**The Chat Format**

In this notebook, you will explore how you can utilize the chat format to have extended conversations with chatbots personalized or specialized for specific tasks or behaviors.

**Setup**

In [ ]:



**import** os

**import** openai

**from** dotenv **import** load\_dotenv, find\_dotenv

\_ **=** load\_dotenv(find\_dotenv()) *# read local .env file*

​

openai.api\_key **=** os.getenv('OPENAI\_API\_KEY')

In [ ]:



**def** get\_completion(prompt, model**=**"gpt-3.5-turbo"):

messages **=** [{"role": "user", "content": prompt}]

response **=** openai.ChatCompletion.create(

model**=**model,

messages**=**messages,

temperature**=**0, *# this is the degree of randomness of the model's output*

)

**return** response.choices[0].message["content"]

​

**def** get\_completion\_from\_messages(messages, model**=**"gpt-3.5-turbo", temperature**=**0):

response **=** openai.ChatCompletion.create(

model**=**model,

messages**=**messages,

temperature**=**temperature, *# this is the degree of randomness of the model's output*

)

*# print(str(response.choices[0].message))*

**return** response.choices[0].message["content"]

In [ ]:



messages **=** [

{'role':'system', 'content':'You are an assistant that speaks like Shakespeare.'},

{'role':'user', 'content':'tell me a joke'},

{'role':'assistant', 'content':'Why did the chicken cross the road'},

{'role':'user', 'content':'I don\'t know'} ]

In [ ]:



response **=** get\_completion\_from\_messages(messages, temperature**=**1)

print(response)

In [ ]:



messages **=** [

{'role':'system', 'content':'You are friendly chatbot.'},

{'role':'user', 'content':'Hi, my name is Isa'} ]

response **=** get\_completion\_from\_messages(messages, temperature**=**1)

print(response)

In [ ]:



messages **=** [

{'role':'system', 'content':'You are friendly chatbot.'},

{'role':'user', 'content':'Yes, can you remind me, What is my name?'} ]

response **=** get\_completion\_from\_messages(messages, temperature**=**1)

print(response)

In [ ]:



messages **=** [

{'role':'system', 'content':'You are friendly chatbot.'},

{'role':'user', 'content':'Hi, my name is Isa'},

{'role':'assistant', 'content': "Hi Isa! It's nice to meet you. \

Is there anything I can help you with today?"},

{'role':'user', 'content':'Yes, you can remind me, What is my name?'} ]

response **=** get\_completion\_from\_messages(messages, temperature**=**1)

print(response)

**OrderBot**

We can automate the collection of user prompts and assistant responses to build a OrderBot. The OrderBot will take orders at a pizza restaurant.

In [ ]:



**def** collect\_messages(\_):

prompt **=** inp.value\_input

inp.value **=** ''

context.append({'role':'user', 'content':f"{prompt}"})

response **=** get\_completion\_from\_messages(context)

context.append({'role':'assistant', 'content':f"{response}"})

panels.append(

pn.Row('User:', pn.pane.Markdown(prompt, width**=**600)))

panels.append(

pn.Row('Assistant:', pn.pane.Markdown(response, width**=**600, style**=**{'background-color': '#F6F6F6'})))

**return** pn.Column(**\***panels)

​

In [ ]:



**import** panel **as** pn *# GUI*

pn.extension()

​

panels **=** [] *# collect display*

​

context **=** [ {'role':'system', 'content':"""

You are OrderBot, an automated service to collect orders for a pizza restaurant. \

You first greet the customer, then collects the order, \

and then asks if it's a pickup or delivery. \

You wait to collect the entire order, then summarize it and check for a final \

time if the customer wants to add anything else. \

If it's a delivery, you ask for an address. \

Finally you collect the payment.\

Make sure to clarify all options, extras and sizes to uniquely \

identify the item from the menu.\

You respond in a short, very conversational friendly style. \

The menu includes \

pepperoni pizza 12.95, 10.00, 7.00 \

cheese pizza 10.95, 9.25, 6.50 \

eggplant pizza 11.95, 9.75, 6.75 \

fries 4.50, 3.50 \

greek salad 7.25 \

Toppings: \

extra cheese 2.00, \

mushrooms 1.50 \

sausage 3.00 \

canadian bacon 3.50 \

AI sauce 1.50 \

peppers 1.00 \

Drinks: \

coke 3.00, 2.00, 1.00 \

sprite 3.00, 2.00, 1.00 \

bottled water 5.00 \

"""} ] *# accumulate messages*

​

​

inp **=** pn.widgets.TextInput(value**=**"Hi", placeholder**=**'Enter text here…')

button\_conversation **=** pn.widgets.Button(name**=**"Chat!")

​

interactive\_conversation **=** pn.bind(collect\_messages, button\_conversation)

​

dashboard **=** pn.Column(

inp,

pn.Row(button\_conversation),

pn.panel(interactive\_conversation, loading\_indicator**=True**, height**=**300),

)

​

dashboard

In [ ]:



messages **=** context.copy()

messages.append(

{'role':'system', 'content':'create a json summary of the previous food order. Itemize the price for each item\

The fields should be 1) pizza, include size 2) list of toppings 3) list of drinks, include size 4) list of sides include size 5)total price '},

)

*#The fields should be 1) pizza, price 2) list of toppings 3) list of drinks, include size include price 4) list of sides include size include price, 5)total price '},*

​

response **=** get\_completion\_from\_messages(messages, temperature**=**0)

print(response)

**Try experimenting on your own!**

You can modify the menu or instructions to create your own orderbot!

In [ ]:



​