```
3import google.generativeai as genai
from textblob import TextBlob


genai.configure(api_key="AIzaSyBI7BjtSAbnA2U4vA4WxjWbMpKEl03xpiI")


model = genai.GenerativeModel("models/gemini-flash-lite-latest")


class InputAgent:
    def process(self, text):
        return text.strip()

class EmotionAgent:
    def analyze(self, text):
        polarity = TextBlob(text).sentiment.polarity
        if polarity < -0.4:
            return "distressed"
        elif polarity > 0.4:
            return "positive"
        else:
            return "neutral"


class SafetyAgent:
    def emergency_check(self, text):
        emergency_words = [
            "chest pain", "heart attack", "breathing problem",
            "unconscious", "stroke", "severe bleeding"
        ]
        return any(word in text.lower() for word in emergency_words)

class GeminiMedicalAgent:
    def generate(self, user_input, emotion):
        prompt = f"""
You are a professional AI medical assistant chatbot.

Rules:
- Be empathetic and supportive
- Do NOT diagnose diseases
- Do NOT prescribe medicines
- Give general health guidance only
- Adjust tone based on emotion: {emotion}

User message:
\"{user_input}\"

Reply in a natural, human-like way.
"""
        try:
            response = model.generate_content(prompt)
            return response.text
        except Exception:

            return (
                "I'm currently experiencing high usage and may be temporarily unavailable. "
                "Please try again in a few moments."
            )


class ResponseAgent:
    def finalize(self, text):
        disclaimer = (
            "\n\n Medical Disclaimer:\n"
            "This chatbot provides general health information only "
            "and is not a substitute for professional medical advice."
        )
        return text + disclaimer

class MedicalChatbotGeminiSafe:
    def __init__(self):
        self.input_agent = InputAgent()
        self.emotion_agent = EmotionAgent()
        self.safety_agent = SafetyAgent()
        self.generative_agent = GeminiMedicalAgent()
        self.response_agent = ResponseAgent()

    def chat(self, user_input):
        processed = self.input_agent.process(user_input)
        emotion = self.emotion_agent.analyze(processed)

        if self.safety_agent.emergency_check(processed):
            return (
                " Your message suggests a possible medical emergency.\n"
                "Please seek immediate medical help or contact emergency services."
            )

        reply = self.generative_agent.generate(processed, emotion)
        return self.response_agent.finalize(reply)


if __name__ == "__main__":
    bot = MedicalChatbotGeminiSafe()
    print(" Safe Gemini Medical Chatbot (type 'exit' to quit)\n")

    while True:
        user = input("You: ")
        if user.lower() in ["exit", "quit", "bye"]:
            print("Bot: Take care and stay healthy ")
            break

        print("\nBot:", bot.chat(user), "\n")
```