**Criando AI AGENTS com PYTHON e LANGCHAIN**

**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.**

Você sabe o que é um agente de IA ou um AI Agent? Nesse vídeo você vai aprender a criar um agente de IA utilizando Python e Language Chain e como criar ferramentas para esse agente de IA utilizar essas ferramentas. Mas antes da gente entrar no conteúdo de agentes de IA, se você quer aprender mais sobre Inteligência Artificial, sobre LLM, chatbots, vector databases, se inscreve aqui no canal, deixa o like nesse vídeo, ativa o sininho para receber notificação quando nós lançarmos mais conteúdo para você ficar por dentro daquilo que tem de mais novo, de mais atual, de mais avançado no mercado de Inteligência Artificial. Bom, mas vamos lá.

O primeiro ponto a gente entender é o que são esses AI Agents, o que são esses agentes de IA, o que eles fazem de diferente, o que eles utilizam de diferente e por fim a gente entender e ver como construir ele na prática. Bom, o primeiro ponto é o que nós temos inicialmente, que são os chatbots simples, chatbots diretos ali onde você tem o usuário, o usuário ele manda um input, no caso uma pergunta, uma mensagem, esse input ele vai para um Large Language Model, um modelo de geração de texto e esse modelo de geração de texto gera um texto, gera um output e esse output ele retorna para o usuário. Então esse é um chatbot simples, onde você tem o input do usuário que vai para o modelo de Inteligência Artificial e o output do modelo ele retorna para o usuário.

Mas a gente também tem uma outra categoria que são as Chains ou Correntes. O que são essas Chains e para que elas servem? É quando nós precisamos adicionar passos extras dentro da nossa aplicação, quando nós queremos fazer uma aplicação mais complexa, mais profunda e adicionar algumas algumas coisas. Então, por exemplo, novamente aqui nós temos o nosso usuário, o usuário ele faz um input, certo? Só que nesse caso, ao invés de ir direto para uma LLM, eu vou fazer aqui um script que vai conferir o contexto desse input do usuário, dessa pergunta do usuário.

Esse conferir o contexto pode, por exemplo, ser um modelo BERT que vai classificar qual é o tópico do input do usuário. Se, por exemplo, o meu usuário está querendo que eu gere uma imagem, então eu não vou enviar o input do usuário para uma LLM, eu tenho que enviar o input do usuário para um modelo de geração de imagem. Então, nesse caso, eu vou checar o contexto, certo? Eu vou enviar, caso seja um contexto de geração de imagem, eu vou enviar para o meu modelo de Stable Diffusion, certo? E aí o meu modelo de Stable Diffusion é aquele que gera imagens, né? Tipo o Dolly, por exemplo, e aí o meu modelo ele vai gerar um output, vai gerar uma imagem, e essa imagem ela vai retornar para o usuário.

Só que, caso o meu usuário esteja fazendo uma pergunta, que seja necessário gerar texto, então eu mando esse input do usuário para uma LLM. Esse LLM ele vai gerar uma resposta, e eu posso fazer um pós-processamento nessa resposta da minha LLM, né? Então, digamos que ele gerou uma resposta ali, e aí eu faço uma avaliação para verificar se a resposta da minha LLM está muito tóxica, se está legível, se está coerente com a pergunta do usuário, com o input do usuário, então eu faço vários tipos de avaliação, certo? E após essa avaliação, eu posso fazer um pós-processamento, eu posso corrigir alguma coisa, eu posso cortar a resposta, caso a minha LLM tenha alucinado, ou até mesmo após essa avaliação, eu posso pedir para a LLM fazer uma nova resposta, certo? Dependendo da avaliação, eu posso ter um loop aqui, certo? Até que essa resposta ela seja aprovada dentro da minha avaliação, certo? E aí ela vai fazer a avaliação, vai fazer esse pós-processamento, e do pós-processamento ele vai para o output do usuário. E aqui é uma chain, é uma corrente, é uma sequência de eventos, certo? Funciona mais ou menos como um grafo, tá? E aí você vai ter essa sequência de eventos que ela vai estar bem estruturada, bem fixa, tá? O input sempre vai passar por aqui, ele vai ter esses dois caminhos, ele vai passar por essa avaliação, vai passar pelo pós-processamento e vai retornar para o usuário.

E caso você queira introduzir um novo caminho, você vai ter que fazer uma terceira opção aqui, e aí essa terceira opção ela vai ter o seu próprio caminho, certo? Todas as vezes que você quiser inserir alguma coisa, você vai ter que aumentar a complexidade dessa corrente. E aí que entram os AI Agents, tá? Os AI Agents vêm para facilitar e permitir com que você traga maiores ferramentas, maiores utilidades para seu chatbot, para suas LLMs. E como é que funciona os AI Agents? Basicamente você vai ter novamente o usuário, né? Aqui eu estou colocando o usuário, mas pode pegar esse input de fazendo um web scrapping, pode pegar esse input de um banco de dados, enfim.

Você pode fazer, você pode pegar esse input de vários outros locais, não de um usuário, mas aqui eu estou usando o usuário só como um exemplo, né? Esse usuário, ele vai mandar o input dele para um agente. Esse agente nada mais é do que um LLM, é um modelo de linguagem. O agente, esse agente aqui, ele vai possuir uma caixa de ferramentas, né? Que a gente vai chamar de um toolkit, né? Um kit de ferramentas.

Ele vai ter várias e várias ferramentas dentro desse toolkit. E aí, o que vai acontecer? Cada uma dessas ferramentas vai ter uma descrição, certo? Cada ferramenta vai ter uma descrição. O que vai acontecer? O agente vai receber esse input e baseado na descrição de cada uma das ferramentas e baseado no input, o agente, ele vai escolher qual que é a melhor ferramenta para ele utilizar para aquele input específico, certo? Então, trazendo isso um pouquinho mais para a prática, digamos que eu tenho aqui uma ferramenta X, que é uma ferramenta de multiplicar.

O X, ele multiplica, o Y, ele soma, o Z, ele eleva ao quadrado. Então, o usuário, ele manda uma pergunta falando quanto é 2 mais 2. O agente, baseado na descrição de cada uma das ferramentas, vai escolher a melhor ferramenta que se adequa à pergunta do usuário. Então, se o usuário quer saber quanto é 2 mais 2, ele vai pegar, opa, essa daqui é uma ferramenta de somar, porque está na descrição dessa ferramenta que ela é uma ferramenta para somar.

Então, eu vou chamar essa função, certo? E isso daqui, esse X aqui, nada mais vai ser do que um def, pensando em Python, já um def edge, que eu vou ter X e Y, e eu vou retornar X mais Y, certo? Isso daqui vai ser a ferramenta. Isso daqui vai ser a ferramenta. E o que vai acontecer? O input do usuário, ele vai ter lá, quero que some 2 mais 2. Ele não vai ter só 2 mais 2. Ele vai ter, quero que some 2 mais 2. O que o agente vai fazer? O que essa LLM vai fazer? Ela vai pegar o 2, vai passar para o X, vai pegar o outro 2, vai passar para o Y. Então, a LLM, segundo a descrição que você vai fazer aqui, a gente vai utilizar uma docstring para fazer a descrição.

E aí, baseado nessa descrição, ela vai passar para essa função edge aqui, quais são os parâmetros que ela vai utilizar, certo? Então, é isso que o agente vai fazer. E ele vai fazer isso para de adição, para de multiplicação, para de elevar ao quadrado. Vai fazer isso para muitas outras coisas, certo? Imagina a infinidade de coisas que você pode fazer utilizando isso aqui.

E hoje, a gente vai fazer duas coisas. Nós vamos trabalhar com duas coisas. A primeira, a gente vai fazer uma ferramenta que vai ajudar, vai nos ajudar a entender documentação, certo? Uma ferramenta que nós vamos enviar o URL de uma documentação e o modelo vai nos explicar o que eu posso fazer com aquela documentação.

E o segundo, ele vai, é uma ferramenta que vai fazer o lint, vai formatar o seu código, vai pegar um arquivo de código Python e vai formatar aquele arquivo utilizando o black. Então, basicamente, a gente vai fazer um agente que nos ajuda a melhorar o nosso código, certo? Nos ajudando com a documentação e nos ajudando com a formatação do nosso código. Então, vamos lá.

Vamos começar a parte do código. Então, vamos lá. Abriu meu VSCode aqui, né? Criei um projeto do zero e para inicializar esse projeto, eu vou criar um projeto poetry, tá? A gente vai utilizar o poetry aqui para gerenciar as nossas dependências, né? Então, eu vou criar aqui package name aiagents, a versão 0.1.0, blá blá blá, beleza, licença.

Eu vou colocar o compatível com Python 3.12.4, que é a versão que eu possuo neste momento. Se eu desejo adicionar as dependências, vou dizer que sim e a gente vai adicionar algumas dependências aqui. A primeira é, obviamente, o lang chain que nós vamos utilizar, né? Então, lang chain, a última versão.

Nós vamos utilizar também o requests. Nós vamos utilizar o black para fazer a formatação, né? Do nosso código. E nós vamos utilizar o lang chain OpenAI.

Além disso, nós também para fazer, como nós vamos fazer a ferramenta que pega uma URL e extrai daquela URL da documentação, para fazer o scrapping da documentação, nós vamos precisar do beautifulsoup4, né? Beautifulsoup4, beleza. Beautifulsoup4, para a gente fazer o scrapping. E, além disso, nós vamos precisar do poetryplugin.yaml. Perfeito.

Terminamos aqui as de desenvolvimento, eu vou dizer que não e vou confirmar a geração do nosso projeto, certo? Vou dar aqui um poetry install e pronto, temos instalado aqui. Esse erro aqui não é nada demais, tá? Temos instalado aqui, então, as nossas dependências. Nós temos o Python, lang chain, requests, black, lang chain OpenAI, beautifulsoup4 e o plugin do .env do poetry, tá? E é o que nós vamos fazer.

Bom, vou criar aqui, então, um arquivo que chama agents.py, né? E, além disso, eu vou criar um .env, tá? Nesse .env, eu vou colocar a minha chave do OpenAI. Você tem que entrar aí no site do OpenAI e criar a sua chave, ou você também pode utilizar um modelo local, um modelo open source do Hugging Face, por exemplo, para ser o seu agente, tá? Você não precisa utilizar, necessariamente, o OpenAI. Eu vou utilizar o OpenAI por praticidade aqui mesmo, né? Porque é muito mais tranquilo de trabalhar localmente utilizando a API do OpenAI.

Bom, feito isso, nós vamos precisar, então, qual que é o nosso objetivo aqui? Primeiro, é criar uma ferramenta para pesquisar documentações criar uma ferramenta para fazer, formatar, né? Formatar o nosso código e, além disso, também criar o nosso agente que vai utilizar essas ferramentas, tá? Esses são os nossos três objetivos aqui. Então, o primeiro ponto é, nós temos que criar a nossa ferramenta que vai fazer esse scrapping, né? Esse scrapping nas documentações. Então, eu já peguei aqui, eu já criei um código de scrapper.

Não é o objetivo aqui do vídeo fazer um conteúdo sobre scrapping, então, eu vou só, realmente, copiar aqui. Mas, basicamente, o que esse código está fazendo, né? Ele faz um request para uma URL que ele recebe, ele verifica se o que retornou foi o código 200 e, baseado nisso, ele cria aqui um objeto do Beautiful Soup e, aí, ele faz um decompose aqui, né? Para tirar tudo o que é de script, tudo o que é de style e ele me retorna só o texto, né? Além disso, ele faz alguns strips aqui para ficar mais organizado no Python o texto, né? Ele me retorna aqui. Caso o request, ele não consiga acessar o conteúdo da URL, ele me retorna aqui, eu ao acessar a página, certo? É, basicamente, esse o código.

Eu só, realmente, copiei e colei aqui, para a gente não ficar perdendo tempo no scrapper. Feito isso, a primeira coisa que a gente tem que fazer é criar a nossa documentação, vou chamar de Documentation Tool, certo? E essa Documentation Tool, ela vai receber o URL e ela vai receber uma pergunta. Lembra que eu falei que o agente, ele consegue definir quais são os parâmetros de cada, de cada uma das funções? Então, a gente consegue separar quais são, o que é aquilo que a nossa ferramenta vai receber e o agente vai passar como parâmetro para a gente, dentro da, para a gente utilizar dentro da nossa ferramenta, tá? Bom, e aí, aqui a gente chega em um dos principais pontos da ferramenta, que é a descrição dessa ferramenta, certo? Então, nós precisamos descrever, descrever direitinho o que que essa ferramenta recebe e qual que é o output dessa ferramenta.

E para ajudar a gente ainda mais, eu acabei esquecendo aqui, a gente tem que passar qual que é o tipo, qual que é o tipo dessa, de cada uma dos nossos parâmetros, né? E nós temos que passar também o retorno da nossa, da nossa função, da nossa ferramenta. E aí, voltando aqui para a descrição, aqui eu vou descrever, então aqui eu fiz uma descrição bem, bem completa, acredito eu, né? Nós vamos ver se ela realmente funciona quando o agente estiver nos passando essa, os parâmetros. Mas, basicamente, o que eu falei foi, essa ferramenta, ela recebe como input a URL da documentação e a pergunta sobre a documentação que o usuário quer que seja respondida, tá? Basicamente, essa é a descrição da ferramenta que nós vamos estar utilizando.

Bom, agora que a gente já tem estruturado a descrição, a gente tem que, então, enviar, primeiro, primeiro, pegar o texto da documentação e, segundo, fazer a pergunta para uma LLM sobre a documentação, utilizando a pergunta que o usuário deseja, né? E a documentação que nós vamos utilizar. Então, o primeiro ponto é pegar o texto, né? Então, a gente vai fazer aqui, from scraper, import, é, qual que é o nome que eu utilizei aqui? GetText. GetTextFromURL.

E aí, a gente vai, então, utilizar aqui como contexto o texto da documentação que vai ser extraída. Então, eu vou fazer GetTextFromURL e vou passar a URL. Tendo esse contexto, a gente consegue, então, criar uma mensagem que vai ser passada para uma LLM e, para isso, a gente vai precisar estruturar a mensagem segundo o OpenAI, certo? Então, para isso, eu vou precisar de importar a human message e o system message, onde eu vou colocar como system message um prompt, um prompt simples para utilizar na LLM e, como human message, como mensagem do usuário, do humano, a pergunta que o nosso usuário fez para a gente, certo? Então, Show, o que eu fiz aqui? Primeiro, eu escrevi aqui um prompt simples, né? Você é um assistente de programação que deve explicar documentação de bibliotecas de programação para usuários as simple as possible, ou tão simples, tão simples quanto você conseguir.

E, depois, a human message, eu passo aqui como content, como conteúdo dessa mensagem do usuário, a documentação, o contexto, que é o texto da URL aqui e a pergunta que o meu usuário, né? Que o meu usuário fez para o nosso agente. Feito isso, agora a gente tem que enviar essa mensagem, enviar esse prompt para nossa LLM, para assim nós conseguimos essa resposta baseada na nossa documentação. E, para isso, eu vou criar aqui uma outra função, né? Para ficar um pouquinho mais organizado, chamada GetResponseFromOpenAI, onde eu vou receber uma mensagem e eu vou chamar aqui uma LLM, né? Deixa eu importar aqui o ChatOpenAI.

Então, importei aqui o ChatOpenAI e a gente vai utilizar ele para, na verdade, isso daqui é o Model e eu vou passar, na verdade, ModelName e eu vou passar aqui o GPT 3.5 Turbo, tá? Nós vamos estar utilizando aqui o GPT 3.5 Turbo, que é um modelo simples, tranquilo, mais barato, né? Do que o, obviamente, do que o GPT 4 e o 4O. Então, eu vou chamar aqui a LLM.invoke com a nossa mensagem, certo? E vou retornar essa response. Então, feito isso aqui, eu só tenho que chamar essa funçãozinha aqui e passar a messages e retornar essa response.

Então, recapitulando o que nós fizemos aqui, a nossa DocumentationTool vai receber uma URL, vai receber uma pergunta e vai extrair dessa URL a documentação, que eu chamei aqui de Contexto, vai passar essa documentação como um contexto aqui para a nossa LLM, juntamente com a pergunta que o usuário fez. Então, por exemplo, ele vai passar aqui a documentação do LangChain sobre AI Agents e vai perguntar como eu posso construir um AI Agent utilizando o LangChain e aí o nosso bot tem que responder utilizando o conhecimento que a documentação forneceu, certo? Qual que é a vantagem de fazer isso? Muitos desses modelos, eles não têm informações atualizadas sobre bibliotecas. Então, eles são treinados com dados até 2021, dados até 2022.

Então, bibliotecas mais recentes ou atualizações mais recentes, esses modelos não possuem. Então, utilizando essa forma aqui de trabalhar com documentação, você consegue fornecer, utilizando uma ferramenta do AI Agent, você consegue fornecer novas informações para o seu chatbot, certo? Então, essa é a primeira ferramenta, essa é a primeira ferramenta que nós temos a criar. E é simples, é só isso.

Eu criei uma ferramenta aqui. Óbvio, a gente vai adicionar algumas outras coisas, mas eu só preciso fazer isso aqui para criar uma ferramenta. Ela é uma função com uma descrição, utilizando docstring.

Acabou, é isso, certo? Agora, a gente vai criar a nossa segunda ferramenta, que é o nosso formatador, nosso formatador de código. Eu vou escrever aqui, vou criar uma outra função, blackformatter, que ele vai receber um path, esse path é uma string, e ele vai retornar também uma string. E, basicamente, o que ele vai fazer, deixa eu importar aqui, import os, basicamente, o que ele vai fazer é executar aqui, opa, não é esse, é o system, ele vai executar um comando, um comando Python no terminal, ele vai fazer poetry run blackpath, e aí eu vou utilizar aqui esse path, e aí eu vou fazer um try.

Então, fiz aqui um try para tentar executar o poetry run black, tentar formatar um arquivo Python que está nesse path aqui. Se ele conseguir, ele vai retornar dom, se ele não conseguir, ele vai retornar did not work, não funcionou. E aí, a gente precisa adicionar uma descrição para a nossa ferramenta.

Então, a descrição que eu utilizei aqui, mais ou menos no padrão da primeira, essa ferramenta recebe como input um caminho do sistema de arquivos para um script Python, para um arquivo que é um script Python, e executa o blackformatter, o formatador black, para formatar esse arquivo de código Python, na verdade, o código Python deste arquivo. Basicamente, é isso. Então, defini aqui a minha segunda ferramenta, vou chamar aqui, continuar no padrão, vou chamar de blackformatter2.

Beleza? Então, defini aqui as minhas duas, as minhas duas ferramentas. O que eu preciso fazer a partir de agora? É aqui que entra a questão da criação dos agentes. Bom, o primeiro ponto é que nós precisamos importar.

Assim, nós definimos a função, as duas funções de ferramenta, mas nós precisamos transformar isso daqui em uma ferramenta do lang chain. E para isso, a gente vai utilizar, unicamente, um decorator, um decorator que o lang chain criou. Então, basicamente, para a gente transformar essa função aqui em uma ferramenta, eu só tenho que utilizar o decorator aqui do lang chain.

Então, arroba tool e pronto. Criei uma ferramenta. É isso.

Está criada uma ferramenta. Só que eu preciso criar a minha caixa de ferramenta, o meu kit de ferramentas, que é um toolkit. E esse toolkit nada mais é do que uma lista com as ferramentas que o meu agente possuirá.

Então, eu criei aqui o meu toolkit. Eu, simplesmente, criei uma lista com o nome das ferramentas que o meu agente poderá utilizar. E após isso, então, eu crio uma LLM.

Novamente, eu vou estar utilizando o chat OpenAI. Vou chamar aqui o ModelName. Vou utilizar o mesmo modelo, exatamente o mesmo modelo, ChatGPT 3.5 Turbo.

Vou utilizar a temperatura em zero. O porquê que eu vou utilizar a temperatura em zero? Se você não sabe, a temperatura, quanto mais alta ela é, mais criativo o modelo fica com as suas respostas. Quanto menor a temperatura é, menos criativo, menos variação das palavras esse modelo vai ter.

Isso é mais fácil para vocês conseguirem reproduzir o mesmo exemplo, mas também é melhor para que o nosso agente não crie coisas muito criativas para passar para as ferramentas. Então, eu acredito que é melhor utilizar a temperatura baixa quando você está lidando com o agente que vai utilizar as ferramentas. Depois disso, a gente vai criar o nosso prompt.

Eu vou copiar aqui um prompt que eu já criei aqui para o nosso agente. E para isso, a gente precisa importar mais algumas coisas. Então, eu importei aqui o ChatPromptTemplate, que é um template de chat, de chatbot, como o próprio nome já fala.

E o MessagesPlaceHolder aqui, só para a gente criar um chat history e, enfim, a gente ter o nosso prompt organizado dentro aqui do nosso agente. E aí, o nosso prompt é basicamente o seguinte, você é um assistente de programação, use suas ferramentas para responder as questões. Se você não tem a ferramenta para responder a questão, fale isso, fale que você não possui a ferramenta.

Retorne apenas as respostas. Isso, para o nosso chatbot, ele fica bem restrito, ele não fica muito criativo, não alucinar, não inventar muita coisa, mas ele fica muito restrito, bem restrito ao conteúdo da nossa ferramenta e ao uso das ferramentas. E aí, após isso, após criado o nosso prompt, a gente vai, enfim, criar o nosso agente.

E para criar o nosso agente, nós também iremos importar duas coisinhas aqui, também do lang chain. E olha só, a gente está usando o lang chain aqui para tudo, mas nós vamos importar aqui do modo agent, a gente vai importar o agent executor, que é para executar o agente, rodar o agente, e essa funçãozinha aqui que a gente vai criar um agente, um OpenAIToolsAgent. O que é esse OpenAIToolsAgent? Se você for olhar a documentação do lang chain sobre agents, AI agents, você vai ver que ele possui vários, uns sete, se eu não me engano, uns sete tipos de agentes diferentes.

E por que que eu estou utilizando esse agente específico aqui? Porque no momento, neste exato momento que eu estou gravando este vídeo, este é o melhor agente para se utilizar. Ele foi treinado especificamente utilizando funções, onde ele pegava uma pergunta e passava, distribuía as características, os elementos ali daquela pergunta em parâmetros. E foi treinado para selecionar a melhor função para uma determinada task, uma determinada tarefa.

Então, neste exato momento, este agente aqui é o melhor, o que tem o melhor desempenho para selecionar a melhor ferramenta baseado na pergunta do usuário, no contexto do usuário, beleza? Então, fica ligado aí, dá uma pesquisada lá na documentação de AI Agents, lá do lang chain, que você vai ver que você pode utilizar outros agentes também, que eles são mais específicos para outras coisas. Mas se o seu foco é utilizar ferramentas, utilizar essas funções, esse agente aqui é o melhor, com absoluta certeza. Então, vamos lá.

A gente agora tem que criar o nosso agente de AI. Então, a gente vai vir aqui em Create Open AI Agent, e a gente tem que passar três coisas para ele. Três coisas.

Então, a primeira coisa é a nossa LLM, a segunda coisa é o nosso Toolkit e a terceira coisa é o nosso Prompt. E pronto, o nosso AI Agents está criado. Eu só preciso passar isso, a LLM, que ele vai utilizar o Toolkit e o Prompt.

E aí você pode estar com uma dúvida aí. Você pode estar com uma dúvida que é, por que eu estou utilizando um outro, eu estou utilizando duas LLMs diferentes, né? Aqui eu estou carregando uma LLM e aqui eu estou carregando uma outra LLM. Bom, uma ferramenta

**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.**