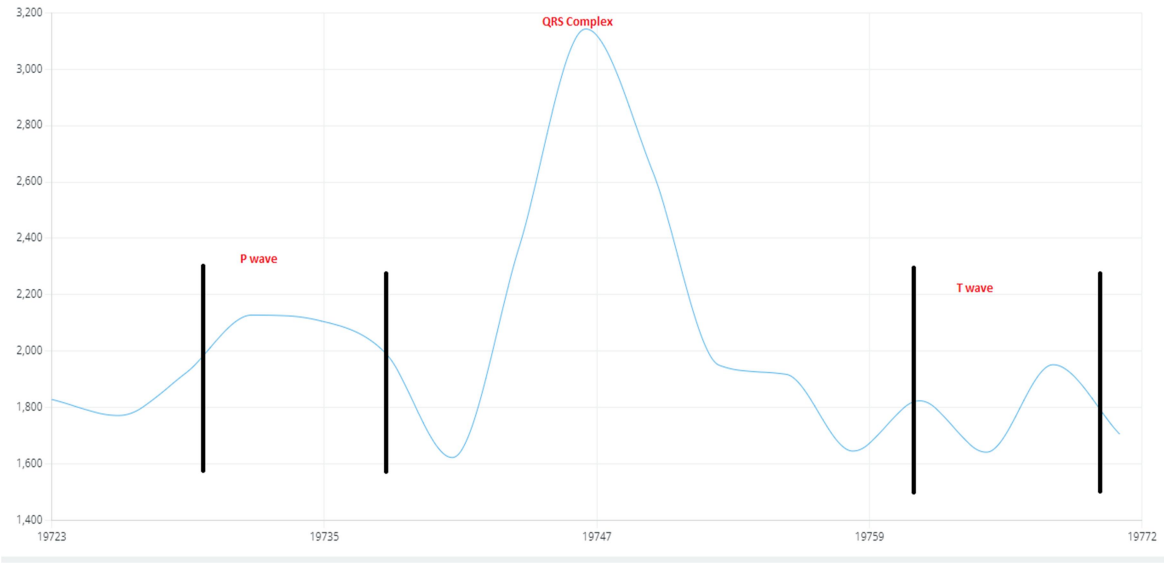

Figure 5-3: Result of ECG wave obtained from the Serial Plotter

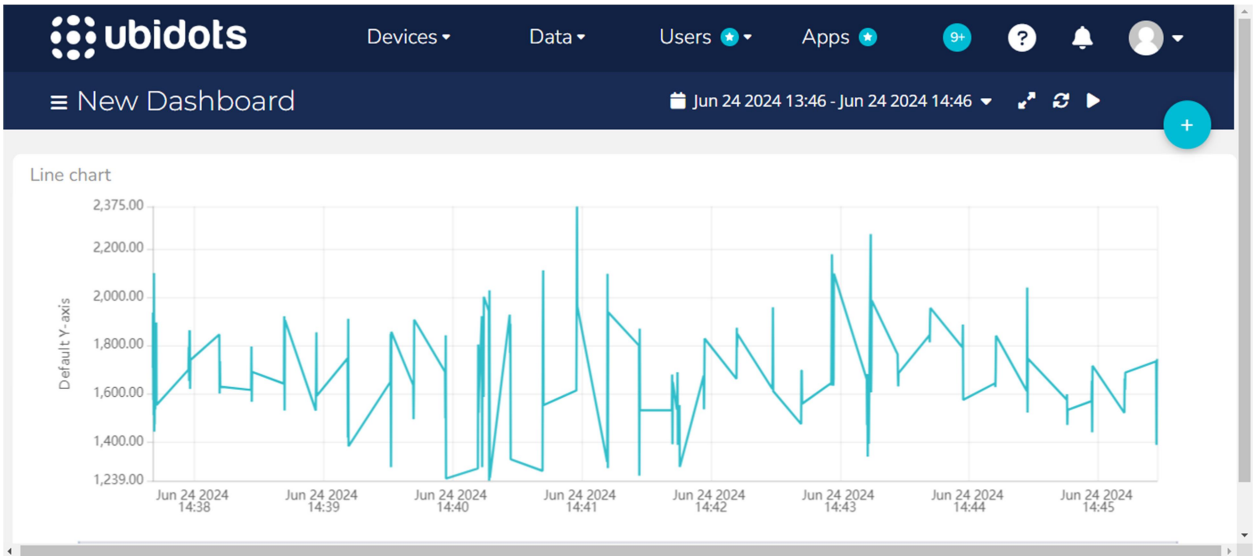

Figure 5-4: Result of the ECG wave on Ubidots over a span of 1 hour

Figure 5-5: Result showing the ECG wave on both Serial Plotter and Ubidots at the same time

5.3 Analog-to-Digital Conversion

The AD8232 sensor outputs an analog voltage signal corresponding to the electrical activity of the heart. This analog signal ranges from 0V to 3.3V and is fed into the ESP32's ADC pin (A0). The ESP32 converts this analog signal into a 12-bit digital value (since the ESP32's ADC resolution is 12 bits), resulting in a digital value range from 0 to 4095.

The relationship between the analog voltage (Vin) and the digital value (Dout) is given by the formula:

$$D_{\text{out}} = \left(\frac{V_{\text{in}}}{V_{\text{ref}}}\right) \times (2^{\text{ADC resolution}} - 1)$$

Where Vref is the reference voltage (3.3V in this case) and the ADC resolution is 12 bits.

$$V_{\text{in}} = \left(\frac{D_{\text{out}}}{2^{\text{ADC resolution}} - 1}\right) \times V_{\text{ref}}$$
$$V_{\text{in}} = \left(\frac{D_{\text{out}}}{4095}\right) \times 3.3$$

When the ESP32 reads the analog signal from the AD8232 sensor, it converts this signal into a digital value between 0 and 4095. This digital value is then used to display the ECG waveform on the serial plotter and Ubidots.

Let's consider a few example digital values and convert them to their corresponding voltages:

1. **Digital Value: 0**
   $$V_{\text{in}} = \left(\frac{0}{4095}\right) \times 3.3 = 0 \text{ V}$$
2. **Digital Value: 2048**
   $$V_{\text{in}} = \left(\frac{2048}{4095}\right) \times 3.3 \approx 1.65 \text{ V}$$
3. **Digital Value: 4095**
   $$V_{\text{in}} = \left(\frac{4095}{4095}\right) \times 3.3 = 3.3 \text{ V}$$

On the Serial Plotter: The serial plotter in the Arduino IDE will plot these digital values directly. For example, a digital value of 2048 will appear as a point in the middle of the plot range (assuming the plot range is 0 to 4095), corresponding to 1.65V from the AD8232 sensor.

On Ubidots: In Ubidots, the digital values are transmitted and visualized in real time. The Ubidots dashboard can be configured to display these values, which represent the analog voltages from the AD8232 sensor.

From Ubidots we are also able to obtain a data sheet of all these digital values including the timestamps which indicate the exact instance at which the heart rate recorded these values i.e.

	A	B	C	D	E
1	date	createdAt	timestamp	context	value
2	2024-06-23 16:35:29.	1.71915E+12	1.71915E+12	{}	4095
3	2024-06-23 16:35:29.	1.71915E+12	1.71915E+12	{}	0
4	2024-06-23 16:35:29.	1.71915E+12	1.71915E+12	{}	4095
5	2024-06-23 16:35:29.	1.71915E+12	1.71915E+12	{}	0
6	2024-06-23 16:35:15.	1.71915E+12	1.71915E+12	{}	2321
7	2024-06-23 16:35:14.	1.71915E+12	1.71915E+12	{}	0
8	2024-06-23 16:35:14.	1.71915E+12	1.71915E+12	{}	3335
9	2024-06-23 16:35:14.	1.71915E+12	1.71915E+12	{}	0
10	2024-06-23 16:35:00.	1.71915E+12	1.71915E+12	{}	0
11	2024-06-23 16:35:00.	1.71915E+12	1.71915E+12	{}	0
12	2024-06-23 16:34:59.	1.71915E+12	1.71915E+12	{}	0
13	2024-06-23 16:34:59.	1.71915E+12	1.71915E+12	{}	0
14	2024-06-23 16:34:44.	1.71915E+12	1.71915E+12	{}	4095
15	2024-06-23 16:34:44.	1.71915E+12	1.71915E+12	{}	4095
16	2024-06-23 16:34:44.	1.71915E+12	1.71915E+12	{}	4095
17	2024-06-23 16:34:44.	1.71915E+12	1.71915E+12	{}	4095
18	2024-06-23 16:34:30.	1.71915E+12	1.71915E+12	{}	1872
19	2024-06-23 16:34:30.	1.71915E+12	1.71915E+12	{}	1411
20	2024-06-23 16:34:29.	1.71915E+12	1.71915E+12	{}	1850
21	2024-06-23 16:34:29.	1.71915E+12	1.71915E+12	{}	1814
22	2024-06-23 16:34:14.	1.71915E+12	1.71915E+12	{}	1423
23	2024-06-23 16:34:14.	1.71915E+12	1.71915E+12	{}	1616
24	2024-06-23 16:34:14.	1.71915E+12	1.71915E+12	{}	1229
25	2024-06-23 16:34:14.	1.71915E+12	1.71915E+12	{}	1644

Figure 5-6: Data Sheet indicating data points on ECG wave exported from Ubidots

5.4 Interpreting the ECG Waveform

The ECG waveform has distinct features represented by the PQRSTU waves:

- **P wave**: A small upward wave before the QRS complex, indicating atrial depolarization.
- **QRS complex**: A series of sharp peaks and troughs representing ventricular depolarization. This is the most prominent part of the ECG.
- **T wave**: A moderate upward wave following the QRS complex, indicating ventricular repolarization.
- **U wave**: Sometimes seen after the T wave, representing further repolarization of the ventricles.

When the AD8232 sensor captures these electrical activities, the resulting analog voltages are converted into corresponding digital values. For instance:

- A small P wave might correspond to digital values ranging from 2000 to 2200 (approximately 1.61V to 1.77V).
- The sharp QRS complex might range from 1000 to 3000 (approximately 0.81V to 2.42V).
- The T wave might range from 1800 to 2200 (approximately 1.45V to 1.77V).

6. CHAPTER 6: CONCLUSION

The final results of the testing phase indicate the system's readiness for deployment. The heart monitoring system demonstrates a high level of accuracy in capturing ECG signals, with minimal discrepancies compared to the reference device. Data transmission to Ubidots is reliable and consistent, with real-time visualization effectively displaying the ECG waveform and associated metrics. The system's robustness is confirmed through stress testing, showing stable performance across various environmental conditions and physiological variations. User feedback is positive, highlighting the system's usability and the clarity of data presentation.

Overall, the testing phase confirms that the heart monitoring system is a reliable and accurate tool for real-time ECG monitoring. Its integration with Ubidots provides a powerful platform for remote health monitoring, enabling healthcare providers to access and analyze patient data from anywhere. The successful testing and validation of the system pave the way for its deployment in real-world healthcare settings, where it can contribute to improved patient care and early detection of cardiac issues.

Practical Implications

- **Heart Rate Monitoring**: The data points can be used to calculate heart rate by analyzing the intervals between successive peaks (R-R intervals) in the ECG waveform.
- **Arrhythmia Detection**: Irregularities in the timing or shape of the captured waveform points can indicate arrhythmias or other cardiac issues. A prolonged QT interval is a risk factor for ventricular arrhythmias and sudden death.
- **Remote Monitoring**: The system allows healthcare providers to monitor patients' heart activity remotely in real time, facilitating timely medical intervention if necessary.

6.1 Recommendations

One critical recommendation regarding electrode placement and the choice of sensor in ECG monitoring is to ensure the electrodes are positioned correctly to obtain accurate and reliable readings. Proper electrode placement is essential for capturing the true electrical activity of the heart, minimizing noise, and reducing artifacts that can distort the ECG waveform. For a typical three-lead ECG configuration, which is often used in portable and simplified ECG devices, the

electrodes should be placed as follows: one electrode on the right arm (RA), one on the left arm (LA), and one on the left leg (LL). Alternatively, in some configurations, the ground electrode may be placed on the right leg (RL).

I could also recommend implementation of a high-pass filter to remove low-frequency components, such as baseline wander, caused by respiration and body movements. Baseline wander can significantly distort the ECG signal, making it difficult to interpret. A cutoff frequency around 0.5 Hz is typically effective in mitigating this issue without affecting the crucial low-frequency components of the ECG waveform, such as the P wave.

REFERENCES

[1] http://www.sciencedirect.com/. International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2017, 6-8 September 2017, Marseille, France.

[2] Partho Kumar Saha, Aisha Singh, Ijraset Journal For Research in Applied Science and Engineering Technology.

[3] https://en.wikipedia.org/wiki/Electrocardiography.

[4] Sahana S Khamitkar M.Tech (CSE) "IoT based System for Heart Rate Monitoring" International Journal of Engineering Research & Technology (IJERT) Vol. 9 Issue 07, July-2020.

[5] Ogunduyile O.O1., Zuva K2 ., Randle O.A3., Zuva T4 "Ubiquitous Healthcare Monitoring System Using Integrated Triaxial Accelerometer, Sp O2and Location Sensors" International Journal of UbiComp (IJU), Vol.4, No.2, April 2013

[6] Sani Abba and Abubakar Mohammed Garba "An IoT-Based Smart Framework for a Human Heartbeat Rate Monitoring and Control System" 15 November 2019– 30 November 2019

[7] Amrita Chatterjee, Snigdha Chowdhury Kolay "IoT Based Pulse Oximeter System"

[8] Krikler DM. Historical aspects of electrocardiography. Cardiol Clin. 1987 Aug;5(3):349-55. [PubMed]

[9] Fye WB. A history of the origin, evolution, and impact of electrocardiography. Am J Cardiol. 1994 May 15;73(13):937-49. [PubMed]

[10] Rundo F, Conoci S, Ortis A, Battiato S. An Advanced Bio-Inspired PhotoPlethysmoGraphy (PPG) and ECG Pattern Recognition System for Medical Assessment. Sensors (Basel). 2018 Jan 30;18(2) [PMC free article] [PubMed]

[11] Spicer DE, Henderson DJ, Chaudhry B, Mohun TJ, Anderson RH. The anatomy and development of normal and abnormal coronary arteries. Cardiol Young. 2015 Dec;25(8):1493-503. [PubMed]

[12] https://techvify-software.com/iot-in-healthcare-exploring-definitions/

[13] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all

APPENDICES:

APPENDIX A: ARDUINO CODE

```
#include <WiFi.h>
#include <WiFiUdp.h>
#include <PubSubClient.h>
#include <NTPClient.h>


#define WIFISSID "xxxxxxxxxxxxx" // Enter WifiSSID here
#define PASSWORD "xxxxxxx" // Enter password here
#define TOKEN "xxxxxxxxxxxxxxxxxxxxxxxx" // Ubidots' TOKEN
#define MQTT_CLIENT_NAME "xxxxxxxxx" // MQTT client Name
// * Define Constants
#define VARIABLE_LABEL "xxxxxxxxxxx" // ubidots variable label
#define DEVICE_LABEL "xxxxxxxxxxxxxxxxxxx" // ubidots device label


#define SENSORPIN A0 // Set the A0 as SENSORPIN

char mqttBroker[]  = "industrial.api.ubidots.com";
char payload[10000];
char topic[150];
// Space to store values to send
char str_sensor[10];
char str_millis[20];
double epochseconds = 0;
double epochmilliseconds = 0;
double current_millis = 0;
double current_millis_at_sensordata = 0;
double timestampp = 0;
int j = 0;
/****************************************
   Auxiliar Functions
 ****************************************/
WiFiClient ubidots;
PubSubClient client(ubidots);
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org");// Space to store values to send

void callback(char* topic, byte* payload, unsigned int length) {
  char p[length + 1];
  memcpy(p, payload, length);
  p[length] = NULL;
```

```
  Serial.write(payload, length);
  Serial.println(topic);
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.println("Attempting MQTT connection...");

    // Attemp to connect
    if (client.connect(MQTT_CLIENT_NAME, TOKEN, "")) {
      Serial.println("Connected");
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 2 seconds");
      // Wait 2 seconds before retrying
      delay(2000);
    }
  }
}

/**************************************
   Main Functions
 **************************************/
void setup() {
  Serial.begin(115200);
  WiFi.begin(WIFISSID, PASSWORD);
  // Assign the pin as INPUT
  pinMode(SENSORPIN, INPUT); // Wi-Fi Connection

  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());
  timeClient.begin();
  client.setServer(mqttBroker, 1883);
```

```
  client.setCallback(callback);
  timeClient.update();
  epochseconds = timeClient.getEpochTime();
  epochmilliseconds = epochseconds * 1000;
  Serial.print("epochmilliseconds=");
  Serial.println(epochmilliseconds);
  current_millis = millis();
  Serial.print("current_millis=");
  Serial.println(current_millis); // NTP Time Synchronization
}

void loop() {
  if (!client.connected()) {
    reconnect();
    j = 0;
  }
  //sprintf(payload, "%s", "{\"ECG_Sensor_data\": [{\"value\":1234,
\"timestamp\": 1595972075},{\"value\":1111, \"timestamp\":
1595971075},{\"value\":2222, \"timestamp\": 1595970075}]}");
  j = j + 1;
  Serial.print("j=");
  Serial.println(j);
  sprintf(topic, "%s%s", "/v1.6/devices/", DEVICE_LABEL);
  sprintf(payload, "%s", ""); // Cleans the payload
  sprintf(payload, "{\"%s\": [", VARIABLE_LABEL); // Adds the variable label
  for (int i = 1; i <= 3; i++)
  {
    float sensor = analogRead(SENSORPIN);
    dtostrf(sensor, 4, 2, str_sensor);
    sprintf(payload, "%s{\"value\":", payload); // Adds the value
    sprintf(payload, "%s %s,", payload, str_sensor); // Adds the value
    current_millis_at_sensordata = millis();
    timestampp = epochmilliseconds + (current_millis_at_sensordata -
current_millis);
    dtostrf(timestampp, 10, 0, str_millis);
    sprintf(payload, "%s \"timestamp\": %s},", payload, str_millis); // Adds the
value
    delay(150);
  }// Data Reading

  float sensor = analogRead(SENSORPIN);
  dtostrf(sensor, 4, 2, str_sensor);
  current_millis_at_sensordata = millis();
  timestampp = epochmilliseconds + (current_millis_at_sensordata -
current_millis);
```

```
  dtostrf(timestampp, 10, 0, str_millis);
  sprintf(payload, "%s{\"value\":%s, \"timestamp\": %s}]}", payload, str_sensor,
str_millis);
  Serial.println("Publishing data to Ubidots Cloud");
  client.publish(topic, payload);
  Serial.println(payload); // Payload construction and Publishing
  // client.loop();


}
```

APPENDIX B: BILL OF MATERIALS

The cost considerations for the materials used is discussed in the table below.

Component	***Quantity Used***	***Cost Per Unit (Ksh.)***	***Cost(Ksh)***
ESP32 Module	1	1600	1600
ECG Module AD8232	1	1500	1500
ECG Electrodes and Connector	3(Electrodes) 1(Connector)	1600	1600
Bread boards	2	250	500
Connecting wires		350	350
Total			5550