Document-Understanding AI Assistant: Detailed Documentation

1. Introduction

The Document-Understanding AI Assistant is an innovative tool designed to process and analyze PDF and TXT documents, allowing users to ask questions and receive AI-generated answers based on the document's content. This system leverages cutting-edge technologies including Streamlit for the user interface, FAISS for efficient vector storage, and Anthropic's Claude 2 model for natural language processing.

2. System Architecture

The system is built on a modular architecture comprising several key components:

1. **User Interface**: Streamlit
2. **Document Processing**: PyPDF
3. **Text Embedding**: HuggingFace Embeddings
4. **Vector Storage**: FAISS
5. **Language Model**: Anthropic's Claude 2
6. **Orchestration**: LangChain

2.1 System Flow

1. User uploads a document through the Streamlit interface.
2. The document is processed and text is extracted.
3. The text is split into chunks and embedded.
4. Embeddings are stored in FAISS for efficient retrieval.
5. User asks a question.
6. Relevant text chunks are retrieved from FAISS.
7. The question and relevant chunks are sent to Claude 2 for answer generation.
8. The answer is displayed to the user.

3. Setup Instructions

3.1 Prerequisites

- Python 3.9+
- Git
- Anthropic API key

3.2 Step-by-step Setup

1. Clone the repository:

```
git clone https://github.com/your-username/document-understanding-ai-assistant.git
cd document-understanding-ai-assistant
```

2. Create and activate a virtual environment:

```
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate
```

3. Install dependencies:

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

4. Set up Anthropic API key:
    - Create a file `.streamlit/secrets.toml`
    - Add your API key: `anthropic_api_key = "your-api-key-here"`
5. Run the application:

```
streamlit run app.py
```

4. Usage Guide

4.1 Launching the Application

- Run `streamlit run app.py`
- Open the provided URL in your web browser

4.2 Uploading a Document

- Click on the "Upload a resume (PDF or TXT)" button
- Select a PDF or TXT file from your computer

4.3 Asking Questions

- Choose a predefined question from the dropdown or type your own
- Click "Submit" or press Enter

4.4 Interpreting Results

- The AI-generated answer will appear below the question
- A confidence slider allows you to rate the answer's perceived accuracy

4.5 Providing Feedback

- Select whether the answer was helpful
- If not helpful, you can provide improvement suggestions

5. Design Choices

5.1 Streamlit for UI

- Rapid development, easy to use, Python-based

5.2 FAISS instead of ChromaDB

- Efficient similarity search, works well with high-dimensional vectors

5.3 Claude 3 instead of OpenAI

- Strong performance, potentially lower cost

5.4 LangChain for Orchestration

- Provides a unified interface for various LLM operations

6. Security Considerations

6.1 API Key Management

- Stored in `.streamlit/secrets.toml`, not tracked in Git

6.2 Document Handling

- Documents are processed in-memory and not stored persistently