TASK 10

SUBJECT:

Programming For AI

PROGRAM:

BS DATA SCIENCE

SUBMITTED TO:

Sir Rasikh Ali

SUBMITTED BY:

FIZZA FAROOQ

ROLL NUMBER:

SU92-BSDSM-F23-017

BSDS (4A)

Lab 10 task

Create any of the following Chatbot (GUI using Flask), develop its "Flask -> HTM:" based front-end

1. University Admission

Flask Application Setup

```
from flask import Flask, render_template, request, jsonify
import re
import random

app = Flask(__name__)
```

- **Flask** is imported, which is a **lightweight web framework** for building web applications.
- **re (regular expressions)** is used to match **user input patterns** for response handling.
- The **random module** is used to allow the chatbot to return **random responses** from a list.
- `app = Flask(__name__)` initializes a **Flask application**, allowing it to handle web requests.

Chatbot Responses (Pairs)

```
pairs = [
    [r"(?i).*hello.*|.*hi.*|.*hey.*",
     ["Hello! How can I assist you with admissions?",
      "Hi there! Do you need admission information?",
      "Hey! Welcome to Superior University Admissions, how can I help?"]],

    [r"(?i).*programs.*offer.*|.*degree programs.*|.*courses available.*",
     ["Superior University offers undergraduate, graduate, and PhD programs in Engineering, Busines
      "We offer a variety of programs, including BS, MS, and PhD in multiple fields. You can visit
      "Our university provides degrees in disciplines such as Computer Science, Business Administra

    [r"(?i).*admission requirements.*|.*entry requirements.*|.*eligibility.*",
     ["Admission requirements vary by program. Generally, you need academic transcripts, an entry t
      "You must have the required qualifications based on your desired program. Some programs also
      "Each program has specific requirements. Visit our website or contact the admission office fo

    [r"(?i).*apply.*admission.*|.*admission process.*|.*steps to apply.*",
     ["You can apply online through the official Superior University website or visit the admission
      "The admission process is simple: fill out the online application form, submit required docum
      "To apply, visit our website, choose your program, and follow the instructions to complete th
```

```
    ["You can apply online through the official Superior University website or visit the admissior
     "The admission process is simple: fill out the online application form, submit required docum
     "To apply, visit our website, choose your program, and follow the instructions to complete th

    [r"(?i).*last date.*apply.*|.*admission deadline.*|.*when.*admission.*close.*",
     ["Admission deadlines vary each semester. Please check the official website or contact the adm
      "The last date to apply depends on the academic session. Keep an eye on our website for impor
      "Superior University announces deadlines for each intake separately. Visit our admissions pag

    [r"(?i).*bye.*|.*goodbye.*|.*see you later.*",
     ["Goodbye! Feel free to ask anytime.",
      "See you later! Good luck with your admission.",
      "Alright, take care! Let us know if you have more questions."]]
```

- This is a **list of lists** containing **regex patterns** and their corresponding responses.
- Each **pattern (regular expression)** is designed to detect a specific type of user query.
- If a **pattern matches**, the chatbot returns a **random response** from the associated list.
- **Example:**
  - If the user types **"Hello"**, the chatbot may respond with:
    "Hello! How can I assist you with admissions?"

Function to Get Response

```python
def get_response(user_input):
    for pattern, responses in pairs:
        if re.search(pattern, user_input):
            return random.choice(responses)
    return "I'm sorry, I don't have information on that. Please ask about university admissions."

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

@app.route("/chat", methods=["POST"])
def chat():
    user_message = request.form["message"]
    response = get_response(user_message)
    return jsonify({"response": response})

if __name__ == "__main__":
    app.run(debug=True)
```

- A **loop** is used to try matching the user's input with each pattern.
- **`re.search(pattern, user_input)`** checks if the user's message matches any predefined pattern.

- If a match is found, the **chatbot returns a random response** from the corresponding category.

- If no match is found, the chatbot returns a **default message**:
"I'm sorry, I don't have information on that. Please ask about university admissions."

- The **"/chat" API route** handles chatbot responses.

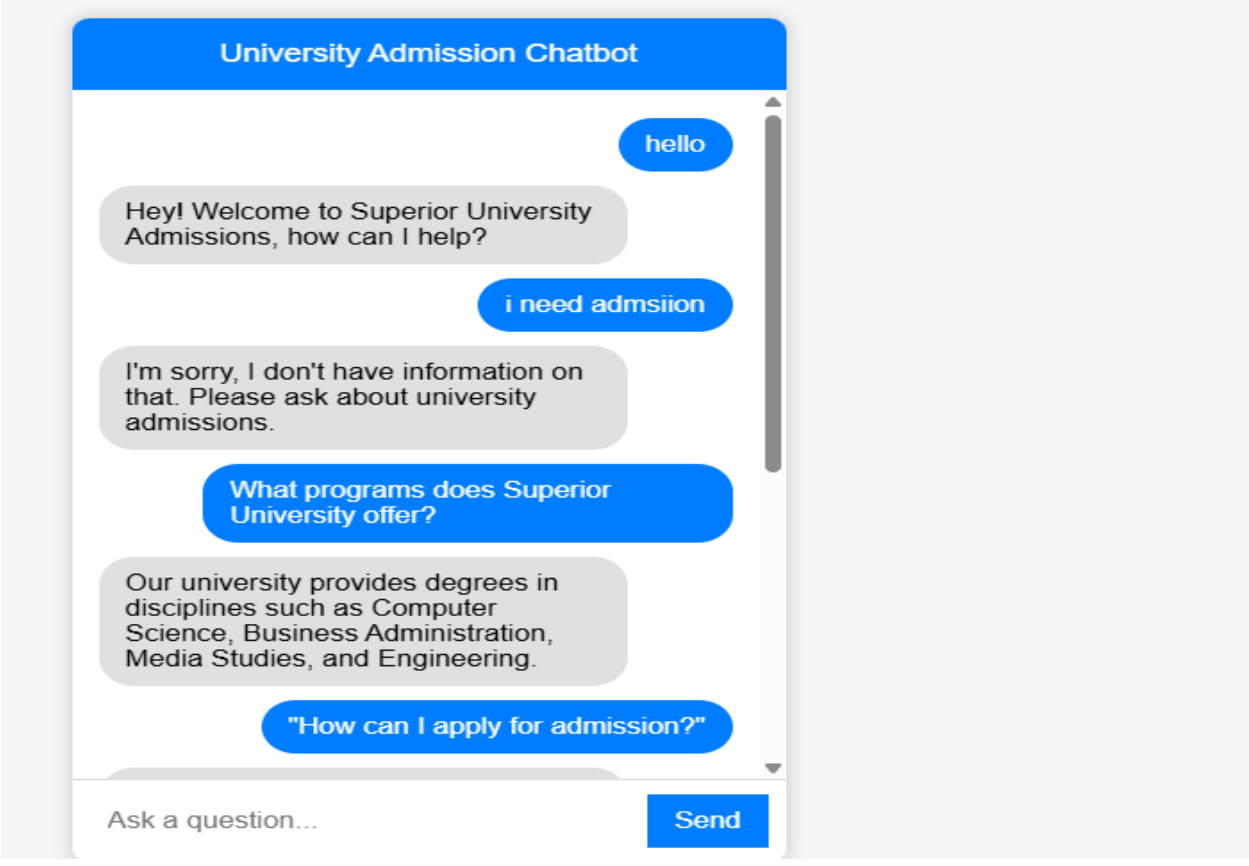
- A **POST request** is used, where:

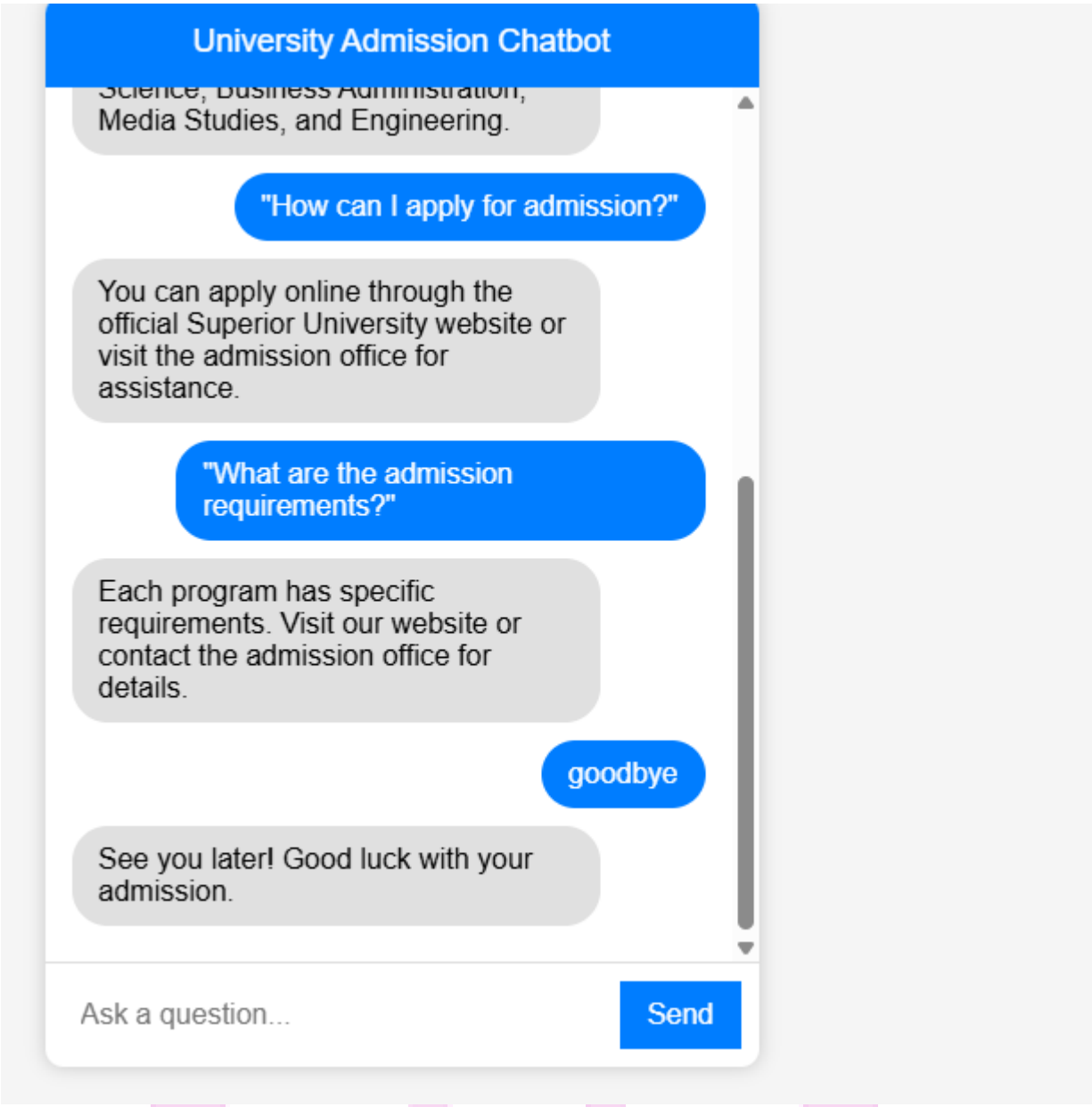
  - The **user's message** is retrieved using `request.form["message"]`.
  - The `get_response(user_message)` function is called to generate the chatbot's reply.
  - The response is sent back as **JSON** in the format `{ "response": response }`.

- `app.run(debug=True)` runs the Flask app in **debug mode**, meaning:

  - Errors will be displayed in the console.
  - The app will **automatically reload** when changes are made.

OUTPUT





1. HTML Structure

- ✓ This part defines the basic layout of the chatbot:
- ✓ ** ** : Wraps the entire chatbot.
- ✓ ** ** : Displays the title "University Admission Chatbot".
- ✓ ** ** : Shows the conversation messages.
- ✓ ** ** : Contains the user input field and send button.
- ✓ **<input type="text">** : Allows the user to type a question.
- ✓ **<button onclick="sendMessage()">Send</button> :** Sends the question when clicked.

2. CSS Styling

- **Background**: Light gray (#f5f5f5).
- **Chat Box**: White (white), rounded corners (border-radius: 10px;), and shadow (box-shadow) for a neat look.
- **Header**: Blue (#007bff) background with white text.
- **User Message**: Blue background with white text (background: #007bff; color: white;).
- **Bot Message**: Light gray background (background: #e0e0e0; color: black;).

3.Connecting with Flask Server

This code works with a **Flask-based AI chatbot**, communicating through the "/chat" API:

- **User's message** is sent to the server.
- **The server processes the message** and sends a JSON response.
- **The bot's response** is displayed in the chat interface.

Summary

This **University Admission Chatbot** allows users to ask admission-related questions. It sends the user's queries to a Flask-based chatbot server, receives AI-generated responses, and displays them in an interactive chat interface.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>University Admission Chatbot</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f5f5f5;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }
        .chat-container {
            width: 400px;
            height: 600px;
            background: white;
            display: flex;
            flex-direction: column;
            border-radius: 10px;
            box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.2);
        }
        .chat-header {
            background: #007bff;
            color: white;
            padding: 15px;
```

```
<head>
    <style>
        .chat-header {
            text-align: center;
            font-size: 18px;
            border-top-left-radius: 10px;
            border-top-right-radius: 10px;
        }
        .chat-box {
            flex: 1;
            padding: 15px;
            overflow-y: auto;
            display: flex;
            flex-direction: column;
        }
        .message {
            max-width: 75%;
            padding: 10px 15px;
            margin: 5px 0;
            border-radius: 20px;
        }
        .user-message {
            align-self: flex-end;
            background: #007bff;
            color: white;
        }
        .bot-message {
            align-self: flex-start;
```

```
<head>
    <style>
        .chat-input {
            display: flex;
            border-top: 1px solid #ddd;
            padding: 10px;
        }
        .chat-input input {
            flex: 1;
            padding: 10px;
            border: none;
            outline: none;
            font-size: 16px;
        }
        .chat-input button {
            padding: 10px 15px;
            background: #007bff;
            color: white;
            border: none;
            cursor: pointer;
            font-size: 16px;
        }
    </style>
</head>
<body>

<div class="chat-container">
    <div class="chat-header">University Admission Chatbot</div>
```

```
<head>
    <style>
    </style>
</head>
<body>

<div class="chat-container">
    <div class="chat-header">University Admission Chatbot</div>
    <div class="chat-box" id="chat-box"></div>
    <div class="chat-input">
        <input type="text" id="user-input" placeholder="Ask a question..." onkeypress="handleKeyPre
        <button onclick="sendMessage()">Send</button>
    </div>
</div>

<script>
    function sendMessage() {
        let userInput = document.getElementById("user-input").value;
        let chatBox = document.getElementById("chat-box");

        if (userInput.trim() === "") return;

        // Display user message
        let userMessage = document.createElement("div");
        userMessage.className = "message user-message";
        userMessage.innerText = userInput;
        chatBox.appendChild(userMessage);
        chatBox.scrollTop = chatBox.scrollHeight;
```

```
<script>
    function sendMessage() {
        userMessage.innerText = userInput;
        chatBox.appendChild(userMessage);
        chatBox.scrollTop = chatBox.scrollHeight;

        fetch("/chat", {
            method: "POST",
            body: new URLSearchParams({ "message": userInput }),
            headers: { "Content-Type": "application/x-www-form-urlencoded" }
        })
        .then(response => response.json())
        .then(data => {

            let botMessage = document.createElement("div");
            botMessage.className = "message bot-message";
            botMessage.innerText = data.response;
            chatBox.appendChild(botMessage);
            chatBox.scrollTop = chatBox.scrollHeight;
        });

        document.getElementById("user-input").value = "";
    }

    function handleKeyPress(event) {
        if (event.key === "Enter") {
            sendMessage();
        }
    }
</script>

</body>
</html>
```