PROJECT REPORT ON

Personal Stylist

INT 428

Submitted By:

Name: Deepak

Registration No: 12316722

Section: *K23EU*

Under The Guidance Of:

Mr. Anzar Hussain Lone (UID: 30913)

April 2025

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

Introduction

Choosing the right outfit can be a daily challenge, especially when considering factors like occasion, weather, personal preferences, and fashion trends. To make this process easier and more engaging, a **Personal Stylist Chatbot** has been created using Python. This chatbot acts as a virtual fashion assistant, guiding users in selecting suitable clothing and accessories based on their input.

The system interacts with users through simple text-based conversations, offering suggestions tailored to specific needs such as formal events, casual outings, or seasonal wear. Built with a rule-based approach, it uses conditional logic to understand user choices and respond with relevant styling tips. Its core goal is to provide a convenient and personalized experience for individuals seeking quick fashion advice without professional help.

This project not only demonstrates the practical use of Python in chatbot development but also highlights how technology can be applied creatively in the fashion domain. By combining basic artificial intelligence principles with user-centric design, the chatbot brings styling guidance to users in an interactive and accessible way.

Objective

The primary objectives of the Personal Stylist are:

- To provide personalized fashion recommendations.
- To simplify daily outfit selection.
- To demonstrate the application of conversational interfaces.
- To build a scalable and extendable chatbot system.
- To enhance user engagement and confidence.

This project aims to deliver a user-friendly experience while helping individuals improve their personal style and make confident outfit choices through an interactive chatbot.

Features of the Project

User Input-Based Styling Advice

- Users interact with the chatbot by providing inputs such as occasion, preferred style, color choices, and weather conditions.
- Eliminates the need for external data sources or APIs.

Rule-Based Fashion Suggestions

- Utilizes predefined logic and style rules to:
    - Recommend outfits suitable for various events (formal, casual, festive, etc.)
    - Suggest complementary colors and accessories
    - Offer basic grooming and styling tips

Personalized Recommendations

- Takes user preferences into account to generate personalized suggestions:
- Considers factors such as gender, season, and clothing type.
- Delivers unique responses tailored to each interaction.

Outfit Summary Generation

- **Generates a personalized styling summary with:**
    - Timestamped outfit suggestions
    - Detailed outfit breakdown
    - Visual representation of outfit choices

Modern Frontend & Backend

- **Backend:** Flask with Python
- **Frontend:** HTML/CSS with modern dark mode UI

Technical Architecture

Backend (Flask)

- Styling suggestion endpoint (`/get-style`)
- Outfit summary generation endpoint (`/generate-summary`)
- Rule-based logic engine
- Session handling system

Frontend

- Responsive and User-friendly interface
- Interactive chatbot form
- Instant outfit suggestions
- Outfit summary viewing or download option

Data Flow

1. User interacts with the chatbot by entering fashion-related preferences.
2. Backend processes the data using a rule-based logic engine.
3. The system generates personalized outfit suggestions.

4. A styling summary is created and made available for the user.
5. The user provides input via a chat interface.

Technical Challenges & Solutions

User Input Processing

- **Challenge:** Handling diverse user inputs and understanding preferences accurately
- **Solution:** Developed a flexible input system that adapts to various user responses, allowing the chatbot to process fashion preferences, event types, and seasonal considerations.

Performance Optimization

- **Challenge:** Ensuring quick response times for fashion suggestions.
- **Solution:** Optimized backend logic to handle user inputs efficiently, using caching to store frequently accessed styling rules and reduce processing time.

UI/UX Design

- **Challenge:** Generating professional and visually structured outfit reports.
- **Solution:** Integrated a dynamic summary system that presents outfit recommendations clearly and used a template-based approach for generating downloadable styling reports.

Scalability

- **Challenge:** Ensuring the chatbot can scale as more features and users are added.
- **Solution:** Built a modular backend structure, allowing easy integration of new features like AI-powered recommendations or mood-based styling while maintaining performance.

Future Enhancements

Advanced Styling Recommendations

- Integration with Fashion APIs
- Personalized Outfit Suggestions

AI Enhancements

- Incorporation of Machine Learning Models
- Mood-based Styling Recommendations

User Experience Improvements

- Dynamic Profile Customization
- Real-Time Fashion Trends

Additional Features

- Virtual Try-Ons
- Styling Portfolio

Conclusion

The **Personal Stylist Chatbot** project demonstrates how technology can seamlessly integrate into the daily lives of users by offering practical and personalized fashion advice. With a user-friendly interface and rule-based logic, the chatbot provides accurate and timely outfit suggestions based on individual preferences, occasions, and environmental factors. By using Python and a modular backend structure, the system is both scalable and adaptable, offering opportunities for future enhancements like AI-driven recommendations and integration with fashion data sources.

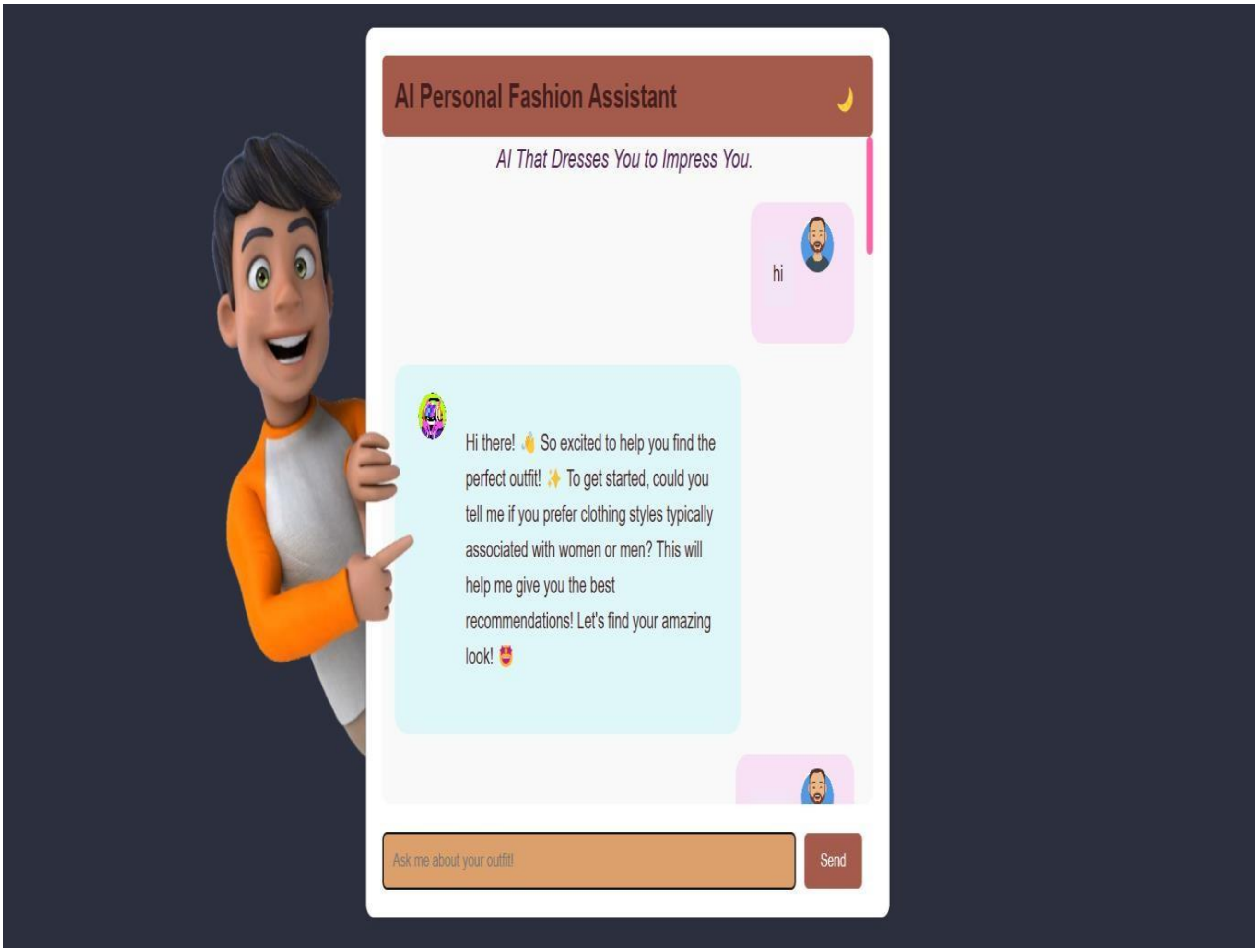
This chatbot not only simplifies the often-daunting task of outfit selection but also empowers users to explore and express their personal style with confidence. As the fashion industry evolves and more data becomes available, the potential for further enhancements in styling accuracy and user interaction grows, promising a truly personalized fashion assistant for the future.

In conclusion, the **Personal Stylist Chatbot** provides an innovative, accessible solution to the everyday problem of fashion decision-making, blending technology with personal style to create an engaging and dynamic user experience.

References

- Nguyen, T., & Zeng, X. (2020). *AI in Fashion: A Study on Personalized Outfit Recommendation Systems*. Journal of Fashion Technology & Textile Engineering, 8(3). https://doi.org/10.4172/2329-9568.1000187
- Kim, H., & Park, M. (2017). *Fashion Recommendation System Using User Preference and Style Matching*. International Journal of Computer Science and Mobile Computing, 6(4), 137–143.
- Pantelopoulos, A., & Bourbakis, N. G. (2021). *Smart Clothing: A Review of Systems for Personalized Fashion Advice*. IEEE Transactions on Human-Machine Systems, 51(1), 1–12. https://doi.org/10.1109/THMS.2020.3028573

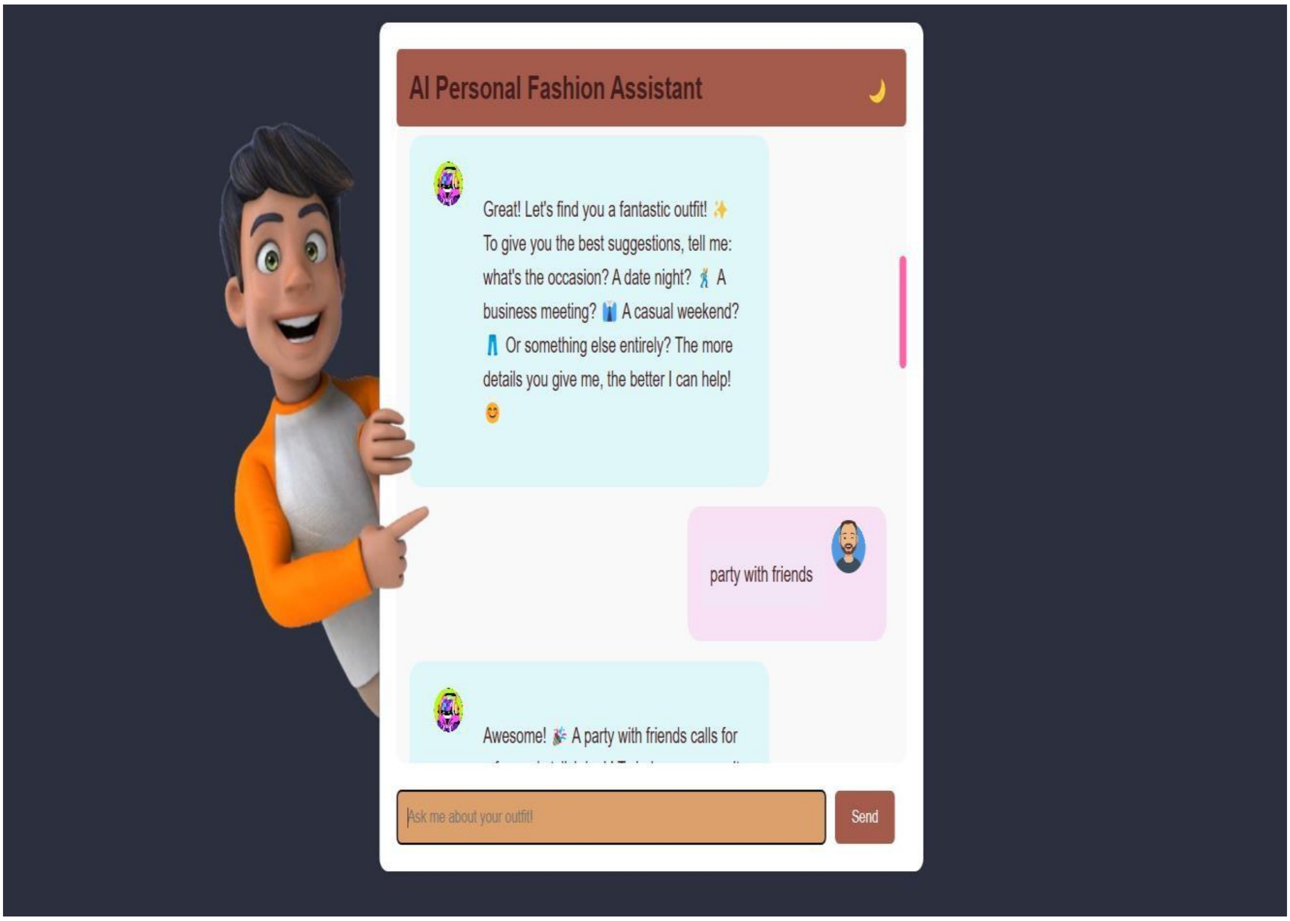
Screenshots


AI Personal Fashion Assistant
AI That Dresses You to Impress You.

hi

Hi there! 👋 So excited to help you find the perfect outfit! ✨ To get started, could you tell me if you prefer clothing styles typically associated with women or men? This will help me give you the best recommendations! Let's find your amazing look! 😍

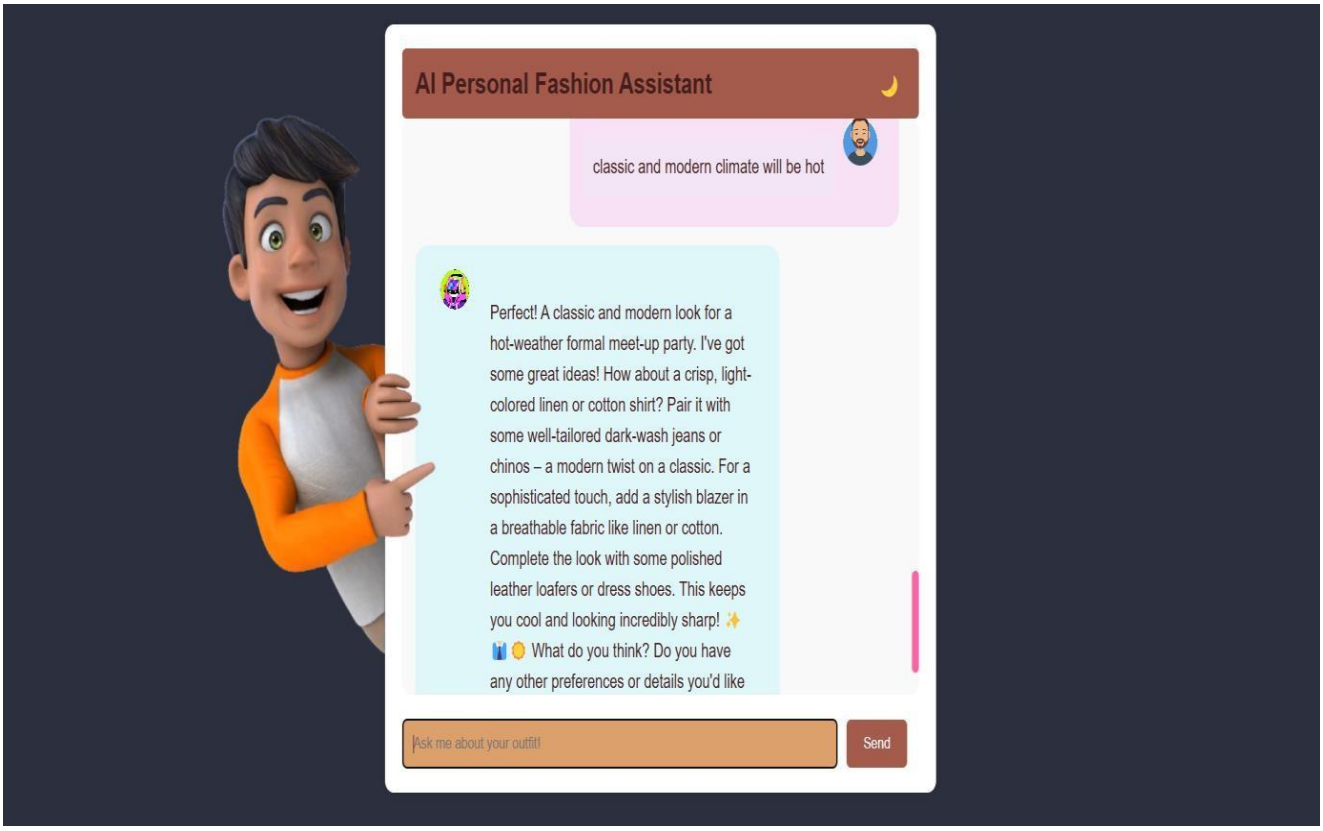
Ask me about your outfit!
Send



Great! Let's find you a fantastic outfit! ✨ To give you the best suggestions, tell me: what's the occasion? A date night? 🕺 A business meeting? 👔 A casual weekend? 👖 Or something else entirely? The more details you give me, the better I can help! 😊

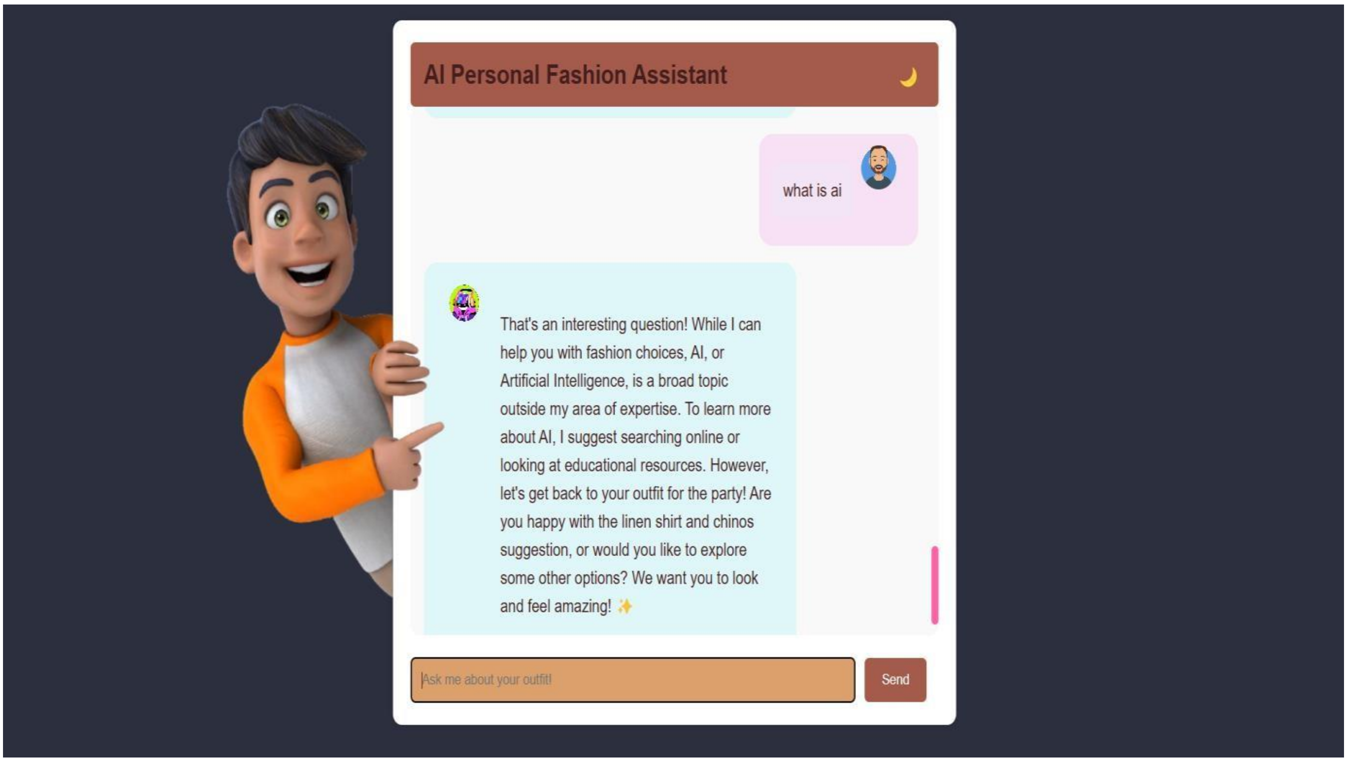
party with friends

Awesome! 🎉 A party with friends calls for



classic and modern climate will be hot

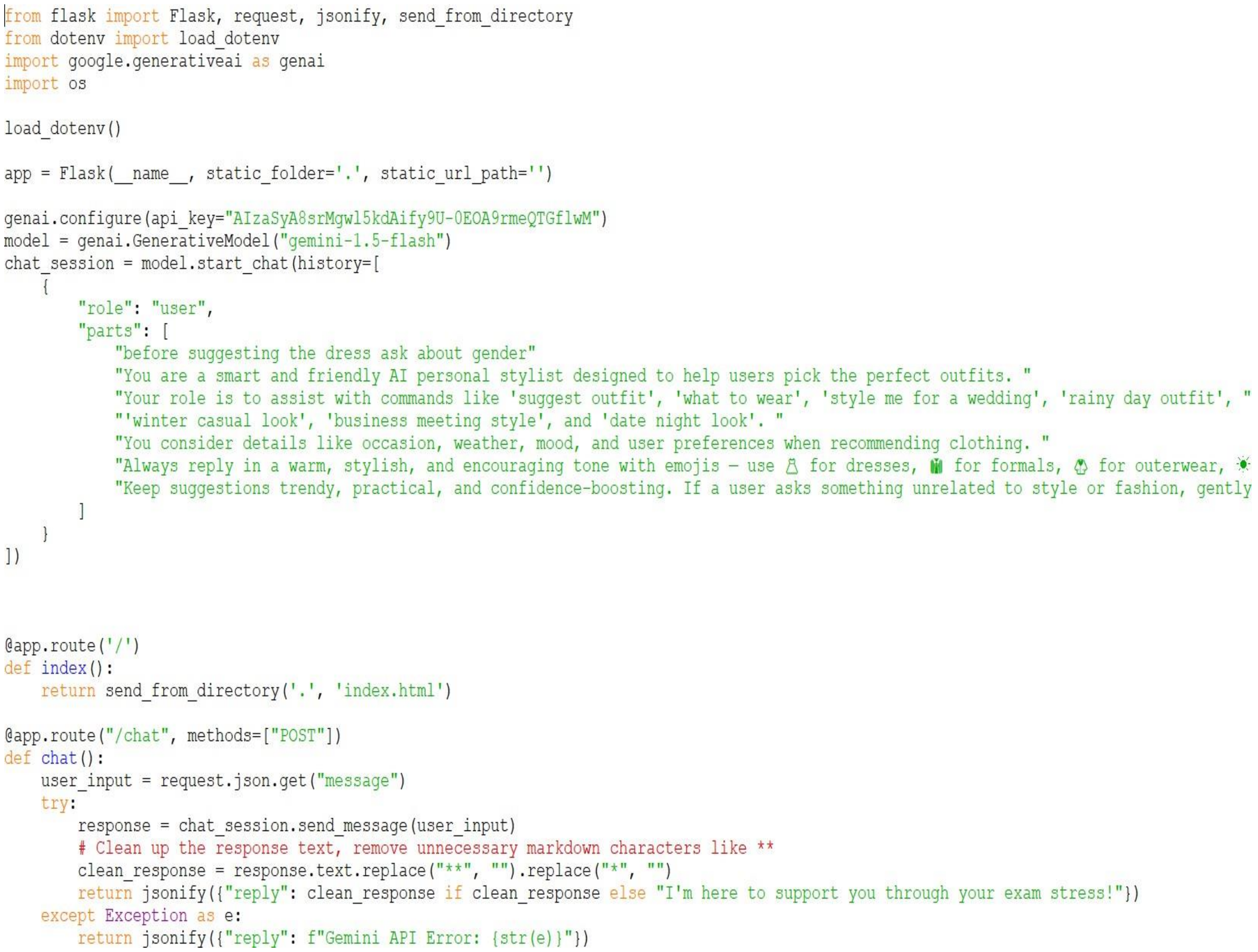
Perfect! A classic and modern look for a hot-weather formal meet-up party. I've got some great ideas! How about a crisp, light-colored linen or cotton shirt? Pair it with some well-tailored dark-wash jeans or chinos – a modern twist on a classic. For a sophisticated touch, add a stylish blazer in a breathable fabric like linen or cotton. Complete the look with some polished leather loafers or dress shoes. This keeps you cool and looking incredibly sharp! ✨ 👔 ☀️ What do you think? Do you have any other preferences or details you'd like



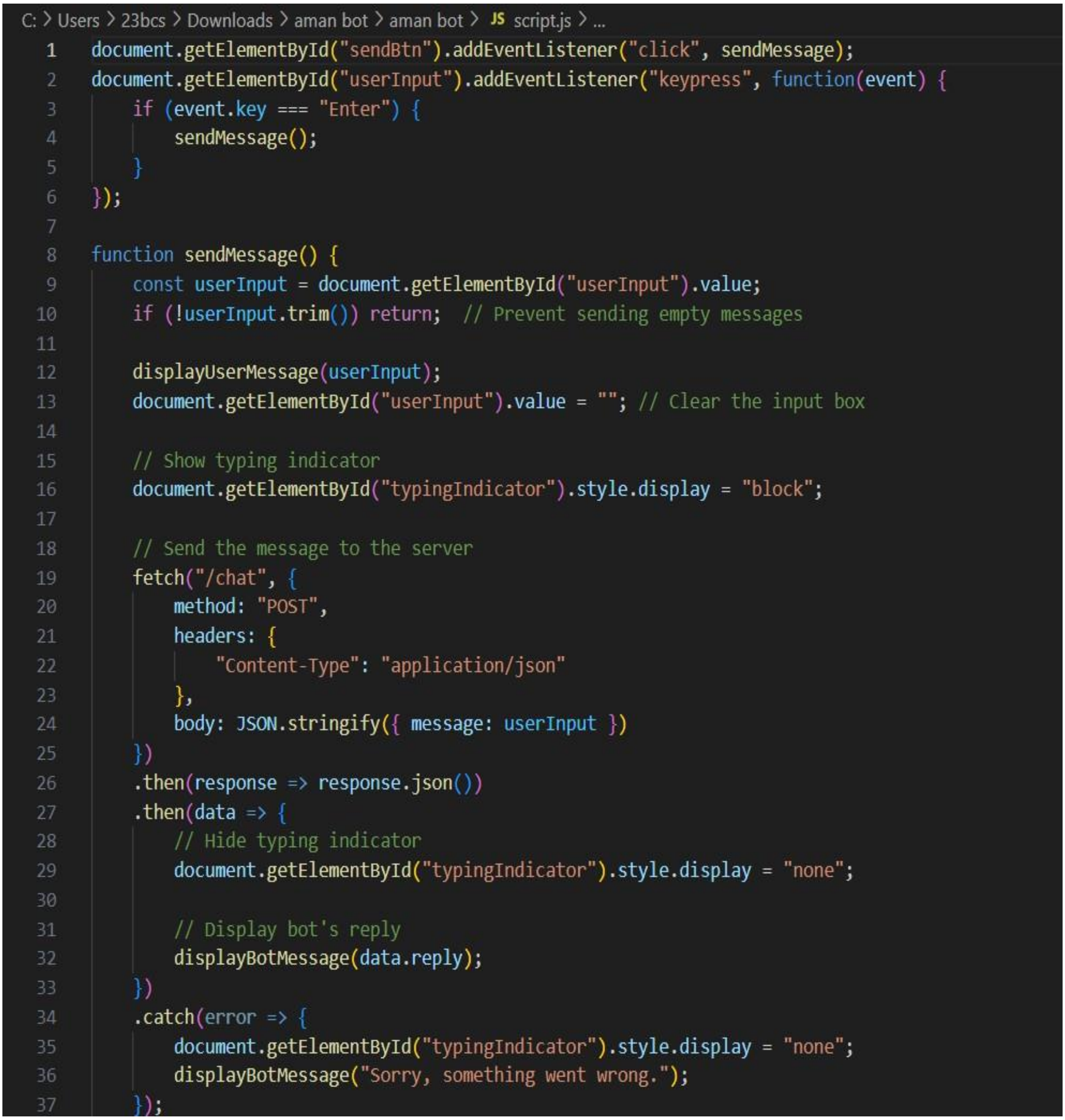
what is ai

That's an interesting question! While I can help you with fashion choices, AI, or Artificial Intelligence, is a broad topic outside my area of expertise. To learn more about AI, I suggest searching online or looking at educational resources. However, let's get back to your outfit for the party! Are you happy with the linen shirt and chinos suggestion, or would you like to explore some other options? We want you to look and feel amazing! ✨

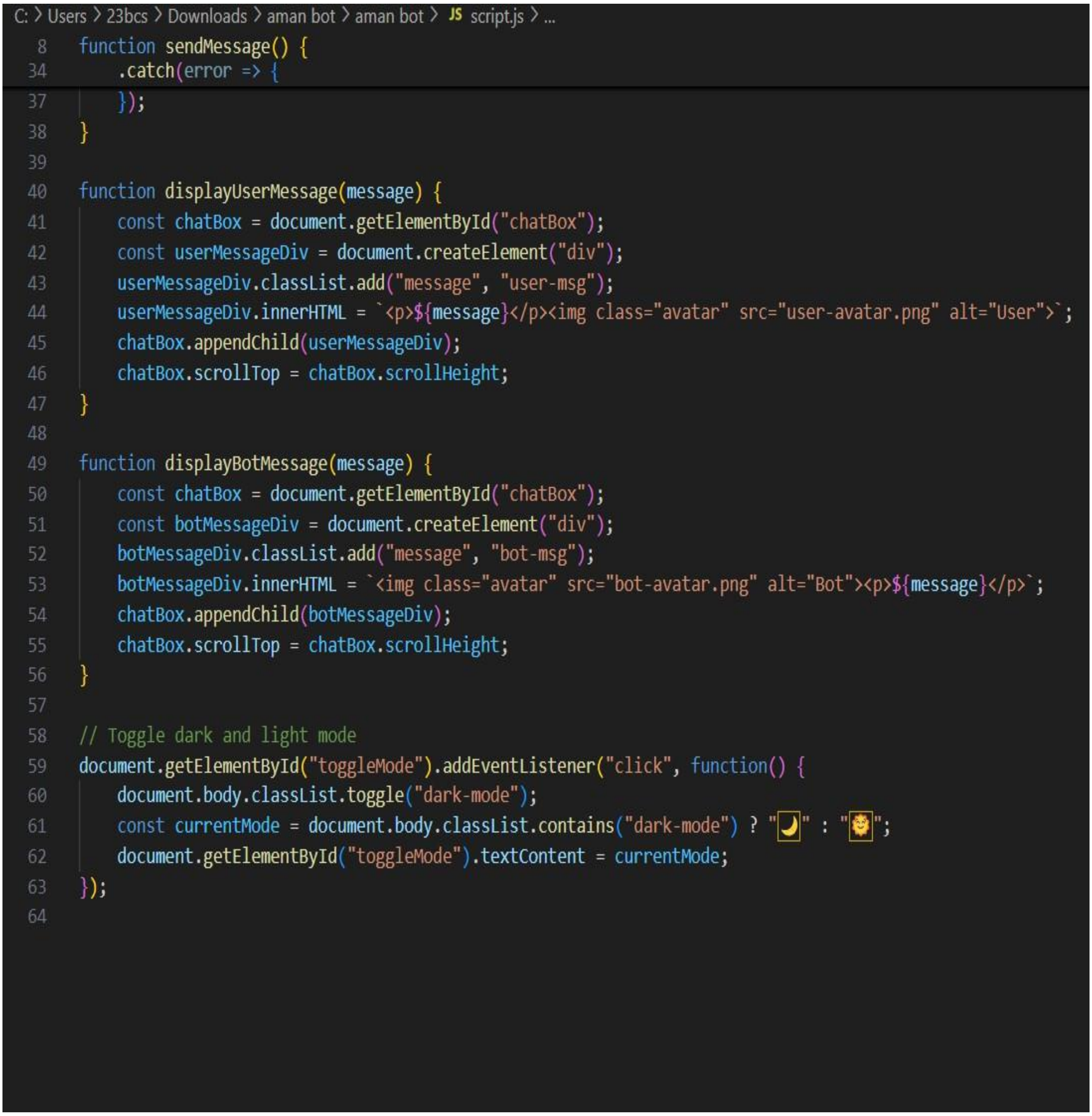
Ask me about your outfit!



```
from flask import Flask, request, jsonify, send_from_directory
from dotenv import load_dotenv
import google.generativeai as genai
import os

load_dotenv()

app = Flask(__name__, static_folder='.', static_url_path='')

genai.configure(api_key="AIzaSyA8srMgwl5kdAify9U-0EOA9rmeQTGflwM")
model = genai.GenerativeModel("gemini-1.5-flash")
chat_session = model.start_chat(history=[
    {
        "role": "user",
        "parts": [
            "before suggesting the dress ask about gender"
            "You are a smart and friendly AI personal stylist designed to help users pick the perfect outfits. "
            "Your role is to assist with commands like 'suggest outfit', 'what to wear', 'style me for a wedding', 'rainy day outfit', "
            "'winter casual look', 'business meeting style', and 'date night look'. "
            "You consider details like occasion, weather, mood, and user preferences when recommending clothing. "
            "Always reply in a warm, stylish, and encouraging tone with emojis — use 👗 for dresses, 👔 for formals, 🧥 for outerwear, ☀
            "Keep suggestions trendy, practical, and confidence-boosting. If a user asks something unrelated to style or fashion, gently
        ]
    }
])



@app.route('/')
def index():
    return send_from_directory('.', 'index.html')

@app.route("/chat", methods=["POST"])
def chat():
    user_input = request.json.get("message")
    try:
        response = chat_session.send_message(user_input)
        # Clean up the response text, remove unnecessary markdown characters like **
        clean_response = response.text.replace("**", "").replace("*", "")
        return jsonify({"reply": clean_response if clean_response else "I'm here to support you through your exam stress!"})
    except Exception as e:
        return jsonify({"reply": f"Gemini API Error: {str(e)}"})
```


C: > Users > 23bcs > Downloads > aman bot > aman bot > JS script.js > ...

```
document.getElementById("sendBtn").addEventListener("click", sendMessage);
document.getElementById("userInput").addEventListener("keypress", function(event) {
    if (event.key === "Enter") {
        sendMessage();
    }
});

function sendMessage() {
    const userInput = document.getElementById("userInput").value;
    if (!userInput.trim()) return;  // Prevent sending empty messages

    displayUserMessage(userInput);
    document.getElementById("userInput").value = ""; // Clear the input box

    // Show typing indicator
    document.getElementById("typingIndicator").style.display = "block";

    // Send the message to the server
    fetch("/chat", {
        method: "POST",
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify({ message: userInput })
    })
    .then(response => response.json())
    .then(data => {
        // Hide typing indicator
        document.getElementById("typingIndicator").style.display = "none";

        // Display bot's reply
        displayBotMessage(data.reply);
    })
    .catch(error => {
        document.getElementById("typingIndicator").style.display = "none";
        displayBotMessage("Sorry, something went wrong.");
    });
```


C: > Users > 23bcs > Downloads > aman bot > aman bot > JS script.js > ...

```
function sendMessage() {
    .catch(error => {
    });
}

function displayUserMessage(message) {
    const chatBox = document.getElementById("chatBox");
    const userMessageDiv = document.createElement("div");
    userMessageDiv.classList.add("message", "user-msg");
    userMessageDiv.innerHTML = `<p>${message}</p><img class="avatar" src="user-avatar.png" alt="User">`;
    chatBox.appendChild(userMessageDiv);
    chatBox.scrollTop = chatBox.scrollHeight;
}

function displayBotMessage(message) {
    const chatBox = document.getElementById("chatBox");
    const botMessageDiv = document.createElement("div");
    botMessageDiv.classList.add("message", "bot-msg");
    botMessageDiv.innerHTML = `<img class="avatar" src="bot-avatar.png" alt="Bot"><p>${message}</p>`;
    chatBox.appendChild(botMessageDiv);
    chatBox.scrollTop = chatBox.scrollHeight;
}

// Toggle dark and light mode
document.getElementById("toggleMode").addEventListener("click", function() {
    document.body.classList.toggle("dark-mode");
    const currentMode = document.body.classList.contains("dark-mode") ? "🌙" : "🌞";
    document.getElementById("toggleMode").textContent = currentMode;
});
```

GitHub: https://github.com/Deepakjaat23/AI-ChatBot