MediAxis: Android based Emergency Alert and Hospital Appointment App

Tanisha Chaudhari[x], Tanuja Patkar[:], Samruddhi Rokade[:], Pratiksha Waghmode[:]
Guided by Prof. Padma R. Ahire Department of Computer Engineering
Gokhale Education Society's R. H. Sapat College of Engineering, Nashik

***Abstract*—MediAxis is comprehensive Android app designed to transform hospital management and patient care across multiple cities by joining the gap between patients, doctors and emergency services. Patients can easily trigger instant emergency alerts that share their exact location with responders and emergency contacts for rapid assistance, put their symptoms to receive rec-ommendations for appropriate specialists book appointments at nearby hospitals using precise location services. However, doctors have a dashboard to view and confirm patient appointments. They can update their schedules, review profiles that show their qualifications, years of experience and preferred consultation times. MediAxis helps to reduce waiting times, improves access to care and ensures real-time communication. This makes it a vital tool for healthcare delivery in busy cities. MediAxis is making healthcare better by connecting patients, doctors and emergency services. It is a tool, for delivering good healthcare in urban settings.**

***Index Terms*—Android Application, Emergency alert, location services, appointment booking, multi-city operation**

I. INTRODUCTION

In India, in big cities the healthcare system is changing really fast. MediAxis is trying to solve some problems that hospitals have been facing for a long time. This is causing problems. More people are getting sick. There are not enough doctors. This means people have to wait a time to see a doctor. It is hard for hospitals to manage everything. They often use ways of doing things like writing down appointments or using the phone to book a time. This can cause more problems like people not showing up or doctors being too busy. This can be really bad for people who need help. MediAxis is trying to help hospitals do things better.

MediAxis is an Android application that helps people man-age hospital things in a way. It is made for people to use on their phones. Patients can use it to book appointments in five cities. They can also tell the application what is wrong with them. It will suggest doctors who can help. For example if they have chest pain it will suggest a heart doctor. If they have a migraine it will suggest a brain doctor. In case of emergency the application can also send emergency alert to hospitals or ambulances. It uses Google Maps to find the location. The doctors have their part of the application. They can see when patients have booked appointments and get messages when someone books or cancels. They also have their profiles where patients can see things like how long they have been a doctor, where they went to school and when they are available. This helps everyone know what is going on and makes things easier, for the patients and the doctors. MediAxis is making it simple for people to use their phones to deal with hospital things. It is helping doctors and patients talk to each other better.

This project is a fit for the work that has already been done on hospital management systems. Other studies have looked at things like apps that help doctors and patients coordinate and telemedicine that knows where you are. These things have made some improvements in how well people keep their appointments and how quickly doctors get back to them.. They do not think about some important things like checking symptoms making sure the system works in many cities and making sure doctors and patients can easily talk to each other. MediAxis takes the things from other projects and adds some new things. Like special plans for emergencies in cities and ways to match patients with the right specialists. It also fixes some problems that people have found with using technology in healthcare like the need for systems that are cheap and work on Android phones because a lot of people in cities cannot afford other options and many people use Android phones. Hospital management systems, like MediAxis are very important.

II. LITERATURE SURVEY

The healthcare system is really different now that it uses technology and mobile apps. People can get the help they need a lot faster. The healthcare system can reduce waiting times for people. It is also easier for doctors and patients to talk to each other. The healthcare system is getting better because of technology and mobile apps. The healthcare system is really getting better, with these technologies and mobile apps.

Vishwakarma and his team looked into an Android app that helps users to create a shared workspace. The app has tools that let users work on their project step by step. It also has features that send out alerts when something bad happens. These alerts are only sent out when the situation is really bad and they get help people to respond. The app uses GPS and cellular networks to track where people are. When there is a crisis the app shares updates with people the user has chosen to contact. This shows how tracking locations can be used in apps like when you need to call an ambulance or find a hospital near you. The app can even help you find hospitals to where you are. Vishwakarma and his team found that the Android app is really helpful, during emergency situations. The location tracking feature of the Android app is very useful. [1]

Speeding help during emergencies happens when phones share live alerts along with the person's location. Instead of waiting, medical teams receive details quickly because soft-ware prioritizes urgent cases. This way, hospitals connect more rapidly with those in need through smarter routing. Messages go directly to responders around precise map locations. Design choices in MediAxis borrow heavily from how apps now guide faster using GPS techniques. Decisions occur earlier since signals travel without delay from device to clinic. [2]

When patients move around the people taking care of them can see what is happening away so they always have the latest information about the patient. For example a study took a group of people. Split them into smaller groups randomly. This study showed that when patients use apps that know where they are using GPS they check in often and it is easier for them to get medical help when they need it. Knowing where patients are is very helpful for clinics because it helps them do their job better. This shows that tools like MediAxis are more useful when they can tell people where the patient is in an emergency and that is a good thing, for patient care. [3]

Early hospital tools often lived online, handling appoint-ments plus operations. Yet they struggled to move around easily or show where things were happening right then. Take Practo, Apollo 24-7 works like a digital clinic where you book visits, find physicians, or talk to someone by video. Even though care comes through your phone, picking the right expert means doing your own searches—which gets tricky if you do not understand what those symptoms might mean. Finding help depends heavily on how well you can match signs to fields of medicine. Without clear clues about conditions, navigation becomes guesswork rather than guidance. Most physicians stay out of it. Besides that, help during crises or tracking by place does not show up here either. [4]

III. SYSTEM ARCHITECTURE

Inside the setup there is an Android app that shows users what they need to see. This app talks to a Python Flask engine that is running in the background. The engine manages tasks and logic. It connects to a SQLite storage spot that holds everything safely. There are three levels stacked up. Each one does its job without causing problems. One level feeds into the next smoothly. It's like water flowing down stairs. The separation of levels keeps things clean when updates are made. The movement between layers is efficient, with nothing slowing things down. The system can grow naturally because its parts can stretch apart if needed. The Android app, the Python Flask engine and the SQLite storage all work together. The levels are separate. Each one handles its own tasks. The system feels natural as it grows.

The Presentation Layer is made using Kotlin and Jetpack Compose. This is what users see when they use the app. The Presentation Layer has lots of things like signing up and logging in. Users can also enter symptoms. Get suggestions from experts. They can schedule visits. Look at a doctors workspace. If they need help fast they can send a signal. It uses Retrofit to talk to the server. The app sends requests to the server. Gets data back in JSON. When the data gets back the screen updates

away. It shows what just happened on the server. The Presentation Layer is smooth because it always shows what is really going on. The Kotlin and Jetpack Compose Presentation Layer makes sure everything is up, to date.

Python is what makes the app layer work through Flask. It sets up a server that uses API. When clients send requests they get processed here. The requests go through some set steps before they get to the data storage. The backend is where things like user logins happen. It is also where symptoms get matched to doctors and visits get booked. Emergency alerts have their special paths and they get handled right away. Each thing that the system does is connected to an endpoint like the one for signing up. This endpoint helps guide the actions one step at a time. A special tunnel is made through something called ngrok. This tunnel lets traffic from outside reach the Flask app, which is connected to a specific port. The system does not just skip checks, on the data that is coming in. Instead it looks at the data before it gets to the database. The system only runs queries after it has checked the data and made sure it is okay. Then it sends back pieces of data in a format called JSON to the person who asked for it.

The app has a database that uses SQLite. This database has a tables, one table is for user details and another table is for doctors. The symptoms have their table too. The specialists are also in this database. It has appointment records and emergency logs. The app uses Flask to handle requests. Flask runs SQL commands to create, read, update or remove things from the database. There are rules in the database that help keep the information accurate. These rules also make sure the relationships between things are correct. Each piece of information goes where it should because the fields are set up in a way. This means there are no ends. The tables are connected in a way that makes sense. This is because the people who made the app planned it carefully. When changes are made to the database they happen safely. There are checks in place to prevent errors. This means the data, in the database is reliable. It stays reliable even when it is updated a lot.

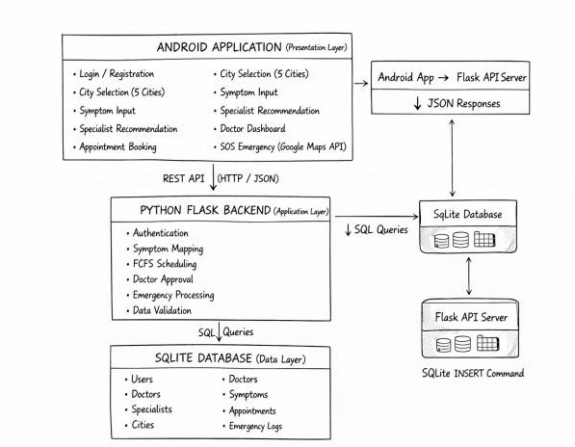

Fig. 1. System Architecture of proposed system

From the beginning information moves one step at a time. When someone enters information in the app for example when signing up the data goes to the server as a POST message using Retrofit. This message is sent to the ngrok link. The message then gets passed to the Flask engine. The engine receives it through a web path that was created to handle it. The server then talks to the SQLite storage. It saves new information or gets existing information before sending a response. On the phone the app reacts immediately. It changes screens or shows alerts based on what the server sent.

IV. METHODOLOGY

A good plan is the thing that needs to be in place for MediAxis to work well. This plan is based on a system where the client and server work together. When we create an app we do it step by step making sure the API paths are simple and the database is controlled properly. It is really important that information moves correctly so each part of the process happens in the order. This means that when people ask for something they get a response and the services they need are available to them, without any problems. MediAxis should be set up in a way that makes it easy for people to get what they need from the services.

The system is made up of three parts: the part that users see(Presentation layer) ,the part that does all the work(Application layer) and the part that stores all the in-formation(Data layer). The part that users see which is made with Kotlin and Jetpack Compose shows users easy to use screens to register log their symptoms, schedule appointments and get emergency alerts. When users put in information it

gets turned into a kind of message that gets sent to the backend services using Retrofit. This helps keep everything organized and makes sure that the part that users see and the part that does all the work do not get mixed up. The system is designed so that each part has its job, like the part that users see is in charge of showing things on the screen and getting input, from users and it makes it easy for information to flow to the backend services, which keeps everything separate and easy to understand.

Messages go to a server that is made using Python Flask. The server takes care of each web request. It does this by using entry points that were set up for this reason. When people try to use the system it first checks who they are. Then it tries to match their symptoms with the doctor. This part is done one step at a time. When people want to book a visit it uses a line-up system. The person who gets there first gets seen first. This keeps everything. If two people try to book the time the system stops this from happening because of the order. Doctors can also respond. They can say yes or no to meetings that people send to them. This is how the system works with messages and doctors and people who need to see a doctor.

Every piece of information- User profiles, doctor records, symptom notes, bookings, crisis entries are saved in a SQLite file. The Flask engine uses scripts to get, update or delete the information that is stored. The system checks everything before it makes any changes so only the right information is used. This way any problems are found early. The system does not break when things are changed behind the scenes.

The emergency SOS module is really important. When you use it, it finds your location using GPS signals. Then it uses Google Maps to find hospitals that're close to you. This step helps you get to a hospital quickly when you really need to. The emergency SOS module is a help, in urgent moments.

The algorithms used for the proposed application are:

A. Emergency SOS Processing Algorithm

A sudden crisis jumps the queue without delay. Priority shifts when danger hits without warning. Urgent needs push aside routine steps instantly.

1) User activates SOS button.
2) System captures real-time GPS location.
3) From time to time, the user's location is recorded in the system behind the scenes. Sometimes the position is passed along to the machine handling requests.
4) Emergency event is logged in the database.
5) Nearby medical centers appear once they are retrieved from the system using the Google Maps API.

B. FCFS Appointment Scheduling Algorithm

The first come, first serve scheduling method. Appointment requests get handled one after another, based on when they arrive, received.

1) Patient submits appointment request.
2) A clock inside marks when it gets the ask. Time stamps lock in place right after entry begins.
3) Folks show up one after another, their requests lined up just like that. First come, first served—nothing jumps ahead. Each person waits their turn, neatly slotted by when they arrived. Time decides the lineup, nobody else.
4) Doctor processes requests sequentially.
5) Once the appointment gets a response, it shows either Confirmed or Rejected. Status changes happen only after confirmation or denial arrives. The system reflects updates once feedback comes through. Either outcome locks in when received. This way works better because it keeps things fair while stopping schedule problems before they start, conflicts.

C. Priority Scheduling for Concurrent Requests

The system works with a way of deciding what to do first. It looks at each person who needs help like someone who wants to make an appointment or tell us about their symptoms or send an emergency alert. The system gives each person a level of importance based on how bad their problem's what they tell us. The people who need help badly like someone who is very sick get help first. The system uses a list, on the backend to make sure these people get help quickly. At the time it still helps people who do not need help as badly but it does so in a way that is fair and does not take too long. This way of doing things helps people who really need help away and it makes the whole process of deciding who to help first work better in the mobile healthcare app.

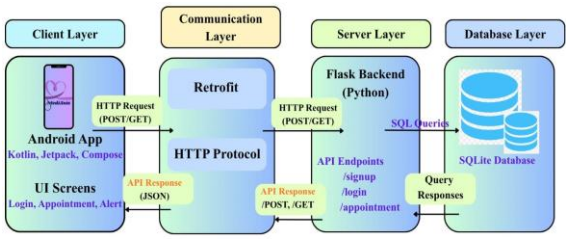

Fig. 2. Implementation of proposed system

V. IMPLEMENTATION

A mobile app built with Kotlin and Jetpack Compose runs on Android devices, handling tasks like signing up users, selecting symptoms, scheduling visits, viewing doctor tools, and triggering emergency alerts. Instead of a traditional setup, it connects through layers—client, server, and storage—where the middle part uses Python Flask to manage requests. On the back end, information is saved into an SQLite database, keeping everything structured yet lightweight. Communication between pieces happens smoothly, each part doing its job without overlap.

Through the Android app, actions are recorded and then sent via Retrofit as HTTP messages packed in JSON. When those reach the server Python Flask takes over managing tasks like signing up users, setting appointments, or triggering urgent warnings. Each arrival triggers checks on the data before any steps follow. What happens next depends entirely on what was requested, handled quietly behind the scenes.

Booking times get handed out based on when people ask, like lining up at a shop door—first there, first served. No one jumps ahead, so each slot goes cleanly to just one person. Inside their private view online, physicians see incoming visits pop up as they arrive. They tap yes or no whenever new ones come in, adjusting as needed without delay. Time fills only when confirmed, keeping gaps between patients clear and set.

Storing things like patient records, physician profiles, symp-tom history, booked visits, and urgent incident reports happens inside an SQLite setup. Operations on that stored info run through standard SQL commands behind the scenes.

When help is needed, the SOS feature grabs the current position through GPS signals. Nearby medical centers pop up on the screen because of live data links to mapping tools. Responses come back after the server handles each alert one at a time. Once processed, fresh details reach the phone app, changing what users see without delay.

A well-built setup keeps data moving smoothly, connects parts of the system without delay, while holding everything together firmly—this shapes a medical tool that grows easily and feels natural to use.

VI. RESULT AND DISCUSSIONS

The system helps patients get access faster. It does this by confirming appointments and handling emergency alerts right away. It cuts down on waiting times and response times. This is done by organizing requests in an order and focusing on urgent cases first not routine ones. The result is that patients get sorted out efficiently. This leads to communication, between patients and healthcare providers. Overall the digital healthcare service gets better at responding and being reliable. The system improves care by making things happen faster and more smoothly. It helps patients get the care they need quickly. The digital healthcare service becomes more efficient and reliable.

The digital healthcare system is really good because it makes it easier for people to get the care they need. It helps doctors and nurses manage the people who need help in a way. The system makes sure that people who really need help away get it but it also helps people who just need regular check ups. The digital healthcare system also helps patients and doctors talk to each other better which means people can get the help they need sooner. This makes the whole healthcare system more reliable. It works better for everyone. The digital healthcare system is a thing because it helps people get the care they need when they need it which is very important, for the digital healthcare system.

VII. FUTURE SCOPE

To make MediAxis better we can add technology that helps it work more efficiently. This means it can do more than the basics. For example MediAxis can use intelligence to really understand symptoms and find the right specialists on its own. If MediAxis is on cloud platforms hospitals can share updates away which is very helpful when many people are working together. Over time MediAxis can connect to health records so clinics can see a

patients entire history in one place. Making changes like these helps MediAxis grow and become more effective in the way it supports care. MediAxis will be better, at helping people. That is what matters.

VIII. CONCLUSION

The MediAxis app has demonstrate a lot of potential to change the way healthcare is delivered digitally. It helps patients get access to care easily and quickly and also helps emergency responders react faster. It helps doctors and nurses prioritize cases while still taking care of routine appointments. This makes it easier for patients and healthcare providers to work together. The system also helps make sure patients get care by keeping track of their medical history and sending alerts in real-time. This is a foundation for adding new features like video consultations, remote health monitoring and connecting with more healthcare providers. Overall MediAxis helps make healthcare services more responsive organized and focused on patients needs whether its a check-up or an emergency. The MediAxis app is really making a difference, in healthcare. MediAxis is helping patients and healthcare providers work together effectively. The MediAxis system is designed to improve care.

REFERENCES

[1]. I. Vishwakarma, A. Joshi, V. Kate, C. Bansal, et al., “SOS Detection App -The Emergency Ally, 24/7,”Journal of Emerging Technologies and Innovative Research (JETIR), vol. 12, no. 5, pp. 368-371, May 2025.

[2]. U. Agomuo, G. E. Okereke, S. C. Echezona, G. B. Akuchu, et al., “A Mobile Application Based Optimized Out-Patient Emergency Request Model,” International Journal of Advanced Trends inComputer Science and Engineering (IJATCSE), vol. 13, no. 2, pp. 44-52, Mar-Apr. 2024.

[3]. K. Clouse, S. Noholoza, S. Madwayi, M. Mrubata, C. S. Camlin, L. Myer, T. K. Phillips, et al., “The Implementation of a GPS-Based Location-Tracking Smartphone App in South Africa to Improve Engagement in HIV Care: Randomized Controlled Trial,” BMJ Open, vol. 12, no. 11, e064946, Nov.2022.

[4]. Marolla C. 2020 ”Health in the Digital World.” In Information and Com-munication Technology for Sustainable Development. London: Taylor and Francis, https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37373505.