Developing a Feasible Model for the IoT-Based Ubiquitous Healthcare Monitoring System with Data Optimization

**Abstract: The Internet of Things (IoT) has the potential to revolutionize the healthcare industry by enabling efficient and real-time monitoring of patients and healthcare systems. IoT devices can be integrated into healthcare monitoring systems to gather valuable data, improve patient care. The Internet of Things (IoT) has potential applications for connecting different medical devices, sensors, and healthcare experts to deliver high-quality medical services in a remote location. In this project we have planned to design a compact wireless Patient Health Monitoring System and also mobile app which is alert patient in real time when any abnormal situation creates. And sensor send real time data efficient way using data optimization. Patient can monitor their health issue like pulse rate, Body Temperature. Patient also know Room Temperature, Humidity.**

**Key Words: IOT, Health monitoring, Blood Pressure, Heart rate, Sensor, Temperature.**

**INTRODUCTION:**

**IoT(Internet of Things) plays a significant role in healthcare monitoring systems by enabling real-time data collection, remote monitoring, and improved patient care.** Health care is now of extreme importance in all countries with the remote areas. So, in this regard, an IoT-based health monitoring system is the best solution for such an outbreak.

**Remote Patient Monitoring: The IoT healthcare monitoring system allows for remote monitoring of reducing the need for frequent hospital visits.**

**Early Warning and Preventive Care: By continuously monitoring body temperature, pulse rate, humidity, and room temperature, the system can identify abnormal patterns or deviations from normal ranges. This early warning capability enables timely intervention, facilitating preventive care and reducing the risk of complications or emergencies. For example, sudden changes in body temperature or pulse rate can indicate the onset of a fever or irregular heart rhythm, prompting immediate medical attention.**

**Enhanced Accuracy and Efficiency: The IoT healthcare monitoring system provides accurate and consistent measurements of vital health parameters, eliminating potential human errors associated with manual readings. The automated data collection and transmission process ensure efficiency and reliability in monitoring patient health, enabling healthcare providers to make informed decisions based on real-time, objective data.**

**Timely Alerts and Notifications: The system can be programmed to generate alerts and notifications when specific thresholds or conditions are met. For instance, if the body temperature exceeds a predefined limit or the room temperature becomes unsuitable for a patient's well-being, immediate alerts can be sent to healthcare providers or caregivers, enabling prompt action and interventions.**

**Cost and Resource Optimization: The IoT healthcare monitoring system can contribute to cost savings and efficient resource allocation in the healthcare sector. By reducing hospital readmissions, minimizing unnecessary visits, and enabling early interventions, the system can optimize healthcare resources, reduce healthcare costs, and alleviate the burden on healthcare facilities.**

**the IoT healthcare monitoring system based on body temperature, pulse rate, humidity, and room temperature sensors revolutionize healthcare by providing real-time monitoring, early warning capabilities, accurate data collection, data-driven insights, and improved patient care. This project holds immense significance in transforming the way healthcare is delivered, promoting proactive and personalized healthcare management, and ultimately enhancing the quality of life for individuals.**

**Proposed System:**

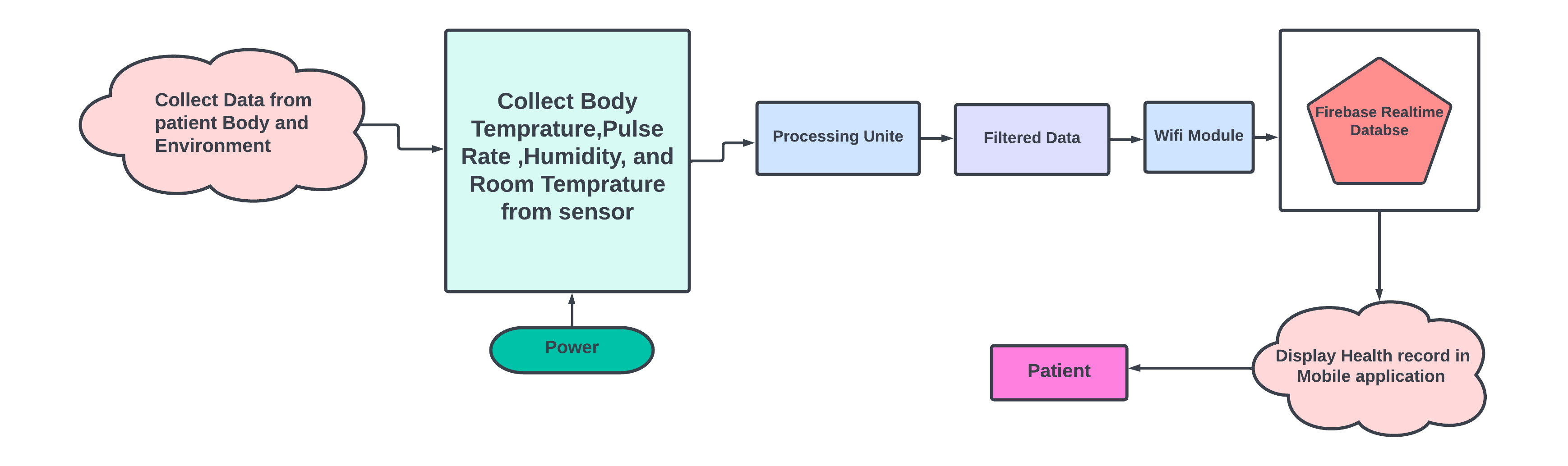
The main objective of the suggested system is to use Wi-Fi to continuously and unbrokenly monitor all of the most important patient health and room condition parameters around-the-clock. Additionally, the system offers an enhanced and automated warning procedure that works with in case of an emergency, internet connectivity and the ability to locate the sufferer. This system send data to firebase Realtime Database patients can monitor their health condition using IOT app which is mobile app in our system. If any abnormal data show then app alert immediately and notify patient.

Patients aware their health condition and take emergency based on real time data. This system use some sensor for collect Realtime data of patient and also send it to Realtime database.

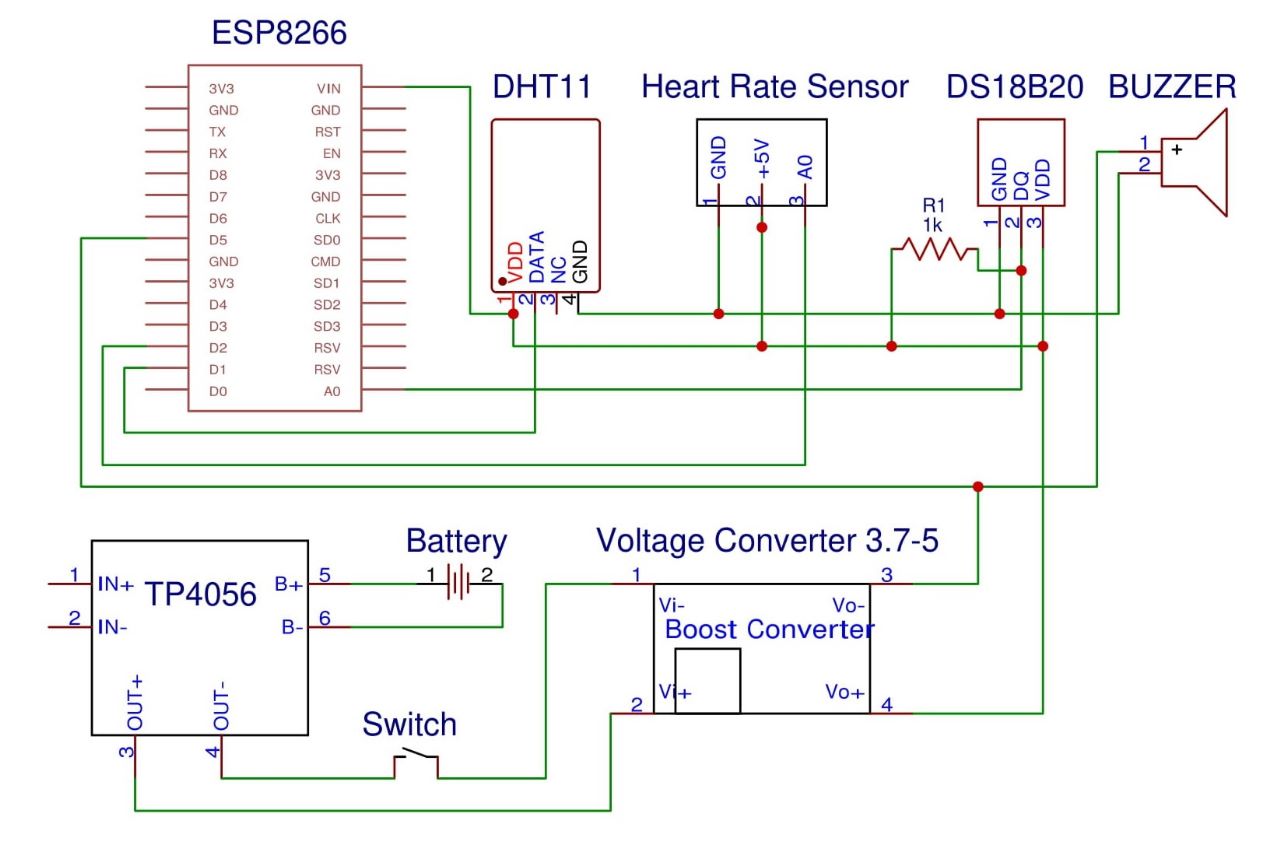
The system integrates several technologies but mainly consists of two components, a hardware device carried by the patient and a mobile application to display the patient’s Realtime and prior health records.

The sensors associated with the device collect the patient’s body and environmental parameters and pass them to the processing unit. Here, we have considered the most valuable health parameters for most common health problems: patient’s body temperature, heart rate or pulse rate. And beside of this collect Room temperature and Humidity.

**Block diagram of the system:**



**Circuit Diagram of System:**

****

MAJOR HARDWARE COMPONENTS

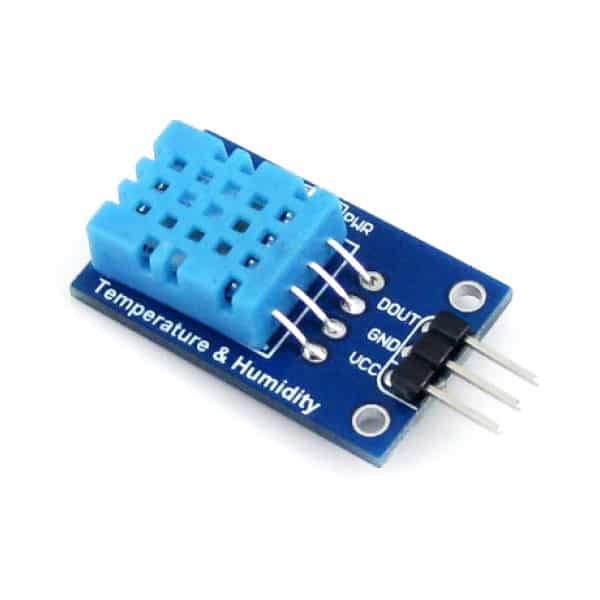
NODE MCU ESP8266: The NODE MCU ESP8266 module is a popular and widely used Wi-Fi module that integrates a microcontroller with built-in Wi-Fi capabilities. It is developed by Express if Systems and has gained significant popularity in the Internet of Things (IoT) and DIY electronics communities. The ESP8266 module allows devices to connect to Wi-Fi networks, enabling them to communicate and interact with other devices or the internet. The module includes a microcontroller unit (MCU) with a low-power 32-bit RISC processor, providing computational capabilities for running applications and executing code. ESP8266 module has GPIO pins. ESP8266 module typically has several general-purpose input/output (GPIO) pins that allow it to interface with various sensors, actuators, and other electronic components. The ESP8266 module can be programmed using various programming languages and development environments. The most commonly used programming language for the module is C/C++, but there are also firmware options, such as NodeMCU, that allow programming in Lua or Arduino IDE.

This Project used ESP8266 module for wifi module which send Realtime Health data to firebase.



**DHT11 Sensor:** The DHT11 is a widely used digital temperature and humidity sensor. It is a low-cost sensor module that provides accurate readings of temperature and relative humidity. The DHT11 sensor is commonly used in various applications, including weather stations, home automation systems, and environmental monitoring devices. In this project used for this sensor collect room temperature and humidity. Compare with this others days temperature. The DHT11 sensor can measure temperature within a range of 0 to 50 degrees Celsius with an accuracy of ±2 degrees Celsius.

The sensor can measure relative humidity within a range of 20% to 90% with an accuracy of ±5%. The DHT11 sensor is known for its affordability, making it a popular choice for applications where cost is a significant consideration. That’s why used this sensor in this project.



Pulse Sensor:

A heart rate sensor module is an electronic device that is designed to measure and monitor a person's heart rate. It consists of a sensor, typically an optical sensor, that detects the blood flow and pulsations in the skin, usually on the wrist or finger. The module may also include additional components such as an analog-to-digital converter, microcontroller, and wireless connectivity options.

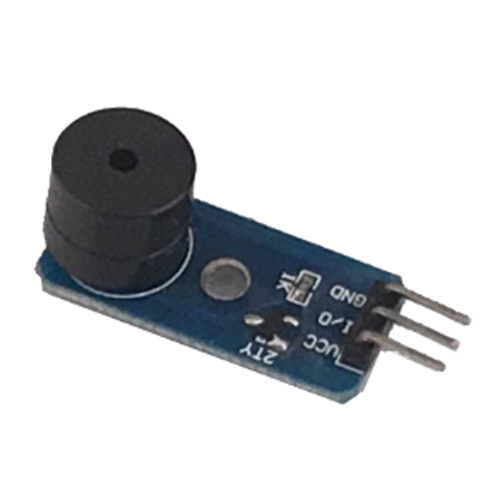
The module measures the heart rate by detecting the changes in blood volume in the capillaries near the skin surface. This data is then processed to calculate the beats per minute (BPM) and provide real-time heart rate information. The module includes signal processing algorithms to filter out noise and extract the heart rate signal from the raw data. These algorithms are designed to provide accurate and reliable heart rate measurements. Heart rate sensor modules may require a power source such as a battery or external power supply. The power consumption varies depending on the module's design and features.



**DS18B20 Sensor**: The DS18B20 is a popular digital temperature sensor that uses the 1-Wire protocol to communicate with a microcontroller or a computer. It is manufactured by Maxim Integrated and is widely used in various applications due to its simplicity, accuracy, and ease of integration. Here are some key features and characteristics of the DS18B20 sensor. The DS18B20 sensor is capable of measuring temperature with a resolution of up to 12 bits, allowing for precise temperature readings. The temperature range supported by the sensor is typically from -55°C to +125°C (-67°F to +257°F). The sensor uses the 1-Wire protocol, which means that multiple DS18B20 sensors can be connected to a single data line. This makes it convenient to use in applications where multiple temperature measurements are required.



Switch: To control external devices or components using an Arduino board, can use various types of switches. The type of switch you choose depends on the specific application and the requirements of project. This project used push Button Switch. A push button switch is a simple mechanical switch that is pressed down to make or break a connection. It is commonly used for momentary actions, such as triggering an event or changing a state. Push button switches can be connected to digital input pins of an Arduino board to detect when the button is pressed or released.

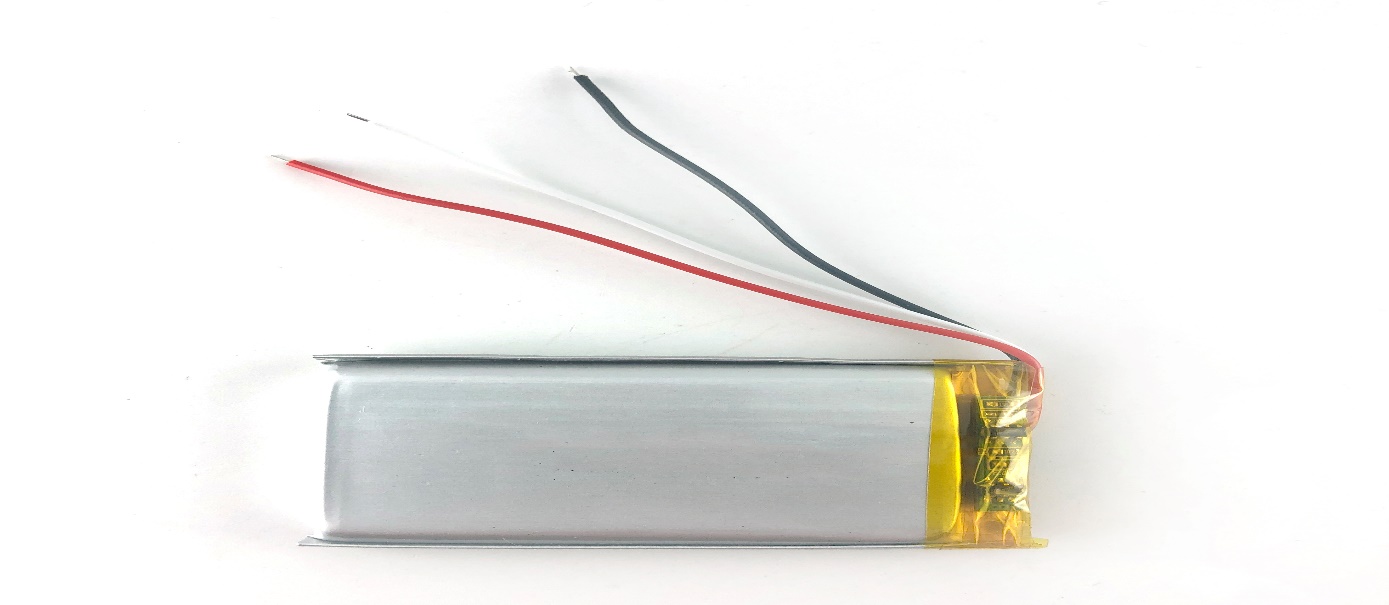


Buzzer: A buzzer for a sensor is a sound-producing device that is often used to provide audible alerts or notifications based on the readings or measurements taken by a sensor. The buzzer is typically connected to the sensor circuit and is activated when certain conditions or thresholds are met. Use Buzzer for alarm when unexpected condition occur in our body.



LIPO Battery :

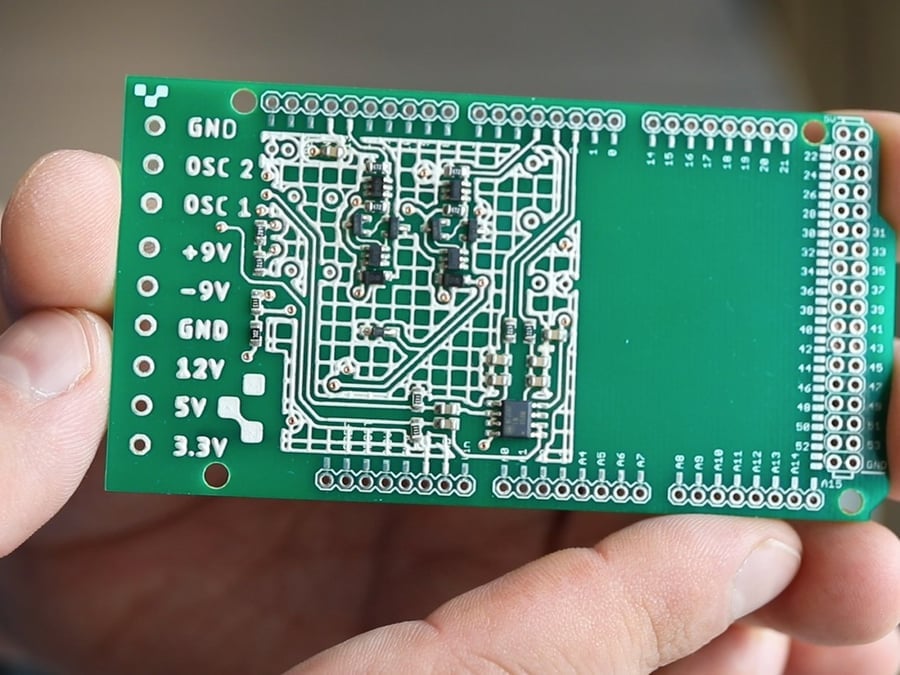
When selecting a battery for a sensor, there are several factors to consider, including the power requirements of the sensor, desired battery life, physical size constraints, and the specific application. Here are some key considerations when choosing a battery for a sensor. Determine the voltage requirement of the sensor. Sensors typically operate at specific voltage levels, such as 3.3V or 5V. Ensure that the battery voltage matches the sensor's requirements or use appropriate voltage regulation circuitry if needed. Consider the power consumption of the sensor and the desired battery life. The capacity of a battery is typically measured in milliampere-hours (mAh) or ampere-hours (Ah) and indicates how much charge the battery can deliver over a specific period.



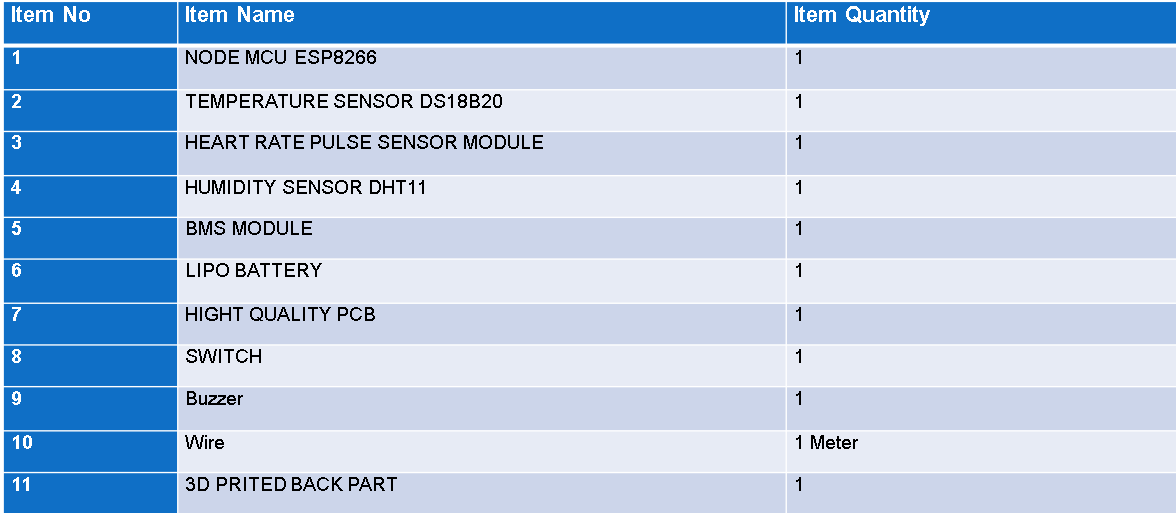
Wire : When attaching a sensor to a surface or object, you may need to use various types of hardware or mounting methods depending on the sensor and its specific requirements. Use I meter wire for attaching senso to Board.



Mounting Plates or PCBs: In some cases, sensors may have mounting plates or PCBs that can be attached to the surface using screws or other mounting methods. These plates or PCBs usually have pre-drilled holes or slots that allow for easy attachment.

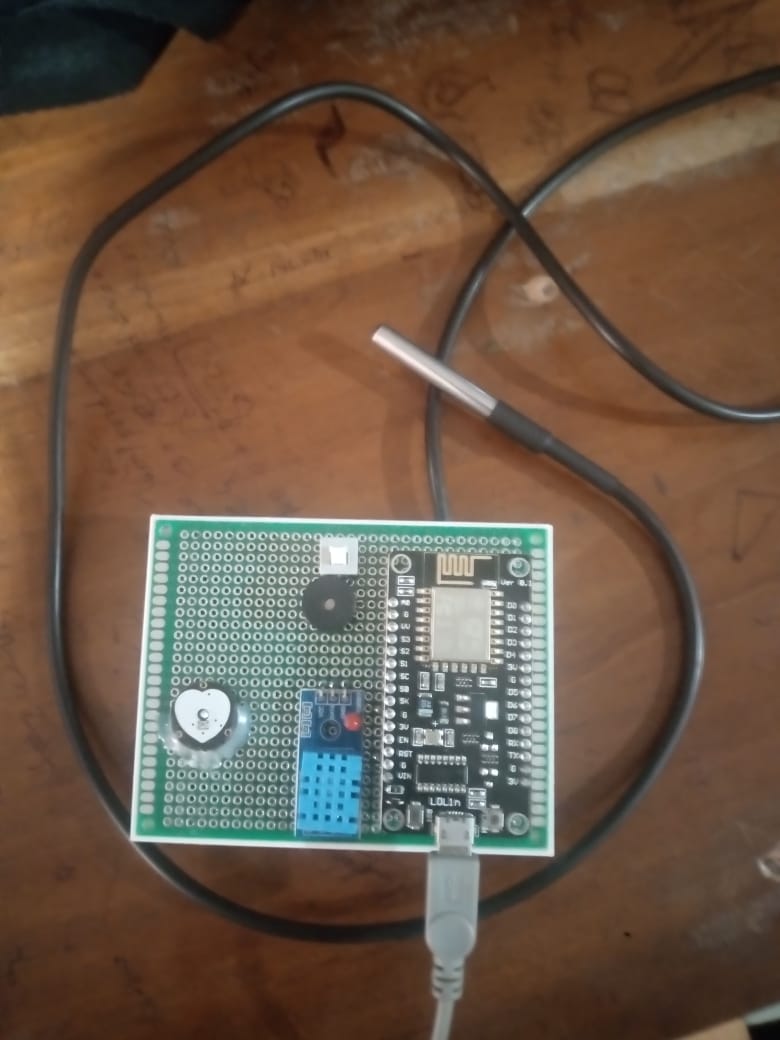


Hardware component List:



Software Components:

Prototype Of the System:





# Project Overview:

In this project first create a IOT device using body temperature, Pulse rate , humidity, and room temperature sensor. WIFI module NODE MCU ESP8266 send data to server we used Firebase Realtime Database for store Realtime health data. For sending sensor data used Arduino IDE and then use Firebase Realtime Database, and show Realtime data to mobile app which is created using android studio with using Java programming language. In this project send data to firebase in efficient way using data optimization techniques like remove redundant data.

The following diagram shows a high-level overview of the project we’ll build.



* Body Temperature
* Room Temperature
* Humidity
* Pulse Rate

Real Time Database

Get Sensor Reading

Store sensor data to Firebase Realtime Database

The ESP8266 gets temperature, humidity and pressure from the BME280 sensor. It gets epoch time right after getting’s the readings (timestamp). The ESP8266 sends temperature, humidity, pressure and timestamp to the database. New readings are added to the database periodically. You’ll have a record of all readings on the Firebase Realtime database. Here add step by step process which is followed by this project.

These are the main steps to complete this project:

1. Use Firebase server
2. [Create Firebase Project](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#create-firebase-project)
3. [Set Authentication Methods](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#Set-Authentication-Methods)
4. [Get Project API Key](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#firebase-project-api-key)
5. [Set up Realtime Database](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#Set-up-Realtime-Database)
6. [Set up Database Security Rules](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#Set-up-Database-Security-Rules)
7. NODE MCU [ESP8266 Datalogging (Firebase Realtime Database)](https://randomnerdtutorials.com/esp8266-data-logging-firebase-realtime-database/#ESP-Datalogging-RTDB)

**Firebase Server**: Firebase Realtime Database is a cloud-hosted NoSQL database provided by Google as part of the Firebase platform. It offers a scalable solution for building real-time applications, allowing developers to store and synchronize data in real time across multiple clients. Google Firebase Google Firebase is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking analytics, reporting and fixing app crashes, creating marketing and product experiment. Firebase offers a number of services, including: • Analytics – Google Analytics for Firebase offers free unlimited reporting on as many as 500 separate events. Analytics presents data about user behavior in iOS and Android apps, enabling better decision-making about improving performance and app marketing.

Authentication – Firebase Authentication makes it easy for developers to build secure authentication systems and enhances the sign-in and onboarding experience for users. This feature offers a complete identity solution, supporting email and password accounts, phone auth, as well as Google, Facebook, GitHub, Twitter login and more.

Realtime database – the Firebase Realtime Database is a cloud-hosted NoSQL database that enables data to be stored and synced between users in real time.

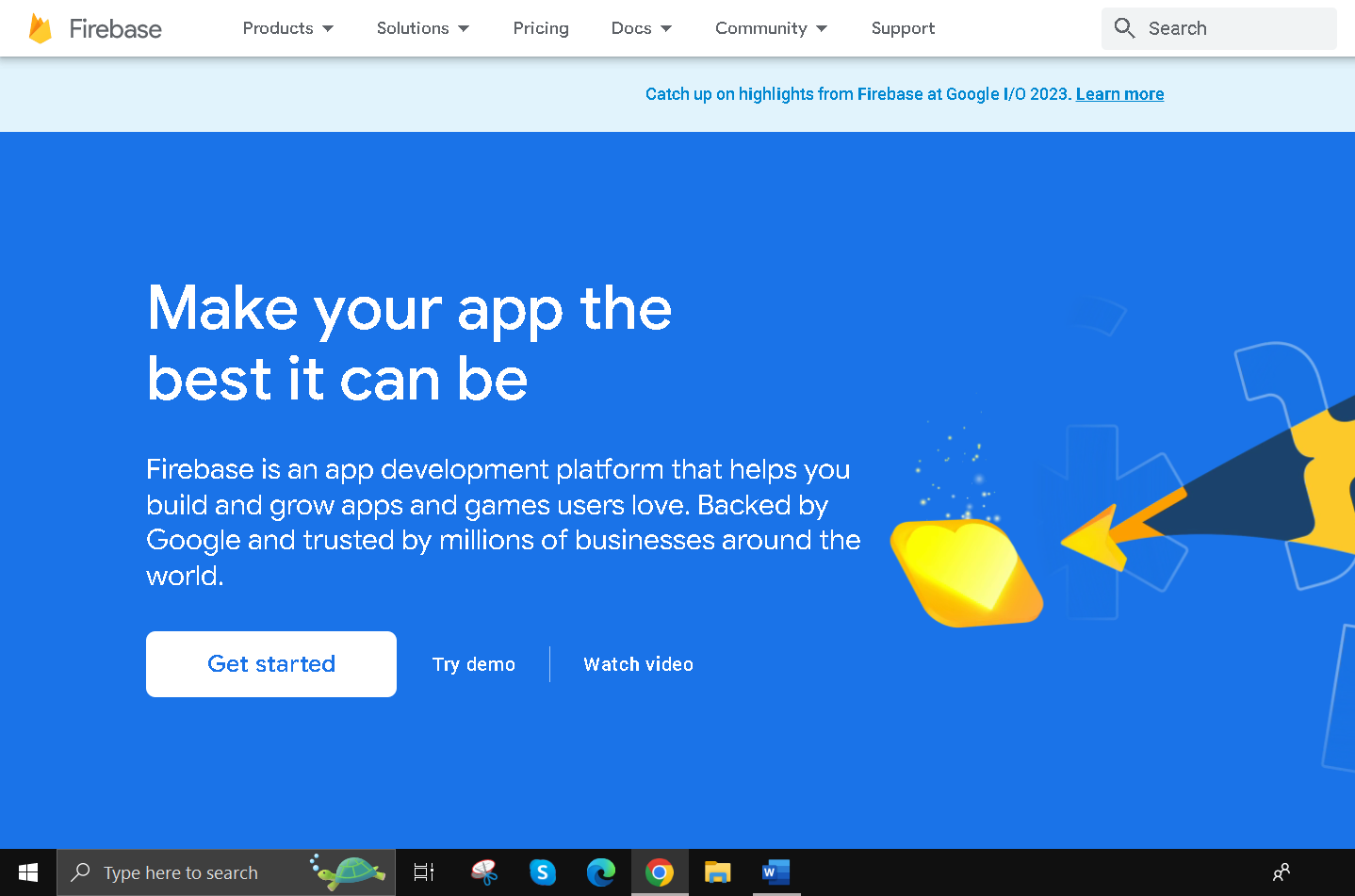
So in this project use Firebase Realtime database to show Realtime health monitoring data like body temperature, pulse rate, humidity and temperature of these environment.



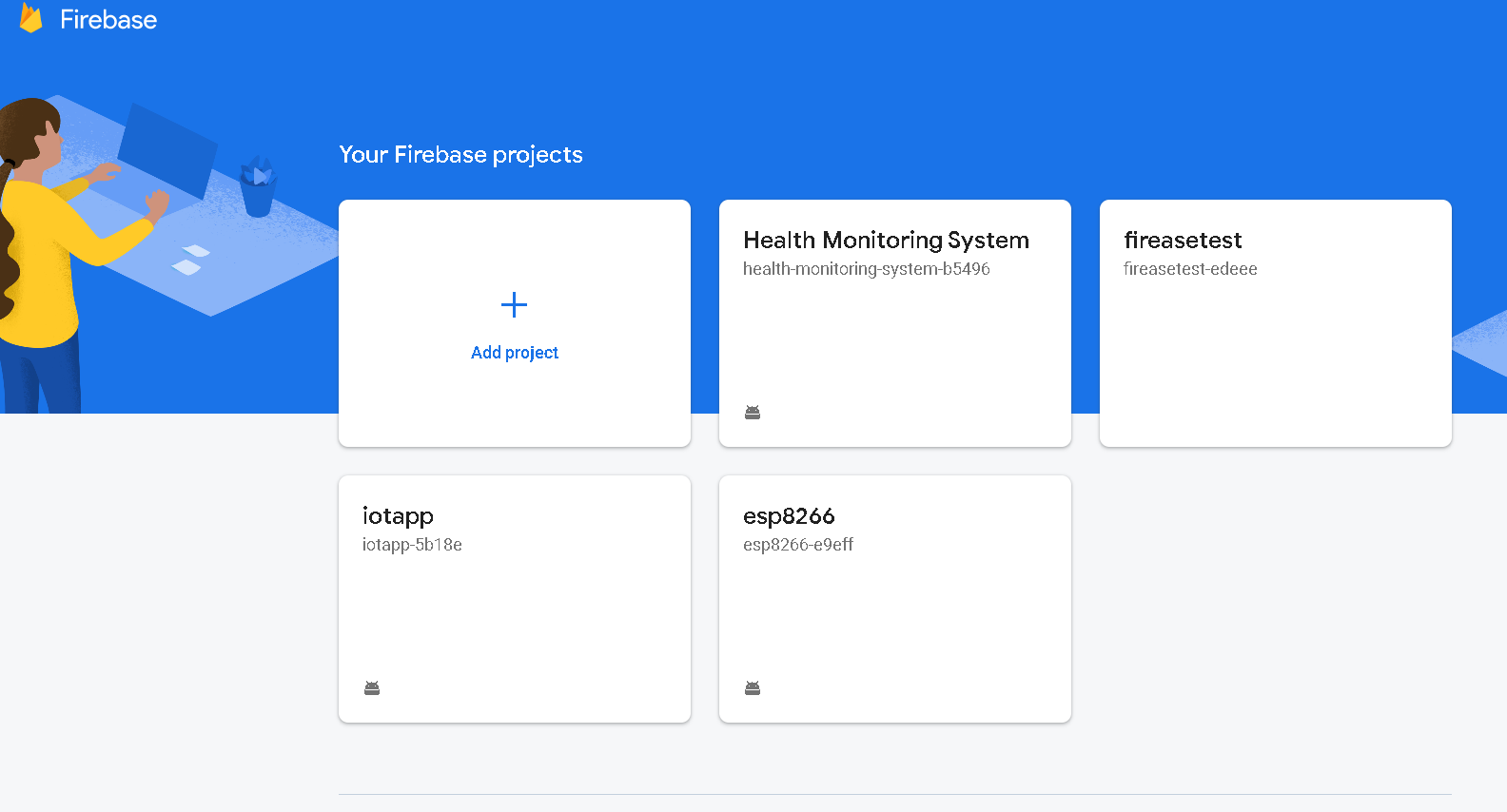
For using firebase Realtime database uses these step.

## Create Firebase Project

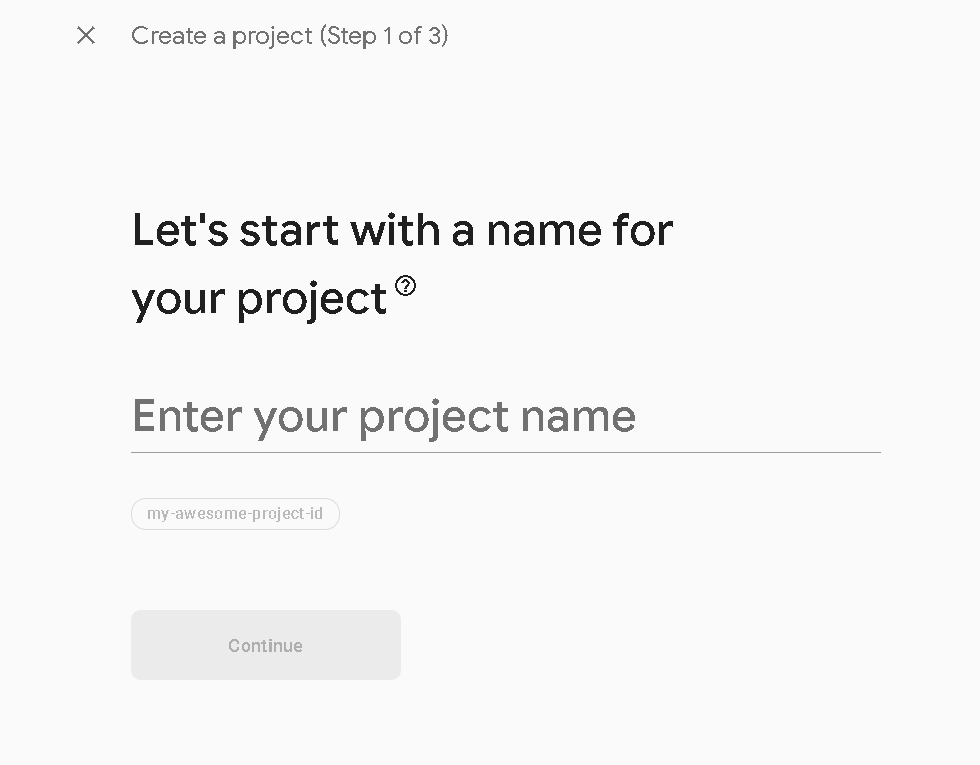
**1)** Go to [Firebase](https://firebase.google.com/)and sign in using a Google Account.

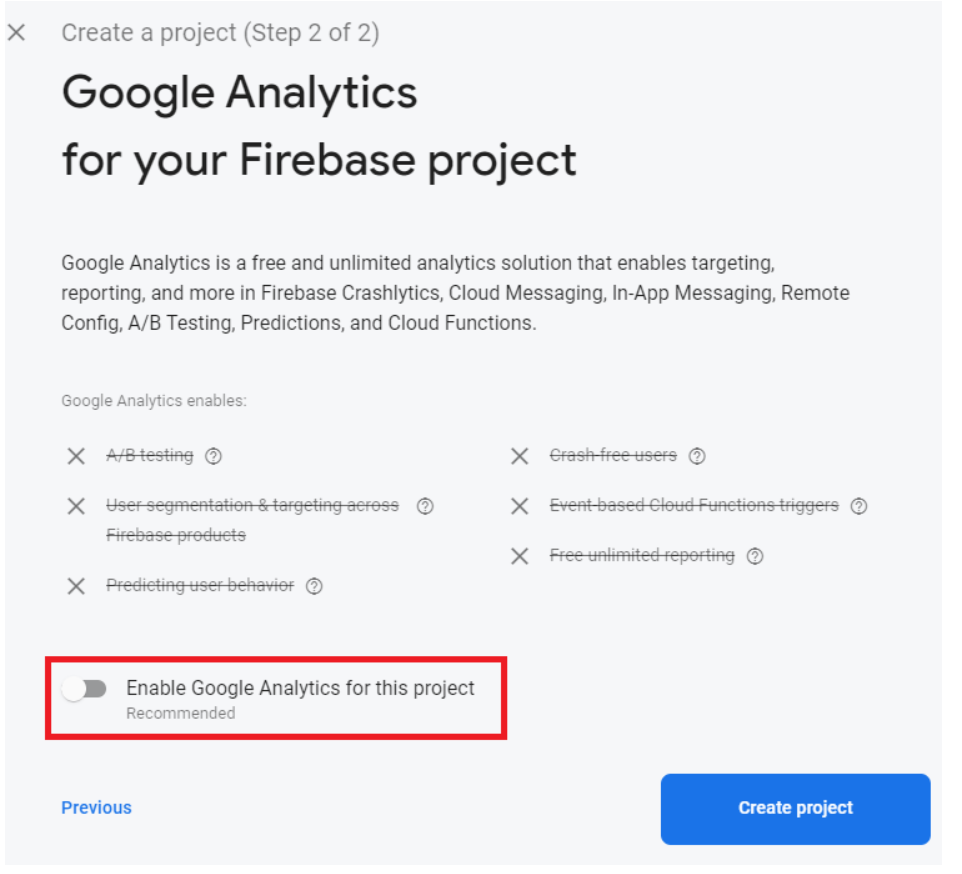


**2)** Click Get Started and then **Add project** to create a new project.



**3)** Give a name to this project, for example, Health Monitoring System.



**4)** Disable the option Enable Google Analytics for this project as it is not needed and click **Create project**. 

**5)** It will take a few seconds to set up this project. Then, click **Continue** when it’s ready.

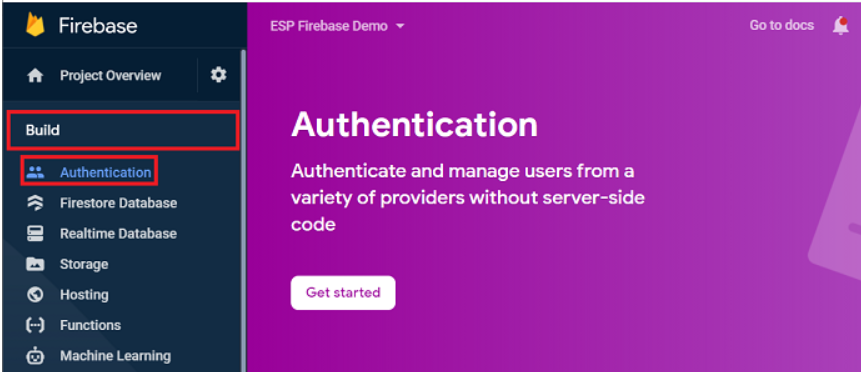
**6)** This’ll be redirected to this Project console page.

## 2) Set Authentication Methods

To allow authentication with email and password, first, you need to set authentication methods for your app.

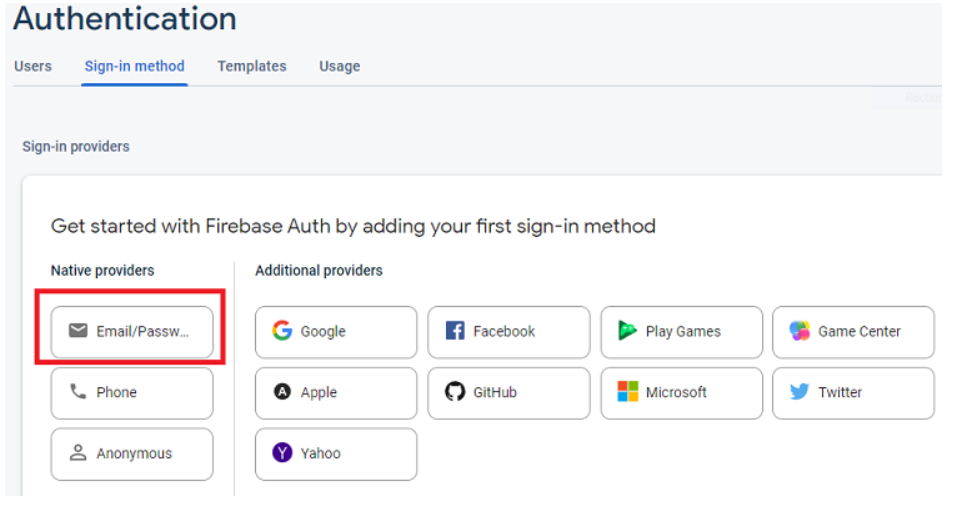
“Most apps need to know the identity of a user. In other words, it takes care of logging in and identifying the users (in this case, the ESP8266). Knowing a user’s identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user’s devices.” To learn more about the authentication methods, you can [read the documentation](https://firebase.google.com/docs/auth).

1. On the left sidebar, click on **Authentication**and then on **Get started**.

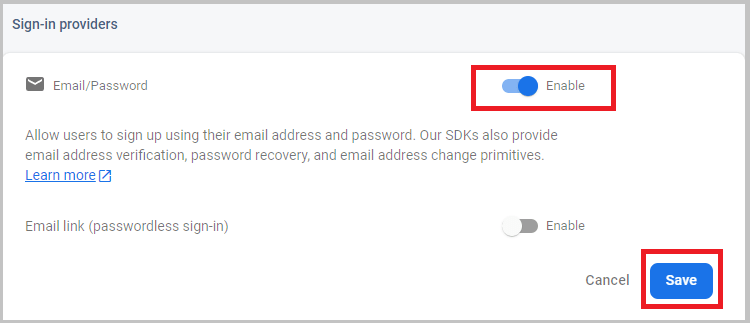


1. Select the Option **Email/Password**.

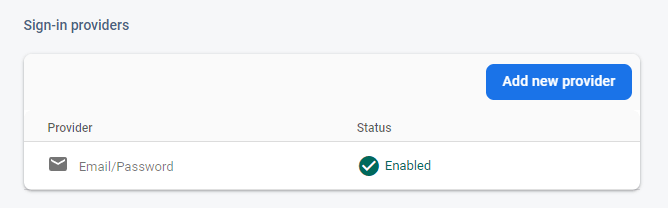
We used others authentication process but in this project we used only Email/Password authentication



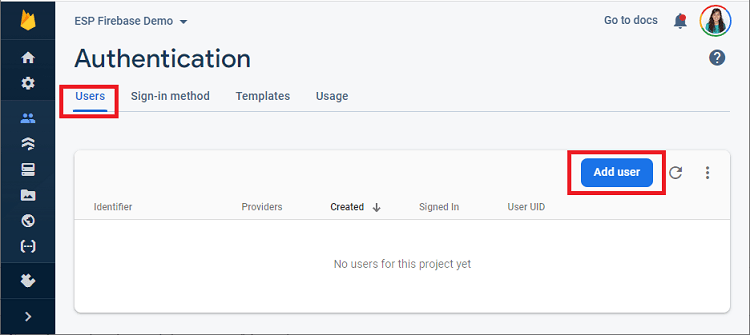
3.Enable that authentication method and click **Save**.



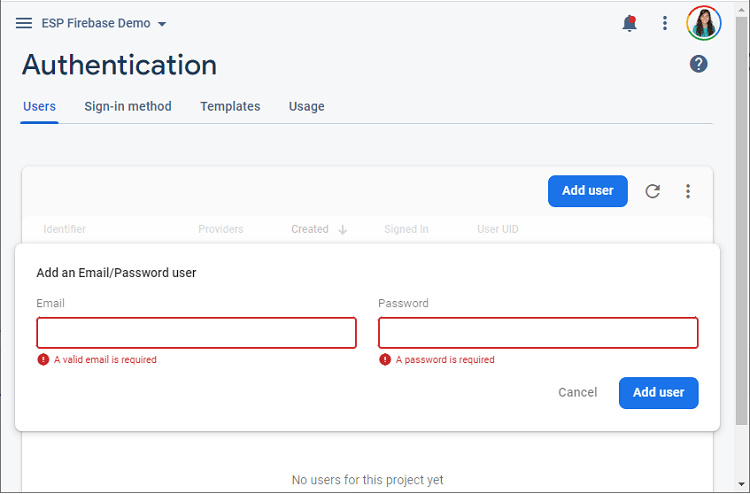
**4)** The authentication with email and password should now be enabled.



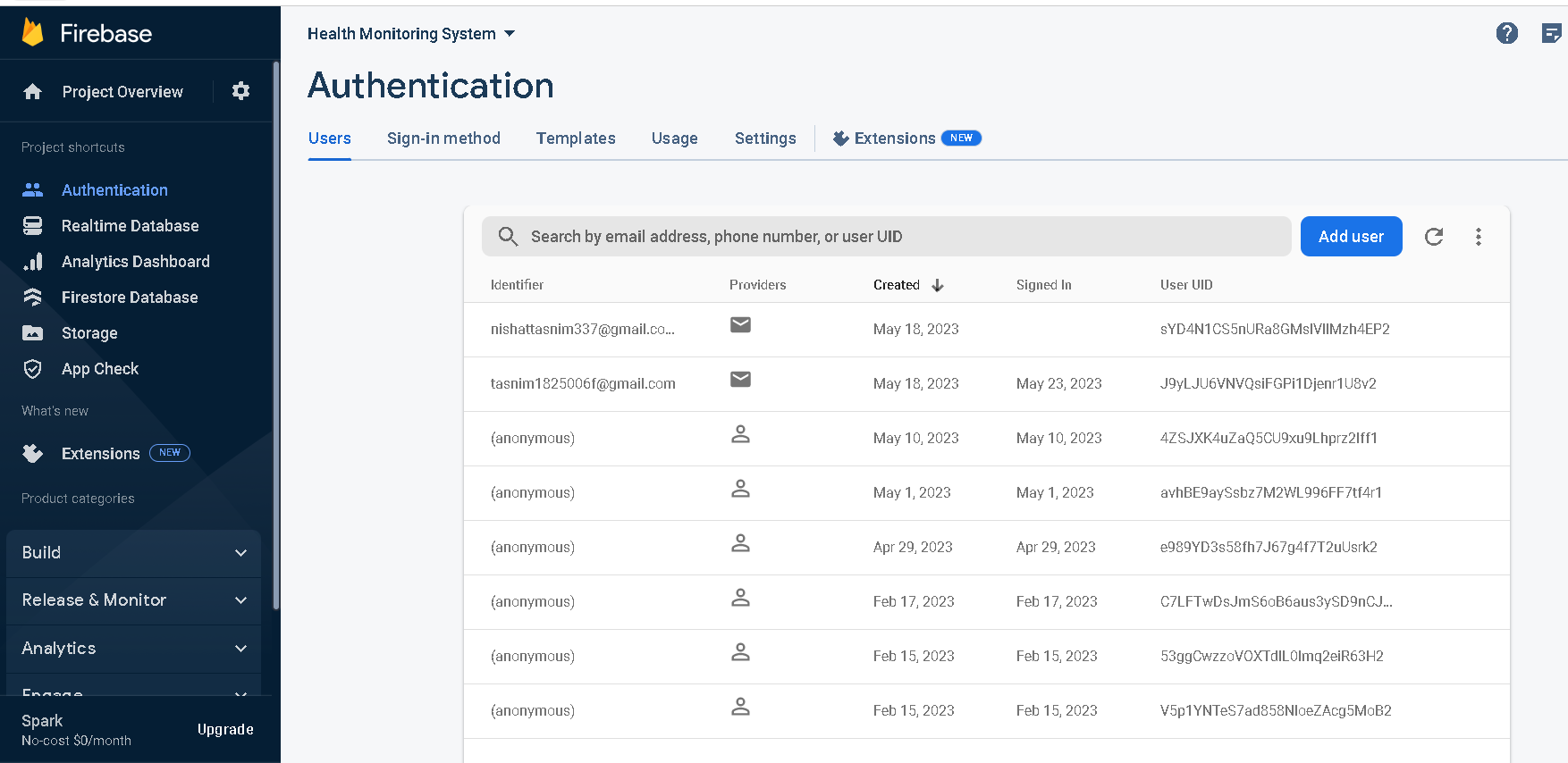
**5)** Now, you need to add a user. On the **Authentication**tab, select the **Users**tab at the top. Then, click on **Add User**.



**6)** Add an email address for the authorized user. It can be your google account email or any other email. You can also create an email for this specific project. Add a password that will allow you to sign in to your app and access the database. Don’t forget to save the password in a safe place because you’ll need it later. When you’re done, click **Add user**.



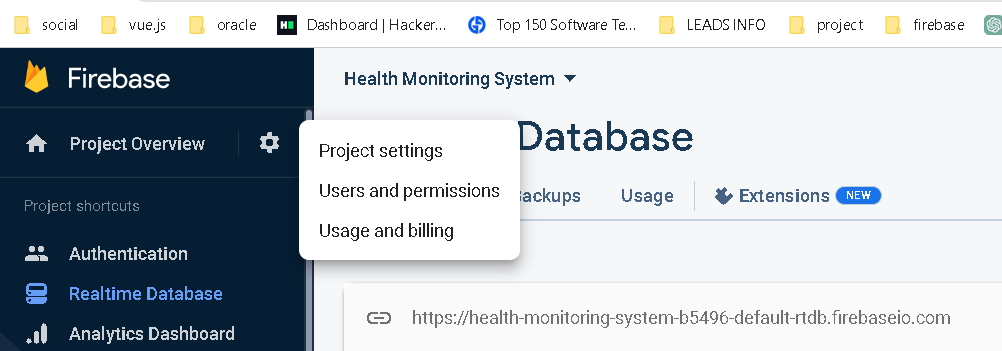
**7)** A new user was successfully created and added to the **Users** table.



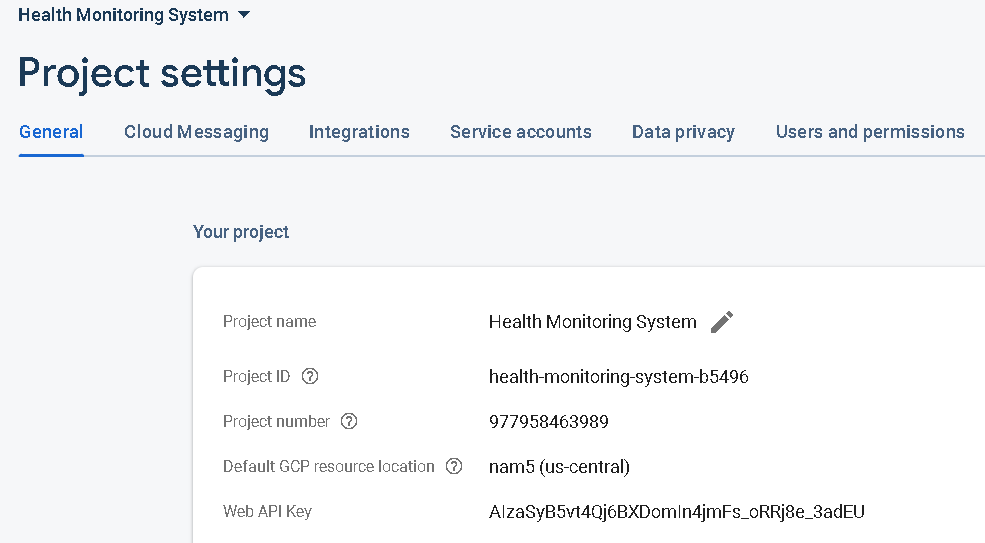
## Get Project API Key

To interface with your Firebase project using the ESP8266 board, you need to get your project API key. Follow the next steps to get your project API key.

**1)** On the left sidebar, click on **Project Settings**.

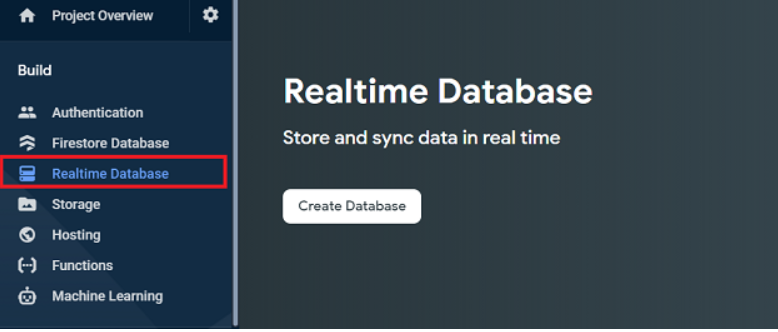


**2)** Copy the Web API Key to a safe place because you’ll need it later.

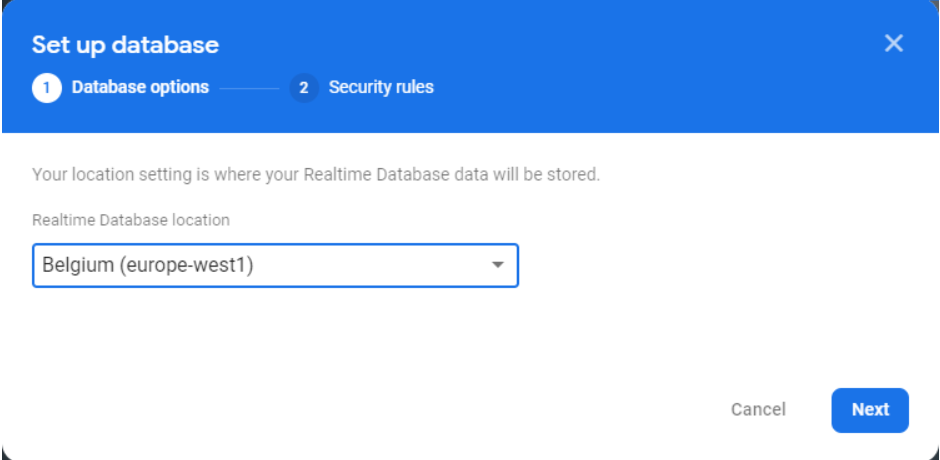


## 4) Set up Realtime Database

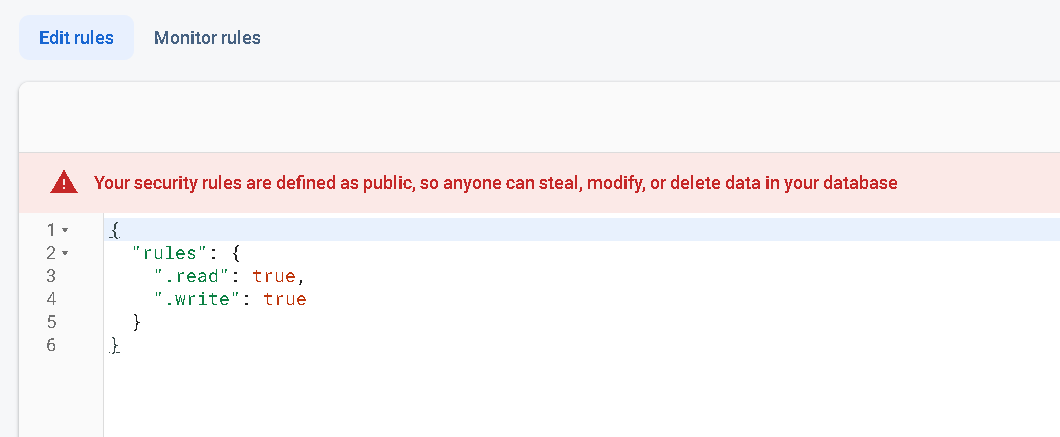
**1)** On the left sidebar, click on **Realtime Database** and then click on **Create Database**.

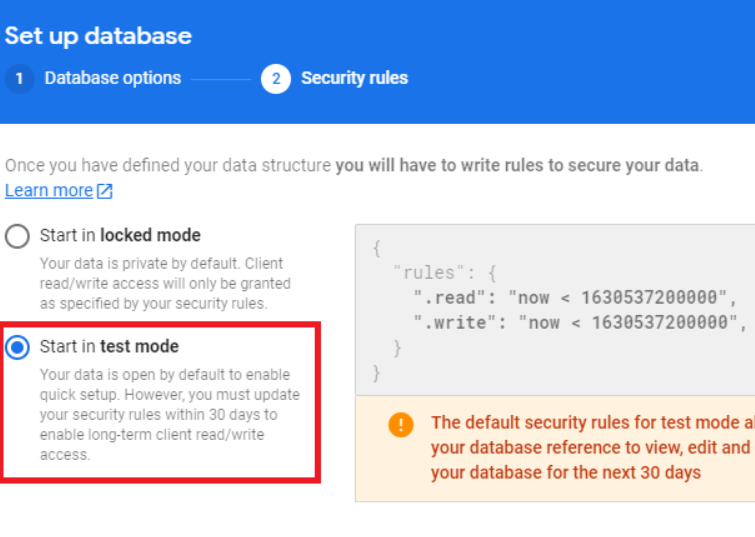


**2)** Select your database location. It should be the closest to your location.

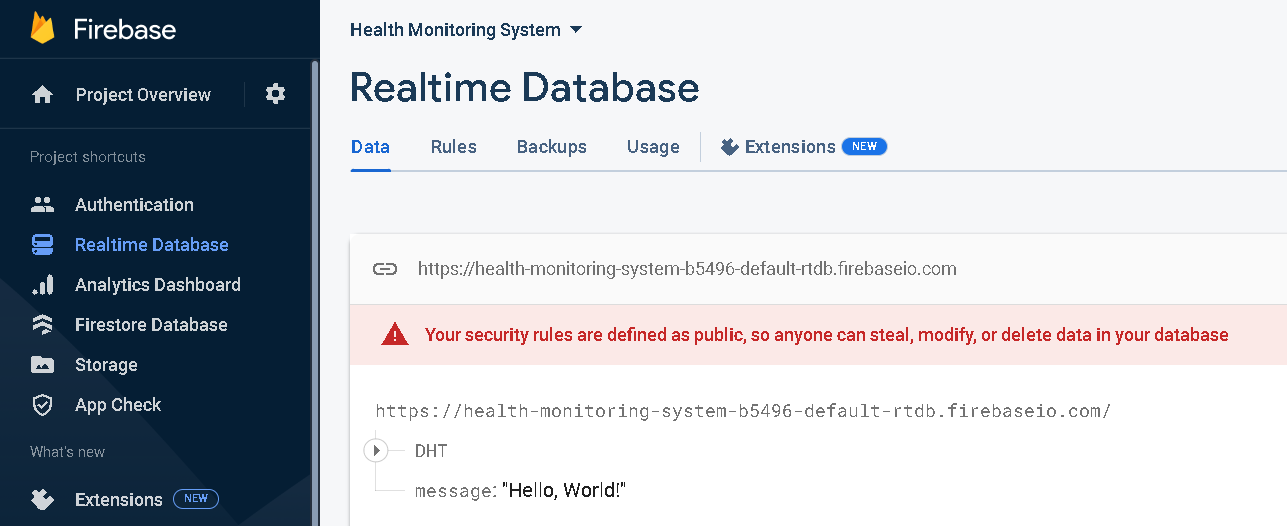


1. Set up security rules for your database. You can select **Start in test mode**. We’ll change the database rules in just a moment.



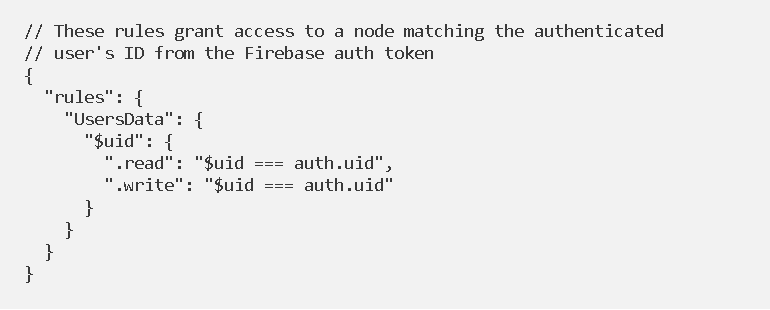


1. Your database is now created. You need to copy and save the database URL—highlighted in the following image—because you’ll need it later in your ESP8266 code.



## 5) Set up Database Security Rules

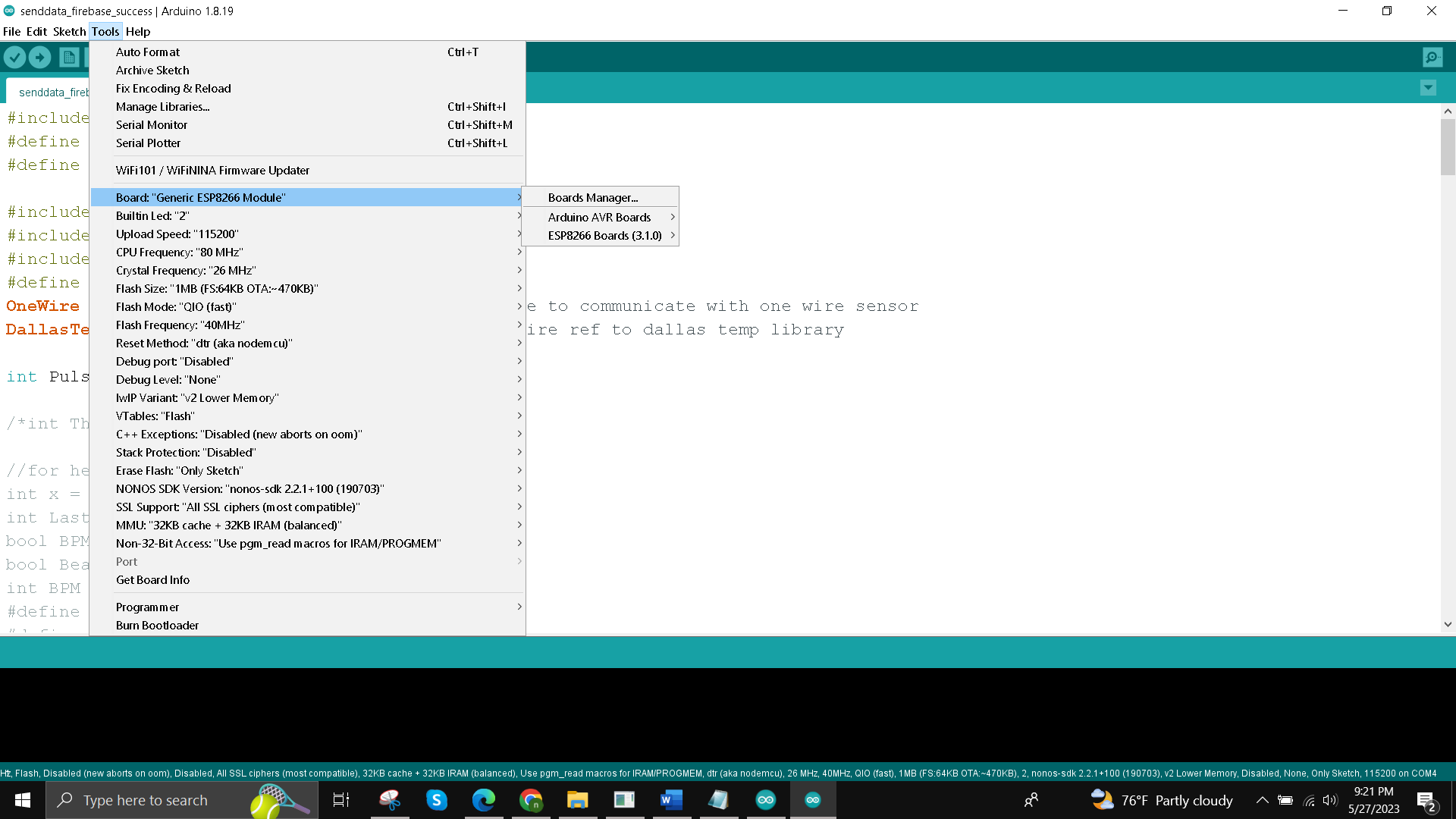
Now, let’s set up the database rules. On the **Realtime Database** tab, select the **Rules** tab at the top. Then, click on **Edit rules**, copy the following rules and then click **Publish**.



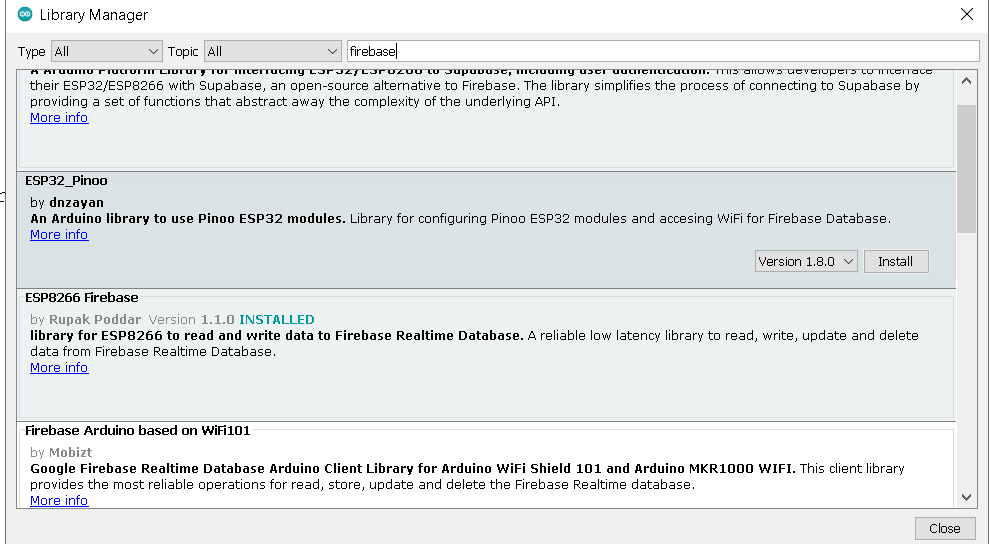
**Software Setup:**

For sending sensor readings to the Firebase Realtime Database. So, you need to wire the BME280 sensor to your board.

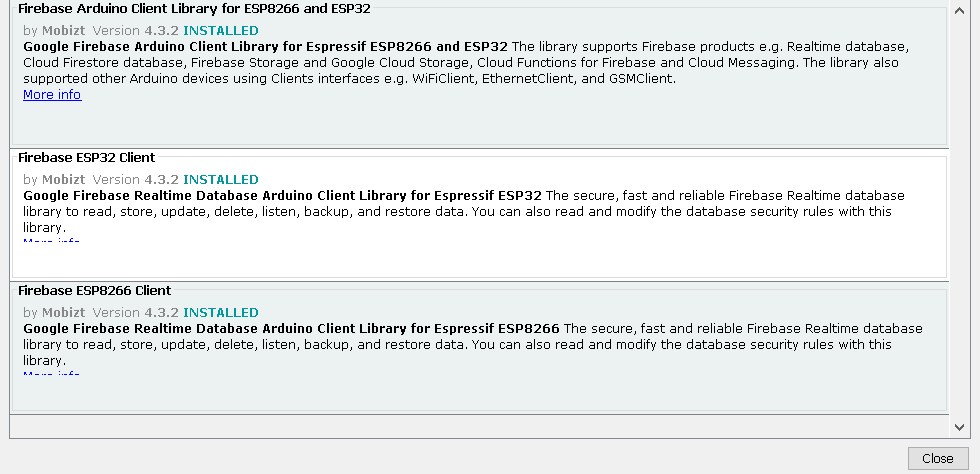
### Installing Libraries



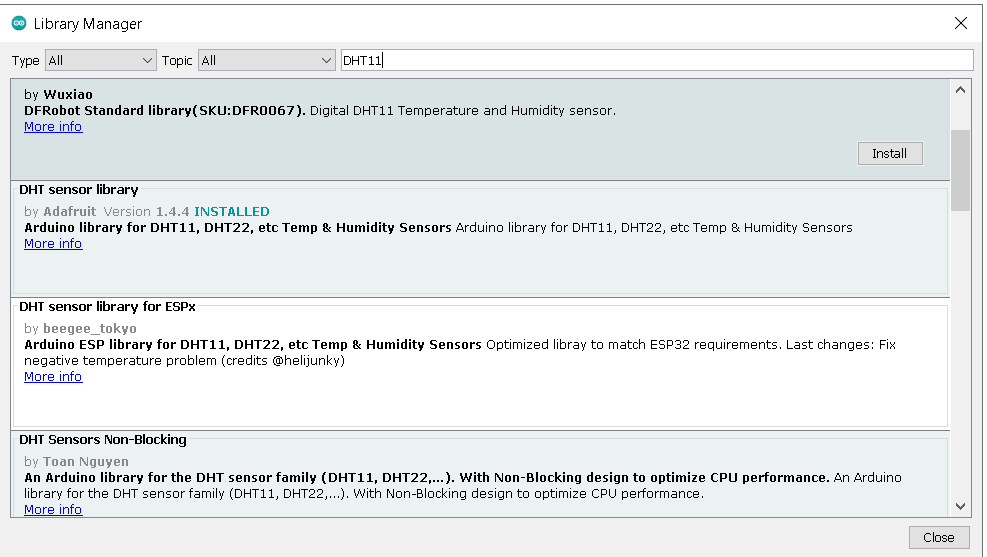
**Install ESP8266 Firebase for sending data to firebase**



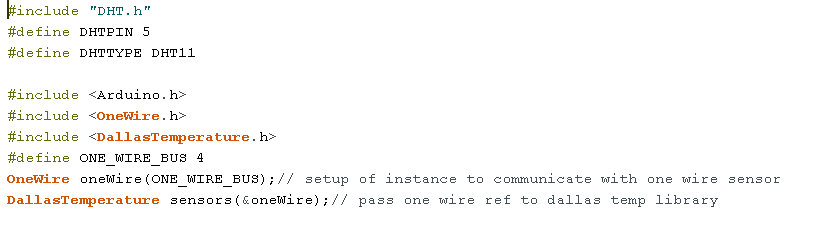
##### **Install the Firebase-ESP-Client Library**

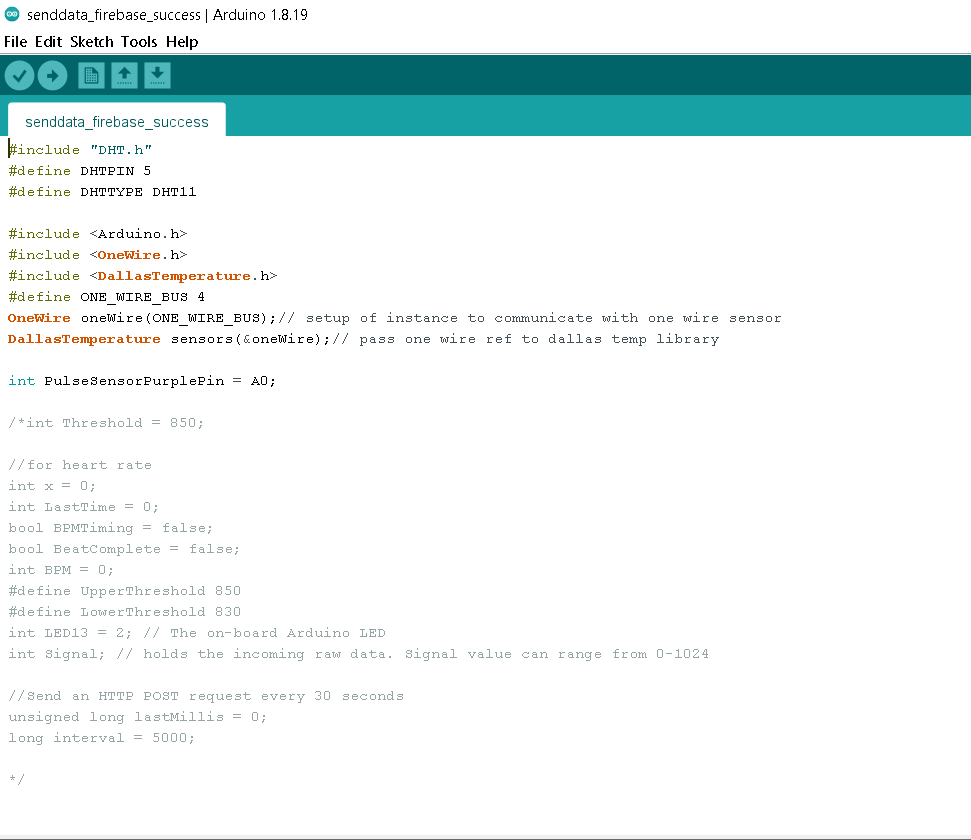


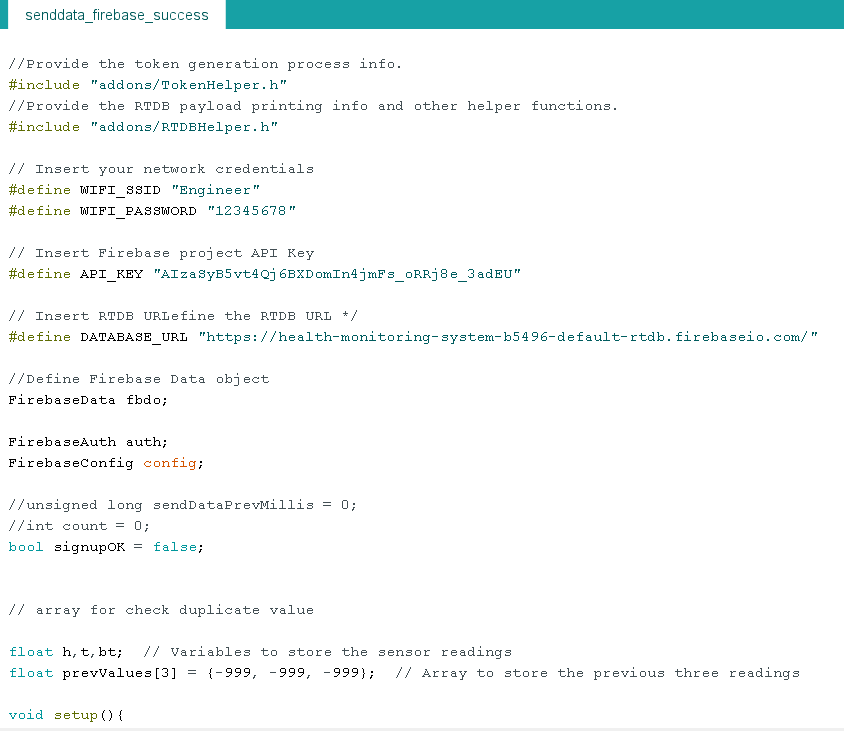
Install DHT11 Library

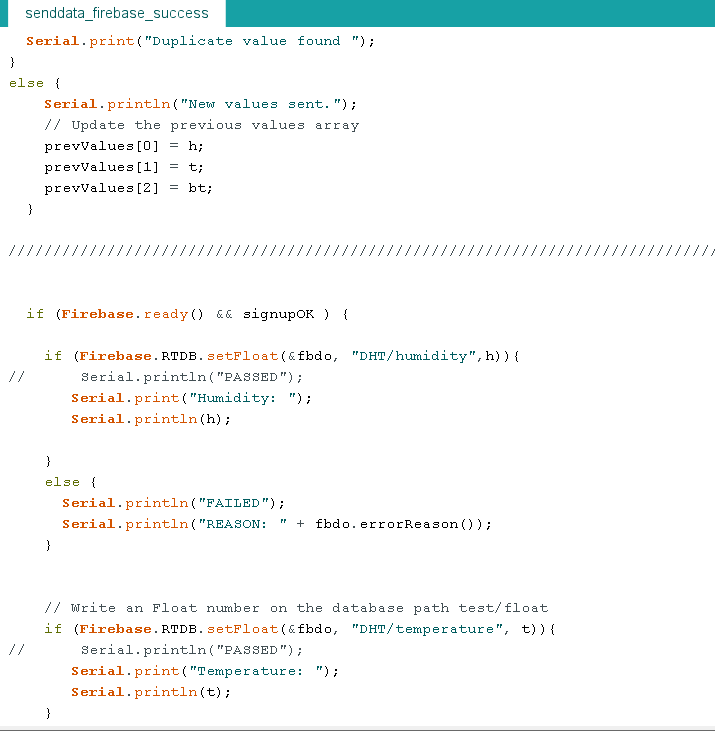


Library needed for this project

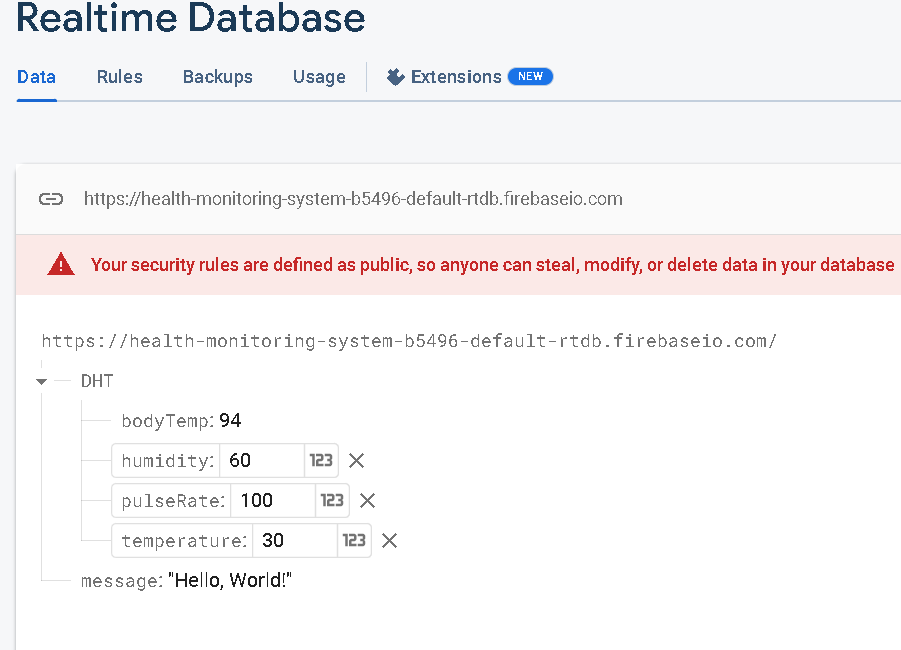








After that Data send to firebase successfully



Mobile application:

Overview: The mobile application for IoT health monitoring system utilizing the Firebase Realtime Database and sensors such as body temperature, pulse rate, humidity, and room temperature brings together the power of Internet of Things (IoT) technology and mobile devices to enable convenient and real-time health monitoring. This application serves as a user-friendly interface, allowing individuals to monitor their vital health parameters and receive insights and alerts based on the data collected from the sensors.

Key Features of the Mobile Application which name is iot\_app for health monitoring system.:

User login with Firebase Authentication: The application includes a user login with firebase authentication system to ensure secure access to personal health data. Registered user can log in using their credentials to access their personalized health monitoring dashboard.

Sensor Integration: The mobile application connects with the sensors measuring body temperature, pulse rate, humidity, and room temperature. Through wireless technologies, the application retrieves data from these sensors in real-time, providing up-to-date information for health monitoring which store in firebase Realtime database.

Real-time Data Display: Users can view their current body temperature, pulse rate, humidity level, and room temperature at a glance, allowing them to track their health status conveniently through the mobile app from firebase server.

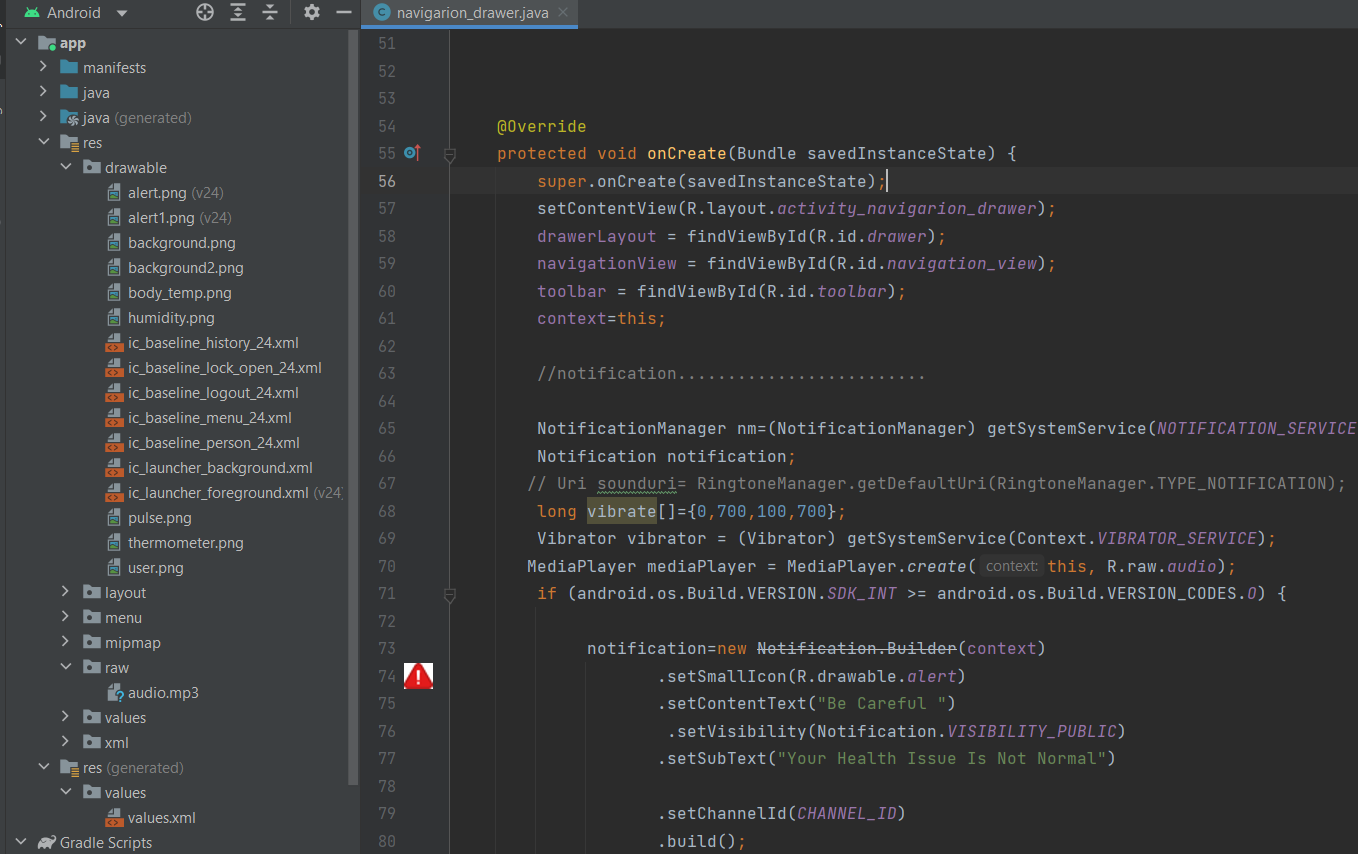
Customized Thresholds and Alerts: Users can set personalized thresholds for each parameter to define acceptable ranges. When the sensor data exceeds or falls below these thresholds, the application generates alerts and notifications with sound and vibration, prompting users to take appropriate actions or seek medical attention.

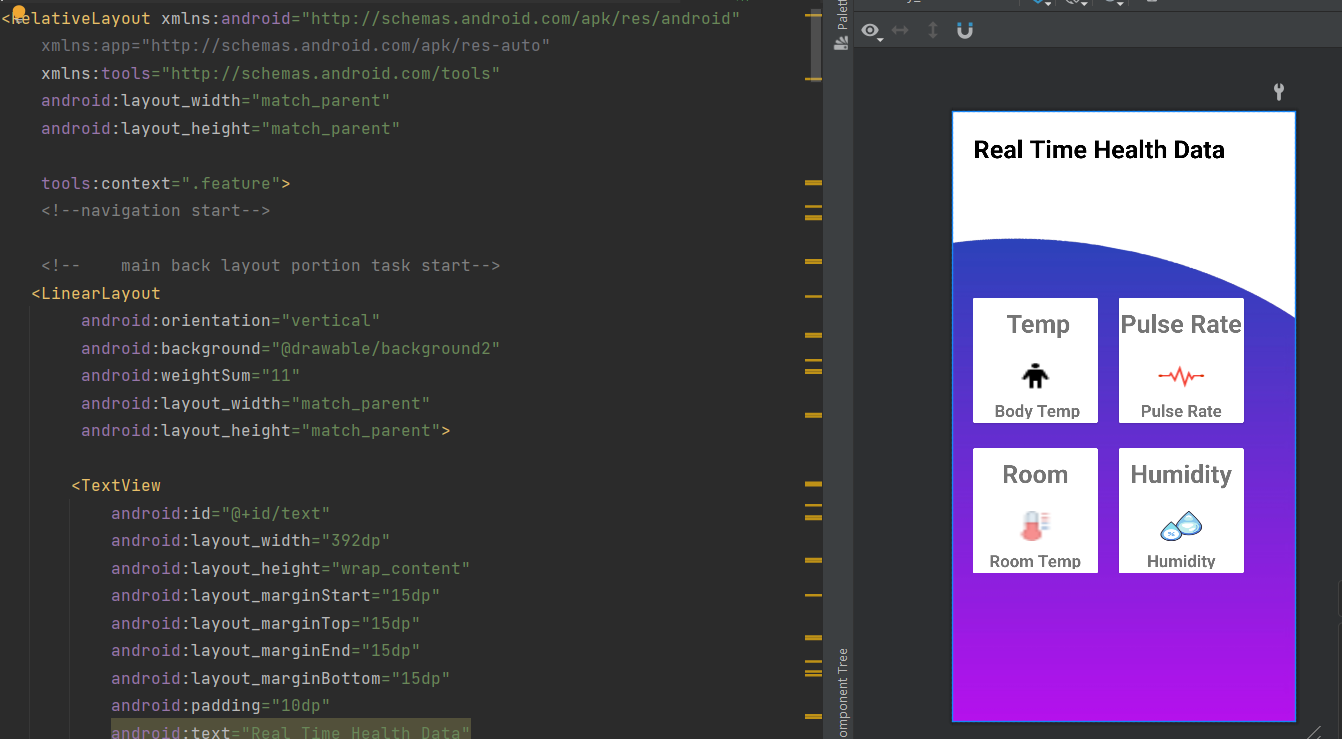
Cloud-based Storage and Synchronization: The Firebase Realtime Database serves as the centralized storage solution for the collected health data. It ensures that the data is securely stored in the cloud and synchronized across multiple devices, allowing users to access their information seamlessly from anywhere.

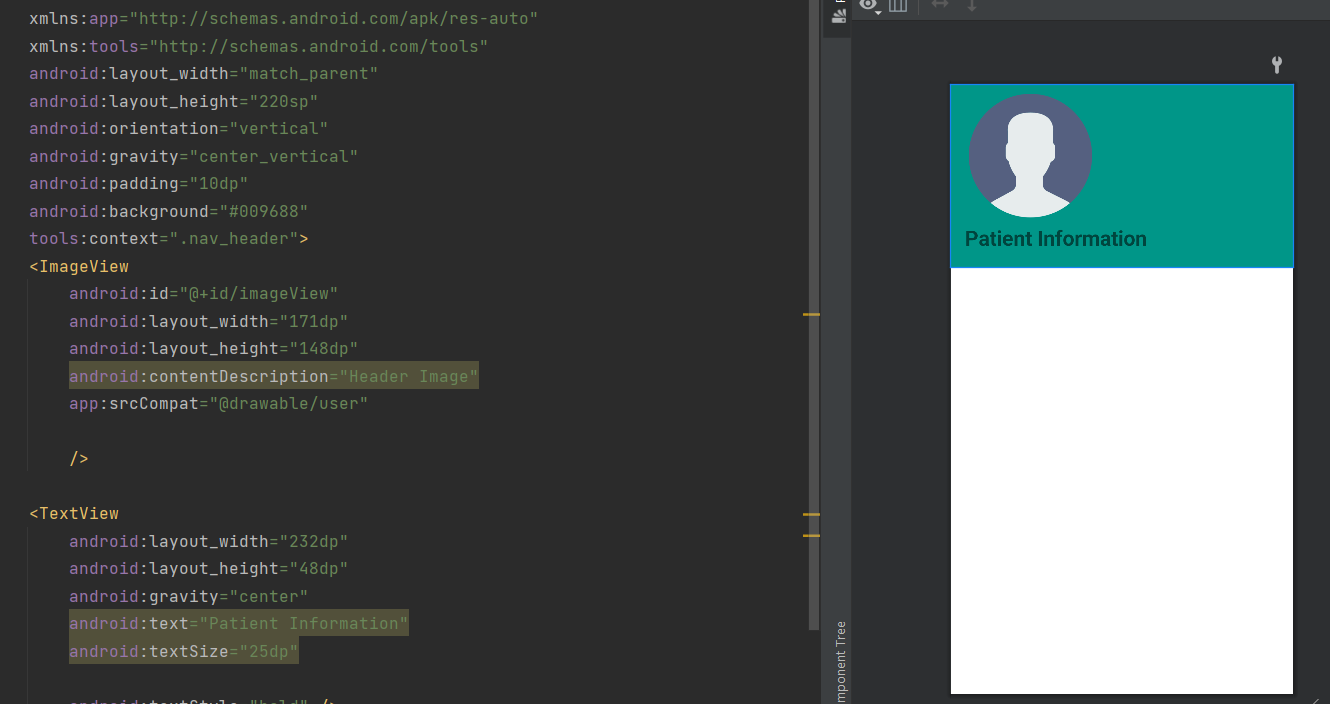
User Engagement and Education: The application can incorporate educational resources, tips, and insights related to health and well-being. This feature promotes user engagement, encourages healthy lifestyle choices, and provides valuable information for users to make informed decisions about their health.

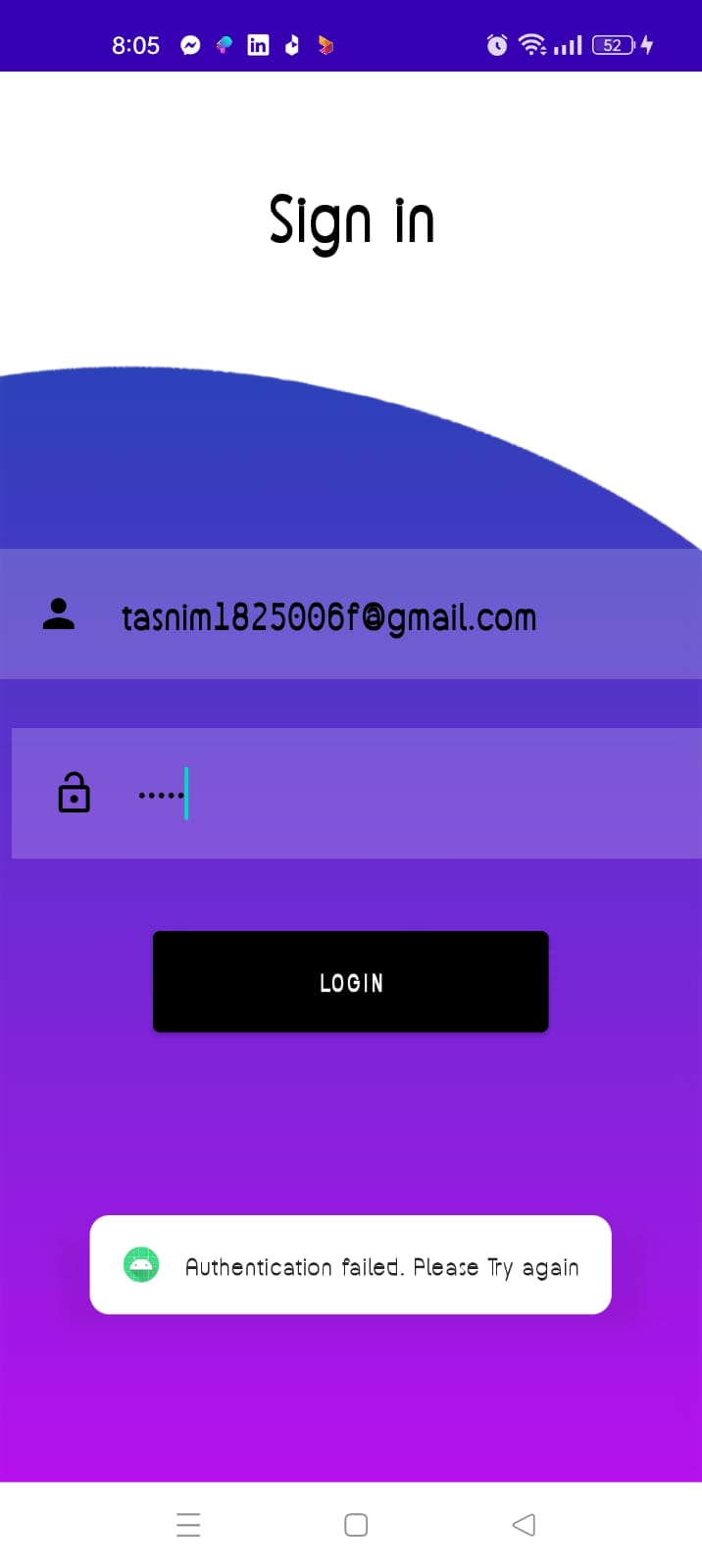
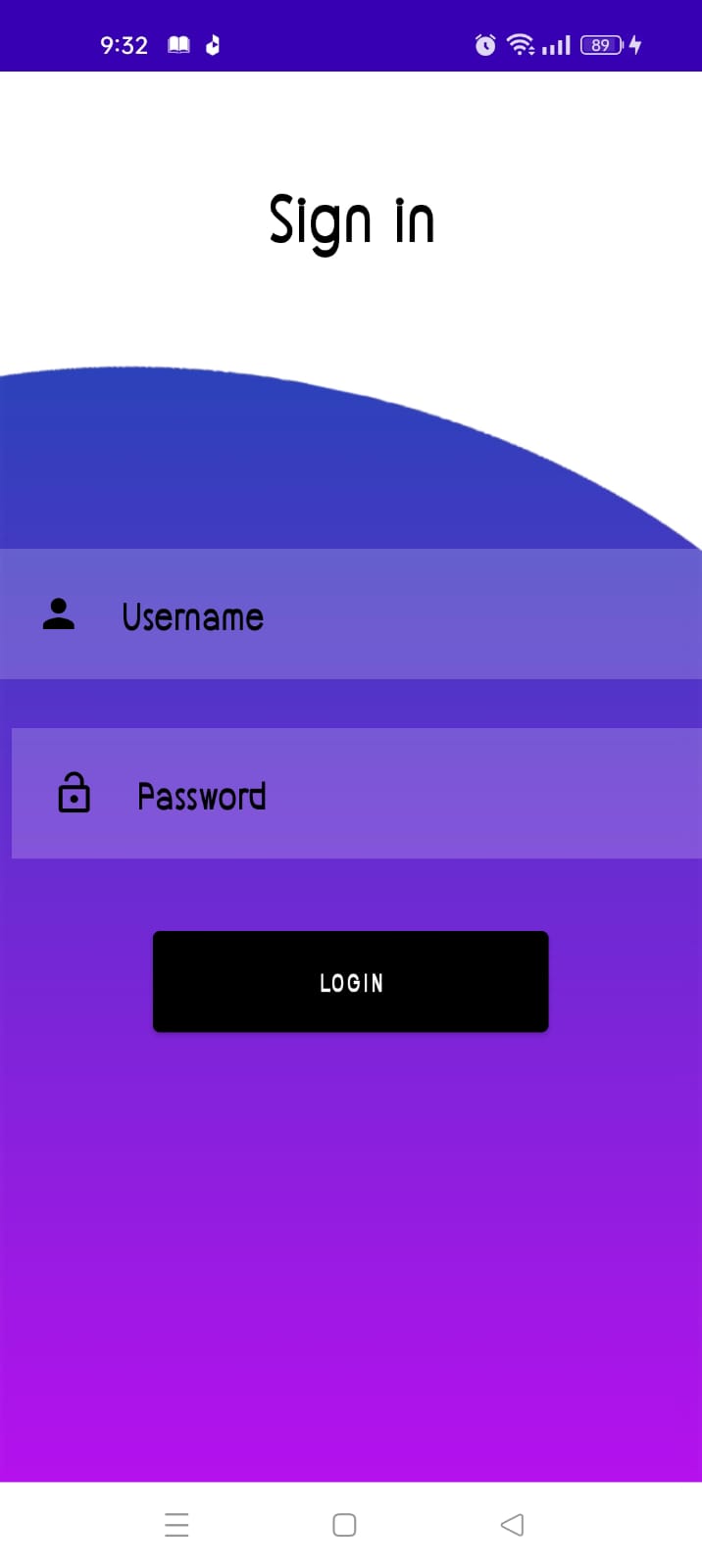
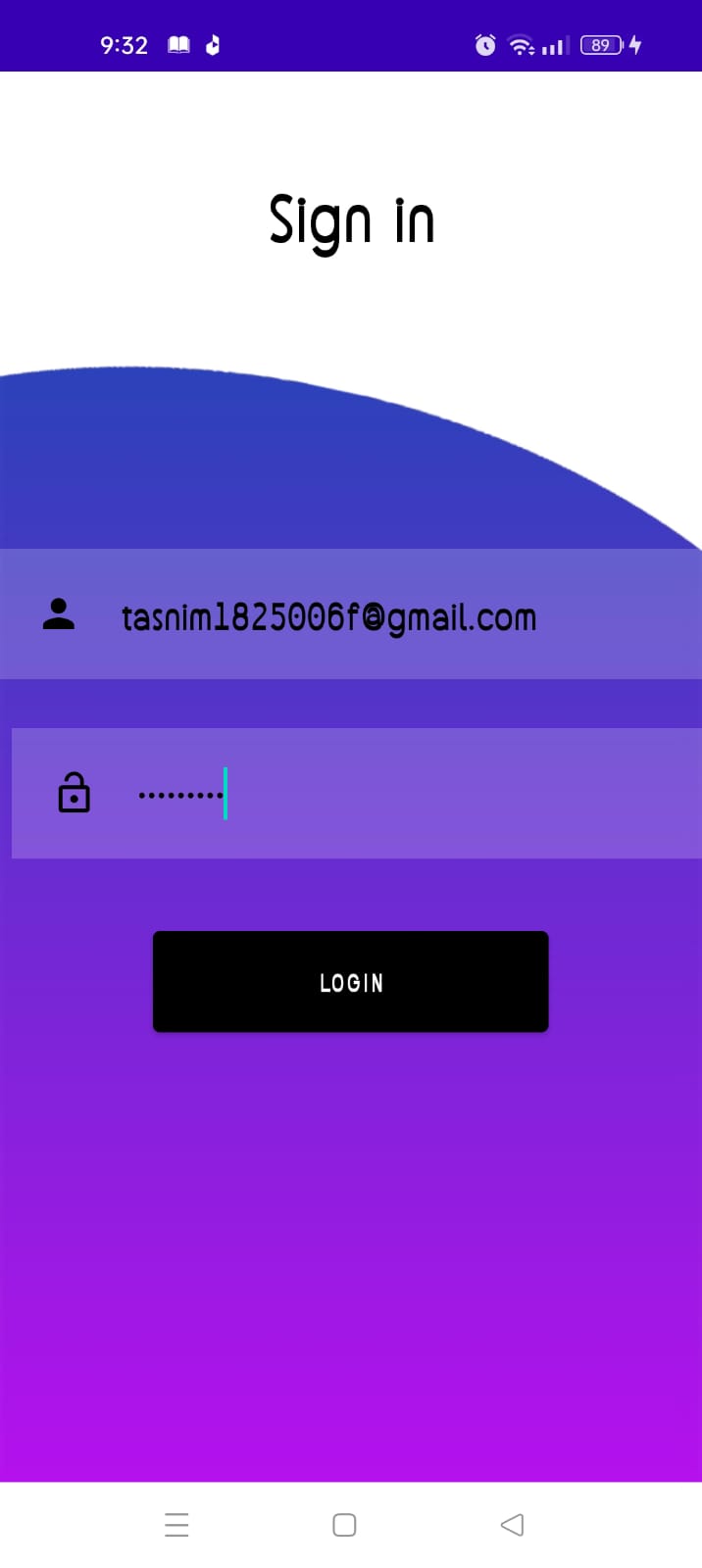
Implementation and user interface of mobile application:

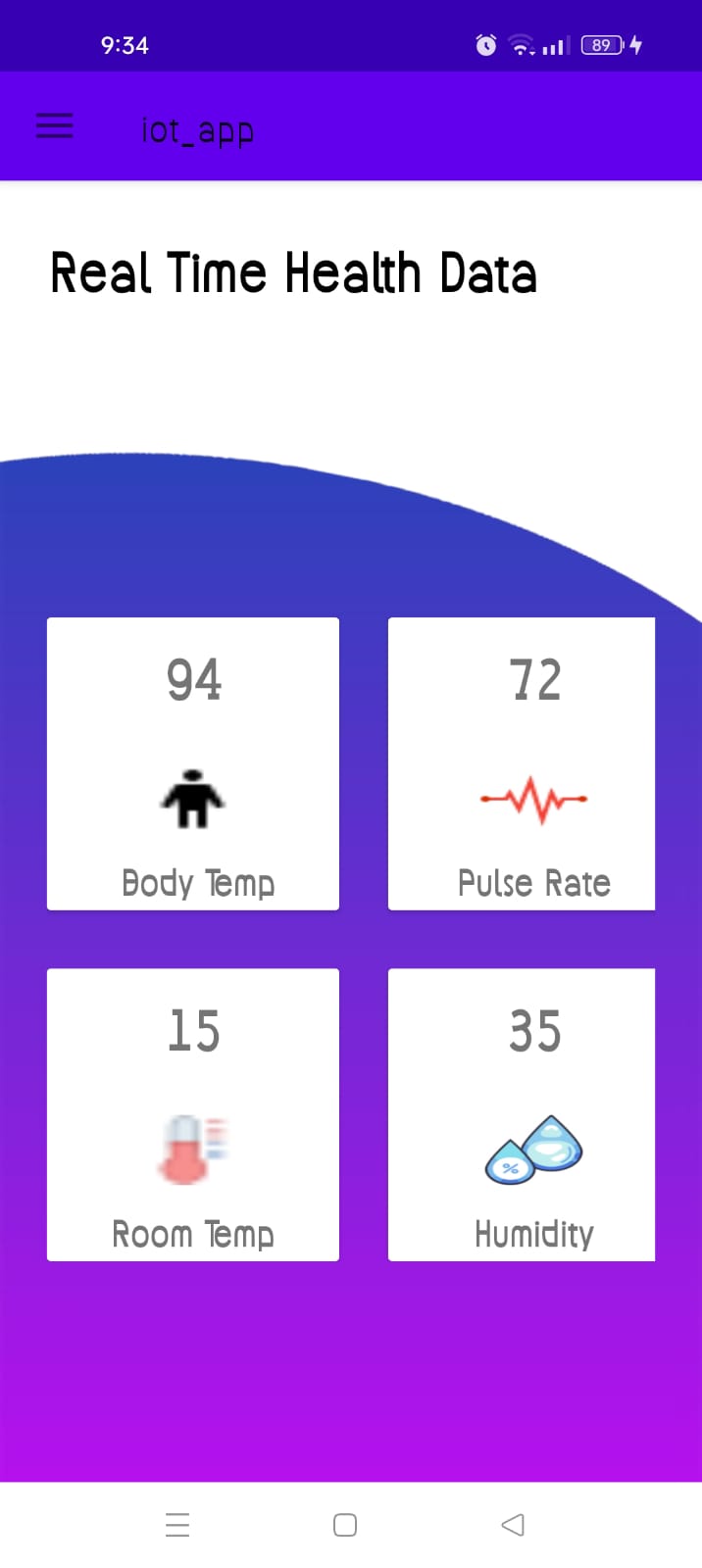
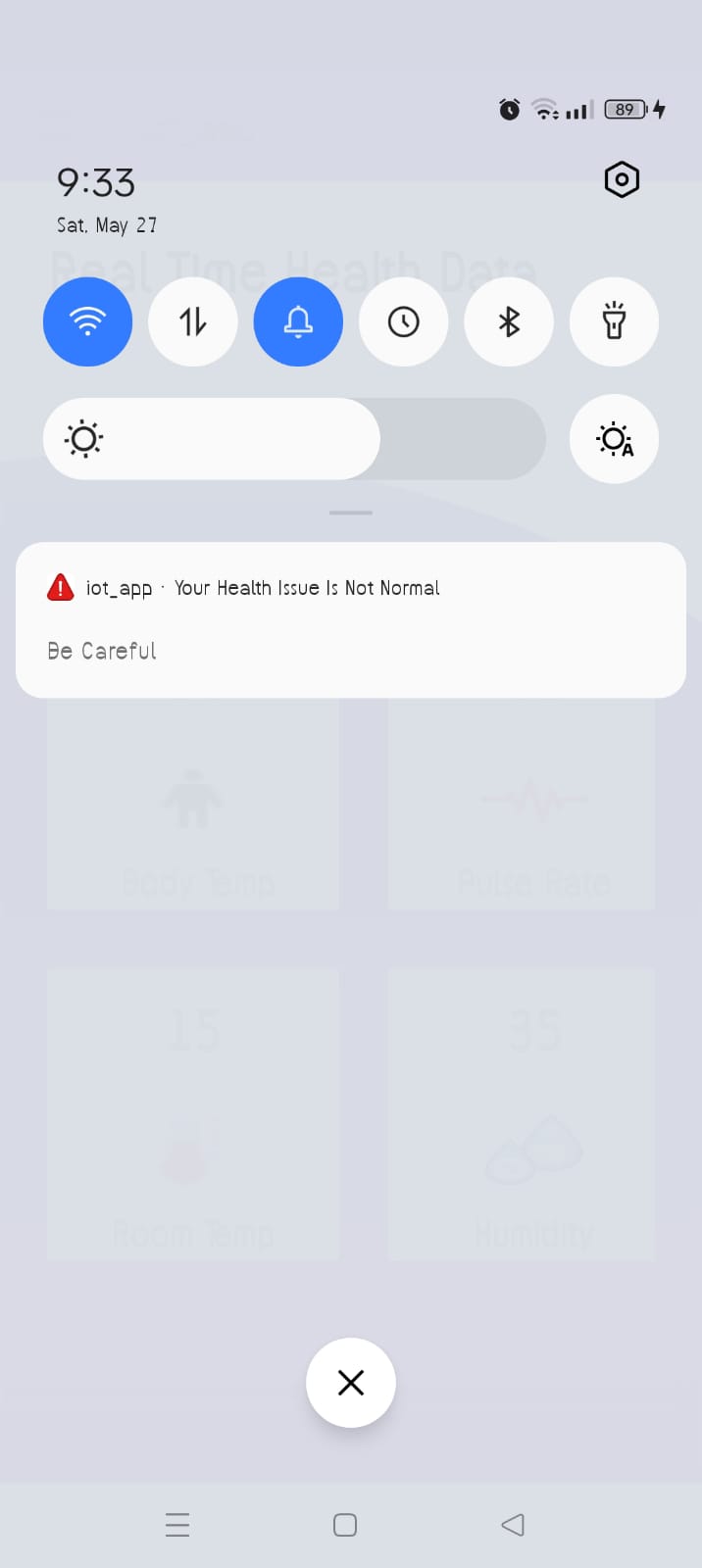
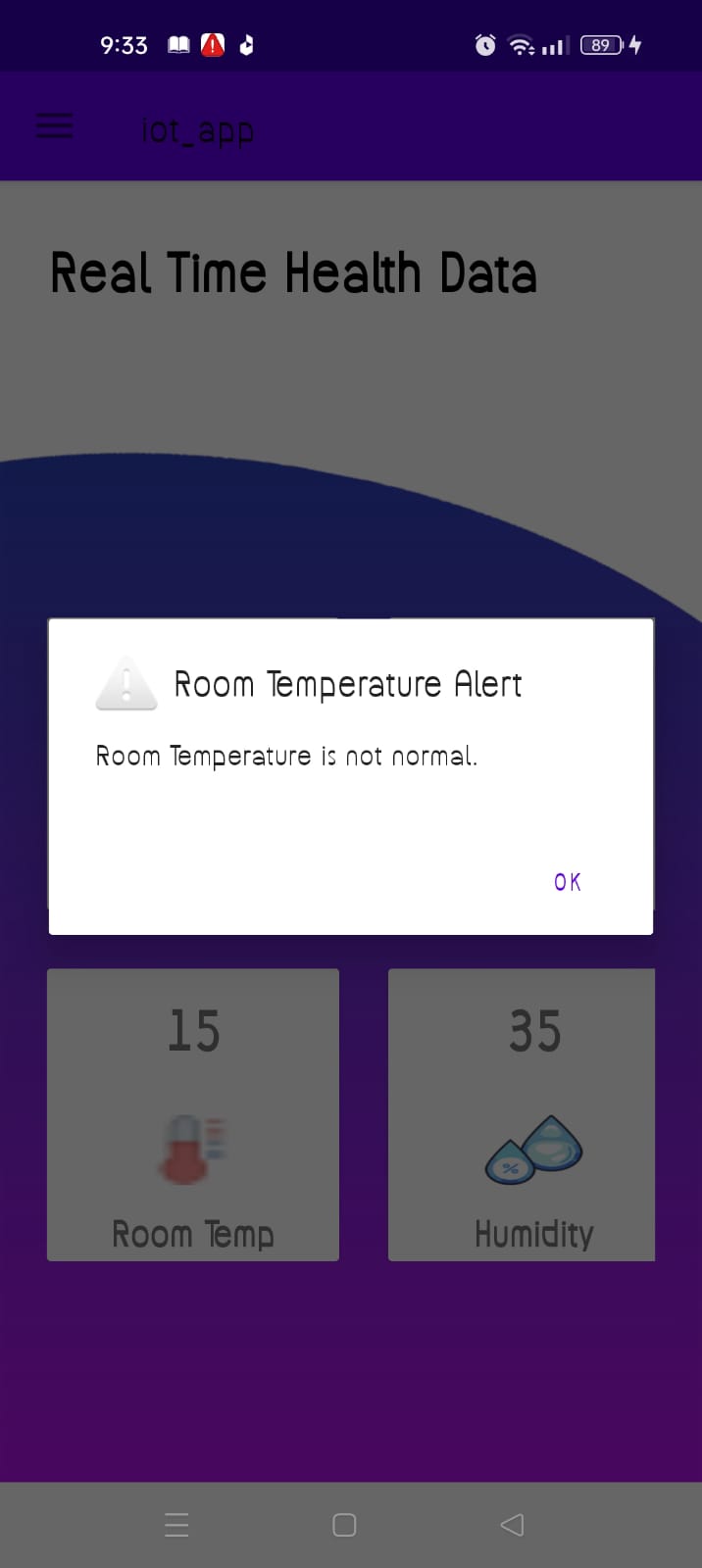
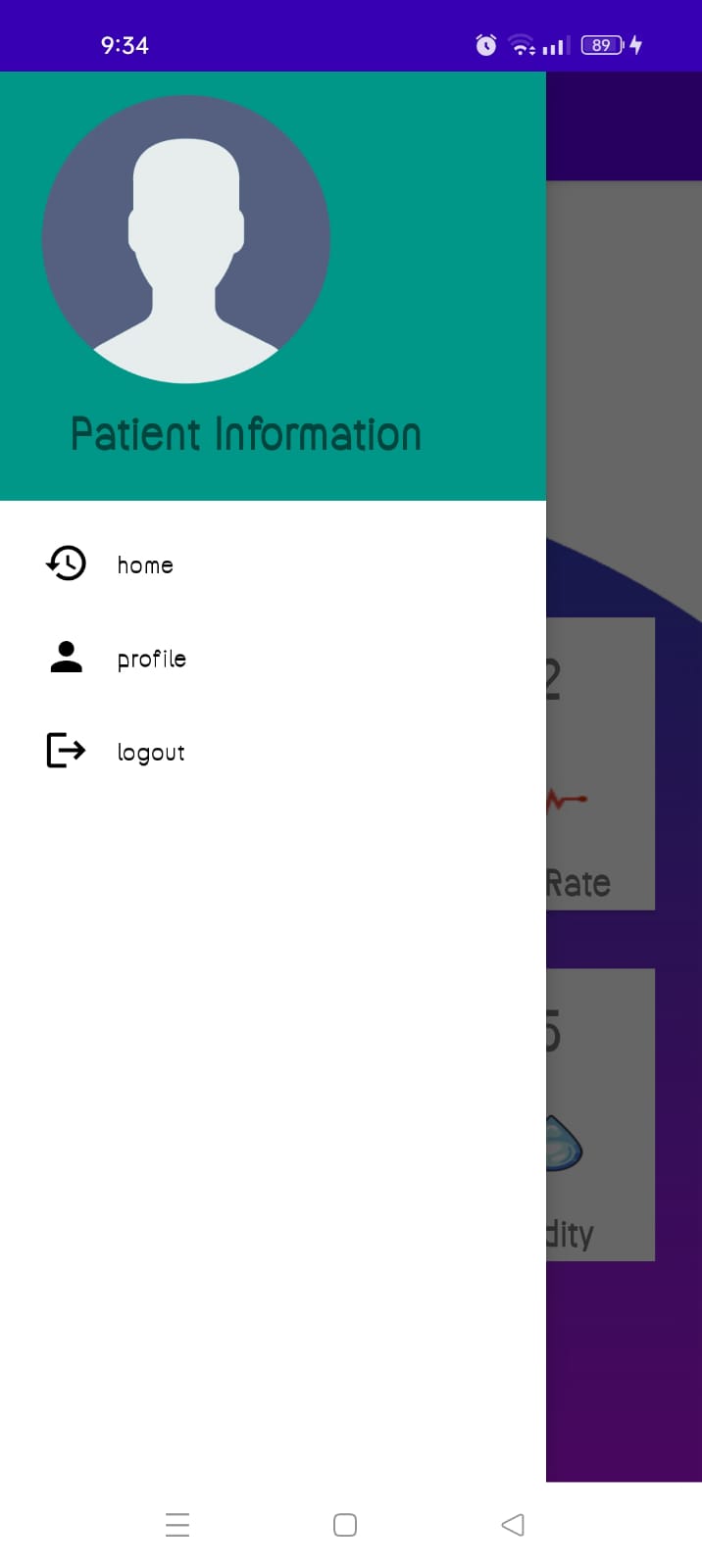
**Only authenticate user login into using patient credentials.**





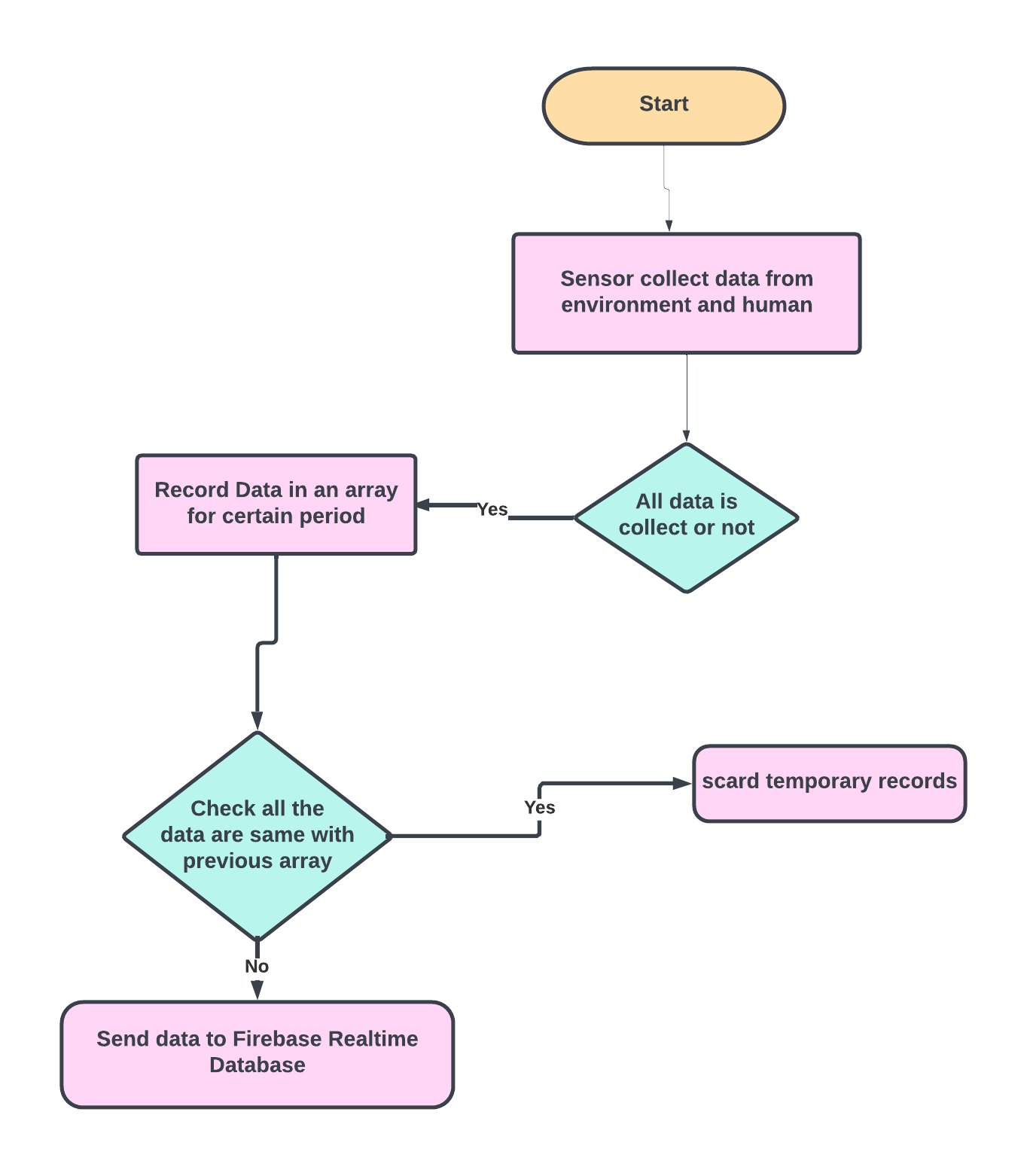


** **

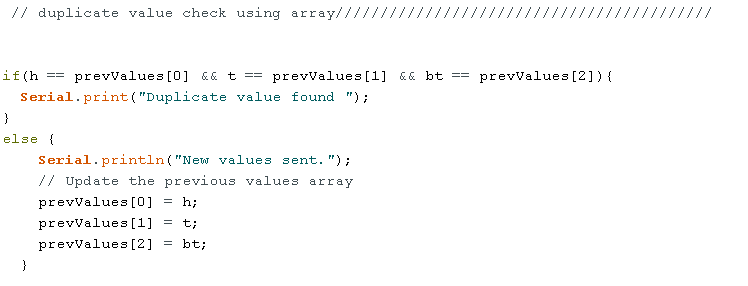
**  **

**Data Optimization**: Remove redundancy of sensor for sending send firebase database. Data optimization refers to the process of improving the efficiency, performance, and storage requirements of data. It involves various techniques and strategies to make data more compact, accessible, and usable. Power saving is a critical issue in wireless sensor networks (WSNs) since sensor nodes are powered by batteries, so its backup is very high when it doesn’t send redundant data to server like Firebase Realtime Database server. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three important cost metrics: area, performance, and power dissipation of the resulting implementation.

Data Deduplication: Deduplication eliminates duplicate data by identifying and removing redundant copies. This is commonly used in storage systems, backup solutions, and file-sharing platforms to reduce storage costs and optimize data transfer. In this project we used this technique for remove redundant data. After filtered data then send it to firebase Realtime database.

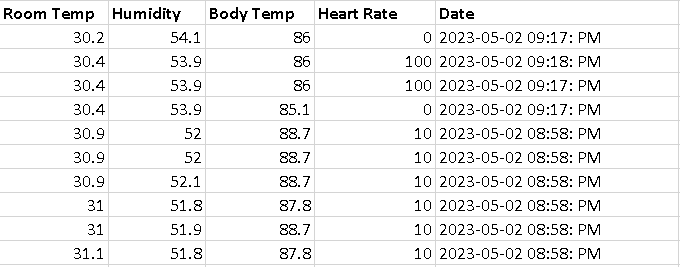
For Data Optimization used filtered in Arduino IDE, so sender filtered data first and then send to receiver. First check all data are match with previous data if match then these data doesn’t send to firebase Realtime database. If not then send data to firebase Realtime Database. 

Check duplicity of sensor data in Arduino IDE

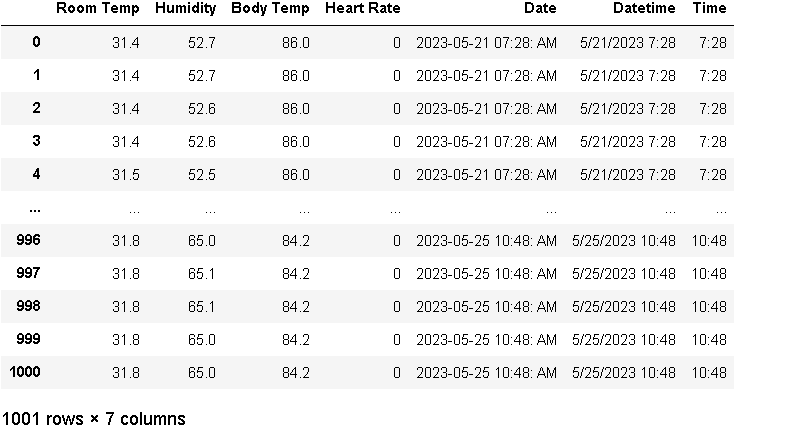


First follow some step to remove redundant data like: First send all data to localhost. Then collect all data analysis it after remove redundancy how must data to stop to send firebase from Arduino IDE. For analyzing this data use Jupyter IDE, and using Python language and some libraries like Pandas, numpy, matplotlib, and datetime.

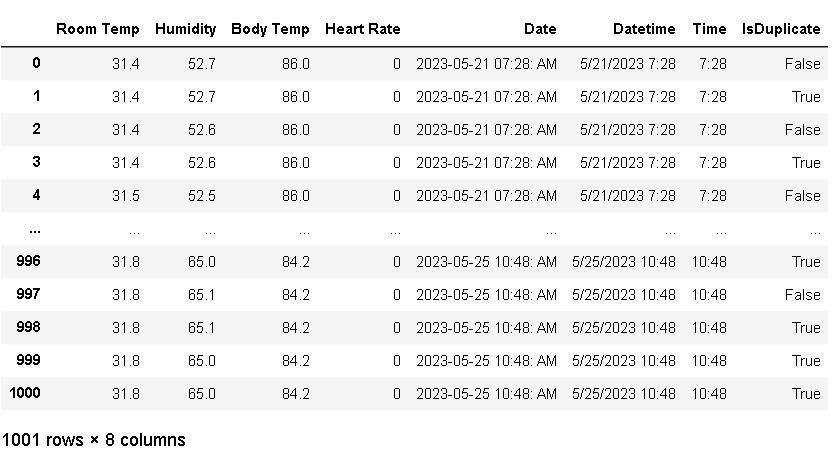
Collect data:



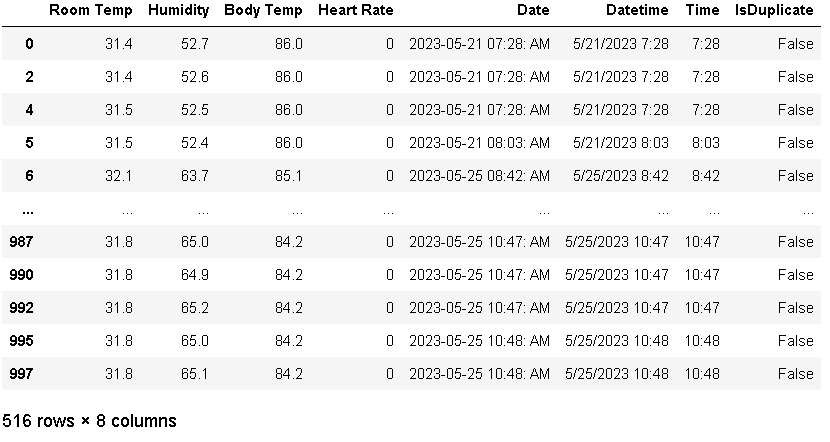
Then Feature extract from Database:



After this find out the duplicate row from the Database:



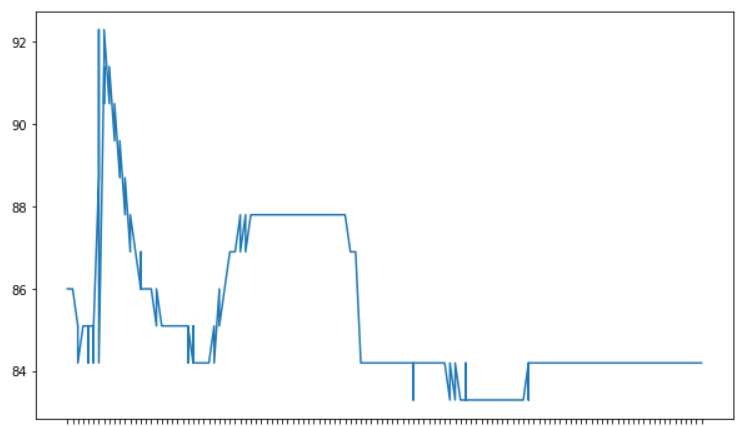
After that remove all duplicate row then ceate new Dataframe.



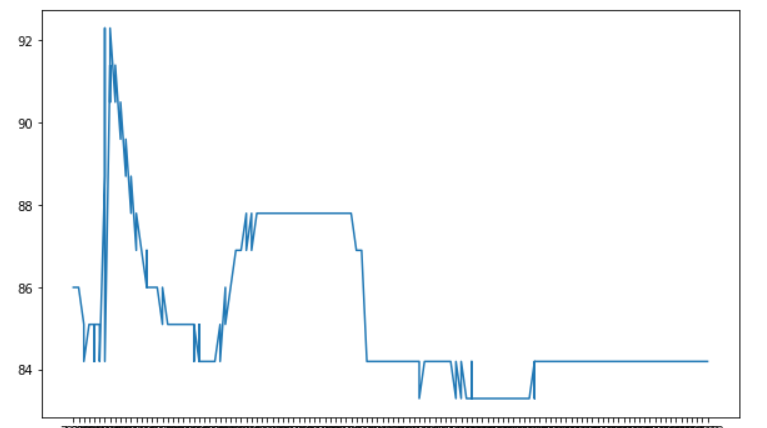
After removing duplicate Data row then new Dataframe contain only 516 which is almost 50% data remove from first created Dataframe.

Show some figure before and after redundant data remove:

Before:

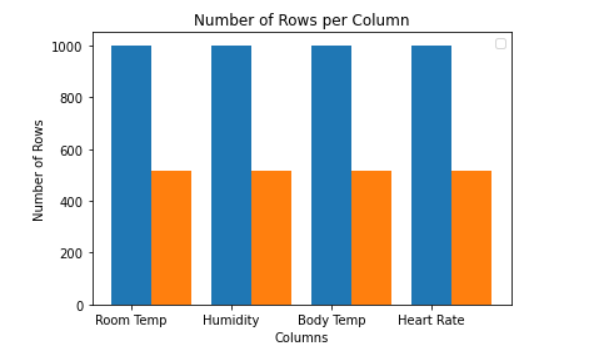


After remove redundancy:



Result: after removing redundant data : Before remove redundancy sensor send 1001 data to server but after remove redundancy sensor send data only 516 .

After remove redundant data it easy to send , storage optimization , use efficient energy to send data.



DISCUSSION:

1. Significance of the work

1) Storage Optimization: For increasing volume of sensor data, storage capability is one of the major concern. By our scheme we are able to reduce the storage requirement of IoT numerical data by removing redundancy.

2) Easy to Implement: The remove redundancy steps that developed and easier to implement. Because it is developed by using several easy computation methods.

3) Error Rate: This project sensor removes the redundant data to all the actual data to be send to firebase.

4) Energy Consumption: It is another big issue for big data. In this project we show that when sensor send all data but after removing redundant data it send data only 50% or 60 % approximate which is less from original like 45% to 50%.

B. Scope and Limitations: The primary limitation of our data approach is that it can show the Realtime data and show it to the mobile app. We cannot store Data from Realtime to Firestore for the purpose of paid cloud function. So that’s why mobile app cannot show the history of patient data.

C. Future Work : In future we plan to transfer data Firebase Realtime database to firestore and show history to mobile app. And also use some others data optimization framework in this project. Also alert patient and doctor , guardian both. Use NLP for making decision based on Realtime Database.

**CONCLUSIONS:** The project work has been studied and implemented a complete working model using an Arduino board. The programming is done on the Arduino software. This project elaborates the design and construction of patient health monitoring system. This work could be a life saving thing for many people and can be very handy for the doctors to monitor the patient health. The design and verification of a Patient Health Monitoring System was done successfully. In this paper, we found the importance and fruitful benefits of implementation of IoT in remote health monitoring systems for patient. Arduino Board send efficient Data to firebase Realtime database. Overall Arduino Board send optimal data to Firebase Realtime databse removing redundant data. The system is technologically rich but economical, reliable, user friendly and serves multiple purposes. Remote healthcare monitoring systems include intelligent sensors and devices that can operate continuously online emergency monitoring.