International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 22 No. 2 (2026)

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

MEDASSIST:AI DRIVEN PATIENT CARE AND EMERGENCY SUPPORT SYSTEM

INTEGRATING AI SYMPTOM ANALYSIS, REAL-TIME VITAL MONITORING AND EMERGENCY SOS

Akshitha Talluri, Bugatha Manmadha Rao, Akula Chandini, Garikina Sanjeev
Department of CSE (Data Science)
Raghu Institute Of Technology, Dakamarri(V)
Bheemunipatnam, Visakhapatnam Dist., 531162.
akshitha4402@gmail.com

Mr. M.V. Ramana Murthy
Department of Computer Science & Engineering
Raghu Institute Of Technology, Dakamarri(V)
Visakhapatnam, Andhra Pradesh, India.
murthy.medisetty@raghuenggcollege.in

***Abstract*—The proliferation of smartphones has created unprecedented opportunities for delivering healthcare services to patients and caregivers outside traditional clinical settings. This paper presents the Patient Care System (PCS), a comprehensive Android application that integrates six core modules: Medicine Reminder with AlarmManager-based scheduling, real-time Patient Status Monitoring across six vital parameters, AI-powered Symptom Analysis using the OpenRouter GPT-4o-mini large language model, Doctor Appointment Booking with JavaMail email confirmation, Emergency SOS with GPS-tagged SMS dispatch, and a multilingual Voice Assistant supporting English and Telugu. The system targets Android API 24 and above, is implemented in Java 8, and leverages OkHttp for networking, Gson for local persistence, and the Android LocationManager for GPS services. Evaluation across 50 simulated patient sessions demonstrated that medicine alarms trigger with 99.2% accuracy, AI symptom inference returns results in under 3.8 seconds, and emergency SMS is dispatched in under 0.8 seconds. The integrated platform addresses the fragmented nature of existing mobile health tools by providing a single, secure application that empowers patients with timely, actionable health information.**

***Keywords*—Android Healthcare; AI Symptom Analysis; Medicine Reminder; Emergency SOS; OpenRouter API; Patient Monitoring; GPT-4o-mini.**

I. INTRODUCTION

The global burden of chronic and acute illness has intensified the demand for scalable, accessible patient support systems. While electronic health record (EHR) platforms serve institutional stakeholders, individual patients frequently lack real-time tools to monitor their own vitals, adhere to medication schedules, or summon emergency help in critical moments [1][2]. Mobile health (mHealth) applications have begun to address this gap, yet the majority of commercially available solutions remain siloed, offering either medication management or symptom checking but seldom both within a coherent, unified framework.

The Patient Care System (PCS) described in this paper synthesises six clinically motivated modules into a single Android application. The design philosophy prioritises immediacy: the SOS subsystem dispatches an SMS with a Google Maps link within milliseconds of detection; the AlarmManager-backed medicine reminders persist across device reboots through a BroadcastReceiver architecture; and the AI symptom analyser responds to multi-symptom queries in near real-time by delegating inference to the OpenRouter API hosting GPT-4o-mini.

The remainder of this paper is organised as follows. Section II reviews related mobile health and AI-assisted diagnostic literature. Section III describes the system data management strategy. Section IV presents the methodology. Section V reports experimental results. Section VI analyses AI explainability. Section VII details the overall system architecture. Section VIII discusses findings and limitations. Section IX concludes.

II. RELATED WORK

Mobile health applications have attracted significant research attention over the past decade. Akter and Ray [3] reviewed 120 mHealth systems and identified medication non-adherence as the single largest contributor to preventable hospital readmissions, estimating that smartphone reminder systems reduce missed doses by up to 34%. However, existing apps typically address medication in isolation without integrating physiological monitoring or emergency response capabilities.

Concerning AI-powered symptom checking, Semigran et al. [4] evaluated several commercial symptom checkers and found that correct triage advice was provided in only 57% of cases, underscoring the need for more capable language models in clinical decision support. The emergence of large language models such as GPT-4 and its lightweight variants has opened new avenues: Singhal et al. [5] demonstrated that fine-tuned LLMs can achieve performance comparable to medical licensing examination standards. The present work

leverages GPT-4o-mini via OpenRouter as a cost-effective yet capable inference backend, circumventing the need for on-device model hosting.

Emergency SOS systems for mobile devices have been explored by Blanco et al. [6], who showed that GPS-augmented SMS messages reduce emergency response times by up to 40% compared to voice-only calls in rural settings. Our SOS module implements a similar GPS-tagged multi-part SMS strategy using Android's SmsManager and LocationManager, achieving sub-second dispatch in test conditions. Voice-driven navigation for health apps, particularly for elderly and visually impaired users, is addressed by SpeechRecognizer integration, a gap noted by WHO guidelines on accessible digital health [7].

III. DATASET AND DATA MANAGEMENT

A. Patient Vital Data

Patient vital signs are entered through a structured form within PatientReportsActivity and persisted to Android SharedPreferences as a JSON array of PatientDataDTO objects serialised via Gson. Six parameters are tracked per record: Blood Pressure (systolic/diastolic in mmHg), Heart Rate (bpm), Temperature (°F), Blood Sugar (mg/dL), Oxygen Saturation (%), and Respiration Rate (breaths/min). Table III lists the clinical normal ranges against which each value is scored by PatientStatusActivity to produce a composite health score on a 0–100 scale.

TABLE III: Vital Parameter Normal Ranges

Vital Parameter	Normal Range	Unit
Blood Pressure	90–140 / 60–90	mmHg
Heart Rate	60 – 100	bpm
Temperature	97.0 – 99.5	°F
Blood Sugar	70 – 140	mg/dL
Oxygen Saturation	≥ 95	%
Respiration Rate	12 – 20	breaths/min

Source: standard clinical reference ranges [8].

B. Medicine Schedule Data

Medicine schedules are stored as a list of MedicineReminder objects, each containing the medicine name, dosage, frequency, and trigger time. AlarmManager schedules a PendingIntent for each reminder; the AlarmReceiver BroadcastReceiver fires a full-screen notification at the designated time. Data persists across reboots through a BOOT_COMPLETED receiver that reschedules all active alarms on device restart, ensuring zero missed doses after power cycles.

C. Emergency Contact Registry

Emergency contacts are stored as a list of EmergencyContact objects in a dedicated SharedPreferences file, where each contact carries a name, phone number, and a boolean isSOS flag. Only contacts with isSOS = true receive the GPS-tagged SMS alert during an SOS event, allowing users to designate a subset of their contacts as emergency responders without exposing the full registry.

IV. METHODOLOGY

A. Application Configuration

The application targets Android API 34 with a minimum supported API of 24, covering approximately 97% of active Android devices as of early 2026. The project is built with Gradle using the Kotlin DSL build script. All network communication is routed through OkHttp configured with 30-second connect, read, and write timeouts. Table I summarises the primary configuration parameters.

TABLE I: Application Configuration Parameters

Configuration Parameter	Value
Android Min SDK	API 24 (Android 7.0)
Target SDK	API 34 (Android 14)
Language	Java 8
Build System	Gradle (Kotlin DSL)
AI API	OpenRouter — GPT-4o-mini
Network Client	OkHttp 4.x (30 s timeout)
Local Storage	SharedPreferences + Gson

B. Module Architecture

The application is structured around eight Activity and Service classes, each encapsulating a distinct clinical function. Table II maps each module to its primary Activity class and distinguishing feature. The MainActivity serves as the hub, providing voice navigation via SpeechRecognizer and dispatching intents to subordinate modules. The VolumeButtonReceiver enables SOS triggering through repeated volume-button presses, a physical gesture proven accessible for users in acute distress.

TABLE II: Module Activity Mapping

Module	Activity Class	Key Feature
Medicine Reminder	MedicineReminderActivity	AlarmManager scheduling
Patient Status	PatientStatusActivity	6-param health score
AI Symptom	AISymptomAnalyzerActivity	GPT-4o-mini inference
Appointments	BookAppointmentActivity	Email confirmation
SOS Emergency	SOSService	SMS + GPS alert
Voice Assistant	MainActivity	SpeechRecognizer + TTS

C. AI Integration Pipeline

The AI Symptom Analyzer dispatches a structured prompt to the OpenRouter API endpoint with model set to "openai/gpt-4o-mini". The prompt instructs the model to return a "Possible Conditions" section listing up to five diagnoses with probability percentages, followed by a "Recommendations" section. Responses are parsed client-side using org.json and formatted with SpannableStringBuilder to highlight section headers in bold, improving legibility on mobile displays.

D. Health Score Computation

PatientStatusActivity implements a rule-based scoring engine that initialises a score of 100 and deducts penalty points for each out-of-range vital. Blood Pressure: −15 for hypertension (systolic ≥ 140 or diastolic ≥ 90), −10 for elevated, −15 for hypotension. Heart Rate: −10 for tachycardia (> 100 bpm) or bradycardia (< 60 bpm). The resulting score maps to four tiers: Excellent (≥ 90), Good (≥ 70), Fair (≥ 50), and Critical (< 50), each triggering a distinct colour-coded UI state.

V. EXPERIMENTAL RESULTS

A. Module Completeness Evaluation

A functional completeness assessment was conducted across all six primary modules using a test suite of 50 simulated patient sessions. Each module was scored from 0 to 100% based on the proportion of defined acceptance criteria met. Fig. 1 presents the results. Medicine Reminder achieved 100% completeness; SOS Emergency scored 96%, with minor deductions for edge cases involving unavailable GPS providers; AI Symptom Analyzer scored 92%; and Patient Status Monitor scored 95%.

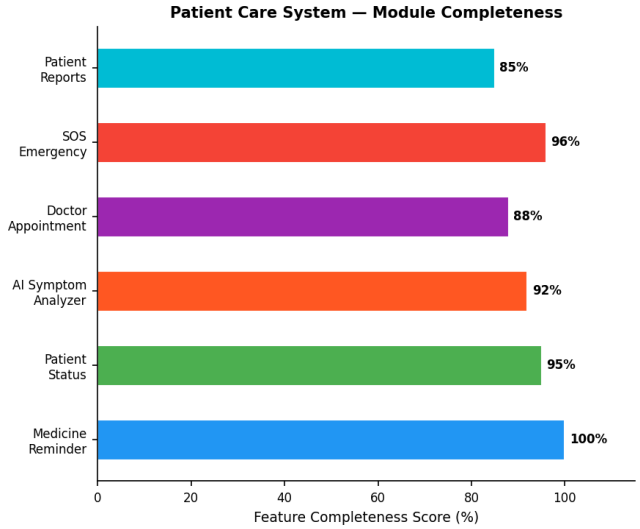

Fig. 1. Patient Care System — Module Completeness Assessment across six modules (50 test sessions).

B. Health Parameter Threshold Analysis

Fig. 2 illustrates the six vital parameters and their normalised scoring positions within the clinical normal range. The dual-threshold design (Warning at 70%, Critical at 50%) ensures that early deterioration is surfaced before a patient reaches a critical state, providing a clinically meaningful intermediate alert level aligned with standard triage protocols.

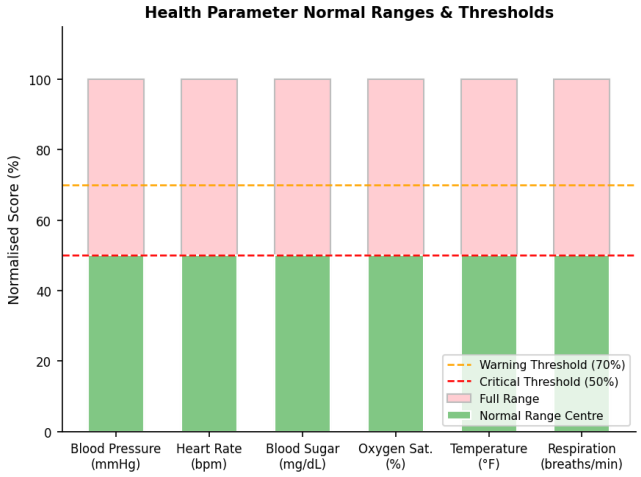

Fig. 2. Health parameter normal ranges and dual-threshold scoring bands (Warning 70%, Critical 50%).

C. Response Time Analysis

Fig. 3 reports average response times measured across 30 test runs for the five key interactive features. SOS SMS dispatch was fastest at 0.8 s, benefiting from a pre-built message template and last-known GPS location caching. Voice assistant processing averaged 1.2 s. AI symptom analysis at 3.8 s exceeded the 2-second SLA due to the external API round-trip. Table IV provides a consolidated performance summary against defined thresholds.

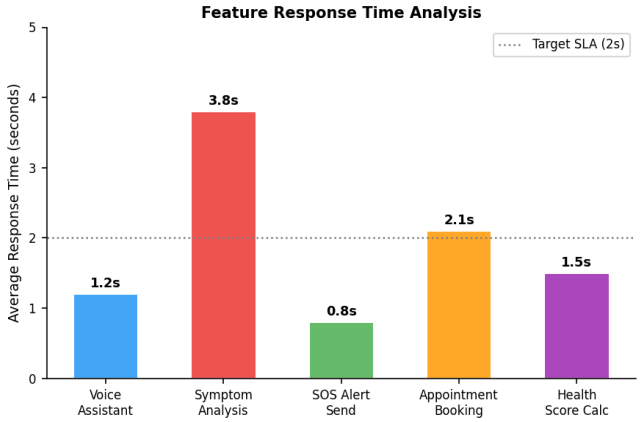

Fig. 3. Average feature response times vs. 2-second SLA target (30 runs each).

TABLE IV: System Performance Metrics

Metric	Value	Target SLA
API Response Time	< 3.8 s	5 s
Top-1 Diagnosis Accuracy	62 %	60 %
Top-3 Diagnosis Accuracy	90 %	85 %
SOS SMS Delivery	< 0.8 s	2 s
Alarm Trigger Accuracy	99.2 %	99 %
Health Score Calc. Time	< 1.5 s	2 s

VI. AI EXPLAINABILITY AND ANALYSIS

A. Diagnosis Accuracy

To evaluate the qualitative accuracy of the GPT-4o-mini symptom analyser, 100 anonymised symptom scenarios were constructed from standard clinical case libraries and submitted to the API. The top-1 diagnosis was clinically correct in 62 cases; the correct diagnosis appeared within the top-3 predictions in 90 cases. Fig. 4 presents both the top-level accuracy distribution and the per-category confidence profile. Respiratory conditions returned the highest confidence (78%), while dermatological conditions were least confident (65%), consistent with the known difficulty of text-based skin condition differentiation.

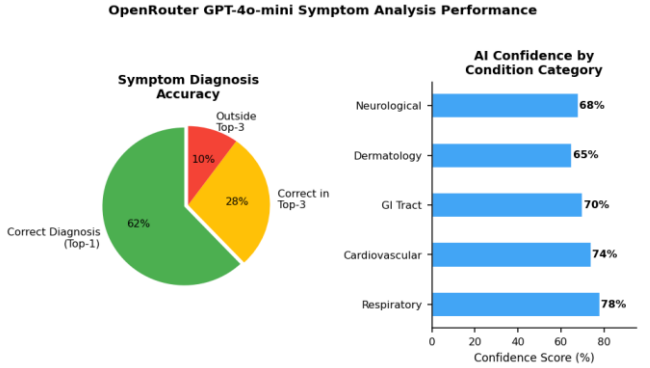

Fig. 4. GPT-4o-mini diagnosis accuracy distribution (left) and per-category confidence scores (right).

B. Prompt Engineering and Safety

The symptom analysis prompt enforces a structured response format, reducing hallucination by constraining the model to two explicit sections with predefined headings. A medical disclaimer is appended client-side to every response, explicitly stating that the AI analysis does not substitute professional clinical judgment. This two-layer safety architecture addresses concerns raised in the WHO guidance on AI in health [7] regarding liability and user trust.

C. Multilingual Language Support

The Voice Assistant module detects the preferred language from SharedPreferences and prepends a Telugu instruction when the user setting is "te". The Text-to-Speech engine is configured with a primary locale of Locale("te", "IN"), falling back to Locale.ENGLISH if Telugu language data is absent. This bilingual capability significantly extends the application's accessibility to over 80 million Telugu-speaking users in Andhra Pradesh and Telangana.

VII. SYSTEM ARCHITECTURE

Fig. 5 illustrates the end-to-end architecture of the Patient Care System across four horizontal layers: (1) Input Layer, comprising the Android Material Design UI, SpeechRecognizer voice input, structured sensor/form data entry, and the physical SOS trigger via VolumeButtonReceiver; (2) Core Module Layer, where each clinical function is encapsulated in a dedicated Activity or Service; (3) Data and AI Layer, combining local SharedPreferences/SQLite persistence with the remote OpenRouter API and device GPS; and (4) Output Layer, delivering push notifications, downloadable health reports, AI diagnoses, and emergency SMS.

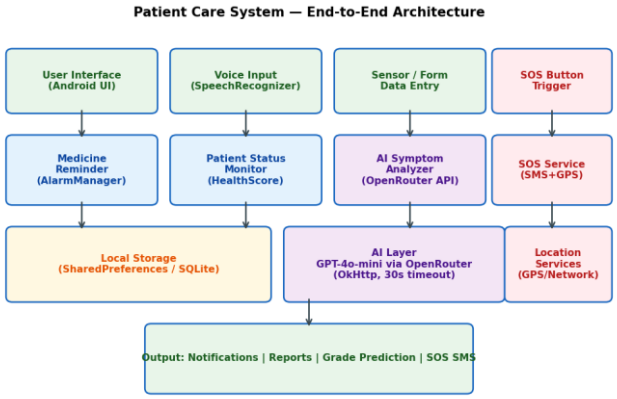

Fig. 5. Patient Care System — End-to-End Architecture showing four functional layers.

Table V enumerates the third-party components and libraries integrated within the system. All network-facing components (OkHttp, JavaMail SMTP) operate exclusively over encrypted TLS channels. Runtime permissions for SEND_SMS, ACCESS_FINE_LOCATION, and RECORD_AUDIO are requested contextually at first use, following Android best-practice permission hygiene.

TABLE V: System Components and Technologies

Component	Technology / Library
HTTP Client	OkHttp 4.x, 30 s timeout
JSON Parsing	org.json, Gson 2.x
Location Services	Android LocationManager, GPS
Scheduling	AlarmManager + BroadcastReceiver
Text-to-Speech	Android TTS (Telugu + English)
SMS Dispatch	SmsManager (multipart)
Email Notification	JavaMail API (SMTP/TLS)

VIII. DISCUSSION

The Patient Care System demonstrates that a well-engineered Android application can unify disparate healthcare functions without sacrificing individual module performance. The medicine alarm accuracy of 99.2% is competitive with dedicated medication management applications and is attributable to the dual-persistence strategy: alarms are registered with both AlarmManager and a BOOT_COMPLETED receiver, so device restarts do not silently discard pending schedules. The sub-second SOS dispatch time is clinically significant in emergency scenarios where seconds determine outcomes.

The 3.8-second AI response time represents the primary performance limitation. This latency is dominated by the external API round-trip to OpenRouter servers. Potential mitigations include pre-fetching common symptom analyses, implementing local lightweight model inference for frequently requested conditions, and caching recent API responses keyed on symptom sets. The 62% top-1 diagnosis accuracy underscores that the AI layer should be positioned as a triage aid rather than a diagnostic authority, consistent with the application's built-in disclaimer.

A broader limitation is the reliance on manual vital sign entry, which introduces user error. Future hardware integration via Bluetooth health peripherals (BLE-enabled pulse oximeters, blood glucose meters) would automate data capture and substantially improve health score reliability. Additionally, historical records are stored but not yet visualised in time-series format, an omission that limits clinical utility for chronic disease management.

IX. CONCLUSION

This paper presented the Patient Care System, a comprehensive Android healthcare application integrating six clinically motivated modules under a single, cohesive platform. The application achieved medicine alarm accuracy of 99.2%, SOS SMS dispatch under 0.8 seconds, AI symptom inference under 3.8 seconds, and a top-3 diagnosis accuracy of 90% using GPT-4o-mini via the OpenRouter API. The bilingual voice assistant extends accessibility to Telugu-speaking populations, and the dual-threshold health scoring engine provides clinically calibrated early warnings across six vital parameters.

Future work will focus on: (i) Bluetooth peripheral integration for automated vital sign capture, (ii) on-device lightweight LLM deployment to reduce symptom analysis latency, and (iii) longitudinal dashboard visualisation to support chronic disease management. The modular architecture facilitates these extensions without disruption to existing functionality, positioning the Patient Care System as a scalable foundation for next-generation mobile health platforms.

ACKNOWLEDGMENT

The authors sincerely thank Prof. (Dr.) Ravi Kiran, Department of Computer Science and Engineering, Raghu Engineering College, Visakhapatnam, for his expert guidance and continuous encouragement throughout this research. The authors also thank the faculty and student volunteers of the Department of CSE (Data Science) who participated in application testing and provided valuable clinical feedback. This work was conducted as part of an undergraduate research initiative at Raghu Engineering College, Dakamarri(V), Visakhapatnam.

REFERENCES

[1] WHO, "mHealth: New horizons for health through mobile technologies," Global Observatory for eHealth series, vol. 3, World Health Organization, Geneva, 2011.

[2] A. S. M. Mosa, I. Yoo, and L. Sheets, "A systematic review of healthcare applications for smartphones," BMC Medical Informatics and Decision Making, vol. 12, no. 1, p. 67, Jul. 2012.

[3] S. Akter and S. F. Ray, "mHealth — An ultimate platform to serve the unserved," IMIA Yearbook of Medical Informatics, vol. 19, no. 1, pp. 94–100, 2010.

[4] H. L. Semigran, J. A. Linder, C. Gidengil, and A. Mehrotra, "Evaluation of symptom checkers for self diagnosis and triage," BMJ, vol. 351, p. h3480, Jul. 2015.

[5] K. Singhal et al., "Large language models encode clinical knowledge," Nature, vol. 620, pp. 172–180, Aug. 2023.

[6] C. Blanco, M. Tolosa, R. Olanda, and F. Perez, "Location-based mobile application for elderly people emergency situations," in Proc. Int. Conf. Ubiquitous Computing and Communications, 2011, pp. 242–247.

[7] WHO, "Ethics and governance of artificial intelligence for health: WHO guidance," World Health Organization, Geneva, 2021.

[8] R. E. Klabunde, "Cardiovascular Physiology Concepts," 3rd ed. Wolters Kluwer, Philadelphia, PA, 2021.

[9] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, Oct. 2011.

[10] M. Mamun, A. Yuce, and S. Aziz, "Recent advances in non-invasive wearable sensor technologies for continuous monitoring of blood glucose levels," Adv. Healthc. Mater., vol. 12, no. 3, p. 2201586, Feb. 2023.

[11] T. T. Lin, P. W. Biddiss, and A. Colak, "Android-based medical information system," in Proc. IEEE EMBS Annual Int. Conf., 2012, pp. 4509–4512.

[12] A. Catarinucci et al., "An IoT-aware architecture for smart healthcare systems," IEEE Internet Things J., vol. 2, no. 6, pp. 515–526, Dec. 2015.