WEATHER APP USING API

Project report submitted to the Karpagam Academy of Higher Education in partial fulfillment of the requirements for the award of the degree of Bachelor of Science (Information Technology)

Submitted by

KAVI PRAKASH M

20ITU022

Under the Guidance of

Dr. K.RANJITH SINGH, M.Sc., M.Phil., Ph.D.

Assistant Professor

Department of Computer Science

Karpagam Academy Of Higher Education

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act, 1956)

(Accredited with A+ Grade by NAAC in the Second Cycle)

COIMBATORE – 641 021

APRIL-2023

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed to be University)

(Established Under Section 3 of UGC Act, 1956)

(Accredited with A+ Grade by NAAC in the Second Cycle)

COIMBATORE – 641 021

Bachelor Of Science (Information Technology)

Bonafide Certificate

This is to certify that the project work entitled **"WEATHER APP USING API"** done by **KAVIPRAKASH M, 20ITU022** during the period November 2022 to April 2023 in partial fulfillment of the degree of **B.Sc.Information Technology**, is submitted for the viva-voce held on _____

Project Guide **Head of the Department**

Examiners :

1.

2.

CONTENT

Enable | Enlighten | Enrich
KARPAGAM
ACADEMY OF HIGHER EDUCATION
(Deemed to be University)
(Established Under Section 3 of UGC Act, 1956)

CERTIFICATE

This is to certify that the project entitled **"WEATHER APP USING API"** is the bonafide project work carried out by **Kavi Prakash M** with register number **20ITU022** student of **B.Sc. Information Technology** submitted to the **Karpagam Academy of Higher Education,** Coimbatore, during the period November 2022 to April 2023, in partial fulfillment of the requirements for the award of the Degree of Bachelor of Science Information Technology, is the record of student's own worked carried out by him under my supervision.

Place : COIMBATORE

Date : Signature of the Guide

DECLARATION

I hereby declare that the Project entitled **"WEATHER APP USING API"** submitted for the **B.Sc.Information Technology** Degree is my original work.

Place : COIMBATORE

Date :

Signature of the Student

(Kavi Prakash M)

ACKNOWLEGDEMENT

If words are considered as symbols of approval and tokens of acknowledgement, then the words play the heralding role of expressing my gratitude to all who have helped me directly and indirectly during my project work.

I will be ever graceful and thankful to **Dr.B.VENKATACHALAPATHY, Vice Chancellor**, **Karpagam Academy of Higher Education,** for allowing me to do my project with his moral support.

I take this opportunity to express my sincere gratitude and thanks to our beloved**, Dr.S.RAVI**, **Registrar, Karpagam Academy of Higher Education,** for allowing me to do my project with his moral support.

I will be ever graceful and thankful to **Dr.N.V.BALAJI, Dean, Faculty of Arts, Science, Commerce and Management, Karpagam Academy of Higher Education,** for allowing me to do my project with his moral support.

I will be ever graceful and thankful to **Dr.P.TAMILSELVAN, Associate Professor, Programme Co-Ordinator (IT&CT), Department of Computer Science,** for providing me this opportunity and extending a constant monitoring throughout the course of the project.

I will be ever graceful and thankful to **Dr. K.RANJITH SINGH, M.Sc., M.Phil., Ph.D., Assistant Professor**, **Department of Computer Science,** Internal Guidance for her valuable guidance and constant monitoring throughout the course of the project.

Finally my heart full of thanks to all other Faculty Members, My Parents, and My Beloved Friends for their moral support, without which I would not been able to complete this project.

ABSTRACT

This project aims to develop a web-based weather application that displays real-time weather information for a user's location. The application utilizes the OpenWeather API to retrieve weather data and displays the information in a user-friendly interface. Users can view the current weather conditions, temperature, humidity, wind speed, and other relevant information for their location. The application is built using HTML, CSS, and JavaScript and is responsive, allowing it to be used on various devices. The user interface is designed to be intuitive and easy to use, making it accessible to a wide range of users. Overall, the project provides a practical and useful tool for users to access up-to-date weather information for their location.

CHAPTER 01

1.INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Weather forecasting is the attempt by meteorologists to predict the weather conditions at some future time and the weather conditions that may be expected. The climatic condition parameters are based on the temperature, wind, humidity, rainfall and size of data set. In this Project I will involve in the creating of the simple and efficient weather API using the front-end web development languages like HTML, CSS and JavaScript

1.2 MODULE DESCRIPTION

- **HTML/CSS :**

  These are the foundational languages used for building the structure and styling of the web pages of the application.

- **JavaScript :**

  JavaScript is used to fetch data from the OpenWeather API, parse JSON responses, and dynamically display the weather data on the user interface.

- **OpenWeather API :**

  This is the API that provides the weather data used in the application. You will need to use the API endpoints to retrieve weather data for the user's location.

- **Bootstrap and other CSS framework :**

  CSS frameworks such as Bootstrap provide pre-designed and customizable templates to create responsive and attractive user interfaces.

- **Geolocation API :**

  This is an API that allows you to get the user's location based on their IP address or GPS coordinates, which can be used to automatically retrieve weather data for the user's location.

- **Error handling :**

  You'll need to include error handling mechanisms in case there is a problem with the API, or if there are issues with the user's input.

- **Deployment :**

  You will need to deploy your application to a web server or hosting platform so that users can access it.

1.3 SYSTEM SPECIFICATION

1.3.1 HARDWARE SPECIFICATION

- Processor : Intel core i3 10^{th} gen
- Hard Disk : 250 GB
- RAM : 4 GB

1.3.2 SOFTWARE SPECIFICATION

- Operating System : Windows 7,8,10 & above
- Backend : OpenWeather API
- Web Browser : Chrome , Edge , Firefox
- Software : Visual Studio Code

CHAPTER 2

2.SYSTEM STUDY

2.1 EXISTING SYSTEM

The existing system for a weather app project may vary depending on the specific application being used as a reference. However, some existing systems that provide similar functionality to a weather app may include.

Native weather apps on mobile devices or desktop computers: Many devices come with pre-installed weather apps that display weather information based on the user's location. These apps often provide current weather conditions, forecasts, and severe weather alerts.

2.1.1 DRAWBACKS OF EXISTING SYSTEM

Limited customization : Many existing weather apps and websites have limited options for customizing the user interface or the type of information displayed. This can make it difficult for users to get the information they need quickly and easily.

Advertisements :

Many weather apps and websites rely on advertising to generate revenue, which can be distracting and may slow down the app or website.

Inaccurate Data :

Weather data can be unpredictable, and even the most reliable sources can occasionally provide inaccurate information. This can lead to frustration and confusion for users who are relying on the weather information for planning or safety purposes.

2.2 PROPOSED SYSTEM

The proposed system for a weather app project involves designing and developing a software application that provides users with accurate and up-to-date weather information. The app will be built using HTML, CSS, and JavaScript, along with the OpenWeather API, which provides real-time weather data for any location in the world.

2.2.1 BENEFITS OF THE PROPOSED SYSTEM

- **Customizable user interface :** Your weather app can be designed with a user-friendly interface that is customizable to the user's preferences, including font sizes, colors, and layout.
- **Accurate and up-to-date weather information** : By using the OpenWeather API, your weather app can provide reliable and accurate weather data to users in real-time.
- **Specific weather information :** Your weather app can provide users with specific weather-related information, such as air quality, pollen counts, or UV index, which may not be available in other weather apps.
- **Improved user experience :** By providing a simple and easy-to-use interface, your weather app can improve the overall user experience for users, making it easy for them to find the information they need.
- **Customizable notifications :** Your weather app can provide customizable notifications for severe weather alerts, or other important weather updates, to keep users informed and safe.

2.3 SOFTWARE FEATURES

2.3.1 VISUAL STUDIO CODE

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust. It is based on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine.

CHAPTER 03

3. SYSTEM DESIGN AND DEVELOPMENT

3.1 INPUT SYSTEM

Input design is a crucial aspect of user interface (UI) design that focuses on creating user-friendly and efficient methods for users to input data into a computer system. It involves designing forms, fields, and other input elements to be clear, organized, and easy to use, with the goal of improving the accuracy and efficiency of data entry.

Effective input design takes into consideration the types of data that need to be entered, the context in which the data entry occurs, and the characteristics of the users who will be entering the data. The design should minimize the number of errors made during data entry, and should also help users to enter data quickly and easily, without causing unnecessary frustration or confusion.

3.2 OUTPUT SYSTEM

Output design is a key aspect of user interface (UI) design that focuses on creating clear and effective ways of presenting information to users. It involves designing the way that data is displayed and organized, with the goal of making it easy for users to understand and interpret.

Effective output design takes into consideration the type of information that is being presented, as well as the context in which it is being presented. The design should be clear and organized, with important information emphasized and presented in a way that is easy to read and comprehend. It should also be visually appealing and engaging, with appropriate use of colors, fonts, and other design elements to enhance the user experience.

CHAPTER 04

4. SYSTEM TESTING & IMPLEMENTATION

System testing is a type of software testing that evaluates the complete system or application as a whole, rather than testing individual components or functions in isolation. It is a black-box testing technique that examines the system's behavior and performance in a real-world environment to ensure that it meets the specified requirements and works as intended.

The primary goal of system testing is to identify defects, errors, or inconsistencies in the system's functionality, usability, reliability, security, and performance. This involves testing various aspects of the system, including its interfaces, functionality, compatibility with other systems, database integration, security features, and scalability.

4.1 TESTING METHODOLOGY

- Unit Testing
- Integration Testing
- Validation Testing
- White Box Testing
- Black Box Testing

Unit Testing

This type of testing is focused on testing individual units or components of the software application to ensure that they are working as expected. It is typically conducted by developers during the development phase and involves testing small, isolated parts of the code to ensure that they meet the requirements and specifications.

Integration Testing

This type of testing is focused on testing the integration between different components or modules of the software application to ensure that they are working together as expected. It is typically conducted after unit testing and involves testing the interaction between different parts of the code to ensure that they integrate seamlessly.

Validation Testing

This type of testing is focused on testing the software application against the user's requirements and specifications to ensure that it meets their needs. It is typically conducted after the development phase and involves testing the application as a whole to ensure that it meets the desired functionality and performance criteria.

White Box Testing

This type of testing is focused on testing the internal workings of the software application, including the code, algorithms, and data structures. It is typically conducted by developers during the development phase and involves testing the application at the code level to ensure that it is functioning as expected and that there are no bugs or errors in the code.

Black Box Testing

Black box testing is a type of testing that is focused on testing the software application's functionality and performance from the user's perspective, without knowledge of the internal workings of the application. It is typically conducted after the development phase and involves testing the application's inputs and outputs to ensure that they meet the user's requirements and specifications.

4.2 SYSTEM IMPLEMENTATION

System implementation is the process of putting a new or upgraded software system into operation within an organization. It involves the installation and configuration of the new system, as well as the migration of data from any existing systems to the new system.

The implementation process typically includes several stages, including ;

- **Planning :** This involves defining the project scope, identifying the key stakeholders, and developing a detailed implementation plan.

- **Installation :** This involves installing the new system software, configuring it to meet the organization's needs, and integrating it with other systems if necessary.

- **Data migration :** This involves transferring data from any existing systems to the new system, ensuring that it is accurate and complete.
- **Testing :** This involves testing the new system to ensure that it meets the organization's needs and is free of bugs and other issues.

- **Training :** This involves providing training to end-users on how to use the new system effectively.

- **Rollout :** This involves deploying the new system across the organization, ensuring that it is fully operational and any issues are addressed promptly.

CHAPTER 05

5.CONCLUSION

5.1 CONCLUSION

In conclusion, the weather app project is a valuable software application that can provide users with reliable, accurate, and specific weather information. By utilizing the OpenWeather API, the app can provide real-time weather data for any location in the world, making it a valuable tool for users who rely on weather information for planning or safety purposes.

The proposed system for the weather app project includes several features, such as current weather conditions, weather forecasts, interactive maps, severe weather alerts, customizable notifications, and a user-friendly interface. By addressing the drawbacks of existing weather systems, the proposed system can provide a more valuable and user-friendly experience for users.

5.2 FUTURE ENHANCEMENT

There are several potential future enhancements that could be made to the weather app project. Here are a few possibilities ;

- **Integration with wearable technology :**

  The weather app could be integrated with wearable technology, such as smartwatches or fitness trackers, allowing users to access weather information on-the-go.

- **Integration with other APIs :**

  The weather app could be integrated with other APIs, such as traffic or air quality APIs, to provide users with more comprehensive information about their environment.

- **Enhanced customization options :**

  The app could include enhanced customization options, allowing users to choose from a wider range of themes, layouts, and color schemes to personalize their experience.

6.BIBLIOGRAPHY

6.1 BIBLIOGRAPHY

Here are some online resources that can be helpful for creating a weather app with an API ;

OpenWeatherMap API : OpenWeatherMap is a popular weather API that provides current weather data, forecasts, and historical weather data for various locations around the world. They offer a free plan as well as paid plans with additional features.

Dark Sky API : Dark Sky is another popular weather API that provides hyperlocal weather information, including minute-by-minute forecasts. The API has been recently acquired by Apple and will be shut down in 2022.

AccuWeather API : AccuWeather is a weather API that provides forecasts, historical weather data, and severe weather alerts. They offer a free plan as well as paid plans with additional features.

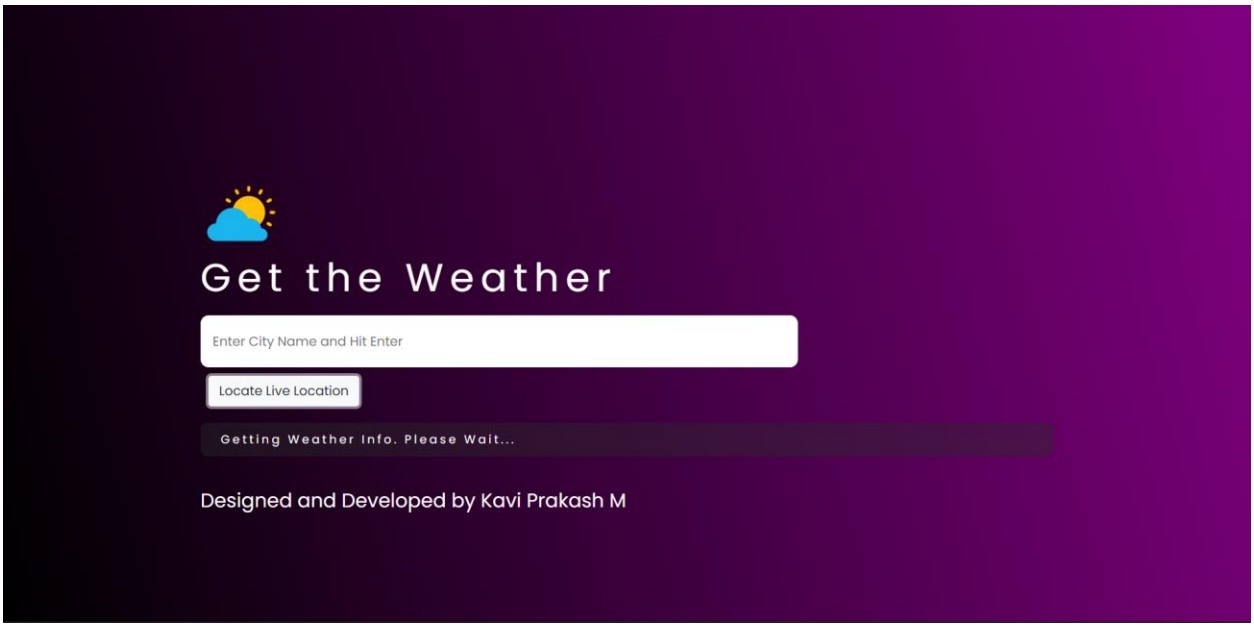
Weatherstack API : Weatherstack is a weather API that provides current weather data and forecasts. They offer a free plan as well as paid plans with additional features.

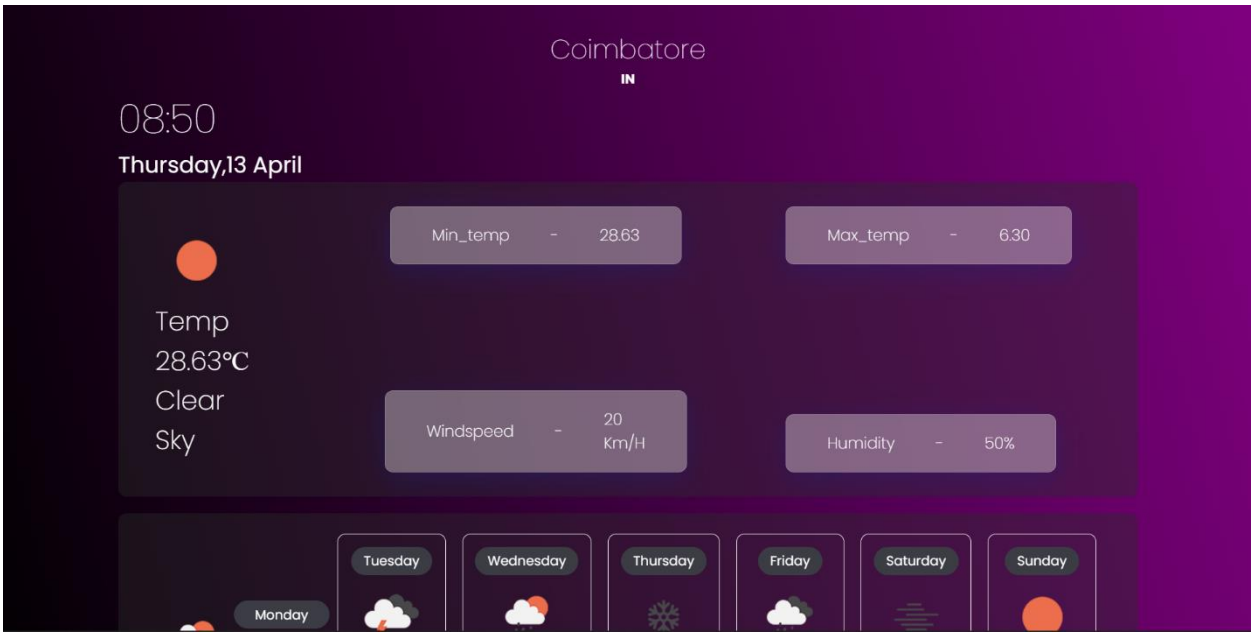
In terms of references or tutorials, there are many resources available online depending on your programming language of choice. Here are a few examples:

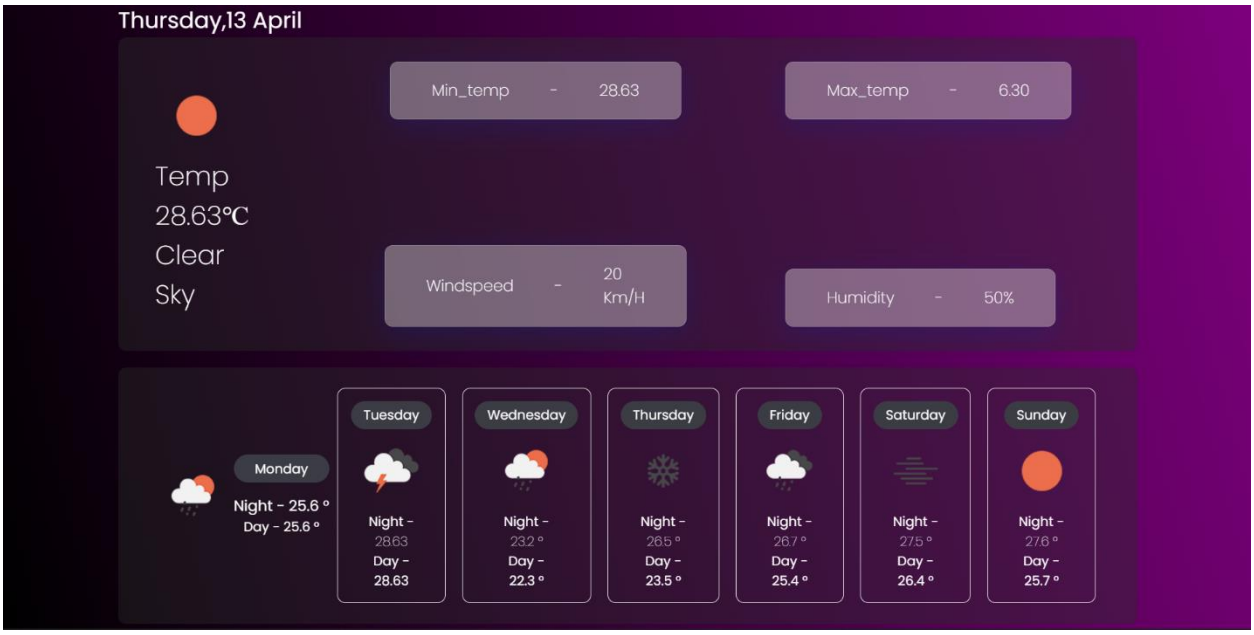
- "How to Build a Weather App with HTML, CSS, and JavaScript" on FreeCodeCamp.org
- "Building a Weather App with OpenWeatherMap API" tutorial on Dev.to
- "How to Build a Weather App in Python" tutorial on Real Python
- "Building a Weather App with React Native and Redux" tutorial on Medium
- I hope these resources help you get started with building your weather app!

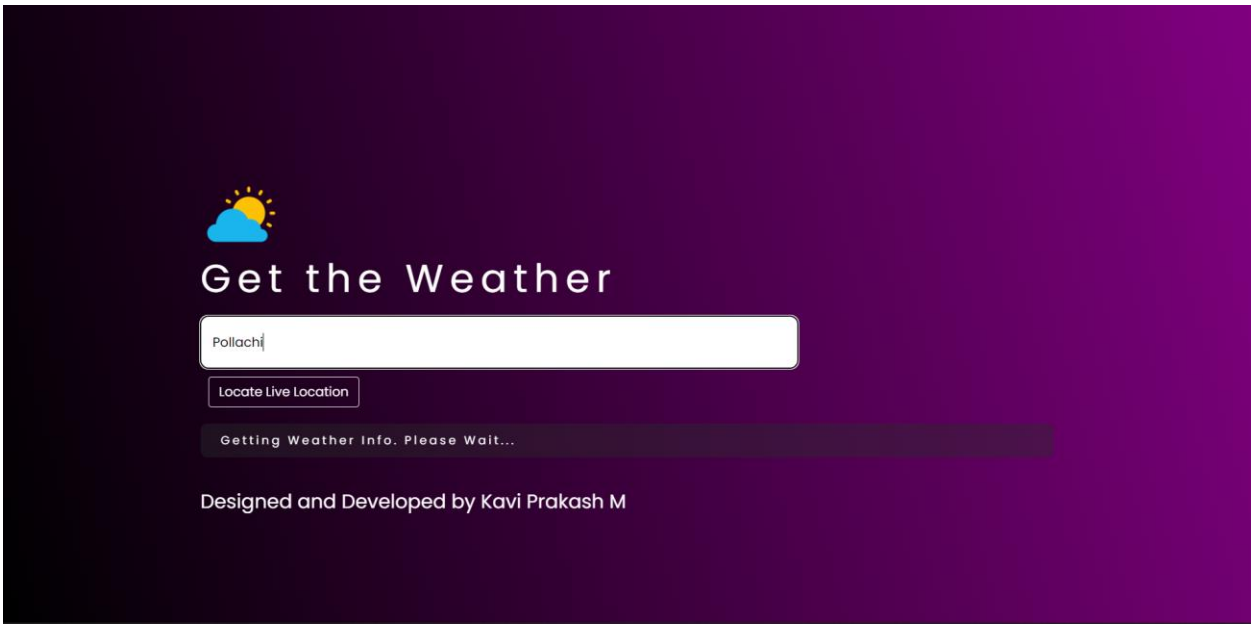
7.APPENDIX

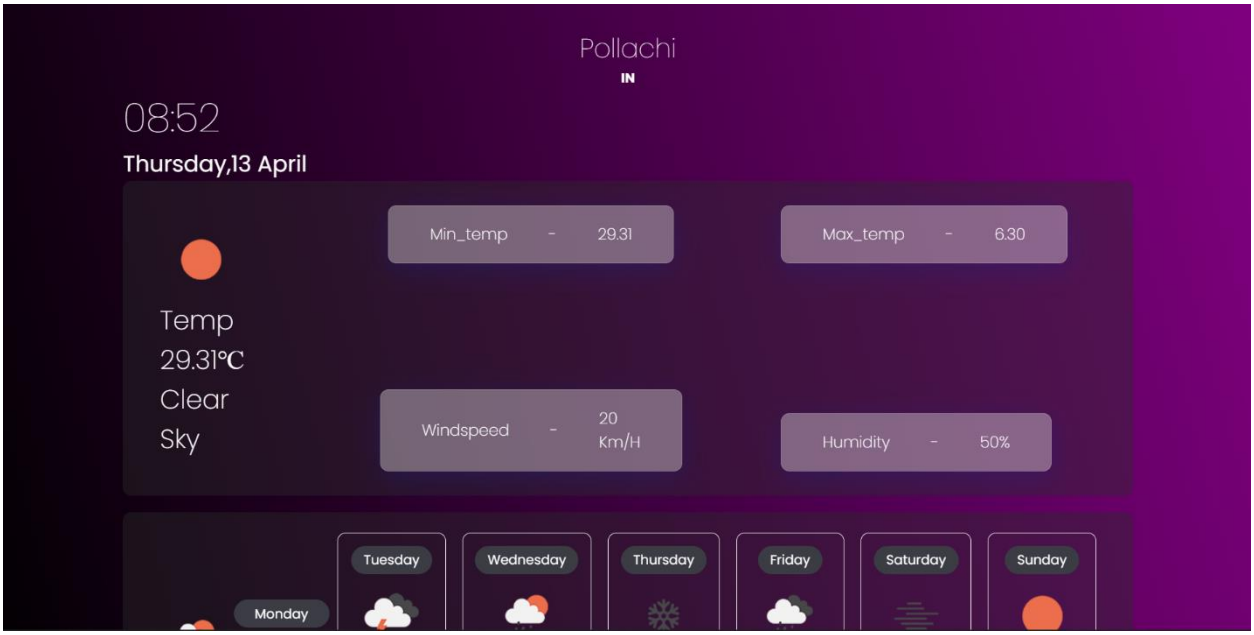
7.1 SCREENSHOTS

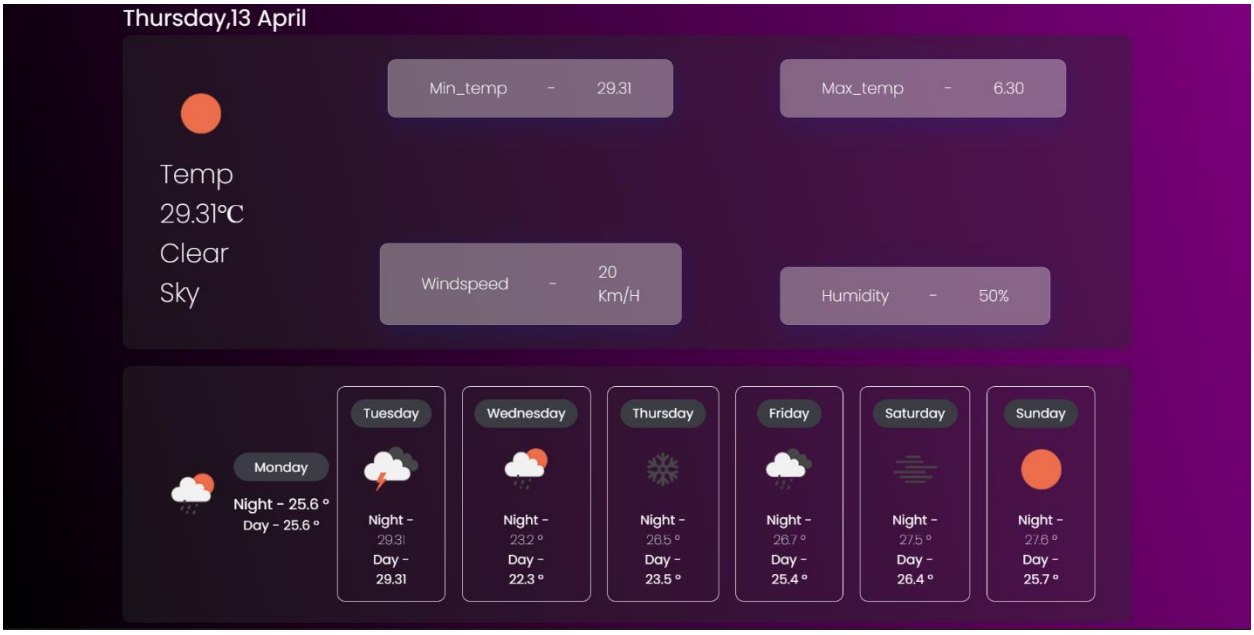





Thursday,13 April
Min_temp - 28.63
Max_temp - 6.30
Temp
28.63℃
Clear
Sky
Windspeed - 20 Km/H
Humidity - 50%
Monday
Night - 25.6 °
Day - 25.6 °
Tuesday
Night - 28.63
Day - 28.63
Wednesday
Night - 23.2 °
Day - 22.3 °
Thursday
Night - 26.5 °
Day - 23.5 °
Friday
Night - 26.7 °
Day - 25.4 °
Saturday
Night - 27.5 °
Day - 26.4 °
Sunday
Night - 27.6 °
Day - 25.7 °


Get the Weather
Pollachi
Locate Live Location
Getting Weather Info. Please Wait...
Designed and Developed by Kavi Prakash M


Pollachi
IN
08:52
Thursday,13 April
Min_temp - 29.31
Max_temp - 6.30
Temp
29.31℃
Clear
Sky
Windspeed - 20 Km/H
Humidity - 50%
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday


Thursday,13 April
Min_temp - 29.31
Max_temp - 6.30
Temp
29.31℃
Clear
Sky
Windspeed - 20 Km/H
Humidity - 50%
Monday
Night - 25.6 °
Day - 25.6 °
Tuesday
Night - 29.31
Day - 29.31
Wednesday
Night - 23.2 °
Day - 22.3 °
Thursday
Night - 26.5 °
Day - 23.5 °
Friday
Night - 26.7 °
Day - 25.4 °
Saturday
Night - 27.5 °
Day - 26.4 °
Sunday
Night - 27.6 °
Day - 25.7 °

7.2 CODING

Weather.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css">
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <link rel="icon" type="image/x-icon"
    href="https://cdn2.iconfinder.com/data/icons/weather-flat-14/64/weather02-512.png">
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
<link rel="stylesheet" href="style.css">
  <title>weather App</title>
</head>
<body>
  <div class="container">
    <div class="search wrapper" id="search">
      <div class="new">
      <img src="https://cdn2.iconfinder.com/data/icons/weather-flat-14/64/weather02-
512.png" id="img1" alt="Image1" height="100px">
      <h1>Get the Weather</h1>
      </div>
      <div class="input-container">
```

```
      <input type="text" placeholder="Enter City Name and Hit Enter" spellcheck="false" /><br>
      <button class="btn btn-outline-light" style="margin: 10px;">Locate Live Location</button>
    </div>
    <div class="error-container">
      <p class="error-text" id="error"></p>
    </div><br>
    <h4 style="color:white;">Designed and Developed by Kavi Prakash M</h4>
  </div>
  <div class="place wrapper">
    <div class="time-zone" id="time-zone">Asia/Koltaka</div>
    <div class="country" id="country">in</div>
  </div>
  <div class="current wrapper">
    <div class="date-container">
      <div class="time" id="time">
        <span id="hours">12</span>:<span id="minutes">20</span>
      </div>
      <div class="date" id="date">
        <span id="day">monday</span>,<span id="date-today">15</span>
        <span id="month">aug</span>
      </div>
      <div class="others" id="current-weather">
        <div class="weather-condition-container">
```

```
  <div class="weather-condition-wrapper">
    <div class="weather-condition">
      <div class="icon"></div>
      <div class="condition">temp <span></span>&#8451;</div>
    </div>
    <span class="description">cloudy</span>
  </div>
</div>
<div class="other-details">
  <div class="grid-left">
    <div class="spaced-content weather-deatils">
      <div class="box"><div>min_temp</div>-
      <div id="min-temp">30 <span>&#8451;</span></div>
    </div></div>
    <div class="spaced-content weather-deatils">
      <div class="box">
      <div>windspeed</div>-
      <div id="wind-speed">20 <span>Km/h</span></div>
    </div></div>
  </div>
  <div class="grid-right">
    <div class="spaced-content weather-deatils">
      <div class="box">
      <div>max_temp</div>-
```

```
            <div id="max-temp">6.30 <span>&#8451;</span></div>
          </div></div>
          <div class="spaced-content weather-deatils">
            <div class="box">
            <div>humidity</div>-
            <div id="humidity">50<span>%</span></div>
          </div></div>
        </div>
      </div>
      <div class="future-forecast">
        <div class="today" id="current-temp">
          <img src="http://openweathermap.org/img/wn/10d@2x.png" alt="weather-icon"
class="w-icon">
          <div class="other">
            <div class="day">Monday</div>
            <div class="temp">Night - 25.6 &#176;</div>
            <div class="temp1">Day - 25.6 &#176;</div>
          </div>
      </div>
      <div class="weather-forecast" id="weather-forecast">
        <div class="weather-forecast-item">
          <div class="day">Tuesday</div>
          <img src="http://openweathermap.org/img/wn/11d@2x.png" alt="weather-icon"
class="w-icon">
```

```
          Night -<div class="temp"> 25.6 &#176;</div>

          Day -<div class="temp1"> 25.6 &#176;</div>

        </div>

        <div class="weather-forecast-item">

          <div class="day">Wednesday</div>

          <img src="http://openweathermap.org/img/wn/10d@2x.png" alt="weather-icon"
class="w-icon">

          Night -<div class="temp"> 25.6 &#176;</div>

          Day -<div class="temp1"> 25.6 &#176;</div>

        </div>

        <div class="weather-forecast-item">

          <div class="day">Thursday</div>

          <img src="http://openweathermap.org/img/wn/13d@2x.png" alt="weather-icon"
class="w-icon">

          Night -<div class="temp"> 25.6 &#176;</div>

          Day -<div class="temp1"> 25.6 &#176;</div>

        </div>

        <div class="weather-forecast-item">

          <div class="day">Friday</div>

          <img src="http://openweathermap.org/img/wn/09d@2x.png" alt="weather-icon"
class="w-icon">

          Night -<div class="temp"> 25.6 &#176;</div>

          Day -<div class="temp1"> 25.6 &#176;</div>

        </div>

        <div class="weather-forecast-item">
```

```
            <div class="day">Saturday</div>
            <img src="http://openweathermap.org/img/wn/50d@2x.png" alt="weather-icon" class="w-icon">
            Night -<div class="temp"> 25.6 &#176;</div>
            Day -<div class="temp1"> 25.6 &#176;</div>
          </div>
          <div class="weather-forecast-item">
            <div class="day">Sunday</div>
            <img src="http://openweathermap.org/img/wn/01d@2x.png" alt="weather-icon" class="w-icon">
            Night -<div class="temp"> 25.6 &#176;</div>
            Day -<div class="temp1"> 25.6 &#176;</div>
          </div>
        </div>
      </div>
    </div>
  </div>
  <script src="script.js"></script>
</body></html>
```

Style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;400;700&display
=swap');

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    background: linear-gradient(75deg,black, purple);
    font-family: 'Poppins', sans-serif;
    min-height: 100vh;
}

.container {
    display: flex;
    flex-direction: column;
    padding: 30px;
    align-items: center;
}

.container h1 {
    color: white;
    letter-spacing: 8px;
}
```

```
.container.active .search {
    display: none; }
.search .input-container input {
    width: 100%;
    border: none;
    font-size: 1rem;
    border-radius: 10px;
}
.search .error-container .error-text {
    font-size: .8rem;
    font-weight: 400;
    letter-spacing: .2rem;
    text-transform: capitalize;
}
.error-text.error {
    color: white;
    background: rgba(255, 0, 0, 0.6);
    padding: 10px 25px;
    border-radius: 8px;
}
.error-text.wait {
    color: white;
    padding: 10px 25px;
    background: rgba(42, 43, 42, 0.418);
```

```
    border-radius: 8px;
}

.container .current {
    justify-content: space-between;
}
.date-container .time {
    font-size: 3rem;
    font-weight: 100;
}
.date-container .time #am-pm {
    font-size: 2.5rem;
    text-transform: uppercase;
}
.date-container .date {
    font-size: 1.7rem;
}
.place {
    text-align: center;
    display: none;
    flex-direction: column;
    color: white;
}
.container.active .place {
```

```
    display: flex;
}
.place .time-zone {
    font-size: 2rem;
    font-weight: 100;
}
.place .country {
    font-size: 1rem;
    font-weight: 700;
}
.current {
    display: none;
    color: white;
}
.container.active .current {
    display: flex;
}
.current .others {
    font-size: 1rem;
    font-weight: 200;
    text-transform: capitalize;
    background-color: rgba(42, 43, 42, 0.418);
    border-radius: 10px;
}
```

```
.current .others .weather-condition-container {
    display: flex;
    align-items: center;
    justify-content: center;
    padding: 20px;
    font-size: 1.5rem;
    margin: 15px;
}
.current .others .other-details {
    display: flex;
    flex-direction: column;
    padding: 20px;
}
.others .other-details .grid-left,
.grid-right {
    display: flex;
    flex-direction: column;
    margin: 10px;
}
.others .other-details .grid-left .spaced-content {
    display: flex;
    justify-content: space-between;
}
.others .other-details .grid-right .spaced-content {
```

```
    display: flex;
    justify-content: space-between;
}
.other-details .grid-left .spaced-content div {
    font-size: 1rem;
}
.other-details .grid-right .spaced-content div {
    font-size: 1rem;
}
@media only screen and (min-width: 769px) {
    h1 {
        font-size: 3rem;
    }
    .container .search {
        margin-top: 15%;
        padding: 15px 15px;
        width: 100%;
        align-items: center;
        justify-content: center;
    }
    .search .input-container {
        font-size: 1.2rem;
        padding: 10px 0px;
    }
```

```
.search .input-container input {
    width: 70%;
    padding: 20px 15px;
}
.search .error-container .error-text {
    font-size: .9rem;
}
.place {
    display: none;
}
.current {
    display: none;
}
.container.active .current {
    display: flex;
}
.current .others {
    display: flex;
    flex-direction: row;
}
.current .others .weather-condition-container {
    display: flex;
    align-items: center;
    justify-content: center;
```

```
    padding: 30px;
    font-size: 2rem;
    margin: 15px;
}
.current .others .other-details {
    display: flex;
    flex-direction: row;
    padding: 20px;
    margin-left: 40px;
}
.others .other-details .grid-left,
.grid-right {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
}
.others .other-details .grid-left .spaced-content {
    display: flex;
    justify-content: space-around;
}
.others .other-details .grid-right .spaced-content {
    display: flex;
    justify-content: space-between;
}
```

```
    .other-details .grid-left .spaced-content div {
        margin: 0 50px;
        font-size: 1.2rem;
    }
    .other-details .grid-right .spaced-content div {
        margin: 0 50px;
        font-size: 1.2rem;
    }
}
.future-forecast {
    margin-top: 20px;
    background-color: rgba(42, 43, 42, 0.418);
    border-radius: 10px;
    padding: 25px;
    display: flex;
    color: white;
    width: 100%;
    align-items: center;
    justify-content: center;
}
.future-forecast .today {
    display: flex;
    align-items: center;
    text-align: center;
```

```
}
.future-forecast .today .day {
    padding: 5px 15px;
    background-color: #3c3c44;
    text-align: center;
    border-radius: 50px;
}
.future-forecast .today .temp {
    font-size: 18px;
    padding-top: 15px;
}
.future-forecast .weather-forecast {
    display: flex;
}
.weather-forecast .weather-forecast-item {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    margin: 0px 10px;
    border: 1px solid #eee;
    border-radius: 10px;
    padding: 15px;
}
```

```
.weather-forecast .weather-forecast-item .day {
    padding: 5px 15px;
    background: #3c3c44;
    border-radius: 50px;
    text-align: center;
}
.weather-forecast .weather-forecast-item .temp {
    font-weight: 100;
}
@media only screen and (max-width:730px) {
    .future-forecast {
        justify-content: start;
        align-items: none;
        overflow-y: scroll;
    }
}
.box
{
    background: rgba( 255, 255, 255, 0.35 );
    box-shadow: 0 8px 32px 0 rgba( 31, 38, 135, 0.37 );
    backdrop-filter: blur( 13px );
    -webkit-backdrop-filter: blur( 13px );
    border-radius: 10px;
    border: 1px solid rgba( 255, 255, 255, 0.18 );
```

```
    display: flex;
    padding: 20px 0px;
    color: white;
    align-items: center;
    justify-content: center;
}
```

Script.js

```
const API = "5f8bd5df4a8f994029aefd3fa0b2efe6",
  currentWeatherItemsEl = document.getElementById("current-weather"),
  timeZoneEL = document.getElementById("time-zone"),
  countryEL = document.getElementById("country"),
  weeklyForcastEl = document.getElementById("weekly-forcast"),
  currentTempEl = document.getElementById("current-temp"),
  inputField = document.querySelector("input"),
  loactionBtn = document.querySelector("button"),
  textError = document.querySelector(".error-text"),
  errContainer = document.querySelector(".error-container"),
  container = document.querySelector(".container"),
  weatherIcon = document.querySelector(".icon"),
  days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday",],
  months = ["January", "February", "March", "April", "May", "June", "July", "August",
"September", "October", "November", "December",];
let api;
inputField.addEventListener("keyup", (e) => {
  if (e.key == "Enter" && inputField.value != "") {
    requestApi(inputField.value);
  }
});
function requestApi(city) {
```

```
  console.log(city);
  api =
`https://api.openweathermap.org/data/2.5/weather?q=${city}&units=metric&cnt=7&appi
d=${API}`;
  fetchData();
}
loactionBtn.addEventListener("click", () => {
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(onSuccess, onError);
  } else {
    alert("Your browser does not support geoloaction api");
  }
});
function onSuccess(position) {
  const { latitude, longitude } = position.coords;
  api =
`https://api.openweathermap.org/data/2.5/weather?lat=${latitude}&lon=${longitude}&ex
clude=hourly,daily&units=metric&appid=${API}`;
  fetchData();
}
function onError(error) {
  textError.innerText = error.message;
  textError.classList.add("error");
}
function fetchData() {
```

```
  textError.innerText = "Getting weather info. Please wait...";
  textError.classList.add("wait");
  fetch(api)
    .then((response) => response.json())
    .then((result) => weatherDetails(result));
}
function weatherDetails(info) {
  textError.classList.replace("wait", "error");
  if (info.cod == "404") {
    textError.innerText = `"${inputField.value}" isn't a valid city name`;
  } else {
    const city = info.name,
      timezone = info.timezone,
      { country } = info.sys,
      { description, icon } = info.weather[0],
      { humidity, temp, temp_max, temp_min } = info.main, speed = info.wind;
      console.log(city)
    container.querySelector(".place .time-zone").innerText = city;
    container.querySelector(".place .country").innerText = country;
    container.querySelector("#min-temp").innerText = temp_min;
    container.querySelector("#max_temp"), (innerText = temp_max);
    container.querySelector("#humidity"), (innerText = humidity);
    container.querySelector("#wind-speed"), (innerText = speed);
```

```
    container.querySelector(".weather-condition-container .description").innerText =
description;

    container.querySelector(".weather-condition-container .condition span").innerText =
temp;

    container.querySelector(".weather-forecast .weather-forecast-item .temp").innerText =
temp;

    container.querySelector(".weather-forecast-item .temp1").innerText = temp_min;


    let iconImg = document.createElement("img");

    iconImg.src = `http://openweathermap.org/img/wn/${icon}@2x.png`;

    iconImg.alt = "weather-icon";

    weatherIcon.appendChild(iconImg);

    textError.classList.remove("wait", "error");

    container.classList.add("active");

    console.log(info);

    console.log(timezone);

    function dispalyTime() {

      let utcHours = timezone / 360 / 10;

      const d = new Date();

      const localTime = d.getTime();

      const localOffset = d.getTimezoneOffset() * 60000;

      const utc = localTime + localOffset;

      const getTime = utc + 3600000 * utcHours;

      const getTimeNow = new Date(getTime);

      const hrs = getTimeNow.getHours();
```

```
    const min = getTimeNow.getMinutes();
    const month = getTimeNow.getMonth();
    const date = getTimeNow.getDate();
    const day = getTimeNow.getDay();
    document.getElementById("day").innerHTML = days[day];
    document.getElementById("month").innerHTML = months[month];
    document.getElementById("date-today").innerHTML = date;
    if (min < 10) {
      document.getElementById("minutes").innerHTML = "0" + min;
    } else {
      document.getElementById("minutes").innerHTML = min;
    }
    if (hrs < 10) {
      document.getElementById("hours").innerHTML = "0" + hrs;
    } else {
      document.getElementById("hours").innerHTML = hrs;
    }
   }
   setInterval(dispalyTime, 100);
  }
}
```