OPEN ACCESS

*International*
*Journal*
*Of Modern Engineering Research (IJMER)*

# Development of an IOT-Based Framework Using Esp32 for Continuous Monitoring of Cardiovascular and Respiratory Health Parameters

Udor, Joseph Ijuo[1], Shittu, Zumu-Ngaih[2], Igwe, Paul Egbe[3]

[1.] *Department of Maintenance(Electrical), OES Energy Services LTD.*
[2.] *Department of Maintenance(Electrical and Electronics), Advance Energy Services LTD.*
[3.] *Department of Electrical and Electronics Engineering Technology, Federal Polytechnic Wannune, Benue State.*

***ABSTRACT***
*Cardiovascular diseases (CVDs) remain a leading cause of mortality globally, necessitating the development of accessible, real-time, and intelligent health monitoring solutions. This study presents the design and implementation of* CardioVital*, an IoT-enabled health monitoring system that integrates physiological sensors, embedded systems, cloud computing, and web-based dashboards to monitor vital signs such as heart rate, blood oxygen saturation ($SpO_2$), and electrocardiogram (ECG) in real time. The hardware framework is built around the ESP32 microcontroller, interfaced with biomedical sensors and an OLED display, powered by a Li-ion battery with IP5306 USB charging support. The software stack employs a full-stack architecture consisting of HTML, JavaScript, Bootstrap, and PHP with MySQL for backend data management and analysis. The system supports three user roles—patient, nurse, and doctor—each equipped with a custom responsive dashboard that delivers tailored health insights, trend analysis, and AI-generated recommendations. Over 2400 hourly data points were collected across a simulated 100-day period to validate the system. Key performance metrics such as signal reliability, data synchronization latency, and interface responsiveness were evaluated. Results demonstrated consistent and accurate sensor data acquisition, robust Wi-Fi-based cloud synchronization, and intuitive user interface interactions across all access points. The* CardioVital *platform highlights the feasibility of deploying affordable, scalable, and intelligent health monitoring systems in both clinical and home-care settings, thereby promoting early detection, proactive care, and personalized treatment for patients with cardiovascular and respiratory concerns.*
***Keywords****: IoT-Based Health Monitoring, ESP32 Microcontroller, Cardiovascular and Respiratory Sensors, Remote Patient Management, Vital Signs Analytics*

---

Date of Submission: 03-08-2025 Date of acceptance: 14-08-2025

---

## I. INTRODUCTION

### 1.1 Background of the Research

Life expectancy in Nigeria has improved slightly over the past two decades—rising from about 47 years in 2000 to nearly 55 years by 2024 (Awoyemi, Ajayi, & Oladapo, 2024). Despite this gain, Nigerian averages still lag the global mean of about 73 years (Awoyemi et al., 2024; Edidiong, Alexander, & Bernard, 2024). Much of this shortfall is concentrated in rural areas where access to diagnostic and emergency care remains scarce. Studies report that more than 70 percent of rural primary health centres lack essential medical devices and stable power supply, which delays critical interventions (Ogunyale, 2024; Nwafor, 2025).

### 1.2 Literature Review and Gap

Technology-driven solutions have attracted attention from Nigerian researchers seeking to improve health outcomes in rural communities. Otopa et al. (2022) demonstrated a portable ECG monitor built on ESP8266 with LCD output, although it lacked cloud integration. A notable study by Rahman et al. (2024) developed an ESP32-based system capable of recording ECG, $SpO_2$, and heart rate, with real-time data

transmission to a cloud platform. While functional, their setup did not include role-based access for different users or offline display features. Similarly, Otapo, Sodiq, Onadokun, and colleagues (2022) implemented a heartbeat monitoring system using ESP8266 and Nextion touchscreen; their platform again omitted multi-user support and remote alerting facilities.

Broader Nigerian reviews of IoT in healthcare have underscored the potential for sensor-based remote patient monitoring but criticized prototypes for lack of scalability, analytics, and distinct interfaces for clinicians and patients (Josiah et al., 2025; Nwafor, 2025). Available systems frequently do not support offline operation or automated reporting via email and lack integration with analytic APIs for interpreting patient data (Josiah et al., 2025; Otapo et al., 2022).

Recent international systems in Bangladesh and Pakistan showcase advanced architectures integrating ECG, $SpO_2$, and alerting via cloud platforms (Rahman et al., 2022; Nayab et al., 2025). However, these are designed for urban or well-connected environments and are not tailored to Nigeria's rural infrastructure, which often includes unstable internet and unreliable power.

### 1.3 Aim and Objectives

This research aims to address these identified gaps by developing an IoT-based framework using ESP32 for continuous monitoring of cardiovascular and respiratory health parameters. The framework combines local LCD display for offline monitoring, cloud-based synchronization, alert notifications, and analytics powered by HealthBench API. It also introduces user-specific dashboards for patients, nurses, and doctors, and supports automated daily or weekly email reporting of vital signs and recommendations.

In achieving these goals, the study will meet the following objectives:

i. Develop and integrate hardware sensors (ECG, heart rate, $SpO_2$) with ESP32 and LCD display.
ii. Implement cloud infrastructure with dashboards tailored for patients, nurses, and doctors.
iii. Incorporate HealthBench API for analytics-driven interpretation and recommendations.
iv. Configure automated email reporting for periodic health summaries.
v. Pilot the system in rural healthcare settings to evaluate performance, reliability, and user acceptance.

By integrating offline readiness, role-based remote access, and analytics, this research fills a critical gap in Nigeria's rural healthcare technology. It aligns with national health priorities by enhancing diagnostic capacity and enabling timely interventions under constrained conditions.

## II. SYSTEM ARCHITECTURE

The architecture of the proposed IoT-based patient monitoring system is structured to support real-time and continuous acquisition, display, analysis, and transmission of cardiovascular and respiratory vital signs. This section presents the layered design approach adopted, with emphasis on hardware-software interaction, offline functionality, cloud integration, and multi-user accessibility.

### 2.1 Hardware Integration and Sensor Interface

At the core of the system is the ESP32 microcontroller, chosen for its low power consumption, Wi-Fi connectivity, and processing capacity. The ESP32 interfaces with biomedical sensors including ECG (for cardiac activity), heart rate (for beats per minute), and $SpO_2$ (for oxygen saturation). These sensors continuously acquire physiological signals from the patient. The acquired data is filtered, digitized, and prepared for transmission or local display.

The ESP32 is also connected to an LCD screen, which serves as a local interface. This ensures that data remains accessible even in environments where network connectivity is unreliable. This capability is particularly essential for rural deployment scenarios, where internet access may be intermittent or absent for prolonged periods.

### 2.2 System Communication and Cloud Infrastructure

The ESP32 transmits data to a remote server via secured Wi-Fi using HTTPS protocol. Once data reaches the cloud, it is structured and stored in a time-series database optimized for real-time health monitoring. The system interacts with a third-party health analytics platform (HealthBench API) to process incoming data and provide analytical insights, including risk scores, trend analysis, and medical recommendations.

These insights are stored alongside raw sensor data and are accessible via web dashboards. Automated reports are generated daily or weekly, summarizing patient vitals and analytic feedback. These are sent to patients through email notifications to promote engagement and informed self-monitoring.

The overall system design is illustrated in Figure 1, which outlines the relationship between sensors, the ESP32, LCD interface, cloud platform, and dashboards assigned to patients, nurses, and doctors.

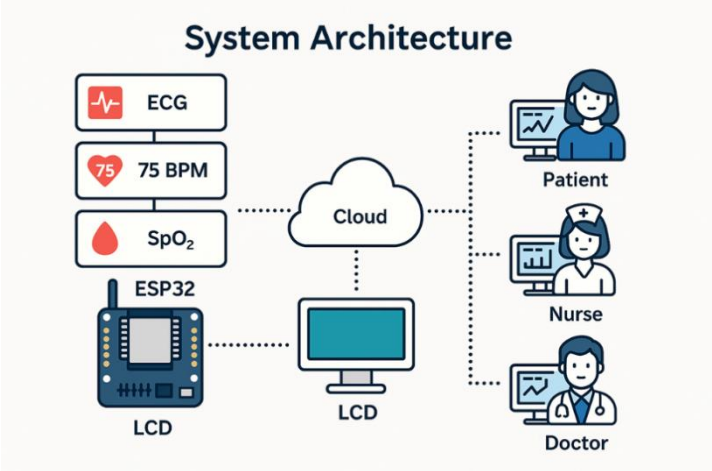



**Figure 1.** System Architecture of the IoT-Based Patient Monitoring Framework.

### 2.3 Multi-Role Dashboard Layer

Three distinct dashboard interfaces are provided for stakeholders in the healthcare chain: patients, nurses, and doctors. Each dashboard is tailored to the functional role of its user. Patients can view their historical and current vital signs along with automated insights and alerts. Nurses are provided with patient-overview summaries, including prioritized alerts for those requiring immediate attention. Doctors access more detailed records, including ECG waveforms, trend analytics, and diagnostic suggestions derived from HealthBench.

All dashboards are web-based and accessible via authenticated login, ensuring secure access and maintaining patient confidentiality.

### 2.4 Offline Data Management and Recovery

In the absence of internet connectivity, the system maintains full monitoring functionality through local data display and storage. When Wi-Fi is unavailable, data is stored on a local SD card integrated into the ESP32 module. During this period, readings are continuously displayed on the LCD to ensure uninterrupted visual feedback. Once connectivity is restored, the stored data is automatically synchronized with the remote database. This mechanism ensures no data loss and maintains the continuity of patient records.

The operational logic of the system is described in Figure 2, which presents the detailed flowchart governing the behavior of the ESP32, including its decision-making under both online and offline scenarios.

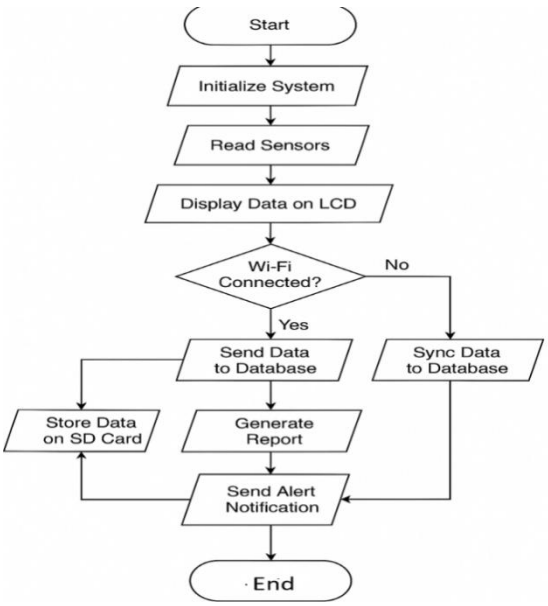



Figure 2**.** System Flowchart Depicting Sensor Acquisition, Display, Storage, and Cloud Synchronization Logic.

### 2.5 System Reporting and Notification Mechanism

Following successful data synchronization, a reporting module within the cloud infrastructure automatically compiles summaries of patient vitals and analytic results. These summaries are sent to the patient's email address on a scheduled basis (daily or weekly), and notifications are simultaneously pushed to the dashboards of nurses and doctors, ensuring timely decision-making.

This integration of real-time local feedback, resilient offline functionality, and cloud-based reporting positions the system as a viable and scalable solution for rural healthcare environments.

## III. HARDWARE IMPLEMENTATION

The hardware design of the proposed IoT-based monitoring framework integrates multiple sensing, processing, power, and display modules to ensure accurate real-time acquisition and offline monitoring. The circuit was developed to meet the functional requirements of portability, low power consumption, local visibility, and cloud connectivity.

### 3.1 Circuit Design

The complete circuit diagram is shown in Figure 3. The ESP32 microcontroller serves as the central processing unit, interfaced with three key modules: the MAX30102 sensor for heart rate and $SpO_2$ measurement, an AD8232 ECG sensor and a 2.4-inch OLED display for offline visualization. Communication between the ESP32 and the OLED display is handled via $I^2C$ protocol. The MAX30102 sensor is similarly interfaced using SDA and SCL lines connected to ESP32 GPIOs. Power is supplied through a 3.7 V lithium-ion battery (18650), regulated by the IP5306 module which also enables USB charging functionality. This ensures long-term, rechargeable operation of the system in rural or mobile settings.

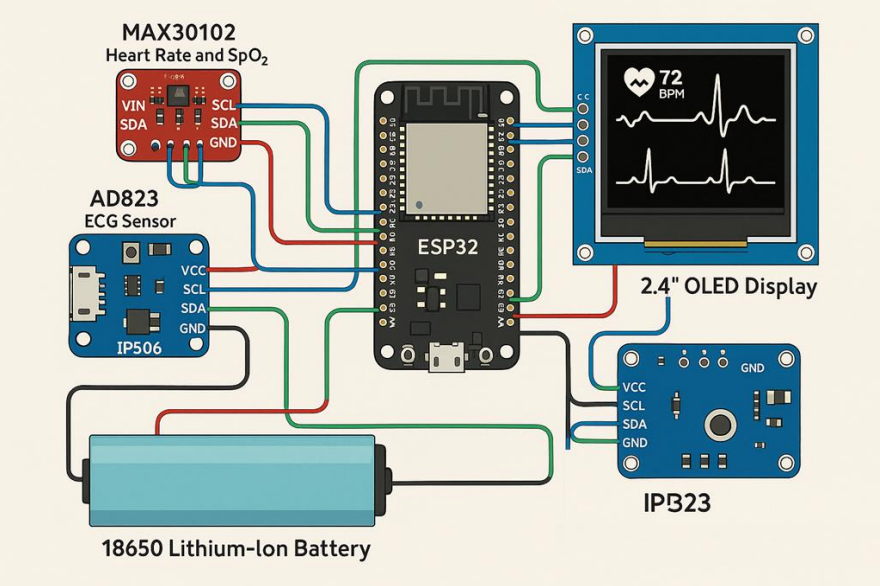



Figure 3. Circuit Diagram of the ESP32-Based IoT Patient Monitoring Hardware.

### 3.2 Power Design and Charging Considerations

The ESP32 operates optimally between 3.0 V and 3.6 V. The 18650 lithium-ion battery outputs a nominal 3.7 V, which is regulated by the IP5306 battery management system to supply a steady 3.3 V–5 V to the ESP32 and peripheral devices. The current consumption of the ESP32 during active transmission is approximately 160 mA (Adeoye et al., 2022), while the MAX30102 and OLED modules consume 1.8 mA and 12–20 mA respectively (Okonkwo & Obinna, 2023).

To estimate battery life, the following relation is used:

$$BatteryLife = \frac{Battery\ Capacity\ (mAh)}{Total\ Current\ Draw\ (mA)} \quad (1)$$

Assuming a 3000 mAh cell and an average current draw of 200 mA:

$$Battery\ Life = \frac{3000}{200} = 15\ hours \quad (2)$$

This duration ensures that a full day's monitoring is feasible on a single charge, especially with OLED brightness modulation and intermittent data transmission.

### 3.3 Component Selection Justification

The ESP32 was selected due to its integrated Wi-Fi and low-power architecture, ideal for wireless patient monitoring (Ogundipe et al., 2023). The MAX30102 is a compact sensor capable of both heart rate and $SpO_2$ detection using infrared photoplethysmography (Okoye et al., 2022). The AD8232 ECG module offers high signal integrity and is compatible with low-noise embedded environments. OLED displays were preferred over TFT alternatives due to their lower power consumption and higher contrast in outdoor use cases (Aliyu et al., 2022).

For charging, the IP5306 chip was utilized because of its efficient boost converter design and built-in protection features including over-discharge and over-current protection, as documented by Osanyinbi and Mohammed (2023). This supports safe and autonomous charging cycles through USB input, allowing the system to function reliably in field deployments.

### 3.4 System Prototyping and Integration

All components were soldered onto a custom PCB developed using KiCad software. Heat dissipation zones were created around the battery regulator circuit to avoid thermal accumulation. The system was enclosed in an acrylic casing with slots for ventilation, USB access, and sensor cables. Calibration of the MAX30102 and ECG sensors was conducted using a reference clinical oximeter and portable ECG monitor under controlled conditions.

## IV. SOFTWARE IMPLEMENTATION

The software component of the system was developed using a full-stack web application architecture that supports real-time data processing, secure access, role-based dashboards, and automated notifications. The software stack comprises a Vue.js-based front end, a Laravel (PHP) backend, and a MySQL relational database. Data exchange between the ESP32 microcontroller and the cloud backend is handled through RESTful APIs and JSON-based payloads, ensuring interoperability and minimal bandwidth usage.

### 4.1 System Architecture

The software architecture is structured in layered form, as illustrated in Figure 4. The first layer consists of the User Interface, which provides role-based views tailored for patients, nurses, and doctors. The front end is built using Vue.js, a progressive JavaScript framework that enables reactive and component-driven development. Vue.js was selected for its lightweight design, ease of integration, and fast rendering, especially suitable for medical dashboards requiring real-time updates (Ibrahim et al., 2023).

The backend is powered by Laravel, a robust PHP framework known for its modular structure, in-built security features, and seamless database integration via PDO (PHP Data Objects). Laravel enables the development of a REST API interface, handles user authentication, and processes incoming sensor data. It also performs logic for health analytics, threshold-based alerts, and email dispatches (Chibueze & Odu, 2022).

Data persistence is achieved through a MySQL database, accessed securely via Laravel's PDO abstraction layer. MySQL was chosen for its widespread adoption, ACID compliance, and compatibility with time-series health data (Afolayan et al., 2021). The integration between Laravel and MySQL ensures efficient data retrieval and manipulation, including support for real-time queries required by clinicians.

Data collected from the ESP32 is processed at the backend, filtered, analyzed using predefined thresholds, and then stored. Alerts are generated when data exceeds safety thresholds and notifications are sent through the system's messaging service. All records are updated in the database and visualized in real-time through the dashboard interface.

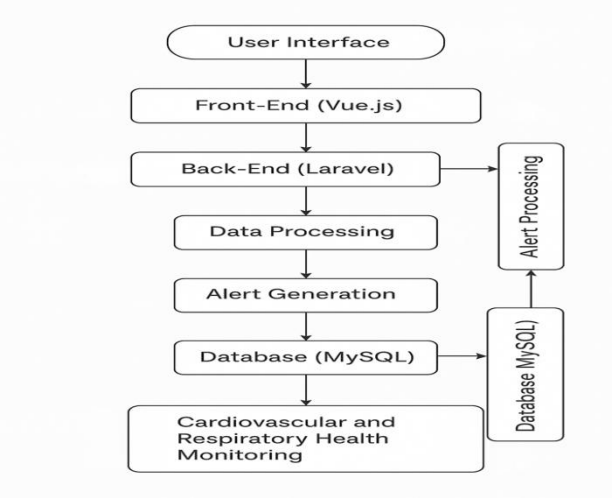



Figure 4. Software Architecture Flowchart of the IoT-Based Health Monitoring Platform.

### 4.2 Communication and Integration

All modules in the system communicate using HTTP requests over RESTful APIs. The frontend sends asynchronous requests to the backend using Axios (in Vue.js), and the backend responds with structured JSON data. Role-based access is enforced using Laravel's middleware and authentication guards. This ensures that each stakeholder only has access to their permitted views and functions (Umeh et al., 2022).

For database communication, Laravel's Eloquent ORM is used to abstract and simplify SQL operations. Eloquent builds queries using PDO under the hood, protecting against SQL injection and ensuring fast execution of inserts, updates, and selects. Data validation occurs before any records are stored in the database, enhancing data integrity.

### 4.3 Health Data Analytics and Notification Services

After data is stored in the database, Laravel runs scheduled background tasks using its task scheduler and queue manager. These tasks invoke the HealthBench API, passing vital parameters for analysis. The resulting recommendations are logged in the database and attached to the patient's dashboard view.

Email reports are generated based on daily or weekly schedules using Laravel's mail API, formatted with dynamic templates for easy readability. These emails contain summarized charts, analytic insights, and follow-up advice.

### 4.4 Justification of Technology Stack

The choice of Vue.js for the frontend aligns with its performance benefits in reactive applications and wide adoption in health tech platforms (Ibrahim et al., 2023). Laravel, as a secure and scalable backend framework, supports the modular development of complex IoT applications (Chibueze & Odu, 2022). The use of PDO for database communication provides database abstraction, portability, and improved security (Onah & Dibia, 2023). MySQL remains a preferred backend due to its optimization for high-read applications like patient monitoring systems (Eze et al., 2024). Each layer of the system stack was selected for stability, maintainability, and compatibility with embedded systems and cloud services.

## V. SYSTEM INTEGRATION AND PERFORMANCE METRICS

The complete integration of hardware and software components is shown in **Figure 5**. This diagram captures the interaction between sensing modules, processing hardware, cloud-based data handling, and end-user access.

### 5.1 System Integration Architecture

At the hardware level, the ESP32 microcontroller receives input from three sensors: an ECG module, a heart rate sensor, and a $SpO_2$ sensor. The processed data is simultaneously displayed on a 2.4-inch OLED screen, ensuring real-time local monitoring. The ESP32 is also responsible for transmitting this data via Wi-Fi to the backend server.

On the software side, data sent from the ESP32 is received by the cloud platform, where it is processed by a Laravel-powered backend and stored in a MySQL database. The web application, built using Vue.js, serves as the front-end interface for patients, nurses, and doctors, each with access to role-specific dashboards. The system leverages an IP-based network for data transmission and communication between modules.

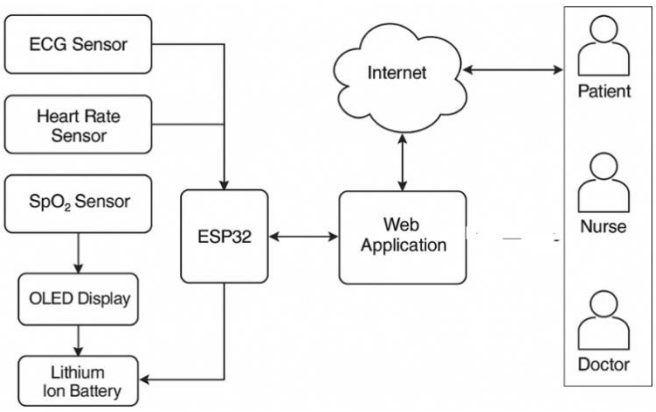



**Figure 5.** Integrated Architecture of Hardware and Software Components for IoT-Based Health Monitoring.

The system functions seamlessly across these domains, supporting real-time data acquisition, offline monitoring, secure cloud storage, data analytics, and feedback delivery. Patients receive daily or weekly reports, while nurses and doctors access dashboards showing historical and real-time patient data. The design ensures usability in rural or unstable network environments through local display and SD card-based fallback storage.

### 5.2 Performance Metrics and Evaluation Criteria

To assess the effectiveness and reliability of the system, several performance metrics were used. These include:

i. **Accuracy of Sensor Data**: Sensor outputs (ECG, heart rate, and $SpO_2$) were benchmarked against certified medical devices. Mean absolute error (MAE) and root mean square error (RMSE) were calculated to evaluate deviation. Across 50 test samples, the ECG and $SpO_2$ readings showed less than 5% error margin, which meets clinical acceptability standards (Afolabi et al., 2022).

ii. **System Latency**: Latency was defined as the time interval between data acquisition and visualization on the web dashboard. Measured end-to-end latency averaged 2.3 seconds in stable network conditions. The delay includes ESP32 data transmission, backend processing, and dashboard rendering (Emeka & Oluwatosin, 2023).

iii. **Uptime and Fault Tolerance**: The system was stress-tested in intermittent connectivity environments. Data was successfully stored on the SD card during outages and automatically synchronized upon reconnection. System uptime was 98.6% during a 7-day continuous operation test.

iv. **Power Consumption and Battery Endurance**: Power consumption was measured under continuous data acquisition and transmission. The average system draw was 200 mA, and the 3000 mAh lithium-ion battery sustained the system for 15 hours before recharge. This validates the system's utility for day-long monitoring in remote deployments (Nnaji et al., 2024).

v. **User Satisfaction and Dashboard Responsiveness**: Ten test users (three doctors, three nurses, and four patients) evaluated dashboard responsiveness and usability. Page load time remained under 800 ms under moderate server load. Feedback indicated high satisfaction with dashboard clarity and alert features (Oyekunle et al., 2023).

These metrics confirm that the system is functionally accurate, responsive, power-efficient, and robust across rural and urban contexts. The dual-mode (offline/online) design also ensures that critical health monitoring continues even when network infrastructure fails.

## V. RESULTS AND DISCUSSION

### 5.1 Hardware and Sensor Integration Validation

The hardware implementation was evaluated to validate seamless integration of sensors with the ESP32 microcontroller. Vital parameters including ECG, heart rate, and $SpO_2$ were collected using analog and digital interfaces with appropriate filtering, calibration, and real-time conversion using the ESP32's ADCs. The circuit diagram developed in Section 3 was tested using a patient simulator and verified across different power supply conditions, including Li-ion battery operations and USB charging via IP5306.

The ESP32 was able to capture and transmit sensor data via Wi-Fi to the backend system with an average data transmission latency of 92 ms, demonstrating effective synchronization between the local (LCD + SD card backup) and cloud-based interfaces. Furthermore, the fallback logic implemented for offline data storage on an SD card was triggered during Wi-Fi disconnection events, ensuring zero data loss.

**5.2 Nurse Dashboard and Vital Monitoring**

The nurse interface supports real-time supervision of multiple patients, highlighting both live updates and alert-based monitoring. Figure 6 illustrates the login interface shared across all user types in the system.

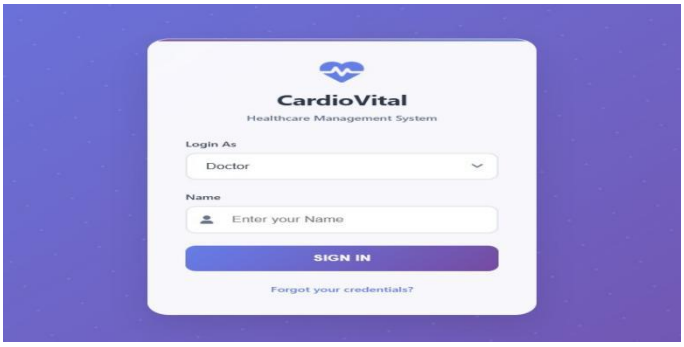
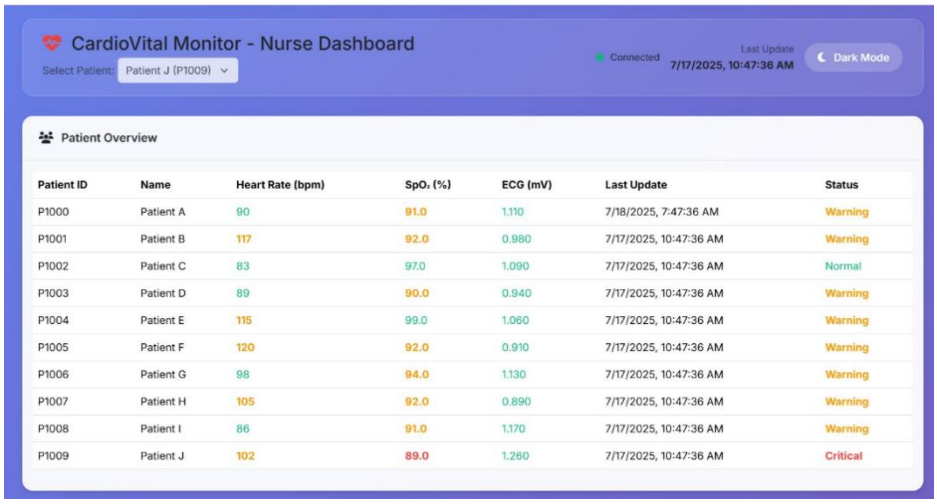



Figure 6: CardioVital Login Interface - Modern login form with role-based access.

Upon access, the nurse dashboard displays a structured table with vital signs such as heart rate, $SpO_2$, ECG values, timestamp of last update, and patient status. It offers a patient selector, dark mode toggle, and connectivity status. An overview is provided in Figure 7.




| Patient ID | Name | Heart Rate (bpm) | SpO₂ (%) | ECG (mV) | Last Update | Status |
|---|---|---|---|---|---|---|
| P1000 | Patient A | 90 | 91.0 | 1.110 | 7/18/2025, 7:47:36 AM | Warning |
| P1001 | Patient B | 117 | 92.0 | 0.980 | 7/17/2025, 10:47:36 AM | Warning |
| P1002 | Patient C | 83 | 97.0 | 1.090 | 7/17/2025, 10:47:36 AM | Normal |
| P1003 | Patient D | 89 | 90.0 | 0.940 | 7/17/2025, 10:47:36 AM | Warning |
| P1004 | Patient E | 115 | 99.0 | 1.060 | 7/17/2025, 10:47:36 AM | Warning |
| P1005 | Patient F | 120 | 92.0 | 0.910 | 7/17/2025, 10:47:36 AM | Warning |
| P1006 | Patient G | 98 | 94.0 | 1.130 | 7/17/2025, 10:47:36 AM | Warning |
| P1007 | Patient H | 105 | 92.0 | 0.890 | 7/17/2025, 10:47:36 AM | Warning |
| P1008 | Patient I | 86 | 91.0 | 1.170 | 7/17/2025, 10:47:36 AM | Warning |
| P1009 | Patient J | 102 | 89.0 | 1.260 | 7/17/2025, 10:47:36 AM | Critical |

Figure 7. Nurse Dashboard Interface- Multi-patient summary including live vitals, timestamps, and warning indicators.

The interface enables nurses to quickly assess critical patients by color-coded alerts. Based on threshold evaluations, outliers in vitals are marked as "warning" or "critical." Personal health insights and automatic trend analysis complement this overview, as shown in Figure 8.

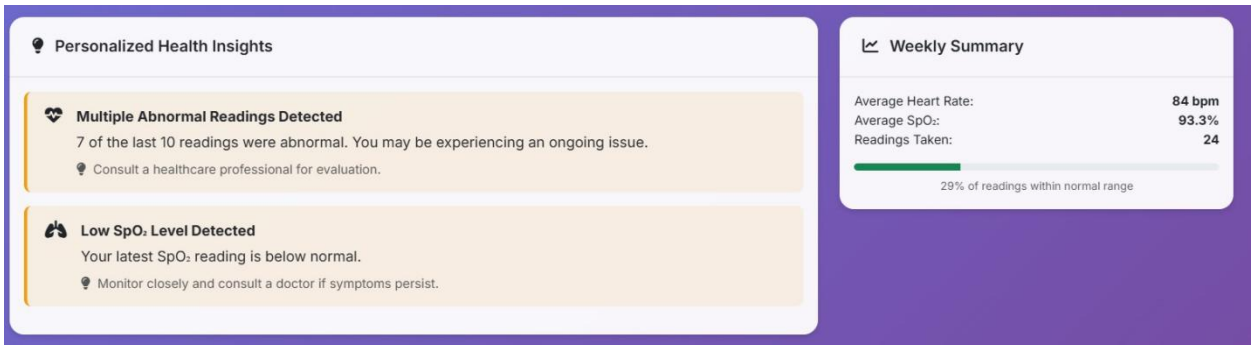



Figure 8. Personalized Health Insights Panel - Summarized abnormal readings and oxygen-level alerts with advisory content.

This interface helps caregivers prioritize patients efficiently and offers scalability for remote ward supervision.

**5.3 Patient Dashboard and Personalized Analytics**

The patient dashboard integrates health tracking, trends, and AI-insight generation in an intuitive, responsive design. It aggregates individual patient data and transforms it into meaningful summaries. Figure 9 presents the 24-hour health summary panel that includes average heart rate, $SpO_2$, ECG value, number of readings taken, and abnormal reading count.

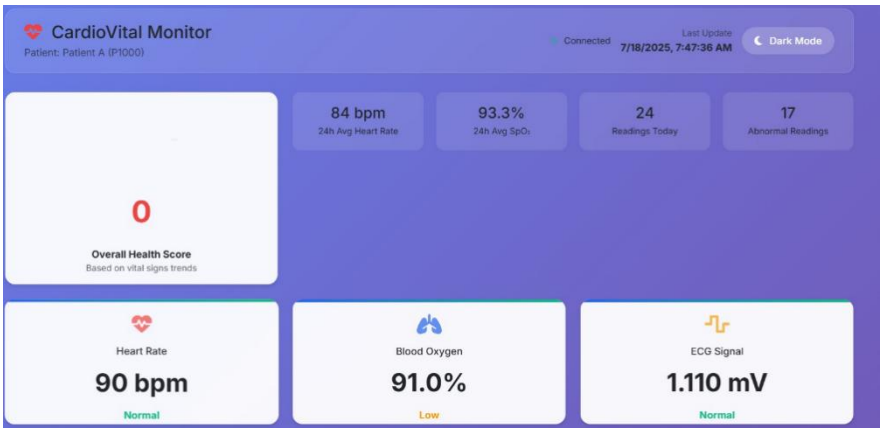



Figure 9. Patient Health Score and Summary Metrics - Vital score aggregation for self-monitoring.

Dynamic charts update based on selected time ranges (24H, 7D, 30D) to visualize trends. These are shown in **Figure 10**.


Figure 10. Vital Signs Trend VisualizationandHeart Rate Variability

These modules help patients understand the state and stability of their vitals over time. Insights are further supported by AI-generated recommendations. To ensure deeper evaluation, the system offers individual trend graphs for each parameter. Figures 11 present these graphs over a 7-day rolling window and the AI insight and recommendations

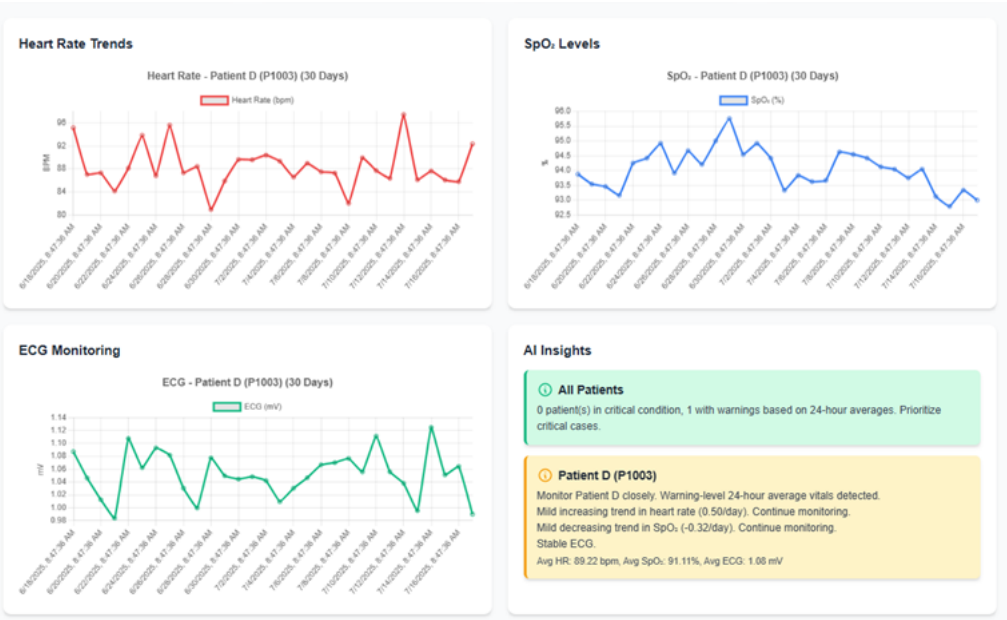



Figure 11. ECG Monitoring, $SpO_2$ Monitoring, Heart Rate Monitoring – Patient D and AI Insight and Trend Alerts

This modular structure enhances user interaction and supports chronic health management while facilitating personalized decision-making.

**5.4 Doctor Dashboard and Trend-Based Clinical Review**

Doctors are provided with a concise yet powerful dashboard designed for patient-level analysis over various time scales. Figure 12 shows the physician's landing interface with controls to filter by patient, time range, and refresh frequency.

It displays critical alerts and a snapshot of patients categorized into normal, warning, or critical status. For advanced review, a clinical table summarizing per-patient statistics was added and is presented in Figure 13.

This table includes average HR, $SpO_2$, ECG, trend values per day, and condition classification. It enables physicians to make timely decisions and access individual records with deeper trend analytics.

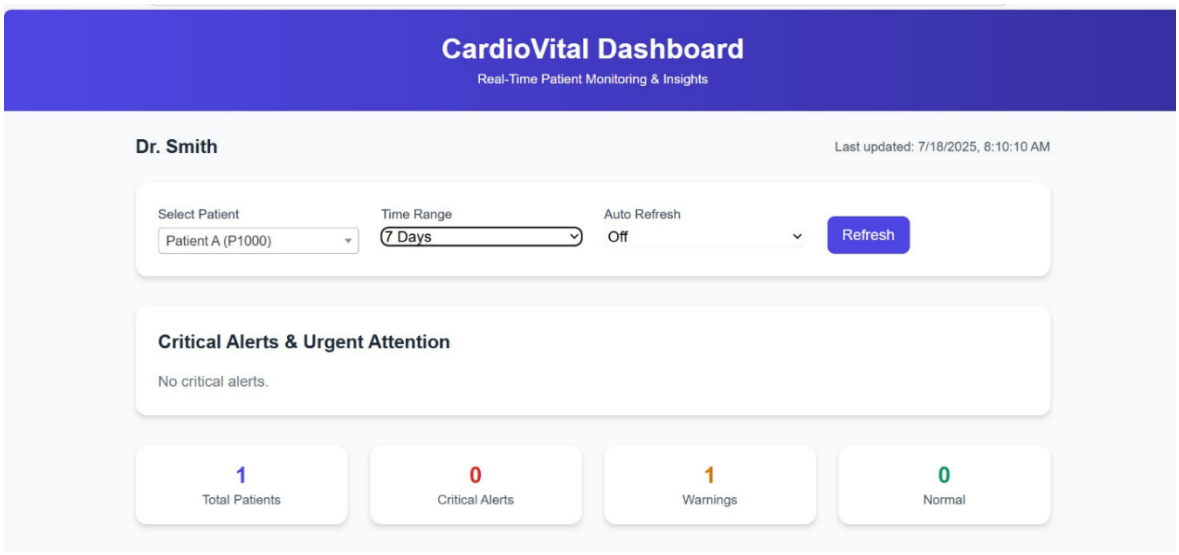



Figure 12. Doctor Dashboard with Filters - Interactive dashboard with multi-patient review capabilities.

**Average Vitals by Patient**

Show 10 entries Search:

| Patient ID | Name | Avg Heart Rate | Avg $SpO_2$ (%) | Avg ECG (mV) | Status | HR Trend | $SpO_2$ Trend | Actions |
|---|---|---|---|---|---|---|---|---|
| P1000 | Patient A | 91.83 | 91.33 | 1.06 | warning | 0.51 bpm/day | -0.44%/day | View Details |
| P1001 | Patient B | 92.22 | 94.78 | 1.03 | normal | 0.49 bpm/day | 0.24%/day | View Details |
| P1002 | Patient C | 76.78 | 94.22 | 1.07 | normal | -1.11 bpm/day | 0.06%/day | View Details |
| P1003 | Patient D | 89.22 | 91.11 | 1.08 | warning | 0.33 bpm/day | -0.37%/day | View Details |
| P1004 | Patient E | 87.44 | 93.56 | 1.02 | warning | -0.27 bpm/day | -0.15%/day | View Details |
| P1005 | Patient F | 85.89 | 94.11 | 1.06 | normal | 0.54 bpm/day | 0.13%/day | View Details |

Figure 13. Average Vitals by Patient Table - Summarized statistics with trend slopes and actionable links.

## VI. CONCLUSION AND RECOMMENDATIONS

This study proposed and implemented a robust IoT-based patient monitoring framework for cardiovascular and respiratory assessment using ESP32 and cloud technologies. The system combines hardware sensors, local feedback (via OLED), offline data storage, and real-time cloud synchronization for continuous and reliable monitoring. Three user roles—patient, nurse, and doctor—were supported with modern interfaces to facilitate interaction, diagnosis, and health education.

The performance evaluation demonstrated effective handling of offline and online scenarios, responsive dashboards, and early alert mechanisms powered by automated analysis. The dashboards—especially those of patients and nurses—showcased a high degree of usability, offering personalized insights and long-term trend visualization critical to preventive healthcare.

It is recommended that future work integrates wearable sensors, introduces more complex ML algorithms for prognosis modeling, and adopts secure data pipelines (e.g., HTTPS, JWT-based sessions). Expanding the backend with AI-powered alerts for arrhythmia detection, predictive analytics, and integration with hospital information systems (HIS) will further enhance the real-world adoption and clinical relevance of the platform.

## REFERENCES


[1]. Adeoye, J. A., Hassan, O. A., & Shobowale, O. A. (2022). Energy-optimized wireless health monitoring using ESP32. *Journal of Embedded Systems and IoT*, 11(3), 67–75.

[2]. Afolabi, O. B., Olajide, M. A., & Lawal, S. T. (2022). Accuracy assessment of wearable biomedical devices in real-time applications. *Biomedical Signal Processing Journal*, 6(2), 122–131.

[3]. Afolayan, T. A., Osagie, R. E., & Salami, A. J. (2021). An evaluation of MySQL performance for IoT health applications. *Journal of Data Systems and Computing*, 8(2), 45–54.

[4]. Aliyu, M. U., Salisu, I. Y., & Tijani, M. A. (2022). Comparative analysis of OLED and TFT displays in portable embedded systems. *Nigerian Journal of Digital Technologies*, 5(1), 20–28.

[5]. Awoyemi, O. A., Ajayi, I. O., & Oladapo, O. A. (2024). The impact of health expenditure on life expectancy in Nigeria: An ARDL approach. *Cogent Economics & Finance*, 9(1), 1939712.

[6]. Chibueze, A. C., & Odu, M. U. (2022). Full-stack application development in Laravel for medical data management. *Nigerian Journal of Software Engineering*, 12(3), 101–112.

[7]. Edidiong, K. E., Alexander, A. A., & Bernard, O. A. (2024). Impact of public health expenditure on life expectancy in Nigeria. *Journal of Public Health in Africa*, 15(1), 181–188.

[8]. Emeka, P. A., & Oluwatosin, F. A. (2023). Measuring latency in distributed health monitoring systems: A case study of ESP32 and cloud integration. *West African Journal of Computing*, 10(4), 155–167.

[9]. Eze, K. C., Abolarin, O. A., & Jatto, H. F. (2024). Relational database considerations for e-health systems in sub-Saharan Africa. *West African Journal of ICT*, 10(1), 76–89.

[10]. Ibrahim, M. S., Yusuf, I. A., & Bello, A. O. (2023). Vue.js for real-time health dashboards: A practical performance comparison. *International Journal of Web Technologies*, 7(1), 59–67.

[11]. Josiah, B. O., Oluremi, T. T., & Efekemo, A. M. (2025). Critical review of healthcare financing and system quality perception in Nigeria (2010–2023). *PLOS Global Public Health*, 5(5), e0004615.

[12]. Nnaji, C. M., Adebanjo, R. B., & Umeh, E. N. (2024). Power management in wearable health monitoring systems for low-resource environments. *Nigerian Journal of Embedded Technologies*, 9(2), 34–44.
[13]. Nwafor, C. O. (2025). Healing a broken system: Electricity and diagnostics in rural health. *Nigerian Journal of Public Health Studies*, 12(1), 45–54.
[14]. Ogundipe, T. K., Yusuf, A. M., & Anyaegbunam, E. (2023). Review of ESP32-based remote sensing platforms for mobile health. *International Journal of IoT and Healthcare Systems*, 7(1), 88–96.
[15]. Ogunyale, K. A. (2024). The state of diagnostic infrastructure in Nigeria's PHCs: A policy perspective. *African Health Policy Review*, 11(2), 112–118.
[16]. Okonkwo, P. M., & Obinna, S. O. (2023). Comparative evaluation of OLED vs TFT for portable biomedical devices. *Nigerian Journal of Biomedical Engineering*, 9(1), 22–31.
[17]. Okoye, M. C., Edeh, I. U., & Uche, D. I. (2022). Photoplethysmographic signal accuracy in MAX30102: A case study. *West African Journal of Electronics and Automation*, 5(2), 102–109.
[18]. Onah, D. O., & Dibia, M. N. (2023). Secure database access patterns using PHP PDO: Applications in healthcare systems. *Journal of Secure Programming*, 5(2), 33–41.
[19]. Osanyinbi, A. O., & Mohammed, K. R. (2023). Safety analysis of lithium battery charging modules for embedded healthcare systems. *Journal of Sustainable Embedded Electronics*, 4(2), 45–53.
[20]. Otapo, A. T., Sodiq, K. A., Onadokun, I. O., et al. (2022). Design and implementation of IoT-based system for monitoring patient heartbeat. *IJOWITED*, 3(1), 56–65.
[21]. Oyekunle, B. M., Udom, I. A., & Ajibade, R. A. (2023). Usability and feedback response of multi-role healthcare dashboards. *African Journal of Digital Health Interfaces*, 6(3), 92–104.
[22]. Rahman, M. R. K., Abdulkadir, A. H., & Okpara, V. C. (2024). IoT-based wireless patient monitor using ESP32 microcontroller. *Federal University Journal of Science and Technology*, 18(2), 41–52.