Here's a step-by-step guide (with code) to build a **LangChain-based chatbot** that uses **OpenAI** to answer questions from **PDF notes**.

□ Prerequisites

Install the required libraries:

```
pip install langchain openai pypdf faiss-cpu tiktoken
```

□ Step-by-Step Code

Step 1: Import Required Modules

Python:-

```python
from langchain.document_loaders import PyPDFLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import FAISS
from langchain.chat_models import ChatOpenAI
from langchain.chains import RetrievalQA
import os
```

Step 2: Set Your OpenAI API Key

Python:-

```python
os.environ["OPENAI_API_KEY"] = "your-openai-api-key"
```

Replace `"your-openai-api-key"` with your actual key.

Step 3: Load and Split PDF

Python:-

```
pdf_path = "your_notes.pdf"  # Replace with your PDF path
loader = PyPDFLoader(pdf_path)
documents = loader.load()

# Split the text into chunks
text_splitter = RecursiveCharacterTextSplitter(chunk_size=1000,
chunk_overlap=200)
chunks = text_splitter.split_documents(documents)
```

Step 4: Create Vector Store (FAISS)

Python:-

```
embedding = OpenAIEmbeddings()
vectorstore = FAISS.from_documents(chunks, embedding)
```

Step 5: Build Retrieval-based QA Chain

Python:-

```
retriever = vectorstore.as_retriever()
llm = ChatOpenAI(temperature=0)

qa_chain = RetrievalQA.from_chain_type(
    llm=llm,
    chain_type="stuff",
    retriever=retriever
)
```

Step 6: Ask Questions

Python:-

```
while True:
    query = input("Ask a question (or type 'exit'): ")
    if query.lower() == 'exit':
        break
    result = qa_chain.run(query)
    print("Answer:", result)
```

Example Questions

You can now ask:

- "What is the definition of photosynthesis?"
- "Summarize chapter 2 of the notes."
- "List the key takeaways from the document."

Optional: Save and Reload Vector Store

To avoid re-processing PDFs every time:

Python:-

```
# Save
vectorstore.save_local("pdf_vectorstore")

# Later, load
vectorstore = FAISS.load_local("pdf_vectorstore", embedding)
retriever = vectorstore.as_retriever()
```

Full Python Code pdf_chatbot.py :-

```
# pdf_chatbot.py

from langchain.document_loaders import PyPDFLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import FAISS
from langchain.chat_models import ChatOpenAI
from langchain.chains import RetrievalQA
```

```
import os

def main():
    # 1. Set your OpenAI API key here
    os.environ["OPENAI_API_KEY"] = "your-openai-api-key"

    # 2. Load your PDF file
    pdf_path = "your_notes.pdf"  # Replace with your PDF filename
    loader = PyPDFLoader(pdf_path)
    documents = loader.load()

    # 3. Split the document into chunks for embeddings
    text_splitter = RecursiveCharacterTextSplitter(chunk_size=1000,
chunk_overlap=200)
    chunks = text_splitter.split_documents(documents)

    # 4. Create embeddings and vector store
    embedding = OpenAIEmbeddings()
    vectorstore = FAISS.from_documents(chunks, embedding)

    # 5. Create retriever and LLM chain for QA
    retriever = vectorstore.as_retriever()
    llm = ChatOpenAI(temperature=0)
    qa_chain = RetrievalQA.from_chain_type(llm=llm, chain_type="stuff",
retriever=retriever)

    print("Chatbot ready! Type 'exit' to quit.")

    # 6. Start interactive QA loop
    while True:
        query = input("\nAsk a question: ")
        if query.strip().lower() == "exit":
            print("Goodbye!")
            break

        answer = qa_chain.run(query)
        print("\nAnswer:", answer)


if __name__ == "__main__":
    main()
```

How to run

1. Replace `"your-openai-api-key"` with your actual OpenAI API key.
2. Replace `"your_notes.pdf"` with your PDF filename (make sure it's in the same folder or use a full path).
3. Run: