INTERNSHIP REPORT

Name: Ayesha Liaqat

Intern ID: DHC-1059

Department: AI / Machine Learning

Advanced task2 Smart Ticket Chatbot

This document provides an overview of the tasks and learning outcomes from the AI/ML internship program at Developers Hub Corporation.

DEVELOPERS HUB CORPORATION - AI/ML INTERNSHIP REPORT

Project Title: Smart Ticket Chatbot

Objective

Build a smart support chatbot that can auto-tag customer support tickets and return app-specific solutions using an ML model and a fallback similarity search — all wrapped in a Flask-based interactive web app.

✅ Tasks Performed

- Preprocessed raw support tickets from multiple apps
- Built a multiclass classifier using TF-IDF + Logistic Regression
- Encoded labels with LabelEncoder
- Created a fallback logic using **cosine similarity** on ticket embeddings
- Stored FAQs/solutions in a JSON-based knowledge base
- Built a **Flask** app to:
    - Predict issue category (tag)
    - Identify app name using keyword matching
    - Fetch solutions or fallback recommendations
- Stored model files using joblib
- Created an interactive chat-like frontend using HTML + Flask sessions

---

🧠 New Concepts Learned

- Auto-tagging support tickets with ML
- Handling unknown/unmapped tickets via similarity fallback
- Text preprocessing + vectorization (TF-IDF)
- Flask session-based multi-step interactions
- Dynamic solution retrieval from a knowledge base
- Clean and user-friendly chatbot interface

📊 Tech Stack

- Python, Scikit-learn, Pandas, Regex
- Flask for backend & routing

- HTML + CSS for the chatbot interface
- joblib for model/vector persistence
- Jinja2 templates for rendering responses
- cosine_similarity for fallback solution matching

🚀 How It Works

1. User submits their issue via the chatbot.
2. App:
    - Predicts the issue category (tag)
    - Extracts app name from the query
    - Tries to fetch a solution from solution_bank.json
    - If not found, uses cosine similarity to retrieve the most relevant past ticket
3. Displays the solution and asks: *"Did that help?"*
4. Cycles through solutions until user confirms or exhausts options.

🔍 Fallback Search

When no direct match is found in the solution bank:

- Uses **TF-IDF** to vectorize the ticket
- Computes **cosine similarity** with all known tickets
- Picks the most similar one for fallback help

🧠 Conclusion

This project demonstrates:

- How **LLM-inspired workflows** can be implemented with lightweight ML
- A great example of **practical text classification** for real-world apps
- How to build **smart helpdesk systems** without using heavy cloud APIs

---

📌 Future Enhancements

- Integrate OpenAI or Hugging Face LLMs for fallback reasoning
- Add button-based user responses instead of typing
- Store chat history

- Add user login for tracking issues