

```
1 import docx2txt
2 from apikey import OPENAI_API_KEY
3 import openai
4 import os
5 import streamlit as st
6 from langchain.text_splitter import CharacterTextSplitter
7 from langchain.embeddings.openai import OpenAIEmbeddings
8 from langchain.vectorstores import FAISS
9 from langchain.chains.question_answering import load_qa_chain
10 from langchain.llms import OpenAI
11 from langchain.callbacks import get_openai_callback
12 from PyPDF2 import PdfReader
13 from docx2pdf import convert
14 from langchain.agents import create_csv_agent
15
16
17 os.environ['OPENAI_API_KEY'] = OPENAI_API_KEY
18 openai.api_base = 'https://api.pawan.krd/v1'
19
20
21 def main():
22     st.set_page_config(page_title="Ask your file")
23     option = st.selectbox(
24         'What is the format of your file?',
25         ('.csv', '.pdf', '.docx'))
26
27
28
29
30     if option == '.csv':
31         st.header("Ask your csv ☐")
32
33         # upload file
34         csv_file = st.file_uploader("Upload your .csv", type=["csv"])
35         if csv_file is not None:
36
37             agent = create_csv_agent(
38                 OpenAI(temperature=0), csv_file, verbose=True)
39
40             user_question = st.text_input("Ask a question about your CSV: ")
41
42             if user_question is not None and user_question != "":
43                 with st.spinner(text="In progress..."):
```

```
44         st.write(agent.run(user_question))
45
46
47
48
49
50     if option == '.pdf':
51         st.header("Ask your pdf 📄")
52
53         # upload file
54         pdf = st.file_uploader("Upload your PDF", type="pdf")
55
56         # extract the text
57         if pdf is not None:
58             pdf_reader = PdfReader(pdf)
59             text = ""
60             for page in pdf_reader.pages:
61                 text += page.extract_text()
62
63             # split into chunks
64             text_splitter = CharacterTextSplitter(
65                 separator="\n",
66                 chunk_size=1000,
67                 chunk_overlap=200,
68                 length_function=len
69             )
70             chunks = text_splitter.split_text(text)
71
72             # create embeddings
73             embeddings = OpenAIEmbeddings()
74             knowledge_base = FAISS.from_texts(chunks, embeddings)
75             #
76             # show user input
77             user_question = st.text_input("Ask a question about your PDF:")
78             if user_question:
79                 docs = knowledge_base.similarity_search(user_question)
80
81                 llm = OpenAI()
82                 chain = load_qa_chain(llm, chain_type="stuff")
83                 with get_openai_callback() as cb:
84                     response = chain.run(input_documents=docs, question=user_question)
85                     print(cb)
86
```

```

87         st.write(response)
88
89
90
91
92
93     if option == '.docx':
94         st.header("Ask your docx")
95
96         # upload file
97         docx_file = st.file_uploader("Upload your .docx", type=["docx"])
98         if docx_file is not None:
99             # Read CSV data into a pandas DataFrame
100            text = docx2txt.process(docx_file)
101
102            # # split into chunks
103            text_splitter = CharacterTextSplitter(
104                separator="\n",
105                chunk_size=1000,
106                chunk_overlap=200,
107                length_function=len
108            )
109            chunks = text_splitter.split_text(text)
110            # create embeddings
111            embeddings = OpenAIEmbeddings()
112            knowledge_base = FAISS.from_texts(chunks, embeddings)
113            #
114            # show user input
115            user_question = st.text_input("Ask a question about your PDF:")
116            if user_question:
117                docs = knowledge_base.similarity_search(user_question)
118
119                llm = OpenAI()
120                chain = load_qa_chain(llm, chain_type="stuff")
121                with get_openai_callback() as cb:
122                    response = chain.run(input_documents=docs, question=user_question)
123                    print(cb)
124
125                st.write(response)
126            #
127            #
128            #
129            #

```

```
130      #
131      #
132      #
133
134
135
136
137
138 if __name__ == '__main__':
139     main()
140
141
```