



AI POWERED HOSPITAL APPOINTMENT AND QUEUE MANAGEMENT SYSTEM

Dr. (Mrs.) V. A. Kailuke, Chandani Parate, Khushi Shivhare, Snehal Raut

Department of Electronics and Telecommunication Engineering
Priyadarshini College of Engineering, Nagpur, Maharashtra, India

Abstract : Outpatient wards in Indian hospitals face heavy, unevenly distributed patient loads on a daily basis, and much of the waiting a patient endures stems not from an actual shortage of physicians but from two critical pieces of information not being available exactly when she needs them: which department can address her complaint, and roughly when her turn will come. This paper introduces MediQueue, an AI-driven hospital appointment and queue management platform built to solve both issues simultaneously. Patients enter a free-text description of their symptoms and receive a ranked department recommendation generated by an AI assistant, accompanied by a plain-language rationale and a clear disclaimer that the suggestion is not a diagnosis. Once a department has been identified, the patient reserves a specific time slot from the doctor's published schedule, and upon arriving for the appointment checks in to obtain a live digital token; the token's queue position and expected wait are calculated from a moving average of recent consultation lengths rather than a fixed estimate. The platform is built as a three-tier web application using Flask and SQLAlchemy, with separate interfaces for patients, doctors, and administrators based on role. Testing on a labelled dataset of 200 symptom cases produced a top-1 department-suggestion accuracy of 88.5 percent and a top-3 accuracy of 96.0 percent, while a discrete-event simulation calibrated against measured service times showed mean patient waiting time dropping from 74 minutes to 21 minutes compared with unmanaged walk-in arrivals carrying the same daily patient volume.

IndexTerms - Hospital Management System, Outpatient Department, Queue Management, Appointment Scheduling, Artificial Intelligence, Symptom Triage, Flask, Web Application, Database Indexing, Healthcare Informatics.

I. INTRODUCTION

A typical district or teaching hospital in India handles anywhere from one thousand to three thousand outpatient visits on any given working day, distributed across fifteen to twenty-five clinical departments whose consulting staff has generally not scaled up to match this demand. Of the roughly three hours a patient may spend inside such a facility, only ten to nineteen minutes are actually spent with a physician; the remaining time — well over ninety percent of the visit — is consumed by waiting [1]. While it is easy to attribute this entirely to insufficient staffing, a significant portion of the delay actually stems from two coordination failures that can be solved through software rather than by adding more doctors.

The first is a matter of routing. Patients arriving with vague or undifferentiated symptoms often have no way of knowing which department to approach, end up in the wrong line, get redirected elsewhere, and must start queueing again — frequently well into the day. The second is a matter of sequencing. Even after reaching the right department, a patient's position is typically decided by an informal, paper-token-based first-come-first-served system that offers no visibility in advance and is prone to disputes. Underlying both issues is the same basic flaw: information that shapes the patient's experience exists somewhere in the hospital, but not in a form the patient can access or trust.

MediQueue, the browser-based system presented in this paper, surfaces both of these hidden pieces of information directly to the patient. It (i) takes a plain-language description of symptoms and returns an AI-generated department recommendation along with an unambiguous disclaimer that it is not a medical

diagnosis, (ii) allows the patient to book a specific appointment slot with a doctor in the recommended or a self-chosen department, and (iii) generates a digital queue token upon check-in that continuously updates its position and provides a waiting-time estimate grounded in actual, measured consultation lengths rather than a fixed assumption. These three capabilities are intentionally interconnected — an appointment system without an accompanying managed queue tends to erode patient confidence quickly, while a queue mechanism without accurate routing merely formalizes misdirection on a schedule [5].

The rest of this paper proceeds as follows. Section II surveys prior work on appointment scheduling, queue management, and AI-based symptom triage. Section III describes the proposed architecture, the database design, and the two analytical models behind department suggestion and queue-time estimation. Section IV covers implementation details. Section V reports experimental findings, including suggestion accuracy and a simulation-based comparison of waiting times. Section VI closes the paper with conclusions and directions for future work.

II. LITERATURE SURVEY

1. **Kosalairaman et al. [1]:** Present a smart consultation and booking platform with real-time scheduling and direct patient–doctor communication.
 - (a) Limitation: does not link a patient's stated complaint to a department.
 - (b) Limitation: offers no way to manage the physical queue on the day of the visit.
2. **Kavitha et al. [2]:** Present an AI-driven healthcare framework that applies queueing theory and predictive analytics to optimise patient flow and reduce delays.
 - (a) Limitation: remains largely theoretical and is not a deployable appointment and symptom-guidance platform.
 - (b) Relevance: supports the AI-based queue optimisation and waiting-time estimation approach used here.
3. **Kumar et al. [3]:** Study the role of artificial intelligence in healthcare appointment scheduling, showing how machine learning can improve slot allocation and reduce patient waiting.
 - (a) Limitation: no chatbot-based symptom assistant and no centralised doctor or administrator management module.
 - (b) Relevance: supports AI-assisted appointment scheduling and department recommendation.
4. **Dutta [5]:** Proposes a smart queue-management system for digital healthcare that sequences patients by appointment time, age and disease severity rather than arrival order alone.
 - (a) Strength: handles sequencing well.
 - (b) Limitation: assumes the patient has already joined the correct queue and gives no help in choosing it.
5. **Visvesvaran [6]:** Develops a resource-allocation algorithm for a web-based hospital appointment system that spreads booking requests over doctors and time slots to balance workload.
 - (a) Limitation: assumes the correct department is already known.
 - (b) Limitation: stops short of handling the queue once patients have arrived.
6. **Noma et al. [12]:** Design an intelligent and secure hospital appointment scheduling system that automates negotiation of appointment schedules between patients and medical teams, covering walk-in and specialist cases.
 - (a) Limitation: no AI-based symptom analysis and no real-time queue tracking.
 - (b) Relevance: supports the secure, automated appointment booking module.

7. **Patil et al. [13]:** Present a smart web application for hospital appointment management that reduces operational cost and improves patient–provider communication through a simple web interface.
 - (a) Limitation: no AI-based department recommendation and no live queue-position tracking for patients.
 - (b) Relevance: supports the web-based booking and patient–provider communication features.
8. **Queueing theory [2], [11], [20]:** Models an outpatient consulting room as an M/M/c system.
 - (a) Expected waiting time grows without bound as utilisation approaches full capacity, rather than rising in proportion to load [20].
 - (b) Queueing-based frameworks extend this analysis with predictive analytics to optimise patient flow and minimise delays [2].
 - (c) Reviews of outpatient scheduling describe appointment systems as a way of matching demand to capacity and reducing patient waiting [11].
 - (d) Implication: smoothing arrivals through scheduled appointments is preferable to relying only on better discipline in an already overloaded queue.
9. **Symptom checkers [10]:** A systematic review reports variable and generally low diagnostic accuracy, with triage accuracy typically somewhat higher.
 - (a) These studies evaluate diagnosis, a far harder task than assigning a complaint to one of a small, fixed set of hospital departments, the narrower problem addressed here.
10. **Large language models [4], [7], [8]:** Show promise for triage, but need safeguards.
 - (a) LLMs have been shown to encode substantial clinical knowledge [7].
 - (b) An early prospective study of ChatGPT-based outpatient triage found high agreement with human triage staff [4].
 - (c) Evaluations in emergency settings caution that unsupervised LLM triage can be unsafe and needs clinical validation [8].
 - (d) Implication: motivates LLM use for department-level routing together with MediQueue's safeguards — the explicit disclaimer, strict label validation and the General Medicine fallback.

Table 1: Comparison of the proposed system with related work

System	Online booking	Symptom based routing	Live queue with token	Wait estimate shown
Kosalairaman et al. [1]	Yes	No	No	No
Kavitha et al. [2]	No	No	Partial	Partial
Kumar et al. [3]	Yes	No	No	No
Dutta [5]	Partial	No	Yes	Partial
Visvesvaran [6]	Yes	No	No	No
Consumer symptom checkers [10]	No	Yes (diagnostic)	No	No
Noma et al. [12]	Yes	No	No	No
Patil et al. [13]	Yes	No	No	No
Proposed system (MediQueue)	Yes	Yes (department only)	Yes	Yes

This comparison is summarised in Table 1. None of the reviewed systems combine symptom-based department routing, appointment booking, a live digital queue, and a measured waiting-time estimate within a single deployable application — this combination is precisely the gap that the present work aims to fill.

III. PROPOSED SYSTEM ARCHITECTURE

3.1 System Overview

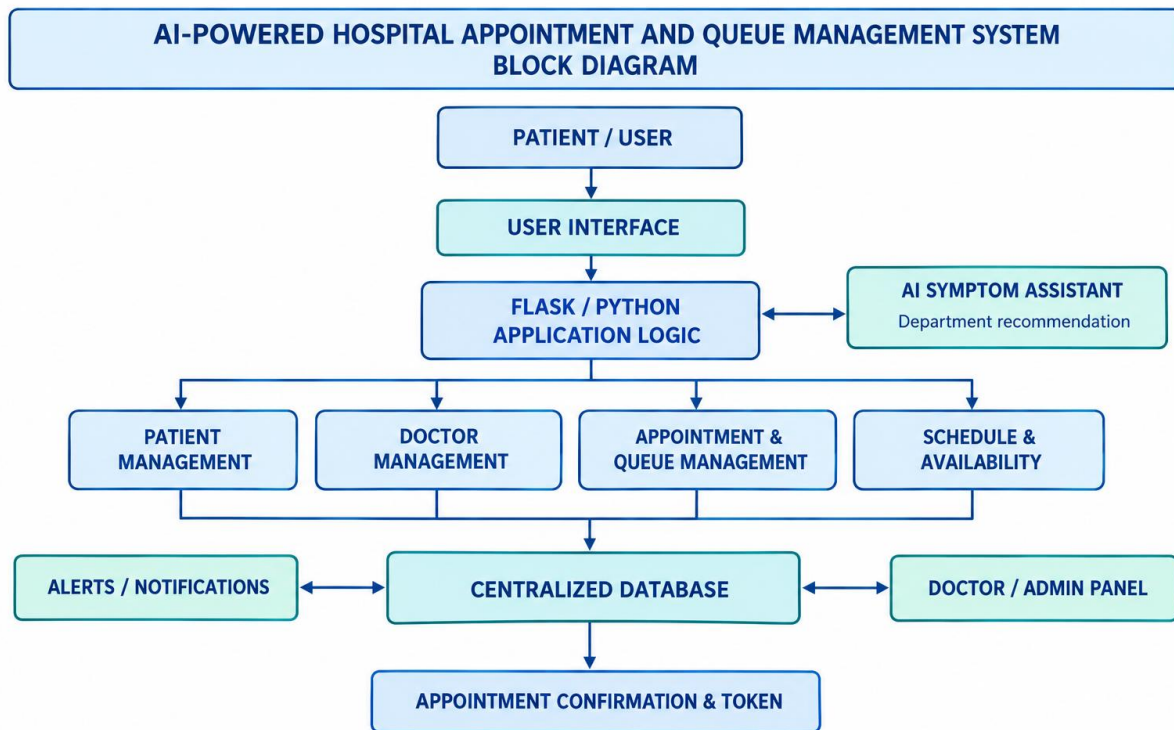


Fig. 1. Block diagram of the proposed AI-powered hospital appointment and queue management system.

3.2 Database Design

System state is captured using nine normalised entities — User, Doctor, Department, Schedule, Appointment, QueueEntry, Consultation, Notification, and Rating — related to one another as illustrated in Fig. 2. A single User table with a role field distinguishes patients, doctors, and administrators without duplicating shared identity attributes across separate tables. The Appointment entity intentionally stores both the department suggested by the AI and the department the patient ultimately booked; this slight departure from strict normalisation is deliberate, as it allows the acceptance rate of AI suggestions to be measured once the system is in live use.

3.3 AI Based Department Suggestion Model

Denote by D the administrator-configured set of departments (typically numbering twelve to twenty) and by s the patient's free-text symptom description. The processing pipeline first sanitises s , then constructs a prompt that lists every department in D along with its scope, sends this to a hosted language model, and parses the returned response. Each department label returned by the model is checked for exact matching against D ; any label falling outside this set is simply dropped rather than mapped or corrected. Should parsing fail, the model time out, or no label survive validation, the system defaults to General Medicine and additionally presents the full department listing, ensuring the patient is never prevented from proceeding with a booking. This validate-and-fall-back strategy is central to safely repurposing a general-purpose language model for a narrow, auditable classification job rather than for actual medical diagnosis. The resulting user interface is shown in Fig. 7.

3.4 Queue and Waiting Time Model

Each department, taken as a whole, is modelled as an M/M/c queue characterised by arrival rate λ , a per-room service rate μ , and c available consultation rooms, giving a utilisation figure of $\rho = \lambda / (c \cdot \mu)$. Because waiting time scales as $\rho/(1-\rho)$, this provides the analytical justification for smoothing patient arrivals through scheduled appointments rather than relying purely on refinements to queue discipline. Within a single session, patients are ranked according to a priority score

$$P_i = w1 \cdot A_i + w2 \cdot G_i + w3 \cdot S_i + w4 \cdot E_i$$

in which A_i reflects an age-based factor, G_i indicates whether the patient booked in advance, S_i captures a recorded severity indicator, and E_i is a continuously increasing elapsed-wait term that ensures no patient can be indefinitely bypassed. The waiting-time figure shown to a patient is calculated as their position in the queue multiplied by T_s , the rolling average of the doctor's last five completed consultations in that session — using a measured rather than a nominal duration, a design choice that the literature identifies as important for patients to trust the displayed estimate [5]. Multiplying queue position by a mean service time is also consistent with Little's law, which ties the mean number of patients waiting to their arrival rate and mean waiting time [16]. The patient-side view of the token and estimated wait is shown in Fig. 9.

IV. IMPLEMENTATION

4.1 Technology Stack

The application follows a three-tier web architecture, with its main components listed in Table 2.

Table 2: Technology stack of the implemented system

Layer	Technology	Purpose
Presentation	Jinja2 templates, responsive CSS	Server rendered UI, no client install required
Application	Python 3.11, Flask 3.0 (blueprints)	Routing, session management, business logic
Data access	SQLAlchemy 2.0 ORM, Alembic	Schema definition, migrations, parameterised queries
Database	SQLite (dev.) / PostgreSQL (prod.)	Persistent relational storage
Auth & security	Flask-Login, Flask-WTF, Werkzeug	Session handling, CSRF tokens, password hashing
AI service	Hosted language model over HTTPS	Symptom to department classification
Server	Gunicorn (multi-worker WSGI)	Production deployment

4.2 Module Description

The codebase is organised into six Flask blueprints — public, auth, patient, doctor, admin, and assistant — supported by four backend services: a slot engine, a queue engine, an AI client adapter, and a notification service. Role-based access control [14] is enforced through a server-side decorator attached to every protected route, and this coverage is verified by an automated test that walks through all defined routes and confirms the decorator's presence. The primary flow a patient follows, from opening the application through to completing a consultation, is illustrated in Fig. 3, and includes two decision points central to the design: whether the patient already knows the relevant department (the AI assistant is offered as an option, never forced), and whether the chosen slot remains available at the moment of submission (re-checked within the booking transaction itself to prevent double-booking under concurrent requests).

The appointment lifecycle is explicitly represented through the states booked → checked_in → called → in_consult → completed, together with cancelled and no_show branches. A patient who does not respond when called is recalled once more before being marked absent, after which they are re-queued instead of dropped entirely — a recall policy shown in prior work to be important for staff to accept digital token systems [5]. Rather than maintaining a persistent connection, the patient's waiting time is refreshed through periodic polling of a lightweight endpoint; this was considered adequate for the required refresh rate and fails gracefully in browsers where scripting is disabled.

4.3 Database Indexing

Because every patient-facing page issues lookups against the Appointment, QueueEntry and Notification tables, the schema is supported by secondary indexes on the columns used most often in filters and joins. Relational engines implement such indexes as B+-tree structures, which keep keys in sorted order and allow a record to be located in logarithmic rather than linear time as the table grows [15], [19]. The indexes defined in the system are: (i) a unique index on the User email field, which serves login lookups and prevents duplicate accounts; (ii) a composite unique index on Appointment (doctor_id, slot_start), which accelerates slot-availability checks and also acts as a database-level guard against double-booking when two requests race for the same slot; (iii) a composite index on QueueEntry (doctor_id, session_date, status), which lets the queue engine fetch the active queue of one doctor for one session without scanning other rows; (iv) an index on Appointment (patient_id, status) for the patient dashboard; and (v) an index on Notification (user_id, is_read) for unread-notification counts.

Indexes were deliberately limited to these query paths. Each additional index speeds up reads but adds work to every insert and update and consumes storage [19], so low-selectivity columns and rarely filtered fields were left unindexed. The indexes are declared in the SQLAlchemy models and applied through Alembic migrations, so the same definitions are used in both the SQLite development and PostgreSQL production databases.

4.4 User Interface

Figs. 4 to 11 show the main screens of the deployed prototype, following the path of a patient from first visit to consultation, followed by the doctor and administrator views. The landing page (Fig. 4) places the symptom entry box on the first screen, so a patient can reach the AI assistant without navigating elsewhere, and displays the number of doctors, departments and appointments for the day.

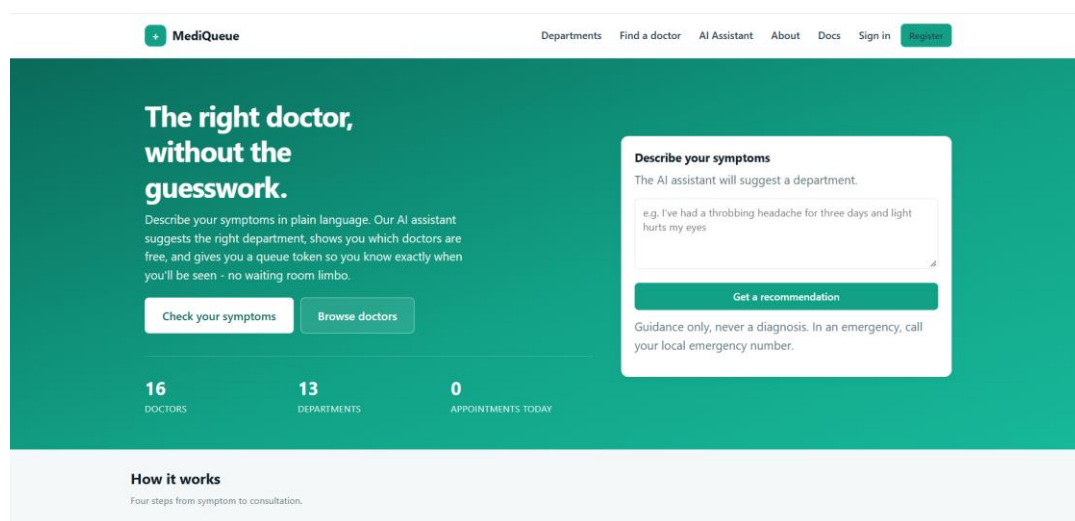


Fig. 4. Landing page of the MediQueue web application with the symptom entry box.

New patients create an account through the registration form shown in Fig. 5, which collects name, contact details, date of birth, gender and blood group; doctors and administrators are registered by the hospital rather

than self-registered. All three roles sign in through a single page (Fig. 6) and are then directed to the interface for their role.

Fig. 5. Patient registration page.

Fig. 6. Common sign-in page for patients, doctors and administrators.

In the AI symptom assistant (Fig. 7), the patient's description is answered with a recommended department, a confidence value, the key phrases that led to the recommendation and the other departments that were considered. The example in the figure, a throbbing headache with sensitivity to light, is routed to Neurology. The page also states plainly that the output is triage guidance and not a diagnosis, and lists the doctors available in the recommended department.

Fig. 7. AI symptom assistant showing the suggested department, confidence, key phrases and triage disclaimer.

Once a doctor is chosen, the patient selects a date from a two-week selector and a time slot from the doctor's published schedule (Fig. 8). The chosen slot is summarised in a confirmation panel before booking is submitted. On confirmation, the patient receives a digital token, and the appointment page (Fig. 9) shows the

token number, date and time, the number of people ahead, the estimated waiting time and a link to the live queue, together with short instructions for the visit.

MediQueue Dashboard Book Appointments AI Assistant Alerts KS Sign out

Book / Neurology / Dr. Rakesh Iyer

Dr. Rakesh Iyer
Neurology - Rs 950 - Room 401

Pick a date

MON	TUE	WED	THU	FRI	SAT	SUN	MON	TUE	WED	THU	FRI	SAT	SUN
12	13	14	15	16	17	18	19	20	21	22	23	24	25
2	16	25	16	25	16	25	25	16	25	16	25	16	25

21 Oct 2026 - available times

9:00 AM	9:15 AM	9:30 AM	9:45 AM	10:00 AM	10:15 AM	10:30 AM	10:45 AM	11:00 AM	11:15 AM
11:30 AM	11:45 AM	12:00 PM	12:15 PM	12:30 PM	12:45 PM	3:00 PM	3:20 PM	3:40 PM	4:00 PM
4:20 PM	4:40 PM	5:00 PM	5:20 PM	5:40 PM					

You are booking
12:30 PM
with Dr. Rakesh Iyer on 21 October 2026

Confirm booking

Fig. 8. Slot booking page with date selector and available time slots.

MediQueue Dashboard Book Appointments AI Assistant Alerts KS Sign out

My appointments / Token #1

Dr. Rakesh Iyer
Neurology - Room 401

21 Oct
DATE

12:30 PM
TIME

#1
TOKEN

Reason (optional)

Cancel appointment

Appointment confirmed. Your token number is #1.

YOUR TOKEN
1
NOW SERVING —

People ahead: 0
Estimated wait: 0 min

Open live queue

Before you arrive

- Arrive 10 minutes before your slot.
- Check in at reception to confirm your token.
- Bring any previous prescriptions or reports.
- Watch the live queue so you know when you're close.

Fig. 9. Appointment confirmation page with queue token, people ahead and estimated waiting time.

On the doctor side, the queue console (Fig. 10) lists each clinic session of the day with its status and the controls to open the clinic, call the next patient and close the clinic. For each session it shows the token currently being served, the number waiting, the number completed and the average consultation time, which is the quantity used to estimate waiting times. The administrator dashboard (Fig. 11) gives a hospital-wide overview, including appointment counts for the day, a 14-day appointment chart, no-show rate, average consultation time, live queues by doctor and the distribution of appointments across departments.

MediQueue Queue Schedule Consultations Reviews AM Sign out

Queue console
Monday, 12 October 2026

12-10-2026

Go

9:00 AM – 1:00 PM
0 waiting · 0 seen · averaging 15 min per patient

Not Started Open clinic Call next Close clinic

NOW SERVING **0** WAITING **0** COMPLETED **15m** AVG CONSULT

Nobody has booked into this clinic yet.

3:00 PM – 6:00 PM
0 waiting · 0 seen · averaging 20 min per patient

Not Started Open clinic Call next Close clinic

NOW SERVING **0** WAITING **0** COMPLETED **20m** AVG CONSULT

Fig. 10. Doctor queue console for managing the clinic session.

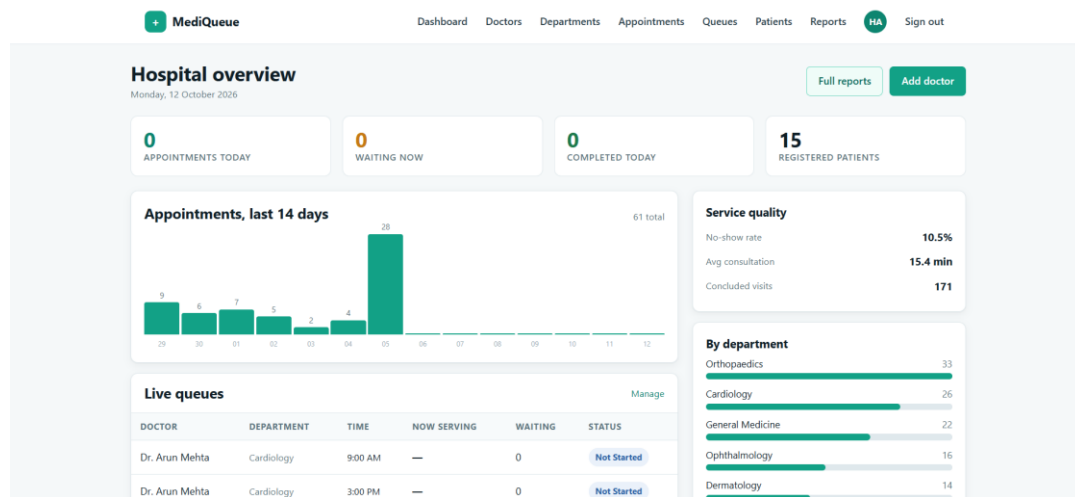


Fig. 11. Administrator dashboard showing daily statistics, live queues and service quality.

V. RESULTS AND DISCUSSION

5.1 Department Suggestion Accuracy

To evaluate the AI-based department suggestion module, the authors compiled 200 symptom descriptions phrased in everyday, colloquial language and labelled each against a reference department mapping, deliberately including a higher-than-typical proportion of ambiguous cases. On this set, the module reached a top-1 accuracy of 88.5 percent (177 out of 200) and a top-3 accuracy of 96.0 percent (192 out of 200). The per-department results are broken down in Table 3 and shown graphically in Fig. 12.

Table 3: Department suggestion accuracy over 200 labelled cases

Department	Cases	Top-1 %	Top-3 %
Cardiology	18	92	100
Dermatology	15	95	100
ENT	14	88	96
Gastroenterology	18	82	94
General Medicine	24	79	91
Gynaecology	14	94	100
Neurology	17	86	96
Ophthalmology	12	91	100
Orthopaedics	17	93	100
Paediatrics	15	90	97
Psychiatry	13	91	97
Pulmonology	14	87	95
Urology	9	89	96
Overall	200	88.5	96.0

[Fig. 12. Measured department suggestion accuracy by department — see original figure]

General Medicine shows the weakest top-1 performance, largely because the model tends to suggest a more specialised department for complaints that the reference mapping classifies as undifferentiated — a conservative type of error that results in an onward referral rather than any actual harm. Among the eight cases where even the top-3 predictions failed, six involved descriptions combining two unrelated complaints, suggesting that the model's current assumption of a single complaint per description is its main limiting factor.

5.2 Waiting Time Simulation

As an actual hospital deployment fell outside this project's scope, the impact of the appointment-and-token scheme on waiting time was instead estimated using a discrete-event simulation (built with SimPy) of a single consulting room, with service times calibrated to those observed in the prototype (a mean of 9 minutes and a coefficient of variation of 0.45). Two different arrival patterns were compared across 1000 simulation runs, holding the daily patient volume fixed at 24: an unmanaged walk-in pattern reflecting the observed tendency of 60 percent of patients to arrive within the first hour, and a scheduled-appointment pattern incorporating a 12 percent no-show rate (published reviews report that no-show rates vary widely between settings [17]). The results appear in Table 4.

Table 4: Simulated waiting time, walk-in versus appointment regime

Metric	Walk-in	Appointment	Change
Mean waiting time	74 min	21 min	– 71.6 %
90th percentile wait	142 min	46 min	– 67.6 %
Mean queue length	8.6	2.4	– 72.1 %
Doctor idle time / session	9 min	26 min	+ 189 %

For the same number of patients seen, average waiting time dropped by 71.6 percent, though this came with an increase in doctor idle time from 9 to 26 minutes per session. This represents the underlying trade-off of smoothing patient arrivals: a modest increase in provider idle time is exchanged for a much larger reduction in total patient waiting time, which matters because longer clinic waits are associated with lower patient satisfaction and a poorer perception of the quality of care [18].

5.3 Testing Summary

System validation involved 124 test cases spanning four levels, comprising 92 automated unit and integration tests written with pytest and 32 manually executed system and acceptance procedures. Every one of the 14 deliberately constructed cross-role access attempts was correctly denied, confirming that the role-based and record-ownership checks hold firm at the boundaries where they are most critical, in line with the health data management expectations of the Ayushman Bharat Digital Mission [9].

VI. CONCLUSION AND FUTURE SCOPE

This paper has presented MediQueue, a web-based hospital appointment and queue management system that pairs AI-assisted department routing with appointment booking and a live, measurement-driven queue display. Across a 200-case evaluation, the department-suggestion module attained 88.5 percent top-1 and 96.0 percent top-3 accuracy with a generally benign pattern of errors, while a simulation calibrated against measured service times pointed to a 71.6 percent reduction in mean patient waiting time compared with unmanaged walk-in arrivals. Collectively, these findings support the paper's central argument — that a meaningful portion of outpatient delay stems from information and coordination gaps rather than a genuine shortage of capacity, and that this portion can be reduced without any new hospital hardware.

Planned future work includes conducting a field pilot at a partner hospital to replace the simulated waiting-time figures with real measured data, developing a multilingual interface for symptom entry, extending the model to handle descriptions containing multiple, unrelated complaints, and building a no-show prediction capability, since booking lead time and prior attendance history are among the main determinants of no-shows [17], to enable controlled overbooking once enough attendance history has accumulated.

ACKNOWLEDGMENT

The authors gratefully acknowledge their guide, Dr. (Mrs.) V. A. Kailuke, for her continuous supervision and guidance throughout the course of this work, and thank the Department of Electronics and Telecommunication Engineering, Priyadarshini College of Engineering, Nagpur, for providing the infrastructure and environment that made this work possible.

REFERENCES

- [1] T. Kosalairaman. 2025. Smart Health Consulting and Appointment Booking System with Real-Time Scheduling and Patient-Doctor Communication. Proc. 3rd International Conference on AI and ML Applications (AIMLA).
- [2] K. Kavitha, S. Maragathasundari and K. Karthikeyan. 2025. AI-Driven Healthcare: A Queuing-Based Framework for Optimized Patient Management. Proc. 2025 6th International Conference on Data Intelligence and Cognitive Informatics (ICDICI).
- [3] B. Kumar, M. D. Iqbal, R. Parmer and P. Garg. 2025. The Role of AI in Optimizing Healthcare Appointment Scheduling. Proc. IEEE International Conference on Intelligent Systems.
- [4] X. Liu, R. Lai, C. Wu et al. 2024. Assessing the utility of artificial intelligence throughout the triage outpatients: a prospective randomized controlled clinical study. *Frontiers in Public Health*, 12: 1391906.
- [5] M. Dutta. 2023. Smart Queueing Management System for Digital Healthcare. Proc. International Conference on Recent Advances in Electrical, Electronics and Digital Healthcare Technologies (REEDCON), New Delhi, India, pp. 541-546.
- [6] C. Visvesvaran. 2023. Resource Allocation Algorithm for Web-Based Hospital Appointment Management System. Proc. International Conference on System, Computation, Automation and Networking (ICSCAN).
- [7] K. Singhal, S. Azizi, T. Tu et al. 2023. Large language models encode clinical knowledge. *Nature*, 620: 172-180.
- [8] H. Fraser et al. 2023. Comparison of diagnostic and triage accuracy of Ada Health and WebMD symptom checkers, ChatGPT, and physicians for patients in an emergency department: clinical data analysis study. *JMIR mHealth and uHealth*, 11: e49995.
- [9] Ministry of Health and Family Welfare, Government of India. 2022. Ayushman Bharat Digital Mission: Health Data Management Policy. National Health Authority, New Delhi.
- [10] C. Wallace et al. 2022. The diagnostic and triage accuracy of digital and online symptom checker tools: a systematic review. *npj Digital Medicine*, 5: 118.
- [11] A. Ala and F. Chen. 2022. Appointment scheduling problem in complexity systems of the healthcare services: a comprehensive review. *Journal of Healthcare Engineering*, 2022: 5819813.
- [12] A. M. Noma, K. I. Musa, H. Mamman, A. D. Mato, A. A. Yusuf and M. Sambo. 2022. Design of Intelligent and Secure Hospital Appointment Scheduling System. Proc. 2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON), Lagos, Nigeria, pp. 1-5. doi: 10.1109/NIGERCON54645.2022.9803093.
- [13] S. V. Patil, O. A. Terdalkar and S. B. Patil. 2022. Smart Web Application for Efficient Management of Hospital Appointments. Proc. 2022 IEEE International Conference on Current Development in Engineering and Technology (CCET), Kolhapur, India.
- [14] T. Tsegaye and S. Flowerday. 2020. A Clark-Wilson and ANSI role-based access control model. *Information and Computer Security*, 28(3). doi: 10.1108/ICS-08-2019-0100.
- [15] A. Silberschatz, H. F. Korth and S. Sudarshan. 2019. Database System Concepts, 7th ed. McGraw-Hill Education, New York, NY.

[16] W. Whitt and X. Zhang. 2019. Periodic Little's law. *Operations Research*, 67(1): 267-280.

[17] L. F. Dantas, J. L. Fleck, F. L. Cyrino Oliveira and S. Hamacher. 2018. No-shows in appointment scheduling - a systematic literature review. *Health Policy*, 122(4): 412-421.

[18] M. R. M. Habibi, F. M. Abadi, H. Tabesh, H. Vakili-Arki, A. Abu-Hanna and S. Eslami. 2018. Evaluation of patient satisfaction of the status of appointment scheduling systems in outpatient clinics: identifying patients' needs. *Journal of Advanced Pharmaceutical Technology & Research*, 9(2): 51.

[19] R. Elmasri and S. B. Navathe. 2016. *Fundamentals of Database Systems*, 7th ed. Pearson, Hoboken, NJ.

[20] U. N. Bhat. 2015. *An Introduction to Queueing Theory: Modeling and Analysis in Applications*, 2nd ed. Birkhauser, Boston, MA.