**Build Your Own Local RAG AI Agent with Python & Ollama - Complete Tutorial**

**Transcrito por** [**TurboScribe.ai**](https://turboscribe.ai/pt/?ref=docx_export_upsell)**.** [**Atualize para Ilimitado**](https://turboscribe.ai/pt/subscribed?ref=docx_export_upsell) **para remover esta mensagem.**

Em nome de Deus, o abençoo e a salve. Nesta leitura de hoje, vamos criar um agente de AI que vamos deployá-lo localmente. E não precisamos pagar nenhuma tarifa para o API key do OpenAI ou o API key do CloudAnthropic.

Na última leitura, usamos os dois API keys para criar um agente com o qual fizemos uma pesquisa. Por exemplo, você dava um keyword e recebíamos resultados múltiplos. E a leitura de AI que temos aqui já está disponível.

Você pode ver na tela também. Se a gente ir para codainx.com e ir para o canal do YouTube de Codainx, você pode ver facilmente. Aqui está a nossa leitura, Python AI Research Agent, Build Your Own GPT.

Então, nós usamos dois API keys aqui. Eu vou deixar o link na descrição também. Hoje, o nosso trabalho será totalmente diferente, porque o agente que nós vamos criar vai ser muito pesado.

Quais são as razões? Por que? Por que eu estou focando mais no agente? Vamos falar um pouco sobre isso. A primeira razão é que o agente que nós vamos criar, esse agente vai ser localizado. Você não precisa ter nada como o OpenAI ou o API key.

Você não precisa instalar nada ou comprar nada. Tudo isso vai ser localizado. Localizado significa que quando você traz um modelo para o seu PC, quando você traz um modelo para o seu PC, esse modelo vai ser localizado no seu PC.

E com esse mesmo PC, você pode localizar seu modelo, você pode localizar seus queries, e você pode criar um sistema baseado no Rack. O meu pen é um pouco desligado. Isso pode causar problemas.

Dê-me um segundo, eu vou tentar conectá-lo wirelessmente. Se eu puder fazer isso. Espero que isso funcione bem agora.

Então, nós temos um modelo. Sim, agora está funcionando bem. Então, nós vamos usar um sistema baseado no Rack.

Ou seja, nós vamos trabalhar através do Retrieval Augmented Generation. E o nosso modelo vai ser o Ulama, que nós vamos usar. E nós vamos usar dois tipos de modelos.

Um vai ser o de Embedding. E o outro vai ser o Large Language Model. E nós vamos utilizar esses dois modelos de Open Source.

O que é o benefício do Open Source? Primeiramente, eles estão disponíveis gratuitamente. E segundo, você não precisa comprar eles. E você pode localizá-los facilmente.

Primeiramente, antes de criar um projeto, o que você tem que fazer é ir à página do Ulama. Se você ainda não instalou o Ulama, você tem que instalá-lo antes. Você tem que instalá-lo antes.

Se você clicar aqui, você terá o Ulama. Ele funciona em macOS, Linux e Windows. Depois de instalado, você terá algo assim.

Ele vai aparecer aqui. Se eu zoomar aqui um pouco, você pode ver que ele está funcionando assim. Você terá um tipo de Ulama.

Ele vai funcionar assim. Você também pode ver os logs e outras coisas. Além disso, você também pode usar o Command Prompt.

Eu clico aqui e vou para o Command Prompt. Depois de entrar no Command Prompt, você pode usar ele muito facilmente. Por exemplo, eu escrevo Ulama.

Eu escrevo "-version". Ele vai me mostrar a versão. A versão atual está em 0.93. 0.9 e 0.3. Eu atualmente updatei isso.

Antes de começar a aula. Se você também ver a versão atual, isso é bom. Senão, precisamos verificar isso.

A documentação também está disponível. Você também pode extrair os diferentes comandos. Por exemplo, se eu escrever Ulama cheat sheet, você pode encontrar alguns comandos.

Você pode usar esses comandos para criar diferentes códigos. As pessoas já fizeram esse tipo de cheat sheet. Você também pode acessá-lo.

Por exemplo, a gente tem um cheat sheet em Word. E o cheat sheet do Microsoft Word também está disponível. Assim.

Então, como usar o cheat sheet, qual é o modelo, e quais são as coisas envolvidas. E assim por diante. Nós podemos ver todas as coisas aqui.

E podemos utilizar esse cheat sheet também. E esse cheat sheet, eu vou postá-lo aqui também. Então, você pode ir no GitHub e copiar o cheat sheet.

E você também pode acessá-lo. Então, eu vou postá-lo aqui também. Certo.

Então, nós instalamos o Ulama. Nós vimos como o Ulama funciona. Eu já mostrei isso nas leituras anteriores.

Agora, vamos para o comando prompt. Eu escrevo Ulama aqui. Depois de Ulama, eu escrevo List.

Então, estes são alguns modelos que já estão funcionando no Ulama. Ou seja, todos os modelos do meu sistema já estão disponíveis. Eu tenho o Lemma 2, o Phi 3, o último do Phi 3, o Mini, e também o Nomic Impact Text.

Você pode deletá-los também. Você pode usá-los também. Isso é totalmente opcional.

Mas hoje, nós vamos... Na leitura de hoje, os dois modelos que nós vamos usar, eu vou te contar sobre eles. Vamos iniciá-los na parte de trás, para que eles se instalem. E depois, vamos ver como utilizá-los.

O que eu já fiz, eu já criei um folder, que vai aparecer no GitHub. O nome dele é 02 Local RAG AI Agent. Tem local, RAG e AI Agent.

Então, eu escrevi o nome dele. E este folder já está funcionando, o folder número 9. Se você sabe, este local RAG AI Agent vai aparecer. E este vai estar disponível no GitHub.

Tudo vai estar disponível lá em cima. Então, você não precisa se preocupar com nada. Ok.

Agora, vamos para o nosso folder. No folder, eu já criei um readme file no começo, com o qual você vai saber o que nós vamos fazer com este AI Agent. Eu vou zoar um pouco.

Para que seja mais fácil para você. Desculpe por isso. Aqui.

Vamos começar. Este vai ser o local RAG AI Agent. Ele vai ser baseado na geração de retrieval aumentada.

No qual, nós vamos dar um file. Com esse file, ele pode ser do CSV, pode ser do .docs, pode ser qualquer outro file. Com esse file, este modelo, este AI Agent que nós estamos criando, ele vai ler isso.

Com esse file, ele vai ler o que está dentro deste file. Ele vai converter isso para o database vector. O que é o database vector? É o formato legítimo do file para o modelo.

Ok. O formato legítimo da máquina. Depois de converter, você pode responder a pergunta.

E este vai ser seu AI Agent conversacional. Ok. Apenas dê um file, e este AI Agent conversacional vai aparecer.

O que ele vai fazer? Ele vai levar a informação relevante do cartão e vai summarizar também. E ele vai dar a informação para você. As características dele serão que ele vai ser integrado no local LLM.

Nós usaremos o LLMA 3.2. Este modelo nós usaremos para a geração de texto. Como eu disse, nós usaremos dois modelos. O primeiro é o LLMA 3.2. Nós usaremos este.

E o segundo modelo que nós usaremos, eu vou falar sobre ele. O LLMA 3.2 vai ser usado para a geração de texto. Para a searcha de vectores, nós vamos implementar o ChromaDB.

O ChromaDB é basicamente um database que nós geramos. E depois, nós vamos searchar dentro do database os diferentes tipos de tokens que nós temos. Nós processamos documentos.

Por exemplo, nós vamos processar revistas de restaurante. Você pode colocar qualquer arquivo de CSV. Mas aqui, por exemplo, nós vamos usar um exemplo de revistas de restaurante.

Então, nós podemos usar diferentes tipos de revistas ou diferentes tipos de arquivos de AI para gerar um arquivo de CSV. Você pode fazer uma conversa interativa dentro do interface da linha de comandos. O seu assinamento será que eu vou mostrar o interface da linha de comandos.

Você vai desenvolver sua aplicação usando o Streamlet. Então, você vai usar este tipo de procedimento. Mas você tem que fazer sua aplicação usando o Streamlet.

Como eu disse nos últimos dias, usando o LLMA, você pode criar uma aplicação usando o Streamlet. Você só tem que cuidar dos portos. Para saber qual é a entrada e qual é a saída.

Então, você pode ver isso nas leituras anteriores. E onde estão disponíveis as leituras? Se você olhar o curso de 4 meses, aqui você pode ver os modelos de linguagem local que eu te ensinei. Nós falamos sobre isso aqui.

Se você olhar aqui, tem aplicação baseada em RAC, tem aplicação baseada em NLP, e aí você pode ir para a frente e aqui você pode ver todos os materiais que eu usei para criar modelos de linguagem grande. Ok. Então, você pode criar uma aplicação Por exemplo, você pode especializar para um restaurante de pizza.

Você pode especializar para outro parâmetro. Por exemplo, se você tem um arquivo relacionado a um carro. Você pode ver o arquivo do carro e criar um arquivo de vetores.

Você pode criar um arquivo de vetores e aí você pode trabalhar com o modelo de linguagem grande. Então, isso varia se você quer criar um arquivo específico ou um arquivo especializado. Nós vamos ver como criar um arquivo específico e um arquivo especializado.

sua arquitetura e vamos continuar. Então, o sistema consiste em três componentes principais. Tem três componentes principais.

Primeiro, o VectorStore. O que é o VectorStore? Basicamente, é um arquivo .py no qual nós vamos carregar o arquivo .csv de revistas de restaurante. Ok.

O arquivo .csv nós vamos carregá-lo. Nós vamos criar embeddings. Você sabe o que é embeddings, porque eu disse isso em detalhes na aula anterior.

E então, nós vamos usar o MxBAI modelo de linguagem grande. Ok. Nós vamos usar esse modelo para fazer o embedding.

Você se lembra que eu já mostrei vários tipos de modelos e eu já mostrei vários tipos de modelos. Então, os modelos de embedding são diferentes. Por exemplo, esse é um modelo que nós vamos usar.

Ele veio um ano atrás e ele está em 335 milhões de parâmetros. Ok. MxBAI Embed Large.

Ok. Além disso, se você escrever lemma aqui, lemma 3.1 também está disponível e 3.2 também está disponível. Então, é totalmente a sua escolha qual você quer usar.

Mas eu recomendo usar lemma 3.2 ou 3 também está disponível, mas 3.2 é o mais recente. Ele veio 9 meses atrás. Então, nós vamos usar ele.

Então, depende da sua escolha, mas o modelo que eu vou usar aqui, eu estou mostrando para você aqui. Ok. Nós temos aqui.

Nós lojamos o arquivo e o guardamos no embedding. E então, nós vamos guardar, por exemplo, os vectores no ChromaDB para retrieval rápido. O ChromaDB é um método através do qual nós guardamos os vectores em um database e com esses vectores nós retrievamos a informação.

Isso é um tipo de retrieval da nossa aplicação. Ou você pode dizer que é um modelo de linguagem grande ou de agente AI. Então, nós vamos para a aplicação principal.

O que vai fazer o main.py? Ela vai implementar a interface de conversão e vai combinar os reviews retrovados que vão vir com isso, ok? Ela vai combinar isso com a nossa query do usuário. Simples. Esses reviews que nós já mostramos, ela vai combinar com a query do usuário.

E isso vai gerar uma resposta usando o modelo lemma 3.2. Ok? Então, isso é o que nós vamos implementar agora. E aí você vai ver que é muito fácil, não é tão difícil. A terceira parte é a parte principal do CSV, onde deve existir dados.

Se não tiver CSV, não vai funcionar. Então, nós vamos criar três fios. Vector.py, main.py e CSV.

Então, esses três fios vão funcionar juntos. O que isso significa? Para criar um agente local, você precisa de três coisas. Qualquer tipo de agente local você pode criar.

Qualquer tipo significa qualquer tipo. Se você quiser retirar o CSV do CSV, retirar o PDF do PDF, se você quiser retirar o DOCS do DOCS, você só precisa adicionar essas bibliotecas ali em cima. No final, as suas embeddings vão ser feitas através desse modelo.

Quando nós do CSV, eles também vão para a embedding. Se nós pegarmos do DOCS, eles também vão para o DOCS, ou se você pegar do PDF, eles também vão para lá. É totalmente a sua decisão qual tipo de modelo você faz.

E, este é o seu assinamento também. Quando você criar sua app, você precisa de três ou quatro tipos de arquivos textos para ver se eles funcionam ou não. No CSV, eu mostro que você tem que usar o PDF, o DOCS e o texto.

A vantagem disso é que você terá uma app completamente feita que você pode mostrar no seu portfólio também. Ok? Até aqui, tudo bem? É fácil, não é? Não está ficando difícil ainda. E eu já tenho o código.

E nós também vamos mudar algumas coisas no código. Ok. Ok.

Se eu ficar um pouco no chat, eu sinto que vocês estão me ouvindo. Certo? Ok. Vamos para o próximo.

Quais são os requisitos? Claro que você vai precisar do Python. Sem o Python, não vai funcionar. Então, você vai precisar do Python.

Então, você vai precisar do Python. você vai precisar Então, do Python. Então, você vai precisar do Python.

do Python. Então, você vai precisar do Python. você vai precisar Então, você vai precisar do Python.

Então, você vai precisar do Python. Então, você vai precisar do Python. Então, você vai precisar você vai precisar de złotins.

Então, você vai precisar do With com 3.2 eu fiz aqui, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada ok, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada, eu não escrevi nada eu não escrevi nada Criar um ambiente de conda Criar o minus n Python 3.11 Criar o minus n Python 3.11 Então vamos fazer ele 11 Certo? Eu tenho dado o requirement.txt Ok Agora eu vou mostrar para vocês duas coisas Primeiro temos eu tenho o vector.py e o outro é o main.py eu vou mostrar os dados para vocês, são dados de restaurantes realistas e eu adaptei o código de um canal do Tim porque ele usou essa metáfora, mas eu fiz algumas coisas de acordo com nossas próprias databases e eu vou mostrar para vocês onde vocês podem ver as mudanças então esse é um arquivo CSV simples você pode pegar qualquer arquivo CSV, qualquer documento e você pode inserir todos esses dados o título é o primeiro, a data é a segunda e o valor é o terceiro, 0-5 ou 1-5 depois você tem as revistas esse é um arquivo de texto, basicamente mas ele está inserido no arquivo CSV nós vamos utilizar isso para criar nossa database se eu for para o vector.py, eu vou mostrar para vocês o que vai acontecer a primeira coisa que vai acontecer é porque nós inserimos o lang.chain.olama nós temos que trazer o olama.embeddings e o olama.chroma depois de trazer essas duas coisas, nós temos que importar o documento também com o lang.chain.code documento para que nós possamos ler qualquer documento nosso e convertê-lo no vector então esse é o nome do arquivo, que nós temos escrito aqui você pode mudar aqui e aqui você pode inserir uma função no app para upload seu arquivo e perguntar ao usuário o que é e depois você pode usar isso como variável através do código dinâmico eu estou te dando dicas, você tem que criar sua própria app e depois o modelo é melhor você usar o código duro ao invés de usar o código soft e selecionar o modelo como nós fizemos na última app vocês se lembram, nós fizemos uma seleção de diferentes modelos no olama mas eu sugeriria que você use o código duro depois aqui, dentro do diretório atual, você vai salvar o database ok? com o nome de lang.chain.db e depois você vai adicionar o documento dentro do diretório atual se ele não estiver disponível ok? significa que se o database estiver disponível, você não vai adicioná-lo se ele não estiver disponível, você vai fazer isso ok então aqui eu fiz algumas mudanças no formato se você observar, se adicionar documento, nós fizemos uma lista vazia e fizemos uma lista vazia de id então nós vamos adicionar a id de cada documento aqui, um por um e depois nós vamos ler isso, baseado no título e dentro disso, todos os dados que nós temos todos os dados disponíveis dentro do documento porque os colunas do nosso documento são algo assim se tem título, tem review, tem rating e tem data são 4 colunas, por isso nós salvamos essas 4 colunas baseado no conteúdo de página e metadata e a id nos dá tudo basicamente, para fazer um database você está convertendo um arquivo CSV no seu formato legítimo ok? e no final, o que você vai ter? você vai guardar os vectores usando o chroma e o nome da coleção, restaurant-review que nós mostramos aqui vamos ver aqui temos o df aqui temos o embedding aqui temos a locação do DB a locação está dentro do diretório atual se você quiser adicionar documento nós temos algumas coisas assim e depois de adicionar o chroma nós vamos salvar o nome da coleção com o nome de restaurant-review e depois nós vamos persistir o diretório o local DB que nós temos agora que está dentro do diretório atual e dentro dele, a coleção será salvada o database tem uma coleção e você pode coletar coleções de diferentes tipos de databases o database é uma categoria major e dentro disso, há coleções e depois você pode adicionar diferentes documentos e essas coleções serão adicionadas ou seja, você está salvando a data base de cada arquivo se tiver outro arquivo, o nome da coleção será mais se tiver outro arquivo, o nome da coleção será mais e o nome da coleção do próximo arquivo será mais e isso é o que nós temos aqui e você tem que ter muito cuidado quando você está adicionando essas coisas porque aqui eu fiz um código duro o título, as revistas e o rating você tem que levar isso para o código dinâmico que quando você ler o arquivo, automaticamente os nomes das colunas serão lidos e depois nós temos aqui nós usamos alguns retrievers aqui basicamente, isso é para o documento que se você adicionar o documento o VectorStore adiciona o documento através do arquivo do documento e ele também salva o documento dentro do arquivo ele não só faz o VectorDatabases ele também salva os documentos e no final, você tem o Retriever que é basicamente uma função do VectorStore e com isso, você pode retirar em base a 5K é basicamente um número random através do qual você pode mudar os valores você tem o VectorDatabase qualidade e quantidade ou você pode dizer que a leitura vai aumentar no algoritmo Retriever se você mudar essas coisas, sua resposta vai mudar mas eu fiz isso de forma simples eu coloquei 5 valores depois, quando nós vamos finalizar o modelo de linguagem grande dentro do RPC então eu vou te dizer como você pode modificar e quais são os parâmetros que você deve ter o que é o objetivo do Vector.py o objetivo principal é que você converte o seu data para o Vector ou seja, transformar em Embeddings e guardar em um VectorDatabase para que quando nós perguntarmos alguma coisa a gente tenha o VectorDatabase Vector e Retriever, desculpe, Curie e Retrieved que são os contextos que nós temos juntos, o nosso sistema vai ser um Rack depois nós temos o Main.py aqui também nós usamos o Ullama para usar templates de chat e prompt como nós fizemos na última vez que nos ajudam automaticamente para refinar o nosso prompt em diferentes tipos de modelos antes de usá-los, quando a Curie aparece ele é refinado no prompt no Agenting Workflow, o objetivo principal é refinar o que você escreveu por exemplo, você escreve no Google imagens e Faisalabad, o Google não vai entender o que você perguntou, mas através do prompt o Google convertiu as palavras em perguntas então esse é basicamente o template de chat e prompt que nós usamos do VectorImportRetriever nós fizemos o Retriever o arquivo do Vector.py e nós usamos essa parte aqui nós escrevemos lemma 3.2 você entendeu? e nós escrevemos o arquivo do Vector no arquivo do Vector.py mxbai-large então no Vector.py nós usamos o arquivo do Embedding e aqui nós usamos o arquivo do Text Generation você pode mudar o nome e o modelo mudará como eu mostrei, eu também tenho lemma 2, eu também posso usar isso então qualquer modelo do Text Generation, você só precisa nomeá-lo e ele vai funcionar e o interessante é que essa biblioteca automaticamente está portando, o que nós fizemos manualmente você se lembra, essa biblioteca automaticamente vai funcionar, se o modelo está funcionando ela vai trazer ele aqui e aqui no template o que nós fizemos, nós dissemos para o nosso modelo de língua maior, você é um experto em responder perguntas sobre um restaurante de pizza nós especificamos que você tem que responder sobre um restaurante de pizza porque o nosso arquivo também é do restaurante de pizza você se lembra, no começo eu disse que você também pode especificar qual é a expertia do modelo você também pode especificá-lo no prompt você também pode escrever um restaurante simples mas agora ele vai treinar sobre um restaurante de pizza e o quanto de conhecimento no lemma 3.2 relacionado à pizza, ele vai nos trazer

**Este arquivo é mais longo que 30 minutos.**

[**Atualize para Ilimitado**](https://turboscribe.ai/pt/subscribed?ref=docx_export_upsell) **em** [**TurboScribe.ai**](https://turboscribe.ai/pt/?ref=docx_export_upsell) **para transcrever arquivos de até 10 horas.**