TalentScout Hiring Assistant Chatbot

Project Overview

The TalentScout Hiring Assistant is a conversational chatbot designed to assist in the initial screening of candidates applying for tech positions. The chatbot gathers essential candidate information, including their name, contact details, years of experience, desired positions, and tech stack. Based on the declared tech stack, it generates technical questions tailored to assess the candidate's proficiency in relevant technologies. It also offers multilingual support, sentiment analysis, and a streamlined user experience via the Streamlit UI.

Features

1. Multilingual Support

- The chatbot asks the user to select their preferred language before starting the conversation. After language selection, all subsequent prompts and input fields are shown in the selected language.
- Languages supported: English (en), Tamil (ta), Telugu (te), Kannada (kn), Malayalam (ml), Hindi (hi).

2. Information Gathering

- The chatbot gathers essential details:
    - Full Name
    - Email Address
    - Phone Number
    - Years of Experience
    - Desired Position(s)

- Current Location
- Tech Stack (e.g., Python, SQL, AI, ML, LLM, etc.)

3. Technical Question Generation

- Based on the tech stack declared by the candidate, the chatbot generates 3-5 technical questions to assess proficiency in each specified technology.

4. Sentiment Analysis

- The sentiment of the candidate's responses is analyzed using TextBlob, providing feedback on the emotional tone of the conversation (Positive, Neutral, Negative).

5. Prompt Engineering

- Effective prompts have been designed to gather candidate information and generate relevant technical questions. These prompts guide the language model to produce clear, contextually relevant outputs.

6. Data Privacy and Security

- Sensitive candidate information is securely handled using Streamlit's session state. Data is anonymized for storage.

7. End Conversation

- Once all information is gathered and technical questions are provided, the chatbot ends the conversation and thanks the candidate.

Technologies Used

1. **Python**: The primary programming language for the project.
2. **Streamlit**: Used to create an interactive UI for the chatbot.
3. **Google Generative AI (Gemini API)**: Used to generate content and technical questions based on the candidate's tech stack.
4. **Google Translate API**: Used to translate text for multilingual support.
5. **TextBlob**: Used for sentiment analysis of candidate responses.
6. **ngrok**: Used to create a public URL for running the Streamlit app locally.

System Requirements

1. **Python 3.12**
2. **Libraries**:
    - streamlit
    - google-generativeai
    - googletrans
    - textblob
    - pyngrok
    - requests

Required Libraries:

You can install the necessary libraries using the requirements.txt file by running the following command:

pip install -r requirements.txt

The requirements.txt file includes:

- streamlit==1.10.0
- google-generativeai==0.1.1
- googletrans==4.0.0-rc1
- textblob==0.15.3
- pyngrok==5.1.0

How to Run the Application

1. Setup the Environment

- Make sure you have Python 3.12 installed.
- Install the required libraries using the requirements.txt file.

2. Run the Streamlit App

- Run the app using Streamlit

3. Access the App

- Once the Streamlit app is running, it will provide a public URL through ngrok. This URL will be displayed in the output, and you can open it in your browser.

Documentation for Code Components

Main Flow of the Application

1. **Language Selection**:
    - When the user accesses the application, they are first prompted to select a language (e.g., Tamil, Telugu, English). This selection impacts the language of all subsequent prompts and labels.
2. **Information Gathering**:
    - After language selection, the chatbot asks the user for basic information (name, email, phone, experience, etc.).
    - This information is captured using Streamlit's input fields.
3. **Tech Stack Declaration**:
    - The chatbot asks the candidate to declare their tech stack (e.g., Python, Django, React, etc.).
    - The tech stack information is stored for generating the technical questions.
4. **Generating Technical Questions**:
    - Once the tech stack is declared, the chatbot generates technical questions using the **Google Gemini API**.
    - The questions are displayed to the user, allowing the chatbot to assess the candidate's technical skills.
5. **Sentiment Analysis**:
    - TextBlob analyzes the sentiment of the technical questions to provide feedback on the emotional tone (Positive, Neutral, Negative).
    -

6. **Multilingual Support**:
    - The **Google Translate API** is used to translate the prompt messages and responses into the selected language.
    - The chatbot displays the translated content along with the technical questions.
7. **End the Conversation**:
    - When the process is complete, the chatbot thanks the candidate and ends the conversation.

GitHub Repository Structure

Github Link: https://github.com/RUKESH-1704/TalentScout-Hiring-Assistant-Chatbot

The project repository includes the following files:

- TalentScout Hiring Assistant Chatbot.ipynb: The main Notebook file containing the Streamlit app.
- requirements.txt: The list of required libraries to run the app.
- README.md: A detailed readme file describing the project and setup instructions.
- Demo Video also attached in Github Repo

Conclusion

This project demonstrates the use of a large language model (Google Gemini) for generating technical questions based on a candidate's tech stack. It integrates multilingual support, sentiment analysis, and a smooth user interface through Streamlit. The solution meets the requirements for gathering candidate information, generating technical assessments, and handling sensitive data securely.