```
# O '!' diz ao Colab para executar como comando de sistema
!pip install SpeechRecognition gTTS pydub
```

```
Requirement already satisfied: SpeechRecognition in /usr/local/lib/pyth
Requirement already satisfied: gTTS in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: pydub in /usr/local/lib/python3.12/dist-
Requirement already satisfied: typing-extensions in /usr/local/lib/pyth
Requirement already satisfied: requests<3,>=2.27 in /usr/local/lib/pyth
Requirement already satisfied: click<8.2,>=7.1 in /usr/local/lib/python
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/l
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.1
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/pyt
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/pyt
```

```
from IPython.display import display, Javascript, Audio
from google.colab import output
import base64

def record_audio(filename='input_audio.wav'):
    js = Javascript('''
        async function recordAudio() {
            const stream = await navigator.mediaDevices.getUserMedia({audio:
            const recorder = new MediaRecorder(stream);
            const chunks = [];

            return new Promise((resolve) => {
                const div = document.createElement('div');
                const button = document.createElement('button');
                button.textContent = '🎤 Clique para Gravar';
                button.style.cssText = "padding: 10px; border-radius: 5px; b
                document.body.appendChild(div);
                div.appendChild(button);

                recorder.ondataavailable = (e) => chunks.push(e.data);

                button.onclick = () => {
                    if (recorder.state === 'inactive') {
                        recorder.start();
                        button.textContent = '🛑 Parar Gravação';
                        button.style.background = "#f44336";
                    } else {
                        recorder.stop();
                        button.textContent = '✅ Processando...';
                    }
                };

                recorder.onstop = async () => {
                    const blob = new Blob(chunks, {type: 'audio/wav'});
                    const reader = new FileReader();
                    reader.readAsDataURL(blob);
                    reader.onloadend = () => resolve(reader.result);
                    div.remove();
                };
            });
        }
    ''')
    display(js)
    data = output.eval_js('recordAudio()')
```

```
        binary = base64.b64decode(data.split(',')[1])

        with open(filename, 'wb') as f:
            f.write(binary)

        # VERIFICAÇÃO: Mostra um player para você testar se o som foi gravado
        print("Verificação: Ouça o áudio abaixo para confirmar se o microfone fu
        display(Audio(filename))

        return filename
```

```
```

```
```

```
import speech_recognition as sr
from pydub import AudioSegment
from IPython.display import Javascript, display
import os

def stt_processor(audio_file):
    recognizer = sr.Recognizer()

    if not os.path.exists(audio_file):
        print("Erro: Arquivo não encontrado.")
        return None

    try:
        # --- PASSO DE CONVERSÃO CRUCIAL ---
        # O navegador grava em WebM/Ogg. Vamos converter para WAV PCM real.
        audio = AudioSegment.from_file(audio_file)
        wav_filename = "converted_audio.wav"
        audio.export(wav_filename, format="wav")
        # ----------------------------------

        with sr.AudioFile(wav_filename) as source:
            # Ajuste de ruído para melhorar a precisão
            recognizer.adjust_for_ambient_noise(source, duration=0.5)
            audio_data = recognizer.record(source)

            # Transcrição via Google
            text = recognizer.recognize_google(audio_data, language="pt-BR")
            return text

    except sr.UnknownValueError:
        print("DEBUG: O áudio foi lido, mas as palavras não foram reconheci
    except Exception as e:
        print(f"DEBUG: Erro na conversão/processamento: {e}")
    return None

def tts_browser_processor(text):
    # Mantendo a versão JavaScript que funcionou bem
    safe_text = text.replace("'", "\\'").replace("\n", " ")
    js_code = f'''
        window.speechSynthesis.cancel(); // Para conversas anteriores
        const utterance = new SpeechSynthesisUtterance('{safe_text}');
        utterance.lang = 'pt-BR';
        window.speechSynthesis.speak(utterance);
```

```
    '''
    display(Javascript(js_code))
```

```
/usr/local/lib/python3.12/dist-packages/pydub/utils.py:300: SyntaxWarni
  m = re.match('([su]([0-9]{1,2})p?) \(([0-9]{1,2}) bit\)$', token)
/usr/local/lib/python3.12/dist-packages/pydub/utils.py:301: SyntaxWarni
  m2 = re.match('([su]([0-9]{1,2})p?)( \(default\))?$', token)
/usr/local/lib/python3.12/dist-packages/pydub/utils.py:310: SyntaxWarni
  elif re.match('(flt)p?( \(default\))?$', token):
/usr/local/lib/python3.12/dist-packages/pydub/utils.py:314: SyntaxWarni
  elif re.match('(dbl)p?( \(default\))?$', token):
```

```
def run_voice_chat():
    print("🎤 Aguardando sua voz... (Clique no botão que aparecerá abaixo)")

    # 1. Grava
    path = record_audio()

    # 2. Transcreve
    user_text = stt_processor(path)

    if user_text:
        print(f"👤 Você: {user_text}")

        # 3. Lógica (Exemplo simples)
        if "ajuda" in user_text.lower():
            reply = "Claro! Eu sou um assistente executado inteiramente no s
        else:
            reply = f"Recebi sua mensagem: {user_text}"

        print(f"🤖 Bot: {reply}")

        # 4. Fala usando o Navegador
        tts_browser_processor(reply)
    else:
        print("❌ Bot: Não consegui processar seu áudio. Tente falar mais pe

# Executar
run_voice_chat()
```

```
🎤 Aguardando sua voz... (Clique no botão que aparecerá abaixo)
Verificação: Ouça o áudio abaixo para confirmar se o microfone funcionc




👤 Você: ajuda
🤖 Bot: Claro! Eu sou um assistente executado inteiramente no seu naveg
```

```
!pip install -q -U google-generativeai yfinance SpeechRecognition gTTS pydub
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 155.1/155.1 kB 5.2 M
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.1/127.1 kB 10.3
```

```
import google.generativeai as genai
```

```
import yfinance as yf
from datetime import datetime

# Configure sua API Key aqui
genai.configure(api_key="AIzaSyAHNkiyNRQNHHIF54sDaSRuZzQv_6OMht8")
model = genai.GenerativeModel('gemini-2.5-flash')

def get_market_data():
    """Busca dados básicos do mercado brasileiro"""
    try:
        tickers = {"Dólar": "USDBRL=X", "Ibovespa": "^BVSP"}
        info = ""
        for name, ticker in tickers.items():
            data = yf.Ticker(ticker).history(period="1d")
            price = data['Close'].iloc[-1]
            info += f"{name}: R$ {price:.2f}. "
        return info
    except:
        return "Não consegui buscar dados em tempo real agora."

def financial_brain(user_query):
    market_context = get_market_data()
    prompt = f"""
    Você é um assistente financeiro brasileiro especializado e amigável.
    Contexto atual do mercado: {market_context}
    Data de hoje: {datetime.now().strftime('%d/%m/%Y')}

    Pergunta do usuário: {user_query}

    Responda de forma curta, clara e objetiva.
    Se o usuário perguntar sobre investimentos, lembre-o que isso não é uma
    """
    response = model.generate_content(prompt)
    return response.text
```

```
def financial_assistant():
    print("--- 💰 ASSISTENTE FINANCEIRO ATIVO ---")
    print("Como deseja interagir? [1] Voz | [2] Texto")

    choice = input("Escolha: ")
    user_input = ""

    if choice == "1":
        path = record_audio() # Usa a função da Célula 2 anterior
        user_input = stt_processor(path) # Usa a função da Célula 3 anterior
    else:
        user_input = input("Digite sua dúvida financeira: ")

    if user_input:
        print(f"\n👤 Você: {user_input}")

        # O Cérebro processa a resposta com contexto do mercado
        response_text = financial_brain(user_input)

        print(f"\n🤖 Assistente: {response_text}")

        # O bot fala a resposta
        tts_browser_processor(response_text) # Usa a função JS anterior
    else:
        print("Não entendi sua mensagem.")

# Para rodar, você precisará ter as funções record_audio, stt_processor
```

```
# e tts_browser_processor das etapas anteriores definidas.
financial_assistant()
```

```
--- 💰 ASSISTENTE FINANCEIRO ATIVO ---
Como deseja interagir? [1] Voz | [2] Texto
Escolha: 2
Digite sua dúvida financeira: Qual o valor do dólar?

👤 Você: Qual o valor do dólar?

🤖 Assistente: Olá! O dólar está em R$ 5,35 hoje.
```