Final Year Project

IoT Based ECG Monitor

By,

Kuiyu DING - 62716
Fahadh Mohamed JAHEER HUSSAN - 62720
Nuhu BELLO - 62713
Muhammad Arshad Shamshudeen AHMED - 62708

Supervised by,

Saad EL JAOUHARI (ISEP)
Marou Mehri (Epsidy)
Shidi Xia (Epsidy)

Table of the content

1.Introduction

In the dynamic landscape of the Internet of Things (IoT), the fusion of healthcare and technology has paved the way for innovative real-time health monitoring solutions. The ECG Monitoring System presented in this project aims to provide real-time monitoring of ECG data with the use of the Pan and Tompkins algorithm for R-peak detection and BPM computation.

This system is designed to integrate with a mobile application, allowing users to monitor their ECG recordings conveniently and store these recordings in a database, this project is given by Epsidy.

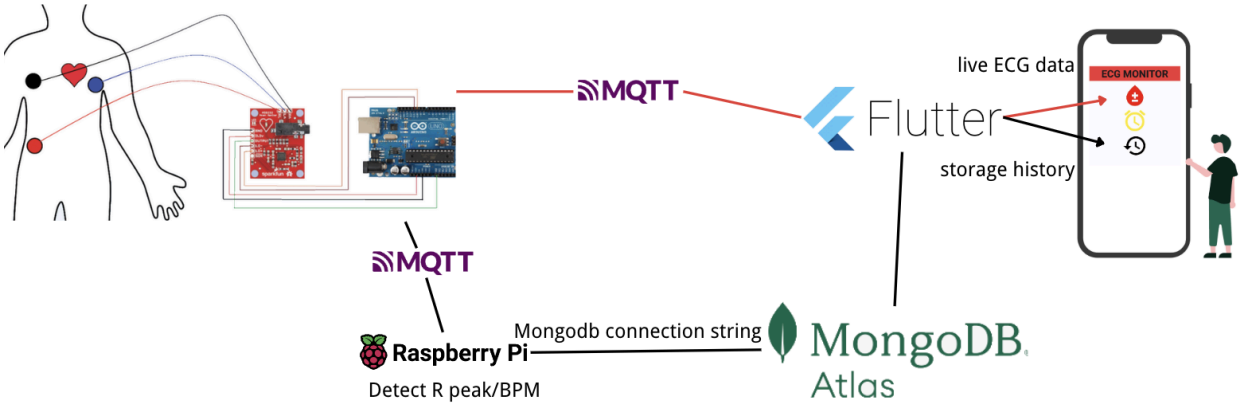
Epsidy is a startup company that focuses on developing innovative solutions in the healthcare sector. The company is known for developing a data-driven garment to diagnose heart disease in women, as well as MRI-compatible tools for use in cardiovascular diseases. Epsidy is also involved in research and development related to generative adversarial networks (GANs) for synthesizing activity.

This project integrates advanced technologies, starting by connecting the ECG sensor to an Arduino for efficient data capture of ecg signal. Using MQTT ensures a reliable data flow between the Arduino and raspberry pi. The Raspberry Pi processes ECG data, identifying R peaks and calculating BPM with the Neurokit2 algorithm, enhancing accuracy for early irregularity detection.

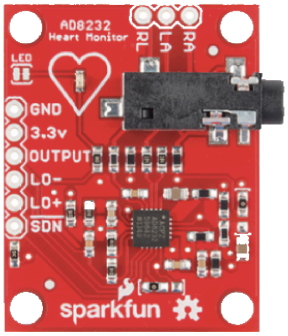
The project achieves real-time ECG data visualization and storage. It presents data from the Arduino through MQTT on an Android app, enabling active health monitoring. Simultaneously, recorded ECG data is securely stored in MongoDB Atlas, accessible via the app. This ensures a comprehensive dataset for analysis, allowing healthcare professionals and users to track trends and gain insights.

The final report explores project phases, detailing ECG sensor integration, data processing intricacies, and visualization/storage implementation.

The architecture of the project:

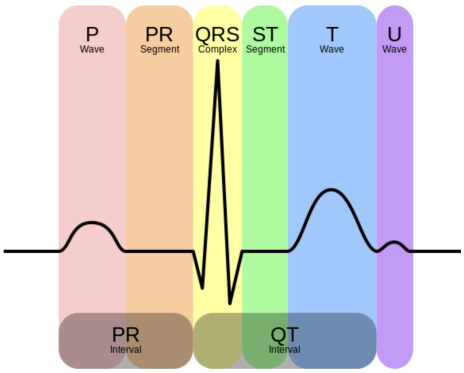


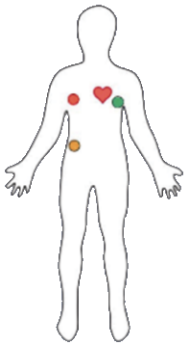
2.AD8232 ECG Sensor and Arduino



The AD8232 is a very useful sensor that measures the heart's electrical activity, Known as ECG. ECG, also known as EKG or Electrocardiography, is measuring electrical changes generated by cardiac movement.

This sensor is a cost-effective board. This electrical activity can be charted as an ECG or Electrocardiogram and output as an analog reading. However, ECGs can be extremely noisy, and the AD8232 Single Lead Heart Rate Monitor acts as an op-amp to help obtain a clear signal from the **PR** and **QT** intervals easily.
Place the sensor adhesive electrode pads as shown in the below right image:

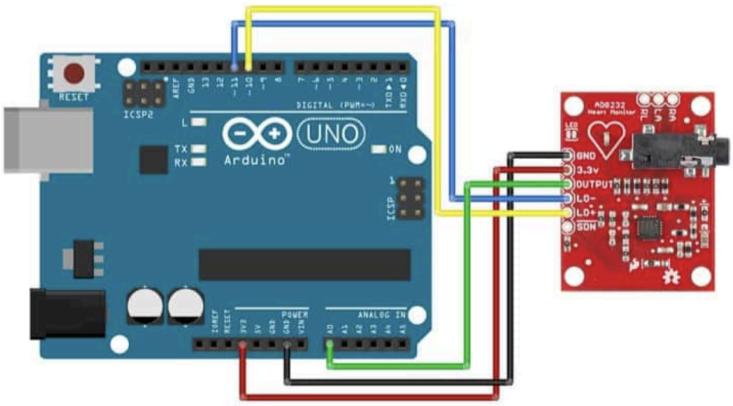




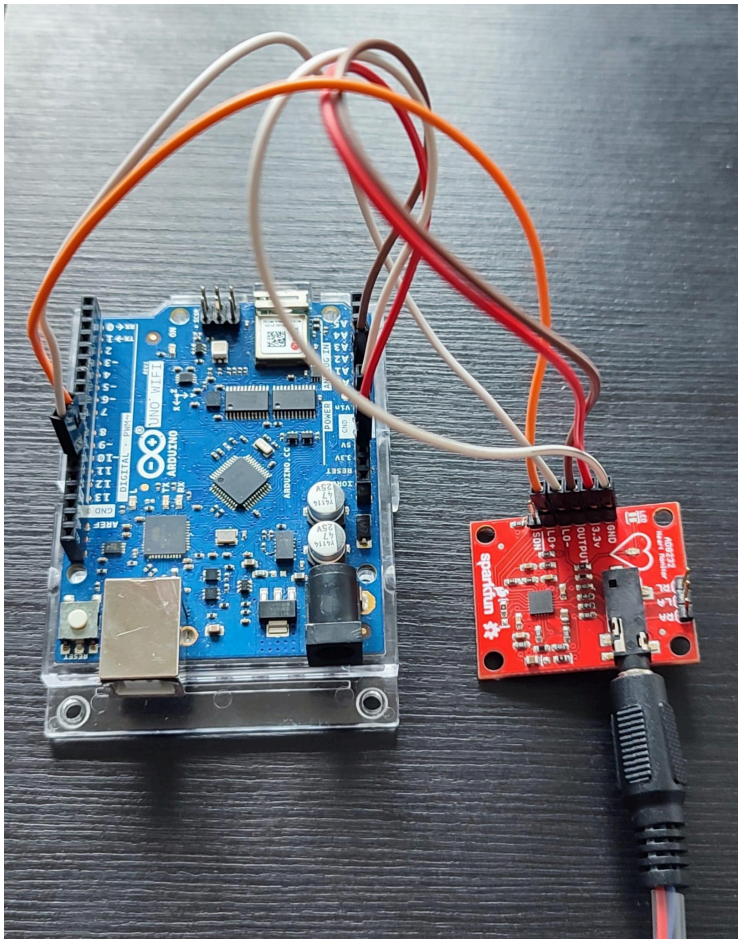
Red: RA (Right Arm)
Yellow: LA (Left Arm)
Green: RL (Right Leg)

Interfacing of AD8232 ECG Sensor With Arduino

Connect the sensor to the Arduino according to the above diagram.



Connect the GND of the sensor to the GND of the Arduino.
Connect 3.3V to the 3.3V of the Arduino.
Connect an output of the sensor to the A0 of the Arduino.
Connect LO- to PIN 11 of the Arduino.
Connect LO+ to PIN 10 of the Arduino.
Keep SDN pin unconnected



3.MQTT

MQTT, which stands for MQ Telemetry Transport, is a lightweight messaging protocol for the Internet of Things (IoT) and machine-to-machine communication. It is designed for resource-constrained devices and provides a simple way to distribute data. MQTT uses a publish/subscribe model, where devices publish messages to a central broker, and other devices subscribe to the messages they are interested in.

This protocol is known for its efficiency, low bandwidth usage, and support for various quality of service levels for message delivery. MQTT is widely used in industries such as automotive, energy, and telecommunications, and it is the de facto standard for IoT messaging. The protocol was initially developed by IBM in 1999 and later became an open standard maintained by the Organization for the Advancement of Structured Information Standards (OASIS).

MQTT Client in Arduino :

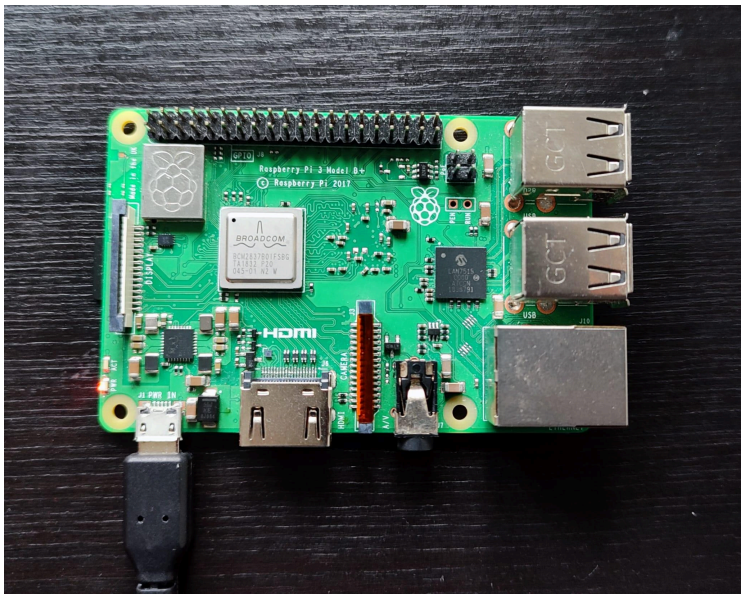
On the Arduino side, we have a program written to acquire ECG signals from the sensor. The Arduino is configured to act as an MQTT client, and using the MQTT protocol,using the MQTT library it publishes the ECG data to a specific topic on the Mosquitto broker running on the Raspberry Pi. The topic could be chosen to reflect the nature of the data being transmitted, such as "ecgData."

MQTT Broker in Raspberry Pi (Mosquitto):

The Raspberry Pi, functioning as an MQTT broker with **Mosquitto** installed, is responsible for managing the communication between the Arduino and potential subscribers interested in receiving the ECG data.

The broker subscribes to the "ecgData" topic, and we got the ecg data by subscribing to this topic and we received real-time ECG data as it is published by the Arduino.

4.Raspberry PI

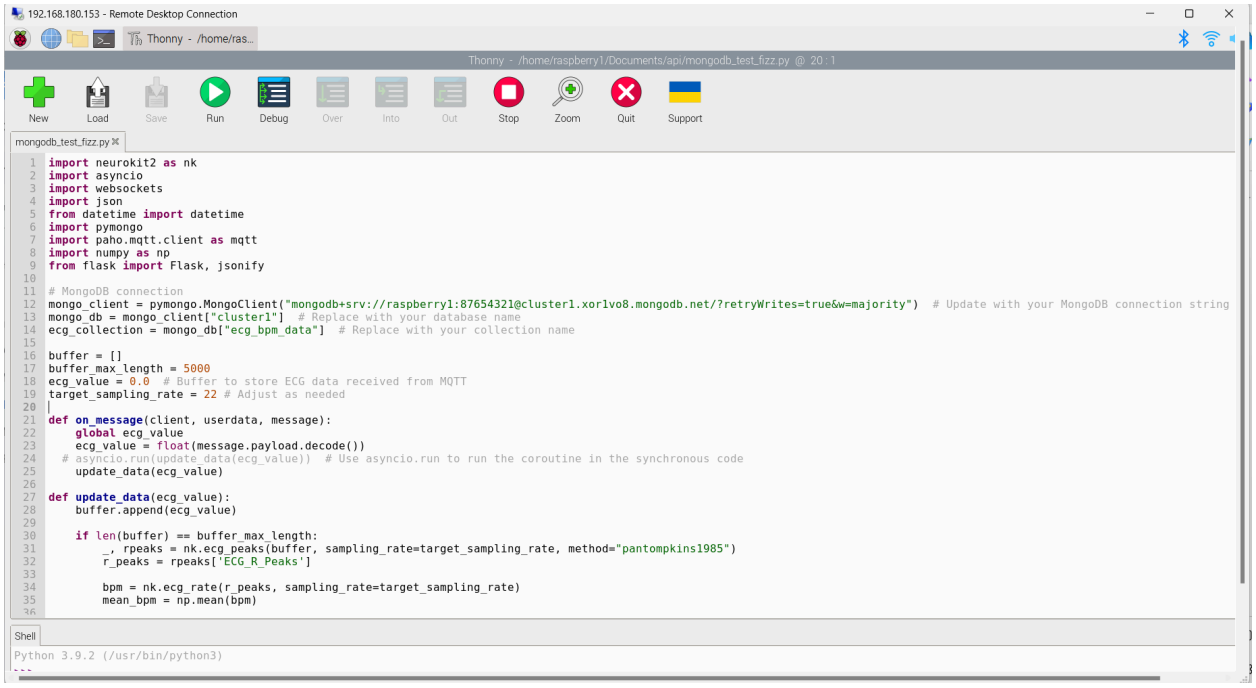


Neurokit2 :

NeuroKit2 is a Python library designed for physiological signal processing and analysis, with a particular focus on signals related to neuroscience and physiology.

Developed to be user-friendly and versatile, NeuroKit2 provides a range of functions that make it easy for researchers and practitioners to perform complex analyses on various physiological signals, including electrocardiography (ECG), electrodermal activity (EDA), and electromyography (EMG).

The library is built on top of other popular scientific computing libraries in Python, such as NumPy, SciPy, and Matplotlib, making it seamlessly integrate into the broader scientific computing ecosystem.



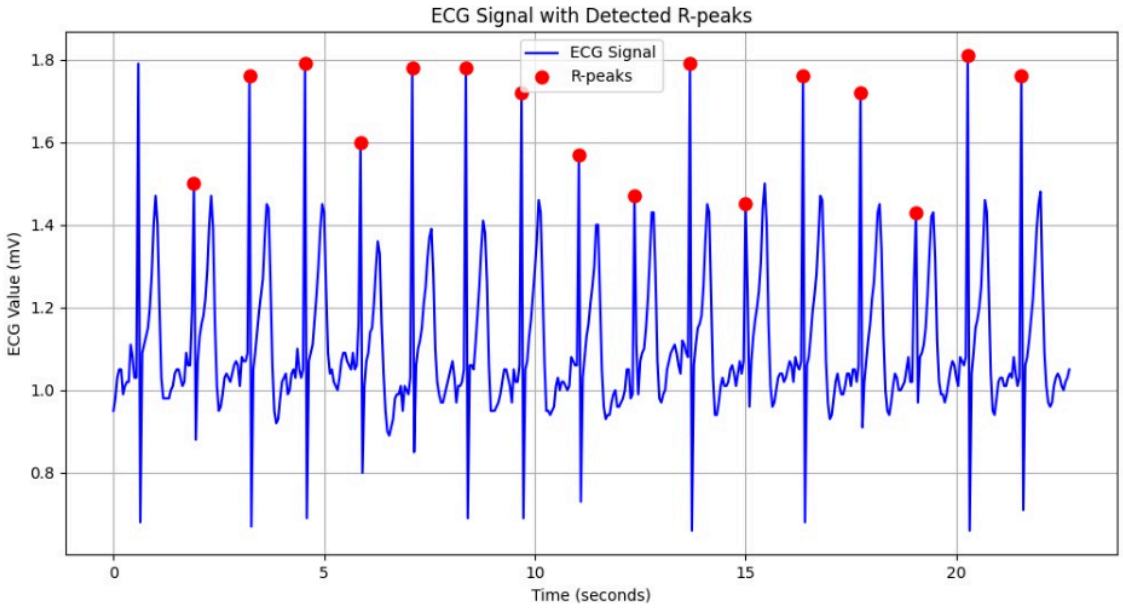
R peak detection and calculation of BPM :

After getting the data from MQTT protocol, using the library “paho.mqtt”. The buffer is a Python list used to store a certain number of ECG data samples (500 samples). In this case, it acts as a sliding window with a maximum length of buffer_max_length. As new ECG data is received, it is appended to the buffer, and when the buffer reaches its maximum length, processing is triggered.

The neurokit2 library, imported as nk, The **nk.ecg_peaks** function from NeuroKit2 is utilized to detect R-peaks in the ECG signal. R-peaks correspond to the peaks in the QRS complex of the ECG waveform and are indicative of individual heartbeats. The algorithm used for peak detection is the Pan-Tompkins algorithm, specified with the **method = "pantompkins1985"** parameter. The ECG data is sampled at a rate of **22Hz.**

The nk.ecg_peaks function in NeuroKit2 is used for detecting R-peaks in an ECG signal. The parameters included in this function are as follows:

Buffer: The ECG signal data stored in the buffer.
Sampling_rate: The target sampling rate, which is set to 22HZ in the code.
Method: The algorithm chosen for R-peak detection, specified as "pantompkins1985."



Pan Tompkins Algorithm :

The Pan-Tompkins algorithm is a widely used real-time method for identifying R-peaks in electrocardiogram (ECG) signals. Developed by Pan and Tompkins, it employs signal processing steps like bandpass filtering, differentiation, squaring, and integration to detect the QRS complexes, specifically the R-peaks. This algorithm is renowned for its simplicity and effectiveness, making it a popular choice in applications like heart rate monitoring and arrhythmia detection, contributing significantly to advancements in ECG signal processing and cardiovascular studies.

In the `neurokit2` library, the Pan and Tompkins algorithm for R-peak detection in ECG signals is implemented with default parameters that typically include:

1. Bandpass Filter Parameters: Usually set to a frequency range of 5 to 15 Hz.
2. Derivative Filter: Based on the standard formula from the original algorithm.
3. Squaring Function: To emphasize larger differences by making all data points positive.
4. Integration Window: Size depends on the sampling rate, often around 150 ms.
5. Adaptive Thresholds: For R-peak detection, initially set based on signal levels and updated dynamically.
6. Refractory Period: Generally around 200-300 ms to avoid multiple detections of the same QRS complex.

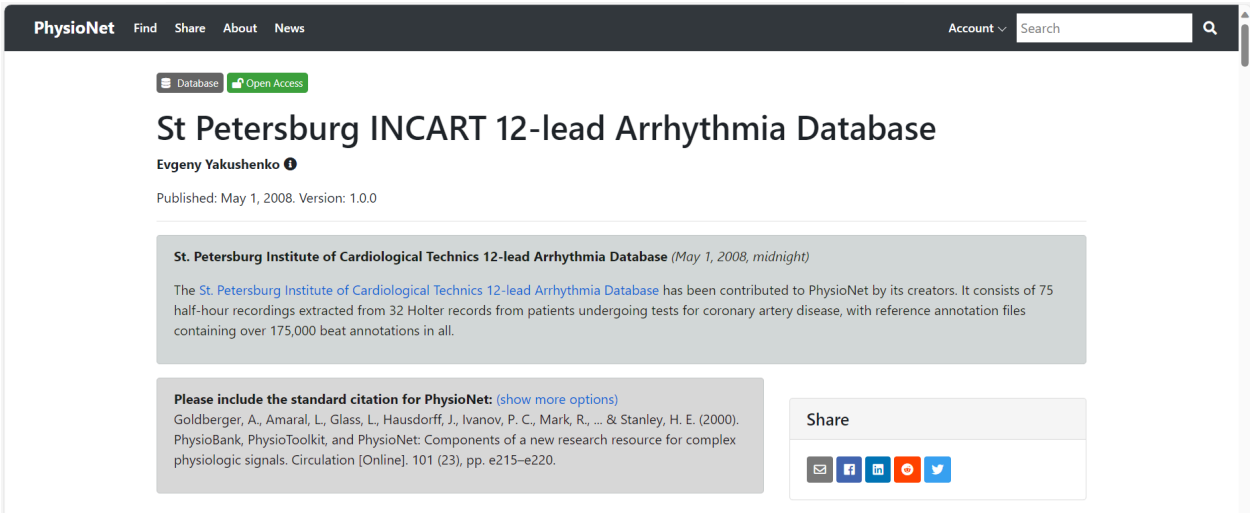
These parameters are set to work well across various ECG signals but can vary slightly depending on the `neurokit2` version.

After detecting R-peaks, the code calculates the heart rate (bpm) using the **nk.ecg_rate** function from NeuroKit2. The function takes the R-peaks and the sampling rate (target_sampling_rate) as parameters. The resulting bpm values are then averaged using NumPy's np.mean function.

```
>>> [ 16  37  69  79  99 109 128 139 157 187 215 246 274 303 330 360 388 417
  442 468 494]
62.78168265237371
Time required to process a 500 sample window: 0.016302824020385742 seconds
[  9  18  49  61  80  92 112 120 145 157 178 193 211 222 243 271 298 324
 350 375 404 437 469]
74.77549994861864
Time required to process a 500 sample window: 0.013710975646972656 seconds
[  8  18  26  33  65  76  98 126 152 181 212 244 277 311 343 374 405 440
 475]
66.6117230796061
```

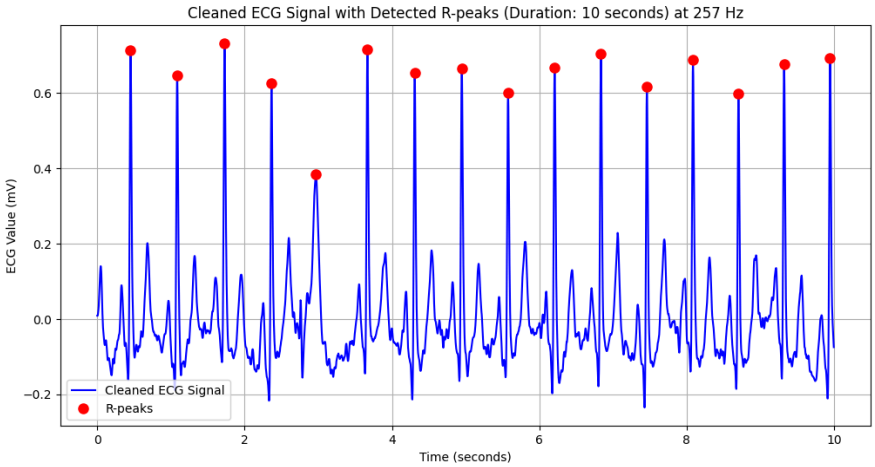
Time required to process a 500-sample window is approximately ~ 0.015 seconds.

5.Physionet :



In our testing process, we used PhysioNet, a repository of physiological signal databases, to obtain ECG signal data. On the Raspberry Pi, we utilized the wfdb library to access and retrieve this ECG data. After obtaining the data, we created a buffer of 5000 samples to store the ECG signal.

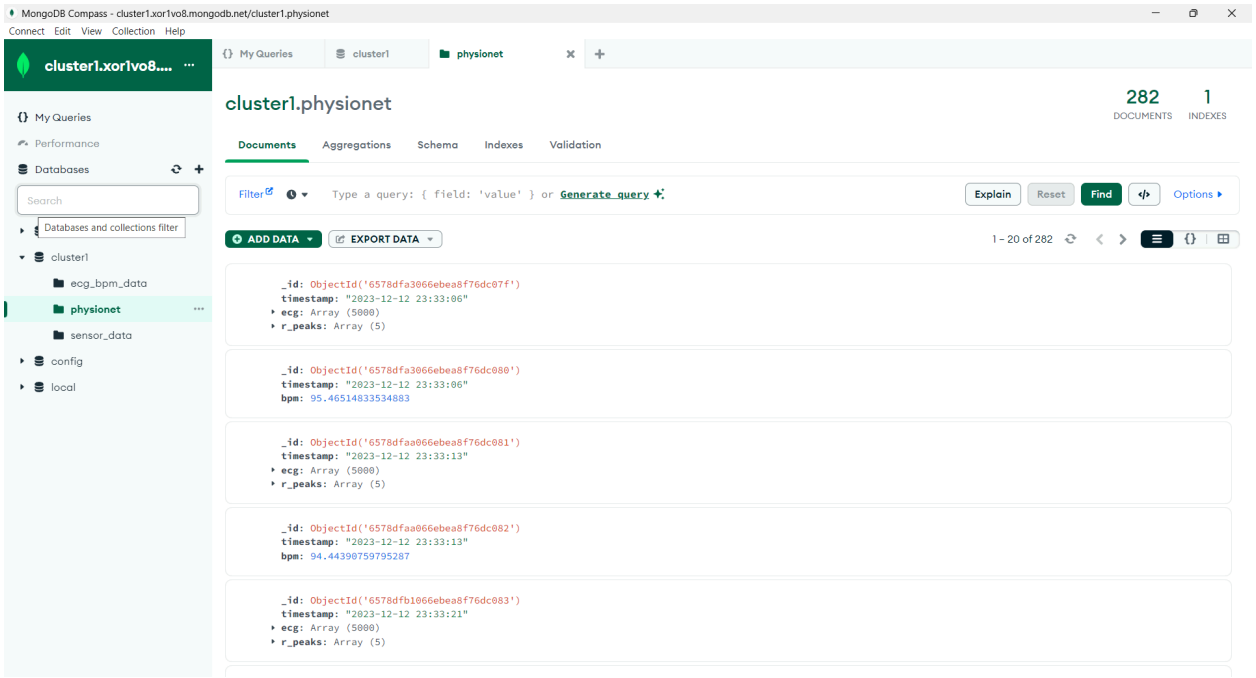
The NeuroKit library, specifically its built-in algorithm, was employed to detect R-peaks within the ECG signal. Simultaneously, the heart rate (bpm) was calculated based on the identified R-peaks. The ECG data is sampled at a rate of **257Hz.**



Websocket :

WebSocket is a communication protocol that enables real-time, bidirectional data exchange between a client (such as a web browser) and a server. It provides a persistent connection, allowing both parties to send messages to each other at any time, facilitating efficient and low-latency communication. WebSocket is commonly used in web applications for features requiring real-time updates, like chat applications, live notifications, and interactive gaming.

Subsequently, the processed data, including the 5000 ECG samples, the detected R-peaks using the NeuroKit algorithm, and the calculated bpm, were stored in a MongoDB Atlas database. This was achieved using WebSocket communication, a communication protocol that enables real-time bidirectional communication between a server and clients. WebSocket likely facilitated the transmission of data from the Raspberry Pi to the MongoDB Atlas database.



6.MongoDB

MongoDB is a popular NoSQL database known for its flexibility and scalability. Unlike traditional relational databases that use tables and rows, MongoDB stores data in JSON-like documents with dynamic schemas, making data integration easier for certain types of applications. It's designed to handle large volumes of data and complex data structures and is well-suited for cloud applications. MongoDB offers features like full index support, replication, high availability, and horizontal scaling through sharding. It's commonly used in various applications, from simple web apps to complex data processing and analytics platforms. MongoDB supports a range of programming languages and platforms, making it a versatile choice for developers.

MongoDB Atlas :

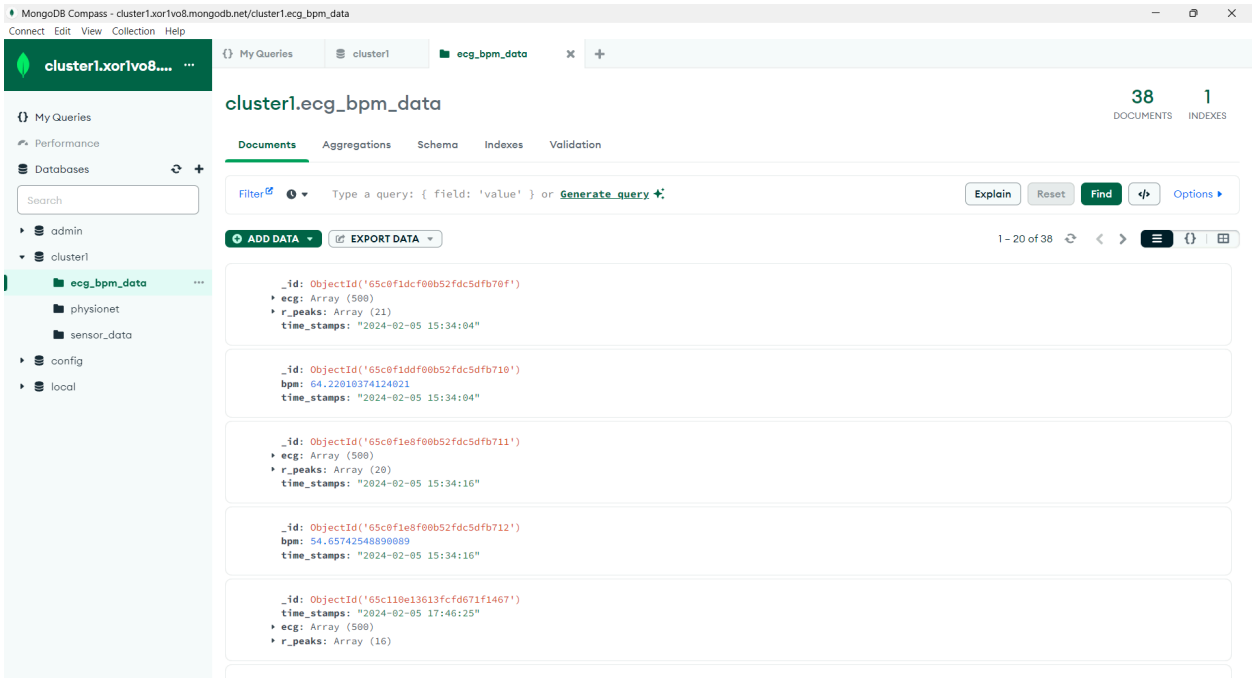
MongoDB Atlas is a fully managed cloud database service by MongoDB. It allows users to easily deploy, scale, and manage MongoDB databases in the cloud, with features such as automated scaling, security measures, backup and restore options, monitoring tools, and global cluster support. It simplifies database management, providing an intuitive interface and handling infrastructure complexities, making it a convenient choice for developers and businesses.

How it Works:

After getting the data of ecg samples (500 samples), r peak and BPM value, a connection to MongoDB Atlas is established using the pymongo library. The MongoDB connection string includes authentication details and the information required to connect to the MongoDB Atlas cluster. The connection initializes a MongoDB client, and from there, a specific database **(cluster1)** and collection **(ecg_bpm_data)** are selected for storing the ECG data, R-peaks, and calculated BPM.

Following the ECG data processing using the NeuroKit library, two recordings are created: ecg_recording and bpm_recording. These recordings encapsulate the timestamp, ECG buffer, R-peaks and BPM, respectively. Subsequently, both recordings are inserted into the specified MongoDB collection **(ecg_bpm_data)** using the insert_one method, ensuring that each buffer's processed data is stored as a document in the collection. This data was sent to Mongodb atlas.

The ObjectId is a unique identifier used by MongoDB to ensure every object in the database can be uniquely identified. The timestamp denotes when the data was recorded. We used Mongodb compass to view this data.




```
_id: ObjectId('65c0f1dcf00b52fdc5dfb70f')
ecg: Array (500)
r_peaks: Array (21)
time_stamps: "2024-02-05 15:34:04"
```

```
_id: ObjectId('65c0f1ddf00b52fdc5dfb710')
bpm: 64.22010374124021
time_stamps: "2024-02-05 15:34:04"
```

```
_id: ObjectId('65c0f1e8f00b52fdc5dfb711')
ecg: Array (500)
r_peaks: Array (20)
time_stamps: "2024-02-05 15:34:16"
```

```
_id: ObjectId('65c0f1e8f00b52fdc5dfb712')
bpm: 54.65742548890089
time_stamps: "2024-02-05 15:34:16"
```

```
_id: ObjectId('65c110e13613fcfd671f1467')
time_stamps: "2024-02-05 17:46:25"
ecg: Array (500)
r_peaks: Array (16)
```

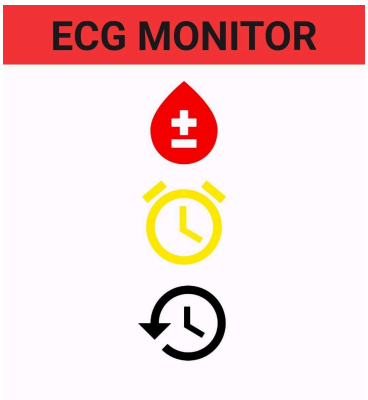
7. Flutter

Flutter is an open-source UI software development toolkit created by Google, designed for building natively compiled applications for mobile, web, and desktop from a single codebase. It allows developers to create visually appealing and high-performance applications with a rich user interface. Flutter uses the Dart programming language and provides a widget-based framework, enabling the creation of expressive and customizable user interfaces.

Application Logo & Name

ECG

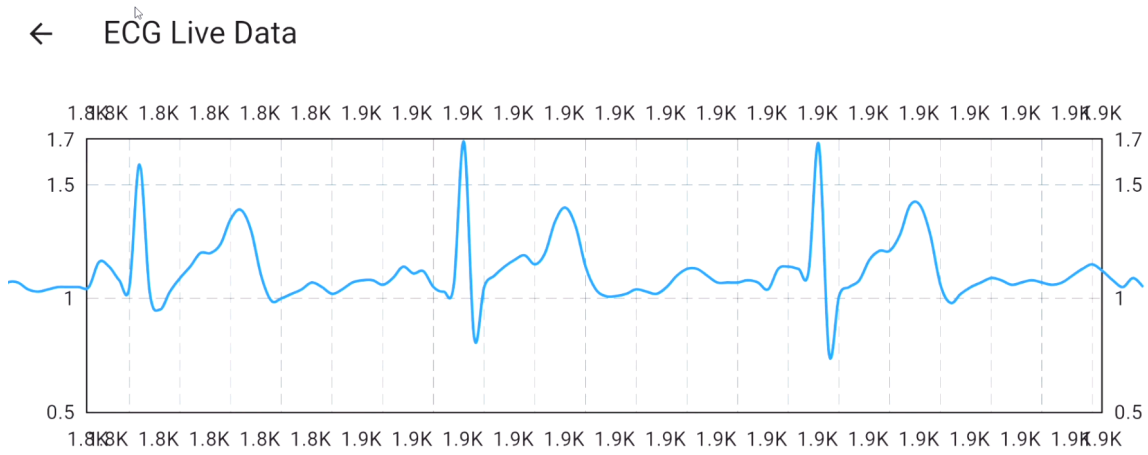
We have developed a mobile application using Flutter to visualize and interact with the ECG data. The mobile application consists of three main parts:



- Live ECG Data
- Alarm
- History

Live ECG Data:

In this section of the application, the live ECG data is obtained from the Arduino using the MQTT protocol. Within the Flutter framework, an MQTT client is established to subscribe to a dedicated topic where the Arduino publishes the ECG data. The Flutter application then receives and processes this real-time data through the MQTT client, allowing users to observe a dynamic and live visualization of the ECG signal within the frontend of the application. Unfortunately, we are currently unable to display the R-peaks in the live ECG data.



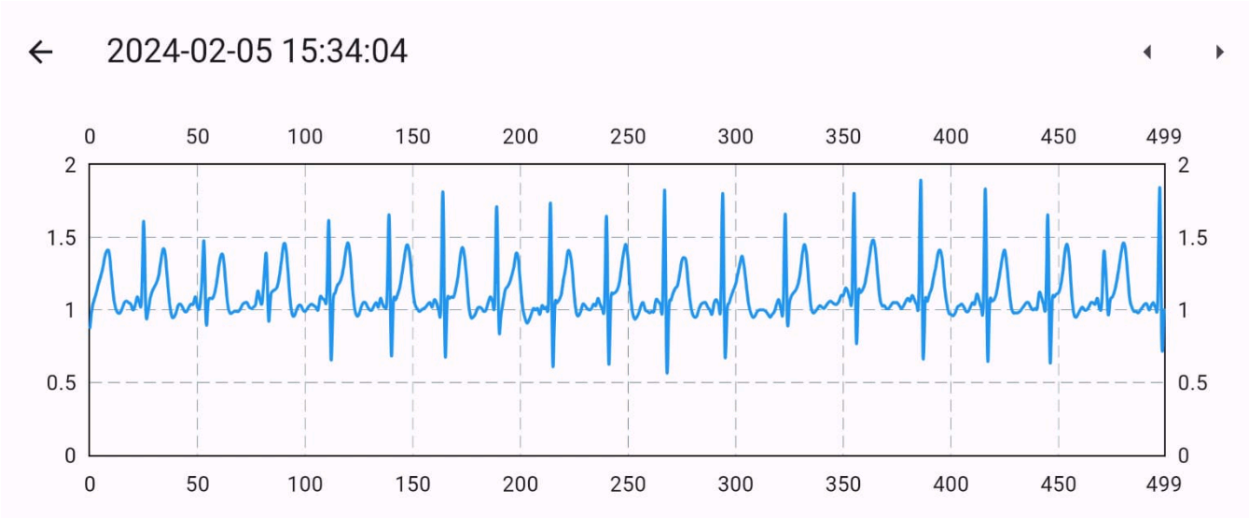
Alarm:

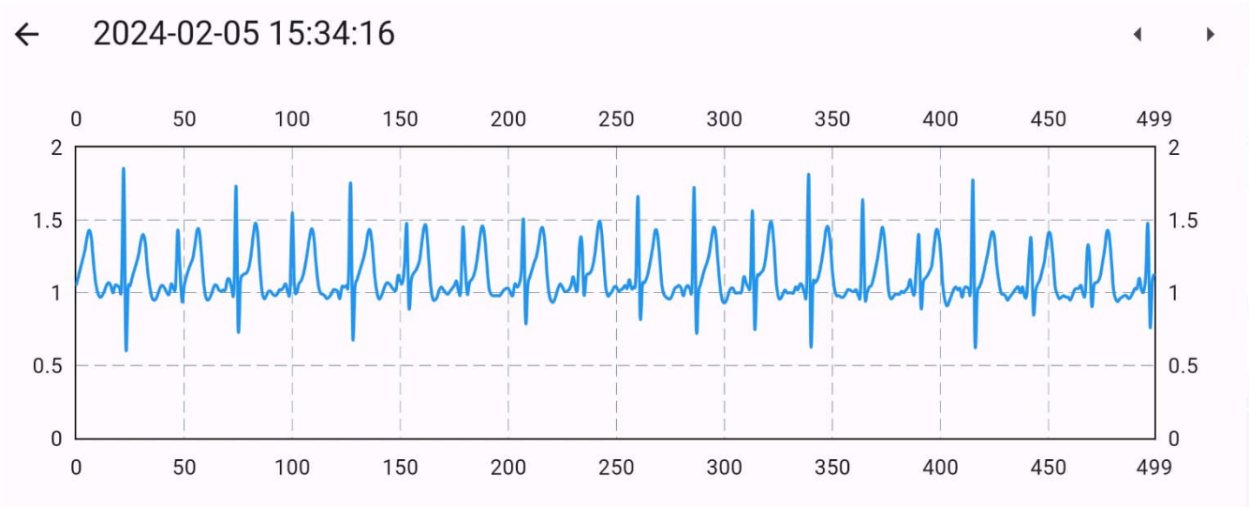
The Alarm section of the application interacts with MongoDB Atlas using a MongoDB connection string in the frontend. It checks the BPM data stored in the database are visually alerts users if the BPM falls below 60 or exceeds 100. This visual indication uses colors, showing red for abnormal BPM values and green for normal ones.

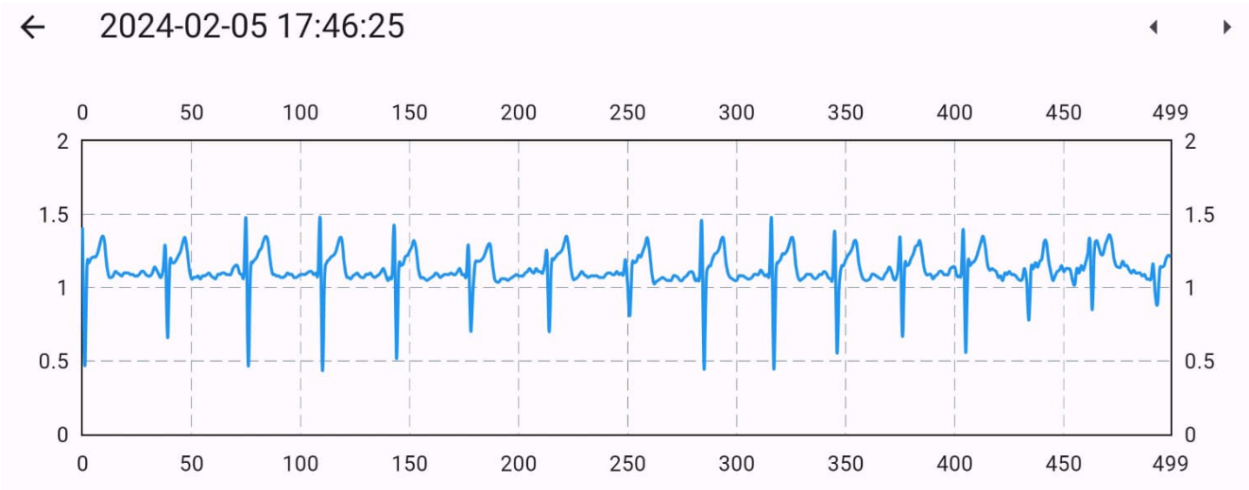
ID	timestamps	bpm
ObjectId("65c151744331d3aeb147b00d")	2024-02-05 23:00:02.618263	46.24517845848016
ObjectId("65c151e839ecf42f31f6d55d")	2024-02-05 23:00:02.618288	72.98292220113852
ObjectId("65c151f339ecf42f31f6d55f")	2024-02-05 23:00:02.618324	72.60539354606772
ObjectId("65c151ff39ecf42f31f6d561")	2024-02-05 23:00:02.618351	83.90312606032688

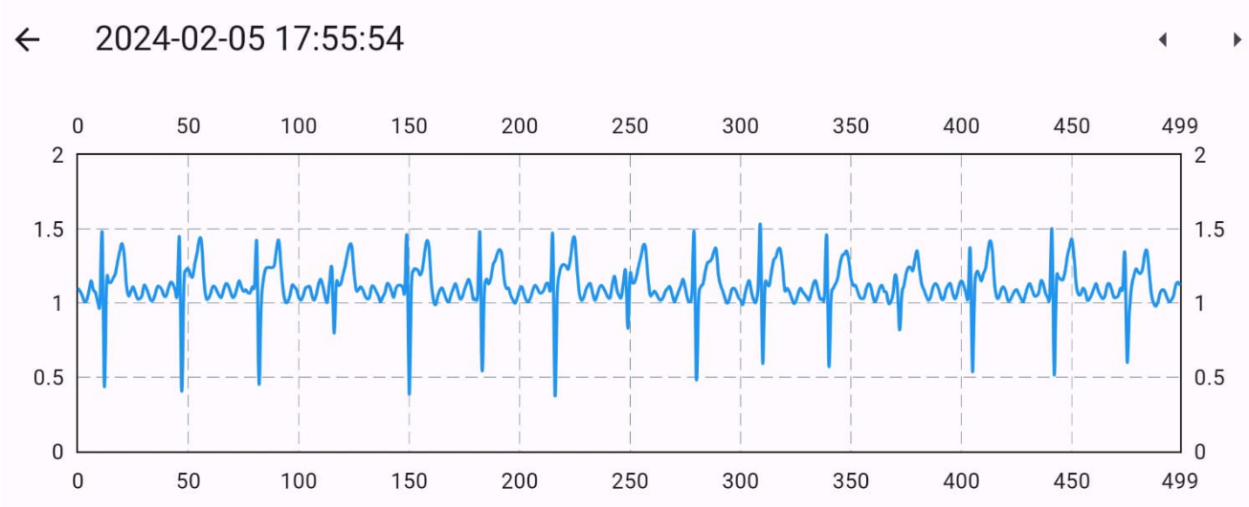
History:

The Storage part of the application retrieves stored ECG samples from MongoDB Atlas using the MongoDB connection string. It allows users to view the ECG signal in graphical form, presenting images of the signal for the last 500 samples stored in the buffer. Additionally, the application supports navigation to the next set of data with a click of the "next" button, enabling users to explore and analyze recorded ECG data.


2024-02-05 15:34:04
0 50 100 150 200 250 300 350 400 450 499
2
1.5
1
0.5
0


2024-02-05 15:34:16
0 50 100 150 200 250 300 350 400 450 499
2
1.5
1
0.5
0


2024-02-05 17:46:25
0 50 100 150 200 250 300 350 400 450 499
2
1.5
1
0.5
0



In summary, Flutter serves as the development framework for your mobile application, providing a unified platform for creating a visually engaging and cross-platform user interface. The application leverages real-time ECG data from the MQTT protocol, as well as recorded data stored in MongoDB Atlas, to offer users features such as live monitoring, alarm notifications, and the ability to explore stored ECG signal graphs.

8.Conclusion

In summary, the project successfully integrates hardware, communication protocols, data processing algorithms, and cloud-based storage to create a holistic ECG monitoring system. The implications extend beyond immediate health monitoring, offering potential applications in preventive healthcare and personalized fitness tracking.

During the course of the project, we encountered a few problems. These include:
Detection of R Peak: Ensuring accurate detection of R peaks in ECG signals, especially in the presence of noise.
Calculation of BPM: Accurately calculating heart rate (BPM) from the detected R peaks while considering variable signal quality.
Live Data Display in Flutter Application:
Achieving real-time data visualization in a Flutter application for a seamless user experience.
Sensor Issues: Addressing potential issues with the ECG sensor, such as signal artifacts or connectivity problems.

Raspberry Pi WiFi Connectivity: Ensuring stable and reliable WiFi connectivity on the Raspberry Pi for continuous data transmission.

This project stands as a testament to the power of technology in transforming healthcare practices and providing valuable tools for both medical professionals and individuals to enhance health and well-being.

Moving forward, this project offers potential avenues for enhancement. Future iterations could focus on integrating machine learning techniques for improved signal processing, expanding compatibility with various ECG devices, and enhancing the user interface for a more seamless experience. The ECG Monitoring System has the potential to evolve into a comprehensive tool for proactive heart health management.