Final Project Presentation on AI Interviewer

Name: Shail Patel
CWID: 885175489

Introduction & Purpose

Introduction to AI-Driven Interview Preparation

- **Background:** Traditional interview preparation often fails to address individual needs and lacks interaction.
- **Objective:** To modernize interview training through an AI-driven tool that personalizes preparation based on user profiles.
- **Key Technologies:** Utilizes OpenAI's GPT-4 for realistic scenario simulation and Whisper for speech-to-text to mimic real interactions.
- **Benefits:** Provides dynamic, tailored training that enhances candidate preparedness across various interview types.

Technical Details & Methodology

Leveraging AI for Enhanced Interview Training

AI Technologies: Introduction to how GPT-4 processes natural language and Whisper converts spoken responses to text, enhancing the realism of practice interviews.

Methodology Overview:

- Data Engineering: Handling diverse data types including resumes and job descriptions.
- Prompt Engineering: Creating context-specific prompts that guide AI to generate relevant questions.
- Feature Engineering: Extracting and using features like education and skills from resumes to tailor questions and feedback.

System Workflow & User Interaction

Workflow and User Engagement

System Workflow: Detailed walkthrough from resume upload to receiving feedback, illustrating the step-by-step user journey within the AI system.

User Interaction:

- Users can select different types of interviews (behavioral, technical, resume-focused).
- Interaction with AI-generated questions based on user's resume and selected job description.

Real-Time Feedback: Highlighting how immediate AI-generated feedback helps users refine their answers and improve interview skills.

Practical Application & Advantages

Real-World Applications and System Benefits

Case Studies: Brief examples or testimonials from users who have significantly improved their interview skills using the AI Interviewer.

Advantages:

- Customization to individual needs enhances preparation effectiveness.
- Real-time feedback mechanism helps in quick skill enhancement.
- Preparation for a wide range of interviews, making users versatile.

User Experience: Share insights into the user-friendly nature of the system and how it has been received by the job-seeking community.

Impact, Conclusions, and Future Directions

Impact: Discussion on how the AI Interviewer has transformed interview preparation, with personalization at its core.

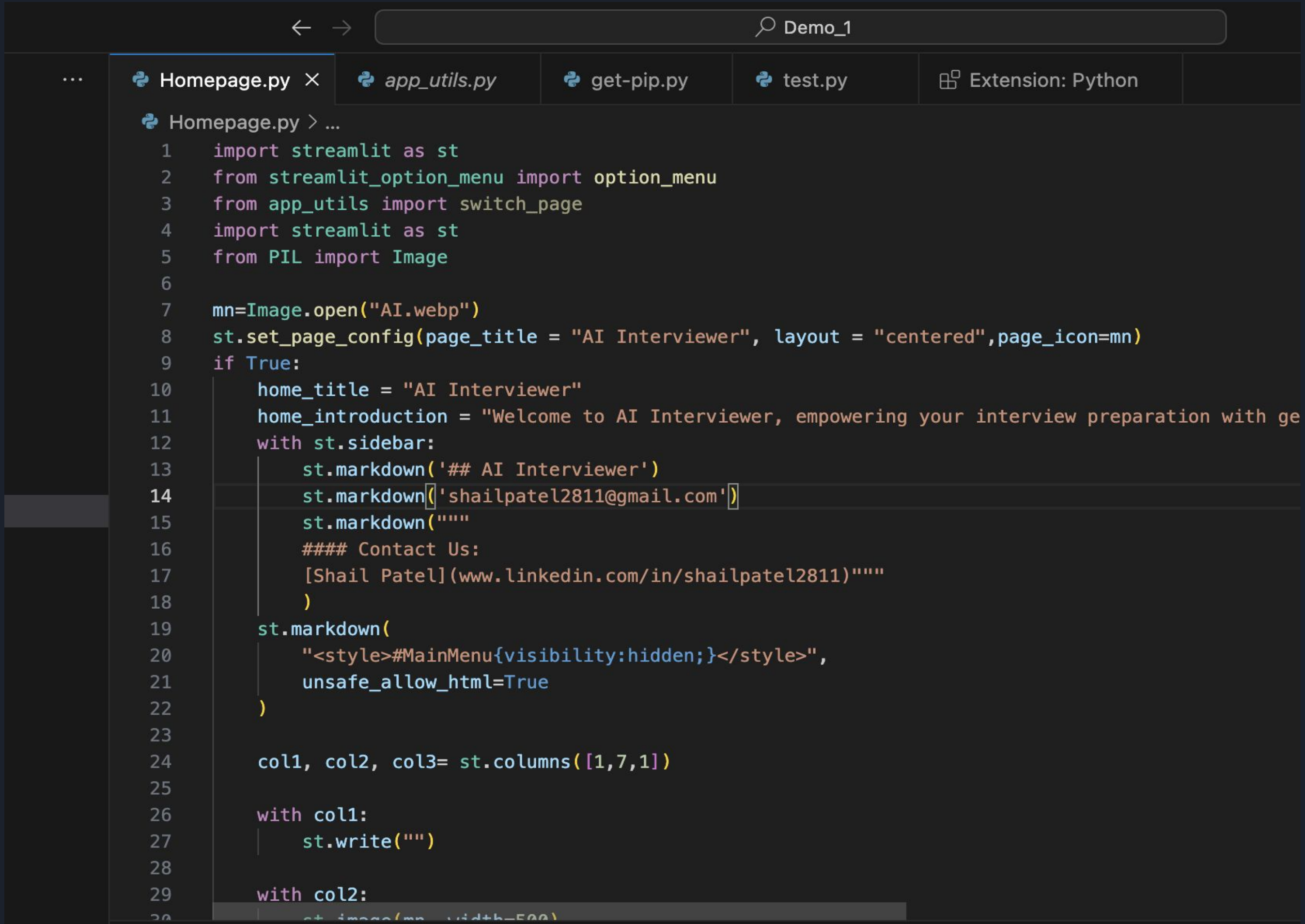
Key Learnings: Reflections on the integration of AI in educational tools and its effectiveness in real-world applications.

Future Directions:

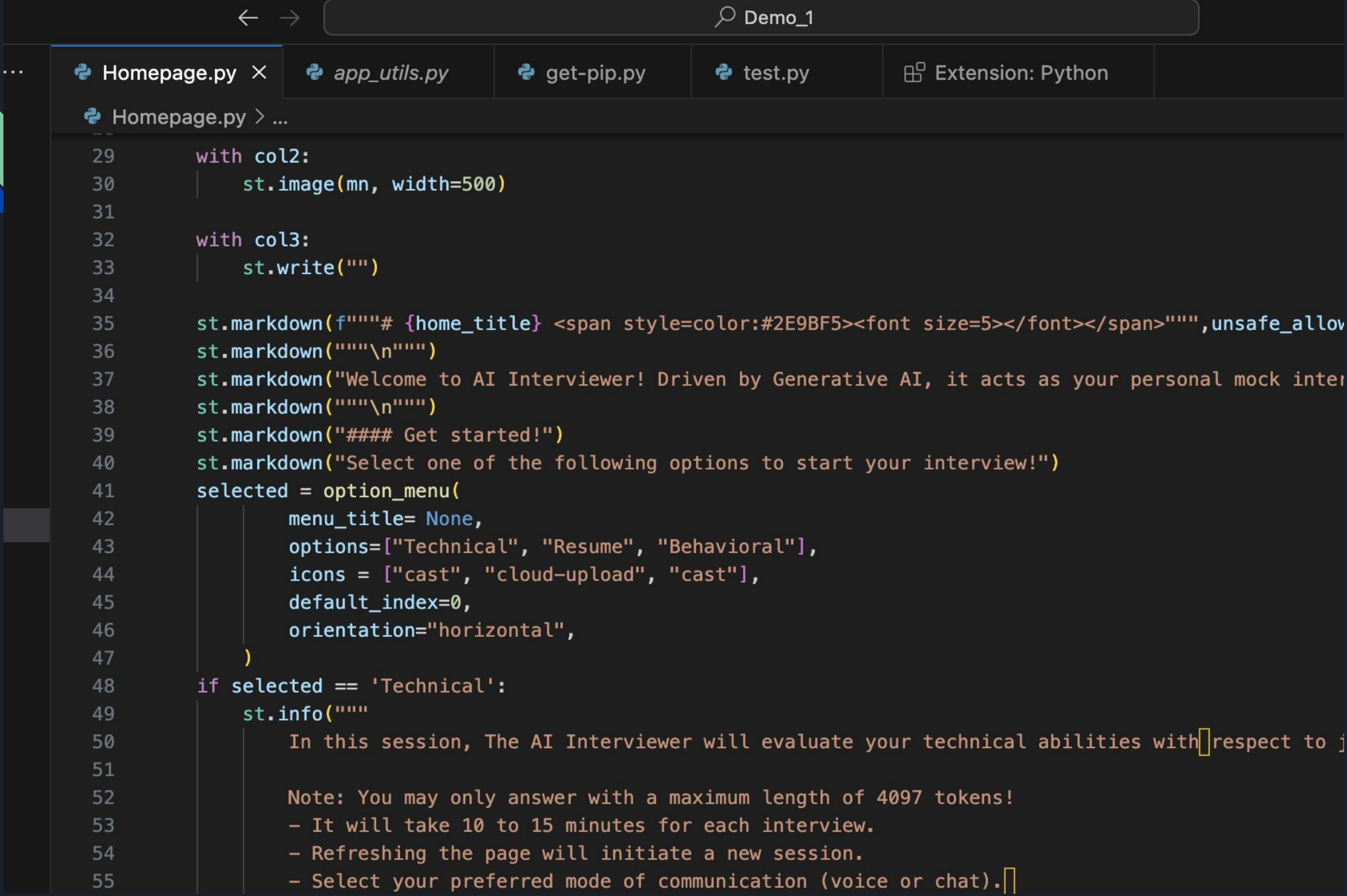
- Expansion to more languages and job sectors.
- Integration of additional features like career advice and market trends.

Closing Thoughts: Emphasize the project's role in shaping future interview preparation tools and potential adaptations for educational sectors.

homepage.py



```python
import streamlit as st
from streamlit_option_menu import option_menu
from app_utils import switch_page
import streamlit as st
from PIL import Image

mn=Image.open("AI.webp")
st.set_page_config(page_title = "AI Interviewer", layout = "centered",page_icon=mn)
if True:
    home_title = "AI Interviewer"
    home_introduction = "Welcome to AI Interviewer, empowering your interview preparation with ge
    with st.sidebar:
        st.markdown('## AI Interviewer')
        st.markdown('shailpatel2811@gmail.com')
        st.markdown("""
        #### Contact Us:
        [Shail Patel](www.linkedin.com/in/shailpatel2811)"""
        )
    st.markdown(
        "<style>#MainMenu{visibility:hidden;}</style>",
        unsafe_allow_html=True
    )

    col1, col2, col3= st.columns([1,7,1])

    with col1:
        st.write("")

    with col2:
        st.image(mn, width=500)
```



```python
    with col2:
        st.image(mn, width=500)

    with col3:
        st.write("")

    st.markdown(f"""# {home_title} <span style=color:#2E9BF5><font size=5></font></span>""",unsafe_allow
    st.markdown("""\n""")
    st.markdown("Welcome to AI Interviewer! Driven by Generative AI, it acts as your personal mock inter
    st.markdown("""\n""")
    st.markdown("#### Get started!")
    st.markdown("Select one of the following options to start your interview!")
    selected = option_menu(
            menu_title= None,
            options=["Technical", "Resume", "Behavioral"],
            icons = ["cast", "cloud-upload", "cast"],
            default_index=0,
            orientation="horizontal",
        )
    if selected == 'Technical':
        st.info("""
            In this session, The AI Interviewer will evaluate your technical abilities with respect to

            Note: You may only answer with a maximum length of 4097 tokens!
            - It will take 10 to 15 minutes for each interview.
            - Refreshing the page will initiate a new session.
            - Select your preferred mode of communication (voice or chat).
```

Share

Homepage

Behavioral Interview

Resume based Interview

Technical Interview

AI Interviewer

Contact Us:

[Email]

[Shail Patel]

AI Interviewer

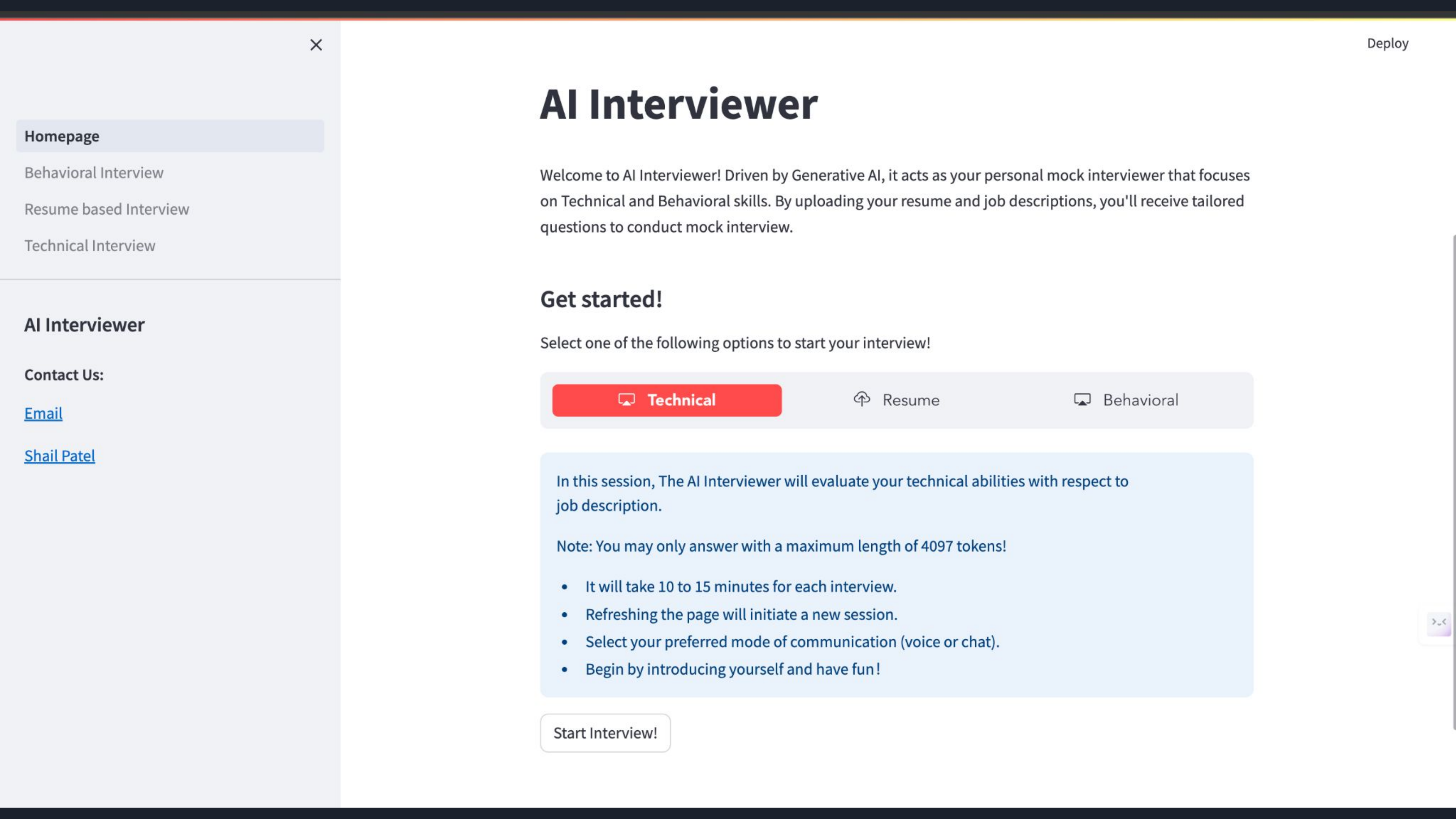
Welcome to AI Interviewer! Driven by Generative AI, it acts as your personal mock interviewer that focuses on Technical and Behavioral skills. By uploading your resume and job descriptions, you'll receive tailored questions to conduct mock interview.

Get started!

Select one of the following options to start your interview!

Technical | Resume | Behavioral

Manage app



AI Interviewer

Contact Us:

Email

Shail Patel

AI Interviewer

Welcome to AI Interviewer! Driven by Generative AI, it acts as your personal mock interviewer that focuses on Technical and Behavioral skills. By uploading your resume and job descriptions, you'll receive tailored questions to conduct mock interview.

Get started!

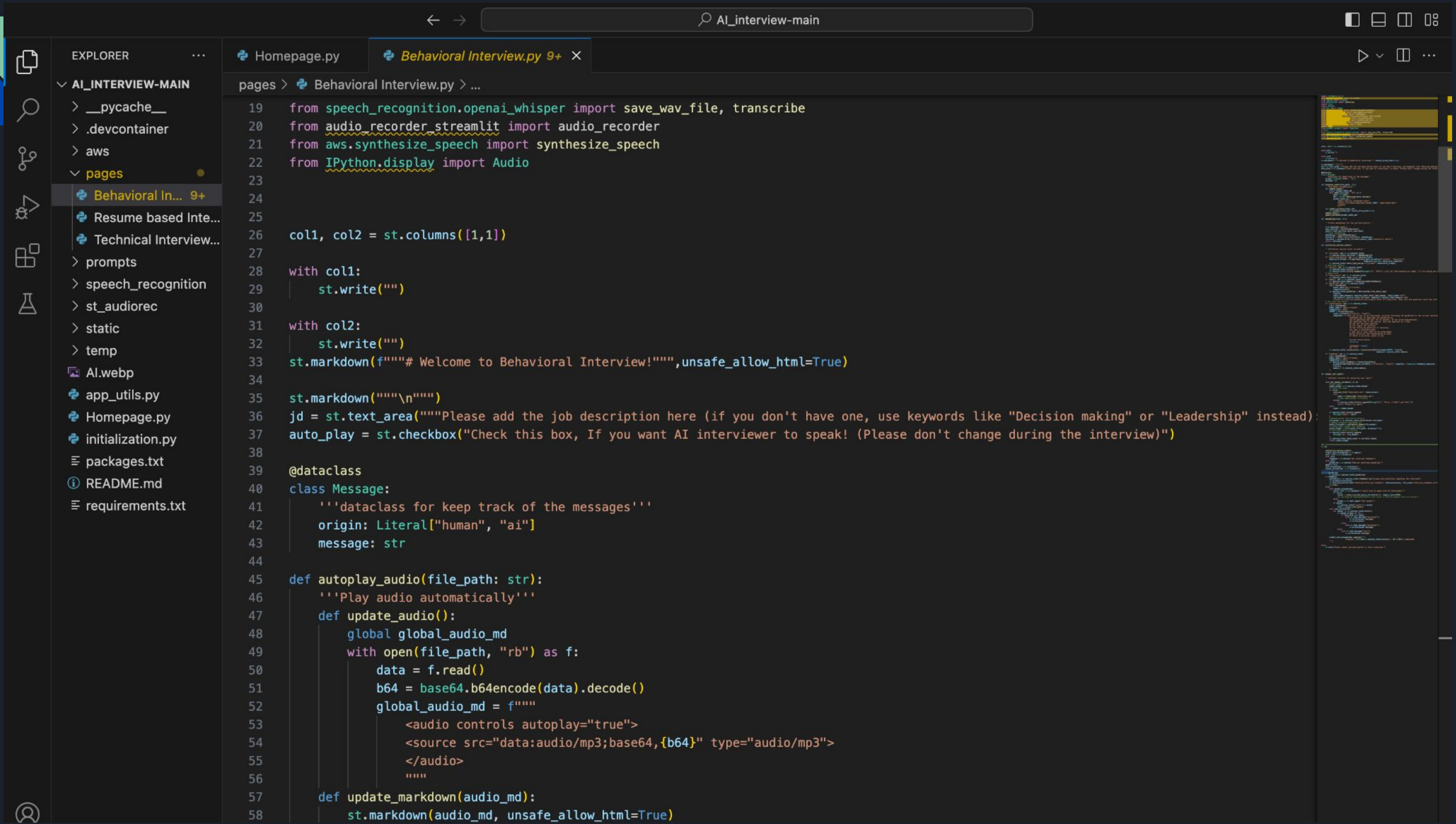
Select one of the following options to start your interview!

In this session, The AI Interviewer will evaluate your technical abilities with respect to job description.

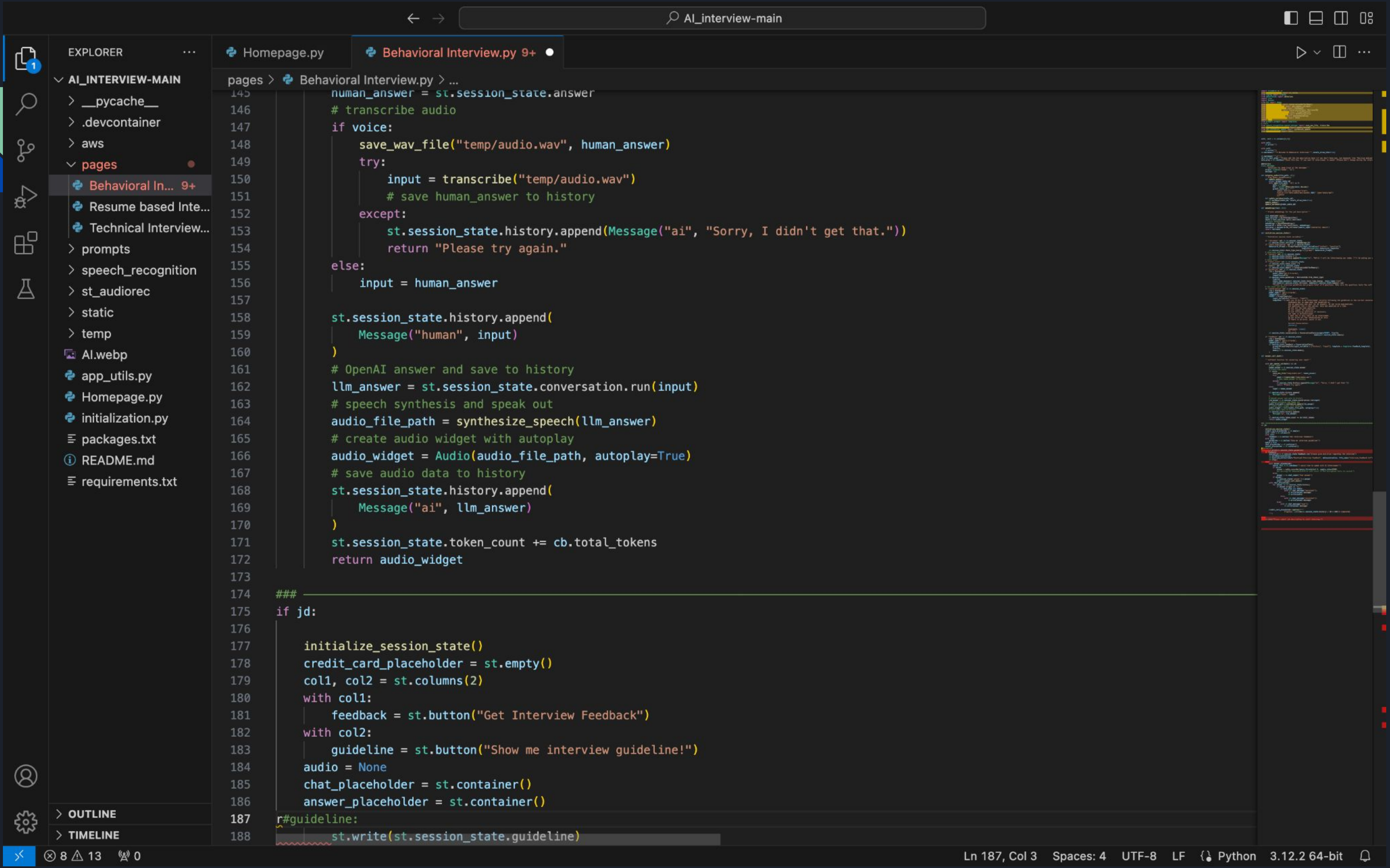
Note: You may only answer with a maximum length of 4097 tokens!

- It will take 10 to 15 minutes for each interview.
- Refreshing the page will initiate a new session.
- Select your preferred mode of communication (voice or chat).
- Begin by introducing yourself and have fun!

Behavioral Interview Page



```python
from speech_recognition.openai_whisper import save_wav_file, transcribe
from audio_recorder_streamlit import audio_recorder
from aws.synthesize_speech import synthesize_speech
from IPython.display import Audio



col1, col2 = st.columns([1,1])

with col1:
    st.write("")

with col2:
    st.write("")
st.markdown(f"""# Welcome to Behavioral Interview!""",unsafe_allow_html=True)

st.markdown("""\n""")
jd = st.text_area("""Please add the job description here (if you don't have one, use keywords like "Decision making" or "Leadership" instead)
auto_play = st.checkbox("Check this box, If you want AI interviewer to speak! (Please don't change during the interview)")

@dataclass
class Message:
    '''dataclass for keep track of the messages'''
    origin: Literal["human", "ai"]
    message: str

def autoplay_audio(file_path: str):
    '''Play audio automatically'''
    def update_audio():
        global global_audio_md
        with open(file_path, "rb") as f:
            data = f.read()
            b64 = base64.b64encode(data).decode()
            global_audio_md = f"""
                <audio controls autoplay="true">
                <source src="data:audio/mp3;base64,{b64}" type="audio/mp3">
                </audio>
                """
    def update_markdown(audio_md):
        st.markdown(audio_md, unsafe_allow_html=True)
```



AI_interview-main

EXPLORER

AI_INTERVIEW-MAIN
- __pycache__
- .devcontainer
- aws
- pages
  - Behavioral In... 9+
  - Resume based Inte...
  - Technical Interview...
- prompts
- speech_recognition
- st_audiorec
- static
- temp
- AI.webp
- app_utils.py
- Homepage.py
- initialization.py
- packages.txt
- README.md
- requirements.txt

OUTLINE

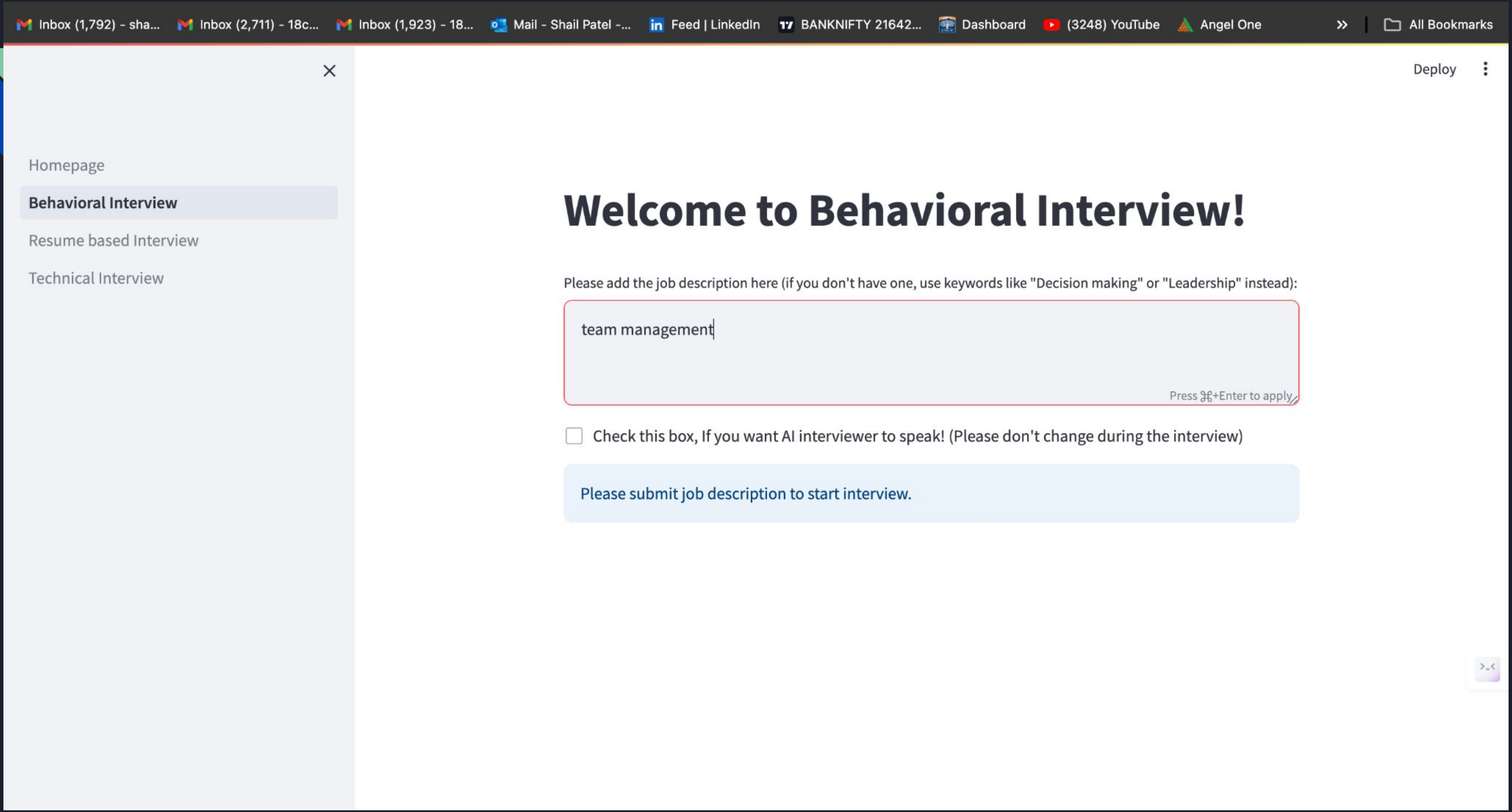
TIMELINE

Homepage.py | Behavioral Interview.py 9+

pages > Behavioral Interview.py > ...

```python
        human_answer = st.session_state.answer
        # transcribe audio
        if voice:
            save_wav_file("temp/audio.wav", human_answer)
            try:
                input = transcribe("temp/audio.wav")
                # save human_answer to history
            except:
                st.session_state.history.append(Message("ai", "Sorry, I didn't get that."))
                return "Please try again."
        else:
            input = human_answer

        st.session_state.history.append(
            Message("human", input)
        )
        # OpenAI answer and save to history
        llm_answer = st.session_state.conversation.run(input)
        # speech synthesis and speak out
        audio_file_path = synthesize_speech(llm_answer)
        # create audio widget with autoplay
        audio_widget = Audio(audio_file_path, autoplay=True)
        # save audio data to history
        st.session_state.history.append(
            Message("ai", llm_answer)
        )
        st.session_state.token_count += cb.total_tokens
        return audio_widget

###
if jd:

    initialize_session_state()
    credit_card_placeholder = st.empty()
    col1, col2 = st.columns(2)
    with col1:
        feedback = st.button("Get Interview Feedback")
    with col2:
        guideline = st.button("Show me interview guideline!")
    audio = None
    chat_placeholder = st.container()
    answer_placeholder = st.container()
r#guideline:
        st.write(st.session_state.guideline)
```

⊗ 8 ⚠ 13 0 | Ln 187, Col 3 Spaces: 4 UTF-8 LF Python 3.12.2 64-bit



Welcome to Behavioral Interview!

Please add the job description here (if you don't have one, use keywords like "Decision making" or "Leadership" instead):

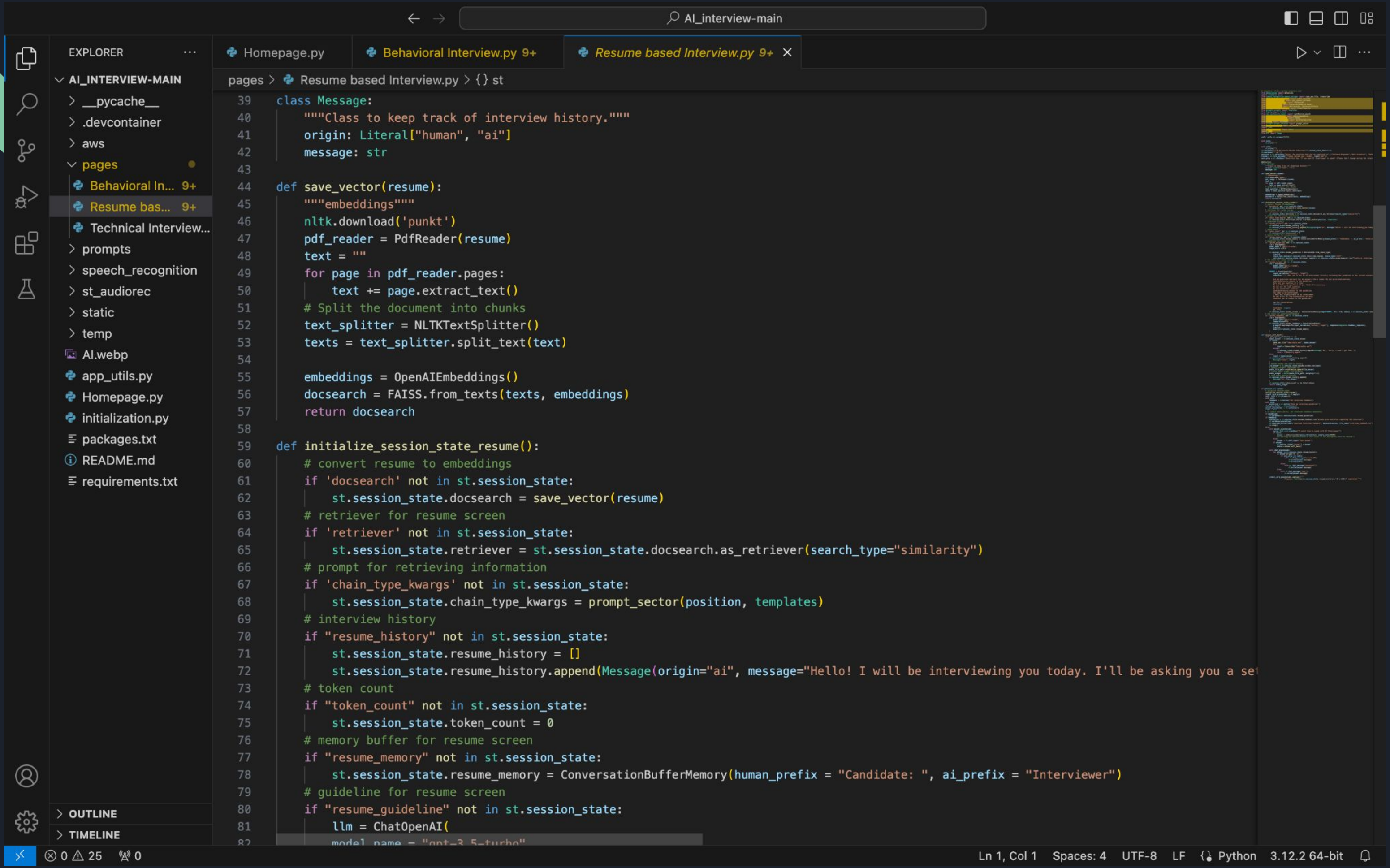
team management

Press ⌘+Enter to apply

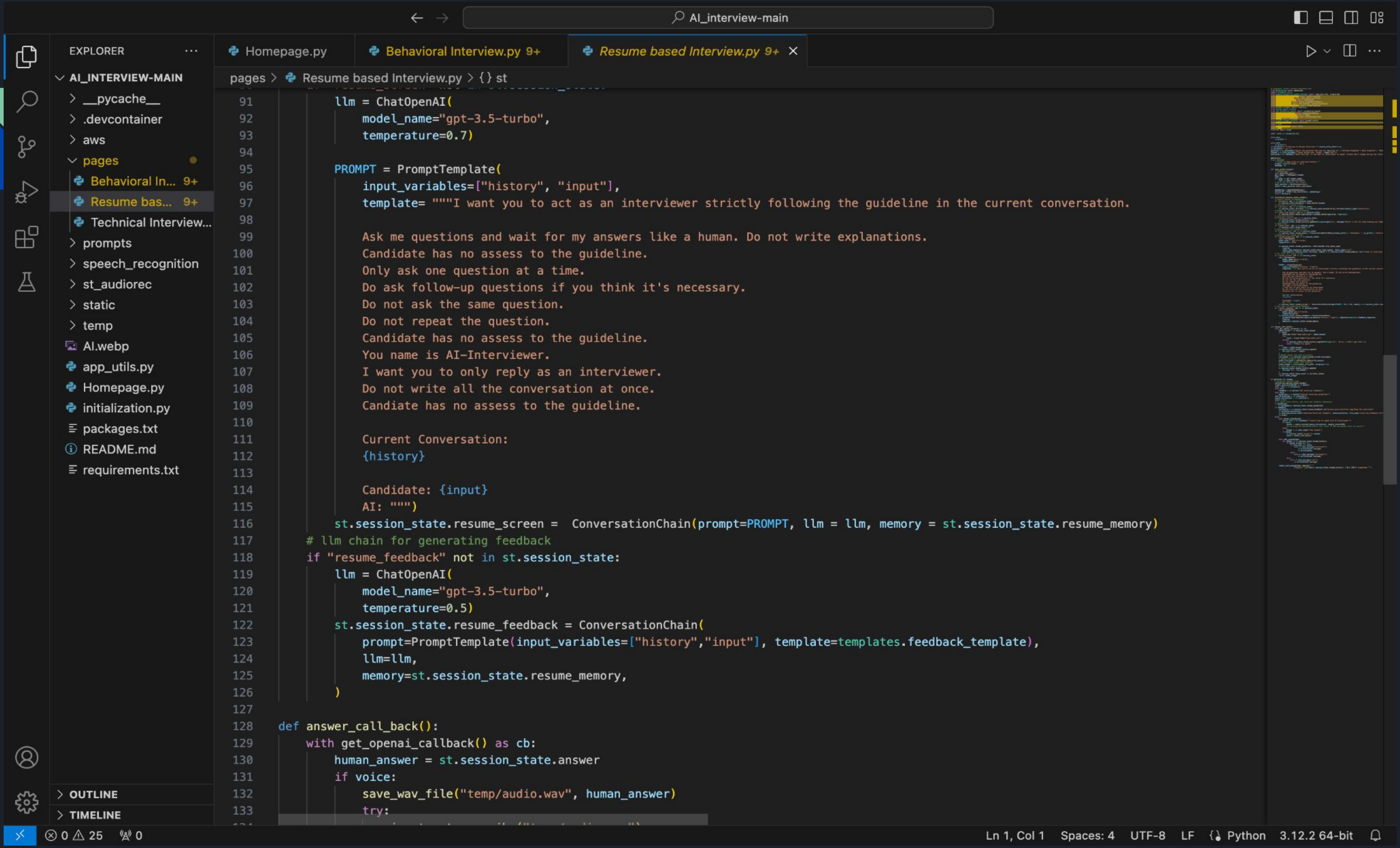
☐ Check this box, If you want AI interviewer to speak! (Please don't change during the interview)

Please submit job description to start interview.

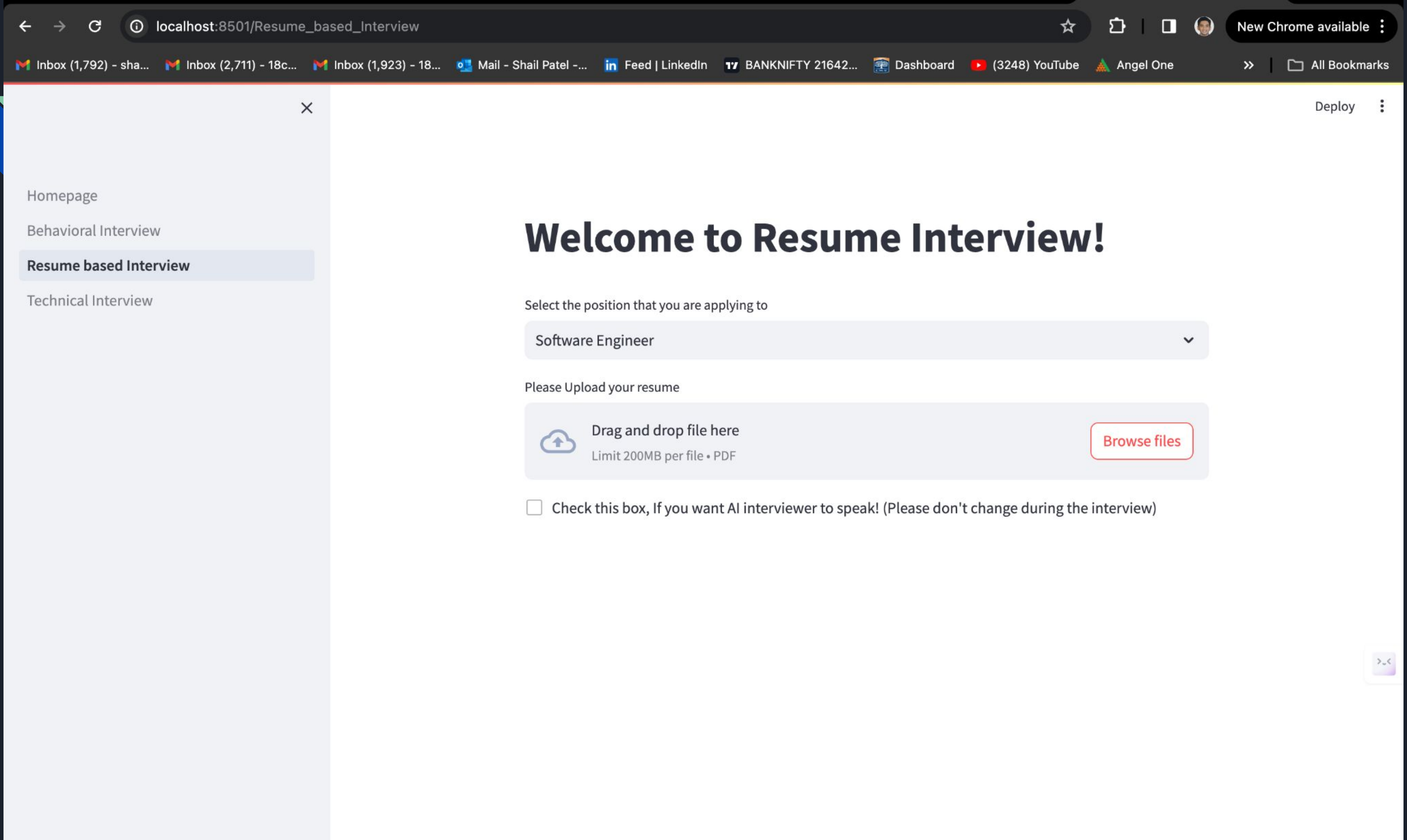
Resume Based Interview Page



```python
class Message:
    """Class to keep track of interview history."""
    origin: Literal["human", "ai"]
    message: str

def save_vector(resume):
    """embeddings"""
    nltk.download('punkt')
    pdf_reader = PdfReader(resume)
    text = ""
    for page in pdf_reader.pages:
        text += page.extract_text()
    # Split the document into chunks
    text_splitter = NLTKTextSplitter()
    texts = text_splitter.split_text(text)

    embeddings = OpenAIEmbeddings()
    docsearch = FAISS.from_texts(texts, embeddings)
    return docsearch

def initialize_session_state_resume():
    # convert resume to embeddings
    if 'docsearch' not in st.session_state:
        st.session_state.docsearch = save_vector(resume)
    # retriever for resume screen
    if 'retriever' not in st.session_state:
        st.session_state.retriever = st.session_state.docsearch.as_retriever(search_type="similarity")
    # prompt for retrieving information
    if 'chain_type_kwargs' not in st.session_state:
        st.session_state.chain_type_kwargs = prompt_sector(position, templates)
    # interview history
    if "resume_history" not in st.session_state:
        st.session_state.resume_history = []
        st.session_state.resume_history.append(Message(origin="ai", message="Hello! I will be interviewing you today. I'll be asking you a se
    # token count
    if "token_count" not in st.session_state:
        st.session_state.token_count = 0
    # memory buffer for resume screen
    if "resume_memory" not in st.session_state:
        st.session_state.resume_memory = ConversationBufferMemory(human_prefix = "Candidate: ", ai_prefix = "Interviewer")
    # guideline for resume screen
    if "resume_guideline" not in st.session_state:
        llm = ChatOpenAI(
            model_name = "gpt-3.5-turbo",
```



```python
        llm = ChatOpenAI(
            model_name="gpt-3.5-turbo",
            temperature=0.7)

        PROMPT = PromptTemplate(
            input_variables=["history", "input"],
            template= """I want you to act as an interviewer strictly following the guideline in the current conversation.

            Ask me questions and wait for my answers like a human. Do not write explanations.
            Candidate has no assess to the guideline.
            Only ask one question at a time.
            Do ask follow-up questions if you think it's necessary.
            Do not ask the same question.
            Do not repeat the question.
            Candidate has no assess to the guideline.
            You name is AI-Interviewer.
            I want you to only reply as an interviewer.
            Do not write all the conversation at once.
            Candiate has no assess to the guideline.

            Current Conversation:
            {history}

            Candidate: {input}
            AI: """)
        st.session_state.resume_screen =  ConversationChain(prompt=PROMPT, llm = llm, memory = st.session_state.resume_memory)
    # llm chain for generating feedback
    if "resume_feedback" not in st.session_state:
        llm = ChatOpenAI(
            model_name="gpt-3.5-turbo",
            temperature=0.5)
        st.session_state.resume_feedback = ConversationChain(
            prompt=PromptTemplate(input_variables=["history","input"], template=templates.feedback_template),
            llm=llm,
            memory=st.session_state.resume_memory,
        )

def answer_call_back():
    with get_openai_callback() as cb:
        human_answer = st.session_state.answer
        if voice:
            save_wav_file("temp/audio.wav", human_answer)
            try:
```



Welcome to Resume Interview!

Select the position that you are applying to

Software Engineer

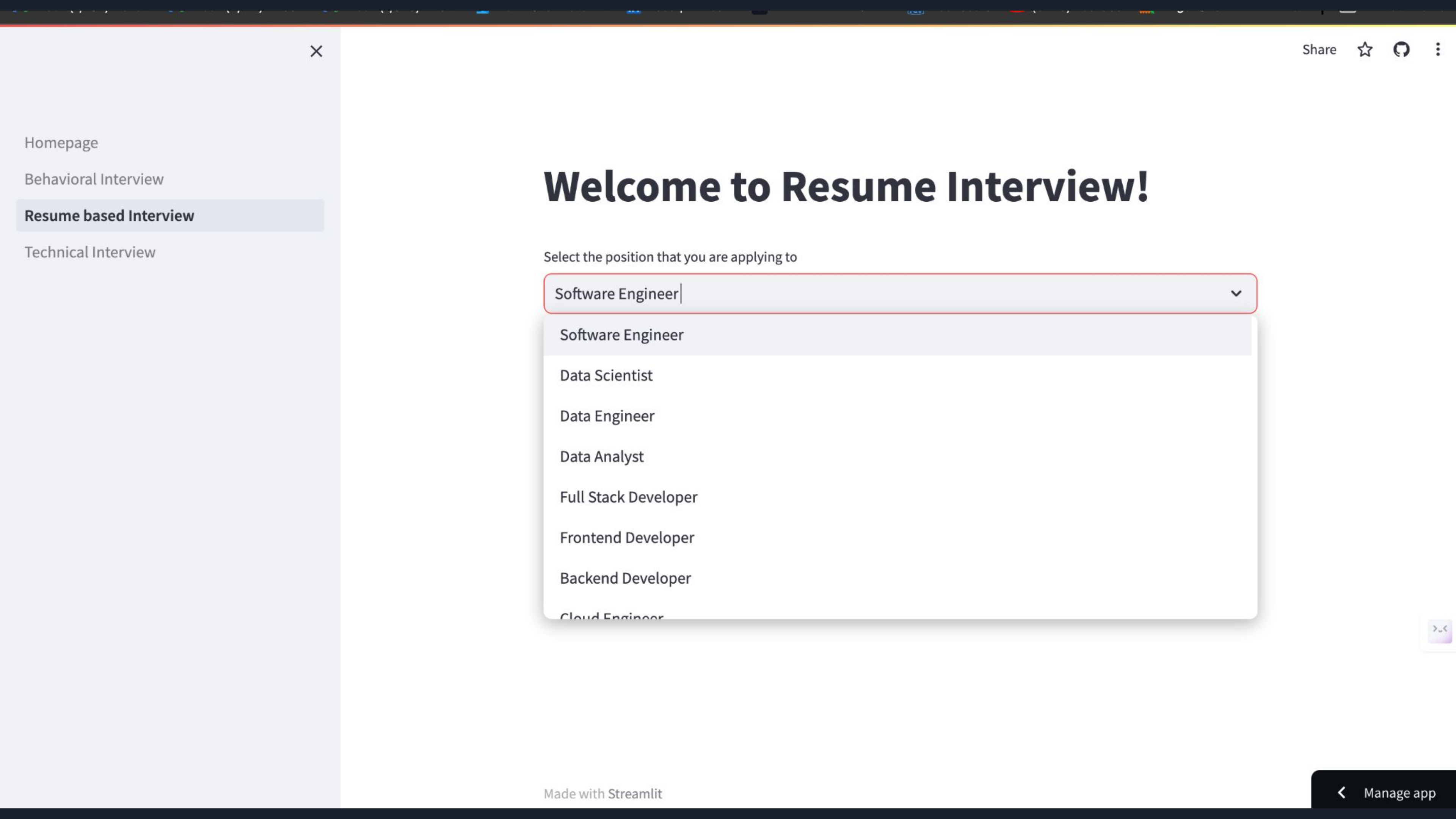
Please Upload your resume

Drag and drop file here

Limit 200MB per file • PDF

Browse files

☐ Check this box, If you want AI interviewer to speak! (Please don't change during the interview)



Share

Homepage

Behavioral Interview

Resume based Interview

Technical Interview

Welcome to Resume Interview!

Select the position that you are applying to

Software Engineer

Software Engineer

Data Scientist

Data Engineer

Data Analyst

Full Stack Developer

Frontend Developer

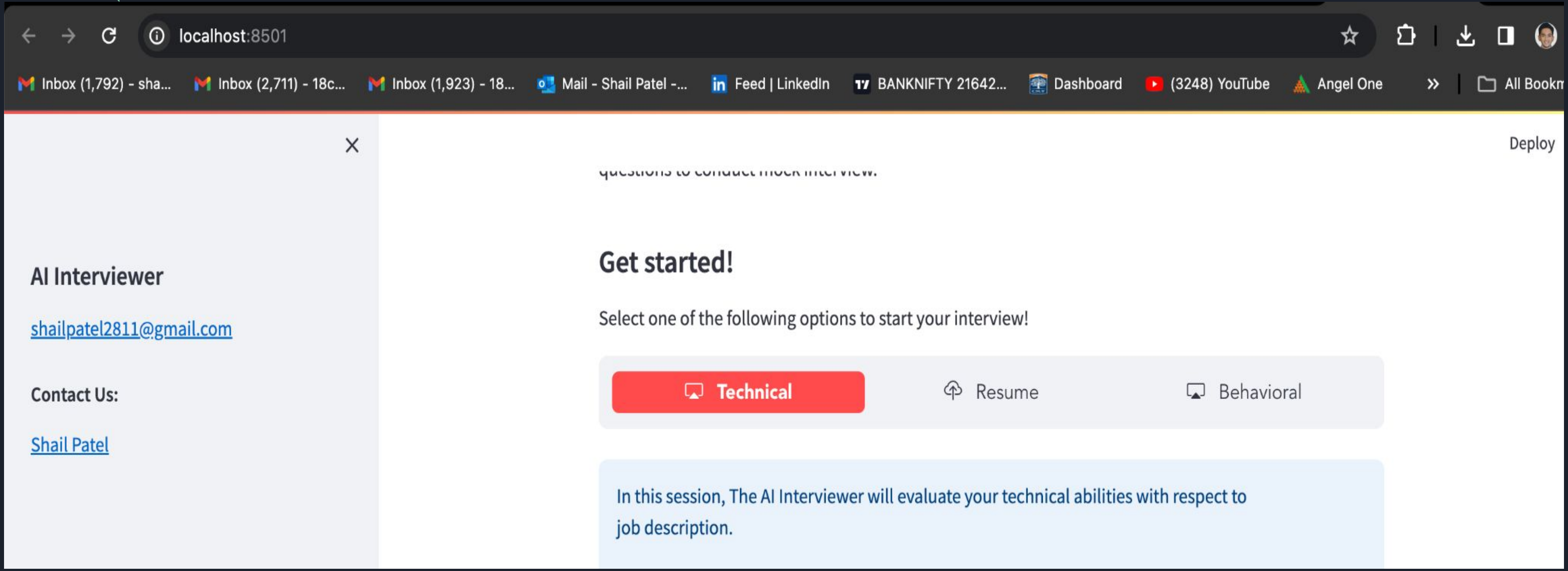
Backend Developer

Cloud Engineer

Made with Streamlit

Manage app

Local host link



AI Interviewer

shailpatel2811@gmail.com

Contact Us:

Shail Patel

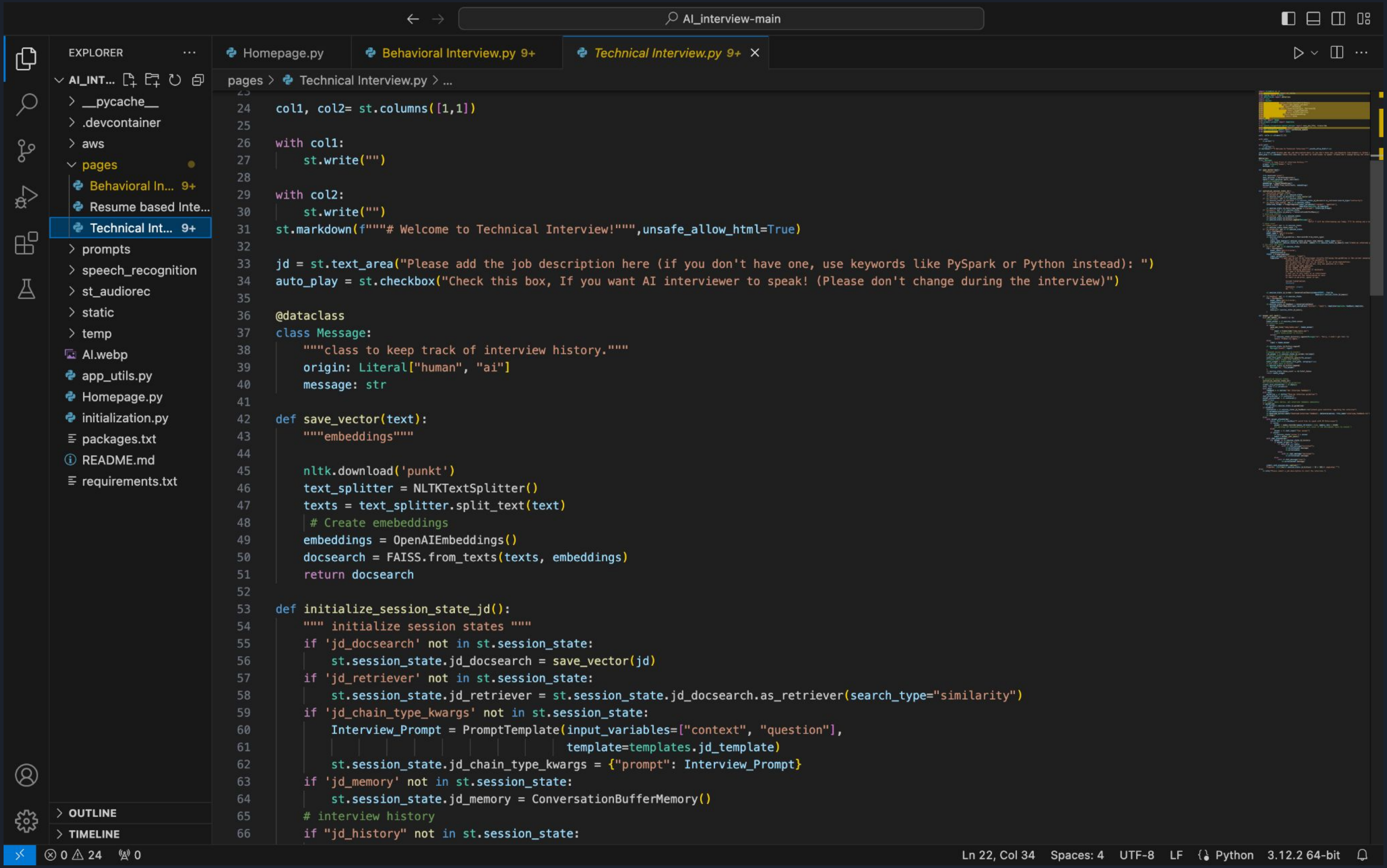
Get started!

Select one of the following options to start your interview!

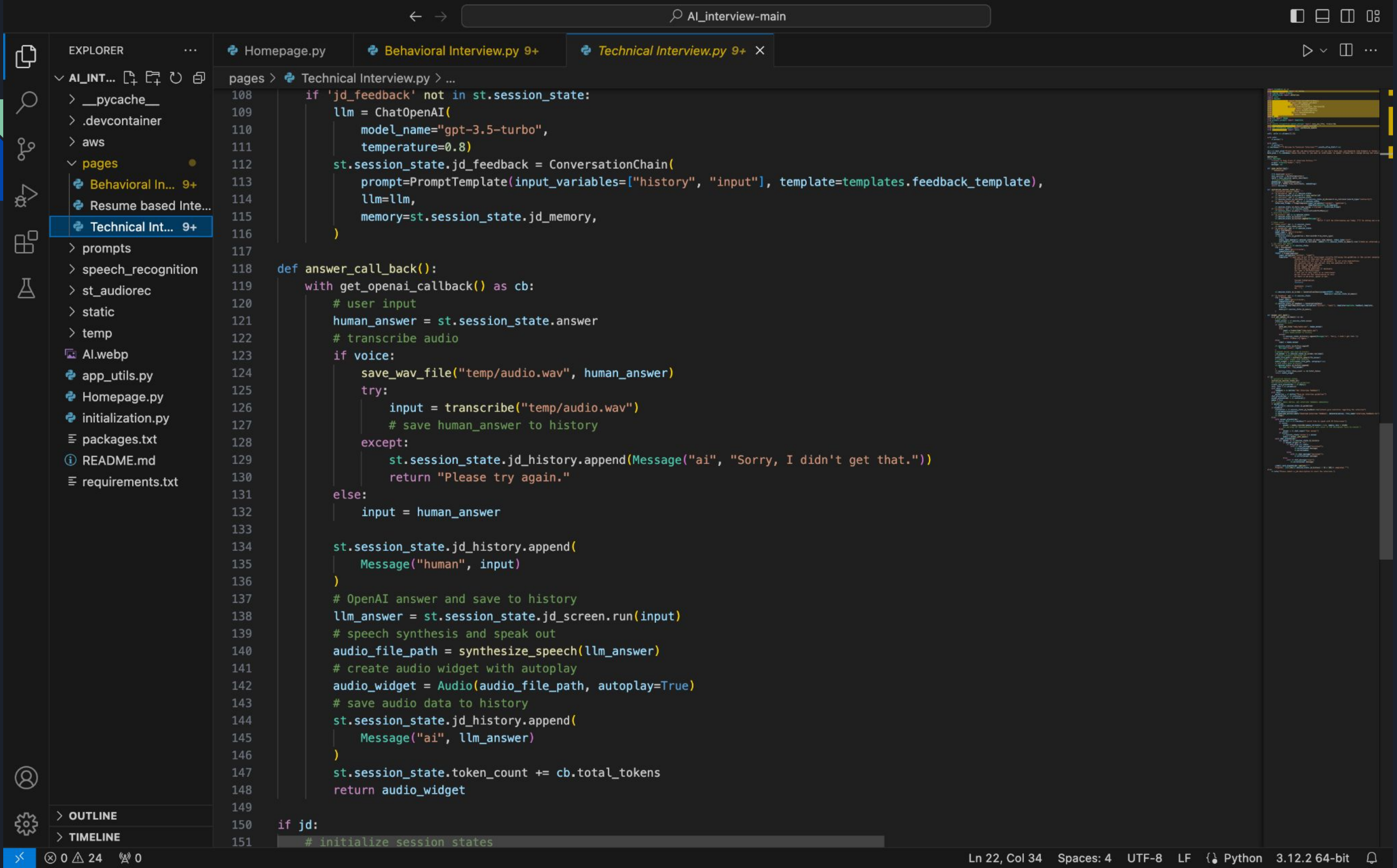
Technical | Resume | Behavioral

In this session, The AI Interviewer will evaluate your technical abilities with respect to job description.

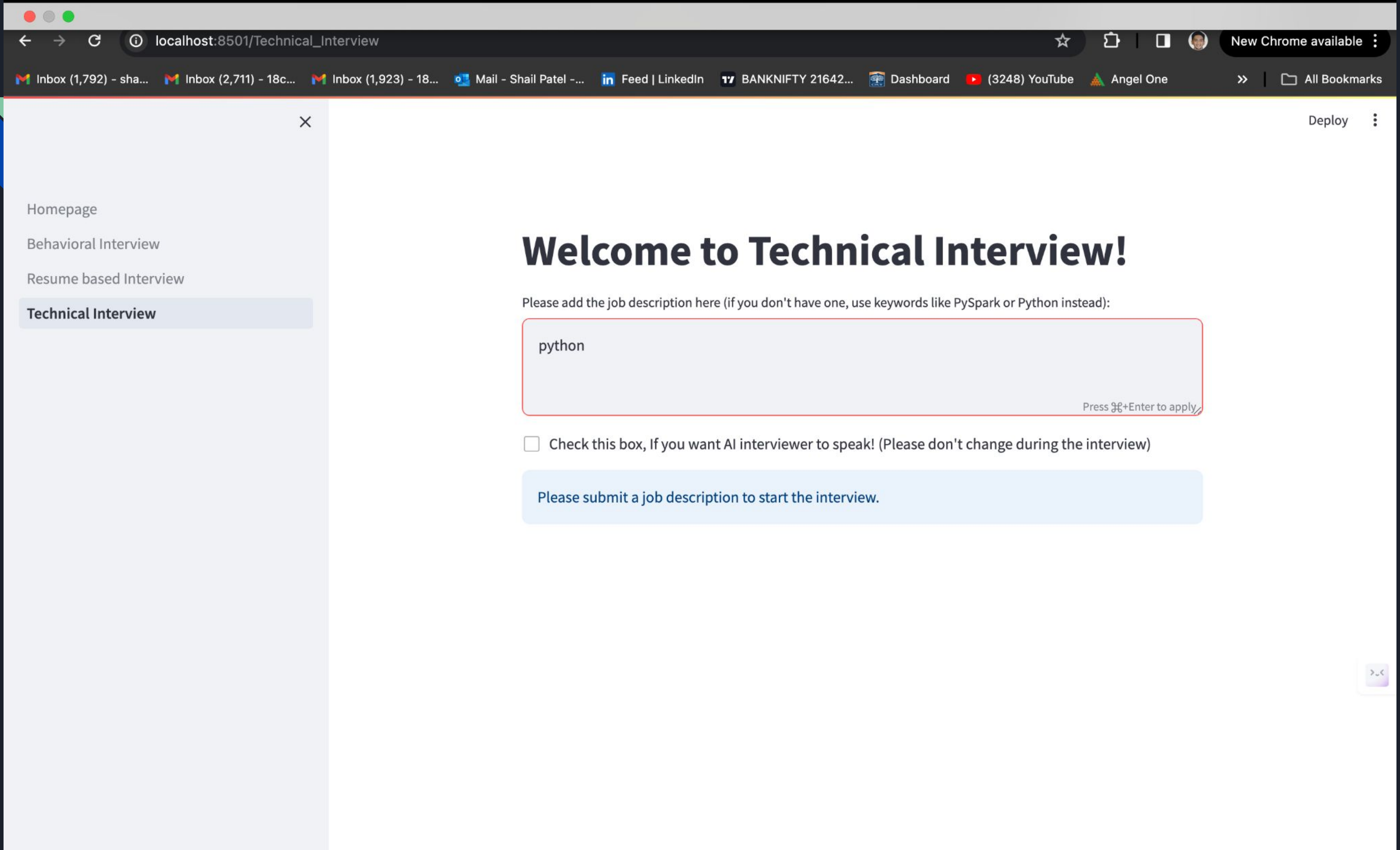
Technical Interview Page



```python
col1, col2= st.columns([1,1])

with col1:
    st.write("")

with col2:
    st.write("")
st.markdown(f"""# Welcome to Technical Interview!""",unsafe_allow_html=True)

jd = st.text_area("Please add the job description here (if you don't have one, use keywords like PySpark or Python instead): ")
auto_play = st.checkbox("Check this box, If you want AI interviewer to speak! (Please don't change during the interview)")

@dataclass
class Message:
    """class to keep track of interview history."""
    origin: Literal["human", "ai"]
    message: str

def save_vector(text):
    """embeddings"""

    nltk.download('punkt')
    text_splitter = NLTKTextSplitter()
    texts = text_splitter.split_text(text)
    # Create emebeddings
    embeddings = OpenAIEmbeddings()
    docsearch = FAISS.from_texts(texts, embeddings)
    return docsearch

def initialize_session_state_jd():
    """ initialize session states """
    if 'jd_docsearch' not in st.session_state:
        st.session_state.jd_docsearch = save_vector(jd)
    if 'jd_retriever' not in st.session_state:
        st.session_state.jd_retriever = st.session_state.jd_docsearch.as_retriever(search_type="similarity")
    if 'jd_chain_type_kwargs' not in st.session_state:
        Interview_Prompt = PromptTemplate(input_variables=["context", "question"],
                                          template=templates.jd_template)
        st.session_state.jd_chain_type_kwargs = {"prompt": Interview_Prompt}
    if 'jd_memory' not in st.session_state:
        st.session_state.jd_memory = ConversationBufferMemory()
    # interview history
    if "jd_history" not in st.session_state:
```



```python
        if 'jd_feedback' not in st.session_state:
            llm = ChatOpenAI(
                model_name="gpt-3.5-turbo",
                temperature=0.8)
            st.session_state.jd_feedback = ConversationChain(
                prompt=PromptTemplate(input_variables=["history", "input"], template=templates.feedback_template),
                llm=llm,
                memory=st.session_state.jd_memory,
            )

    def answer_call_back():
        with get_openai_callback() as cb:
            # user input
            human_answer = st.session_state.answer
            # transcribe audio
            if voice:
                save_wav_file("temp/audio.wav", human_answer)
                try:
                    input = transcribe("temp/audio.wav")
                    # save human_answer to history
                except:
                    st.session_state.jd_history.append(Message("ai", "Sorry, I didn't get that."))
                    return "Please try again."
            else:
                input = human_answer

            st.session_state.jd_history.append(
                Message("human", input)
            )
            # OpenAI answer and save to history
            llm_answer = st.session_state.jd_screen.run(input)
            # speech synthesis and speak out
            audio_file_path = synthesize_speech(llm_answer)
            # create audio widget with autoplay
            audio_widget = Audio(audio_file_path, autoplay=True)
            # save audio data to history
            st.session_state.jd_history.append(
                Message("ai", llm_answer)
            )
            st.session_state.token_count += cb.total_tokens
            return audio_widget

    if jd:
        # initialize session states
```



Welcome to Technical Interview!

Please add the job description here (if you don't have one, use keywords like PySpark or Python instead):

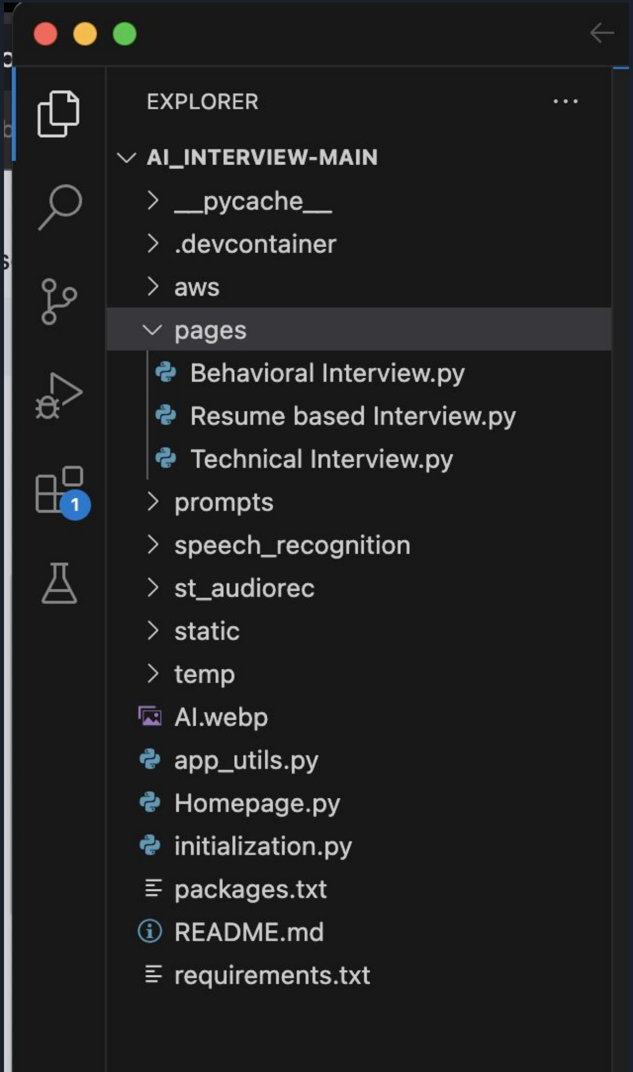
python

Press ⌘+Enter to apply

☐ Check this box, If you want AI interviewer to speak! (Please don't change during the interview)

Please submit a job description to start the interview.

Pages



```
EXPLORER
AI_INTERVIEW-MAIN
  > __pycache__
  > .devcontainer
  > aws
  v pages
      Behavioral Interview.py
      Resume based Interview.py
      Technical Interview.py
  > prompts
  > speech_recognition
  > st_audiorec
  > static
  > temp
  AI.webp
  app_utils.py
  Homepage.py
  initialization.py
  packages.txt
  README.md
  requirements.txt
```

Thank You