CREATE A CHATBOT IN PYTHON

PHASE 5 : DOCUMENTATION

PROBLEM STATEMENT:

Developing a chatbot using Python is a response to the growing demand for intelligent virtual assistants and automated customer support solutions. The problem at hand is the need for effective and efficient human-computer interaction in various domains, including e-commerce, customer service, healthcare, and more. Traditional customer support and information retrieval methods often require significant human intervention, leading to delays, inefficiencies, and increased operational costs. This creates a pressing challenge for businesses and organizations seeking to improve user experiences, enhance response times, and reduce the workload on human operators. By creating a chatbot, we aim to bridge this gap and provide a natural language interface that can understand and respond to user inquiries, perform tasks, and provide information in a conversational manner. The problem statement encompasses not only the technical aspects of natural language processing and machine learning but also the design and integration of the chatbot into existing systems and platforms. The goal is to develop an intelligent, versatile, and user-friendly chatbot that can address a wide range of user needs, ultimately leading to more efficient and satisfying interactions across various industries.

DESIGN THINKING:

Information Retrieval and FAQs:

Chatbots can answer frequently asked questions by providing userswith relevant information from a database or knowledge base. This is particularly useful for customer support and service-related queries**.**

Conversational Engagement:

Chatbots can engage users in natural language conversations, providinga more interactive and engaging experience. They can chat with users on topics ranging from general conversation to specific subjects**.**

Task Automation:

Chatbots can automate repetitive tasks and processes. For example, they can schedule appointments, set reminders, order products, and performvarious other tasks on behalf of users

Personal Assistance:

Virtual personal assistants like Siri and Google Assistant are examples ofchatbots that provide users with information, perform tasks, and interact withother apps on the user's device**.**

PHASES OF DEVELOPMENT:

Phase1: Problem Definition

Phase2: Innovation

Phase3: Development Part1

Phase4: Development Part2

CREATING DATASET :

In the context of a chatbot, a "dataset" refers to a structured collection of text or data that is used for various purposes, primarily for training and fine-tuning the chatbot's natural language processing (NLP) capabilities.Chatbot datasets typically consist of pairs or sequences of user messages and corresponding bot responses.

DATASET USED:

Dataset Link: https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot

```
hi, how are you doing?  i'm fine. how about yourself?
i'm fine. how about yourself?   i'm pretty good. thanks for asking.
i'm pretty good. thanks for asking.     no problem. so how have you been?
no problem. so how have you been?       i've been great. what about you?
i've been great. what about you?        i've been good. i'm in school right now.
i've been good. i'm in school right now.        what school do you go to?
what school do you go to?       i go to pcc.
i go to pcc.    do you like it there?
do you like it there?   it's okay. it's a really big campus.
it's okay. it's a really big campus.    good luck with school.
good luck with school.  thank you very much.
how's it going? i'm doing well. how about you?
i'm doing well. how about you?  never better, thanks.
never better, thanks.   so how have you been lately?
so how have you been lately?    i've actually been pretty good. you?
i've actually been pretty good. you?    i'm actually in school right now.
i'm actually in school right now.       which school do you attend?
which school do you attend?     i'm attending pcc right now.
i'm attending pcc right now.    are you enjoying it there?
are you enjoying it there?      it's not bad. there are a lot of people there.
it's not bad. there are a lot of people there.  good luck with that.
good luck with that.    thanks.
how are you doing today?        i'm doing great. what about you?
```

DATA PREPROCESSING STEPS:

Designing a chatbot project involves several steps, and preprocessing the dataset is one of the critical initial steps. Preprocessing ensures that your data is clean, structured, and ready for analysis and model training. Here's a step-by-step guide on how to preprocess a dataset for a chatbot project, including various analyses you may need to perform.

1. DATA COLLECTION:

Data collection for a chatbot refers to gathering, curating, and organizing the data needed to develop and train a chatbot. This data is crucial for enabling the chatbot to understand user input, generate appropriate responses, and continuously improve its performance. This could be text conversations, customer support logs, or any other relevant data.

2. **SEGMENTATION:**

Text segmentation, in the context of a chatbot or natural language processing (NLP), refers to the process of breaking down a continuous text or conversation into meaningful units or segments. These segments can help the chatbot better understand and respond to the user's input.

3. NORMALIZATION:

Text normalization in the context of a chatbot refers to the process of standardizing and transforming text input to make it more consistent and easier for the chatbot to process and understand. It involves various techniques aimed at cleaning and structuring text data.

4. **TOKENIZATION:**

Tokenization is a crucial natural language processing (NLP) technique used in chatbots and various other text-processing applications. It involves the process of breaking down a continuous text string, typically a sentence or paragraph, into individual units known as "tokens." Tokens are usually words or subwords, which are the basic building blocks of text that the chatbot can then analyze, manipulate, or respond to.

5. DATA ENCODING:

Convert text data into numerical representations using techniques like one-hot encoding or word embeddings (e.g., Word2Vec, GloVe).

6. PADDING SEQUENCES:

Ensure that text sequences have a uniform length by padding shorter sequences with zeros or truncating longer ones.

TECHNIQUE USED: NLP

NLP stands for Natural Language Processing, which is a subfield of artificial intelligence (AI) and linguistics. NLP focuses on the interaction between computers and human language.

Natural Language Processing (NLP) is not a single machine learning algorithm but rather a field within artificial intelligence (AI) that encompasses a wide range of machine learning and deep learning techniques and algorithms. Training a chat bot involves several NLP tasks.

MACHINE LEARNING ALGORITHM:

Sequence-to-Sequence (Seq2Seq) Models:

Seq2Seq models, often based on recurrent neural networks (RNNs) or transformer architectures, are commonly used for chatbot development. These models are designed to generate natural language responses given an input message or question.

Recurrent Neural Networks (RNNs):

RNNs, especially Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) variants, can be used for sequence generation and chatbot development. They are well-suited for handling sequential data.

MODEL TRAINING:

DATA COLLECTION AND LABELING:

Gather a diverse dataset of user queries or messages and corresponding responses. These conversations should cover a wide range of topics and intents.

Label the dataset with intents and entities, indicating the purpose of each user query and any specific parameters or entities mentioned.

PREPROCESSING AND DATA CLEANING:

Preprocess the text data by applying techniques such as tokenization, stopword removal, and lemmatization.

Handle data-specific noise, including special characters, spelling errors, and variations in writing style.

MODEL EVALUATION:

Intent Classification Accuracy:

Measure the chatbot's accuracy in correctly classifying user intents. Use a test dataset with labeled intents to assess how well the chatbot understands the purpose of user queries.

INNOVATIVE IDEA:

We are going to implement a personalized, conversational, intelligent, and integrated ordering chatbot:

1. Identify our business goals and customer needs:

What kind of ordering experience we want to provide our customers?

2. Choose a chatbot programming language.

We are building the chatbot by using a programming language Python.

3. Design the chatbot conversation flow.

This involves mapping out the different paths that a conversation which the chatbot might take.

4. Develop the chatbot.

This involves implementing the chatbot conversation flow and adding any necessary functionality.

5. Test the chatbot.

Once the chatbot is developed, we will test it thoroughly to ensure that it is working properly and that it can handle a variety of customer requests.

6. Deploy the chatbot.

Once the chatbot is tested and ready to go, we need to deploy it so that customers can start using it. This may involve integrating the chatbot with some messaging app that had been already present.

SCOPE FOR CHATBOT:

Develop chatbots that handle customer inquiries, provide support, and assist with common issues. Implement chatbots on websites, mobile apps, and social media platforms to improve customer service accessibility.

ROADMAP FOR IMPLEMENTATION:

Define the purpose and objectives: Identify specific objectives and goals that the chatbot should achieve.

Determine the target Audience: Understand the needs and preferences of the users who will interact with the chatbot.

Choosing the technology Stack: Select the programming language (python) and any relevant libraries or frameworks for chatbot development.

The Technology Stack: Select the programming language (python) and any relevant libraries or frameworks for chatbot development.

Integration: Implement API connections to access external data or services if necessary.

Testing and Quality Assurance: Implement a feedback loop for continuous implement. Deployment: Deploy the chatbot to be desired platforms or channels, ensuring it is accessible to user.

4.KEY FEATURES

Natural Language understanding (NLU): Chatbots use NLU techniques to comprehend and interpret user input, allowing them to understand and respond to natural language queries and commands.

Text-Based communication: Chatbots communicate with users through text messages, making them accessible via chat interfaces on.

Multi-platform supports: Chatbots can be deployed on multiple platforms, including websites, mobile apps, messaging apps (e.g., Facebook messenger and voice assistants (e.g., Amazon Alexa. Google Assistant)

IMPORTING LIBRARIES:

```
!pip install langchain
!pip install openai
!pip install gradio
!pip install huggingface_hub
```

```
Collecting langchain
  Downloading langchain-0.0.327-py3-none-any.whl (2.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 19.2 MB/s eta 0:00:00
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-packages (from langchain) (6.0.1)
Requirement already satisfied: SQLAlchemy<3,>=1.4 in /usr/local/lib/python3.10/dist-packages (from langchain) (2.0.22)
Requirement already satisfied: aiohttp<4.0.0,>=3.8.3 in /usr/local/lib/python3.10/dist-packages (from langchain) (3.8.6)
Requirement already satisfied: anyio<4.0 in /usr/local/lib/python3.10/dist-packages (from langchain) (3.7.1)
Requirement already satisfied: async-timeout<5.0.0,>=4.0.0 in /usr/local/lib/python3.10/dist-packages (from langchain) (4.0.3)
Collecting dataclasses-json<0.7,>=0.5.7 (from langchain)
  Downloading dataclasses_json-0.6.1-py3-none-any.whl (27 kB)
Collecting jsonpatch<2.0,>=1.33 (from langchain)
  Downloading jsonpatch-1.33-py2.py3-none-any.whl (12 kB)
Collecting langsmith<0.1.0,>=0.0.52 (from langchain)
  Downloading langsmith-0.0.56-py3-none-any.whl (44 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.4/44.4 kB 5.7 MB/s eta 0:00:00
Requirement already satisfied: numpy<2,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain) (1.23.5)
Requirement already satisfied: pydantic<3,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain) (1.10.13)
Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-packages (from langchain) (2.31.0)
Requirement already satisfied: tenacity<9.0.0,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain) (8.2.3)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (23.1.0)
Requirement already satisfied: charset-normalizer<4.0,>=2.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (3.3.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (1.9.2)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (1.4.0)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain) (1.3.1)
```

```
import os
import gradio as gr
from langchain.chat_models import ChatOpenAI
from langchain import LLMChain, PromptTemplate
from langchain.memory import ConversationBufferMemory

OPENAI_API_KEY="OPENAI_API_KEY"
os.environ["OPENAI_API_KEY"] = OPENAI_API_KEY
```

- Go to https://platform.openai.com/account/api-keys
- Create a new Secret Key
- Copy the Secret Key for your use.

```
template = """You are a helpful assistant to answer user queries.
{chat_history}
User: {user_message}
Chatbot:"""

prompt = PromptTemplate(
    input_variables=["chat_history", "user_message"], template=template
)

memory = ConversationBufferMemory(memory_key="chat_history")
```

- Similar to Open AI Mondel we can also use HuggingFace Transformer Models.
- Reference links:
  https://python.langchain.com/docs/integrations/providers/huggingface ,
  https://python.langchain.com/docs/integrations/llms/huggingface_hub.html

Hugging Face is a popular website and platform that is known for its contributions to the field of natural language processing (NLP) and machine learning. It was founded by Clément Delangue, Thomas Wolf, and Julien Chaumond in 2016. Hugging Face offers a variety of resources and tools for NLP practitioners, researchers, and developers.

```
#from langchain.llms import HuggingFacePipeline
#hf = HuggingFacePipeline.from_model_id(
#model_id="gpt2",
#task="text-generation",)
```

```
llm_chain = LLMChain(
    llm=ChatOpenAI(temperature='0.5', model_name="gpt-3.5-turbo"),
    prompt=prompt,
    verbose=True,
    memory=memory,
)
```

Langchain is a Python library that simplifies and enhances the process of working with natural language data and text processing. It offers a wide range of tools and functionalities for tasks such as text classification, sentiment analysis, named entity recognition, and more. Langchain leverages the power

of machine learning and deep learning techniques to provide accurate and efficient natural language processing capabilities. Developers and data scientists can easily integrate.

Langchain into their projects, making it an invaluable tool for tasks that involve understanding and working with human language. Whether it's building chatbots, conducting sentiment analysis on social media data, or extracting valuable insights from textual content, Langchain streamlines the development process and empowers users to harness the potential of natural language data in their applications.

```
def get_text_response(user_message,history):
    response = llm_chain.predict(user_message = user_message)
    return response



demo = gr.ChatInterface(get_text_response, examples= "How are you
doing?" "What are your interests?" "Which places do you like to
visit?"



if __name__ == "__main__":
    demo.launch() #To create a public link, set `share=True` in
`launch()`. To enable errors and logs, set `debug=True` in `launch()`.
```

```
from huggingface_hub import notebook_login

notebook_login()
```

Copy a token from your Hugging Face tokens page and paste it below.

Immediately click login after copying your token or it might be stored in plain text in this notebook file.

Token:

Add token as git credential?

Login

Pro Tip: If you don't already have one, you can create a dedicated 'notebooks' token with 'write' access, that you can then easily reuse for all notebooks.

```
from huggingface_hub import HfApi
api = HfApi()
```

```
HUGGING_FACE_REPO_ID = "<Hugging Face User Name/Repo Name>"
```

Adding Secret Variables in Hugging Face Account:

1. Open your Space
2. Click on Settings Button
3. Checkout to **Variables and secrets** section
4. Create New Secrets

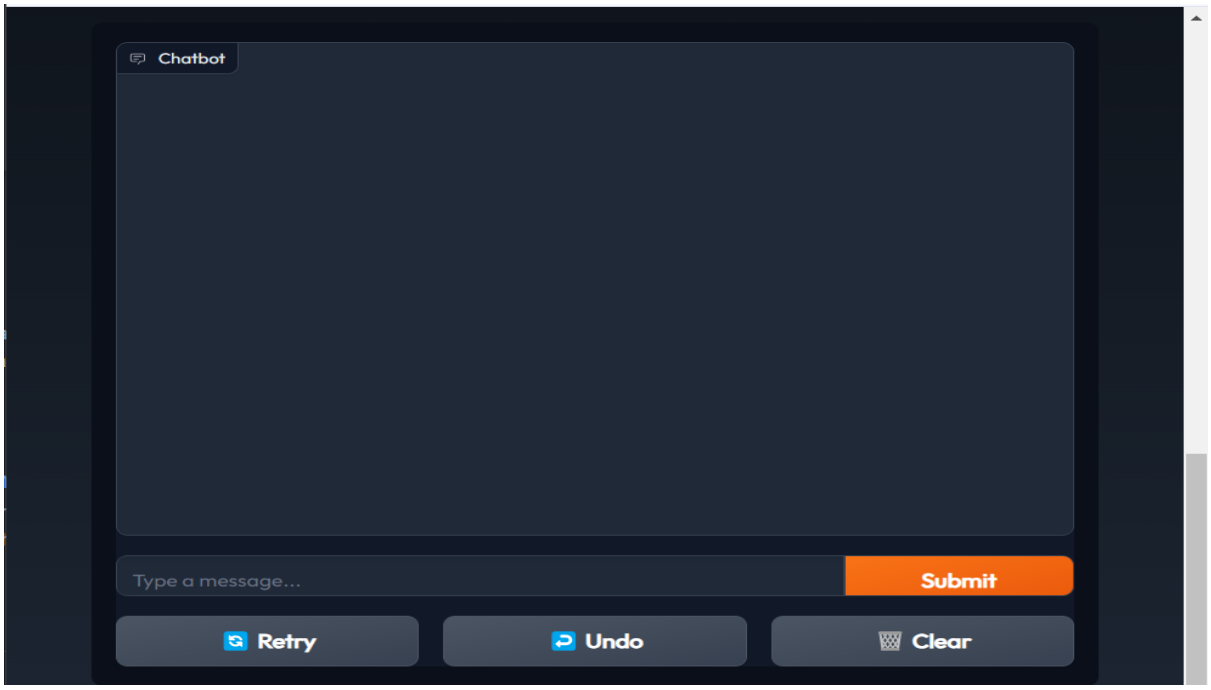
```
%mkdir /content/ChatBotWithOpenAI
!wget -P /content/ChatBotWithOpenAI/ https://s3.ap-south-1.amazonaws.com/cdn1.ccbp.in/GenAI-Workshop/ChatBotWithOpenAIAndLangChain/app.py
!wget -P /content/ChatBotWithOpenAI/ https://s3.ap-south-1.amazonaws.com/cdn1.ccbp.in/GenAI-Workshop/ChatBotWithOpenAIAndLangChain/requirements.txt
```

```
%cd /content/ChatBotWithOpenAI

api.upload_file(
    path_or_fileobj="./requirements.txt"
    path_in_repo="requirements.txt",
    repo_id=HUGGING_FACE_REPO_ID,
    repo_type="space")

api.upload_file(
    path_or_fileobj="./app.py",
    path_in_repo="app.py",
    repo_id=HUGGING_FACE_REPO_ID,
    repo_type="space")
```

OUTPUT:



The above picture is the visualization of my chat bot ,

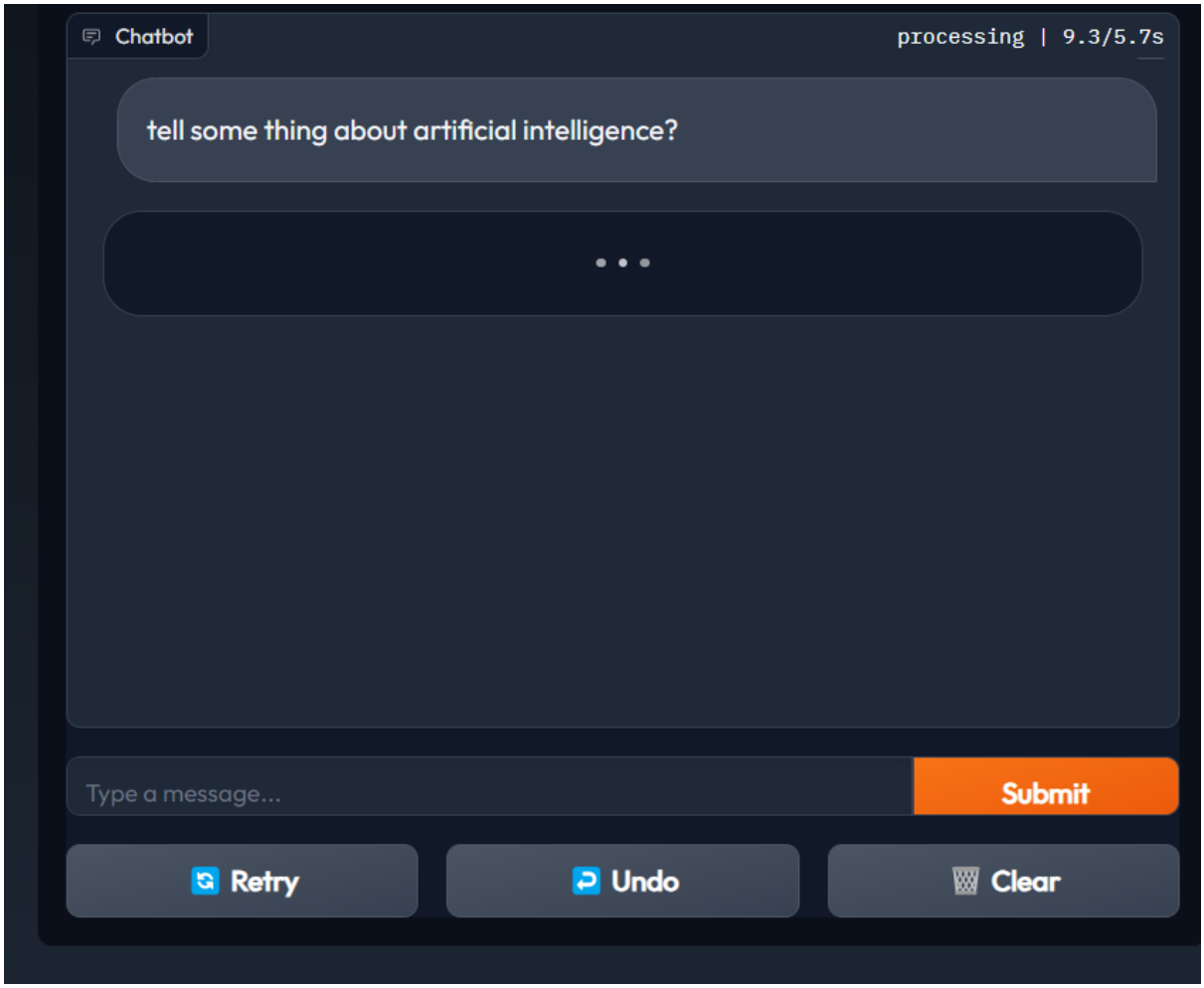
1.Start by typing your query or question in the chatbot's input field.

2.Make sure your query is clear and concise for the best results.

3.After entering your query, review it for accuracy and completeness.

4.Once you are satisfied with your input, click the "Submit" or "Send" button to send your query to the chatbot.

5.Wait for the chatbot's response, and be ready to engage in the conversation or follow any further instructions it provides**.**



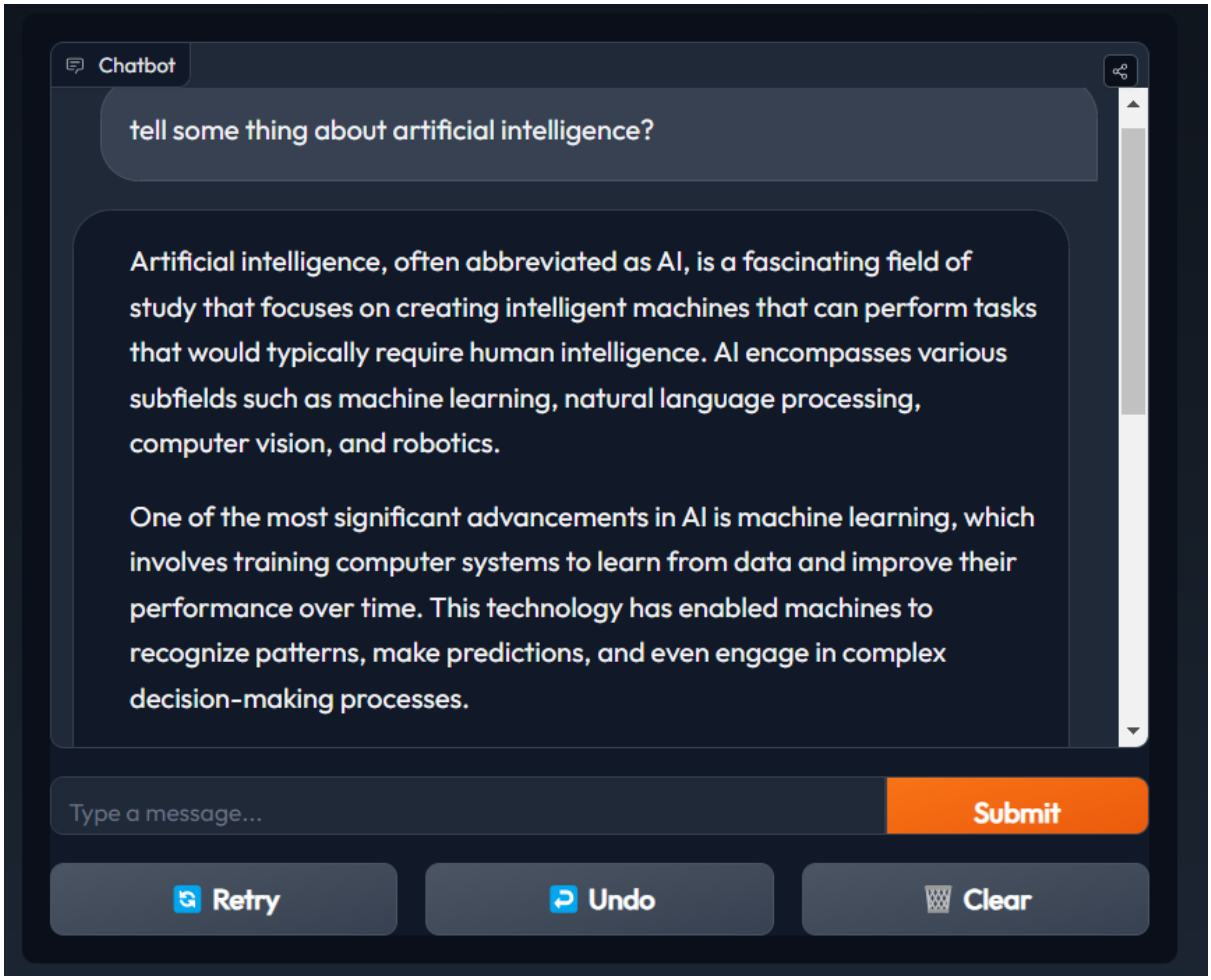
1.After sending your initial message, patiently wait for the chatbot's response.

2.Stay attentive to the chat and be ready to engage in the conversation.

3.Be open to following any additional instructions or prompts that the chatbot provides.

4.Respond promptly and clearly to ensure a smooth and efficient interaction.

5.Enjoy the conversation and make the most of your interaction with the chatbot.



When you ask a question to a chatbot, it typically follows a multi-step process to generate a response. First, it analyzes the input to understand the context and intent of the question. Then, it searches its knowledge base or database for relevant information. Next, it may use natural language processing algorithms to structure a coherent and grammatically correct response. The chatbot may also consider user-specific data or preferences if available. Finally, it generates and delivers the response, aiming to provide accurate and helpful information in a conversational manner.

CASE STUDY: REAL-WORLD IMPLEMENT

Company XYZ is an e-commerce platform that sells a wide range of products online. They have a growing customer base, and their customer support team is struggling to handle the increasing volume of customer inquiries and requests. To improve customer service efficiency and provide 24/7 support, Company XYZ decides to build a customer support chatbot.

Identify the common customer queries and FAQs. Determine the integration point with existing systems for order tracking and customer data. Gather customer support FAQs, product information and order tracking data. Annotate and preprocess the data for training the chatbot.

CONCLUSION:

In conclusion, generative AI chatbots represent a significant advancement in natural language processing and human-computer interaction. These chatbots have the ability to understand and generate human-like text, enabling more engaging and personalized conversations. They have found applications in customer support, content creation, and various industries, streamlining processes and improving user experiences. However, challenges remain, including the need to mitigate biases and improve the quality of responses. As these technologies continue to evolve, the potential for generative AI chatbots to revolutionize communication and information exchange is substantial, and their impact on various aspects of our lives is expected to grow significantly in the coming years. Ethical considerations and ongoing research will be crucial in ensuring their responsible and effective use.

By the end of phase 5, successfully completed outline the problem statement, design thinking process, and the phases of development.Describe the dataset used, data preprocessing steps, and feature extraction techniques. Explain the choice of machine learning algorithm, model training, and evaluationmetrics.Document any innovative techniques or approaches used during the development