The error you're encountering happens because the `outputs` argument of the `click` method expects Gradio components, not strings. The component IDs are used internally by Gradio to manage the data flow between user interactions and processing functions.

To fix the issue, you need to create the actual Gradio components that will be used to display the results of the semantic search and the thank you message.

Here's the corrected code with appropriate Gradio components:

```python

import gradio as gr

from sentence\_transformers import SentenceTransformer, util

from transformers import pipeline

import nltk

import time

# Load models

semantic\_model = SentenceTransformer('paraphrase-multilingual-mpnet-base-v2')

nli\_model = pipeline("text-classification", model="path\_to\_deberta\_v3\_model") # Update with your model path

# Initialize Bhashini API (mock for demonstration)

def bhashini\_api\_mock(text, source\_lang, target\_lang):

# Mock function to represent translation, ASR, transliteration

return text # In actual implementation, connect to Bhashini API

# Welcome messages and audio files (mock paths)

welcome\_messages = {

"en": ("Welcome!", "path\_to\_welcome\_audio\_en"),

"hi": ("स्वागत है!", "path\_to\_welcome\_audio\_hi"),

"mr": ("स्वागत आहे!", "path\_to\_welcome\_audio\_mr"),

"gu": ("સ્વાગત છે!", "path\_to\_welcome\_audio\_gu"),

"pa": ("ਸੁਆਗਤ ਹੈ!", "path\_to\_welcome\_audio\_pa"),

"bn": ("স্বাগতম!", "path\_to\_welcome\_audio\_bn"),

}

# Function to handle language selection

def select\_language(language):

message, audio\_path = welcome\_messages[language]

return message, audio\_path

# Function to process the document

def process\_document(language, topic, problem, document, document\_lang):

# OCR and translation (mock for demonstration)

ocr\_text = bhashini\_api\_mock(document, document\_lang, language) # Perform OCR and translate to user's language

# Semantic search (mock for demonstration)

def semantic\_search(query, documents, threshold=0.4):

query\_embedding = get\_embeddings([query])

doc\_embeddings = get\_embeddings(documents)

similarities = util.pytorch\_cos\_sim(query\_embedding, doc\_embeddings)[0]

relevant\_indices = [i for i in range(len(similarities)) if similarities[i] >= threshold]

return [(documents[i], similarities[i].item()) for i in relevant\_indices]

# NLI classification (mock for demonstration)

matched\_paragraphs = semantic\_search(topic, [ocr\_text])

results = []

for para, similarity in matched\_paragraphs:

sentences = nltk.sent\_tokenize(para)

for sentence in sentences:

result = nli\_model({"premise": sentence, "hypothesis": problem})

results.append((sentence, result))

# Final output (mock for demonstration)

thank\_you\_message = bhashini\_api\_mock("Thank you!", "en", language)

return results, thank\_you\_message

# Gradio interface

with gr.Blocks() as demo:

gr.Markdown("## Welcome! Please wait while we load...")

with gr.Tabs():

with gr.TabItem("Step 1: Welcome"):

language = gr.Radio(["English", "Hindi", "Marathi", "Gujarati", "Punjabi", "Bengali"], label="Select your language")

welcome\_output = gr.Textbox()

welcome\_audio = gr.Audio()

gr.Button("Submit").click(select\_language, inputs=language, outputs=[welcome\_output, welcome\_audio])

with gr.TabItem("Step 2: Input Language and Details"):

user\_language = gr.Radio(["en", "hi", "mr", "gu", "pa", "bn"], label="Preferred Language")

user\_topic = gr.Textbox(label="Topic of Concern")

user\_problem = gr.Textbox(label="Detailed Problem Description")

with gr.TabItem("Step 3: Upload Document"):

document = gr.File(label="Upload your legal document")

document\_lang = gr.Dropdown(["English", "Hindi", "Marathi", "Gujarati", "Punjabi", "Bengali"], label="Document Language")

results\_output = gr.Dataframe(headers=["Sentence", "NLI Result"]) # To display results of semantic search

thank\_you\_message = gr.Textbox(label="Thank You Message")

gr.Button("Submit").click(process\_document, inputs=[user\_language, user\_topic, user\_problem, document, document\_lang], outputs=[results\_output, thank\_you\_message])

demo.launch()

```

### Changes Made:

1. \*\*Outputs Correction\*\*:

- Defined Gradio components (`results\_output` and `thank\_you\_message`) to display the results and thank you message.

- Used these components in the `outputs` argument of the `click` method.

2. \*\*Results Display\*\*:

- Changed the `results\_output` to a `gr.Dataframe` for displaying the sentences and their NLI results.

3. \*\*Component Names\*\*:

- Updated component names to ensure they match the inputs and outputs expected by the `process\_document` function.

This should resolve the error and provide a structure to build on for further development.