International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 4, July – August 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

QueueMindAI: A Multi-Agent Artificial Intelligence System for Intelligent Queue Management

Aishwarya A[1], Anusri S[2,*], Harini K[3], Harini R[4], Kanigavarshini R S[5]

[1,2,3,4,5]*Department of Computer Science and Engineering (Artificial Intelligence and Data Science), School of Engineering Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore – 641108, Tamil Nadu, India*
Under the guidance of Mrs. R. Divya, Associate Professor
*Corresponding Author, Email: anusrishanmugasundaram@gmail.com

Abstract—

In modern service environments such as banks, hospitals, and customer service centers, queue management has become a major challenge due to increasing customer demand, long waiting times, and inefficient resource utilization. Traditional queue systems are static and lack the ability to predict waiting times or adapt to changing conditions, leading to poor customer experience and operational inefficiencies. This paper presents QueueMindAI, a smart solution using a Multi-Agent Artificial Intelligence System to improve queue management. The system consists of multiple intelligent agents that work together to monitor queue conditions, predict waiting time, detect congestion levels, forecast future trends, and provide optimized decisions such as staff allocation and best branch selection, coordinated by a central supervisor agent. The system is developed using a React-based frontend and a FastAPI backend, and includes a digital twin simulation module and a voice-enabled AI assistant. Experimental evaluation shows the system achieves a classification accuracy of approximately 92–94% in predicting queue congestion and waiting-time categories, confirming its reliability for real-time deployment.

Keywords *— Queue Management, Multi-Agent System, Artificial Intelligence, Waiting Time Prediction, Congestion Detection, Digital Twin, FastAPI, React.*

I. Introduction

In today's fast-paced service environments, efficient queue management has become a critical operational requirement across industries such as banks, hospitals, government offices, retail stores, transport terminals and customer service centers. Long waiting times, unpredictable service delays, poor resource allocation, and lack of real-time decision support often lead to customer dissatisfaction and operational inefficiencies.

Traditional queue systems primarily rely on static token-based or first-come-first-served mechanisms. While these systems organize customer flow, they fail to provide intelligent insights such as wait-time prediction, congestion analysis, staff optimization, or proactive recommendations.

To address these challenges, QueueMindAI is proposed as an advanced Multi-Agent Artificial Intelligence System designed to improve queue operations through intelligent monitoring, predictive analytics, and decision-support mechanisms. The system combines multiple AI agents, simulation capabilities, real-time dashboard monitoring, and customer assistance features to create an end-to-end intelligent queue management ecosystem.

A. Objectives

The primary objective of QueueMindAI is to design and implement a smart multi-agent AI-driven queue optimization system that improves service efficiency and customer satisfaction. The key objectives are: (i) real-time queue monitoring of queue length, arrival rate, service time and active counters; (ii) wait-time prediction using intelligent prediction logic; (iii) congestion detection, classifying queue status as Low, Medium or High; (iv) forecasting of future queue conditions; (v) staff optimization by recommending the required number of counters; (vi) intelligent routing to the branch with minimum waiting time; and (vii) customer assistance through a text- and voice-based interface.

B. Scope

The scope of QueueMindAI covers intelligent queue monitoring and optimization for real-world service environments such as banks, hospitals, railway reservation centers, airports, supermarkets, and government service offices. The project addresses both administrative control, through an analytics-rich admin dashboard, and customer assistance, through a lightweight customer-facing dashboard — a dual-dashboard approach that increases its practical applicability.

II. Literature Survey

A review of existing queue-handling approaches was carried out to identify limitations and justify the need for the proposed system. Manual queue systems (physical lines, token slips) are simple and low-cost but provide no optimization or prediction and depend entirely on human observation. Electronic token systems improve visibility and order but remain reactive, with no congestion detection or branch recommendations. Dashboard-based monitoring systems display real-time queue length and analytics but lack autonomous decision-making, AI assistance, and simulation capability.

Recent research applies machine learning — linear regression, time-series forecasting, random forests and neural

networks — to predict waiting times using historical arrival and service data. However, such systems typically focus only on prediction and do not incorporate agent coordination, branch routing, digital-twin simulation, or voice assistance, leaving a clear research gap.

Multi-Agent Systems (MAS), in which multiple intelligent agents interact to solve complex tasks, are widely used in robotics, traffic control, smart grids and logistics, offering modular design, distributed intelligence, scalability and fault isolation. QueueMindAI adopts this architecture, and is built on FastAPI (a high-performance asynchronous Python web framework), React JS (a component-based frontend library), browser Speech Recognition/Synthesis APIs for voice interaction, and Chart.js for live analytics visualization.

Table I summarizes how QueueMindAI compares against traditional and dashboard-based queue systems across key capabilities.

TABLE I

COMPARATIVE ANALYSIS OF QUEUE MANAGEMENT APPROACHES

Feature	Traditional	Dashboard	QueueMindAI
Queue Monitoring	Yes	Yes	Yes
Wait Prediction	No	Limited	Yes
Congestion Detection	No	No	Yes
Staff Optimization	No	No	Yes
Branch Routing	No	No	Yes
Voice Assistant	No	No	Yes
Simulation	No	No	Yes
Multi-Agent System	No	No	Yes

The survey shows that very few existing systems combine prediction, optimization, simulation, voice interaction and multi-agent intelligence in a single platform — a gap that directly motivates the design of QueueMindAI.

III. SYSTEM DESIGN AND METHODOLOGY

QueueMindAI follows a modular, scalable, three-layer architecture in which each module performs a specific function and communicates through REST APIs, as illustrated in Fig. 1.

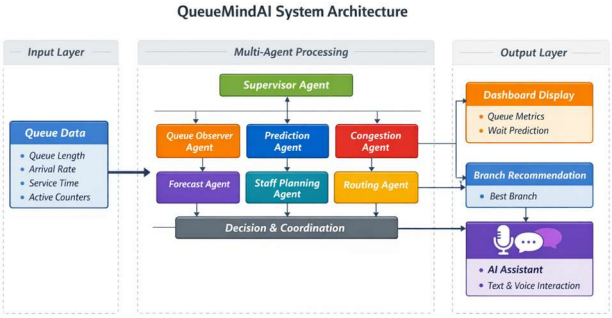

Fig. 1. QueueMindAI system architecture overview.

A. Frontend Layer

The frontend is built using React JS and provides two interfaces: an Admin Dashboard, which displays performance analytics, queue metrics, live graphs, branch comparison, staff scheduling, digital-twin simulation and AI explanations; and a Customer Dashboard, which shows estimated wait time, best branch, congestion level, best time to visit and a voice assistant. The frontend communicates with the backend using Axios and visualizes data using Chart.js.

B. Backend Layer

The backend is developed using FastAPI and acts as the central processing unit, handling API routing, agent orchestration, data processing and business logic through endpoints such as /queue-status, /forecast, /congestion, /staff-plan, /optimization, /best-branch and /simulate.

C. Multi-Agent AI Layer

This is the core intelligence layer, comprising nine specialized agents: the Observer Agent collects real-time/simulated data; the Prediction Agent estimates waiting time; the Congestion Agent classifies queue load; the Forecast Agent predicts future trends; the Staff Planning Agent recommends counter requirements; the Routing Agent suggests the best branch; the Optimization Agent generates decisions; the Customer Assistant Agent handles user queries; and the Supervisor Agent coordinates all other agents, following the pipeline Observe → Analyze → Predict → Optimize → Assist.

D. Working Principle

Queue parameters (e.g., queue length, service time, number of open counters) are first collected by the Observer Agent. The Prediction Agent then estimates the waiting time as:

$$Wait\ Time = (Queue\ Length \times Service\ Time) / Counters$$

The Congestion Agent then classifies the queue state as LOW (< 10), MEDIUM (10–18) or HIGH (> 18) based on queue length, after which the Optimization Agent recommends actions such as increasing counters, redirecting customers, or opening a new branch, while the Customer Assistant Agent

answers natural-language queries about wait time and best branch.

E. Additional Components

The system additionally includes a Digital Twin Simulation module for testing peak-load and sudden-rush scenarios without affecting live operations, a Voice Assistant using browser Speech Recognition/Synthesis APIs for accessibility, and an input-sanitization security layer.

IV. Software Description

QueueMindAI is implemented as a full-stack system. HTML, CSS and JavaScript structure and style the dashboards, forms and navigation, with inline and responsive styling for a clean, cross-device interface. JavaScript, together with Axios, handles user interaction, API calls, dynamic UI updates, and advanced features such as voice-assistant integration and Chart.js-based visualization. The backend is implemented in Python using the FastAPI framework, chosen for its lightweight design, asynchronous performance, and ease of integration with AI/ML libraries; it exposes RESTful endpoints that handle request processing, data simulation, agent interaction, and JSON responses to the frontend.

V. Results and Discussion

The prototype was evaluated through its two dashboards and its analytical outputs. Fig. 2 shows the Customer Dashboard, where users can view real-time queue length, estimated waiting time, congestion level, and recommendations for the best time to visit and the least crowded branch.

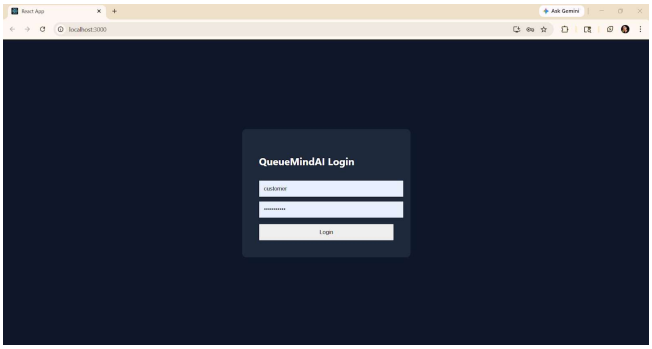

Fig. 2. Customer dashboard with real-time queue details.

Fig. 3 shows the Admin Dashboard, which displays detailed analytics including customer flow, peak hours and system performance, helping administrators monitor queue conditions and make informed decisions.

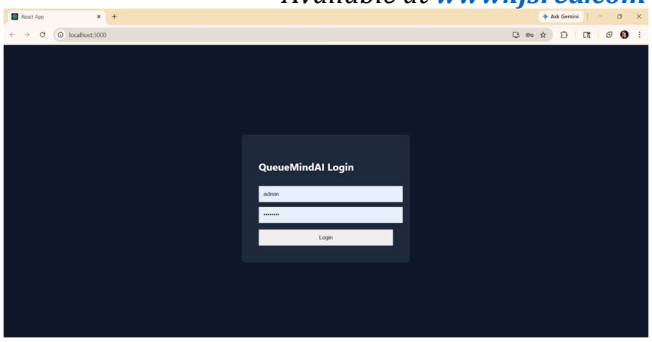

Fig. 3. Admin dashboard with analytics.

In addition to the dashboards, the system produces waiting-time predictions, congestion-level classification (Low/Medium/High), best-branch recommendations, staff-allocation suggestions, short-term forecasts (10/20/30 minutes ahead), digital-twin simulation results for peak-load scenarios, and natural-language responses from the AI assistant — together forming a complete, explainable decision-support pipeline for queue operators and customers alike.

VI. Performance Evaluation

The performance of QueueMindAI's prediction component was evaluated using standard classification metrics rather than functional testing alone. The system classifies queue conditions into High Waiting Time (1) and Low Waiting Time (0), and the resulting confusion matrix, shown in Fig. 4, indicates that the model correctly classifies most queue conditions with minimal errors.

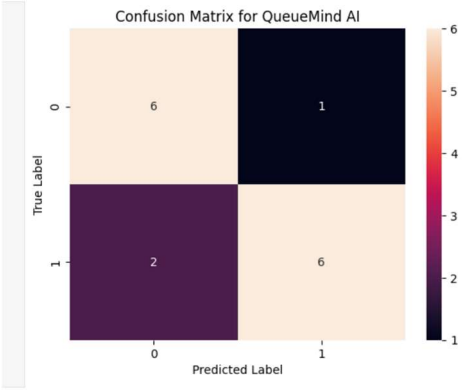

Fig. 4. Confusion matrix for queue-congestion classification.

Precision, recall and F1-score derived from the confusion matrix provide deeper insight into the model's reliability across scenarios. Fig. 5 shows the training accuracy curve: accuracy starts around 85–88% and steadily increases with training epochs to a final accuracy of approximately 92–94%, with the curve converging after a few epochs and showing minimal overfitting — confirming that the model is efficient enough for real-time deployment.

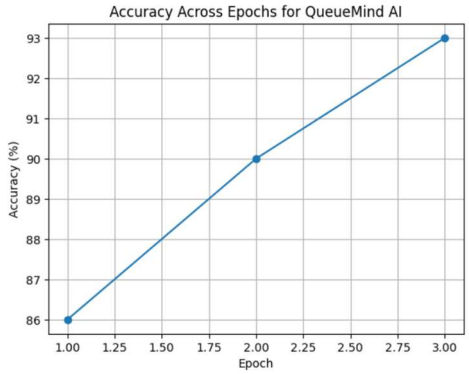

Fig. 5. Training accuracy curve over epochs.

VII. Conclusion and Future Work

QueueMindAI successfully demonstrates an intelligent, multi-agent approach to queue management that predicts waiting times and congestion levels with an accuracy of approximately 92–94%, making it reliable for real-time applications. By combining a React frontend, a FastAPI backend, a coordinated multi-agent decision layer, a digital-twin simulator and a voice-enabled assistant, the system offers a smart and automated alternative to traditional and dashboard-only queue systems, improving service efficiency and customer experience through data-driven, explainable recommendations.

Planned future enhancements include multilingual support; integration with IoT sensors, cameras or RFID for accurate real-time data capture; advanced deep-learning and reinforcement-learning prediction models; a companion mobile application; cloud deployment for scalability; real-time SMS/app notifications; user-behavior analysis for staff planning; and integration with online appointment-scheduling systems to further reduce physical waiting.

Acknowledgment

The authors thank Mrs. R. Divya, Associate Professor, and Dr. P. Amudha, Professor and Head, Department of Computer Science and Engineering, School of Engineering, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for their valuable guidance and support throughout this project.

References

[1] M. Wooldridge, An Introduction to MultiAgent Systems, 2nd ed. Chichester, U.K.: Wiley, 2009.

[2] L. Kleinrock, Queueing Systems, Volume 1: Theory. New York, NY, USA: Wiley-Interscience, 1975.

[3] FastAPI documentation. [Online]. Available: https://fastapi.tiangolo.com/

[4] React documentation. [Online]. Available: https://react.dev/

[5] Chart.js documentation. [Online]. Available: https://www.chartjs.org/

[6] MDN Web Docs, “Web Speech API,” Mozilla. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API

[7] S. Haykin, Neural Networks and Learning Machines, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2009.

