PROJECT REPORT

1. Introduction

Overview

A brief description about your project

purpose

The use of this project. what can be achieved using this.

2. LITERATURE SURVEY

Existing problem

Existing approaches (or) method to solve this problem

3. THEORITICAL ANALYSIS

Block diagram

Diagrammatic overview of the project

Hardware / software designing

.Hardware and software requirements of the project

4. RESULT

final findings of the project along with screenshots

5. Advantages & Disadvantages

List of advantages and disadvantages of proposed solution

6. Application

the areas where this solution can be applied

7. Conclusion

conclusion Summarizing the entire work and findings

8. Future Scope

Enhancements that can be made in the future

PROJECT REPORT FORMAT

Introduction

overview

Weather app is a one step - Solution for staying up-to-date with real-time weather forecast

This project is an exciting endeavor in front end development" aimed at providing users with a sleek and inductive weather application our mission is to deliver an Engaging user Experience by presenting weather data in a visually appealing and informative manner

purpose:

Weather plays a significant role in our daily lives influencing ever our activities clothing choices and overall well being people constantly seek accurate weather information to plan their schedules accordingle while many weather application exist, weather app stands out by priortizing user Experience and Simplicity

The purpose of a weather app project is to create Software application that provide users with real-time weather information and fore casts for a Specific location or multiple location

objectives

LITERATURE SURVEY

Real-time weather data: The app should able to fetch and display current weather conditions, including temperature humidity wind speed and visibility for the users chosen per location

weather fore cast: providing accurate weather forecasts for the next few days is crucial as it help users plan a head for events travel or outdoor activities.

location based services

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

user-friendly interface:

The app should have an intative and as 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 (eg: celsium or farenheit) or choose the time format.

PROPOSED SOLUTION

weather alerts: the App may include a feature to send weather alerts & notifications to users for severs weather conditions like storms, extreme temp

Maps and Radar

Including weather maps and radar data can help users visualize weather patterns and storms in real-time

Energy Efficiency:

weather app projects may focus on optimizing battery usage, Especially for mobile devices

Integration with APIs

The app may utalize third-party weather APIs to access accurate and up-to date weather data

Cross-platform compatability:

To reaches a brodes audience the app Shoutd be compatible with different operating system such as Android, iOS, windows and web browser

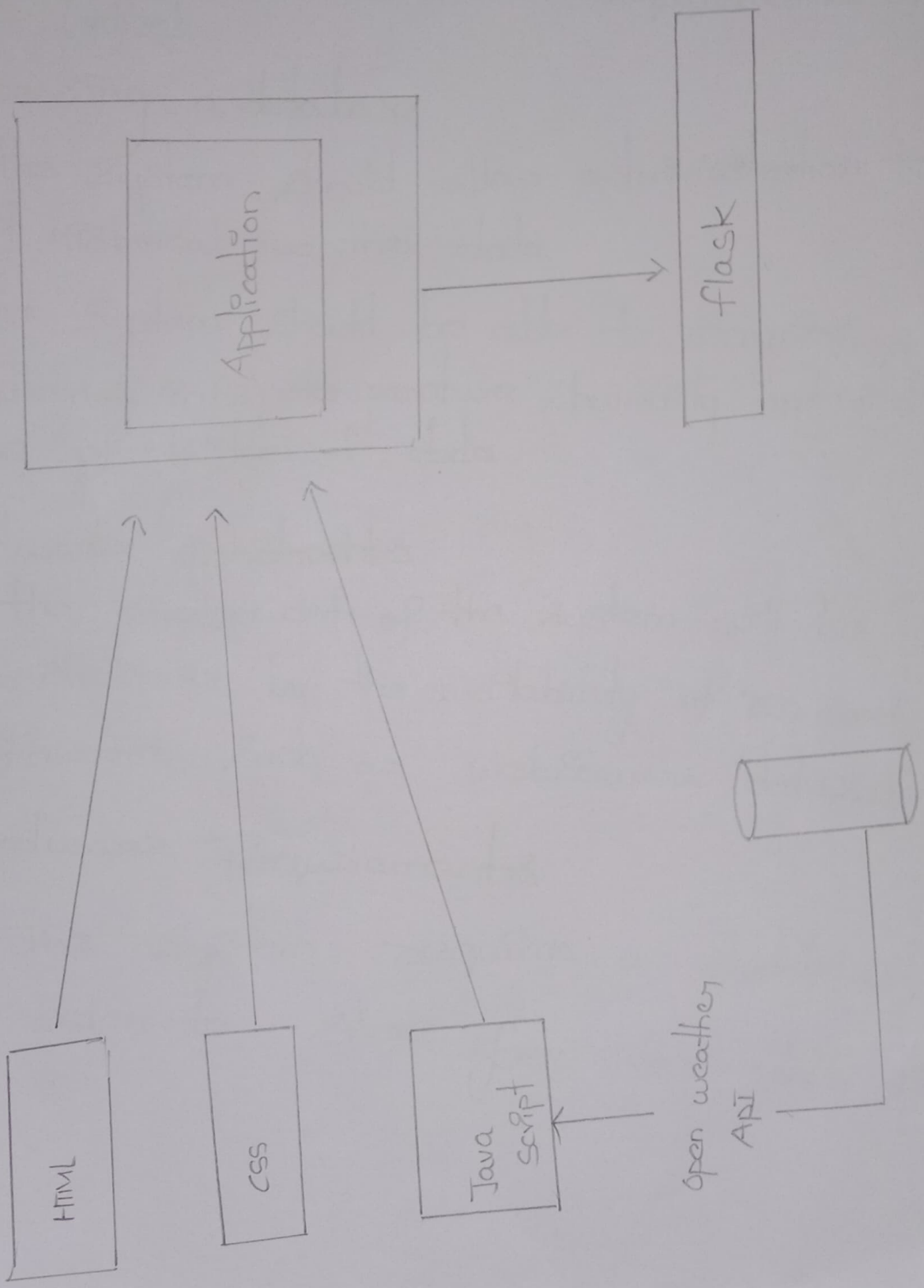
offline access:

although real-time data is Essential the app project might consider providing basic weather information Even when the device is offline

overall, the primary purpose of a weather app project is to offer users a Convenient and reliable to or to access weather information at their fingertips allowing them to make informed decisions based on current and fore casted weather conditions.

Theoritical Analysis

→ Block diagram



→ Hardware/Software designing

Hardware and Software requirements of the project

Accessing a database

→ the System should allow administration to add historical weather data

→ the System should be able to recognize patterns in(s) temperature, humidity and wind use of historical data

Software Constraints

→ the development of the system will be constrained by the availability of required Software such as webServers, dataset

Hardware Requirements

→ The System requires a database in order to store persistend data.

INDEX.HTML

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

<head>
  <!-- ---------- Created By InCoder ---------- -->
  <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 - InCoder</title>
  <link rel="stylesheet" href="style.css">
  <link rel="icon" href="https://img.freepik.com/free-photo/rainbow-colors-reflect-tranquil-forest-pond-generated-by-ai_188544-44849.jpg?size=626&ext=jpg&ga=GA1.2.1809880757.1688533325&semt=sph" type="png">
</head>

<body class="w_bg">
  <div class="mainContainer">
    <div class="searchInput">
      <input type="text" placeholder="Enter City Name" id="searchInput" value="visakhapatnam"/>
      <button id="searchButton"><i class="fa-solid fa-magnifying-glass"></i></button>
    </div>
    <div class="weatherDetails">
      <div class="weatherIcon">
        <img src="Images/sun.png" alt="Clouds" id="weatherIcon">
      </div>
      <div class="cityDetails">
        <div class="weather" id="weather"></div>
        <div class="desc"></div>
      </div>
      <div class="windDetails">
```

INDEX.HTML

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

<head>
  <!-- ---------- Created By InCoder ---------- -->
  <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 - InCoder</title>
  <link rel="stylesheet" href="style.css">
  <link rel="icon" href="https://img.freepik.com/free-photo/rainbow-colors-reflect-tranquil-forest-pond-generated-by-ai_188544-44849.jpg?size=626&ext=jpg&ga=GA1.2.1809880757.1688533325&semt=sph" type="png">
</head>

<body class="w_bg">
  <div class="mainContainer">
    <div class="searchInput">
      <input type="text" placeholder="Enter City Name" id="searchInput" value="visakhapatnam"/>
      <button id="searchButton"><i class="fa-solid fa-magnifying-glass"></i></button>
    </div>
    <div class="weatherDetails">
      <div class="weatherIcon">
        <img src="Images/sun.png" alt="Clouds" id="weatherIcon">
      </div>
      <div class="cityDetails">
        <div class="weather" id="weather"></div>
        <div class="desc"></div>
      </div>
      <div class="windDetails">
```

```
            <div class="humidityBox">
              <img
src="https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEhgr7XehXJkOPXbZr8xL42sZEFYlS-1fQcvUMsS2HrrV8pcj3GDFaYmYmeb3vXfMrjGXpViEDVfvLcql7pJ03pKb_9ldQm-Cj9SlGW2Op8rxArglhlD6oSLGQQKH9IqH1urPpQ4EAMCs3KOwbzLu57FDKv01PioBJBdR6pqlaxZTJr3HwxOUlFhC9EFyw/s320/thermometer.png" alt="Humidity">
              <div><p class="humidity"></p><span>Humidity</span></div>
            </div>
            <div class="windSpeed">
              <img
src="https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEiyalguDPkbBMnUDQkGp3wLRj_kvd_GIQ4RHQar7a32mUGtwg3wHLIe0ejKqryX8dnJu-gqU6CBnDo47O7BlzCMCwRbB7u0Pj0CbtGwtyhd8Y8cgEMaSuZKrw5-62etXwo7UoY509umLmndsRmEqqO0FKocqTqjzHvJFC2AEEYjUax9tc1JMWxIWAQR4g/s320/wind.png"
>
              <div><p id="windSpeed"></p><span>Wind Speed</span></div>
              </div>
            </div>
          </div>
        </div>
      </div>

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

</html>
```

STYLE.CSS

```
@import
url("https://fonts.googleapis.com/css2?family=Ubuntu:ital,wght@0,300;0,400;0,500;0,700;1,300;1,4
00;1,500;1,700&display=swap");

/* ---------- Created By InCoder ---------- */

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Ubuntu", sans-serif;
}



body {
  display: flex;
  height: 100vh;
  align-items: center;
  justify-content: center;
  background-color: #9c9dde;
  background-image: url("bg.jpg");
  background-size: cover;

}/*
.w_bg{
    background-image: url(./nature_landscape_bg_img.jpg);
}*/
.mainContainer {
  width: 20rem;
  height: auto;
  border-radius: 1rem;
  backdrop-filter: blur(10px);
```

```
}

.searchInput {
  width: 100%;
  display: flex;
  padding: 1rem 1rem;
  justify-content: center;
}

.searchInput input {
  width: 100%;
  height: 2rem;
  outline: none;
  font-size: 0.8rem;
  color: #525050;
  padding: 0.2rem 0.5rem;
  border-radius: 1.5rem;
  border: 1px solid #b3b3b3;
}

.searchInput input:focus {
  border: 1px solid #9c9dde;
}

.searchInput button {
  width: 2.2rem;
  height: 2rem;
  cursor: pointer;
  color: #f8f4f4;
  border-radius: 50%;
  margin-left: 0.4rem;
```

```
 transition: all 0.3s ease;
 background-color: #080808;
 border: 1px solid #b3b3b3;
}

.searchInput button:hover {
 color: #8a8686;
 background-color: #9c9dde;
 border: 1px solid #9c9dde;
}

.weatherIcon {
  display: flex;
  padding-top: 0.5rem;
  padding-bottom: 0.5rem;
  justify-content: center;
 }

 .weatherIcon img {
   max-width: 100%;
   width: 8rem;
 }

 .cityDetails {
   color: #f5f4f4;
   font-size: 2.5rem;
   text-align: center;
   margin-bottom: 0.5rem;
 }

 .windDetails {
```

```
 display: flex;
 margin-top: 1rem;
 margin-bottom: 1.5rem;
 justify-content: space-around;
}

.windDetails .humidityBox {
 display: flex;
 font-size: 1rem;
 color: #f1eded;
}

.windDetails .humidityBox img {
  max-height: 3rem;
  margin-right: 0.5rem;
}

.windDetails .windSpeed {
  display: flex;
  font-size: 1rem;
  color: #f5f2f2;
}

.windDetails .windSpeed img {
  max-height: 3rem;
  margin-right: 0.5rem;
 }
```

RIPT.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.coor
ds.longitude}&format=json`

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


Weather App
Weather App - InCoder
File
C:/Users/prasa/OneDrive/Desktop/uday/index.html
visakhapatnam
28°c
mist
94%
Humidity
1.03km/h
Wind Speed

5. Advantages And Disadvantages

Advantage

→ Advantages of a weather App in front End developer

1. Skill Enhancement: Developing a weather app as a front End project allows front-End project allows front End developers to improve their skills in html, css and java
2. Real word application: a weather app is practical project that provides Real-word on value to users if allows developers to work on something relavant and useful Enhancing their portfolio Employes or clients.
3. user interface design: weather apps required on create and visually appealing user interface building such as interface keeps front-End developer shapes this design and user Experience (ux) skills
4. API integration: Integration weather data APIs into the app teaches developers how to work with External data source and handle asynchronous request a crucial aspect of modern web development.
5. cross-browser compatibility: front-End development must Ensure the app functions correctly across different web browser handles - on Experience in dealing with cross browser Compatibility issues.

Disadvantages

Disadvantages of a weather app in front-end develops

1. Limited scope: a weather app, while useful may be considered a relatively simple project in terms of functionality front-end developers may miss the opportunity to work on more complex applications that involve backend development or database integration.
2. 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 database management (or) back end architecture
3. Data limitations: front end developers rely on weather APIs to fetch weather data the amount data and of the available features are dependent on the capability of the chosen API, limiting the scope for data manipulation and analysis
4. Security concerns: Handling APIs and external data sources requires careful consideration of security to prevent data breaches or unauthorized access to sensitive information
5. performative challenges: Depending on the API and data retrieval methods front-end developers may encounter performance issues if the app requires frequent updates (or) data

Applications

→ Real-time weather information: Display current weather conditions including temperature humidity wind speed and direction along with an icon representing the weather type [eg: sunny, cloudy, rainy]

→ location-based forecast: allow users to enter their location or use their devices Gps to get localized weather forecasts for the current day and the upcoming days.

→ multiple locations: Enable users to save and switch between multiple locations, so they can check the weather for places they frequency visit or plan to travel

→ weather Radar and maps: Implement weather radar and interactive maps to visualize weather patterns including rain, snow and cloud cover

→ weather alerts and warnings: Display severe weather alerts and warnings for the users 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 a head giving users a compre-hensive view of what to expect

Historical weather data: offer access to historical weather data, allowing user to explore past weather patterns and trends

user preferences: let users customize the app by setting temperative units [Eg: celsius or fahrenheit] language there and other personal preferences

weather widgets: Create small weather widgets that can be embeded on the websites or shared on social media plat form

Responsive design: Ensure the app is fully responsive and optimized for various devices including desktops tables and mobile phones

Accessibility: make the app accessible to users the disabilities by adhering to accessibility standards and guidelines.

Animations and transitions: use subtle animations and smooth transitions to enhance the user Experience and make the app feel more interactive.

offline Support: Implement caching and storage mechanisms to provide basic weather information even when the user offline.

Social media integration: allow users to share weather updates on social media platforms.

Conclusion.

Project overview briefly summarize the purpose of the weather app its target audience and technolo -gies used in its development.

Feature and functionality : highlight the key features implemented in the app such as real-time weather data retrival location-based weather forecasts interactive UI elements i.e responsive design for different devices and any element other unique function -alities

Design and ues Experience discuss the design choices made through out the project including color schemes typography conography and overall ues interface such as integrating third-party APIs handling asynchronous data retrieval or Ensuring cross-browser Compatibility Explain how you over Create these challenges

Responsiveness and compatibility discuss the efforts put into Ensuring that the weather app is respon -sive and compatible with various devices and brauser mention any specific break points or media queries used to optimize the apps displays on different screen sizes

8. FUTURE SCOPE

- future scope of weather app in front-end developer

1. Real-time data and personalization: weather apps of the future will likely offer more real time and hyper localized data front-end developers can beverage technologies like geolocations and APIs that provides up to data weather information for users exact locations personalization feature may be incorporated to codes to individual preferences and user behaviours
2. Enhanced user interfaces: front-end developers will play a crucial role in designing immersive and interfaces weather apps can incorporative Engaging animations 3D elements and intertive gestures to makes the experience more enjoyable and user friendly
3. progressive web apps (PWAS): the adoption of PWAS is expected to grow offering users an app-like experience create weather PWAS that work offline load quickly and seamlessly adopt to various screen sizes and devices
4. Voice and creature controll: with the rise of voice assistants and gesture based interactions, front end developers can explore integrating voice commands and gesture controls into weather apps. this approach enhances accessibility and