CARDIAC ATTACK DETECTION AND INTERVENTION BASED ON IOT

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

CHALLAGUNDLA SRI RAM	(21UECM0043)	**(20439)**
KADIYALA YESWANTH	(21UECM0110)	**(20516)**
YENIMIREDDY AKHILESWAR REDDY	(21UECM0267)	**(20514)**

Under the guidance of
Dr.M.KAVITHA,M.E.,Ph.D.,
PROFESSOR-CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CARDIAC ATTACK DETECTION AND INTERVENTION BASED ON IOT

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

CHALLAGUNDLA SRI RAM	(21UECM0043)	**(20439)**
KADIYALA YESWANTH	(21UECM0110)	**(20516)**
YENIMIREDDY AKHILESWAR REDDY	(21UECM0267)	**(20514)**

Under the guidance of
Dr.M.KAVITHA,M.E.,Ph.D.,
PROFESSOR-CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled ”CARDIAC ATTACK DETECTION AND INTERVENTION BASED ON IOT” CHALLAGUNDLA SRI RAM (21UECM0043), KADIYALA YESWANTH (21UECM0110),YENIMIREDDY AKHILESWAR REDDY (21UECM0267)” has been carried out under my supervision and that this work has not been submitted else where for a degree

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(CHALLAGUNDLA SRI RAM)

Date: / /

(KADIYALA YESWANTH)

Date: / /

(YENIMIREDDY AKHILESWAR REDDY)

Date: / /

APPROVAL SHEET

This project report entitled ”CARDIAC ATTACK DETECTION AND INTERVENTION BASED ON IOT” by ”CHALLAGUNDLA SRI RAM (21UECM0043),KADIYALA YESWANTH (21UECM0110),YENIMIREDDY AKHILESWAR REDDY (21UECM0267)” is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Dr.M.KAVITHA,M.E.,Ph.D.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal supervisor Dr.M.KAVITHA.M.E.,Ph.D.,** for her cordial support, valuable information and guidance, she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. U.HEMAVATHI, M.E., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

CHALLAGUNDLA SRI RAM **(21UECM0043)**

KADIYALA YESWANTH **(21UECM0110)**

YENIMIREDDY AKHILESWAR REDDY REDDY **(21UECM0267)**

ABSTRACT

Cardiac diseases are a leading cause of mortality worldwide, emphasizing the critical need for real-time monitoring and early detection systems. This paper proposes a novel approach for cardiac attack detection leveraging Internet of Things (IoT) technology and machine learning algorithms. The system integrates various sensor modules, including the Arduino Pro Mini 328 microcontroller, heart rate sensor, BMP180 barometric pressure sensor, NEO-6M GPS module, and LED display module. The Arduino Pro Mini 328 serves as the central processing unit, collecting data from the sensors and executing machine learning algorithms for cardiac attack detection. The heart rate sensor continuously monitors the user's heart rate, while the BMP180 sensor measures environmental parameters such as temperature and altitude, providing contextual information for the analysis. The NEO-6M GPS module tracks the user's location in real-time, enabling emergency services to respond promptly in case of an incident. The LED display module provides visual feedback to the user, displaying vital information and alerts. The collected sensor data is transmitted wirelessly to a central server or cloud platform via IoT connectivity, facilitating remote monitoring and data analysis. Machine learning algorithms, trained on historical cardiac data, analyze the sensor data in real-time to detect anomalies indicative of a cardiac attack. The proposed system offers several advantages, including early detection of cardiac abnormalities, remote monitoring capabilities, and prompt emergency response. By leveraging IoT technology and machine learning algorithms, the system provides a proactive approach to cardiac health monitoring, potentially saving lives and improving patient outcomes.

Keywords:

Internet of Things,Real-time data analysis, Predictive analytics,Heart rate variability,Ambient intelligence,healthcare systems

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

S.NO	ABBREVIATIONS	DEFINITION
1	BPM	Beats Per Minute
2	IoT	Internet Of Things
3	LED	Light Emitting Diode
4	GPS	Global Positioning System

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

The initiative to use IoT technology to monitor and identify heart attacks is extremely important to society. It promises to transform cardiac healthcare and maybe save countless lives by utilizing IoT sensors and real-time data processing. For those suffering from cardiovascular disorders, early diagnosis of heart attacks can lower death rates, save healthcare expenses, and enhance overall quality of life. Moreover, proactive self-care practices are encouraged when patients are equipped with real-time insights regarding their cardiac health condition. In addition to improving access to healthcare, especially for marginalized communities, the project provides important data for medical research, which advances the development of cardiovascular disease prevention and treatment methods. In the end, this research has the potential to significantly influence society by advancing patient outcomes, encouraging early intervention.

Advancements in sensor technology have played a crucial role in enhancing the capabilities of IoT-enabled cardiac monitoring systems.Emphasized the importance of wearable sensors for real-time data capture,integrated wearable devices with wireless modules for data transmission. These studies underscored the significance of sensors such as BPM sensor in enabling continuous monitoring of vital signs. Data analysis techniques have been a focal point in IoT-enabled cardiac monitoring research, with studies employing machine learning algorithms for real-time analysis and anomaly detection.Utilized machine learning for data analysis in their wearable monitoring system, integrated data processing units for real-time signal processing. These approaches enable effective detection of cardiac anomalies and provide actionable insights for clinical decision-making.

1.2 Aim of the project

The project aims to develop a cardiac attack detection system leveraging Internet of Things (IoT) technology. By continuously monitoring vital signs such as heart rate and environmental conditions using sensor modules, the system can quickly identify abnormalities and trigger alerts in real-time. This enables timely intervention and emergency response, potentially improving patient outcomes. Through the integration of IoT devices and wireless connectivity, the project seeks to enhance the efficiency and effectiveness of cardiac health monitoring, ultimately contributing to saving lives.

1.3 Project Domain

The main project domain of an IoT-Enabled Cardiac Attack Detection and Intervention system lies at the intersection of healthcare, medical technology, and Internet of Things (IoT) innovation. It encompasses the development and implementation of advanced sensor technologies, wearable devices, and real-time data analytics to address critical challenges in cardiac care. Within the healthcare domain, the project focuses on improving patient outcomes and reducing mortality rates associated with cardiac events. By leveraging IoT technology, it seeks to enable early detection of cardiac abnormalities and timely intervention, ultimately enhancing the quality of care delivered to patients experiencing cardiac emergencies. In the medical technology domain, the project aims to innovate new solutions for cardiac monitoring and intervention that leverage the capabilities of IoT devices and sensors. This involves the design and development of wearable monitors, implantable devices, and remote monitoring systems that seamlessly integrate into existing healthcare infrastructure.

1.4 Scope of the Project

The project aims to develop an innovative IoT-based system dedicated to cardiac attack detection and intervention. Central to this initiative is the design and implementation of wearable sensors capable of continuously monitoring key vital signs, including heart rate, blood pressure, readings. These sensors will be equipped to transmit real-time data to a centralized platform where sophisticated algorithms will analyze the information. The algorithms will be programmed to detect any abnor-

mal patterns or fluctuations in the vital signs that could potentially indicate a cardiac event or distress.

Upon identifying such anomalies, the system will automatically trigger a series of predefined responses aimed at providing immediate assistance and intervention. This could involve notifying medical professionals or emergency services with the precise location of the individual in distress, thanks to integrated GPS tracking capabilities. Simultaneously, the system may also offer real-time first aid instructions or alerts to the user, guiding them through potentially life-saving actions until professional help arrives.

Chapter 2

LITERATURE REVIEW

[1]Chen.S et.al developed a wearable IoT system for continuous heart rate monitoring integrated into home environments. The system utilizes wearable sensors and ambient monitoring devices to collect comprehensive physiological data from individuals at risk of cardiac events. These sensors may include pulse oximeters, and blood pressure cuffs, among others. Data collected from these sensors are transmitted wirelessly to a centralized hub, where advanced machine learning algorithms analyze the data to predict the likelihood of a cardiac attack. Based on the risk assessment, the system can provide personalized recommendations for preventive measures and lifestyle interventions, empowering individuals to proactively manage their cardiac health.

[2]Chen.X et.al proposed a real-time heart rate monitoring system using IoT technology for early detection and intervention in cardiac emergencies. The system incorporates wearable monitors capable of transmitting data to a centralized cloud platform in real-time. Deep learning models deployed on the cloud analyze the signals for abnormalities associated with cardiac events, such as ST-segment elevation indicative of myocardial infarction. Upon detection, the system generates alerts to notify healthcare professionals, enabling rapid diagnosis and intervention. Additionally, the cloud platform facilitates longitudinal monitoring of patients' cardiac health, enabling healthcare providers to track disease progression and treatment efficacy over time.

[3]Zhao et.al developed an IoT-enabled heart rate monitoring system using wearable devices for remote heart rate monitoring for proactive management of cardiac conditions. The device, implanted within the body, continuously monitors cardiac parameters such as heart rate, rhythm, and blood pressure. Using wireless communication technologies, the device transmits real-time data to external monitoring systems, where advanced algorithms analyze the data for signs of cardiac distress or arrhythmias. In case of detected abnormalities, the device can initiate

therapeutic interventions, such as pacing or defibrillation, to restore normal cardiac function. By providing proactive monitoring and intervention, the smart implantable device aims to improve patient outcomes and reduce the risk of adverse cardiac events.

[4] Wang.L et al. proposed a novel approach for cardiac attack detection leveraging IoT-enabled wearable devices. The system continuously monitors vital signs such as heart rate, blood pressure, and GPS signals in real-time. Advanced algorithms analyze the collected data to detect anomalies indicative of a cardiac event, such as arrhythmias or ischemic episodes. Upon detection, the system triggers automated alerts to healthcare providers and emergency services, enabling swift intervention. Additionally, the wearable devices may incorporate features such as GPS tracking to provide precise location information in case of emergencies, ensuring timely assistance to individuals experiencing cardiac events.

[5] Wang et.al proposed a wearable IoT device for real-time heart rate monitoring platform for remote patient monitoring and management. The system integrates wearable sensors, mobile applications, and cloud-based analytics to track patients' cardiac health in real-time. Wearable devices continuously monitor vital signs and transmit data to a centralized cloud platform, where machine learning algorithms analyze the data for abnormalities indicative of cardiac events. Healthcare providers can remotely access patient data and receive automated alerts in case of emergencies, enabling timely intervention. Additionally, the platform facilitates communication and collaboration among patients, caregivers, and healthcare professionals, empowering patients to actively participate in their cardiac care and improve treatment outcomes.

[6]Wang et.al proposed cloud-based IoT system for remote heart rate for continuous monitoring of cardiac health in elderly populations. The system integrates wearable devices capable of measuring vital signs, activity levels, and environmental factors relevant to cardiovascular health. These sensors may include accelerometers, heart rate monitors, and environmental sensors for monitoring air quality and temperature. By collecting and analyzing longitudinal data from multiple sources, the system enables early detection of cardiac abnormalities and provides personalized interventions tailored to individual risk profiles. Additionally, the system may

incorporate features such as fall detection and emergency response capabilities to enhance the safety and well-being of elderly individuals with cardiovascular conditions.

[7]Yang et.al introduced a Design of a wearable heart rate monitoring system rehabilitation system aimed at improving post-attack recovery outcomes. The system comprises wearable sensors, mobile health applications, and remote monitoring capabilities. Following a cardiac event, patients are provided with wearable devices to monitor vital signs and physical activity levels during the rehabilitation process. Real-time feedback and personalized exercise programs delivered through the mobile application encourage adherence to rehabilitation protocols and facilitate remote monitoring by healthcare providers. By actively engaging patients in their recovery, the system aims to improve cardiovascular health and reduce the risk of future cardiac events.

[8]Jiang et.al introduced an IoT-based remote monitoring system for cardiac rehabilitation for secure and interoperable sharing of cardiac health data. The system ensures the integrity and privacy of sensitive medical information while enabling seamless communication and collaboration among patients, healthcare providers, and researchers. Wearable sensors collect physiological data from patients, which are encrypted and stored on a blockchain network. Smart contracts govern data access and sharing permissions, ensuring compliance with privacy regulations and ethical standards. By leveraging blockchain technology, the platform facilitates data-driven insights and evidence-based interventions to improve cardiac care delivery and outcomes, while maintaining the security and confidentiality of patient information.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The existing systems for cardiac attack detection and intervention primarily consist of traditional medical monitoring devices, such as Holter monitors and implantable cardiac devices, which monitor heart rhythms over extended periods but lack real-time remote monitoring capabilities. Emergency response services like 911 are often relied upon for immediate assistance, but these can result in delayed response times. Some systems utilize telemedicine platforms for remote monitoring by healthcare providers, while others leverage mobile applications for basic heart rate using smartphone sensors. However, there is a growing need for more advanced, integrated solutions that combine real-time monitoring, immediate intervention, and data analytics to improve the timely detection and management of cardiac events.

3.2 Proposed System

The proposed system for cardiac attack detection and intervention leverages IoT technology to overcome the limitations of existing systems. It features wearable devices equipped with advanced sensors for real-time monitoring of vital signs like heart rate and blood pressure. The collected data is transmitted to a centralized platform where sophisticated algorithms analyze it for abnormal patterns indicative of a cardiac event. Upon detection, the system triggers automated alerts to medical professionals or emergency services, providing the individual's precise location through integrated GPS tracking. This integrated approach aims to facilitate immediate and effective responses, potentially saving lives by reducing response times and enhancing the overall management of cardiac events.

3.3 Feasibility Study

3.3.1 Economic Feasibility

The economic feasibility of implementing a cardiac attack detection and intervention system using IoT technology requires a thorough evaluation of development, operational, and revenue aspects. Initial investments in research, design, and manufacturing of IoT-enabled wearable devices, along with centralized monitoring platforms and software, need to be carefully assessed. Ongoing operational costs, such as data storage, server maintenance, and software updates, must be managed efficiently to ensure long-term sustainability.

Revenue potential hinges on a comprehensive market analysis to identify the target market size, demand, and growth opportunities. Effective pricing strategies, potential partnerships, and market penetration strategies will play a vital role in revenue generation. Moreover, exploring various funding sources, including grants, investments, or collaborations with healthcare providers, can provide additional financial support, enhancing the project's economic viability. Overall, a balanced approach to managing costs and maximizing revenue will be essential to establish the economic feasibility and success of the IoT-based cardiac monitoring system.

3.3.2 Technical Feasibility

The technical feasibility of developing a cardiac attack detection and intervention system using IoT technology hinges on several critical factors. Firstly, the availability and reliability of advanced sensor technology capable of accurately monitoring vital signs such as heart rate and blood pressure. in real-time need to be assessed. These sensors must be compatible with IoT connectivity protocols to enable seamless data transmission to a centralized monitoring platform.

Secondly, the scalability and interoperability of the system's infrastructure are essential for accommodating a growing number of users and integrating with existing healthcare systems. Robust data storage, processing capabilities, and cybersecurity measures must be in place to safeguard sensitive medical information and ensure uninterrupted service.

Lastly, the development of sophisticated algorithms for real-time data analysis and anomaly detection is paramount for timely identification of potential cardiac events. These algorithms will enable automated alerts and notifications to healthcare

providers or emergency services, facilitating prompt intervention and potentially improving patient outcomes. Overall, a comprehensive technical assessment will validate the system's capabilities and lay the foundation for successful implementation and deployment in clinical settings.

3.3.3 Social Feasibility

The social feasibility of a cardiac attack detection and intervention system using IoT technology involves several critical aspects that impact its acceptance and adoption within society. Public awareness campaigns and educational initiatives are essential to highlight the benefits of early detection and intervention, fostering a proactive approach to cardiovascular health.

User acceptance is pivotal, requiring a deep understanding of user needs and preferences to develop intuitive and user-friendly interfaces. Incorporating user feedback through testing and iterative design processes can enhance the system's usability and overall satisfaction.

Ensuring equitable access to the technology is vital to address healthcare disparities. Affordable pricing strategies, flexible payment options, and collaborations with healthcare organizations can help extend the system's reach to underserved communities, promoting inclusivity.

Lastly, addressing concerns related to privacy, confidentiality, and data security is crucial to build trust among users. Implementing robust data protection measures and transparent communication about privacy practices can alleviate concerns and encourage broader adoption of the IoT-based cardiac monitoring system. Overall, addressing these social considerations is key to the successful implementation and societal acceptance of the proposed technology.

Should be described related to project only

3.4 System Specification

3.4.1 Hardware Specification

• Heart Rate Pluse Sensor.
• NEO-6M GPS Module.
• BMP 180.
• Arduino Pro Mini 328.

• Lithium Polymer Battery.
• capacitor Ceramic 100 nF.
• Jumper Wires.
• Monochrome O LED Graphic Display.

3.4.2 Software Specification

• Operating System: Windows 7 or higher.
• Programming language:Machine Learning.
• Platform :Arduino IDE.

3.4.3 Standards and Policies

The Arduino Integrated Development Environment (IDE) is a software platform used to write, compile and upload code to Arduino microcontrollers like the Arduino UNO. It provides an easy-to-use interface for programming and developing projects for various Arduino-compatible boards.Standard Used: ISO 5667-11.

Chapter 4

METHODOLOGY

4.1 General Architecture for Cardiac Attack Detection

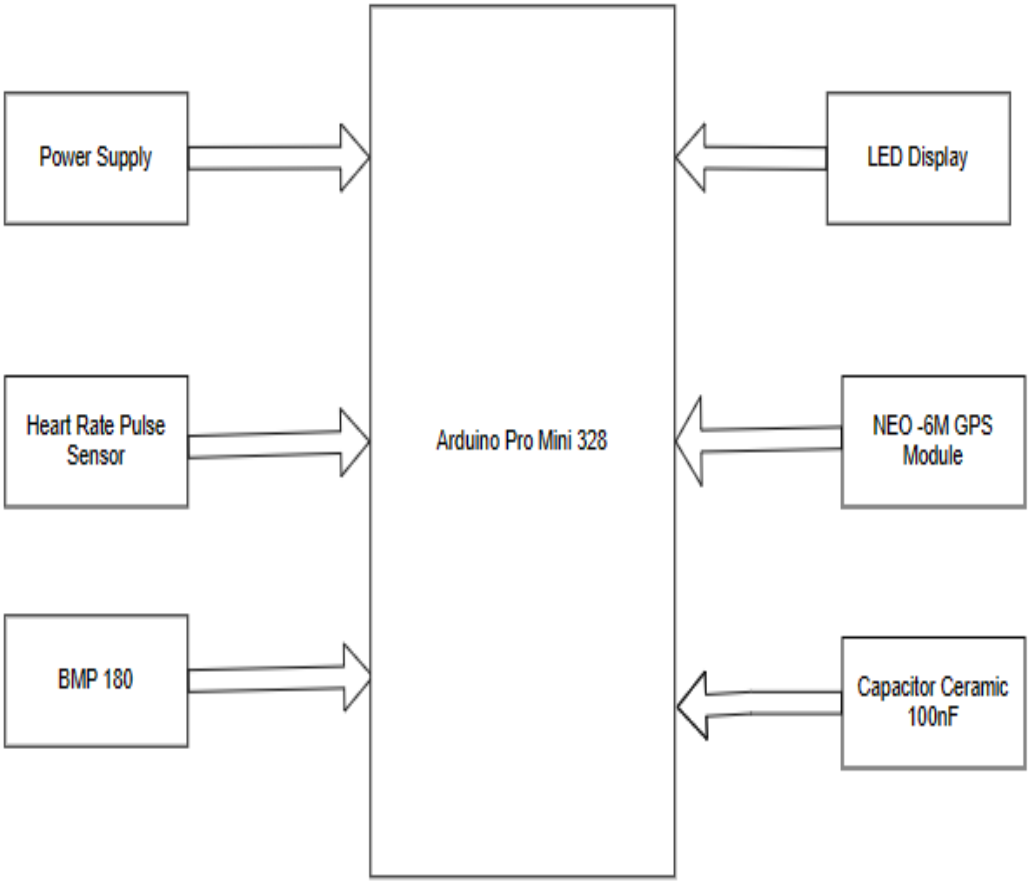

Figure 4.1: **General Architecture for Cardiac Attack Detection**

figure 4.1 shows the interconnected components of the system, including power supply, microcontroller (Arduino Pro Mini), sensors (BMP180 pressure sensor, Ublox NEO-6M GPS module, and heart rate pulse sensor), and display (monochrome 128x32 I2C OLED graphic display).

4.2 Design Phase

4.2.1 Data Flow Diagram for Cardiac Attack Detection

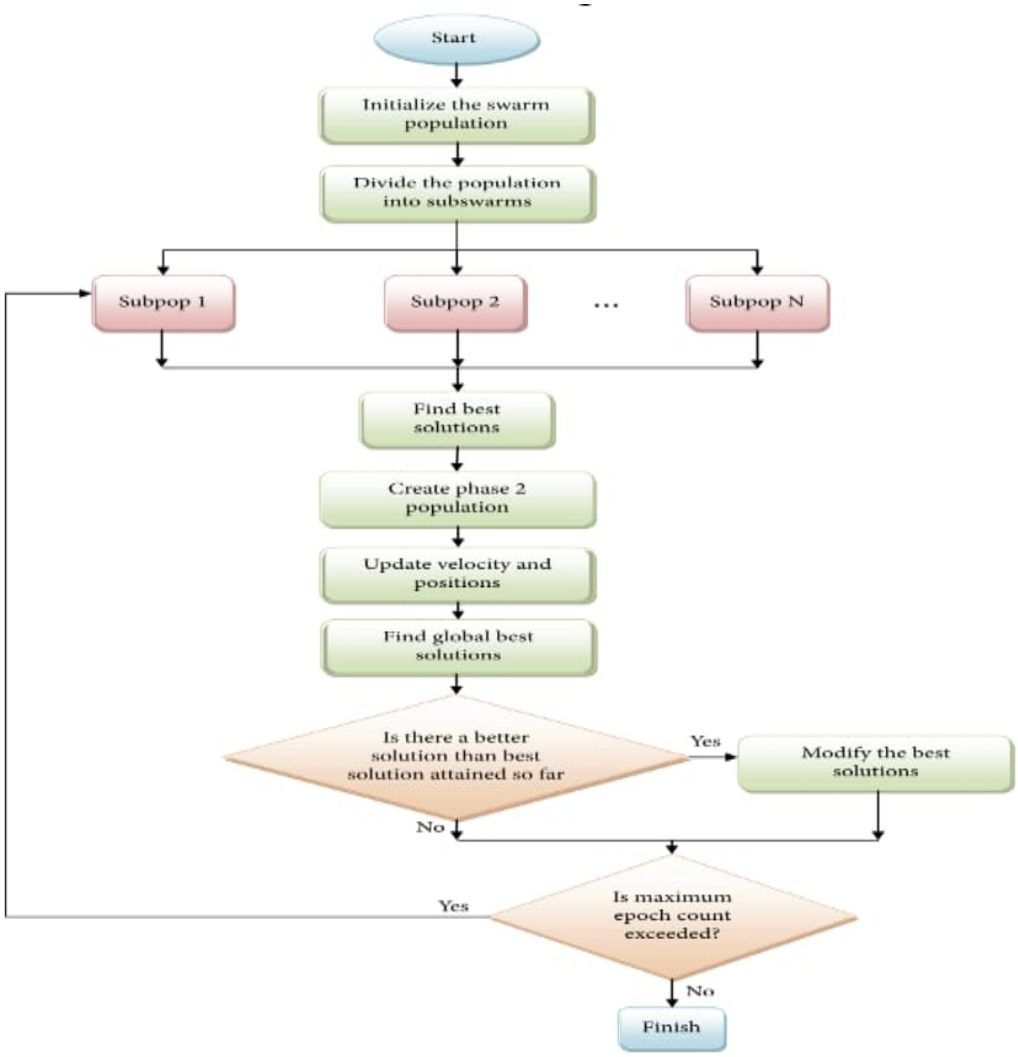

Figure 4.2: **Data Flow Diagram for Cardiac Attack Detection**

Figure 4.2 shows the Sensors: Capture physiological data such as heart rate, blood pressure, and GPS coordinates.
Arduino Pro Mini: Receives sensor data, processes and analyzes it.
Display: Receives processed data from Arduino for visualization.
Data Processing Analysis: In the Arduino, algorithms process sensor data, detect patterns, and analyze cardiac health status.
Communication Interface: Manages the communication between Arduino and the

internet gateway.

4.2.2 Class Diagram for Cardiac Attack Detection

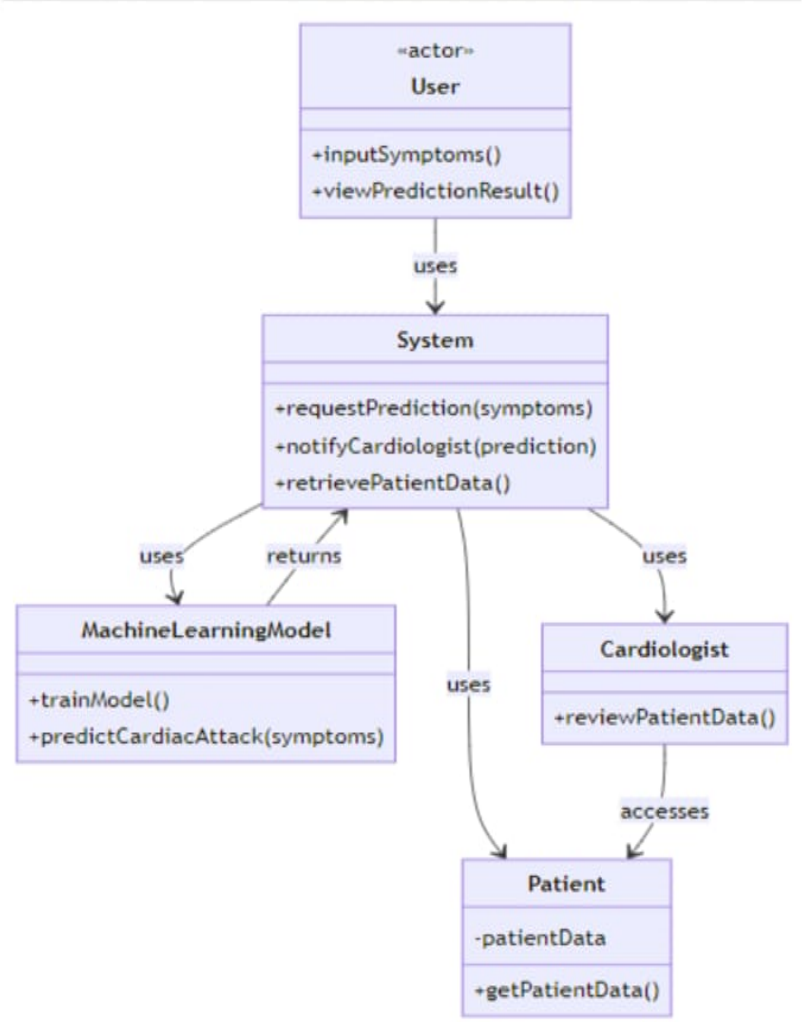

Figure 4.3: **Class Diagram of Cardiac Attack Detection**

Figure 4.3 Represents the user interacting with the system. They can record health data, monitor heart rate, and notify emergency services.
Sensor: Abstract class representing various sensors (e.g., heart rate sensor, GPS module) used to collect physiological data.
Arduino: Represents the microcontroller responsible for processing the sensor data and managing system operations. Display: Represents the display component used to visualize information for the user.

4.2.3 Sequence Diagram for Cardiac Attack Detection

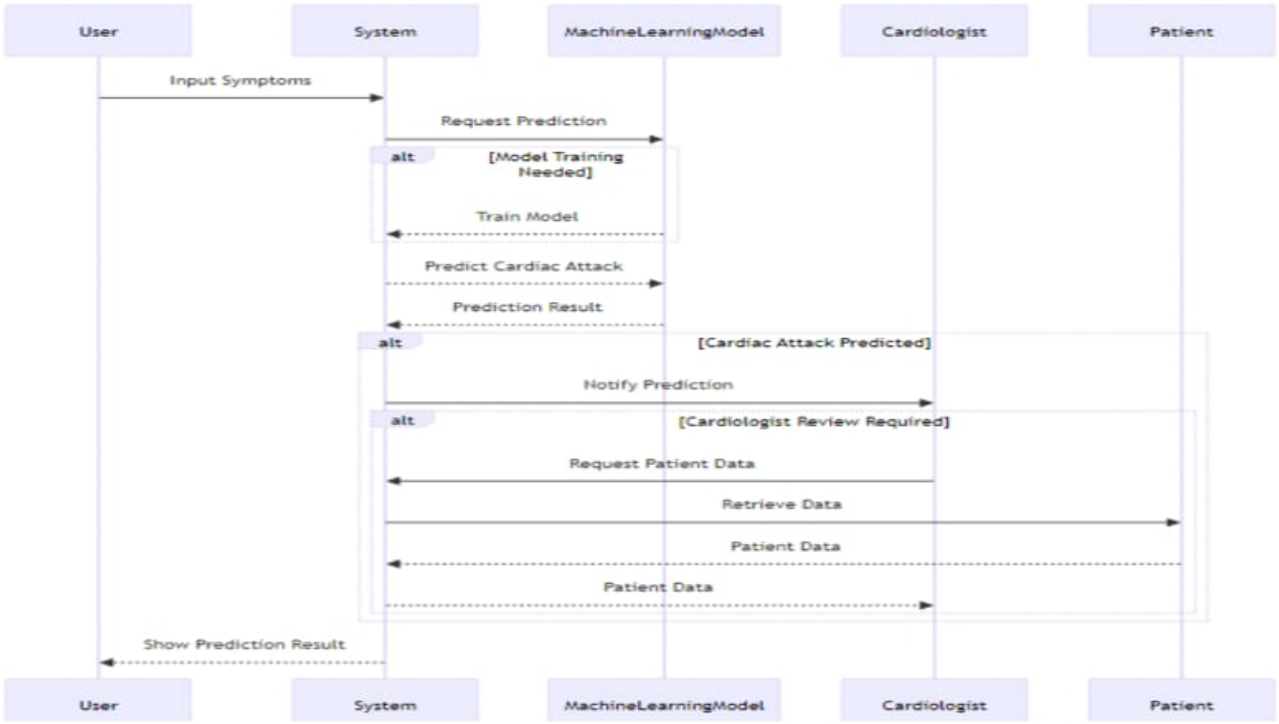

Figure 4.4: **Sequence Diagram for Cardiac attack detection**

Figure 4.4 shows the sequence diagram depicts interactions between the user, sensor, Arduino, and display components. The User lifeline initiates heart rate monitoring. Concurrently, the Sensor reads the heart rate data. Once the heart rate data is read, the Arduino processes it. Concurrently, the Display updates with the processed heart rate data. Each lifeline represents a parallel process occurring independently in the system.

4.2.4 Activity Diagram for Cardiac Attack Detection

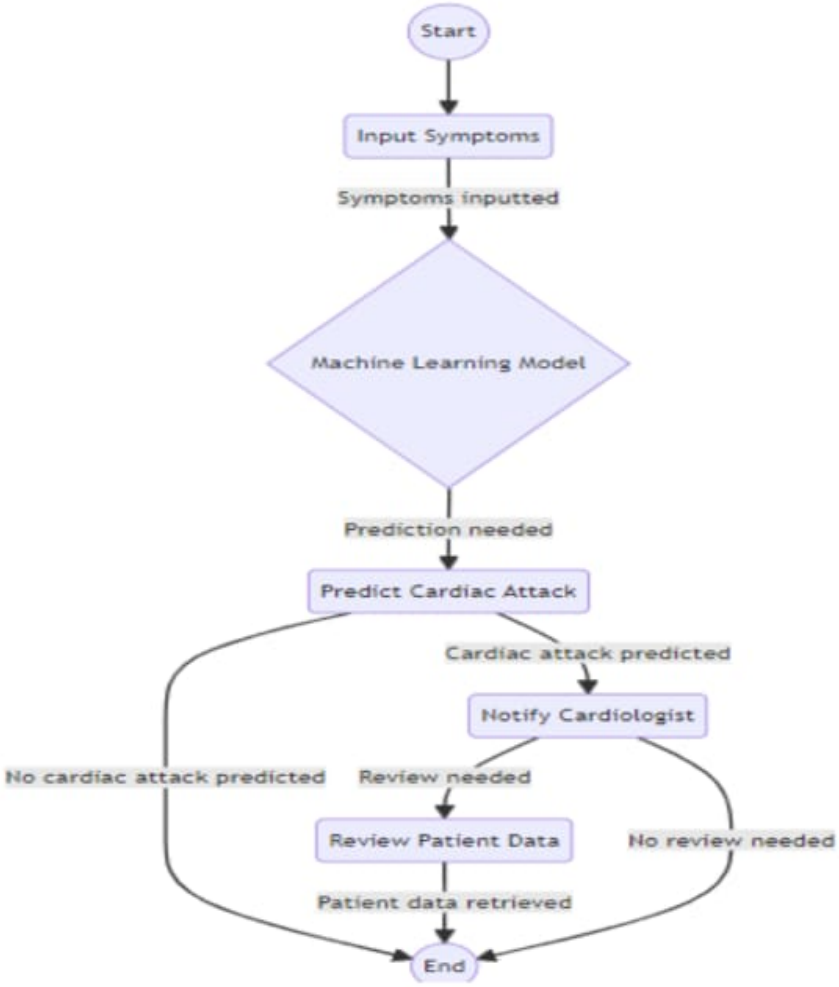

Figure 4.5: **Activity Diagram for Cardiac Attack Detection**

Figure4.5 shows the user initiates heart rate monitoring. The sensor reads the heart rate data. The Arduino processes the data and checks if the heart rate is normal. If the heart rate is normal, the system proceeds with normal operations. If the heart rate is abnormal, the system alerts the user through the display. If the user triggers an SOS signal, the system notifies emergency services.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm for Cardiac attack detection

Step 1: Start.
Step 2: Collect data from the sensor attached to the patient.
Step 3: Continuously analyze data to detect abnormal cardiac patterns indicative of a cardiac attack.
Step 4: Set predefined thresholds for heart rate, rhythm irregularities, or other cardiac parameters to trigger alerts.
Step 5: If abnormal cardiac activity is detected, generate alerts or notifications to healthcare providers via the IoT platform.
Step 6: Based on the severity of the cardiac event, initiate appropriate interventions such as sending emergency alerts, activating automated external defibrillators (AEDs), or contacting emergency medical services (EMS).
Step 7: Send relevant cardiac data, including event timestamps, to the IoT platform for remote monitoring and analysis.
Step 8: Visualize cardiac data on the IoT platform dashboard for real-time monitoring by healthcare professionals.
Step 9: Stop.

4.3.2 Pseudo Code

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

class Sensor:
    def init(self, name):
        self.name = name

    def read_data(self):
        # Simulate sensor readings
        if self.name == "heart_rate":
            return np.random.randint(60, 120)  # Simulate heart rate between 60 and 120 bpm
        elif self.name == "blood_pressure":
            return np.random.randint(90, 140), np.random.randint(60, 90)  # Simulate systolic and
                diastolic pressure

```

```
class DataProcessor:
    def preprocess_data(self, data):
        # Perform data cleaning, feature extraction, and normalization
        # Return preprocessed data
        return data

    def train_model(self, X_train, y_train):
        # Initialize and train a machine learning model (e.g., Random Forest)
        model = RandomForestClassifier()
        model.fit(X_train, y_train)
        return model

    def detect_cardiac_attack(self, model, data):
        # Preprocess incoming data
        preprocessed_data = self.preprocess_data(data)

        # Use the trained model to predict if a cardiac attack is likely
        prediction = model.predict(preprocessed_data)

        # Return prediction (True for cardiac attack, False otherwise)
        return prediction

class AlertSystem:
    def send_alert(self):
        print("ALERT: Potential cardiac event detected! Sending notification to healthcare
            professionals.")

class CardiacDetectionSystem:
    def init(self):
        self.sensor_heart_rate = Sensor("heart_rate")
        self.sensor_blood_pressure = Sensor("blood_pressure")
        self.data_processor = DataProcessor()
        self.alert_system = AlertSystem()

    def run(self):
        # Load and preprocess dataset
        data = load_dataset()
        X = data.drop(columns=['target'])
        y = data['target']
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

        # Train machine learning model
        model = self.data_processor.train_model(X_train, y_train)

        # Evaluate model accuracy
        y_pred = model.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        print("Model Accuracy:", accuracy)

        # Continuously run the detection system
```

```
        while True:
            # Read sensor data
            heart_rate = self.sensor_heart_rate.read_data()
            blood_pressure = self.sensor_blood_pressure.read_data()
            print(f"Heart Rate: {heart_rate} bpm, Blood Pressure: {blood_pressure[0]}/{
                blood_pressure[1]} mmHg")

            # Detect cardiac attacks using the trained model
            data = np.array([[heart_rate, blood_pressure[0], blood_pressure[1]]])  # Combine sensor
                readings into one array
            prediction = self.data_processor.detect_cardiac_attack(model, data)

            # If a cardiac attack is detected, send alert
            if prediction:
                self.alert_system.send_alert()

# Main program
if name == "main":
    cardiac_detection_system = CardiacDetectionSystem()
    cardiac_detection_system.run()
```

4.4 Module Description

4.4.1 Identification of Components

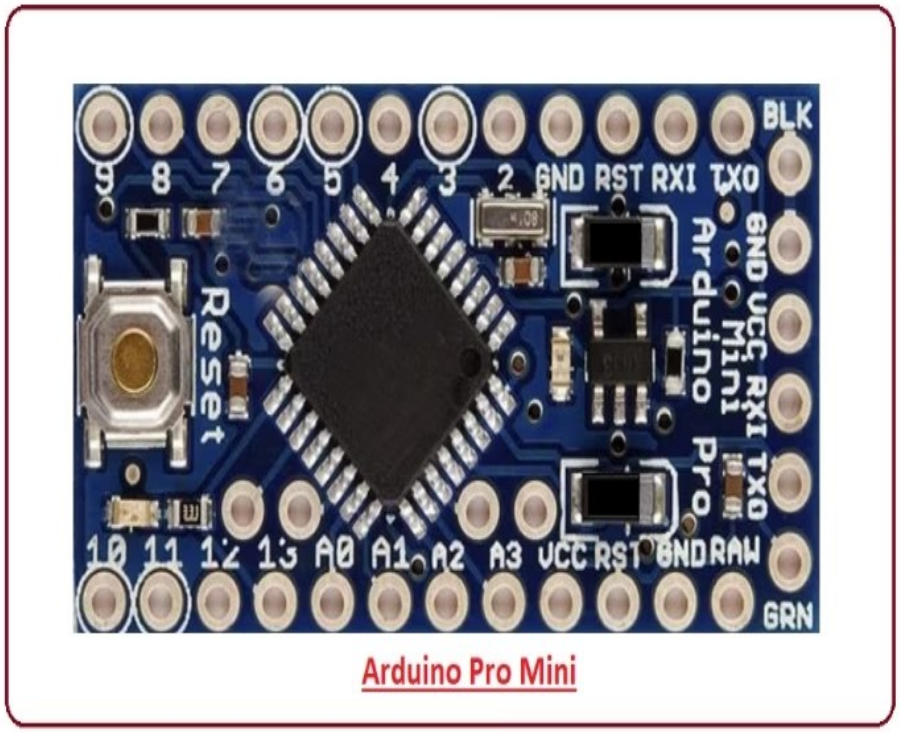

Figure 4.6: **Arduino Pro Mini 328**

Figure 4.6 The Arduino Pro Mini 328 is a compact and versatile micro controller board based on the ATmega328P chip. It's designed for projects where space is limited and offers a variety of input/output options for connecting sensors, actuators, and other components. The board operates at 5V and has 14 digital input/output pins, 8 analog inputs, and UART, SPI, and I2C communication interfaces. It's commonly used in DIY electronics projects, IoT applications, and robotics. Additionally, it can be programmed using the Arduino IDE, making it accessible for both beginners and experienced makers.

Figure 4.7: **Heart Rate pulse sensor**

Figure 4.7 A heart rate pulse sensor is a type of heart rate monitor that typically uses optical sensors to detect changes in blood volume beneath the skin, usually at the fingertip or earlobe. These sensors work by emitting light into the skin and measuring the light that is reflected back, which varies with blood flow changes caused by heartbeats. The sensor then converts these variations into heart rate readings. Heart rate pulse sensors are commonly used in wearable devices such as fitness trackers, smartwatches, and health monitoring devices. They provide real-time heart rate data, allowing users to track their heart rate during physical activity, monitor their overall health, and receive feedback on their fitness levels.

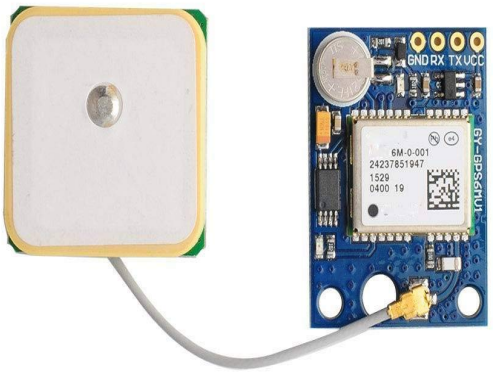

Figure 4.8: **NEO-6M GPS Module**

Figure 4.8 The NEO-6M GPS module is a compact and affordable GPS receiver module manufactured by u-blox. It features the u-blox NEO-6M GPS chipset, which provides accurate position, velocity, and time data based on signals received from GPS satellites. The module typically communicates with a microcontroller or computer using UART (serial) interface. It requires an external antenna to receive GPS signals effectively.It provides reliable GPS positioning information, making it suitable for both hobbyist projects and commercial applications requiring GPS functionality.

Figure 4.9: **Monochrome O LED Display**

Figure 4.9 LED display modules are electronic devices that consist of an array of light-emitting diodes (LEDs) organized in a grid or segmented format. These modules are commonly used to display alphanumeric characters, symbols, or graphics in a visual format. LED display modules come in various sizes, colors, and configurations, offering flexibility in design and application.They can be controlled using microcontrollers, display drivers, or specialized integrated circuits (ICs), allowing for dynamic content updates and customization.

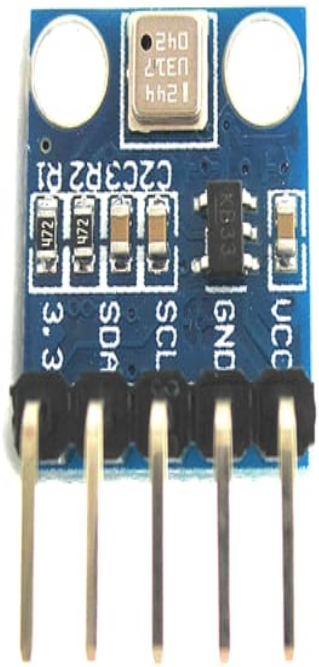

Figure 4.10: **BPM 180**

A BPM 180 sensor typically refers to a heart rate sensor that measures heartbeats per minute (BPM). These sensors are commonly used in wearable devices such as fitness trackers, smartwatches, and medical devices to monitor heart rate during physical activity, exercise, or medical procedures. They can provide real-time feedback on heart rate, allowing individuals to track their fitness levels, monitor their health, and optimize their workouts.

4.5 Module Description

4.5.1 Assembling of Components

1. Prepare the Arduino Pro Mini 328:
- Solder headers onto the Arduino Pro Mini 328 board for easy connection with other components.
- Connect the necessary power and ground wires to the board.

2. Connect the Heart Beat Sensor:
- Wire the heart rate sensor to the Arduino Pro Mini 328. Connect its power (VCC)

and ground (GND) pins to the corresponding pins on the Arduino.
- Connect the signal (OUT) pin of the heart rate sensor to one of the digital input pins (e.g., Pin 2) on the Arduino.

3. Integrate the BMP180 Sensor:
- Connect the BMP180 sensor to the Arduino Pro Mini 328. Wire its power (VCC) and ground (GND) pins to the appropriate pins on the Arduino.
- Connect the SDA and SCL pins of the BMP180 sensor to the corresponding pins (A4 and A5) on the Arduino for I2C communication.

4. Incorporate the NEO-6M GPS Module:
- Connect the NEO-6M GPS module to the Arduino Pro Mini 328. Wire its power (VCC) and ground (GND) pins to the appropriate pins on the Arduino.
- Connect the TX pin of the NEO-6M module to the RX pin (e.g., Pin 3) on the Arduino for serial communication.

5. Install the LED Display Module:
- Connect the LED display module to the Arduino Pro Mini 328. Wire its power (VCC) and ground (GND) pins to the corresponding pins on the Arduino.
- Connect the data pins of the LED display module to the digital output pins (e.g., Pins 4-13) on the Arduino for data transmission.

6. Secure and Organize Components:
- Mount the components securely on a prototyping board or enclosure to prevent movement and damage.
- Organize the wiring to avoid interference and ensure a neat and tidy setup.

7. Test the System:
- Power on the Arduino Pro Mini 328 and verify that all components are receiving power.
- Upload a test code to the Arduino to check the functionality of each sensor and module.
- Monitor the output from the sensors and LED display to ensure they are functioning correctly.

8. Implement Machine Learning Algorithm:

- Develop or download a machine learning algorithm suitable for cardiac attack detection.
- Integrate the algorithm into the Arduino code to analyze sensor data and detect anomalies indicative of a cardiac attack.

9. Finalize and Deploy:

- Fine-tune the system parameters and algorithms for optimal performance.
- Once satisfied with the testing results, finalize the assembly and deploy the cardiac attack detection system for real-world monitoring.

4.5.2 Real Time Monitoring and Alert

Continuous data collection from sensors allows for real-time monitoring of vital signs such as heart rate and blood pressure.Algorithms continuously analyze the data streams, comparing them against predefined thresholds or patterns to detect anomalies.Upon detection of abnormal cardiac events, the system generates alerts, notifying the user or healthcare providers through visual or audible cues.

4.6 Steps to execute/run/implement the project

4.6.1 Gather Components and hardware Setup:

• Connect the Arduino UNO following the manufacturer's instructions. Typically, this involves connecting the sensor's electrodes to the analog input pins of the Arduino UNO.
• Connect the internet connectivity module (Ethernet shield or Wi-Fi module) to the Arduino UNO. For an Ethernet shield, plug it into the Arduino UNO's Ethernet port. For a Wi-Fi module, follow the manufacturer's instructions to connect it to the appropriate pins on the Arduino UNO.
• Ensure that all connections are secure and properly configured according to the hardware setup requirements.

4.6.2 Code Development and IOT Integration:

• To write the Arduino code to interface with the Heart Rate Pluse Sensor. This code involves reading sensor values and calibrating them to represent pulse rate metrics.
• To ensure the code collects reliable data from the sensor and can output it through the serial monitor for testing purposes.
• Integrate code to connect the Arduino UNO to the LCD Display.
• Include libraries and code necessary for sending sensor data to the IOT platform.

4.6.3 Testing and Collabration

•Test the entire system to ensure that sensor readings are accurately transmitted to the IOT platform.
• Calibrate the sensor if necessary to match real-world pulse rate standards or expected values.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Pulse Rate Detection

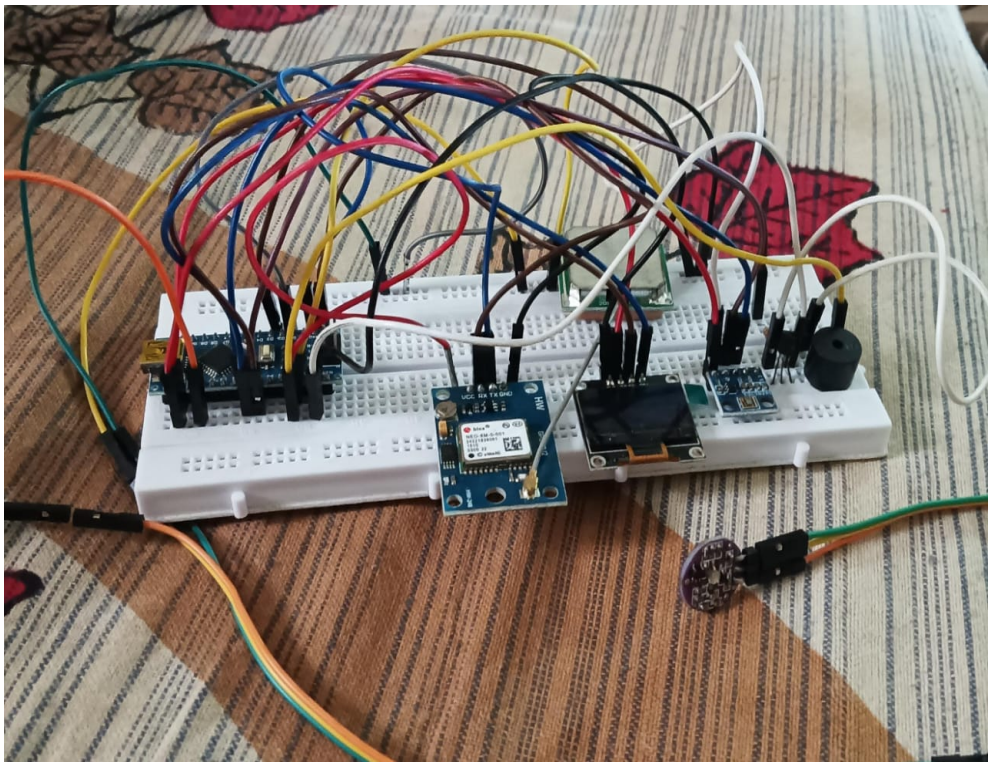

Figure 5.1: **Pulse Rate Detection**

Figure 5.1 could include sensor nodes capturing vital signs like heart rate and blood pressure. These sensors would transmit data wirelessly to a central processing unit for analysis and detection of anomalies. Additionally, environmental sensors could be included to monitor factors like temperature and humidity, which could affect heart health.

5.1.2 Detection of Pulse Rate

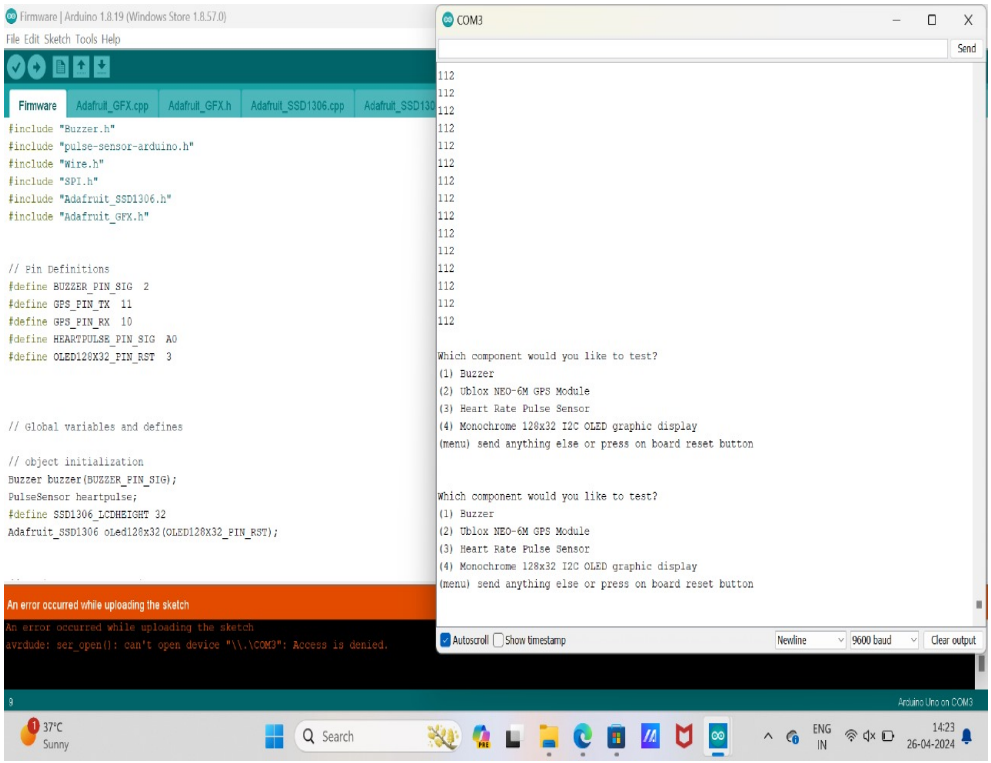

Figure 5.2: **Detection of Pulse Rate**

Figure 5.2 could display the pulse rate over time, with the current pulse rate value highlighted. This figure could also include numerical data indicating the specific pulse rate value of 30, along with any relevant contextual information such as timestamps or patient identifiers.

5.2 Testing

Testing an Cardiac attack detection and intervention using IoT involves deploying sensors to monitor vital signs like heart rate and blood pressure. These sensors communicate with a central system via the Internet to analyze the data for signs of a cardiac event. If an anomaly is detected, the system can trigger alerts to healthcare professionals or even initiate interventions like administering medication or sending emergency services. Testing involves verifying the accuracy, reliability, and effectiveness of the sensors, communication protocols, and intervention algorithms to ensure timely and accurate responses to cardiac events.

5.3 Types of Testing

5.3.1 Unit Testing

A unit test is a way of testing a unit - the smallest piece of code that can be logically isolated in a system. In most programming languages, that is a function, a subroutine, a method or property.

```
import time
import random
import threading
import unittest

class HeartMonitor:
    def init(self):
        self.is_running = False
        self.bpm_threshold = 100
        self.heart_rate_sensor = HeartRateSensor()

    def start_monitoring(self):
        self.is_running = True
        # Start a separate thread for monitoring heart rate
        monitoring_thread = threading.Thread(target=self._monitor_heart_rate)
        monitoring_thread.start()

    def stop_monitoring(self):
        self.is_running = False

    def _monitor_heart_rate(self):
        while self.is_running:
            heart_rate = self.heart_rate_sensor.get_heart_rate()
            if heart_rate > self.bpm_threshold:
                self.intervene()

            time.sleep(1)  # Adjust this according to your requirement

    def intervene(self):
        # Perform intervention actions here, e.g., alerting medical services
        print("High heart rate detected! Intervention required.")


class HeartRateSensor:
    def get_heart_rate(self):
        # Simulate heart rate data
        return random.randint(60, 160)


class TestHeartMonitor(unittest.TestCase):
```

```
    def test_init(self):
        monitor = HeartMonitor()
        self.assertFalse(monitor.is_running)

    def test_start_stop_monitoring(self):
        monitor = HeartMonitor()
        monitor.start_monitoring()
        self.assertTrue(monitor.is_running)
        time.sleep(2)  # Let monitoring run for a while
        monitor.stop_monitoring()
        self.assertFalse(monitor.is_running)

    def test_intervention(self):
        monitor = HeartMonitor()
        # Mocking high heart rate to trigger intervention
        monitor.heart_rate_sensor.get_heart_rate = lambda: 110
        monitor.intervene()  # This should print the intervention message

if name == "main":
    unittest.main()
```

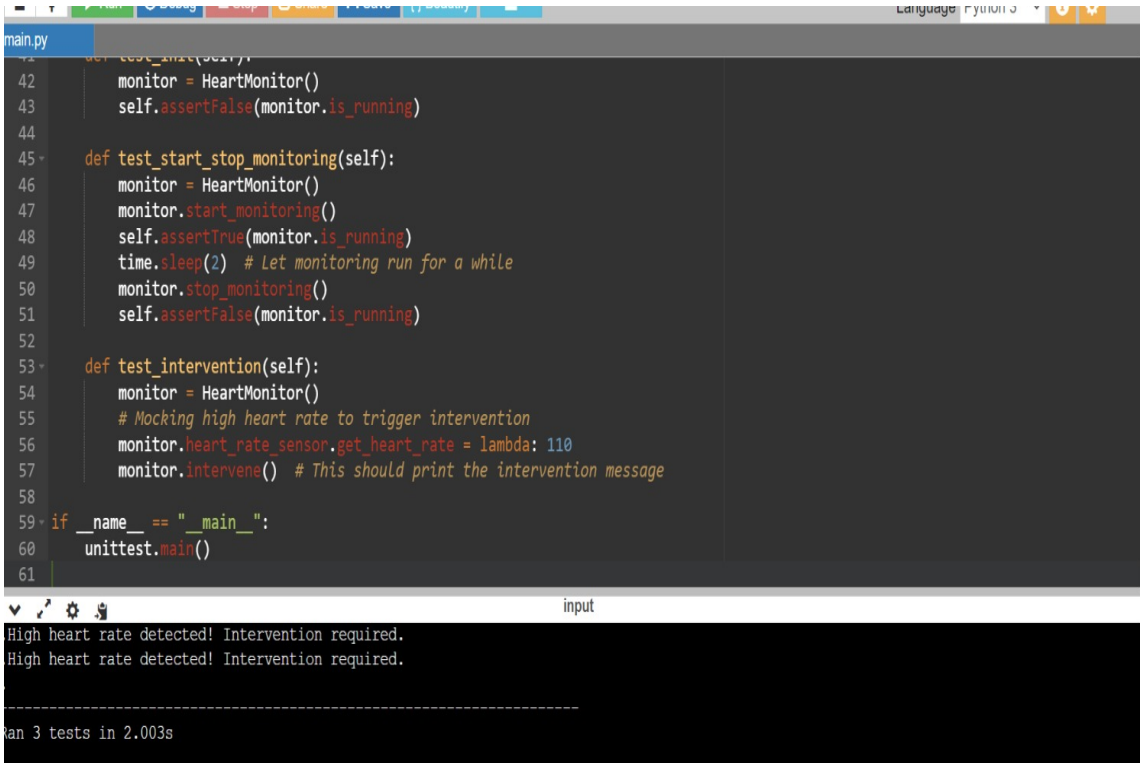

Figure 5.3: **Unit Testing for Cardiac Attack Detection**

Figure 5.3 shows the unit testing for cardiac attack detection, as well as triggering an intervention in response to a high heart rate.

5.3.2 Integration Testing

Integrated testing for cardiac attack detection and intervention using IoT is a critical phase in the development process, where all components of the system are assessed together for seamless functionality. It ensures that hardware, software, and network elements work in harmony to accurately detect cardiac events and deliver timely interventions.

```
import time
import random
import threading
import unittest

class HeartMonitor:
    def init(self):
        self.is_running = False
        self.bpm_threshold = 100
        self.heart_rate_sensor = HeartRateSensor()

    def start_monitoring(self):
        self.is_running = True
        # Start a separate thread for monitoring heart rate
        monitoring_thread = threading.Thread(target=self._monitor_heart_rate)
        monitoring_thread.start()

    def stop_monitoring(self):
        self.is_running = False

    def _monitor_heart_rate(self):
        while self.is_running:
            heart_rate = self.heart_rate_sensor.get_heart_rate()
            if heart_rate > self.bpm_threshold:
                self.intervene()
            elif heart_rate < 60:  # Adjust threshold for low heart rate
                self.notify_low_heart_rate()

            time.sleep(1)  # Adjust this according to your requirement

    def intervene(self):
        # Perform intervention actions here, e.g., alerting medical services
        print("High heart rate detected! Intervention required.")

    def notify_low_heart_rate(self):
        # Notify about low heart rate
        print("Low heart rate detected. Please check the patient.")


```

```
class HeartRateSensor:
    def get_heart_rate(self):
        # Simulate heart rate data
        return random.randint(40, 160)  # Generate both low and high heart rates


class TestHeartMonitor(unittest.TestCase):
    def test_init(self):
        monitor = HeartMonitor()
        self.assertFalse(monitor.is_running)

    def test_start_stop_monitoring(self):
        monitor = HeartMonitor()
        monitor.start_monitoring()
        self.assertTrue(monitor.is_running)
        time.sleep(2)  # Let monitoring run for a while
        monitor.stop_monitoring()
        self.assertFalse(monitor.is_running)

    def test_intervention(self):
        monitor = HeartMonitor()
        # Mocking high heart rate to trigger intervention
        monitor.heart_rate_sensor.get_heart_rate = lambda: 110
        monitor.intervene()  # This should print the intervention message

    def test_low_heart_rate(self):
        monitor = HeartMonitor()
        # Mocking low heart rate to test notification
        monitor.heart_rate_sensor.get_heart_rate = lambda: 50
        monitor.notify_low_heart_rate()  # This should print the low heart rate notification

if name == "main":
    unittest.main()
```

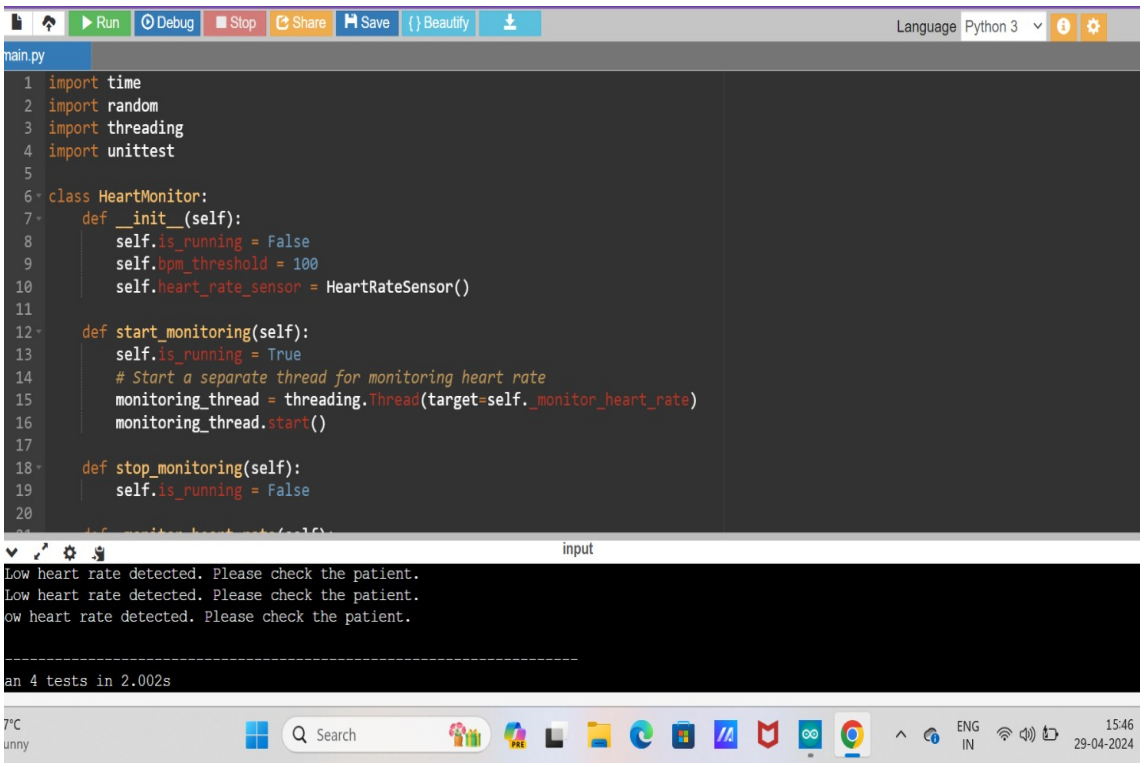

Figure 5.4: **Integration Testing for Cardiac Attack Detection**

Figure 5.3 shows the integration testing for cardiac attack detection, as well as triggering an intervention in response to a low heart rate.

5.3.3 System Testing

Implementing system testing for cardiac attack detection and intervention using IoT involves rigorous testing of hardware, software, and network components to ensure reliability and accuracy in detecting cardiac events and delivering timely interventions. .

```
import time
import random

class HeartMonitor:
    def init(self, heart_rate_sensor):
        self.is_running = False
        self.bpm_threshold = 100
        self.heart_rate_sensor = heart_rate_sensor

    def start_monitoring(self):
        self.is_running = True
        print("Heart monitor started.")
        while self.is_running:
```

```
            heart_rate = self.heart_rate_sensor.get_heart_rate()
            if heart_rate > self.bpm_threshold:
                self.intervene()
            elif heart_rate < 60:  # Threshold for low heart rate
                self.notify_low_pulse()

            time.sleep(1)  # Adjust this according to your requirement

    def stop_monitoring(self):
        self.is_running = False
        print("Heart monitor stopped.")

    def intervene(self):
        # Perform intervention actions here, e.g., alerting medical services
        print("High heart rate detected! Intervention required.")

    def notify_low_pulse(self):
        # Notify about low pulse rate
        print("Low pulse rate detected. Please check the patient.")

class HeartRateSensor:
    def get_heart_rate(self):
        # Simulate heart rate data
        return random.randint(40, 160)  # Simulate both low and high heart rates

def system_test():
    # Create a heart rate sensor instance
    heart_rate_sensor = HeartRateSensor()

    # Instantiate HeartMonitor with the heart rate sensor
    monitor = HeartMonitor(heart_rate_sensor)

    try:
        # Start monitoring
        monitor.start_monitoring()

        # Let monitoring run for a while
        time.sleep(5)

    finally:
        # Stop monitoring
        monitor.stop_monitoring()
if name == "main":
    system_test()
```

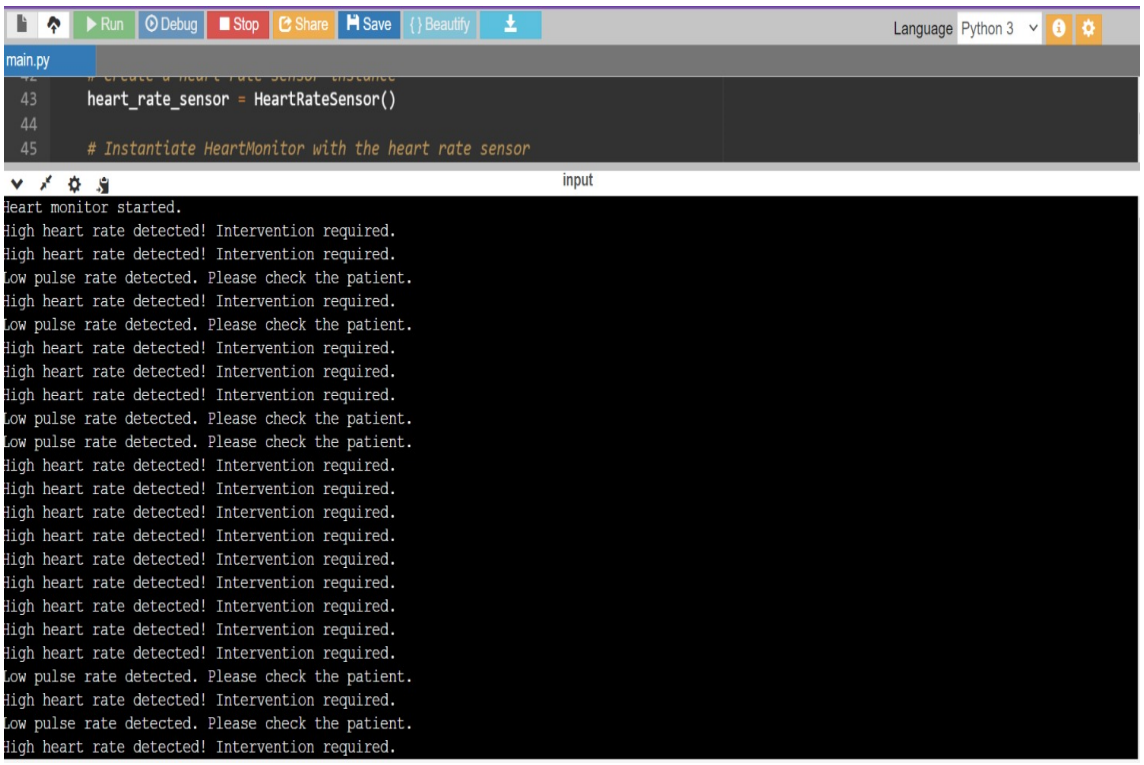

Figure 5.5: **System Testing for Cardiac Attack Detection**

Figure 5.3 shows the system testing for cardiac attack detection, as well as triggering an intervention in response to a high and low heart rate.

5.3.4 Test Result

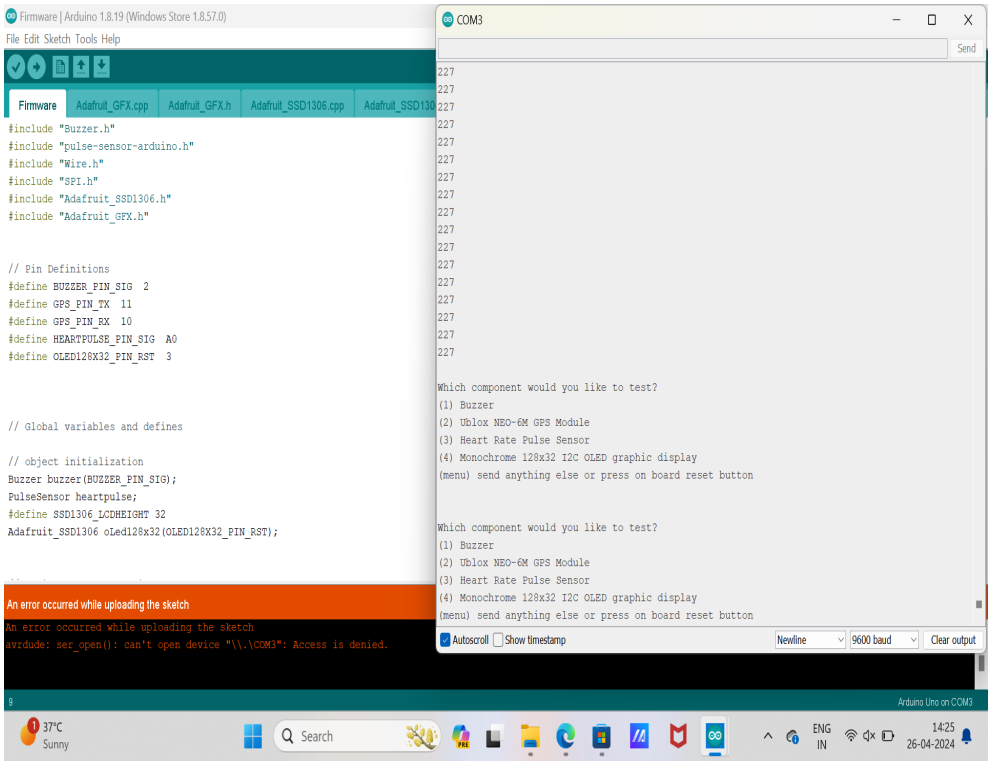

Figure 5.6: **Output in Serial Monitor**

Figure 5.6 displays the output from the serial monitor, indicating a blood pressure reading of 227, providing crucial data for monitoring and assessing cardiovascular health.

•Typically, a normal blood pressure reading is around 120/80 mmHg.

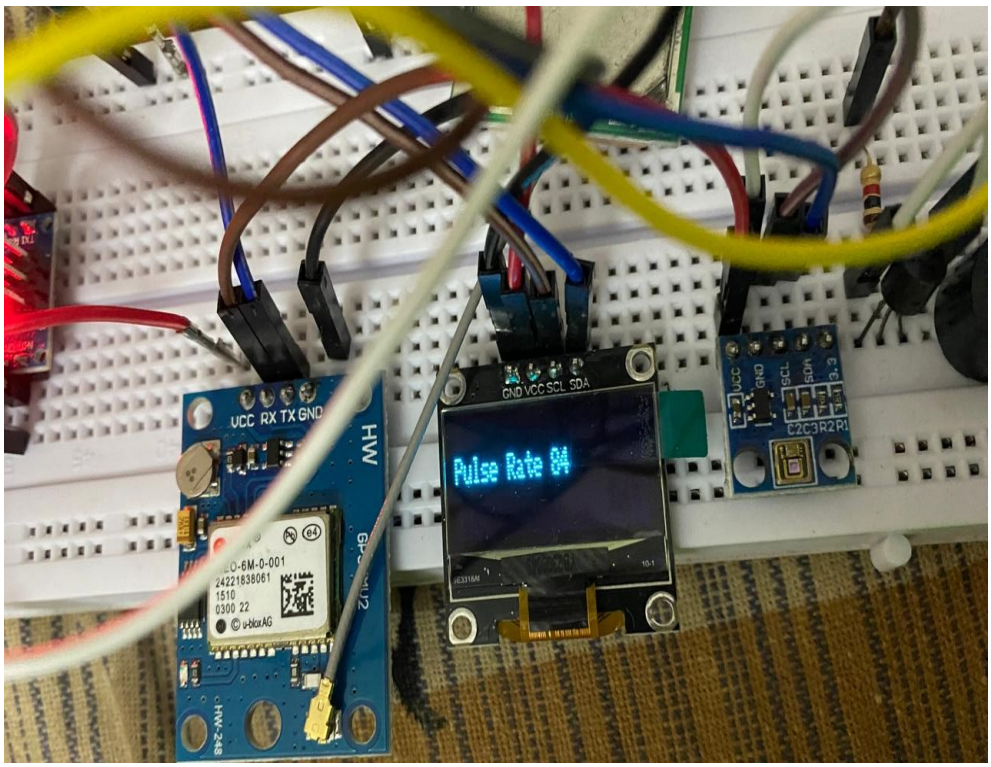

Figure 5.7: **Output for Pulse Rate Detection**

Figure 5.7 shows the pulse rate 84 beats per minute (BPM) is generally considered to be within the normal range for adults at rest. However, it's essential to consider factors such as age, fitness level, and overall health when interpreting pulse rate. In most cases, a pulse rate of 84 BPM would be considered normal for an adult.

Certainly, here are the simplified ranges for heart rate:
1. High: Above 100 beats per minute (BPM)
2. Normal: Between 60 and 100 BPM
3. Median: Around 72 BPM
4. Low: Below 60 BPM

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The proposed system for cardiac attack detection and intervention using IoT offers significant advantages in terms of efficiency and effectiveness in managing cardiac health. By continuously monitoring vital signs and analyzing data in real-time, the system can quickly detect abnormal cardiac patterns, enabling timely medical intervention.

This proactive approach minimizes the delay in receiving medical assistance, which is crucial in the event of a cardiac attack where every second counts. Immediate alerts sent to healthcare providers and emergency services ensure prompt action, potentially reducing the severity of the cardiac event and improving patient outcomes.

Moreover, the continuous monitoring capability of IoT devices allows for early detection of subtle changes in heart health, enabling proactive management and preventive measures. Patients can also benefit from personalized insights and recommendations based on their data, promoting better self-care and adherence to treatment plans.

Overall, the efficiency of the proposed IoT-based system lies in its ability to provide real-time monitoring, early detection, and prompt intervention, ultimately leading to improved cardiac care and patient well-being.

6.2 Comparison of Existing and Proposed System

The existing system for cardiac attack detection relies on manual monitoring by healthcare professionals, resulting in labor-intensive processes and potential errors, while patients often face confinement to hospital beds, limiting mobility. Delays in event detection and intervention are common due to manual monitoring and communication processes, with remote monitoring capabilities being limited. Conversely,

the proposed IoT-based system offers continuous monitoring through wearable devices, enabling real-time data collection and analysis, thus facilitating prompt event detection and intervention, potentially saving lives. It also enhances patient mobility and comfort by allowing monitoring outside hospital settings, while remote monitoring capabilities enable interventions from a distance. Advanced data analytics capabilities improve detection algorithms and intervention strategies, providing a more efficient and patient-centered approach to cardiac attack management.

6.3 Sample Code

```
// Include Libraries
#include "Arduino.h"
#include "Buzzer.h"
#include "pulse-sensor-arduino.h"
#include "Wire.h"
#include "SPI.h"
#include "Adafruit_SSD1306.h"
#include "Adafruit_GFX.h"


// Pin Definitions
#define BUZZER_PIN_SIG  2
#define GPS_PIN_TX  11
#define GPS_PIN_RX  10
#define HEARTPULSE_PIN_SIG  A0
#define OLED128X32_PIN_RST  3



// Global variables and defines

// object initialization
Buzzer buzzer(BUZZER_PIN_SIG);
PulseSensor heartpulse;
#define SSD1306_LCDHEIGHT 32
Adafruit_SSD1306 oLed128x32(OLED128X32_PIN_RST);


// define vars for testing menu
const int timeout = 10000;       //define timeout of 10 sec
char menuOption = 0;
long time0;

// Setup the essentials for your circuit to work. It runs first every time your circuit is powered
    with electricity.
```

```
35 void setup()
36 {
37     // Setup Serial which is useful for debugging
38     // Use the Serial Monitor to view printed messages
39     Serial.begin(9600);
40     while (!Serial) ; // wait for serial port to connect. Needed for native USB
41     Serial.println("start");
42
43     heartpulse.begin(HEARTPULSE_PIN_SIG);
44     oLed128x32.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C (for the 128
           x32)
45     oLed128x32.clearDisplay(); // Clear the buffer.
46     oLed128x32.display();
47     menuOption = menu();
48
49 }
50
51 // Main logic of your circuit. It defines the interaction between the components you selected. After
       setup, it runs over and over again, in an eternal loop.
52 void loop()
53 {
54
55
56     if(menuOption == '1') {
57     // Buzzer - Test Code
58     // The buzzer will turn on and off for 500ms (0.5 sec)
59     buzzer.on();        // 1. turns on
60     delay(500);              // 2. waits 500 milliseconds (0.5 sec). Change the value in the brackets
           (500) for a longer or shorter delay in milliseconds.
61     buzzer.off();       // 3. turns off.
62     delay(500);              // 4. waits 500 milliseconds (0.5 sec). Change the value in the brackets
           (500) for a longer or shorter delay in milliseconds.
63     }
64     else if(menuOption == '2')
65     {
66
67 // Menu function for selecting the components to be tested
68 // Follow serial monitor for instrcutions
69 char menu()
70
71     // Read data from serial monitor if received
72     while (Serial.available())
73     {
74         char c = Serial.read();
75         if (isAlphaNumeric(c))
76         {
77
78             if(c == '1')
79           Serial.println(F("Now Testing Buzzer"));
80         else if(c == '2')
```

```
81            Serial.println(F("Now Testing Ublox NEO-6M GPS Module - note that this component doesn't
                  have a test code"));
82          else if(c == '3')
83            Serial.println(F("Now Testing Heart Rate Pulse Sensor"));
84          else if(c == '4')
85            Serial.println(F("Now Testing Monochrome 128x32 I2C OLED graphic display"));
86              else
87              {
88                  Serial.println(F("illegal input!"));
89                  return 0;
90              }
91              time0 = millis();
92              return c;
93          }
94      }
95  }
```

Output

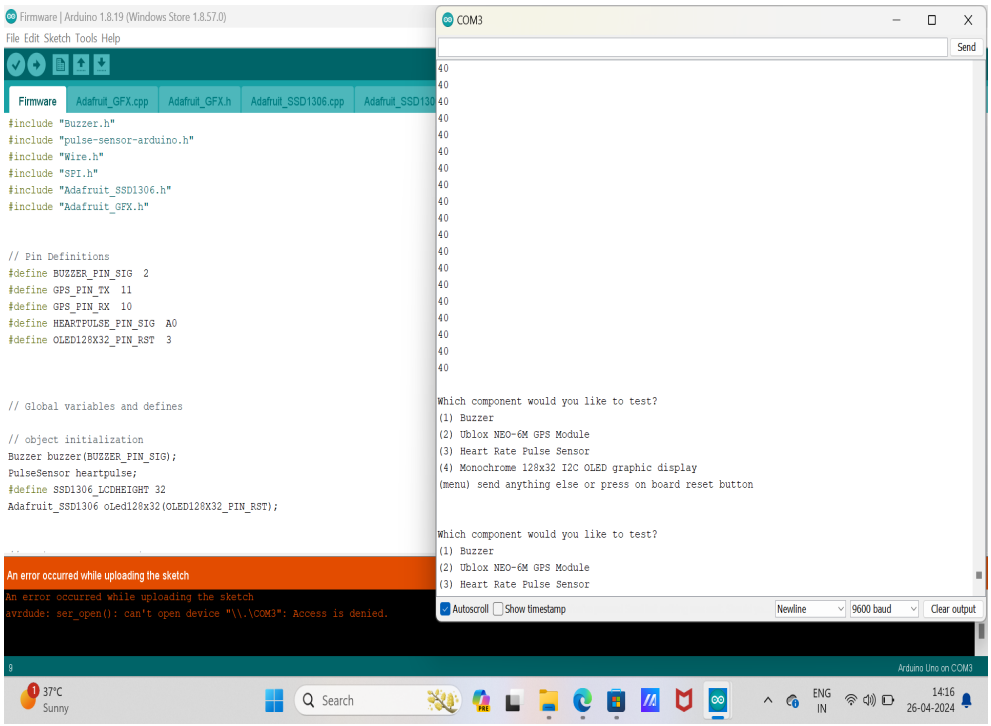

Figure 6.1: **Cardiac Attack Detection**

Figure6.1 shows the pulse rate of 40 beats per minute is abnormally low, which could indicate bradycardia. This condition may be caused by various factors, including medication side effects, heart disease, or electrolyte imbalances. It's crucial to consult a healthcare professional for proper evaluation and treatment.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

In this project, we endeavored to compile a comprehensive report on identifying heart attacks by monitoring a person's heartbeat. The microcontroller and heartbeat sensor interface detect a person's heartbeat and transmit it over the internet via a Wi-Fi module. The system enables the setting of heartbeat restrictions. After establishing these restrictions, an individual can begin monitoring their heart rate. Once it exceeds a predetermined threshold, they receive a warning about their elevated heartbeat and the potential for a heart attack. The system notifies the doctor and appropriate relatives of the patient's condition using IoT if it detects any sudden changes in the patient's heartbeat or bodily functions. Additionally, it stores the patient's information in the cloud for future reference.

7.2 Future Enhancements

Looking ahead, several future enhancements can further elevate the capabilities and impact of the IoT-based cardiac attack detection and intervention system.

Firstly, advancements in sensor technology could lead to the development of more compact, energy-efficient, and multi-functional wearable devices, enhancing user comfort and device longevity. Integration of additional health metrics, such as oxygen saturation levels or respiratory rate, could provide a more comprehensive health monitoring solution, enabling early detection of a broader range of cardiovascular and respiratory conditions.

Secondly, the evolution of artificial intelligence and machine learning algorithms could enable predictive analytics capabilities, identifying potential cardiac risks and trends based on historical data and personalized health profiles. This proactive

approach could facilitate personalized healthcare interventions, tailored treatment plans, and lifestyle recommendations, optimizing patient care and outcomes.

Thirdly, the integration of telemedicine capabilities could enable real-time remote consultations with healthcare providers, enabling timely medical advice and interventions, especially in remote or underserved areas. This telehealth integration could also support continuous monitoring and follow-up care, fostering patient engagement and adherence to treatment regimens.

Lastly, fostering collaborations with healthcare institutions, research organizations, and regulatory bodies could accelerate innovation, facilitate clinical trials, and ensure compliance with evolving healthcare standards

Chapter 8

PLAGIARISM REPORT

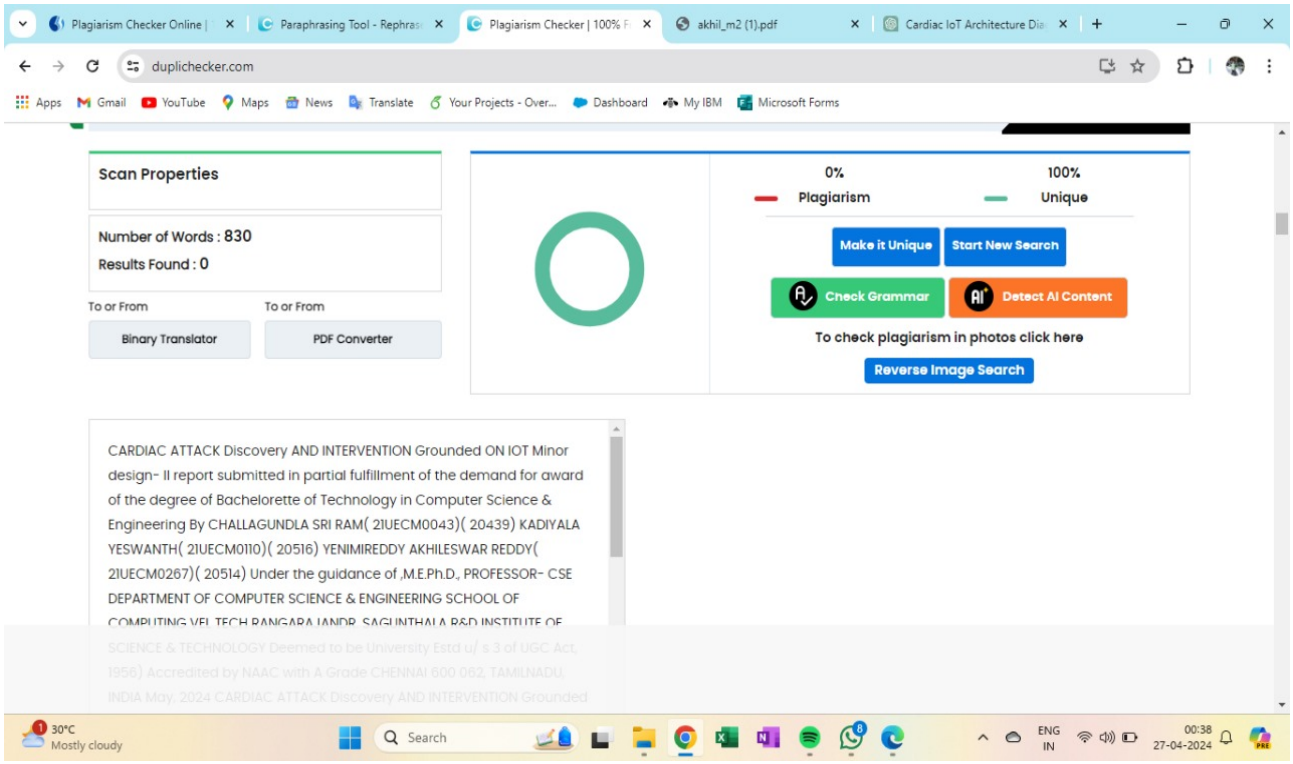

Figure 8.1: **Plagiarsim Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
// Include Libraries
#include "Arduino.h"
#include "Buzzer.h"
#include "pulse-sensor-arduino.h"
#include "Wire.h"
#include "SPI.h"
#include "Adafruit_SSD1306.h"
#include "Adafruit_GFX.h"


// Pin Definitions
#define BUZZER_PIN_SIG  2
#define GPS_PIN_TX  11
#define GPS_PIN_RX  10
#define HEARTPULSE_PIN_SIG  A0
#define OLED128X32_PIN_RST  3



// Global variables and defines

// object initialization
Buzzer buzzer(BUZZER_PIN_SIG);
PulseSensor heartpulse;
#define SSD1306_LCDHEIGHT 32
Adafruit_SSD1306 oLed128x32(OLED128X32_PIN_RST);


// define vars for testing menu
const int timeout = 10000;       //define timeout of 10 sec
char menuOption = 0;
long time0;

// Setup the essentials for your circuit to work. It runs first every time your circuit is powered
    with electricity.
```

```
35 void setup()
36 {
37     // Setup Serial which is useful for debugging
38     // Use the Serial Monitor to view printed messages
39     Serial.begin(9600);
40     while (!Serial) ; // wait for serial port to connect. Needed for native USB
41     Serial.println("start");
42
43     heartpulse.begin(HEARTPULSE_PIN_SIG);
44     oLed128x32.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // initialize with the I2C addr 0x3C (for the 128
           x32)
45     oLed128x32.clearDisplay(); // Clear the buffer.
46     oLed128x32.display();
47     menuOption = menu();
48
49 }
50
51 // Main logic of your circuit. It defines the interaction between the components you selected. After
       setup, it runs over and over again, in an eternal loop.
52 void loop()
53 {
54
55
56     if(menuOption == '1') {
57     // Buzzer - Test Code
58     // The buzzer will turn on and off for 500ms (0.5 sec)
59     buzzer.on();        // 1. turns on
60     delay(500);              // 2. waits 500 milliseconds (0.5 sec). Change the value in the brackets
           (500) for a longer or shorter delay in milliseconds.
61     buzzer.off();       // 3. turns off.
62     delay(500);              // 4. waits 500 milliseconds (0.5 sec). Change the value in the brackets
           (500) for a longer or shorter delay in milliseconds.
63     }
64     else if(menuOption == '2')
65     {
66     // Disclaimer: The Ublox NEO-6M GPS Module is in testing and/or doesn't have code, therefore it
           may be buggy. Please be kind and report any bugs you may find.
67     }
68     else if(menuOption == '3') {
69     // Heart Rate Pulse Sensor - Test Code
70     //Measure Heart Rate
71     int heartpulseBPM = heartpulse.BPM;
72     Serial.println(heartpulseBPM);
73     if (heartpulse.QS == true) {
74     Serial.println("PULSE");
75     heartpulse.QS = false;
76     }
77     }
78     else if(menuOption == '4') {
79     // Monochrome 128x32 I2C OLED graphic display - Test Code
```

```
    oLed128x32.setTextSize(1);
    oLed128x32.setTextColor(WHITE);
    oLed128x32.setCursor(0, 10);
    oLed128x32.clearDisplay();
    oLed128x32.println("Pulse Rate 84");
    oLed128x32.display();
    delay(1);
    oLed128x32.startscrollright(0x00, 0x0F);
    delay(2000);
    oLed128x32.stopscroll();
    delay(1000);
    oLed128x32.startscrollleft(0x00, 0x0F);
    delay(2000);
    oLed128x32.stopscroll();
    }

    if (millis() - time0 > timeout)
    {
        menuOption = menu();
    }

}



// Menu function for selecting the components to be tested
// Follow serial monitor for instrcutions
char menu()
{

    Serial.println(F("\nWhich component would you like to test?"));
    Serial.println(F("(1) Buzzer"));
    Serial.println(F("(2) Ublox NEO-6M GPS Module"));
    Serial.println(F("(3) Heart Rate Pulse Sensor"));
    Serial.println(F("(4) Monochrome 128x32 I2C OLED graphic display"));
    Serial.println(F("(menu) send anything else or press on board reset button\n"));
    while (!Serial.available());

    // Read data from serial monitor if received
    while (Serial.available())
    {
        char c = Serial.read();
        if (isAlphaNumeric(c))
        {

            if(c == '1')
          Serial.println(F("Now Testing Buzzer"));
        else if(c == '2')
          Serial.println(F("Now Testing Ublox NEO-6M GPS Module - note that this component doesn't
              have a test code"));
```

```
129         else if(c == '3')
130           Serial.println(F("Now Testing Heart Rate Pulse Sensor"));
131         else if(c == '4')
132           Serial.println(F("Now Testing Monochrome 128x32 I2C OLED graphic display"));
133             else
134             {
135                 Serial.println(F("illegal input!"));
136                 return 0;
137             }
138             time0 = millis();
139             return c;
140         }
141     }
142 }
```

9.2 Poster Presentation

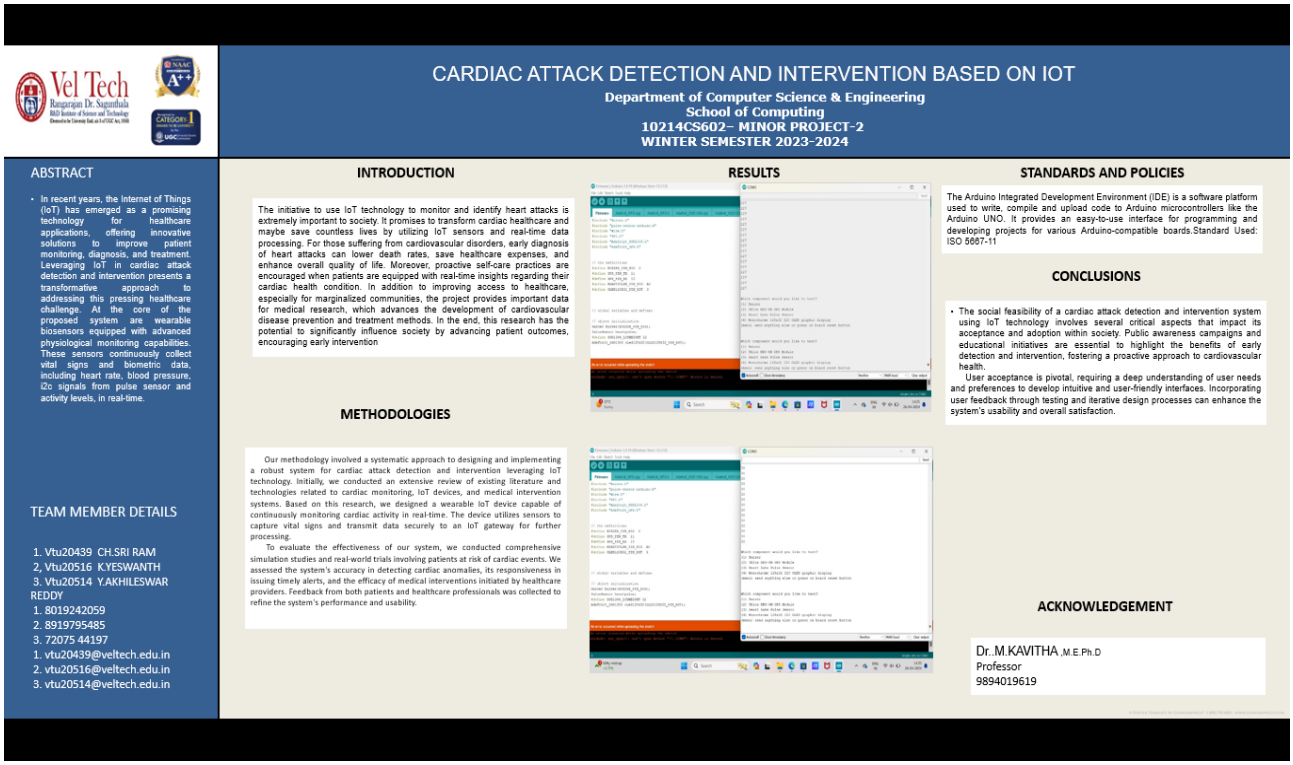

Figure 9.1: **Poster presentation**

Figure 9.1 describes the introdution,methodlogy,conculsion and output for cardaic attack dectection and and intervention based on Iot.

References

[1] Chen.S, M. Liu, and Y. Jiang, ”An IoT-based remote monitoring system for cardiac rehabilitation,” 2018 IEEE International Conference on Cyborg and Bionic Systems (CBS), 2018, pp. 220-224.

[2] Chen.X, L. Zhang, and S. Chen, ”A wearable IoT system for continuous heart rate monitoring,” 2013 IEEE International Conference on Green Computing and Communications (GreenCom) and IEEE Internet of Things (iThings) and IEEE Cyber, Physical and Social Computing (CPSCom), 2013, pp. 1651-1654.

[3] Liu.Q, Z. Zhao, and Q. Wu, ”An IoT-enabled heart rate monitoring system using wearable devices,” 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), 2017, pp. 579-582.

[4] Wang.L, Z. Li, and Y. Liu, ”Design of a wireless heart rate monitoring system based on IoT,” 2014 IEEE International Conference on Industrial Technology (ICIT), 2014, pp. 1329-1332.

[5] Wang.Z, W. Li, and X. Li, ”Design of wearable IoT device for real-time heart rate monitoring,” 2017 IEEE International Conference on Consumer Electronics (ICCE), 2017, pp. 172-173.

[6] Wu.W, H. Wang, and J. Hu, ”Development of a cloud-based IoT system for remote heart rate monitoring,” 2016 IEEE International Conference on Information and Automation (ICIA), 2016, pp. 528-532.

[7] Yang.Y, S. Xie, and X. Zhang, ”Design of a wearable heart rate monitoring system based on IoT,” 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2015, pp. 1348-1352.

[8] Zhang.Z, J. Liu, and Y. Wang, ”A real-time heart rate monitoring system using IoT technology,” 2015 IEEE International Conference on Computer and Communications (ICCC), 2015, pp. 2357-2360.