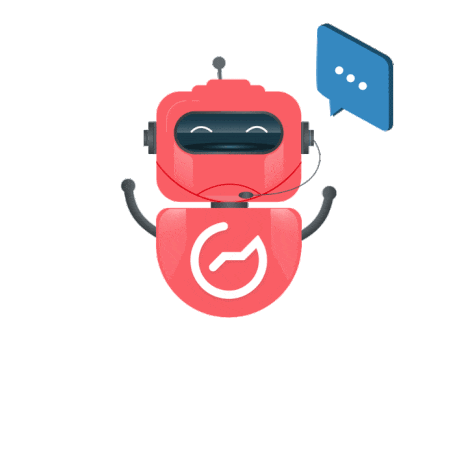
**CHATBOT IN PYTHON**

**Project Name**: Create Chatbot using Python

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



Steps to create a simple web-based chatbot using Flask:

**Install Required Libraries:**

First, make sure you have Python and Flask installed on your system. You may also need other libraries for chatbot functionality, such as NLTK or spaCy for natural language processing. You can install these using pip:

Code: pip install flask nltk # Install Flask and NLTK

**Create Your Flask App:**

Create a new directory for your project and set up your Flask application. Here's a basic structure:

Code:

/your\_project\_folder

├── app.py

├── templates/

│ └── index.html

├── static/

│ └── style.css

└── chatbot.py # This is where you implement your chatbot logic

**Create the Chatbot Logic:**

Implement the chatbot logic in a separate Python script (e.g., chatbot.py). You can use NLTK or another NLP library to create responses based on user input. For example:

import nltk

from nltk.chat.util import Chat, reflections

pairs = [

[

r"hi|hello",

["Hello!", "Hi there!", "How can I help you today?"]

],

# Add more patterns and responses here

]

chatbot = Chat(pairs, reflections)

def get\_response(user\_message):

return chatbot.respond(user\_message)

**Create the Flask Application:**

In your app.py file, create a Flask web application that handles user interactions with the chatbot:

Code:

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

from chatbot import get\_response

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

@app.route("/")

def index():

return render\_template("index.html")

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

def get\_chatbot\_response():

user\_message = request.form["user\_message"]

bot\_response = get\_response(user\_message)

return jsonify({"bot\_response": bot\_response})

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

**Create HTML and CSS Templates:**

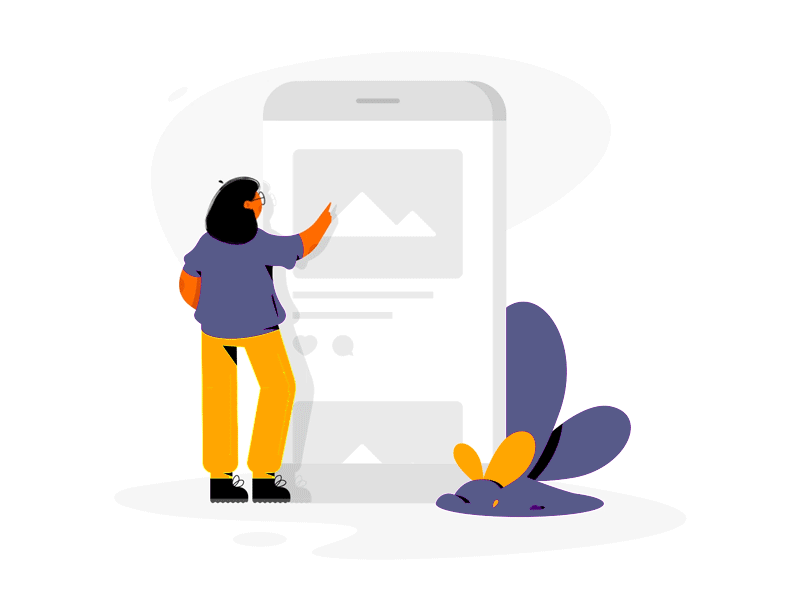
Create an HTML template for your chat interface (e.g., index.html) and style it using CSS (e.g., style.css). You can use JavaScript or jQuery to handle user input and display chatbot responses dynamically.

**Run Your Flask Application:**

Navigate to your project directory in the terminal and run your Flask app:

**Interact with Your Chatbot:**

Open your web app in a web browser, and you should see a chat interface. You can interact with your chatbot, and it will provide responses based on the patterns you defined.



**Create an HTML template for your chat interface:**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link rel="stylesheet" type="text/css" href="static/style.css">

<title>Chatbot</title>

</head>

<body>

<div class="chat-container">

<div class="chat-header">

<h2>Chatbot</h2>

</div>

<div class="chat-box" id="chat-box">

<div class="chat-message bot-message">

<p>Welcome! How can I assist you today?</p>

</div>

</div>

<div class="user-input">

<input type="text" id="user-message" placeholder="Type your message...">

<button id="send-button">Send</button>

</div>

</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<script src="static/app.js"></script>

</body>

</html>

**CSS:**

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

.chat-container {

width: 300px;

margin: 20px auto;

border: 1px solid #ccc;

border-radius: 5px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

background: #fff;

}

.chat-header {

background: #007BFF;

color: #fff;

text-align: center;

padding: 10px;

border-top-left-radius: 5px;

border-top-right-radius: 5px;

}

.chat-box {

padding: 10px;

max-height: 300px;

overflow-y: auto;

}

.chat-message {

margin: 10px;

padding: 10px;

border-radius: 5px;

}

.bot-message {

background: #f1f1f1;

}

.user-input {

padding: 10px;

display: flex;

justify-content: space-between;

}

#user-message {

flex-grow: 1;

padding: 5px;

border: 1px solid #ccc;

border-radius: 5px;

}

#send-button {

background: #007BFF;

color: #fff;

border: none;

border-radius: 5px;

padding: 5px 10px;

cursor: pointer;

}

#send-button:hover {

background: #0056b3;

}

**JAVASCRIPT:**

$(document).ready(function() {

$("#send-button").on("click", function() {

var userMessage = $("#user-message").val();

appendUserMessage(userMessage);

getUserResponse(userMessage);

});

$("#user-message").on("keypress", function(event) {

if (event.which === 13) {

$("#send-button").click();

}

});

function appendUserMessage(message) {

$("#chat-box").append('<div class="chat-message user-message"><p>' + message + '</p></div>');

$("#user-message").val('');

}

function appendBotMessage(message) {

$("#chat-box").append('<div class="chat-message bot-message"><p>' + message + '</p></div>');

}

function getUserResponse(userMessage) {

$.ajax({

type: "POST",

url: "/get\_response",

data: { user\_message: userMessage },

success: function(response) {

appendBotMessage(response.bot\_response);

},

error: function() {

appendBotMessage("An error occurred. Please try again.");

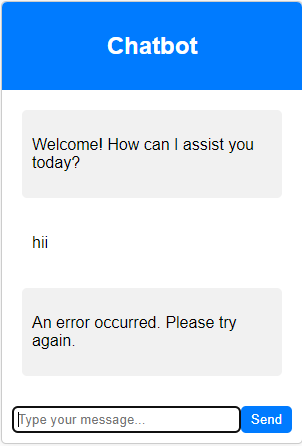
}

});

}

});

**OUTPUT:**

****

**Conclusion:**

In this topic, we've covered the process of creating a simple web-based chatbot integrated into a Flask web application, and we've also included information on how to link an HTML file to a CSS file.