Creating UI using flask:

import random

import json

import torch

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

from long\_responses import NeuralNet

from chatbot import bag\_of\_words, tokenize, stem

import os

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

# Load chatbot data

os.chdir('C:/Users/arara/OneDrive/Desktop/chatbot/')

with open('intents.json', 'r') as f:

    intents = json.load(f)

FILE = "data.pth"

data = torch.load(FILE)

input\_size = data["input\_size"]

hidden\_size = data["hidden\_size"]

output\_size = data["output\_size"]

all\_words = data["all\_words"]

model\_state = data["model\_state"]

tags = data["tags"]

model = NeuralNet(input\_size, hidden\_size, output\_size)

model.load\_state\_dict(model\_state)

model.eval()

bot\_name = "zoro"

# Route for rendering the chat interface

@app.route('/')

def chat\_interface():

    os.chdir('C:/Users/arara/OneDrive/Desktop/chatbot/')

    return render\_template('index.html')

# Function to handle chatbot responses

def get\_chatbot\_response(user\_input):

    sentence = tokenize(user\_input)

    x = bag\_of\_words(sentence, all\_words)

    x = x.reshape(1, x.shape[0])

    x = torch.from\_numpy(x)

    output = model(x)

    \_, predicted = torch.max(output, dim=1)

    tag = tags[predicted.item()]

    probs = torch.softmax(output, dim=1)

    prob = probs[0][predicted.item()]

    if prob.item() > 0.75:

        for intent in intents["intents"]:

            if tag == intent["tag"]:

                return random.choice(intent['responses'])

    else:

        return "I do not understand... please only ask queries related to food delivery."

# Route for handling chat requests

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

def chat():

    user\_input = request.json['message']

    chatbot\_response = get\_chatbot\_response(user\_input)

    return jsonify({'response': chatbot\_response})

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

this code is used to embed html code for creating an UI for the user to interact with the chatbot AI.

Html and css code used:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Chatbot</title>

    <meta charset="UTF-8">

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

    <meta http-equiv="X-UA-Compatible" content="ie edge">

    <link rel="stylesheet" href="{{ url\_for('static', filename='chatbot.css') }}">

</head>

<body>

    <!-- partial:index.partial.html -->

    <section class="msger">

        <header class="msger-header">

            <div class="msger-header-title">

                <i class="fas fa-bug"></i> Chatbot <i class="fas fa-bug"></i>

            </div>

        </header>

        <main class="msger-chat">

            <div class="msg left-msg">

                <div class="msg-img" style="background-image: url(https://image.flaticon.com/icons/svg/327/327779.svg)"></div>

                <div class="msg-bubble">

                    <div class="msg-info">

                        <div class="msg-info-name">Chatbot</div>

                        <div class="msg-info-time">12:45</div>

                    </div>

                    <div class="msg-text">

                        Hi, welcome to ChatBot! Go ahead and send me a message. 😄

                    </div>

                </div>

            </div>

        </main>

        <form class="msger-inputarea">

            <input type="text" class="msger-input" id="textInput" placeholder="Enter your message...">

            <button type="submit" class="msger-send-btn">Send</button>

        </form>

    </section>

    <!-- partial -->

    <script>

        const msgerForm = document.querySelector(".msger-inputarea");

        const msgerInput = document.querySelector(".msger-input");

        const msgerChat = document.querySelector(".msger-chat");

        const BOT\_IMG = "https://image.flaticon.com/icons/svg/327/327779.svg";

        const PERSON\_IMG = "https://image.flaticon.com/icons/svg/145/145867.svg";

        const BOT\_NAME = " zoro";

        const PERSON\_NAME = "You";

        msgerForm.addEventListener("submit", event => {

            event.preventDefault();

            const msgText = msgerInput.value;

            if (!msgText) return;

            appendMessage(PERSON\_NAME, PERSON\_IMG, "right", msgText);

            msgerInput.value = "";

            botResponse(msgText);

        });

        function appendMessage(name, img, side, text) {

            const msgHTML = `

                <div class="msg ${side}-msg">

                    <div class="msg-img" style="background-image: url(${img})"></div>

                    <div class="msg-bubble">

                        <div class="msg-info">

                            <div class="msg-info-name">${name}</div>

                            <div class="msg-info-time">${formatDate(new Date())}</div>

                        </div>

                        <div class="msg-text">${text}</div>

                    </div>

                </div>

            `;

            msgerChat.insertAdjacentHTML("beforeend", msgHTML);

            msgerChat.scrollTop += 500;

        }

        function botResponse(rawText) {

            // Send a POST request to the /chat route to get the chatbot response

            fetch('/chat', {

                method: 'POST',

                headers: {

                    'Content-Type': 'application/json',

                },

                body: JSON.stringify({ message: rawText }),

            })

                .then(response => response.json())

                .then(data => {

                    const msgText = data.response;

                    appendMessage(BOT\_NAME, BOT\_IMG, "left", msgText);

                });

        }

        function formatDate(date) {

            const h = "0" + date.getHours();

            const m = "0" + date.getMinutes();

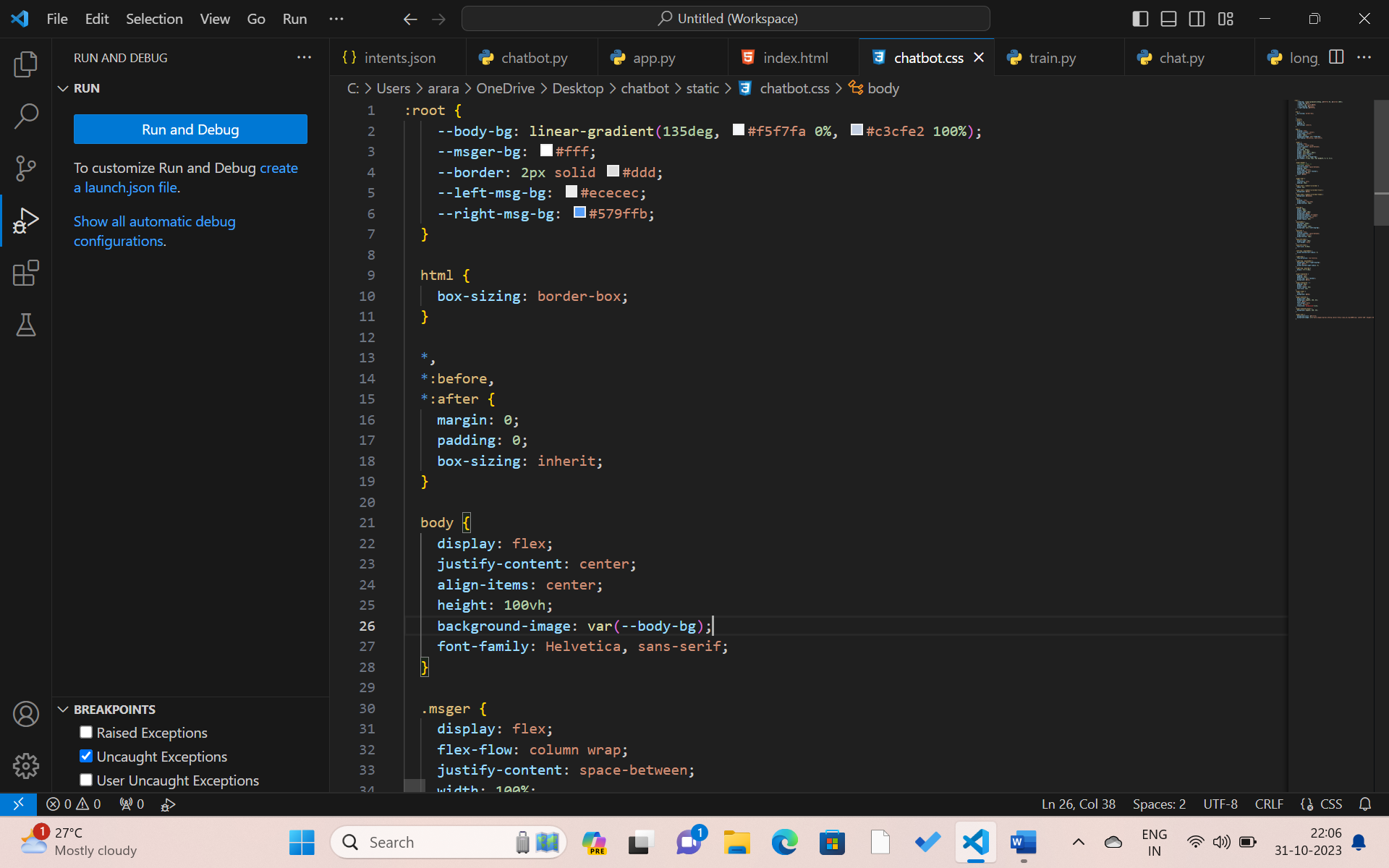
            return `${h.slice(-2)}:${m.slice(-2)}`;

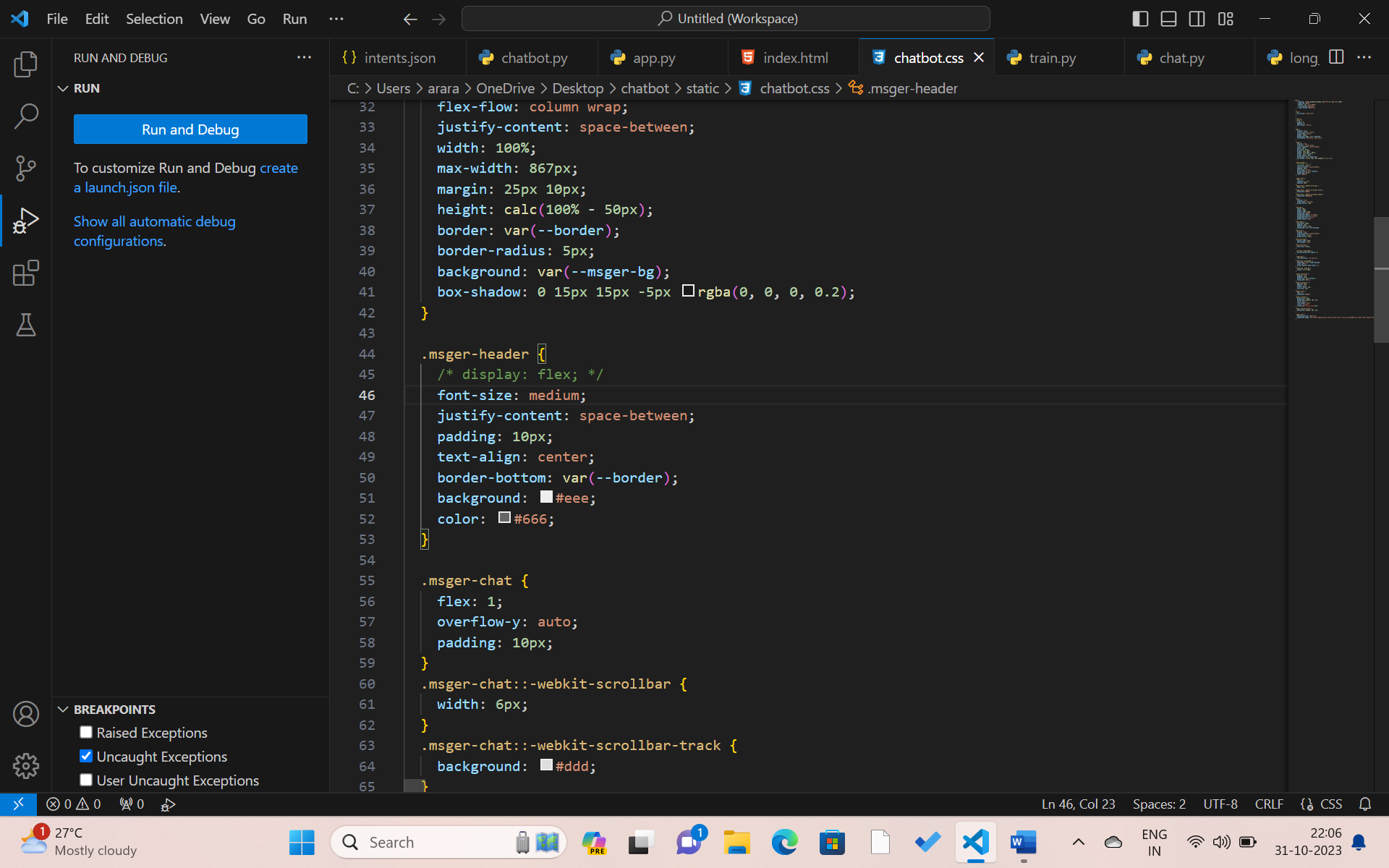
        }

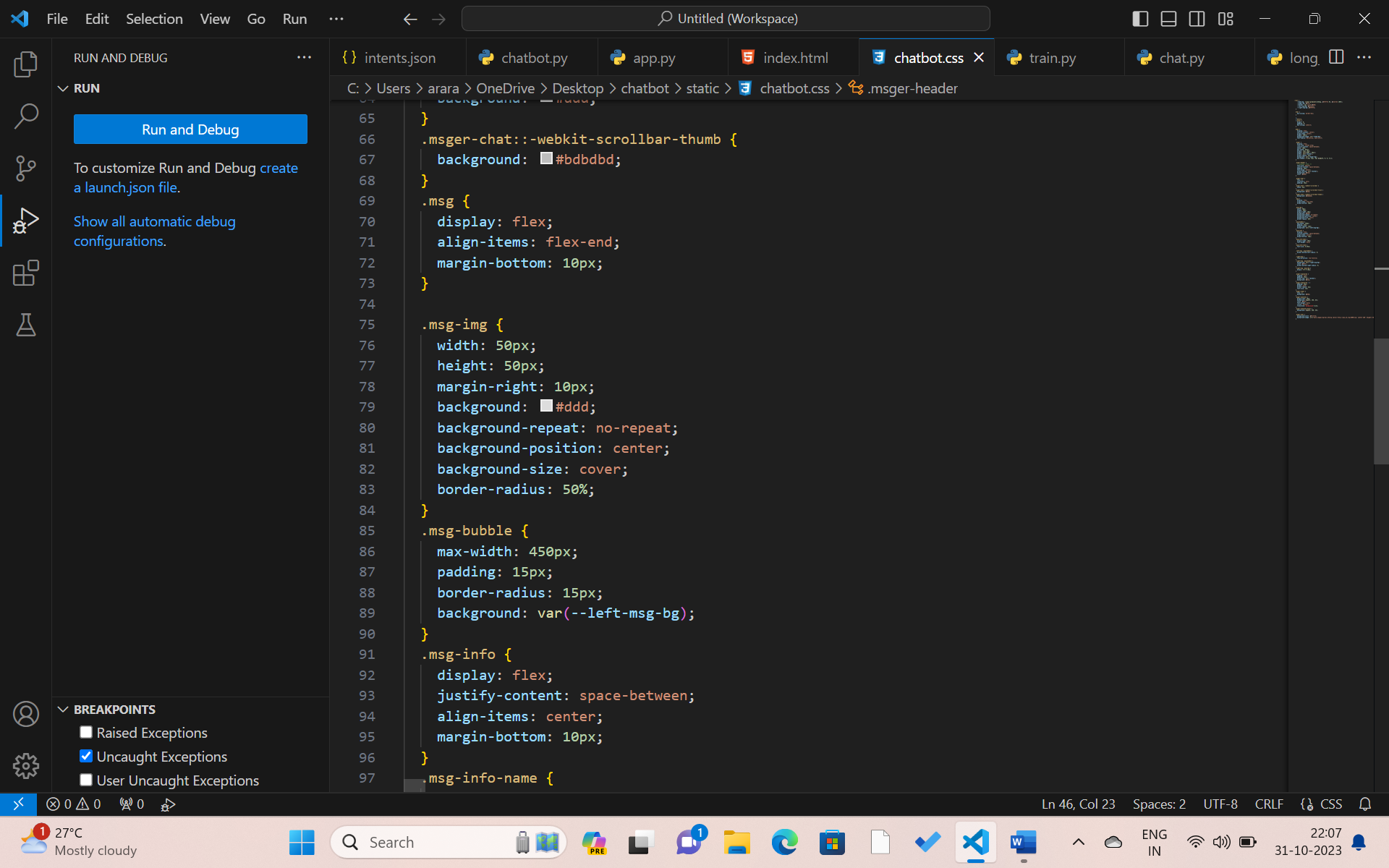
    </script>

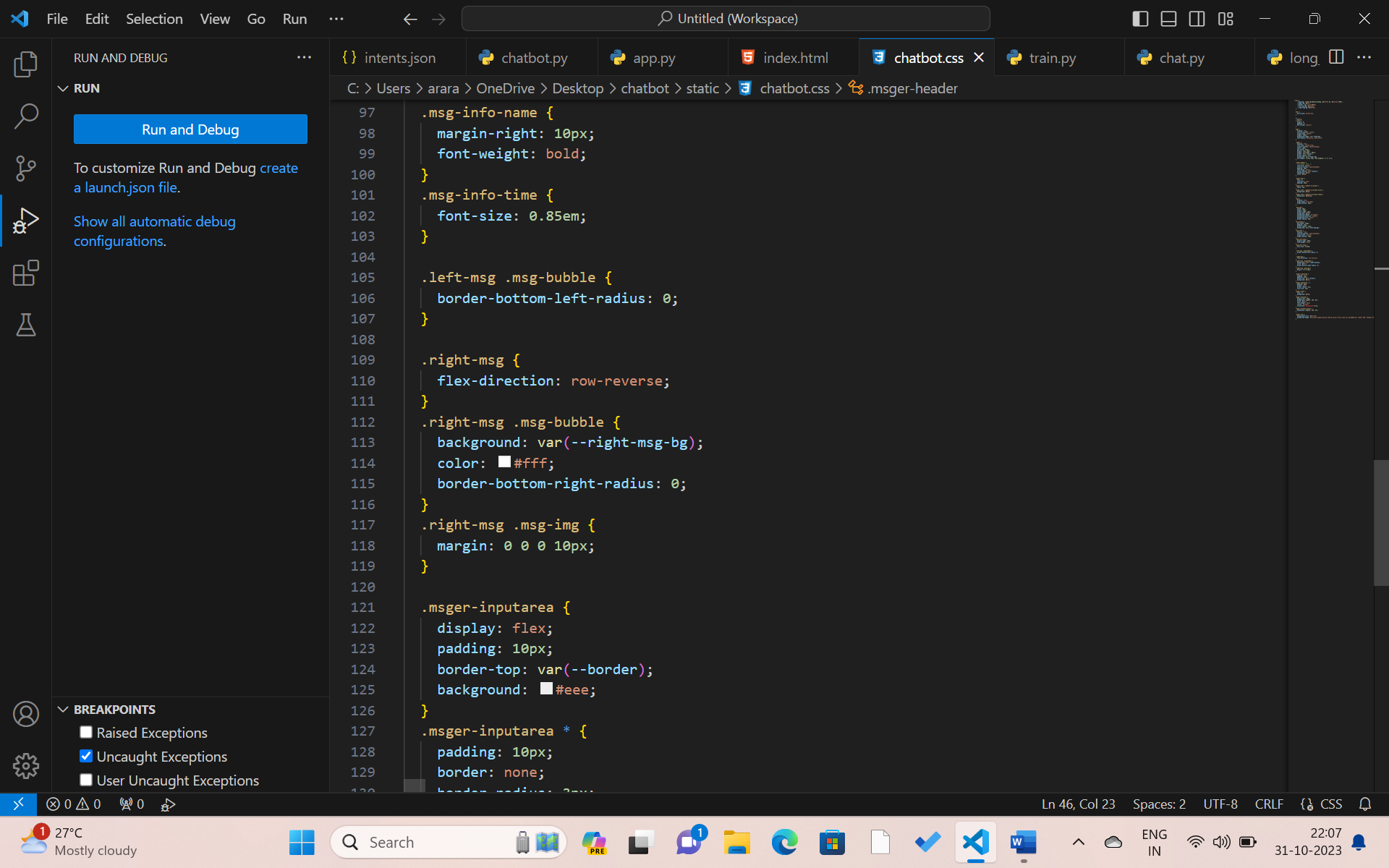
</body>

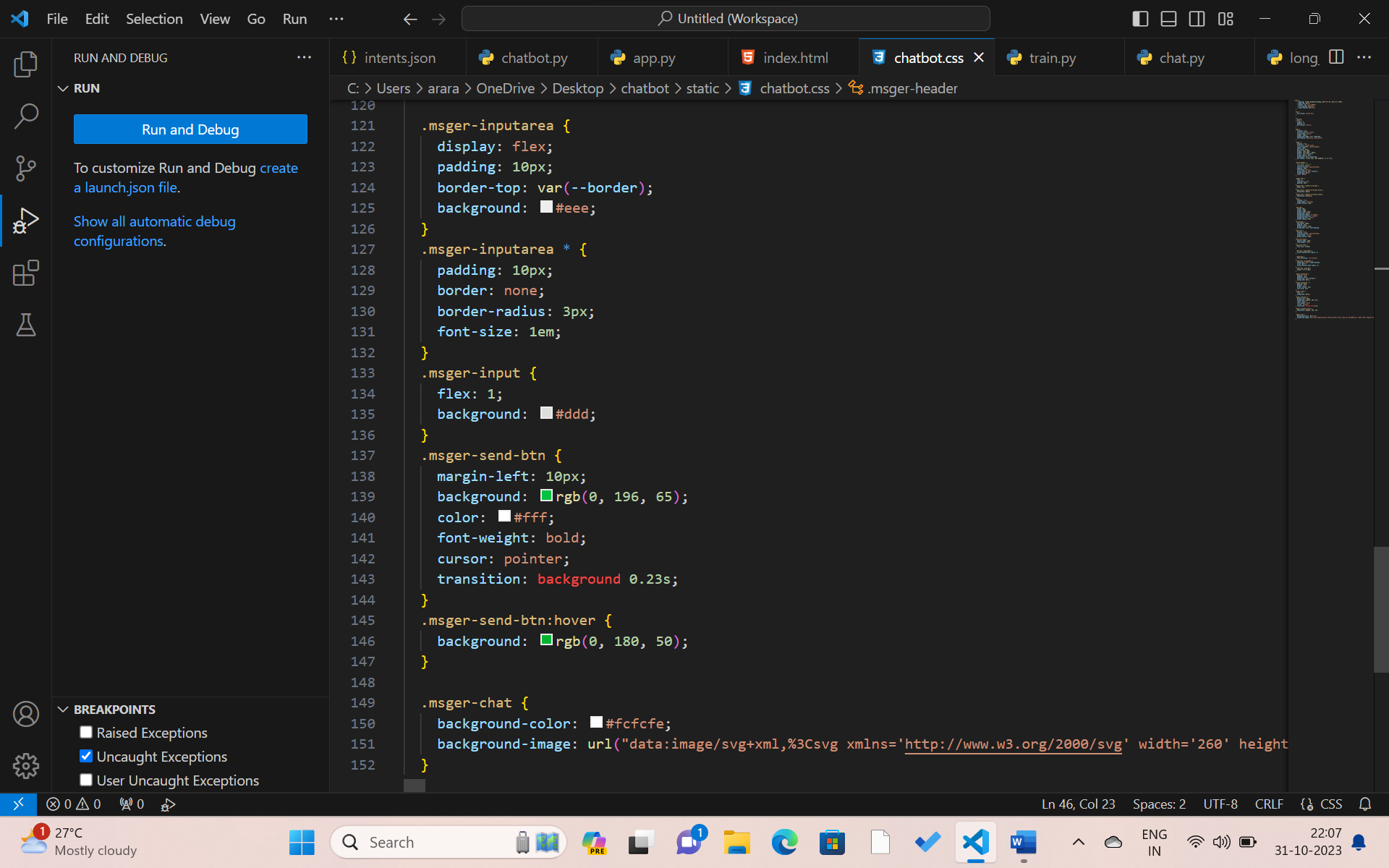
</html>











Output:

