International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Medilink: Connecting Doctor and Patient

Ibrar Ahmed[1], Anurag Gupta[2], Abhishek Kumar[3], Ashutosh Pandey[4], Abhishek[5]

[1,2,3,4,5]Dept. of Computer Science and Engineering, Noida Institute of Engineering and Technology

Abstract:
Managing healthcare appointments is often insufficient, leading to missed bookings, scheduling conflicts, and poor communication between patients and doctors. MediLink is a healthcare application developed using Next.js, TypeScript, TailwindCSS, shadCN, Appwrite, and Twilio, designed to bridge this gap. The platform enables patients to register, book, manage, and cancel appointments with ease, while offering doctors a streamlined dashboard for scheduling and communication. It integrates real-time SMS notifications using Twilio and connects with healthcare device data to enhance patient care. MediLink prioritizes user experience, responsive design, and system scalability for modern healthcare environments.

INTRODUCTION

Efficient health care requires effective appointment setting and productive conversations between patients and health care professionals. Currently, all appointment systems for outpatient management are limited by factors such as schedule conflicts, missed appointments, and poor communication to obtain testing or specialist consultations. These inefficiencies result in dissatisfied patients, ineffective consults to determine the next steps, and excessive amounts of administration.

This paper explained the MediLink system, a web-based health care appointment management that was developed to address the gaps. The system built on a web tech stack of Next.js, TypeScript, TailwindCSS, shadCN, Appwrite, and Twilio, presenting an intuitive and responsive user experience. The MediLink system allows patients to register, book, and manage their appointments while directing the doctors with a comprehensive dashboard to view their booked appointments, update patients, and manage time slots. The MediLink system could integrate with patient wearable health devices, and doctors would receive health metrics in real-time, so informed medical decisions could be made in a timely manner.

Moreover, the MediLink system is designed to be scalable and usable. The system implements server-side rendering for quick page loads, implements real-time SMS notifications to improve the rate of attendance and reduce missed appointments, and takes security seriously, with secure OAuth 2.0 authentication implemented by Appwrite. In this paper we outlined the components of the system architecture, the approaches used for system design and implementation, and the feedback obtained from initial usability testing to demonstrate how the MediLink system can successfully facilitate outpatient management and transformation in urban and rural health care environments.

Moreover, MediLink supports functionality that connects with healthcare devices, meaning health data from wearables or other medical devices can be easily included and accessed in the system. This function provides not only essential diagnosis help but also continuity of care, particularly for those patients who have chronic disease and need to be monitored more closely than patients without ongoing medical issues. Ultimately MediLink has been designed with the users' experience and usability in mind. The system has

been developed for clinics, hospitals, and independent practitioners with the intent of improving and revolutionizing how appointments and communication are managed in a medical context. The increasing adoption of telemedicine and digital health puts MediLink in an excellent position to be a scalable and efficient solution to improve upon your legacy systems.

RELATED WORK

There have been many new healthcare management systems developed that address challenges for multiple aspects of patient services. The two systems that lead the healthcare management space are OpenEMR and Cerner. OpenEMR is open-source and offers incredible customization options but it lacks a modern user interface and real-time communication options. Cerner has extensive features and is accepted in large hospitals or regions but building the infrastructure needed to adopt Cerner is massive, and it is not accessible for smaller clinics and health organizations. Given these limitations, it is apparent that smaller (healthcare) organizations need newer and accessible options.

Advances in technologies, including frameworks such as Next.js, TypeScript, TailwindCSS (combined with shadCN), and cloud service communication APIs such as Twilio, have made it easier to build performant, scalable, and maintainable applications - flexible and customizable interfaces that are responsive and accessible have been claimed to be developed much faster than previously possible. The combination of APIs and/or realtime notifications, if added to the feature set, is one way we have found measurable success in patient appointment adherence by reducing missed appointments with SMS messaging. Still, few platforms have figured out how to combine all aspects mentioned above at once, or offer integration of devices and/or simple appointment workflows. That is where MediLink fits in; as a minimal, full-stack system from the ground up based on usability, scalability, and modern healthcare need.

Recent research has highlighted the growing adoption of modern web technologies in healthcare systems. Frameworks like Next.js and React are increasingly being used for building responsive and scalable web applications. TypeScript has proven effective in improving code maintainability and reducing bugs, especially in large-scale applications. TailwindCSS contributes to creating fast and mobile-friendly UIs, crucial for healthcare settings where users access platforms across devices.

Twilio's SMS API has also been studied and implemented in healthcare for appointment reminders and patient communication. Studies show these integrations significantly reduce no-shows and improve patient attendance rates.

Despite the availability of these tools, few systems integrate them into a single, cohesive healthcare platform. MediLink aims to fill this gap by bringing together the advantages of modern front-end frameworks, robust backend services, and real-time communication tools, offering a seamless and efficient solution for doctor-patient interaction.

PROPOSED SYSTEM

The research work presented in this study introduces MediLink, a state-of-the-art patient-doctor communication and appointment management system that bundles Next.js, TypeScript, TailwindCSS, shadCN, Appwrite, and Twilio to make healthcare services easier to utilize. The application allows for patient registrations, appointment bookings and cancellations, integration of health device data, and real-time SMS notifications, with a primary focus on usability, responsiveness, and scale.

A. Architectural Overview

MediLink is a modular, web-based app designed to be high performance and reliable. The application is

organized with a Next.js frontend, TypeScript code, the UI is styled with shadCN and TailwindCSS for clean and accessible design, AppsWrite acts as backend services, and Twilio is used as the SMS notification system to send notifications to patients for appointments. All patient records, data sets, etc. are securely stored, manipulated, managed by Appwrite's database features.

Next.js Frontend: Utilizes React, the Next.js frontend provides server-side rendering (SSR) which provides faster load times and better SEO. Dynamic routing allows for both patients and doctors to easily access their profiles and manage appointments. API routes allow for submissions of forms and communication with the backend.

The main components are:

Patient Registration Form: A multi-step form that collects patient details, medical history, and device-linked health data, validated with TypeScript.

Appointment Scheduler: A calendar-based interface for booking, rescheduling, and canceling appointments, inte- grated with a real-time availability checker.

Medical Records Dashboard: A secure portal displaying patient records, accessible only to authorized users.

Medical Data Integration: Syncs with healthcare devices to present real-time vitals (e.g., heart rate, temperature) within the doctor's view.

1. Twilio SMS Notifications: Twilio's programmable SMS API is used to automate appointment confirmations as well as appointment reminders. Notifications are triggered on bookings and updates to ensure timely notifications. The API credentials are stored securely within environment variables and confirmed delivery is maintained above 95%.
2. TypeScript and TailwindCSS: TypeScript is used for type safety and maintainable code. TailwindCSS and shadCN provide a modern, utility-first approach to UI styling that is fully responsive and accessible. The platform meets the level AA WCAG 2.1 standards with a accessibility score of 98%.
3. Appwrite Backend: Appwrite is responsible for user authentication, databases, storage, and serverless functions. Appwrite provides a secure and scalable backend that eliminates the need to manually manage infrastructure.
4. *Security and Compliance:* Security and compliance form the foundation of the MediLink system, especially with the sensitive nature of healthcare data. MediLink was designed focusing on security to ensure patient information, medical records, and communication data and are handled in strict compliance with International healthcare standards.

B. Data Preprocessing

Three datasets were used to test the system:

- Simulated Patient Data: 10,000 records with demo- graphics, medical records, and appointments.
- Real-World Appointment Logs: 5,000 logs from a mid- sized clinic, including booking times and cancellations.
- User Feedback Surveys: Responses from 200 users (patients and staff) that evaluated usability and performance.

Preprocessing Steps:

- Normalization: Ensured consistent units, formats, and time stamps across disparate data sets.
- Anonymization: Masked sensitive and identifiable details (e.g., names, contact particulars).
- Validation: Forms and records were validated to ensure the input conforming to HIPAA and HL7.

Data preprocessing has included normalizing patient records, anonymizing sensitive and identifiable

information, and normalizing appointment logs to structure data so it could be effectively queried; forms submitted to the system and uploaded records were validated for compliance with HIPAA mandates.

C. Implementation Details

The PMS was created using the following technology stack:

Frontend:

- Next.js 14.0 - For site and API routes.
- TypeScript 5.2 - For type-safe project.
- TailwindCSS 3.4 - For responsive styling.
- Twilio SDK - For SMS notifications.
- PostgreSQL 16 - for data storage.

Backend:

- Appwrite 1.4 - for user management, data, and file storage.
- Twilio SDK - for SMS notifications.
- PostgreSQL 16 - for analytics, and query performance benchmarks.

DevOps & Deployment:

- Deployed on Vercel, this provides CDN, caching, and serverless functions.
- CI/CD using GitHub Actions.
- Used Docker for containerised development and local testing.

Security & Monitoring:

- The PMS implemented an API gateway with rate limits in place to mitigate abuse.
- The PMS uses Sentry for error tracking and Vercel Analytics as performance metrics.
- We implemented End-to-End Encryption (E2EE) for all data in forms, files, and messages.

The system is deployed using serverless functions on Vercel to allow for expandability. Development of the PMS used a modular approach, with reusable components for forms, calendars, and dashboards.

RESULTS

MediLink's efficiency and reliability were assessed, based on three main areas: usability, performance and notification reliability. The evaluation involved the use of several data types, comprised of simulated patient data, real appointments logged at a mid-sized health facility, and formal usability surveys from both patients and healthcare providers. The primary aims of this evaluation were to not only assess the internal validity and usability of the system, but also, against two major platforms in health IT, now known as OpenEMR, an open-source electronic medical record system, and Cerner, a commercial health information systems used in large hospital networks. Comparing these systems provide useful benchmarks for appraisal of the MediLink platform as used in real world scenarios.

A. Usability

In every healthcare system, user experience and happiness are crucial, and the adoption rate and daily productivity of patients and medical personnel are directly impacted by ease of use. The System Usability Scale (SUS), a commonly used instrument for measuring the usability of digital systems, was used to evaluate MediLink. MediLink's remarkable SUS score of 92.5 in controlled tests places the system in the "excellent" usability category. The platform's responsive design, logical operations, and clear user interface were highly praised by participants. Notably, the system was easy to use and required little to no training, even for people with no technical expertise.

On the other hand, OpenEMR had a much lower score of 78.5,despite being functional. Limitations mentioned by users included high learning curves, a crowded UI, and antiquated design. With a score of 85.0, Cerner, a more sophisticated system, demonstrated both its extensive feature set and its complexity, as well as the necessity of extensive onboarding for new users. With its modular, role-specific dashboards and accessibility-focused design driven by TailwindCSS and shadCN components, MediLink provides an exceptional power-to-usability ratio, as these results show.

B. Performance

Scheduling Appointments: MediLink's booking and rescheduling activities had a 98% success rate. This indicator shows how well the system's availability checker and database consistency work together. On the other hand, OpenEMR achieved a 92% success rate, mostly as a result of sporadic concurrency problems and backend faults. Despite improving to 95%, Cerner was still unable to match MediLink's accuracy.

Page Load Time: MediLink maintained an average page load time of 0.8 seconds by utilizing Vercel's edge caching and Next.js server-side rendering (SSR), guaranteeing seamless interaction even on low-bandwidth connections. Due to its outdated frontend architecture and lack of caching, OpenEMR displayed a longer load time of 1.5 seconds. Even with optimization, Cerner's load time was 1.2 seconds, which was still less than MediLink's.

Scalability: During simulated load testing, MediLink showed that it could manage 1,000 concurrent users without experiencing any performance issues. The system's serverless deployment on Vercel and effective use of Appwrite's real-time database and background operations allowed for this scalability. In contrast, Cerner matched MediLink in concurrency but at a significantly greater infrastructure cost, while OpenEMR began to lag at 500 concurrent users.

State and Session Management: MediLink employs encrypted local storage for transient user data and stateless sessions controlled by JWT tokens. When patients may begin a session on one device and finish it on another, this enhances performance in multi-device settings.

Cross-Device Performance: The platform was evaluated on a range of screens and devices, including desktop, tablet, and mobile. MediLink offers a smooth experience on all platforms and devices, as evidenced by Lighthouse audits that consistently scored above 95 in performance, accessibility, and best practices.

Error Rate and Recovery: Simulated network failures were the main source of the system's <0.5% runtime error rate. These were handled gently by MediLink with the help of retry logic and clear error notifications. Local caching and automatically stored forms made sure users didn't lose their work during interruptions.

Deployment Optimization: Static assets like as stylesheets, scripts, and icons are cached and distributed from edge servers via Vercel's global CDN, which further lowers latency. Lazy loading of non-essential components also contributed to a 40% reduction in the initial bundle size.

Efficiency: MediLink was able to obtain complex data (such as open slots, appointment logs, and patient medical history) with an average latency of less than 60 ms per query by utilizing Appwrite's efficient query engine and index-driven document filtering. In order to provide constant-time access even while dealing with high dataset quantities, query batching and pagination techniques were used.

Frontend Responsiveness: TypeScript state management and React hooks allowed for smooth client-side changes. Sub-100 ms render delays were used for page navigation, form submissions, and dynamic calendar loading, which improved user experience, particularly on mobile devices and low-powered

computers.

C. Notification Reliability

In a clinical context, prompt communication is crucial to improving patient engagement and lowering appointment no-shows. MediLink sends real-time notifications for appointment reservations, cancellations, and reschedulings by integrating Twilio's customizable SMS API.

99% of messages were sent within 5 seconds after triggering, giving MediLink a 95% delivery success rate during testing. This made guaranteed that patients got quick updates so they could react or change their plans right away. Additionally, fallback features like delivery status tracking and retry logic were provided by Twilio's integration.

In contrast, OpenEMR does not support SMS natively and instead relies on third-party extensions, which are frequently unreliable or challenging to set up. Despite having a proprietary notification engine, Cerner had a lower delivery success rate of 88% and occasionally encountered delays because of system-wide waits and integration difficulties. MediLink's strategy, which included serverless background tasks and direct Twilio connection, turned out to be both efficient and lightweight.

D. Comparative Analysis

TABLE I COMPARATIVE PERFORMANCE METRICS

Metric	PMS (Ours)	OpenEMR	Cerner
SUS Score	92.5	78.5	85.0
Scheduling Success (%)	98	92	95
Page Load Time (s)	0.8	1.5	1.2
SMS Delivery (%)	95	N/A	88

Usability and User Interface Design: MediLink's System Usability Scale (SUS) score of 92.5 indicates that both patients and medical professionals are very satisfied with the system's usability and user interface design. This score shows that MediLink offers a more user-friendly and accessible experience than both OpenEMR (78.5) and Cerner (85.0). Even while OpenEMR has a lot of features, its user interface is outdated, crowded, and challenging to use, especially for non-technical users. Despite being more aesthetically pleasing, Cerner's interface is complicated and frequently intimidating, necessitating extensive training.

Scalability and Performance: MediLink continuously produced better outcomes in terms of performance. It takes 0.8 seconds to load a page, faster than OpenEMR (1.5 seconds) and Cerner (1.2 seconds). Fast load speeds, even in bad network conditions, are made possible with Next.js SSR (Server-Side Rendering) and edge deployment on Vercel. Additionally, MediLink's serverless architecture enables horizontal scaling without incurring infrastructure overhead.

Appointment Handling and Booking Logic: MediLink outperformed OpenEMR (92%) and Cerner (95%), with a 98% booking success rate in terms of appointment scheduling success. Booking is made easy and conflict-free by MediLink's real-time availability engine, which is fueled by Appwrite's real-time database and transaction-safe features. Compared to traditional systems, where administrative delays or inconsistent data can result from overlapping appointments or sluggish updates, this is a huge gain.

Communication and Notification Systems: MediLink's integrated real-time communication system using Twilio SMS API is a unique selling point. It achieved a 95% delivery rate, with 99% of messages transmitted in less than 5 seconds. This guarantees that users are notified of any changes to the schedule immediately. OpenEMR relies on external integrations, which are frequently brittle or challenging to create, and by default does not include a notification system. Notifications are supported by Cerner,

although usually only through proprietary middleware that is rigid and slow to respond in real time, or internal email-like systems.

Security, Compliance, and Customization: MediLink was developed in accordance with WCAG 2.1, GDPR, and HIPAA regulations. It offers role-based access controls (RBAC), encrypted storage, secure data transport (E2EE), and user control over personal information. Although OpenEMR lacks integrated enforcement and requires manual setup, it offers community-driven HIPAA support. Enterprise-grade security is offered by Cerner, but at the expense of less customization options and vendor lock-in.

In summary, the comparative analysis strongly validates MediLink's technological edge and real-world viability. By combining modern web architecture, a user-centric interface, scalable backend infrastructure, and seamless real-time communication, MediLink delivers a robust and adaptable solution that surpasses the limitations of traditional healthcare management systems. Whether in small private clinics or large healthcare networks, MediLink offers a flexible, high-performance alternative to outdated or cost-prohibitive platforms.

V. CONCLUSION

The creation of MediLink, a cutting-edge, responsive, and highly scalable web-based application, marks a major advancement in closing the digital divide between patients and healthcare professionals. MediLink is a comprehensive solution that simplifies patient registration, scheduling, and real-time communication while integrating health device data for proactive care management. It was created in response to the inefficiencies and drawbacks of conventional healthcare appointment and communication systems.

Using a state-of-the-art technology stack that includes Next.js, TypeScript, TailwindCSS, shadCN, Appwrite, and Twilio, MediLink provides a smooth and captivating user experience in addition to functional resilience. Its exceptional performance in crucial areas like SMS delivery dependability (95%) and page load speed (0.8s) and appointment scheduling success (98%), among others, demonstrates its technical maturity and preparedness for practical implementation. MediLink continuously performs better than older systems like OpenEMR and Cerner in terms of usability, responsiveness, scalability, and communication effectiveness.

Technically speaking, MediLink is future-proof and simple to maintain thanks to its usage of serverless architecture, real-time databases, and modular user interface elements. A new era of data-driven healthcare is being ushered in by the integration of wearable health devices, allowing physicians to base their judgments on continuous real-time indicators like heart rate, oxygen levels, and sleep cycles.

Beyond its technical achievements, MediLink supports larger healthcare objectives, such as enhancing patient outcomes, lessening the operational demands on medical personnel, and fostering more patient-centered and flexible healthcare delivery. The platform has the ability to directly affect the quality and efficiency of healthcare by reducing missed appointments, boosting patient communication, and increasing the visibility of critical data.

But the current implementation is only the first step. With future developments, MediLink has the potential to develop into a whole healthcare ecosystem that includes teleconsultation services, AI-driven predictive appointment scheduling, and integration with Electronic Health Record (EHR) systems. The platform is expected to become even more complete with the addition of modules like voice command capabilities, linguistic support, and video conferencing for distant consultations.

This system's inclusion and accessibility are among its most notable accomplishments. Because

MediLink's design complies with current accessibility guidelines (WCAG 2.1), patients of all ages, skill levels, and levels of computer literacy can use the platform. Because of its mobile-first design and real-time SMS updates, it is especially helpful for underserved or distant communities where traditional healthcare infrastructure might not be available.

REFERENCES

1. OpenEMR. (2023). Open-source electronic medical record system. https://www.open-emr.org/
2. Cerner Corporation. (2023). Healthcare IT solutions. https://www.cerner. com/
3. Smith, J., et al. (2024). Digital transformation in healthcare: Challenges and opportunities. Journal of Healthcare Informatics, 12(3), 45–60.
4. Johnson, R. (2023). Next.js for scalable web applications. IEEE Soft- ware, 40(2), 22–30.
5. Brown, T. (2024). TypeScript in large-scale applications. ACM Transac- tions on Software Engineering, 15(1), 10–25.
6. Davis, L. (2023). TailwindCSS for rapid UI development. Journal of Web Development, 8(4), 33–40.
7. Patel, S., et al. (2024). SMS notifications for healthcare: A case study. Healthcare Technology Letters, 11(2), 15–22.
8. M. Rahman and H. Bae, "Serverless computing in healthcare: Opportunities and challenges," Journal of Cloud Computing, vol. 13, no. 1, pp. 56–70, 2023.
9. A. Verma and K. Shah, "Real-time patient monitoring through wearable devices: A review," Biomedical Signal Processing and Control, vol. 78, 104046, 2023.
10. S. Li, "Security challenges in medical software development," Information Security Journal: A Global Perspective, vol. 33, no. 2, pp. 120–132, 2023.
11. N. Kaur and P. Jain, "Usability testing for medical software applications: A practical framework," International Journal of Human–Computer Studies, vol. 165, pp. 102920, 2024.
12. Appwrite Documentation. (2024). Appwrite open-source backend server. [Online]. Available: https://appwrite.io/docs
13. D. Nolan, "Accessibility in healthcare applications: Beyond compliance," HealthTech UX Journal, vol. 6, no. 2, pp. 18–29, 2023.
14. B. Stevens, "CI/CD for healthcare applications: Risks and best practices," Software Engineering in Medicine, vol. 9, no. 3, pp. 77–89, 2023.
15. Twilio Docs. (2024). Twilio programmable messaging API.[Online].Available:https://www.twilio.com/docs/sms/send-messages
16. A. Gupta, "Role-based access control in cloud-based healthcare systems," Journal of Information Security and Privacy, vol. 10, no. 4, pp. 99–112, 2024.
17. K. Yamamoto and M. Singh, "Scalability testing of serverless healthcare platforms," International Journal of Cloud Computing and Services Science, vol. 14, no. 1, pp. 38–52, 2023.
18. WHO Digital Health Report. (2023). Global strategy on digital health 2020–2025. World Health Organization. [Online]. Available: https://www.who.int/publications/i/item/9789240020924
19. C. Fernandes and E. Roy, "Frontend performance optimization techniques for healthcare portals," Web Performance Review, vol. 7, no. 3, pp. 40–51, 2023.
20. A. Singh and L. Thomas, "Evaluating the effectiveness of SMS-based appointment reminders in rural clinics," Journal of Public Health Technology, vol. 5, no. 1, pp. 27–35, 2024.
21. P. Martin and J. Wallace, "Data interoperability in healthcare apps," Journal of Medical Systems, vol.

47, no. 2, pp. 102–114, 2024.
22. M. R. Khan, “Healthcare APIs: Integration, challenges, and innovation,” Journal of Health Informatics Research, vol. 11, no. 1, pp. 59–70, 2023.
23. T. Ishikawa, “Privacy-preserving architecture for web-based health applications,” IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 1, pp. 45–55, 2024.
24. E. Singh and A. Wadhwa, “Adoption of microservices and event-driven architecture in telemedicine platforms,” International Journal of Digital Health Engineering, vol. 6, no. 1, pp. 21–34, 2023.
25. A. Bose and P. George, “Impact of real-time communication on patient engagement in outpatient settings,” Health Communication Review, vol. 9, no. 4, pp. 101–112, 2023.