Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa

GRADUAÇÃO EM BANCO DE DADOS

Projeto de Bloco: Ciência de Dados Aplicada [24E3\_5]

WANDERSON RAFAEL MENDONÇA BATISTA

TESTE DE PERFORMANCE – TP4

PROF. DIEGO DA SILVA RODRIGUES

RIO DE JANEIRO, 2024

**Configuração do Ambiente de Desenvolvimento:**

Configure seu ambiente de desenvolvimento, incluindo Git para controle de versão e preparação para deploy. Lembre-se de seguir a estrutura do CRISP-DM para organizar seu projeto de forma eficiente e escalável.

https://github.com/wanderaf/ProjetoDeBloco

1. **Identificação e Escolha do Modelo LLM (Local):**

* **Critérios de Seleção: Pesquise e selecione o modelo de linguagem natural mais adequado para a sua aplicação, considerando os seguintes critérios:**
* **Desempenho: Avalie a precisão, a capacidade de resposta e o tipo de tarefa para a qual o modelo foi treinado (ex: GPT, BERT, T5).**
* **Custo Computacional Local: Verifique os recursos de hardware necessários para rodar o modelo no ambiente local (como GPU, memória RAM, etc.).**
* **Acessibilidade Local: Garanta que o modelo escolhido pode ser carregado localmente, utilizando bibliotecas como Transformers da HuggingFace para download e execução do modelo em sua própria máquina.**

O modelo escolhido para a aplicação foi o t5-small, da arquitetura T5, devido à sua capacidade de realizar tarefas de transformação de texto com precisão e eficiência, como previsão baseada em padrões históricos. Esse modelo apresenta requisitos computacionais moderados, sendo adequado para execução em CPUs com pelo menos 8 GB de RAM ou em GPUs para maior desempenho, garantindo viabilidade em ambientes locais com recursos limitados. Além disso, sua acessibilidade é facilitada pela integração com a biblioteca Hugging Face Transformers, permitindo carregamento e execução local de forma independente, com suporte a otimizações como cache de recursos para maior eficiência operacional.

1. **Integração de LLM com FastAPI no Ambiente Local:**

**Execução Local: Utilize um modelo da HuggingFace ou um modelo treinado localmente para realizar tarefas de processamento de linguagem natural (como geração de texto, resumo automático, classificação de sentimento, etc.) e integre-o ao seu backend FastAPI, sem necessidade de conexão com a nuvem.**

**Rota FastAPI: Implemente uma rota em FastAPI que se conecte ao modelo LLM rodando localmente para processar dados textuais fornecidos pela aplicação. Exemplo de rota:**

**POST /processar\_texto: Rota que recebe um texto enviado pelo usuário e retorna a análise ou processamento realizado pelo modelo de linguagem, como um resumo ou análise de sentimento.**

**Execução Local do HuggingFace: Baixe o modelo diretamente em seu ambiente de desenvolvimento local e utilize as funções da biblioteca Transformers para carregar e executar o modelo sem depender de serviços externos.**

Modelo HuggingFace

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Get -> Busca nome das unidades através de palavras chaves

**Texto

Descrição gerada automaticamente**

Post -> Acrescenta novo registro no banco

**Texto

Descrição gerada automaticamente**

1. **Manipulação das Respostas da API:**

**Response Models: Crie Response Models em FastAPI para garantir que as respostas da API sejam consistentes e estruturadas, utilizando modelos de resposta claros e bem definidos.**

**Exemplo: A rota /processar\_texto deve retornar um JSON estruturado com informações detalhadas, como o texto processado, o tipo de análise realizada (resumo, classificação, etc.) e os resultados gerados.**

**Validação das Respostas: Assegure-se de que todos os retornos da API estejam validados, garantindo que os dados enviados e recebidos estejam no formato correto e sejam compreensíveis para o cliente.**

Foram realizadas modificações em relação ao código apresentado na questão anterior afim de aplicar o Response Models:

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

1. **Tratamento Robusto de Erros na API:**

**Exceções HTTP: Implemente um tratamento robusto de erros na API, utilizando exceções HTTP específicas em FastAPI para lidar com problemas que possam ocorrer durante as requisições (como falha ao carregar o modelo, problemas com os dados de entrada, ou timeouts).**

**Exemplo: Se o modelo LLM não estiver carregado corretamente ou ocorrer um erro na manipulação dos dados, a aplicação deve retornar um código de erro adequado (ex: 503 Serviço Indisponível) e uma mensagem explicativa.**

**Melhoria da Experiência do Usuário: Garanta que os erros sejam tratados de forma clara e informativa, proporcionando feedback útil para o usuário final e facilitando o diagnóstico de possíveis problemas.**

Exemplos de tratamento

**Texto

Descrição gerada automaticamente**

**Texto

Descrição gerada automaticamente** **Texto

Descrição gerada automaticamente**