SMART HEALTH MONITORING SYSTEM USING IOT

A PROJECT REPORT

Submitted By:

RAJ GODARA
GURLEEN KAUR
VARDAN BALIYAN
ANNU

University Roll Number:

20BCS4503
20BCS4526
20BCS4561
20BCS4608

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

(INTERNET OF THINGS)

Under the Supervision of:

MR. GAURAV SONI

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI – 140413, PUNJAB

APRIL, 2024

SMART HEALTH MONITORING SYSTEM USING IOT

A PROJECT REPORT

Submitted by

RAJ GODARA (20BCS4503)
GURLEEN KAUR (20BCS4526)
VARDAN BALIYAN (20BCS4561)
ANNU (20BCS4608)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

(INTERNET OF THINGS)

CHANDIGARH UNIVERSITY

APRIL, 2024

BONAFIDE CERTIFICATE

Validated that the work described in the project report, **"SMART HEALTH MONITORING SYSTEM USING IOT",** was completed under my or our supervision and is the legitimate work of **"RAJ GODARA (20BCS4503), GURLEEN KAUR (20BCS4526), VARDAN BALIYAN (20BCS4561), ANNU (20BCS4608)."**

MR. AMAN KAUSHIK

HEAD OF THE DEPARTMENT (AIT-CSE)

MR. GAURAV SONI (E9610)

SUPERVISOR

DECLARATION

I, **'RAJ GODARA', 'GURLEEN KAUR', 'VARDAN BALIYAN', 'ANNU'** student of **'Bachelor of Engineering in the Internet of Things**, **session: 2020-2024**, Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, at this moment, declare that the work presented in this Project Work entitled **'SMART HEALTH MONITORING SYSTEM USING IOT'** is the result of our own legitimate work that has been done with consideration for engineering ethics. It is accurate to the best of our knowledge. It doesn't contain any content that has already been published or written by someone else, nor does it contain any content that has been approved for the award of any other degree or certificate from the university or another higher education institution, unless appropriate credit has been indicated within the text.

CANDIDATE NAME & UID

RAJ GODARA (20BCS4503)
GURLEEN KAUR (20BCS4526)
VARDAN BALIYAN (20BCS4561)
ANNU (20BCS4608)

Place: Chandigarh University

Month & Year: APRIL, 2024

TABLE OF CONTENTS

ABSTRACT

This project describes the creation of an Internet of Things-based smart health surveillance system that allows for the real-time data tracking of a person's vital signs. An ESP8266 Node MCU microcontroller is integrated with temperature, humidity, and heart rate sensors in the system to gather information on the user's body temperature, ambient humidity, and heart rate. A specially created Android mobile application shows the gathered data, which is wirelessly sent to a Firebase cloud database. With the use of the mobile app, users can easily create individualized profiles, view vital sign data, and get warnings. Early health issue diagnosis and proactive illness management are aided by successful implementation, which shows that low-cost IoT devices, cloud computing, and mobile technologies may all be used for effective remote health monitoring.

Keywords - Heart Rate, Temperature, Sensors, Smart Health Monitoring, Internet of Things, Humidity.

ACKNOWLEDGEMENT

Apart from the efforts of all the crew members, the segment of this challenge document subject matter relies upon in large part the encouragement and steerage of our teachers. We take this possibility to specifically our gratitude to the academics who've been instrumental in the approval of this challenging subject matter. We would like to show our greatest appreciation to **Mr. Gaurav Soni** and our Chancellor **Mr. Satnam Singh Sandhu** for giving us this golden opportunity to complete a great project on "**SMART HEALTH MONITORING SYSTEM"** which also helped us do extensive study, and we learned a great deal of fresh information and achieved a good amount of knowledge through this project. The contribution and support received from all the team members **'RAJ DODARA', 'GURLEEN KAUR', 'VARDAN BALIYAN', and 'ANNU',** including are high-spirited. The team spirit shown by all has made this project successful.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In recent years, the Internet of Things (IoT) has emerged as a revolutionary technology that has transformed various sectors, including healthcare. The integration of IoT with wearable devices and sensor networks has paved the way for the development of smart health monitoring systems, enabling real-time monitoring of vital signs and environmental conditions. This report presents a comprehensive overview of a smart health monitoring system that leverages the power of IoT and various components to provide seamless and efficient health monitoring.

The proposed system utilizes the ESP8266 NodeMCU, a low-cost and versatile microcontroller, as the core component. This microcontroller is programmed using the Arduino IDE, a widely adopted open-source software for developing embedded systems. The system incorporates two essential sensors: the DHT11 temperature and humidity sensor, and a heart rate sensor. The DHT11 sensor accurately measures ambient temperature and humidity, while the heart rate sensor monitors the user's pulse rate, providing valuable insights into their cardiovascular health.

The collected data from the sensors is transmitted wirelessly to a cloud-based platform, Firebase, which serves as a robust and scalable real-time database. Firebase ensures secure data storage and enables seamless integration with various platforms, including Android applications.

To provide a user-friendly interface and enable remote monitoring, an Android application has been developed using Android Studio and Java programming language. This application not only displays the real-time sensor data but also offers features such as historical data visualization, alert notifications, and the ability to share health

data with healthcare professionals or caregivers.

The integration of these components creates a comprehensive and intelligent health monitoring system that empowers users to take proactive measures in managing their well-being. By continuously monitoring vital signs and environmental conditions, the system can detect potential health risks and alert users or healthcare providers, enabling timely intervention and preventive care.

This report delves into the design, implementation, and evaluation of the smart health monitoring system, highlighting its potential applications, challenges faced, and future enhancements. It serves as a valuable resource for researchers, healthcare professionals, and technology enthusiasts interested in exploring the intersection of IoT, wearable devices, and healthcare.

1.1. IDENTIFICATION

In the context of the "Smart Health Monitoring System using IoT" project, the identification of the issue or problem statement is crucial as it sets the foundation for the project's objectives and justifies its significance. Here's a potential identification of the issue:

Health monitoring has traditionally been a challenge, particularly in scenarios where regular monitoring is required or when access to healthcare facilities is limited. Traditional methods of health monitoring often rely on infrequent visits to healthcare providers, which may lead to delayed diagnosis and treatment of health conditions. Moreover, environmental factors such as temperature and humidity can significantly impact an individual's health, but monitoring these factors continuously in real-time has been difficult with conventional approaches.

The rising prevalence of chronic diseases, an aging population, and the increasing demand for preventive healthcare have highlighted the need for innovative and efficient health monitoring solutions. Additionally, the limitations of traditional healthcare systems, including high costs, limited accessibility, and inefficient data management, have further exacerbated the need for alternative approaches.

The advent of the Internet of Things (IoT) and the proliferation of wearable devices and sensor networks have opened up new possibilities for addressing these challenges. However, existing IoT-based health monitoring solutions often lack comprehensive integration of vital sign monitoring, environmental condition monitoring, and user-friendly interfaces, limiting their widespread adoption and effectiveness.

The proposed "Smart Health Monitoring System using IoT" project aims to address these issues by developing a comprehensive and user-friendly solution that combines the monitoring of vital signs (such as heart rate) and environmental conditions (temperature and humidity) using IoT technology. By leveraging the capabilities of the ESP8266 Node MCU microcontroller, DHT11 temperature and humidity sensor, heart rate sensor, and the integration with Firebase and an Android application, the project seeks to provide real-time monitoring, data storage, and user-friendly visualization of health-related data.

The identification of this issue highlights the need for an innovative and integrated approach to health monitoring that can provide continuous and efficient monitoring, early detection of potential health risks, and improved accessibility to healthcare services. By addressing this issue, the proposed project has the potential to contribute to better health outcomes, enhanced preventive care, and improved quality of life for individuals across various settings, including remote areas, elderly care facilities, and fitness and wellness applications.

1.2. IDENTIFICATION OF PROBLEM

Traditional healthcare systems face several challenges in providing continuous, real-time monitoring of vital signs and environmental conditions that can significantly impact an individual's health. The lack of efficient and accessible health monitoring solutions can lead to delayed diagnosis and treatment of health conditions, putting individuals at risk of developing or exacerbating chronic diseases.

Existing IoT-based health monitoring solutions often lack comprehensive integration of vital sign monitoring, environmental condition monitoring, and user-friendly interfaces. Many solutions focus on monitoring either vital signs or environmental factors, but fail to provide a holistic approach that considers the interplay between these factors and their impact on overall health.

Furthermore, current solutions may suffer from limitations such as:

1. Lack of real-time data collection and analysis: Traditional methods often rely on infrequent visits to healthcare facilities, leading to gaps in data collection and delayed response to potential health issues.

2. Limited accessibility and portability: Some existing solutions are limited to specific healthcare settings or require specialized equipment, making them less accessible and convenient for individuals in remote areas or those requiring continuous monitoring.

3. Inefficient data management and visualization: Health data may be scattered across multiple platforms or presented in a complex or non-intuitive manner, making it difficult for individuals or healthcare providers to interpret and act upon the information efficiently.

4. Limited user-friendliness and engagement: Many solutions lack user-friendly interfaces or fail to engage individuals in actively monitoring and managing their health, reducing compliance and limiting the potential benefits of the technology.

The proposed "Smart Health Monitoring System using IoT" project aims to address these problems by developing an integrated solution that combines the monitoring of vital signs (heart rate) and environmental conditions (temperature and humidity) using

IoT technology. By leveraging the capabilities of the ESP8266 Node MCU microcontroller, DHT11 temperature and humidity sensor, heart rate sensor, and the integration with Firebase and an Android application, the project seeks to provide a comprehensive, real-time, and user-friendly health monitoring solution.

Addressing these problems can lead to early detection of potential health risks, improved preventive care, better disease management, and enhanced accessibility to healthcare services, ultimately contributing to better health outcomes and quality of life for individuals across various settings.

1.3. IDENTIFICATION OF TASKS

1. Hardware Setup and Configuration:
   - Set up and configure the ESP8266 Node MCU microcontroller
   - Interface the DHT11 temperature and humidity sensor with the ESP8266
   - Interface the heart rate sensor with the ESP8266
   - Design and implement the circuit for connecting all hardware components
   - Ensure proper power management and battery considerations
2. Arduino IDE Programming:
   - Program the ESP8266 Node MCU using the Arduino IDE
   - Implement code for reading data from the DHT11 sensor and heart rate sensor
   - Process and format the collected data for transmission
   - Establish wireless communication between the ESP8266 and Firebase
3. Firebase Integration:
   - Set up a Firebase project and configure the real-time database
   - Implement code for securely transmitting sensor data from the ESP8266 to Firebase
   - Ensure proper data storage and retrieval mechanisms in Firebase
4. Android Application Development:

- Design the user interface (UI) for the Android application using Android Studio
- Implement code for retrieving data from Firebase and displaying it in the app
- Develop data visualization techniques (e.g., charts, graphs) for sensor data
- Implement notification systems for alerting users or healthcare providers
- Incorporate user authentication and data privacy measures

5. System Integration and Testing:

- Integrate the hardware components with the Arduino IDE programming
- Test the data acquisition and transmission from sensors to Firebase
- Test the Android application for proper data retrieval and visualization
- Conduct unit testing for individual components and integration testing for the complete system
- Evaluate system performance (data transmission rates, latency, battery life)
- Perform user experience testing and gather feedback

6. Data Analysis and Interpretation:

- Analyze the collected data (temperature, humidity, heart rate) using appropriate statistical methods
- Interpret the analyzed data and derive insights into potential health implications
- Identify patterns or correlations between environmental factors and vital signs

7. Documentation and Reporting:

- Document the project's design, implementation, and testing processes
- Prepare a comprehensive report detailing the system architecture, components, results, and findings
- Include relevant diagrams, code snippets, and visualizations in the report

8. Deployment and Application:

- Explore potential applications and use cases for the smart health monitoring system
- Identify target users or settings (e.g., remote patient monitoring, elderly care, fitness

tracking)

- Develop deployment strategies and guidelines for implementation in real-world scenarios

9. Future Enhancements:

- Identify potential future enhancements or extensions to the system
- Explore integration with other wearable devices or sensors
- Investigate the application of machine learning for pattern recognition and anomaly detection
- Evaluate interoperability with electronic health record (EHR) systems

10. Presentation and Defense:

- Prepare a comprehensive presentation summarizing the project's objectives, approach, and findings
- Defend the project, addressing questions and concerns from the audience or evaluation committee

These tasks should be carefully planned, scheduled, and executed in a systematic manner to ensure the successful completion of the "Smart Health Monitoring System using IoT" project.

1.4. TIMELINE

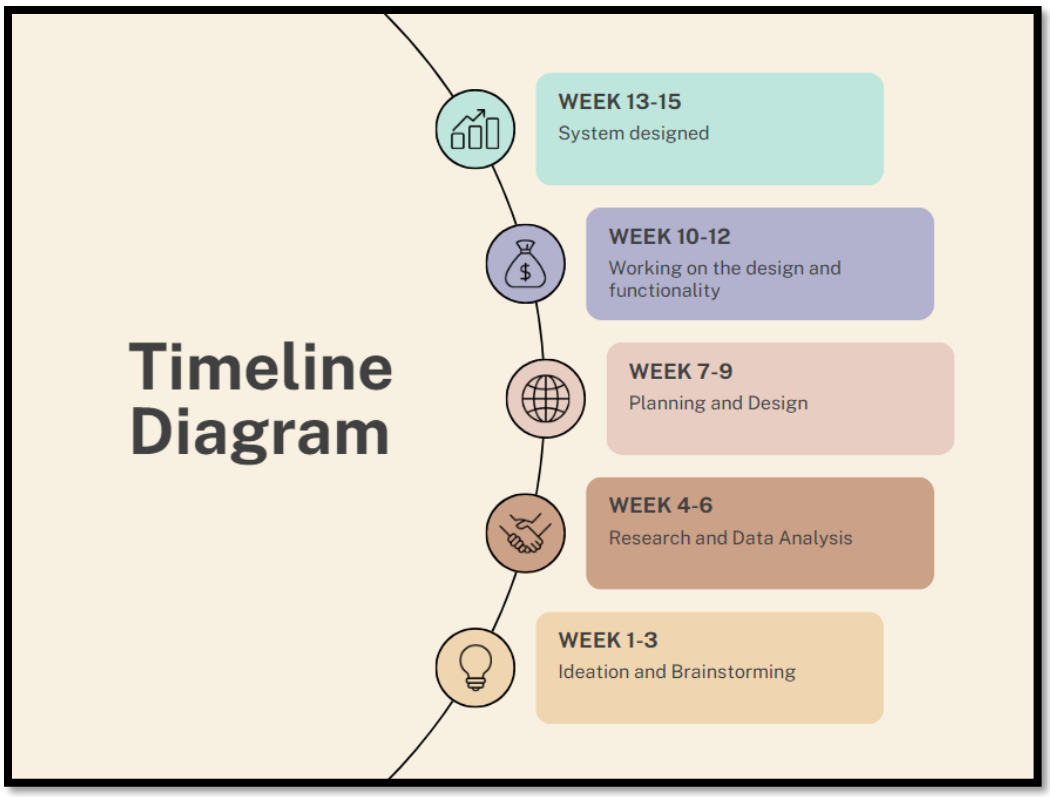


Figure 1: Timeline Diagram

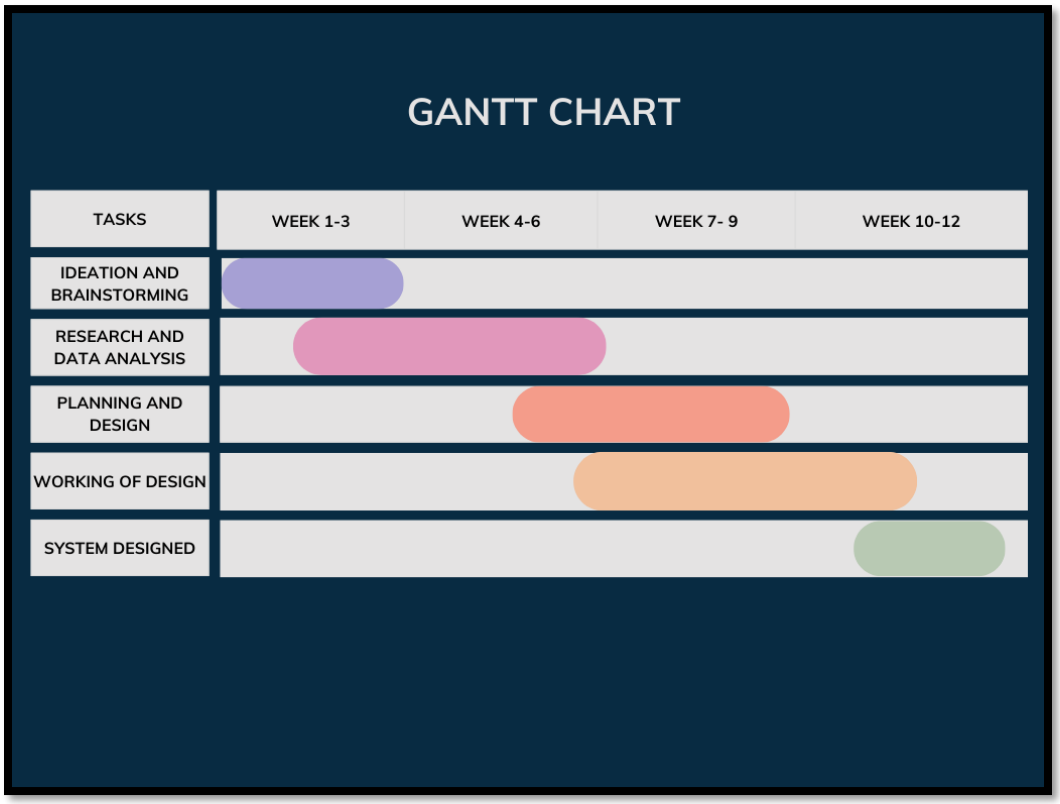


Figure 2: Gantt Chart

1.5. ORGAIZATION OF THE REPORT

The report is organized as follows:

Chapter 2 discusses Literature Review and the Background Study of the System.

Chapter 3 discusses the Design Flow process and the methodology used in the implementation of the system.

Chapter 4 validates the designed system by obtaining the results after implementation.

Chapter 5 concludes the report followed by References of the project.

CHAPTER 2

LITERATURE REVIEW/BACKGROUND STUDY

2.1. EXISTING SOLUTIONS

The existing systems include predicting health issues using various IOT models. Some of these are:

a. The paper "Smart Health Monitoring System through IoT" presented at the International Conference on Communication and Signal Processing in April 2019 discusses the role of IoT in healthcare. The authors emphasize the potential of IoT in connecting individuals through wearable gadgets, enabling health monitoring and providing timely solutions for abnormal health conditions. The research focuses on capturing sensor data, analyzing it, and providing feedback to patients based on various health parameters. The paper reviews various applications and methods related to IoT in healthcare, discussing the use of wireless monitoring systems, Raspberry Pi-3 board, and Arduino board as gateways for capturing real-time data. The proposed work involves continuous monitoring of different health parameters through a smartwatch, with collected data sent to the cloud for further analysis. The paper also discusses the communication and interaction between smartwatches, smartphones, ThingSpeak cloud, and IFTTT, emphasizing the use of Bluetooth low energy for efficient wireless communication. The paper concludes that the proposed system provides flexibility and continuous monitoring of heart conditions, offering a preventive approach to healthcare.

b. The "IoT-Based Health Monitoring System" is a study that explores the potential of the Internet of Things (IoT) in healthcare, mainly in remote health surveillance. The system uses wearable sensors and smartphones to continuously monitor a patient's vital parameters, such as heartbeat and temperature. The benefits of IoT-based health monitoring include disease prevention, remote diagnosis, and reduced

healthcare costs. The system a uses wireless communication to transmit the data to a medical server, which can then be accessed by authorized personnel through IoT platforms. The system's experimental setup includes sensors for body temperature, pulse rate, and room humidity, the process of data transmission using IoT. The study also presents experimental results, including sensor calibration and diagnosis. The paper concludes by emphasizing the potential of IoT in remote health monitoring and disease diagnosis, allowing for continuous patient monitoring, reduced hospital stays, and remote diagnosis by medical practitioners.

c. The document presents a "Smart Health Monitoring System" designed to provide continuous healthcare monitoring, especially in rural or remote areas. The system uses biomedical sensors connected to an Arduino UNO controller to gather patient data, which is then transmitted to a server and visualized on a smartphone through a dedicated Android application. The architecture includes the integration of medical sensors and a controller responsible for collecting patients' physical parameters. The data is saved in a CSV format on an SD card and uploaded to an online database. The hardware description includes the Arduino UNO and the calculation of temperature and heartbeat rate sensors. The development of an Android application named s-Health is highlighted, offering functionalities such as BMI calculation, medication reminders, nearby hospital information, and home remedies. The conclusion emphasizes the successful implementation of the system and proposes future enhancements, such as incorporating more sensors and refining the Android application's features to make it more dynamic. The document provides a comprehensive overview of the Smart Health Monitoring System and offers insights into potential future enhancements, including the expansion of sensor capabilities and the refinement of the Android application's features.

d. The "IoT-Based Healthcare-Monitoring System towards Improving Quality of Life: A Review" is a comprehensive analysis of IoT-based healthcare-monitoring

systems, focusing on remote patient monitoring and improving quality of life. In reviewing recent research on these systems, the report compares their efficacy, security, privacy, and data protection. It also covers issues with healthcare security and privacy, wearable and wireless sensor-based IoT monitoring systems, and recommendations for future IoT healthcare applications. The review emphasizes these systems' advantages such as real-time monitoring, preventive care, and remote healthcare, while addressing challenges such as range and bandwidth limitations, security and privacy concerns, and improved data protection. The paper also discusses the Internet of Wearable Things (IoWT) and its potential to revolutionize healthcare by integrating sensors into wearable devices for continuous monitoring of health and activity. The review emphasizes the importance of addressing security, privacy, and data protection concerns in the development of IoT healthcare applications.

e. The research article "IoT-Based Health Monitoring System Development and Analysis" describes the development and deployment of an Internet of Things-based health monitoring system. The system measures a patient's body temperature, heartbeat, and oxygen saturation levels, sending the data to a mobile application via Bluetooth. This innovative electronic device aims to identify irregularities within the body, especially for patients in rural areas with limited access to healthcare facilities. The system aims to increase affordability and easy access to personal healthcare, particularly for patients with chronic diseases and during the COVID-19 pandemic. The study details hardware components, cost analysis, and real-life testing results, showcasing its potential for real-time health monitoring and remote patient care. It compares the developed system with existing IoT-based health monitoring systems, emphasizing its unique features and capabilities. Future improvements include integrating new algorithms for enhanced security, using Raspberry Pi as a microcontroller, adding additional sensors, and optimizing IoT

management processes. The research presents a comprehensive approach to developing an IoT-based health monitoring system, demonstrating its potential for real-time health monitoring, remote patient care, and affordable healthcare solutions.

f. The document presents an "IoT-based Health Monitoring System" that measures health-related parameters like body temperature, pulse, ECG, and blood pressure to predict diseases. The system aims to provide timely medical assistance and reduce the need for frequent health check-ups. It is user-friendly and can be implemented in homes, old-age homes, and work environments. The hardware part uses a PCB package and sensors, which are sent to a cloud server using machine learning algorithms to predict different diseases based on the measured parameters. The system uses microcontrollers and sensors like the ESP32 and MAX30100 to sense and calculate health parameters. Experimental results show a prototype of the system, calibrated using a microcontroller and displayed on an LCD. The future scope suggests increasing the number of sensors and predicting other diseases using different datasets. The conclusion emphasizes the importance of IoT in remote health monitoring and the system's potential to accurately predict heart disease.

This document provides an in-depth analysis of an IoT-based Integrated Health Monitoring System, highlighting its potential benefits in real-time patient monitoring and healthcare quality improvement. It emphasizes the growing importance of healthy aging and the need for in-home health monitoring systems. The system uses Node MCU (ESP8266) for remote healthcare, a pulse/heart rate sensor, and Google Firebase for database storage. It uses two Android applications, Smart Care Patient and Smart Care Admin, for patient and healthcare provider interfaces. The system faces challenges such as expanding sensor capabilities, enhancing data storage, and improving security. Upcoming tasks include adding more sensors and growing the database, and introducing a web-based control system. The document serves as a

comprehensive resource on the potential impact of IoT on healthcare, its architecture, implementation, and future directions for improvement. The proposed IoT-based Integrated Health Monitoring System has the potential to address critical healthcare challenges, particularly in remote patient monitoring and real-time healthcare decision-making.

2.2. PROPOSED SYSTEM

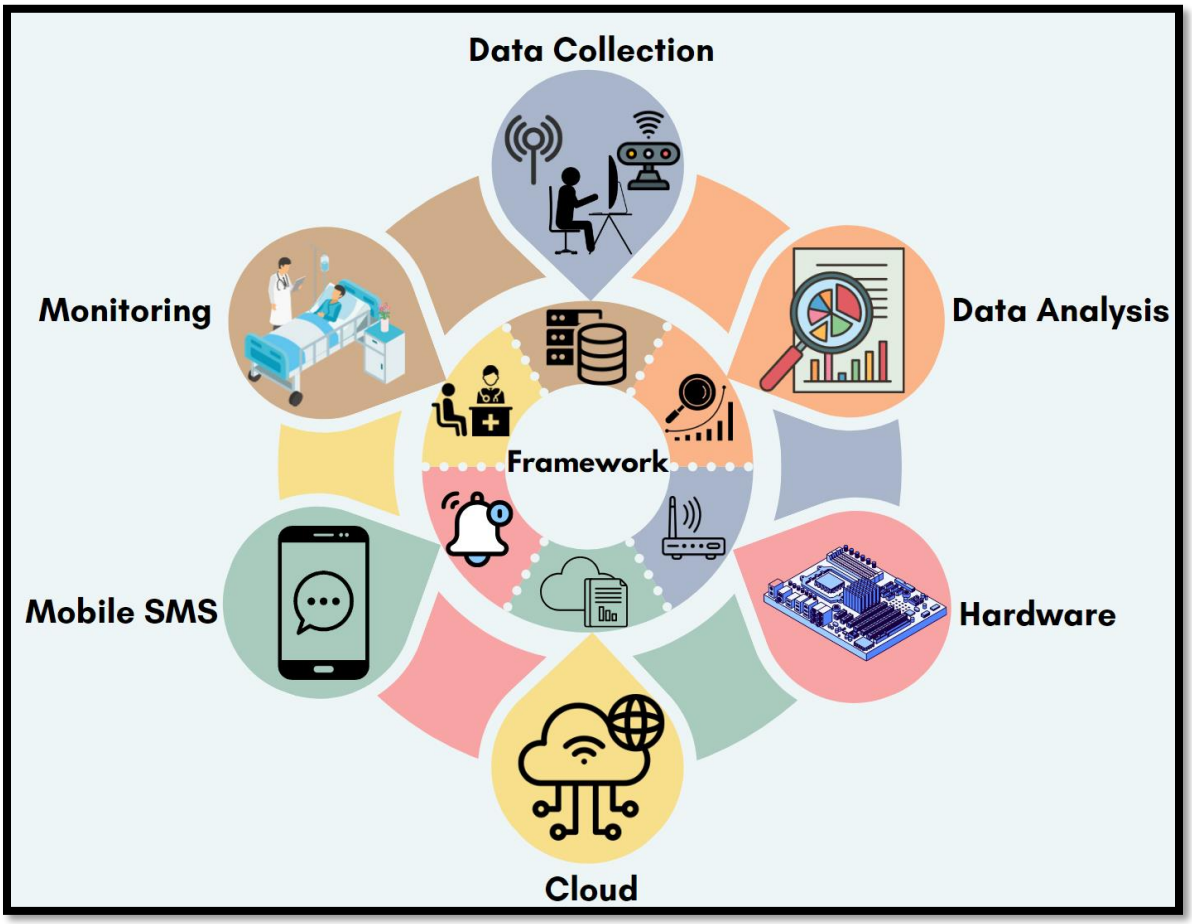


Figure 3: System Overview/Framework

Figure 3 represents an overview of the proposed system that involves utilizing various hardware components and software tools to create a system that continuously monitors patients' health conditions and sends data to healthcare professionals.

2.3. REVIEW SUMMARY

Sr.No.	Author	Article	Tools/Techniques
1.	Madhan Mohan, Sathya Pichandi	Smart Health Monitoring System through IOT	ThingSpeak, IFTTT, Android Application
2.	Prajoona Valsalan, Ahmed Tariq, Ali Hussain	IOT based Health Monitoring System	Temperature sensor, Heartbeat sensor, IOT servers
3.	Tarannum Khan, Manju K. Chattopadhyay	Smart Health Monitoring System	Arduino UNO, LCD Serial Monitor, Temperature sensor, Heartbeat rate sensor
4.	Suliman Abdulmalek, Abdul Nasir abd ghafar bin, Waheb A. Jabbar Al-Areeqi, Mukarram A M Almuhaya	IOT-based Healthcare-Monitoring System towards Improving Quality of Life: A Review	IOT Gateway, Local server
5.	Mohammad Monirujjaman Khan, Turki M. Alanazi, Amani Abdulrahman Albraikan, Faris A Almalki	IOT-based Health Monitoring System Development and Analysis	Arduino UNO, LM35, Bluetooth Module HC-05, MAX30100, LCD display, Jumper wires, Breadboard
6.	Mansi Mhalsakant Gajare, Manas A Dani, Payal D Deshmukh, Pritesh Chaudhari	IOT based Health Monitoring System	ESP32, MAX30100, PCB, Blynk app, ML Prediction website

7.	Salma Sultana, Sadia Rahman, Md. Atikur Rahman, Narayan Ranjan Chakraborty	An IOT based Integrated Health Monitoring System	ESP8266, LM35, SEN-11574, Breadboard, Jumper wires

2.4. PROBLEM DEFINITION

In today's fast-paced world, the prevalence of lifestyle-related health issues and chronic diseases has increased significantly. Monitoring vital signs such as body temperature, heart rate, and humidity levels plays a crucial role in early detection and proactive management of various health conditions. However, traditional methods of monitoring these parameters often require frequent hospital visits or the use of expensive medical equipment, making it inconvenient and inaccessible for many individuals. The Smart Health Monitoring System using IoT aims to address this challenge by providing a cost-effective and user-friendly solution for continuous remote monitoring of vital signs.

2.5. GOALS / OBJECTIVES

The primary goals of the Smart Health Monitoring System are to develop a cost-effective and user-friendly solution for continuous remote monitoring of vital signs, including body temperature, heart rate (BPM), and ambient humidity levels. By leveraging Internet of Things (IoT) technologies, the system aims to empower individuals to take an active role in managing their well-being, while also providing healthcare professionals with timely access to critical health data for informed decision-making. The specific objectives include seamless integration of an ESP8266 NodeMCU microcontroller with sensors, wireless transmission and secure storage of data in the Firebase cloud, development of a user-friendly Android mobile application for real-time visualization of vital signs, and incorporation of features

like personalized profiles, customizable alerts, and historical data tracking.

- Develop a cost-effective solution for remote monitoring of vital signs.
- Continuously track body temperature, heart rate (BPM), and humidity levels.
- Integrate IoT technologies for wireless data transmission and cloud storage.
- Create a user-friendly Android app for real-time visualization of health data.
- Enable personalized profiles, customizable alerts, and historical data tracking.
- Empower individuals in managing their well-being and support informed healthcare decisions.

CHAPTER 3

DESIGN FLOW/PROCESS

3.1. EVALUATION & SELECTION OF SPECIFICATIONS/FEATURES

The proposed architectural design may include specifications/features such as:

Hardware Components:

a. ESP8266 NodeMCU

Figure 4: ESP8266 NodeMCU

Figure 4, the ESP8266 NodeMCU used in the research. The ESP8266 NodeMCU is commonly used in smart health monitoring systems based on IoT technology. These systems leverage the NodeMCU's capabilities to gather, process, and transmit health-related data for monitoring purposes. The NodeMCU, integrated with sensors like heartbeat sensors, temperature sensors, and blood pressure sensors, enables the real-time monitoring of vital health parameters such as heart rate, temperature, and blood pressure. By connecting these sensors to the NodeMCU, the system can analyze the data to detect normal or abnormal conditions, allowing for timely intervention and healthcare support. Additionally, the NodeMCU facilitates remote monitoring and notification functionalities, enabling doctors to receive alerts via mobile messages in case of abnormal health conditions, thus enhancing healthcare services and improving

patient outcomes.

b. Heart rate Pulse Sensor sensor module

Figure 5: Heart Rate Pulse Sensor Module

Figure 5 shows the heart rate pulse sensor module used in the research. The heart rate pulse sensor module is a plug-and-play sensor designed for Arduino and compatible boards, allowing users to easily incorporate live heart rate data into their projects. It features a compact and discrete design, making it suitable for wearables, medical devices, and various projects where space is a key consideration. This sensor is versatile and can be used in applications ranging from automation and robotics to environmental and industrial monitoring. However, it is essential to note that this heart rate sensor module is not intended for medical use and should not be used for diagnosing, preventing, or treating any medical condition. The measurements and statistics provided by this sensor are for informational and educational purposes only, not for medical purposes. Additionally, the heart rate pulse sensor module operates at a working voltage of 3 to 5 volts, with a working current of 4 mA at 5 volts, and provides connections for GND, VCC, and analog signal out. It is a valuable tool for STEM projects, helping users develop and enhance their skills in science, technology,

engineering, and mathematics.

c. DHT11 Temperature and Humidity Sensor

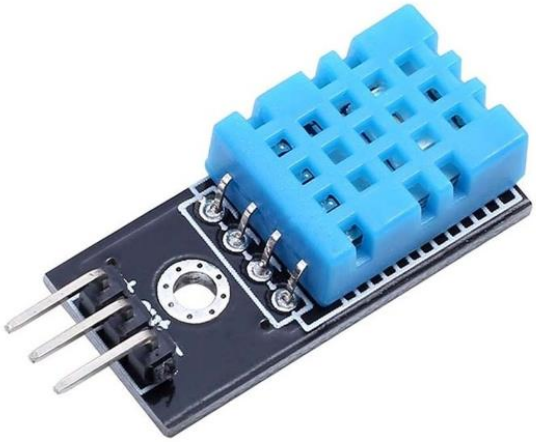


Figure 6: DHT11 Temperature Sensor

Figure 6, the DHT11 temperature and humidity sensor is a crucial component for smart health monitoring systems using IoT technology. This sensor plays a vital role in measuring room temperature (0-50 degrees Celsius) and humidity (20%-80% with an accuracy of +/- 5%). It utilizes a capacitive humidity sensor and a thermistor to provide accurate readings of the surrounding environment. In the context of health monitoring systems, the DHT11 sensor can be integrated with microcontrollers like Arduino and Raspberry Pi to monitor environmental conditions that are essential for patient comfort and well-being. By incorporating the DHT11 sensor into IoT-based health monitoring systems, healthcare providers can track and analyze temperature and humidity levels in real-time, ensuring optimal conditions for patients. Additionally, the sensor's simplicity of use, reliability, fast response time, and long-term stability make it a valuable tool for continuous monitoring of environmental parameters in healthcare settings.

d. Breadboard

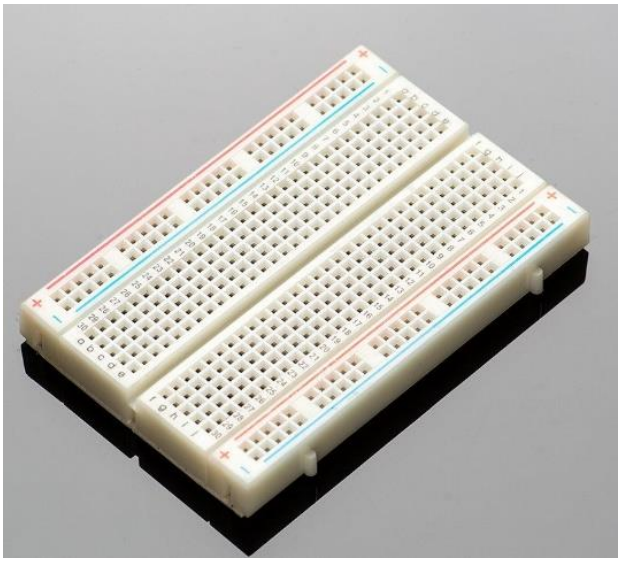


Figure 7: Breadboard

Figure 7, A breadboard is a fundamental tool in electronics used for building and testing circuits without the need for soldering. It is a solderless device that allows for easy prototyping and temporary circuit construction. Breadboards consist of multiple rows and columns of interconnected metal clips that hold components and wires in place. They are commonly used for creating prototypes, testing new parts, and analyzing circuits before finalizing a design. Breadboards come in various sizes, from tiny premium breadboards to half-sized premium breadboards, offering different tie points for accommodating circuits of varying complexity. These versatile tools are essential for both beginners and experienced electronics enthusiasts, providing a platform to experiment, learn, and develop electronic projects efficiently.

e. Jumper Wires

Figure 8: Jumper Wires

Figure 8, Jumper wires are essential components in electronics used for making quick and easy connections in low voltage circuits without the need for tools or soldering. They come in various forms, such as ribbons of multicolored wires or preformed breadboard jumper wire kits, offering different lengths and connector types to suit different circuit requirements. Jumper wires are commonly used with breadboards and other prototyping tools to facilitate the easy modification of circuits as needed. These wires typically come in male-to-male, male-to-female, and female-to-female versions, each serving different connection purposes. While the colors of jumper wires do not have specific meanings, they can be used to differentiate between types of connections, such as ground or power. Jumper wires are versatile components that enable efficient circuit building and testing in electronics projects, making them indispensable for both beginners and experienced enthusiasts in the field.

Software Components:

a. Arduino IDE

The Arduino IDE plays a crucial role in smart health monitoring systems using IoT

technology. It serves as the programming environment where developers can write, compile, and upload code to microcontrollers like Arduino and ESP8266 for health monitoring applications. By using the Arduino IDE, developers can easily program the microcontrollers to interact with sensors such as temperature sensors, pulse sensors, and other health monitoring devices. The IDE provides a user-friendly interface and a vast library of functions that simplify the development process, making it accessible for both beginners and experienced developers. Additionally, the Arduino IDE allows for seamless integration with IoT platforms like Thingspeak, enabling the transmission of health data to the cloud for remote monitoring and analysis. Overall, the Arduino IDE is a fundamental tool in the implementation of smart health monitoring systems, enabling the creation of efficient and reliable IoT solutions for healthcare applications.

b. Firebase

Firebase is a powerful tool for smart health monitoring systems using IoT technology. It serves as a real-time database that can store and synchronize data across multiple clients, making it ideal for monitoring and analyzing health data remotely. In the context of smart health monitoring systems, Firebase can be used to store patient health parameters, sensor readings, and alerts generated by the system. By integrating Firebase into the IoT infrastructure, health data can be securely stored and accessed from anywhere, enabling healthcare providers to monitor patients' conditions in real-time. Additionally, Firebase's real-time capabilities allow for instant updates and notifications, ensuring that doctors receive timely alerts in case of abnormal health conditions. This seamless data management and communication provided by Firebase enhance the efficiency and effectiveness of smart health monitoring systems, bridging the gap between healthcare providers and patients.

c. Android Studio (Java)

Android Studio, utilizing Java programming language, plays a pivotal role in smart health monitoring systems using IoT technology. It serves as the development

environment for creating Android applications that interact with IoT devices and platforms to monitor and manage health data. In the context of smart health monitoring systems, Android Studio enables the creation of user-friendly mobile applications that can collect, display, and analyze health parameters obtained from IoT sensors. These applications can provide real-time monitoring of vital signs, such as body temperature, heart rate, and oxygen levels, allowing healthcare providers and patients to access and track health data conveniently. Additionally, Android Studio facilitates the integration of IoT functionalities, such as data transmission to cloud services like Firebase, enabling remote monitoring and analysis of health information. Overall, Android Studio, in conjunction with Java programming, empowers the development of intuitive and efficient mobile applications that enhance the monitoring and management of health data in IoT-based health monitoring systems.

System Features:

1. Real-time Monitoring:

The system should continuously monitor temperature, humidity, and heart rate data from the respective sensors in real-time, without any significant delays or lags.

The sampling rate for data collection should be configurable to meet the desired monitoring frequency and accuracy requirements.

The system should be capable of handling and processing data streams from multiple sensors simultaneously, ensuring seamless and synchronized monitoring.

2. Data Storage and Retrieval:

The sensor data collected from the ESP8266 NodeMCU should be securely transmitted to the Firebase real-time database using appropriate communication protocols and data formats.

The data should be organized and stored in the database in a structured manner, allowing for efficient retrieval and querying based on factors such as sensor type, timestamps, or user identifiers.

The Android application should be able to retrieve data from the Firebase database in real-time, enabling up-to-date display of sensor data and historical trends.

Appropriate data caching mechanisms should be implemented to ensure smooth data retrieval and minimize delays or interruptions in the application's user experience.

3. User-friendly Interface:

The Android application's user interface should follow best practices in design and usability, ensuring an intuitive and user-friendly experience.

The interface should be clean, uncluttered, and visually appealing, with clear labels and icons for easy navigation and understanding.

The application should support both landscape and portrait orientations, adapting the layout and content accordingly for optimal viewing on different device sizes and orientations.

Consideration should be given to accessibility features, such as support for different font sizes, color contrast, and compatibility with screen readers or other assistive technologies.

4. Data Visualization:

The Android application should incorporate data visualization techniques, such as line charts, bar graphs, or other suitable visualizations, to present sensor data in a clear and understandable manner.

The visualizations should be interactive, allowing users to zoom, pan, or filter data based on specific time ranges or sensor types.

Appropriate labels, legends, and color coding should be used to enhance the clarity and interpretation of the visualized data.

The application should provide options to switch between different visualization modes or types, catering to user preferences and specific use cases.

5. Notification System:

The system should have the capability to generate notifications or alerts based on predefined thresholds or rules related to the sensor data.

For example, if the temperature or heart rate exceeds a certain value, the system should generate an alert or notification to prompt the user or healthcare provider for appropriate action.

The notifications should be customizable, allowing users to set their preferred thresholds or rules based on their specific health conditions or requirements.

The notifications should be delivered through the Android application, as well as other channels such as push notifications, emails, or SMS (if integrated), depending on the user's preferences and the system's capabilities.

6. User Authentication and Data Privacy:

The Android application should implement secure user authentication mechanisms, such as username/password or biometric authentication (e.g., fingerprint or face recognition), to ensure data privacy and access control.

User data, including sensor readings and personal information, should be encrypted and securely stored in the Firebase database, adhering to industry-standard data privacy and security practices.

The application should implement role-based access control, allowing different levels of access and permissions for users, healthcare providers, or administrators.

Appropriate measures should be taken to ensure compliance with relevant data privacy regulations, such as GDPR or HIPAA, depending on the target region and application domain.

7. Portability and Accessibility:

The hardware components, including the ESP8266 NodeMCU, sensors, and associated circuitry, should be designed to be compact, lightweight, and easily transportable,

enabling monitoring in various settings (e.g., at home, during physical activities, or in healthcare facilities).

The system should be powered by rechargeable batteries or other portable power sources, ensuring extended operation without the need for constant external power sources.

The Android application should be compatible with a wide range of Android devices and versions, ensuring accessibility and usability for a broad user base.

Consideration should be given to remote monitoring capabilities, allowing healthcare providers or caregivers to access and monitor sensor data from remote locations, enabling timely interventions and support.

3.2. DESIGN CONSTRAINTS

The design constraints for the "Smart Health Monitoring System using IoT" project:

1. Hardware Constraints:

The ESP8266 NodeMCU has limited memory (64-128KB RAM) and processing power, which may restrict the complexity of the firmware and the number of concurrent tasks it can handle. Power consumption is a critical factor, as the sensor module may need to run on batteries or low-power sources. Optimizing power usage for the ESP8266 and sensors is crucial for prolonged operation. The physical size and form factor of the sensor module should be compact and wearable, considering user comfort and portability. The wireless communication range and reliability of Wi-Fi or Bluetooth can be affected by environmental factors, interference, and obstacles, which may impact real-time data transmission.

2. Sensor Constraints:

The DHT11 temperature and humidity sensor has an accuracy of ±2°C for temperature and ±5% for humidity, which may require calibration or error handling mechanisms for precise measurements. The sampling rate and response time of the heart rate sensor should be optimized to capture accurate readings without

introducing significant delays or data loss. Sensor calibration and error handling mechanisms should be implemented to account for potential sensor drift, noise, or environmental factors that may affect sensor readings.

3. Software Constraints:

The limited memory and processing power of the ESP8266 may restrict the size and complexity of the firmware code, requiring efficient memory management and optimization techniques. Compatibility issues may arise between the Arduino IDE, libraries, and the ESP8266 board, potentially limiting the available functionality or requiring workarounds. Efficient data transmission protocols and techniques should be employed to ensure real-time data transfer without introducing significant latency or data loss.

4. Firebase Constraints:

Firebase pricing and usage limits may impose constraints on the number of simultaneous connections, data storage, and operations performed, impacting scalability and cost-effectiveness. The data structure and schema design in Firebase Realtime Database or Cloud Firestore should optimize data retrieval, querying, and synchronization performance for real-time updates. Handling real-time data streams and push notifications using Firebase Cloud Functions or Cloud Messaging may introduce latency or limitations on the number of concurrent requests or messages.

5. Android Application Constraints:

The wide range of screen sizes and resolutions on different Android devices may require responsive UI design and adaptive layouts to ensure optimal user experience. Battery consumption should be minimized through efficient data handling, background processing, and optimizations for continuous data monitoring and visualization. Offline data storage and synchronization mechanisms should be implemented to ensure data availability and consistency when the device is offline or has intermittent connectivity. User interface design and responsiveness should be optimized for real-time data visualization, ensuring smooth updates and minimizing

lag or stuttering. Compatibility with different Android versions and device configurations should be addressed to ensure broad device compatibility and consistent functionality.

6. Security and Privacy Constraints:

Secure authentication and authorization mechanisms should be implemented for the mobile application and Firebase services to protect user data and prevent unauthorized access. Encryption and secure transmission of sensitive health data, such as heart rate, should be implemented to comply with data privacy regulations and protect user privacy. Compliance with data privacy regulations (e.g., GDPR, HIPAA) should be ensured when handling personal health information, including data storage, transmission, and access controls.

7. Scalability and Performance Constraints:

The system should be designed to handle an increasing number of connected devices and users without compromising performance or introducing significant latency. Real-time data processing and visualization performance should be optimized for large datasets, ensuring smooth updates and efficient data handling. Load balancing and scalability of Firebase services (e.g., Cloud Functions, Realtime Database) should be considered to accommodate increased usage and data volume.

8. Usability and User Experience Constraints:

The mobile application interface should be intuitive and user-friendly, with clear navigation and feedback mechanisms to enhance the overall user experience. Accessibility considerations, such as support for screen readers, high-contrast modes, and alternative input methods, should be addressed for users with disabilities or special needs. Localization and internationalization support should be implemented to accommodate users from different languages and regions, ensuring accurate display of text, dates, and measurements.

9. Maintenance and Upgrade Constraints:

Over-the-air (OTA) firmware updates for the ESP8266 and sensor module should be

supported to address bug fixes, security vulnerabilities, and feature enhancements without requiring physical access to the devices. Mobile application updates should be seamless and compatible with newer Android versions, ensuring continued functionality and user experience improvements. Continuous integration and deployment (CI/CD) pipelines should be established for efficient updates and releases, automating testing, building, and deployment processes for both the firmware and mobile application.

These design constraints cover various aspects of the project, including hardware limitations, sensor accuracy and reliability, software efficiency, data storage and transmission, security and privacy, scalability and performance, usability, and maintenance. Addressing these constraints throughout the project's lifecycle is crucial for developing a robust, secure, and user-friendly Smart Health Monitoring System using IoT.

3.3. ANALYSIS OF FEATURES AND FINALIZATION SUBJECT TO CONSTRAINTS

1. Real-time monitoring of temperature and humidity:
    - The DHT11 sensor provides real-time data on ambient temperature and humidity levels.
    - This feature allows users to monitor environmental conditions that can impact health and take necessary actions.
2. Heart rate monitoring:
    - The heart rate sensor continuously measures the user's heart rate, which is a crucial vital sign.

- Real-time heart rate data can be used to detect potential health issues or monitor physical activities.

3. Cloud-based data storage:

   - The integration of Firebase real-time database enables secure and reliable storage of health data in the cloud.
   - Historical data can be accessed and analyzed for trends or patterns.

4. Mobile app integration:

   - The Android app developed using Android Studio and Java provides a user-friendly interface for accessing and visualizing the health data.
   - Users can monitor their health parameters remotely and receive notifications or alerts based on defined thresholds.

Finalization subjects to constraints:

1. Power consumption:

   - The system components, such as the ESP8266 NodeMCU and sensors, should be optimized for low power consumption to ensure long battery life or efficient power management.

2. Wireless connectivity:

   - Reliable and stable Wi-Fi connectivity is essential for the ESP8266 NodeMCU to communicate with the Firebase database seamlessly.
   - Potential interference or signal strength issues need to be addressed for uninterrupted data transmission.

3. Data security and privacy:

    - As the system handles sensitive health data, appropriate security measures should be implemented to protect user privacy and prevent unauthorized access to the data.

    - Encryption techniques and secure authentication protocols should be employed for data transmission and storage.

4. Scalability and extensibility:

    - The system should be designed with scalability in mind, allowing for easy integration of additional sensors or health monitoring devices in the future.

    - The Firebase database and Android app should be able to handle increasing amounts of data and user load.

5. User experience:

    - The Android app should provide a user-friendly interface with clear data visualization and intuitive navigation.

    - Proper error handling and feedback mechanisms should be implemented for a smooth user experience.

6. Hardware compatibility:

    - The hardware components, such as the ESP8266 NodeMCU, sensors, and their interfaces, should be compatible and work seamlessly together.

    - Appropriate calibration and testing should be performed to ensure accurate data readings.

By considering these features and addressing the constraints, the Smart Health

Monitoring System Using IoT can effectively monitor and track health parameters, providing users with valuable insights and enabling timely interventions or preventive measures.

3.4. DESIGN FLOW

The data flow diagram for the proposed system is as follows in Figure 9:

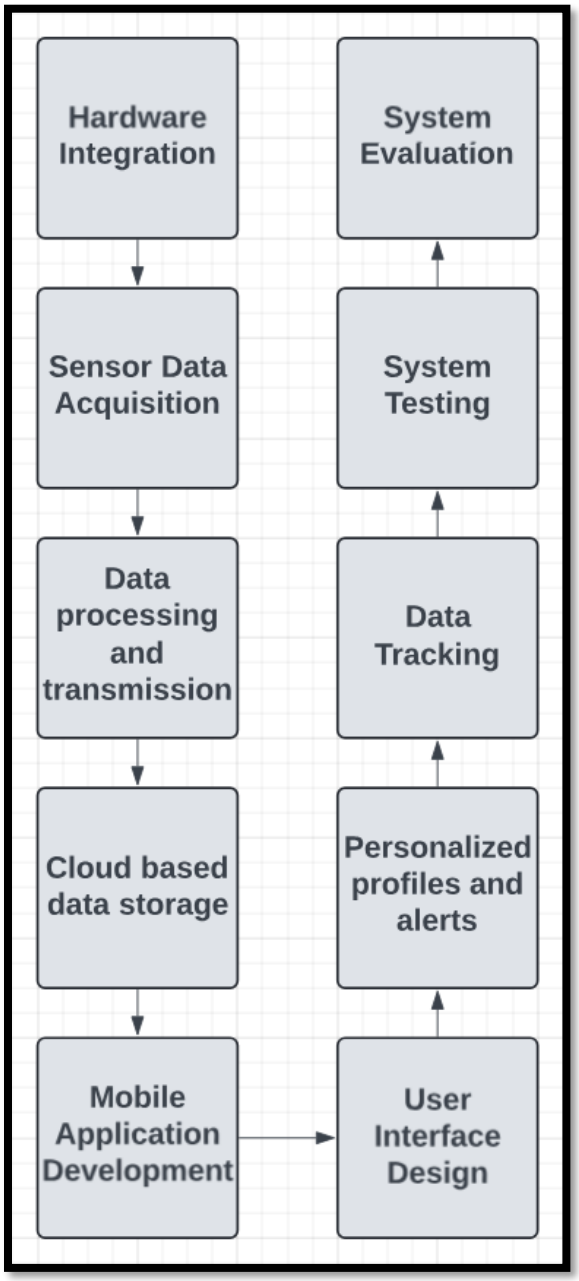


Figure 9: Data/Design Flow Diagram

The development and implementation of the Smart Health Monitoring System using IoT technologies involve a systematic methodology that integrates various hardware and software components. The methodology can be outlined as follows:

1. Hardware Setup

- ESP8266 NodeMCU microcontroller is selected as the central processing unit for its Wi-Fi capabilities, low power consumption, and ease of programming.
- A heart rate pulse sensor module and a DHT11 temperature and humidity sensor are chosen for their accuracy and reliability in measuring vital signs.
- The sensors are connected to the ESP8266 NodeMCU using a breadboard and jumper wires, enabling seamless communication and data acquisition.

2. Firmware Development

- The Arduino Integrated Development Environment (IDE) is used to develop the firmware for the ESP8266 NodeMCU microcontroller.
- The firmware code is written to initialize and control the connected sensors, read and process the sensor data, and establish a wireless connection with the Firebase Realtime Database.
- Relevant libraries are imported to support the functionality of the sensors and the Firebase integration.

3. Firebase Integration

- The Firebase Realtime Database is chosen as the cloud-based data storage solution for its real-time data synchronization, scalability, and ease of integration with mobile applications.
- The Firebase project is set up, and the necessary configurations are made to enable secure data transfer from the ESP8266 NodeMCU to the database.

4. Mobile Application Development

- Android Studio, an official Integrated Development Environment (IDE) for Android app development, is utilized to create the mobile application.
- The mobile app is developed using the Java programming language and follows the Model-View-Controller (MVC) architecture pattern.
- The Firebase Android SDK is integrated into the mobile app, allowing seamless retrieval and display of the vital sign data from the Firebase Realtime Database.
- The user interface (UI) of the mobile app is designed to provide a clear and intuitive experience for monitoring vital signs, including real-time display of temperature, heart rate (BPM), and humidity levels.

5. Data Visualization and User Experience
- The mobile app incorporates charts, graphs, and other visual elements to present the vital sign data in an easy-to-understand format.
- User experience (UX) principles are applied to ensure a smooth and intuitive navigation flow within the app, enabling users to access and interact with their health data effortlessly.

6. User Profile Management and Alerts
- The mobile app allows users to create personalized profiles, enabling them to set customized alert thresholds based on their individual health conditions or preferences.
- If any of the monitored vital signs deviate from the set thresholds, the app generates alerts to notify the user or their healthcare provider.

7. Data Tracking
- The mobile app incorporates features for tracking historical data trends, allowing users to access and analyze their vital sign data over time.
- This feature enables users to identify patterns, monitor progress, and make

informed decisions about their health and well-being.

8. Testing and Evaluation

- Extensive testing and evaluation of the Smart Health Monitoring System are conducted to ensure the reliability, accuracy, and performance of the overall solution.
- This includes testing the hardware integration, wireless data transmission, cloud-based data storage, and the functionality of the mobile application.
- User acceptance testing is performed to gather feedback from potential users and refine the system based on their requirements and preferences.

9. System Deployment and Maintenance

- After successful testing and evaluation, the Smart Health Monitoring System is prepared for deployment in real-world scenarios.
- Ongoing maintenance and updates are planned to ensure the system remains up-to-date with the latest security patches, bug fixes, and feature enhancements.

3.5. DESIGN SELECTION

For the "Smart Health Monitoring System using IoT" project, the following design selections were made:

- Hardware Design:

- ESP8266 Node MCU as the microcontroller for its Wi-Fi capabilities and ease of programming.

- Heart rate pulse sensor module and DHT11 temperature/humidity sensor for accurate vital sign monitoring.

- Breadboard and jumper wires for prototyping and interconnecting components.

- Software Design:

- Arduino IDE for developing the firmware and programming the NodeMCU.

- Firebase Realtime Database for secure cloud storage and real-time data synchronization.

- Android Studio and Java for developing the mobile application with a user-friendly interface.

- System Architecture:

- NodeMCU collects vital sign data from sensors and transmits it wirelessly to Firebase.

- Mobile app retrieves data from Firebase and displays temperature, BPM, and humidity levels in real-time.

- Additional features like personalized profiles, alerts, and historical data tracking.

This design selection ensures a cost-effective, modular, and scalable solution for remote health monitoring, leveraging IoT technologies and providing a seamless user experience.

3.6. IMPLEMENTATION PLAN/METHODOLOGY

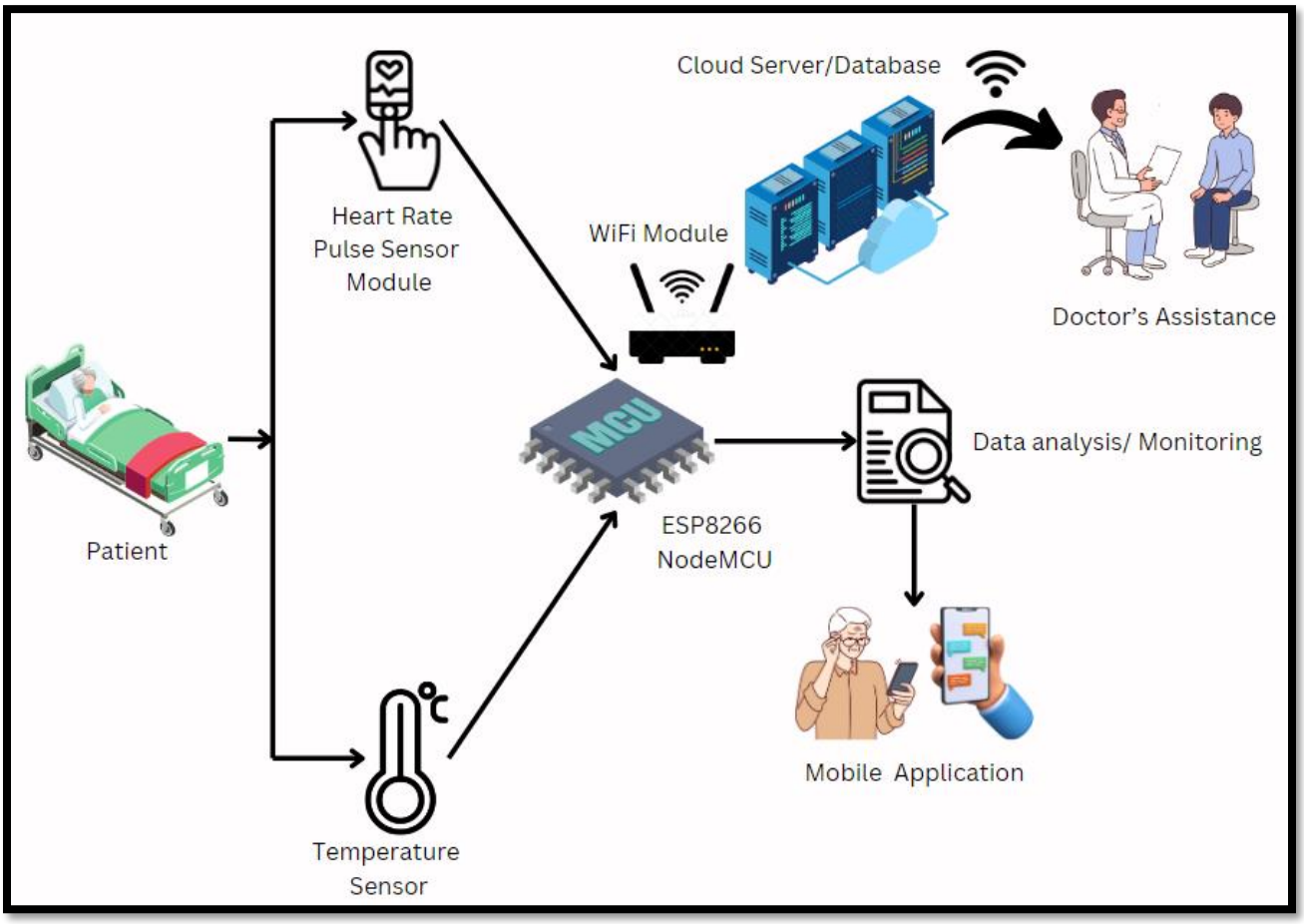


Figure 10: Methodology

Figure 10 is a virtual representation of the methodology and data flow from the patient to the doctor. The hardware components used in this research include the ESP8266 Node MCU, DHT11 temperature and humidity sensor, heart rate pulse sensor module, jumper wires, breadboard, and Firebase for cloud storage. The software tools involved are Arduino IDE for programming the microcontroller, Android Studio for developing the mobile application, and Firebase for cloud data storage.

The Smart Health Monitoring System using IoT consists of four main components: hardware setup, data acquisition, data transmission, and data presentation. These components are interconnected to form a cohesive ecosystem designed to facilitate personalized healthcare monitoring and predictive analytics.

The connections in the system involve wiring the sensors to the ESP8266 NodeMCU for data acquisition and processing. The data collected from the sensors is then transmitted to Firebase for storage and can be accessed through the mobile application developed in Android Studio. The Arduino IDE is used to program the microcontroller to ensure proper functioning of the system and communication with the sensors. This integrated setup allows for real-time health monitoring and data transmission, enhancing patient care and enabling remote monitoring efficiently.

The patient's data will be gathered by the sensors and then sent to the cloud via the WiFi module. The user will then receive notification of the gathered data via a mobile app.

CHAPTER 4

RESULTS ANALYSIS AND VALIDATION

4.1. CODE

ARDUINO CODE

```
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>
#include <DHT.h>
#include <PulseSensorPlayground.h>

// Define DHT sensor pin and type
#define DHTPIN D1
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Define Pulse Sensor pin
const int PulseWire = A0;
const int LED = LED_BUILTIN;          // The on-board
Arduino LED, close to PIN 13.
int Threshold = 480;
PulseSensorPlayground pulseSensor;

// Enter your network credentials
const char* ssid = "Vardan";
const char* password = "vardan2312";

// Enter Firebase web API Key
#define API_KEY "AIzaSyBigggABHtgVnZuuvRRP52Mtl0PpeCNIS0"

// Enter Realtime Database URL
#define DATABASE_URL "test-825d4-default-rtdb.asia-
southeast1.firebasedatabase.app/"
#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"
```

```
FirebaseData Firebase_dataObject;
FirebaseAuth auth;
FirebaseConfig config;

String UID;

// Database main path
String database_path;

String temperature_path = "/temperature";
String humidity_path = "/humidity";
String heart_rate_path = "/heart_rate";
// String time_path = "/epoch_time";

// Send new readings every 5 minutes
unsigned long previous_time = 0;
unsigned long Delay = 3000;

bool signupOK = false;

void setup(){
  Serial.begin(115200);

  dht.begin(); // Initialize the DHT sensor
  pulseSensor.analogInput(PulseWire); // Initialize the
Pulse Sensor

  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print('.');
    delay(1000);
  }
  Serial.println(WiFi.localIP());
  Serial.println();

  config.api_key = API_KEY;
  config.database_url = DATABASE_URL;
```

```
  if (Firebase.signUp(&config, &auth, "", "")){
    Serial.println("ok");
    signupOK = true;
  }
  else{
    Serial.printf("%s\n",
config.signer.signupError.message.c_str());
  }

  Firebase.reconnectWiFi(true);
  Firebase_dataObject.setResponseSize(4096);

  config.token_status_callback = tokenStatusCallback;
  config.max_token_generation_retry = 5;

  Firebase.begin(&config,&auth);

  database_path = "/Sensor_Readings";
}

void loop(){
  if (Firebase.ready() && (millis() - previous_time >
Delay || previous_time == 0)) {
    previous_time = millis();
    int heart;

    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();

    if (pulseSensor.sawStartOfBeat()) {
      int myBPM = pulseSensor.getBeatsPerMinute();  .
         heart =
myBPM;
     Serial.println(" HeartBeat");
     Serial.print("BPM: ");
     Serial.println(myBPM);
}
```

```
    FirebaseJson json;
    json.set(temperature_path.c_str(),
String(temperature));
    json.set(humidity_path.c_str(), String(humidity));
    json.set(heart_rate_path.c_str(), String(heart));
    // json.set(time_path.c_str(), String(epoch_time)); //
Add epoch time to JSON data

    String parent_path = database_path;
    Serial.printf("Set JSON...%s\n",
Firebase.RTDB.setJSON(&Firebase_dataObject,
parent_path.c_str(), &json) ? "ok" :
Firebase_dataObject.errorReason().c_str());
  }
}
```

MAIN_ACTIVITY.JAVA

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:background="@color/white"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.appbar.AppBarLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar"
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="?attr/colorPrimary"
            android:minHeight="?attr/actionBarSize"
            android:theme="?attr/actionBarTheme"
            tools:layout_editor_absoluteX="0dp"
            tools:layout_editor_absoluteY="0dp" />

    </com.google.android.material.appbar.AppBarLayout>

    <ImageView
        android:id="@+id/thermostat"
        android:layout_width="88dp"
        android:layout_height="159dp"
        android:layout_marginStart="40dp"
        android:layout_marginTop="68dp"
        android:background="@color/white"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/temp" />

    <TextView
        android:id="@+id/temperature"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="132dp"
        android:text="Temperature"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.369"
        app:layout_constraintStart_toEndOf="@+id/thermosta
t"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:background="@color/white"
        android:id="@+id/heartRate"
        android:layout_width="88dp"
        android:layout_height="159dp"
```

```
        android:layout_marginStart="40dp"
        android:layout_marginTop="28dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/thermost
at"
        app:srcCompat="@drawable/heartratesensor" />

    <TextView
        android:id="@+id/bpm"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="168dp"
        android:text="Heart rate"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.342"
        app:layout_constraintStart_toEndOf="@+id/heartRate
"
        app:layout_constraintTop_toBottomOf="@+id/temperat
ure" />

    <ImageView
        android:background="@color/white"
        android:id="@+id/humidity"
        android:layout_width="88dp"
        android:layout_height="159dp"
        android:layout_marginStart="40dp"
        android:layout_marginTop="32dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/heartRat
e"
        app:layout_constraintVertical_bias="0.019"
        app:srcCompat="@drawable/images" />

    <TextView
        android:id="@+id/humidity_value"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_marginTop="172dp"
        android:text="Humidity"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.342"
        app:layout_constraintStart_toEndOf="@+id/humidity"
        app:layout_constraintTop_toBottomOf="@+id/bpm" />

    <Button
        android:id="@+id/refresh"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="128dp"
        android:text="Refresh"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/humidity
_value"
        app:layout_constraintVertical_bias="0.0" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

SPLASH_SCREEN.XML

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:background="@color/white"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SplashScreenActivity">

    <ImageView
        android:background="@color/white"
        android:id="@+id/imageViewHeart"
```

```
        android:layout_width="230dp"
        android:layout_height="344dp"
        android:layout_centerInParent="true"
        android:src="@drawable/heart_rate_icon" />

    <TextView
        android:id="@+id/textViewWelcome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="WELCOME"
        android:textSize="24sp"
        android:textStyle="bold"
        android:layout_centerHorizontal="true"
        android:layout_below="@id/imageViewHeart"
        android:layout_marginTop="16dp" />

</RelativeLayout>
```

MAIN_ACTIVITY.JAVA

```
package com.example.monitoring;

import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.graphics.Bitmap;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;

import androidx.annotation.NonNull;
```

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.content.res.ResourcesCompat;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;


public class MainActivity extends AppCompatActivity {

    private static final String CHANNEL_ID = "mychannel";
    private static final int NOTIFICATION_ID = 100;


    FirebaseDatabase firebaseDatabase;
    DatabaseReference databaseReference;

    private TextView Temperaturevalue , humidityvalue ,
heartratevalue;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findView
ById(R.id.main), (v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom);
            return insets;
```

```
        });

//        Drawable drawable =
ResourcesCompat.getDrawable(getResources(),R.drawable.imag
es,null);
//        BitmapDrawable bitmapDrawable = (BitmapDrawable)
drawable;
//        Bitmap largeIcon = bitmapDrawable.getBitmap();
//
//        NotificationManager mn =
(NotificationManager)getSystemService(NOTIFICATION_SERVICE
);
//        Notification notification;
//        if (android.os.Build.VERSION.SDK_INT >=
android.os.Build.VERSION_CODES.O) {
//             notification = new
Notification.Builder(this)
//                    .setLargeIcon(largeIcon)
//                    .setSmallIcon(R.drawable.images)
//                    .setContentText("Message")
//                    .setSubText("New Message")
//                    .setChannelId(CHANNEL_ID)
//                    .build();
//            mn.createNotificationChannel(new
NotificationChannel(CHANNEL_ID,"New
channel",NotificationManager.IMPORTANCE_HIGH));
//
//        }else{
//            notification = new
Notification.Builder(this)
//                    .setLargeIcon(largeIcon)
//                    .setSmallIcon(R.drawable.images)
//                    .setContentText("Message")
//                    .setSubText("New Message")
//                    .build();
//        }

//        mn.notify(NOTIFICATION_ID,notification);
```

```
        Toolbar toolbar = (Toolbar)
findViewById(R.id.toolbar);

        // using toolbar as ActionBar
        setSupportActionBar(toolbar);

        firebaseDatabase = FirebaseDatabase.getInstance();
        databaseReference =
firebaseDatabase.getReference("DHT");
        Temperaturevalue = findViewById(R.id.temperature);
        humidityvalue = findViewById(R.id.humidity_value);
        heartratevalue = findViewById(R.id.bpm);

//        getdata();

        Button btnRefresh = findViewById(R.id.refresh);
        btnRefresh.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                getdata();
            }
        });

        initialdata();


    }

    private void initialdata() {
        Temperaturevalue.setText("Temperature : " + "00");
        humidityvalue.setText("Humidity : " + "00");
        heartratevalue.setText("BPM : " + "00");
    }

    private void getdata() {
```

```
        // calling add value event listener method
        // for getting the values from database.
        databaseReference.addValueEventListener(new
ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot
snapshot) {
                // this method is call to get the realtime
                // updates in the data.
                // this method is called when the data is
                // changed in our Firebase console.
                // below line is for getting the data from
                // snapshot of our database.
                String temp_value =
snapshot.child("temperature").getValue(String.class);
                String humidity_value =
snapshot.child("humidity").getValue(String.class);
                String heart_value =
snapshot.child("heartbeat").getValue(String.class);

                int temp = Integer.parseInt(temp_value);
                if(temp>38){

                }


                // after getting the value we are setting
                // our value to our text view in below
line.
                Temperaturevalue.setText("Temperature : "
+ temp_value);
                humidityvalue.setText("Humidity : " +
humidity_value);
                heartratevalue.setText("BPM : " +
heart_value);
            }

            @Override
```

```
            public void onCancelled(@NonNull DatabaseError
error) {
                // calling on cancelled method when we
receive
                // any error or we are not able to get the
data.
                Toast.makeText(MainActivity.this, "Fail to
get data.", Toast.LENGTH_SHORT).show();
            }
        });
    }

    public void getNotification(){
        Drawable drawable =
ResourcesCompat.getDrawable(getResources(),R.drawable.imag
es,null);
        BitmapDrawable bitmapDrawable = (BitmapDrawable)
drawable;
        Bitmap largeIcon = bitmapDrawable.getBitmap();



        NotificationManager mn =
(NotificationManager)getSystemService(NOTIFICATION_SERVICE
);
        Notification notification;
        if (android.os.Build.VERSION.SDK_INT >=
android.os.Build.VERSION_CODES.O) {
            notification = new Notification.Builder(this)
                    .setLargeIcon(largeIcon)
                    .setSmallIcon(R.drawable.images)
                    .setContentText("Message")
                    .setSubText("New Message")
                    .setChannelId(CHANNEL_ID)
                    .build();
            mn.createNotificationChannel(new
NotificationChannel(CHANNEL_ID,"New
channel",NotificationManager.IMPORTANCE_HIGH));
```

```
        }else{
            notification = new Notification.Builder(this)
                    .setLargeIcon(largeIcon)
                    .setSmallIcon(R.drawable.images)
                    .setContentText("Message")
                    .setSubText("New Message")
                    .build();
        }

    }
}
```

SPLASH_SCREEN.JAVA

```
package com.example.monitoring;

import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import
android.view.animation.AccelerateDecelerateInterpolator;
import android.widget.ImageView;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class SplashScreenActivity extends
AppCompatActivity {

    private ImageView imageViewHeart;
    private static final int ANIMATION_DURATION = 5000; //
5 seconds
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_splash_screen);
        ViewCompat.setOnApplyWindowInsetsListener(findView
ById(R.id.main), (v, insets) -> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom);
            return insets;
        });

        imageViewHeart =
findViewById(R.id.imageViewHeart);
        // Create ObjectAnimator for scaling up
        ObjectAnimator scaleUp =
ObjectAnimator.ofFloat(imageViewHeart, "scaleY", 1.2f);
        scaleUp.setDuration(3000); // Duration for scaling
up (3 seconds)
        scaleUp.setInterpolator(new
AccelerateDecelerateInterpolator());

        // Create ObjectAnimator for scaling down
        ObjectAnimator scaleDown =
ObjectAnimator.ofFloat(imageViewHeart, "scaleY", 1.0f);
        scaleDown.setDuration(3000); // Duration for
scaling down (3 seconds)
        scaleDown.setInterpolator(new
AccelerateDecelerateInterpolator());

        // Create AnimatorSet to play the animations
sequentially
        AnimatorSet animatorSet = new AnimatorSet();
        animatorSet.play(scaleUp).before(scaleDown); //
Scale up first, then scale down
```

```
        animatorSet.play(scaleDown).after(3000); // Start
scaling down after 3 seconds

        // Start the animation
        animatorSet.start();

        // Delay transition to MainActivity after
animation duration
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                startActivity(new
Intent(SplashScreenActivity.this, MainActivity.class));
                finish();
            }
        }, ANIMATION_DURATION);

    }
}
```

4.2. IMPLEMENTATION OF SOLUTION

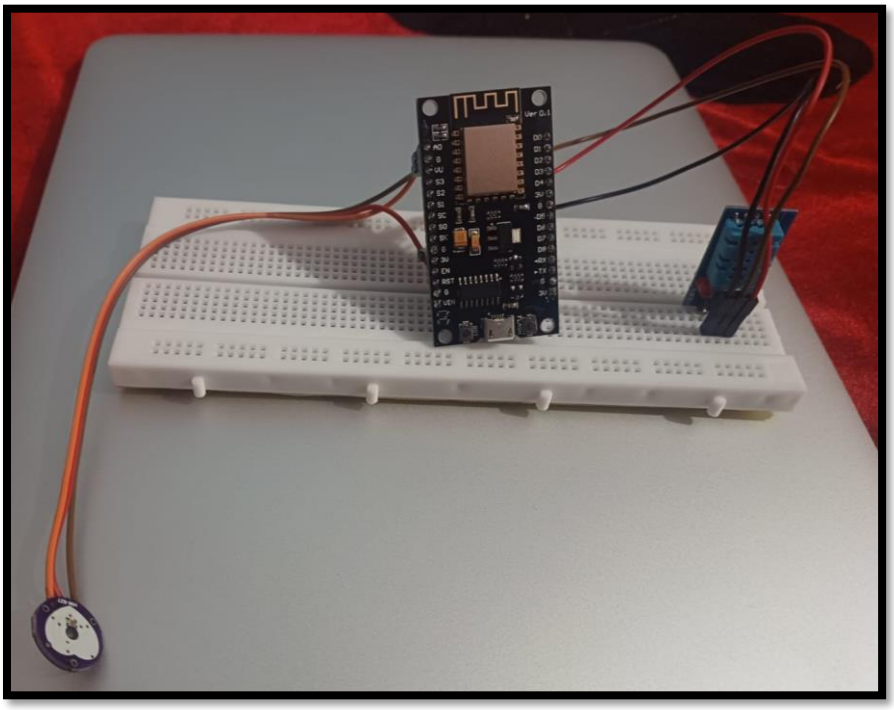


Figure 11: Hardware Setup

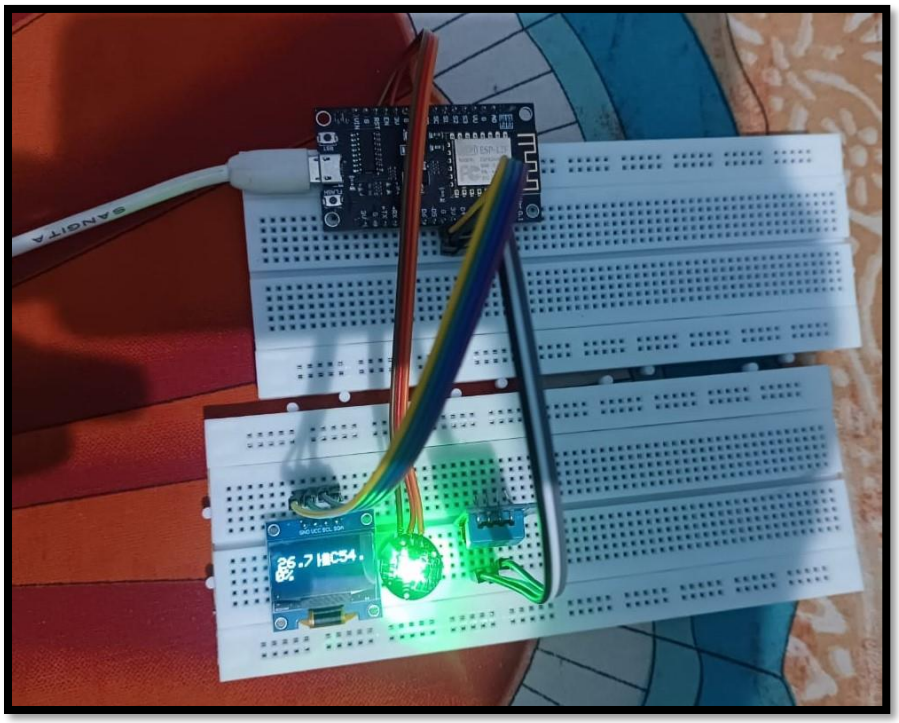


Figure 12: Working Model

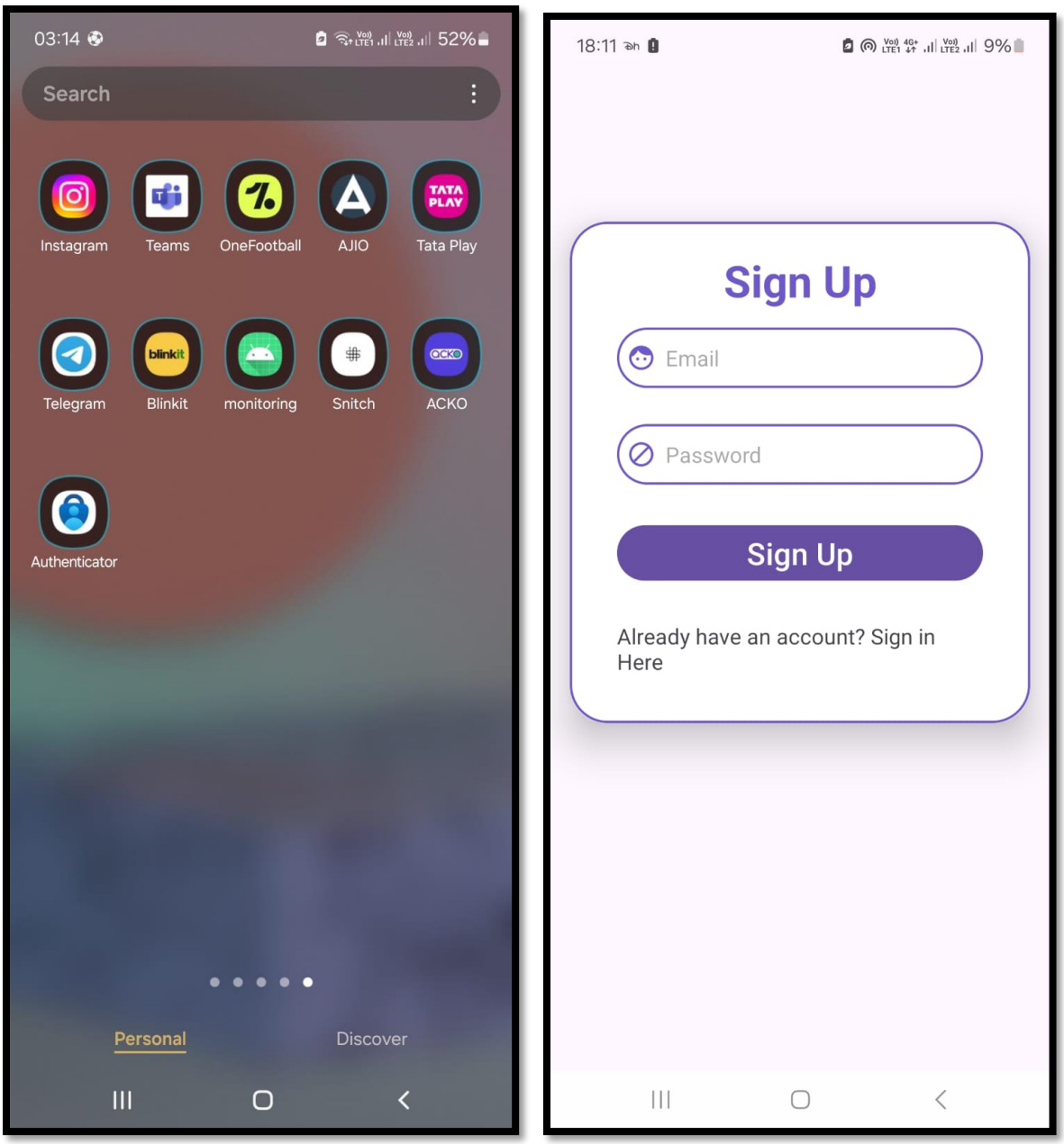


Figure 13, 14: Designed Application (Monitoring), Login Page

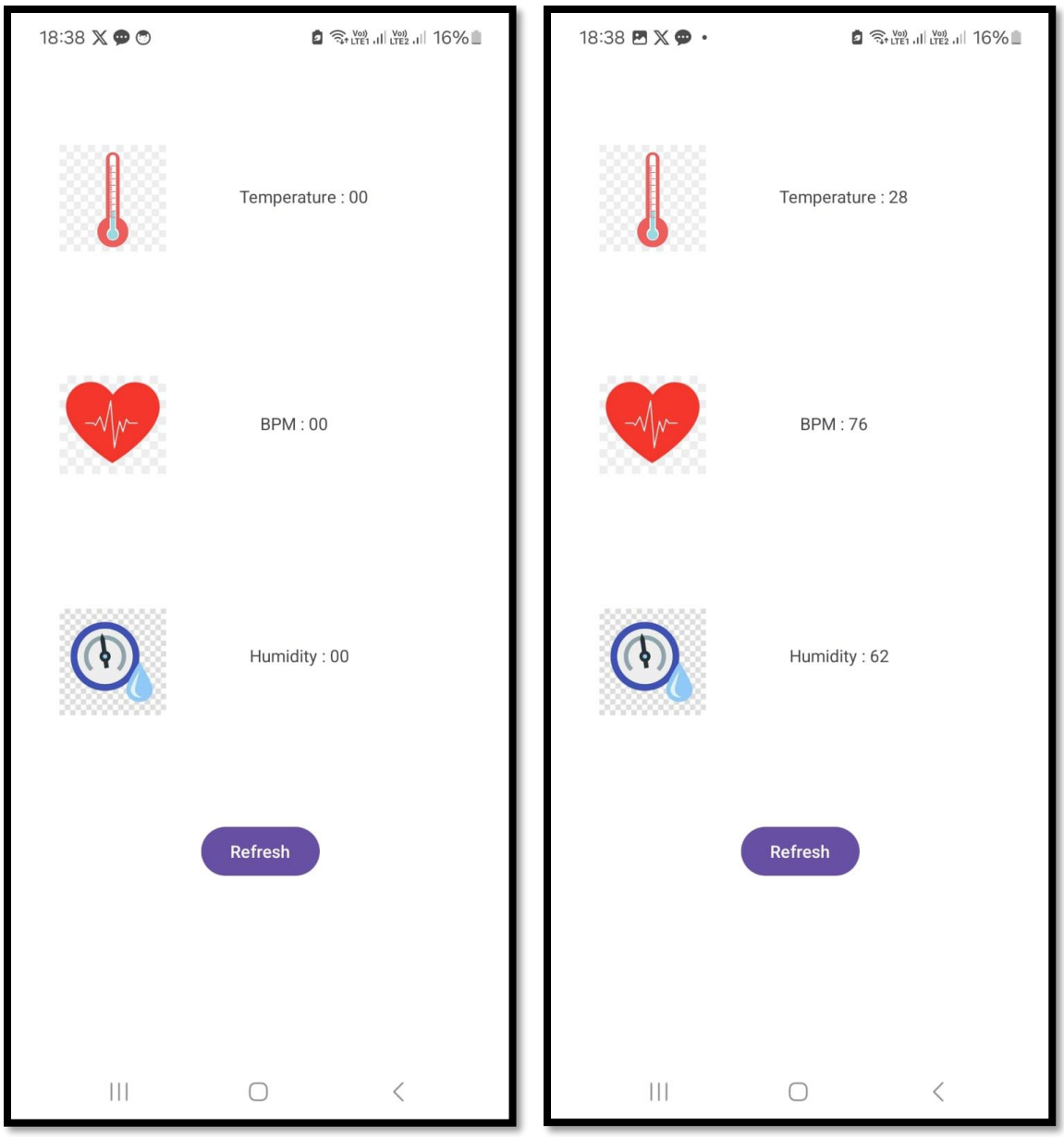


Figure 15, 16: Application Interface and Readings

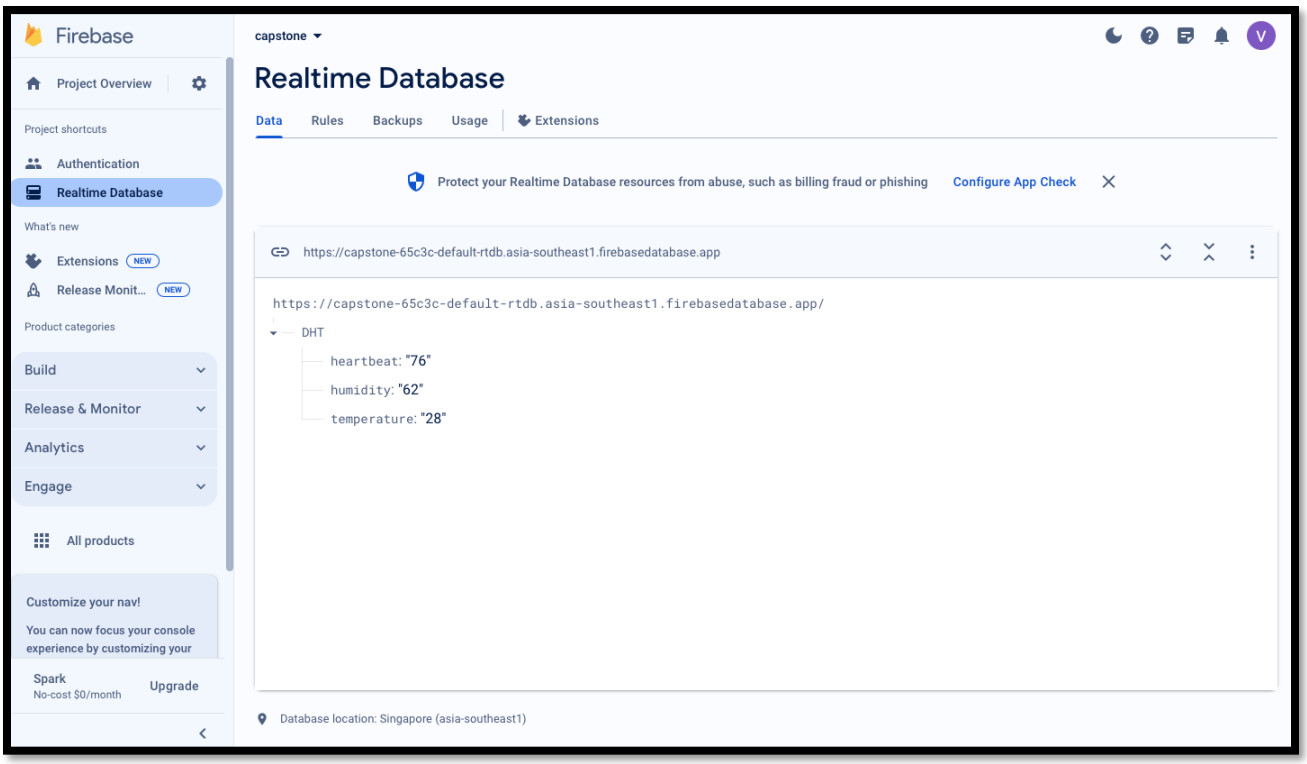


Figure 17: Realtime Database Page

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1. CONCLUSION

The designed system, which was developed with the help of Internet of Things technologies, has effectively shown that it is possible to use mobile applications, cloud computing, and inexpensive hardware components to achieve effective remote health monitoring. The system effectively monitors body temperature, heart rate (BPM), and ambient humidity levels by merging the ESP8266 Node MCU, temperature sensor, humidity sensor, and heart rate sensor. Real-time monitoring, customizable profile management, and timely notifications are made possible by the user-friendly Android mobile app and the wireless data transmission to the Firebase Realtime Database. The system has demonstrated its accuracy, dependability, and usefulness via rigorous testing and user input, supporting proactive disease management, early health issue diagnosis, and preventative care efforts. This initiative sets the stage for future developments in Internet of Things (IoT)-based healthcare solutions, giving people the ability to take charge of their health and assisting medical professionals in providing high-quality treatment.

5.2. FUTURE WORK

While the developed Smart Health Monitoring System successfully demonstrates the feasibility of remote vital sign monitoring using IoT technologies, there are several avenues for future work to enhance its capabilities further. Integrating additional sensors for tracking parameters like blood pressure, oxygen saturation levels, or glucose levels can provide a more comprehensive health monitoring solution. Incorporating machine learning algorithms for data analysis and pattern recognition could enable early detection of potential health risks and predictive analytics. Furthermore, exploring the integration of wearable devices or smart home sensors could enhance the system's ability to monitor users' daily activities and environmental conditions, contributing to a holistic approach to health management. Additionally, investigating interoperability standards and seamless integration with existing electronic health record systems could facilitate better collaboration between patients and healthcare providers, leading to improved patient outcomes and more personalized care delivery.

TENTATIVE CHAPTER PLAN FOR THE PROPOSED WORK

CHAPTER 1: INTRODUCTION

This chapter will cover the overview of the system.

CHAPTER 2: LITERATURE REVIEW/BACKGROUND STUDY

This chapter includes the literature available for our system and the findings of the researchers will be highlighted which will become the basis of the current implementation.

CHAPTER 3: DESIGN FLOW/PROCESS/METHODOLOGY

This chapter will cover the technical details of the proposed approach and the techniques used for the implementation.

CHAPTER 4: RESULT ANALYSIS AND VALIDATION

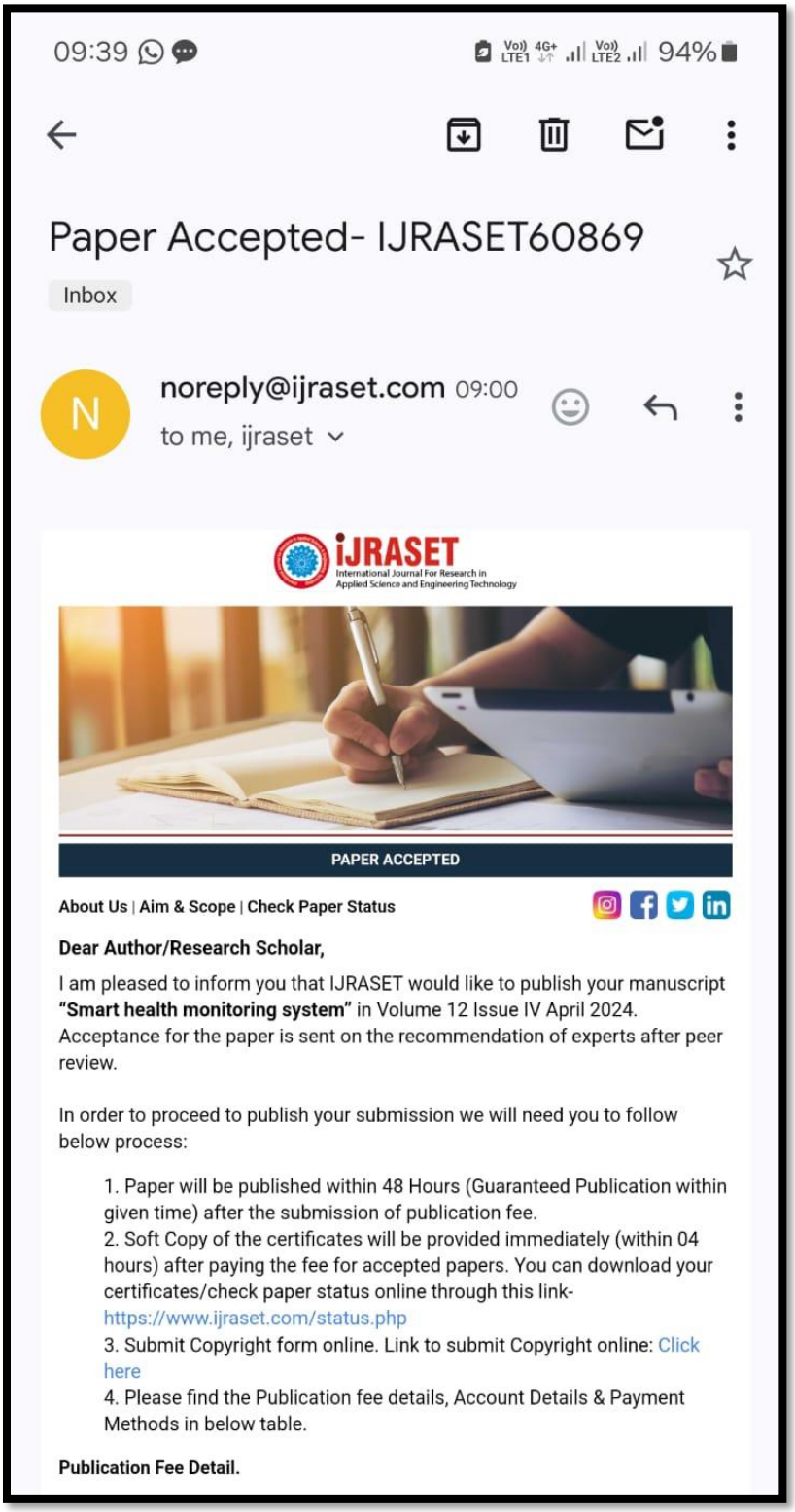
This chapter will include the code and output or a glimpse of the system.

CHAPTER 5: CONCLUSION AND FUTURE SCOPE

The major findings of the work will be presented in this chapter. Also, directions for extending the current study will be discussed.

PAPER ACCEPTANCE PROOF

DOI: https://doi.org/10.22214/ijraset.2024.60869



09:39 94%

Paper Accepted- IJRASET60869

Inbox

noreply@ijraset.com 09:00

to me, ijraset

iJRASET

International Journal For Research in Applied Science and Engineering Technology

PAPER ACCEPTED

About Us | Aim & Scope | Check Paper Status

Dear Author/Research Scholar,

I am pleased to inform you that IJRASET would like to publish your manuscript **"Smart health monitoring system"** in Volume 12 Issue IV April 2024. Acceptance for the paper is sent on the recommendation of experts after peer review.

In order to proceed to publish your submission we will need you to follow below process:

1. Paper will be published within 48 Hours (Guaranteed Publication within given time) after the submission of publication fee.
2. Soft Copy of the certificates will be provided immediately (within 04 hours) after paying the fee for accepted papers. You can download your certificates/check paper status online through this link- https://www.ijraset.com/status.php
3. Submit Copyright form online. Link to submit Copyright online: Click here
4. Please find the Publication fee details, Account Details & Payment Methods in below table.

Publication Fee Detail.

REFERENCES

[1] Mohan, Madhan & Pichandi, Sathya. (2019). Smart Health Monitoring System through IOT.

[2] Valsalan, Prajoona & Tariq, Ahmed & Hussain, Ali. (2020). IOT BASED HEALTH MONITORING SYSTEM. 2020. 10.31838/jcr.07.04.137.

[3] Khan, Tarannum & Chattopadhyay, Manju. (2017). Smart health monitoring system. 1-6. 10.1109/ICOMICON.2017.8279142.

[4] Abdulmalek, Suliman & bin, Abdul Nasir & Al-Areeqi, Waheb & Almuhaya, Mukarram & Bairagi, Anupam & Khan, Md. Al-Masrur & Kee, Seong-Hoon. (2022). IoT-Based Healthcare-Monitoring System towards Improving Quality of Life: A Review. 10.3390/healthcare10101993.

[5] Khan, Mohammad & Alanazi, Turki & Albraikan, Amani & Almalki, Faris. (2022). IoT-Based Health Monitoring System Development and Analysis. Security and Communication Networks. 2022. 1-11. 10.1155/2022/9639195.

[6] Gajare, Mansi & Dani, Manas & Deshmukh, Payal & Chaudhari, Pritesh. (2021). IOT based Health Monitoring System.

[7] Sultana, Salma & Rahman, Sadia & Rahman, Md & Chakraborty, Narayan & Hasan, Tanveer. (2021). An IoT Based Integrated Health Monitoring System. 549-554. 10.1109/ICCCA52192.2021.9666412.

[8] Sekhar, Y. Ravi. "Android based health care monitoring system." *Department Of ECE, Vignan's University, Vadlamudi, India*.

[9] D. A. M. Budida and R. S. Mangrulkar, "Design and implementation of smart HealthCare system using IoT," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, 2017, pp. 1-7, doi: 10.1109/ICIIECS.2017.8275903.

[10] H. Al-Hamadi and I. R. Chen, "Trust-Based Decision Making for Health IoT Systems," in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1408-1419, Oct. 2017, doi: 10.1109/JIOT.2017.2736446.

[11] Rahaman, A., Islam, M.M., Islam, M.R., Sadi, M.S., Nooruddin, S. (2019). Developing IoT based smart health monitoring systems: A review. Revue d'Intelligence Artificielle, Vol. 33, No. 6, pp. 435-440. https://doi.org/10.18280/ria.330605

[12] Joyia, G.J., Liaqat, R.M., Farooq, A. and Rehman, S., 2017. Internet of medical things (IoMT): Applications, benefits and future challenges in healthcare domain. *J. Commun.*, *12*(4), pp.240-247.

[13] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain and K. -S. Kwak, "The Internet of Things for Health Care: A Comprehensive Survey," in IEEE Access, vol. 3, pp. 678-708, 2015, doi: 10.1109/ACCESS.2015.2437951.

[14] K. Perumal, M. Manohar, A Survey on Internet of Things: Case Studies, Applications, and Future Directions, In Internet of Things: Novel Advances and Envisioned Applications, Springer International Publishing, (2017) 281-297.

[15] Islam, M.M., Rahaman, A. & Islam, M.R. Development of Smart Healthcare Monitoring System in IoT Environment. *SN COMPUT. SCI.* **1**, 185 (2020). https://doi.org/10.1007/s42979-020-00195-y

[16] Azzawi, Mustafa & Hassan, Rosilah & Abu Bakar, Khairul Azmi. (2016). A Review on Internet of Things (IoT) in Healthcare. International Journal of Applied Engineering Research. 11. 10216-10221.

[17] Shubham Banka, Isha Madan and S.S. Saranya, Smart Healthcare Monitoring using IoT. International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 15, pp. 11984-11989, 2018. 4.

[18] Ennafiri, Mohamed & Mazri, Tomader. (2020). INTERNET OF THINGS FOR SMART HEALTHCARE: A REVIEW ON A POTENTIAL IOT BASED SYSTEM AND TECHNOLOGIES TO CONTROL COVID-19 PANDEMIC. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. XLIV-4/W3-2020. 219-225. 10.5194/isprs-archives-XLIV-4-W3-2020-219-2020.

[19] M. Hummady, Muna & Fadhil, Heba. (2023). Smart Healthcare Medical Bracelet using the Internet of Things. 10.1109/ICCA56443.2022.10039622.

[20] M. Asaduzzaman Miah, Mir Hussain Kabir, M. Siddiqur Rahman Tanveer and M. A. H. Akhand, "Continuous heart rate and body temperature monitoring system using Arduino UNO and Android device," 2015 2nd International Conference on Electrical Information and Communication Technologies (EICT), Khulna, Bangladesh, 2015, pp. 183-188, doi: 10.1109/EICT.2015.7391943.

[21] Taştan, Mehmet. (2018). IoT Based Wearable Smart Health Monitoring System. Celal Bayar Üniversitesi Fen Bilimleri Dergisi. 14. 343-350. 10.18466/cbayarfbe.451076.

[22] Mohammed, B. G., & Hasan, D. S. (2023). Smart Healthcare Monitoring System Using IoT. *International Journal of Interactive Mobile Technologies (iJIM)*, *17*(01), pp. 141–152.

[23] M. M. Masud, M. Adel Serhani and A. N. Navaz, "Resource-Aware Mobile-Based Health Monitoring," in IEEE Journal of Biomedical and Health Informatics, vol. 21, no. 2, pp. 349-360, March 2017, doi: 10.1109/JBHI.2016.2525006.

[24] Poongodi, Manoharan, et al. "Smart healthcare in smart cities: wireless patient monitoring system using IoT." *The Journal of Supercomputing* (2021): 1-26.

[25] Islam, Md Milon, Ashikur Rahaman, and Md Rashedul Islam. "Development of smart healthcare monitoring system in IoT environment." *SN computer science* 1 (2020): 1-11.

[26] Valsalan, Prajoona, Tariq Ahmed Barham Baomar, and Ali Hussain Omar Baabood. "IoT based health monitoring system." *Journal of critical reviews* 7.4 (2020): 739-743.

[27] Kadhim, Kadhim Takleef, et al. "An overview of patient's health status monitoring system based on internet of things (IoT)." *Wireless Personal Communications* 114.3 (2020): 2235-2262.

[28] Rani, Sita, et al. "IoT equipped intelligent distributed framework for smart healthcare systems." *Towards the Integration of IoT, Cloud and Big Data: Services, Applications and Standards*. Singapore: Springer Nature Singapore, 2023. 97-114.

[29] Akkaş, M. Alper, Radosveta Sokullu, and H. Ertürk Çetin. "Healthcare and patient monitoring using IoT." *Internet of Things* 11 (2020): 100173.

[30] Ahad, Abdul, et al. "Technologies trend towards 5G network for smart health-care using IoT: A review." *Sensors* 20.14 (2020): 4047.

[31] Ali, Farman, et al. "A smart healthcare monitoring system for heart disease prediction based on ensemble deep learning and feature fusion." *Information Fusion* 63 (2020): 208-222.

[32] Papa, Armando, et al. "E-health and wellbeing monitoring using smart healthcare devices: An empirical investigation." *Technological Forecasting and Social Change* 153 (2020): 119226.

[33] Taiwo, Olutosin, and Absalom E. Ezugwu. "Smart healthcare support for remote patient monitoring during covid-19 quarantine." *Informatics in medicine unlocked* 20 (2020): 100428.

[34] Taiwo, Olutosin, and Absalom E. Ezugwu. "Smart healthcare support for remote patient monitoring during covid-19 quarantine." *Informatics in medicine unlocked* 20 (2020): 100428.

[35] Jacob Rodrigues, Mariana, Octavian Postolache, and Francisco Cercas. "Physiological and behavior monitoring systems for smart healthcare environments: A review." *Sensors* 20.8 (2020): 2186.

[36] Ghazal, Taher M., et al. "IoT for smart cities: Machine learning approaches in smart healthcare—A review." *Future Internet* 13.8 (2021): 218.

[37] Ghazal, Taher M., et al. "IoT for smart cities: Machine learning approaches in smart healthcare—A review." *Future Internet* 13.8 (2021): 218.

[38] Selvaraj, Sureshkumar, and Suresh Sundaravaradhan. "Challenges and opportunities in IoT healthcare systems: a systematic review." *SN Applied Sciences* 2.1 (2020): 139.

[39] Greco, Luca, et al. "Trends in IoT based solutions for health care: Moving AI to the edge." *Pattern recognition letters* 135 (2020): 346-353.

[40] Ullo, Silvia Liberata, and Ganesh Ram Sinha. "Advances in smart environment monitoring systems using IoT and sensors." *Sensors* 20.11 (2020): 3113.

[41] Li, Wei, et al. "A comprehensive survey on machine learning-based big data analytics for IoT-enabled smart healthcare system." *Mobile networks and applications* 26 (2021): 234-252.

[42] Ali, Farman, et al. "An intelligent healthcare monitoring framework using wearable sensors and social networking data." *Future Generation Computer Systems* 114 (2021): 23-43.

[43] Tuli, Shreshth, et al. "HealthFog: An ensemble deep learning based Smart Healthcare System for Automatic Diagnosis of Heart Diseases in integrated IoT and fog computing environments." *Future Generation Computer Systems* 104 (2020): 187-200.

[44] Alshehri, Fatima, and Ghulam Muhammad. "A comprehensive survey of the Internet of Things (IoT) and AI-based smart healthcare." *IEEE access* 9 (2020): 3660-3678.

[45] Zhang, Quan, et al. "Wearable triboelectric sensors enabled gait analysis and waist motion capture for IoT-based smart healthcare applications." *Advanced Science* 9.4 (2022): 2103694.

[46] Vedaei, Seyed Shahim, et al. "COVID-SAFE: An IoT-based system for automated health monitoring and surveillance in post-pandemic life." *IEEE access* 8 (2020): 188538-188551.

[47] Pasika, Sathish, and Sai Teja Gandla. "Smart water quality monitoring system with cost-effective using IoT." *Heliyon* 6.7 (2020).

[48] Mujawar, Mubarak A., et al. "Nano-enabled biosensing systems for intelligent healthcare: towards COVID-19 management." *Materials Today Chemistry* 17 (2020): 100306.

[49] Muhammad, Ghulam, et al. "A comprehensive survey on multimodal medical signals fusion for smart healthcare systems." *Information Fusion* 76 (2021): 355-375.

[50] Tripathi, Gautami, Mohd Abdul Ahad, and Sara Paiva. "S2HS-A blockchain based approach for smart healthcare system." *Healthcare*. Vol. 8. No. 1. Elsevier, 2020.

[51] Kumar, Adarsh, et al. "A novel smart healthcare design, simulation, and implementation using healthcare 4.0 processes." *IEEE access* 8 (2020): 118433-118471.

[52] Zaabar, Bessem, et al. "HealthBlock: A secure blockchain-based healthcare data management system." *Computer Networks* 200 (2021): 108500.

[53] Jamil, Faisal, et al. "Towards a remote monitoring of patient vital signs based on IoT-based blockchain integrity management platforms in smart hospitals." *Sensors* 20.8 (2020): 2195.

[54] Mansour, Romany Fouad, et al. "Artificial intelligence and internet of things enabled disease diagnosis model for smart healthcare systems." *IEEE Access* 9 (2021): 45137-45146.

[55] Javaid, Mohd, and Ibrahim Haleem Khan. "Internet of Things (IoT) enabled healthcare helps to take the challenges of COVID-19 Pandemic." *Journal of oral biology and craniofacial research* 11.2 (2021): 209-214.

[56] Mbunge, Elliot, Benhildah Muchemwa, and John Batani. "Sensors and healthcare 5.0: transformative shift in virtual care through emerging digital health technologies." *Global Health Journal* 5.4 (2021): 169-177.

[57] Serhani, Mohamed Adel, et al. "ECG monitoring systems: Review, architecture, processes, and key challenges." *Sensors* 20.6 (2020): 1796.

[58] Pradhan, Bikash, Saugat Bhattacharyya, and Kunal Pal. "IoT-based applications in healthcare devices." *Journal of healthcare engineering* 2021 (2021): 1-18.

[59] Sangeethalakshmi, K., U. Preethi, and S. Pavithra. "Patient health monitoring system using IoT." *Materials Today: Proceedings* 80 (2023): 2228-2231.

[60] Wu, Xingdong, et al. "Internet of things-enabled real-time health monitoring system using deep learning." *Neural Computing and Applications* (2023): 1-12.

[61] Saranya, K. Durga, et al. "IoT-based health monitoring system using beaglebone black with optical sensor." *Journal of Optical Communications* 44.3 (2023): 359-365.

[62] Rani, Sita, et al. "IoT equipped intelligent distributed framework for smart healthcare systems." *Towards the Integration of IoT, Cloud and Big Data: Services, Applications and Standards*. Singapore: Springer Nature Singapore, 2023. 97-114.

[63] Rahman, Muhammad Zia, et al. "An intelligent health monitoring and diagnosis system based on the internet of things and fuzzy logic for cardiac arrhythmia COVID-19 patients." *Computers in Biology and Medicine* 154 (2023): 106583.

[64] Hosseinzadeh, Mehdi, et al. "An elderly health monitoring system based on biological and behavioral indicators in internet of things." *Journal of Ambient Intelligence and Humanized Computing* (2023): 1-11.

[65] Shrivastava, Anurag, Midhun Chakkaravarthy, and Mohd Asif Shah. "Health Monitoring based Cognitive IoT using Fast Machine Learning Technique." *International Journal of Intelligent Systems and Applications in Engineering* 11.6s (2023): 720-729.

[66] Nasser, Nidal, et al. "A smart healthcare framework for detection and monitoring of COVID-19 using IoT and cloud computing." *Neural Computing and Applications* (2023): 1-15.

[67] Dahan, Fadl, et al. "A smart IoMT based architecture for E-healthcare patient monitoring system using artificial intelligence algorithms." *Frontiers in Physiology* 14 (2023): 1125952.

[68] Munnangi, Ashok Kumar, et al. "Survival study on deep learning techniques for IoT enabled smart healthcare system." *Health and Technology* 13.2 (2023): 215-228.

[69] Akhbarifar, Samira, et al. "A secure remote health monitoring model for early disease diagnosis in cloud-based IoT environment." *Personal and Ubiquitous Computing* 27.3 (2023): 697-713.

[70] Boopathi, Sampath. "Internet of things-integrated remote patient monitoring system: Healthcare application." *Dynamics of Swarm Intelligence Health analysis for the next generation*. IGI Global, 2023. 137-161.

[71] Boopathi, Sampath. "Internet of things-integrated remote patient monitoring system: Healthcare application." *Dynamics of Swarm Intelligence Health analysis for the next generation*. IGI Global, 2023. 137-161.

[72] Alsudani, Mustafa Qahtan, et al. "RETRACTED ARTICLE: Smart logistics with IoT-based enterprise management system using global manufacturing." *Journal of Combinatorial Optimization* 45.2 (2023): 57.

[73] Sharif, Zubair, et al. "Priority-based task scheduling and resource allocation in edge computing for health monitoring system." *Journal of King Saud University-Computer and Information Sciences* 35.2 (2023): 544-559.