# Healthcare Management System using Next.js

Ms. Shivaranjini C[1], Sanskar Dewangan[2], Shobhit Singh[3], Supriya Singh[4]
*Dept. of Computer Science and Engineering, Sir M. Visvesvaraya Insitute of Technology Bengaluru, KA, India*

***Abstract*— Efficient healthcare appointment management is a critical challenge in modern clinical operations. Traditional systems are often inefficient, fragmented, and lack automation. This paper presents a robust, responsive web-based Health- care Management System, developed using Next.js, Appwrite, TailwindCSS, and Twilio, which automates patient registration, appointment booking, confirmation, and SMS-based notifications. The application supports admin-level monitoring, performance tracking via Sentry, and secure file upload with Appwrite Storage. The system aims to improve patient engagement, reduce no-shows, and streamline administrative workflows. Results demonstrate its efficiency, responsiveness, and scalability, making it an ideal candidate for real-world deployment and future enhancements.**

***Index Terms*— Healthcare Management, Next.js, Appwrite, Twilio, SMS notifications, Appointment booking, Sentry, Scalability**

## I. INTRODUCTION

In today's digitally connected world, healthcare services must be accessible, reliable, and efficient. The core of this experience lies in effective communication and appointment handling between patients and healthcare providers. Traditional appointment systems are typically paper-based or built on outdated software, leading to errors, delays, and dissatisfaction.

This paper introduces a modern Healthcare Management System built using Next.js, focusing on seamless appointment scheduling, confirmation via SMS, and responsive design. The system is powered by Appwrite for backend operations, Twilio for real-time alerts, and Sentry for application monitoring.

## II. PROBLEM STATEMENT

Existing systems often suffer from one or more of the following issues:

- Lack of real-time scheduling and confirmation mechanisms.
- Limited accessibility on mobile devices.
- Manual appointment tracking and high risk of human error.
- No centralized performance monitoring or feedback sys- tem.
- Poor integration of file upload/storage and patient communication tools.

## III. LITERATURE REVIEW

Past studies in healthcare technology have emphasized the shift from manual to digital appointment systems. Several solutions have been proposed using platforms like PHP, Laravel, and Firebase. However, these often come with:

- Steep learning curves,
- Scalability issues,
- Fragmented third-party service integration.

React-based architectures improved modularity but lacked server-side rendering support. Next.js bridges this gap by offering performance-focused server-side rendering, routing, and a superior developer experience.

## IV. SYSTEM ARCHITECTURE

Our system addresses these issues by offering a modular, scalable, and user-friendly appointment management platform that is accessible across all devices and integrates with modern cloud and messaging services.

The system follows a modular multi-layered architecture:

### *A. Frontend Layer*

Technology: Next.js + TypeScript + TailwindCSS
Role: UI rendering, routing, and dynamic interaction. Responsiveness: Tailored for all devices.

### *B. Backend Layer*

Technology: Appwrite
Role: Authentication, Database, Storage
Collections:

- Patient: Stores user profiles.
- Appointment: Tracks appointments, statuses.

### *C. Notification Layer*

Service: Twilio

Function: Sends SMS to patients upon appointment booking or cancellation.

*D. Monitoring Layer*

Tool: Sentry

Function: Real-time error tracking and application performance monitoring.

## V. FEATURES OVERVIEW

TABLE I: SYSTEM FEATURES OVERVIEW

| Feature | Description |
|---|---|
| Patient Registration | New users can register and create profiles securely. |
| Appointment Booking | Patients can book appointments based on availability. |
| Admin Controls | Admins can view, confirm, reschedule, or cancel bookings. |
| SMS Confirmation | Real-time SMS updates sent to users. |
| File Upload | Patients can securely upload medical documents. |
| Fully Responsive | Compatible with desktop, tablet, and mobile devices. |
| Sentry Monitoring | Tracks system logs and user-side errors. |

## VI. COMPARATIVE ANALYSIS

TABLE II: COMPARISON BETWEEN TRADITIONAL AND PROPOSED SYSTEM

| Parameter | Traditional Systems | Our System |
|---|---|---|
| Accessibility | Limited to clinics or apps | Web-based, responsive |
| Confirmation | Manual or phone calls | Automated via SMS |
| Scalability | Local database, less flexible | Cloud-native (Appwrite) |
| Monitoring | Absent or minimal | Integrated Sentry |
| Ease of Use | Varies | High (modern UI with Tailwind) |

## VII. LIMITATIONS & FUTURE ENHANCEMENTS

*A. Limitations:*

- Limited role-based access (only admin & patient roles)
- No payment gateway integration
- Basic appointment time selection (not real-time calendar)

*B. Future Enhancements:*

- Add teleconsultation module (via WebRTC)
- Role-based dashboards (doctor, staff, admin)
- Integration with payment gateways (e.g., Razorpay, Stripe)
- Mobile app version using React Native or Flutter

## VIII. SECURITY CONSIDERATIONS

The system ensures secure data handling by implementing strong encryption protocols for data transmission and storage. Appwrite provides secure authentication methods like JWT, and the entire system follows the best practices in data protection, including compliance with relevant healthcare standards (e.g., HIPAA).

## IX. USER INTERFACE DESIGN

The UI of the system was designed with usability in mind, using TailwindCSS to ensure responsiveness across all devices. The design follows modern UX principles, such as clear navigation, accessibility, and easy-to-understand workflows for both patients and healthcare providers.

## X. INTEGRATION WITH OTHER HEALTHCARE SYSTEMS

The system is designed to be easily integrated with other healthcare solutions, including Electronic Health Records (EHR) and telemedicine platforms. With minimal modifications, it could interface with these systems, further streamlining healthcare workflows.

## XI. COST ANALYSIS

By using modern web technologies, such as Next.js, Appwrite, and Twilio, the cost of infrastructure is minimized. Appwrite offers a scalable backend with a free tier, while Twilio's pay-per-use model makes SMS notifications cost- effective.

## XII. REAL-WORLD USE CASES

This system is ideal for deployment in small to medium- sized clinics, telemedicine services, or even large hospital networks. Future versions could

include features tailored for enterprise use, such as advanced reporting and analytics.

## XIII. CONCLUSION

The Healthcare Management System presented here demonstrates a practical and scalable approach to digitizing patient appointment workflows. It is fast, responsive, and easy to maintain due to the modular tech stack. With minor enhancements, the system can serve small clinics, hospitals, or even telemedicine startups. The project highlights how modern web technologies can reshape healthcare service delivery through automation and user-friendly design.

The authors would like to thank [Insert names of people or organizations] for their support in this research.

## REFERENCES

[1] S. Kumar, J. Ojha, M. M. Tripathi, and K. Garg, "Integrated e-healthcare management system using machine learning and Flask," *ResearchGate*, 2023.

[2] Twilio, "Appointment Management with EHR Integration," *Twilio CodeExchange*, 2023.

[3] Appwrite, "Health API Reference," *Appwrite Documentation*, 2023

[4] Sentry, "Release Health," *Sentry Documentation*, 2023.

[5] Pagepro, "Building Healthcare Platforms with Next.js," *Pagepro Blog*, 2023

[6] A. Arnob, "HealthCare-NextJS-Website," *GitHub Repository*, 2023

[7] JETIR, "Nexpulse- A Nextjs Health Support Platform with Real- Time Video Consultations," *JETIR*, 2025.

[8] Sentry, "Application Monitoring Resources," *Sentry Resources*, 2023