REAL TIME PATIENT SCHEDULING AND ALERTS

Team Zybots

240361V **L.H.H.R.Kumar**

240470E **K.A.R.Nishaya**

240553L **W.B.Y.Ranaweera**

240558G **J.L.E.Rathnapala**

240710R **W.W.N.D.Wickramarathna**

Project Background

Hospitals commnly use fixed appointment schedules that do not adapt to real time delays caused by varying consultation durations, emergencies, or workflow interruptions.. This results in long patient waiting times, overcrowded waiting rooms and inefficient use of medical staff time.. The Just in time healthcare project addresses this problem by introducing a dynamic scheduling and notification system that synchronies patient arrival times with a doctor's real-time availability, reducing unnecessary waiting and imprving the overall healthcare experience..

Project Objectives

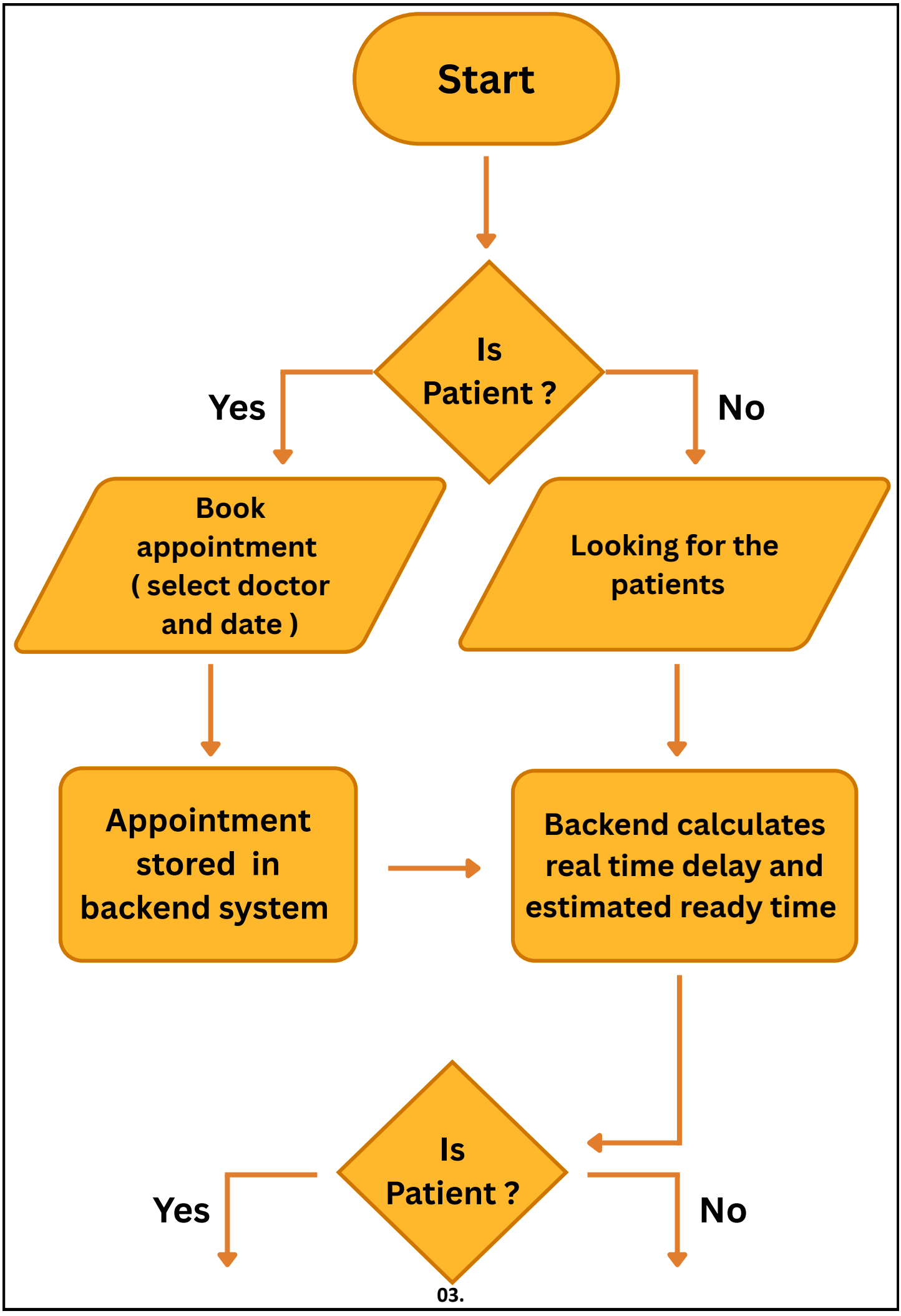
- Minimize patient waiting time in hospitals and clinics.
- Dynamically adjust appointment schedules based on real time doctor progress.
- Notify patients to arrive exactly when the doctor is ready.
- Improve efficiency of doctors and hospital operations.
- Reduce congestion and stress in hospital waiting areas.

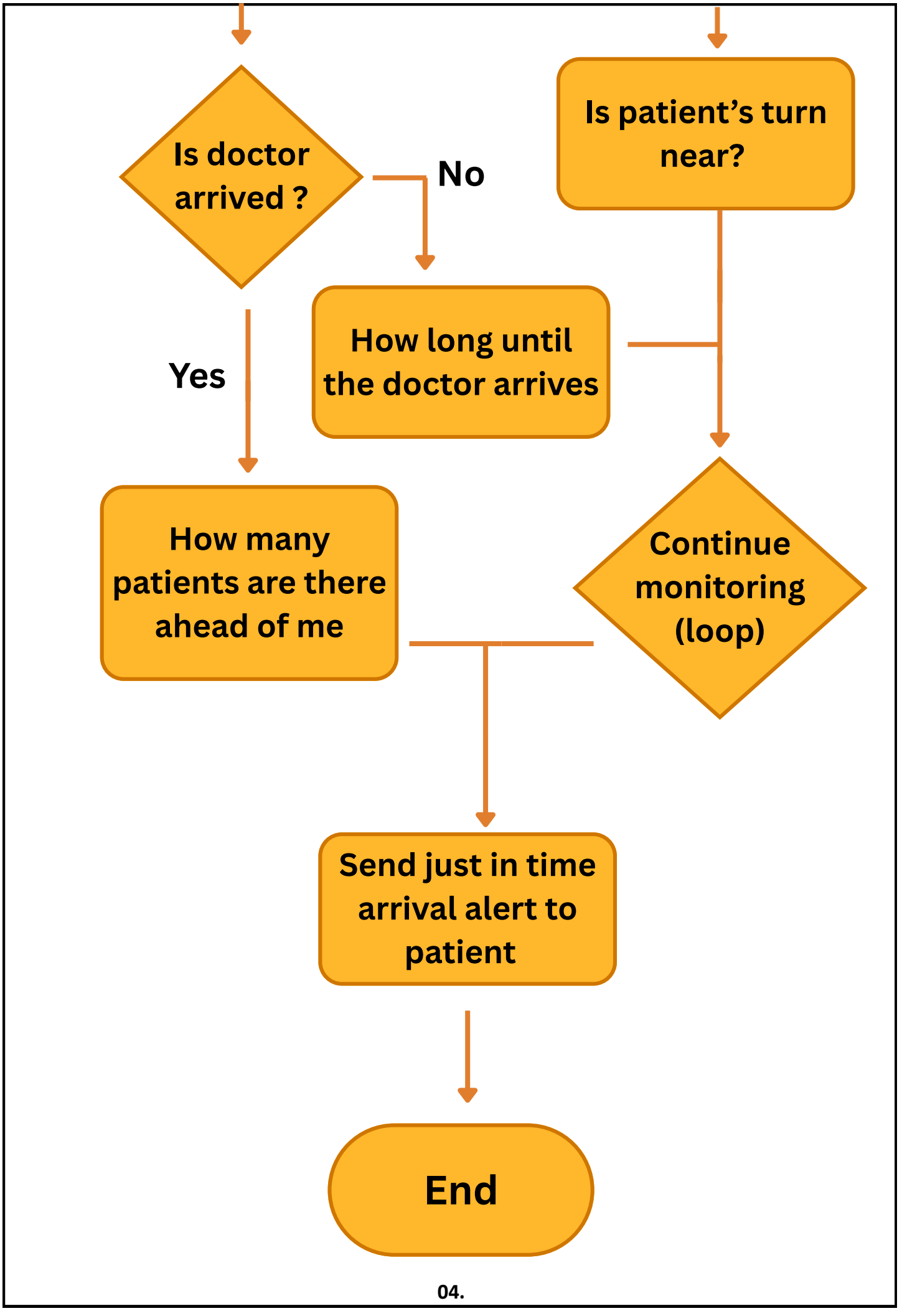
Project Deliverables

- Patient mobile application for appointment booking and real time arrival notifications.
- Doctor interface to update consultation progress and manage schedules.
- Admin dashboard for hospital staff to manage queues and monitor system performance.
- Backend ystem for real-time scheduling logic and notifications.
- Project documentation, including system design and user manuals.

Stakeholders

- Patients : Receive just in time notifications and experience reduced waiting.
- Doctors : Manage consultation flow and schedules more efficiently.
- Hospital Administration : Oversee operations, scheduling and resource allocation.
- Reception and Support Staff : Use the system for queue and appointment management.
- System Developers and Maintainers : Build, deploy and maintain the application.


Start
Is Patient ?
Yes
No
Book appointment (select doctor and date)
Looking for the patients
Appointment stored in backend system
Backend calculates real time delay and estimated ready time
Is Patient ?
Yes
No


Is doctor arrived ?
No
Yes
How long until the doctor arrives
Is patient's turn near?
How many patients are there ahead of me
Continue monitoring (loop)
Send just in time arrival alert to patient
End

Flow Explanation

Appointment booking

The patient uses the mobile app to book an appointment with a doctor.
This information is securely stored in the backend system.

Doctor consultation tracking

As the doctor consults patients they update their current consultation status (started, completed, delayed etc...) using a doctor interface.

Real time processing

The backend continuously

- Monitors doctor progress..
- Calculates delays..
- Predicts when the next patient should arrive..

This replaces fixed appointment times with dynamic estimation...

Just in time decision

The system checks:
“Is the patient’s turn aproaching soon?”

- No → Keep monitoring doctor progress..
- Yes → Send notification..

Patient notification

The patient receives a Just in time alert like:
“Please arrive at the hospital in 15 minute.”

Arival and consultation

The patient reaches the hospital just before the doctor is ready minimizing waiting time and congestion.

Functional Requirements

Patient functional requirements

- Ability to register and log in to the mobile app.
- View available doctors and their consultation schedules.
- Book, reschedule or cancel appointments.
- Receive Just in time arrival notifications based on doctor progress.
- View estimated waiting time and doctor availability.
- Provide feedback on consultation experience.

Doctor functional requirements

- Login to the doctor interface.
- Update consultation status (started, in-progress, completed, delayed).
- View queue of upcoming patients.
- Adjust consultation duration or mark emergency delays.
- Access daily schedule and upcoming appointments.

Hospital admin functional requirements

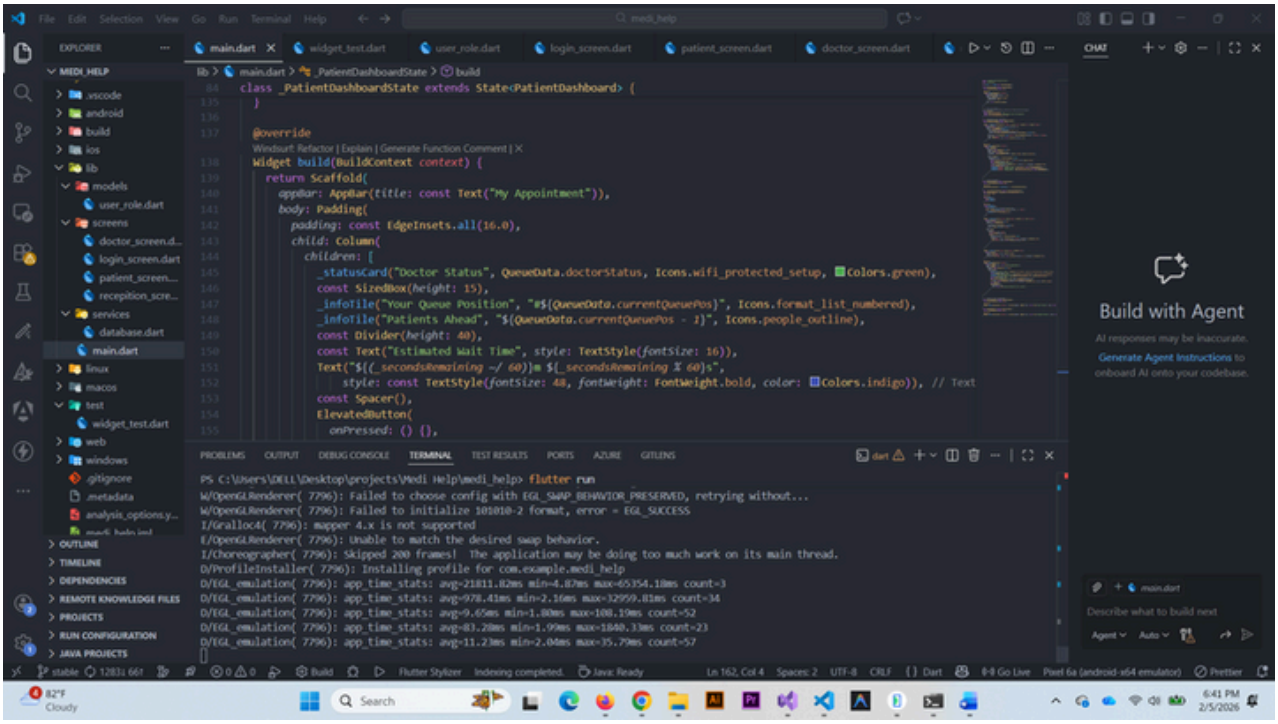
- Manage doctors schedules and patient appointments.
- Monitor real-time patient queue and doctor availability.
- Generate reports on average waiting times and patient flow.
- Update hospital information and resource availability.
- Manage notification rules and system settings.

Backend system functional requirements

- Store and retrieve appointments and patient data securely.
- Track doctor consultation progress in real-time.
- Calculate dynamic estimated arrival times for patients.
- Trigger push notifications or sms alerts for patients.
- Ensure data synchronization between mobile app doctor interface and admin dashboard.
- Maintain audit logs for appointments and notifications.

Advanced functional requirements (future enhancements)

- Predict consultation duration using ML/AI based on past data.
- Optimize scheduling to reduce idle time for doctors.
- Provide analytics dashboard for hospital management.
- Integrate with IoT devices (e.g... smart check-ins, room sensors).



- Flutter is used to develop the patient, doctor and admin interfaces using a single codebase.

- Firebase is used for backend services including authentication, database management, real time updates and notifications.

- The ESP32 is installed near the doctor’s consultation room and acts as a smart status controller...

- The ESP32 cam provides visual confirmation of consultation room status..

- This ensures that notifications are triggered only when the doctor is actually ready...

NonFunctional Requirements

Essential

Usability and User friendliness
Mobile app and interfaces must be easy to navigate for patients, doctors, and admin staff.

Reliability and Dependability
The system must work correctly and provide notifications consistently.

Availability
System must be operational 24/7 for booking and notifications.

Responsiveness and Timeliness
Just-in-time notifications must be delivered in real-time.

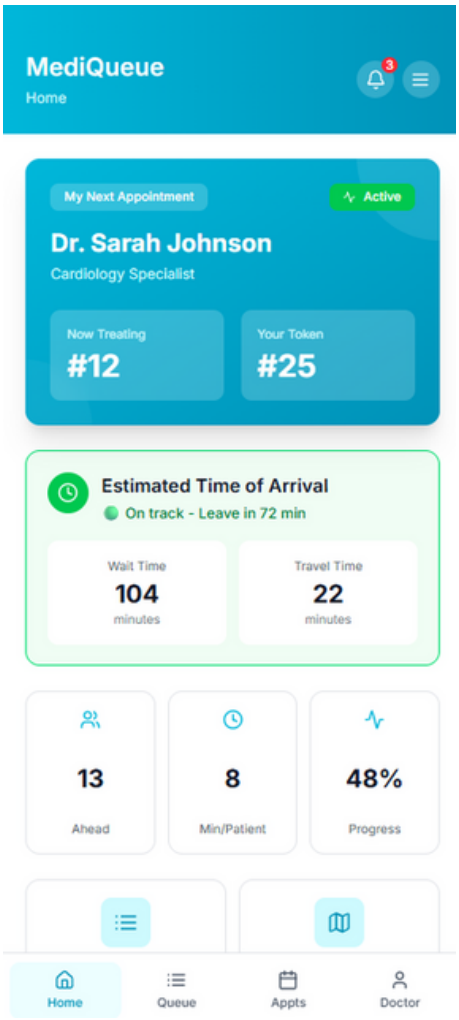
Security and Integrity
Patient and doctor data must be secure and protected from unauthorized access.

Accuracy and Correctness
Estimated consultation times and notifications must be precise.

Maintainability
System should be easy to update, fix bugs, and add new features.

Scalability
System should handle increasing numbers of patients and doctors without performance loss.

Flexibility and Modifiability
Ability to modify scheduling rules, notifications, or UI without major changes.



Inter operability and Compatibility

System should work with different mobile platforms (iOS, Android) and possibly hospital software.

Adaptability and Customizability

Patients or doctors can customize notification preferences or schedule views.

Performance and Efficiency

Backend calculations and notifications should be fast with minimal delays.

Traceability and Verifiability

Logs should allow tracking of notifications and appointments for auditing.

Portability

Mobile app should run on different devices and screen sizes.

Robustness an Fault tolerance

System should handle minor failures (network loss) without crashing.

Clarity and Simplicity

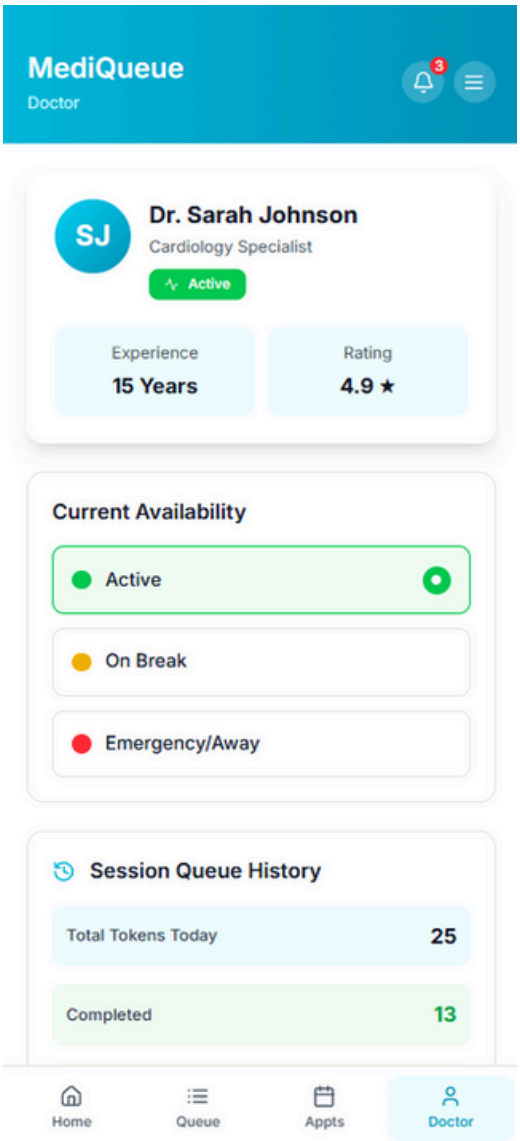
UI should be intuitive and not confusing for users.

Aesthetics and Comfort

Visual design can be improved for user appeal but doesn't affect core functionality.

Autonomy and Self healing

Advanced AI based self-healing or decision making not required in the first version.

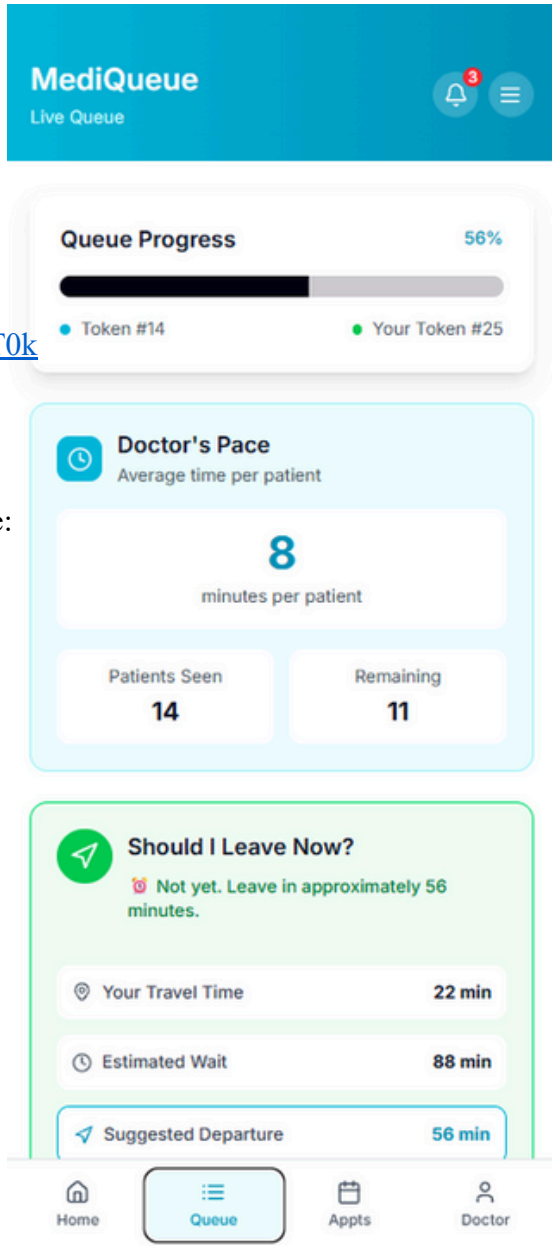


Long-lasting Battery
Optimizations for battery use are secondary since the app runs on user devices.

Survivability
Extreme disaster recovery or offline operation is optional.

Ubiquity
Worldwide availability is not required in initial implementation.

Cost / Time to market
Low priority for NFR classification; project will follow normal development constraints.



Project repository :
https://github.com/lasitheshanJR/lasitheshanJR/

Figma file link :
https://www.figma.com/design/PmtArwRMNZRCEpwT0kQTsH/Medi_Help?node-id=0-1&m=dev&t=evSHxTCbQ63Y92nz-1

We are creating machine lerning models through Kaggle:
https://www.kaggle.com/lasitheshan