Phase 4: Artificial Intelligence

Let's begin by installing the chatterbot library. For creating chatbot also need to install chatterbot corpus. Corpus - literal meaning is a collection of words. This contains a corpus of data that is included in the chatterbot module. Each corpus is nothing but a prototype of different input statements and their responses. These corpus are used by bots to train themselves. The most recommended method for installing chatterbot and chatterbot_corpus is by using pip.

Installation commands for terminal:

pip install chatterbot:

pip install chatterbot_corpus;

Connect Database

Storage Adapters allows you to connect to a particular storage unit or network. For using a storage adapter, we need to specify it. We will position the storage adapter by assigning it to the import path of the storage we want to use. Here we are using SQL Storage Adapter, which permits chatbot to connect to databases in SQL. By using the database parameter, we will create a new SQLite Database.

```
bot = ChatBot(
    'Buddy',
    storage_adapter='chatterbot.storage.SQLStorageAdapter',
    database_uri='sqlite:///database.sqlite3'
)
```

Matching the keyword

You can also position the logical adapter with a chatbot object. As the name implies, Logical Adapter regulates the logic behind the chatterbot, i.e., it picks responses for any input provided to it. This parameter contains a list of logical

operators. Chatterbot allows us to use a number of logical Adapters. When more than one logical adapter is put to use, the chatbot will calculate the confidence level, and the response with the highest calculated confidence will be returned as output. Here we have used two logical adapters: BestMatch and TimeLogicAdapter.

```
# Create object of ChatBot class with Logic Adapter
bot = ChatBot(
   'Buddy',
   logic_adapters=[
      'chatterbot.logic.BestMatch',
      'chatterbot.logic.TimeLogicAdapter'],
)
```

Chatbot Testing

The last step of this tutorial is to test the chatterbot's conversational skills. For testing its responses, we will call the get_responses() method of Chatbot instance.

```
# Get a response to the input text 'I would like to book a flight.'
response = bot.get_response('I have a complaint.')
print("Bot Response:", response)
```

output:

Bot Response: Please elaborate, your concern

In this example, we will create a simple chatbot that can answer general questions about an e-commerce store. We'll use ChatterBot for natural language processing.

Coding

```
from chatterbot import ChatBot
from chatterbot.trainers import ChatterBotCorpusTrainer

# Create a chatbot instance
chatbot = ChatBot('EcommerceBot')

# Create a new trainer for the chatbot
trainer = ChatterBotCorpusTrainer(chatbot)

# Train the chatbot on the English language
trainer.train("chatterbot.corpus.english")

while True:
    user_input = input("You: ")

    if user_input.lower() == 'quit':
        break

    response = chatbot.get_response(user_input)
    print("EcommerceBot:", response)
```

Product Enquires

```
def get_product_info(product_name):
    if product_name in products:
        return response_prompts["product"].format(
            product_name=product_name,
            product_price=products[product_name]["price"],
            product_description=products[product_name]["description"]
        )
    else:
        return response_prompts["not_found"]


def chatbot_response(user_input):
    user_input = user_input.lower()


    if any(greet in user_input for greet in greetings):
        return response_prompts["greet"]
    elif "bye" in user_input:
        return response_prompts["goodbye"]
    elif "price" in user_input:
        return "You can ask about product prices by specifying the product name."
    else:
        for product_name in products:
            if product_name in user_input:
                return get_product_info(product_name)
    return "I'm sorry, I didn't understand that. Please ask another question or
specify a product."
```

End

We will create a while loop for our chatbot to run in. When statements are passed in the loop, we will get an appropriate response for it, as we have already entered data into our database. If we get "Bye" or "bye" statement from the user, we can put an end to the loop and stop the program.

```
name=input("Enter Your Name: ")
print("Welcome to the Bot Service! Let me know how can I help you?")
while True:
    request=input(name+':')
    if request=='Bye' or request =='bye':
        print('Bot: Bye')
        break
    else:
        response=bot.get_response(request)
        print('Bot:',response)
```