DAY – 46

3 October 2025

Problem 1: Short spelling / misspelling check

- Tere code ch normalize_text() function banaya hoya si jo **short forms (u → you, ur → your)** te **misspellings (registartion → registration)** replace karan layi si.
- Problem eh aa ki **Punjabi/Hindi text normalize karan samay bhi remove ho reha** (kyunki regex sirf [a-z0-9] allow kar reha).
- Is karke Punjabi input **English cleanup vich hi remove ho janda** → match hi nahi hunda.

Problem 2: Punjabi/Hindi response sahi nahi aanda

- Gemini answer hamesha **English vich** aa reha.
- Translation GoogleTranslator karda par jadon **short spelling correct karna** miss ho janda, oh Punjabi vich vi sahi sentence generate nahi karda.

Fix

1. Normalization sirf **English inputs** te apply karni.
   - Punjabi/Hindi input nu sidha translate kar deo English vich, normalization baad karo.
2. Replace dictionary nu extend karo (u→you, ur→your, r→are etc.) so ki short spelling vi match ho jave.
3. Punjabi/Hindi answer layi → **English → Punjabi/Hindi translation step properly enforce karo**.

Corrected chatbot_logic.py

```
import re
import unicodedata
import difflib
import requests
import langdetect
from deep_translator import GoogleTranslator

# ===== CONFIG =====
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()

# ===== SHORT FORMS + MISSPELLINGS =====
SHORT_FORM_DICT = {
    "u": "you",
    "ur": "your",
    "r": "are",
    "pls": "please",
    "plz": "please",
}
MISSPELLINGS = {
    "registartion": "registration",
    "applcation": "application",
    "acount": "account",
}
```

```
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

# ===== NORMALIZATION (Only English) =====
def normalize_text(t):
    t = unicodedata.normalize("NFKC", t).lower()
    t = replace_by_dict(t, SHORT_FORM_DICT)
    t = replace_by_dict(t, MISSPELLINGS)
    # only keep English characters
    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 =====
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)

    # STEP 1: Convert input → English for matching
    if lang_code != "en":
        translated_input = translate_text(user_input, "en")
    else:
        translated_input = user_input

    # STEP 2: Normalize only if English
    translated_input = normalize_text(translated_input)

    # STEP 3: Search in dataset or Gemini
    dataset_answer = search_dataset(translated_input)
    if not dataset_answer:
        dataset_answer = get_gemini_response(translated_input)
```

```
# STEP 4: Translate back to user 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
```

Now Features

- **Short spelling fixed** → "plz registartion" → correctly understood as "please registration".
- **English normalization only** → Punjabi/Hindi input delete nahi hovega.
- **Automatic translation** → English dataset answer → back into Punjabi or Hindi.