NAME: TEJA SINGURU

COURSE:PYTHON(3MONTHS)INTERNSHIP

DEVTERN CHATBOT PYTHON PROJECT

INTRODUCTION :

1. OBJECTIVE OF THE TASK
    - Develop an interactive system that understands user input and responds appropriately.
    - Automate customer support or engagement services.
    - Provide conversational interfaces to enhance user experience.
    - Implement machine learning or NLP to improve chatbot intelligence
    - Offer personalized responses based on user history or preferences
2. RELEVANCE
    - High demand for automation in customer service, sales, and marketing.
    - Python's simplicity and versatility make chatbot development efficient.
    - Chatbots enhance user experience by providing interactive, real-time responses.
    - Easy integration with AI and NLP technologies for intelligent conversations.
    - Chatbots improve business efficiency by handling routine queries.

TASKDESCRIPTION :

3. PROBLEM STATEMENT
    - Develop a Python-based chatbot for interactive, conversational user engagement.
    - Automate responses to user queries, reducing manual intervention.
    - Implement NLP for understanding and generating human-like responses.
    - Optionally offer personalized responses based on user data
    - Enhance user experience and business efficiency.
4. . REQUIREMENTS
    - Use Python with NLP libraries (NLTK, etc.,) for response generation
    - Provide a simple, scalable user interface (text-based or platform-integrated).
    - Ensure fast response time and handle unexpected input gracefully.

- Test thoroughly for performance and effectiveness.
- Optional features: multi-language support, AI integration, and machine learning.

METHODOLOGY:

5. APPROACH

- Define the chatbot's scope and design conversation flow using decision trees.
- Integrate NLP for understanding user inputs and intents.
- Implement the chatbot with Python, starting simple and gradually adding complexity.
- Test and refine the chatbot with real-world inputs and edge cases.
- Deploy on a web platform or messaging app, and monitor performance for improvements.

6. TOOLS & TECHNOLOGIES

- **Python** as the primary programming language.
- NLP libraries: **NLTK**, **SpaCy**, or **Transformers** for text processing.
- Machine learning frameworks: **Scikit-learn**, **TensorFlow** (optional).
- Conversation management: **Rasa** or **Dialogflow** for handling conversation flows.
- Deployment on platforms like **Flask**, **Telegram API**, or cloud services (e.g., **Heroku**, **AWS**)

IMPLEMENTATION:

7.CODE/ALGORITHM DETAILS :

```
import os

import nltk

import ssl

import streamlit as st

import random

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.linear_model import LogisticRegression
```

```
ssl._create_default_https_context = ssl._create_unverified_context
nltk.data.path.append(os.path.abspath("nltk_data"))
nltk.download('punkt')

intents = [
  {
    "tag": "greeting",
    "patterns": ["Hi", "Hello", "Hey", "How are you", "What's up"],
    "responses": ["Hi there", "Hello", "Hey", "I'm fine, thank you", "Nothing
much"]
  },
  {
    "tag": "goodbye",
    "patterns": ["Bye", "See you later", "Goodbye", "Take care"],
    "responses": ["Goodbye", "See you later", "Take care"]
  },
  {
    "tag": "thanks",
    "patterns": ["Thank you", "Thanks", "Thanks a lot", "I appreciate it"],
    "responses": ["You're welcome", "No problem", "Glad I could help"]
  },
  {
    "tag": "about",
        "patterns": ["What can you do", "Who are you", "What are you", "What is
your purpose"],

        "responses": ["I am a chatbot", "My purpose is to assist you", "I can
answer questions and provide assistance"]

    },

    {

        "tag": "help",

        "patterns": ["Help", "I need help", "Can you help me", " What shoul I do"],

        "responses": ["Sure, what do you need help with?", "I'm here to help.
What's the problem?", "How can I assist you?"]

    },

    {

        "tag": "age",

        "patterns": ["How old are you", "What's your age"],

        "responses":["I don't have an age. I'm a chatbot."," I was just born in th
digital word.", "Age is just a number for me."]

    },

    {

        "tag":"weather",

        "patterns": ["What's the weather like", "How's the weather today"],

        "responses": ["I'm sorry, I cannot provide real-time weather information.",
"You can check the Wheather app or Website."]

    },

    {

        "tag":"budjet",
        "patterns": ["How can I make a budget ", "What's a good budgeting strategy", "How do i create a budget"],
        "responses": ["To make a budget, start by tracking your income and expenses .Then, allocate your income towards essntial expenses by categorizing them into fixed (like rent) and variable (like groceries). Subtract your total expenses from your income to see how much you can save or spend. Adjust as necessary to ensure you’re living within your means.", "One effective budgeting strategy is the 50/30/20 rule: allocate 50% of your income to needs (essentials), 30% to wants (discretionary spending), and 20% to savings and debt repayment. This method helps you balance spending and saving, making it easier to manage your finances.", "To create a budget, follow these steps:Determine your income: Include all sources, like salary, freelance work, or investments.List your expenses: Categorize them into fixed and variable costs.Set financial goals: Decide what you want to achieve, such as saving for a vacation or paying off debt.Create your budget: Allocate your income to cover expenses, savings, and goals.Review and adjust: Regularly check your budget and make adjustments based on your spending habits and financial goals. "]
    },
]


vectorizer = TfidVectorizer()
clf = LogisticRegression(random_state=0, max_itern=10000)


#Preprocess the data
tags = []
patterns = []
for intent in intents:
    for pattern in intent['patterns']:
        tags.append(intent['tag'])
        patterns.append(pattern)


#Train the model
x= vectorizer.fit_transform(patterns)
clf.fit(x, tags)


def chatbot(user_input):
    user_input_vector = vectorizer.transform([user_input])
    predicted_tag = clf.predict(user_input_vector) [0]


    for intent in intents:
        if intent['tag'] == predicted_tag:
            response =  random.choice(intent['responses'])
            return response
    return "I'm sorry, I didn't understand that."


counter  = 0


def main():
    global counter
    st.title("Chatbot")
    st.write("Welcome to the chatbot. Please type a message and press Enter to
start the conversation.")


    counter += 1
    user_input = st.text_input("You:", key=f"user_input_{counter}")


    if user_input:
        response = chatbot(user_input)
        st.text_area("Chatbot:", value = response, height=100, max_chars=None,
key=f"chatbot_response_{counter}")


        if response.lower() in ['goodbye', 'bye']:
            st.write("Thank you for chatting with me. Have a great day!")
            st.stop()


if __name__ == '__main__':
    main()
```

EXPALNATION OF THE ABOVE CODE :

1. Importing Libraries and Setting Up:

- **Libraries Imported:**
  - `os`, `nltk`, `ssl`, `streamlit as st`, `random` for various utilities.
  - `TfidfVectorizer` and `LogisticRegression` from `sklearn` for text processing and classification.
- **SSL Configuration:**
  - Adjusts SSL settings to avoid verification issues when downloading NLTK data.
- **NLTK Setup:**
  - Specifies the path for NLTK data and downloads the 'punkt' tokenizer.

2.Defining Intents:

- **Intents Structure:**

  - A list of dictionaries where each dictionary represents an intent with:
    - `tag`: The intent name (e.g., "greeting").
    - `patterns`: Possible user inputs for that intent.
    - `responses`: Possible chatbot replies for that intent.

- **Example Intents:**

  - Greetings, goodbyes, thanks, about, help, age, weather, and budgeting.

3.Preparing the Data for Training:

- **Data Extraction:**

  - Extracts all `patterns` and corresponding `tags` from the intents.

- **Vectorization:**

  - Uses `TfidfVectorizer` to convert text patterns into numerical vectors suitable for machine learning.

- **Model Training:**

  - Trains a `LogisticRegression` classifier on the vectorized patterns and their tags to predict intent based on user input

    .

4.Defining the Chatbot Function:

Function `chatbot(user_input)`:

- **Vectorizes Input:** Converts the user's input text into the same TF-IDF vector space.
- **Predicts Intent:** Uses the trained classifier to determine the intent tag of the input.
- **Generates Response:** Selects a random response from the matched intent's responses.
- **Fallback:** Returns a default message if the intent isn't recognized.

5. Building the Streamlit Interface:

- **Function `main()`:**

  - **Title and Welcome Message:** Displays the chatbot title and instructions.
  - **User Input:** Provides a text input box for the user to type messages.

- **Display Response:**
    - When the user submits input, the chatbot function generates a response.
    - The response is shown in a text area below the user input.
- **Ending Conversation:**
    - If the response is a goodbye message, it thanks the user and stops the app.

- **Execution:**

    - Runs the `main()` function when the script is executed directly.

6. Handling User Sessions:

Counter Variable:

- Uses a `counter` to create unique keys for multiple user inputs and responses, ensuring each interaction is tracked separately in Streamlit.

7. Optional Enhancements:

- **Personalization & Context:**

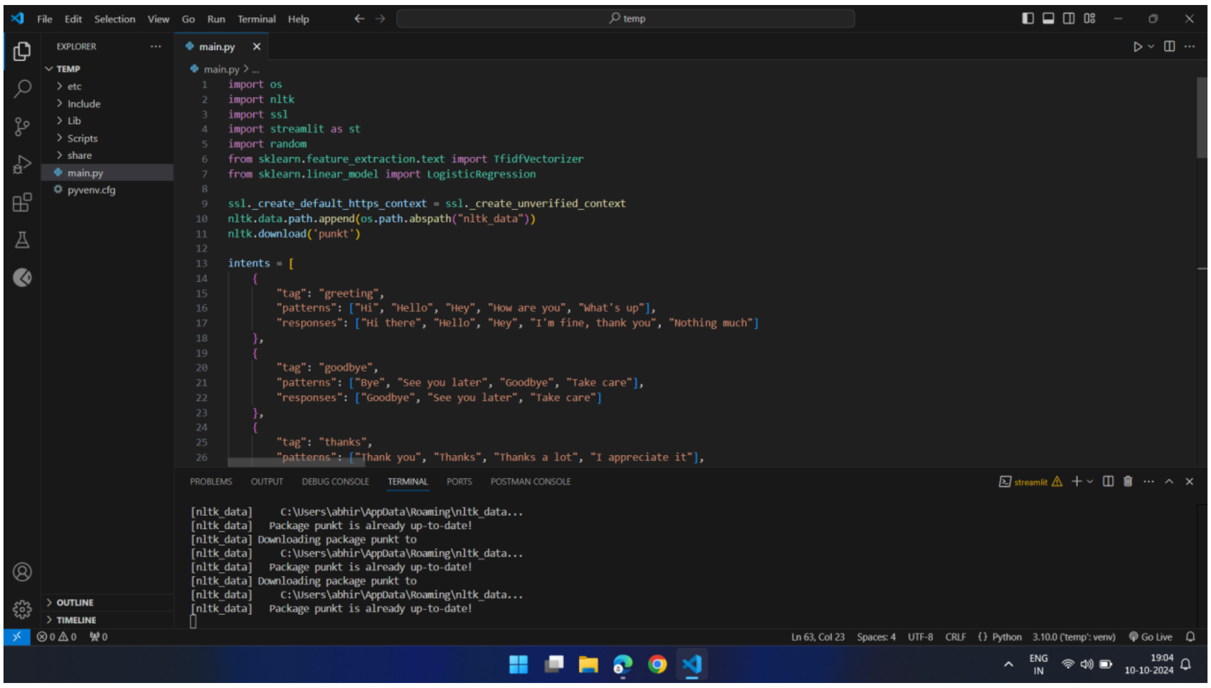
    - Currently, the chatbot provides predefined responses. Future improvements could include personalized replies based on user history.

- **Scalability:**

    - The chatbot is designed to handle multiple intents and can be expanded with more intents and responses as needed.

WORKING CONDITIONS :

9.RESULTS

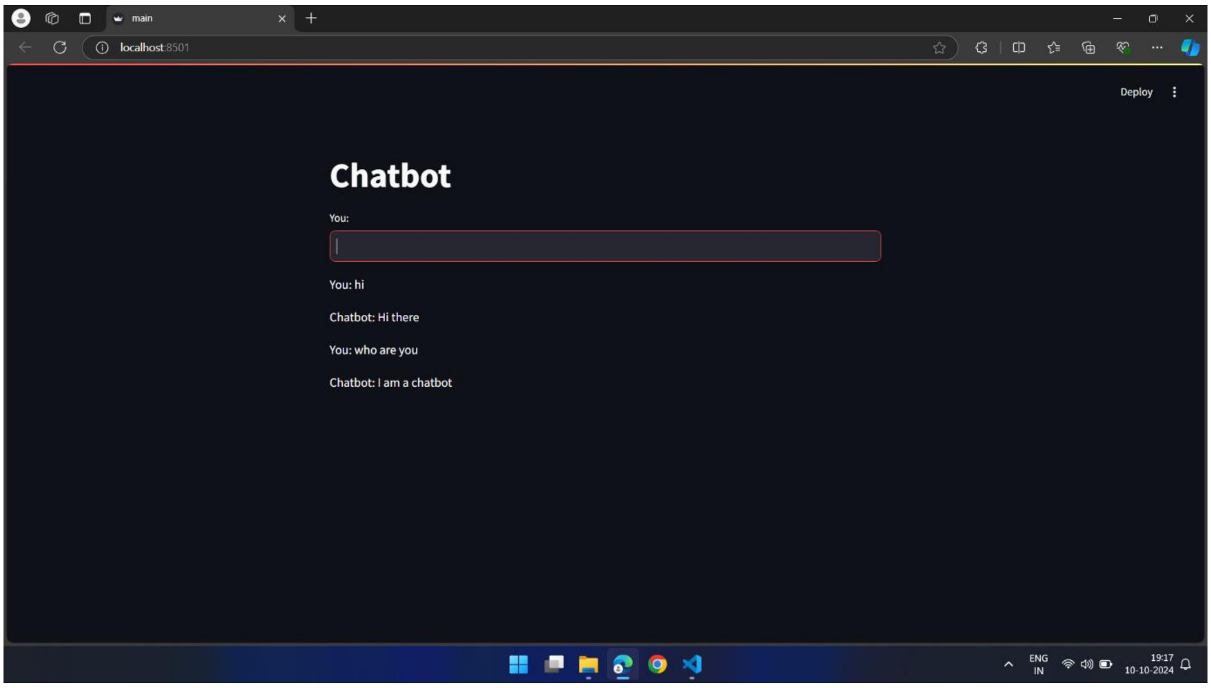


```python
import os
import nltk
import ssl
import streamlit as st
import random
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression

ssl._create_default_https_context = ssl._create_unverified_context
nltk.data.path.append(os.path.abspath("nltk_data"))
nltk.download('punkt')

intents = [
    {
        "tag": "greeting",
        "patterns": ["Hi", "Hello", "Hey", "How are you", "What's up"],
        "responses": ["Hi there", "Hello", "Hey", "I'm fine, thank you", "Nothing much"]
    },
    {
        "tag": "goodbye",
        "patterns": ["Bye", "See you later", "Goodbye", "Take care"],
        "responses": ["Goodbye", "See you later", "Take care"]
    },
    {
        "tag": "thanks",
        "patterns": ["Thank you", "Thanks", "Thanks a lot", "I appreciate it"],
```

```
[nltk_data]     C:\Users\abhir\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\abhir\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\abhir\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```



Chatbot

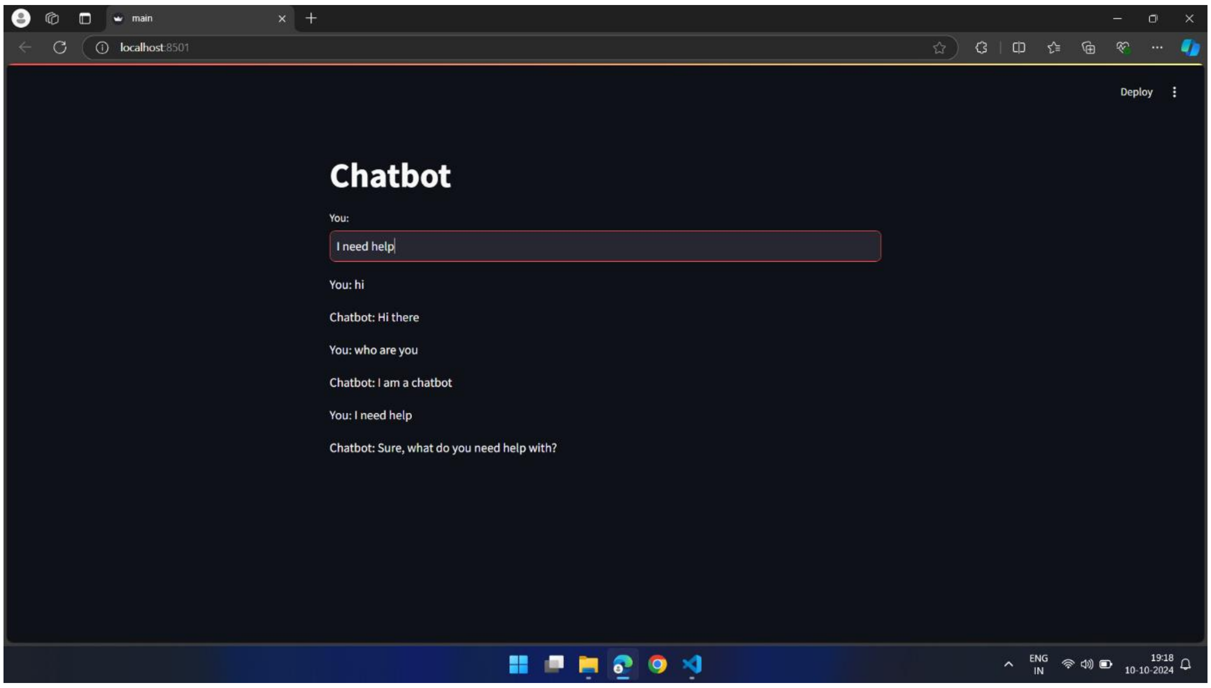
You:

You: hi

Chatbot: Hi there

You: who are you

Chatbot: I am a chatbot



Chatbot

You:

I need help

You: hi

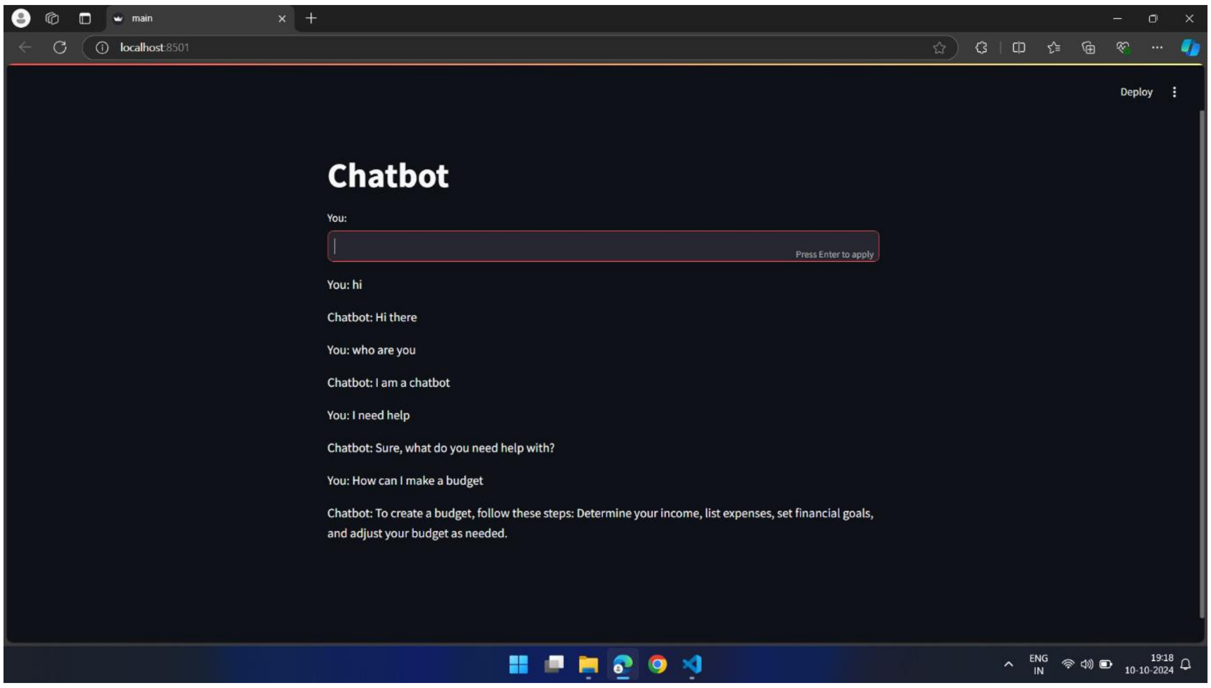
Chatbot: Hi there

You: who are you

Chatbot: I am a chatbot

You: I need help

Chatbot: Sure, what do you need help with?



Chatbot

You:

Press Enter to apply

You: hi

Chatbot: Hi there

You: who are you

Chatbot: I am a chatbot

You: I need help

Chatbot: Sure, what do you need help with?

You: How can I make a budget

Chatbot: To create a budget, follow these steps: Determine your income, list expenses, set financial goals, and adjust your budget as needed.

CONCLUSION:

10.SUMMARY

- **Setup:** Imports necessary libraries and configures NLTK.
- **Intents Definition:** Lists various user intents with patterns and responses.
- **Data Preparation:** Extracts patterns and tags, then trains a TF-IDF vectorizer and Logistic Regression model.
- **Chatbot Logic:** Processes user input to predict intent and generate appropriate responses.
- **User Interface:** Uses Streamlit to create a web-based interface for interacting with the chatbot.

11.LEARNINGS

- Learned to use NLP techniques to preprocess and understand user input.
- Trained a machine learning model to classify user intents.
- Built a web-based chatbot interface using Streamlit for real-time interaction.
- Designed conversation flows by mapping inputs to predefined responses.
- Gained insight into building scalable and expandable chatbot systems.

APPENDICES:

https://www.simplilearn.com/tutorials/python-tutorial/how-to-make-a-chatbot-in-python

https://www.geeksforgeeks.org/chat-bot-in-python-with-chatterbot-module/