TASK 9-Chatbot system

a. Simple ChatGPT using openai

PROGRAM

```
Pip install openai
import openai
openai.api_key = "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"
completion = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages=[{"role": "user",
"content": "Give me 3 ideas that i could build using openai apis"}])
print(completion.choices[0].message.content)
```

Output:

```
1. Personalized Content Recommendation System: Develop an AI-powered content
recommendation system that suggests personalized content to users based on their interests and
search history. Use OpenAI's language generation APIs to generate relevant content
descriptions and summaries, and employ their natural language processing (NLP) APIs to
understand user preferences and interests.
2. Intelligent Chatbot: Build a conversational AI-enabled chatbot that can answer customer
queries, provide helpful recommendations, and complete transactions seamlessly. Use
OpenAI's language processing APIs to train the chatbot to understand user inputs and respond
in natural language. Integration with other APIs such as payment gateways and customer
databases can make the chatbot efficient and effective.
3. Fraud Detection System: Develop a machine learning model that can identify and prevent
fraudulent activities using OpenAI's anomaly detection and classification APIs. Train the
model using historical data of fraudulent transactions, and use the APIs to continuously scan
for and identify suspicious activities. Such a system can be deployed in a range of applications
```

such as finance or e-commerce platforms.

b. ChatGPT Assistant using openai

Code:

```
import openai

openai.api_key = "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"

messages = []

system_msg = input("What type of chatbot would you like to create?\n")

messages.append({"role": "system", "content": system_msg})

print("Your new assistant is ready! Type your query")

while input != "quit()":

 message = input()

 messages.append({"role": "user", "content": message})

 response = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages=messages)

 reply = response["choices"][0]["message"]["content"]

 messages.append({"role": "assistant", "content": reply})

 print("\n" + reply + "\n")
```

Output:

```
What type of chatbot would you like to create?

Nila’s personal chatbot

(ctrl enter)

Your new assistant is ready!
```

c. CHATBOT CHAT ASSISTANT WEBSITE

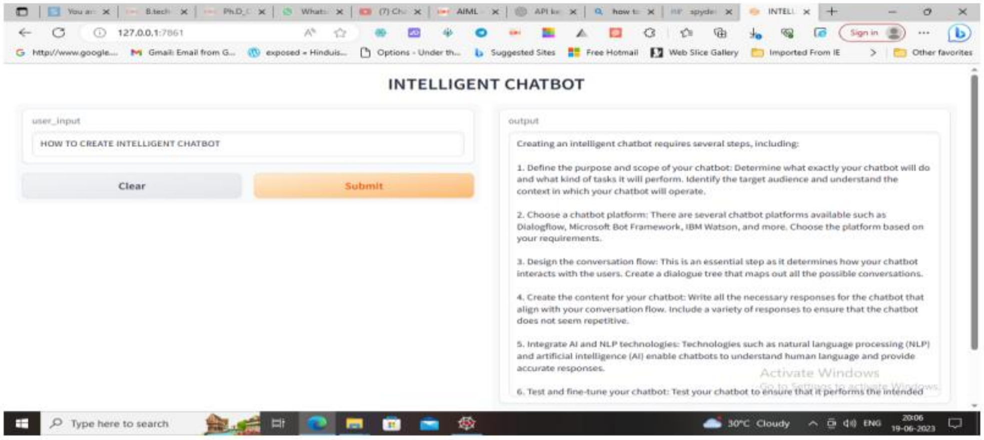
Code:

```
import openai

import gradio
```

```
openai.api_key = "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"
messages = [{"role": "system", "content": "You are a financial experts that specializes in real estate investment and negotiation"}]
def CustomChatGPT(user_input):
 messages.append({"role": "user", "content": user_input})
 response = openai.ChatCompletion.create(
 model = "gpt-3.5-turbo",
 messages = messages
 )
 ChatGPT_reply = response["choices"][0]["message"]["content"]
 messages.append({"role": "assistant", "content": ChatGPT_reply})
 return ChatGPT_reply
demo = gradio.Interface(fn=CustomChatGPT, inputs = "text", outputs = "text", title = "INTELLIGENT CHATBOT")
demo.launch(share=True)
```

OUTPUT:



INTELLIGENT CHATBOT

user_input

HOW TO CREATE INTELLIGENT CHATBOT

output

Creating an intelligent chatbot requires several steps, including:

1. Define the purpose and scope of your chatbot: Determine what exactly your chatbot will do and what kind of tasks it will perform. Identify the target audience and understand the context in which your chatbot will operate.

2. Choose a chatbot platform: There are several chatbot platforms available such as Dialogflow, Microsoft Bot Framework, IBM Watson, and more. Choose the platform based on your requirements.

3. Design the conversation flow: This is an essential step as it determines how your chatbot interacts with the users. Create a dialogue tree that maps out all the possible conversations.

4. Create the content for your chatbot: Write all the necessary responses for the chatbot that align with your conversation flow. Include a variety of responses to ensure that the chatbot does not seem repetitive.

5. Integrate AI and NLP technologies: Technologies such as natural language processing (NLP) and artificial intelligence (AI) enable chatbots to understand human language and provide accurate responses.

6. Test and fine-tune your chatbot: Test your chatbot to ensure that it performs the intended