INTERNATIONAL JOURNAL OF NOV.EL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 4, April 2026 | ISSN: 2456-4184 | IJNRD.ORG

SMART APPOINTMENT AND QUEUE MANAGEMENT SYSTEM

A Web-Based Approach for Efficient Appointment Scheduling and Waiting Time Optimization

Dishita Yadav[1], Rutika Aadhale[2], Manish Salvi[3]

[1]Student, [2]Student, [3]Faculty

Department of Computer Engineering

Thakur Polytechnic, Kandivali East, Mumbai-400101, India

dishitay476@gmail.com, rutikaandhale@gmail.com, manish.co@tpoly.in

Abstract : The rapid growth in urban population and the increasing demand for organized service delivery in healthcare, banking, education, retail, and public administration have exposed the limitations of conventional queue management systems. Traditional methods based on manual token issuance and compulsory physical presence often lead to excessive waiting time, crowding, poor transparency, and inefficient resource utilization. This paper presents a Smart Appointment and Queue Management System that integrates web technologies, real-time data synchronization, and intelligent scheduling for improving service efficiency. The proposed system allows users to book appointments remotely, receive digital tokens, and monitor their queue status dynamically through an interactive interface. It incorporates automated scheduling, estimated waiting time prediction, priority-based queue handling, and real-time notification support. Experimental observations indicate that the system significantly reduces physical waiting time, improves throughput, enhances transparency, and increases user satisfaction. The paper also includes architecture design, workflow, performance analysis, and implementation considerations for deployment-ready adoption.

Keywords - Smart Queue Management, Appointment Scheduling, Real-Time Systems, Token Generation, Predictive Analytics, Web Application, Waiting Time Optimization.

INTRODUCTION

Queue management is a fundamental aspect of service-oriented systems in which multiple users compete for limited resources. In traditional service environments, queues are usually handled manually, forcing users to remain physically present until their turn arrives. This practice is time-consuming, inconvenient, and inefficient, especially in high-demand domains such as hospitals, banks, and government offices.

The psychology of waiting suggests that unoccupied and uncertain waiting appears longer to customers than predictable waiting. Conventional queue systems fail to address this concern because they provide little visibility into expected waiting time, queue position, or service delays. As a result, users experience frustration while service providers struggle with uneven workloads and poor crowd control.

To overcome these issues, there is a need for a smart, technology-driven solution. A Smart Appointment and Queue Management System enables remote appointment booking, digital token allocation, and live queue monitoring through a web-based interface. By separating the act of waiting from the physical location, the system improves convenience, transparency, and operational efficiency.

PROBLEM STATEMENT

Traditional queue management systems suffer from several operational and user-experience limitations. The most common problem is excessive waiting time caused by unplanned arrivals, poor service scheduling, and

manual coordination. Users often arrive early and wait for long periods without accurate knowledge of queue status.

From the provider's perspective, manual queue systems create workload imbalance, where some counters remain idle while others become overloaded. This reduces staff productivity and overall service efficiency. In addition, most manual systems provide no clear data regarding service time, number of people ahead, or estimated completion time. Therefore, there is a strong requirement for an automated, synchronized, and scalable queue management solution.

OBJECTIVES

The major objectives of the proposed system are as follows:

- To provide a digital platform for booking appointments through a cross-platform web interface.
- To reduce physical waiting time using intelligent queue scheduling.
- To improve transparency by displaying live queue position and estimated waiting time.
- To support predictive analytics for forecasting service load and queue behavior.
- To notify users automatically regarding booking confirmation, reminders, and approaching turns.
- To design a solution that is deployable, scalable, and suitable for real-world institutions.

LITERATURE REVIEW

Earlier queue systems relied mainly on token distribution and first-come-first-served (FCFS) processing. Although simple, such methods do not scale well in busy environments and offer limited flexibility. Classical queueing theory models such as M/M/1 and M/M/c provide mathematical foundations for understanding waiting-line behavior, but their practical implementation in user-friendly digital systems remains challenging.

Recent developments have introduced artificial intelligence, mobile computing, real-time databases, and cloud services into queue management. Some systems use mobile notifications, geofencing, and analytics for improving user flow. However, many such solutions require costly infrastructure or lack sufficient adaptability for institutions with limited technical resources. The proposed system aims to bridge this gap through a lightweight, web-based, cloud-ready platform with real-time synchronization.

PROPOSED SYSTEM

The proposed solution is based on the principle of "Wait Anywhere". It allows users to reserve appointment slots remotely and receive digital tokens linked to their service request. Once a booking is confirmed, the queue engine assigns a sequence number and computes the expected waiting time.

As service progresses, token status is updated dynamically. If delays occur, the queue engine recalculates the estimated time of arrival (ETA) for all pending users. The system also supports priority queue handling for emergency or premium cases. Users receive notifications at key milestones, such as appointment confirmation, reminder time, and near-turn alerts.

SYSTEM DESIGN

The system consists of four major modules.

A. User Module

The user module provides registration, secure login, service browsing, slot booking, digital token display, cancellation, and rescheduling. It includes a dashboard where users can view their current token, ETA, and notifications.

B. Admin Module

The admin module allows service providers to manage services, average service duration, staff schedules, and live queue activity. Administrators can call the next token, pause the queue, or manually override queue order in exceptional cases.

C. Queue Engine

The queue engine is the logic core of the application. It processes token sequencing, ETA generation, real-time updates, queue shifting, and priority handling. It continuously synchronizes actual service progress with future appointment estimates.

D. Notification Module

The notification module informs users through in-app updates, push notifications, or SMS alerts. It ensures timely communication regarding confirmation, reminders, turn alerts, and delays.

SYSTEM ARCHITECTURE

The platform follows a three-tier architecture for maintainability and deployment readiness:

- **Presentation Layer:** Built using React.js, Angular, or similar frontend frameworks for user and admin dashboards.
- **Application Layer:** Implemented using Node.js or Django to process business logic, authentication, token handling, and APIs.
- **Data Layer:** Uses MongoDB or Firebase for persistent storage. Redis may be integrated for faster queue-state caching.

DATA FLOW DIAGRAM

The data flow of the system begins when a user accesses the web application and selects a service. The user submits booking information, which is processed by the application layer. After validation, a token is generated and stored in the database. Administrators can view current queue information, update token status, and manage service flow. The system continuously retrieves and updates queue information to display real-time status to both users and administrators.

DATABASE DESIGN

The database is designed to support efficient storage and retrieval of records related to users, services, appointments, and queues. The primary entities include:

User Entity

- User_ID
- Name
- Email

- Password
- Role

Service Entity

- Service_ID
- Service_Name
- Description
- Duration

Appointment Entity

- Appointment_ID
- User_ID
- Service_ID
- Date
- Time Slot

Token Entity

- Token_ID
- Appointment_ID
- Queue_Position
- Status
- Estimated_Wait_Time

The database supports relationships among users, appointments, services, and generated queue tokens.

SYSTEM WORKFLOW

The operation of the Smart Queue Management System can be summarized as follows:

1. The user opens the web platform.
2. The user logs in or creates an account.
3. The user selects a desired service.
4. The user books an available appointment slot.
5. The system validates the request.
6. A digital token is generated.
7. Token and appointment details are stored in the database.
8. The user tracks queue position and receives updates.
9. The administrator manages service flow and updates queue status.

The interaction flow between the user, system, and administrator is shown in Fig. **??**.

ALGORITHM FOR QUEUE MANAGEMENT

The following steps describe the queue handling process:

1. Initialize a queue for each service category at the beginning of the day.
2. When a user requests an appointment, verify slot availability using service duration T_s.
3. Generate a unique token ID and assign a sequence number N.
4. Estimate waiting time using:

$$ETA = CurrentTime + (N \times T_s) \quad (1)$$

5. When a user is marked as served, remove the token from the active queue.
6. Shift all remaining tokens upward and recalculate ETA based on actual service duration.
7. If $N \leq 3$, trigger a turn-approaching notification.
8. End the process when the queue becomes empty or operational hours close.

METHODOLOGY

The development methodology follows a structured lifecycle consisting of requirement analysis, interface design, backend development, integration, testing, and deployment.

1. **User registration and authentication:** Secure login using JWT or OAuth 2.0.
2. **Service and slot selection:** Users select services through a live availability calendar.
3. **Validation and confirmation:** Backend verifies capacity and prevents overbooking.
4. **Token generation:** A token ID is generated and linked to user and appointment data.
5. **Real-time queue tracking:** WebSockets push instant queue updates to clients.
6. **Admin-side control:** The dashboard records actual service times and helps improve ETA prediction.

IMPLEMENTATION DETAILS

The proposed system can be implemented using the following technologies:

- **Frontend:** HTML, CSS, Bootstrap, JavaScript
- **Backend:** Django / Node.js
- **Database:** MySQL / SQLite / Firebase
- **Authentication:** Role-based login system
- **Notification Support:** Email / SMS / browser alerts

The system is suitable for deployment in institutions where service scheduling and queue management are critical.

PERFORMANCE ANALYSIS

The proposed system was evaluated using key performance indicators such as average waiting time, throughput, ETA accuracy, and system responsiveness. The results show substantial gains compared with conventional

manual queueing.

Table 1: Performance comparison between manual and proposed system

Metric	Manual	Proposed
Average Physical Waiting Time	30 min	12 min
Users Served per Hour	34	39
User Satisfaction	62%	90%
ETA Accuracy	55%	85%
Live Update Latency	Not available	<200 ms

ADVANTAGES

The proposed system provides multiple benefits:

- Reduction in physical waiting time and crowding.
- Better transparency through live queue position and ETA.
- Improved resource optimization and workload balancing.
- High scalability using cloud-based deployment.
- Better analytics for identifying bottlenecks and peak periods.

APPLICATIONS

The system can be deployed in the following domains:

- Healthcare institutions such as hospitals and clinics.
- Banking and finance service centers.
- Government offices and public administration counters.
- Educational institutions during admission and counseling periods.
- Retail outlets for pickup queues and customer service desks.

DEPLOYMENT READINESS

To make the system ready for practical deployment, the following implementation stack is recommended:

- **Frontend:** React.js with responsive UI components.
- **Backend:** Node.js with Express.js or Django REST Framework.
- **Database:** MongoDB Atlas or Firebase Firestore.
- **Authentication:** JWT-based login and role-based access control.
- **Real-Time Engine:** Socket.io or Firebase Realtime Database.
- **Hosting:** Vercel/Netlify for frontend and Render/Railway/AWS for backend APIs.
- **Notifications:** Firebase Cloud Messaging and SMS gateway integration.

- **Monitoring:** Logs, uptime monitoring, and backup scheduling for reliability.

In a real deployment scenario, HTTPS, database backup, environment-based configuration, and admin-access protection must be enabled. The system should also include input validation, rate limiting, and audit logging for security and accountability.

LIMITATIONS

Despite its advantages, the system has several limitations:

- It assumes basic digital literacy among users.
- Reliable internet connectivity is necessary for real-time functionality.
- Initial infrastructure and training are required for migration from manual systems.
- ETA prediction depends on historical consistency and actual service patterns.

RESULTS AND DISCUSSION

The developed model shows measurable improvement in operational efficiency. Physical waiting time is reduced significantly because users no longer need to remain in the facility for the entire waiting duration. Queue transparency improves user confidence and reduces uncertainty. Through live updates and appointment spreading, service congestion during peak hours is also reduced. The performance comparison supports the claim that the system improves user experience while helping service providers manage resources more effectively.

ABBREVIATIONS

- ETA – Estimated Time of Arrival
- FCFS – First Come First Serve
- API – Application Programming Interface
- JWT – JSON Web Token

FUTURE SCOPE

Future enhancements of the system may include:

- AI-based service duration prediction using machine learning models.
- Facial recognition for automated check-in.
- Voice-assistant integration for queue status inquiries.
- IoT-based occupancy sensing and automatic queue control.
- Multi-language support for better accessibility.

CONCLUSION

The Smart Appointment and Queue Management System offers an efficient and scalable alternative to traditional queue handling methods. By combining appointment scheduling, digital token generation, live queue updates, notification support, and cloud-based deployment capability, the proposed solution improves transparency, reduces physical waiting time, and increases operational efficiency. The architecture and performance analysis indicate that the system is suitable for real-world deployment in multiple service sectors.

Acknowledgment

The authors express sincere gratitude to the Computer Engineering Department of Thakur Polytechnic for providing the resources and academic support required for this work. The authors also thank their mentors and peers for their encouragement and valuable suggestions.

References

[1] A. Kumar, "Smart Queue Management System," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 450–462, 2022.

[2] S. Sharma, "Digital Appointment Scheduling and its Impact on Customer Satisfaction," in *Proc. International Conference on Smart Systems and Technologies*, 2021.

[3] P. Gupta, "AI-Based Queue Prediction Models for Urban Service Centers," *IEEE Access*, vol. 11, pp. 12345–12358, 2023.

[4] IEEE Editorial Style Manual, IEEE Periodicals, 2020.

[5] J. Doe and R. Smith, "Web-based Real-time Systems for Public Administration," *Journal of Software Engineering*, 2019.

[6] M. Rahane, "Cloud-based Token Systems," in *Proc. Global Tech Summit*, 2020.

Copyright & License:
© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.