* Make a simple snake game in JavaScript with a leaderboard

/\*---------------------------Snake.html-------------------------\*/

<!DOCTYPE html>

<html>

    <head>

        <meta charset="UTF-8">

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

        <title>Snake Lader</title>

        <link rel="stylesheet" href="snake.css">

        <script src="snake.js"></script>

    </head>

    <body>

        <div class="snake" style="align-items: center;"></div>

        <h1>Snake Lader</h1>

        <canvas id="board"></canvas>

    </body>

</html>

/\*---------------------------Snake.css------------------------\*/

body {

    font-family: 'Courier New', Courier, monospace;

    text-align: center;

}

/\*---------------------------Snake.js-----------------------\*/

//board

var blockSize = 25;

var rows = 20;

var cols = 20;

var board;

var context;

//snake head

var snakeX = blockSize \* 5;

var snakeY = blockSize \* 5;

var velocityX = 0;

var velocityY = 0;

var snakeBody = [];

//food

var foodX;

var foodY;

var gameOver = false;

window.onload = function() {

    board = document.getElementById("board");

    board.height = rows \* blockSize;

    board.width = cols \* blockSize;

    context = board.getContext("2d"); //used for drawing on the board

    placeFood();

    document.addEventListener("keyup", changeDirection);

    // update();

    setInterval(update, 1000/10); //100 milliseconds

}

function update() {

    if (gameOver) {

        return;

    }

    context.fillStyle="black";

    context.fillRect(0, 0, board.width, board.height);

    context.fillStyle="red";

    context.fillRect(foodX, foodY, blockSize, blockSize);

    if (snakeX == foodX && snakeY == foodY) {

        snakeBody.push([foodX, foodY]);

        placeFood();

    }

    for (let i = snakeBody.length-1; i > 0; i--) {

        snakeBody[i] = snakeBody[i-1];

    }

    if (snakeBody.length) {

        snakeBody[0] = [snakeX, snakeY];

    }

    context.fillStyle="lime";

    snakeX += velocityX \* blockSize;

    snakeY += velocityY \* blockSize;

    context.fillRect(snakeX, snakeY, blockSize, blockSize);

    for (let i = 0; i < snakeBody.length; i++) {

        context.fillRect(snakeBody[i][0], snakeBody[i][1], blockSize, blockSize);

    }

    //game over conditions

    if (snakeX < 0 || snakeX > cols\*blockSize || snakeY < 0 || snakeY > rows\*blockSize) {

        gameOver = true;

        alert("Game Over");

    }

    for (let i = 0; i < snakeBody.length; i++) {

        if (snakeX == snakeBody[i][0] && snakeY == snakeBody[i][1]) {

            gameOver = true;

            alert("Game Over");

        }

    }

}

function changeDirection(e) {

    if (e.code == "ArrowUp" && velocityY != 1) {

        velocityX = 0;

        velocityY = -1;

    }

    else if (e.code == "ArrowDown" && velocityY != -1) {

        velocityX = 0;

        velocityY = 1;

    }

    else if (e.code == "ArrowLeft" && velocityX != 1) {

        velocityX = -1;

        velocityY = 0;

    }

    else if (e.code == "ArrowRight" && velocityX != -1) {

        velocityX = 1;

        velocityY = 0;

    }

}

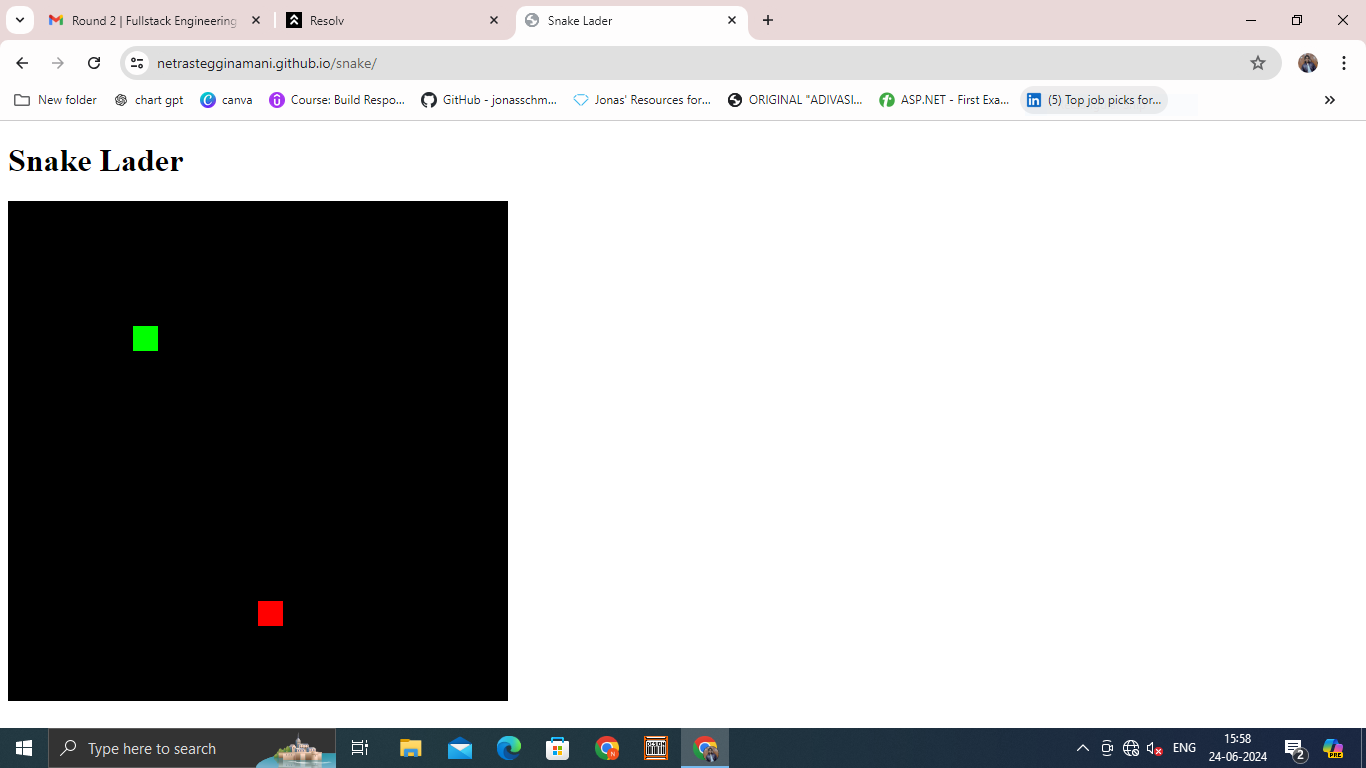
function placeFood() {

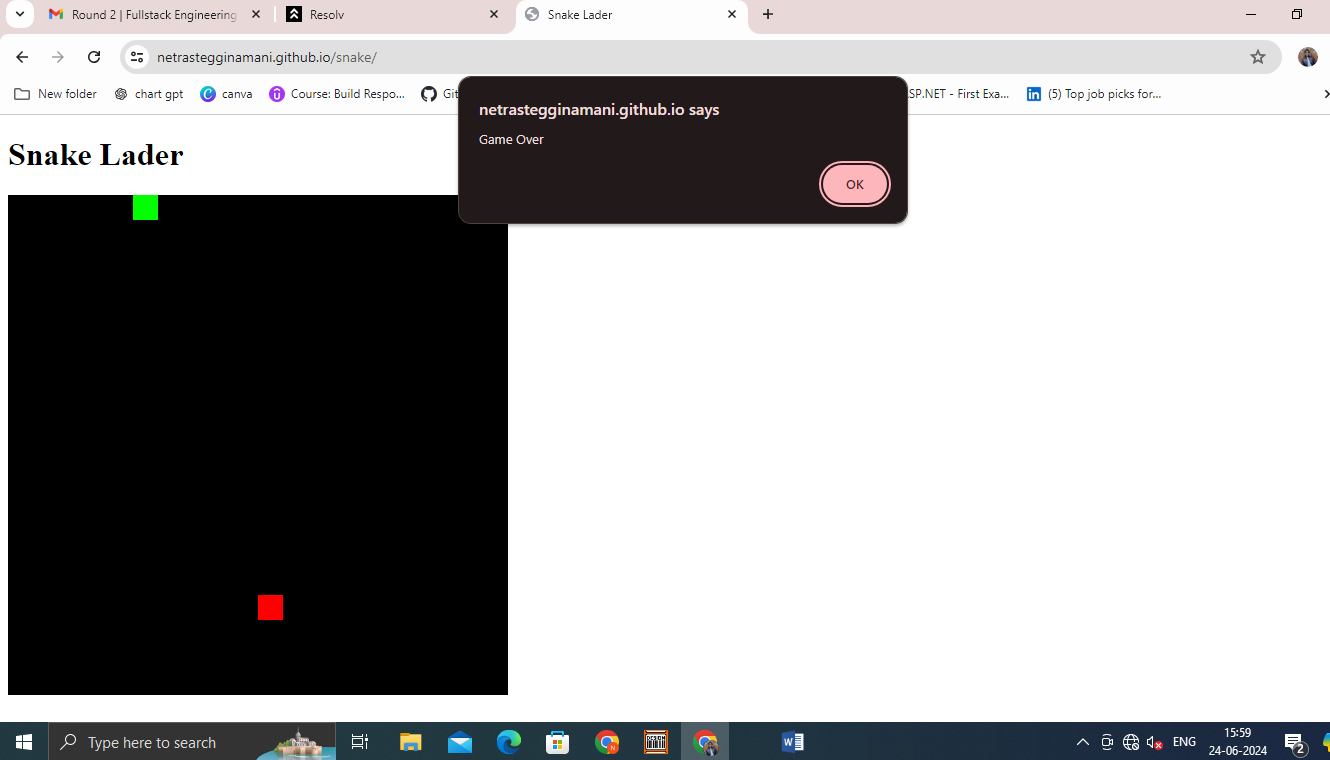
    //(0-1) \* cols -> (0-19.9999) -> (0-19) \* 25

    foodX = Math.floor(Math.random() \* cols) \* blockSize;

    foodY = Math.floor(Math.random() \* rows) \* blockSize;

}





Develop a weather forecasting app.

/\*----------------Weather.html------------\*/

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Weather App</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<div id="weather-container">

<h2>Weather App</h2>

<input type="text" id="city" placeholder="Enter city">

<button onclick="getWeather()">Search</button>

<img id="weather-icon" alt="Weather Icon">

<div id="temp-div"></div>

<div id="weather-info"></div>

<div id="hourly-forecast"></div>

</div>

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

</body>

</html>

/\*----------------Weather.css------------\*/

body {

background: #8C52FF;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

display: flex;

align-items: center;

justify-content: center;

height: 100vh;

margin: 0;

}

#weather-container {

background: rgba(255, 255, 255, 0.3);

max-width: 400px;

padding: 20px;

border-radius: 15px;

box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);

backdrop-filter: blur(10px);

border: 1px solid rgba(255, 255, 255, 0.1);

text-align: center;

}

h2, label, p {

color: #fff;

margin: 8px 0;

}

input {

width: calc(100% - 16px);

padding: 8px;

box-sizing: border-box;

border-radius: 10px;

border: 1px solid white;

margin-top: 20px;

}

button {

background: #debff4;

color: white;

padding: 10px;

border: none;

border-radius: 10px;

cursor: pointer;

margin-top: 20px;

width: 100px;

font-size: 15px;

}

button:hover {

background: #8b48d7;

}

#temp-div p {

font-size: 60px;

margin-top: -30px;

}

#weather-info {

font-size: 20px;

}

#weather-icon {

width: 200px;

height: 200px;

margin: 0 auto 10px;

margin-bottom: 0;

display: none;

}

#hourly-forecast {

margin-top: 50px;

overflow-x: auto;

white-space: nowrap;

display: flex;

justify-content: space-between;

}

.hourly-item {

flex: 0 0 auto;

width: 80px;

display: flex;

flex-direction: column;

align-items: center;

margin-right: 10px;

color: white;

}

.hourly-item img {

width: 30px;

height: 30px;

margin-bottom: 5px;

}

#hourly-heading {

color: #fff;

margin-top: 10px;

}

/\*----------------Weather.js------------\*/

function getWeather() {

const apiKey = 'YOUR-API-KEY';

const city = document.getElementById('city').value;

if (!city) {

alert('Please enter a city');

return;

}

const currentWeatherUrl = `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}`;

const forecastUrl = `https://api.openweathermap.org/data/2.5/forecast?q=${city}&appid=${apiKey}`;

fetch(currentWeatherUrl)

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

.then(data => {

displayWeather(data);

})

.catch(error => {

console.error('Error fetching current weather data:', error);

alert('Error fetching current weather data. Please try again.');

});

fetch(forecastUrl)

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

.then(data => {

displayHourlyForecast(data.list);

})

.catch(error => {

console.error('Error fetching hourly forecast data:', error);

alert('Error fetching hourly forecast data. Please try again.');

});

}

function displayWeather(data) {

const tempDivInfo = document.getElementById('temp-div');

const weatherInfoDiv = document.getElementById('weather-info');

const weatherIcon = document.getElementById('weather-icon');

const hourlyForecastDiv = document.getElementById('hourly-forecast');

// Clear previous content

weatherInfoDiv.innerHTML = '';

hourlyForecastDiv.innerHTML = '';

tempDivInfo.innerHTML = '';

if (data.cod === '404') {

weatherInfoDiv.innerHTML = `<p>${data.message}</p>`;

} else {

const cityName = data.name;

const temperature = Math.round(data.main.temp - 273.15); // Convert to Celsius

const description = data.weather[0].description;

const iconCode = data.weather[0].icon;

const iconUrl = `https://openweathermap.org/img/wn/${iconCode}@4x.png`;

const temperatureHTML = `

<p>${temperature}°C</p>

`;

const weatherHtml = `

<p>${cityName}</p>

<p>${description}</p>

`;

tempDivInfo.innerHTML = temperatureHTML;

weatherInfoDiv.innerHTML = weatherHtml;

weatherIcon.src = iconUrl;

weatherIcon.alt = description;

showImage();

}

}

function displayHourlyForecast(hourlyData) {

const hourlyForecastDiv = document.getElementById('hourly-forecast');

const next24Hours = hourlyData.slice(0, 8); // Display the next 24 hours (3-hour intervals)

next24Hours.forEach(item => {

const dateTime = new Date(item.dt \* 1000); // Convert timestamp to milliseconds

const hour = dateTime.getHours();

const temperature = Math.round(item.main.temp - 273.15); // Convert to Celsius

const iconCode = item.weather[0].icon;

const iconUrl = `https://openweathermap.org/img/wn/${iconCode}.png`;

const hourlyItemHtml = `

<div class="hourly-item">

<span>${hour}:00</span>

<img src="${iconUrl}" alt="Hourly Weather Icon">

<span>${temperature}°C</span>

</div>

`;

hourlyForecastDiv.innerHTML += hourlyItemHtml;

});

}

function showImage() {

const weatherIcon = document.getElementById('weather-icon');

weatherIcon.style.display = 'block'; // Make the image visible once it's loaded

}