

Roteiro do Curso: Construindo um Chatbot com Gemini e Python

Abertura

“Bom, pessoal, nós somos Raimundo, Carlos Eduardo e Ulisses, e a ideia aqui é construir com vocês um chatbot funcional usando a linguagem de programação Python e o Gemini.

Alguém aqui sabe o que é linguagem de programação?

Explicação simples:

Linguagem de programação é como se fosse um idioma que usamos para “conversar” com o computador e ensinar ele a fazer algo. No nosso caso, vamos ensinar o computador a responder mensagens como um assistente virtual, ou seja, um chatbot.

Vocês sabem o que é um chat bot?

Explicação simples:

Um chatbot é um programa que simula uma conversa com uma pessoa, respondendo perguntas automaticamente.

Passo 1 – Conectando nosso chatbot ao "cérebro"

Para que o nosso chatbot funcione, ele precisa ser inteligente, certo? Quem vai ser o "cérebro" dele é o Gemini, que é um sistema de inteligência artificial criado pela Google. Ele é capaz de entender perguntas, gerar respostas e conversar de forma natural.

Mas, para usar o Gemini, precisamos de uma chave de acesso secreta chamada API KEY. É como se fosse a “senha” para deixar nosso código conversar com o cérebro.

Obter a Chave Mágica (API Key):

1. Todo mundo aqui tem uma conta no Gmail? Se não tiver, é só criar rapidinho.
2. Acesse o Google IA studio (link: https://aistudio.google.com/prompts/new_chat)
3. Faça login com o Gmail.
4. Lá no canto direito, clique em GET API Key.
5. Copie a chave gerada e cole no bloco de notas. Vamos usar ela no código.

✓ Passo 2 – Criando nosso ambiente de trabalho

Agora que temos o cérebro e a chave, precisamos de um lugar para escrever as instruções, ou seja, o nosso código.

Esse lugar é o Google Colab – um site que funciona como um caderno online para escrever código Python.

Acesse:

- Vamos acessar o Google colab(link: <https://colab.google/>)
- Clique em “Novo notebook”
- Renomeie o nome do arquivo para: **chatbot_gemini**

✓ Passo 3 – Escrevendo o código do nosso chatbot


Agora chegou a parte mais divertida: programar o chatbot! Vamos dar uma espécie de "receita de bolo" pro computador seguir. A gente diz como ele deve funcionar e como responder.

CÓDIGO

INSTALAR O PACOTE NECESSÁRIO

Primeiro, vamos instalar um pacote chamado 'google-genai'.

Esse pacote é como uma caixa de ferramentas que nos permite conversar com a IA do Google (o Gemini).

```
 # Instala o pacote
!pip install google-genai
```

IMPORTAR BIBLIOTECAS

Aqui, iremos importar duas ferramentas:

Iremos importar a OS que Serve para mexer nas configurações do sistema, como guardar nossa chave da API.

```
[ ] # Importa bibliotecas
import os
```

E também iremos importar a Genai, que é a ferramenta principal que nos conecta com o Gemini. No caso, estamos importando a biblioteca genai que faz parte do pacote google.

```
from google import genai
```

CONFIGURAR A CHAVE DE ACESSO (API KEY)

Agora vamos dizer para o computador qual é a nossa chave secreta para acessar o Gemini.

Nessa parte entre aspas simples Copie e cole a chave da API

```
[ ] # Define a chave da API como variável de ambiente  
os.environ['GEMINI_API_KEY'] = ''
```

CRIAR O CLIENTE

Agora vamos dar um comando para criar o "cliente", que é a conexão com o cérebro do chatbot (o Gemini).

```
[ ] # Cria o cliente  
client = genai.Client()
```

DEFINIR O COMPORTAMENTO DO CHATBOT

Agora, vamos criar um histórico com instruções para o nosso chatbot saber como se comportar.

```
# Inicia o histórico  
history = ["Seu nome é Mundinho Perneteta. Você é um assistente de IA",  
           "Você sempre tenta ser engraçado",  
           "Apresente-se no início da conversa",  
           "Responda as perguntas, mas sempre tente falar o menos possível"]
```

Agora vamos dar um print(impressão na tela) de uma mensagem de boas-vindas quando o programa começar.

```
print("Olá! Sou seu chatbot mundinho perneteta. Digite 'Sair' quando quiser encerrar.\n")
```

Obs.: Esse \n no final é para o computador dar uma quebra de linha, ou seja, para pular uma linha

Agora vamos criar uma estrutura de repetição e fazer o computador conversar com o usuário em um loop infinito.

Só que, iremos adicionar um parâmetro, Enquanto o usuário não digitar "sair", o chatbot continua respondendo.

#Iniciamos com while true

```
# Loop principal
while True:
```

#Na linha seguinte, adicionaremos uma variável (pergunta) com uma entrada para (input) pro usuário fazer uma pergunta

```
pergunta = input("Pergunte alguma coisa: ")
```

Agora iremos adicionar um parametro. Se o usuário digitar "sair", o chatbot se despede e para.

```
if pergunta.strip().lower() == "sair":
    print("Encerrando o chatbot. Até logo!")
    break
```

Nessa linha do IF, esse strip e lower juntos vão servir para verificar se o usuário digitou "sair", mesmo que ele use letras maiúsculas ou espaços extras.

O **STRIP** remove espaços no começo e no fim do texto.

O **LOWER** transforma todas as letras em minúsculas.

O **BERAK** quebra o loop

Agora, iremos preparar a pergunta do usuário para ser enviada ao modelo Gemini, em um formato que ele entende.

#O Gemini precisa que as mensagens da conversa estejam organizadas com:

- quem está falando (**role**)
- e o que está sendo dito (**parts(é uma lista de partes que compõem a mensagem)**).

```
# Cria conteúdo do tipo Content com role='user'
user_input = genai.types.Content(
    role="user",
    parts=[genai.types.Part(text=pergunta)]
)
```

#**genai.types.Content**: é um comando que cria uma nova mensagem no formato padrão do Gemini.

#Podemos imaginar que está dizendo:

"Ei, Gemini! Aqui está uma nova fala da conversa."

#**genai.types.Part(text=pergunta)**:

Aqui estamos dizendo:

"A parte da mensagem é esse texto que o usuário acabou de digitar."

Agora iremos adicionar a pergunta ao histórico

```
# Adiciona ao histórico
history.append(user_input)
```

Agora vamos dar um comando para enviar tudo para o modelo do Gemini responder

```
# Chama o modelo com o histórico
response = client.models.generate_content(
```

Adicionamos **"model="gemini-2.5-flash"**, que serve para escolher o modelo rápido e eficiente

```
model="gemini-2.5-flash",
```

#Adicionamos **"contents=history"** que serve para enviar toda a conversa e manter o contexto

```
contents=history
)
```

Agora iremos dar um comando para pegar a resposta e mostrar ela na tela

Adicionaremos um comando que pega a resposta do Gemini (IA) e transforma em texto

```
# Obtém a resposta e imprime
resposta_texto = response.candidates[0].content.parts[0].text
```

#Agora adicionaremos um print para mostrar a resposta do chatbot na tela

```
print("Chatbot:", resposta_texto)
```

#Por fim, finalizamos com um comando que adiciona a resposta da IA ao histórico para manter o contexto da conversa, ou seja, que o chatbot sempre lembre do que você perguntou anteriormente

```
# Adiciona resposta ao histórico
history.append(response.candidates[0].content)
```

✓ **Recapitulando os passos:**

1. 📌 **Conseguimos nossa chave secreta (API KEY) no site do Google IA Studio.**
2. 📄 **Criamos um ambiente no Google Colab para escrever o código.**
3. 🧠 **Escrevemos o código passo a passo para montar o nosso chatbot.**
4. 🗣️ **Interagimos com o chatbot pelo terminal.**