

chat.py

```
1 import requests
2 import json
3 import sqlite3
4 from datetime import datetime
5
6
7
8 with open("gemini_key.txt", "r") as f:
9     API_KEY = f.read().strip()
10
11
12 MODEL = "models/gemini-2.0-flash"
13
14 API_URL = f"https://generativelanguage.googleapis.com/v1/{MODEL}:generateContent?key={API_KEY}"
15
16
17 ASSISTANT_STYLE = (
18     "You are a friendly, professional AI assistant. "
19     "You explain things clearly in simple language, like a good teacher. "
20     "You remember previous conversation context and keep answers concise but helpful. "
21 )
22
23
24 conversation_history = [] # list of (user, bot) messages
25
26
27
28
29 conn = sqlite3.connect("chat_history.db", check_same_thread=False)
30 cur = conn.cursor()
31 cur.execute("""
32     CREATE TABLE IF NOT EXISTS conversations (
33         id INTEGER PRIMARY KEY AUTOINCREMENT,
34         timestamp TEXT,
35         user_message TEXT,
36         bot_reply TEXT
37     )
38 """)
39 conn.commit()
40
41
42 def save_to_db(user_msg, bot_reply):
43     ts = datetime.now().isoformat(timespec="seconds")
44     cur.execute(
45         "INSERT INTO conversations (timestamp, user_message, bot_reply) VALUES (?, ?, ?)",
46         (ts, user_msg, bot_reply),
47     )
```

```
48     conn.commit()
49
50
51 # ----- CHATBOT FUNCTION -----
52
53 def build_prompt(user_message: str) -> str:
54     """
55     Build a single text prompt with:
56     - Personality
57     - Recent conversation history
58     - Current user message
59     """
60     history_text = ""
61     # last 5 exchanges only
62     for u, b in conversation_history[-5:]:
63         history_text += f"User: {u}\nAssistant: {b}\n"
64
65     prompt = (
66         f"{ASSISTANT_STYLE}\n\n"
67         f"Here is the recent conversation:\n{history_text}\n"
68         f"User: {user_message}\n"
69         f"Assistant:"
70     )
71     return prompt
72
73
74 def chatbot(message: str) -> str:
75     global conversation_history
76
77     if not message.strip():
78         return "Please type something so I can help you 😊"
79
80     full_prompt = build_prompt(message)
81
82     payload = {
83         "contents": [
84             {
85                 "parts": [
86                     {"text": full_prompt}
87                 ]
88             }
89         ]
90     }
91
92     try:
93         response = requests.post(API_URL, json=payload)
94         data = response.json()
95
96         if "error" in data and data["error"].get("code") == 503:
97             raise Exception("API is currently unavailable")
```

```
98     response = requests.post(API_URL, json=payload)
99     data = response.json()
100
101    if "candidates" in data:
102        reply = data["candidates"][0]["content"]["parts"][0]["text"]
103    else:
104        reply = "⚠ Error from AI: " + json.dumps(data)
105
106
107    conversation_history.append((message, reply))
108    save_to_db(message, reply)
109
110    return reply
111
112 except Exception as e:
113     return f"⚠ Error: {e}"
114
```