Agentic AI: Sample Document for RAG Testing

1. Definition of Agentic AI

Agentic AI refers to artificial intelligence systems that can independently plan, reason, and execute tasks toward a goal. Unlike traditional AI systems that respond passively to user inputs, agentic systems actively decide what actions to take next. These systems use reasoning loops, tools, and memory to accomplish complex objectives. Agentic AI is often implemented using frameworks like LangChain and LangGraph.

2. Key Components of Agentic Systems

Agentic AI systems typically include several core components. First, the LLM (Large Language Model) acts as the brain. Second, tools allow the agent to interact with external systems such as web search, databases, or APIs. Third, memory enables the agent to retain context across steps. Fourth, a planner decides the sequence of actions required to complete a task. Finally, an executor performs those actions.

3. Agent Loop

The agent loop is the core mechanism behind agentic behavior. The loop consists of perception, reasoning, action, and observation. The agent observes the current state, reasons about what to do next, performs an action, and then observes the result. This cycle continues until the goal is achieved. LangGraph is commonly used to implement this loop in production systems.

4. Tools in Agentic AI

Tools extend the capabilities of agents. Examples include Wikipedia search tools, Arxiv research tools, database query tools, and code execution tools. By using tools, agents can retrieve external information instead of relying only on their training data. This makes the system more accurate and up-to-date.

5. Retrieval Augmented Generation (RAG)

RAG combines vector search with language models. Documents are converted into embeddings and stored in a vector database. When a question is asked, relevant documents are retrieved and provided to the model. This improves factual accuracy and allows the system to answer questions about private data. This document can be used as a source in RAG testing.

6. LangGraph Architecture

LangGraph allows developers to define agent workflows as graphs. Each node represents a step such as planning, tool execution, or response generation. Edges define transitions between steps. This structure provides better control, debugging, and scalability compared to simple chains.

7. Real-world Applications

Agentic AI is used in research assistants, autonomous coding agents, customer support systems, and workflow automation tools. Companies use agentic systems to automate complex business processes and decision-making.

8. Conclusion

Agentic AI represents the next evolution of AI systems. Instead of simple question answering, agents can plan, act, and solve multi-step problems. Combined with RAG, agentic AI can provide accurate, reliable, and context-aware responses.

Additional Content for Chunking Tests

This page exists to increase document length for better chunking during RAG testing. Chunking is the process of splitting documents into smaller pieces before embedding them. Good chunking improves retrieval accuracy. Testing with multiple sections ensures your RAG pipeline works correctly.

This page exists to increase document length for better chunking during RAG testing. Chunking is the process of splitting documents into smaller pieces before embedding them. Good chunking improves retrieval accuracy. Testing with multiple sections ensures your RAG pipeline works correctly.

This page exists to increase document length for better chunking during RAG testing. Chunking is the process of splitting documents into smaller pieces before embedding them. Good chunking improves retrieval accuracy. Testing with multiple sections ensures your RAG pipeline works correctly.

This page exists to increase document length for better chunking during RAG testing. Chunking is the process of splitting documents into smaller pieces before embedding them. Good chunking improves retrieval accuracy. Testing with multiple sections ensures your RAG pipeline works correctly.

This page exists to increase document length for better chunking during RAG testing. Chunking is the process of splitting documents into smaller pieces before embedding them. Good chunking improves retrieval accuracy. Testing with multiple sections ensures your RAG pipeline works correctly.