

LangChain Chat with Your Data: SFBU Customer Support System – text

This code uses a Retrieval-Augmented Generation (RAG) workflow to build a chatbot that can answer questions based on document content. The workflow involves loading documents, creating a vector database, performing similarity search, initializing an LLM, setting up RetrievalQA and Conversational Retrieval chains, and building a web-based user interface.

Step 1: Overview of the Workflow for RAG:

- **Environment Setup:**

```
1 # Define your OpenAI API key
2 OPENAI_API_KEY = "Replace with your actual API key"
3 USER_AGENT = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36"
```

```
SFBU_Customer_Support.py > CustomerSupportChatbot > call_load_db
1 import os, openai
2 import datetime
3 from dotenv import load_dotenv, find_dotenv
```

- **Load API Key:**

```
# Load the OpenAI API key and select the LLM model based on the current date
load_dotenv(find_dotenv())
openai.api_key = os.getenv('OPENAI_API_KEY')
```

- **LLM Model Selection:**

```
current_date = datetime.datetime.now().date()
llm_name = "gpt-3.5-turbo-0301" if current_date < datetime.date(2023, 9, 2) else "gpt-3.5-turbo"
print(f"Using LLM Model: {llm_name}")
```

Output:






```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 7$ python3 SFBU_Customer_Support.py  
Using LLM Model: gpt-3.5-turbo
```

Step 2: Load Document and Create VectorDB (Vectorstore):

```
SFBU_Customer_Support.py > ...  
29 # Step 2: Load document and create VectorDB (i.e., Vectorstore)  
30 persist_directory = 'docs/chroma/'  
31 embedding = OpenAIEmbeddings()  
32 vectordb = Chroma(persist_directory=persist_directory, embedding_function=embedding)  
33
```

Step 3: Similarity Search to Select Relevant Chunks (Splits):

```
SFBU_Customer_Support.py > ...  
35 # Step 3: Similarity Search to select relevant chunks (splits)  
36 question = "What are major topics?"  
37 docs = vectordb.similarity_search(question)  
38 print("\nNumber of documents: ",len(docs))  
39
```

Name	Date modified	Type	Size
 MachineLearning-Lecture01	31-10-2024 22:49	PDF File	64 KB
 sfbu-2024-2025-university-catalog-8-20-2024	08-11-2024 13:47	PDF File	1,470 KB
 sfbu-2024-2025-university-catalog-8-20-2024_1	08-11-2024 13:47	PDF File	1,470 KB
 sfbu-2024-2025-university-catalog-8-20-2024_2	08-11-2024 13:47	PDF File	1,470 KB
 sfbu-2024-2025-university-catalog-8-20-2024_3	08-11-2024 13:47	PDF File	1,470 KB

Output:

```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 7$ python3 SFBU_Customer_Support.py  
Using LLM Model: gpt-3.5-turbo  
Number of documents: 4
```

Step 4: Create LLM:

```
SFBU_Customer_Support.py > ...  
39 # Step 4: Create LLM  
40 # Initialize the LLM and vectorstore to handle the similarity search and storage of embeddings  
41 llm = ChatOpenAI(model=llm_name, temperature=0)  
42
```

Step 5: RetrievalQA Chain:

```
SFBUCustomerSupport.py > ...
44 # Step 5: RetrievalQA Chain - optional
45 # Step 5.1: Create a prompt template for the QA chain
46 template = """Use the following context to answer the question at the end.
47 If you don't know the answer, say you don't know. Use three sentences maximum.
48 Always end with "thanks for asking!".
49 {context}
50 Question: {question}
51 Helpful Answer: """
52 QA_CHAIN_PROMPT = PromptTemplate(input_variables=["context", "question"], template=template)
53
```

Step 6: ConversationalRetrievalChain:

```
SFBUCustomerSupport.py > ...
54 # Step 6: ConversationalRetrievalChain
55 # Step 6.1: Create Memory - use memory to track conversation history for context continuity
56 memory = ConversationBufferMemory(memory_key="chat_history", return_messages=True)
57 retriever = vectordb.as_retriever()
58 # Step 6.2: QA with ConversationalRetrievalChain - setting up conversational retrieval with memory
59 qa_chain = ConversationalRetrievalChain.from_llm(
60     llm,
61     retriever=retriever,
62     memory=memory
63 )
64
```

Step 7: Create a Chatbot That Works on Your Documents:

```
SFBUCustomerSupport.py > ...
64
65 # Step 7: Create a chatbot that works on your documents
66 # Function to load documents and create vector database with relevant parameters
67 def load_db(file, chain_type="stuff", k=4):
68     loader = PyPDFLoader(file)
69     documents = loader.load()
70     text_splitter = RecursiveCharacterTextSplitter(chunk_size=1000, chunk_overlap=150)
71     docs = text_splitter.split_documents(documents)
72     db = Chroma.from_documents(docs, embedding)
73     retriever = db.as_retriever(search_type="similarity", search_kwargs={"k": k})
74     return ConversationalRetrievalChain.from_llm(
75         llm=llm,
76         retriever=retriever,
77         return_source_documents=True,
78         return_generated_question=True
79     )
80
```

Step 7.1: Create Business Logic:

```
SFBU_Customer_Support.py > ...
81 # Step 7.1: Create Business logic
82 # Define a class for the chatbot, handling loading, conversation, and history management
83 class CustomerSupportChatbot(param.Parameterized):
84     chat_history = param.List([])
85     answer = param.String("")
86     db_query = param.String("")
87     db_response = param.List([])
88
89     def __init__(self, **params):
90         super().__init__(**params)
91         self.panels = []
92         self.loaded_file = "docs/cs229_lectures/sfbu-2024-2025-university-catalog-8-20-2024.pdf"
93         self.qa = load_db(self.loaded_file)
94
95     def call_load_db(self, count):
96         # Load PDF and set up vectorstore for new file if provided
97         if count == 0 or file_input.value is None:
98             return pn.pane.Markdown(f"Loaded File: {self.loaded_file}")
99         else:
100             file_input.save("temp.pdf")
101             self.loaded_file = file_input.filename
102             self.qa = load_db("temp.pdf")
103         self.clr_history()
104         return pn.pane.Markdown(f"Loaded File: {self.loaded_file}")
105
106     def clr_history(self):
107         # Clear the chat history for new interactions
108         self.chat_history.clear()
109         self.panels.clear()
110
111     def convchain(self, query):
112         # Process user input, perform conversational QA and update response panel
113         if not query:
114             return pn.WidgetBox(pn.Row('User:', pn.pane.Markdown("")), scroll=True)
115         result = self.qa.invoke({"question": query, "chat_history": self.chat_history})
116         self.chat_history.extend([(query, result["answer"])])
117         self.db_query = result["generated_question"]
118         self.db_response = result["source_documents"]
119         self.answer = result["answer"]
120         self.panels.extend([
121             pn.Row('User:', pn.pane.Markdown(query)),
122             pn.Row('ChatBot:', pn.pane.Markdown(self.answer))
123         ])
124         inp.value = '' # Clears input field
125         return pn.WidgetBox(*self.panels, scroll=True)
126
127     def display_db_query(self):
128         # Display the generated database query based on user input
129         return pn.pane.Markdown(f"DB query: {self.db_query}")
130
131     def display_db_response(self):
132         # Display the response documents retrieved from the database
133         rlist = [pn.Row(pn.pane.Markdown("Result of DB lookup:"))]
134         for doc in self.db_response:
135             rlist.append(pn.Row(pn.pane.Markdown(str(doc))))
136         return pn.WidgetBox(*rlist, scroll=True)
137
```


Step 7.2: Create a Web-Based User Interface:

```
SFBU_Customer_Support.py > ...
138 # Step 7.2: Create a web-based user interface
139 # Define the UI components and interaction handlers for the web-based chatbot
140 cbfs_instance = CustomerSupportChatbot()
141 inp = pn.widgets.TextInput(name="Your Query")
142 button = pn.widgets.Button(name="Submit Query", button_type="primary")
143 button.on_click(lambda event: cbfs_instance.convchain(inp.value)) # Passes inp.value to convchain
144 file_input = pn.widgets.FileInput(accept=".pdf")
145 button_load = pn.widgets.Button(name="Load PDF", button_type="primary")
146 button_load.on_click(lambda event: cbfs_instance.call_load_db(count=1))
147
148 # Layout to display the web-based chatbot UI
149 dashboard = pn.Column(
150     pn.pane.Markdown("## Customer Support Chatbot"),
151     pn.Row(file_input, button_load),
152     pn.Row(inp, button),
153     cbfs_instance.display_db_query,
154     cbfs_instance.display_db_response
155 )
156
157 # Launch the web interface for the chatbot
158 dashboard.show()
```

Web Interface Output:

```
(venv) vaishnavi@DESKTOP-9V8KJG2:/mnt/c/Users/Mohit/Desktop/Gen AI/Week 7$ python3 SFBU_Customer_Support.py
Using LLM Model: gpt-3.5-turbo

Number of documents: 4
/mnt/c/Users/Mohit/Desktop/Gen AI/Week 7/SFBU_Customer_Support.py:57: LangChainDeprecationWarning: Please see the migration
guide at: https://python.langchain.com/docs/versions/migrating_memory/
  memory = ConversationBufferMemory(memory_key="chat_history", return_messages=True)
Launching server at http://localhost:51180
```

Customer Support Chatbot

Browse...

sfbu-2024-2025-univers....catalog-8-20-2024.pdf

Load PDF

Your Query

Submit Query

DB query: How can I apply to SFBU?

Result of DB lookup:

page_content="2024 – 2025 University Catalog 92
Application Requirements
To apply for admission into a bachelor's degree program, the applicant is required to complete the application form online and submit the following to the SFBU Admissions Office:
📄 Domestic Students:

1. Unofficial and/or Official transcripts from ALL previously attended colleges; first-year applicants are required to submit their official high school transcript upon high school graduation. Applicants must have been in good academic standing at the last institution attended.
A high school/college CGPA below 2.0 does not qualify for admission.
2. An English proficiency document is required for non-native English speakers: An official transcript with English course records or TOEFL/IELTS/ITEP/PTE Academic/Duolingo/Cambridge B2 First test score report or equivalent will suffice. See English Proficiency Requirement below for detailed information on the English entrance requirement." metadata={"page": 92, "source": "docs/cs229_lectures/sfbu-2024-2025-university-catalog-8-20-2024.pdf"}

page_content="2024 – 2025 University Catalog 92
Application Requirements
To apply for admission into a bachelor's degree program, the applicant is required to complete the application form online and submit the following to the SFBU Admissions Office:
📄 Domestic Students:

1. Unofficial and/or Official transcripts from ALL previously attended colleges; first-year applicants are required to submit their official high school transcript upon high school graduation. Applicants must have been in good academic standing at the last institution attended.
A high school/college CGPA below 2.0 does not qualify for admission.
2. An English proficiency document is required for non-native English speakers: An official transcript with English course records or TOEFL/IELTS/ITEP/PTE Academic/Duolingo/Cambridge B2 First test score report or equivalent will suffice. See English Proficiency Requirement below for detailed information on the English entrance requirement." metadata={"page": 92, "source": "temp.pdf"}

page_content="2024 – 2025 University Catalog 7
Frequently Asked Questions
If you have any questions or concerns, please call the University Information number. A recording will give you a choice of offices to contact:
Telephone: (510) 803-SFBU (7328)
The university website address is <https://www.sfbu.edu>.
For Admissions Office: e- mail admissions@sfbu.edu; Telephone: (510) 803-7328 ext. 1
• How can I apply to SFBU?
See admission and application information on page 10 (Admission Policies)
University Academic Programs - page 64
• How can I get an application form? What should I submit for the application?
Start the application by creating an account on the SFBU applicant portal, accessible from the SFBU website. Admissions officers are also available to assist with the application.
For degree programs, the required application materials are listed on SFBU's website in the "Admissions" section and in the "Undergraduate Admissions" and "Graduate" metadata={"page": 7, "source": "docs/cs229_lectures/sfbu-2024-2025-university-catalog-8-20-2024.pdf"}

page_content="2024 – 2025 University Catalog 7
Frequently Asked Questions
If you have any questions or concerns, please call the University Information number. A recording will give you a choice of offices to contact:
Telephone: (510) 803-SFBU (7328)
The university website address is <https://www.sfbu.edu>.
For Admissions Office: e- mail admissions@sfbu.edu; Telephone: (510) 803-7328 ext. 1
• How can I apply to SFBU?
See admission and application information on page 10 (Admission Policies)
University Academic Programs - page 64
• How can I get an application form? What should I submit for the application?
Start the application by creating an account on the SFBU applicant portal, accessible from the SFBU website. Admissions officers are also available to assist with the application.
For degree programs, the required application materials are listed on SFBU's website in the "Admissions" section and in the "Undergraduate Admissions" and "Graduate" metadata={"page": 7, "source": "temp.pdf"}

GitHub URL:

<https://github.com/vaishnavi477/Machine-Learning/tree/main/LangChain%20Chat%20with%20your%20Data/SFBU%20Customer%20Support%20System>

Google Slide:

<https://docs.google.com/presentation/d/1Ok4fm9OKMcT58bjfuOkTZQCHoOO7vXtQMI5cGJWChkE/edit?usp=sharing>