| **Tech Saksham**  Final Project Report  **TS-WEB TRACK CAPSTONE PROJECT** |  |  |
| --- | --- | --- |

**“WEATHER FORECASTING”**

**“AVANTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY”**

| **ROLL NO** | **NAME** |
| --- | --- |
| 19PT1A0533 | T.RUTHU |
| 19PT1A0534 | T.JAHNAVI |
| 19PT1A0524 | M.ANJALI DEVI |

|  |  |
| --- | --- |
|  | **Trainer Name:** |
|  | Uma maheshwari |

**ABSTRACT**

A weather forecasting HTML/JavaScript project would involve creating a web application that allows users to access current and forecasted weather information for a given location. The project would involve using APIs provided by weather data providers, such as OpenWeatherMap or AccuWeather, to retrieve weather data and present it to users in a user-friendly way.

The web application would need to have a user interface that allows users to input their location or select it from a pre-defined list. The application would then use this information to retrieve the corresponding weather data and display it to the user.

The application would also need to provide features such as temperature, wind speed, humidity, and precipitation. It would need to be able to handle different units of measurement, such as Celsius or Fahrenheit, and display the information accordingly.

To implement the project, you would need to have knowledge of HTML, CSS, and JavaScript. You would also need to be familiar with working with APIs and making HTTP requests to retrieve data.

Overall, a weather forecasting HTML/JavaScript project would be a great opportunity to learn about web development and working with APIs while also providing a useful tool for users to access weather information.

**INDEX**

| **Sr. No.** | **Table of Contents** | **Page No.** |
| --- | --- | --- |
| 1 | Chapter 1: Introduction | 1-2 |
| 2 | Chapter 2: Services and Tools Required | 3-6 |
| 3 | Chapter 3: Project Architecture | 7-8 |
| 4 | Chapter 4: HTML Source code | 9-11 |
| 5 | Chapter 5: Javascript Source Code | 12-14 |
| 6 | Conclusion | 15 |
| 7 | future scope | 16 |
| 8 | references | 17-18 |

**CHAPTER 1**

**INTRODUCTION**

Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Ancient weather forecasting methods usually relied on observed patterns of events, also termed pattern recognition. For example, it might be observed that if the sunset was particularly considered, the following day often brought fair weather. However, not all of these predictions prove reliable

**1.1OBJECTIVE OF THE SYSTEM**

This project will serve the following objectives:-

1. Provides the user with an easy and friendly interface

2. Provides the user with the temperature of a particular region

3 It will also show humidity, wind speed and cloud

**1.2 JUSTIFICATION AND NEED FOR THE SYSTEM**

Weather is something everybody deals with, and accurate data about it like what is coming can help users to make informed decisions. With weather apps for iOS and Android, people can exactly know when to expect a change in the weather conditions. Weather apps can give urgent alerts too.

Undoubtedly, weather forecasting has come a long way, helping people to know about weather conditions. So, if you are in an area where weather frequently changes from sunny to torrential rain in a matter of minutes, then what is the easiest way to make sure to be prepared? A suitable answer is a weather application.

**1.3 ADVANTAGES**

**1. Real-Time Data**

One of the biggest advantages of weather monitoring been going in for weather stations is because of the ability to get their information in real-time.

**2. Accurate Local Forecast**

In reality, the meteorological department may be located far from your home and weather forecasts are made for regions, not a specific area. That’s a reason why in these instances, the weather predictions that they give are not always the most accurate.

**3. Ease Of Use**

Ease to use is definitely a big advantage of the weather monitoring system. Weather stations like all other weather devices are designed to be efficient and straightforward, therefore, everyone can use them.

It is so convenient and comfortable for users to get the most accurate information in the simplest way possible.

**1.4 Previous work or related systems; how they are used.**

Before we begin a new system it is important to study the system that will be improved or replaced (if there is one). We need to analyze how this system uses hardware, software, network and people resources to convert data resources, such as transaction data, into information products.

Following are the problems associated with the previous project which led to the creation of the proposed project:-

1. Not user-friendly: The existing system is not user-friendly because the information like humidity cloud and wind etc are not in one place.

2. Not a good UI: The user interface of the previous systems are not that good

**CHAPTER 2**

**SERVICES AND TOOLS REQUIRED**

**2.Functional Requirements**

Functional requirements are the requirements that describe the functionalities of the system elements. It may involve functional user requirements or functional system requirements.

For example:

The operator shall be able to input the region to the system to view the desired weather parameters.

The system shall provide the following weather parameters: temperature, pressure, wind speed ,date /time and humidity.

**2.1.1 USER REQUIREMENTS**

The system specifications that a user may want are as follows:

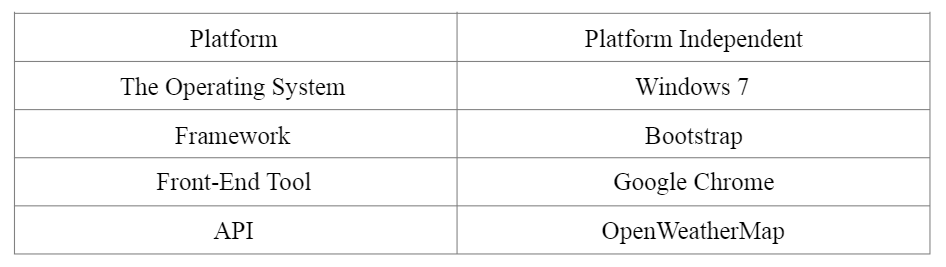
1. It should be easy to understand

2. Must be interactive

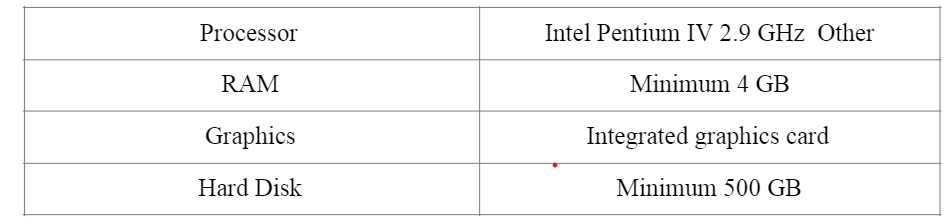
3. Should provide a good user interface

4. Security should be maintenance

**SOFTWARE REQUIREMENTS**

****

**HARDWARE REQUIREMENTS:**

****

**3. system requirements:**

Mainly there are six requirements which an SRS must satisfy.

(a) It should specify the external behaviour.

(b) It should specify the constraints.

(c) It should be easy to change.

(d) It should be a reference tool.

(e) It should record throughout the lifecycle.

(f) It should have the capacity to expect an undesired event.

**Functional Requirements**

Functional requirements are the requirements that describe the func)onali)es of the system elements. It may involve func)onal user requirements or func)onal system requirement**s.**

**For example:**

The operator shall be able to input the region to the system to view the desired weather parameters.

The system shall provide the following weather parameters: temperature, pressure, wind speed & direction, rainfall, and humidity.

**3.3 Design Requirements**

The main objectives of input design are:

(a) Controlling the amount of input

(b) Keeping the process simple.

(c) The best thing in the input design is to achieve all the objectives mentioned in the simplest manner possible.

The main objectives of output design are:

(a) Identifying the specific outputs.

The primary goal of the system analysis is to improve the efficiency of the existing system. For that the study of specification of the requirements is very essential. For the development of the new system, an preliminary survey of the existing system will be conducted. Investigation done whether the upgradation of the system into an application program could solve the problems and eradicate the inefficiency of the existing system

**CHAPTER 3**

**PROJECT ARCHITECTURE**

**3.1 Architecture**

A weather forecast project using HTML and CSS would involve creating a web page that displays weather information for a particular location. The project architecture would include the following components:

**HTML structure:** The HTML structure of the project would define the layout of the web page. It would include the necessary HTML tags to structure the content, such as headers, paragraphs, and divs.

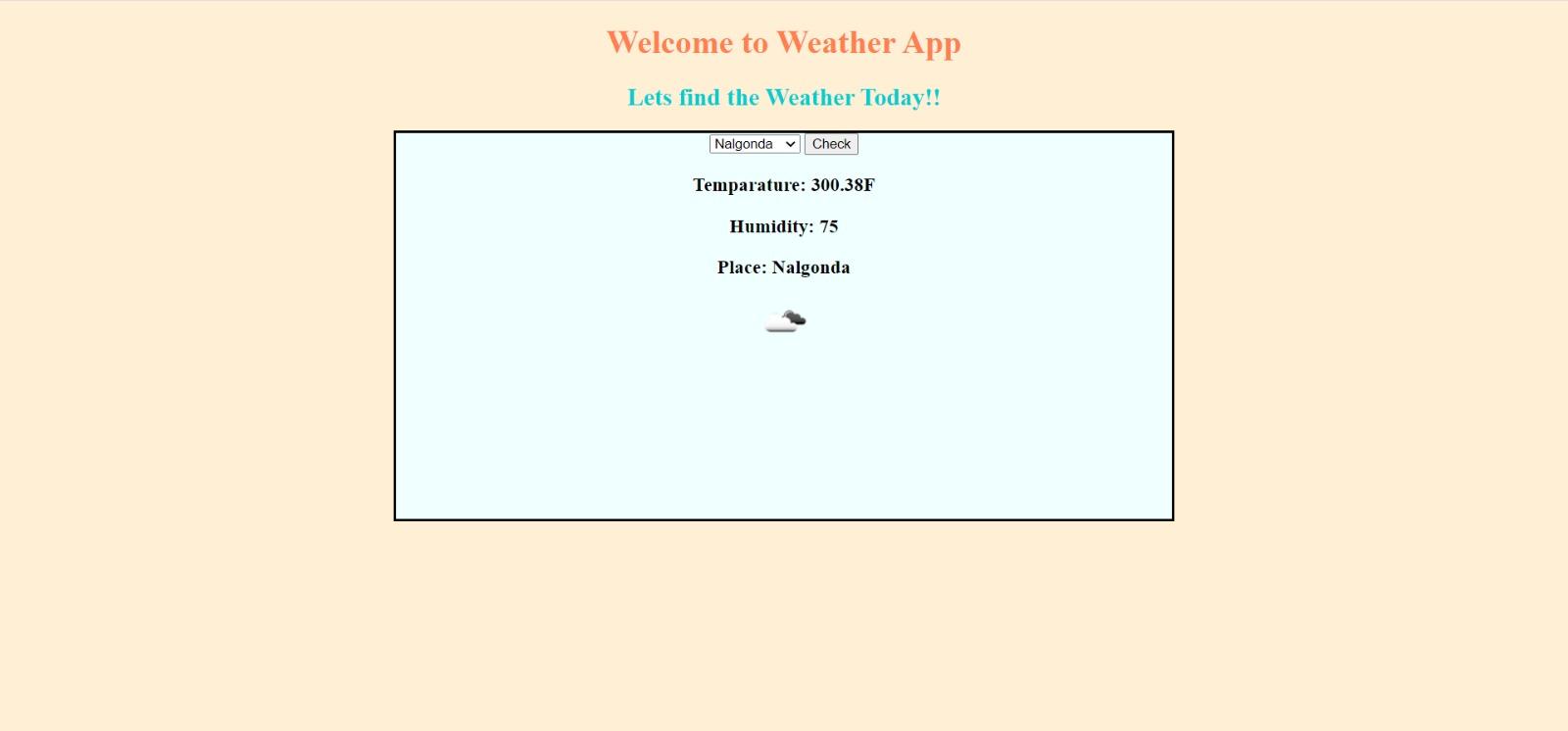
**CSS styling:** The CSS styling of the project would define the visual appearance of the web page. It would include the necessary CSS rules to set the font, color, and layout of the content.

**Weather API:** To retrieve weather data for a particular location, the project would use a weather API, such as OpenWeatherMap or AccuWeather. The API would be called using JavaScript and the data retrieved would be displayed on the web page.

**JavaScript code:** The JavaScript code would handle the interaction between the web page and the weather API. It would make the necessary HTTP requests to the API and parse the JSON data returned by the API. The JavaScript code would also handle any user interaction, such as changing the location or unit of measurement.

**User interface:** The user interface of the project would allow the user to input their location or select it from a pre-defined list. The interface would also provide options to select the unit of measurement and display the weather information accordingly.

Overall, a weather forecast project using HTML and CSS would involve creating a visually appealing web page that displays weather information for a particular location. It would require a good understanding of HTML and CSS, as well as JavaScript and working with APIs.

****

****

**HTML SOURCE CODE**

<!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</title>

style>

h2{

text-align: center;

}

#myimg{

display: block;

margin-left: auto;

margin-right: auto;

}

.drops{

text-align: center;

border: 3px solid black ;

height: 10cm;

margin-left: 10cm;

margin-right: 10cm;

background-color: azure;

}

.button{

position: relative;

}

</style>

<script src="whetherapp.js"></script>

</head>

<body style="background-color:papayawhip;">

<h1 style="text-align: center;color: coral;">Welcome to Weather App</h1>

<h2 style="color: darkturquoise;">Lets find the Weather Today!!</h2>

<div class="drops">

<select id="myDropdown">

<option value = "Hyderabad">Hyderabad</option>

<option value = "Chennai">Chennai</option>

<option value = "Nalgonda">Nalgonda</option>

<option value = "Delhi">Delhi</option>

<option value = "Kerala">Kerala</option>

</select>

<button value="submit" name="submit" onclick="fun()" class="button">Check</button><br>

<h3 id="mypara"></h3>

<h3 id="humidity"></h3>

<h3 id="place"></h3>

<img id="myimg" src='' />

</div>

</body>

</html>

**JAVA SCRIPT SOURCE CODE**

let request = new XMLHttpRequest();

//open a connection

function fun() {

var dropdown = document.getElementById("myDropdown");

if (dropdown) {

var selectedOption = dropdown.value;

console.log(selectedOption);

} else {

console.log("Dropdown not found");

}

var place = dropdown.value;

console.log(place.value)

let strr = `https://api.openweathermap.org/data/2.5/weather?q=${place}&appid=93f26e3c57081a6210de53b8dcfdfea4`

request.open('GET', strr, true);

// console.log(selectedOption)

request.onload = function () {

if (request.status >= 200 && request.status < 400) {

//get the data

console.log("success!!!");

let data = JSON.parse(request.responseText);

console.log(data)

let imgsrc = 'https://openweathermap.org/img/w/' + data.weather[0].icon + '.png';

document.getElementById('mypara').innerHTML = "Temparature: "+data.main.temp + 'F';

document.getElementById('humidity').innerHTML = "Humidity: "+data.main.humidity;

document.getElementById('myimg').src = imgsrc;

document.getElementById('place').innerHTML = "Place: "+place;

}

else {

console.log("couldnot connect to server");

}

}

//error checking

request.onerror = function () {

console.log("error!!");

}

//send the request

request.send();

}

**CONCLUSION**

Now a day’s there is a big demand of different types of applications, which is because IT has become the main part of our New World. There is a big need of different applications. People want application for every specific task from work to entertainment. We have developed the application “Weather WebApp” which works easy on any given web browser

.

The application has been tested and found to be working as per the given criteria. It can be safely concluded that the application possesses a highly efficient UI system and is working properly and meeting to all the requirements of the user. The application gives the user maximum flexibility in the types of touch and other device movements.

**FUTURE SCOPE**

Every project whether large or small has some limitations no matter how diligently developed. In some cases, limitations are small while in other cases they may be broad also. The new system has got some limitations. Major areas where modifications can be done are as follows:

•Our system does not have weather information for foreign countries or cities.

•There is no provision for complaint handling so further it can be added

**REFERENCES**

Here are some references for weather app projects using HTML and CSS:

"Build a Weather App" by Kevin Powell - This YouTube tutorial walks you through building a weather app using HTML, CSS, and JavaScript. **https://www.youtube.com/watch?v=GuA0\_Z1llYU**

"Weather App using HTML, CSS and JS" by Web Dev Simplified - This tutorial includes step-by-step instructions for creating a weather app using HTML, CSS, and JavaScript. **https://www.youtube.com/watch?v=wPElVpR1rwA**

"Build a Weather App with HTML, CSS, and JavaScript" by FreeCodeCamp - This tutorial guides you through creating a weather app using HTML, CSS, and JavaScript. **https://www.freecodecamp.org/news/how-to-build-a-weather-app-with-html-css-and-javascript/**

"Build a Weather App with HTML, CSS and JavaScript" by Traversy Media - This tutorial covers creating a weather app using HTML, CSS, and JavaScript, and also includes a section on using the OpenWeatherMap API. **https://www.youtube.com/watch?v=1K7YvOoBgZU**

"Creating a Weather App with Vanilla JavaScript" by Scotch.io - This tutorial teaches you how to create a weather app using HTML, CSS, and vanilla JavaScript, and also covers using the OpenWeatherMap API. **https://scotch.io/tutorials/create-a-simple-weather-app-with-vanilla-javascript**

These tutorials should provide you with a good starting point for building your own weather app using HTML and CSS.

**CODE**

**Please Provide Code through Git Hub Repo Link**