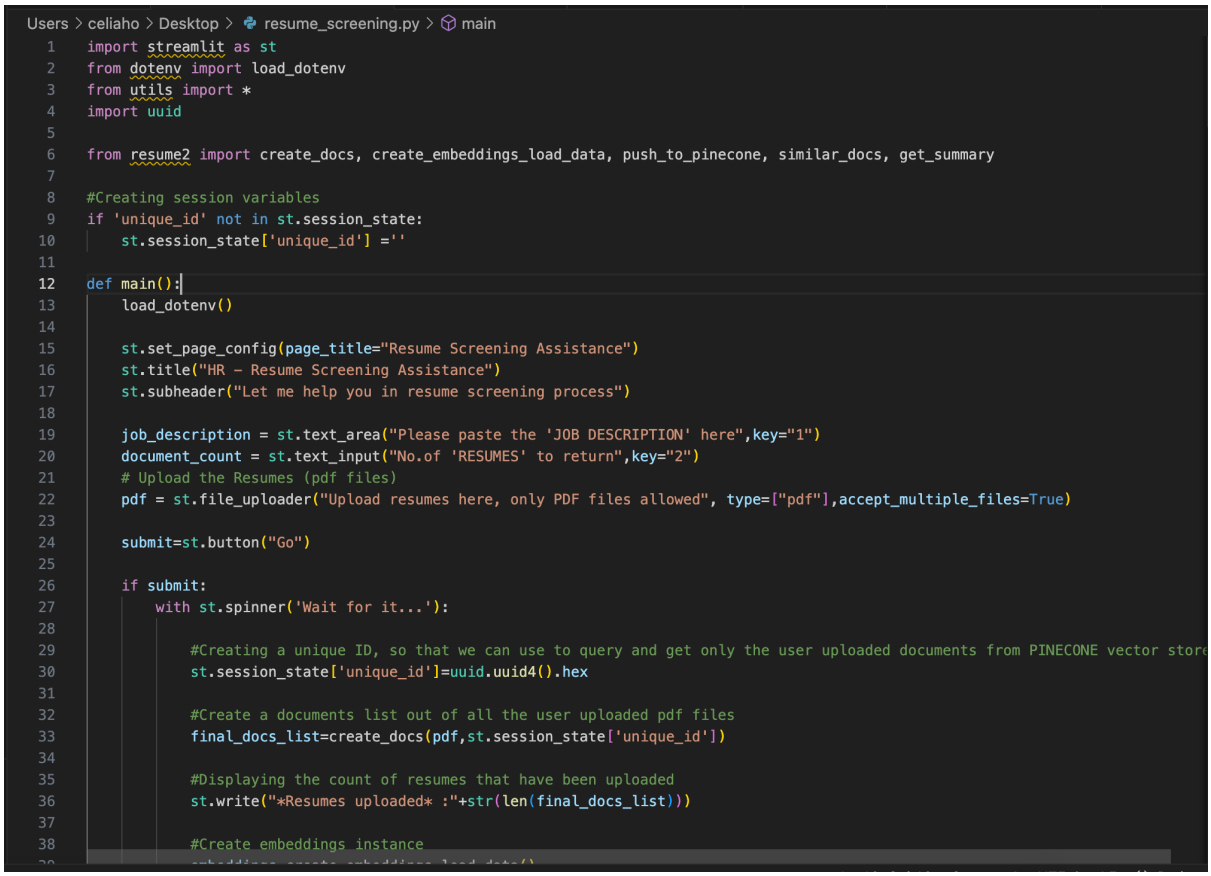
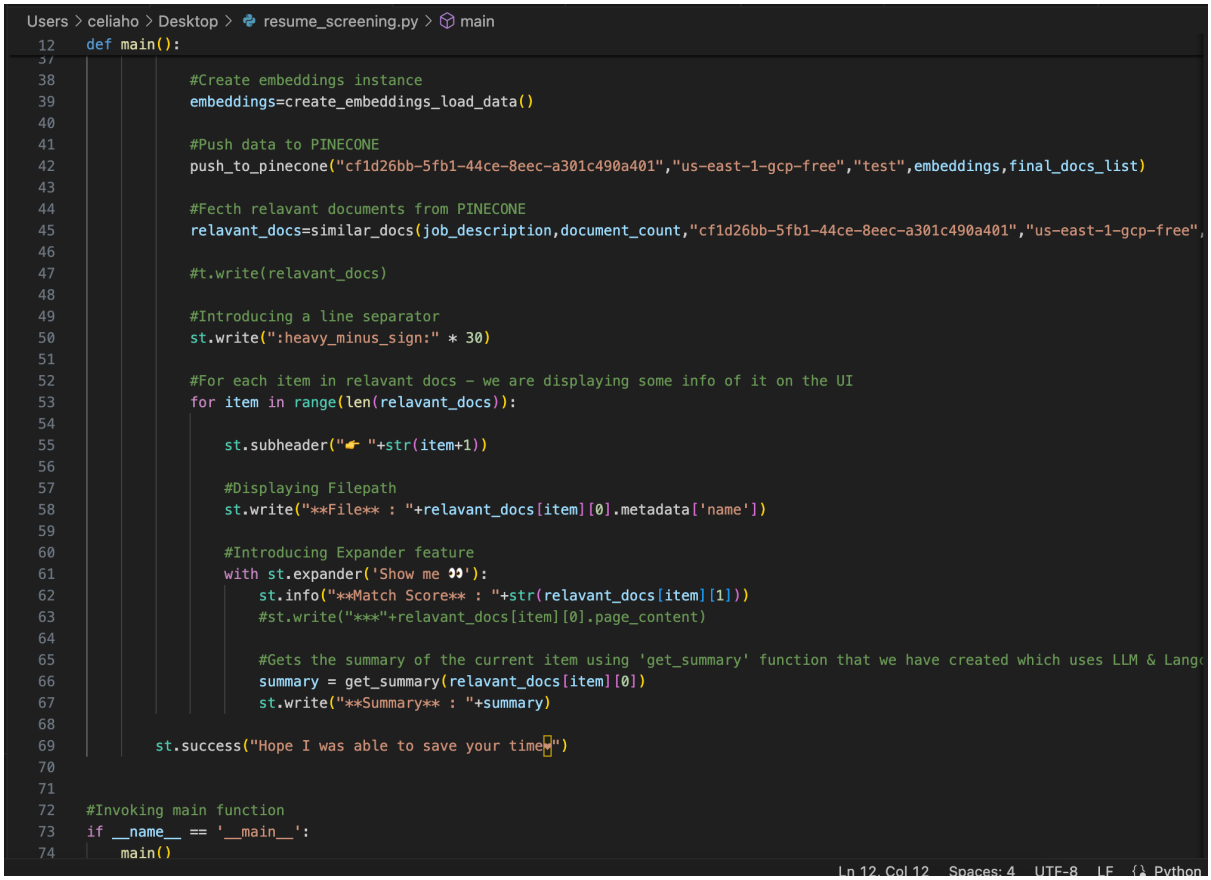
Celia M11209802



```python
import streamlit as st
from dotenv import load_dotenv
from utils import *
import uuid

from resume2 import create_docs, create_embeddings_load_data, push_to_pinecone, similar_docs, get_summary

#Creating session variables
if 'unique_id' not in st.session_state:
    st.session_state['unique_id'] =''

def main():
    load_dotenv()

    st.set_page_config(page_title="Resume Screening Assistance")
    st.title("HR - Resume Screening Assistance")
    st.subheader("Let me help you in resume screening process")

    job_description = st.text_area("Please paste the 'JOB DESCRIPTION' here",key="1")
    document_count = st.text_input("No.of 'RESUMES' to return",key="2")
    # Upload the Resumes (pdf files)
    pdf = st.file_uploader("Upload resumes here, only PDF files allowed", type=["pdf"],accept_multiple_files=True)

    submit=st.button("Go")

    if submit:
        with st.spinner('Wait for it...'):

            #Creating a unique ID, so that we can use to query and get only the user uploaded documents from PINECONE vector store
            st.session_state['unique_id']=uuid.uuid4().hex

            #Create a documents list out of all the user uploaded pdf files
            final_docs_list=create_docs(pdf,st.session_state['unique_id'])

            #Displaying the count of resumes that have been uploaded
            st.write("*Resumes uploaded* :"+str(len(final_docs_list)))

            #Create embeddings instance
```



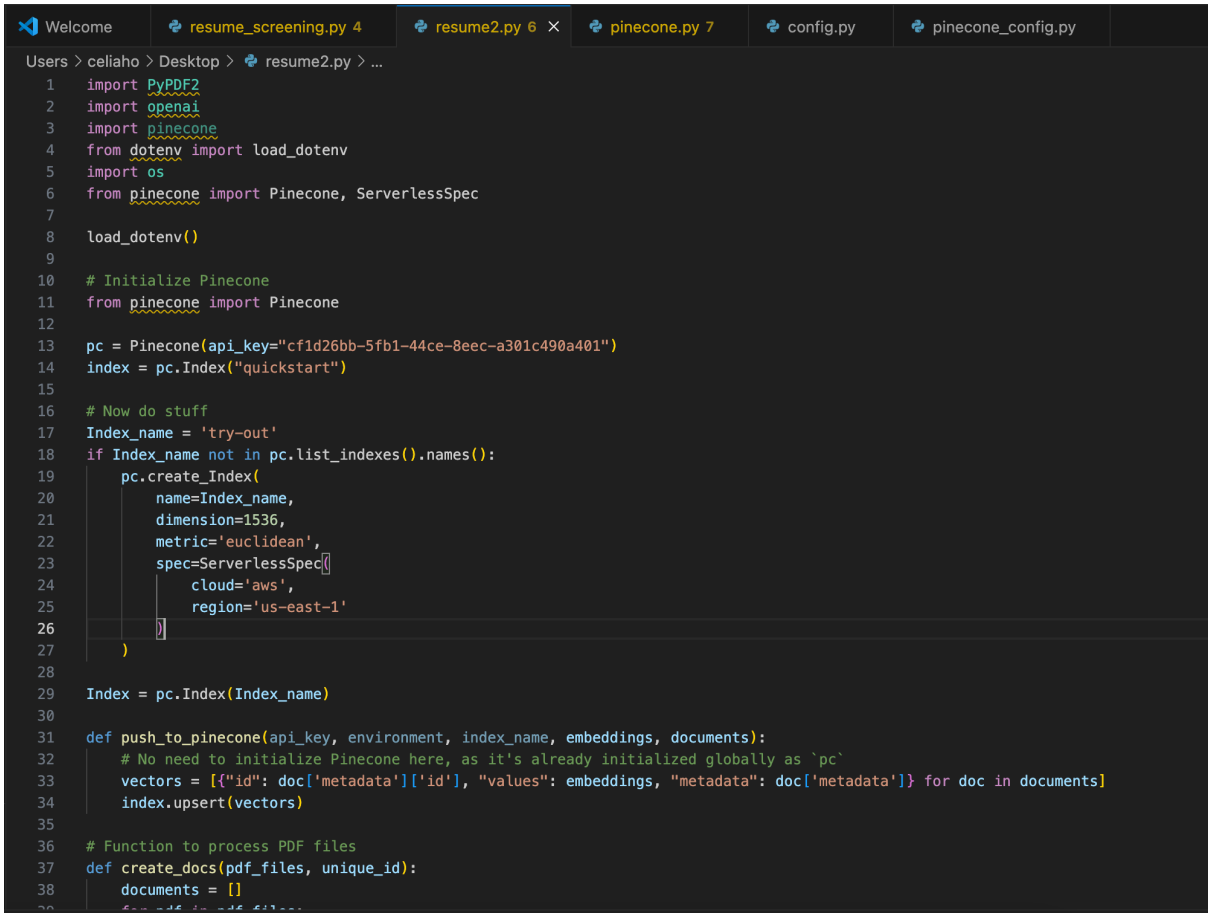
```python
def main():
            #Create embeddings instance
            embeddings=create_embeddings_load_data()

            #Push data to PINECONE
            push_to_pinecone("cf1d26bb-5fb1-44ce-8eec-a301c490a401","us-east-1-gcp-free","test",embeddings,final_docs_list)

            #Fecth relavant documents from PINECONE
            relavant_docs=similar_docs(job_description,document_count,"cf1d26bb-5fb1-44ce-8eec-a301c490a401","us-east-1-gcp-free",

            #t.write(relavant_docs)

            #Introducing a line separator
            st.write(":heavy_minus_sign:" * 30)

            #For each item in relavant docs - we are displaying some info of it on the UI
            for item in range(len(relavant_docs)):

                st.subheader("👉 "+str(item+1))

                #Displaying Filepath
                st.write("**File** : "+relavant_docs[item][0].metadata['name'])

                #Introducing Expander feature
                with st.expander('Show me 👀'):
                    st.info("**Match Score** : "+str(relavant_docs[item][1]))
                    #st.write("***"+relavant_docs[item][0].page_content)

                    #Gets the summary of the current item using 'get_summary' function that we have created which uses LLM & Lang
                    summary = get_summary(relavant_docs[item][0])
                    st.write("**Summary** : "+summary)

        st.success("Hope I was able to save your time❤")


#Invoking main function
if __name__ == '__main__':
    main()
```

Ln 12, Col 12 Spaces: 4 UTF-8 LF {} Python

try-out

METRIC	DIMENSIONS	HOST	CLOUD	REGION	TYPE	VECTOR COUNT
euclidean	1536	https://try-out-hkpumv8.svc.aped-4627-b74a.pinecone.io	AWS	us-east-1	Serverless	0

I started off with ‘pip install pinecone-client’ to ensure the installation of pinecone so I can apply it.



Welcome | resume_screening.py 4 | resume2.py 6 | pinecone.py 7 | config.py | pinecone_config.py

Users > celiaho > Desktop > resume2.py > ...

```python
import PyPDF2
import openai
import pinecone
from dotenv import load_dotenv
import os
from pinecone import Pinecone, ServerlessSpec

load_dotenv()

# Initialize Pinecone
from pinecone import Pinecone

pc = Pinecone(api_key="cf1d26bb-5fb1-44ce-8eec-a301c490a401")
index = pc.Index("quickstart")

# Now do stuff
Index_name = 'try-out'
if Index_name not in pc.list_indexes().names():
    pc.create_Index(
        name=Index_name,
        dimension=1536,
        metric='euclidean',
        spec=ServerlessSpec(
            cloud='aws',
            region='us-east-1'
        )
    )

Index = pc.Index(Index_name)

def push_to_pinecone(api_key, environment, index_name, embeddings, documents):
    # No need to initialize Pinecone here, as it's already initialized globally as `pc`
    vectors = [{"id": doc['metadata']['id'], "values": embeddings, "metadata": doc['metadata']} for doc in documents]
    index.upsert(vectors)

# Function to process PDF files
def create_docs(pdf_files, unique_id):
    documents = []
```

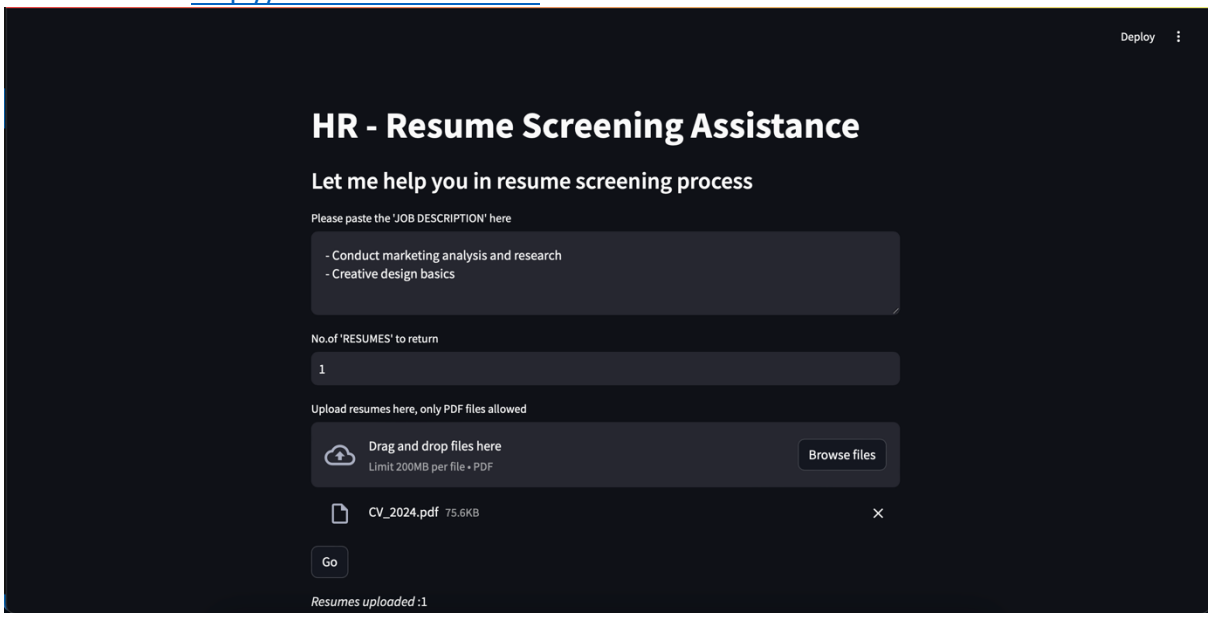
Welcome | resume_screening.py 4 | resume2.py 6 | pinecone.py 7 | config.py | pinecone_config.py

Users > celiaho > Desktop > resume2.py > ...

```python
def push_to_pinecone(api_key, environment, index_name, embeddings, documents):
    index.upsert(vectors)

# Function to process PDF files
def create_docs(pdf_files, unique_id):
    documents = []
    for pdf in pdf_files:
        reader = PyPDF2.PdfReader(pdf)
        content = ""
        for page in reader.pages:
            content += page.extract_text()
        documents.append({"content": content, "metadata": {"name": pdf.name, "id": unique_id}})
    return documents

# Function to create/load embeddings
def create_embeddings_load_data():
    # Placeholder implementation. Replace with actual embedding creation code.
    return None

# Function to fetch similar documents
def similar_docs(job_description, document_count, api_key, environment, index_name, embeddings, unique_id):
    # No need to initialize Pinecone here, as we're not directly using it
    query_vector = embeddings  # Replace with actual query embedding for job description
    result = index.query(query_vector, top_k=int(document_count), include_metadata=True)
    return [(res['metadata'], res['score']) for res in result['matches']]

# Function to get summary of a document
def get_summary(document):
    # Placeholder for document summarization. Replace with actual code.
    openai.api_key = os.getenv("OPENAI_API_KEY")
    response = openai.Completion.create(
      engine="davinci",
      prompt=f"Summarize the following document: {document['content']}",
      max_tokens=150
    )
    return response.choices[0].text.strip()
```

I was able to load the website with:

Local URL: http://localhost:8501
Network URL: http://192.168.0.153:8501



But, I still received an error that it expects a list or list like data structure.

Resumes uploaded :1

ListConversionException: Expected a list or list-like data structure, but got: None

Traceback:

```
File "/Users/celiaho/HuggingFaceGuidedTourForMac/lib/python3.12/site-packages/
    exec(code, module.__dict__)
File "/Users/celiaho/Desktop/resume_screening.py", line 74, in <module>
    main()
File "/Users/celiaho/Desktop/resume_screening.py", line 42, in main
    push_to_pinecone("cf1d26bb-5fb1-44ce-8eec-a301c490a401","us-east-1-gcp-fre
File "/Users/celiaho/Desktop/resume2.py", line 34, in push_to_pinecone
    index.upsert(vectors)
File "/Users/celiaho/HuggingFaceGuidedTourForMac/lib/python3.12/site-packages/
    return func(*args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^
File "/Users/celiaho/HuggingFaceGuidedTourForMac/lib/python3.12/site-packages/
    return self._upsert_batch(vectors, namespace, _check_type, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "/Users/celiaho/HuggingFaceGuidedTourForMac/lib/python3.12/site-packages/
    vectors=list(map(vec_builder, vectors)),
            ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "/Users/celiaho/HuggingFaceGuidedTourForMac/lib/python3.12/site-packages/
```