**Project Report: Document Analysis and Retrieval Chatbot**

**1. Introduction**

This project involves analyzing large documents more than 256 pages, generating vector embeddings, storing them in a vector database, and enabling user queries via a chatbot. The system supports:

* **Retrieval-Augmented Generation (RAG)**: Stores document embeddings and retrieves relevant sections based on user queries.
* **AI Insights**: Extracts meaningful insights from documents.
* **Summarization**: Summarizes retrieved document sections.
* **Chatbot**: Allows users to ask questions and receive contextual responses.

**2. Technology Stack**

The project uses the following technologies:

* **Streamlit**: Web interface for interaction.
* **LangChain**: Document processing and retrieval framework.
* **Ollama**: Embedding model for text vectorization.
* **Pinecone**: Vector database for efficient retrieval.
* **Groq API (Llama3-8b-8192)**: LLM for response generation.
* **PyPDFLoader**: Extracts text from PDFs.

**3. Implementation Details**

**3.1. PDF Processing and Vector Embedding**

1. Users upload a PDF document via Streamlit.
2. The document is split into chunks (2,000 characters with 1,000 overlaps) using **RecursiveCharacterTextSplitter**.
3. Embeddings are generated using **Ollama (Llama3.2:1b)**.
4. Embeddings and metadata (including chunk content) are stored in **Pinecone**.

**3.2. Query Processing**

1. The user inputs a query.
2. The system retrieves the most relevant chunks from Pinecone using cosine similarity.
3. Retrieved document content is passed to **Groq API (Llama3-8b-8192)** for summarization and response generation.
4. The chatbot displays the response along with relevant document context.

**3.3. Error Handling and Validations**

* Ensures that uploaded documents are successfully loaded.
* Checks for missing embeddings and incorrect chunk splits.
* Validates the Pinecone index before inserting or querying.

**4. Results**

* Successfully extracted, embedded, and stored document chunks.
* Fast and accurate retrieval of relevant sections.
* Summarization and chatbot responses were contextually accurate.
* User-friendly Streamlit interface for interaction.

**5. Deliverables**

* **Chatbot** for querying documents.
* **Summary Generation** of retrieved sections.
* **AI Insights** extraction.
* **GitHub repository** with clean and structured code.
* **Video Demonstration** explaining development and usage.

**6. Future Enhancements**

* Add support for multiple file formats (Word, TXT, etc.).
* Implement more efficient chunking techniques.
* Optimize retrieval speed using different vector search methods.
* Extend to handle multimodal documents (text + images).

This report documents the complete functionality and implementation of the project. Let me know if any sections need further details or modifications.