DAY – 44

30 September 2025

Reason for Error

1. **Gemini API** (jo tusi use kar rahe ho) jadon answer generate karda, oh **by default English vich hi likhda** hai → even if tusi system prompt ch "Punjabi vich reply karo" likho.
    - Gemini gemini-pro text generation API nu koi proper target_lang parameter nahi ditta.
    - Is karke oh sirf English text return karda.
2. Tuhada translate_text() function Gemini nu hi translation layi use kar reha hai → but Gemini har vaar **English output hi generate karda**.
    - Matlab: translation wala step vi fail ho janda.
    - Result → sirf English hi chatbot answer karde.

Solution

- Translation layi **HuggingFace Transformers** ya Google Translate API use karo.
- Gemini nu **sirf English base answer** generate karan layi rakho.
- Fir automatically user di language detect karke us vich **translate back** karo.

 Corrected chatbot_logic.py

```
import os
import re
import unicodedata
import requests
import difflib
import langdetect
from deep_translator import GoogleTranslator   # Add Google Translate
```

```
# ===== CONFIG =====
HF_API_KEY = "your_huggingface_key"
GEMINI_API_KEY = "your_gemini_key"
DATASET_PATH = "dataset.txt"

# ===== LOAD DATASET =====
faq_data = {}
with open(DATASET_PATH, "r", encoding="utf-8") as f:
    for line in f:
        if "=" in line:
            q, a = line.split("=", 1)
            faq_data[q.strip().lower()] = a.strip()

# ===== NORMALIZATION =====
SHORT_FORM_DICT = {"u": "you", "ur": "your", "r": "are"}
MISSPELLINGS = {"registartion": "registration", "applcation": "application"}
SYNONYMS = {"sign up": "register", "apply": "register"}
STOPWORDS = {"the","is","am","are","a","an","and","to","of","in","on","for"}

def replace_by_dict(text, mapping):
    for k,v in mapping.items():
        text = re.sub(r"\b" + re.escape(k) + r"\b", v, text)
    return text

def normalize_text(t):
    t = unicodedata.normalize("NFKC", t).lower()
    t = replace_by_dict(t, SHORT_FORM_DICT)
    t = replace_by_dict(t, MISSPELLINGS)
    t = replace_by_dict(t, SYNONYMS)
    t = re.sub(r"[^a-z0-9\s]", " ", t)
    return re.sub(r"\s+", " ", t).strip()

# ===== LANGUAGE DETECTION =====
def detect_language(text):
```

```
    try:
        return langdetect.detect(text)
    except:
        return "en"

# ===== TRANSLATION (Google) =====
def translate_text(text, target_lang):
    try:
        return GoogleTranslator(source="auto", target=target_lang).translate(text)
    except:
        return text

# ===== GEMINI SMART RESPONSE =====
def get_gemini_response(prompt):
    url = f"https://generativelanguage.googleapis.com/v1/models/gemini-
pro:generateContent?key={GEMINI_API_KEY}"
    headers = {"Content-Type": "application/json"}
    data = {"contents": [{"parts": [{"text": prompt}]}]}
    try:
        res = requests.post(url, headers=headers, json=data, timeout=10)
        if res.status_code == 200:
            return res.json()["candidates"][0]["content"]["parts"][0]["text"]
    except:
        pass
    return "Sorry, I couldn't process your question."

# ===== DATASET SEARCH =====
def search_dataset(user_input):
    matches = difflib.get_close_matches(user_input.lower(), faq_data.keys(), n=1, cutoff=0.6)
    if matches:
        return faq_data[matches[0]]
    return None

# ===== MAIN RESPONSE =====
```

```
def get_response(user_input):
    lang_code = detect_language(user_input)

    # Translate non-English to English for dataset search
    if lang_code != "en":
        translated_input = translate_text(user_input, "en")
    else:
        translated_input = user_input

    translated_input = normalize_text(translated_input)

    dataset_answer = search_dataset(translated_input)
    if not dataset_answer:
        dataset_answer = get_gemini_response(translated_input)

    # Translate back to user's language
    if lang_code == "pa":
        return translate_text(dataset_answer, "pa")
    elif lang_code == "hi":
        return translate_text(dataset_answer, "hi")
    else:
        return dataset_answer
```

---

app.py (same as before)

```
from flask import Flask, render_template, request, jsonify
from chatbot_logic import get_response

app = Flask(__name__)

@app.route("/")
def index():
    return render_template("index.html")
```

```
@app.route("/get", methods=["POST"])
def chatbot_reply():
    data = request.get_json()
    user_message = data.get("msg", "")
    reply = get_response(user_message)
    return jsonify({"response": reply})

if __name__ == "__main__":
    app.run(debug=True)
```

Now What Will Happen

- User Punjabi/Hindi input likhega → code English vich convert karega.
- Dataset + Gemini answer English vich hovega.
- Answer **Google Translate** naal automatically Punjabi/Hindi vich dubara translate ho ke aayega.
- Hun chatbot bilingual ho javega bina dataset ch Punjabi/Hindi manually likhan to.