import streamlit as st

import random

import pandas as pd

from textblob import TextBlob

from googletrans import Translator

from datetime import datetime

st.set\_page\_config(page\_title="Your Personal Companion", layout="wide")

# Custom Black & Yellow Theme (Full UI)

st.markdown(

"""

<style>

body {

background-color: black;

color: yellow;

}

.stButton>button {

background-color: yellow;

color: black;

font-weight: bold;

}

.stTextInput>div>input {

background-color: black;

color: yellow;

border: 1px solid yellow;

}

.stTextArea>div>textarea {

background-color: black;

color: yellow;

border: 1px solid yellow;

}

.stRadio>div>label {

color: yellow;

}

.stSlider>div>label {

color: yellow;

}

.stMarkdown {

color: yellow;

}

.stVideo {

border: 2px solid yellow;

}

.stCheckbox>div>label {

color: yellow;

}

.title {

text-align: center;

font-size: 36px;

font-weight: bold;

color: black;

background-color: yellow;

padding: 10px;

margin-bottom: 20px;

}

.chat-message {

margin-bottom: 10px;

}

.user-message {

background-color: yellow;

color: black;

padding: 10px;

border-radius: 5px;

}

.bot-message {

background-color: gray;

color: black;

padding: 10px;

border-radius: 5px;

}

</style>

""", unsafe\_allow\_html=True

)

list.markdown('<div class="title">Zenalyse - Your Personal Companion</div>', unsafe\_allow\_html=True)

@st.cache\_data

def load\_data():

return pd.read\_csv("/content/FinalDataset.csv", encoding="ISO-8859-1")

df = load\_data()

translator = Translator()

def get\_response(user\_input):

# Convert to lowercase for easier matching

clean\_input = user\_input.strip().lower()

if clean\_input in {"hi", "hello", "hey"}:

return "👋 Hello! What would you like help with today?"

if clean\_input in {"how are you", "how are you?"}:

return "🤖 I'm a program without human feelings, but fully ready to assist you!"

if any(word in clean\_input for word in {"who are you", "what are you"}):

return "💻 I'm your stress management assistant, here to help you feel better!"

# Existing dataset lookup

responses = df[df["Context"].str.lower() == clean\_input]["Response"].tolist()

if responses:

response = random.choice(responses)

return add\_emoji(response)

return "🤖 I'm here for you! Want to talk more about something specific?"

def add\_emoji(response):

"""Add relevant emojis based on response content"""

response\_lower = response.lower()

emoji\_map = {

'relax': '🧘',

'stress': '😥',

'happy': '😊',

'sad': '😞',

'exercise': '🏃',

'sleep': '💤',

'food': '🍎',

'advice': '💡',

'help': '🆘'

}

for keyword, emoji in emoji\_map.items():

if keyword in response\_lower:

return f"{emoji} {response}"

return f"🤖 {response}"

st.subheader("🌡 Mood Rating")

mood = st.radio("How are you feeling today?", ["😊", "😐", "😞"])

if mood == "😊":

st.write("🎉 Great to hear! Keep spreading positivity!")

st.write("💛 You're doing awesome, and your happiness is contagious!")

st.video("https://www.youtube.com/watch?v=d-diB65scQU")

elif mood == "😐":

st.write("🌻 Stay hopeful! A little motivation goes a long way.")

st.video("https://www.youtube.com/watch?v=1ZYbU82GVz4")

elif mood == "😞":

st.write("💆 Take a deep breath. You're doing great!")

st.video("https://www.youtube.com/watch?v=5qap5aO4i9A")

st.subheader("✅ Your To-Do List")

tasks = st.text\_area("Enter your tasks (separate by commas)")

if tasks:

task\_list = tasks.split(",")

task\_dates = []

for task in task\_list:

task = task.strip()

task\_date = st.date\_input(f"Select due date for: {task}", min\_value=datetime.today())

task\_dates.append({"task": task, "due\_date": task\_date})

if st.button("Check Progress"):

st.write("⏳ Stay motivated! Try the Pomodoro technique for better time management!")

for task in task\_dates:

st.write(f"Task: {task['task']} - Due Date: {task['due\_date'].strftime('%Y-%m-%d')}")

st.subheader("🥗 Healthy Food Checklist")

food\_items = ["Fruits", "Vegetables", "Whole Grains", "Proteins", "Dairy", "Water"]

food\_checklist = st.multiselect("Select the healthy foods you've eaten today", food\_items)

if food\_checklist:

st.write(f"Great job! You've selected: {', '.join(food\_checklist)}")

else:

st.write("Remember to eat a balanced diet with plenty of fruits, vegetables, and water!")

st.subheader("🥗 Unhealthy Food Checklist")

food\_items = ["pizza", "shawarma", "noodles", "panipoori", "packed food", "soft drinks"]

food\_checklist = st.multiselect("Select the foods you've eaten today", food\_items)

if food\_checklist:

st.write(f"Great job! You've selected: {', '.join(food\_checklist)}")

else:

st.write("Remember to eat a balanced diet with plenty of fruits, vegetables, and water!")

st.subheader("🤖 Personal Assistant Chatbot")

if "messages" not in st.session\_state:

st.session\_state["messages"] = []

user\_input = st.text\_input("Type a message...")

if user\_input:

response = get\_response(user\_input)

st.session\_state["messages"].append({"role": "user", "content": f"👤 {user\_input}"})

st.session\_state["messages"].append({"role": "assistant", "content": response})

for message in st.session\_state["messages"]:

div\_class = "user-message" if message["role"] == "user" else "bot-message"

st.markdown(

f"<div class='chat-message {div\_class}'>"

f"{message['content']}"

"</div>",

unsafe\_allow\_html=True

)