PHASE -V

**CREATE A CHATBOT USING PYTHON**

**Problem Definition and Design Thinking:**

When using an app or website, customers expect outstanding service. They can become disinterested in the app if they can't locate the solution to a question they have. To avoid losing customers and having an adverse effect on your bottom line, you must provide the highest quality service possible while developing a website or application.

**IMPORT LIBRARIES**

We need to import all the libraries which required for the project

**from sklearn.model\_selection import train\_test\_split**

**import unicodedata**

**import re**

**import numpy as np**

**import warnings**

**warnings.filterwarnings('ignore')**

**Read the dataset**

Depending on the format of your dataset, use appropriate libraries or functions in Python (such as pandas for CSV or Excel files) to load the data into your programming environment.

**df=open('/content/dialogs1.txt','r').read()**

**print(df)**

**SAMPLE OUTPUT**

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?

i'm doing great. what about you? i'm absolutely lovely, thank you.

i'm absolutely lovely, thank you. everything's been good with you?

everything's been good with you? i haven't been better. how about yourself?

i haven't been better. how about yourself? i started school recently.

i started school recently. where are you going to school?

where are you going to school? i'm going to pcc.

i'm going to pcc. how do you like it so far?

how do you like it so far? i like it so far. my classes are pretty good right now.

i like it so far. my classes are pretty good right now. i wish you luck.

it's an ugly day today. i know. i think it may rain.

i know. i think it may rain. it's the middle of summer, it shouldn't rain today.

it's the middle of summer, it shouldn't rain today. that would be weird.

that would be weird. yeah, especially since it's ninety degrees outside.

yeah, especially since it's ninety degrees outside. i know, it would be horrible if it rained and it was hot outside.

i know, it would be horrible if it rained and it was hot outside. yes, it would be.

yes, it would be. i really wish it wasn't so hot every day………….

**Data preprocessing**

**Lowercasing**

**Tokenization**

**Removing Special Characters and Punctuation**

**Removing Stop Words**

**import nltk**

**from nltk.tokenize import word\_tokenize**

**from nltk.corpus import stopwords**

**import string**

**nltk.download('punkt')**

**nltk.download('stopwords')**

**def normalize\_text(text)**

**# Convert to lowercase**

**text = text.lower()**

**# Tokenize the text**

**tokens = word\_tokenize(text)**

**sentences = nltk.sent\_tokenize(text)**

**# Remove punctuation and special characters**

**tokens = [word for word in tokens if word.isalnum()]**

**# Remove stopwords**

**stop\_words = set(stopwords.words('english'))**

**tokens = [word for word in tokens if word not in stop\_words]**

**# Join the tokens back into a normalized string**

**normalized\_text = ' '.join(tokens)**

**return normalized\_text**

**# Example usage**

**input\_text = open('/content/dialogs1.txt','r').read()**

**normalized\_text = normalize\_text(input\_text)**

**print("Normalized Text:", normalized\_text)**

**OUTPUT**

**Normalized Text:** hi fine fine pretty good thanks asking pretty good thanks asking problem problem great great good school right good school right school go school go go pcc go pcc like like okay really big campus okay really big campus good luck school good luck school thank much going well well never better thanks never better thanks lately lately actually pretty good actually pretty good actually school right actually school right school attend school attend attending pcc right attending pcc right enjoying enjoying bad lot people bad lot people good luck good luck thanks today great great absolutely lovely thank absolutely lovely thank everything good everything good better better started school recently started school recently going school going school going pcc going pcc like ……..

**Stemming and Lemmatization**

**import nltk**

**from nltk.stem import PorterStemmer**

**from nltk.stem import WordNetLemmatizer**

**# Download NLTK resources (only needed once)**

**nltk.download('punkt')**

**nltk.download('wordnet')**

**# Example text**

**text = open('/content/dialogs1.txt','r').read()**

**# Tokenize the text**

**words = nltk.word\_tokenize(text)**

**# Perform Stemming**

**stemmer = PorterStemmer()**

**stemmed\_words = [stemmer.stem(word) for word in words]**

**print("Stemmed Words:")**

**print(stemmed\_words)**

**# Perform Lemmatization**

**lemmatizer = WordNetLemmatizer()**

**lemmatized\_words = [lemmatizer.lemmatize(word) for word in words]**

**print("\nLemmatized Words:")**

**print(lemmatized\_words)**

**OUTPUT:**

**Stemmed and Lemmatized Words:**

['hi', ',', 'how', 'are', 'you', 'do', '?', 'i', "'m", 'fine', '.', 'how', 'about', 'yourself', '?', 'i', "'m", 'fine', '.', 'how', 'about', 'yourself', '?', 'i', "'m", 'pretti', 'good', '.', 'thank', 'for', 'ask', '.', 'i', "'m", 'pretti', 'good', '.', 'thank', 'for', 'ask', '.', 'no', 'problem', '.', 'so', 'how', 'have', 'you', 'been', '?', 'no', 'problem……]

**Entity Recognition**

**import spacy**

**# Load the English language model**

**nlp = spacy.load("en\_core\_web\_sm")**

**# Example text for entity recognition**

**text = open('/content/dialogs1.txt','r').read()**

**# Process the text using spaCy**

**doc = nlp(text)**

**# Extract and print entities**

**entities = [(ent.text, ent.label\_) for ent in doc.ents]**

**print("Entities in the text:")**

**for entity, label in entities:**

**print(f"Entity: {entity}, Label: {label}")**

**BUILDING MODEL**

Using Machine Learning algorithm we can build the chatbot

For building the chatbot We convert the text data into csv data then we apply the scikit-learn library for text vectorization and a simple Linear Support Vector Machine (SVM) model for training the chatbot

**CODE**

**import pandas as pd**

**from sklearn.feature\_extraction.text import TfidfVectorizer**

**from sklearn.svm import LinearSVC**

**#Load the data and prepare the dataset**

**df = pd.read\_csv('/content/dialogs .csv')**

**questions = data['question'].tolist()**

**answers = data['answer'].tolist()**

**#Vectorize the text data for the question**

**vectorizer = TfidfVectorizer()**

**X = vectorizer.fit\_transform(questions)**

**y = answers**

**#Train the chatbot using Linear support vector Machine**

**model = LinearSVC()**

**model.fit(X, y)**

**#Implement the chatbot functionality**

**def chatbot\_response(user\_input):**

**input\_vector = vectorizer.transform([user\_input])**

**predicted\_answer = model.predict(input\_vector)**

**return predicted\_answer[0]**

**#Test the chatbot**

**print("Chatbot: Hello! Ask me a question (type 'exit' to end)")**

**while True:**

**user\_input = input("User: ")**

**if user\_input.lower() in ['exit', 'quit', 'bye', 'goodbye']:**

**print("Chatbot: Goodbye! Have a great day!")**

**break**

**response = chatbot\_response(user\_input)**

**print("Chatbot:", response)**

**OUTPUT:**

Chatbot: Hello! Ask me a question (type 'exit' to end)

User: hi, how are you doing?

Chatbot: i'm fine. how about yourself?

User: i'm pretty good. thanks for asking.

Chatbot: no problem. so how have you been?

User: i've been great. what about you?

Chatbot: i've been good. i'm in school right now.

User: what school do you go to?

Chatbot: i go to pcc.

User: do you like it there?

Chatbot: it's okay. it's a really big campus.

User: bye

Chatbot: Goodbye! Have a great day!

**INTEGRATING CHATBOT WITH FLASK**

**CODE**

**from flask import Flask, render\_template, request**

**import pandas as pd**

**from sklearn.feature\_extraction.text import TfidfVectorizer**

**from sklearn.svm import LinearSVC**

**app = Flask(\_\_name\_\_)**

**data = pd.read\_csv('your\_dataset.csv')e**

**questions = data['Question'].tolist()**

**answers = data['Answer'].tolist()**

**vectorizer = TfidfVectorizer()**

**X = vectorizer.fit\_transform(questions)**

**y = answers**

**model = LinearSVC()**

**model.fit(X, y)**

**def chatbot\_response(user\_input):**

**input\_vector = vectorizer.transform([user\_input])**

**predicted\_answer = model.predict(input\_vector)**

**return predicted\_answer[0]**

**# Flask Routes**

**@app.route('/')**

**def index():**

**return render\_template('index.html')**

**@app.route('/get\_response', methods=['POST'])**

**def get\_response():**

**user\_input = request.form['user\_input']**

**response = chatbot\_response(user\_input)**

**return render\_template('index.html',user\_input=user\_input, bot\_response=response)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**app.run(debug=True)**

**INDEX.HTML CODE**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Chatbot</title>**

**</head>**

**<body>**

**<h1>Chatbot</h1>**

**<form action="/get\_response" method="post">**

**<label for="user\_input">You:</label>**

**<input type="text" id="user\_input" name="user\_input" required>**

**<input type="submit" value="Send">**

**</form>**

**<div>**

**<p>You: {{ user\_input }}</p>**

**<p>Chatbot: {{ bot\_response }}</p>**

**</div>**

**</body>**

**</html>**

**CONCLUSION**

In conclusion, a well-thought-out and properly built chatbot can be a valuable tool for companies and organizations. The use of chatbots has the potential to improve customer service, expedite procedures, and give customers useful information in a conversational setting. Planning the objective, responses, and integration of the chatbot with user wants and preferences in mind is essential to maximizing their impact. A high-quality chatbot must undergo continuous improvement to remain relevant and respond to changing user needs. This progress must be driven by user feedback and testing. Chatbots can be effective tools for offering users quick and interesting interactions if they are used with the appropriate approach and a constant dedication to improvement.