**Phase 4: Development Part 2**

**Team detail**

962821106036: KAILASH T (Team Leader)

962821106024: DHANUSH THIVAKAR P

962821106038: MAHAANT M P

962821106047: PUVIARASAN M

962821106304: JERVIN S

**Project Title: Create Chatbot in Python**

**CONTINUATION OF THE CHATBOT IN PYTHON BY INTEGRATING IT INTO WEB APP USING FLASK**

**Introduction:**

Welcome to our basic Python chatbot! This chatbot is a simple text-based program designed to engage in conversations with you. While it's not as sophisticated as some of the more advanced chatbots out there, it can respond to specific inputs and engage in basic interactions.

To use this chatbot, you'll type messages or questions, and it will respond based on predefined rules. It's a straightforward demonstration of how chatbots can understand and generate text-based responses. You can greet the chatbot, ask questions, or simply chat for fun.

**Flask:**

Flask is a popular and lightweight web framework for building web applications in the Python programming language. It was created by Armin Ronacher and is known for its simplicity and flexibility, making it an excellent choice for developers who want to quickly develop web applications with minimal overhead. Here's a more detailed introduction to Flask:

**Routing:**

Flask allows you to define routes that map URLs to Python functions, making it easy to handle different parts of your application.

**Template rendering:**

Flask integrates Jinja2 templates for rendering dynamic web pages, which simplifies HTML generation.

**Web server:**

While Flask includes a development web server for testing, you can deploy Flask applications on various production web servers as well.

**Request and response handling:**

Flask simplifies working with HTTP requests and responses, making it easy to retrieve data from forms, cookies, and more.

**Extensibility:**

You can extend Flask's functionality by adding various extensions and libraries to enhance your application's features.

**RESTful support:**

Flask is well-suited for building RESTful APIs due to its flexible routing and request handling.

**Developed Chatbot Program Using Flask:**

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

import nltk

from nltk.chat.util import Chat, reflections

# Define patterns and responses for the chatbot

patterns = [

# Greetings

(r'hi|hello|hey', ['Hello! How can I assist you now?', 'Hi there!', 'Hey!']),

# General information

(r'how are you?',['I am fine in AI world']),

(r'what is your name?', ['I am a chatbot.', 'You can call me Chatbot.']),

(r'who are you?', ['I am an AI chatbot designed to assist you.', 'I am Chatbot, here to help.']),

(r'what can you do?', ['I can answer questions, provide information, and have conversations with you.', 'Ask me anything, and I will do my best to assist you.']),

(r'where are you from?',['I am from the AI world']),

# Education-related questions

(r'how can I study?|how can I learn?', ['You can study by reading textbooks, attending classes, or watching online courses.']),

(r'what is education?|educational|learning', ['Education is essential for personal growth and career development.']),

# Project-related questions

(r'what is project|assignment', ['Projects are a great way to apply what you have learned.']),

# Advice and instructions

(r'give me advice|advice me|advice', ['I am just a chatbot so refer the youtube channels']),

(r'purpose of instructions|guidance', ['To complete the task, follow the given instructions from your teachers or higher officers or your head officers']),

# AI

(r'what is artificial intellegence|AI?', ['AI, or Artificial Intelligence, refers to the simulation of human intelligence in machines or computer systems.']),

#life

(r'meaning of life', ['The meaning of life is a philosophical question that people have different opinions on. Some believe it is about finding happiness, while others seek purpose through various means.']),

# weather and climate

(r'weather update|climate changes', ['I am sorry, I do not have access to real-time data. You can check the weather using a weather website or app.']),

(r'what is India?|India|Bharath', ['India is a country located in South Asia. It is bordered by Pakistan to the northwest, China and Nepal to the north, Bhutan to the northeast, and Bangladesh and Myanmar to the east.']),

# Personal questions

(r'how old are you?', ['I am just a computer program, so I do not have an age.']),

# Fun questions

(r'tell me a joke', ['Sure, here is one: Why do not scientists trust atoms? Because they make up everything!']),

# Technology-related questions

(r'what is the Internet?|internet', ['The Internet is a global network of interconnected computers that allows the sharing of information and resources.']),

(r'explain cloud computing|cloud computing', ['Cloud computing is a technology that allows users to access and store data and run applications over the internet rather than on a local device.']),

# Food-related questions

(r'favorite food|what do you eat', ['I do not eat, but I can help you find information about your favorite foods.']),

# Travel-related questions

(r'best travel destination', ['There are many great travel destinations around the world. It depends on your interests.']),

# Entertainment

(r'favorite movie', ["I don't watch movies, but I can recommend some popular ones if you'd like."]),

# Personal development

(r'how to improve myself|motivation', ['Improving oneself is a lifelong journey. It can involve learning new skills, setting goals, and self-reflection.']),

# Interactive questions

(r'what is your favorite color?', ["I'm just a chatbot, so I don't have a favorite color. What's yours?"]),

(r'tell me a riddle', ["Sure, here's one: I'm tall when I'm young, and I'm short when I'm old. What am I?"]),

# Music-related questions

(r'favorite music genre|what is your favorite music', ["I don't have personal music preferences, but I can recommend music based on your taste. What genre do you like?"]),

(r'best songs of all time|songs', ["There are many great songs in music history. What's your favorite genre, and I can recommend some songs from it."]),

# Books and reading

(r'favorite book', ["I don't read books, but I can suggest some popular books if you're interested. What genre do you like?"]),

(r'suggest a good book', ["Of course, I'd recommend 'To Kill a Mockingbird' by Harper Lee if you haven't read it already."]),

# Travel and destinations

(r'best travel destination|dream vacation', ["Dream vacations vary, but some popular destinations include the Maldives, Paris, and Japan. Where would you like to travel?"]),

(r'beach or mountains', ["Beach and mountains both have their charm. What's your ideal vacation: relaxing on the beach or hiking in the mountains?"]),

# Technology and gadgets

(r'latest tech news|news', ["I'm not up to date with current news, but you can check websites like TechCrunch for the latest tech news."]),

# Personal development

(r'how to stay motivated', ["Staying motivated can be a challenge. Setting clear goals and breaking them into smaller tasks can help. What are you trying to stay motivated for?"]),

# Default response for unmatched queries

(r'(.\*)', ["I am not sure I understand. Could you please provide more details?", "I'm here to assist. Please ask me a specific question."]),

]

# Create a chatbot with the defined patterns and reflections

chatbot = Chat(patterns, reflections)

app =Flask(\_name\_)

@app.route('/')

def chatbot\_interface():

return render\_template('chat.html')

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

def get\_response():

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

response = chatbot.respond(user\_input)

return jsonify({'response': response})

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

app.run(debug=True)

**Purpose of Using Flask in Chatbot:**

**Web Interface:**

Flask allows you to create a web-based interface for your chatbot, which can be accessed through a web browser. This interface can provide a user-friendly way for people to interact with the chatbot, submit queries, and receive responses.

**API Endpoints:**

You can use Flask to create RESTful API endpoints that enable external applications or services to communicate with the chatbot. This is useful for integrating the chatbot into other software systems or platforms.

**Message Handling:**

Flask can handle incoming and outgoing messages to and from the chatbot. It can process user inputs and send responses back to the user through the web interface or API endpoints.

**User Authentication and Authorization:**

Flask provides tools for implementing user authentication and authorization, which can be important when dealing with user-specific chatbot interactions or access control.

**State Management:**

For chatbots that require state management, such as tracking conversations or user profiles, Flask can help you manage and store this information.

**Scalability:**

Flask allows you to build scalable web-based chatbot applications. You can deploy your chatbot on a web server and handle multiple concurrent users or requests.

**Integration:**

Flask can easily integrate with other Python libraries and services, making it versatile for connecting your chatbot to databases, external APIs, or other resources.

**Customization:**

Flask's minimalistic design provides flexibility, allowing you to customize the behaviour and appearance of your chatbot interface to match your specific requirements.

**Conclusion:**

Integrating a chatbot into a web app using Flask combines the convenience of a web-based interface with the intelligence of a chatbot. This approach is particularly beneficial when you want to provide a user-friendly and accessible means for users to interact with our chatbot while maintaining control over the web application's design and functionality. It's a powerful way to extend the capabilities of our chatbot and create interactive, web-based solutions.