GEMINI-AI VOICEBOT USING STREAMLIT

by

Lakshmi Hasa

MST03-0055

Submitted to Scifor Technologies

Meta

Scifor Technologies

Script. Sculpt. Socialize

UNDER GUIDIANCE OF

Urooj Khan

TABLE OF CONTENTS

ABSTRACT

The Gemini AI Voicebot project explores the development of a sophisticated voice-interactive chatbot designed to enhance user engagement through advanced natural language processing (NLP) and speech recognition technologies. This project is built on the Gemini AI platform, which provides powerful tools for understanding and generating human-like responses in real-time. The voicebot's primary objective is to create an intuitive and responsive conversational interface that can interpret spoken queries, provide accurate responses, and retain conversation history to ensure continuity and context-awareness.

The project utilizes Python as the core programming language, leveraging its extensive libraries and frameworks to build a robust and scalable application. Key technologies integrated into the voicebot include the SpeechRecognition library, which is responsible for converting user voice input into text, and

The development process followed a structured methodology, beginning with the integration of essential libraries and tools required for AI-driven voice processing. Environment configuration was a critical step, involving the secure management of API keys and other sensitive data to ensure that the application operates within a protected and reliable environment.

The voicebot was subjected to rigorous testing across various scenarios to evaluate its performance in understanding and responding to different types of queries. The tests revealed that the bot is highly effective in maintaining conversation context, allowing it to remember previous interactions and provide more personalized and relevant responses over time. This capability is particularly beneficial in applications where continuous interaction with the user is necessary, such as customer support, virtual assistants, and interactive learning environments.

In conclusion, the Gemini AI Voicebot project represents a significant advancement in the field of conversational AI, showcasing the potential of voice-enabled interfaces to transform how users interact with digital systems.

INTRODUCTION

The rapid advancement of artificial intelligence (AI) has revolutionized how humans interact with digital systems, particularly through the development of conversational AI. Voice-enabled interfaces are becoming increasingly popular, providing a more natural and efficient means of communication between users and machines. The Gemini AI Voicebot project seeks to capitalize on these advancements by developing a sophisticated voice-interactive chatbot that leverages state-of-the-art natural language processing (NLP) and speech recognition technologies.

The primary objective of the Gemini AI Voicebot is to create a responsive and intuitive conversational agent capable of understanding and generating human-like responses in real time. By integrating advanced NLP algorithms with voice recognition software, the chatbot is designed to interpret spoken queries, process them accurately, and deliver contextually relevant responses. This project aims to demonstrate the potential of voicebots in enhancing user experiences across various applications, from customer support to personal virtual assistants.

The project utilizes Python as the core programming language, given its flexibility and extensive library support for AI development. Key components of the system include the SpeechRecognition library for converting voice inputs into text and Google Text-to-Speech (gTTS) for producing spoken responses. The voicebot also features a custom web interface that facilitates user interaction, making it accessible across multiple devices.

In this, we will explore the development process of the Gemini AI Voicebot, detailing the methodologies, technologies, and challenges encountered. We will also present the results of testing the bot’s capabilities in real-world scenarios, demonstrating its effectiveness in creating seamless and natural interactions between humans and machines. The project is built on the foundation of the Gemini AI platform, which provides powerful tools for processing natural language and generating coherent, human-like responses. The voicebot’s primary goal is to facilitate a smooth, conversational experience for users, whether they are seeking information, performing tasks, or simply interacting with the system.

METHODOLOGY

1. Library Integration

Integrated essential libraries like Gemini AI API for NLP, SpeechRecognition for converting voice to text, and Google Text-to-Speech (gTTS) for generating spoken responses. These libraries formed the backbone of the chatbot's interactive capabilities.

2. Environment Setup

Configured a secure environment to handle sensitive data like API keys. This ensured secure operations and minimized risks of breaches, while maintaining a reliable development and deployment process.

3. Voice Processing

Processed user voice inputs by converting them to text, analyzing them with Gemini AI for intent, and generating responses. These responses were then converted back to speech for natural interaction, all in real-time.

4. Interface Design

Developed a custom, user-friendly web interface that allowed seamless voice interactions across devices. The design focused on simplicity and ease of use, enhancing the overall user experience.

5. Testing and Optimization

Conducted thorough testing to ensure the chatbot's accuracy and responsiveness across different scenarios. Optimized the system based on feedback to improve performance and reliability.

TECHNOLGY USED

Programming Language: Python

Python was selected as the primary programming language due to its extensive ecosystem and ease of use. Its rich set of libraries supports rapid AI development, making it ideal for integrating complex functionalities like NLP and voice processing.

Framework: Custom Web Interface

A custom web interface was built to provide a seamless and intuitive platform for user interaction. This interface is responsive and designed for compatibility across various devices, ensuring that users can easily interact with the voicebot on desktops, tablets, and smartphones.

Libraries:

Gemini AI API: This API is the core of the voicebot's intelligence, enabling it to perform advanced natural language processing. It helps the bot understand user queries, detect intent, and generate appropriate responses, making conversations feel more human-like.

SpeechRecognition:

Used for capturing and converting voice input into text, this library is essential for interpreting user speech. It supports various speech engines, ensuring high accuracy in different environments.

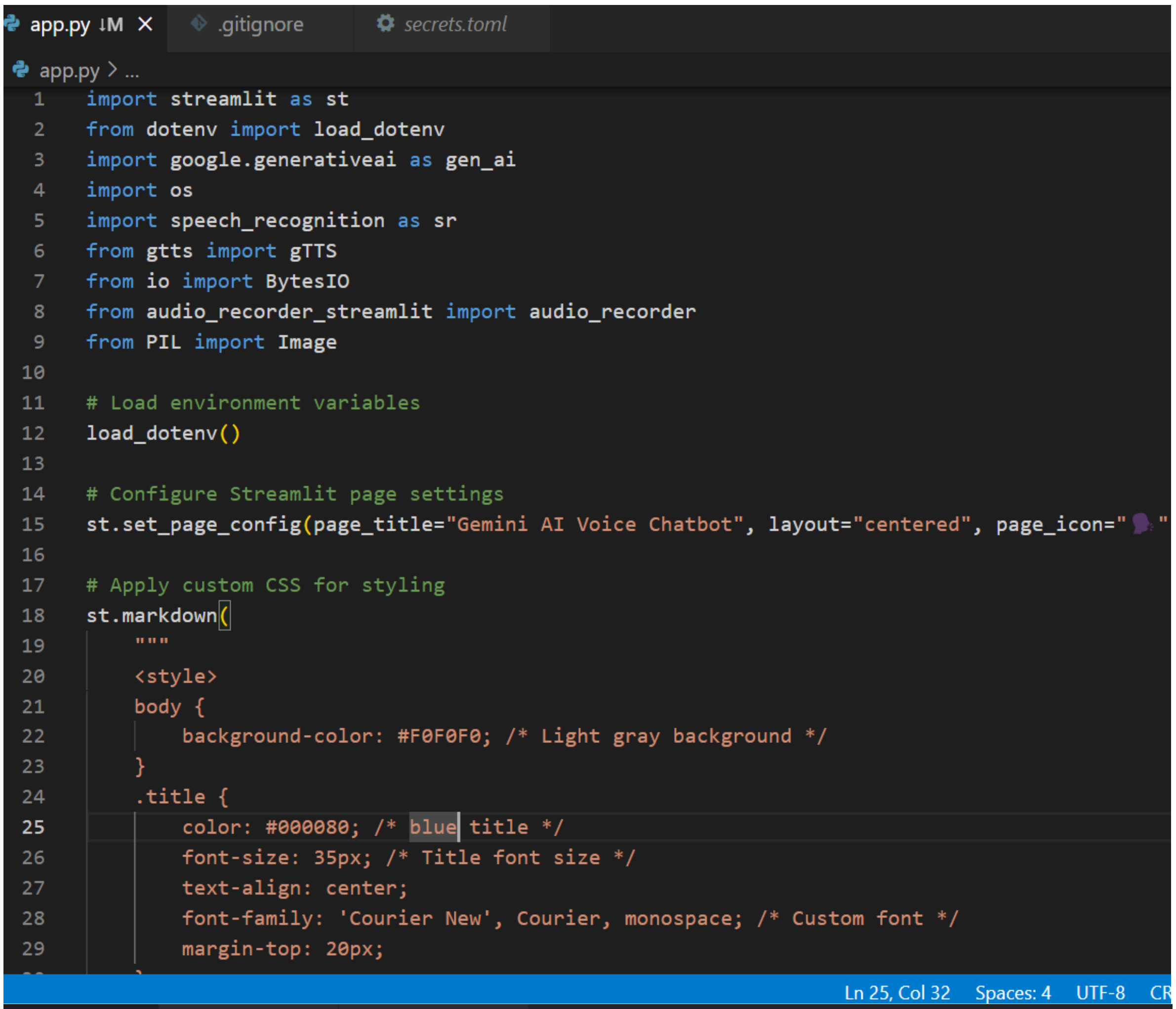
Google Text-to-Speech (gTTS):

gTTS is responsible for converting text-based responses into spoken language. It supports multiple languages and accents, providing a natural and clear voice output, enhancing the user experience.

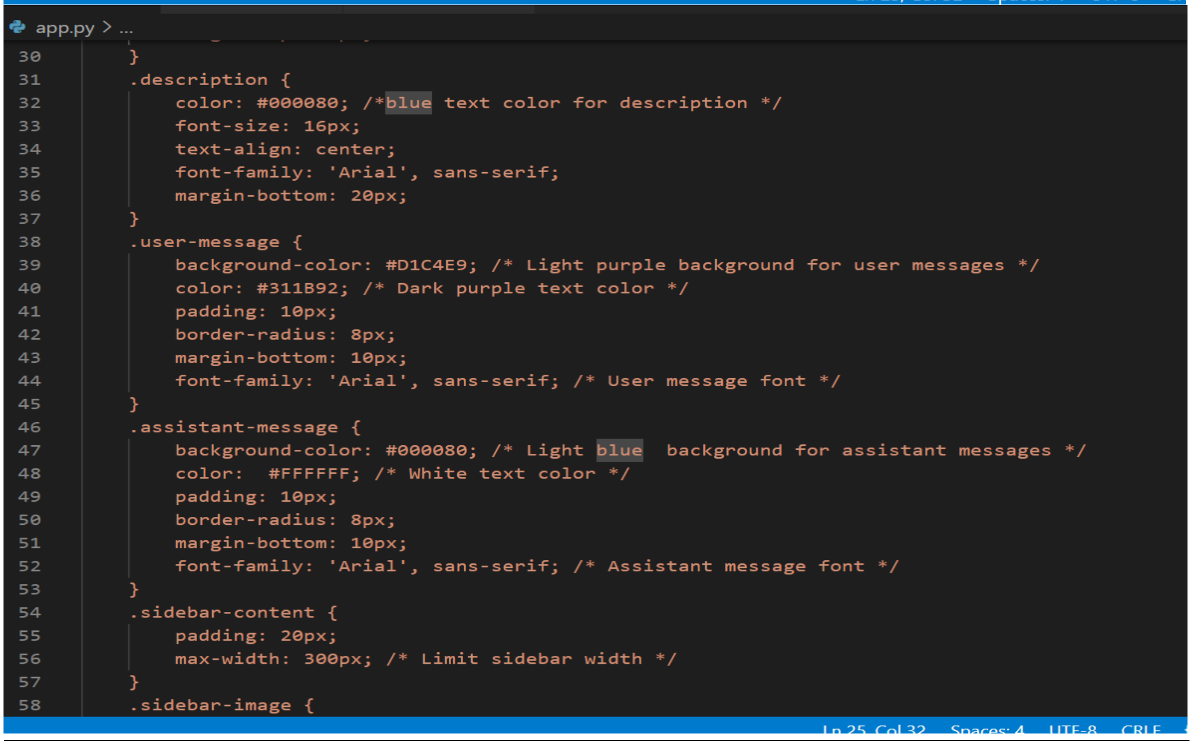
Environment Management:

Environment management involved securely storing and managing API keys, tokens, and other sensitive information using environment variables or secure vaults. This step ensured the integrity of the application, protecting it from security breaches and unauthorized access. Proper environment setup also included configuring dependencies and maintaining version control to support stable development and deployment.

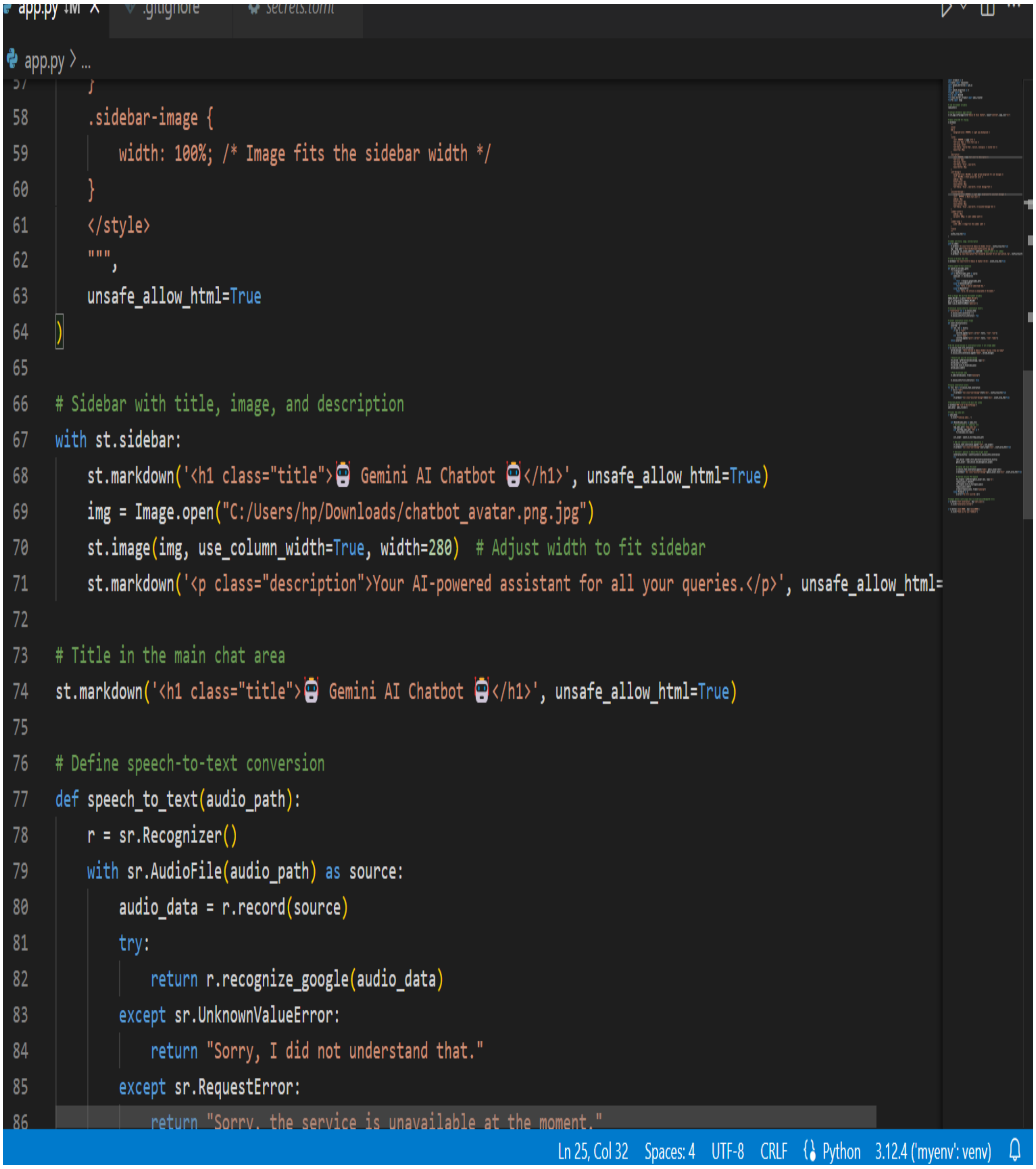
CODE SNIPPET



```python
import streamlit as st
from dotenv import load_dotenv
import google.generativeai as gen_ai
import os
import speech_recognition as sr
from gtts import gTTS
from io import BytesIO
from audio_recorder_streamlit import audio_recorder
from PIL import Image

# Load environment variables
load_dotenv()

# Configure Streamlit page settings
st.set_page_config(page_title="Gemini AI Voice Chatbot", layout="centered", page_icon="🗣"

# Apply custom CSS for styling
st.markdown(
    """
    <style>
    body {
        background-color: #F0F0F0; /* Light gray background */
    }
    .title {
        color: #000080; /* blue title */
        font-size: 35px; /* Title font size */
        text-align: center;
        font-family: 'Courier New', Courier, monospace; /* Custom font */
        margin-top: 20px;
```



```css
    }
    .description {
        color: #000080; /*blue text color for description */
        font-size: 16px;
        text-align: center;
        font-family: 'Arial', sans-serif;
        margin-bottom: 20px;
    }
    .user-message {
        background-color: #D1C4E9; /* Light purple background for user messages */
        color: #311B92; /* Dark purple text color */
        padding: 10px;
        border-radius: 8px;
        margin-bottom: 10px;
        font-family: 'Arial', sans-serif; /* User message font */
    }
    .assistant-message {
        background-color: #000080; /* Light blue  background for assistant messages */
        color:  #FFFFFF; /* White text color */
        padding: 10px;
        border-radius: 8px;
        margin-bottom: 10px;
        font-family: 'Arial', sans-serif; /* Assistant message font */
    }
    .sidebar-content {
        padding: 20px;
        max-width: 300px; /* Limit sidebar width */
    }
    .sidebar-image {
```

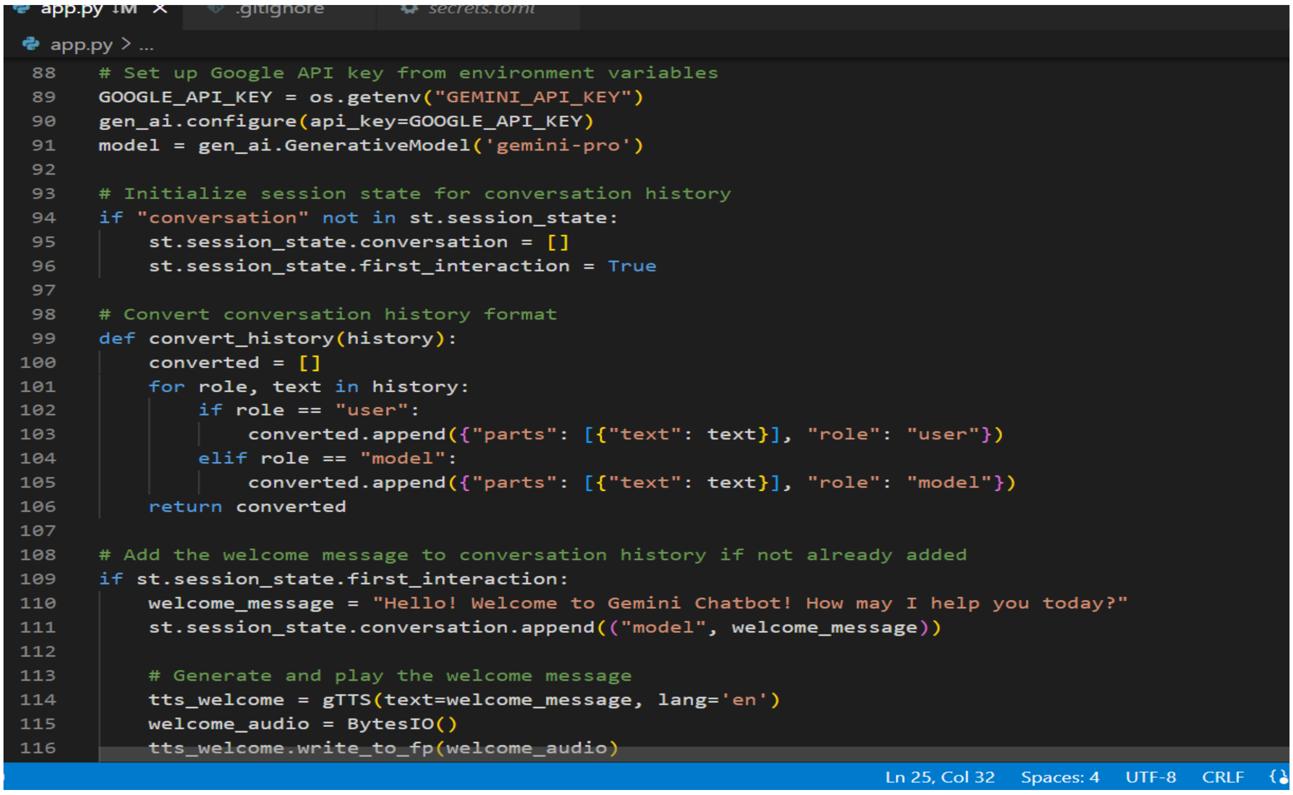


app.py 1M ×   .gitignore   secrets.toml

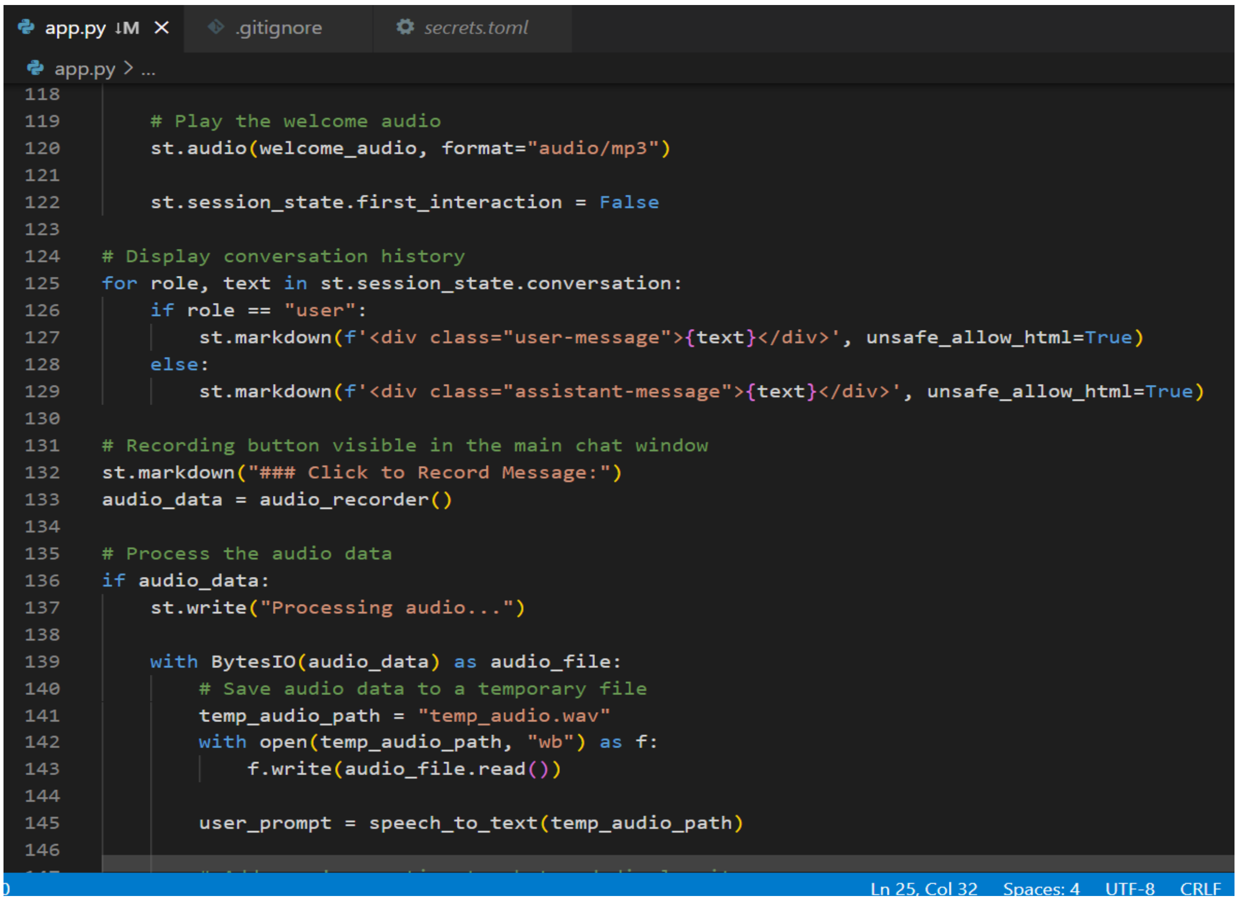
app.py > ...

```
    }
    .sidebar-image {
        width: 100%; /* Image fits the sidebar width */
    }
    </style>
    """,
    unsafe_allow_html=True
)

# Sidebar with title, image, and description
with st.sidebar:
    st.markdown('<h1 class="title">🤖 Gemini AI Chatbot 🤖</h1>', unsafe_allow_html=True)
    img = Image.open("C:/Users/hp/Downloads/chatbot_avatar.png.jpg")
    st.image(img, use_column_width=True, width=280)  # Adjust width to fit sidebar
    st.markdown('<p class="description">Your AI-powered assistant for all your queries.</p>', unsafe_allow_html=

# Title in the main chat area
st.markdown('<h1 class="title">🤖 Gemini AI Chatbot 🤖</h1>', unsafe_allow_html=True)

# Define speech-to-text conversion
def speech_to_text(audio_path):
    r = sr.Recognizer()
    with sr.AudioFile(audio_path) as source:
        audio_data = r.record(source)
        try:
            return r.recognize_google(audio_data)
        except sr.UnknownValueError:
            return "Sorry, I did not understand that."
        except sr.RequestError:
            return "Sorry, the service is unavailable at the moment."
```

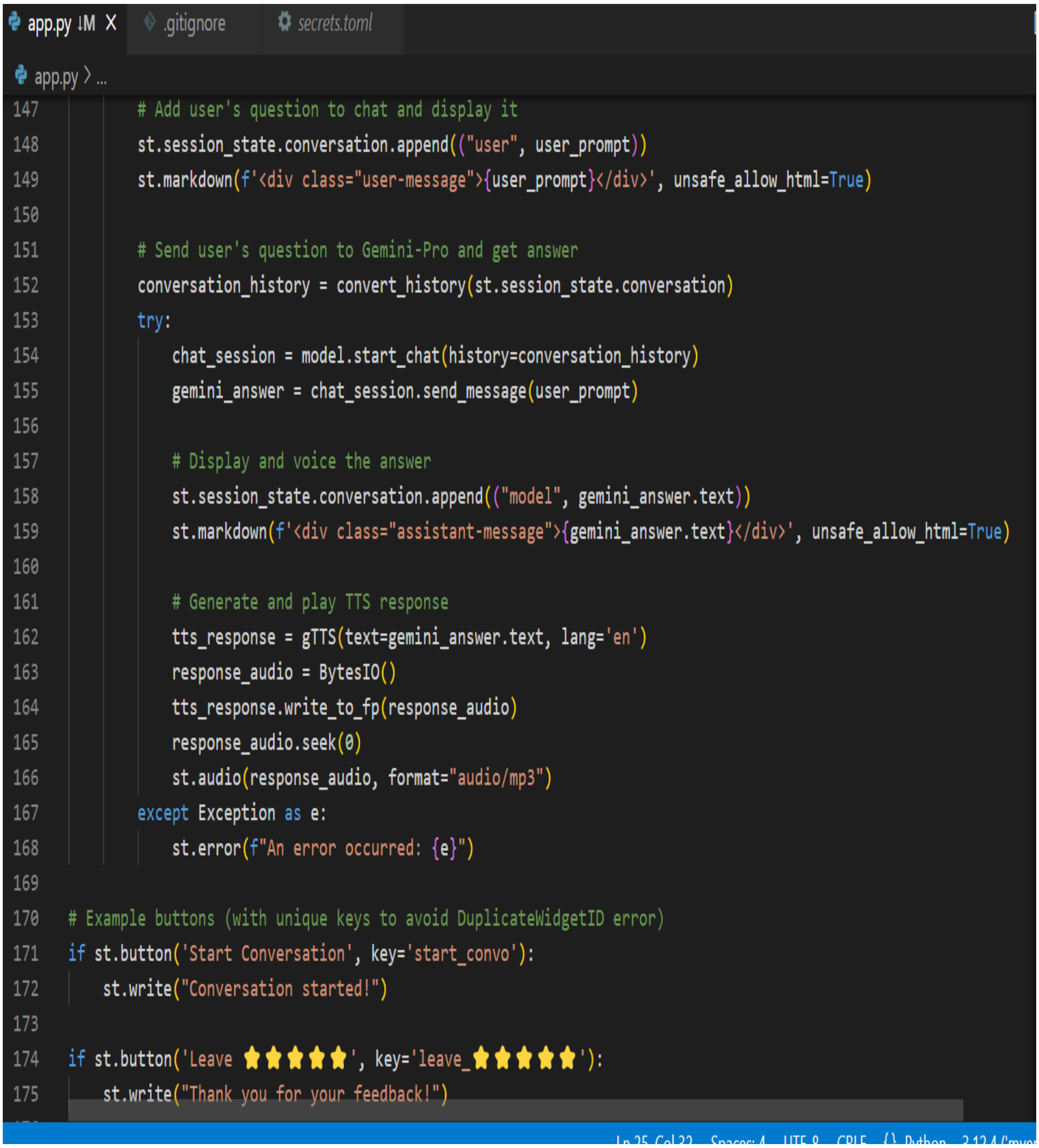
Ln 25, Col 32   Spaces: 4   UTF-8   CRLF   {} Python   3.12.4 ('myenv': venv)



```python
# Set up Google API key from environment variables
GOOGLE_API_KEY = os.getenv("GEMINI_API_KEY")
gen_ai.configure(api_key=GOOGLE_API_KEY)
model = gen_ai.GenerativeModel('gemini-pro')

# Initialize session state for conversation history
if "conversation" not in st.session_state:
    st.session_state.conversation = []
    st.session_state.first_interaction = True

# Convert conversation history format
def convert_history(history):
    converted = []
    for role, text in history:
        if role == "user":
            converted.append({"parts": [{"text": text}], "role": "user"})
        elif role == "model":
            converted.append({"parts": [{"text": text}], "role": "model"})
    return converted

# Add the welcome message to conversation history if not already added
if st.session_state.first_interaction:
    welcome_message = "Hello! Welcome to Gemini Chatbot! How may I help you today?"
    st.session_state.conversation.append(("model", welcome_message))

    # Generate and play the welcome message
    tts_welcome = gTTS(text=welcome_message, lang='en')
    welcome_audio = BytesIO()
    tts_welcome.write_to_fp(welcome_audio)
```



```python

    # Play the welcome audio
    st.audio(welcome_audio, format="audio/mp3")

    st.session_state.first_interaction = False

# Display conversation history
for role, text in st.session_state.conversation:
    if role == "user":
        st.markdown(f'<div class="user-message">{text}</div>', unsafe_allow_html=True)
    else:
        st.markdown(f'<div class="assistant-message">{text}</div>', unsafe_allow_html=True)

# Recording button visible in the main chat window
st.markdown("### Click to Record Message:")
audio_data = audio_recorder()

# Process the audio data
if audio_data:
    st.write("Processing audio...")

    with BytesIO(audio_data) as audio_file:
        # Save audio data to a temporary file
        temp_audio_path = "temp_audio.wav"
        with open(temp_audio_path, "wb") as f:
            f.write(audio_file.read())

        user_prompt = speech_to_text(temp_audio_path)

```



```python
        # Add user's question to chat and display it
        st.session_state.conversation.append(("user", user_prompt))
        st.markdown(f'<div class="user-message">{user_prompt}</div>', unsafe_allow_html=True)

        # Send user's question to Gemini-Pro and get answer
        conversation_history = convert_history(st.session_state.conversation)
        try:
            chat_session = model.start_chat(history=conversation_history)
            gemini_answer = chat_session.send_message(user_prompt)

            # Display and voice the answer
            st.session_state.conversation.append(("model", gemini_answer.text))
            st.markdown(f'<div class="assistant-message">{gemini_answer.text}</div>', unsafe_allow_html=True)

            # Generate and play TTS response
            tts_response = gTTS(text=gemini_answer.text, lang='en')
            response_audio = BytesIO()
            tts_response.write_to_fp(response_audio)
            response_audio.seek(0)
            st.audio(response_audio, format="audio/mp3")
        except Exception as e:
            st.error(f"An error occurred: {e}")

# Example buttons (with unique keys to avoid DuplicateWidgetID error)
if st.button('Start Conversation', key='start_convo'):
    st.write("Conversation started!")

if st.button('Leave ⭐⭐⭐⭐⭐', key='leave_⭐⭐⭐⭐⭐'):
    st.write("Thank you for your feedback!")
```

Ln 25, Col 22   Spaces: 4   UTF-8   CRLF   {} Python   3.12.4

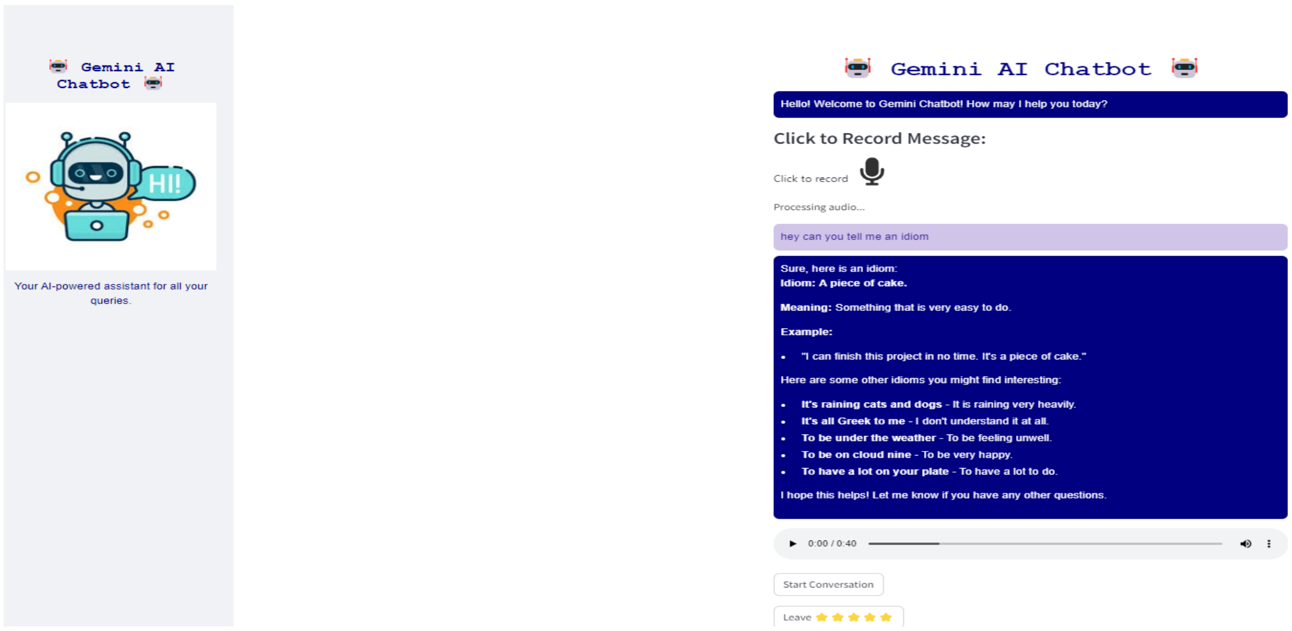
Results and Discussion

The Gemini AI Voicebot project yielded significant results in creating a responsive and intelligent conversational interface. During testing, the voicebot demonstrated high accuracy in understanding user queries, with the SpeechRecognition library effectively converting voice input into text under various conditions. The integration with the Gemini AI API allowed the bot to interpret and respond to a wide range of queries, maintaining context over multiple interactions. This context-awareness was particularly notable, as it enabled the bot to handle more complex conversations that required understanding of previous exchanges.

In terms of response quality, the Google Text-to-Speech (gTTS) module provided clear and natural-sounding voice output, making interactions feel more human-like. Users reported that the bot’s responses were timely and contextually appropriate, contributing to a positive user experience. However, the discussion also highlighted areas for improvement, such as the need for better handling of ambiguous queries and improving the voicebot's ability to manage multiple tasks simultaneously.

The custom web interface proved effective in facilitating user interaction, with most users finding it intuitive and easy to navigate. This positive feedback underscores the importance of a well-designed interface in enhancing the overall functionality of conversational AI systems.

Overall, the project successfully demonstrated the potential of combining NLP, speech recognition, and voice synthesis to create a versatile and user-friendly AI voicebot. Future work could focus on refining these capabilities, particularly in improving error handling and expanding the bot’s ability to engage in more complex dialogues.



Hello! Welcome to Gemini Chatbot! How may I help you today?

hey can you tell me an idiom

Sure, here is an idiom:
Idiom: A piece of cake.

Meaning: Something that is very easy to do.

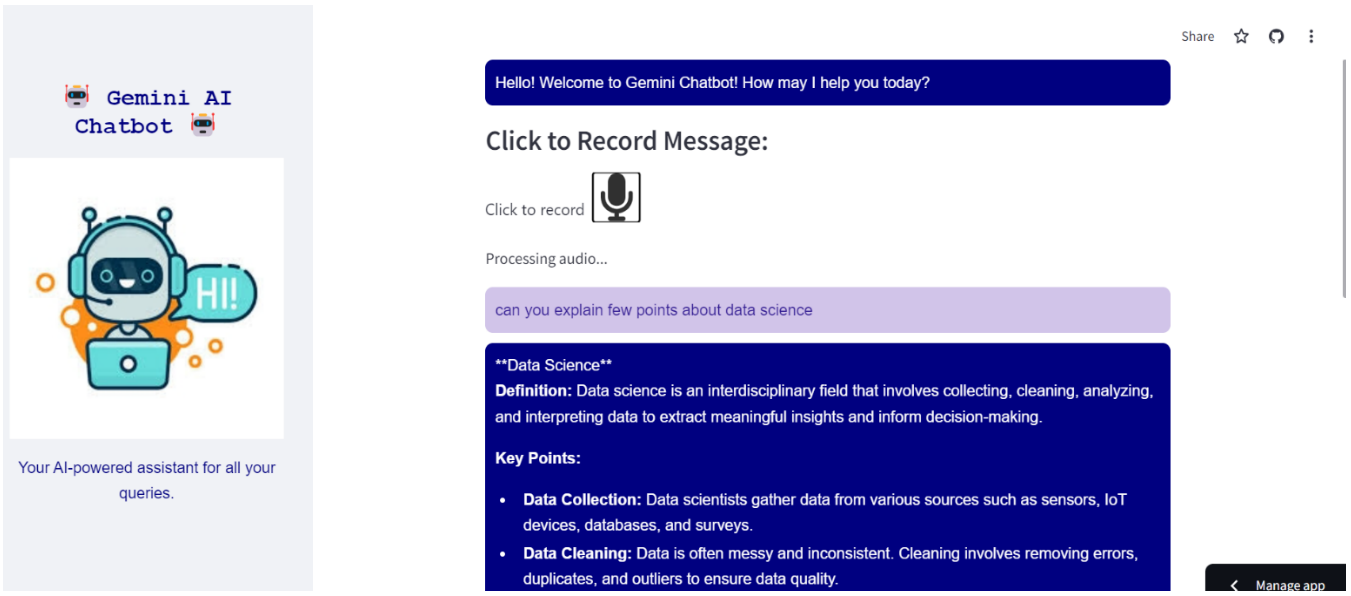
Example:

- "I can finish this project in no time. It's a piece of cake."

Here are some other idioms you might find interesting:

- **It's raining cats and dogs** - It is raining very heavily.
- **It's all Greek to me** - I don't understand it at all.
- **To be under the weather** - To be feeling unwell.
- **To be on cloud nine** - To be very happy.
- **To have a lot on your plate** - To have a lot to do.

I hope this helps! Let me know if you have any other questions.



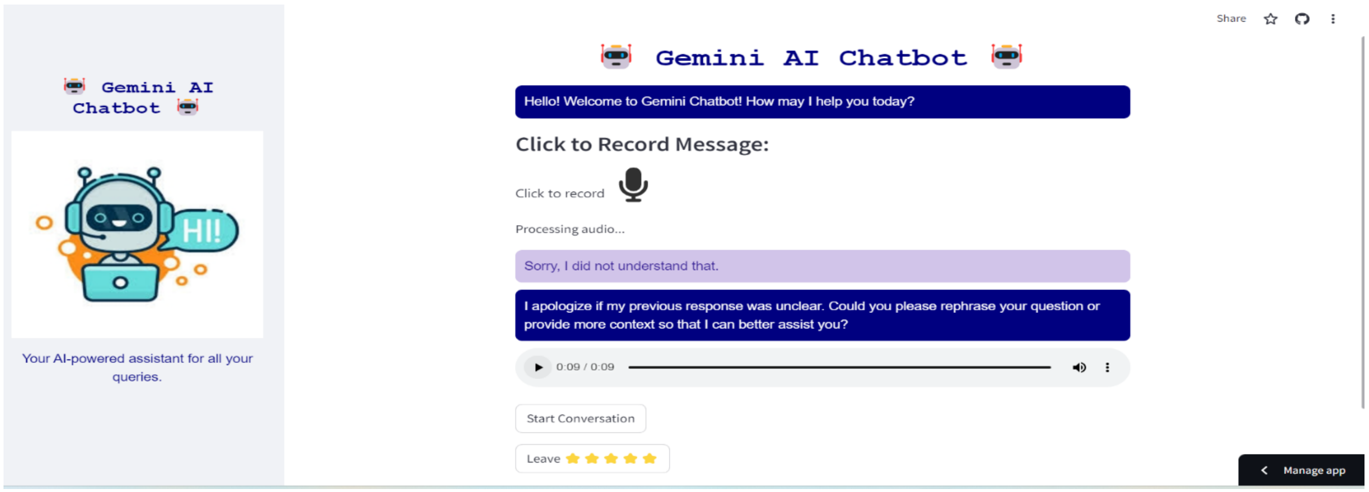
Hello! Welcome to Gemini Chatbot! How may I help you today?

can you explain few points about data science

Data Science
Definition: Data science is an interdisciplinary field that involves collecting, cleaning, analyzing, and interpreting data to extract meaningful insights and inform decision-making.

Key Points:

- **Data Collection:** Data scientists gather data from various sources such as sensors, IoT devices, databases, and surveys.
- **Data Cleaning:** Data is often messy and inconsistent. Cleaning involves removing errors, duplicates, and outliers to ensure data quality.



Hello! Welcome to Gemini Chatbot! How may I help you today?

Sorry, I did not understand that.

I apologize if my previous response was unclear. Could you please rephrase your question or provide more context so that I can better assist you?

CONCLUSION

The Gemini AI Voicebot project stands as a testament to the transformative power of modern artificial intelligence in enhancing human-computer interactions. By seamlessly integrating cutting-edge technologies such as natural language processing, speech recognition, and text-to-speech synthesis, the project successfully created a voicebot capable of engaging users in dynamic, context-aware conversations. This achievement not only highlights the practical applications of conversational AI but also underscores its potential to revolutionize how we interact with digital systems in our daily lives.

The voicebot's ability to understand and respond to a wide range of user queries in real-time demonstrates the effectiveness of combining advanced AI techniques with intuitive interface design. Users experienced a natural, fluid interaction that closely mimics human conversation, which is crucial in making AI-driven systems more accessible and user-friendly.

While the results are promising, they also pave the way for future enhancements. Addressing challenges such as handling ambiguous queries and managing more complex dialogues will be key to further refining the voicebot's capabilities. The Gemini AI Voicebot is not just a functional tool but a glimpse into the future of AI, where voice-activated systems become an integral part of our digital landscape, making technology more personal, responsive, and intelligent.

REFERENCES

Youtube video : **https://www.youtube.com/watch?v=o4ZhXSVuPyc&t=63s**

Information : Wikipedia

Github ref : https://github.com/ilhansevval/Gemini-Chatbot-Interface-with-Streamlit