Surya Group Of Instutions

Naan Mudhalvan

IBM -Artificial Intelligence

GNANAVEL B

Reg :42221104012

Team - 03

Chatbot In Python

Design For Creating A Chatbot In Python :

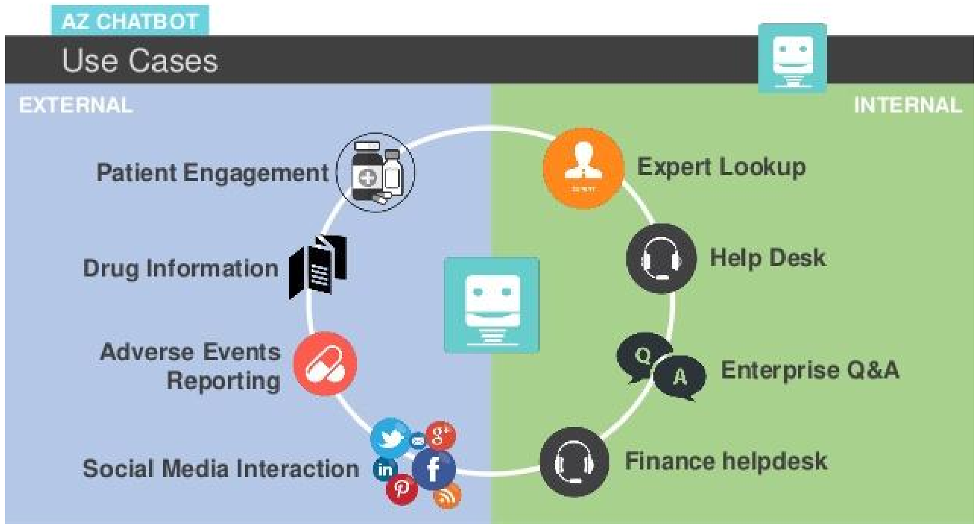
Chatbots are software applications that can interact with humans using natural language. They can be used for various purposes, such as customer service, entertainment, education, and more.

There are many ways to design and build a chatbot in Python, depending on your goals and preferences. One of the most popular frameworks for creating chatbots is ChatterBot, which uses natural language processing and machine learning to generate responses based on previous conversations. You can also use TensorFlow, a powerful library for deep learning, to train your own neural network model for chatbot dialogue.

Another option is to use OpenAI’s GPT-3, a state-of-the-art language model that can generate coherent and diverse texts based on a given prompt3. You can access GPT-3 through an API and use it to power your chatbot. However, you will need to request access from OpenAI and pay for the usage.

To create a user interface for your chatbot, you can use Gradio, a framework that makes it easy to develop web-based interfaces for machine learning models. You can also use FastAPI and WebSockets to create APIs for your chatbot and connect them with a front-end framework like React.

As you can see, there are many possibilities for creating a chatbot in Python. You can explore the links I provided for more details and tutorials on each approach. I hope this helps you get started on your chatbot project. Have fun! .



Steps to create a ChatBot with OpenAI and Gradio in Python :

Here we are going to see the steps to use OpenAI in Python with Gradio to create a chatbot.

Step 1 : Log in to your OpenAI account after creating one.

Step 2 : As shown in the figure below, after logging in, select Personal from the top-right menu, and then select "View API keys".

Step 3 : After completing step 2, a page containing API keys is displayed, and the button "Create new secret key" is visible. A secret key is generated when you click on that, copy it and save it somewhere else because it will be needed in further steps.

Step 4 : Import the openai, gradio library, and then do as follows. Store the created key in the below-mentioned variable.

```
import os
import openai
import gradio
```

Step 5 : Here we are getting the user chat history and storing it in a list and adding it to the previous state.

```
def message_and_history(input, history):
history = history or []
print(history)
s = list(sum(history, ()))
print(s)
s.append(input)
```

```
print('##########################################')
print(s)
inp = ' '.join(s)
print(inp)
output = api_calling(inp)
history.append((input, output))
print('------------------')
print(history)
print(history)
print("*********************")
return history, history
```

Step 6 : Now we create the header for the gradio application and we are defining the gradio UI Submitting the input we are changing the current state of the Chatbot.

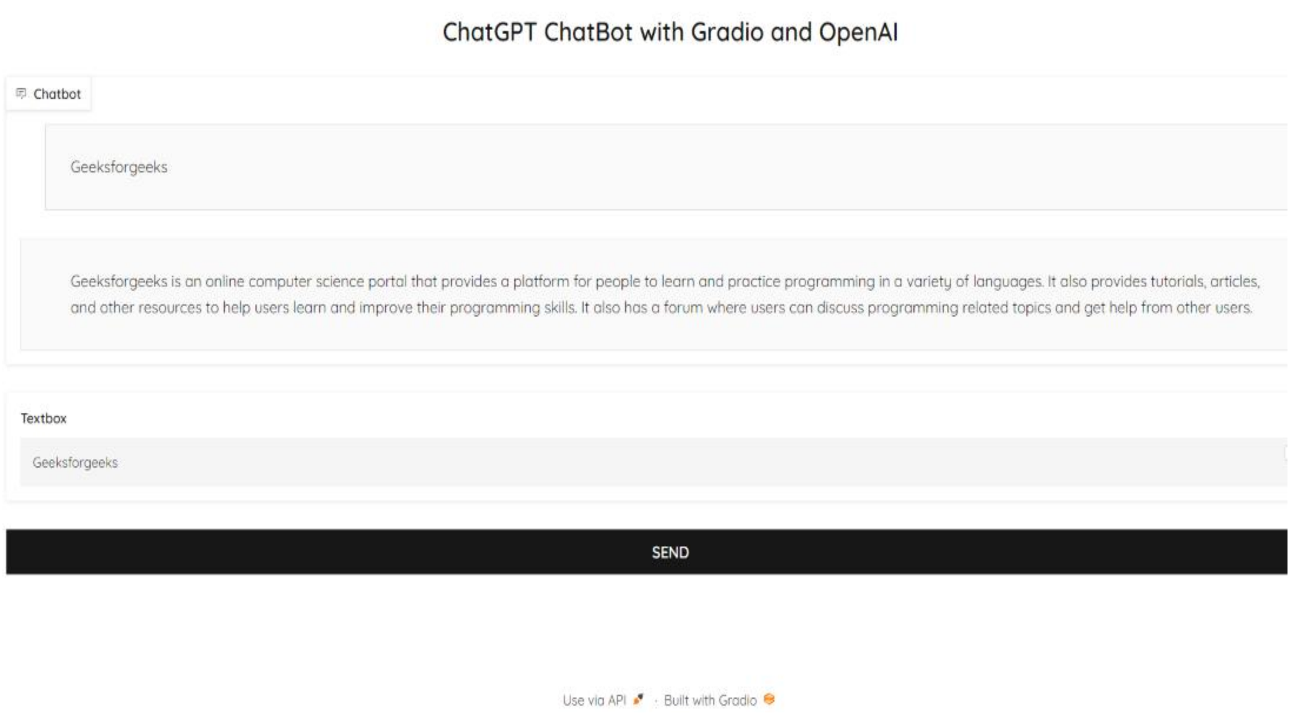
```
block = gradio.Blocks(theme=gradio.themes.Monochrome())
with block:
        gradio.Markdown("""<h1><center>ChatGPT ChatBot
        with Gradio and OpenAI</center></h1>
        """)
        chatbot = gradio.Chatbot()
        message = gradio.Textbox(placeholder=prompt)
        state = gradio.State()
        submit = gradio.Button("SEND")
        submit.click(message_and_history,
                        inputs=[message, state],
                        outputs=[chatbot, state])
block.launch(debug = True)
```

Complete Code :

```
import os
import openai

import gradio

openai.api_key = "YOUR_API_KEY"

prompt = "Enter Your Query Here"

def api_calling(prompt):

    completions = openai.Completion.create(

        engine="text-davinci-003",

        prompt=prompt,

        max_tokens=1024,

        n=1,

        stop=None,

        temperature=0.5,

    )

    message = completions.choices[0].text

    return message

def message_and_history(input, history):

    history = history or []

    s = list(sum(history, ()))

    s.append(input)

    inp = ' '.join(s)
```

```
        output = api_calling(inp)
        history.append((input, output))
        return history, history
block = gradio.Blocks(theme=gradio.themes.Monochrome())
with block:
        gradio.Markdown("""<h1><center>ChatGPT
        ChatBot with Gradio and OpenAI</center></h1>
        """)
        chatbot = gradio.Chatbot()
        message = gradio.Textbox(placeholder=prompt)
        state = gradio.State()
        submit = gradio.Button("SEND")
        submit.click(message_and_history,
                              inputs=[message, state],
                              outputs=[chatbot, state])
block.launch(debug = True)
```

Output :



ChatGPT ChatBot with Gradio and OpenAI

Geeksforgeeks

Geeksforgeeks is an online computer science portal that provides a platform for people to learn and practice programming in a variety of languages. It also provides tutorials, articles, and other resources to help users learn and improve their programming skills. It also has a forum where users can discuss programming related topics and get help from other users.

Geeksforgeeks