PROJECT DOCUMENTATION

MindCare AI: Mental Health FAQ Chatbot

Submitted by: Muneeb

Artificial Intelligence Intern | *CodeAlpha*

February 2026

1. Executive Summary

This document outlines the architecture, methodology, and technical implementation of 'MindCare AI', an intelligent FAQ chatbot developed to fulfill Task 2 of the CodeAlpha Artificial Intelligence Internship. The objective was to build a robust NLP-based system capable of matching user inquiries with pre-defined answers regarding mental health. By integrating a Python and Flask backend with a responsive HTML/CSS/JS frontend, the project successfully demonstrates a full-stack AI deployment.

2. Technology Stack

- Core Language: Python 3.x
- Natural Language Processing: NLTK (Tokenization, Lemmatization, Stop-word removal)
- Machine Learning Core: scikit-learn (TfidfVectorizer, cosine_similarity)
- Data Manipulation: Pandas
- Backend Web Server: Flask, Flask-CORS
- Frontend: HTML5, CSS3 (Glassmorphism UI), Vanilla JavaScript
- Version Control & Deployment: Git, GitHub

3. System Architecture & Data Flow

The application operates on a seamless Client-Server model to separate the user interface from the heavy machine learning computations:

3.1 The Client (Frontend)

The user interfaces with a highly responsive web application featuring a modern 'glassmorphism' aesthetic. When a user types a query, an asynchronous JavaScript Fetch request is triggered. The UI instantly displays a dynamic 'AI is thinking...' animation, ensuring a natural, interactive user experience while waiting for the server's response.

3.2 The Server (Backend)

A lightweight Flask server listens for POST requests on the '/ask' endpoint. Upon receiving a user's JSON payload, the server passes the text through the NLP pipeline, calculates the best mathematical match using the loaded Pickle (.pkl) models trained in Google Colab, and returns

the optimized text response back to the client.

4. NLP Methodology & Mathematical Model

4.1 Dataset Expansion & Augmentation

The foundation of the bot is built upon a curated mental health dataset initially sourced from Kaggle (98 core Q&A pairs). To improve the model's resilience against varied user conversational phrasing, programmatic data augmentation was applied. Conversational prefixes were added, effectively multiplying the dataset to 588 unique interactions.

4.2 Text Preprocessing Pipeline

Raw text is inherently messy. To prepare the user's input for the AI, the text undergoes strict normalization:

- Normalization: All characters are converted to lowercase and stripped of punctuation using regular expressions.
- Tokenization: Sentences are split into discrete words using NLTK's punkt_tab tokenizer.
- Stop-word Removal: Low-information words (e.g., 'the', 'and', 'is') are discarded to reduce noise.
- Lemmatization: Words are reduced to their dictionary root (e.g., 'running' becomes 'run') using the WordNetLemmatizer, maximizing the matching accuracy of core concepts.

4.3 TF-IDF Vectorization & Cosine Similarity

The processed text strings are converted into numerical arrays using the Term Frequency-Inverse Document Frequency (TF-IDF) algorithm. This technique assigns higher mathematical weight to rare, highly relevant keywords and lower weight to frequent, generic words.

To find the correct answer, the system calculates the Cosine Similarity between the user's input vector and the vectors of all known FAQs in the dataset. This mathematical formula measures the cosine of the angle between two multi-dimensional vectors, returning a score between 0 and 1. The bot returns the answer mapped to the highest score, provided it surpasses a 20% confidence threshold.

5. Development Challenges & Solutions

During local deployment, a key challenge emerged regarding NLTK's data dependencies, specifically the missing 'punkt_tab' resource required for tokenization in newer NLTK versions. This was successfully resolved by explicitly defining the download command within the Flask application's initialization sequence, ensuring seamless cross-platform execution without manual environment setup.

6. Conclusion

MindCare AI effectively showcases the practical application of Natural Language Processing in a real-world scenario. By completing this project from data preparation in Google Colab to local deployment with an interactive web UI, fundamental concepts of AI integration and software engineering have been solidified, successfully fulfilling the requirements of the CodeAlpha AI Internship program.