DexterSptizu Create app.py dc83d6b VERIFIED 4 minutes ago

raw | Copy download link | history | blame | edit | delete | Safe | 5.78 kB

```python
import gradio as gr
from langchain_core.prompts import ChatPromptTemplate, MessagesPlaceholder
from langchain_core.messages import SystemMessage, HumanMessage, AIMessage
from langchain_openai import ChatOpenAI
import os

class AirlineChatbot:
    def __init__(self):
        self.chat_history = []
        self.api_key = None
        self.chat = None

    def set_api_key(self, api_key):
        self.api_key = api_key
        os.environ["OPENAI_API_KEY"] = api_key
        self.chat = ChatOpenAI(
            temperature=0.7,
            model="gpt-4o-mini"
        )
        return "API Key set successfully! You can now start chatting."

    def format_chat_history(self, chat_history):
        formatted_messages = []
        for message in chat_history:
            if isinstance(message, HumanMessage):
                formatted_messages.append(("human", message.content))
            elif isinstance(message, AIMessage):
                formatted_messages.append(("assistant", message.content))
        return formatted_messages

    def respond(self, message, history):
        if not self.api_key:
            return "Please set your OpenAI API key first."

        # System prompt template
        system_prompt = """You are an airline customer service representative.
        Your role is to assist customers with:
        - Flight bookings and inquiries
        - Schedule information
        - Baggage policies
        - Check-in procedures
        - General airline policies

        Be professional, courteous, and provide accurate information.
        Always maintain the role of an airline representative throughout the conversation."""

        # Convert Gradio history to LangChain format
        self.chat_history = []
        for human, ai in history:
            self.chat_history.append(HumanMessage(content=human))
            self.chat_history.append(AIMessage(content=ai))

```

```python
        # Create appropriate template based on message content
        if "book" in message.lower():
            template = ChatPromptTemplate.from_messages([
                ("system", system_prompt),
                ("human", "I want to book a flight from {origin} to {destination} on {date}"),
                ("assistant", "I'll help you book a flight. Let me check available options."),
                MessagesPlaceholder(variable_name="chat_history"),
                ("human", "{input}")
            ])
        elif "baggage" in message.lower():
            template = ChatPromptTemplate.from_messages([
                ("system", system_prompt),
                ("human", "What's the baggage allowance for {flight_type} flights?"),
                MessagesPlaceholder(variable_name="chat_history"),
                ("human", "{input}")
            ])
        elif "schedule" in message.lower():
            template = ChatPromptTemplate.from_messages([
                ("system", system_prompt),
                ("human", "I need flight schedules between {origin} and {destination}"),
                MessagesPlaceholder(variable_name="chat_history"),
                ("human", "{input}")
            ])
        else:
            template = ChatPromptTemplate.from_messages([
                ("system", system_prompt),
                MessagesPlaceholder(variable_name="chat_history"),
                ("human", "{input}")
            ])

        # Format the prompt with the current message
        prompt = template.format_messages(
            chat_history=self.chat_history,
            input=message,
            origin="",
            destination="",
            date="",
            flight_type=""
        )

        try:
            # Get response from the model
            response = self.chat(prompt)
            return response.content
        except Exception as e:
            return f"Error: {str(e)}"

def create_demo():
    chatbot = AirlineChatbot()

    with gr.Blocks(theme=gr.themes.Soft()) as demo:
        with gr.Row():
            gr.Markdown("## ✈️ Airline Customer Service Assistant")

        with gr.Row():
            with gr.Column():
                api_key_input = gr.Textbox(
                    label="Enter your OpenAI API Key",
                    type="password",
                    placeholder="sk-..."
                )
                set_key_button = gr.Button("Set API Key")
                api_key_status = gr.Textbox(label="Status", interactive=False)

        chatbot_interface = gr.ChatInterface(
            fn=chatbot.respond,
            description="""Welcome to our Airline Customer Service!
```

```
            Ask questions about flights, bookings, baggage, and more.""",
            examples=[
                "What's the baggage allowance for international flights?",
                "How early should I arrive for check-in?",
                "I want to book a flight from New York to London",
                "What documents do I need for international travel?",
                "Can I change my flight date?",
                "Do you provide meals on long-haul flights?"
            ]
        )

        set_key_button.click(
            fn=chatbot.set_api_key,
            inputs=[api_key_input],
            outputs=[api_key_status]
        )

    return demo

# Custom CSS for better styling
custom_css = """
.gradio-container {
    font-family: 'Arial', sans-serif;
}
.chat-message {
    padding: 15px;
    margin: 5px;
    border-radius: 10px;
}
"""

if __name__ == "__main__":
    # Create and launch the demo
    demo = create_demo()
    demo.launch(
        share=True,  # Set to False if you don't want to generate a public URL
        server_name="0.0.0.0",
        server_port=7860,
        debug=True
    )

```