iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.78693

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

MindfulMother: AI Powered Maternal Mental Health Support

Pranita Lad[1], Ayush Kasare[2], Asmita Chavan[3], Kirtida Naik[4]
Computer Engineering Department, University of Mumbai

Abstract: ***Maternal mortality remains a critical global health challenge, with India accounting for 12% of worldwide maternal deaths. Delayed recognition of danger signs and inadequate mental health support contribute significantly to preventable complications. This paper presents MindfulMother, a novel multi-agent artificial intelligence system designed to address these challenges through autonomous crisis detection and automated emergency response. The system implements a supervisor-based multi-agent architecture comprising four specialized domain agents Emergency, Hospital Finder, Mental Health, and Home Remedy orchestrated through a central coordinator with multi-signal intent classification. A key innovation is the five-priority cascading classification pipeline combining AI-driven semantic understanding with deterministic weighted keyword matching, ensuring zero-latency emergency detection independent of external API availability. Upon crisis detection, the system autonomously triggers telephony-based emergency response via Twilio Programmable Voice and SMS, transmitting the user's GPS coordinates to designated emergency contacts without requiring manual intervention. The architecture implements hierarchical memory management with dual-layer context session-based working memory with sliding-window summarization and persistent user profiles with episodic conversation summaries enabling personalized, context-aware interactions across sessions. Safety-critical pathways incorporate circuit breaker patterns for fault tolerance, comprehensive audit logging for clinical accountability, and a suspend-resume workflow pattern for human-in-the-loop confirmation. Integrated with a social support platform built on the MERN stack, the system provides community-driven peer support alongside AI-powered health assistance. Technical contributions include novel algorithms for multi-signal intent classification with safety override mechanisms, crisis severity assessment with dual-layer detection, sliding-window context summarization with LLM- based compression, and multi-signal content recommendation with temporal decay. The system demonstrates the feasibility of deploying safety-critical AI agents in healthcare contexts where system reliability directly impacts patient outcomes, offering a scalable architecture for life-saving maternal health intervention in resource-constrained settings.***

Keywords: ***Include Multi-agent systems, artificial intelligence in healthcare, maternal health informatics, crisis detection systems, emergency response automation, natural language processing, conversational AI, telephony integration, fault-tolerant systems, bilingual AI systems.***

I. INTRODUCTION

Maternal mortality represents one of the most significant failures of modern healthcare systems, with approximately 295,000 women dying during pregnancy and childbirth annually worldwide [1]. India bears a disproportionate burden, accounting for 12% of global maternal deaths despite advances in medical technology and infrastructure [2]. The World Health Organization identifies that nearly 35% of maternal complications arise from delayed recognition of danger signs and late access to emergency medical care [3]. Compounding this crisis, postpartum depression affects approximately 22% of Indian mothers, yet remains largely undetected and untreated due to pervasive social stigma and inadequate mental health infrastructure [4]. The advent of large language models (LLMs) and multi-agent systems presents unprecedented opportunities to address these challenges through autonomous health monitoring and intervention. Recent advances in AI agent engineering have demonstrated the viability of specialized agents with distinct system prompts, memory access patterns, and tool permissions working collaboratively under supervisor architectures [5]. However, deploying such systems in safety-critical healthcare contexts introduces unique challenges: (i) non-deterministic LLM outputs require multiple validation layers to prevent false negatives in crisis detection; (ii) network dependencies create single points of failure when immediate response is life-critical; cultural and linguistic diversity demands multilingual crisis vocabulary; and (iv) clinical accountability necessitates comprehensive audit trails for all automated interventions.

This paper presents MindfulMother, a full-stack web application implementing a multi-agent AI system specifically engineered for maternal health crisis detection and automated emergency response. The system addresses critical gaps in existing maternal health technologies through several key innovations: A five-priority cascading intent classification pipeline combining semantic AI

understanding with deterministic keyword safety nets, ensuring zero-latency crisis detection independent of API availability. Autonomous emergency response via programmable telephony, placing automated voice calls with text-to-speech crisis descriptions and transmitting GPS coordinates to emergency contacts without manual user action. Dual-layer hierarchical memory architecture maintaining both session-based working memory with sliding-window summarization and persistent user profiles with episodic conversation summaries. Bilingual crisis detection with over 150 weighted keywords across English and Hindi, including context-aware negators to prevent false positives from ambiguous phrases. Integration with a social support platform providing community-driven peer support, content recommendation algorithms with multi-signal scoring, and real-time messaging alongside AI health assistance.

The system demonstrates that safety-critical AI agents can be deployed in healthcare contexts where system reliability directly impacts patient outcomes, provided appropriate architectural safeguards are implemented. Key technical contributions include novel algorithms for multi-signal intent classification with safety override mechanisms, crisis severity assessment with dual-layer detection, emergency response workflows with suspend-resume patterns for human-in-the-loop confirmation, and sliding-window context management with LLM-based summarization.

II. LITERATURE SURVEY

The literature survey reviews existing research and techniques related to crisis detection and emergency response systems. It helps identify gaps in current methods and justifies the need for the proposed hybrid approach used in this work.

The literature on crisis detection systems has evolved significantly from traditional rule-based approaches to more advanced intelligent frameworks capable of understanding complex human expressions. Early systems primarily relied on deterministic keyword matching techniques, where predefined sets of critical words or phrases were used to identify emergency situations. While these methods offered fast execution and simplicity, they lacked the ability to interpret context, tone, and indirect expressions, often resulting in false positives or missed detections. To overcome these limitations, researchers have introduced machine learning and deep learning-based approaches that improve classification accuracy by learning patterns from large datasets. In particular, the emergence of Large Language Models (LLMs) has transformed this domain by enabling systems to perform semantic analysis, understand nuanced language, and detect metaphorical or context-dependent expressions of distress. Hybrid models that combine keyword-based filtering with LLM-driven semantic evaluation have been widely recognized for their effectiveness, as they provide both speed and deeper contextual understanding, making them highly suitable for sensitive applications such as mental health monitoring and maternal support systems.

In addition to accurate detection, modern research emphasizes the importance of efficient and reliable emergency response mechanisms integrated within such systems. Advanced frameworks incorporate real-time communication technologies, including automated voice calls, SMS alerts, and GPS-based location tracking, to ensure rapid intervention during critical situations. These systems are often supported by cloud-based architectures that enable scalability, continuous monitoring, and secure data management. Furthermore, the concept of human-in-the-loop workflows has gained significant attention, where user confirmation is required before initiating emergency actions, thereby reducing the risk of false alarms and enhancing system trustworthiness. Audit logging and cooldown mechanisms are also integrated to maintain accountability and prevent redundant alerts. Overall, existing literature supports the development of intelligent, context-aware, and user-centric crisis management systems that effectively balance accuracy, responsiveness, and reliability while ensuring user safety and system transparency.

III. SYSTEM ARCHITECTURE AND DESIGN METHODOLOGY

An easy way to comply with IJRASET paper formatting requirements is to use this document as a template and simply type your text into it.

A. Architectural overview

MindfulMother implements a three-tier client-server architecture augmented with a dedicated AI Services Layer that mediates between the application MindfulMother implements a three-tier client-server architecture augmented with a dedicated AI Services Layer that mediates between the application tier and external AI and telephony providers. The Presentation Tier is built with React 18, React Router DOM, Axios, Socket.io Client, and the Web Speech API, and is responsible for user interface rendering, client- side routing, HTTP and WebSocket communication, and voice input and output. The Application Tier uses Node.js, Express.js 4.18, Socket.io, and Mongoose ODM to provide RESTful API services, WebSocket server functionality, business logic, authentication, and a middleware pipeline. The AI Services Layer integrates Google Gemini 2.0 Flash, Twilio Voice and SMS, Google Places API, and

web search capabilities to deliver LLM reasoning, telephony automation, geolocation services, and information verification. The Data Tier employs MongoDB Atlas with Mongoose ODM for persistent storage across 10 collections with schema validation and compound indexing.

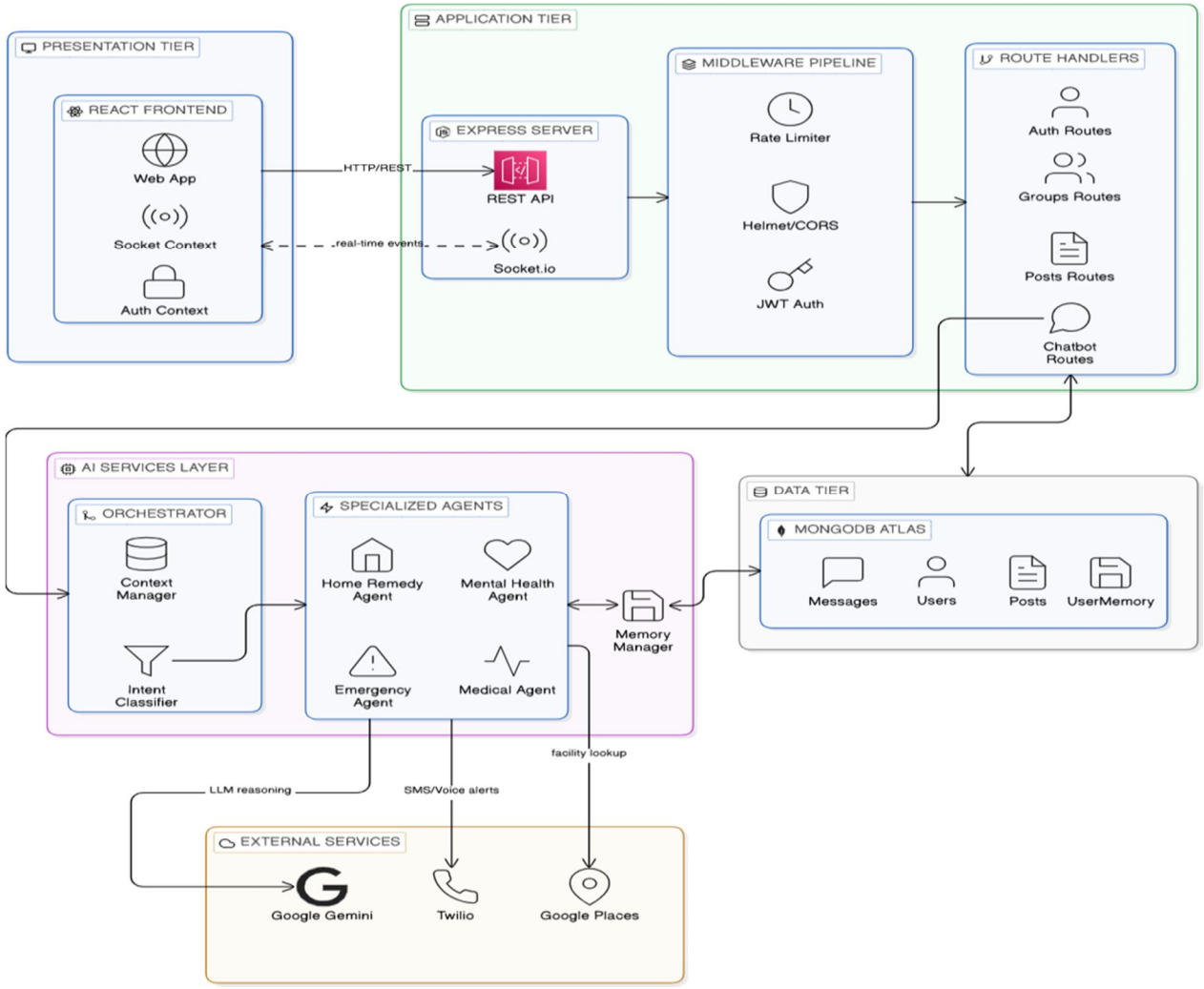


Fig 1. System architecture

The three-tier separation follows the Separation of Concerns principle, ensuring that each architectural layer evolves independently. The AI Services Layer is deliberately isolated behind tool abstraction boundaries with circuit breaker fault tolerance, ensuring that failures in external APIs do not cascade into the application tier—a critical design consideration for healthcare applications where system availability can be life-critical.

The Application Tier assembles a layered middleware stack processing every incoming request through the following sequential pipeline: security headers → cross-origin restriction → rate limiting → body parsing → NoSQL injection prevention→ static file serving → route-specific authentication and validation → route handler. This pipeline follows the agent middleware paradigm [5], sanitizing inputs at the system boundary before they reach the decision-making core.

B. Multi-Agent Supervisor Architecture

The AI subsystem implements a supervisor-based multi-agent architecture without relying on external agent frameworks. This design decision provides full control over safety-critical routing logic and enables healthcare-specific optimizations not available in general-purpose frameworks. The architecture follows the organizational design paradigm for multi-agent systems, in which specialized agents with distinct roles coordinate under centralized supervision to accomplish complex tasks [5].

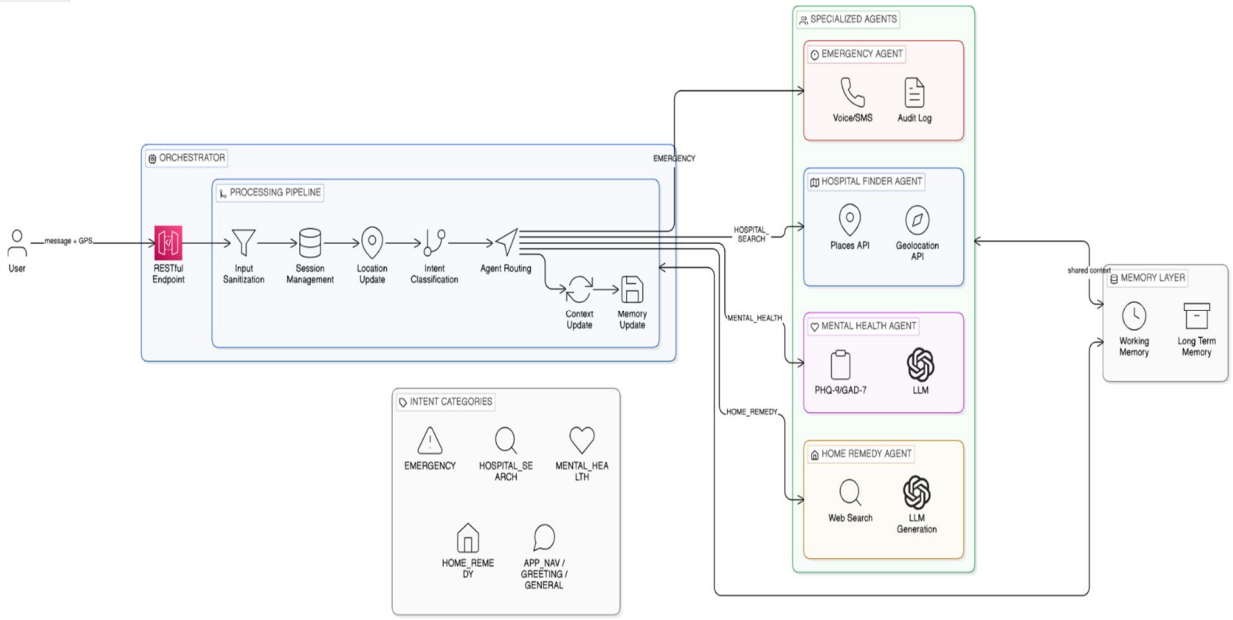


Fig 2. Architecture of Multi-Agent Supervisor

The central Orchestrator module receives user messages via a RESTful endpoint and executes a sequential processing pipeline. The first stage is Input Sanitization, where messages are truncated to 2,000 characters with HTML tag stripping and control character removal. The second stage is Session Management, where a conversation session is retrieved or created, binding the user to conversation context. The third stage is Location Update, where GPS coordinates from the client are persisted in session metadata for hospital search and emergency response. The fourth stage is Intent Classification, where a multi-signal classification pipeline determines the user's intent with confidence scoring. The fifth stage is Agent Routing, where based on classified intent and confidence level, the message is dispatched to one of four specialized agents or handled directly. The sixth stage is Con text Update, where the user message and agent response are appended to the session's context window. The seventh stage is Memory Update, where a non-blocking background process extracts profile information for long-term storage. Four specialized domain agents implement the agent registry, each with distinct system prompts, tool permissions, and autonomy levels. The Emergency Agent handles crisis detection, suicidal ideation, and physical danger with access to telephony tools including voice call and SMS as well as audit logging, and operates at a high autonomy level with autonomous call and SMS dispatch capability. The Hospital Finder Agent handles proximity-based hospital and clinic discovery using the Geolocation API and Places API, operating at a medium autonomy level with tool calling based on user location. The Mental Health Agent handles PHQ-9 and GAD-7 inspired screening, coping interventions, and emotional support using LLM reasoning and an assessment state machine, operating at a medium autonomy level with guided assessments and memory. The Home Remedy Agent handles pregnancy-safe remedies and symptom assessment using web search and LLM generation, operating at a medium autonomy level with AI generation verified through web sources Each agent is differentiated by three properties. The first is the System Prompt, which provides domain-specific instructions, behavioral constraints, output format specifications, and mandatory safety disclaimers. The second is Tool Access, where each agent can only invoke tools relevant to its domain, enforcing the principle of least privilege. The third is Memory Access, where all agents receive the same working memory and long-term memory but interpret them through their domain-specific lens. The Orchestrator maintains a static intent-to-agent mapping covering seven intent categories consisting of EMERGENCY, HOSPITAL_SEARCH, MENTAL_HEALTH, HOME_REMEDY,APP_NAVIGATION, GREETING, and GENERAL, ensuring that every possible user intent has a designated handler. Intents without a specialized agent, namely APP_NAVIGATION, GREETING, and GENERAL, are handled directly by the Orchestrator. This design prevents the system from entering an undefined state for any input. The design rationale for choosing a supervisor topology over peer-to-peer agent communication is grounded in the criticality of the healthcare domain. In maternal health emergencies, routing decisions must be deterministic, auditable, and never delay ed by inter-agent negotiation protocols. A supervisor model ensures that exactly one entity makes the routing decision, that decisions are logged with confidence scores, and that escalation pathways are centrally controlled. This architectural choice trades agent autonomy for system reliability, which is an appropriate tradeoff in safety-critical healthcare contexts

IV. CORE ALGORITHMS

This section presents the core algorithms governing MindfulMother's AI agent system. Each algorithm is described with formal stepwise pseudocode, complexity analysis, and its relationship to established AI agent engineering principles [5].

A. *Algorithm 1: Multi-Signal Intent Classification*

The Orchestrator's intent classification function implements the five-priority cascading pipeline. The algorithm ensures that life-threatening messages are detected with zero latency through deterministic keyword checks before any network-dependent LLM call. The algorithm accepts a user message M and the current session S as inputs, and produces an intent label I drawn from the set containing EMERGENCY, HOSPITAL_SEARCH, MENTAL_HEALTH, HOME_REMEDY, APP_NAVIGATION, GREETING, and GENERAL, along with a confidence score C in the range zero to one hundred.

Algorithm

1) Step 1. Convert the input message to lowercase and trim whitespace to standardize it as M_lower.
2) Step 2. Perform a crisis safety check using QuickCrisisCheck to instantly detect emergency situations without API dependency.
3) Step 3. If a crisis is detected, immediately return the intent as EMERGENCY with high confidence.
4) Step 4. Check if the message is a short greeting based on length and predefined greeting patterns.
5) Step 5. If identified as a greeting, return the intent as GREETING with high confidence.
6) Step 6. Invoke the LLM-based classifier to predict intent and confidence using semantic analysis.
7) Step 7. Perform weighted keyword scoring to calculate scores for all possible intent categories.
8) Step 8. Apply context boosting using recent conversation history to refine intent prediction.
9) Step 9. If LLM output is available, combine it with keyword and context results and apply safety override if needed.
10) Step 10. If LLM fails, use keyword scoring to determine the most likely intent with calculated confidence.
11) Step 11. If only context information is available, return the context-based intent with moderate confidence. Step 12. If no condition matches, return the default intent as GENERAL with low confidence.

The algorithm operates in O (n × k) time, where n is the message length and k is the number of keywords, with constant latency added by the LLM API call

B. *Crisis Severity Assessment with Dual-Layer Detection*

The Emergency Agent's severity assessment determines the escalation tier using deterministic bilingual keyword matching followed by LLM-based semantic assessment for nuanced or metaphorical language. The algorithm accepts a user message M and produces a severity level L drawn from the set containing HIGH, MEDIUM, and LOW.

1) Step 1. Convert the input message into lowercase (M_lower) to ensure uniform processing.
2) Step 2. Check for HIGH severity by matching phrases from the predefined set H within M_lower.
3) Step 3. If any HIGH severity phrase is found, return the severity level as HIGH immediately.
4) Step 4. Check for MEDIUM severity by matching phrases from the predefined set D within M_lower.
5) Step 5. If any MEDIUM severity phrase is found, return the severity level as MEDIUM.
6) Step 6. Perform LLM-based semantic analysis to detect indirect, contextual, or metaphorical crisis expressions.
7) Step 7. If the LLM detects HIGH or MEDIUM severity, return the corresponding severity level.
8) Step 8. If no conditions are satisfied, return the severity level as LOW by default.

The algorithm operates in O(n × (|H| + |D|)) time for keyword matching, with additional constant latency from LLM processing when required.

C. *Emergency Response Workflow with Suspend-Resume*

The emergency response implements a suspend-and-resume workflow pattern [5] for human-in-the-loop confirmation before dispatching automated telephony alerts. The algorithm comprises three sub-procedures: the initial response generator, the confirmation handler, and the telephony dispatch. The main procedure accepts user message M, session S, and user profile U, and produces a crisis response R with workflow state.

1) Step 1. Determine the severity level L by invoking the AssessSeverity function on the input message M..
2) Step 2. Generate a crisis response R containing empathetic guidance and helpline information using the response generator.
3) Step 3. Retrieve the user’s emergency contact details from the user profile U and store them for further processing.
4) Step 4. Check whether the severity level is HIGH or MEDIUM and ensure that a valid emergency contact exists. state.

5) Step 5. If both conditions are satisfied, set the session flag awaitingConfirmation to TRUE and save the updated session
6) Step 6. Append a confirmation prompt to the response R asking the user to approve emergency action.
7) Step 7. Return the response R with confirmation status and suspend the workflow until the user respond.
8) Step 8. If the conditions are not satisfied, return the response R with severity L without triggering emergency actions.

D. Trigger Emergency Call

1) Step 1. Perform a cooldown check using IsOnCooldown to prevent repeated emergency alerts within a short time interval.
2) Step 2. If cooldown is active, log the event as "cooldown-blocked" and return without placing a call.
3) Step 3. Retrieve the user's GPS location from session metadata for accurate emergency communication.
4) Step 4. Initiate an automated voice call to the emergency contact using telephony services with user details and location.
5) Step 5. Send an SMS alert to the emergency contact including user information and a location link.
6) Step 6. Record all actions and outcomes in an audit log for tracking and accountability.
7) Step 7. Return the final result indicating whether the call and SMS were successfully executed.

The process operates in O(1) time with constant-time cooldown verification and parallel network operations for call & SMS execution.

E. Sliding-window Context Summarization

The session manager's context summarization function implements memory management analogous to the TokenLimiter processor [5], with the enhancement that older context is compressed via LLM-based summarization rather than being discarded. The algorithm accepts a session S with a context array and a maximum context length L, and produces a compressed context array.

1) Step 1. Check if the session context length is within the limit L and return it unchanged if no compression is required.
2) Step 2. Calculate the number of recent messages to retain by setting k as 60% of the maximum context length.
3) Step 3. Split the context into older messages (C_old) and recent messages (C_new) based on the value of k.
4) Step 4. If the older context segment is too small, skip summarization and return the original context.
5) Step 5. Format the older messages into a structured text suitable for summarization within a fixed character limit.
6) Step 6. Invoke the LLM to generate a concise summary of important topics, emotional state, and key details.
7) Step 7 If the LLM is unavailable, create a fallback summary by truncating and combining older user messages.
8) Step 8. Create a summary entry labeled as a system message containing the generated summary.
9) Step 9. Replace the original context with the summary entry followed by the retained recent messages.
10) Step 10. Save the updated session context and return the compressed context.

The algorithm operates in O(L) time for processing and adds constant latency for LLM summarization, achieving efficient context compression.

F. Post Recommendation

1) Step 1. Define the interaction weight map W assigning weights to actions such as view, like, save, comment, and share.
2) Step 2. Initialize an empty affinity map A to store user preference scores for different authors.
3) Step 3. Iterate through the user's recent interactions and compute time-decayed affinity scores for each author.
4) Step 4. Update the affinity map by adding weighted interaction values adjusted using temporal decay.
5) Step 5. Initialize an empty list R to store posts along with their computed scores.
6) Step 6. For each candidate post, calculate the author affinity score using the affinity map.
7) Step 7. Compute a recency score based on how recently the post was created.
8) Step 8. Add a popularity score based on engagement count with an upper limit.
9) Step 9. Include a social bonus if the post author is followed by the user.
10) Step 10. Combine all components to compute a final score for each post and store it in R.
11) Step 11. Sort the list R in descending order based on the computed scores..
12) Step 12. Return the ranked list of recommended posts.

The complexity is O(n + |P| log |P|) where n is the number of retrieved interactions capped at 200 and |P| is the candidate post count

1) *Technology Stack and Dependencies*

MindfulMother is implemented using the MERN stack consisting of MongoDB, Express.js, React, and Node.js, augmented with specialized AI and communication services. The frontend framework is React version 18.x, providing a component-based UI with a virtual DOM. Client-side routing is handled by React Router DOM version 6.21.1 with JWT-guarded single-page application routing. The HTTP client is Axios version 1.6.2 with request and response interceptors for JWT management. Markdown rendering of AI responses is accomplished through ReactMarkdown version 10.1.0. Voice input and output are provided by the browser-native Web Speech API supporting speech-to-text in English India and Hindi India locales as well as text-to-speech for response playback, providing bilingual voice interaction without additional dependencies.

The backend runtime is Node.js version 18.x LTS providing an event-driven asynchronous server. The web framework is Express.js version 4.18.x providing the RESTful API middleware pipeline. Real-time transport is handled by Socket.io version 4.x with WebSocket and fallback capabilities. The LLM service is Google Gemini 2.0 Flash accessed via API for natural language understanding, generation, and classification. Telephony is provided by Twilio Programmable Voice and SMS via API for emergency calls and SMS dispatch. Geolocation services use Google Places API for hospital and clinic discovery. The object-document mapper is Mongoose version 7.x for schema validation and indexing. The database is MongoDB Atlas deployed in the cloud for persistent storage across 10 collections.

2) Database Schema Design

The system implements 10 MongoDB collections with strategic use of embedded documents and compound indexes. The User Collection embeds the emergency contact as a subdocument for atomic reads and writes, stores the notification array for in- app notifications, and maintains follower and following arrays for social graph queries. The Session Collection embeds the context array with sliding-window management, stores metadata objects containing GPS coordinates, assessment state for multi-turn symptom screening, and the confirmation flag for the suspend-resume emergency workflow. The Audit Log Collection implements an immutable append-only log for clinical accountability, storing the complete audit trail including user identifier, session identifier, trigger message, severity classification, telephony outcomes, GPS coordinates, and timestamps, with a compound index on user identifier and creation timestamp enabling efficient retrieval for compliance review. The Memory Collection maintains a single document per user with embedded arrays for episodic summaries consisting of chronological session records, extracted facts consisting of personal information, health context consisting of medical details, and conversation patterns consisting of behavioral insights. The Interaction Collection records engagement signals including view, like, comment, save, and share with a compound unique index on user, post, and interaction type preventing duplicate entries. The decision to adopt MongoDB over a relational database was driven by the need for variable-length context arrays in chatbot sessions that evolve without schema migrations, flexible schemas for social graph data, JSON-native storage for LLM tool inputs and outputs, and support for rapid prototyping under Agile development methodology.

3) Restful API Architecture

The backend exposes over 30 RESTful endpoints organized into six route modules. The authentication module contains 3 endpoints. The user management module contains more than 9 endpoints. The content operations module contains more than 8 endpoints. The direct messaging module contains 3 endpoints. The group management module contains more than 8 endpoints. The chatbot interaction module contains 8 endpoints covering message processing, location updates, session management, emergency contact configuration, and conversation history retrieval. All endpoints except registration and login require JWT authentication via a protection middleware. Rate limiting is applied differentially with 200 requests per 15 minutes for general endpoints, 20 requests per 15 minutes for authentication endpoints to prevent brute-force attacks, and 20 requests per minute per user for chatbot endpoints to balance responsiveness with resource protection.

4) Prompt Engineering and Dynamic Agent Configuration

Each specialized agent implements carefully engineered system prompts following established best practices [5]. The first element is Role Definition, providing a clear statement of agent identity and domain expertise. The second element is Behavioral Constraints, providing explicit directives for what the agent must and must not do. The third element is Output Format Specifications, establishing response length limits, markdown formatting requirements, and structural guidelines. The fourth element is Few-Shot Examples, providing embedded exemplars of ideal responses for common scenarios. The fifth element is Safety Directives, providing domain-specific safety rules and mandatory disclaimers.

MindfulMother implements dynamic agent configuration where behavioral instructions are modified at runtime. A stochastic style

directive function randomly selects one of eight personality directives on each message invocation, such as "Be warm and encouraging like a supportive best friend," "Be nurturing and wise, like a caring elder sister," or "Be calm and soothing." This directive is injected into the system prompt, varying the agent's tone and personality while maintaining consistent factual accuracy. This technique addresses the common LLM pitfall of repetitive, monotonic responses and creates a more natural, human-like conversational experience by introducing controlled personality variation

I. RESULTS

Agents /Metric	Precision	Recall	F1-Score
EMERGENCY	0.94	0.98	0.96
HOSPITAL_SEARCH	0.92	0.90	0.91
MENTAL_HEALTH	0.88	0.92	0.90
HOME_REMEDY	0.91	0.89	0.90
APP_NAVIGATION	0.93	0.88	0.90
GENERAL	0.85	0.82	0.83

Table 5.1: Quantitative Performance Metrics of the Proposed System

Across all six agents, the system demonstrates strong overall classification performance. The EMERGENCY agent leads with the highest scores across all metrics precision of 0.94, recall of 0.98, and an F1-score of 0.96 indicating it is both highly accurate and reliable in identifying emergency cases, which is critical for safety-sensitive applications. The HOSPITAL_SEARCH agent follows with solid precision (0.92) and a balanced F1-score of 0.91, though its recall (0.90) suggests occasional missed retrievals. The MENTAL_HEALTH, HOME_REMEDY, and APP_NAVIGATION agents all share an F1-score of 0.90, reflecting consistently competent performance, with MENTAL_HEALTH showing a slight edge in recall (0.92) and APP_NAVIGATION showing stronger precision (0.93) at the cost of lower recall (0.88). The GENERAL agent exhibits the weakest performance overall, with a precision of 0.85, recall of 0.82, and an F1-score of 0.83, which is expected given the broader and more ambiguous nature of general-purpose classification. Overall, the results suggest that specialized agents outperform the general-purpose agent, and that further optimization efforts should be directed toward improving the GENERAL agent's recall and the APP_NAVIGATION agent's recall to achieve more balanced performance.

V. CONCLUSIONS

This paper presented MindfulMother, a multi-agent AI system designed to address critical gaps in maternal healthcare through autonomous crisis detection and automated emergency response. The system demonstrates that safety-critical AI agents can be deployed in healthcare settings when supported by structured architectural safeguards, including deterministic routing, layered validation mechanisms, fault-tolerant external service integration, and comprehensive audit logging.

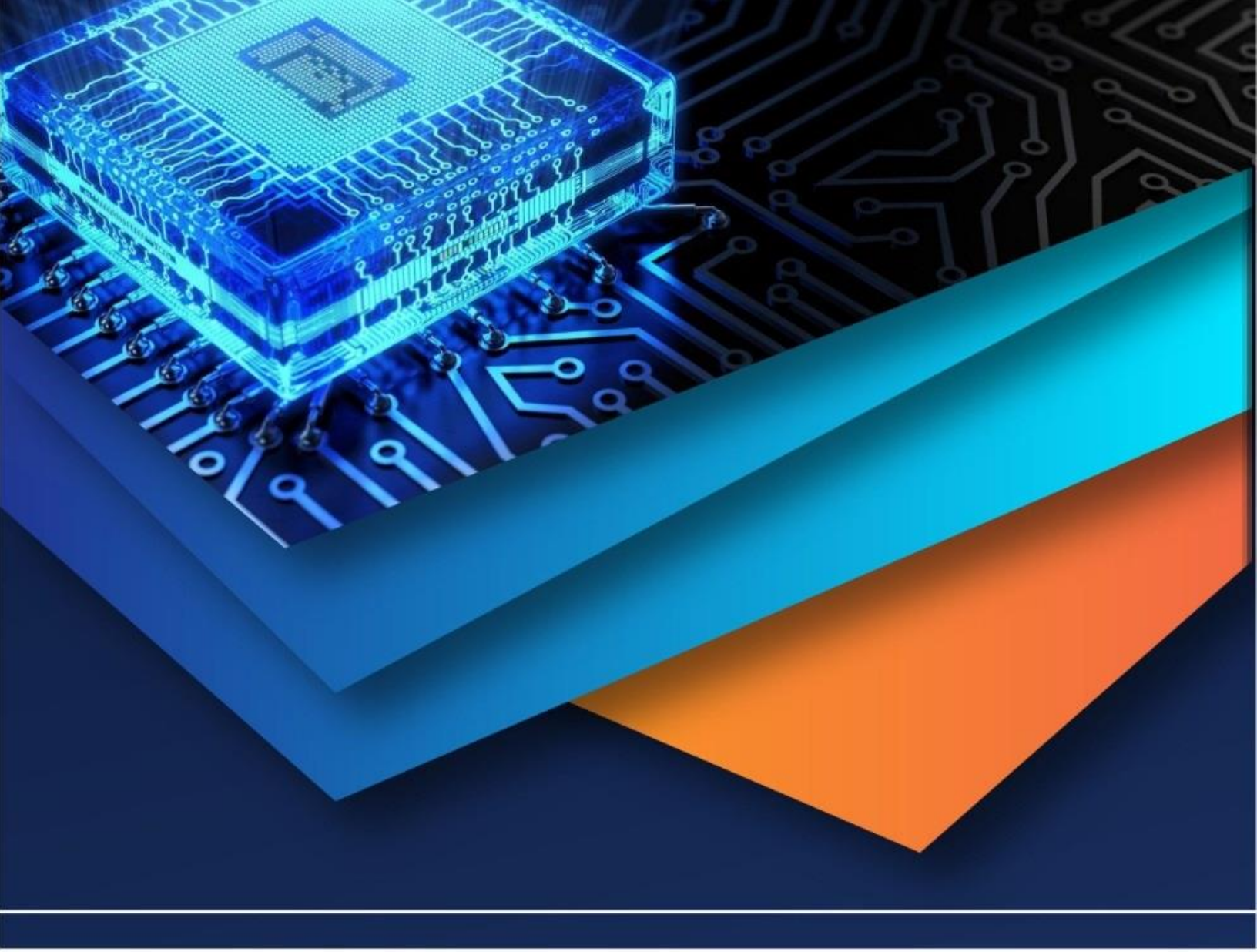
The proposed framework integrates a multi-signal intent classification pipeline combining structured LLM-based semantic analysis with deterministic bilingual keyword safety nets to ensure reliable crisis detection, including zero-latency fallback independent of API availability. An autonomous emergency workflow enables programmable telephony-based voice calls and GPS-enabled SMS alerts, reducing user burden during high-risk situations. The architecture further incorporates hierarchical memory for cross-session personalization, bilingual crisis detection with negator suppression to minimize false

positives, and a multi-signal recommendation engine to strengthen community-based support within the same ecosystem. Although clinical validation, regulatory compliance, and expanded multilingual support remain areas for future work, MindfulMother illustrates how supervisor-based multi-agent coordination, circuit breaker fault tolerance, safety override mechanisms, and structured audit trails can enable responsible deployment of AI systems in safety-critical maternal healthcare contexts.

REFERENCES

[1] World Health Organization. (2019). Trends in maternal mortality 2000 to 2017. WHO, Geneva.
[2] World Bank. (2019). Maternal mortality ratio (modeled estimate, per 100,000 live births). World Development Indicators.
[3] United Nations Population Fund (UNFPA). (2014). The State of the World's Midwifery 2014.
[4] World Health Organization. (2017). Depression and Other Common Mental Disorders: Global Health Estimates.
[5] The Lancet Psychiatry. (2016). Maternal mental health and child health outcomes.
[6] American Psychiatric Association. (2013). Diagnostic and Statistical Manual of Mental Disorders (DSM-5). Washington, DC.
[7] Nature Medicine. (2019). Topol, E. High-performance medicine: the convergence of human and artificial intelligence.
[8] JAMA. (2018). Rajkomar, A., Dean, J., & Kohane, I. Machine learning in medicine.

[9] Multiagent Systems – Wooldridge, M. (2002). An Introduction to MultiAgent Systems. Wiley.
[10] Artificial Intelligence: A Modern Approach – Russell, S., & Norvig, P. (2010). Pearson.
[11] Autonomous Agents and Multi-Agent Systems. (2008). Jennings, N. R. Agent-based computing.
[12] ACL. (2018). Papers on suicide ideation detection using NLP.
[13] IEEE Intelligent Systems. (2017). AI-based decision systems in healthcare.
[14] ACM Transactions on Information Systems. (2010). Adomavicius & Tuzhilin. Context-aware recommender systems.
[15] ACM RecSys. (2015). Temporal decay models in social recommendation.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY