**Bem-vindo**

**Desenvolvedor!!**

A seguir você encontrará uma breve explicação do código e de cada função que foi desenvolvida, possíveis melhorias e falhas já encontradas!

Sumário

[Funcionalidades do Sistema 1](#_Toc166323667)

[Futuros desenvolvedores 2](#_Toc166323668)

[Falhas Conhecidas: 2](#_Toc166323669)

[Possíveis Melhorias Futuras: 3](#_Toc166323670)

[Contribuindo 3](#_Toc166323671)

[Requisitos 3](#_Toc166323672)

[Install 3](#_Toc166323673)

[Conversar com IA 4](#_Toc166323674)

[Assunto 4](#_Toc166323675)

[Dicas de resposta 5](#_Toc166323676)

[Primeira Frase utilizada: 5](#_Toc166323677)

[JSON Convert 6](#_Toc166323678)

[Model Tools 6](#_Toc166323679)

[Funções e Métodos 7](#_Toc166323680)

[Função chat\_principal 7](#_Toc166323681)

[Função `texto\_padrao\_dicas(texto, idioma="inglês")`: 8](#_Toc166323682)

[Função `possiveis\_respostas(texto)`: 9](#_Toc166323683)

[Função `texto\_padrao\_JSON\_dicas(texto)`: 9](#_Toc166323684)

[Função `modelo\_dicas\_schema(texto)`: 10](#_Toc166323685)

[Função `modelo\_dica\_regex(text)`: 10](#_Toc166323686)

[Função `unica\_pergunta(text)`: 11](#_Toc166323687)

[Função `validaConversa(texto)`: 11](#_Toc166323688)

# Funcionalidades do Sistema

**# Funcionalidades do Sistema**

Com esse código, é possível criar um sistema interativo para ajudar os usuários a praticarem conversação em um idioma estrangeiro de sua escolha.

No geral, esse sistema fornece uma maneira interativa e eficaz para os usuários praticarem conversação em um idioma estrangeiro, oferecendo suporte com dicas, traduções e validações de idioma ao longo do processo. Ele pode ser uma ferramenta útil para estudantes de idiomas que desejam melhorar suas habilidades de conversação de forma autônoma.

Aqui estão algumas das funcionalidades e utilidades desse sistema:

**\*\*Escolha do Idioma\*\***: O usuário pode escolher qual idioma estrangeiro deseja estudar, selecionando entre as opções disponíveis, como inglês, francês, alemão, espanhol e italiano.

**\*\*Treinamento de Conversação\*\***: O sistema oferece a opção de treinar conversação no idioma selecionado. Ele gera respostas automáticas com base nas interações do usuário, ajudando-o a praticar a conversação em situações simuladas.

**\*\*Geração de Opções de Resposta\*\***: Quando o usuário solicita uma dica durante a conversação, o sistema pode gerar várias opções de resposta para uma determinada frase. Isso ajuda o usuário a expandir seu vocabulário e a compreender diferentes maneiras de expressar uma ideia no idioma estrangeiro.

**\*\*Tradução de Frases\*\***: O sistema é capaz de fornecer traduções de frases do idioma estrangeiro para o português. Isso ajuda o usuário a entender o significado das frases em sua língua nativa.

# Futuros desenvolvedores

\*\*Documentação do Programa: Prática de Conversação em Idiomas\*\*

Este programa oferece uma plataforma interativa para praticar conversação em diferentes idiomas, proporcionando uma experiência de aprendizado dinâmica e envolvente. No entanto, há algumas falhas e possíveis melhorias que podem ser consideradas para futuras versões:

## Falhas Conhecidas:

1. **Interação Limitada:** Atualmente, a interação do usuário é limitada a responder e receber dicas ou traduções. Adicionar recursos adicionais de interação, como perguntas abertas ou jogos de palavras, pode aumentar o engajamento do usuário.
2. **Validação de Entrada:** A validação da entrada do usuário é mínima, o que pode levar a respostas inválidas ou fora de contexto. Implementar uma validação mais robusta da entrada do usuário pode melhorar a qualidade das interações.
3. **Variedade de Assuntos:** O programa atualmente oferece apenas um conjunto limitado de assuntos para conversação. Expandir a variedade de tópicos de conversação pode tornar a experiência mais abrangente e interessante para os usuários.
4. **Feedback Personalizado:** O programa fornece dicas genéricas, mas não oferece feedback personalizado com base no desempenho do usuário. Incorporar algoritmos de aprendizado de máquina para fornecer feedback adaptativo pode melhorar a eficácia do programa.
5. **Limitações de Idiomas:** O programa atualmente oferece suporte a um conjunto limitado de idiomas. Expandir essa lista pode melhorar a acessibilidade para usuários de diversos backgrounds linguísticos.
6. **Respostas Genéricas:** Algumas respostas geradas automaticamente podem ser genéricas ou inadequadas para certos contextos de conversação. Aprimorar os algoritmos de geração de respostas pode melhorar a qualidade da interação.
7. **Experiência do Usuário:** A interface do usuário pode ser aprimorada para fornecer uma experiência mais intuitiva e amigável, facilitando a navegação e o uso do programa.

## Possíveis Melhorias Futuras:

1. **Integração de Reconhecimento de Voz:** Adicionar suporte para reconhecimento de voz pode permitir que os usuários pratiquem a pronúncia correta das palavras.
2. **Funcionalidade de Compartilhamento:** Implementar recursos de compartilhamento para que os usuários possam compartilhar suas experiências de aprendizado em redes sociais ou com outros usuários.
3. **Personalização de Experiência:** Permitir que os usuários personalizem sua experiência de aprendizado, como ajustar a dificuldade ou escolher tópicos específicos de interesse e adaptem a experiência do programa às suas necessidades e preferências individuais.
4. **Aprimoramento da Interface do Usuário:** Melhorar a interface do usuário para tornar a navegação mais intuitiva e agradável.
5. **Expansão de Idiomas:** Adicionar suporte para uma gama mais ampla de idiomas para atender a uma variedade maior de usuários.
6. **Inteligência Artificial Avançada:** Implementar algoritmos de IA mais avançados para melhorar a qualidade das respostas e a relevância das dicas fornecidas durante a conversa.

## Contribuindo

Este programa é de código aberto e recebe contribuições da comunidade de desenvolvedores interessados em melhorar sua funcionalidade e usabilidade. Se você tem sugestões de melhorias, correções de bugs ou deseja contribuir com novos recursos, sinta-se à vontade para enviar pull requests para nosso repositório no GitHub.

Obrigado por escolher nosso Programa de Prática de Conversação em Idiomas. Estamos ansiosos para ver como você irá ajudar a tornar esta ferramenta ainda mais útil e eficaz para estudantes de idiomas em todo o mundo!

# Requisitos

## Install

Para utilizar o projeto foi necessário instalar as seguintes lib.

pip install ipython

pip install -q -U google-generativeai

pip install -q -U google-generativelanguage

# Conversar com IA

Essa conversa será feita em cima de um assunto que o usuário escolher e contará com dicas para responder a IA, e correção das respostas escritas pelo usuário.

“Vamos iniciar uma conversa em inglês, nós acabamos de nos conhecer, vamos começar na apresentação, o assunto é viagens, meu nível em inglês é avançado, a conversa deverá seguir esses parâmetros.”

Para essa conversação foi necessário utilizar o modelo “Gemini 1.5 pro” ao invés do “Gemini 1.0 Pro” pelo fato do modelo 1.5 se sair melhor na simulação de uma conversa real, enquanto o modelo 1.0 realizava a conversa inteira sozinho e não dava o tempo do usuário digitar uma resposta.

Gemini 1.5

Tela de celular

Descrição gerada automaticamente

Foi solicitado para realizar uma conversa e o modelo compreendeu o que era esperado.

## Assunto

Pedir para a IA disponibilizar 5 assuntos para uma conversa, mostrar ao usuário e pedir para ele escolher um, a IA começará o assunto.

Frase utilizada:

Me mostre 5 assuntos para conversar com uma pessoa que tem o nível básico na língua francesa, mostre uma breve descrição do que poderá ser conversado nesse assunto: Assunto 1: texto;   
Descricao 1: texto;   
  
Assunto 2: texto;   
Descricao 2: texto;

. . .

Substituir a língua pela que o usuário esta utilizando

Resposta esperada:

Texto

Descrição gerada automaticamente

## Dicas de resposta

A dica de resposta foi dividida em duas partes

Uma em que responde a pergunta e outra que transforma o conteúdo em JSON.

### Primeira Frase utilizada:

Me de 3 opções de resposta para responder a seguinte frase:

“frase”

A resposta escreva em frances e a tradução em português.

Siga o exemplo de resultado que eu espero:

Resposta 1: texto;

Traducao 1: texto;

Resposta 2: texto;

Traducao 2: texto;

Resposta 3: texto;

Traducao 3: texto;

Substitua a palavra texto pelo resultado obtido. Mostre também um tópico de "Dicas:"

Substituir o idioma pelo escolhido do aluno

Substituir a frase pela resposta do modelo.

Resposta esperada da frase

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

### JSON Convert

“””

Por favor retorne JSON descrevendo as respostas, a tradução e as dicas desse texto usando o seguinte schema:   
    {"resposta": list[RESPOSTA], "traducao":list[TRADUCAO], "dica":list[DICA}  
  
    RESPOSTA = {"numero": int, "descricao": str}  
    TRADUCAO = {"numero": int, " descricao ": str}  
    DICA = {"numero": int, " titulo ": str, " descricao ": str}  
  
    Todos os campos são necessários  
  
    Importante: Só retorne um único texto valido de JSON.  
  
    Aqui está o conteúdo:  
  
    """) + texto

### Model Tools

<https://ai.google.dev/gemini-api/docs/function-calling/python>

# Funções e Métodos

## Função chat\_principal

Essa é uma visão geral do que a função `chat\_principal()` faz:

1. Boas-vindas e Instruções:

- A função `chat\_principal()` começa imprimindo uma mensagem de boas-vindas e instruções para o usuário. Ela informa que o módulo permite a prática de um idioma escolhido pelo usuário e fornece algumas ferramentas durante a conversa, como dicas e traduções.

2. Inicia a conversa:

- Após as boas-vindas, a função inicia uma conversa utilizando um modelo de chat (`model.start\_chat()`) e passa uma lista vazia como histórico de mensagens.

3. Introdução da conversa:

- Em seguida, a função fornece algumas informações sobre a conversa, como o nível de dificuldade (neste caso, básico) e o assunto da conversa (quem está se conhecendo).

4. Envio da primeira mensagem:

- O código envia a primeira mensagem para iniciar a conversa, utilizando as informações de nível, assunto e idioma escolhido pelo usuário.

5. Troca de mensagens:

- Após o envio da primeira mensagem, o código entra em um loop que permite a troca de mensagens entre o usuário e o sistema. O usuário pode digitar sua resposta, solicitar dicas, traduções ou encerrar a conversa digitando "fim".

6. Processamento das respostas:

- Dentro do loop, o código verifica se a resposta do usuário é uma solicitação de dica ou tradução. Se for uma mensagem comum, ela é enviada para o modelo de chat para processamento.

7. Validação da conversa:

- O código também realiza uma validação para verificar se a resposta do usuário faz sentido na conversa. Se sim, ele fornece uma mensagem incentivando o usuário a continuar no mesmo idioma.

8. Oferta de dicas:

- Se o usuário solicitar uma dica, o código gera possíveis respostas utilizando a função `possiveis\_respostas()` e exibe as opções para o usuário escolher.

9. Tradução de frases:

- Se o usuário solicitar uma tradução, o código utiliza a função `unica\_pergunta()` para solicitar ao usuário que traduza a frase para o português e exibe o resultado.

10. Fim da conversa:

- O loop continua até que o usuário digite "fim", encerrando a conversa.

## Função `texto\_padrao\_dicas(texto, idioma="inglês")`:

Descrição:

Esta função gera um texto padrão para solicitar ao modelo de geração de texto que forneça três opções de resposta para uma determinada frase em um idioma específico. O texto também inclui um exemplo de como o resultado deve ser apresentado, com as respostas e suas traduções correspondentes.

Parâmetros:

- `texto`: A frase para a qual se deseja gerar opções de resposta.

- `idioma`: O idioma em que a resposta deve ser gerada (padrão é "inglês").

Retorno:

- Uma string contendo o texto padrão com as instruções para gerar opções de resposta.

Funcionamento:

1. Gera um texto com instruções para o modelo de geração de texto.

2. Inclui a frase fornecida pelo usuário no texto.

3. Define o idioma padrão como "inglês", a menos que um idioma diferente seja especificado.

4. Retorna o texto padrão.

## Função `possiveis\_respostas(texto)`:

Descrição:

Esta função utiliza o modelo de geração de texto para gerar opções de resposta, traduções e dicas com base em um texto padrão.

Parâmetros:

- `texto`: A frase para a qual se deseja gerar opções de resposta.

Retorno:

- Um dicionário contendo as opções de resposta, traduções e dicas.

Funcionamento:

1. Chama a função `texto\_padrao\_dicas()` para obter o texto padrão com instruções.

2. Utiliza o modelo de geração de texto para gerar opções de resposta com base no texto padrão.

3. Formata o resultado para extrair as opções de resposta, traduções e dicas.

4. Retorna um dicionário com as informações obtidas.

## Função `texto\_padrao\_JSON\_dicas(texto)`:

Descrição:

Esta função gera um texto padrão com instruções para retornar um JSON descrevendo as respostas de frases, as traduções e as dicas com base em um texto fornecido.

Parâmetros:

- `texto`: O texto a ser incluído nas instruções.

Retorno:

- Uma string contendo o texto padrão com as instruções para retornar um JSON.

Funcionamento:

1. Gera um texto com instruções para retornar um JSON com informações específicas.

2. Inclui o texto fornecido nas instruções.

3. Retorna o texto padrão com as instruções.

## Função `modelo\_dicas\_schema(texto)`:

Descrição:

Esta função configura um modelo de geração de texto para adicionar frases de respostas, traduções e dicas a um banco de dados.

Parâmetros:

- `texto`: O texto contendo as frases de respostas, traduções e dicas a serem adicionadas ao banco de dados.

Retorno:

- Não há retorno explícito. A função configura o modelo para adicionar as informações ao banco de dados.

Funcionamento:

1. Configura um modelo de geração de texto com uma ferramenta para adicionar informações ao banco de dados.

2. Utiliza o modelo para gerar texto com instruções para adicionar as informações ao banco de dados.

3. Retorna a função de chamada para adicionar informações ao banco de dados.

## Função `modelo\_dica\_regex(text)`:

Descrição:

Esta função não está implementada no código fornecido. Ela parece ser destinada a criar um modelo baseado em expressões regulares para processar texto, mas atualmente retorna uma string vazia.

    Parâmetros:

        - `text`: O texto da pergunta a ser gerada.

    Retorno:

        - A ideia e gerar um retorno estruturado e até transformado em lista de cada tópico.

## Função `unica\_pergunta(text)`:

Descrição:

Esta função utiliza o modelo de geração de texto para fazer uma pergunta com base no texto fornecido.

Parâmetros:

- `text`: O texto da pergunta a ser gerada.

Retorno:

- Uma string contendo a pergunta gerada pelo modelo.

Funcionamento:

1. Utiliza o modelo de geração de texto para gerar uma pergunta com base no texto fornecido.

2. Retorna a pergunta gerada pelo modelo.

## Função `validaConversa(texto)`:

Descrição:

Esta função faz uma pergunta ao usuário para validar se um determinado texto está no idioma especificado.

Parâmetros:

- `texto`: O texto a ser validado.

Retorno:

- Um valor booleano indicando se o texto está no idioma especificado (True) ou não (False).

Funcionamento:

1. Utiliza a função `unica\_pergunta()` para fazer uma pergunta ao usuário sobre o idioma do texto.

2. Verifica a resposta do usuário para determinar se o texto está no idioma especificado.

3. Retorna True se o texto estiver no idioma especificado e False caso contrário.