Progress Agentic RAG As a Service

Author: Jitesh Kamra

Affiliation: Five Frogs Technologies Pvt Ltd.

Date: January 19, 2025

Executive Summary

In the rapidly evolving landscape of artificial intelligence, Retrieval-Augmented Generation (RAG) has emerged as a powerful technique for enhancing the capabilities of Large Language Models (LLMs) by integrating external data sources. Traditional RAG focuses on retrieving relevant information and generating responses, but it often falls short in handling complex, multi-step queries or dynamic environments.

Agentic RAG represents an advanced evolution, incorporating an autonomous agent layer that enables planning, decision-making, iterative reasoning, and even real-world actions. This white paper explores the fundamentals of RAG, introduces Agentic RAG, and highlights the Progress® Agentic RAG platform—a high-value SaaS solution designed to index diverse data sources, enrich knowledge with LLMs, and ensure high-quality outputs through robust RAG metrics.

Key benefits include autonomous reasoning, dynamic retrieval, improved accuracy, and seamless integration with databases like Progress OpenEdge. We discuss use cases across domains such as HR, healthcare, finance, and customer support, and compare it with other platforms. By adopting Agentic RAG, organizations can automate tasks, enhance user experiences, and stay competitive in an AI-driven world.

Introduction to Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) is a hybrid approach that combines information retrieval with generative AI to produce more accurate and contextually relevant responses. At its core, RAG addresses the limitations of standalone LLMs, which may hallucinate or lack up-to-date knowledge, by fetching pertinent data from external sources before generation.

Traditional RAG operates in two primary phases:

- **Retrieval**: The system queries external sources—such as documents, databases, or APIs—to gather relevant information. Common retrievers include vector databases like FAISS or search engines like Elasticsearch.
- **Generation**: An LLM, such as GPT-4 or Flan-T5, synthesizes the retrieved data into coherent outputs, such as answers, summaries, or recommendations.

This methodology ensures that responses are grounded in real data, reducing errors and improving reliability. RAG is particularly valuable in enterprise settings where data privacy, accuracy, and domain-specific knowledge are paramount.

Classic RAG Examples

Traditional RAG has been successfully applied across various domains, demonstrating its versatility in handling structured and unstructured data. Below are illustrative examples:

Example	Domain	Retriever	Generator	Outcome
Document Q&A	Enterprise Docs	FAISS	GPT-4	Policy answers
Support Chatbot	Retail	Elasticsearch	Flan-T5	Product help
Legal Assistant	Legal	Weaviate	GPT-3.5	Clause summaries
Medical Assistant	Healthcare	FAISS	BioBERT	Research summaries
HR Assistant	Enterprise	Cognitive Search	GPT-4	HR queries

These examples showcase how RAG can be tailored to specific industries, leveraging domain-optimized retrievers and generators to deliver precise, actionable insights.

Introducing Agentic RAG

Agentic RAG builds upon traditional RAG by introducing an "agent" layer that imbues the system with autonomy and intelligence. Unlike static RAG, which performs a one-time retrieval and generation, Agentic RAG enables the AI to act proactively to achieve user goals.

Key capabilities of the agent layer include:

- **Planning Actions**: Deciding the sequence of steps needed for complex queries.
- **Tool Selection**: Choosing appropriate sources, tools, or APIs dynamically.
- **Iterative Reasoning**: Refining responses through multiple cycles of retrieval and evaluation.
- **Execution**: Performing actions beyond generation, such as invoking workflows or updating databases.

This evolution transforms RAG from a passive query-answering tool into an active problem-solving system, ideal for scenarios requiring multi-step logic or real-time adaptations.

Components of Agentic RAG

Agentic RAG comprises several interconnected components that work in harmony to deliver intelligent outcomes:

Component	Role	Example
Planner / Agent	Determines next actions (retrieve, reason, or execute).	Logic engine deciding query strategy
Retriever	Fetches data from structured/unstructured sources.	Vector search or database query
Generator (LLM)	Synthesizes outputs or intermediate steps.	LLM like GPT for response creation
Tool / API Layer	Enables calls to external systems, databases, or workflows.	API integrations for actions
Memory	Maintains context across sessions for improved reasoning.	Persistent storage of conversation history

These components ensure the system is flexible, scalable, and capable of handling diverse tasks.

Sample Workflow in Agentic RAG

A typical Agentic RAG workflow begins with a user query, which the planner decomposes into actionable steps. For instance:

1. The agent assesses the query and retrieves initial data from a vector store.
2. If more context is needed, it invokes tools (e.g., API calls) for additional information.
3. The LLM generates a draft response, which the agent evaluates and refines iteratively.
4. Finally, the system outputs a polished answer or triggers an action.

This sample process highlights the dynamic nature of Agentic RAG, allowing for adaptive problem-solving.

Benefits of Agentic RAG

Agentic RAG offers significant advantages over traditional methods:

- **Autonomous Reasoning**: Handles complex, multi-step queries without human intervention.
- **Dynamic Retrieval**: Adapts strategies based on query complexity or data availability.
- **Real-World Action**: Can initiate workflows, such as updating records or sending notifications.
- **Improved Accuracy**: Verifies and corrects outputs through self-evaluation.
- **Context Awareness**: Retains memory for personalized, ongoing interactions.

These benefits make Agentic RAG a transformative tool for enterprises seeking efficiency and innovation.

Comparison with Traditional RAG

The following table contrasts Agentic RAG with classic RAG:

Feature	RAG	Agentic RAG
Retrieval	Static / one-time	Dynamic and iterative
Generation	Single-step answer	Multi-step reasoning
Tool Usage	Limited	Multiple tools / APIs
Autonomy	None	High
Memory	Stateless	Contextual and persistent

Agentic RAG's enhancements enable it to tackle more sophisticated challenges.

Progress® Agentic RAG Platform

The Progress® Agentic RAG platform is a comprehensive SaaS solution that automates the indexing of files and documents to support diverse LLM and AI agent use cases. It ensures high-quality outputs via built-in RAG quality metrics, making it an ideal choice for enterprises.

Key features include automatic data indexing, LLM integration for knowledge enrichment, customizable retrieval strategies, and evaluation of embedding models.

Why Choose Progress Agentic RAG?

- **Versatile Indexing**: Handle any data type in any language.
- **LLM Enrichment**: Leverage any LLM to derive insights from your data.
- **Custom Retrieval**: Define strategies tailored to your needs.
- **Embedding Flexibility**: Evaluate and select optimal models.
- **Competitive Edge**: Integrate AI to enhance product value.
- **Time Savings**: Automate tasks and save hundreds of hours.
- **Advanced Capabilities**: Unlock AI on unstructured data.
- **Enhanced Search**: Natural language queries for accurate, relevant responses.

These advantages position Progress Agentic RAG as a strategic investment for AI-driven innovation.

Data Sources

Agentic RAG supports a wide array of data sources, including documents, databases, and APIs, ensuring comprehensive coverage for retrieval tasks.

Integration with Progress OpenEdge

While Progress does not yet offer a native Agentic RAG product, it can be achieved by integrating with relational databases like Progress OpenEdge, SQL Server, PostgreSQL, Oracle, and MySQL. This unification is facilitated through:

- **Progress APIs / JDBC Connectors**: For data access.
- **External AI Orchestration**: Tools like LangChain for agent logic.
- **Common Schema or Vector Store**: To normalize and store data.

A sample integration architecture includes:

Component	Role	Example
Retriever Layer	Pulls data from databases.	JDBC/ODBC drivers
Schema Normalization Layer	Unifies schemas into a vector store.	ETL or AI pipelines
RAG/Agent Layer	Manages retrieval and reasoning.	LangChain or custom APIs
LLM Layer	Generates responses.	GPT or similar models

For more details, refer to: https://docs.rag.progress.cloud/docs/.

Use Cases for Agentic RAG

Agentic RAG excels in practical applications:

Domain	Example
Enterprise HR (Hartlink)	Retrieves policies, checks eligibility, automates updates.
Healthcare	Finds clinical data, interprets results, drafts reports.
Finance (Hartlink)	Analyzes regulatory data, generates compliance summaries.
Customer Support (Hartlink)	Fetches FAQs, triggers requests, logs issues autonomously.

These use cases demonstrate Agentic RAG's potential to streamline operations.

Competing Platforms

Several platforms offer similar capabilities:

Platform	Features
OpenAI's RAG-as-a-Service (Azure)	Fully managed enterprise RAG.
Databricks Mosaic AI	Retrieval + LLM pipeline with data connectors.
Cohere Command RAG	API-driven retrieval and generation.
Google Vertex AI Search + Gemini	Enterprise search combined with LLMs.

Progress Agentic RAG stands out for its focus on OpenEdge integration and customizable metrics.

Conclusion

Agentic RAG represents a paradigm shift in AI, empowering systems with agency to handle complex tasks autonomously. The Progress® platform provides a robust, SaaS-based solution for enterprises to harness this technology, driving efficiency, accuracy, and innovation.

By adopting Agentic RAG, organizations like Five Frogs Technologies Pvt Ltd. can transform data into actionable intelligence, ensuring a competitive advantage in the AI era.

References

- Progress RAG Documentation: https://docs.rag.progress.cloud/docs/
- LangChain and related AI orchestration tools.
- Various LLM providers (e.g., OpenAI, Anthropic).