International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Smart Queue Management System

Harsh Tiwari, Shubham Mangal, Manish Chotia
2022pietcdtiwari057@poornima.org, 2022pietcdshubham053@poornima.org,
2022pietcdmanish040@poornima.org
Poornima Institute of Engineering and Technology, Jaipur, India

***Abstract*—An increasing need for fast services has brought queue management system in the limelight in hospitals, banks and offices etc. Traditional queueing systems result in long waits and crowding. In this paper we present Smart Queue Management System that allow the users to join the queues remotely using mobile application. The system records real time queue tracking by Socket.io and the queue is reserved by that when the user chooses one. The platform is developed with Flutter on the front end and Node.js/Express.js with MongoDB on the back end.**

***Index Terms*—Queue Management, Flutter, Node.js, MongoDB, Real-Time Systems, Smart Systems**

I. Introduction

Queue management is a pivotal issue in the service delivery environment. The classical system encourages the necessity of physical attendance at the counters, leading to inefficiency, congestion and discontent among users.

With the advancement in mobile app and cloud computing, digitalized approach can be used to replace over the traditional queuing mechanism. The users can also join queues at a distance, and monitor their progress using the real time system.

The specific goals are:

- To reduce the waiting time,
- To increase the service efficiency,
- To enable customers to get real-time queue information.

II. Problem Statement

In a system of queues, since customers must be physically located within the service facility to receive service, congestion is endemic and customersfrequently waste time. Excessively long queues may degrade both user experienceand service efficiency.

It would be nice to have a system that lets people remotely join queues, track their status in a queue in real time, and only show up when needed.

III. Objectives

System objectives are:

- Eliminate the queues of people waiting in person.
- Enabling real-time monitoring ofqueues
- In order to raise service efficiency.
- In order to provide more convenience forusers through mobile phone.

IV. Literature Review

Queue systems in the past were manual and inefficient. The current systems are web and mobile based. But several does not real-time synchronization, and scalability. This platform is designed for simplicity and immediacy, and offers real-time interaction.

V. Proposed System

Through the mobile application, you can take advantage of Smart Queue Management System and it allows the users to queue from a distance and also monitor of token status.

A. System Workflow

The process consists of user login, Services selection,token generation, managing staffs and live monitoring, tracking.

VI. System Architecture

The system is built on the basis of client-server structure. The backend server is communicated by the mobile application via REST APIs. All the data is stored in MongoDB and the real-time updates are powered by Socket.io.

VII. Detailed Implementation

A. User Module

System users can do their jobs from a single point Hub whichis the mobile app. The registered users can login, search service centers and take tokens. The app has simple and easy to use interface to get best from it.

B. Token Management

Token generation is done via backend APIs. Each token in the system has a unique number and queue position. The system stores the sequence correctly and does not repeats.

C. Real-Time Queue System

Emailjs is built on top of Socket. IO for real-time communication. All the connected clients will receive a notification on every token status change in the staff. In manual, this remove refresher. This will help a lot for busy receiver.

D. Admin/Staff Module

A queue web portal is also available for the use of employees. Token status can be modified, Queue flow can be viewed and Counters can be Opened/Closed.

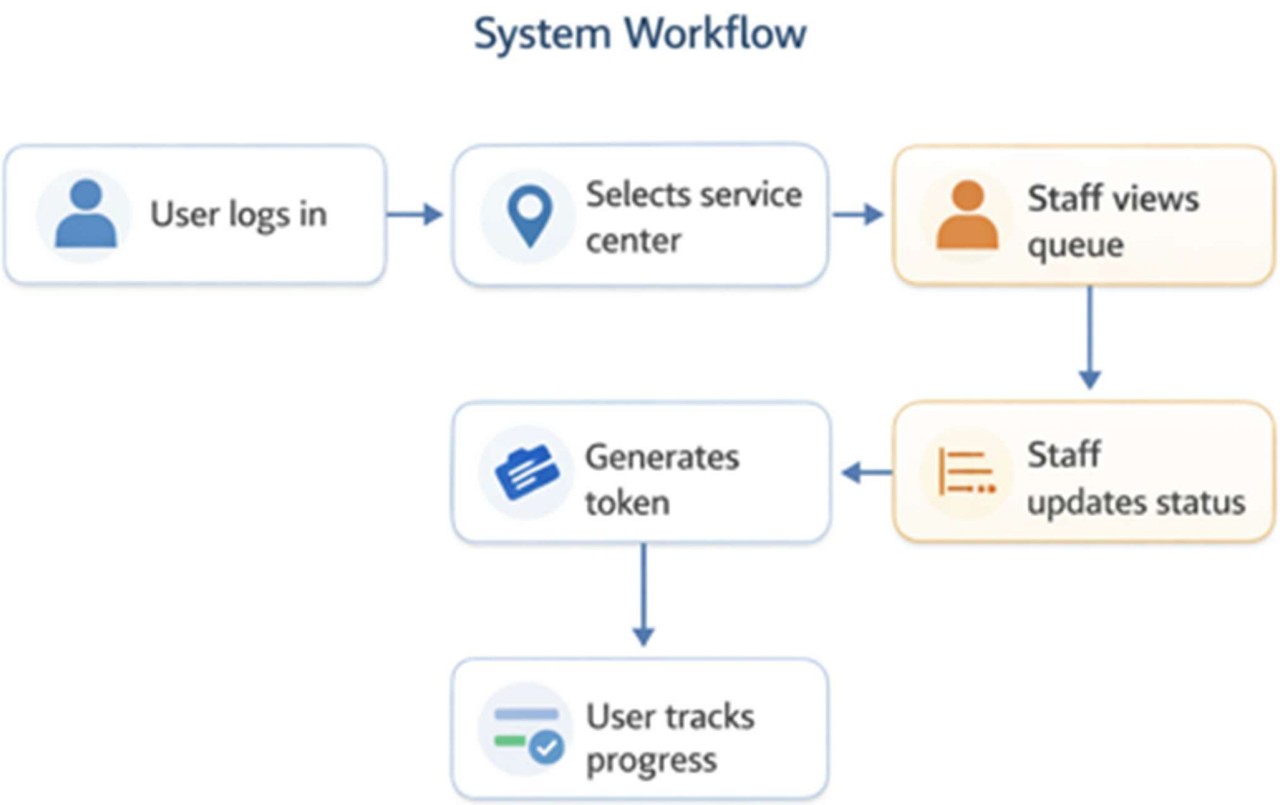

Fig. 1. System Workflow

E. Service Center Interface

This page shows the service center details like address, opening hours and average wait time. So, before you get in a line, it gives users some idea of what to expect.

F. Token Preview and Confirmation

Prior to purchasing the ticket, users are presented with a preview of the wait time and the service information of the selected service.This increases the transparency and also ensures the user to be sure with his/her action.

VIII. Features

IX. User Interface and User Experience

Smart Queue Management System is designed to have very simple and user friendly UI with fluid user experience. The app sport a neat and user-friendly layout, enabling you to easily navigate among the different areas.

The home screen also shows the active tokens, services and latest queue history. This also makes it better for users to search for what they want in one go.

Time, place, waiting time and opening hours are all displayed on the service centre's screen. This will help users to locate the nearest or best service center.

Assert Token Preview Token preview Before generating the users know the length of the queue, and how long they will wait for service. This is a better way to minimize hesitation and enable better choices.

Live updates provide a more polished experience, as users are aware of their position inv the queue. When your turn nears, you get a notification with Descartes "Near" and a second one with a countdown clock continuously running Descartes "Active", so you won't miss your turn!. The simple, clear DCOS UI and system response time were the motivation behind the design and development of the system leading to a delighting and efficient end user experience.

TABLE I
System Features

Feature	Description
Token Generation	Users generate tokens remotely
Real-Time Tracking	Live updates using Socket.io
Service Selection	Choose service centers
Staff Dashboard	Manage queue efficiently
Token History	View past tokens

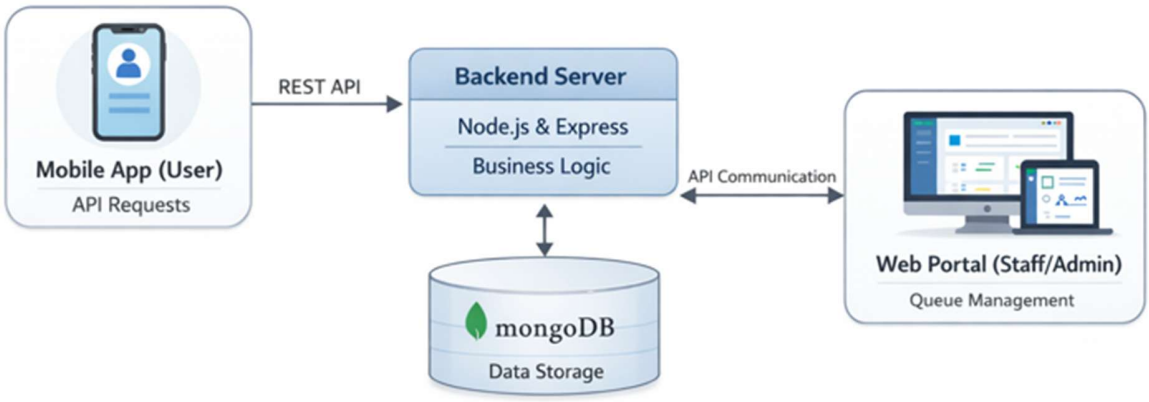

Fig. 2. System Architecture

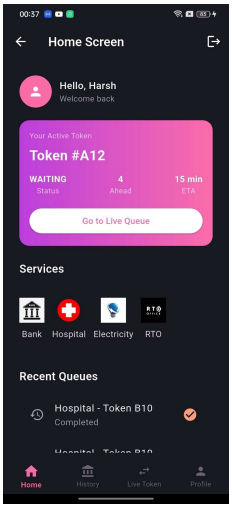

Fig. 3. Home Screen of Mobile Application

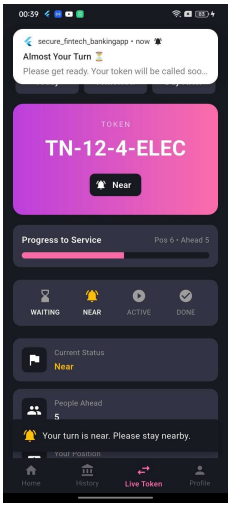

Fig. 4. Live Queue Tracking Screen

TABLE II
TECHNOLOGY STACK

Component	Technology
Mobile App	Flutter
Backend	Node.js, Express.js
Database	MongoDB
Real-Time	Socket.io

X. BENEFITS OF THE PROPOSED SYSTEM

- Reduces crowding at the service counters.
- Saves users time
- Enhances the service efficiency
- Gives updates on the go
- User friendly and scalable

XI. SYSTEM SCALABILITY

Scalability should be considered in system design to serve the increasing number of users and access points. The proposed Smart Queue Management System is designed in a modular fashion so that it can be scaled up as per the need.

Node.js is used for the backend, and it provides the ability to process multiple requests in parallel without any delay. As is the case with any NoSQL database, you can store unstructured content in MongoDB and scale to massive amounts of data. The architecture can be generalized to multi-server scenario with spatial distribution. Every service center can have their own interface, but the data is stored centrally. Regarding future work, deployment of the system on cloud platforms to improve scalability could be considered as well. In this way resources could be dynamically assigned to the user load and kept were satisfactory performance was maintained even under heavy load.

XII. SECURITY AND DATA HANDLING

Security is a concern in all digital platforms, and this is especially true for those that deal with user data and service information. The Smart Queue Management System

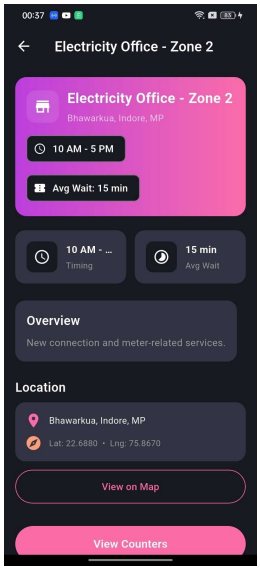

Fig. 5. Service Center Details Screen

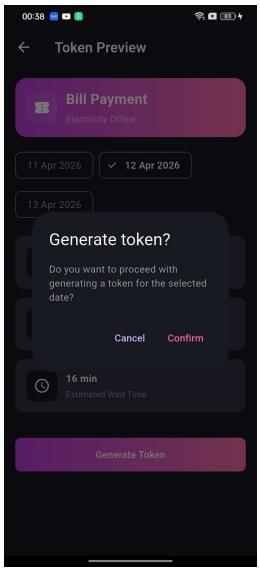

Fig. 6. Token Preview and Confirmation Screen

has been divided into different User levels and it has simple authentication mechanism whereby only authorized persons can login to the system.

Behind the code User credentials are validated against the backend, And your sensitive information must never be exposed directly to the client's side. It possesses a well-defined API so that all communication between the mobile client and the server can be controlled and verified.

Furthermore, data are also stored in MongoDB as nested documents to ensure Data Integrity and Consistency. Every token created is unique and cannot be reused.

The proposed scheme provides protection against the basic security threats and can be extended to incorporate schemes for encryption, security authentication and role-based access for different classes of users e.g admin, staff.

XIII. Applications of the System

The Smart Queue Management System can be utilized in the following areas:

- Hospitals – to prioritize an orderly stream of patients.
- Banks- Cut down on the customers' wait time.
- Government Offices– Enhance Service Delivery.
- Service Centers - Handle the crowd of customers, during the rush hours particularly.

XIV. Process Flow Representation

The overall system process flow is illustrated in Fig. 7, depicting each step from user login to service completion.

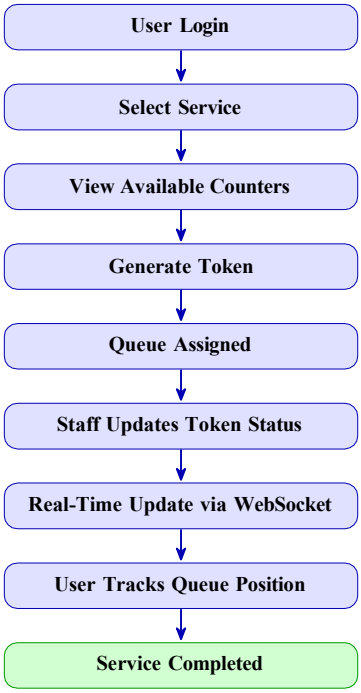

Fig. 7. Overall System Process Flow

XV. Comparison with Traditional System

Table III compares the traditional manual queue systemand a smart queue management system.

XVI. Challenges and Limitations

Although the Smart Queue Management System (SQMS) brings the above mentioned merits, but there are some prob- lems to be solved in this system. A big drawback is that you must have internet connectivity. Since the live communications are at the core of the system, a strong network connection is required to be met.

The challenge of system dependence on a back end server should be added. Should the server be down, It will be impossible for the users to access to queue information, or generate tokens.

TABLE III
COMPARISON BETWEEN TRADITIONAL AND SMART QUEUE SYSTEM

Parameter	Traditional System	Smart System
Queue Type	Physical / Manual	Digital / Virtual
Waiting Time	High	Reduced
Real-Time Tracking	Not Available	Yes (Live Updates)
Token Generation	Paper-Based	Automated via App
Staff Dashboard	Not Available	Web Interface
Notifications	None	Push / SMS Alerts
Scalability	Limited	Multi-Center Support
Overall Efficiency	Low	High

Currently the system does not contain any more functional- ity, such as forecasting waiting time or planning automatically. These are delightfully cool features that result from more data crunching and machine learning.

Given the possibility of having access to users' data, there is a greater need for privacy and security. The security and de- pendability of the system is the subject for future work. When these challenges are met, dramatic performance and robustness improvements in the overall system become possible.

Meeting these challenges will contribute to making the system more robust and effective.

XVII. SYSTEM EVALUATION

The system is rated from three perspectives, which are performance, usability and scalability. The system is being tested by multiple users at the same time, with token being generated simultaneously.

The result shows that the system can effectively.support multiple users at the same time. Executed real-time.updates were sent immediately by Socket.io.

The user interface was tested for friendliness and simplicity. Users managed to keep track of the procedure without much difficulty, and there were no complaints about disorientation when receiving a token.

In addition, The system has excellent scalability for multi- service centers and a large number of users can be supported.

A. Performance Metrics

- Response Time: Fast API response
- Latency: Small delay for real-time updates
- Scalability: Multiple users can join
- Reliability: Stable performance

XVIII. RESULTS AND DISCUSSION

Easy Queue makes waiting more productive and queuing more enjoyable. Real-time position of users in the queue enhances the cooperation between users and staffs. The system is stable and reliable .

The proposed system is evaluated in various cases. The results show the system reduces waiting and crowd congestion significantly.

People can get tokens and monitor their progress remotely in the queue without being physically present. This led to time management and increased satisfaction for the users. Users could purchase tokens in advance and track their progress ina queue.

With Socket.io we guaranteed the experience of both the parties, without any latency issues. The user app and the admin panel are in full sync.

In this respect the system provided satisfactory results in general and we expect the proposed system can be applied to large scale such as Hospitals, Government Applications...etc. It was also tested for usability to see if potential users could operate the robot successfully.

The observerd that users did not find using the application to get records such as generate token and track in the queue too hard. It was also reported that the user interface design and real-time feedback have a positive effect on positive user experience. The app removes the confusion and is transparent at every stage of the process.

XIX. DISCUSSION AND INSIGHTS

The Smart Queue Management Systen describes the usage of the most recent technologies for a mass service waiting time in a real environment. Mobile application with real-time backend communication, is an easy flow for both user and admin.

One of the key lessons from this system is real-time synchronization. With Socket.io, the app naturally keeps you stuck to your queue now no need to refresh. This results in higher user engagement and less anxiety.

In addition, they are also focused on convenience for the customers. It is also different from the traditional system in that users do not need to queue in person. These virtual lines provideparticipants with a way to avoid unnecessary crowding, as they should only appear when it is nearly their turn.

In terms of technology, the architecture is scalable. RESTful APIs and MongoDB are a good match for customizable and scalable data handling. It is modular in design and so it is easy to extend for example with notifications, analytics and predictions using AI.

The system's success depends on the face of it is. Thanks to the seamless design of the app, non-tech users can handle the app without any trouble. Features like token preview, real- time updates, simple navigation, contribute to a better user experience.

However, the system tells us that the refinement of the system is an ongoing process.

Additional features like predictive analytics and automated scheduling could be added to further enhance system perfor- mance and user satisfaction.

Overall, this system illustrates that the traditional service system is greatly enhanced by the integration of mobile technology and real time communication.It is based on a user- friendly practical efficient solution that can be seamlessly scaled and also it is possible to be applied in different industries.

XX. Future Scope

Future improvements include:

- Push notifications
- AI-based wait prediction
- Cloud deployment

XXI. Conclusion

The SQMS has been designed to provide a great solution to the problem faced by the traditional queue system. This system reduces physical congestion and improves the efficiency of the entire service by enabling users to remotely queue and check their status in real time.

The use of modern technology such as Flutter, Node.js and Socket.io enhances the user experience with real time updates and improved group coordination. It offers a clean, scalable and easy to use API to suit a variety of assumptions related to the service environment.

Overall, our method makes the user's life easier and the service cost and flow complexity smaller. Future research may involve the development of notifications and/or predictions, which allow for even more intelligent queue management.

References

1. S. Sharma and R. Gupta, "Smart Queue Management Systems: A Review," International Journal of Computer Applications, vol. 182, no. 12, pp. 25–30, 2023.
2. M. Patel, "Real-Time Queue Monitoring Using Web Technologies," IEEE Conference on Smart Systems, pp. 120–125, 2022.
3. Node.js Documentation, Available: https://nodejs.org
4. Flutter Documentation, Available: https://flutter.dev
5. MongoDB Documentation, Available: https://www.mongodb.com
6. Express.js Guide, Available: https://expressjs.com
7. Socket.io Documentation, Available: https://socket.io

A. Kumar and P. Singh, "Digital Queue Systems in Healthcare," Journal of Information Systems, vol. 15, no. 3, pp. 45–52, 2024.

8. R. Verma, "Improving Customer Experience Using Smart Queue Sys- tems," International Journal of Engineering Research, vol. 11, no. 2, pp. 88–94, 2023.
9. K. Mehta, "Cloud-Based Queue Management Solutions," IEEE Access, vol. 10, pp. 56789–56800, 2022.
10. P. Singh and A. Verma, "Queue Management Optimization in Service Systems," International Journal of Advanced Computer Science, vol. 9, no. 4, pp. 112–118, 2022.
11. R. K. Sharma, "Mobile-Based Queue Management Solutions," Journal of Emerging Technologies, vol. 14, no. 2, pp. 55–60, 2023.

A. Joshi, "Design and Implementation of Smart Service Systems," IEEE Conference on Computing Systems, pp. 210–215, 2021.

12. D. Meena and S. Kulkarni, "Real-Time Data Handling in Distributed Systems," International Journal of Data Science, vol. 8, no. 1, pp. 30–36, 2022.
13. T. Roy, "Improving Customer Experience Using Digital Queue Sys- tems," International Journal of Engineering Research, vol. 10, no. 5, pp. 75–80, 2023.

