Diário de Estudo — LangChain + Azure OpenAI

Objetivo

Este diário de estudo tem como objetivo registrar a jornada de aprendizado sobre a construção de agentes conversacionais utilizando a biblioteca LangChain em conjunto com o Azure OpenAI (ChatGPT).

Planejamento inicial

Antes de iniciar a prática, defini dois caminhos possíveis: (1) implementar um agente funcional usando LangChain + Azure OpenAI, ou (2) documentar em detalhes a teoria, os conceitos principais e os exemplos de código explorados. Escolhi o caminho da documentação, pois queria compreender a fundo os componentes da arquitetura.

Conceitos-chave aprendidos

- LangChain organiza aplicações em blocos: LLMs, Chains, Agents, Tools e Memory.
- Agents permitem que o modelo decida qual ferramenta usar para resolver uma tarefa.
- O Azure OpenAI exige configuração de endpoint, chave de API e nome do deployment.
- A integração com LangChain pode variar conforme a versão da biblioteca (mudanças frequentes).
- É importante proteger credenciais usando variáveis de ambiente (.env).

Exemplo de código explorado

Durante o estudo, escrevi um exemplo mínimo que conecta o LangChain a um deployment do Azure OpenAI. O código inicial utiliza a classe ChatOpenAI, configurada com as variáveis de ambiente AZURE_OPENAI_ENDPOINT, AZURE_OPENAI_KEY e AZURE_OPENAI_DEPLOYMENT.

Principais dificuldades

- Configuração correta do deployment no Azure OpenAI (nome exato precisa coincidir).
- Mudanças na API do LangChain: algumas versões usam AzureChatOpenAI, outras ChatOpenAI.
- Controle de custos e limites de uso dos modelos no Azure.
- Necessidade de sanitizar entradas para evitar problemas de segurança.

Reflexões pessoais

Aprender a usar o LangChain com o Azure foi um exercício valioso para entender como estruturar aplicações de IA mais complexas. A documentação oficial nem sempre é clara, então manter um

diário de estudo ajudou a consolidar conceitos. Pretendo evoluir para a implementação de agentes com memória e múltiplas ferramentas.

Próximos passos

- Implementar um agente completo com memória de conversação.
- Adicionar ferramentas personalizadas (busca, calculadora).
- Explorar integração com bancos de dados vetoriais (retrievers).
- Documentar resultados em notebooks com exemplos reproduzíveis.