IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

Clinic Queue Handler Web Portal

Yash Sanjay Kandekar[1], Kunal Dnyaneswar Murkute[2], Sanskar Sanjay Nagare[3], Prathamesh Ravindra Salunke[4], Ms. S. P. Morade[5]
Students, Department of Computer Engineering[1-4]
Guide, Department of Computer Engineering[5]
Matoshri Aasarabai Intitute of Technology and Research Centre, Eklahare, Nashik, Maharashtra, India

Abstract: *The Clinic Queue Handler Web Portal is a web-based system designed to automate and streamline patient queue management in clinics and hospitals. Traditional healthcare facilities face significant challenges including long waiting times, patient overcrowding, and inefficient use of available resources, all of which negatively impact patient care and satisfaction. This paper presents the design, architecture, and implementation of an intelligent Hospital Queue Management System (HQMS) that optimizes the patient admission process through real-time queue tracking, automated token generation, appointment scheduling, and role-based access control for clinic staff. The system integrates a Patient Treatment Time Prediction (PTTP) algorithm and a Hospital Queuing-Recommendation (HQR) module to estimate waiting times and suggest efficient treatment schedules. The proposed system features an advanced bed management module, notification services via SMS and email, a doctor dashboard for real-time consultation management, and seamless integration with existing Electronic Health Record (EHR) and billing systems. Results demonstrate significant reductions in manual record-keeping and improved transparency between patients and healthcare providers*

Keywords: Queue Management System, Hospital Information System, Patient Flow, Appointment Scheduling, WebSocket, Spring Boot, Real-Time Queue, Healthcare Automation, PTTP Algorithm

I. INTRODUCTION

Healthcare is a critical sector where the quality of service is directly linked to patient outcomes and overall satisfaction. One of the most persistent challenges faced by modern healthcare facilities is the lack of an effective technique to manage patient queues. Long wait times lead to patient overcrowding, increased frustration, and in critical cases, delayed medical treatment.

The Clinic Queue Handler Web Portal addresses this challenge by providing an automated, web-based solution for managing patient flow in clinics and hospitals. The system enables patients to register online, book appointments, and track their queue position in real time. Simultaneously, doctors and administrative staff can efficiently manage patient flow through role-specific dashboards, significantly reducing manual effort and improving service quality.

Queuing is a challenge for all healthcare systems globally. Considerable research has demonstrated the applicability of queuing theory and machine learning models to improve waiting time estimation and resource allocation in hospital settings. This paper proposes a comprehensive system that combines these approaches with modern web technologies to deliver a practical, scalable, and secure solution for clinics of all sizes.

The system is built using Java Spring Boot for the backend, React.js for the frontend, MySQL for data storage, and WebSockets for real-time communication. It supports role-based access control for Admin, Doctor, Receptionist, and Patient roles, automated notifications, and integration with existing clinic systems.

II. LITERATURE SURVEY

Extensive research has been conducted on hospital queue management and patient wait time prediction. Ozcan (2006) established foundational quantitative methods for healthcare management, while McQuarrie (1983) and Green et al. (2006) demonstrated the practical application of queuing theory to improve staffing in emergency departments.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 8, March 2026 Impact Factor: 8.2

Murray (2000) emphasized understanding patient flow patterns as essential to reducing congestion. More recently, machine learning approaches have been widely explored. Researchers have applied the Random Forest (RF) algorithm to build the Patient Treatment Time Prediction (PTTP) model, which predicts patient treatment durations and minimizes time delays for each treatment task. The RF algorithm, an ensemble learning technique that builds multiple decision trees and aggregates results, was found to outperform traditional methods in handling imbalanced medical datasets.

Apache Spark cloud environments have been proposed alongside PTTP to predict queue waiting times and recommend optimized treatment plans. Logistic regression models have been applied to predict no-show risks for scheduled appointments, improving resource planning in radiology and general practice settings. Studies on multi-stage queuing models for outpatient clinics highlighted the importance of data-driven strategies in optimizing resource allocation.

Arena simulation software has been used to model appointment scheduling and patient flow dynamics, providing insights that can be integrated into machine learning-based predictive systems. Obulor and Eke (2016) analyzed hospital queue management systems in Nigeria, underscoring the necessity of digital transformation in healthcare services. Wong's portable electronics queue control system demonstrated practical hardware-software integration for clinical queue management.

III. SYSTEM ARCHITECTURE

A. Architectural Overview

The Clinic Queue Handler Web Portal follows a Three-Tier Architecture separating the Presentation Layer (client interface), Application Layer (business logic), and Data Layer (persistent storage). This separation improves maintainability, security, and scalability of the overall system.

B. System Components

1. Patient Interface: Web and mobile portal for patient registration, symptom entry via text or voice input, and real-time queue tracking.

2. Voice Input Module: Speech-to-text conversion enabling accessible symptom entry for patients less comfortable with typing.

3. Symptom Processing Module: Keyword extraction and matching to classify patient conditions and route them to the appropriate department.

4. Queue Management Module: Automatic token assignment, real-time queue ordering, and priority handling for emergency or senior citizen patients.

5. Doctor Module: Dashboard for viewing assigned patient queue, managing consultations, skipping or holding patients, and updating status.

6. Receptionist Dashboard: Centralized view of all queues, doctor availability, patient status, and emergency case handling.

7. Admin Module: System configuration, user management, queue rule setup, report generation, and analytics.

8. Notification Service: Automated SMS and email alerts for token generation, queue movement, and consultation readiness.

9. Database Layer: MySQL storage for patient records, doctor schedules, queue tokens, appointments, and audit logs.

C. Technology Stack

Frontend: HTML5, CSS3, JavaScript, Bootstrap, React.js for dynamic component-based UI. Backend: Java with Spring Boot framework, Hibernate (JPA) for Object-Relational Mapping, RESTful API architecture. Database: MySQL with JDBC connectivity. Real-time communication: WebSockets for live queue updates without page refresh. Security: Spring Security, JWT (JSON Web Token), HTTPS/SSL encryption. Deployment: Apache Tomcat, Docker

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

containerization, AWS or Local Server hosting. Development tools: Eclipse/IntelliJ IDEA, Postman, Git/GitHub, Maven.

D. Hardware Requirements

Server: Intel Core i3 or equivalent, 4 GB RAM, 20 GB free disk space, 64-bit architecture, broadband internet connection. Client System: Intel Core i3 or higher, 2 GB RAM, 1024×768 display resolution, internet connection.

IV. REQUIREMENTS ANALYSIS

A. Stakeholder Identification

Primary stakeholders include Patients (queue status tracking, reduced waiting time), Doctors (queue management, patient flow), and Receptionists/Front Desk Staff (patient registration, appointment management). Secondary stakeholders include Clinic Administrators (operational monitoring, decision-making reports) and System Administrators (performance, security, backups). Tertiary stakeholders include Cloud/Hosting Service Providers and Government Healthcare Regulatory Authorities.

B. Functional Requirements

The system encompasses twelve core functional modules:

- User Registration and Authentication with role-based access control for patients, doctors, receptionists, and admins.
- Patient Registration Management with unique patient ID generation supporting both walk-in and appointment-based flows.
- Queue Generation and Management with automatic token assignment, real-time updates, and multi-doctor queue support.
- Appointment Scheduling with double-booking prevention, rescheduling, and cancellation features.
- Real-Time Queue Display showing current token, estimated waiting time, and automatic refresh via WebSockets.
- Notification System sending SMS and email alerts for token generation, queue movement, and appointment confirmations.
- Doctor Dashboard for viewing queue, marking consultation start/end, and skipping or holding patients.
- Receptionist Dashboard for managing live queues, doctor availability, and emergency cases.
- Admin Management Module for clinic settings configuration, queue rules, and data export.
- Report Generation for patient visits, doctor performance, and queue efficiency with date and doctor filtering.
- Secure Data Storage and Retrieval maintaining real-time and historical data consistency.
- Session Management with automatic timeout and secure logout.

C. Non-Functional Requirements

- Performance: System response within 2–3 seconds; real-time queue updates without noticeable delay under peak loads.
- Security: Encrypted communications (HTTPS/SSL), JWT authentication, BCrypt password hashing, input validation and sanitization.
- Usability: Intuitive interface designed for non-technical users including receptionists and doctors.
- Scalability: Support for multiple doctors, departments, and clinic branches without major redesign.
- Availability: 99% uptime during clinic operational hours with access from multiple device types.
- Portability: Browser-based, device-agnostic, compatible with Google Chrome, Firefox, and Microsoft Edge.
- Data Backup: Regular automated backups with tested restoration procedures ensuring data integrity after recovery.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

V. SYSTEM DESIGN

A. Data Flow Design (DFD)

At DFD Level 0, the central process 'Clinic Queue Handler Web Portal' receives Patient Details/Symptoms (text or voice) and Appointment Requests from the Patient entity, and Patient Registration Data from the Receptionist entity. It returns Queue Number/Queue Status/Consultation Details to the Patient and Patient Queue List/Patient Details and Consultation Status/Advice/Prescription to the Doctor.

At DFD Level 1, the system is decomposed into five processes: P1 (Patient Registration & Login), P2 (Symptom Submission & Processing), P3 (Queue Generation & Management), P5 (Doctor Dashboard & Consultation), and supporting data stores D1 (Patient Records & Health Info), D2/D3 (Queue Database), and D4 (Doctor Records/Reports). At DFD Level 2, the symptom processing is further decomposed into P2.1 (Symptom Entry & Input Processing with Speech-to-Text Module) and P2.2 (Keyword Extraction & Symptom Analysis), and queue management is decomposed into P3.1 (Queue Number Assignment) and P3.2 (Queue Monitoring & Position Update).

B. Use Case Design

The Use Case Diagram identifies four main actors: System Admin, Queue Admin, Queue Operator, and Queue Client (Patient). System Admin can Register Hospital, Register Queue Admin, and Create Queue. Queue Admin can Register Queue Operator and Create Queue. Queue Operator can Manage Queue. Queue Client can View Queue Status.

C. Entity-Relationship Design

The E-R model identifies User as the central entity with three specializations: Secretary, Admin, and Doctor (connected via 'Is A' relationships). Secretary manages Patient Reservations (M:M). Admin creates and manages multiple records. Doctor manages Patient Visits (M:M). Patient Profile has a 1:M relationship with Patient Visits and a M:1 relationship with Patient Reservations.

D. System Workflow

1. Patient logs in or registers on the web portal.
2. Patient enters details and symptoms using text or voice input.
3. System processes the input, extracts keywords, and retrieves existing patient health records.
4. Queue token is generated automatically based on doctor availability and priority rules.
5. Patient is added to the live queue; queue status is displayed and notifications are sent.
6. Doctor views the real-time queue and calls the next patient.
7. Consultation is completed, patient status is updated, and the queue advances automatically.

VI. IMPLEMENTATION

A. Deployment Architecture

The system is deployed using a three-tier architecture on Apache Tomcat. The Presentation Layer consists of web browsers used by clinic staff providing user interfaces for token generation, queue monitoring, and appointment updates. The Application Layer hosts core queue handling logic, manages authentication, token generation, scheduling, and notifications, and interfaces between the client and database via RESTful APIs and WebSocket communication. The Data Layer manages persistent MySQL storage ensuring data integrity, consistency, and availability through indexing, referential integrity constraints, and scheduled backups.

B. Real-Time Queue Management

Real-time queue management is implemented using WebSockets, enabling persistent bidirectional communication between the server and all connected clients. Queue state changes — including token assignments, consultation completions, and doctor status updates — are broadcast instantly to all subscribed clients without requiring page

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

refresh. This ensures that every receptionist, doctor, and patient viewing the portal sees a synchronized, up-to-date queue state at all times.

C. Integration with Existing Systems

The portal integrates with existing clinic systems through secure REST APIs and middleware adapters. Key integrations include: Patient Registration System (automatic patient detail retrieval and token generation without duplicate data entry); Appointment Management System (synchronization of scheduled appointments with real-time queue); Doctor Scheduling System (real-time doctor availability and automatic queue redirection); Billing System (transfer of consultation details for invoice generation); Laboratory/Diagnostic Systems (test request coordination and status updates in patient queue); Electronic Health Record (EHR) Systems (access to patient medical history during consultation); and Digital Display/Token Calling Systems (real-time display of queue numbers and audio announcements in waiting areas).

D. Notification System

The notification system uses asynchronous processing via SMTP-based Email API and SMS Gateway API. Notifications are queued independently of core queue handling functions to prevent blocking during high-traffic periods. Patients receive automated alerts for token generation, expected waiting time, and consultation readiness. Staff receive alerts for queue delays, doctor unavailability, and emergency cases.

VII. SECURITY AND SCALABILITY

A. Security Measures

- User Authentication: Unique username and password authentication with secure Spring Security integration.
- Role-Based Access Control (RBAC): Strict permission enforcement for Admin, Doctor, Receptionist, and Patient roles.
- Password Security: BCrypt hashing ensures passwords are never stored in plain text.
- Secure Communication: All client-server data transmission encrypted using HTTPS/SSL.
- JWT Session Management: JSON Web Tokens validate every API request, preventing session hijacking.
- Input Validation: Server-side and client-side validation protects against SQL injection and Cross-Site Scripting (XSS).
- Database Security: Restricted access credentials; only the application server interacts with the database directly.
- Audit Logs: All login attempts, token generations, and queue updates are recorded for monitoring and compliance.
- Session Timeout: Automatic session expiry after a defined period of inactivity.

B. Scalability Considerations

- Horizontal Scaling: Multiple application server instances supported behind a load balancer for high-traffic scenarios.
- Vertical Scaling: Server CPU, RAM, and storage can be upgraded to accommodate growing patient loads.
- Stateless Design: Backend is stateless allowing independent request handling across multiple server instances.
- Database Optimization: Indexing on frequently queried fields, normalized schema, and query optimization ensure stable performance under peak loads.
- Asynchronous Processing: SMS/email notifications and report generation are processed asynchronously preventing bottlenecks.
- Caching: Frequently accessed data such as doctor availability and current queue status is cached to reduce database load.
- Cloud Deployment: AWS/Azure compatibility with dynamic auto-scaling based on patient traffic.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

VIII. CHALLENGES AND LESSONS LEARNED

A. Challenges Encountered

1. Legacy System Integration: Differences in database formats and communication protocols between the queue handler and existing registration, billing, and appointment systems required custom middleware and API design.

2. Real-Time Synchronization: WebSocket implementation required careful concurrency management to ensure consistent queue state across simultaneous updates from multiple clients.

3. Notification Reliability: Varying network reliability and third-party API limitations made real-time SMS and email delivery challenging, requiring retry mechanisms and asynchronous queuing.

4. Database Performance: Large volumes of patient and queue data under peak clinic loads required extensive indexing, query optimization, and schema normalization.

5. Security Implementation: Configuring multiple security layers (HTTPS, JWT, password hashing, input validation) without compromising usability required careful testing and iteration.

6. UI Accessibility: Designing an interface intuitive enough for non-technical receptionists and doctors while displaying complex real-time queue information required multiple user testing cycles.

B. Key Learnings

- Thorough requirement analysis prevents costly design changes during development.
- Modular architecture enables independent scaling and maintenance of system components.
- User-centered design is essential in healthcare applications where errors can have serious consequences.
- Comprehensive testing including peak load and concurrent user scenarios is critical for reliability.
- Role-based access control must be designed and tested systematically from the earliest stages.

IX. FUTURE ENHANCEMENTS

- Mobile Application: Android and iOS apps for patients and staff enabling remote queue management, appointment booking, and push notifications.
- AI-Based Queue Prediction: Machine learning algorithms for wait time prediction and dynamic queue optimization based on historical patient patterns and doctor efficiency metrics.
- Telemedicine Integration: Support for virtual consultations with unified queue management for both in-person and online patients.
- Multi-Clinic Management: Centralized platform for managing multiple clinic branches with consolidated analytics and reporting.
- IoT Integration: Smart token displays, wearable device connectivity, and automated patient notifications to reduce waiting area congestion.
- Automated Billing & Insurance Processing: Fully integrated billing and insurance claim submission reducing administrative workload and errors.
- Voice and Chatbot Assistance: AI-powered assistants guiding patients through registration, queue updates, and basic inquiries.
- Enhanced Security: Biometric authentication (fingerprint/facial recognition), multi-factor authentication (MFA), and compliance with national healthcare data protection standards.
- Cloud SaaS Deployment: Platform-as-a-service model enabling clinics of any size to adopt the system without local server investment.

X. CONCLUSION

The Clinic Queue Handler Web Portal presents an effective and scalable digital solution for patient queue management in healthcare facilities. By automating token generation, enabling real-time queue tracking through WebSocket communication, providing role-specific dashboards for doctors and receptionists, and integrating notification and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 8, March 2026
Impact Factor: 8.2

appointment systems, the portal substantially reduces patient waiting times, minimizes manual administrative effort, and enhances the overall quality of clinic operations.

The system's modular, secure, and cloud-compatible architecture ensures it can scale from a single-doctor clinic to a multi-branch hospital network. The integration of PTTP and HQR algorithms provides data-driven wait time estimation, contributing meaningfully to the ongoing digital transformation of healthcare delivery. Future enhancements including mobile applications, AI-based optimization, and IoT integration will further extend the system's impact on patient care and operational efficiency.

REFERENCES

[1] Y. A. Ozcan, Quantitative Methods in Health Care Management: Techniques and Applications, 1st ed. Jossey-Bass Publications, 2006.

[2] D. G. McQuarrie, "Hospital utilization levels: The application of queuing theory to a controversial medical economic problem," Minnesota Medicine, vol. 66, pp. 679–686, 1983.

[3] L. V. Green, J. Soares, J. Giulio, and R. Green, "Using queuing theory to increase the effectiveness of emergency department provider staffing," Academic Emergency Medicine, vol. 13, no. 1, pp. 61–68, 2006.

[4] S. C. Murray, "Understanding the patient flow," Decision Line, pp. 8–10, Mar. 2000.

[5] M. Adele and S. Barry, "Modeling patient flow in hospital using queuing theory," Unpublished Manuscript, 2005.

[6] S. Ivalis and P. Millard, "Health care modeling operating the black box," British Journal of Health Care Management, vol. 8, no. 7, pp. 251–255, 2003.

[7] R. Arun and P. P. Priyesh, "Smart queue management system using GSM technology," Advances in Electronic and Electric Engineering, vol. 3, no. 8, pp. 941–950, 2013.

[8] R. Obulor and B. O. Eke, "Outpatient queuing model development for hospital appointment system," International Journal of Scientific Engineering and Applied Science (IJSEAS), vol. 2, no. 4, Apr. 2016.

[9] C. Y. Wong, "Portable electronics queue control system," University Malaysia Pahang.

[10] A. H. Anderson, Successful Training Practice: A Manager's Guide to Personnel Development. Oxford, UK: Blackwell Business Publishers, 1993.

[11] Smith and G. van der Pijl, "A private hospital management system," Postgraduate Institute of Medicine, University of Colombo, Sri Lanka, 2015.