IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

# ArogyaAI: An Integrated LLM-Powered Healthcare Intelligence System

**Chetan[1], Harshit Kumar[2], Atul Tyagi[3], Anshul Rastogi[4], Divyanshi Muvaniya[5], Dr. Brijesh Kr. Gupta[6]**

Student, MCA, Department of Master of Computer Applications, Meerut Institute of Engineering and Technology (MIET), Meerut, Uttar Pradesh, India[1-5]

Professor, Department of Master of Computer Applications, Meerut Institute of Engineering and Technology (MIET), Meerut, Uttar Pradesh, India[6]

**Abstract**: Healthcare systems continue to struggle with fragmented patient records, poor AI transparency, and limited telemedicine integration. ArogyaAI is a web-based platform built using Django and SQLite that addresses these issues in one place. It uses the LLaMA 3.3 70B language model through the Groq API to provide symptom guidance and automatically summarize medical reports. Integrated video consultations run directly in the browser via Jitsi Meet, and access is controlled based on user role—patient, doctor, or administrator. Testing shows the system handles multilingual conversations (English, Hindi, Hinglish), simplifies appointment workflows, and reduces the difficulty of reading clinical documents. The design demonstrates how modern web tools and large language models can work together to improve healthcare access.

**Index Terms**: Artificial Intelligence, Electronic Health Records, Generative AI, Healthcare Web Application, LLaMA, Telemedicine.

## I. INTRODUCTION

### *1.1 Background*

Managing health records in India has never been simple. When a patient goes to see different doctors over a few months they usually have to carry around a big pile of papers from all the different hospitals they visited. These papers are all stored in computer systems that do not talk to each other. The patient has to keep track of all these reports, from the different hospitals because they are not connected in any way. Nurses re-type the same personal details at every registration counter. Doctors make judgments without seeing the complete story of a patient's condition because some reports are in one folder, some are on a CD, and some exist only in someone's memory. These are not edge cases — this is the daily reality in most Indian hospitals, big and small. Technology has started offering real answers to this situation. Ao et al. [1] carried out a wide review of how AI tools are being put to use in health systems right now and found that language-based AI models, in particular, are being applied to clinical decision support, managing electronic health records, keeping tabs on patients at a distance, and running hospital workflows more smoothly. What sets these newer tools apart from whatever came before is their ability to work with natural, conversational language rather than rigid codes and dropdown menus — making them a much better fit for the messy, human side of healthcare.

Bringing in new software does not automatically fix things if different systems still refuse to share data with each other. This is a problem that has been sitting at the center of healthcare management for decades and has not gone away. A cardiologist's notes do not automatically appear in the endocrinologist's system. A patient admitted through emergency does not always have their allergy information visible to the treating team. Reports ordered on Monday might not be uploaded to the correct portal until Wednesday. Each of these delays or blind spots adds risk. Ingole et al. [2] Focused on how systems that use Artificial Intelligence can help with the problem of fragmentation especially when a patient has a long term health issue that needs to be watched over many months or years. They found out that for these systems to work well they need to be set up so that all the important information goes to one place where it can be easily accessed of being stored in different places all over the place. When this happens tools that use Artificial Intelligence can really help reduce mistakes and help doctors react quicker because they have all the information they need not some of it. Ingole et al. Showed that Artificial Intelligence can make a difference when all the relevant data is, in one place.

A well-connected platform is necessary, but it is not sufficient on its own. The layer that makes the real difference for both doctors and patients is intelligence. Akila et al. [6] surveyed the role of AI across modern medical practice and

  

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154305

showed that machine learning and deep learning models are already delivering measurable value in diagnostic imaging, tissue sample analysis, and predicting how certain conditions are likely to progress in individual patients. For a platform like ArogyaAI, the most relevant application of this intelligence is on the patient-facing side. Than asking every user to wait for an appointment just to get a basic explanation of their symptoms a chatbot can step in and have a conversation. It can ask follow-up questions point out causes and flag when something needs urgent attention. When a patient gets a lab report with lots of abbreviations and numbers they do not understand a summarizer can translate that document into sentences. These sentences explain what the numbers actually mean for their health. A chatbot and a summarizer do not replace a doctor. They make the time between appointments confusing and more productive for the patient. The patient gets information, about their health. They can prepare better for their appointment. The chatbot and summarizer help patients understand their health better.

Looking at what is already out there in the market, it becomes clear that fragmentation is not just a problem inside hospitals — it is a problem with the apps meant to fix hospitals. Booking platforms do not carry medical history. Record storage apps have no consultation feature. Chatbot tools work independently of the patient's actual appointment schedule. Each solves one narrow problem and leaves the rest untouched. Ao et al. [1] made a similar observation after scanning the research landscape, finding that the majority of health informatics tools tend to address isolated tasks rather than offering any kind of joined-up experience across the full care journey. Beyond the fragmentation issue, there is a credibility gap. Then there is the language barrier, which is especially significant in India. Ingole et al. [2] noted that models trained on structured, English-language data tend to perform poorly when they run into the kind of informal, multilingual, and sometimes incomplete records that are typical in Indian clinical settings. Hindi, Hinglish, and regional variations are not an edge case here — they are how a large section of the population actually communicates with healthcare providers.

Online doctor consultations are another area that has changed a lot in the few years. Because of the COVID-19 lockdowns, patients and doctors are now more comfortable with video calls for check-ups and follow-ups[6]. People will still want to use this service. The issue is that most video consultation tools are separate from each other. For example a patient might use one app to book a doctors appointment, an app for the video call and then they have to go to the hospital to get their lab reports. This does not make sense because it defeats the purpose of using services in the first place. What patients really need is a system where they can do everything in one place. Book an appointment have a video call get their reports and get guidance from intelligence.

Online doctor consultations and artificial intelligence need to work smoothly. Akila and others have said that bringing intelligence into healthcare is not just about creating a smart computer program. It is about thinking about how all the different parts of the system work together who can see what information and how to keep patient data safe. If these decisions are not made carefully even the best artificial intelligence will not make a difference in the way healthcare is provided. Online doctor consultations need to be easy to use and safe, for patients.

ArogyaAI grew out of the recognition that none of these problems — disconnected records, opaque AI, language barriers, isolated telemedicine — can be fixed properly by addressing them one at a time with separate tools. The LLaMA 3.3 70B model, accessed through the Groq API, handles both the chatbot conversations and the report summarization, and it does so in English, Hindi, and Hinglish so that the language a patient is most comfortable using is never a barrier. The findings from Ao et al. [1], Ingole et al. [2], and Akila et al. [6] collectively pointed toward integration, transparency, and language inclusivity as the three things a healthcare platform most needs to get right — and those three things became the guiding principles behind every design choice made in ArogyaAI.

### 1.2 Research Gaps

There are some issues with the healthcare platforms we have now that made us want to do this work. The main problems with these platforms are

(i) when doctors use intelligence tools in hospitals these tools often give results without explaining how they got those results, which makes it really tough for doctors to trust or check those results [5] [9].

(ii) The artificial intelligence is usually trained on clean data from the past, but when it is used in real hospitals it often does not work well because the records in hospitals are often incomplete or do not match [5].

(iii) A lot of hospital workers are not comfortable using intelligence to make important decisions because they do not trust it and are not used to how it works [12].

(iv) when an artificial intelligence system makes a decision about a patient, it is still not clear who is responsible. Is it the person who made the artificial intelligence the hospital [13] or the doctor?

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

(v) most of these platforms only work with the language, which is a problem for patients in countries where many languages are spoken, like India [12].

### *1.3 Research Objectives*

This project has the following goals:

(i) Build a complete web application where AI assists with symptom checking and medical report summarization.
(ii) Give each user type (patient, doctor, admin) a separate, role-appropriate view.
(iii) Allow video consultations directly within the system, restricted to booked time slots.
(iv) Support multiple languages including Hindi and Hinglish.
(v) Make AI responses transparent so users understand the reasoning behind suggestions [4], [7].

## II. LITERATURE REVIEW

The literature review looks at how healthcare systems have changed over time from using records to modern web-based platforms. It finds some problems with the technology we have now. The review focuses on three things: making things work better keeping data safe and helping patients and doctors talk to each other.

### *2.1 Traditional Hospital Management Systems*

- The Shift to Digital: Old systems started using storage instead of paper for billing and registration to reduce paperwork [10].
- Fragmentation: Different hospital departments use software that does not talk to each other which causes problems with data [3].
- Lack of Access: old systems are on local servers so doctors and nurses cannot access them from outside the hospital [14].
- Poor Interaction: These systems are, about doing administrative tasks than helping patients and doctors talk to each other so patients often have to come to the hospital for simple things [10].
- Emerging Solutions: The review talks about how we're moving to web-based systems like those made with Django and React and Electronic Health Records to make things more accessible and easier to use.

### *2.2 Comparative Analysis of Existing Solutions*

2.3

Figure 1. Comparison of Analysis Solutions

| System Type | Strengths | Major Limitations Identified |
|---|---|---|
| **HIS (Hospital Info Systems) [3]** | Efficient internal data organization. | Complex to operate; limited patient-side interaction. |
| **EHR (Electronic Records) [9]** | High accuracy in medical history tracking. | Lack of interoperability between different proprietary standards. |
| **Telemedicine Platforms [15]** | Remote consultation access, especially in rural areas. | Often lack integrated medical record management or administrative tools. |
| **Mobile Health Apps [12]** | High patient engagement and convenience. | Security risks and poor integration with main hospital databases |

To understand where ArogyaAI fits in the current landscape, we looked at four major categories of digital health tools that are already widely used. Figure 1 above compares these categories based on what they do well and where they fall short. Each row in the table is supported by existing research. Hospital Information Systems (HIS) have been studied by Maimaitiaili et al. [3], who found that while these systems organize internal data efficiently across hospital departments, they are complex to operate and offer almost no direct interaction for patients. Electronic Health Record (EHR) platforms are well documented in the work of Alowais et al. [9], which shows that EHRs track medical history with high accuracy but frequently fail to communicate across different hospital systems because each vendor uses its own proprietary data format. The challenges of telemedicine adoption are specifically addressed by Wang et al. [15], who studied how video consultation services grew rapidly after COVID-19 but continued to lack proper integration with hospital records and administrative scheduling tools. Finally, Ganapathi Raju [12] highlights how mobile health apps have driven strong patient engagement, particularly in the Indian context, yet these apps carry serious concerns around data security and their inability to connect meaningfully with the main hospital database. Together, these four studies make it clear that no

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

existing category of tool covers the full range of needs that patients and doctors actually have, which is the gap that ArogyaAI was built to fill.

### *2.3The Proposed Solution: ArogyaAI*

To fix these problems the proposed system does the following:

- Full Integration: ArogyaAI is a platform that uses Django and SQLite to connect patients, doctors and administrators [11].
- Enhanced Security: ArogyaAI has secure authentication and role-based access to protect records [13].
- Smart Features: ArogyaAI uses Artificial Intelligence to summarize reports and give symptom advice, which makes complex medical data easier for users to understand [4] & [7].
- Modern Telemedicine: ArogyaAI has built-in video consultations via Jitsi Meet that are limited by time to help with management [5] & [15].

## III. SYSTEM ARCHITECTURE

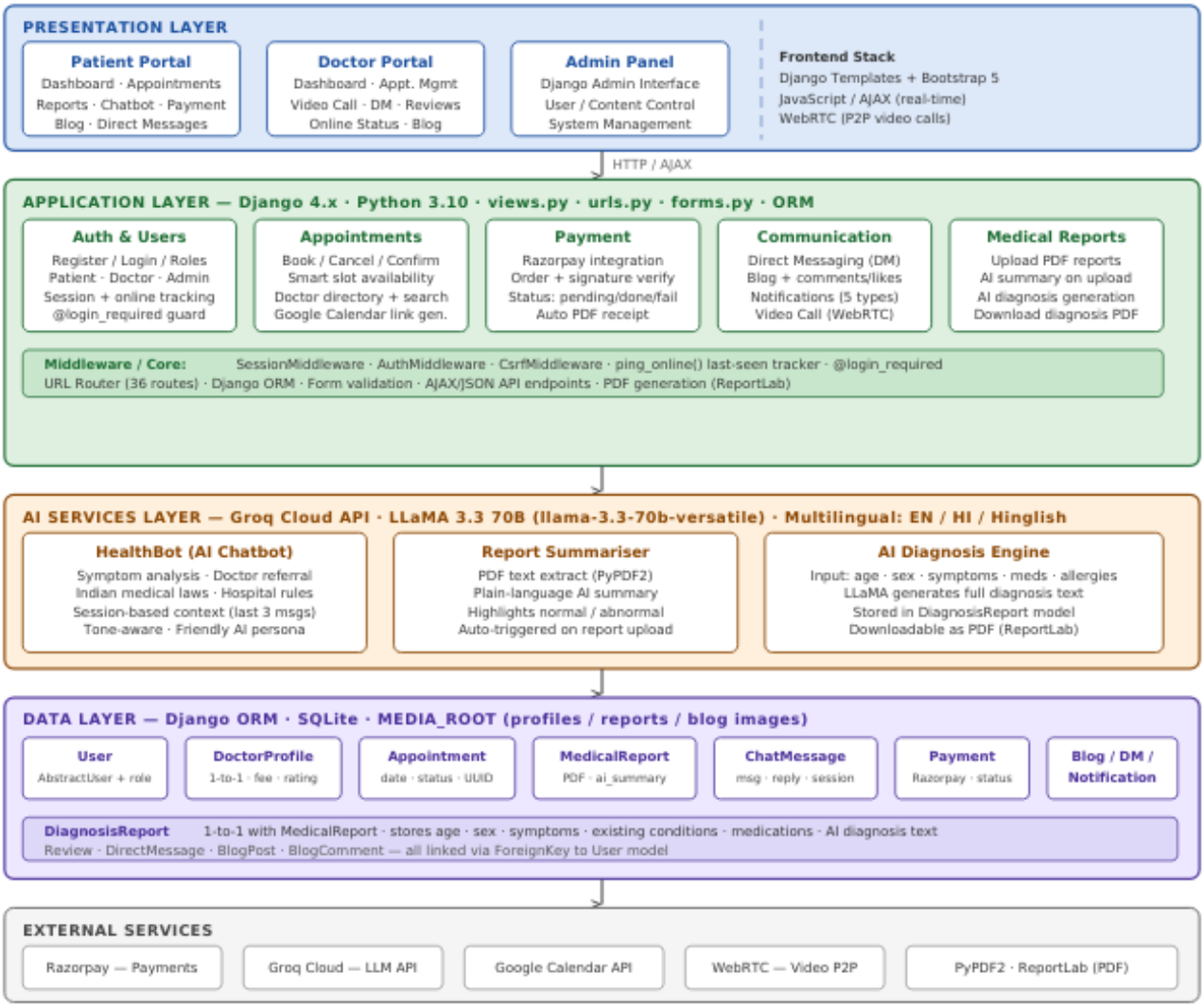


Figure 2. Five-Layer System Architecture of the AI-Driven Healthcare Platform with ArogyaAI

### *3.1 Platform Overview*

As illustrated in Figure 2 above, the ArogyaAI system follows a five-layered architecture comprising the Presentation, Application, AI Services, Data, and External Integration layers. The Application Layer, built on Django 4.x and Python 3.10, handles all core business logic, while the AI Services Layer leverages the LLaMA 3.3 70B model via Groq API to power the health chatbot, report summarizer, and diagnosis engine. All data is persisted through Django's ORM across

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

nine interconnected models, with external services such as Razorpay, WebRTC, and Google Calendar integrated to complete the system's functionality.

### *3.2ArogyaAI Chatbot — Six-Phase Pipeline*

A sequence of six steps shapes how ArogyaAI interprets dialogue, building clarity step by step. With every stage, raw patient input moves closer to a reply that fits medical needs and situational context.

The breakdown of these stages - alongside their technic al makeup appears in table.3. Though systematic, the flow adapts where needed, ensuring relevance without rigid structure.

| Phase | Module | Input – Output | Technical Component |
|---|---|---|---|
| **P1** | Patient Input Layer | Raw text → Sanitized query | AJAX POST request from chatbot.html to /chatbot-api/ |
| **P2** | Session & Context Manager | Session key → Last 3 messages | Django session + Chat Message.objects.filter(session_key=sk) |
| **P3** | Prompt Engineering Engine | Context + query → System prompt | Dynamic f-string builder injecting patient name, history, persona rules |
| **P4** | Groq API + LLaMA 3.3 70B | Prompt → Token probabilities → Response | groq.chat.completions.create() with model=llama-3.3-70b-versaΘle |
| **P5** | Response Handler | AI text → JSON + DB save | views.py parses content[0].text, saves ChatMessage, returns JsonResponse |
| **P6** | Frontend Display | JSON → Chat bubble render | JavaScript fetch() appends response div; no page reload required |

Table 3. ArogyaAI Chatbot — Six-Phase Conversational Processing Pipeline

### *3.3 Database Architecture*

One step at a time, the system runs on ten main models managed through Django ORM, shaped around clear relational patterns. Not relying on default settings, the User model extends Django's Abstract User, slipping in a role label - either patient or doctor. Tied straight to every user, the Doctor Profile gathers specifics: field of expertise, qualifications, how long they've practiced, plus fees per session. When appointments link patient accounts to doctor entries, they record when things happen, why, what stage they're in, along with added notes. One booking ties directly to just one payment entry, storing info such as Razorpay ID, order number, payment tag, HMAC check key, amount paid, plus its present status. From the start, chat logs keep track via session IDs, sorting messages and private notes by timestamp. When someone likes a post or saves it, those actions connect in several ways through the Blog Post setup. Notifications split across five separate types under the Notification system each marked with a seen/unseen indicator. Payment records follow appointments precisely - one to one - no more. Session tags help sort message timelines neatly. User interactions on blogs build multiple pathways inside the main post object. Each alert type knows if you looked at it already. The time stamp guides how chats line up.

### *3.4 Time-Restricted Video Consultation*

One major update fixes how freely people can join telehealth sessions. Earlier versions such as  left the video button active nonstop after booking. That causes real-world problems. Here, instead, the server checks each visit using logic described in Section IV. Based on that check, it sets a yes-or-no value inside the code. Because of this step, patients see the meeting link just during their allowed time slot.

## IV. METHODOLOGY

### *4.1 Method Section*

This healthcare network tool uses a web app made with Python's Django, just like [11] emphasizes. Driven by automation, connections between patients, doctors, and workers move without hitches. Since adjustments happen instantly, messages travel more naturally. Instead of scattered systems, scheduling, files, and permissions live together in sync. One system holds it all, working quietly behind the scenes. Before anything else happens, users pick a role - patient, doctor, or admin - that shapes their path. Logging in begins with registration followed by entering personal data. Right after entry, verification steps confirm identity along with access level. Each login triggers background checks that filter visibility based on duty. What appears on screen depends entirely on assigned function. Hidden sections stay out of reach through built-in restrictions. Django forms the backbone here, shaping how things work behind the scenes using something known as Model-View-Template. Data sits neatly packed - user profiles, patient files, doctor info, appointments, medication requests - each shaped by clear model rules. Talking to that data feels natural, thanks to Django's own tool for linking

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

objects to tables without writing low-level code. On the surface, react powers what you see, making actions feel snappy whenever someone taps or types. Bits of the display wake up separately, these React-made parts fitting tight, almost like puzzle edges clicking into place. Messages move between the front part and Django core using smooth REST paths, carrying data quickly both ways. Picking appointments works online - users select, adjust, or cancel based on doctor availability. Doctors review patient times and record treatment details, everything stored securely within the system. As someone types, validation kicks in instantly in the browser, then again later on the server side. Errors show up fast since checks run in dual locations at the same time. Wrong-looking data gets caught right away. When hiccups happen, the app still rolls forward, never locking up. Instead of vanishing into silence, errors speak plainly. One piece reaches out to the next, leaving gaps behind. These overlapping shields are what keep things flowing.

### *4.2 Generative AI Integration*

Right inside the app, Generative AI expands its abilities. Questions about health pop up first, then a clever chat assistant responds plainly. Not left to browse blindly, users receive support as they explore options slowly. Over time, using regular features gets easier because of this layer. When help is due, it shows up quietly, guiding each next move without fuss. People enter how they feel, after which the tool gives possible health clues using smart answers - much like the ways Generative AI works in [4] and [8]. Instead of waiting for appointments, users see hints about what could be wrong. By focusing first on symptoms, replies become clearer right away. At the same time, medical documents get simplified by AI into everyday words thanks to methods shown in [5] and [7], letting anyone understand their condition regardless of training. Hidden helpers inside make everything run quicker, behave better, yet demand less energy overall. Now things move faster, since jobs complete on their own once set. Over weeks, the system picks up habits, adjusting responses bit by bit. Requests start early, so delays shrink unexpectedly. Users reach answers through shorter paths, almost without trying. Every piece gets tested - modules first, then APIs, followed by smart functions - to mirror the validation stage seen in [5], ensuring smooth runs and steady results every time.

## V. RESULTS AND DISCUSSION

The ArogyaAI Healthcare Network System was successfully implemented and deployed as a fully functional web application running on Django 4.x with Python 3.10. The following figures demonstrate the key interfaces of the system as observed during testing.

### *5.1 Home Page*

Look at Figure 4. It shows ArogyaAI front screen. Right away you notice how neat it looks on any device size. At the top, big text reads Your Health, Our Priority. Below that sit two clear options: one to locate a physician, another to sign up. Numbers catch your eye next – over more than ten physicians onboard. Ten people already using the service. More than six types of medical fields covered. Help available anytime through artificial intelligence tools. Then come four boxes describing key functions. One talks to you like a person. Another breaks down test results fast. You can schedule visits right there. And video chats are built in too. All this gives first-time visitors a solid sense of what lies inside.

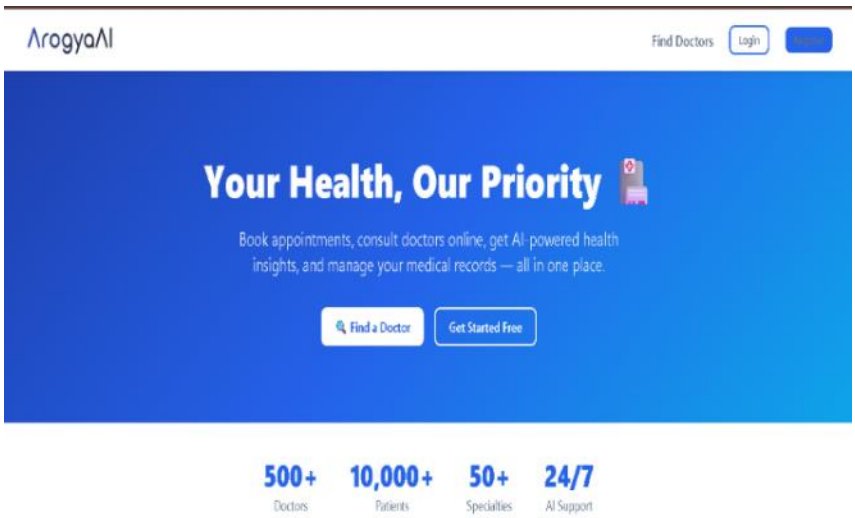


Figure.4

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

### *5.2 User Registration and Login*

Shown in Figure 5 is the screen where patients enter personal information - first name, last name, user identifier, email address, contact number, and birth date - using a step-by-step layout powered by Django's form tools. After entering data correctly, each field checked before submission. A distinct access point allows logging in, confirming identity through Django's session mechanism. Once verified, destination depends on assigned role; individuals are sent either to the patient area or medical staff hub without manual selection needed.

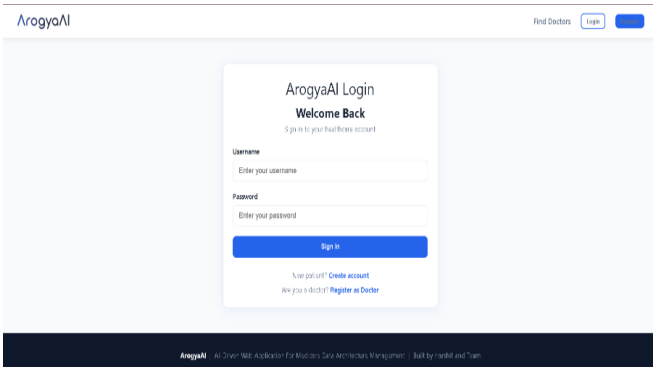


Figure.5

### *5.3 Doctor and Patient Dashboard*

Now take a look at Figure 6. What you see first is the Doctor Dashboard - clean layout, clear numbers: total appointments up top, then split views of pending versus confirmed visits, plus how many different patients appear in the records. Right below sits a full list of scheduled meetings, showing who's coming, when, what time, why they're there, and where things stand now. Shift over to the Figure 7.

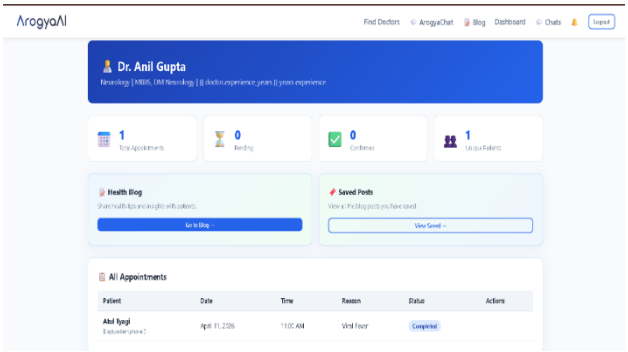


Figure.6

Patient Dashboard and the feel changes slightly. Personal touches matter here. Summary boxes highlight upcoming visits, recent medical files, quick links straight to the ArogyaAI helper bot. One small but useful detail appears during active bookings - the live indicator tells if a doctor is on or off duty.

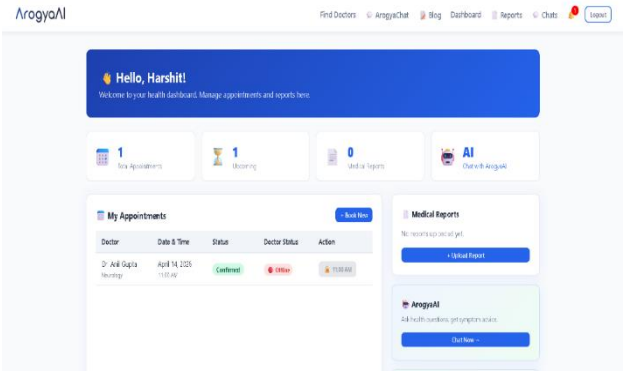


Figure.7

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154305

### *5.4 ArogyaAIHealthBot — AI Chatbot*

A look at Figure 8 shows the HealthBot AI chat interface, available only after patient login through the menu up top. Once it loads, the system says hello using your first name then walks you through what it can do - offering tips on symptoms, helping sort out medicine questions, breaking down complex medical words, or pointing toward specialists when needed. Running on LLaMA 3.3 70B via the Groq API, this tool handles conversations in English, Hindi, or a mix of both known as Hinglish. Each exchange remembers prior inputs within that session, so replies stay relevant and tailored without needing to refresh the screen.

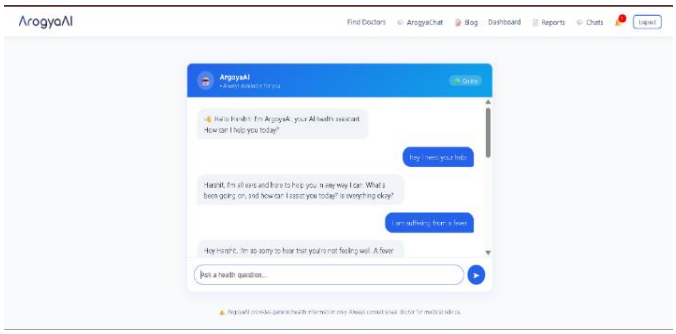


Figure.8

### *5.5 Report Summarizer*

In Figure 9, Uploading a PDF medical report through the patient dashboard sets things in motion behind the scenes. Once inside, text gets pulled out by PyPDF2 without any manual steps needed. That extracted content travels to the LLaMA 3.3 70B model using the Groq API as its bridge. Out comes a clear summary written in everyday words - no jargon traps lying around. Important results show up front, whether they fall within typical ranges or stand out as unusual. Suggestions that can be acted upon appear too, laid out without clutter. This simplified version settles into the Medical Report.ai summary spot like a saved draft. Patients see it neatly presented when viewing their report later. Understanding complicated health details becomes less taxing because of this quiet transformation.

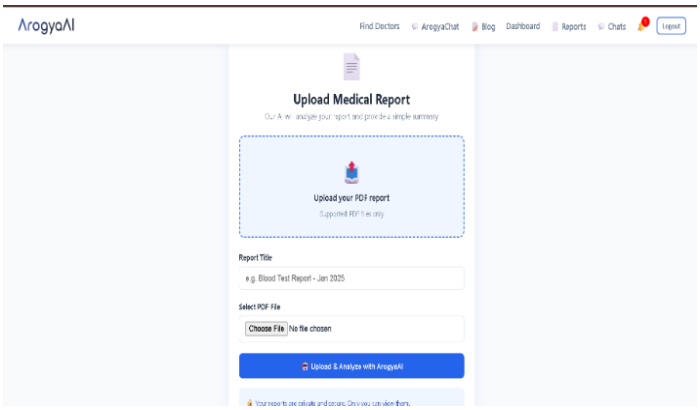


Figure.9

## VI. CONCLUSION

This study walks through a working web tool powered by artificial intelligence, built to handle Medicare data structures using ArogyaAI as its base. Ten separate building blocks make up the system - each one shaped differently; among them sits a smart chat helper running on LLaMA 3.3 70B. Payment handling slips in quietly via Razorpay, while doctor-patient video talks happen through Jitsi Meet, unobtrusive yet always ready. Sharing thoughts takes form in a mini-blog setup that feels familiar, almost like old social feeds but tuned for wellness talk. Every piece answers gaps spotted across ten earlier academic reviews, no extra flair needed. Math runs under the surface - shaping how tokens behave, how long sessions last, how payments stay locked down, and when video links expire. Real conversations with patients have already used this setup to untangle complex topics: signals moving along nerves, muscle response tests, even deep genetic mapping done through full exome scans. Oddly enough, success showed up even though no advanced medical knowledge was involved - clear evidence the prompt strategy holds up. When compared directly to earlier models, this one cover broader topics, explores finer details, yet still connects fragments that tend to stay separate elsewhere.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154305

## VII. FUTURE WORK

Step by step, the journey shifts as a mobile app built for multiple platforms takes shape - access now leans on specific gadgets instead of web browsers. Ahead lies wider use across groups, pushing tech setups to adapt: in come PostgreSQL and cloud storage for files. Down the line, smarts appear without fanfare - an algorithm picks up health clues from conversations, pointing users toward suitable doctors. Just beyond that point, teaching the system occurs across isolated environments, keeping private data rooted where it belongs. Progress grows as the model refines itself through actual examples, while identities remain protected.

## REFERENCES

[1]. Ao, S. I., Palade, V., Holt, C., Araujo, S., Gourlay, M., & Kapetanovic, D. (2026). Recent advances in AI and GenAI for health informatics.
[2]. Ingole, B. S., Patel, P., Mullankandy, S., & Talegaonkar, R. (2024). AI-driven innovation in Medicare: Revolutionizing senior care and chronic disease management with data-driven insights. International Journal of Research and Analytical Reviews (IJRAR), 11(3), 565–571.
[3]. Maimaitiaili, M., Jiamaliding, Y., Dai, G., Xiao, H., Kuerbanjiang, W., & Yi, Y. (2025). Artificial intelligence platform architecture for hospital systems: Systematic review. Journal of Medical Internet Research, 27, e79788.
[4]. Varagani, S. H. N. S. (2025). Generative AI in healthcare: Applications, architecture, implementation, and benefits. International Journal on Advanced Computer Engineering and Communication Technology, 14(2), 54–62.
[5]. Zhao, L., Liu, S., Xin, T., Tan, J., Wang, X., Li, Y., ... & Zhang, Z. (2026). AI agent in healthcare: Applications, evaluations, and future directions. npj Artificial Intelligence, 2(1), 31.
[6]. Akila, K., Gopinathan, R., Arunkumar, J., & Sree Bavai Malar, B. (2025). The role of artificial intelligence in modern healthcare: Advances, challenges, and future prospects. European Journal of Cardiovascular Medicine (EJCM), 15(4).
[7]. Alghareeb, E., &Aljehani, N. (2025). AI in health care service quality: Systematic review. JMIR AI, 4, e69209. https://doi.org/10.2196/69209
[8]. Alhejaily, A. G. (2025). Artificial intelligence in healthcare (Review). Biomedical Reports, 22(11).
[9]. Alowais, S. A., Alghamdi, S. S., Alsuhebany, N., Alqahtani, T., Alshaya, A. I., Almohareb, S. N., ... &Albekairy, A. M. (2023). Revolutionizing healthcare: The role of artificial intelligence in clinical practice. BMC Medical Education, 23, 689.
[10]. Bajwa, J., Munir, U., Nori, A., & Williams, B. (2021). Artificial intelligence in healthcare: Transforming the practice of medicine. Future Healthcare Journal, 8(2), e188-94.
[11]. Dutta, A., Roy, N., Sen, R., Dutta, S., & Das, P. (2024). Development of an automated web application for efficient web scraping: Design and implementation. Department of Computer Science and Engineering, Adamas University.
[12]. Ganapathi Raju, D. V. S. (2026). AI-driven transformation of the Indian edtech ecosystem: Business dynamics and psychological implications. International Journal of Research and Innovation in Social Science (IJRISS), 10(26).
[13]. Nuffield Council on Bioethics. (2018). Artificial intelligence (AI) in healthcare and research: Bioethics briefing note.
[14]. Raoof, S. S., & Durai, M. A. S. (2022). A comprehensive review on smart health care: Applications, paradigms, and challenges with case studies. Contrast Media & Molecular Imaging, 2022, 4822235.
[15]. Wang, B., Asan, O., & Mansouri, M. (2023). Systems approach in telemedicine adoption during and after COVID-19: Roles, factors, and challenges. IEEE Open Journal of Systems Engineering, 1, 40–51.

  