Multi-Agent FullStack RAG Pipeline

Intelligent Agent Communication & Real-time Evaluation System

LLM Assignment 3.1 & 3.2

Anjila Subedi
Roll No: 29
Large Language Models Course
Kathmandu University

June 28, 2025

Abstract

A comprehensive multi-agent RAG system featuring intelligent agent communication, real-time evaluation metrics, and advanced content processing capabilities. The system implements a message bus architecture with specialized agents for document processing, web scraping, and quality evaluation.

Assignment Context: This project demonstrates advanced implementation of Large Language Model concepts including multi-agent systems, retrieval-augmented generation (RAG), real-time evaluation metrics, and intelligent agent communication patterns as part of LLM Assignment 3.1 and 3.2.

1 Key Features

- **Multi-Agent Communication**: Message bus architecture with shared memory
- **Real-time Evaluation**: 4-metric assessment system with detailed feedback
- **Advanced Content Processing**: PDF and web content extraction
- **Performance Monitoring**: Agent activity tracking and health monitoring
- **Modern UI**: Next.js frontend with evaluation interface

2 System Workflow

2.1 Step 1: Content Upload & Processing

Process Flow

```
User Upload → Document Agent → Text Chunking → Vector Storage →
Ready for Queries
```

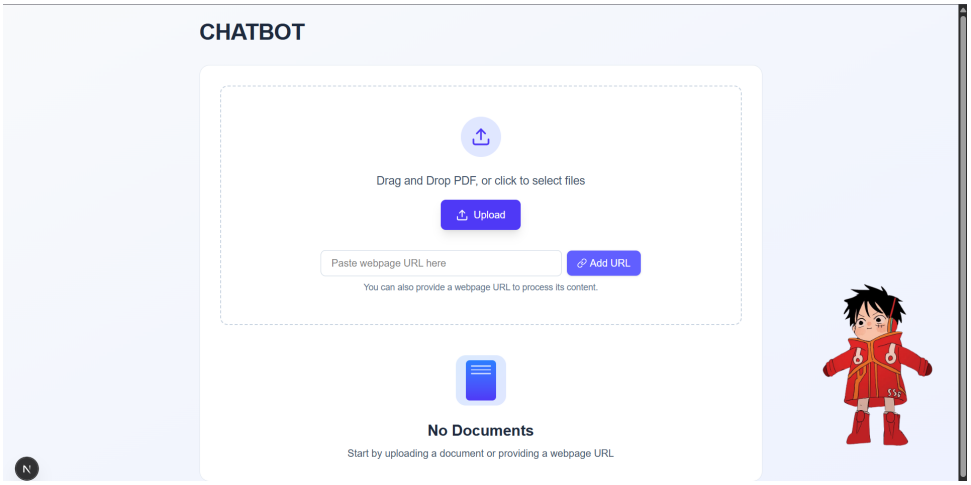

Figure 1: Content Upload Interface

2.2 Step 2: Interactive Chat Interface

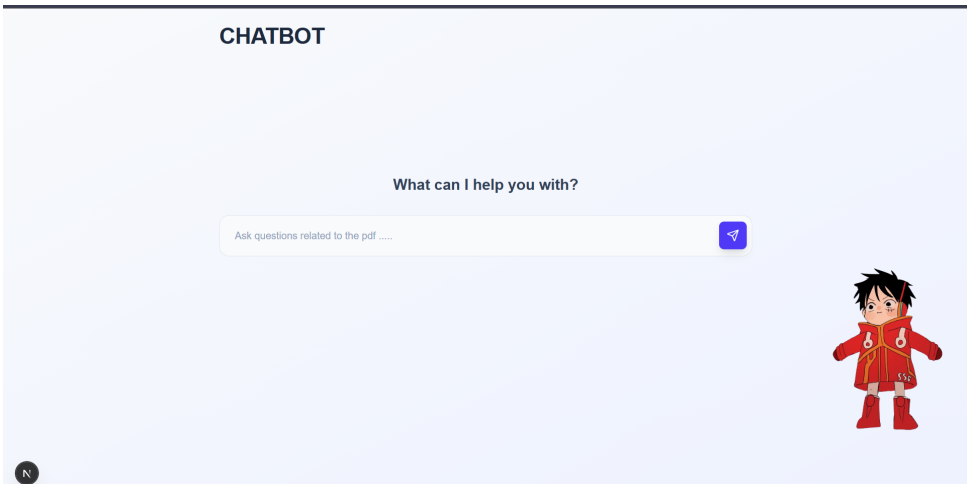

Figure 2: Interactive Chat Interface

Chat Features:

- Real-time messaging with markdown support
- Evaluation toggle for quality assessment
- Ground truth input for correctness testing

2.3 Step 3: Intelligent Query Processing

2.4 Step 4: Real-time Evaluation System

Evaluation Metrics:

- Correctness: Answer accuracy against ground truth
- Relevance: Question-answer alignment (1-5 scale)

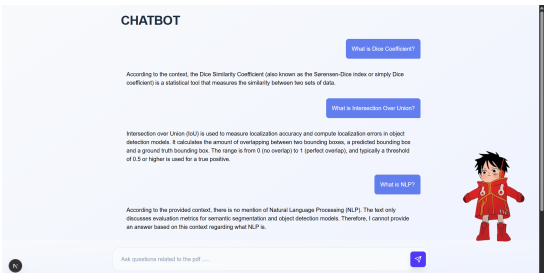

Figure 3: PDF Processing Example

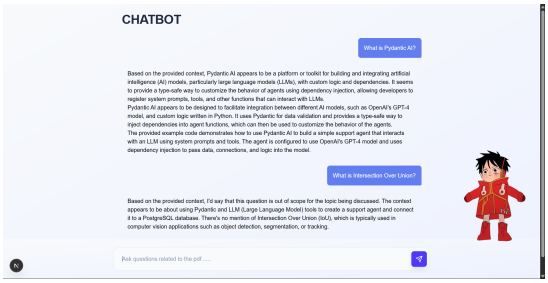

Figure 4: Web Content Processing

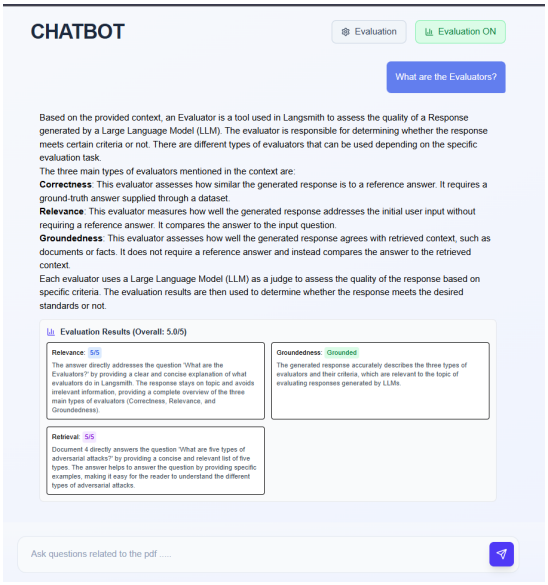

Figure 5: Evaluation Interface

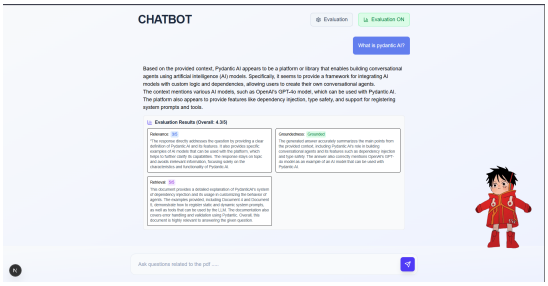

Figure 6: Detailed Evaluation Results

- **Groundedness**: Context-based answer validation
- **Retrieval Quality**: Document relevance assessment

3 Multi-Agent Architecture

3.1 System Architecture Overview

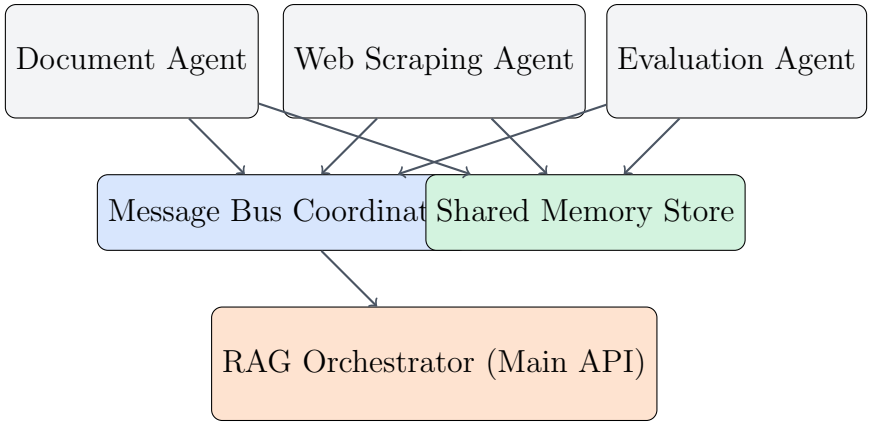

Figure 7: Multi-Agent System Architecture

3.2 Agent Communication Flow

Message Bus System

- **Centralized Hub**: All agents communicate through a single message bus
- **Async Messaging**: Non-blocking message passing between agents
- **Event Broadcasting**: Status updates propagated to all interested agents

Listing 1: Shared Memory Architecture

```
# Agent A stores data
shared_memory.set("pdf_chunks", processed_data)

# Agent B retrieves data
data = shared_memory.get("pdf_chunks")
```

Agent Coordination Example:

1. Document Agent: "PDF processing started"
2. Message Bus: Broadcast to System Coordinator
3. Shared Memory: Store processing results
4. Evaluation Agent: "Ready for evaluation"
5. System Coordinator: "Pipeline ready"

3.3 Specialized Agent Roles

Document Processing Agent

- PDF text extraction and chunking
- Metadata preservation
- Status reporting to message bus

Web Scraping Agent

- Async HTTP requests
- HTML parsing and cleaning
- Content validation and storage

Evaluation Agent

- Multi-metric assessment
- Real-time quality scoring
- Detailed feedback generation

System Coordinator

- Agent health monitoring
- Activity logging
- Error handling and recovery

4 Evaluation Process Deep Dive

4.1 4-Metric Assessment System

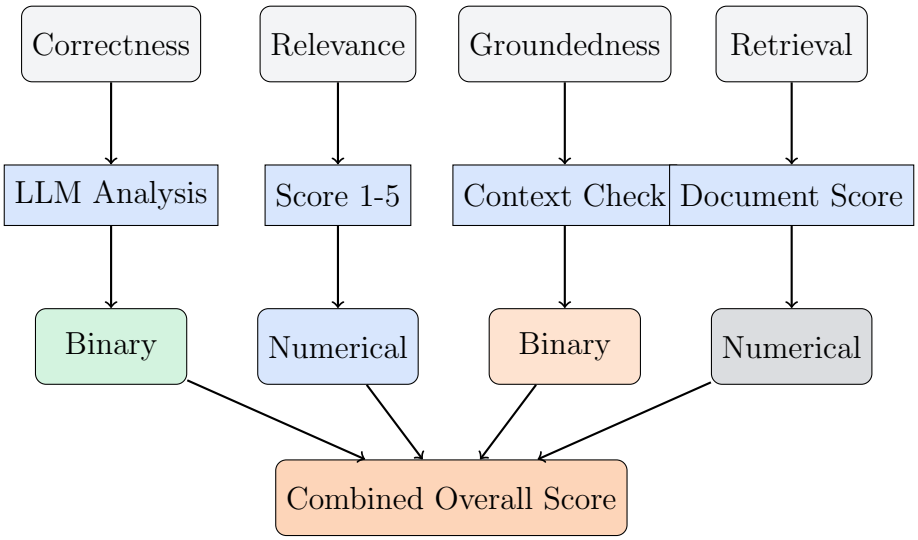

Figure 8: Evaluation Workflow Architecture

4.2 Evaluation Metrics Details

Correctness Evaluation

`Ground Truth Input → LLM Comparison → Binary Score (/)`

- Semantic similarity analysis
- Factual accuracy check
- Key information coverage

Relevance Scoring (1-5 Scale)

`Question Analysis → Answer Relevance → Numerical Score`

- Topic alignment verification
- Context appropriateness
- Response completeness

Groundedness Assessment

`Retrieved Context → Answer Validation → Support Score`

- Citation accuracy check
- Hallucination detection
- Source material alignment

Retrieval Quality (1-5 Scale)

`Query Vector → Document Matching → Relevance Score`

- Semantic similarity measurement
- Content coverage analysis
- Information completeness

5 API Endpoints

5.1 Core Operations

Endpoint
POST /uploadPDF processing via Document Agent POST /urlWeb content via Scraping Agent POS

5.2 Evaluation Endpoints

Endpoint
POST /evaluate/correctnessBinary accuracy assessment POST /evaluate/relevance1-5 scale relevanc

5.3 Agent Monitoring

Endpoint
GET /agents/statusReal-time agent health GET /agents/activitiesRecent agent activity logs GET /a

6 Quick Start

6.1 Backend Setup

```
# Install dependencies
pip install -r backend/requirements.txt

# Start FastAPI server
cd backend && python rag.py

# Verify Ollama models
ollama list | grep -E "(mxbai-embed-large|llama3)"
```

6.2 Frontend Setup

```
# Install and start Next.js
cd chatbot
npm install && npm run dev
```

6.3 Access Points

- **Frontend**: `http://localhost:3000`
- **API Docs**: `http://localhost:8000/docs`
- **Agent Status**: `http://localhost:8000/agents/status`

7 System Benefits

- ✓ **Modular Architecture**: Independent, specialized agents
- ✓ **Real-time Evaluation**: Instant quality feedback
- ✓ **Transparent Operations**: Complete activity visibility
- ✓ **Scalable Design**: Easy agent addition and modification
- ✓ **Robust Error Handling**: Isolated failure management
- ✓ **Modern UI/UX**: Responsive design with rich interactions

8 Assignment Reflection & Learning Outcomes

8.1 Technical Achievements

Assignment 3.1 Accomplishments

Multi-Agent System Implementation:

- Successfully designed and implemented 4 specialized agents
- Created robust message bus communication system
- Implemented shared memory architecture for data persistence
- Achieved asynchronous processing with proper error handling

Assignment 3.2 Accomplishments

RAG Evaluation Framework:

- Developed comprehensive 4-metric evaluation system
- Implemented real-time quality assessment capabilities
- Created detailed feedback mechanisms with explanations
- Built performance tracking and analytics dashboard

8.2 Key Learning Insights

1. Multi-Agent Coordination Complexity

- Understanding the challenges of distributed system design
- Learning effective inter-agent communication patterns
- Managing state consistency across multiple agents

2. RAG System Evaluation Challenges

- Implementing objective quality metrics for subjective responses
- Balancing automated evaluation with human judgment
- Creating meaningful performance benchmarks

3. System Integration & Scalability

- Building modular systems that can grow and adapt
- Implementing proper error isolation and recovery
- Creating maintainable and extensible architectures

8.3 Future Enhancements

Potential Improvements for Advanced LLM Applications:

1. **Advanced Agent Reasoning**: Implement chain-of-thought processing
2. **Dynamic Agent Spawning**: Create agents on-demand based on workload
3. **Multi-Modal Processing**: Extend to handle images, audio, and video
4. **Federated Learning**: Implement distributed model training capabilities
5. **Advanced Evaluation Metrics**: Include bias detection and fairness assessment

Technology Stack

FastAPI • LangChain • ChromaDB • Ollama • Next.js • TypeScript

Submitted by: **Anjila Subedi** — Roll No: 29 — LLM Assignment 3.1 & 3.2