WEATHER DETAILS OF A CITY

A Real-Time / Field-Based Research Project (22DT284) report submitted to the Jawaharlal Nehru Technological University, Hyderabad

Submitted by

P RAKESH 22B81A6694
K RAVI TEJ 22B81A6696
P SANDEEP 22B81A66B0

Under the guidance of
Mrs.K.Sowjanya Bharathi

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

CVR COLLEGE OF ENGINEERING

(***An Autonomous Institution, NAAC Accredited and Affiliated to JNTUH, Hyderabad***)
Vastunagar, Mangalpalli(V), Ibrahimpatnam(M),
Rangareddy (D), Telangana- 501 510

JUNE 2024

CVR COLLEGE OF ENGINEERING

(*An Autonomous Institution , NAAC Accredited and Affiliated to JNTUH, Hyderabad*)
Vastunagar,Mangalpalli(V),Ibrahimpatnam(M),
Rangareddy (D), Telangana- 501 510

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

CERTIFICATE

This is to certify that the Real time/ Field-Based research project (22DT284) report entitled **"WEATHER DETAILS OF A CITY"** is a record of work carried out by **P RAKESH, K RAVI TEJ, P SANDEEP** submitted to Department of **CSE (AI & ML),** CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University, Hyderabad during the year 2023-2024.

Mrs.K.Sowjanya Bharathi **Project Guide**	**Dr.M.Surya Bhupal Rao** Associate Professor & **Project Coordinator**	**Dr.Lakshmi H N** Professor & HOD

DECLARATION

We hereby declare that the Real time/ Field-Based research project (22DT284) report entitled **"WEATHER DETAILS OF A CITY"** is an original work done and submitted to the Department of **CSE (AI & ML)**, CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University, Hyderabad and it is a record of bonafide project work carried out under the guidance of**, Mrs.K.Sowjanya Bharathi.**

We further declare that the work reported in this project has not been submitted, either in part or in full, for the award of any other degree or diploma in this Institute or any other Institute or University.

Signature of the Student
P RAKESH
22B81A6694

Signature of the Student
K RAVI TEJ
22B81A6696

Signature of the Student
P SANDEEP
22B81A66B0

Date:
Place:

ACKNOWLEDGEMENT

We are thankful and fortunate enough to get constant encouragement, support, and guidance from all **Teaching staff of CSE (AI & ML)** Department who helped us in successfully completing this project work.

We respect and thank our internal guide **Mrs.K.Soujanya Bharathi**, for giving us all the support and guidance, which made us complete the project duly.

We thank the Project Review Committee **Dr. M. Surya Bhupal Rao**, Project Coordinator and **Ms. K. Anuradha,Mrs.K.Vineela Krishna** for their valuable guidance and support which helped us to complete the project work successfully.

We would like to thank **Dr. H. N. Lakshmi**, Professor & Head of the Department, for allowing us to do this project and extending support and guidance.

We thank our Vice-Principal **Prof. L. C. Siva Reddy** for providing excellent computing facilities and a disciplined atmosphere for doing our work.

We wish a deep sense of gratitude and heartfelt thanks to **Dr. Rama Mohan Reddy**, Principal and the Management for providing excellent lab facilities and tools. Finally, we thank all those guidance helpful to us in this regard.

LIST OF FIGURES AND TABLES

LIST OF ABBREVIATIONS

API	Application Programming Interface
CSS	Cascading Style Sheets
HTML	Hyper Text Markup Language
IDE	Integrated Development Environment
JS	Java Script
PDF	Portable Document Format
RAM	Random Access Memory

ABSTRACT

A weather forecasting application is a software tool designed to provide users with up-to-date and accurate information about current and future weather conditions.

Weather forecasting is one of the most scientifically and technologically challenging problems around the world in the last century. To make an accurate prediction is indeed, one of the major challenges that meteorologists are facing all over the world. To predict the conditions of the atmosphere for a given location, Weather Forecasting is used. Weather forecasting is made by collecting numerous data predicted by very proper understanding of the collected data. Weather simply refers to the condition of air on the earth at given place and time.

It is a continuous, data-intensive, multidimensional, dynamic and chaotic process. These processes make weather forecasting a formidable challenge. Forecasting is the process of estimation in unknown situations from the historical data. It is the application of science and technology. Weather forecast is more helpful for people as it predicts how the future weather is going to be and people may plan accordingly.

Farmers will be most beneficial one's as they may know the rainfall prediction accordingly. The weather forecast can be done in many ways like using the previous data or analyzing the current clouds.

The application provides a seamless experience for users to access critical weather information, ultimately contributing to improved safety and enhanced decision-making in response to ever-changing weather patterns.

TABLE OF CONTENTS

INTRODUCTION

1.1 Overview

Weather prediction is the application of technology to predict the action of the atmosphere for a given location. It is becoming increasingly vital for business, agriculturists, farmers, disaster management and related organizations to understand the natural phenomena. The art of weather prediction began with using the reoccurring astronomical and meteorological events to help them to monitor the seasonal changes in the weather. Throughout these centuries, this attempt is madeto produce forecasts based on weather changes and personal observations. Weather prediction has been one of the most interesting domains. The scientists are been trying to forecast the meteorological data using a big set of methods, some of them more accurate than others.
Weather forecasting is anessential application in meteorology and has been one of the most scientifically challenging problems around the world. Weather condition is a state of atmosphere at given time and the weather parameters are temperature, humidity, and wind speed. The accuracy of the prediction depends on knowledge of prevailing weather condition over large areas. Weather is the non-linear and dynamic process as it varies day to day even minute to minute; the big challenge of weather is data intensive and the frenzied nature.

1.2 Aim

People can get accurate weather information is the main aim of this application. The important issue faced in our country is climatic changes and that can be resolved by our application “WEATHER FORECASTING SYSTEM”. The goal of weather prediction is to provide information. People and organizations can use to reduce weather related loses and enhanced societal benefits, including protection of life and property, public health and support of economic prosperity and quality of life.

1.3 Problem Statement

Current weather apps lack simplicity and speed, making it challenging for users to quickly access accurate information for specific cities. There is a need for a streamlined web application that prioritizes user-friendly interfaces, delivering real-time, precise weather details for informed decision-making in travel, planning, and daily activities.

1.4 Usage

Overall, live weather websites play a crucial role in everyday life, providing valuable information and insights that help individuals, businesses, and communities make informed decisions and stay safe in various weather conditions.

1.5 Key Features

1.Real-time Updates:

The website provides up-to-date weather information, including current temperature, humidity, wind speed, and direction.

2.Weather forecasting improves transportation safety: It is particularly useful for people, as it predicts future weather conditions and allows them to plan accordingly.

3.Location Access:

Developed a user friendly location access allowing users to allow their locations for getting the weather updates of their particular location.

4.Weather Data Retrieval:

Connect to a Weather API to fetch real-time weather data and provide users with the ability to specify their location for weather information retrieval.

5.User-Friendly Interface:

- Customizable dashboards to display preferred weather information.
- Mobile-friendly applications for access on smartphones and tablets.

1.6 Objectives

The objective of the Todays weather project is to design and implement an efficient and user-friendly system that helps user to know about weather details of any city using its name only.

1.7 Software & Hardware Specifications

1.8

Requirement specification is used in programming contexts to describe a system's functionality and understand the client's needs within an organisation. The purpose of an SRS is to comprehensively plan the requirements of a system or subsystem. It outlines the necessary conditions for meeting the client's needs at a specific point in time before finalizing the work.

Software Specifications

- Operating System – Windows
- Languages – HTML, CSS, JavaScript
- Tools/IDE – Visual Studio Code

Hardware Specifications

- RAM – 8 GB
- Processor – Core i5

DESIGN METHODOLOGY

2.1 Existing System

In ancient times, forecasting was mostly based on weather pattern observation. Over the years, the study of weather patterns has resulted in various techniques for rainfall forecasting. Present rainfall forecasting embodies a combination of computer models, interpretation, and an acquaintance of weather patterns. So, we created a web application to predict the weather in accuracy format to help user can get the weather detail.

2.2 Proposed System

Weather report application is a web based application through which you will able to get all the reports related to weather forecasting of any locations and able to present its weather details such as Temperature, Direction of Wind, Humidity etc using weather api with accurate weather data.

2.3 Scope of the System

Weather forecasts are made by collecting as much data as possible about the current state of the atmosphere (particularly the temperature, humidity and wind) to determine how the atmosphere evolves in the future.

However, the chaotic nature of the atmosphere makes the forecasts less accurate as therange of the forecast increases.

Traditional observations made at the surface of atmospheric pressure, temperature, wind speed, wind direction, humidity, precipitation are collected routinely from trained observers, automatic weather stations or buoys.During the data assimilation process,information gained from the observations is used In conjunction with a numericalmodel's most recent forecast for the time that observations were made to produce themeteorological analysis.

The complicated equations which govern how the state of afluid changes with time require supercomputers to solve them..

2.4 Feasibility Study

The primary objective of the Feasibility Study is to evaluate the economic sustainability of the proposed enterprise. The feasibility study must address the inquiry of whether the idea is economically viable. The study should include a comprehensive examination of the business [10]. The result of the feasibility study will determine whether to continue with the proposed endeavor. If the study yields a favorable outcome, the cooperative can proceed with the formulation of a business plan. A feasibility study should assess three primary domains - technical aspects, economic viability and operational feasibility.

2.4.1 Technical Feasibility

Evaluate the technologies needed for front-end (HTML5, CSS3, JavaScript frameworks like React, Angular, Vue.js) and back-end development (Node.js, Python, PHP). Assess the compatibility and reliability of weather APIs (e.g., OpenWeatherMap, Weather API) for real-time data. Ensure the team has the necessary skills or identify the need for hiring or training.

2.4.2 Economic Feasibility

Estimate the costs involved in development, API subscriptions, hosting, domain registration, and ongoing maintenance. Compare the estimated costs against the available budget. Explore monetization options such as ads, premium features, or partnerships.

2.4.3 Operational Feasibility

Assess the availability of human resources, including developers, designers, and support staff.
Define the processes for regular updates, data accuracy checks, and system maintenance.

Plan for customer support and troubleshooting for end-users. Ensure the system can handle growth in user base and data volume without performance degradation.

2.5 Use Case Diagram.

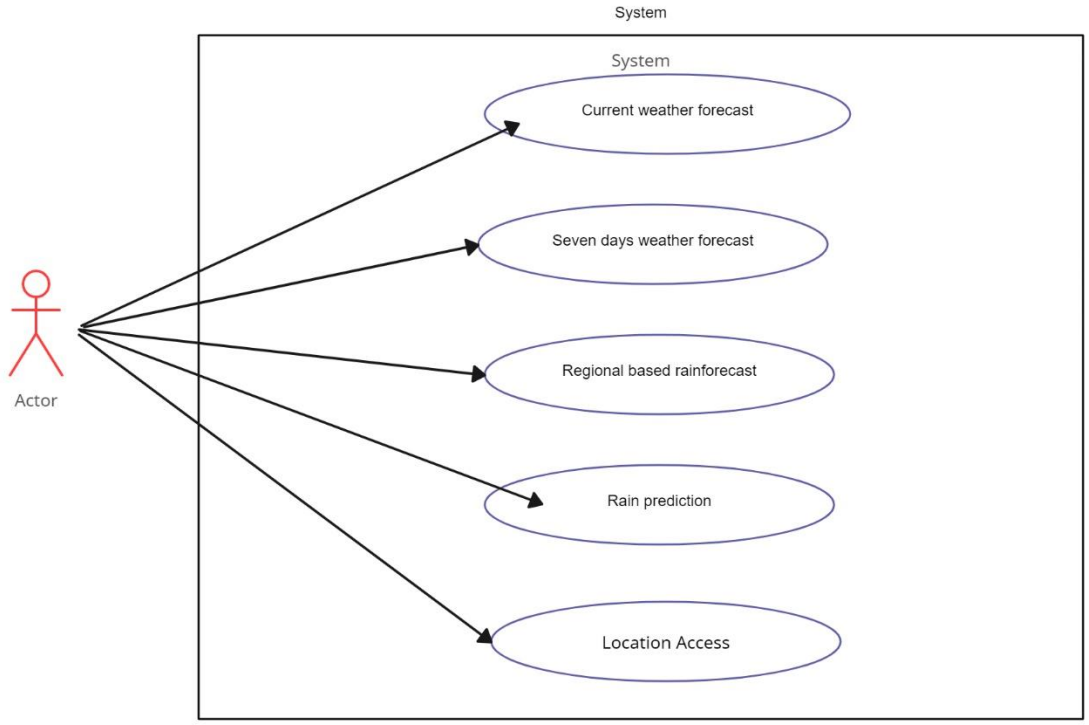
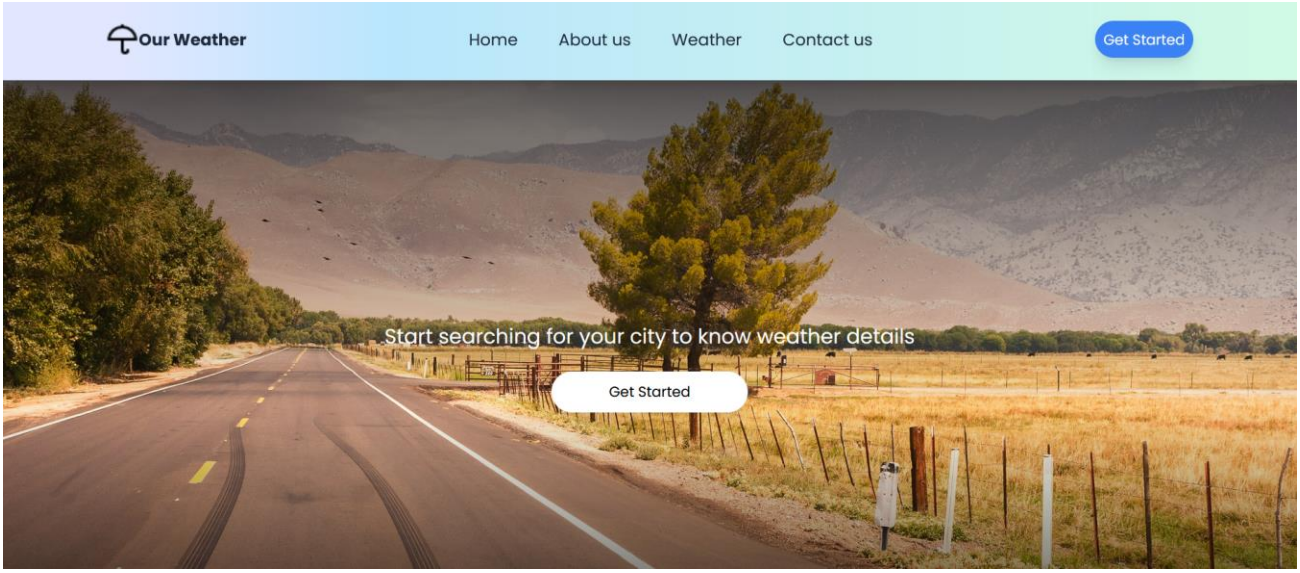


Figure 1.2 : USE CASE DIAGRAM ON LIVE WEATHER WEBSITE

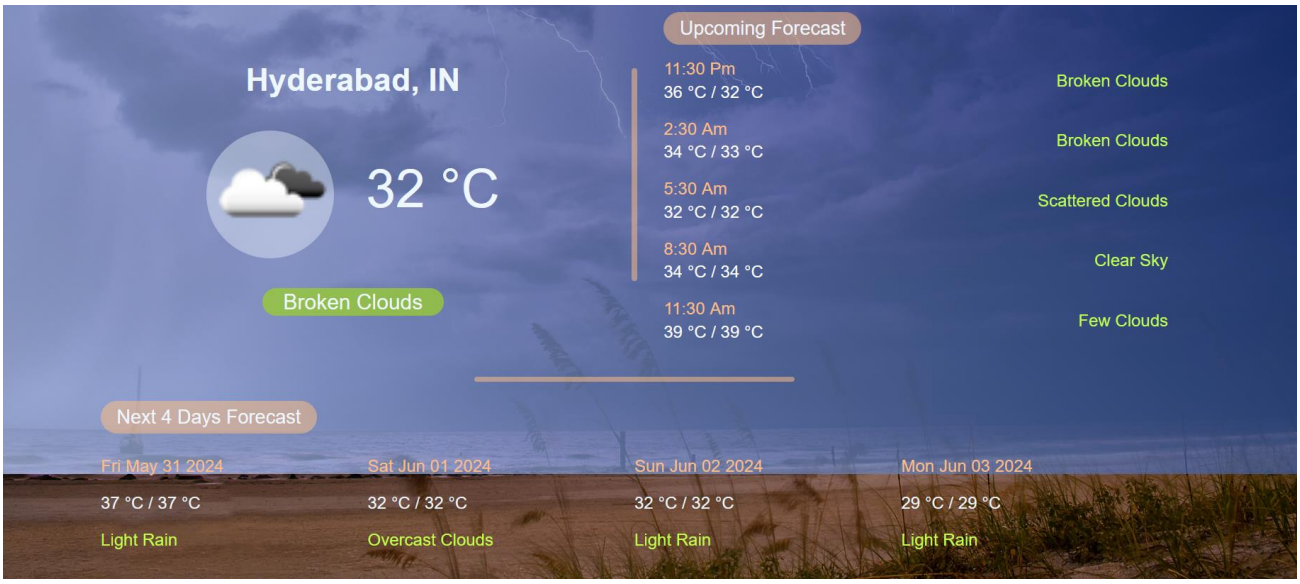
Use Case Diagram Explanation :When the end user request the weather forecasting information from the web application and it will directly go the API call. API call is the process of a user application submitting a request to an API and that API retrieving the requested data from the external server or program and delivering it back to the client. After API call it will go to the cloud database in which all the weather data information are saved in a database. From database it will go into JSON data. JSON data is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute value pairs and arrays. It is a common data format with diverse uses in electronic data interchange.

HomePage Diagram:



Result page1 Diagram:

Result page2 Diagram:



2.6 Design Methodology

The methodologies we have used to develop the user interface of the project are as follows

1. **HTML5:** This is the latest version of Hyper Text Markup Language (HTML) with new elements, new attributes and behaviors, and a large set of technologies which allows the building of more diverse and powerful websites and applications.
2. **CSS3:** This is the latest standard of Cascading Style Sheets (CSS) which is basically used for styling the web pages to make it look attractive. The CSS3 includes more features like rounded corners, text shadows, background gradients, opacity.
3. **JavaScript:** It is a programming language of HTML and web. It is a prototype based, dynamic language and it supports object-oriented styles and imperative styles.

IMPLEMENTATION & TESTING

3.1 Code Snippets

Function getweatherData():

```
function weatherReport(data) {
  var urlcast =
    `http://api.openweathermap.org/data/2.5/forecast?q=${data.name}&` +
    `appid=${apikey}`;

  fetch(urlcast)
    .then((res) => {
      return res.json();
    })
    .then((forecast) => {
      console.log(forecast.city);
      hourForecast(forecast);
      dayForecast(forecast);

      console.log(data);
      document.getElementById("city").innerText =
        data.name + ", " + data.sys.country;
      console.log(data.name, data.sys.country);

      console.log(Math.floor(data.main.temp - 273));
      document.getElementById("temperature").innerText =
        Math.floor(data.main.temp - 273) + " °C";

      document.getElementById("clouds").innerText = data.weather[0].description;
      console.log(data.weather[0].description);

      let icon1 = data.weather[0].icon;
      let iconurl = "http://api.openweathermap.org/img/w/" + icon1 + ".png";
      document.getElementById("img").src = iconurl;
    });
}
```

Function SearchByCity Name :

```
function searchByCity() {
  var place = document.getElementById("input").value;
  var urlsearch =
    `http://api.openweathermap.org/data/2.5/weather?q=${place}&` +
    `appid=${apikey}`;

  fetch(urlsearch)
```

```
    .then((res) => {
      return res.json();
    })
    .then((data) => {
      console.log(data);
      weatherReport(data);
    });
  document.getElementById("input").value = "";
}
```

Function for accessing location of a user:

```
window.addEventListener("load", () => {
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition((position) => {
      let lon = position.coords.longitude;
      let lat = position.coords.latitude;
      const url =
        `http://api.openweathermap.org/data/2.5/weather?lat=${lat}&` +
        `lon=${lon}&appid=${apikey}`;

      fetch(url)
        .then((res) => {
          return res.json();
        })
        .then((data) => {
          console.log(data);
          console.log(new Date().getTime());
          var dat = new Date(data.dt);
          console.log(dat.toLocaleString(undefined, "Asia/Kolkata"));
          console.log(new Date().getMinutes());
          weatherReport(data);
        });
    });
  }
```

Function for hour forcast:

```
function hourForecast(forecast) {
  document.querySelector(".templist").innerHTML = "";
  for (let i = 0; i < 5; i++) {
    var date = new Date(forecast.list[i].dt * 1000);
    console.log(
      date.toLocaleTimeString(undefined, "Asia/Kolkata").replace(":00", "")
    );

    let hourR = document.createElement("div");
    hourR.setAttribute("class", "next");
```

```
    let div = document.createElement("div");
    let time = document.createElement("p");
    time.setAttribute("class", "time");
    time.innerText = date
      .toLocaleTimeString(undefined, "Asia/Kolkata")
      .replace(":00", "");

    let temp = document.createElement("p");
    temp.innerText =
      Math.floor(forecast.list[i].main.temp_max - 273) +
      " °C" +
      " / " +
      Math.floor(forecast.list[i].main.temp_min - 273) +
      " °C";

    div.appendChild(time);
    div.appendChild(temp);

    let desc = document.createElement("p");
    desc.setAttribute("class", "desc");
    desc.innerText = forecast.list[i].weather[0].description;

    hourR.appendChild(div);
    hourR.appendChild(desc);
    document.querySelector(".templist").appendChild(hourR);
  }
}
```

Function for dayForcast:

```
function dayForecast(forecast) {
  document.querySelector(".weekF").innerHTML = "";
  for (let i = 8; i < forecast.list.length; i += 8) {
    console.log(forecast.list[i]);
    let div = document.createElement("div");
    div.setAttribute("class", "dayF");

    let day = document.createElement("p");
    day.setAttribute("class", "date");
    day.innerText = new Date(forecast.list[i].dt * 1000).toDateString(
      undefined,
      "Asia/Kolkata"
    );
    div.appendChild(day);

    let temp = document.createElement("p");
    temp.innerText =
```

```
      Math.floor(forecast.list[i].main.temp_max - 273) +
      " °C" +
      " / " +
      Math.floor(forecast.list[i].main.temp_min - 273) +
      " °C";
    div.appendChild(temp);

    let description = document.createElement("p");
    description.setAttribute("class", "desc");
    description.innerText = forecast.list[i].weather[0].description;
    div.appendChild(description);

    document.querySelector(".weekF").appendChild(div);
  }
}
```

CONCLUSION

In the era of the global warming, research in weather measurement, monitoring and forecasting are become more and more relevant. This research demonstrates the design and implementation of an affordable mini weather monitoring system that ensures flexibility, portability, scability and user friendly operations which can provide data of some weather variables including temperature, humidity and pressure. With the advacement of technology weather forecasting has developed to its level best, but there is yet to develop, as far as a nature is so unpredictable. Weather forecasts are increasingly accurate and useful, and their benefits extend widely across the economy. While much has been accomplished in improving weather forecasts, there remains much room for improvement. Simultaneously, they are developing new technologies and observational netwoeks that can enchance forecaster skill and the value of their services to their users.

FUTURE ENHANCEMENT

The website we created in this project can be futher developed into a mobile application so that it can give timely weather updates. These updates will be received in the form of notification in the user's mobile based on the location they are present in. So the users don't even have to get into that particular application to know the weather and it saves their valuable time.

BIBLIOGRAPHY

[1] Our technology **Time Machine**, has allowed us to enhance data in the Historical Weather Collection: historical weather data is now available for any coordinates and the depth of historical data has been extended to 40 years.
https://openweathermap.org

[2] Background images and internal images.
https:freepik.com

[3] Internal fonts.
https://fonts.google.com/

[4] Icons and also fonts in some other places.
https://fontawesome.com

[5] famous css framework.
http://tailwindcss.com

APPENDIX: (SOURCE CODE)

index.html :

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="shortcut icon" href="favicon.ico" type="image/x-icon" />
    <title>Our Weather</title>
    <script src="https://cdn.tailwindcss.com"></script>
    <script src="https://unpkg.com/lucide@latest/dist/umd/lucide.js"></script>
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link
        href="https://fonts.googleapis.com/css2?family=Akshar:wght@300..700&family=P
oppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;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" />
</head>
<style>
    * {
        font-family: 'poppins';
    }

    body {
    background-image: linear-gradient(rgba(0, 0, 0.5, 0.8) 0, rgba(0, 0.5, 0, 0)
60%, rgba(0, 0, 0.5, 0.8) 100%),url(hb.jpg);
    position: relative;
    background-size: cover;
    background-position: center;
```

```
    }
</style>

<body class="">
    <div
        class="flex justify-around items-center h-20 bg-gradient-to-r from-indigo-
100 from-10% via-sky-200 via-30% to-emerald-100 text-slate-800 drop-shadow-lg">
        <div class='flex justify-center items-center'>
            <img class='w-8 h-8 cursor-pointer' src="umbrella.png"></>
            <h2 class='font-bold cursor-pointer'>Our Weather</h2>
        </div>
        <div>
            <ul class='flex justify-center items-center gap-10'>
                <li class='hover:border-b-2 border-slate-600 cursor-pointer text-
md'><a href="#">Home</a></li>
                <li class='hover:border-b-2 border-slate-600 cursor-pointer'>About
us</li>
                <li class='hover:border-b-2 border-slate-600 cursor-
pointer'>Weather</li>
                <li class='hover:border-b-2 border-slate-600 cursor-pointer'>Contact
us</li>
            </ul>
        </div>
        <div>
            <button class='shadow-lg hover:bg-blue-800 w-24 h-9 bg-blue-500 rounded-
full text-white text-sm'><a
                    href="weather.html">Get Started</a></button>
        </div>
    </div>
    <div class="w-screen h-screen flex flex-col gap-5 items-center justify-center">
        <p class="text-white text-xl m">Start searching for your city to know
weather details</p>
        <a href="weather.html"><button class='shadow-lg hover:bg-slate-300 w-48 h-10
bg-white rounded-full text-black text-sm'>Get
                Started</button></a>
    </div>
    <div>

        <div className='border-t-2 border-slate-400 shadow-lg flex justify-center
text-slate-600 items-center text-sm h-20 bg-gradient-to-r from-indigo-100 from-10%
via-sky-200 via-30% to-emerald-100'>
        Copyright<span className="font-bold text-2xl mx-1">©</span> 2024 CVR
Hospitals Inc.
        </div>
      </div>

</body>
```

```
</html>
```

Weather.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>WEATHER APP</title>
    <link rel="stylesheet" href="weather.css">
</head>
<body>
    <div class="header">
        <h1>OUR WEATHER</h1>
        <div>
            <input type="text" name="" id="input" placeholder="Enter city name">
            <button id="search" onclick="searchByCity()">Search</button></input>
        </div>
    </div>

    <main>
        <div class="weather">
            <h2 id="city">Delhi,IN</h2>
            <div class="temp-box">
                <img src="/weathericon.png" alt="" id="img">
                <p id="temperature">26 °C</p>
            </div>
            <span id="clouds">Broken Clouds</span>
        </div>
        <div class="divider1"></div>

        <div class="forecstH">
            <p class="cast-header">Upcoming forecast</p>
            <div class="templist">

                <div class="next">
                    <div>
                        <p class="time">8:30 PM</p>
                        <p>29 °C / 29 °C</p>
```

```
                    </div>
                    <p class="desc">Light Rain</p>
                </div>

                <div class="next">
                    <div>
                        <p class="time">8:30 PM</p>
                        <p>29 °C / 29 °C</p>
                    </div>
                    <p class="desc">Light Rain</p>
                </div>

                <div class="next">
                    <div>
                        <p class="time">8:30 PM</p>
                        <p>29 °C / 29 °C</p>
                    </div>
                    <p class="desc">Light Rain</p>
                </div>

                <div class="next">
                    <div>
                        <p class="time">8:30 PM</p>
                        <p>29 °C / 29 °C</p>
                    </div>
                    <p class="desc">Light Rain</p>
                </div>

                <div class="next">
                    <div>
                        <p class="time">8:30 PM</p>
                        <p>29 °C / 29 °C</p>
                    </div>
                    <p class="desc">Light Rain</p>
                </div>

            </div>
        </div>
    </main>

    <div class="forecstD">
        <div class="divider2"></div>
        <p class="cast-header"> Next 4 days forecast</p>
        <div class="weekF">

            <div class="dayF">
                <p class="date">Sun Jul 03 2022</p>
                <p>31 °C / 31 °C</p>
```

```
                <p class="desc">Overcast Clouds</p>
            </div>

            <div class="dayF">
                <p class="date">Sun Jul 03 2022</p>
                <p>31 °C / 31 °C</p>
                <p class="desc">Overcast Clouds</p>
            </div>

            <div class="dayF">
                <p class="date">Sun Jul 03 2022</p>
                <p>31 °C / 31 °C</p>
                <p class="desc">Overcast Clouds</p>
            </div>

            <div class="dayF">
                <p class="date">Sun Jul 03 2022</p>
                <p>31 °C / 31 °C</p>
                <p class="desc">Overcast Clouds</p>
            </div>
        </div>
    </div>
    <script src="/weather.js"></script>
</body>
</html>
```

Weather.css:

```
@import
url('https://fonts.googleapis.com/css2?family=Concert+One:wght@300&display=swap');
@import
url('https://fonts.googleapis.com/css2?family=Orbitron:wght@700&display=swap');
@import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,
400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,
900&display=swap');

html{
    height: 100%;
}
h1{
    text-align: center;
}
body{
    background: whitesmoke;
    padding: 1rem 5rem;
    font-family: 'Roboto', sans-serif;
    text-transform: capitalize;
    background: linear-gradient(45deg,rgba(183, 204, 248, 0.717),rgba(7, 43, 127,
0.679)), url('./background.jpg');

    background-repeat: no-repeat;
    background-position: center;
    background-size: cover;
}
.header{
    display: flex;
    justify-content: space-between;
    align-items: center;
}
.header h1{
    color: white;
    text-decoration: underline;
    font-family: 'Orbitron', sans-serif;
}
#input{
    font-size: 18px;
    padding: 5px 10px;
    outline: none;
    border: none;
    border-radius: 15px;
    background: aliceblue;
}
#search{
```

```
    background: none;
    padding: 5px 20px;
    color: aliceblue;
    outline: none;
    background: cadetblue;
    font-size: 17px;
    border-radius: 15px;
    cursor: pointer;
    border: none;
}
.weather{
    text-align: center;
    color: aliceblue;
}
#city{
    font-size: 30px;
}
.weather img{
    width: 120px;
    height: 120px;
    border-radius: 50%;
    background: rgba(240, 248, 255, 0.408);
}
#temperature{
    font-size: 50px;
    margin: 0;
    margin-left: 30px;
    margin-bottom: 10px;
}
.temp-box{
    display: flex;
    align-items: center;
    justify-content: center;
    margin: 30px 0;
}
#clouds{
    font-size: 20px;
    background: rgba(153, 205, 50, 0.778);
    padding: 2px 20px;
    border-radius: 15px;
}
main{
    display: grid;
    grid-column-gap: 25px;
    grid-template-columns: 1fr 5px 1fr;
    align-items: center;
    margin: 0 50px;
    color: white;
```

```
}
.next{
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin: 10px 0;
}
.next p,.next h3{
    margin: 3px 0;
}
.forecstD{
    margin: 20px 50px;
    color: aliceblue;
}
.weekF{
    display: grid;
    grid-template-columns: repeat(4,1fr);
}
.cast-header{
    color: aliceblue;
    background: rgba(254, 189, 132, 0.539);
    width: max-content;
    padding: 5px 15px;
    border-radius: 20px;
    font-size: 18px;
    margin-bottom: 5px;
}
.divider1,.divider2{
    background: rgba(254, 189, 132, 0.539);
    height: 200px;
    border-radius: 5px;
}
.divider2{
    height: 5px;
    width: 30%;
    margin: 0 auto;
}
.time,.date{
    color: rgb(254, 189, 132);
}
.desc{
    color: rgb(196, 255, 77);
}
```

weather.js:

```
const apikey = "c1133d409de6e6edf1c8f48d824a11bd";
window.addEventListener("load", () => {
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition((position) => {
      let lon = position.coords.longitude;
      let lat = position.coords.latitude;
      const url =
        `http://api.openweathermap.org/data/2.5/weather?lat=${lat}&` +
        `lon=${lon}&appid=${apikey}`;

      fetch(url)
        .then((res) => {
          return res.json();
        })
        .then((data) => {
          console.log(data);
          console.log(new Date().getTime());
          var dat = new Date(data.dt);
          console.log(dat.toLocaleString(undefined, "Asia/Kolkata"));
          console.log(new Date().getMinutes());
          weatherReport(data);
        });
    });
  }
});

function searchByCity() {
  var place = document.getElementById("input").value;
  var urlsearch =
    `http://api.openweathermap.org/data/2.5/weather?q=${place}&` +
    `appid=${apikey}`;

  fetch(urlsearch)
    .then((res) => {
      return res.json();
    })
    .then((data) => {
      console.log(data);
      weatherReport(data);
    });
  document.getElementById("input").value = "";
}

function weatherReport(data) {
  var urlcast =
    `http://api.openweathermap.org/data/2.5/forecast?q=${data.name}&` +
```

```
    `appid=${apikey}`;

  fetch(urlcast)
    .then((res) => {
      return res.json();
    })
    .then((forecast) => {
      console.log(forecast.city);
      hourForecast(forecast);
      dayForecast(forecast);

      console.log(data);
      document.getElementById("city").innerText =
        data.name + ", " + data.sys.country;
      console.log(data.name, data.sys.country);

      console.log(Math.floor(data.main.temp - 273));
      document.getElementById("temperature").innerText =
        Math.floor(data.main.temp - 273) + " °C";

      document.getElementById("clouds").innerText = data.weather[0].description;
      console.log(data.weather[0].description);

      let icon1 = data.weather[0].icon;
      let iconurl = "http://api.openweathermap.org/img/w/" + icon1 + ".png";
      document.getElementById("img").src = iconurl;
    });
}

function hourForecast(forecast) {
  document.querySelector(".templist").innerHTML = "";
  for (let i = 0; i < 5; i++) {
    var date = new Date(forecast.list[i].dt * 1000);
    console.log(
      date.toLocaleTimeString(undefined, "Asia/Kolkata").replace(":00", "")
    );

    let hourR = document.createElement("div");
    hourR.setAttribute("class", "next");

    let div = document.createElement("div");
    let time = document.createElement("p");
    time.setAttribute("class", "time");
    time.innerText = date
      .toLocaleTimeString(undefined, "Asia/Kolkata")
      .replace(":00", "");

    let temp = document.createElement("p");
```

```
    temp.innerText =
      Math.floor(forecast.list[i].main.temp_max - 273) +
      " °C" +
      " / " +
      Math.floor(forecast.list[i].main.temp_min - 273) +
      " °C";

    div.appendChild(time);
    div.appendChild(temp);

    let desc = document.createElement("p");
    desc.setAttribute("class", "desc");
    desc.innerText = forecast.list[i].weather[0].description;

    hourR.appendChild(div);
    hourR.appendChild(desc);
    document.querySelector(".templist").appendChild(hourR);
  }
}

function dayForecast(forecast) {
  document.querySelector(".weekF").innerHTML = "";
  for (let i = 8; i < forecast.list.length; i += 8) {
    console.log(forecast.list[i]);
    let div = document.createElement("div");
    div.setAttribute("class", "dayF");

    let day = document.createElement("p");
    day.setAttribute("class", "date");
    day.innerText = new Date(forecast.list[i].dt * 1000).toDateString(
      undefined,
      "Asia/Kolkata"
    );
    div.appendChild(day);

    let temp = document.createElement("p");
    temp.innerText =
      Math.floor(forecast.list[i].main.temp_max - 273) +
      " °C" +
      " / " +
      Math.floor(forecast.list[i].main.temp_min - 273) +
      " °C";
    div.appendChild(temp);

    let description = document.createElement("p");
    description.setAttribute("class", "desc");
    description.innerText = forecast.list[i].weather[0].description;
    div.appendChild(description);
```

```
    document.querySelector(".weekF").appendChild(div);
  }
}
```