© NOV 2025 | IRE Journals | Volume 9 Issue 5 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I5-1712066

Doctor Appointment Booking System Using MERN Stack: A Systematic Review and Analysis

SALONI PATIL[1], SHRUTI ROHIDAS RANSING[2], SHILPA SANJAY VANTALKAR[3], PURVII SANJAY DHAYBAR[4]
[1, 2, 3,] *4Navsahyadri Group of Institutes – [NESGI], Pune*

Abstract-

Scope: The healthcare sector is increasingly reliant on digital appointment-scheduling platforms to overcome inefficiencies of manual/phone-based booking (e.g., long waiting times, poor resource utilisation). This paper focuses on digital appointment systems specifically built using the MERN stack (MongoDB, Express.js, React.js, Node.js) architecture.

Methodology: A systematic review of published works from 2020–2025 that implement or analyse MERN (or closely related full-stack JavaScript) solutions for doctor/health-provider appointment scheduling.

Findings: The MERN stack offers benefits such as single-language full-stack development (JavaScript/TypeScript) and rapid front-end/back-end integration. However, critical challenges emerge: NoSQL databases like MongoDB may complicate relational modelling (doctor-patient-appointment links); authentication/authorization practices vary widely; and scalability/security issues (e.g., regulatory compliance, data integrity) are often under-analysed.

Conclusion: This review identifies key gaps in existing MERN-based appointment-systems research—particularly around data integrity, security/compliance, and standard healthcare-data interoperability—and proposes future research avenues (e.g., integration with FHIR, microservices/serverless, stronger access control).

I. INTRODUCTION

1.1 Background on Healthcare Scheduling

Begin by discussing how traditional appointment-booking systems (paper, phone calls) suffer from delays, resource mismatch (e.g., over-booking or under-utilisation), poor patient experience (long waits), and limited analytics (no real-time dashboards). Digital systems aim to reduce these inefficiencies, improve patient satisfaction, and optimise clinician/clinic workloads.

1.2 Introduction to Full-Stack Architectures

Define the MERN stack: MongoDB (NoSQL database), Express.js (server framework), React.js (front-end library), Node.js (JavaScript runtime). Emphasise the "JavaScript everywhere" advantage—shared language across client & server, enabling faster development. Note the popularity of full-stack JavaScript frameworks for interactive web apps.

Thesis statement: This paper systematically analyses the literature on MERN-based doctor appointment-booking systems to assess their architectural design, feature completeness, performance, and security, and to identify key limitations and areas for advanced research.

II. LITERATURE REVIEW

Structure this section into logical categories, summarise existing works, then synthesise.

2.1 Category A: Feature-Centric Implementations

For example:

- The paper "Doctor Appointment Booking and Handwriting Recognition System" (2025) uses MERN + OCR to digitise prescriptions and integrate booking. IJISRT+1
- "Development of Online Doctor's Appointment System for Bengaluru" uses MERN with JWT authentication and real-time booking. ijsci.com

Summarise typical features: dashboards for patients/doctors, search/filter of doctors, booking/cancellation workflows, reminders/notifications, payment integration.

2.2 Category B: Performance & Scalability Studies

E.g., "Real-Time Appointment Scheduling Using MERN Stack and WebSockets" (2025) uses WebSockets for immediate updates and avoids

double-booking. Jetir
Discuss metrics like API latency, concurrent booking handling, real-time slot availability.

2.3 Category C: Security & Compliance

Papers that discuss JWT authentication, role-based access control, encryption of data in transit/at rest. (e.g., "Optimizing Healthcare Scheduling Through Machine Learning…" uses MERN + JWT + Bcrypt for password hashing. ISJEM Journal
Few works fully address regulatory compliance (HIPAA, GDPR) or the deeper issues of NoSQL transactional integrity.

2.4 Critical Synthesis

Here is where you move beyond summarising:

- Many papers focus on "we built it and it works" but lack deep evaluation of security, compliance, long-term data integrity.
- Example: heavy reliance on MongoDB in MERN systems—yet few discuss the relational nature of doctor-patient-appointment schemas and the possible problems (e.g., lack of ACID transactions, referential integrity).
- Many implementations omit advanced features (telemedicine, AI scheduling, FHIR interoperability).
- Scalability studies exist but are limited to prototypes, not full real-world deployment under heavy load.

III. METHODOLOGY (REVIEW PROCESS)

3.1 Search Strategy

Describe which databases (e.g., IEEE Xplore, Springer, ACM Digital Library, Google Scholar) were searched; keywords like:
"MERN doctor appointment", "MERN healthcare scheduling", "MongoDB healthcare system", "full-stack JavaScript appointment booking".

3.2 Selection Criteria

Inclusion: Papers from 2020–2025, implementing or analysing systems using MERN (or closely related stack) for doctor/health-appointment booking, with at least a prototype or implementation.

Exclusion: Papers only proposing conceptual designs without implementation; systems using non-JavaScript back-ends (e.g., Django, Spring Boot) unless explicitly MERN variant; unrelated booking domains (e.g., hotel rooms) unless healthcare context.

3.3 Data Extraction

From each included paper extract:

- Authentication/authorization methods (e.g., JWT, OAuth, role-based)
- Database design (MongoDB schema, collections, relationships)
- Performance metrics (latency, concurrency, real-time updates)
- Security features (encryption, vulnerability analysis)
- Feature list (patient/doctor dashboards, cancellation/reschedule, reminders)
- Scalability/architecture (monolithic vs microservices, WebSockets, serverless)

IV. CRITICAL ANALYSIS & DISCUSSION

This is the core section where you deeply analyse *why* and *how* the MERN stack is chosen and used, and critique.

4.1 Architectural Evaluation

Strengths (The MERN Advantage)

- Unified language (JavaScript) across client and server → simpler skill-set, faster development.
- Express + Node.js enable rapid API development; React enables dynamic, interactive front-end UI.
- MongoDB's schema-less model allows flexible data structures.

Weaknesses (The NoSQL Dilemma)

- Doctor-patient-appointment relationships are inherently relational (many-to-many, constraints, cascading updates). MongoDB's document model may complicate these relations: e.g., lack of FK constraints, transaction support (though multi-doc transactions are now supported but less mature).

- Ensuring data consistency (e.g., when rescheduling appointments, cancelling doctors, updating availability) is harder without rigid schema constraints.
- Scalability beyond a prototype—sharding, indexing, large data volumes—are under-explored in these systems.

4.2 Security & Authentication Review

- Many systems adopt JWT-based authentication; however, fewer consider token expiry, refresh tokens, secure storage of tokens (httpOnly cookies vs localStorage).
- Role-based access: typical roles are Admin, Doctor, Patient—but many papers do not deeply analyse privilege escalation, cross-role data leaks, audit logging.
- Compliance: Very few references to healthcare regulation (HIPAA, GDPR) or encryption at rest/in transit; few discuss logging, audit trails, breach-handling.
- Data encryption & protection: Use of bcrypt for password hashing is common (e.g., the ML scheduling paper ISJEM Journal) but encryption of medical records, secure transmission, SOC2-type auditing rarely covered.

4.3 Feature Completeness and Modern Gaps

Core Features:

- Slot availability, booking/reschedule/cancellation, user dashboards (patient/doctor), appointment history.
- Many systems embed notifications/reminders (email/push) or integrate payment gateways.

Missing/Immature Features:

- Telemedicine (video/audio) integration is rare or only mentioned as future work.
- Smart scheduling/AI – few real-world implementations of machine-learning for optimising doctor time, patient routing, predictive no-shows.
- Interoperability with standard healthcare data models (e.g., FHIR) is almost absent.
- Microservices/serverless architectures for extreme scalability, multi-tenant clinics, distributed hospitals are only tangentially discussed.
- Robust real-time synchronization (WebSocket, push notifications) is in prototype stage (e.g., WebSockets paper Jetir) but not widely validated in production.

Paper Title	Method Used	Key Findings	Main Gaps	How Proposed System Fixes It
Doctor Appointment Booking + OCR (2025)	MERN + OCR	Digital booking + prescription scan	Weak security, no FHIR, shallow MongoDB design	Adds RBAC, encryption, FHIR-ready model
Online Doctor Appointment for Bengaluru (2025)	MERN + JWT + real-time	Real-time booking, basic auth	No scalability, no audit logs, no refresh tokens	Adds refresh tokens, audit trails, load-balanced APIs
MediTrack Healthcare System (2025)	MERN + EMR + multi-role	Better usability, role dashboards	No data-integrity checks, no compliance	Uses transactions/hybrid DB + compliance-oriented security
Medicheck MERN System (2025)	MERN prototype	Integrated workflow	No scalability tests, no ACID consistency	Adds safe transaction booking + horizontal scaling
Full-Stack Appointment System (2025)	MERN + payments	Secure payments + booking	Missing WebSockets, no AI features	Adds WebSockets + optional ML reminders

Real-Time MERN Scheduling (2025)	MERN + WebSockets	Prevents double-booking	Prototype only, no large-scale tests	Adds distributed WebSockets + failover
ML-Based Scheduling (2025)	MERN + ML	Improves scheduling efficiency	Not real-time, weak privacy/security	Adds secure ML pipeline + real-time integration
SQL vs NoSQL in Healthcare (2021)	SQL vs MongoDB comparison	MongoDB scalable	Weak ACID, relational challenges	Polyglot DB: MongoDB + SQL for relational data
HL7 FHIR Interoperability (2013)	FHIR standard	Shows value of data exchange	Not used in MERN systems	Integrates FHIR resources for interoperable data

V. FUTURE RESEARCH DIRECTIONS

Based on the analysis, propose concrete directions:

- Data Standardization & Interoperability: Research into integrating MERN systems with healthcare data standards like FHIR to enable patient data exchange across systems.
- Serverless/Microservices Architectures: Compare monolithic MERN deployments to serverless/back-end-for-front-end (e.g., AWS Lambda, Azure Functions) or microservices to handle high load, multi-clinic environments.
- Enhanced Security Protocols: Multi-factor authentication (MFA), zero-trust architecture, data partitioning (per-clinic/patient), audit logging, compliance frameworks (HIPAA, GDPR).
- Hybrid Database Strategies: Use of polyglot persistence: combining MongoDB for flexible data and a relational database for transaction-intensive, strongly relational data (e.g., appointments, billing).
- AI/ML-Driven Scheduling & Telehealth Integration: Smart scheduling algorithms (predict no-shows, optimise slot usage), telemedicine (video calls, chatbots), integrated remote monitoring.
- Scalability & Real-Time Systems: Research into real-time availability updates, WebSocket/graphQL subscriptions, distributed systems, failover and disaster-recovery scenarios in healthcare context.
- Usability & Adoption Studies: Investigate user acceptability, workflow change for clinics, impact on patient waiting times, digital divide issues in emerging markets.

VI. CONCLUSION

Reiterate that the MERN stack is a robust choice for rapid development of doctor-appointment systems thanks to its full-stack JavaScript coherence and flexible architecture. However, the key takeaway is that future work must prioritise data integrity (especially given the relational nature of healthcare scheduling), regulatory-grade security, interoperability with healthcare standards, and scalable architectures if these systems are to move from prototypes to mission-critical production in clinical settings.

REFERENCES

[1] Nandwalkar, J. R.; Bokde, Aryan K.; Adke, Parth A.; Jethe, Riddhesh S. (2025). “Doctor Appointment Booking and Handwriting Recognition System”. *International Journal of Innovative Science and Research Technology*. ISSN 2456-2165. IJISRT+1

[2] Gade, Neha; Wagaskar, Aarti; Gursali, Apurva; Takale, Rupali; Kanade, R. S. (2025). “MediTrack Healthcare System”. *International Journal of Scientific Research in Science, Engineering and Technology*. MERN Stack, EMR, role-based access. IJSRSET

[3] Jambagi, R. S.; Dharani, N. V. (2025). “Development of Online Doctor’s Appointment System for Bengaluru”. *IJSCI*. Description: MERN stack, JWT, real-time scheduling. ijsci.com

[4] Kumar, A.; Yadav, Ayush K.; Kalse, Prem V.; Singh, Aditya K. (2025). “Medicheck: A MERN Stack Based Healthcare Management System With Multi-Role Integration”. *International Journal of Science, Engineering and Technology*. IJSENT

[5] Suman; Gossain, K. (2025). "Design and Implementation of a Full-Stack Healthcare Appointment Scheduling System". *IJSRET*. MERN stack, real-time updates, secure payments. IJSRET

[6] Ravi, G.; Kamal, M. (2025). "Optimizing Healthcare Scheduling Through Machine Learning: A Role-Based Doctor Appointment System". *ISJEM Journal*. MERN stack with JWT, dynamic scheduling. ISJEM Journal

[7] Sharma, Sunny; Patel, Rashid; Khan, Sehar. (2025). "Real-Time Appointment Scheduling Using MERN Stack and WebSockets". *JETIR Feb 2025*, Vol.12, Issue 2. Jetir

[8] "Hospital Management Application using MERN Stack." GeeksforGeeks, last updated 23 Jul 2025. (Tutorial/overview). GeeksforGeeks

[9] Kaur, H., & Rani, S. (2021). *Comparative Analysis of SQL and NoSQL Databases for Healthcare Big Data Applications.* Journal of King Saud University – Computer and Information Sciences, 33(6), 693–703. https://doi.org/10.1016/j.jksuci.2019.01.010

[10] Bender, D., & Sartipi, K. (2013). *HL7 FHIR: An Agile and RESTful Approach to Healthcare Information Exchange.* IEEE 26th International Symposium on Computer-Based Medical Systems (CBMS), 326–331. https://doi.org/10.1109/CBMS.2013.6627810