PROVA PARA DESENVOLVEDOR BACKEND COM FOCO EM IA

Instruções:

- [ ] Responda todas as questões abaixo.
- [ ] Utilize o Python para implementar suas soluções quando necessário.
- [ ] Certifique-se de comentar seu código para explicar sua lógica.

Questão 1: Desenvolvimento de API com Django/Flask/FastAPI

Desenvolva uma API simples que permite aos usuários cadastrar e consultar livros em uma biblioteca virtual. A API deve incluir as seguintes funcionalidades:

1. Cadastro de livros com os campos: título, autor, data de publicação e resumo.
2. Consulta de livros por título ou autor.
3. Implemente a API utilizando um dos frameworks: Django, Flask ou FastAPI.

Certifique-se de:

- Criar endpoints claros e bem documentados.
- Utilizar um banco de dados SQLite para armazenamento.
- Implementar testes unitários para os endpoints criados.

Dicas:

- Para Django, considere utilizar o Django Rest Framework (DRF).
- Para Flask, considere utilizar Flask-RESTful.
- Para FastAPI, utilize os recursos nativos do framework para criação de APIs.

Questão 2: Implementação de Chatbot com IA Generativa (Langchain, Langsmith, LLMs)

Você precisa desenvolver um chatbot que utilize um modelo de linguagem (LLM) como o GPT-4 da OpenAI para responder perguntas dos usuários sobre programação em Python. O chatbot deve:

1. Receber perguntas dos usuários via input de texto.
2. Utilizar o Langchain para gerenciar o fluxo de conversação e integrar com o LLM.
3. Responder às perguntas utilizando o modelo da OpenAI.

Implemente um exemplo simples onde o usuário possa perguntar algo como "Como criar uma lista em Python?" e o chatbot responda com uma explicação detalhada.

Dicas:

- Utilize o Langchain para facilitar a integração e gerenciamento das respostas do LLM.
- Certifique-se de configurar corretamente a API da OpenAI.
- Forneça exemplos de perguntas e respostas para demonstrar o funcionamento do chatbot.

Questão 3: Trabalhando com Vector Stores e Embeddings

Você deve criar um sistema de busca semântica de documentos utilizando embeddings e vector stores. Para isso, siga as etapas abaixo:

1. Utilize um conjunto de documentos de texto (pode ser um conjunto de artigos ou posts de um blog).
2. Gere embeddings para esses documentos utilizando um modelo de embeddings.
3. Armazene esses embeddings em uma vector store como FAISS ou Milvus.
4. Implemente uma função de busca que, dado um texto de consulta, retorne os documentos mais relevantes com base na similaridade semântica.

Dicas:

- Utilize bibliotecas como transformers para gerar embeddings.
- Documente o processo de criação dos embeddings e armazenamento na vector store.
- Demonstre a busca semântica com exemplos de consultas e resultados relevantes.