WEB DEVELOPMENT

PROJECT

NAME : SEELAM HARI NAGNEDRA

**SOURCE CODE :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>Weather App</title>

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

<link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>

<link rel="shortcut icon" href="https://imgs.search.brave.com/R1spI\_e-nikMqNJkZV1eFXo-Ou389rVd5-n5\_Jn1aH0/rs:fit:860:0:0/g:ce/aHR0cHM6Ly9jZG4u/aWNvbi1pY29ucy5j/b20vaWNvbnMyLzE1/MjcvUE5HLzUxMi90/aHVuZGVyXzEwNjY5/MS5wbmc" type="image/x-icon">

<link href="https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,7

00;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap" rel="stylesheet">

<style>

@import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;500;600;700&display=swap');

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Poppins', Arial, Helvetica, sans-serif;

}

body {

background: #131216;

margin: 0;

padding: 0;

}

p{

color: white;

font-size: 20px;

margin-bottom: 20px;

}

h2{

text-align: center;

padding: 18px 0;

color: white;

font-size: 50px;

font-weight: bold;

}

.container {

display: block;

gap: 35px;

padding: 30px;

width: 100%;

margin: 0 auto;

}

.weather-input {

width: 400px;

}

.weather-input input {

color: white;

height: 46px;

width: 90%;

outline: none;

font-size: 1.3rem;

padding: 0 17px;

margin: 10px 0 20px 0;

border-radius: 4px;

border:none;

outline: none;

background-color: transparent;

}

.weather-input input:focus {

padding: 0 16px;

border:none;

}

.weather-input .separator {

height: 1px;

width: 100%;

margin: 25px 0;

background: #BBBBBB;

display: flex;

align-items: center;

justify-content: center;

}

.weather-input .separator::before{

content: "or";

color: #6C757D;

font-size: 1.18rem;

padding: 0 15px;

margin-top: -4px;

background: #E3F2FD;

}

.location-btn {

width: 100%;

padding: 10px 0;

cursor: pointer;

outline: none;

border: none;

border-radius: 4px;

font-size: 1rem;

color: #fff;

background: #1D1B1F;

transition: 0.2s ease;

}

.search-btn{

border-radius: 50%;

height:46px;

width: 46px;

padding: 10px;

font-size: 25px;

border: none;

color: black;

background-color: white;

display: flex;

align-items: center;

cursor: pointer;

}

.search{

display: flex;

justify-content: center;

align-items: center;

border: 1px solid #ccc;

border-radius: 7px;

padding: 0 10px;

margin-top: 5px;

}

.weather-input .search-btn:hover {

background: white;

transition: .2s ease;

}

.weather-input .location-btn {

background: #6C757D;

}

.weather-input .location-btn:hover {

background: #5c636a;

}

.weather-data {

margin-top: 40px;

width: 100%;

}

.weather-data .current-weather {

color: black;

background:#5c636a ;

border-radius: 5px;

padding: 20px 70px 20px 20px;

display: flex;

justify-content: space-between;

}

.current-weather h2 {

font-weight: 700;

font-size: 1.7rem;

}

.weather-data h6 {

margin-top: 12px;

font-size: 1rem;

font-weight: 500;

}

.current-weather .icon {

text-align: center;

}

.current-weather .icon img {

max-width: 120px;

margin-top: -15px;

}

.current-weather .icon h6 {

margin-top: -10px;

text-transform: capitalize;

}

.days-forecast h2 {

margin: 20px 0;

font-size: 1.5rem;

}

.days-forecast .weather-cards {

display: flex;

gap: 20px;

}

.weather-cards .card {

color: black;

padding: 18px 16px;

list-style: none;

width: calc(100% / 5);

background: #6C757D;

border-radius: 5px;

}

.weather-cards .card h3 {

font-size: 1.3rem;

font-weight: 600;

}

.weather-cards .card img {

max-width: 70px;

margin: 5px 0 -12px 0;

}

@media (max-width: 1400px) {

.weather-data .current-weather {

padding: 20px;

}

.weather-cards {

flex-wrap: wrap;

}

.weather-cards .card {

width: calc(100% / 4 - 15px);

}

}

@media (max-width: 1200px) {

.weather-cards .card {

width: calc(100% / 3 - 15px);

}

}

@media (max-width: 950px) {

.weather-input {

width: 450px;

}

.weather-cards .card {

width: calc(100% / 2 - 10px);

}

}

@media (max-width: 750px) {

h1 {

font-size: 1.45rem;

padding: 16px 0;

}

.container {

flex-wrap: wrap;

padding: 15px;

}

.weather-input {

width: 100%;

}

.weather-data h2 {

font-size: 1.35rem;

}

}

.first{

width: 100%;

text-align: center;

display: flex;

justify-content: center;

}

</style>

</head>

<body>

<h2>Weather App</h2>

<div class="container">

<div class="first">

<div class="weather-input">

<p>Search City By Name</p>

<div class="search">

<input class="city-input" type="text" placeholder="E.g., New York, London, Tokyo">

<button class="search-btn"><i class='bx bx-search'></i></button>

</div>

<div class="separator"></div>

<button class="location-btn">Use Current Location</button>

</div>

</div>

<div class="weather-data">

<div class="current-weather" id="c\_w" style="display: none;">

<div class="details">

<h2></h2>

<h6></h6>

<h6></h6>

<h6></h6>

</div>

</div>

<div class="days-forecast">

<h2 id="headra" style="display: none;">Weather Forecast For Next Days</h2>

<ul id="u\_l" class="weather-cards" style="display: none;">

<li class="card">

</li>

</ul>

</div>

</div>

</div>

<script>

const cityInput = document.querySelector(".city-input");

const searchButton = document.querySelector(".search-btn");

const locationButton = document.querySelector(".location-btn");

const currentWeatherDiv = document.querySelector(".current-weather");

const weatherCardsDiv = document.querySelector(".weather-cards");

const API\_KEY = "2adb2388f50a1bd68e2536b84d1196e6";

document.querySelector(".search-btn").onclick= function(){

if(!cityInput.value==""){

document.getElementById("u\_l").style.display="flex";

document.getElementById("c\_w").style.display="flex";

document.getElementById("headra").style.display="block";

}}

const createWeatherCard = (cityName, weatherItem, index) => {

if(index === 0) {

return `<div class="details">

<h2>${cityName} (${weatherItem.dt\_txt.split(" ")[0]})</h2>

<h6>Temperature: ${(weatherItem.main.temp - 273.15).toFixed(2)}°C</h6>

<h6>Wind: ${weatherItem.wind.speed} M/S</h6>

<h6>Humidity: ${weatherItem.main.humidity}%</h6>

</div>

<div class="icon">

<img src="https://openweathermap.org/img/wn/${weatherItem.weather[0].icon}@4x.png" alt="weather-icon">

<h6>${weatherItem.weather[0].description}</h6>

</div>`;

} else {

return `<li class="card">

<h3>(${weatherItem.dt\_txt.split(" ")[0]})</h3>

<img src="https://openweathermap.org/img/wn/${weatherItem.weather[0].icon}@4x.png" alt="weather-icon">

<h6>Temp: ${(weatherItem.main.temp - 273.15).toFixed(2)}°C</h6>

<h6>Wind: ${weatherItem.wind.speed} M/S</h6>

<h6>Humidity: ${weatherItem.main.humidity}%</h6>

</li>`;

}

}

const getWeatherDetails = (cityName, latitude, longitude) => {

const WEATHER\_API\_URL = `https://api.openweathermap.org/data/2.5/forecast?lat=${latitude}&lon=${longitude}&appid=${API\_KEY}`;

fetch(WEATHER\_API\_URL).then(response => response.json()).then(data => {

const uniqueForecastDays = [];

const fiveDaysForecast = data.list.filter(forecast => {

const forecastDate = new Date(forecast.dt\_txt).getDate();

if (!uniqueForecastDays.includes(forecastDate)) {

return uniqueForecastDays.push(forecastDate);

}

});

cityInput.value = "";

currentWeatherDiv.innerHTML = "";

weatherCardsDiv.innerHTML = "";

fiveDaysForecast.forEach((weatherItem, index) => {

const html = createWeatherCard(cityName, weatherItem, index);

if (index === 0) {

currentWeatherDiv.insertAdjacentHTML("beforeend", html);

} else {

weatherCardsDiv.insertAdjacentHTML("beforeend", html);

}

});

}).catch(() => {

alert("An error occurred while fetching the weather forecast!");

});

}

const getCityCoordinates = () => {

const cityName = cityInput.value.trim();

if (cityName === "") return;

const API\_URL = `https://api.openweathermap.org/geo/1.0/direct?q=${cityName}&limit=1&appid=${API\_KEY}`;

fetch(API\_URL).then(response => response.json()).then(data => {

if (!data.length) return alert(`No coordinates found for ${cityName}`);

const { lat, lon, name } = data[0];

getWeatherDetails(name, lat, lon);

}).catch(() => {

alert("An error occurred while fetching the coordinates!");

});

}

const getUserCoordinates = () => {

navigator.geolocation.getCurrentPosition(

position => {

const { latitude, longitude } = position.coords;

const API\_URL = `https://api.openweathermap.org/geo/1.0/reverse?lat=${latitude}&lon=${longitude}&limit=1&appid=${API\_KEY}`;

fetch(API\_URL).then(response => response.json()).then(data => {

const { name } = data[0];

getWeatherDetails(name, latitude, longitude);

}).catch(() => {

alert("An error occurred while fetching the city name!");

});

},

error => {

if (error.code === error.PERMISSION\_DENIED) {

alert("Geolocation request denied. Please reset location permission to grant access again.");

} else {

alert("Geolocation request error. Please reset location permission.");

}

});

}

locationButton.addEventListener("click", getUserCoordinates);

searchButton.addEventListener("click", getCityCoordinates);

cityInput.addEventListener("keyup", e => e.key === "Enter" && getCityCoordinates());

</script>

</body>

</html>