CHATBOT USING PYTHON

TEAM MEMBER

Phase-1 : Problem Definition and Design Thinking

Project Title: *chatbot using python*

OBJECTIVE:

The objective of this project is to develop CHATBOT

- ## Data Source:

The data source for this project is collected from www.Kaggle.com, which containing information simple dailogues

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.
```

Python program:

To create a chatbot in Python that delivers exceptional customer service and provides high-quality support to users in a web application, you can follow these steps:

1. **Download the CSV file**:

Start by downloading the CSV file from the provided Kaggle link. This file contains a collection of simple dialogs for a chatbot.

2. **Install the necessary libraries**: Install the required libraries for creating a chatbot in Python. You can use NLTK and ChatterBot for this purpose. Install them using pip:

Python:

```
pip install nltk
pip install chatterbot
```

3. **Import the necessary modules**:

Import the required modules into your Python script:

Python

```
import csv
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from chatterbot import ChatBot
from chatterbot.trainers import ChatterBotCorpusTrainer
```

4. **Preprocess the dialogs**:

Before training the chatbot, preprocess the dialogs to remove unnecessary elements such as stopwords and punctuations. You can use NLTK for this purpose:

Python:

```
nltk.download('punkt')
nltk.download('stopwords')

def preprocess_text(text):
    # Tokenize the text
```

```
    tokens = word_tokenize(text.lower())

    # Remove stopwords and punctuations
    stop_words = set(stopwords.words('english'))
    preprocess_tokens = [token for token in tokens if token.isalnum() and token not in stop_words]

    # Convert tokens back to text
    preprocess_text = ' '.join(preprocess_tokens)

    return preprocess_text

def preprocess_dialogs(dialogs):
    preprocessed_dialogs = []
    for dialog in dialogs:
        user_input = dialog[0]
        bot_response = dialog[1]

        preprocessed_user_input = preprocess_text(user_input)
        preprocessed_bot_response = preprocess_text(bot_response)

        preprocessed_dialogs.append([preprocessed_user_input, preprocessed_bot_response])

    return preprocessed_dialogs
```

5. **Read and preprocess the CSV file:**

Read the CSV file and preprocess the dialogs using the previously defined function:

Python:

```
dialogs = []
with open(' https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot ', 'r') as file:
    csv_reader = csv.reader(file)
    next(csv_reader)  # Skip the header row
    for row in csv_reader:
        user_input = row[0]
        bot_response = row[1]
        dialogs.append([user_input, bot_response])

preprocessed_dialogs = preprocess_dialogs(dialogs)
```

Make sure to replace 'path_to_csv_file' with the actual path to the downloaded CSV file.

6. **Train the chatbot:**

Use the preprocessed dialogs to train the chatbot using the ChatterBot library:

Python:

```
chatbot = ChatBot('Customer Support Bot')
trainer = ChatterBotCorpusTrainer(chatbot)

# Train the chatbot using the preprocessed dialogs
trainer.train(preprocessed_dialogs)
```

7. **Integrate the chatbot into your web application:**

You can integrate the chatbot into your web application using a web framework such as Flask or Django. Here's an example using Flask:

Python:

```
from flask import Flask, request, render_template

app = Flask(__name__)
chatbot = ChatBot('Customer Support Bot')

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/get_response', methods=['POST'])
def get_response():
    user_input = request.form['user_input']
    preprocessed_input = preprocess_text(user_input)
    response = chatbot.get_response(preprocessed_input)
    return str(response)

if __name__ == '__main__':
    app.run()
```

In this example, the chatbot is integrated into a Flask web application. The /get_response route handles the user's input, preprocesses it, gets the chatbot's response, and returns it as a response to the user.

8. **Create a front-end for the web application:**

Create HTML templates and CSS stylesheets to create a user-friendly front-end for your web application. You can use frameworks like Bootstrap to make it responsive and visually appealing.

Html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Chatbot Demo</title>
</head>
<body>
    <h1>Chatbot Demo</h1>
    <form action="/get_bot_response" method="post">
        <input type="text" name="user_query" placeholder="Enter your query">
        <input type="submit" value="Submit">
    </form>
    <div id="bot_response"></div>
</body>
</html>
```

9. Deploy the web application:

Deploy your web application to a server or a hosting platform so that users can access it over the internet.

10. Running the Web Application:

Run the web application by executing the app.py file. Open a web browser and go to http://localhost:5000 to see the chatbot interface.

```
python -m notebook chatbot.ipynb
```

The web application will display a chatbot interface with an input field. When the user enters a query and submits it, the chatbot will use the trained model to provide a response, which will be displayed below the input field.

Conclusion:

This is a basic implementation of a web application with a chatbot using the provided CSV file. You can enhance it further by adding more features, improving the user interface, or deploying it to a web server.