AyurMind: A Multi-Agent RAG Framework for Intelligent Ayurvedic Prakriti Assessment and Personalized Wellness Recommendations

Research Proposal for MBCC 2026

Conference Tracks: Ayurveda • Natural Language Processing for Indian Languages • Cognitive Science and AI

Executive Summary

Problem: Ayurvedic diagnosis requires rare expert practitioners, and current AI systems hallucinate incorrect medical advice from ancient texts, making traditional knowledge inaccessible and unreliable for modern healthcare applications.

Solution: We create an intelligent multi-agent system that accurately retrieves and reasons over Ayurvedic texts to provide personalized Prakriti assessment and treatment recommendations without hallucinations.

Method: Using specialized RAG-powered agents (Prakriti assessor, Dosha analyzer, Treatment recommender) coordinated by an orchestrator, all running on open-source Llama-3 models with ChromaDB vector storage of digitized Ayurvedic scriptures.

1. Research Problem

1.1 Background

Ayurveda, a 5,000-year-old system of medicine rooted in Indian Knowledge Systems (IKS), offers holistic approaches to health and wellness. The core principle of Ayurvedic diagnosis involves determining an individual's Prakriti (constitutional type) and Vikriti (current imbalances) to provide personalized treatment recommendations.

However, accessing authentic Ayurvedic knowledge faces several critical challenges:

- **Scarcity of Expert Practitioners:** Qualified Ayurvedic doctors are limited, especially in rural areas, creating accessibility barriers for millions seeking traditional healthcare.
- **AI Hallucination Problem:** Large Language Models (LLMs) like GPT-4 and Claude generate plausible-sounding but factually incorrect Ayurvedic advice, potentially causing harm when users follow inaccurate recommendations.
- **Knowledge Preservation:** Ancient Ayurvedic texts (Charaka Samhita, Sushruta Samhita, Ashtanga Hridaya) contain invaluable wisdom but remain locked in Sanskrit and classical languages, inaccessible to modern practitioners and patients.

- **Limited AI Integration:** Existing Ayurvedic apps use rigid rule-based systems that lack the nuanced reasoning required for personalized diagnosis and treatment planning.

1.2 Research Gap

No existing system combines the retrieval accuracy of RAG (Retrieval-Augmented Generation) with the specialized reasoning capabilities of multi-agent architectures for Ayurvedic diagnosis. Current approaches either suffer from hallucinations (pure LLMs) or lack flexibility (rule-based systems).

2. Proposed Solution: AyurMind

2.1 System Overview

AyurMind is a multi-agent RAG framework where specialized AI agents collaborate to provide holistic Ayurvedic consultations. Each agent is empowered with retrieval capabilities from authenticated Ayurvedic texts, ensuring factual accuracy while maintaining the reasoning flexibility of modern LLMs.

2.2 Architecture

The system consists of four specialized agents:

1. **Prakriti Assessor Agent:** Determines the user's constitutional type (Vata, Pitta, Kapha, or combinations) by analyzing physical characteristics, behavioral patterns, and psychological traits through guided questioning.
2. **Dosha Imbalance Detector Agent:** Identifies current health imbalances (Vikriti) by correlating reported symptoms with disease causation patterns documented in classical texts.
3. **Treatment Recommender Agent:** Suggests personalized interventions including dietary modifications, lifestyle changes, herbal remedies, and therapeutic practices based on Prakriti-Vikriti analysis.
4. **Orchestrator Agent:** Coordinates agent interactions, manages conversation flow, synthesizes recommendations from specialist agents, and presents holistic guidance to users.

Key Innovation: Unlike single-agent systems, our multi-agent architecture mirrors the collaborative diagnostic process in traditional Ayurvedic practice, where different aspects of health are analyzed independently before synthesis.

3. System Architecture

The AyurMind system employs a layered architecture that separates concerns while enabling seamless integration between components. This section presents two complementary views of the system: the structural architecture showing component relationships, and the operational data flow showing runtime behavior.

3.1 Architectural Overview

The system architecture follows a six-layer design pattern, progressing from user interaction at the top to knowledge storage at the bottom. Each layer has well-defined responsibilities and interfaces:

1. **User Interface Layer:** Provides bilingual (English/Hindi) web interface via Gradio, handles user input/output, manages conversation state.
2. **Orchestration Layer:** Central coordinator that decomposes queries, delegates to specialist agents, and synthesizes responses into holistic recommendations.
3. **Specialized Agent Layer:** Three domain-specific agents (Prakriti Assessor, Dosha Detector, Treatment Recommender) that perform focused reasoning with retrieved context.
4. **RAG Layer:** Implements semantic retrieval pipeline using LangChain for document processing, sentence-transformers for embeddings, and ChromaDB for vector storage.
5. **Language Model Layer:** Hosts Llama-3-8B (or Mistral-7B) via Ollama for local inference, shared by orchestrator and all specialist agents.
6. **Knowledge Base Layer:** Contains processed and indexed Ayurvedic classical texts (Charaka Samhita, Sushruta Samhita, Ashtanga Hridaya) organized by topic and chapter.

Link to architecture I was working to shorten it currently access the architecture from here : (Architecture image)

The architecture diagram illustrates component relationships and data pathways. Solid arrows represent primary data flow (queries, responses, retrieved content), while dashed arrows indicate LLM API calls. Color coding distinguishes functional layers: blue for user interface, orange for orchestration, green for specialized agents, purple for RAG infrastructure, pink for language models, and yellow for knowledge repositories.

3.2 Operational Data Flow

When a user submits a health query, the system executes a four-phase workflow orchestrated by the central coordination agent:

Phase 1 - Constitutional Assessment:

The Prakriti Assessor Agent analyzes physical and behavioral characteristics to determine the user's constitutional type. It queries the RAG pipeline for relevant text sections describing Vata, Pitta, and Kapha traits, then combines retrieved context with user information to generate a Prakriti classification. This assessment provides the foundation for personalized recommendations in subsequent phases.

Phase 2 - Imbalance Detection:

The Dosha Imbalance Detector correlates reported symptoms with disease causation patterns documented in classical texts. It retrieves sections on pathology, symptom manifestations, and diagnostic criteria. By synthesizing this knowledge with

the user's symptom profile and constitutional type, the agent identifies current Dosha imbalances (Vikriti) and their severity.

Phase 3 - Treatment Recommendation:

The Treatment Recommender Agent generates personalized interventions based on Prakriti and Vikriti analysis. It retrieves therapeutic protocols, dietary guidelines, herbal formulations, and lifestyle practices specific to the identified imbalances. Recommendations are tailored to the user's constitution, ensuring compatibility and effectiveness.

Phase 4 - Synthesis and Presentation:

The Orchestrator synthesizes outputs from all three specialist agents into a coherent consultation report. It resolves any conflicts between recommendations, adds explanatory context, includes citations to source texts, and formats the response for user presentation. The final output provides holistic guidance addressing constitution, current state, and treatment path.

Figure 2: Data Flow Sequence Diagram I was working on the diagram presently check them I will redraw them currently these are the reference.

Data flow image

The sequence diagram traces a complete user interaction from query submission through multi-phase agent collaboration to final response delivery. Color-coded rectangles highlight distinct operational phases, while swim lanes show concurrent activities across system components. The diagram emphasizes the grounding mechanism: every agent decision point involves RAG retrieval, ensuring recommendations trace back to authenticated classical sources.

3.3 Key Architectural Decisions

Several design choices distinguish AyurMind from conventional AI health systems:

- **Multi-Agent vs Single-Agent:** Decomposition into specialized agents mirrors the Ayurvedic diagnostic process where constitution, imbalance, and treatment are evaluated through distinct analytical lenses. This specialization improves retrieval relevance and reasoning quality compared to monolithic single-agent approaches.
- **RAG vs Fine-Tuning:** While fine-tuning could adapt the base LLM to Ayurvedic domain, RAG offers critical advantages: no hallucinations (all outputs grounded in retrieved text), easy knowledge updates (add new texts without retraining), and explicit source attribution (users can verify recommendations). RAG addresses the core problem of AI hallucination in medical contexts.
- **Local Deployment:** Using Ollama for local LLM inference eliminates API costs, ensures patient privacy (no data leaves user's machine), enables offline operation, and makes the system accessible in resource-constrained settings. This aligns with the goal of democratizing Ayurvedic knowledge.

- **Layered Architecture:** Clear separation between UI, orchestration, agents, retrieval, and knowledge layers enables modular development, independent testing of components, and future extensibility (e.g., adding new agents for pulse diagnosis or replacing the LLM backend without affecting other layers).

4. Technical Implementation

5.1 Technology Stack (Open Source)

Component	Technology
LLM	Llama-3-8B / Mistral-7B (via Ollama - local deployment)
Agent Framework	LangGraph / CrewAI (multi-agent orchestration)
RAG Framework	LangChain (document processing, retrieval pipeline)
Vector Database	ChromaDB (semantic search over Ayurvedic texts)
Embeddings	sentence-transformers (multilingual models for English/Hindi)
User Interface	Gradio (interactive web interface with chat)
Knowledge Base	Digitized Ayurvedic texts (Charaka Samhita, Sushruta Samhita, Ashtanga Hridaya - public domain translations)

5.2 RAG Pipeline Design

Document Processing

1. **Text Extraction:** Convert PDF/HTML versions of Ayurvedic texts into structured plain text
2. **Semantic Chunking:** Split texts into logical sections (slokas, chapters, treatment protocols) maintaining context boundaries
3. **Metadata Tagging:** Annotate chunks with source text, chapter, topic (Prakriti/Vikriti/Treatment), and Sanskrit terms
4. **Vector Embedding:** Generate semantic embeddings using multilingual sentence transformers, store in ChromaDB

Retrieval Strategy

Each agent performs specialized retrieval:

- **Prakriti Agent** → Retrieves constitution definitions, characteristic traits
- **Dosha Agent** → Retrieves disease causation, symptom correlations
- **Treatment Agent** → Retrieves therapeutic protocols, dietary guidelines, herbal formulations

Top-K retrieval (K=5) with relevance scoring ensures agents access most pertinent classical knowledge for each query.

5. Research Methodology

5.1 Dataset Preparation

1. **Source Texts:** Charaka Samhita (focus on Vimana Sthana, Chikitsa Sthana), Sushruta Samhita, Ashtanga Hridaya (English translations available in public domain)
2. **Test Cases:** Develop 30 clinical scenarios covering various Prakriti types and common imbalances (e.g., Vata aggravation causing anxiety, Pitta imbalance causing acidity)
3. **Ground Truth:** Consult 3-5 qualified Ayurvedic practitioners (BAMS degree holders with 5+ years experience) to provide expert diagnoses and treatment plans for test cases

5.2 Experimental Design

We conduct comparative evaluation across three conditions:

System	Architecture	Knowledge Source	Expected Issue
Baseline: Vanilla LLM	Single Llama-3	Pre-training only	Hallucinations
Condition 2: Single-Agent RAG	Single Llama-3	RAG over texts	Generic reasoning
AyurMind: Multi-Agent RAG	4 specialized agents	Targeted RAG per agent	Specialized + holistic

5.3 Evaluation Metrics

1. **Prakriti Classification Accuracy:** Percentage agreement with expert-assigned constitutional types
2. **Treatment Relevance Score:** Expert rating (1-5 scale) on appropriateness of dietary, lifestyle, and herbal recommendations
3. **Hallucination Rate:** Percentage of recommendations not grounded in retrieved Ayurvedic texts
4. **Response Completeness:** Coverage of diagnosis, reasoning, and treatment (binary checklist)
5. **Bilingual Performance:** Accuracy comparison for English vs Hindi queries

6. Expected Results

Metric	Vanilla LLM	Single-Agent RAG	AyurMind
Prakriti Accuracy	45%	68%	**78%**

Treatment Relevance	52%	71%	**81%**
Hallucination Rate	38%	8%	**<3%**
Expert Agreement	41%	64%	**73%**

7. Novel Contributions

1. **First Multi-Agent Architecture for Ayurveda:** Pioneering specialized agent decomposition for traditional medical diagnosis
2. **Open Ayurvedic Knowledge Base:** Digitized, structured, and semantically indexed classical texts available to research community
3. **Validation Framework:** Novel methodology for evaluating AI systems against traditional medical expertise
4. **IKS-AI Integration Blueprint:** Replicable approach for applying modern AI to preserve and democratize ancient wisdom systems
5. **Bilingual Capability:** Bridges language barriers between Sanskrit scholarship and contemporary Hindi/English speakers

8. Implementation Timeline

Here ignore the phases, I will work parallel on project and paper writing.

Phase	Tasks
Week 1-2	Data Collection & RAG Setup: Download Ayurvedic texts, process into chunks, create embeddings, build ChromaDB vector store, test basic retrieval
Week 3-4	Multi-Agent Development: Implement 4 agents using LangGraph/CrewAI, design prompts for specialized reasoning, build orchestrator coordination logic, create Gradio interface
Week 5	Expert Validation: Develop 30 test cases, consult Ayurvedic practitioners for ground truth, implement baseline systems (vanilla LLM, single-agent RAG)
Week 6-7	Experiments & Analysis: Run comparative evaluation across 3 conditions, compute metrics, perform ablation studies (effect of each agent, retrieval strategies), analyze failure cases
Week 8	Paper Writing: Draft manuscript, create visualizations (agent workflow diagram, result charts), prepare demo video, finalize submission

9. MBCC 2026 Conference Alignment

This research perfectly aligns with MBCC 2026's mission to integrate Indian Knowledge Systems with cutting-edge technology:

- **Primary Track - Ayurveda:** Core focus on digitizing and operationalizing Ayurvedic diagnostic principles
- **Secondary Track - NLP for Indian Languages:** Bilingual system supporting English and Hindi with Sanskrit terminology preservation
- **Tertiary Track - Cognitive Science and AI:** Novel multi-agent reasoning architecture for medical diagnosis
- **Holistic Integration:** Bridges empirical AI research with timeless IKS wisdom, fostering dialogue between disciplines

The interdisciplinary nature of this work—combining computer science, traditional medicine, and knowledge preservation—exemplifies MBCC's vision of redefining consciousness and cognitive science studies through IKS integration.

10. Impact and Future Work

10.1 Immediate Impact

- **Healthcare Access:** Democratizes Ayurvedic consultation for underserved populations
- **Knowledge Preservation:** Creates digital infrastructure preventing loss of traditional wisdom
- **Research Tool:** Enables systematic study of Ayurvedic principles through computational methods

10.2 Future Extensions

1. **Fine-Tuning Layer:** Add domain-specific fine-tuning on Ayurvedic clinical case studies for improved accuracy
2. **Additional Languages:** Expand to Tamil, Bengali, Marathi, Telugu for pan-India accessibility
3. **Integration with Modern Medicine:** Hybrid system combining Ayurvedic and allopathic recommendations with drug interaction checks
4. **Clinical Trials:** Partner with Ayurvedic hospitals to validate system recommendations through patient outcomes
5. **Generalization to Other IKS:** Apply framework to Yoga therapy, Unani medicine, Traditional Chinese Medicine

11. Conclusion

AyurMind represents a paradigm shift in how artificial intelligence can serve to preserve, democratize, and advance traditional knowledge systems. By combining the retrieval precision of RAG with the collaborative reasoning of multi-agent architectures, we address the critical dual challenge of hallucination prevention and specialized medical reasoning.

This work is not merely a technical contribution but a bridge between ancient wisdom and modern innovation—exactly the interdisciplinary vision that MBCC 2026 champions. The proposed system has immediate practical applications while opening new research directions in AI-augmented traditional medicine.

We are confident that AyurMind will generate significant interest at MBCC 2026, spark meaningful discussions between IKS scholars and AI researchers, and inspire future work at the intersection of computational intelligence and timeless human knowledge.

Appendix: Technical Details

A. Sample Agent Prompts

Prakriti Assessor Agent:

"You are an expert Ayurvedic practitioner specializing in Prakriti assessment. Based on the user's responses about physical characteristics (body frame, skin type, hair texture), mental tendencies (stress response, decision-making style), and behavioral patterns (appetite, sleep, energy levels), determine their constitutional type. Use the retrieved classical texts to support your assessment. Ask clarifying questions when needed. Provide assessment only after gathering sufficient information."

Dosha Imbalance Detector Agent:

"You are an Ayurvedic diagnostician identifying current health imbalances (Vikriti). Analyze the user's reported symptoms and correlate them with Dosha aggravation patterns from classical texts. Consider: digestive issues → Agni status, skin problems → Pitta/Kapha imbalance, anxiety/insomnia → Vata aggravation. Use retrieved text passages to justify your diagnosis. Indicate confidence level (definite/probable/possible)."

B. Code Repository Structure

Complete code will be released as open-source upon publication:

- data/ - Processed Ayurvedic texts and embeddings
- src/rag/ - RAG pipeline (chunking, embedding, retrieval)
- src/agents/ - Multi-agent implementation (4 specialist agents + orchestrator)
- src/evaluation/ - Evaluation scripts and test cases
- app.py - Gradio web interface
- requirements.txt - Python dependencies (all open-source)
- README.md - Setup instructions and usage guide