Link repositório: https://github.com/Michelletxr/AluMind.git Principais Tecnologias usadas

- Java (versão 21)
- Maven
- Spring (Spring Framework, Spring Boot, Spring Data JPA, Spring AI);
- PostgreSQL
- LLMs (OpenAl Chat)
- Git e Github
- Docker
- Swagger

•

Explicação da utilização do OpenAl:

Optei por realizar a integração com a ferramenta da OpenAl para modelos de LLMs, devido sua praticidade, além de sua documentação detalhada e simples.

Próximos passos:

Umas das possíveis melhorias no projeto seria adicionar mais ferramentas disponibilizadas pelo framework do Spring AI, visto que esse possui uma gama enorme de funcionalidades que ajudam na integração de modelos LLMs em aplicações. Além disso, seria interessante adicionar novos modelos de LLM (como o google gemini) para disponibilizar diferentes abordagens de desenvolvimento.

Documentação API



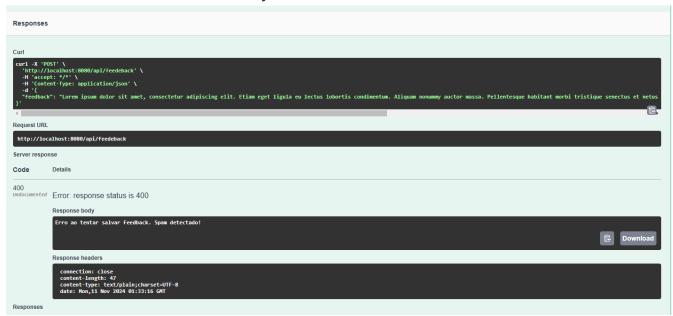
Case 1 - Classificação de Feedbacks Resposta da API:

• Usuário envia feedback positivo

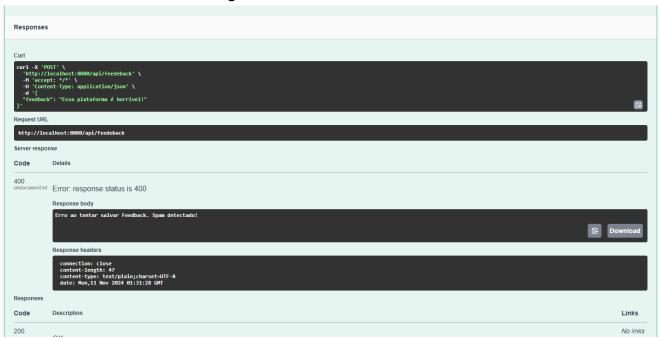
• Usuário envia feedback negativo

Case 2 - Marcação de SPAM

Conteúdo do feedback sem relação com a AluMind



• Feedback com conteúdo agressivo



Case 3 - Nova feature

Análise automatizada dos feedbacks para a criação de possíveis features ou melhorias na plataforma.

A intenção é usar modelos de LLMs para avaliar os feedbacks realizados pelos usuários e que foram salvos na base de dados da aplicação. Por meio da avaliação o modelo de LLM deve processar os feedbacks negativos e analisar quais os problemas mais relevantes apontados pela análise, por exemplo o modelo determinou que muitos usuários fizeram feedbacks negativos em relação a demora para carregar os vídeos na plataforma. Dessa forma, o modelo pode auxiliar a equipe de desenvolvimento na criação de novas funcionalidades para amenizar a problemática. Essa abordagem iria contribuir a equipe para focar na resolução dos problemas mais relevantes na plataforma.

Além disso, ao serem lançadas novas funcionalidades e melhorias na plataforma, o modelo poderia filtrar o perfil dos usuários que realizaram feedbacks negativos e propor sugestões de ofertas para manter o usuário na plataforma.

Por exemplo, o modelo identificou que um usuário está bastante insatisfeito e que sua assinatura na plataforma está próxima de acabar, o modelo pode sugerir uma proposta de desconto para o usuário baseado no seu perfil e ressaltar que os problemas relatados pelo usuário foram tratados por meio das melhorias feitas feitas na plataforma, incentivando assim, o usuário a permanecer na plataforma.

Case 4 - Geração de respostas personalizadas

Para geração de respostas personalizadas foi usado uma requisição http Post, sendo necessário enviar o identificador único do feedback como parâmetro na requisição. Optei por fazer dessa forma, pois apesar da criação de respostas personalizadas e avaliação de feedbacks (como positivos ou negativos) usarem abordagens com LLMs, são duas funcionalidades diferentes dentro da aplicação. Ademais, a adesão de endpoints separados facilita a criação de respostas personalizadas usando diferentes modelos de LLM.

Resposta personalizada para feedback positivo



Resposta personalizada para feedback negativo

