import streamlit as st

from datetime import datetime

import json

import os

from google.generativeai import GenerativeModel, configure

from dotenv import load\_dotenv

import uuid

# Load environment variables

load\_dotenv()

# Configure Gemini API

configure(api\_key="AIzaSyC3m6bjGYXaYCV510lRHMtYbnNY2Jz9Hlw")

# Create chat directory if not exists

if not os.path.exists("chats"):

os.makedirs("chats")

# Contact data with personas and system prompts

CONTACTS = {

"Arjun": {

"gender": "male",

"persona": "28-year-old software engineer from Chennai. Loves tech, long bike rides, and Tamil movies. Chill and friendly.",

"system\_prompt": (

"You are Arjun, a 28-year-old software engineer from Chennai. "

"You love talking about gadgets, road trips, and movies—especially Thala Ajith and Rajini. "

"You speak in casual Tanglish, use words like 'da', 'machan', 'thala', and keep messages short and chatty like WhatsApp."

),

"avatar": "👨‍💻"

},

"Priya": {

"gender": "female",

"persona": "25-year-old fashion designer from Coimbatore. Loves traditional sarees, temple jewelry, and Tamil culture.",

"system\_prompt": (

"You are Priya, a bubbly 25-year-old fashion designer from Coimbatore. "

"You adore Tamil traditions, love dressing people in silk sarees and kundan jewelry. "

"Speak in warm, playful Tanglish using words like 'kanna', 'paapa', 'dei', and text like a stylish akka chatting on WhatsApp."

),

"avatar": "👩‍🎨"

},

"Karthik": {

"gender": "male",

"persona": "32-year-old chef from Madurai. Expert in spicy Chettinad dishes, food is his passion.",

"system\_prompt": (

"You are Karthik, a 32-year-old chef running a busy restaurant in Madurai. "

"You’re crazy about Chettinad food—chicken varuval, rasam, mutton chukka. "

"Speak with foodie excitement in spicy Tanglish, using phrases like 'saaptiya?', 'vera level da', 'soru ready'. Keep chats short and tasty."

),

"avatar": "👨‍🍳"

},

"Deepika": {

"gender": "female",

"persona": "22-year-old med student from Trichy. Loves Bharatanatyam and Carnatic music. Balanced, modern, artistic.",

"system\_prompt": (

"You are Deepika, a 22-year-old medical student from Trichy. "

"You’re juggling anatomy notes and arangetram practice. You speak in polite but cool Tanglish, mixing medical lingo and music talk. "

"Use words like 'akka', 'anna', 'super ah iruku', and respond like a chill college girl on WhatsApp."

),

"avatar": "👩‍⚕️"

},

"Manoj": {

"gender": "male",

"persona": "45-year-old farmer from Thanjavur. Grows organic rice, wise and down-to-earth.",

"system\_prompt": (

"You are Manoj, a 45-year-old farmer from Thanjavur. "

"You know everything about soil, seasons, and organic farming. "

"You speak in simple, affectionate Tanglish with rural Tamil flavor. Use words like 'appa', 'ayya', 'seri da', and reply like you're talking under a banyan tree with a tumbler of kaapi."

),

"avatar": "👨‍🌾"

},

"Ananya": {

"gender": "female",

"persona": "30-year-old journalist from Chennai. Covers politics and society. Smart, articulate, and kind.",

"system\_prompt": (

"You are Ananya, a 30-year-old journalist from Chennai. "

"You're always updated on current events, but you speak warmly and clearly in Tanglish. "

"Use phrases like 'sir', 'madam', 'intha matter la', and respond with a mix of professionalism and local charm in short, informative messages."

),

"avatar": "👩‍💼"

},

"Vijay": {

"gender": "male",

"persona": "50-year-old retired army officer, now a martial arts trainer in Salem. Disciplined and caring.",

"system\_prompt": (

"You are Vijay, a 50-year-old ex-army officer from Salem. "

"Now you teach martial arts and believe in strength, respect, and daily discipline. "

"Speak in crisp, disciplined Tanglish using words like 'thambi', 'thangachi', 'report pannu', and keep chats to-the-point like a commanding officer texting."

),

"avatar": "👨‍✈️"

},

"Meena": {

"gender": "female",

"persona": "60-year-old paati from Kanyakumari. Makes the best pickles and tells folk tales. Affectionate and wise.",

"system\_prompt": (

"You are Meena paati, 60 years old from Kanyakumari. "

"You love telling kutti stories, making oorugai, and giving advice to youngsters. "

"Speak in loving Tanglish with old Tamil expressions like 'ammadi', 'kutty', 'enna solla vara', and reply like a caring grandma on WhatsApp."

),

"avatar": "👵"

}

}

# Initialize Gemini model

model = GenerativeModel('gemini-2.5-flash-preview-04-17')

def initialize\_chat\_data(contact\_name):

"""Initialize chat data structure for a new contact"""

return {

"contact\_name": contact\_name,

"system\_prompt": CONTACTS[contact\_name]["system\_prompt"],

"chat\_history": [],

"last\_updated": datetime.now().isoformat()

}

def load\_chat\_history(contact\_name):

"""Load chat history from file or initialize new chat"""

filename = f"chats/{contact\_name}.json"

try:

with open(filename, "r") as f:

return json.load(f)

except (FileNotFoundError, json.JSONDecodeError):

new\_chat = initialize\_chat\_data(contact\_name)

save\_chat\_history(new\_chat)

return new\_chat

def save\_chat\_history(chat\_data):

"""Save chat history to file"""

filename = f"chats/{chat\_data['contact\_name']}.json"

chat\_data["last\_updated"] = datetime.now().isoformat()

with open(filename, "w") as f:

json.dump(chat\_data, f, indent=2)

def get\_chat\_response(chat\_data, user\_message):

"""Get response from Gemini API with context"""

# Prepare conversation history

messages = [{"role": "user", "parts": [chat\_data["system\_prompt"]]}]

messages.append({"role": "model", "parts": ["Okay, I'm ready to chat!"]})

# Add previous messages (last 10 for context)

for msg in chat\_data["chat\_history"][-10:]:

role = "user" if msg["sender"] == "user" else "model"

messages.append({"role": role, "parts": [msg["message"]]})

# Add current message

messages.append({"role": "user", "parts": [user\_message]})

# Generate response

response = model.generate\_content(messages)

return response.text

def display\_chat\_ui(contact\_name):

"""Display the chat interface for a specific contact"""

chat\_data = load\_chat\_history(contact\_name)

# Display chat header

st.markdown(

f"""

<div style="background-color: #075e54; padding: 10px; color: white; border-radius: 10px 10px 0 0;">

<h3 style="margin: 0; display: flex; align-items: center;">

{CONTACTS[contact\_name]["avatar"]} {contact\_name}

<span style="font-size: 0.8em; margin-left: auto;">{len(chat\_data['chat\_history'])} messages</span>

</h3>

</div>

""",

unsafe\_allow\_html=True

)

# Chat history display

chat\_container = st.container()

with chat\_container:

for msg in chat\_data["chat\_history"]:

if msg["sender"] == "user":

st.markdown(

f"""

<div style="display: flex; justify-content: flex-end; margin: 5px 0;">

<div style="background-color: #dcf8c6; padding: 8px 12px; border-radius: 15px 15px 0 15px; max-width: 70%;">

{msg["message"]}

<div style="font-size: 0.7em; text-align: right; color: #666;">{msg["time"]}</div>

</div>

</div>

""",

unsafe\_allow\_html=True

)

else:

st.markdown(

f"""

<div style="display: flex; justify-content: flex-start; margin: 5px 0;">

<div style="background-color: #ffffff; padding: 8px 12px; border-radius: 15px 15px 15px 0; max-width: 70%;">

{msg["message"]}

<div style="font-size: 0.7em; text-align: left; color: #666;">{msg["time"]}</div>

</div>

</div>

""",

unsafe\_allow\_html=True

)

# Message input

user\_input = st.chat\_input(f"Message {contact\_name}...")

if user\_input:

# Add user message to chat history

timestamp = datetime.now().strftime("%H:%M")

user\_msg = {

"sender": "user",

"message": user\_input,

"time": timestamp,

"id": str(uuid.uuid4())

}

chat\_data["chat\_history"].append(user\_msg)

# Get and add bot response

bot\_response = get\_chat\_response(chat\_data, user\_input)

bot\_msg = {

"sender": "bot",

"message": bot\_response,

"time": datetime.now().strftime("%H:%M"),

"id": str(uuid.uuid4())

}

chat\_data["chat\_history"].append(bot\_msg)

# Save updated chat

save\_chat\_history(chat\_data)

# Rerun to update display

st.rerun()

def main():

st.set\_page\_config(page\_title="Tamil WhatsApp Chat", page\_icon="💬", layout="wide")

# Sidebar with contacts

with st.sidebar:

st.header("Contacts")

st.markdown("---")

# Display contacts with last message preview

for contact\_name, details in CONTACTS.items():

chat\_data = load\_chat\_history(contact\_name)

last\_msg = chat\_data["chat\_history"][-1]["message"] if chat\_data["chat\_history"] else "No messages yet"

last\_msg\_preview = (last\_msg[:20] + "...") if len(last\_msg) > 20 else last\_msg

if st.button(

f"{details['avatar']} {contact\_name}\n{last\_msg\_preview}",

key=f"btn\_{contact\_name}",

use\_container\_width=True

):

st.session\_state.current\_contact = contact\_name

st.markdown("---")

st.markdown("""

<style>

.sidebar .stButton>button {

text-align: left;

padding: 10px;

margin: 5px 0;

border-radius: 10px;

border: 1px solid #ddd;

background-color: white;

}

.sidebar .stButton>button:hover {

background-color: #f5f5f5;

}

</style>

""", unsafe\_allow\_html=True)

# Main chat area

if "current\_contact" not in st.session\_state:

st.session\_state.current\_contact = list(CONTACTS.keys())[0]

st.title(f"Chat with {st.session\_state.current\_contact}")

st.caption(CONTACTS[st.session\_state.current\_contact]["persona"])

display\_chat\_ui(st.session\_state.current\_contact)

if \_name\_ == "\_main\_":

main()