PROJECT REPORT FORMAT

INTRODUCTION

⇒ Over View :-

Weather App is an one step Solution for Staying up-to-date with real-time weather forecasts.

This project is an existing endever in "front-end Development" aimed to providing applications. Our mission is to deliver an engaging user experience by presenting weather data in a visually appealing and informative.

⇒ purpose :-

Weather plays a Significant role in our daily lives, influencing our purpose activities, clothing choices & Over all well-being. people Conitantly seek accurate weather information to plan their Schedules accordinly, while many weather application exist. Weather app Stands out by priortiting user experience and Simplicity.

The purpose of a weather app project is to Create a Software application that provide users with real-time weather information and forecasts for a Specific location (or) a Multiple location.

LITERATURE SURVEY

⇒ Existing problem:

Real-Time Weather Data:

The app should be able to fetch and display current weather conditions, including temprature, humidity, wind speed & visibility.

Weather forecasts:

Providing accurate weather forecasts for the next few days is crucial, as it help users plan ahead for events, travel or outdoor activities.

Location Based Services:-

The app should be able to determine the users location & allow them to input a specific location for weather information.

User-friendly Interface:-

The app should have an intutive and visually appealing interface, making it easy for users to understand and navigate.

Customization:-

users may want to customize the app to display weather units in their preferred format

⇒ Proposed Solution:-

Weather Alerts:

The app may include a feature to send weather alerts and notifications to users for severe weather conditions.

Maps and Radar:-

Including weather maps and radar data can help users visualize weather patterns and track storms.

Intergration with API's:

The app may utilize third-party weather API's to access accurate and up-to-date weather data.

Cross-platform Compatability:-

To reacher a boarder audience, the app should be compatiable with different operating systems such as andriod, ios, web browsers.

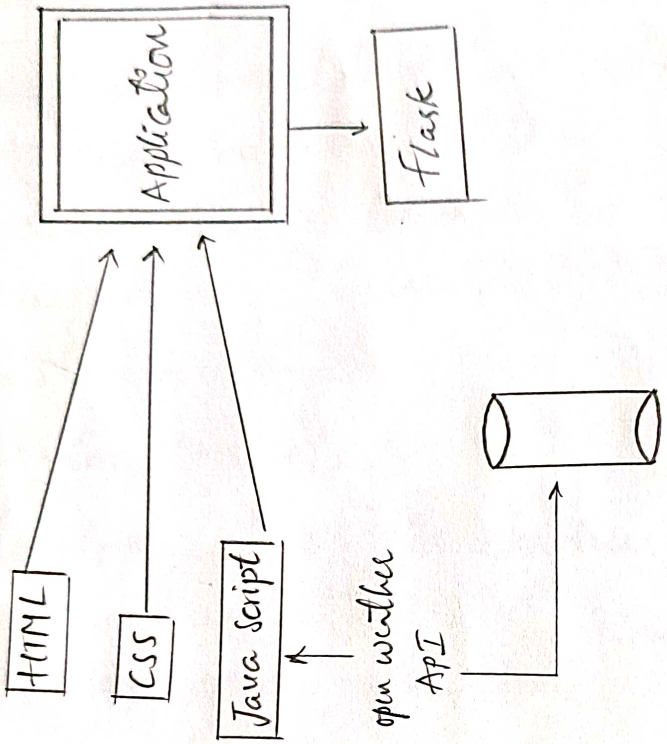
offline Access:-

Although real-time data is cscutial, the app might consider providing basic weather information even when the device is offline.

Overall, the primary purpose of weather app project is to offer users a convenient and reliable tool to access weather information.

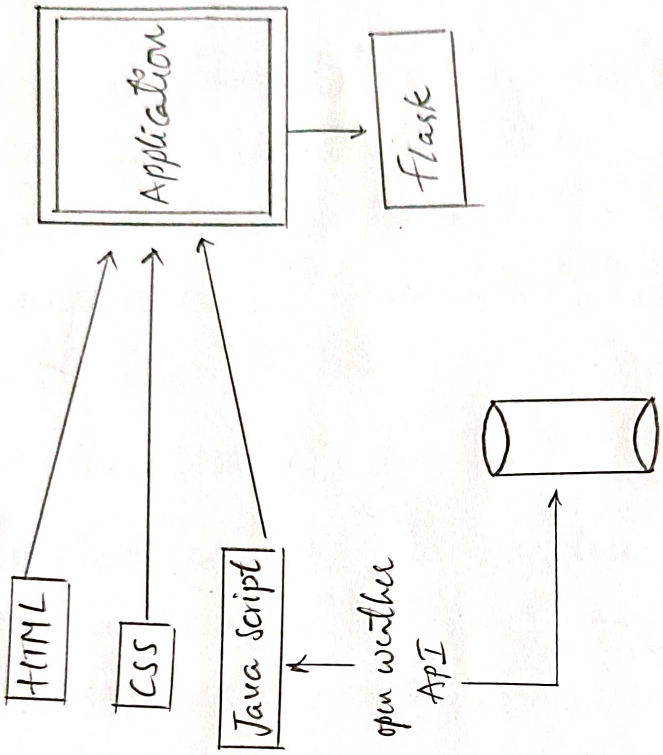
THEORITICAL ANALYSIS

⇒ Block Diagram:



THEORITICAL ANALYSIS

⇒ Block Diagram:



⇒ Hardware / software Designing :-

Hardware and software requirements of the project

Accessing a database :-

* The System should allow administrator to add historical weather data.
* The System should be able to recognize patterns in CSI temprature, humidity, and wind with use of historical data.

Software Constraints :-

* The development of the System will be Constrai-ned by the availability of required Software such as webservers, dataset

Hardware Requirements :-

* The System requires a database in order to store persistent data.

Weather.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Weather App</title>

  <!-- Stylesheet -->
  <link rel="stylesheet" href="style.css" />
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
  <link  rel="icon" href="https://cdn-icons-png.flaticon.com/128/2272/2272221.png" type="png">
 </head>
 <body>
<div class="video-container">
  <video autoplay muted loop>
   <source src="https://v1.pinimg.com/videos/mc/720p/03/fa/2a/03fa2ac3414e1052acec1687c11e5bfe.mp4" />
  </video>
 </div>
 <div class="wrapper">
  <div class="container">
   <div class="search-container">
    <input
     type="text"
     placeholder="please enter a location"
     id="city"
     value="visakhapatnam"
    />
   <button id="search-btn">search</button>
  </div>
  <div id="result"></div>
  <div id="location-details">
```

```
      <button id="get-location"><img src="https://img.icons8.com/?size=512&id=85049&format=png"
alt="location logo"></button>
    </div>
    </div>
  </div>
  <!-- Script -->
  <script src="script1.js"></script>
  <script src="script.js"></script>
  <script>
   function autocomplete(inp, arr) {
    /*the autocomplete function takes two arguments,
    the text field element and an array of possible autocompleted values:*/
    var currentFocus;
    /*execute a function when someone writes in the text field:*/
    inp.addEventListener("input", function(e) {
      var a, b, i, val = this.value;
      /*close any already open lists of autocompleted values*/
      closeAllLists();
      if (!val) { return false;}
      currentFocus = -1;
      /*create a DIV element that will contain the items (values):*/
      a = document.createElement("DIV");
      a.setAttribute("id", this.id + "autocomplete-list");
      a.setAttribute("class", "autocomplete-items");
      /*append the DIV element as a child of the autocomplete container:*/
      this.parentNode.appendChild(a);
      /*for each item in the array...*/
      for (i = 0; i < arr.length; i++) {
       /*check if the item starts with the same letters as the text field value:*/
       if (arr[i].substr(0, val.length).toUpperCase() == val.toUpperCase()) {
        /*create a DIV element for each matching element:*/
        b = document.createElement("DIV");
        /*make the matching letters bold:*/
```

```
      <button id="get-location"><img src="https://img.icons8.com/?size=512&id=85049&format=png"
alt="location logo"></button>
    </div>
    </div>
  </div>
  <!-- Script -->
  <script src="script1.js"></script>
  <script src="script.js"></script>
  <script>
   function autocomplete(inp, arr) {
    /*the autocomplete function takes two arguments,
    the text field element and an array of possible autocompleted values:*/
    var currentFocus;
    /*execute a function when someone writes in the text field:*/
    inp.addEventListener("input", function(e) {
      var a, b, i, val = this.value;
      /*close any already open lists of autocompleted values*/
      closeAllLists();
      if (!val) { return false;}
      currentFocus = -1;
      /*create a DIV element that will contain the items (values):*/
      a = document.createElement("DIV");
      a.setAttribute("id", this.id + "autocomplete-list");
      a.setAttribute("class", "autocomplete-items");
      /*append the DIV element as a child of the autocomplete container:*/
      this.parentNode.appendChild(a);
      /*for each item in the array...*/
      for (i = 0; i < arr.length; i++) {
       /*check if the item starts with the same letters as the text field value:*/
       if (arr[i].substr(0, val.length).toUpperCase() == val.toUpperCase()) {
        /*create a DIV element for each matching element:*/
        b = document.createElement("DIV");
        /*make the matching letters bold:*/
```

```
      b.innerHTML = "<strong>" + arr[i].substr(0, val.length) + "</strong>";
      b.innerHTML += arr[i].substr(val.length);
      /*insert a input field that will hold the current array item's value:*/
      b.innerHTML += "<input type='hidden' value='" + arr[i] + "'>";
      /*execute a function when someone clicks on the item value (DIV element):*/
      b.addEventListener("click", function(e) {
        /*insert the value for the autocomplete text field:*/
        inp.value = this.getElementsByTagName("input")[0].value;
        /*close the list of autocompleted values,
        (or any other open lists of autocompleted values:*/
        closeAllLists();
      });
      a.appendChild(b);
    }
  }
});
/*execute a function presses a key on the keyboard:*/
inp.addEventListener("keydown", function(e) {
  var x = document.getElementById(this.id + "autocomplete-list");
  if (x) x = x.getElementsByTagName("div");
  if (e.keyCode == 40) {
    /*If the arrow DOWN key is pressed
increase the currentFocus variable:*/
    currentFocus++;
    /*and and make the current item more visible:*/
    addActive(x);
  } else if (e.keyCode == 38) { //up
    /*If the arrow UP key is pressed,
    decrease the currentFocus variable:*/
    currentFocus--;
    /*and and make the current item more visible:*/
    addActive(x);
  } else if (e.keyCode == 13) {
```

```
    /*If the ENTER key is pressed, prevent the form from being submitted,*/
    e.preventDefault();
    if (currentFocus > -1) {
     /*and simulate a click on the "active" item:*/
     if (x) x[currentFocus].click();
    }
   }
});
function addActive(x) {
 /*a function to classify an item as "active":*/
 if (!x) return false;
 /*start by removing the "active" class on all items:*/
 removeActive(x);
 if (currentFocus >= x.length) currentFocus = 0;
 if (currentFocus < 0) currentFocus = (x.length - 1);
 /*add class "autocomplete-active":*/
 x[currentFocus].classList.add("autocomplete-active");
}
function removeActive(x) {
 /*a function to remove the "active" class from all autocomplete items:*/
 for (var i = 0; i < x.length; i++) {
  x[i].classList.remove("autocomplete-active");
 }
}
function closeAllLists(elmnt) {
 /*close all autocomplete lists in the document,
 except the one passed as an argument:*/
 var x = document.getElementsByClassName("autocomplete-items");
 for (var i = 0; i < x.length; i++) {
  if (elmnt != x[i] && elmnt != inp) {
   x[i].parentNode.removeChild(x[i]);
  }
 }
```

```
}
/*execute a function when someone clicks in the document:*/
document.addEventListener("click", function (e) {
  closeAllLists(e.target);
});
}

/*An array containing all the country names in the world:*/
var countries =
["adoni","Amaravati","Anantapur","Chandragiri","Chittoor","Dowlaiswaram","Eluru","Guntur","Kadapa","Kakin
ada","Kurnool","Machilipatnam","Nagarjunakonda","Rajahmundry","Srikakulam","Tirupati","Vijayawada","Visa
khapatnam","Vizianagaram","Yemmiganur","Arunachal
Pradesh","Itanagar","Assam","Dhuburi","Dibrugarh","Dispur","Guwahati","Jorhat","Nagaon","Sivasagar","silch
ar","Tezpur","Tinsukia","Bihar","Ara","Barauni","Begusarai","Bettiah","Bhagalpur","Bihar Sharif","Bodh
Gaya","Buxar","Chapra","Darbhanga","Dehri","Dinapur
Nizamat","Gaya","Hajipur","Jamalpur","Katihar","Madhubani","Motihari","Munger","Muzaffarpur","Patna","P
urnia","Pusa","Saharsa","Samastipur","Sasaram","Sita","marhi","Siwan","Chandigarh (union
territory)","Chandigarh","Chhattisgarh","Ambikapur","Bhilai","Bilaspur","Dhamtari","Durg","Jagdalpur","Raipu
r","Rajnandgaon","Dadra","Daman","Diu","Silvassa","New
Delhi","Goa","Madgaon","Panaji","Gujarat","Ahmadabad","Amreli","Bharuch","Bhavnagar","Bhuj","Dwarka","
Gandhinagar","Godhra","Jamnagar","Junagadh","Kandla","Khambhat","Kheda","Mahesana","Morbi","Nadiad",
"Navsari","Okha","Palanpur","Patan","Porbandar","Rajkot","Surat","Surendranagar","Valsad","Veraval","Harya
na","Ambala","Bhiwani","Chandigarh","Faridabad","Firozpur
Jhirka","Gurugram","Hansi","Hisar","Jind","Kaithal","Karnal","Kurukshetra","Panipat","Pehowa","Rewari","Roh
tak","Sirsa","Sonipat","Himachal
Pradesh","Bilaspur","Chamba","Dalhousie","Dharmshala","Hamirpur","Kangra","Kullu","Mandi","Nahan","Shi
mla","Una","Anantnag","Baramula","Doda","Gulmarg","Jammu","Kathua","Punch","Rajouri","Srinagar","Udha
mpur","Jharkhand","Bokaro","Chaibasa","Deoghar","Dhanbad","Dumka","Giridih","Hazaribag","Jamshedpur","
Jharia","Rajmahal","Ranchi","Saraikela","Karnataka","Badami","Ballari","Bengaluru","Belagavi","Bhadravati","B
idar","Chikkamagaluru","Chitradurga","Davangere","Halebid","Hassan","Hubballi-
Dharwad","Kalaburagi","Kolar","Madikeri","Mandya","Mangaluru","Mysuru","Raichur","Shivamogga","Shravan
abelagola","Shrirangapattana","Tumakuru","Vijayapura","Kerala","Alappuzha","Vatakara","Idukki","Kannur","K
ochi","Kollam","Kottayam","Kozhikode","Mattancheri","Palakkad","Thalassery","Thiruvananthapuram","Thriss
ur","Ladakh","Kargil","Leh","Madhya
Pradesh","Balaghat","Barwani","Betul","Bharhut","Bhind","Bhojpur","Bhopal","Burhanpur","Chhatarpur","Chh
indwara","Damoh","Datia","Dewas","Dhar","Dr. Ambedkar
Nagar","Guna","Gwalior","Hoshangabad","Indore","Itarsi","Jabalpur","Jhabua","Khajuraho","Khandwa","Kharg
one","Maheshwar","Mandla","Mandsaur","Morena","Murwara","Narsimhapur","Narsinghgarh","Narwar","Ne
emuch","Nowgong","Orchha","Panna","Raisen","Rajgarh","Ratlam","Rewa","Sagar","Sarangpur","Satna","Seho
re","Seoni","Shahdol","Shajapur","Sheopur","Shivpuri","Ujjain","Vidisha","Maharashtra","Ahmadnagar","Akola
","Amravati","Aurangabad","Bhandara","Bhusawal","Bid","Buldhana","Chandrapur","Daulatabad","Dhule","Jal
gaon","Kalyan","Karli","Kolhapur","Mahabaleshwar","Malegaon","Matheran","Mumbai","Nagpur","Nanded","
Nashik","Osmanabad","Pandharpur","Parbhani","Pune","Ratnagiri","Sangli","Satara","Sevagram","Solapur","T
hane","Ulhasnagar","Vasai-
Virar","Wardha","Yavatmal","Manipur","Imphal","Meghalaya","Cherrapunji","Shillong","Mizoram","Aizawl","Lu
nglei","Nagaland","Kohima","Mon","Phek","Wokha","Zunheboto","Odisha","Balangir","Baleshwar","Baripada",
"Bhubaneshwar","Brahmapur","Cuttack","Dhenkanal","Kendujhar","Konark","Koraput","Paradip","Phulabani",
"Puri","Sambalpur","Udayagiri","Puducherry","Karaikal","Mahe","Puducherry","Yanam","Punjab","Amritsar","
```

```
America","Uruguay","Uzbekistan","Vanuatu","Vatican City","Venezuela","Vietnam","Virgin Islands
(US)","Yemen","delhi","Zambia","Zimbabwe"];


    /*initiate the autocomplete function on the "myInput" element, and pass along the countries array as
possible autocomplete values:*/
    autocomplete(document.getElementById("city"), countries);
    </script>
    <footer>


      <div class="footer">
<p>Copyright &amp;copy; 2023 by prasanth kumar </p>
<p>Contact me by</p>
<a href="https://www.facebook.com/prasanth.sheek?mibextid=2JQ9oc" class="fa fa-facebook"
target="_blank">fb</a><br>
<a href="https://instagram.com/__p_r_a_s_a_n_t_h?igshid=MzRlODBiNWFlZA==" class="fa fa-instagram"
target="_blank">Insta</a>
</div>


</footer>
 </body>
</html>
```

Style.css

```
* {
   padding: 0;
   margin: 0;
   box-sizing: border-box;
   font-family: "papyrus", fantasy;
 }
.video-container {
  height: 100%;
  width: 100%;
  overflow: hidden;
  position: relative;
```

```
}

.video-container video {
 min-width: 100%;
 min-height: 100%;
 position: absolute;
 top: 50%;
 left: 50%;
 transform: translateX(-50%) translateY(-50%);
}

/* Just styling the content of the div, the *magic* in the previous rules */
.video-container .caption {
 z-index: 1;
 position: relative;
 text-align: center;
 color: #dc0000;
 padding: 10px;
}
.search-container #city{
 color: aliceblue;
}
 :root {
  --white: #ffffff;
  --off-white: #e5e5e5;
  --transp-white-1: rgba(255, 255, 255, 0.25);
  --transp-white-2: rgba(255, 255, 255, 0.1);
  --green: #47e46e;
  --blue: #4475ef;
  --shadow: rgba(2, 28, 53, 0.2);
 }
 body {
```

```
  height: 100vh;
  background: linear-gradient(135deg, var(--green), var(--blue));

}
.wrapper {
  font-size: 16px;
  width: 90vw;
  max-width: 28em;
  position: absolute;
  transform: translate(-50%, -50%);
  top: 50%;
  left: 50%;
}
.container {
  width: 100%;
  background: var(--transp-white-2);
  backdrop-filter: blur(10px);
  padding: 3em 1.8em;
  border: 2px solid var(--transp-white-2);
  border-radius: 0.6em;
  box-shadow: 0 1.8em 3.7em var(--shadow);
  text-align: center;
}
.search-container {
  font-size: 1em;
  display: grid;
  grid-template-columns: 9fr 3fr;
  gap: 1.25em;
}
.search-container input,
.search-container button {
  outline: none;
  font-size: 1em;
```

```
  border: none;
}
.search-container input {
  padding: 0.7em;
  background-color: transparent;
  border-bottom: 2px solid var(--transp-white-1);
  color:white;
  font-weight: 300;
}
.search-container input::placeholder {
  color: var(--off-white);
}
.search-container input:focus {
  border-color: var(--white);
}
.search-container button {
  color: var(--blue-2);
  background-color: var(--white);
  border-radius: 0.3em;
}
#result h2 {
  color: var(--white);
  text-transform: uppercase;
  letter-spacing: 0.18em;
  font-weight: 600;
  margin: 1.25em 0;
}
.weather,
.desc {
  color: var(--off-white);
  text-transform: uppercase;
  letter-spacing: 0.2em;
  font-size: 0.9em;
```

```
  font-weight: 500;
  line-height: 2em;
}
.weather {
  margin-top: -0.7em;
}
#result img {
  margin: 0.6em 0 0 0;
  width: 6.2em;
  filter: drop-shadow(0 1.8em 3.7em var(--shadow));
}
#result h1 {
  font-size: 4em;
  margin: 0.3em 0 0.7em 0;
  line-height: 0;
  font-weight: 400;
  color: var(--white);
}
.footer
{
 background:linear-gradient(135deg,black,white);
 color: #ffffff;

}
 .temp-container {
  display: flex;
  justify-content: center;
 }
 .temp-container div {
  padding: 0.3em 1em;
 }
 .temp-container div:first-child {
  border-right: 1px solid var(--transp-white-1);
```

```
  }
  .temp-container .title {
   font-weight: 600;
   color: var(--white);
  }
  .temp-container .temp {
   font-weight: 300;
   color: var(--off-white);
  }
  .msg {
   margin-top: 1.8em;
   color: var(--white);
   font-weight: 500;
   text-transform: uppercase;
   letter-spacing: 0.1em;
  }
  #location-details{
   height: 2px;
   width: px;
   color: antiquewhite;
  }
  #get-location img{
   height: 30px;
   width: 20px;
   position: absolute;
    bottom: 8px;
   left: 16px;
  }
  @media screen and (max-width: 450px) {
   .wrapper {
    font-size: 14px;
   }
  }
```

Script.js

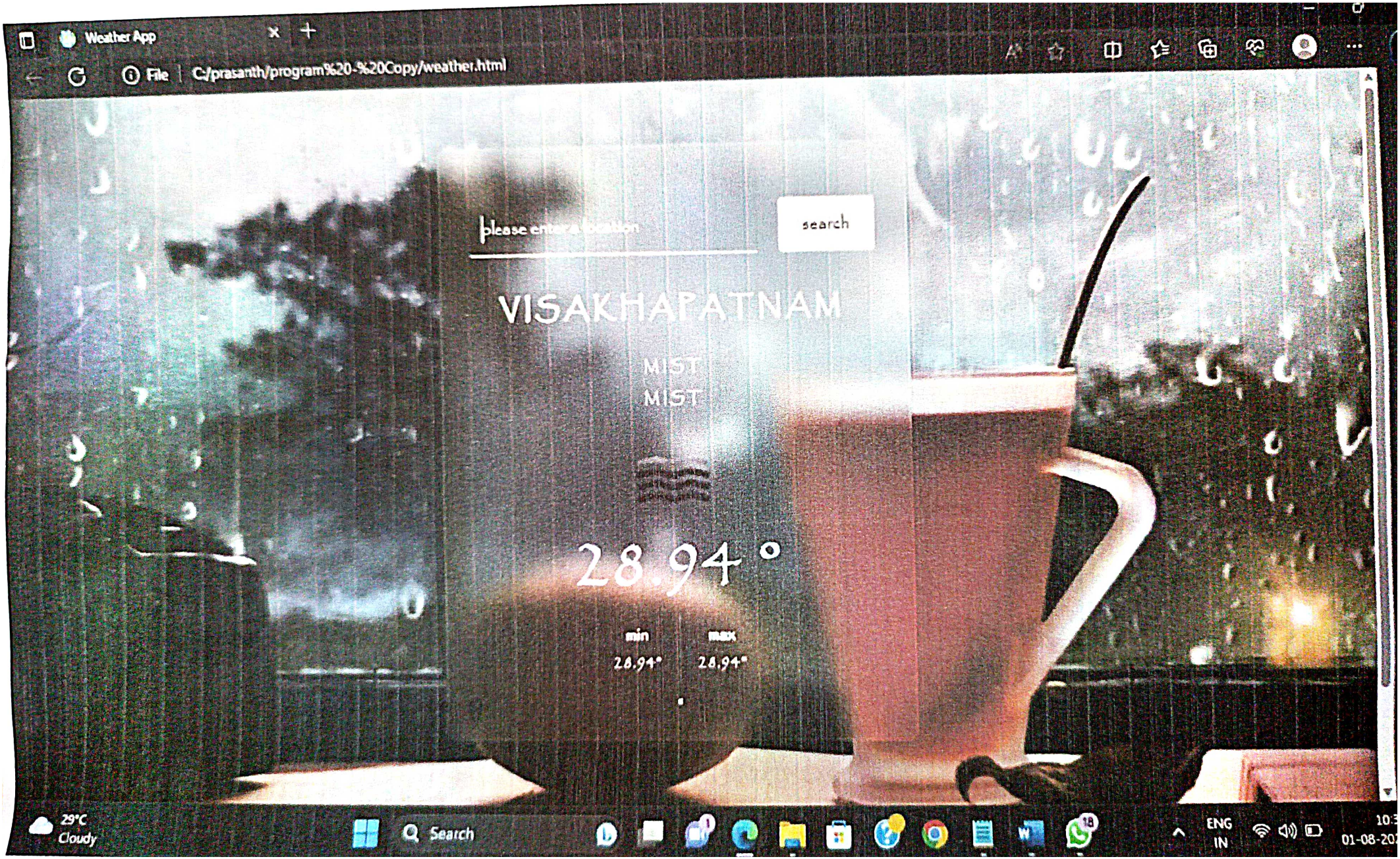
```
let result = document.getElementById("result");
let searchBtn = document.getElementById("search-btn");
let cityRef = document.getElementById("city");
 const key = "1946852246b6fe5011a8a256564b9718";


//Function to fetch weather details from api and display them
let getWeather = () => {
 let cityValue = cityRef.value;
 //If input field is empty
 if (cityValue.length == 0) {
  result.innerHTML = `<h3 class="msg">Please enter a city name</h3>`;
 }
 //If input field is NOT empty
 else {
  let url = `https://api.openweathermap.org/data/2.5/weather?q=${cityValue}&appid=${key}&units=metric`;
  //Clear the input field
  cityRef.value = "";
  fetch(url)
   .then((resp) => resp.json())
   //If city name is valid
   .then((data) => {
    console.log(data);
    console.log(data.weather[0].icon);
    console.log(data.weather[0].main);
    console.log(data.weather[0].description);
    console.log(data.name);
    console.log(data.main.temp_min);
    console.log(data.main.temp_max);
    result.innerHTML = `
    <h2>${data.name}</h2>
    <h4 class="weather">${data.weather[0].main}</h4>
```

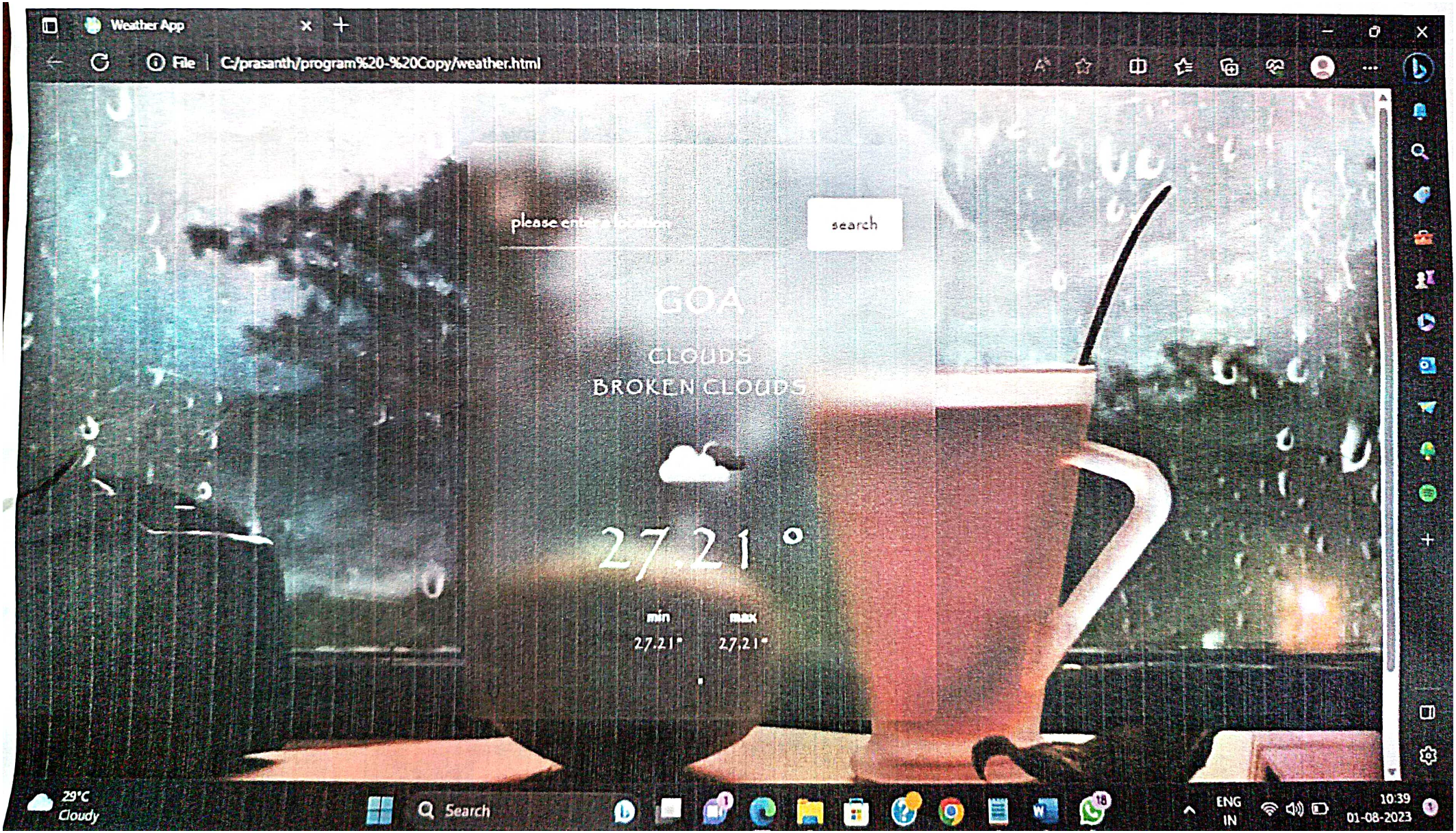
```
    <h4 class="desc">${data.weather[0].description}</h4>
    <img src="https://openweathermap.org/img/w/${data.weather[0].icon}.png">
    <h1>${data.main.temp} &#176;</h1>
    <div class="temp-container">
      <div>
        <h4 class="title">min</h4>
        <h4 class="temp">${data.main.temp_min}&#176;</h4>
      </div>
      <div>
        <h4 class="title">max</h4>
        <h4 class="temp">${data.main.temp_max}&#176;</h4>
      </div>
    </div>
    `;
  })
  //If city name is NOT valid
  .catch(() => {
    result.innerHTML = `<h3 class="msg">City not found</h3>`;
  });
 }
};
searchBtn.addEventListener("click", getWeather);
window.addEventListener("load", getWeather);
```

script1.js

```
let locationButton = document.getElementById("get-location");
let locationDiv =document.getElementById("location-details");

locationButton.addEventListener("click", () => {
 //Geolocation APU is used to get geographical position of a user and is available inside the navigator object
 if (navigator.geolocation) {
  //returns position(latitude and longitude) or error
  navigator.geolocation.getCurrentPosition(showLocation, checkError);
 } else {
```

```
        //For old browser i.e IE
        locationDiv.innerText = "The browser does not support geolocation";
      }
    ));

    //Error Checks
    const checkError = (error) => {
     switch (error.code) {
      case error.PERMISSION_DENIED:
       locationDiv.innerText = "Please allow access to location";
       break;
      case error.POSITION_UNAVAILABLE:
       //usually fired for firefox
       locationDiv.innerText = "Location Information unavailable";
       break;
      case error.TIMEOUT:
       locationDiv.innerText = "The request to get user location timed out";
     }
    };

    const showLocation = async (position) => {
     //We user the NOminatim API for getting actual addres from latitude and longitude
     let response = await fetch(

`https://nominatim.openstreetmap.org/reverse?lat=${position.coords.latitude}&lon=${position.coords.longitude}&format=json`

     );
     //store response object
     let data = await response.json();
     locationDiv.innerText = `${data.address.city}, ${data.address.country}`;
    };
```


Weather App
File
C:/prasanth/program%20-%20Copy/weather.html
please enter a location
search
VISAKHAPATNAM
MIST
MIST
28.94°
min
28.94°
max
28.94°
29°C
Cloudy
Search
ENG
IN


Weather App
File | C:/prasanth/program%20-%20Copy/weather.html
please enter a location
search
GOA
CLOUDS
BROKEN CLOUDS
27.21 °
min
27.21°
max
27.21°
29°C
Cloudy
Search
ENG
IN
10:39
01-08-2023

ADVANTAGES AND DISADVANTAGES

⇒ Advantages:

Skill Enhancement:

Developing a weather app as a front-end project allows front end developers to improve their skills in HTML, CSS and Java Script.

Real-world Application:

A weather app is a pratical project that provides real-world value to users. It also allows developers to work on something relevant.

User Interface Design:-

Weather apps require an intative and wally appealing user intuface. Building Such an interface helps to sharpen their design.

Reliance on Technology:-

Weather forecasting relies heavily on technology and if the technology fails (or) is unavailable accurate predictions cannot be made.

Limited Time Frame:-

Forecasts are usally only accurate for a short time frame, making it different to plan ahead.

→ Disadvantages :-

Data limitations

Front-end developers rely on on Weather API's to fetch weather data. The amount of data and the available features are dependent.

Lack of Backend Experience :

Building a weather app purely as a front end project may not provide opportunities to gain experience in Server-side programming.

Security Concerns :

Handling API's and external data sources requires careful consideration of security to prevent data breaches to sensitive information.

Confusing Terminology :-

The terminology used in weather forecasting can be confusing, making it difficult for some people to understand predictions.

Limited Reach :

Weather forecasts are not available for many remote or sparsely populated areas, making it difficult for people in these areas to prepare for severe weather.

APPLICATIONS

Real-Time Weather Information:-

Display current weather conditions, including temprature, humidity, wind speed and direction, along with an icon representing weather type. Ex: Sunny, Cloudy, Rainy.

Location-Based Forecast:-

Allow users to enter their locations (or) use their devices Gps to get localized weather forecasts for the current day and upcoming days.

Multiple locations:-

Enable users to save and switch b/w multiple locations, so they can check weather for places they frequently visit.

Weather Radar and Maps:-

Implement weather radar and interactive maps to visualize weather, patterns including rain, snow and cloud covers.

Weather Alerts and Warnings:-

Display severe weather alerts & warnings for the user's location (or) selected regions, ensuring users stay informed about potentially dangerous conditions.

Hourly and Daily forecasts:-

Provide detailed weather forecasts for the next few hours and several days ahead, giving users a comprehensive view of what to expect

User prefrences:-

Let users customize the app by setting the temprature units.

Ex:- Celsius (or) Fahrenheit.

Historical Weather Data:-

Offer access to historical weather data, allowing user to explore past weather patterns.

Social Media Integration:-

Allow users to share weather updates on social media platforms.

Responsive Designs:

Ensure the app is fully responsive and optim-ized for various devices.

Accessibility:

Make the app accessible to users with the disabilities by gadhering to accessibility stand-ards and guidlines.

CONCLUSION

The Weather apps are increasingly accurate and useful, and their benefits extend widely across the economy. While much has been accomplished in improving weather forecasts, their remains much room for improvements.

Simultaneously, they are developing new technologies and observational networks that can enhance forecaster skills.

FUTURE SCOPE:

The demand for weather and climate forecast information in support of critical decision making has grown rapidly during the last decade, and will grown even faster in the coming years. Great advances has been made in the utilization of predictions in many years of human activities.

The future of weather applications is promising, with the increasing demand for real time and accurate weather information.