NAAN MUDHALVAN

ARTIFICIAL INTELLIGENCE

PHASE 5

PROJECT TITLE

CREATE A CHATBOT IN PYTHON

NAME: RITHANYA P

DEPT: CSE

REG NO: 712221104017

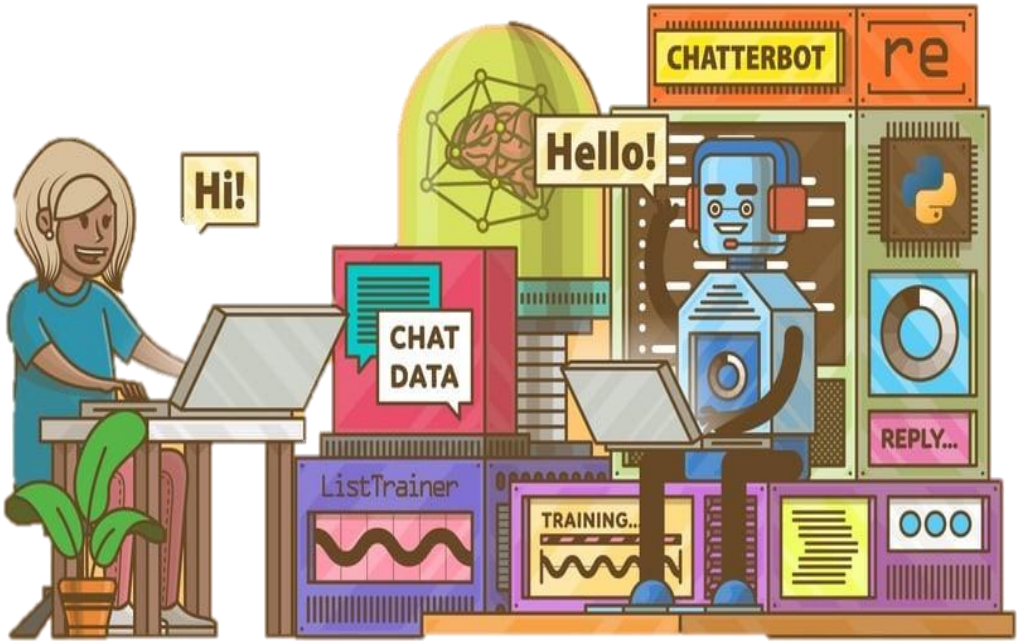
YEAR&SEM:3rd &5th

COLLEGE : PARK COLLEGE OF ENGINEERING AND TECHNOLOGY

PROBELEM DEFINITION AND DESIGN THINKING

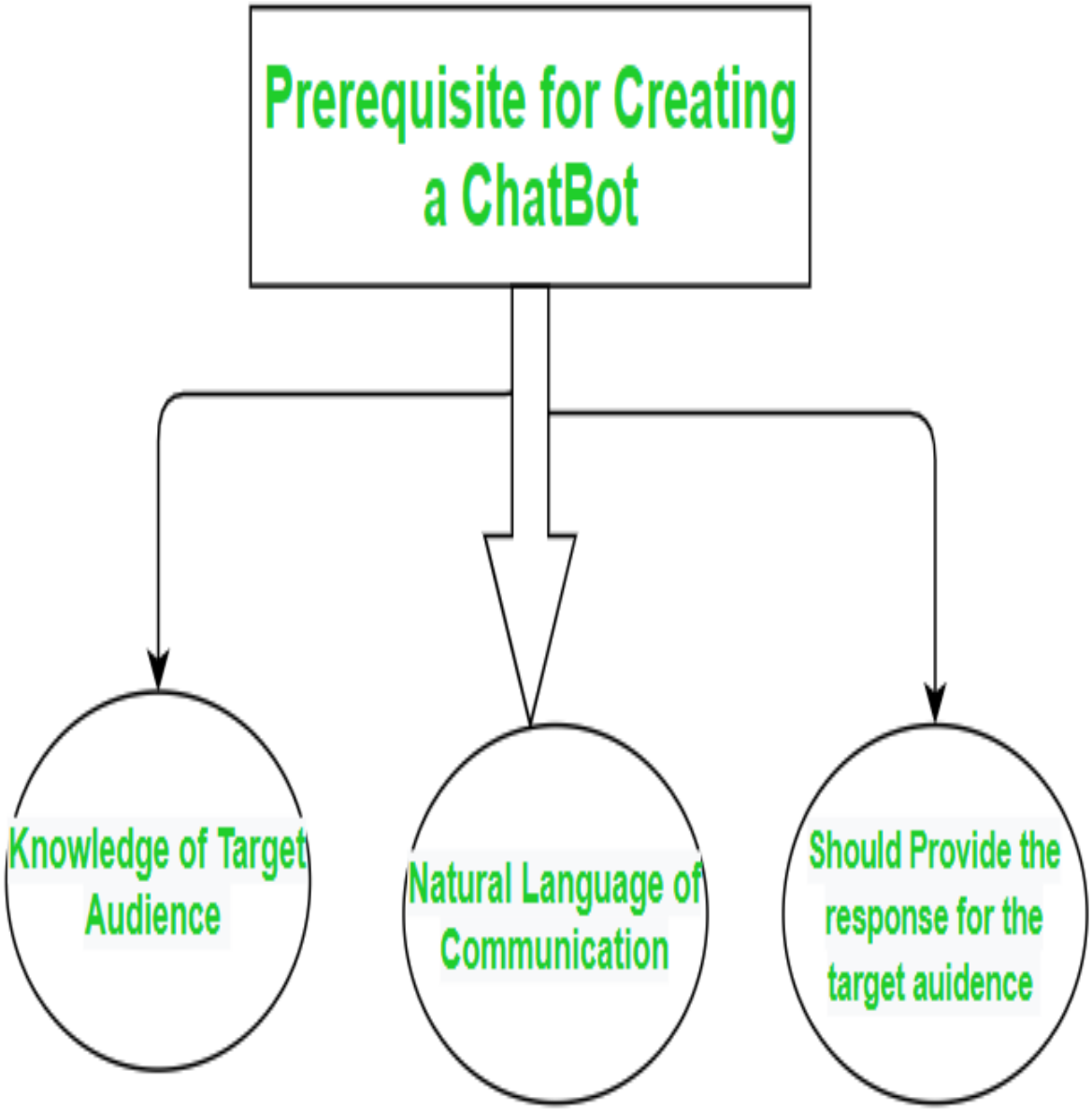
PROBLEM DEFINITION:

Chatbot Python has gained widespread attention from both technology and business sectors in the last few years. These smart robots are so capable of imitating natural human languages and talking to humans that companies in the various industrial sectors accept them. They have all harnessed this fun utility to drive business advantages, from, e.g., the digital commerce sector to healthcare institutions.



DESIGN THINKING:

Python is one of the best languages for building chatbots because of its ease of use, large libraries and high community support.



What is a Chatbot?

Artificial intelligence is used to construct a computer program known as "a chatbot" that simulates human chats with users. It employs a technique known as NLP to comprehend the user's inquiries and offer pertinent information. Chatbots have various functions in customer service, information retrieval, and personal support.

How Does the Chatbot Python Work?

1. Rule-Based Approach

The Chatbot Python adheres to predefined guidelines when it comprehends user questions and provides an answer. The developers often define these rules and must manually program them

2. Self-Learning Approach:

RetrievalBased Models: Based on an input question, these models can obtain predefined responses from a knowledge base. They evaluate user input and compare it to the closest equivalent response in the knowledge base.

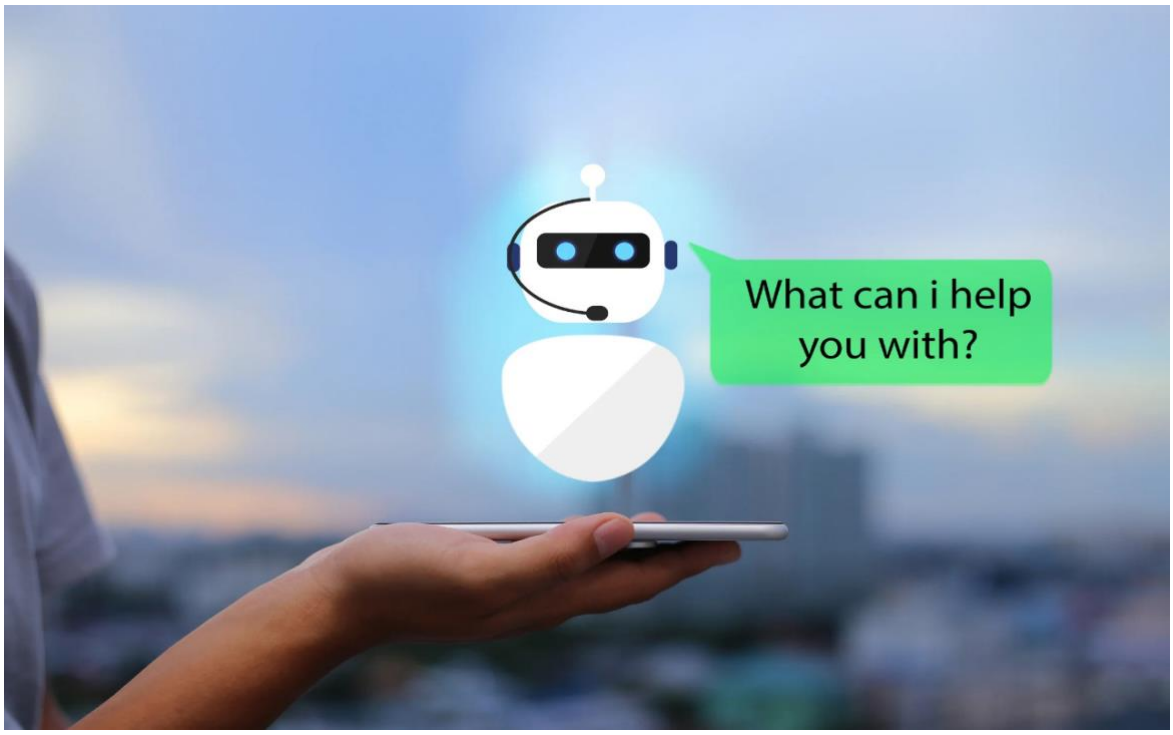
Generative Models: Generative models create responses from scratch based on the input query. They employ approaches like sequence-to-sequence models or transformers, for example, to produce human-like answers.

What is ChatterBot Library?

A Chatbot Python library called The ChatterBot makes it simpler to create chatbots. It manages the challenges of natural language processing and provides a specific API. The following are some of Chatterbot's primary features:

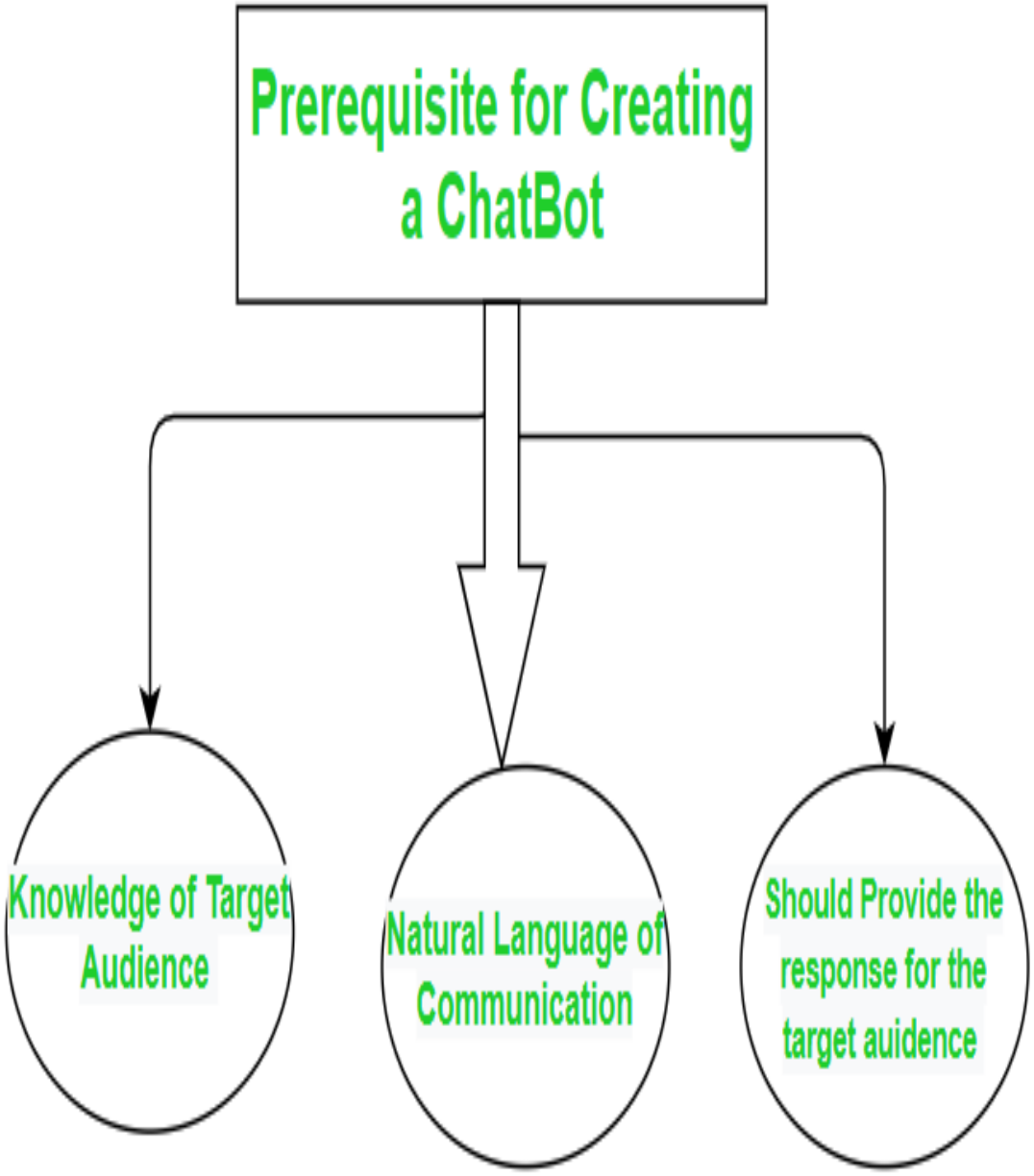
Limitations With A Chatbot:

Lack of semantic understanding: Chatbots may require assistance comprehending the discourse, which could result in misinterpretation or incorrect responses.



INNOVATION

DESIGN THINKING:



KNOWLEDGE OF TARGET AUDIENCE:

- Understanding your target audience is crucial when developing a chatbot in Python. To gather knowledge about your target audience.

Define Your Audience:

Identify who your chatbot is intended for. Consider demographics, interests, and needs.

Conduct Surveys or Research:

Collect data through surveys, user interviews, or market research to gain insights into your audience's preferences and pain points.

Create User Personas:

Develop fictional characters that represent different segments of your audience. This helps in visualizing and understanding their needs.

Analyze User Data:

If you have an existing platform or website, analyze user data to learn about user behavior and preferences.

User Testing:

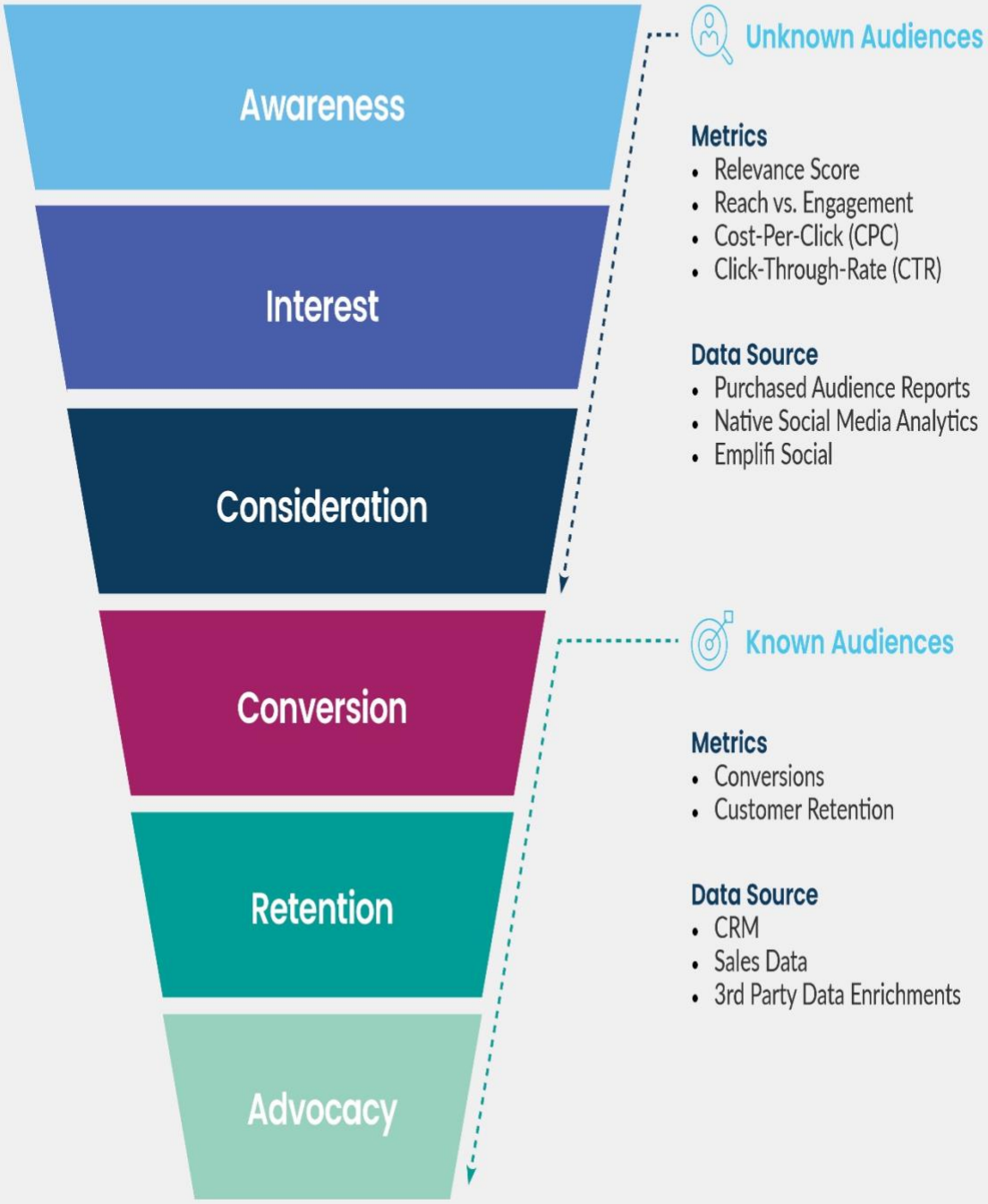
Involve real users in the development process to get feedback and adjust the chatbot accordingly.

Use Analytics Tools:

Implement analytics tools to track user interactions with your chatbot, allowing you to gather data on user behavior.

- Once you understand your target audience, you can tailor your chatbot's responses, tone, and features to better serve their needs and provide a more personalized experience.

How to understand audience types across the customer lifecycle



NATURAL LANGUAGE OF COMMUNICATION:

In natural language communication with a chatbot, the goal is to make the interaction between the user and the chatbot feel as human-like as possible. This involves using natural language, which means the chatbot should understand and generate responses in a way that is similar to how humans converse. Here are some key aspects of natural language communication in chatbots:

Natural language processing:

The chatbot should be able to understand and interpret the user's messages in a way that doesn't require users to use rigid, pre-defined commands.

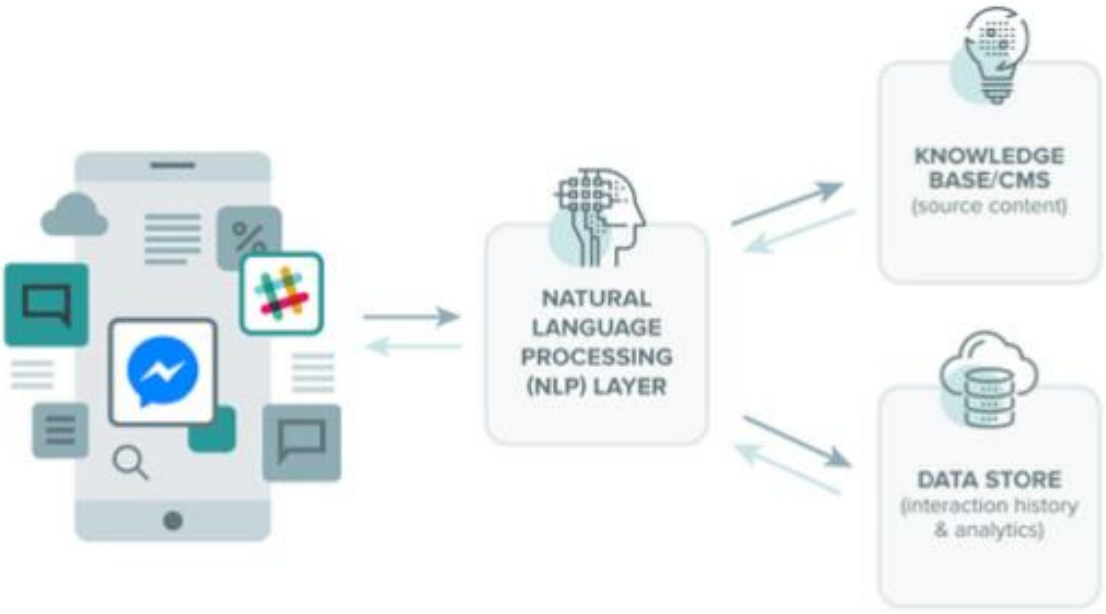
Natural language understanding (NLU) and natural language processing (NLP) techniques are used to process and comprehend user input.

SHOULD PROVIDE THE RESPONSE FOR THE TARGET AUDIENCE:

Absolutely, providing responses that are tailored to the specific target audience is a crucial aspect of creating an effective and user-centered chatbot. To do this, you should consider the following:

Audience Understanding:

Understand the demographics, preferences, and needs of your target audience. You may



CREATE A CHATBOT IN PYTHON

Introduction:

Creating a chatbot in Python can be a fun and educational project. A chatbot is a program that simulates conversation with users, providing responses to their inputs. Here's a basic introduction to get you started:

❖ Choose a Chatbot Type:

Decide what type of chatbot you want to create. Is it a rule-based chatbot that follows predefined rules, or an AI-powered chatbot that uses natural language processing (NLP) and machine learning to understand and respond to user inputs? For this introduction, we'll focus on a rule-based chatbot.

❖ Python Environment:

You'll need Python installed on your system. If you don't have it, download and install it from Python's official website.

❖ Select a Library:

For a rule-based chatbot you can use libraries like NLTK or spaCy Jifor text processing. For more advanced AI-powered chatbots, libraries like Rasa or Dialogflow are good choices.

❖ Data Preparation:

You'll need a dataset of predefined questions and responses. You can create this manually or use existing datasets. For instance, you can create a dictionary of questions and corresponding an

❖ Deployment:

You can deploy your chatbot on a website, integrate it with messaging platforms, or create a standalone application depending on your project's goals.

GIVEN DATASET:

	input	target
0	hi, how are you doing?	i'm fine. how about yourself?
1	i'm fine. how about yourself?	i'm pretty good. thanks for asking.
2	i'm pretty good. thanks for asking.	no problem. so how have you been?
3	no problem. so how have you been?	i've been great. what about you?
4	i've been great. what about you?	i've been good. i'm in school right now.

___Necessary step to follow:

1.Import libraries:

Step by importing the necessary libraries

Program:

```
import pandas as pd
import numpy as np
import string
from string import digits
import matplotlib.pyplot as plt
import seaborn as sns
import re
from sklearn.model_selection impor
t train_test_split
import tensorflow as tf
from keras.layers import Input, LS
TM, Embedding, Dense, Bidirectiona
l, Concatenate, Dot, Activation, T
imeDistributed
from keras.models import Model
from keras.utils import plot_model
```

2.Load the dataset:

Load your dataset into a pandas Dataframe.You Can typically find create a Chatbot in python dataset in CSV format ,but you can adapt code to other format as needed.

Program:

```
data_path = "/kaggle/input/simple-
dialogs-for-chatbot/dialogs.txt"
```

```
with open(data_path, 'r', encoding
='utf-8') as f:
    lines = f.read().split('\n')
inputs = []
targets = []
num_samples = 10000  # Number of s
amples to train on.
for line in lines[: min(num_sample
s, len(lines) - 1)]:

  input, target = line.split('\t')
  inputs.append(input)
  targets.append(target)
```

```
lines = pd.DataFrame({'input':inpu
ts, 'target':targets})
```

```
lines.shape
```

```
lines.head()
```

Challenge involved in loading preprocessing a create a Chatbot in python

Data set:

Data Collection: Gathering and curating a diverse and comprehensive dataset for training the chatbot is crucial. It may involve web scraping, using existing chat logs, or generating synthetic data.
Data Cleaning: The collected data often requires thorough cleaning to remove noise, irrelevant information, and potentially offensive content. This can be a time-consuming process.
Data Preprocessing: Tokenization: Breaking down text into tokens (words, phrases, or sentences).Stopword Removal: Eliminating common words that don't carry much meaning.Lemmatization or Stemming: Reducing words to their base or roo.

How to overcome the challenge involved in loading and preprocessing a create a Chatbot in python:

- ❖ Tokenization: Breaking down text into tokens (words, phrases, or sentences).
- ❖ Stopword Removal: Eliminating common words that don't carry much meaning.
- ❖ Lemmatization or Stemming: Reducing words to their base or root forms.
- ❖ Handling Spelling Errors: Dealing with misspelled words.
- ❖ Sentence Segmentation: Dividing text into sentences.

Loading the dataset

➢ Choose a Dataset:

Organize your dataset into a format suitable for training. Common formats are JSON, CSV, or plain text.

Typically, you'll need pairs of input (user messages) and output (bot responses) for training.

➢ Load the Dataset:

You can use Python's standard file I/O operations or a library like pandas (for CSV) or json (for JSON) to load your dataset into your code.

Program:

```
def cleanup(lines):
  # Since we work on word level, if we normalize the text to lower case, this will re
e the vocabulary. It's easy to recover the case later.
  lines.input=lines.input.apply(lambda x: x.lower())
  lines.target=lines.target.apply(lambda x: x.lower())

  # To help the model capture the word separations, mark the comma with special token
  lines.input=lines.input.apply(lambda x: re.sub("'", '', x)).apply(lambda x: re.su
(",", ' COMMA', x))
  lines.target=lines.target.apply(lambda x: re.sub("'", '', x)).apply(lambda x: re.
(",", ' COMMA', x))

  # Clean up punctuations and digits. Such special chars are common to both domains,
can just be copied with no error.
  exclude = set(string.punctuation)
  lines.input=lines.input.apply(lambda x: ''.join(ch for ch in x if ch not in exclu
e))
  lines.target=lines.target.apply(lambda x: ''.join(ch for ch in x if ch not in exc
e))

  remove_digits = str.maketrans('', '', digits)
  lines.input=lines.input.apply(lambda x: x.translate(remove_digits))
  lines.target=lines.target.apply(lambda x: x.translate(remove_digits))
```

```
st_tok = 'START_'
end_tok = '_END'
def data_prep(lines):
  cleanup(lines)
  lines.target = lines.target.apply(lambda x : st_tok + ' ' + x + ' ' + end_tok)
```

```
data_prep(lines)
```

```
lines.head()
```

Loading the dataset:

Output:

	input	target
0	hi COMMA how are you doing	START_ im fine how about yourself _END
1	im fine how about yourself	START_ im pretty good thanks for asking _END
2	im pretty good thanks for asking	START_ no problem so how have you been _END
3	no problem so how have you been	START_ ive been great what about you _END
4	ive been great what about you	START_ ive been good im in school right now _END

CREATE A CHATBOT IN PYTHON

Introduction:

In this part you will continue Building the project It continue Building the chatbot by integrating it into a web app using flask.

Building the chatbot by integrating it into a web app using flask:

Integrating a chatbot into a web app using Flask is a great idea! Here's a highlevel overview of the steps you can follow to achieve this:

Create a Flask Web App: Create a new Flask

web app by defining routes and views. You can set up a basic web page where the chatbot will be embedded.

Choose a Chatbot Platform: Decide on the chatbot platform you want to use. You can use pre-built chatbot services like Dialogflow, Wit.ai, or build a custom chatbot using

libraries like ChatterBot or Rasa.

Integrate the Chatbot: For a simple integration, you can embed the chatbot as a chat interface within an HTML template. Use JavaScript to handle user interactions and

send/receive messages from the chatbot.Handle User Input: In your Flask app, create a route or endpoint to handle user input sent from the chat interface. Parse the user's message and send it to the chatbot for processing.

Chatbot Responses:

Once the chatbot processes the user's message, it will generate a response. Send the response back to the web page and display it in the chat interface.User Authentication (Optional): If your web app requires user authentication, make sure to implement it so that the chatbot can provide personalized responses based on the user's identity.

Styling and Customization:

Customize the appearance and behavior of the chat interface

to match your web app's design.

Continuous Improvement:

Continuously improve your chatbot's capabilities and responses based on user feedback and usage analytics.

PROGRAM:

```
Print("Hello, I'm a Chatbot \n")
User_Name = input("What is your name? ")
Print("How are you {0}.
\n".format(User_Name))
User_Age = input("What is your age? ")
Print("Oh, so your age is {0}.
\n".format(User_Age))
User_Job = input("What is your job profile? ")
Print("So you're a {0}. \n".format(User_Job))
```

OUTPUT:

```
Hello, I'm a Chatbot
What is your name? pythonscholar
How are you pythonscholar.
What is your age? 26
Oh, so your age is 26.
What is your job profile? Python developer
So you're a python developer.
```

CONCLUSION:

➢ This following projects contains the it continues building the chatbot by integrating it into a web app using flask

➢ It conclude the project with the create a chatbot in python

THANK YOU!!