***Chatbot* com *Retrieval-Augmented Generation* (RAG)**

Atendimento de Reclamações Aéreas

1. Introdução

- **Descrição do Problema:**
    - No setor de atendimento ao cliente da ANAC, há um alto volume de reclamações frequentemente repetitivo sobre direitos do passageiro, bagagens extraviadas, cancelamentos, (ex: regras de bagagem, cancelamentos, atrasos).
    - As equipes de atendimento podem se sobrecarregar, levando a tempos de espera mais longos e potenciais inconsistências nas informações fornecidas ao buscar em manuais/PDFs.
    - Os passageiros podem ter dificuldade em encontrar rapidamente informações precisas e relevantes em meio a extensos documentos ou FAQs.
- **Objetivos do Chatbot:**
    - Desenvolver um chatbot com RAG para fornecer respostas rápidas e precisas às perguntas dos passageiros sobre seus direitos, conforme regulamentações da ANAC.
    - Reduzir a carga de trabalho da equipe de atendimento, automatizando respostas a perguntas frequentes.
    - Melhorar a experiência do cliente, oferecendo acesso fácil e eficiente à informação.

2. Metodologia

- **Etapas de Desenvolvimento:**
    1. **Definição do Problema:** Conforme descrito na seção anterior, o problema central é a dificuldade em fornecer atendimento rápido e eficiente às consultas de passageiros.
    2. **Planejamento da Solução:**
        - O chatbot utilizará a técnica RAG para recuperar informações relevantes de um documento PDF contendo o histórico de reclamações respondidas pela ANAC.

        - A LangChain será usada para orquestrar o processo de recuperação e geração de respostas.
        - O modelo gpt-3.5-turbo da OpenAI será empregado para gerar respostas concisas e informativas.
        - O Streamlit será utilizado para criar a interface do chatbot.
    3. **Implementação:**
        - A aplicação foi desenvolvida em Python.
        - As bibliotecas utilizadas incluem: Streamlit, LangChain, OpenAI, PyPDF2, FAISS.
        - O código foi estruturado para inicializar o estado da sessão do Streamlit de forma adequada, garantindo o funcionamento correto dos widgets.
        - A lógica RAG foi implementada para carregar, processar o PDF, criar um índice vetorial e gerar respostas contextuais.
    4. **Entrega:**
        - A aplicação foi implantada no Streamlit Cloud para acesso online.
- **Detalhamento das Etapas e Escolhas Técnicas:**
    - **Linguagem de Programação:** Python foi escolhido devido à sua vasta disponibilidade de bibliotecas de IA e facilidade de uso.
    - **Framework de IA:** LangChain simplifica a implementação de cadeias de linguagem, incluindo a integração com modelos de linguagem e bancos de dados vetoriais.
    - **Modelo de Linguagem:** O modelo gpt-3.5-turbo da OpenAI oferece um bom equilíbrio entre custo e desempenho para geração de texto.
    - **Biblioteca de PDF:** PyPDF2 é usada para extrair o texto do arquivo PDF.
    - **Banco de Dados Vetorial:** FAISS permite a busca eficiente de similaridade em embeddings, acelerando a recuperação de informações relevantes.
    - **Framework de Interface:** Streamlit facilita a criação de interfaces web interativas para aplicações de aprendizado de máquina.

- **Estado da Sessão:** O st.session_state do Streamlit é utilizado para persistir informações entre as interações do usuário, como o histórico de conversas e o vetor store carregado.
- **Tratamento de Erros:** Blocos try-except são usados para capturar e exibir mensagens de erro amigáveis ao usuário.

3. Funcionamento

- **Orientações de Uso do Chatbot:**
  - O usuário interage com o chatbot através de um campo de texto na interface web.
  - O usuário digita sua pergunta sobre direitos do passageiro (ex: "Quais os direitos em caso de cancelamento?").
  - O chatbot processa a pergunta e exibe a resposta gerada.
  - O usuário pode digitar "sair" para encerrar a conversa.
  - Um expander "Histórico de Conversas" exibe o histórico das interações.
- **Estrutura da Base de Conhecimento:**
  - A base de conhecimento é um arquivo PDF ("Chatbot_SAC.pdf") contendo um compilado das reclamações recebidas e respondidas pela ANAC.
  - O PDF é carregado e processado, sendo dividido em chunks de texto.
  - Esses chunks são convertidos em embeddings e armazenados em um banco de dados vetorial FAISS para recuperação eficiente.
- **Limitações Conhecidas:**
  - A precisão das respostas depende da qualidade e abrangência das informações no PDF fornecido.
  - O chatbot pode ter dificuldades em responder a perguntas fora do escopo do documento.
  - O modelo de linguagem pode ocasionalmente gerar respostas imprecisas ou ambíguas.
  - O desempenho do chatbot pode ser afetado pelo tamanho do PDF e pela complexidade das perguntas.

4. Resultados

- **Impacto Esperado ou Observado da Solução no Ambiente Laboral:**
    - Espera-se uma redução significativa no tempo de resposta às consultas dos passageiros.
    - A equipe de atendimento poderá se concentrar em casos mais complexos e que exigem interação humana.
    - Haverá um aumento na satisfação do cliente devido ao acesso rápido e fácil à informação.
    - A consistência das respostas será aprimorada, garantindo que todos os passageiros recebam as mesmas informações corretas.
    - O chatbot pode funcionar 24 horas por dia, 7 dias por semana, fornecendo suporte contínuo.

Nota explicativa: Esse trabalho foi desenvolvido com auxílio de ferramentas de inteligência artificial.