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 [8]:
```
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 [9]:
```
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 out
    )
    return response.choices[0].message["content"]

def get_completion_from_messages(messages, model="gpt-3.5-turbo", temperatur
    response = openai.ChatCompletion.create(
        model=model,
        messages=messages,
        temperature=temperature, # this is the degree of randomness of the m
    )
#     print(str(response.choices[0].message))
    return response.choices[0].message["content"]
```

In [10]:
```
messages =  [
{'role':'system', 'content':'You are an assistant that speaks like Shakespea
{'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 [11]:
```
response = get_completion_from_messages(messages, temperature=1)
print(response)
```

```
Why did the chicken cross the road, you asketh?
To avoid the terrible fate of being roasted in a flask.
With feathers sleek and a yearning heart,
It sought a life where freedom held a part.
Forsooth, 'twas no simple whim or jest,
But a quest for liberation, at its behest.
So, good sir or madam, now thou dost know,
The reason why the chicken wished to go!
```

In [12]:
```
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)
```

```
Hi Isa! Nice to meet you. How can I help you today?
```

In [13]:
```
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)
```

```
I'm sorry, but as a chatbot, I don't have access to personal information abou
t users. Therefore, I don't know your name. Is there anything else I can assi
st you with?
```

In [14]:
```
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)
```

```
Your name is Isa. Is there anything else you would like me to help you with?
```

OrderBot

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

In [15]:
```
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={'b

    return pn.Column(*panels)
```

In [16]:
```
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 restaur
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 fina
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
```

Enter text here…

Chat!

User:

Assistant: Hello! Welcome to our pizza restaurant. How can I assist you today?

User: Hi

Assistant: Hi there! How can I help you today? Are you looking to place an order?

User: want to order a pizza

Assistant: Great! I can definitely help you with that. What kind of pizza would you like to order? ' pepperoni, cheese, and eggplant pizza.

User: cheese

Assistant: Great choice! Our cheese pizza comes in three sizes: small, medium, and large. Whi would you like?

User: medium

In [18]:
```
messages =  context.copy()
messages.append(
{'role':'system', 'content':'create a json summary of the previous food orde
 The fields should be 1) pizza, include size 2) list of toppings 3) list of
)
 #The fields should be 1) pizza, price 2) list of toppings 3) list of drinks

response = get_completion_from_messages(messages, temperature=0)
print(response)
```

```
Sure! Here's a JSON summary of your food order:

{
  "pizza": {
    "size": "medium",
    "type": "cheese"
  },
  "toppings": [],
  "drinks": [],
  "sides": [
    {
      "item": "fries",
      "size": "small"
    }
  ],
  "total_price": 9.25
}

Please note that the total price includes the medium cheese pizza and the sma
ll order of fries. Let me know if you need any further assistance!
```

Try experimenting on your own!

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

In [ ]: