Notes

AWS Project Documentation

Set Up a RAG Chatbot in Bedrock

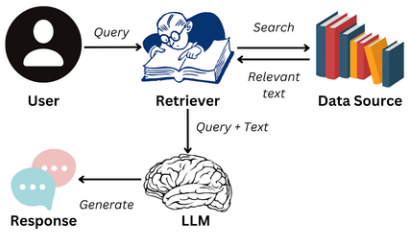
@sAnandha

Notes

Building a RAG Chatbot in Amazon Bedrock

Introduction :

Retrieval-Augmented Generation (RAG) is a powerful approach that enhances AI chatbots by integrating an external knowledge base. Instead of relying solely on predefined responses, RAG enables chatbots to retrieve relevant information from stored documents, ensuring accurate and dynamic responses.



Why Use the RAG Model?

- Enhanced Accuracy: Provides fact-based responses by retrieving relevant context.
- Scalability: Supports large datasets without increasing computational load.
- Flexibility: Adapts to new information without retraining the AI model.

What is Amazon Bedrock?

Amazon Bedrock is a fully managed AI service that allows developers to build and scale generative AI applications using foundation models (FMs) from various AI providers. It simplifies integrating AI capabilities into applications without requiring infrastructure management.

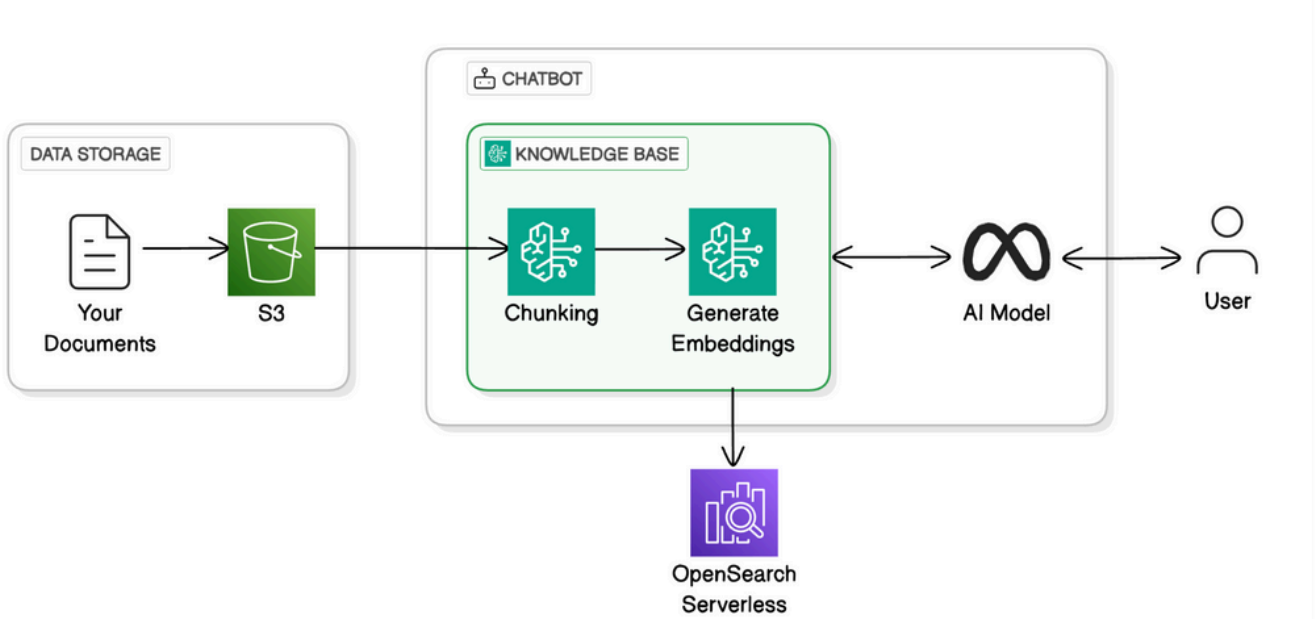
Why Use Amazon Bedrock?

- Access to Multiple AI Models: Supports models from Anthropic, AI21 Labs, Stability AI, and Amazon Titan.
- Customization: Fine-tune models with your own data for domain-specific AI.
- Seamless Integration: Works with AWS services like S3, OpenSearch, and Lambda.
- Scalability: Handles high-volume requests without manual scaling.
- Security & Compliance: Built with enterprise-grade security and access controls.

Use Cases of Amazon Bedrock

- Chatbots and Virtual Assistants: Power AI-driven customer support bots.
- Knowledge Management: Enhance enterprise search with RAG-based models.
- Content Generation: Automate blog writing, reports, and marketing content.
- Code Assistance: Improve developer productivity with AI-driven code suggestions.
- Medical and Legal Research: Process large text datasets for insights.

ARCHITECTURE DIAGRAM :



Architecture Overview :

1. Data Storage (Amazon S3): Stores raw documents uploaded by users.
2. Chunking: Splits large documents into smaller sections for efficient processing.
3. Embedding Generation: Converts chunks into vector representations.
4. OpenSearch Serverless: Indexes and retrieves embeddings for quick lookup.
5. AI Model (Amazon Bedrock): Processes queries and retrieves relevant data.
6. User Interaction: The chatbot interacts with users based on retrieved knowledge

Steps to Build a RAG Chatbot:

1. **Store Documents in Amazon S3**
   - Upload raw documents (PDFs, text files, etc.) to an S3 bucket.
   - Configure permissions to allow secure access.
2. **Chunking the Documents**
   - Break large documents into smaller, manageable text chunks.
   - Use natural language processing (NLP) to ensure meaningful segmentation.
3. **Generate Embeddings**
   - Convert each text chunk into a vector representation using an embedding model.
   - Store these embeddings for efficient similarity search.
4. **Index and Search in OpenSearch Serverless**
   - Store embeddings in Amazon OpenSearch Serverless.
   - Use vector search to retrieve the most relevant document chunks for a query.
5. **AI Model Processing in Amazon Bedrock**
   - Query the AI model with user input and retrieved document context.
     - Generate an accurate response based on the retrieved knowledge.

- Compare different models for response quality:
  - Titan Text Embeddings V2: Converts text into numerical embeddings.
  - Llama 3.1 8B Instruct: Optimized for conversational AI.
  - Llama 3.3 70B Instruct: Advanced reasoning and response generation.

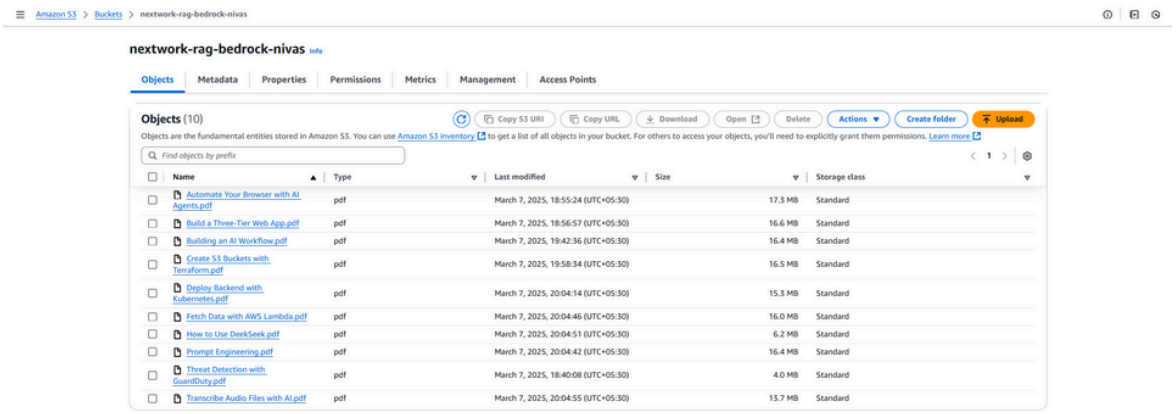
6. **User Interaction and Refinement**
   - Deploy the chatbot for real-time user queries.
   - Continuously improve response accuracy by refining the knowledge base.
   - Experiment with additional models like Claude, Mistral, or Command R+ for better performance.

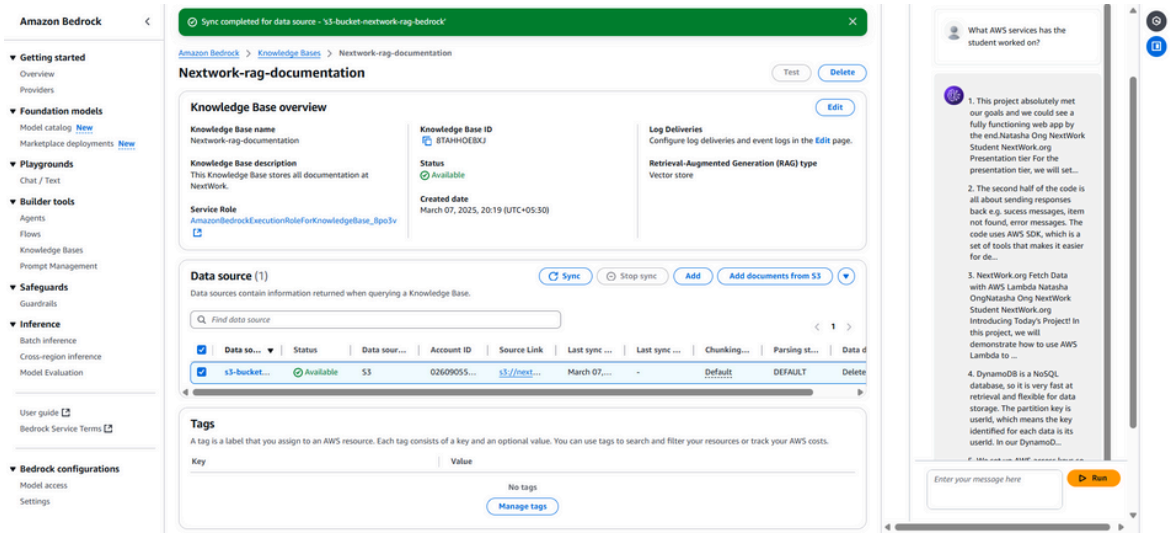
Services Used

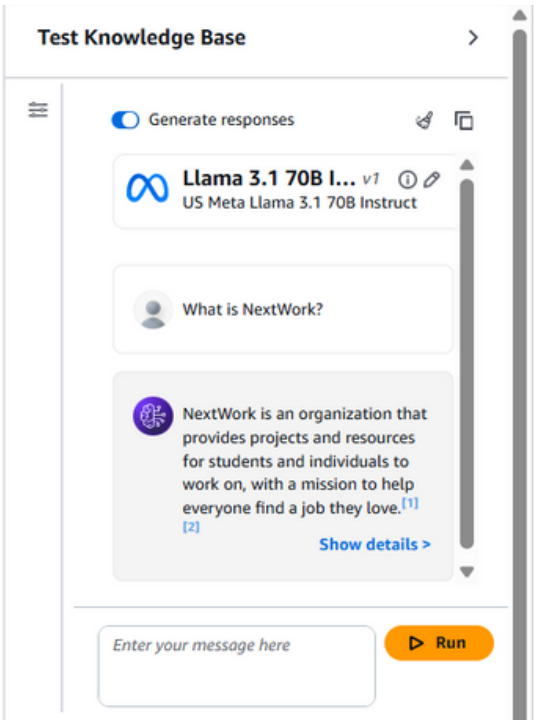
- Amazon S3: Stores documents securely.
- Amazon OpenSearch Serverless: Enables fast retrieval of relevant information.
- Amazon Bedrock: Hosts AI models for intelligent responses.
- Embedding Model: Converts text into vector representations.

Notes

SAMPLE OUTPUT :







For More References :
https://learn.nextwork.org/projects/ai-rag-bedrock?_gl=1*hmudkx*_ga*MjAzODY1Mzc3My4xNzI5NTMwMDAy*_ga_P3ZJGC0XCG*MTc0MTM2NDE0Mi4xNDcuMS4xNzQxMzY1OTQyLjAuMC4w&track=high