NAAN MUDHALVAN

ASSIGNMENT

PHASE - 4

NAME : CJ ANUROOPA

ROLL NO : 2021105505

Building the chatbot by integrating it into a web app using Flask.

To build a chatbot using Python and integrate it into a web app using Flask, you can follow these steps: 1. Install Flask: Start by installing Flask, a web framework for Python, using the command pip install flask on your terminal. 2. Import Flask: Create a new Python file and import the Flask module. python from flask import Flask, request, jsonify 3. Initialize Flask: Initialize a new Flask app. python app = Flask(__name__) 4. Create a webhook endpoint: Set up a route for a webhook endpoint where your chatbot can receive messages. python @app.route('/webhook', methods=['POST']) def webhook(): # Process incoming message return jsonify(response) 5. Create a function to process incoming messages: Write a function that can process and generate responses to incoming messages. python def process_message(message): # Process message and generate response return response 6. Retrieve message from POST request: In the webhook function, retrieve the message from the POST request and call the process_message function. python @app.route('/webhook', methods=['POST']) def webhook(): data = request.get_json() message = data['message'] response = process_message(message) return jsonify(response) 7. Start the server: At the end of your Python file, start the Flask app. python if __name__ == '__main__': app.run() 8. Integrate your chatbot logic: Implement the logic to process messages and generate responses inside the process_message function. You can use any chatbot framework or library of your choice, such as NLTK, ChatterBot, or even a custom-built model. python def process_message(message): # Implement chatbot logic # You can use NLTK, ChatterBot, or any other chatbot library # Example using ChatterBot: from chatterbot import ChatBot from chatterbot.trainers import ChatterBotCorpusTrainer bot = ChatBot('MyChatBot') trainer = ChatterBotCorpusTrainer(bot)

trainer.train("chatterbot.corpus.english") response = bot.get_response(message) return str(response) 9. Customize your chatbot: Customize the chatbot logic, training, and responses based on your requirements. 10. Run the web app: Finally, run your Python file, and your Flask web app with the integrated chatbot will be up and running. This is a basic outline of how you can build a chatbot using Python and integrate it into a web app using Flask. You can further enhance the chatbot's capabilities by integrating it with APIs, databases, or even machine learning models for more advanced interactions.