International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Smart, Secure and Scalable Cloud-Based Healthcare System

Mr. Rajesh T. H.
Department of Computer Science and Engineering
PESITM, Shimoga
Email: rajesh@pestrust.edu.in

Mr. **Aditya S. Jadhav**
Department of Computer Science and Engineering
PESITM, Shimoga
Email: 06jadhavaditya@gmail.com

Ms. **Aishwarya K.**
Department of Computer Science and Engineering
PESITM, Shimoga
Email: aishwaryaa2k4@gmail.com

Ms. **Ananya Aithal**
Department of Computer Science and Engineering
PESITM, Shimoga
Email: ananyaaithal26@gmail.com

Mr. B. V. Mallikarjuna Gouda
Department of Computer Science and Engineering
PESITM, Shimoga
Email: mallikarjunagouda2807@gmail.com

Abstract*—The management of Electronic Health Records (EHRs) requires secure, scalable, and user-friendly platforms that ensure data privacy while supporting efficient healthcare workflows. Traditional systems often face challenges such as weak security, lack of scalability, and poor accessibility for patients and healthcare providers. This paper presents a Smart, Secure, and Scalable Cloud-Based Healthcare System implemented as a modern web application using a Python Flask backend and a React-based frontend. The system incorporates secure authentication through JSON Web Tokens (JWT), Role-Based Access Control (RBAC), structured EHR management, appointment scheduling, and medication tracking. Two advanced features are introduced: an AI-powered CareMate Chatbot, integrated using the OpenAI API, which enables patients to interact with their health data through natural language; and an Emergency Blood Request Broadcast Module, enabling administrators to instantly notify all users during critical situations. The backend is implemented using Flask-SQLAlchemy with SQLite for development, designed for future scalability to PostgreSQL or other cloud-based databases. The frontend utilizes Tailwind CSS for a responsive user interface and Axios for RESTful communication. Testing using Postman and UI validation confirms secure access control, functional correctness, and responsive performance. The results demonstrate that the system delivers a secure, interactive, and extensible healthcare platform aligned with UN Sustainable Development Goals (SDG 3: Good Health and Well-being, and SDG 9: Industry, Innovation, and Infrastructure), providing a strong foundation for future cloud deployment and interoperability enhancements.

***Index Terms*—Electronic Health Records (EHR), Healthcare, Security, Scalability, Python Flask, React, JSON Web Tokens (JWT), Role-Based Access Control (RBAC)**

I. Introduction

Digital transformation in healthcare demands secure and accessible systems for managing patient information. Electronic Health Records (EHRs) improve data accuracy, authorized access, and clinical decision support compared to paper-based methods, yet many existing platforms still suffer from fragmented data, poor usability, and difficulties in scaling. These limitations show the need for a web-based system that is secure, role-aware, and designed on a scalable, cloud-ready architecture.

The proposed Smart, Secure and Scalable Cloud-Based Healthcare System addresses this need through a full-stack web application built with Python Flask for the backend API and React for the frontend interface. Security is enforced using JWT-based authentication and Role-Based Access Control (RBAC), so Patients, Doctors, and Administrators only access permitted operations. Patients can view EHRs, manage appointments, and see medications, while doctors manage patient lists, create and update EHRs (with edit rights restricted to the creating doctor), handle appointments, and prescribe medications; administrators oversee user management and overall system governance. Beyond core EHR workflows, the system integrates the CareMate AI Assistant and MannMitra mental-health companion within the patient dashboard, enabling natural-language queries, empathetic guidance, and seamless transition into a validated Mental Health Check-in module for structured screening and personalised suggestions. An enhanced Emergency Blood Request System allows authorized staff to broadcast urgent blood needs to the registered community in real time and share curated mental-health resources with high-risk users. By uniting strong security, scalable architecture, intelligent wellness support, and emergency coordination, the project aims to enhance patient accessibility, doctor efficiency, and overall quality of digital healthcare, contributing to SDG 3 (Good Health and Well-being) and SDG 9 (Industry, Innovation and Infrastructure).

A. Contributions:

The main contributions of this paper are as follows:

1) To implement secure user authentication (Registration, Login, Password Reset) using JWT.
2) To enforce Role-Based Access Control (RBAC) for Patients, Doctors, and Administrators.
3) To develop functionalities for creating, reading, updating, and deleting Electronic Health Records (EHRs) with appropriate permissions.

4) To implement an appointment booking system allowing patients to book and cancel appointments, and doctors to view their schedules.
5) To create a medication management module allowing doctors to prescribe and patients to view medications.
6) To integrate the CareMate AI Assistant and MannMitra mental-health companion within the patient dashboard to provide conversational, context-aware guidance while clearly remaining non-replacement for professional therapy.
7) To embed a validated Mental Health Check-in workflow that can be invoked from the AI companion, enabling structured screening, wellness scoring, and personalised self-care suggestions stored alongside the patient's records.
8) To extend the Emergency Blood Request System and notification layer so that authorized staff can broadcast urgent blood requirements and disseminate curated mental-health support resources to high-risk users in real time.

II. Literature Survey

The field of Electronic Health Records (EHR) is well-established, with numerous commercial and open-source systems available. Existing works include large-scale enterprise solutions like Epic, Cerner, and Allscripts and open-source alternatives like OpenEMR.

1) Role of Cloud Computing in Healthcare Systems: Nidhi Prasad and Mahima Chaurasia (2022) proposed this how cloud computing transforms healthcare by enhancing collaboration, scalability, security, and efficiency. It enables centralized access to digital health records, promotes telemedicine, improves patient care, and supports real-time analytics for personalized treatment. The cloud also ensures cybersecurity, cost-effective resource management, flexible access, supply chain optimization, and collaborative interoperability across the healthcare ecosystem, thus significantly improving data management and healthcare delivery.

2) Security framework for cloud based Electronic Health Record (EHR) system : According to Raghavendra Ganiga, Radhika M. Pai, Manohara Pai M. M., and Rajesh Kumar Sinha (2020) this paper proposes a security framework for cloud-based Electronic Health Record (EHR) systems that addresses confidentiality, integrity, and availability of health data. Threats are analyzed using STRIDE and DREAD, and solutions include Single Sign-On, attribute-based encryption, and advanced access control methods, all aligned with security standards to ensure safe and scalable healthcare data management.

3) Application of Cloud Computing in Healthcare: According to Neethu George and Surekha Mariam Varghese (2024), a Google Glass-based real-time scene analysis system was introduced to assist the visually impaired by enhancing their environmental awareness. This system captures live video through the Glass's camera, processes the visual data in real-time using a cloud server, and identifies objects using the SURF (Speeded-Up Robust Features) algorithm. Once recognized, object labels are relayed to the user through bone-conduction audio output, allowing users to receive spatial and semantic cues about their surroundings hands-free and non-intrusively.

4) The cloud-based health tracking and monitoring system with AWS: According to Vaishnavi Raosaheb Thoke and Prachi V. Kale(2021), this paper explores how AWS-based cloud platforms enable advanced health tracking and monitoring systems. Leveraging AWS cloud, healthcare providers can securely store, access, and manage vast amounts of electronic health records, patient profiles, lab reports, and daily health updates. The system employs robust encryption, digital signatures, and authentication protocols to safeguard data against unauthorized access and breaches.Patients and providers benefit from real-time monitoring, automated alerts, and remote consultations, facilitating efficient emergency responses and streamlined medical workflows. Additionally, AWS infrastructure supports diverse smart devices and mobile applications, allowing flexible health data collection, remote patient management, and integration with e-health platforms. The platform's scalability and reliability ensure uninterrupted service delivery, even as demands grow. Online patient portals provide easy access for scheduling appointments, viewing medical records, and checking the availability of services like blood banks. This approach not only ensures better healthcare delivery but also supports compliance, privacy, and cost reduction across modern healthcare systems.

5) An Optimized Role-Based Access Control Using Trust Mechanism in E-Health Cloud Environment : According to Ateeq Ur Rehman Butt and co-authorset (2023) this paper presents a cloud-based e-health access control approach that combines role-based access with dynamic trust evaluation, continuously monitoring user actions to adjust permissions. The trust mechanism rates users based on behavioral analysis, such as access time, feedback, and operations, allowing fine-grained, real-time access decisions. This model enhances security, adaptability, and efficiency compared to traditional static RBAC systems, while supporting secure and responsive healthcare data management.

6) Efficient and scalable patients clustering based on medical big data in cloud platform: According to Yongsheng Zhou and Majid Ghani Varzaneh *et al.*,the paper proposes a novel, efficient, and scalable patient clustering method based on medical big data in cloud platforms, addressing challenges brought by the rapid growth and distribution of patient data across multiple healthcare facilities. Their method employs Locality-Sensitive Hashing (LSH) for time-efficient, volume-scalable, and privacy-preserving integration and clustering of such data. The approach improves on existing methods by significantly reducing privacy leakage and computational time, while maintaining high clustering accuracy, as demonstrated through simulated experiments and comparisons with related literature. The work contributes toward enabling more effective and efficient healthcare data management in big data contexts under privacy constraints.

7) A Secure and Scalable Cloud Framework for Next-Generation Healthcare Systems: According to Mekala Ramesh (2024), the paper presents a secure and scalable cloud framework designed to address limitations of traditional healthcare data management such as slowness, inaccessibility, and data loss risks. This framework uses Attribute-Based Encryption (ABE) coupled with lossless compression to achieve secure, efficient storage and transmission of patient data on the cloud. Performance assessments reveal its advantages over conventional encryption methods like AES and RSA in terms of processing time, latency, and throughput. The study affirms the framework's potential to enhance patient data security, system scalability, and healthcare accessibility, while emphasizing compliance with privacy standards and its applicability in both urban and rural healthcare settings.

8) Privacy-Preserving Heterogeneous Federated Learning for Sensitive Healthcare Data: According to Yukai Xu, Jingfeng Zhang, and Yujie Gu (2024), the paper introduces a novel framework called Abstention-Aware Federated Voting (AAFV) that addresses the challenges of data privacy and model heterogeneity in federated learning for sensitive healthcare data. The framework integrates an abstention-aware voting mechanism with differential privacy techniques to enable collaborative and confidential training of heterogeneous local models. The authors demonstrate the method's effectiveness through experiments on real-world healthcare tasks, including diabetes prediction and in-hospital patient mortality, showing improved accuracy and enhanced privacy protection compared to traditional federated learning approaches.

III. PROPOSED METHODOLOGY

The proposed *Cloud-Based Smart,Secure and Scalable Healthcare System* integrates backend (Flask) and frontend (React) software modules to provide a secure platform for managing healthcare information

A. System Overview

The proposed Smart, Secure and Scalable Healthcare System integrates backend and frontend software modules to provide a secure and user-friendly platform for managing essential healthcare information. The system consists of a Python Flask backend API and a React frontend user interface, integrated with software modules for secure user authentication (JWT), Role-Based Access Control (RBAC), EHR management, and appointment booking. In addition, the platform now includes modules for medication management, the CareMate AI Assistant and MannMitra mental-health companion, a validated Mental Health Check-in workflow, and an enhanced Emergency Blood Request and notification system that together enable proactive wellness support and real-time emergency coordination.

B. Software Modules

1) *Backend API (Python Flask):* Backend API (Python Flask): A modular backend built using Python Flask and Flask Blueprints serves as the core processing unit. It handles all business logic, API requests, and database interactions via the Flask-SQLAlchemy ORM.
2) *Frontend UI (React):* A responsive single-page application built with React provides the user interface for patients, doctors, and admins. It uses Axios for RESTful API communication and Tailwind CSS for styling.
3) *Authentication and Authorization (JWT):*Secure user authentication is managed using JSON Web Tokens (JWT), implemented via Flask-JWT-Extended. This is combined with Role-Based Access Control (RBAC) to enforce strict permissions for Patients, Doctors, and Administrators.
4) *EHR Management:* This module provides functionalities for doctors to create, view, and edit Electronic Health Records (EHRs). It includes specific permissions, such as ensuring doctors can only edit EHRs they created, while patients have read-only access to their own records.
5) *Appointment and Medication Management:* Includes functionality for patients to book and cancel their own appointments , and for doctors to view their schedules and update appointment statuses. It also allows doctors to add medications linked to an electronic health record, which patients can then view.
6) *AI Chatbot Assistance Module (CareMate:)* An intelligent conversational assistant integrated using the OpenAI API. It provides personalized responses to patient queries, summarizes EHR and medication details, retrieves doctor availability, and supports natural language interactions. This module enhances usability and provides quick access to key health information.
7) *Emergency Blood Request Broadcast Module:* A critical-response communication system that allows the Admin to broadcast emergency blood requests to all registered users. When triggered, the system sends real-time alerts through the dashboard, enabling faster donor engagement during urgent medical situations.

C. System Flow Diagram

Fig. 1. System Flow Diagram: captures input, processes detection, and outputs audio feedback.

The diagram illustrates the system's architecture: a React single-page application (SPA) frontend communicates with a Python Flask backend server through a REST API. The Flask server, which interacts with an SQLite database via the SQLAlchemy ORM, processes these requests and handles authentication using JWT. This flow ensures that all data is handled securely and displayed to the user based on their authenticated role.

IV. SYSTEM ARCHITECTURE

The proposed *Cloud based Smart, Secure and Scalable HealthCare* system integrates backend and frontend software modules to provide a secure web application for managing healthcare data and delivering intelligent patient support. Figure 3.1 illustrates the system architecture, highlighting the interaction between the client-side application, the backend server, external AI services, and the database.

At the core of the system is the Python Flask back-end server, which serves as the main processing unit. The React single-page application (SPA) frontend continuously captures user inputs such as login credentials, appointment requests, EHR operations, conversational queries to the CareMate/MannMitra assistants, and Mental Health Check-in responses. These inputs are processed through dedicated software modules, structured as Flask Blueprints, running on the server, including Authentication and RBAC, EHR Management, Appointment and Medication Management, the Mental Health Check-in service, the Emergency Blood Request and notification module, and integration endpoints that securely relay AI-assistant requests to external models while enforcing role-based access and data protection.

- **Authentication and Authorization Module:** Implements secure user authentication using JSON Web Tokens (JWT) and enforces Role-Based Access Control (RBAC) for Patients, Doctors, and Administrators.
- **EHR Management Module:** Manages the creation, viewing, and updating of Electronic Health Records (EHRs), with specific permissions such as restricting edit rights to the creator doctor while providing read-only access to patients.
- **Appointment Management Module:** Handles the booking and cancellation of appointments by patients and allows doctors to view their schedules and update appointment statuses.
- **Medication Management Module:** Allows doctors to add medications linked to a specific EHR and enables patients to view their prescribed medications through their dashboard.
- **Mental Health Companion & Check-in Module:** Integrates the CareMate and MannMitra assistants for conversational support and connects them to a validated Mental Health Check-in workflow that records wellness scores and personalised suggestions.
- **Emergency Blood Request & Notification Module:** Enables authorized staff to broadcast urgent blood requirements and share curated support resources with registered users via email or in-app notifications.
- **Database Module:** Utilizes Flask-SQLAlchemy to manage all interactions with the relational database and Flask-Migrate for schema evolution, with a development setup on SQLite and a migration path to PostgreSQL/MySQL.

The processed information is returned through the React frontend as responsive, role-specific dashboards for Patients, Doctors, and Administrators. These views present EHRs, appointments, medications, and mental-health insights in a single interface. This ensures secure, convenient access to all core clinical and wellness features.

V. IMPLEMENTATION AND RESULTS

The Secure and Scalable Healthcare System uses a Python Flask backend (Python 3.9+) and a React (Vite) frontend. The backend relies on Flask-SQLAlchemy (SQLite in development, with a path to PostgreSQL) and Flask-JWT-Extended for secure authentication, along with services for Mental Health Check-in and Emergency Blood Requests. The React single-page application uses Axios and Tailwind CSS, and embeds the CareMate/MannMitra assistants to present EHRs, appointments, medications, and wellness insights in a unified interface.

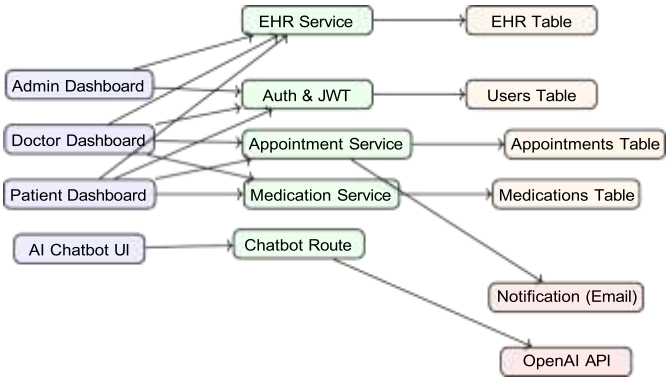

Fig. 1. System Architecture Diagram of the Smart, Secure, and Scalable Healthcare Platform.

A. Software Implementation

The system's software modules were developed using Python (Flask) for the backend and React for the frontend, and they include :

- *Backend API:*Python Flask with Blueprints serving a RESTful API , handling logic and database interactions via Flask-SQLAlchemy.
- *Frontend User Interface:* React dashboards using Axios for API calls and Tailwind CSS for styling.
- *Authentication and Authorization:*Combines Flask-JWT-Extended for JWT authentication with Role-Based Access Control (RBAC) for user permissions.
- *Appointment and Medication Management:* Functionality for patients to book/cancel appointments and view meds , and for doctors to manage schedules and add prescriptions.

B. Experimental Setup

- **Testing environment:** Development setup using a Flask backend, SQLite database, and React frontend in a web browser.
- **Testing methodology:** API endpoint validation using Postman and frontend-backend integration testing by performing actions in the UI.
- **Evaluation metrics:** Functional correctness (e.g., successful login, role-based access), security permission enforcement, API response times, and UI responsiveness/navigation.
- **Test inputs:** Valid and invalid data for CRUD operations (for EHRs, appointments, etc.) sent via Postman and the UI to test logic and permissions.

C. Results and Discussion

The system was evaluated in multiple scenarios. Observed outcomes include:

- **Patient Dashboard:** Successfully displayed EHR information, medications, upcoming appointments, and Mental Health Check-in results, and allowed users to book/cancel appointments and interact with the embedded CareMate/MannMitra assistants.
- **Doctor Functionality:** Allowed doctors to view appointments and manage patient EHRs with appropriate edit restrictions enforced, as well as review medications and relevant wellness information for their patients.
- **Appointment Management:** The appointment booking modal correctly filtered doctor availability and allowed patients to create and cancel appointments with immediate feedback.
- **Authentication & Security:** The system provided a secure login interface and enforced role-based access controls, ensuring that sensitive EHR and mental-health data were accessible only to authorized users.
- **Smart Wellness & Emergency Support:** The integrated mental-health companion and check-in flow improved engagement with wellness features, while the Emergency Blood Request module successfully broadcast test alerts to registered users.

The implementation and testing results show that the Smart, Secure and Scalable Healthcare System provides a functional and secure platform for managing EHRs, appointments, medications, and integrated mental-health features. Informal API testing in the development environment indicated response times of roughly 100 ms, and the React frontend reliably rendered role-specific dashboards for patients and doctors. The system consistently delivered a usable Patient Dashboard with key clinical and wellness information, along with essential Doctor functionalities under strict authentication and role-based access controls.

D. Performance Review

- **Responsive performance:** Achieved with informal Postman tests showing API CRUD operations under 100ms on SQLite. The React frontend also showed responsive rendering and smooth navigation.
- **System performance limitations:** The use of SQLite limits scalability and concurrent access, making it unsuitable for production-level deployment. Formal performance testing under simulated load was not conducted.
- **Frontend-backend integration:** Integration between the React frontend and the modular Flask backend (using Blueprints) is seamless, which was verified by testing user actions in the UI against the corresponding API calls.

III. Discussion

The implementation and testing of the Smart, Secure and Scalable Healthcare System demonstrate that integrating a Python Flask backend with a React frontend can provide a functional and secure platform for managing core healthcare interactions, including EHRs, appointments, medications, and integrated mental-health workflows. The combination of JWT-based authentication, strict Role-Based Access Control (RBAC), and a responsive user interface enables efficient and secure data management for both patients and clinicians, while embedded CareMate/MannMitra assistants and Mental Health Check-ins enhance patient engagement and proactive wellness support.

Several observations were noted during development:

- **Database Scalability:** The current implementation uses SQLite, which is not suitable for production-level scalability or handling high volumes of concurrent access.
- **Feature Completion:** Certain Doctor and Admin dashboard capabilities, as well as deeper analytics for mental-health and emergency modules, are not yet fully implemented.
- **User Interface Robustness:** Error handling, inline validations, and user feedback in the frontend can be further improved to make the application more comprehensive and user-friendly, especially around wellness assessments and emergency alerts.
- **System Interoperability:** The system currently lacks integration with external systems or established healthcare interoperability standards, such as FHIR, which would be important for exchanging EHR and mental-health data with other platforms.
- **Performance Validation:** Formal usability testing and performance/load testing under simulated production conditions were not conducted during this phase; only informal checks (for example, API latency and dashboard responsiveness) were performed.

Overall, the system demonstrates functional performance for core features and a robust security architecture. API response times in the development environment were responsive, generally under 100ms, and the React frontend provided smooth navigation. The results indicate that combining a modern, modular web stack with strong security principles can significantly enhance the accessibility, management, and security of health information for patients and providers. Future enhancements should focus on migrating to a production-grade database, cloud deployment, and FHIR standards integration to improve scalability, reliability, and interoperability in diverse, real-world healthcare environments.

IV. Conclusion and Future Work

The project successfully developed the core components of a Smart, Secure and Scalable Healthcare System. Key objectives were met, including implementing secure JWT-based authentication, distinct role-based access control, and functional modules for managing EHRs, appointments, medications, and integrated mental-health workflows. The Patient Dashboard provides a comprehensive view of clinical data and wellness insights, while foundational Doctor functionalities allow for patient data management with appropriate security restrictions and access to relevant wellness information. The system demonstrates the feasibility of building a secure, modern healthcare application using Python Flask and React, and directly contributes to SDG 3 (Good Health and Well-being) and SDG 9 (Industry, Innovation, and Infrastructure). Future work will focus on the following improvements:

- Migrating the database from SQLite to a production-grade RDBMS like PostgreSQL or MySQL.
- Deploying the application to a cloud platform (e.g., AWS, Azure, GCP) for scalability and accessibility.
- Exploring integration with healthcare interoperability standards like HL7 FHIR for data exchange.
- Implementing a more comprehensive logging and auditing system for security and compliance.

ACKNOWLEDGMENT

The authors express their sincere gratitude to Mr. Rajesh T. H, Assistant Professor, Department of Computer Science and Engineering, PES Institute of Technology and Management, Shivamogga, for his valuable guidance and support throughout this project.

REFERENCES

[1] N. Prasad and M. Chaurasia, "Role of Cloud Computing in Healthcare Systems," *International Journal of Trend in Scientific Research and Development (IJTSRD)*, vol. 6, no. 3, pp. 102-111, Mar.-Apr. 2022.

[2] R. Ganiga, R. M. Pai, M. Pai, and R. K. Sinha, "Security framework for cloud based electronic health record EHR system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 455-466, Feb. 2020.

[3] A. Bose, S. Roy, and R. Bose, "Application of Cloud Computing in Healthcare: A Comprehensive Study," *International Journal of Future Computer and Communication*, vol. 13, no. 2, pp. 23-33, Apr. 2024.

[4] V. R. Thoke and P. V. Kale, "The Cloud-Based Health Tracking and Monitoring System With AWS," *International Journal of Scientific Research in Science and Technology*, vol. 9, issue 4, pp. 155-162, Apr. 2021.

[5] A. U. R. Butt *et al.*, "An Optimized Role-Based Access Control Using Trust Mechanism in E-Health Cloud Environment," *IEEE Access*, vol. X, no. Y, pp. Z-Z, 20XX. DOI: 10.1109/ACCESS.2023.DOI.

[6] Y. Zhou and M. G. Varzaneh, "Efficient and scalable patients clustering based on medical big data in cloud platform," *Journal of Cloud Computing*, vol. 11, no. 49, 2022, doi: 10.1186/s13677-022-00324-3.

[7] M. Ramesh, "A Secure and Scalable Cloud Framework for Next-Generation Healthcare Systems," *IRACST International Journal of Computer Networks and Wireless Communications (IJCNWC)*, vol. 14, no. 4, pp. 261-268, 2024.

[8] Y. Xu, J. Zhang, and Y. Gu, "Privacy-Preserving Heterogeneous Federated Learning for Sensitive Healthcare Data," 2024.