SMART HOSPITAL APPOINTMENT SYSTEM BASED ON WEB TECHNOLOGY

QU KAI, Wandeep Kaur A/P Ratan Singh

Faculty of Information Science & Technology Universiti Kebangsaan Malaysia
43600 Bangi, Selangor

ABSTRACT

Modern healthcare systems face critical challenges in patient flow management and resource allocation, often leading to overcrowding, extended waiting times, and diminished patient satisfaction. (Francetic et al., 2024) Traditional queueing and appointment methods are frequently inadequate, creating significant inefficiencies and delays in care delivery. To address these issues, this project delivered an innovative, web-based solution: the Smart Hospital Appointment System, built on a modular, three-tier architecture that serves three distinct user groups: Patients, Staff, and Administrators. A central innovation is its real-time digital queue management system, which provides patients with live tracking and wait time estimations while empowering medical staff to dynamically manage the patient flow. The appointment booking process is significantly enhanced through an intuitive, HTMX-driven interface that simplifies navigation of a hierarchical department structure and clearly visualizes doctor availability. Furthermore, an AI-powered triage assistant guides users to the appropriate specialty via natural language conversation, effectively reducing booking errors and user uncertainty. By integrating these features, the system is designed to optimize medical resource allocation, improve navigational clarity, and streamline the entire appointment lifecycle from initial booking to real-time queuing. This project delivers a robust and scalable model for improving both operational efficiency and the user experience in healthcare delivery.

1. INTRODUCTION

Modern healthcare systems are increasingly challenged by issues related to patient flow management and equitable resource allocation. Despite advancements, many hospitals still grapple with inefficiencies stemming from traditional queuing methods, leading to prolonged waiting times, overcrowded facilities, and a general decline in patient satisfaction. Existing digital appointment solutions, while aiming to alleviate some of these pressures, frequently present their own usability hurdles, such as cumbersome department selection processes, a lack of transparent guidance, and insufficient real-time feedback on doctor availability or queue status.(Esan & Elegbeleye, 2024) This persistent gap highlights a pressing need for more intuitive, efficient, and patient-centered systems that can streamline the healthcare journey.

This project introduces a comprehensive, web-based Smart Hospital Appointment System designed to address these multifaceted challenges. By leveraging modern web technologies, the system aims to transform the conventional appointment and queuing experience, making it more accessible, transparent, and responsive for all stakeholders. (Chanpen, 2024)The core innovation lies in its ability to offer flexible scheduling, real-time queue updates, and enhanced user interaction, thereby reducing unnecessary waiting, improving resource utilization, and fostering a more satisfying healthcare experience.

The primary objectives guiding this development are as follows:

a) To establish a modular, role-based system architecture: Develop distinct yet interconnected web portals tailored for patients, medical staff, and system administrators, ensuring workflow efficiency and optimized user experience for each group.
b) To engineer an advanced, real-time queue management system: Implement a

dynamic digital queuing system that allows staff to efficiently manage patient flow and provides patients with live tracking, estimated wait times, and automated notifications.

c) To design an intuitive, guided appointment booking interface: Create a multi-step, user-friendly booking process enhanced by smart features, including an AI-powered triage assistant, to simplify department and doctor selection, thereby improving appointment accuracy and patient confidence.

d) To test and validate system-wide functional integrity: Conduct comprehensive functional and non-functional testing across all user portals to ensure the system's quality, reliability, performance underload, and adherence to security and usability standards prior to deployment.

This system is specifically developed for outpatient appointment services and self-service waiting line management within a hospital in Hong Kong. While it facilitates patient bookings with designated or randomly assigned doctors, it does not encompass electronic medical records, online consultations, or electronic prescriptions. Designed for accessibility, the program runs on standard web browsers (Chrome 90+ recommended) with a minimum screen resolution of 720p and supports interaction exclusively in English. The development process adopted the Agile methodology, favoring an iterative approach that emphasized flexibility, continuous refinement, and early delivery of working features to align closely with user needs and project constraints.

2. LITERATURE REVIEW

Effective hospital appointment systems are a cornerstone of modern healthcare delivery, aiming to streamline operations, reduce patient waiting times, and improve overall patient satisfaction (Wan Malissa Wan Mohd Aminuddin & Wan Rosmanira Ismail, 2021). A review of existing literature reveals that while numerous solutions have been developed, they often address specific facets of the problem without providing a comprehensive, integrated experience.

Recent research has explored various technological approaches to enhance these systems. A significant focus has been placed on improving patient accessibility and user-friendliness. For instance, some studies have developed systems with voice commands and chatbots to assist users with physical disabilities (Monrroy & Castañeda, 2024), while others have created efficient web applications with features like self-check-in and online booking to minimize operational costs (Patil et al., 2022). These efforts underscore a growing emphasis on user-centric design in digital health. Concurrently, other researchers have focused on integrating advanced technologies for operational efficiency, using resource allocation algorithms to optimize scheduling and reduce wait times (C et al., 2023), or employing data compression techniques to support telehealth and remote data transfer (Al Siyabi et al., 2023).

Given the sensitive nature of patient data, security and scalability are also paramount concerns. Studies have highlighted the importance of robust security measures, such as role-based access control and AI-driven automation, to protect patient information (Noma et al., 2022). To ensure systems can handle a growing user base, modern, full-stack technologies and microservices architectures have been employed to create scalable and secure platforms (Mahanthi & Jacob, 2023). These works provide a solid foundation for building trustworthy healthcare applications.

However, a critical analysis of these past studies reveals a significant gap: the lack of a truly holistic and integrated system. While individual studies have successfully implemented features like appointment booking, user-friendly interfaces, and security protocols, few have combined these with more advanced, dynamic functionalities. Specifically, core features that directly address the on-site patient experience, such as real-time queue management and automatic appointment recommendations, are largely absent from the reviewed literature. While some systems utilize QR codes for a more efficient check-in process (Custodio-Chavarria et al., 2022), they do not provide patients with live visibility of their position in the queue or an estimated wait time.

Furthermore, none of the reviewed academic studies implemented an AI-driven triage assistant to guide patients to the correct department, a feature that could significantly reduce booking errors and user friction. This analysis indicates that while the building blocks for a better system exist, they have yet to be assembled into a single, cohesive platform that addresses the patient journey from pre-arrival booking to on-site waiting. This project aims to fill that gap by developing a system that not only incorporates established best practices but also introduces these crucial, missing features to create a truly smart and patient-centered appointment and queuing experience.

3. METHODOLOGY

This chapter details the systematic methodology employed to transition the "Smart Hospital Appointment System" from its conceptual phase to a detailed design. A structured approach was essential to ensure that the final product would be robust, scalable, and user centric. By combining a modern development model with comprehensive requirements analysis and a modular architectural design, we established a clear blueprint for implementation. This process ensured that all functional requirements and quality attributes were systematically addressed, laying a solid foundation for the system's success.

3.1 Development Approach

The Agile lifecycle for this project involved a continuous loop of planning, designing, developing, and testing (see Figure 3.1). Each sprint focused on delivering a specific set of prioritized features, which were then reviewed and tested.(mijacobs, 2022) This iterative cycle facilitated a responsive development process, ensuring that the final system was not only technically sound but also precisely aligned with the needs of its end-users.

Figure 3.1: Agile Model Development

3.2 System Requirements Analysis

A thorough requirements analysis was conducted to define the system's objectives, scope, and functionalities, ensuring that the design was grounded in clear and measurable specifications. This process involved identifying the needs of all stakeholder groups and translating them into detailed functional and non-functional requirements.

The functional requirements define what the system must do. To ensure a clear separation of concerns and tailored user experience, the system was designed to serve three primary user roles, each with a distinct set of capabilities:

a) Patients: This user group requires a seamless and intuitive interface for managing their healthcare journey. Key functionalities include registering an account, booking new appointments, viewing and canceling existing appointments, tracking their real-time queue status, and receiving system notifications. The system also supports managing multiple patient profiles under a single user account to accommodate family members.

b) Medical Staff (Doctors & Nurses): This group needs efficient tools to manage their daily schedules and patient interactions. Their core requirements include viewing their appointment schedule, managing the patient queue (e.g., calling the next patient, marking consultations as complete), and accessing essential patient information for upcoming appointments.

c) System Administrators: Administrators require comprehensive oversight and control over the entire platform. Their responsibilities include managing hospital departments and doctor profiles, configuring schedules, overseeing user accounts (both patient and staff), and analyzing system-wide data, such as appointment trends and queue efficiency, to support operational decision-making.

Use Case diagrams were employed throughout this phase to visually model these interactions, providing a clear and unambiguous framework for the system's functional design.

Non-functional requirements define the quality attributes and operational standards the system must meet. These are critical for ensuring a positive user experience, system stability, and data security. The key non-functional requirements were:

a) Reliability: The system must be built using mature and proven technologies to ensure high stability and minimize downtime.
b) Performance Efficiency: The system must be highly responsive. Key user interactions and interface rendering should complete within three seconds, while server-side processing and API responses should not exceed five seconds under normal load.
c) Adaptability: The user interface must be adaptive, automatically optimizing its layout for a seamless experience across various devices and screen resolutions, from desktops to mobile phones.
d) Security: As the system handles sensitive patient data, it must implement

robust security measures, including the encryption of all data in transit, strict role-based access controls, and adherence to data privacy best practices.

3.3 System Architecture and Design

Following the requirements analysis, a detailed system architecture was designed to serve as the blueprint for development. This design prioritizes modularity, scalability, and maintainability, ensuring the system can evolve to meet future demands while remaining robust and efficient.

The system is built upon a Three-Tier Browser/Server (B/S) architecture, a modern and widely adopted pattern that separates the application into distinct logical layers: a Presentation Layer, a Business Logic Layer, and a Data Access Layer. This separation of concerns is fundamental to the system's design, as it allows for independent development, testing, and maintenance of each component, thereby reducing complexity and enhancing stability (Figure 3.2).

Presentation Layer: This is the user-facing layer, responsible for rendering the user interface (UI) and capturing user input. It is designed to be lightweight and intuitive, providing distinct portals for patients, staff, and administrators. Business Logic Layer: This layer acts as the core of the system, containing all the rules and processes that govern the application's behavior. It handles tasks such as appointment scheduling validation, queue management, and user authentication. Data Access Layer: This layer is responsible for all interactions with the database, including storing, retrieving, and updating data. It abstracts the database operations from the business logic, ensuring data integrity and consistency.

To implement this modular design within the Flask framework, the application was organized using Blueprints. Each major functional area (e.g., patient portal, admin dashboard) was encapsulated within its own Blueprint, creating a clean, organized, and scalable codebase.

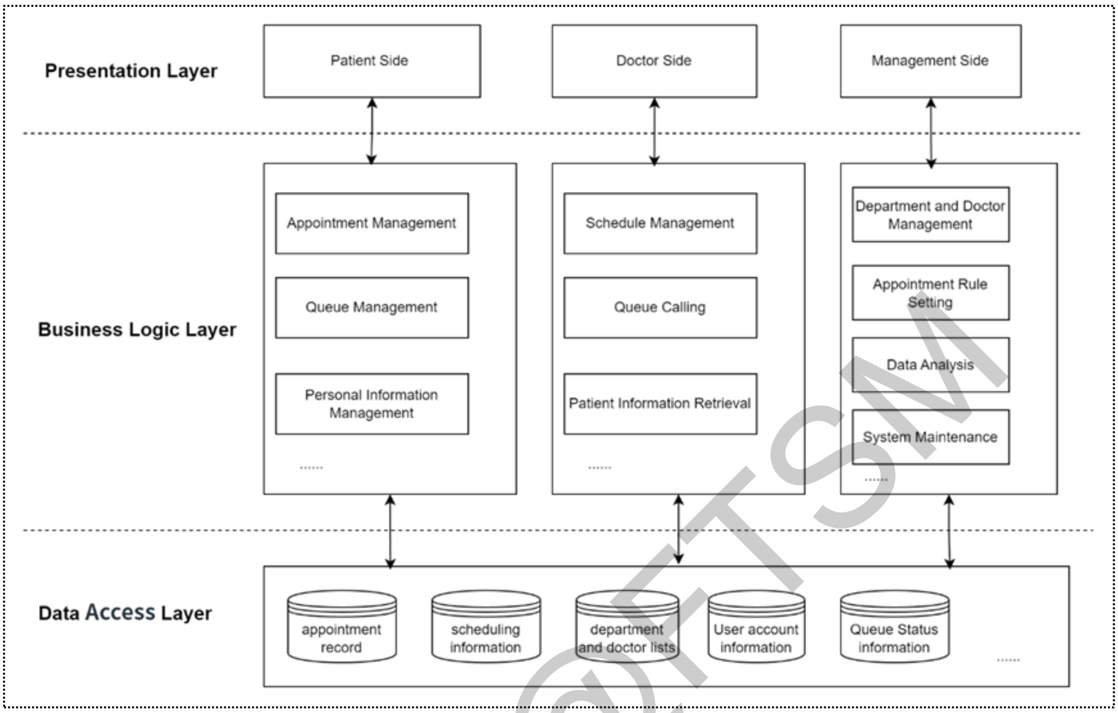

Figure 3.2: System Structure Diagram

A well-structured database is critical to the system's performance and data integrity. The database schema was designed using an Entity-Relationship (E-R) model to define the core entities and the relationships between them. This model provides a clear and logical representation of the system's data structure, ensuring that information is stored efficiently and consistently.

The E-R diagram (Figure 3.3) illustrates the key entities, including User, Patient, Doctor, Department, Appointment, and Queue. Relationships are clearly defined; for example, a doctor is associated with a Department, and an Appointment links a Patient to a specific time Slot with a Doctor. This structured approach ensures referential integrity and provides a robust foundation for all data-driven operations within the system.

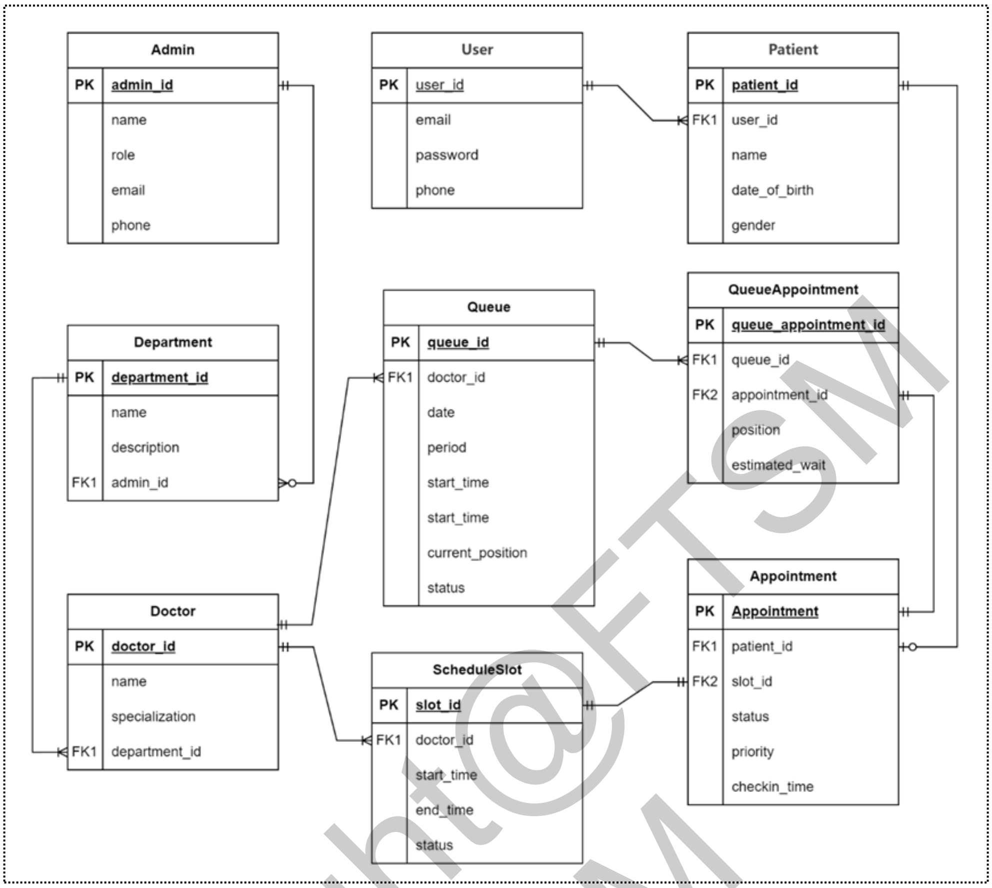

Figure 3.3: System E-R Diagram

3.4 Key Algorithms and Intelligent Design

Beyond a robust architecture, the "Smart Hospital Appointment System" incorporates intelligent algorithms and design patterns to address specific, complex user challenges. These features are what elevate the system from a simple booking platform to a truly smart solution, focusing on guided user experiences and dynamic, real-time data processing.

A primary challenge for patients is determining the correct medical department for their symptoms. To solve this, an AI-Powered Triage Assistant was designed not as a general-purpose chatbot, but as a focused guidance tool. Its logic is built on three core principles to ensure reliability and effectiveness. First, it follows a structured conversational flow, limiting interactions to a maximum of three clarifying questions to guide the user efficiently without causing fatigue. Second, it utilizes dynamic prompt engineering, where the system prompt is continuously updated with the latest list of hospital departments from the database. This ensures the AI's recommendations are always current and strictly limited to available options. The prompt also explicitly defines the AI's role, its constraints (e.g., never to provide medical advice), and the required output format.

Finally, and most critically, the system enforces a structured JSON output. Instead of parsing unpredictable natural language, the AI is compelled to respond in a predefined, machine-readable format. This transforms the Large Language Model from a conversationalist into a reliable system component, whose output can be safely and accurately processed by the backend. The entire workflow, including branches for potential emergencies, is visualized in the flowchart below (Figure 3.4).

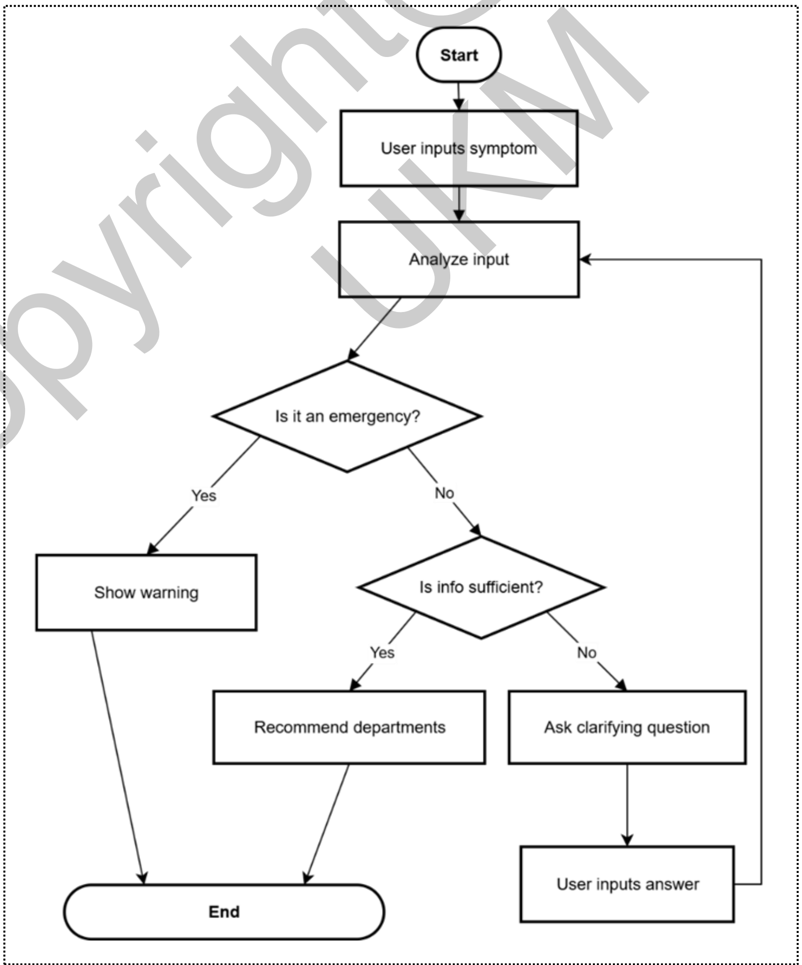

Figure 3.4: AI Triage Assistant Conversation Flowchart

Providing an accurate wait time estimate is notoriously difficult due to the high variability in consultation durations. A simple, static average is often misleading. To address this, the system was designed to use a Sliding Window Averaging algorithm.(Wang & Shao, 2024) This method provides a dynamic and far more accurate estimation by calculating the average consultation duration based only on a moving window of the most recent completed appointments (e.g., the last 10).

As illustrated in Figure 3.5, as each new consultation is completed, its duration is added to the window, and the oldest data point is simultaneously removed. This ensures the average is always reflective of the doctor's current pace and the complexity of recent cases. This method effectively balances stability and responsiveness, smoothing out the impact of unusually short or long appointments while remaining highly adaptive to recent trends. The operational logic, executed after each new consultation, is detailed in the flowchart in Figure 3.6. This algorithm allows the system to provide patients with a meaningful, real-time wait estimation, significantly reducing uncertainty and improving the on-site experience.

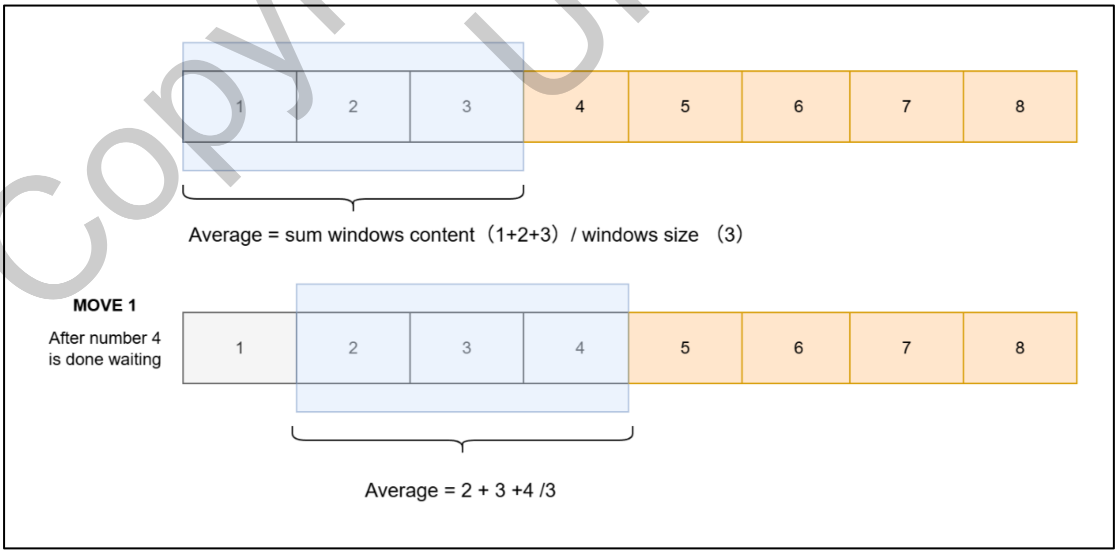

Figure 3.5: Conceptual Diagram of Sliding Window Averaging

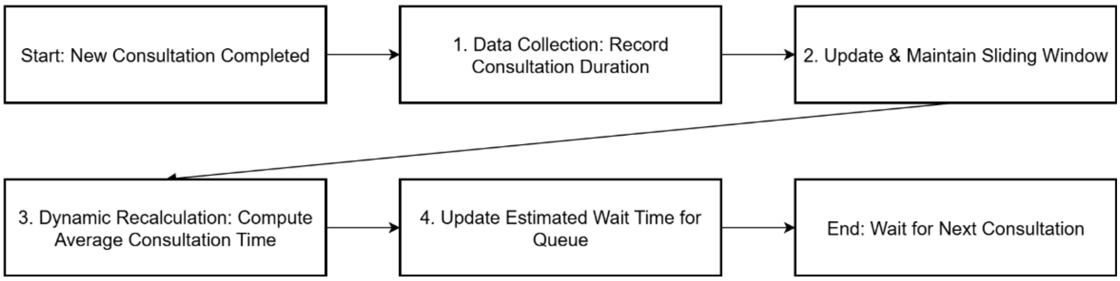

Figure 3.6: Flowchart of Queue Waiting Time Estimation Algorithm

4. SYSTEM IMPLEMENTATION AND TESTING

This chapter details the technical implementation of the "Smart Hospital Appointment System," chronicling its transition from the architectural designs outlined in Chapter 3 into a tangible and functional software product. We will focus on the implementation highlights of core feature modules and present the results of a rigorous, multi-faceted testing phase. This comprehensive evaluation validates the system's correctness, reliability, and readiness for practical deployment, confirming that it successfully meets its specified design objectives.

4.1 Core Feature Implementation

The system's core functionalities were implemented across three distinct user portals—Patient, Staff, and Administrator—each designed to provide a tailored and efficient user experience. The following sections highlight the key features that demonstrate the system's advanced capabilities and user-centric design.

The primary interface for patients is a streamlined, multi-step appointment booking process designed for clarity and ease of use. The system guides the user from selecting a broad medical category to choosing a specific time slot with a doctor, utilizing dynamic content loading via HTMX to create a fluid and responsive experience.

The process begins on the "Find Appointment" page (Figure 4.1), which features a responsive layout that adapts to both desktop and mobile devices. On desktops, a hierarchical list of hospital departments is displayed in a persistent sidebar. Upon selecting a department and date, the main content area is dynamically populated with a list of available doctors, avoiding a full page reload and making the interaction feel seamless.

Once a doctor is chosen, the user is directed to the "Select Time" page (Figure 4.2). This view presents the specific available time slots for that doctor alongside a mini-calendar that allows for quick navigation to check availability on other nearby dates. This focused interface simplifies the final decision-making step, clearly presenting all necessary information for booking. The entire workflow is optimized for mobile devices as well (Figure 4.3), ensuring a consistent and intuitive experience across all platforms as the user moves from department selection to final time confirmation.

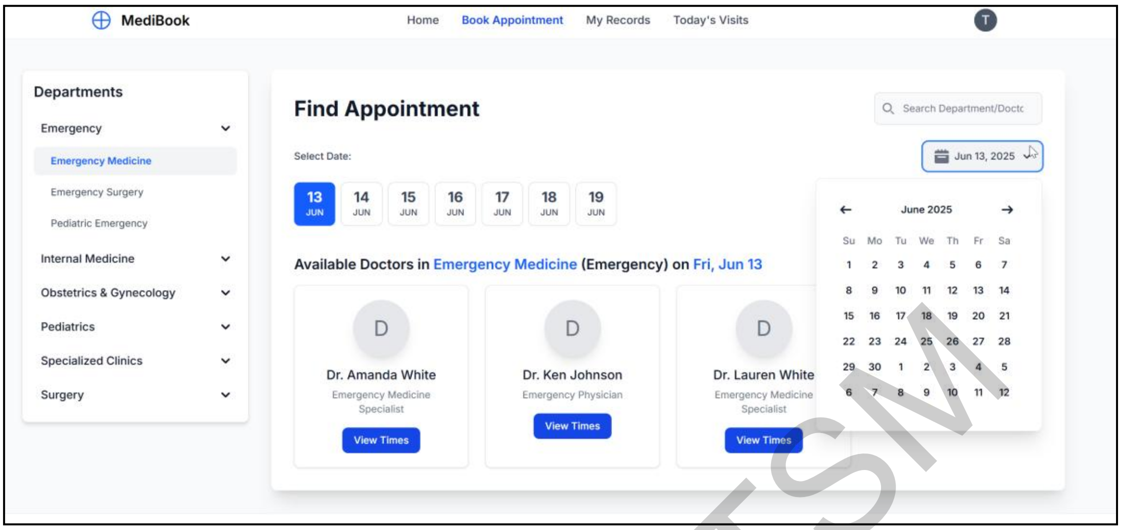

Figure 4.1: "Find Appointment" Page on Desktop

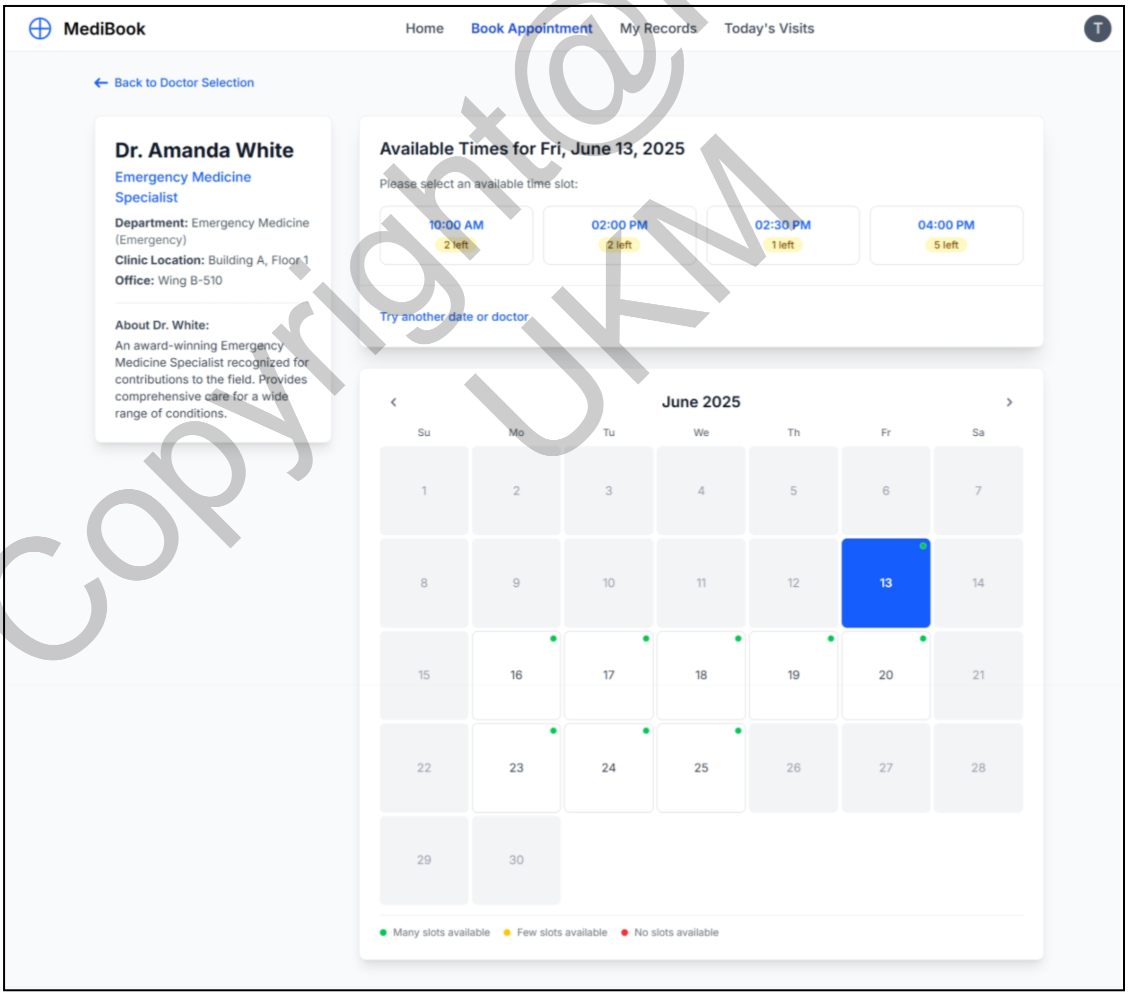

Figure 4.2: "Select Time" Page with Doctor Availability

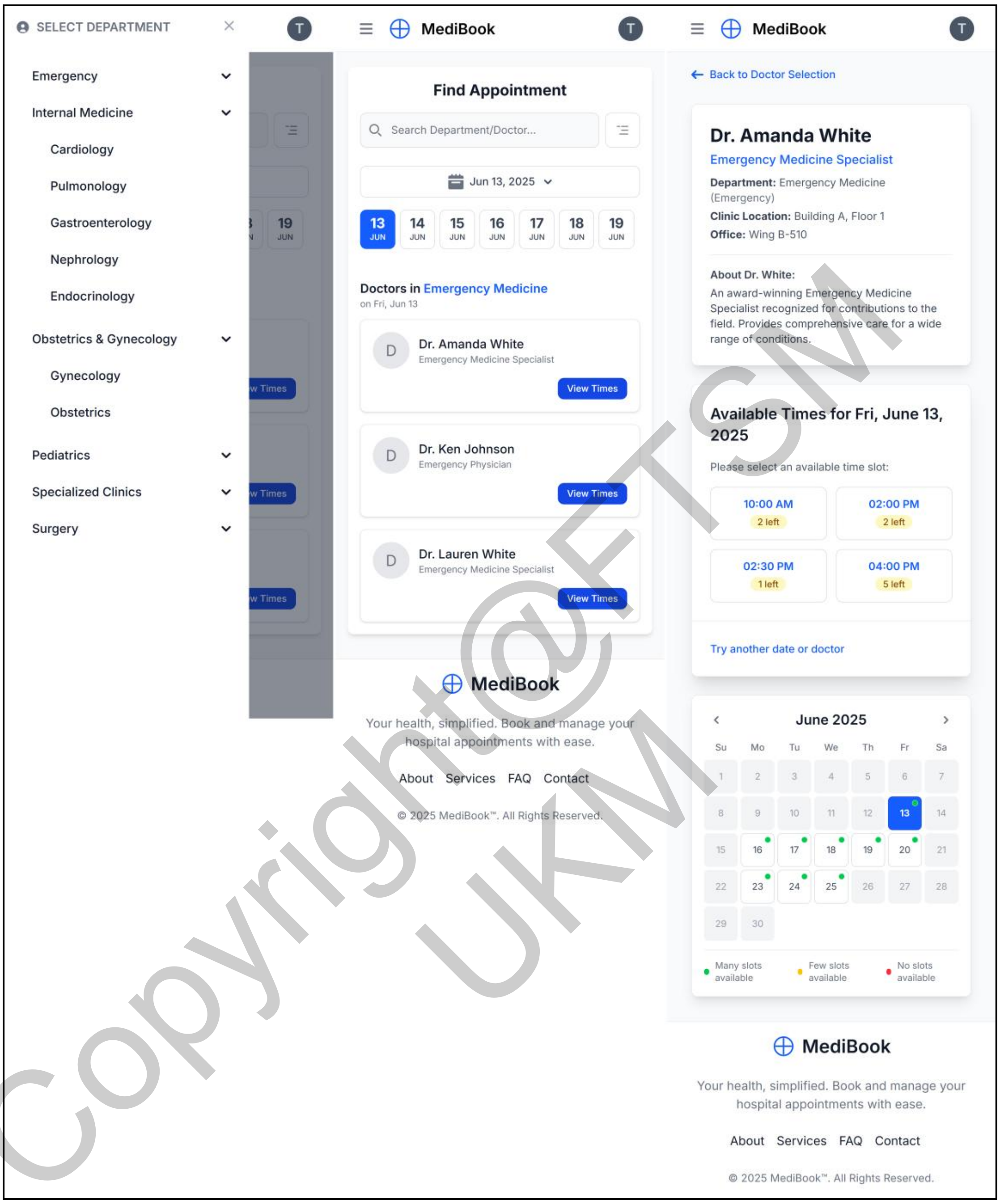

Figure 4.3: Appointment Booking Flow on Mobile

To address the common user challenge of selecting the correct medical specialty, the system features an integrated AI Triage Assistant. As shown in Figure 4.4, this tool provides an interactive, conversational guide that interprets a user's description of their symptoms and suggests the most relevant departments. By leveraging structured prompting and schema-enforced JSON output, the assistant offers reliable, goal-oriented recommendations and provides direct links to the booking page for the suggested departments, greatly simplifying the initial step of the booking process.(Shorten et al., 2024)

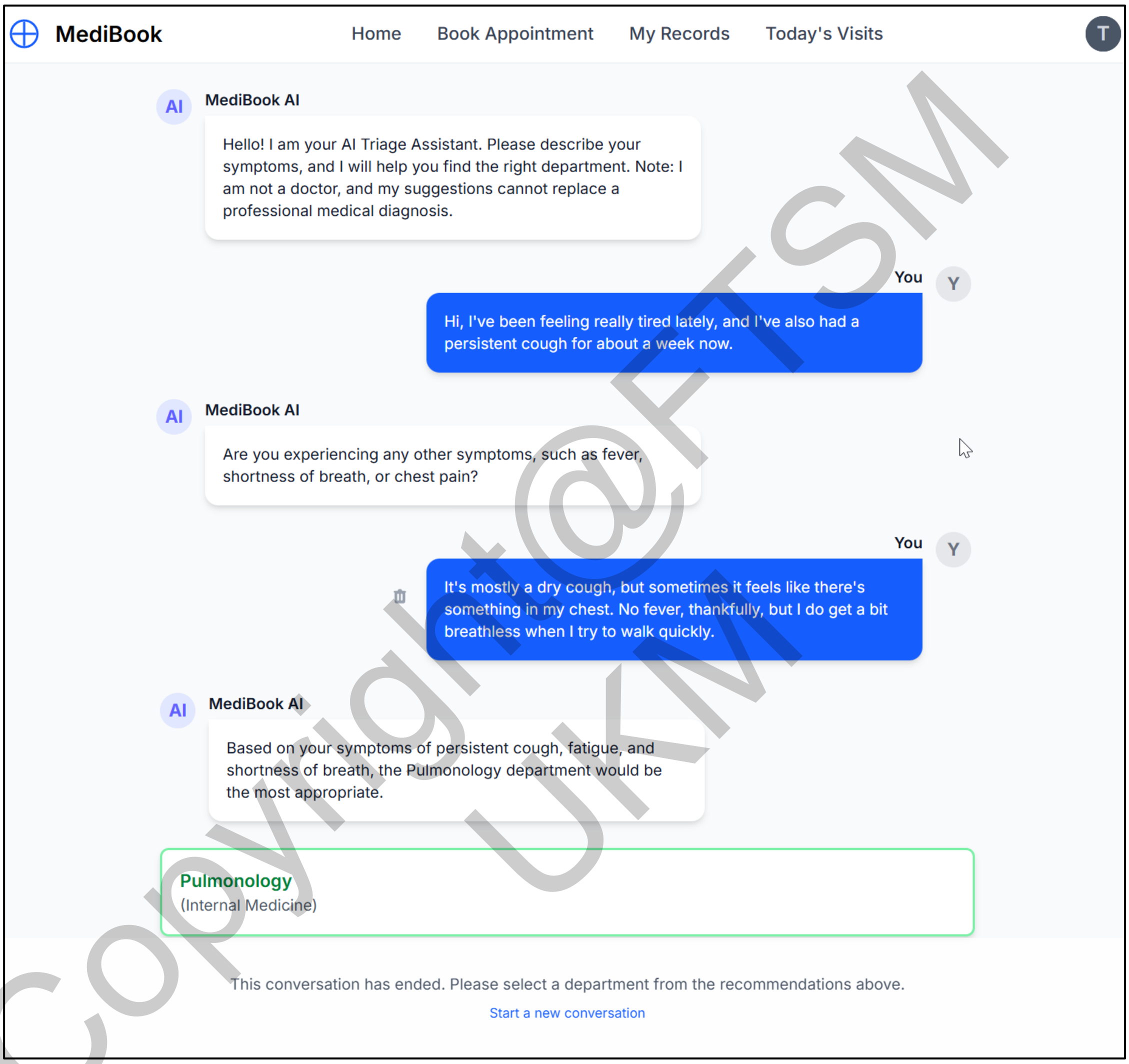

Figure 4.4: AI Triage Assistant Chat Interface

The Staff Portal is centered around an efficient and intuitive interface for managing the daily patient queue. The queue management dashboard (Figure 4.5) provides a real-time overview of waiting, called, and serving patients. This interface is built using an event-driven pattern with HTMX, allowing staff to perform actions—such as calling the next patient or marking a consultation as complete—and see immediate UI updates without requiring full page reloads. When a staff member performs an action, a request is sent to the backend, which processes it and returns a response with a special HX-Trigger header. This header signals a custom event that prompts relevant components on the page to automatically refresh their content, ensuring the entire UI remains synchronized in real-time.

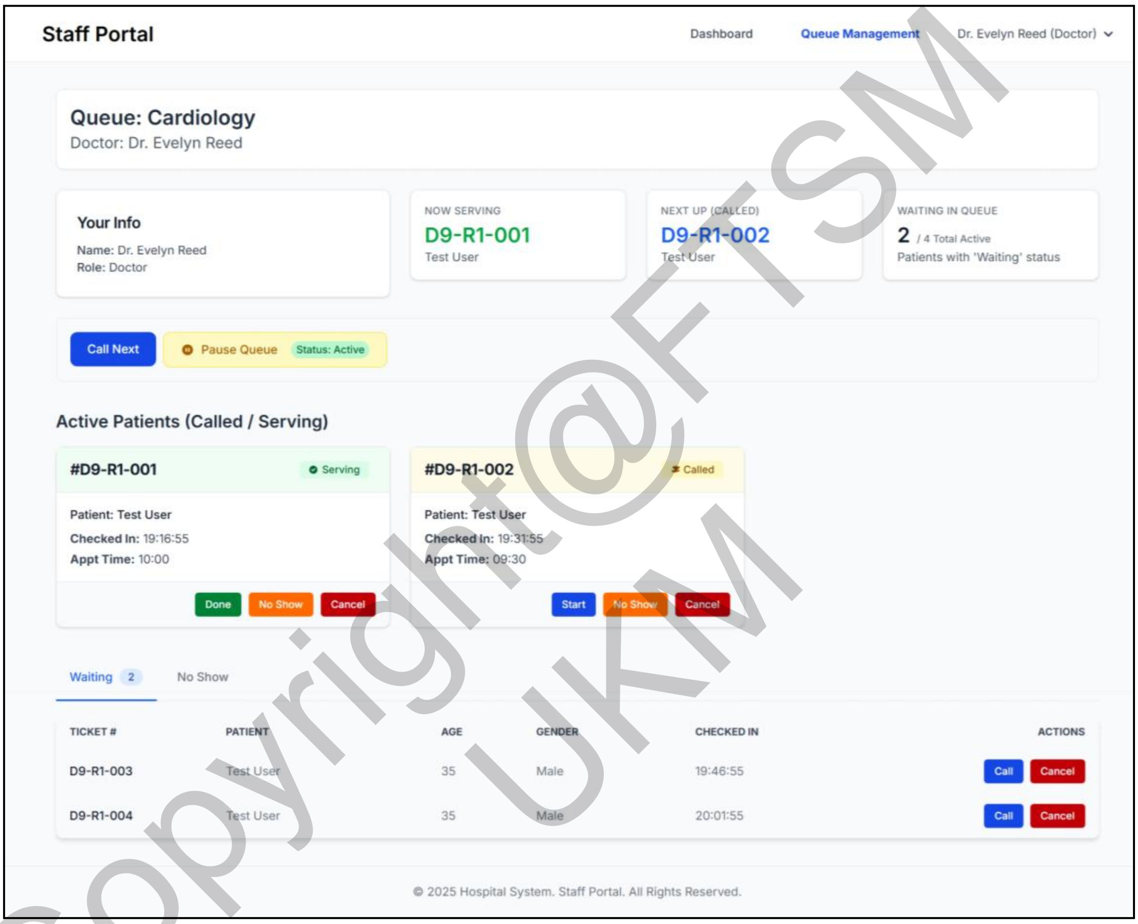

Figure 4.5: Staff Queue Management Dashboard

For administrators, the system provides a powerful and highly interactive appointment management dashboard. A primary challenge was to present a vast amount of appointment data in a clear and filterable manner. The solution is a dashboard featuring a cascading filter system (Figure 4.6). An administrator can progressively narrow down the data they wish to see by first selecting a department, which then dynamically populates a list of doctors belonging only to that department. Selecting a doctor or any other filter then triggers a final update to the main appointment table and a corresponding data visualization chart. This multi-stage filtering, powered by HTMX, provides a fast and intuitive way for administrators to drill down into specific data without overwhelming them with information.

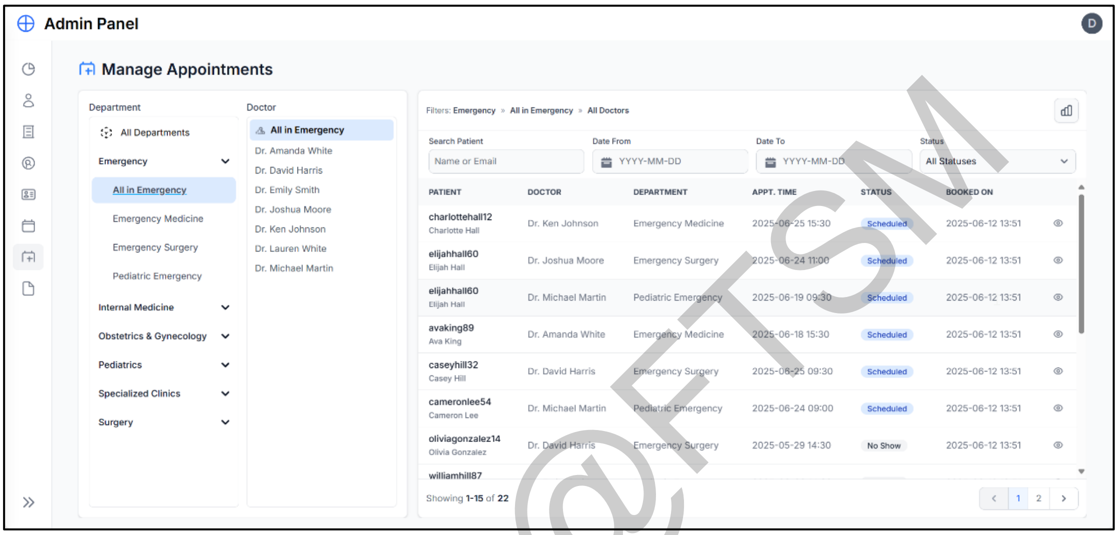

Figure 4.6: Admin Appointment List with Cascading Filters

The administrator's schedule management interface allows for comprehensive oversight of every doctor's working hours. The implementation combines server-side rendering for the initial layout with HTMX and client-side JavaScript for dynamic updates. As shown in Figure 4.7, the main view features a large calendar panel alongside a "Daily Preview" panel. Clicking a day on the main calendar dynamically loads that day's specific time slots into the preview panel via an asynchronous HTMX request. This hybrid approach avoids full page reloads, creating a responsive and seamless user experience for managing complex schedules on a granular, day-by-day basis.

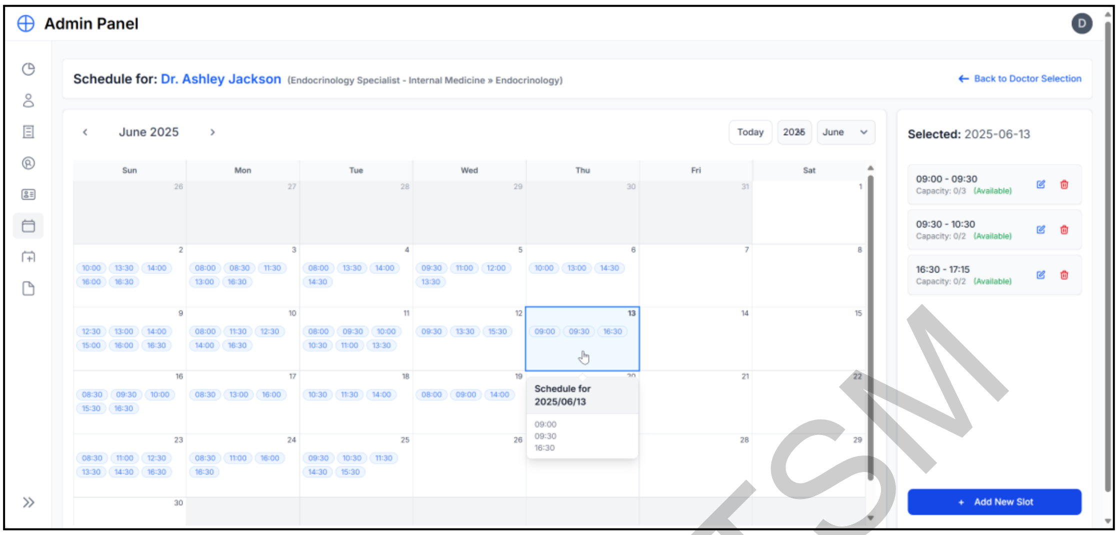

Figure 4.7: Admin Interface for Managing Doctor Schedules

4.2 System Testing and Evaluation

Following the implementation of all core features, the "Smart Hospital Appointment System" underwent a rigorous and multi-faceted testing process to validate its quality, reliability, and alignment with all specified requirements. The testing strategy was designed to be comprehensive, covering functional correctness, performance under load, security, and overall usability to ensure the software's readiness for deployment.

Functional testing was conducted to verify that the system's features operate as specified in the requirements analysis. End-to-end test scenarios were executed for each user role—Patient, Staff, and Administrator—simulating real-world user journeys. The results confirmed that all core functionalities, from patient registration and appointment booking to queue management and administrative oversight, performed correctly. All executed test cases were consistent with their expected outcomes, leading to a "Pass" status and demonstrating the system's functional integrity. A summary of key test cases is presented in Table 4.1.

Table 4.1: Summary of Key Functional Test Cases

ID	Portal	Test Scenario	Expected Result	Status
T-U-04	Patient	Confirm Appointment Booking	A new appointment is successfully created, and the corresponding time slot's booked_count is incremented.	Pass
T-U-07	Patient	AI Emergency Detection	The AI correctly identifies an emergency symptom and advises the user to seek immediate medical help, terminating the triage.	Pass
T-S-02	Staff	Call Next Patient	The first patient in the "Waiting" list is moved to the "Called" state, and the UI updates in real-time to reflect the change.	Pass
T-A-03	Admin	Create Doctor with Staff Account	A new Doctor record and a linked Staff Account record are created simultaneously with a single form submission.	Pass
T-A-05	Admin	Prevent Schedule Time Slot Conflict	The system correctly prevents the creation of a new time slot that overlaps with an existing one for the same doctor.	Pass

Performance and load testing was conducted to measure the system's responsiveness, stability, and throughput under simulated concurrent user traffic. Using the Locust framework, tests were run for five-minute durations with a progressively increasing number of virtual users.

The patient portal, as the most high-traffic component, was tested with 2,000 concurrent users. The results, illustrated in Figure 4.8, were highly positive. The system sustained an average of over 480 Requests Per Second (RPS) with a near-zero failure rate. Response times remained consistently low, well under the 500ms target, even at peak load. The staff and admin portals, tested with 200 and 20 concurrent users respectively, also demonstrated excellent stability and acceptable response times under their simulated loads (Figure 4.9). These results conclusively demonstrate that the system is well-architected to handle its anticipated operational load efficiently.

Type	Name	# reqs	# fails	Avg	Min	Max	Med	req/s	failures/s
POST	/login [LOGIN]	2000	0(0.00%)	185	45	850	160	10.50	0.00
GET	/home [Patient]	18530	0(0.00%)	210	55	1230	190	97.53	0.00
GET	/queue/_get_detailed_status_partial [HTMX]	55821	2(0.00%)	95	25	650	80	293.79	0.01
GET	/records?s_page=[page]	7890	0(0.00%)	350	110	1560	320	41.53	0.00
GET	/appointment/find	4105	0(0.00%)	155	60	980	140	21.61	0.00
GET	/appointment/_get_doctors_list?dept_id=[id]&date=[date]	4010	1(0.02%)	280	90	1340	250	21.11	0.01
	Aggregated	92356	3(0.00%)	158	25	1560	130	486.07	0.0

Figure 4.8: Patient Portal Load Test Results (2,000 Concurrent Users)

Type	Name	# reqs	# fails	Avg	Min	Max	Med	req/s	failures/s
POST	/staff/auth/login [LOGIN]	200	0(0.00%)	160	50	680	145	1.85	0.00
GET	/staff/queue-mgmt/view?dept_id=[id]	1830	0(0.00%)	255	80	1150	230	16.94	0.00
GET	/staff/queue-mgmt/partial/stats [HTMX]	3650	0(0.00%)	120	40	720	110	33.80	0.00
POST	/staff/queue-mgmt/call-next [ACTION]	615	0(0.00%)	280	95	1450	250	5.69	0.00
POST	/staff/queue-mgmt/complete_specific [ACTION]	605	0(0.00%)	310	105	1680	280	5.60	0.00
	Aggregated	6800	0(0.00%)	198	40	1680	180	62.96	0.00

Type	Name	# reqs	# fails	Avg	Min	Max	Med	req/s	failures/s
POST	/admin/auth/login	20	0(0.00%)	190	65	550	170	0.55	0.00
GET	/admin/dashboard/main	158	0(0.00%)	480	150	1800	450	4.39	0.00
GET	/admin/stats/main-chart?period=[days]	155	0(0.00%)	850	350	2500	780	4.31	0.00
GET	/admin/stats/today-summary	156	0(0.00%)	250	80	950	220	4.33	0.00
GET	/admin/appointments/	45	0(0.00%)	550	210	1900	510	1.25	0.00
GET	/admin/doctors/	42	0(0.00%)	320	120	1100	290	1.17	0.00
	Aggregated	576	0(0.00%)	495	65	2500	460	16.00	0.00

Figure 4.9: Staff & Admin Portals Load Test Results

Security testing focused on the robustness of the application's role-based access control (RBAC) mechanisms. Manual penetration tests confirmed that the system effectively segregates the three main portals, preventing unauthorized cross-module access. All attempts by an authenticated user of one role to directly access URLs intended for another role failed as expected, demonstrating that sensitive data and administrative functions are adequately protected.

To quantitatively assess the user experience, a usability survey was administered to a group of test participants. The survey employed a 5-point Likert scale to measure user sentiment across key areas. As shown in Figure 4.10, feedback was overwhelmingly positive. A significant majority of participants "Agreed" or "Strongly Agreed" that the system was easy to use, that navigation was straightforward, and that the information presented was clear. The core appointment booking and queue management workflows also received very high ratings, confirming that the system successfully achieved its usability and user-friendly design goals.

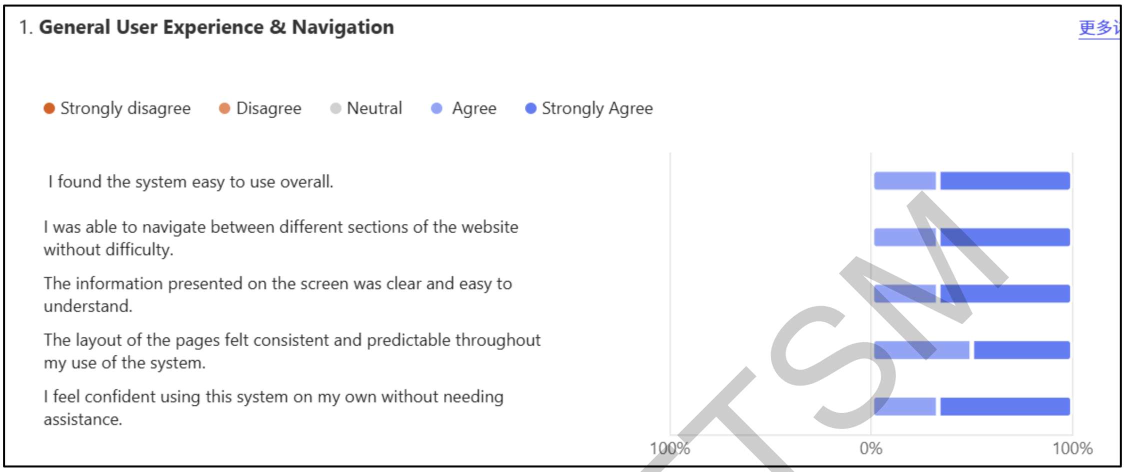

Figure 4.10: Usability Survey Results for General User Experience & Navigation

5. CONCLUSION AND FUTURE WORK

This project was conceived to address the significant challenges of inefficiency and poor user experience that characterize traditional hospital appointment and patient flow management. It successfully delivered a modern, comprehensive, and modular web-based platform—the "Smart Hospital Appointment System"—designed to streamline these processes for patients, medical staff, and system administrators. The system was developed using an Agile methodology, ensuring flexibility and continuous refinement, and leverages a robust technology stack to provide a dynamic and responsive user experience.

The project successfully met all its primary objectives. A modular, role-based architecture was fully realized through the implementation of three distinct and secure user portals. An advanced, real-time queue management system was engineered, providing a dynamic dashboard for staff and a live tracking interface for patients. The goal to design an intuitive appointment booking interface was achieved through a multi-step, guided process enhanced by an AI Triage Assistant that simplifies department selection. Finally, system-wide functional integrity was confirmed through a rigorous testing protocol, which validated that the system is correct, reliable, and performs efficiently under load.

From a technical standpoint, this work makes several notable contributions. It demonstrates how a traditional server-side framework can be integrated with modern libraries like HTMX to create a highly interactive user experience without the complexity of a full-fledged SPA. Furthermore, the integration of a Large Language Model for AI Triage, constrained by structured JSON output, showcases a reliable method for incorporating AI into goal-oriented system workflows. Academically, this project addresses a significant gap identified in the literature by successfully integrating real-time queue management and advanced, AI-guided search functionalities, presenting a more complete model for future research in digital healthcare systems.

While the project achieved its core goals, it is important to acknowledge its current limitations, which in turn inform a clear roadmap for future work. The most impactful enhancement would be the integration with existing hospital systems, such as Electronic Health Record (EHR) platforms, using industry standards like HL7 FHIR to provide a more comprehensive clinical context. (Heryawan et al., 2025) The current notification system is also basic; implementing automated email or SMS reminders for appointments would significantly improve patient communication. To support a production-grade, multi-server deployment, the current lightweight background task handler could be replaced with a more robust, distributed task queue like Celery or Redis Queue. Finally, developing native mobile applications for iOS and Android and integrating a secure telemedicine module would greatly expand the system's accessibility and functionality, aligning it with the future trajectory of digital healthcare.

In conclusion, the "Smart Hospital Appointment System" project successfully translated a set of complex requirements into a functional, validated, and user-centric software solution. It stands as a robust and scalable foundation, demonstrating the potential of well-designed web technology to significantly improve the healthcare experience for all stakeholders.

REFERENCES

Al Siyabi, R. N., Al Hattali, M. R., Al Kharusi, B. A., Al Abri, J. R., Vikraman, B. P., Mahadevan, V., & Rakesh, B. (2023). Hospital Appointment Management System for Telemedicine System Reinforced with Diagnostic Data Transfer. *2023 Third International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, 1–6. https://doi.org/10.1109/ICAECT57570.2023.10117887

C, V., R, S. Y., T, S., & K, S. (2023). Resource Allocation Algorithm for Web Based Hospital Appointment Management System. *2023 International Conference on System, Computation, Automation and Networking (ICSCAN)*, 1–6. https://doi.org/10.1109/ICSCAN58655.2023.10395877

Chanpen, S. (2024). Comparative study of waiting time for emergency operative case between online queue and conventional queue at Ratchaburi hospital. *Journal of the Thai Medical Informatics Association*, *10*(1), Article 1.

Custodio-Chavarria, G., Soldan-Araujo, R. P., & Burga-Durango, D. (2022). System to Optimize the Process of Medical Consultations Using QR Codes in the Hospitals of Peru. *2022 Congreso Internacional de Innovación y Tendencias En Ingeniería (CONIITI)*, 1–4. https://doi.org/10.1109/CONIITI57704.2022.9953670

Esan, O. A., & Elegbeleye, Femi. A. (2024). Effective Approach to Reduce Patient Long Queues in Hospital Emergency Department. *2024 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*, 1–8. https://doi.org/10.1109/icABCD62167.2024.10645273

Francetic, I., Meacock, R., Siciliani, L., & Sutton, M. (2024). Disorderly queues: How does unexpected demand affect queue prioritisation in emergency care? *Health, Econometrics and Data Group (HEDG) Working Papers*, Article 24/14. https://ideas.repec.org//p/yor/hectdg/24-14.html

Heryawan, L., Mori, Y., Yamamoto, G., Kume, N., Lazuardi, L., Fuad, A., & Kuroda, T. (2025). Fast Healthcare Interoperability Resources (FHIR)–Based Interoperability Design in Indonesia: Content Analysis of Developer Hub's Social Networking Service. *JMIR Formative Research*, *9*(1), e51270. https://doi.org/10.2196/51270

Mahanthi, A. N., & Jacob, T. P. (2023). Transforming Traditional Medical Appointment Booking with Advanced Technologies and User-Centric Design. *2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN)*, 1656–1660. https://doi.org/10.1109/ICPCSN58827.2023.00277

mijacobs. (2022, November 28). *What is Agile? - Azure DevOps*. https://learn.microsoft.com/en-us/devops/plan/what-is-agile

Monrroy, G. P., & Castañeda, P. S. (2024). System to optimize the process of booking medical appointments for people with upper extremity disabilities. *2024 7th International Conference on Information and Computer Technologies (ICICT)*, 539–543. https://doi.org/10.1109/ICICT62343.2024.00094

Noma, A. M., Ibrahim Musa, K., Mamman, H., Mato, A. D., Abdulwadud Yusuf, A., & Sambo, M. A. (2022). Design of Intelligent and Secure Hospital Appointment Scheduling System. *2022 IEEE Nigeria 4th International Conference on Disruptive Technologies for Sustainable Development (NIGERCON)*, 1–5. https://doi.org/10.1109/NIGERCON54645.2022.9803093

Patil, S. V., Patil, S. B., Terdalkar, O. A., & Yelure, B. S. (2022). Smart Web Application for Efficient Management of Hospital Appointments. *2022 IEEE International Conference on Current Development in Engineering and Technology (CCET)*, 1–5. https://doi.org/10.1109/CCET56606.2022.10080100

Shorten, C., Pierse, C., Smith, T. B., Cardenas, E., Sharma, A., Trengrove, J., & van Luijt, B. (2024). *StructuredRAG: JSON Response Formatting with Large Language Models*. https://doi.org/10.48550/ARXIV.2408.11061

Wan Malissa Wan Mohd Aminuddin & Wan Rosmanira Ismail. (2021). Improving emergency department overcrowding in Malaysian government hospital. *Journal of Quality Measurement and Analysis*, *17*(1), Article 1.

Wang, G., & Shao, Q. (2024). Design of a smart medical service quality evaluation system based on a hybrid multi-criteria decision model. *Scientific Reports*, *14*(1), Article 1. https://doi.org/10.1038/s41598-024-71224-6

Qu Kai(A192201) Dr. Wandeep Kaur A/P Ratan Singh
Faculty of Information Technology & Science
National University of Malaysia