### **PROJECT:CREATE A CHATBOT IN PYTHON**

### **1.Prepare the Dependencies**

The first step in creating a chatbot in Python with the ChatterBot library is to install the library in your system. It is best if you create and use a new Python virtual environment for the installation. To do so, you have to write and execute this command in your Python terminal

Pip install\_\_\_upgrade chatterpot \_corpus

Pip install--- upgrade chatterpot

### 2. Import Classes

Importing classes is the second step in the Python chatbot creation process. All you need to do is import two classes – ChatBot from chatterbot and ListTrainer from chatterbot.trainers. To do this, you can execute the following command

From chatterbot import chatbot

From chatterpot.trainers import ListTrainer

### 3. Create and Train the Chatbot

This is the third step on creating chatbot in python. The chatbot you are creating will be an instance of the class “ChatBot.” After creating a new ChatterBot instance, you can train the bot to improve its performance. Training ensures that the bot has enough knowledge to get started with specific responses to specific inputs. You have to execute the following command n

My\_bot =chatbot (name =’pybot’, read \_only=True,

[‘chatterbot.logic.Mathematical evaluation’,chatterpot. Logic.bestmatch]

# Main chat loop

print("Problem-Solving Chatbot: Hi! How can I help you today?")

while True:

user\_input = input("You: ").lower

if user\_input == "exit":

print("Problem-Solving Chatbot: Goodbye!")

break

response = solve\_math\_problem(user\_input)

if response:

print("Problem-Solving Chatbot:", response)

else:

closest\_operation = find\_closest\_match(user\_input, math\_operations.keys())

if closest\_operation:

print(f"Problem-Solving Chatbot: Did you mean '{closest\_operation}'?")

else:

print("Problem-Solving Chatbot: I couldn't understand the input.")

4.Gather Data and Knowledge:

For your chatbot to solve problems effectively, it needs access to relevant data and knowledge. This could include databases, APIs, or pre-trained models. In this example, we'll focus on a general problem-solving chatbot without external

5.Data sources.Natural Language Processing (NLP):

Implement natural language processing to understand and interpret user input. Libraries like NLTK, spaCy, or Hugging Face Transformers can help with this.

6.Define Responses:

Create a set of predefined responses for different problem-solving scenarios. You can use dictionaries or a response generation model if you have one.

7.Implement Logic:

Based on the problem domain, write the logic to analyze the user's query and generate a relevant response. This may involve using regular expressions, keyword matching, or more advanced NLP techniques.

8.User Interaction Loop:

Set up a loop that continually interacts with the user, receives their queries, processes them, and provides responses. You can use a while loop for this.