📄Mini RAG App

This project is a **Retrieval-Augmented Generation (RAG) application** built using **Streamlit**, **LangChain**, **NVIDIA AI Endpoints**, and **Pinecone**. It allows users to upload PDF documents, embed them into a vector database, and query them interactively with an LLM.

⚙️Chunking Parameters

- **Chunk size:** `1000`
- **Chunk overlap:** `120`

👉This ensures that chunks are large enough to preserve context while maintaining overlap to avoid cutting off important information.

🔍Retriever & Reranker Settings

- **Base Retriever:**
- Source: **Pinecone VectorStore**
- Embedding: **NVIDIAEmbeddings**

- Top-k: **5** similar chunks retrieved per query

- **Reranker / Compressor:**

- Component: **LLMChainExtractor** (powered by ChatNVIDIA)

- Function: Filters and compresses retrieved chunks to keep only the most relevant context.

- **Final Retriever:**

- Wrapped as **ContextualCompressionRetriever** (retriever + reranker in one).

🪥LLM & Embeddings Provider

- **Provider:** NVIDIA AI Endpoints
- **LLM Model:** `meta/llama-3.1-70b-instruct`
- **Embeddings Model:** `NVIDIAEmbeddings`

🧮Quick Start

1. Clone the Repository

```
git clone https://github.com/your-repo/mini-rag-app.git
cd mini-rag-app
```

2. Install Dependencies

```
pip install -r requirements.txt
```

3. Set Environment Variables

Create a `.env` file in the project root:

```
NVIDIA_API_KEY=your_nvidia_api_key
PINECONE_API_KEY=your_pinecone_api_key
```

4. Run the App

```
streamlit run app.py
```

5. Usage

- Upload one or more PDF documents.
- Click **Documents Embedding** to process and store them in Pinecone.
- Ask questions in the input box.
- View generated answers and inspect retrieved document chunks.

🔗Features

- Multi-PDF ingestion and embedding
- Pinecone namespace isolation per user session
- Retriever + reranker pipeline for high-quality context
- NVIDIA-powered embeddings and LLM
- Transparent similarity search inspection

📂Tech Stack

- **Frontend:** Streamlit
- **Vector DB:** Pinecone
- **Embeddings & LLM:** NVIDIA AI Endpoints
- **Framework:** LangChain

🛠️Pinecone Index Configuration

Suggested Configuration

- **Dimension:** `1024` (NVIDIA embeddings dimension)
- **Metric:** `cosine`
- **Pod Type:** `p1.x1`
- **Replicas:** `1`

CLI Command Example

```
pinecone index create database
  --dimension 1024
  --metric cosine
  --pods 1
  --replicas 1
  --pod-type p1.x1
```

👉Each session uses a **unique namespace** (`session-xxxxxxx`) ensuring data isolation between users.