Remote Patient Monitoring System

[1]Dr. J. Narendra Babu, [2]Dr. Deepak S Sakkari, [3]Simmi Kumari, [4]Sinchana M, [5]Sona Vincent, [6]Srashti R Poojary, [7]Srushti

[1]Professor, Department of Data Science, Sapthagiri NPS University

[2]Professor & Director, Department of Data Science, Sapthagiri NPS University

[3,4,5,6,7] Student, Department of Data Science, Sapthagiri NPS University, India.

***Abstract*—The growing demand for continuous healthcare monitoring has led to the development of IoT-based remote patient monitoring systems that allow doctors to observe patient health parameters without requiring physical presence. This paper presents a low-cost, efficient, and real-time remote patient monitoring system developed using the ESP8266 Node MCU microcontroller integrated with a MAX30100 pulse oximeter sensor and a DHT11 temperature and humidity sensor. The system continuously collects vital health parameters including heart rate, blood oxygen saturation levels, body temperature, and humidity. The collected data is transmitted wirelessly through Wi-Fi communication to a Firebase cloud database, which is further displayed on a web-based dashboard developed using React.js. An AI-powered healthcare assistant is also integrated into the dashboard to provide intelligent health analysis and recommendations. The system incorporates an emergency alert mechanism using a buzzer that activates when sensor readings cross predefined thresholds. The proposed system is particularly beneficial for elderly patients, individuals with chronic illnesses, and patients who require continuous health observation. The integration of embedded systems, cloud computing, and artificial intelligence in this project demonstrates a practical and scalable approach to modern smart healthcare development.**

***Index Terms*—Remote Patient Monitoring, Internet of Things (IoT), ESP8266, NodeMCU, MAX30100, DHT11, Firebase, React.js, Healthcare Monitoring, Wireless Communication, Artificial Intelligence, Cloud Integration.**

I. INTRODUCTION

1.1 Overview

The rapid advancement of Internet of Things technology has opened new possibilities in the healthcare domain, enabling real-time monitoring of patients from remote locations. Traditional healthcare systems require patients to visit hospitals frequently for routine health checkups,

which is inconvenient, expensive, and sometimes medically risky for elderly or critically ill patients. Remote Patient Monitoring (RPM) systems address this challenge by providing continuous health tracking through wearable and embedded sensor devices that transmit data wirelessly to healthcare providers.

The system proposed in this paper utilizes the ESP8266 NodeMCU, a low-cost and highly capable Wi-Fi enabled microcontroller, along with two biomedical sensors namely the MAX30100 pulse oximeter and the DHT11 temperature sensor.

1.2 Motivation

The motivation behind developing this system stems from the increasing healthcare burden on hospitals and the growing elderly population that requires continuous monitoring. Studies indicate that a significant percentage of hospital readmissions occur due to lack of timely monitoring after discharge. An affordable IoT-based solution that provides real-time alerts

1.3 Objectives

The primary objectives of this project are as follows:

- To develop an IoT-based patient monitoring system using ESP8266 NodeMCU and biomedical sensors.
- To monitor vital health parameters including heart rate, SpO_2, body temperature, and humidity in real time.
- To transmit sensor data wirelessly to Firebase cloud database using Wi-Fi communication.
- To develop a web-based dashboard using React.js for remote data visualization.
- To integrate Firebase Authentication for secure doctor login and account management.
- To incorporate an AI-powered healthcare assistant for intelligent health query responses.
- To implement an emergency buzzer alert system for critical health condition detection.

II. LITERATURE SURVEY

The development of remote healthcare monitoring systems has gained significant attention in recent years owing to the widespread adoption of IoT devices and cloud computing technologies. Several research works have explored the use of low-cost microcontrollers and biomedical sensors for continuous health monitoring applications.

Early research by Rajendra et al. demonstrated the feasibility of using Arduino-based systems with pulse oximeter sensors to monitor heart rate and oxygen saturation levels in clinical environments. However, these systems lacked wireless communication capability and cloud integration, limiting their application to hospital settings only.

The introduction of ESP8266 NodeMCU as a Wi-Fi enabled microcontroller opened new possibilities for wireless health monitoring systems. Studies by Kumar and Sharma explored ESP8266-based temperature and humidity monitoring systems and demonstrated reliable data

transmission through MQTT and HTTP protocols to IoT cloud platforms such as ThingSpeak and Firebase.

Research published in IEEE conferences and journals on biomedical engineering has highlighted the MAX30100 sensor as an effective and reliable module for non-invasive heart rate and SpO_2 measurement. The sensor uses photoplethysmography principles to measure blood oxygen levels and pulse rate, making it suitable for continuous patient monitoring applications.

Cloud-integrated patient monitoring systems using Firebase Realtime Database have been studied and demonstrated in multiple research works. Firebase offers real-time data synchronization, secure authentication, and easy integration with web and mobile applications, making it a suitable choice for healthcare IoT systems

III. PROPOSED SYSTEM

3.1 System Architecture

The proposed Remote Patient Monitoring System follows a layered architecture that separates hardware, communication, cloud, and application components for better modularity and scalability. The overall system architecture is divided into four main layers as described below.

The Sensor Layer forms the physical data acquisition component of the system. The MAX30100 pulse oximeter sensor continuously measures the patient's heart rate and blood oxygen saturation through the finger placement method. The DHT11 sensor simultaneously monitors ambient body temperature and humidity levels. Both sensors are connected to the ESP8266 Node MCU through appropriate GPIO pins and communicate via I2C and digital signal protocols respectively.

The Processing and Communication Layer is handled by the ESP8266 NodeMCU microcontroller, which reads data from the connected sensors, processes the raw sensor values, and checks them against predefined health threshold values. If any reading exceeds the safe range, the buzzer alert is activated immediately. The processed data is then transmitted wirelessly to the Firebase Real time Database through the built-in Wi-Fi module of the ESP8266 using HTTP communication.

The Cloud Layer consists of Firebase Realtime Database and Firebase Authentication services. The database stores all incoming sensor readings in real time and organizes data under patient-specific nodes. Firebase Authentication provides secure login and registration functionality for doctors and healthcare personnel accessing the monitoring dashboard.

The Application Layer is a web-based dashboard developed using React.js that fetches live patient data from Firebase and displays it in a visually organized manner. The dashboard includes patient information cards, live sensor reading displays, health alert notifications, doctor notes functionality, medical report sections, and an AI-powered healthcare assistant chat interface.

3.2 Hardware Components

The hardware components used in the system and their respective roles are described below:

Component	Specification	Function
ESP8266 NodeMCU	Wi-Fi MCU, 80MHz, 3.3V	Main controller and wireless communication
MAX30100 Sensor	I2C, 1.8V-3.3V, IR+Red LED	Heart rate and SpO_2 measurement
DHT11 Sensor	Digital, 3.3V-5V supply	Temperature and humidity monitoring
Buzzer	Active buzzer, 3.3V	Emergency audio alert generation
Breadboard	830 tie points	Circuit prototyping and connections
Jumper Wires	Male-to-male, Male-to-female	Component interconnections
USB Cable	Micro USB type	Power supply and code uploading

Table 1: Hardware Components and Specifications

3.3 Circuit Connections

The hardware components are assembled on a breadboard using jumper wires. The MAX30100 sensor communicates with the ESP8266 through the I2C protocol, with SDA connected to pin D2 and SCL connected to pin D1. The sensor is powered through the 3.3V output pin of the ESP8266. The DHT11 sensor's data pin is connected to pin D4, while the buzzer positive terminal is connected to pin D5. All components share a common ground connection through the GND pin of the ESP8266.

3.4 Software Components

The software stack of the system is divided into firmware for the ESP8266 and a frontend web application. The ESP8266 firmware is written in embedded C using the Arduino IDE environment with the following libraries: ESP8266WiFi for wireless communication, FirebaseESP8266 by Mobizt for Firebase integration, DHT sensor library by Adafruit for DHT11 interfacing, and MAX30100 PulseOximeter library for sensor communication.

The web dashboard is developed using React.js as the frontend framework with Firebase SDK for real-time database integration and authentication.. The entire application is structured as a single-page application with multiple pages including a home page, login page, signup page, dashboard, and AI assistant page.

3.5 OOP Concepts Implemented

The software architecture of the system incorporates fundamental object-oriented programming concepts as described below:

Encapsulation: Patient health data including heart rate, temperature, and SpO_2 values are encapsulated within dedicated state variables in the React components, ensuring that data is managed and accessed through defined state management methods only.

Inheritance: Sensor data processing functions follow a structured hierarchy where specific sensor handler functions inherit common read and validate logic from generalized data processing routines.

Polymorphism: The alert handling functions in the ESP8266 firmware can process different types of sensor readings using a unified alert checking interface, allowing the same function structure to handle multiple sensor types.

Abstraction: Complex Firebase communication, sensor interfacing, and API interaction processes are abstracted through dedicated function modules, providing a simplified interface for the main application logic.

3.6 System Workflow

The operational workflow of the system follows a continuous cycle. When the ESP8266 is powered on, it initializes all connected sensors and establishes a Wi-Fi connection to the configured network. Once connected, it authenticates with the Firebase cloud database and begins the sensor reading cycle. Every five seconds, the system reads heart rate and SpO_2 values from the MAX30100 sensor and temperature and humidity values from the DHT11 sensor. The readings are validated and checked against predefined health thresholds. If any critical values are detected, the buzzer is activated immediately to alert nearby caregivers. All sensor readings are simultaneously pushed to the Firebase Realtime Database.

IV. RESULTS

The system was tested under controlled conditions with the sensors properly connected and the ESP8266 programmed with the developed firmware. The Serial Monitor of the Arduino IDE was used to observe real-time sensor readings and Firebase communication status during testing.

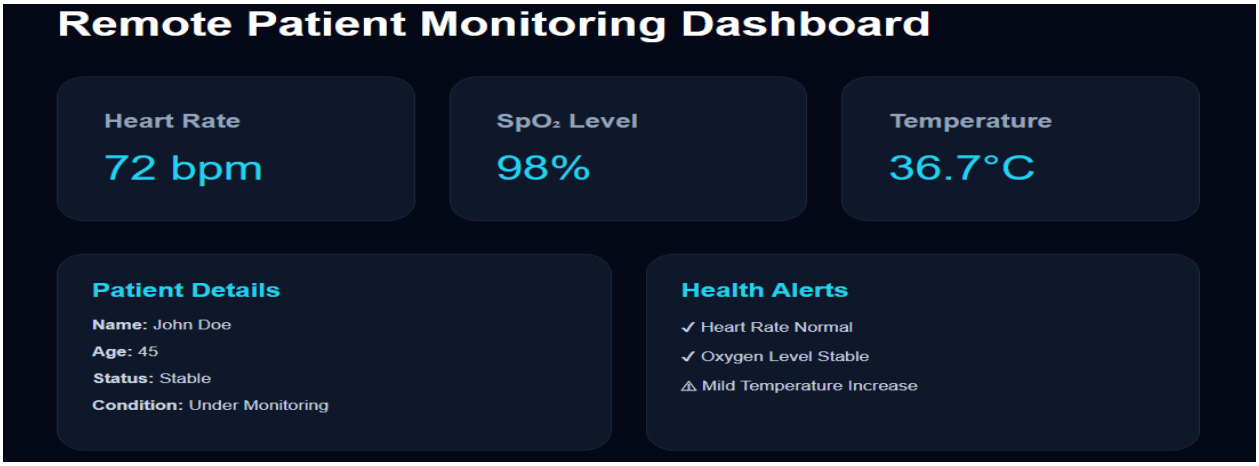
The DHT11 sensor consistently provided accurate temperature readings within the range of 36 to 37.5 degrees Celsius during testing, with humidity values recorded between 55 and 70 percent. The MAX30100 sensor successfully measured heart rate values in the range of 68 to 82 beats per minute and SpO_2 levels above 95 percent during normal test conditions, which fall within the accepted healthy ranges for an adult patient.

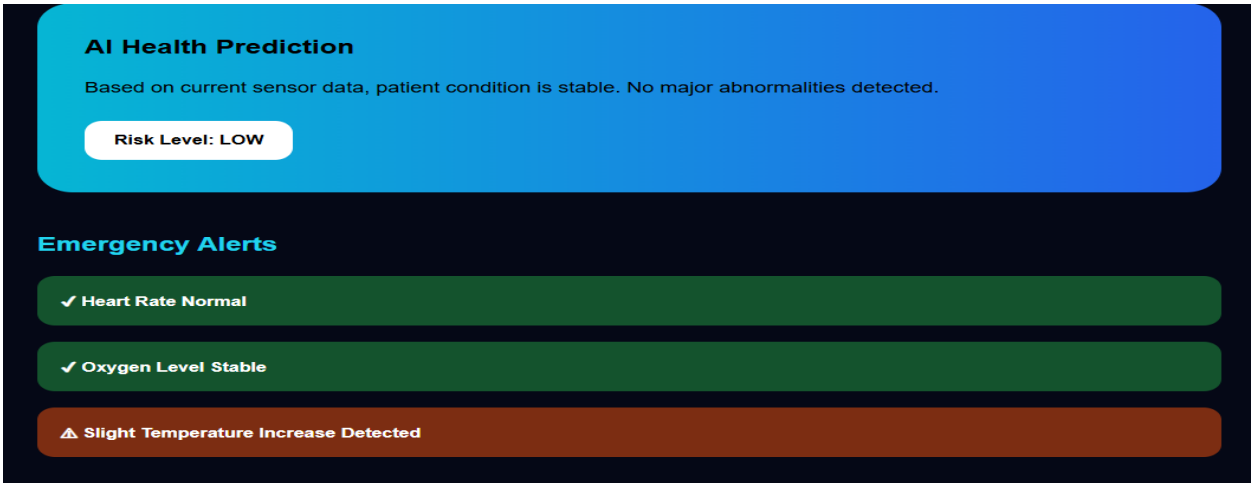
The Firebase data transmission was verified to operate with a latency of approximately two to four seconds from sensor reading to dashboard display, demonstrating near real-time performance suitable for patient monitoring applications. The React.js dashboard successfully

displayed all live sensor readings, patient information, health alerts, and AI chat responses without any significant performance issues.

The emergency buzzer alert was tested by simulating abnormal sensor values, and the system correctly triggered the buzzer within the same reading cycle in which the threshold was exceeded. The Firebase Authentication system successfully restricted dashboard access to registered users only, ensuring secure access to patient data.

The AI healthcare assistant integrated into the dashboard responded to health-related queries with relevant and informative answers, demonstrating the potential of AI integration in enhancing the usability and intelligence of the monitoring system.





V. CONCLUSION

This paper presented the design, development, and implementation of an IoT-based Remote Patient Monitoring System using ESP8266 NodeMCU, MAX30100 pulse oximeter sensor, and DHT11 temperature sensor. The system successfully demonstrated real-time acquisition, wireless transmission, and cloud-based visualization of patient health parameters including heart rate, blood oxygen saturation, body temperature, and humidity.

The integration of Firebase Realtime Database provided reliable and fast cloud storage and synchronization, while the React.js dashboard offered a user-friendly interface for remote monitoring. The inclusion of Firebase Authentication ensured secure access to sensitive patient health data. The AI-powered healthcare assistant further enhanced the system by providing intelligent health query responses directly within the monitoring dashboard.

The proposed system offers a practical and cost-effective solution for continuous patient monitoring, particularly suited for elderly patients, post-surgical monitoring, and individuals with chronic health conditions.

REFERENCES

[1] Rajendra, A. K., et al. (2022). IoT-Based Patient Health Monitoring System Using Arduino and Pulse Oximeter. International Journal of Engineering Research and Technology (IJERT), 11(4), 112-118.

[2] Kumar, S., & Sharma, P. (2023). ESP8266-Based Wireless Health Monitoring System with Cloud Integration. IEEE International Conference on IoT and Smart Systems, 45-51.

[3] Dr. J. Narendra Babu, Strengths on Internet of Things. Journal of Advanced Research in Dynamical & Control Systems, Vol. 11, No. 11, 2019.

[4] Dr. J. Narendra Babu, Enhancement of RVM Using FPGA. Journal of Advanced Research in Dynamical & Control Systems, Vol. 11, No. 11, 2019.

[5] Dr. J. Narendra Babu, Brain Signal Processing: Analysis, Technologies and Application. Journal of Advanced Research in Dynamical & Control Systems (JARDCS), Vol. 11, No. 12, 2019.

[6] Verma, R., & Singh, A. (2022). Real-Time Patient Monitoring Using MAX30100 and NodeMCU with Firebase Database. International Journal of Advanced Computer Science and Applications (IJACSA), 13(6), 78-85.

[7] Google Firebase. (2024). Firebase Realtime Database Documentation. Retrieved from https://firebase.google.com/docs/database

[8] Mohan, N., & Reddy, K. (2023). AI-Integrated Healthcare Monitoring Systems: A Review. Journal of Biomedical Informatics, 142, 104-115.

[9] Adafruit Industries. (2024). DHT11 Temperature and Humidity Sensor Documentation. Retrieved from https://learn.adafruit.com/dht

[10] OXulloIntersecans. (2023). MAX30100 Pulse Oximeter Library for Arduino. GitHub Repository. Retrieved from https://github.com/oxullo/Arduino-MAX30100

[11] Dr. J. Narendra Babu, et al. Experimental Detection of Vehicles for Toll Gate Billing. International Journal of Science Technology and Management, Vol. 4, Issue No. 12, December 2015, ISSN 2394-1537.

[12] Mobizt. (2024). Firebase ESP8266 Client Library Documentation. GitHub Repository. Retrieved from https://github.com/mobizt/Firebase-ESP8266

[13] Dr. J. Narendra Babu, Steganography Based Message Hiding in Image. International Journal of Advance Research in Science and Engineering, Vol. 4, Issue No. 12, December 2015, ISSN 2319-8354.

Authors Biography

Dr. J. Narendra Babu is a seasoned academician with over 28 years of experience in teaching and the software industry. He currently serves as a Professor in the Department of Data Science at Sapthagiri NPS University. He holds B.Tech, M.Tech, and Ph.D. degrees. He has published extensively in reputed journals and conferences and plays a key role in mentoring students and coordinating academic activities.

.