# Eco ChatBot Internship Project Report

Name: Harshita

Internship assginment at: UplyftTitle

ECO CHATBOT – Smart AI Assistant for Green Product Promotion

Date: [9june -14 june]

(2025)

# Table Of Contents

1-Introduction

2-Objective

3-Technology Stack

4-System Architecture

5-User interface screenshots

6-Implementation details

7-Backend Integration

8-learning outcomes

9-Challenges Faced

10-Conclusion

11-Declaration

# 1.Introduction

This report outlines the development of an AI-powered chatbot system named "Eco ChatBot" as a part of my internship project under Uplyft.

The chatbot is designed to assist users by providing information about eco-friendly products and encouraging sustainable living.

Users often find it difficult to locate eco-friendly alternatives to their daily-use products. Traditional e-commerce platforms may not prioritize sustainability or provide dedicated support for green choices. A smart chatbot can solve this by offering instant, engaging recommendations focused on sustainable living.

# 2.Objectives

- To build an AI-based chatbot that promotes eco-friendly products.

- To design a user-friendly interface for interactive communication.

- To enable real-time responses using Flask backend.

- To contribute to sustainable development goals through digital solutions

# 3.Technology Stack

| Category | Tools/Tech Used

| -------- | -------------------------------- |

| Frontend | HTML, CSS, JavaScript

| Backend | Python (Flask)

| AI/Chat | OpenAI or Custom API

| Hosting | Localhost (Flask Dev Server)

| Design | Image background, CSS animations

| Others | GitHub for version control

# 4.System Architecture

[ User ]

↓

[ Frontend (HTML + CSS + JS) ]

↓

[ Flask Backend (Python) ]

↓

[ Chat API (e.g., OpenAI or logic) ]

User enters a message → Frontend captures it

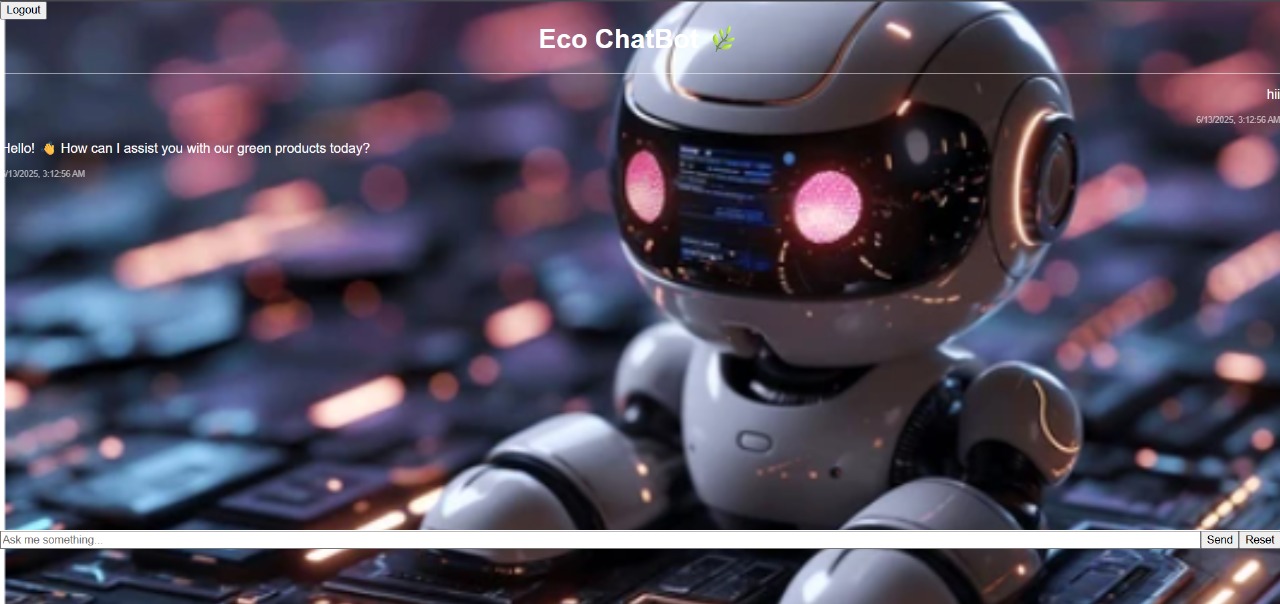
Request sent to Flask backend

Backend forwards it to the chatbot logic or AI

Response is received and shown in the UI

# 5.User Interface Screenshots

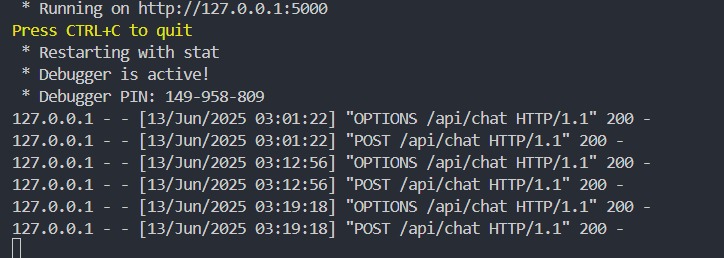
Screenshot 1:



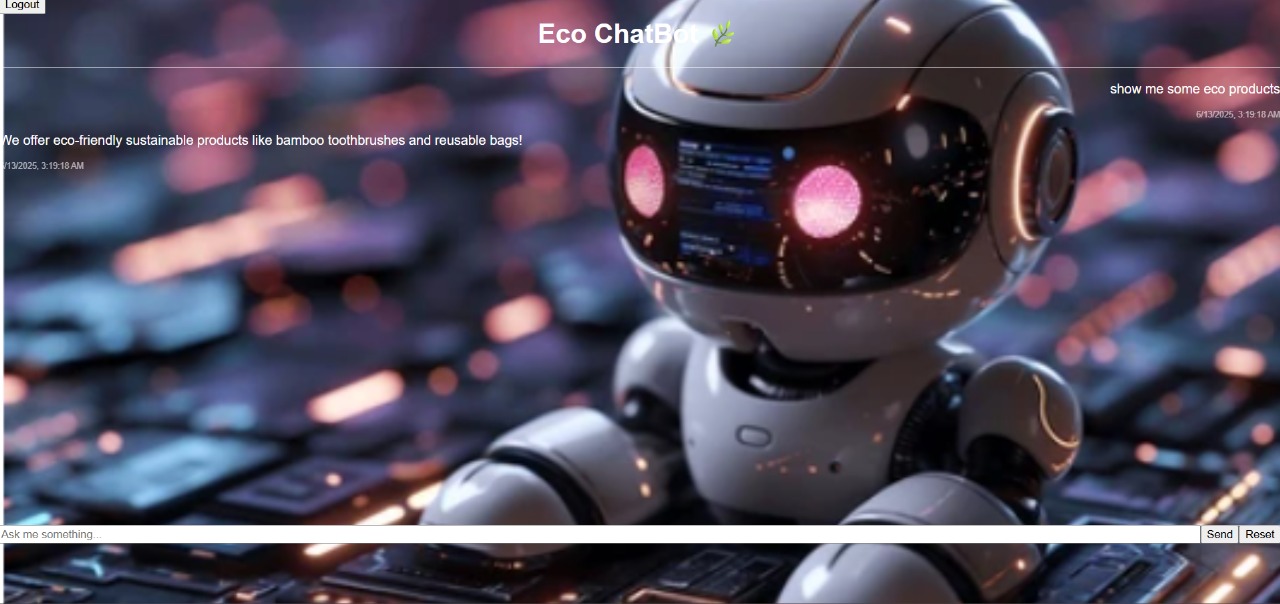
2.Screenshot 2:



3.Screenshot 3:



4.Screenshot 4:



# 6.ImplementImplementation details

A visually engaging landing page shows the login interface.

After login, the main chat window opens with a friendly UI robot image background.

Chat messages are dynamically shown.

Backend server handles /api/chat POST requests and returns a response.

Chat responses are displayed with timestamps.

Login Feature:

Simple login via email field for user identification. No password or authentication layer implemented in this prototype.

\*key features

Features of Eco ChatBot

🔹 Real-time chat interface

🔹 AI-generated replies about eco-friendly products

🔹 Friendly and aesthetic UI

🔹 Light-weight backend using Flask

🔹 Easy to run locally

🔹 User session with logout button

# 7.Backend Integration

The backend for the Eco ChatBot is built using Flask, a lightweight Python web framework. It serves as the backbone for the chatbot, handling message processing and communication between the frontend and the AI logic.

📍 1. Backend Setup

The Flask server runs locally and listens for incoming HTTP requests from the frontend.

A simple /api/chat endpoint is created to receive user messages and send back appropriate responses.

🔄 2. Request-Response Cycle

User Message is typed in the chatbox.

Frontend sends the message as a POST request to http://localhost:5000/api/chat.

Flask Server receives the message and processes it using keyword-based logic or a custom AI function.

Response is returned in JSON format to the frontend.

Frontend then displays the bot’s reply in the UI.

@app.route("/api/chat", methods=["POST"])

def chat():

data = request.get\_json()

user\_msg = data.get("message", "").lower()

if "toothbrush" in user\_msg:

reply = "You can try a bamboo toothbrush – it's eco-friendly and biodegradable!"

elif "bag" in user\_msg:

reply = "Reusable cloth bags are great alternatives to plastic."

else:

reply = "Let's make eco-friendly choices! 🌿"

return jsonify({"response": reply})

⚙ 3. Technologies in Backend

Flask – Python micro web framework to create REST APIs

CORS – For frontend-backend interaction

JSON – For data communication

Debug Mode – Real-time monitoring of logs

📁 4. File Structure (Simplified)

project/

│

├── static/

│ └── styles.css ← frontend styling

├── templates/

│ └── index.html ← main HTML file

├── app.py ← Flask backend

└── requirements.txt

🧪 5. Local Testing Postman for simulating API calls

Browser DevTools → Network Tab to observe requests

Terminal Logs to track backend message handling

🚀 6. Future Backend Upgrades

Add a database (e.g., MongoDB) for chat history

Implement user login system

Integrate OpenAI API or NLP engine

Host it on Render, Heroku, or Vercel y

# 8.Learning Outcomes

Improved proficiency in frontend and backend integration.

- Gained experience in real-time chatbot design.

- Learned to use Flask for RESTful API development.

- Understood deployment and testing of web apps.

# 9.Challenges Faced

Debugging API request failures and CORS issues.

- Managing UI layout and responsive design.

- Optimizing chatbot messages for clarity.

# 10.Conclusion

The Eco ChatBot project was an enriching experience that helped me develop a full-stack application with meaningful impact.

It showcases how technology can support sustainability and how internships like this enhance hands-on learning.

This chatbot can be extended to:

Connect to real product APIs

Add real-time authentication and user tracking

Improve AI integration with NLP

It is an excellent showcase of practical web development and real-world application potential.

# 11.Declaration

I, Harshita, declare that this project report is a result of my own efforts and the work done during my internship assignment at Uplyft.

All the screenshots and implementations are from the project I developed and tested myself.