Project: Customer Support Ticket Analyzer

Project Overview

This project simulates a real-world NLP task: analyzing customer support tickets for a product-based company.

You are provided with a synthetic dataset containing 1000 support messages. These messages are messy and
include real-world noise such as spelling mistakes, emojis, HTML tags, all-caps writing, slang, and inconsistent formatting.

Your task is to perform end-to-end text processing: clean the data, extract insights, classify messages into types,
and summarize long complaints using NLP techniques. You may optionally build an interactive app for showcasing results.

Step-by-Step Tasks

1. 1. Load and inspect the dataset (CSV format). Identify missing values, duplicates, and column types.
2. 2. Perform deep text cleaning on 'message_body': remove emojis, HTML, repeated punctuation, extra spaces, and fix common spelling errors.
3. 3. Apply tokenization, lowercasing, lemmatization, and POS tagging using spaCy or NLTK.
4. 4. Use Named Entity Recognition (NER) to extract product names, user names, locations, and issue types.
5. 5. Classify each message into one of the categories (Complaint, Bug Report, Feature Request, Praise, etc.) using traditional ML or transformers.
6. 6. For messages with >100 words, apply text summarization using pretrained models like T5 or BART.
7. 7. Create a new CSV with cleaned text, predicted category, named entities, and summary.
8. 8. (Optional) Build a Streamlit or Gradio app with filters for category, city, or keyword search.

Deliverables

- Cleaned dataset (CSV) with new columns for category, summary, and extracted entities
- Python notebook with all steps (cleaning, NLP, classification, summarization)
- Output CSV (cleaned and annotated)
- PDF summary (optional)
- (Optional) Streamlit/Gradio-based mini web app to showcase insights

Brownie Points

- You may use regex, BeautifulSoup, or emoji libraries for cleaning.
- For classification: try both TF-IDF + LogisticRegression and Transformer-based pipelines (like BERT).
- For summarization: use HuggingFace models like 't5-small' or 'facebook/bart-large-cnn'.
- Keep modular code for each phase of the pipeline.