Weather Monitoring System Using IOT and Android Application

KEVALKUMAR MANDANKA
INFORMATION TECHNOLOGY
CHANDUBHAI S PATEL
INSTITUTE OF TECHNOLOGY
GUJARAT,INDIA
19it063@charusat.edu.in

AKSHAY MALAVIYA
INFORMATION TECHNOLOGY
CHANDUBHAI S PATEL
INSTITUTE OF TECHNOLOGY
GUJARAT,INDIA
19it061@charusat.edu.in

Under Guidance of:

Prof. Ravi Patel

Chandubhai S Patel Institute Of Technology
Department Of Information And Technology
At. Changa, Dist Anand-
388421,Gujarat,India

ABSTRACT :

The weather station is established to collect quantitative data about the weather conditions in a certain place. Due to uncertain weather changes every day, monitoring the weather conditions in today's environment is considered very important. This research attempts to make the weather station accessible through the website through the Internet of Things platform. Users do not need to enter the area to know the weather changes in the area. This design uses Arduino ESP8266 as the microcontroller. The measured weather parameters include temperature and humidity using DHT-11 sensor. The measurement results of all sensor is stored in the Google Firebase and displayed on the I2C LCD Display and Android Application.

KEYWORDS: IOT TECHNOLOGY; WEATHER MONITORING SYSTEM; EMBEDDED SYSTEM, MICROCONTROLLER, ENVIRONMENTAL DATA STUDY,ANDROID APPLICATION

I. INTRODUCTION

A weather station is a technology for collecting data Different use related to weather and environment Electronic sensor. There are two types of weather stations, One is installing sensors, the second is The weather station is where we extract data from the weather station server. In this project, we designed it according to weather conditions station. We all know that the weather station is not one Equipment, but it is a combination of many small tools that form a Larger system. It contains various sensors and gadgets, Work together, but communicate the appropriate Accurate data on weather parameters. It is quite tricky to uses of WEB server based weather station to non-technical peoples, so we are providing web server-based user interface as well as Android application. We are well known today most mobile units running on Android OS, and many peoples are well known to use the android phone. So, our application is beneficial for such purposes. This device is all about IoT based Live Weather data Monitoring Using NodemCU ESP8266. We will interface DHT11 Humidity & Temperature Sensor, NodeMCU ESP8266-12E wi-fi Module.

II. RELATED WORKS

The Internet of Things uses a variety of equipment, protocols, technologies, networks, middleware, Applications, systems, and data form a heterogeneous network. This will increase the degree of interoperability and complexity. Due to this situation, several groups such as ITU,ETSI and OpenIoT are developing interoperability standards and IoT protocols. However, the high degree of dispersion and development of vertical IoT systems has been increasing.

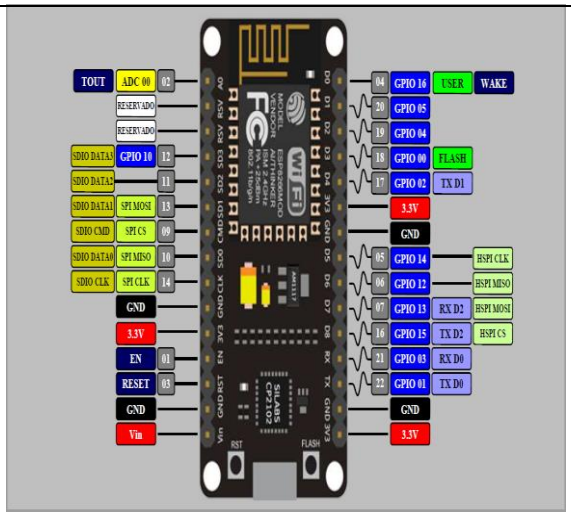
Multi-standard context, in which functions, functions and devices are combined together. The principle trouble of this examination is to exhibit and approve that microcontrollers can be associated with an information securing network with their sensors to fabricate a data set framework dependent on

the qualities of the climate. The current thought helps the microcontroller sensors to foresee conceivable outcomes fixated on the apparent information as opposed to stringently following the gadget. Monetarily, a solitary sensor known as DHT sensor is utilized by the proposed framework to give temperature and humidity furthermore, stickiness readings that were utilized to establish the unorthodox system of the environment information base This paper gives a straightforward method of distantly following/putting away the information, that is, the shopper ought to introduce the application at a particular area and begin recording and putting away the information.

III. WORKING

Gather all framework according to circuit graph. Program the NodeMCU utilizing Arduino IDE. You will get affirmation on your screen once. The NodeMCU is a programable regulator which has inbuilt wi-fi module. We interface a sensor named DHT11 to NodeMCU. By utilizing this sensor, we can gather the necessary climate information for observing reason. This pooled

information is stream over the Internet to show it or read it from anyplace. After the effectively customized equipment, the NodeMCU get one IP address. We can peruse this IP address from any of WEB program like Chrome, Firefox, Internet Explorer etc.so we show the necessary live information which got by sensors in delightful Graphical User Interface design. The climate boundaries that we screen are Temperature and Pressure. Additionally, you can check whether information through anyplace utilizing Internet as we facilitated this worker freely. We built up an android application for simple admittance to our climate checking framework.



ESP8266 NODE MUC Board

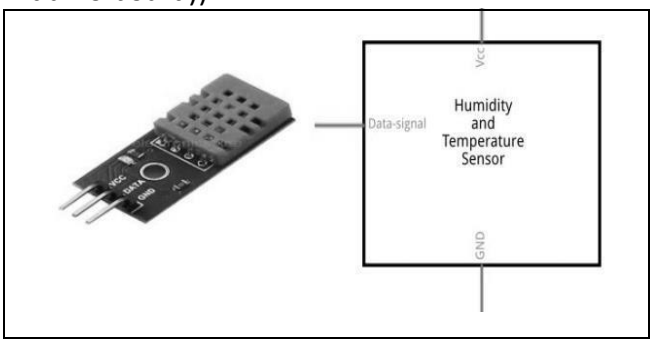
HARDWARE COMPONENTS

1. ESP8266 NODE MCU Board

A Node MCU may be a trendy microcontroller piece that had been created by Arduino Enterprises. This piece works as Arduino with necessities regardless of AVR processors [11, 12] that excite Arduino IDE C++ compilers to compile the complete packet. because of the features enforced by the second sight team, the package is taken into account to be an entire kit supposed to reduce the precise sectors that are needed to be connected to varied roles of the perform. The term Core has been given to the package cluster that is required to right the Arduino C++ headers mistreatment MCU language. The innovation of the ESP8266 module contributes to the creation of strong and complete systems as critical the look methodology that developed the Arduino core beneath the form of government of the ESP8266 Wi-Fi supported GitHub ESP8266 core website. This module may be a platform for machine learning, incorporating between ESP8266 and NodeMCU. The unit shown in Figure one operates beneath the management of networks 802.11n and 802.11b. this implies it is used as Associate in Nursing access purpose AP and Wi-Fi system or each along.

2. Temperature and Humidity Sensor (DHT)

The temperature/humidity sensor (DHT) shown in the figure periodically measures the values of (T) and (H) according to the update cycle, and adjusts the digital signal output. Due to the temperature/humidity sensing module based on proprietary digital signals and data, the sensor is considered to be reliable and stable. The DHT sensor is composed of a resistive element that reads humidity and a negative temperature coefficient NTC element that reads temperature. The sensor uses 8-bit microcontrollers for processing, these microcontrollers have reliability, sensitivity, stability, high responsiveness, no interference, and the final cost is low. The temperature/humidity sensor (DHT) used is provided by 3 pins (VCC connected to 5V of Arduino, GND (connected to Arduino GND) and DATA (connected to digital pins of Arduino board)).



DHT SENSOR MODULE

3. LCD 16 × 2 display module

The proposed liquid crystal LCD display is considered to be the focus of this work, and it regularly displays several data simultaneously on 16 columns and 2 rows. The main function of the proposed LCD display is to display the information reported by the sensors used as several attributes, so as to clarify the system conditions regularly. According to the connection scheme shown in Figure 3 written by the Fritzing program, it is recommended to interface with the Arduino board.

4. JUMP WIRES:-

A hop wire (otherwise called jumper, jumper wire, jumper link, DuPont wire or link) is an electrical wire, or gathering of them in a link, with a connector or pin at each end (or at times without them – essentially "tinned"), which is regularly used to interconnect the parts of a breadboard or other model or test circuit, inside or with other gear or segments, without binding.

SOFTWARE TOOLS

1.Google Firebase:

Firebase

Firebase is a toolset to “build, improve, and grow your app”, and the tools it gives you cover a large portion of the services that developers would normally have to build themselves, but don’t really want to build, because they’d rather be focusing on the app experience itself. This includes things like analytics, authentication, databases, configuration, file storage, push messaging, and the list goes on. The services are hosted in the cloud, and scale with little to no effort on the part of the developer.

2.Android Studio:

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

3.Arduino IDE:

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

METHODOLOGY

WORKING ON ARDUINO

A. ALGORITHMS

```
#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#include <DHT.h>

#define FIREBASE_HOST "esp8266-fd0a4-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "Auvh3ybpHhZuRJSr71i8XLAeqStYOMzKSqlkjS8Y"

#define WIFI_SSID "OnePlus"
#define WIFI_PASSWORD "12345678"

#define DHTPIN D2                                   // Digital pin connected to DHT11
#define DHTTYPE DHT11                               // Initialize dht type as DHT 11
DHT dht(DHTPIN, DHTTYPE);

void setup()
{
  Serial.begin(115200);
  dht.begin();                                        //reads dht sensor data

  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to ");
  Serial.print(WIFI_SSID);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println();
  Serial.print("Connected");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);

}

void loop()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t))
  {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  Serial.print("Humidity: ");
  Serial.print(h);
  String fireHumid = String(h) + String("%");

  Serial.print("%  Temperature: ");
  Serial.print(t);
  Serial.println("°C ");
  String fireTemp = String(t) + String("°C");
  delay(5000);


  Firebase.pushString("/DHT11/Humidity", fireHumid);
Firebase.pushString("/DHT11/Temperature", fireTemp);
 if (Firebase.failed())
  {

    Serial.print("Send data to FIREBASE failed:");
    Serial.println(Firebase.error());
    return;
  }
}
```

B. CODE EXPLANATION

Libraries:-

```
#include <ESP8266WiFi.h>
#include <FirebaseArduino.h>
#include <DHT.h>
```

Firebase:-

```
#define FIREBASE_HOST "esp8266-fd0a4-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "Auvh3ybpHhZuRJSr7li8XLAeqStYOMzKSqlkjS8Y"
```

Wifi:-

```
#define WIFI_SSID "OnePlus"
#define WIFI_PASSWORD "12345678"
```

DHT11 Sensor:-

```
#define DHTPIN D2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
```

Connecting with wifi:-

```
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to ");
  Serial.print(WIFI_SSID);
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println();
  Serial.print("Connected");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
```

Read Data From Sensor:-

```
float h = dht.readHumidity();
float t = dht.readTemperature();
```

Check the sensor working:-

```
if (isnan(h) || isnan(t))
{
  Serial.println(F("Failed to read from DHT sensor!"));
  return;
}
```

Convert into String:-

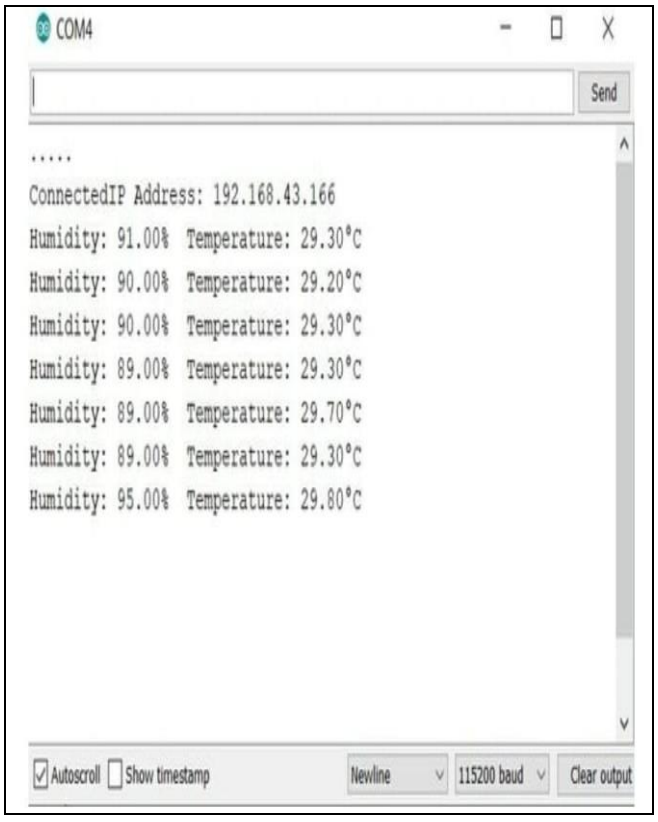
```
Serial.print("Humidity: ");
Serial.print(h);
String fireHumid = String(h) + String("%");

Serial.print("%  Temperature: ");
Serial.print(t);
Serial.println("°C ");
String fireTemp = String(t) + String("°C");
delay(5000);
```

Sending data to Firebase:-

```
Firebase.pushString("/DHT11/Humidity", fireHumid);
Firebase.pushString("/DHT11/Temperature", fireTemp);
```

C.OUTPUTS:-

Serial Monitor



Firebase Database

WORKING ON JS REACT FOR ANDROID APPLICATION

A. ALGORITHMS

Weather Application Code:-

```
import React, { useState } from "react";
import "./css/style.css";
import GlobelTemp from "./GlobelTemp";
import Local from "./Local";



const Weatherapp = () => {

   const [active,setActive] = useState("Globel");

    return(
        <>

            { active === "Globel" && <GlobelTemp/> }
            { active === "Local" && <Local/> }
            <button className="btn1" onClick={ ()=> setActive("Globel")}>Globel</button>
            <button className="btn2" onClick={ ()=> setActive("Local")}>Local</button>

       </>
    );
}


export default Weatherapp;
```

MEASURING LOCAL DATA:-

```
import React, { useEffect, useState } from "react";
import "./css/style.css";
import Animation from "./Animation";
import DateAndTime from "./DateAndTime";
import firebase from "firebase";
import firebaseConfig from "./FirebaseConfig";


export default function Local()  {

    const [humList,setHumList] = useState();
    const [tempList,setTempList] = useState();

    useEffect(()=>{
        const database = firebase.database().ref().child('hum');
        database.on("value", datasnap1=>{
            const hum = datasnap1.val();
            const humList = [];
            for(let id1 in hum){
                humList.push(hum[id1]);
            }
            console.log(humList);
            setHumList(humList);

        });

        const database1 = firebase.database().ref().child('temp');
        database1.on("value", datasnap2=>{
            const temp = datasnap2.val();
            const tempList = [];
            for(let id2 in temp){
                tempList.push(temp[id2]);
            }
            console.log(tempList);
            setTempList(tempList);
        });
    },[]) ;
```

```
    return(
        <>
            <div className="box">
                <div className="info">
                    <h2 className="location">Local Temp</h2>
                    <div className="localdiv">
                        <h1 className="localtemp">temp: {tempList}°C</h1><br/>
                        <h1 className="localtemp">humidity: {humList}%</h1>
                    </div>
                    <DateAndTime/>
                </div>
                <Animation />
            </div>
        </>
    );
```

MEASURING GLOBAL TEMPERATURE:-

```
import React, { useEffect,useState } from "react";
import "./css/style.css";
import Animation from "./Animation";
import DateAndTime from "./DateAndTime";


const Local = () => {

    const [city,setCity] = useState("surat");
    const [search,setSearch] = useState("surat");
    const [js,setJS] = useState(city);
    const [more,setMore] = useState("nomore");

    useEffect( () => {
        const fetchApi = async () => {
            const url = `http://api.openweathermap.org/data/2.5/weather?q=${search}&units=metric&appid=e106a83ca0cec4c2d2e72c1fb8ba8000`;
            const response = await fetch(url);
            //console.log(response);
            const resJson = await response.json();
            setCity(resJson.main);
            setJS(resJson);
            console.log(resJson);
        }
        fetchApi();
    },[search] )


    return(
```

```
<>
{
     more === "nomore" && (
        <div className="box">
        <div className="inputData">
            <input
            type="search"
            value={search}
            className="inputFeild"
            onChange = { (event) => { setSearch(event.target.value) }

            }
            />
        </div>

        {
    !city ? ( <h2 className="errorMsg">No data found</h2> ) : (

        <div>
            <div className="info">
                <h2 className="location">
                <i className="fas fa-thumbtack"></i> {search}
                </h2>
                <h1 className="temp">{city.temp}°C</h1>
                <h3 className="tempmin_max">Min: {city.temp_min}°C | max: {city.temp_max}°C</h3>
                <DateAndTime />
                <button className="btn123"  onClick={ ()=> setMore("more")}>More</button>
            </div>

        </div>

    )
}
<Animation />
    </div>
```

```
        )
    }
    {
         more === "more" && (
            <div className="box1">


            {
        !city ? ( <h2 className="errorMsg">No data found</h2> ) : (

            <div>
                <div className="info">
                    <h2 className="location">
                    <i className="fas fa-thumbtack"></i> {search},({js.sys.country})
                    </h2>
                    <h1 className="temp1">{city.temp}°C</h1>
                    <h3 className="tempmin_max">
                        Min : {city.temp_min} °C <br />
                        Max : {city.temp_max} °C <br />
                        Humidity : {city.humidity} <br />
                        Air Pressure : {city.pressure} hPa <br />
                        Visibility : {(js.visibility)/1000} km <br />
                        Wind : {js.wind.speed} km/h <br />
                    </h3>
                    <DateAndTime />
                    <button className="btn123"  onClick={ ()=> setMore("nomore")}>Back</button>
                </div>

            </div>

        )
    }
    <Animation />
        </div>

        )
```

ANIMATIONS :-

```
import React from "react";


function Animation (){
    return (
        <>
            <div className="wave -one"></div>
            <div className="wave -two"></div>
            <div className="wave -three"></div>
        </>
    )
}

export default Animation;
```

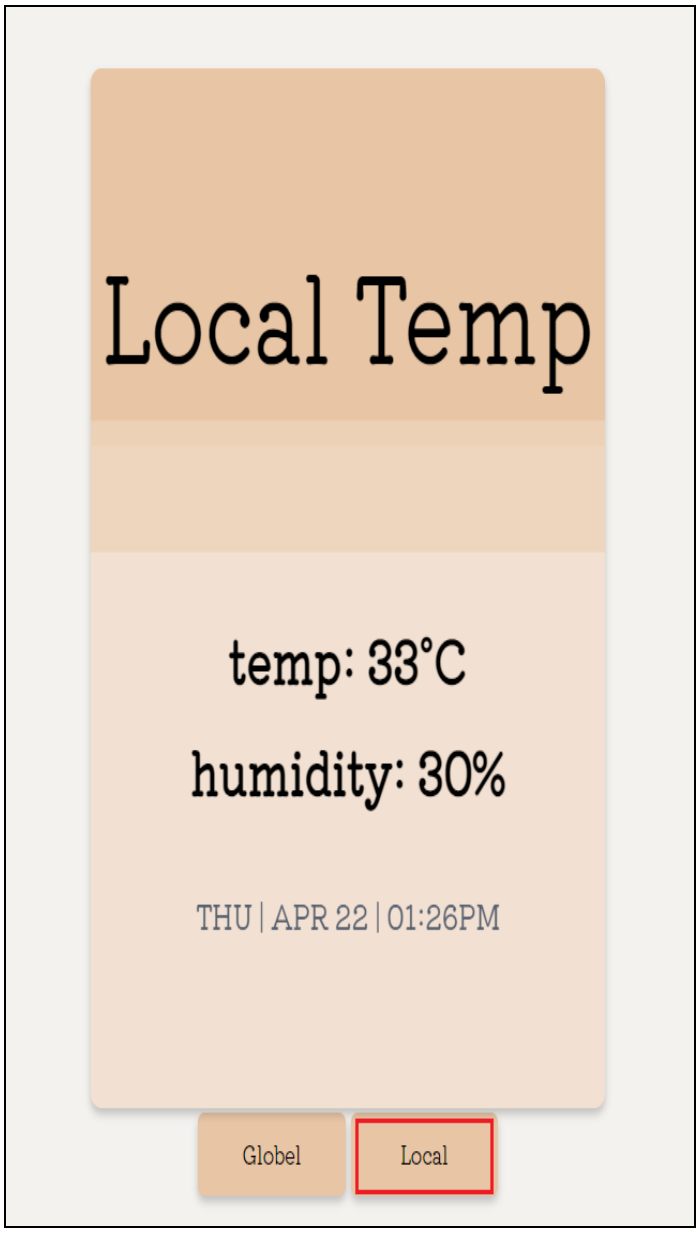
STYLES USING CSS:-

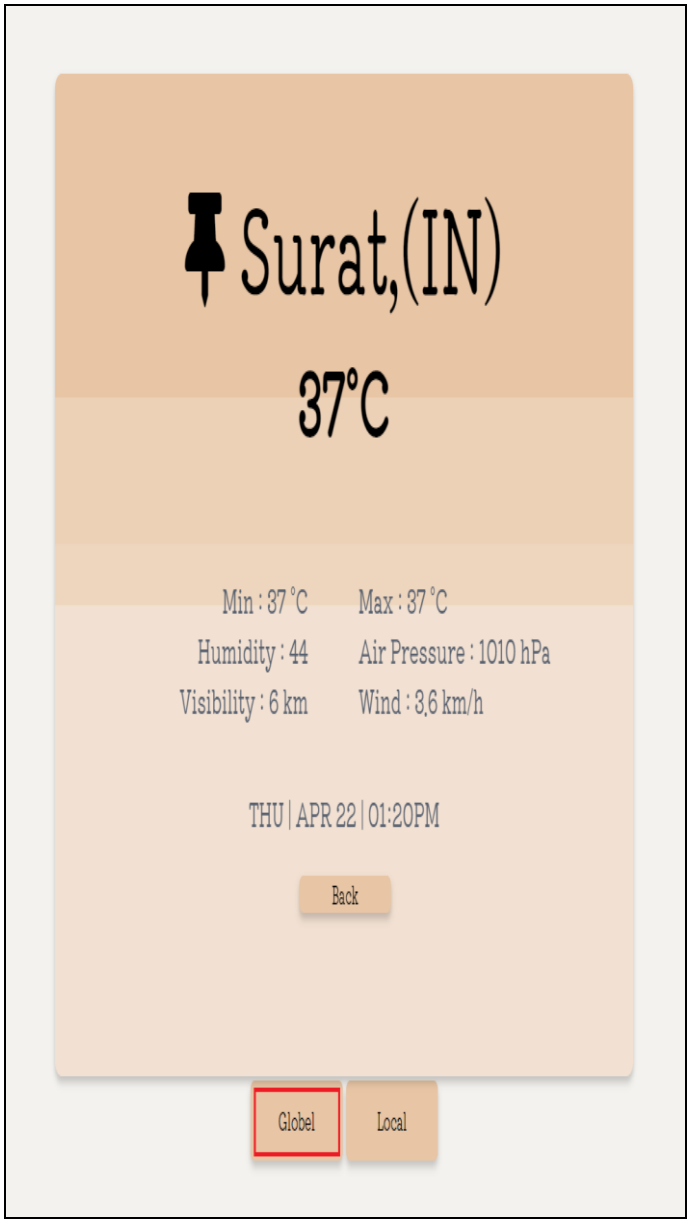
https://docs.google.com/document/d/1jXb1cAo1s0tsMVbsBFZnpcFtLuWyyd_sWPA1GxSp17o/edit

B. OUTPUT

GLOBAL TEMPERATURE:-

LOCAL TEMPERATURE:-





CONCLUSION:-

This paper affords an progressive and reliable thought of a affordable easy weather monitoring and controlling system.
The gadget operates beneath IoT science supervision which correctly optimizes faraway areas. The creativity of this progressive climate station allows monitoring and controlling of the net server-based local weather prerequisites the usage of the ESP8266 node MCU microcontroller. The outputs of the measurements employed are intended to be proven by using our Andriod Application.
In phrases of community connectivity, the devices can be became ON or OFF at any second and anywhere.
The whole dependence on the webserver manipulate gadget and the applicability of the neighborhood IP given by way of the ESP8266 means that the design's price is inexpensive. The device contributes to being relevant in two fields.

The first contribution is extraordinarily beneficial to organizations and different agencies that are tasked with getting ready and coping with their operations based totally on climate situations; such as high-priority transport systems, airways, and forestry, etc. The 2nd contribution is particularly designedto manage place concerning the modifications in person interface reputation based totally on information generated by using enhancements in output due to climate disturbances; such as monitoring residences, stores, hospitals, universities, and clever vehicles.

REFRENCES:-

[1] Dhanalaxmi, B., & Naidu, G. A. (2017, February). A survey on design and analysis of robust IoT architecture. In 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA) (pp. 375-378). IEEE

. [2] Krishnamurthi, K., Thapa, S., Kothari, L., & Prakash, A. (2015). Arduino based weather monitoring system. International Journal of Engineering Research and General Science, 3(2), 452-458.

[3] Sabharwal, N., Kumar, R., Thakur, A., & Sharma, J. (2014). A Low-Cost Zigbee Basedautomatic Wireless Weather Station With Gui And Web Hosting Facility. International Journal of Electrical and Electronics Engineering, 1.

[4] Mahmood, S. N., & Hasan, F. F. (2017). Design of weather monitoring system using Arduino based database implementation. Journal of Multidisciplinary Engineering Science and Technology (JMEST), 4(4), 7109

[5] Yacchirema, D. C., Palau, C. E., & Esteve, M. (2017, January). Enable IoT interoperability in ambient assisted living: Active and healthy ageing scenarios. In 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC) (pp. 53-58). IEEE.

[6] Pan, J., Jain, R., Paul, S., Vu, T., Saifullah, A., & Sha, M. (2015). An internet of things framework for smart energy in buildings: designs, prototype, and experiments. IEEE Internet of Things Journal, 2(6), 527-537.

Links:

Blog:
https://charusatsgpsem-4.blogspot.com/2021/03/weather-with-react-js-hello-everyone.html

Github Code:
https://github.com/akshaymalaviya/weather-app-in-ReactJS.git

ABOUT AUTHORS

AKSHAY MALAVIYA
STUDENT
INFORMATION TECHNOLOGY
CSPIT
CHANGA,ANAND
GUJARAT,INDIA.
19it061@charusat.edu.in

KEVAL MANDANKA
STUDENT
INFORMATION TECHNOLOGY
CSPIT
CHANGA,ANAND
GUJARAT,INDIA.
19it063@charusat.edu.in

TURNITIN PLAGARISM REPORT

Design of Weather Monitoring System Using IOT and Android Application

ORIGINALITY REPORT

37%	22%	13%	28%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	7%
2	Submitted to Charotar University of Science And Technology Student Paper	5%
3	www.jmest.org Internet Source	5%
4	Pablo Palacios, Andres Cordova. "Approximation and Temperature Control System via an Actuator and a Cloud: An Application Based on the IoT for Smart Houses", 2018 International Conference on eDemocracy & eGovernment (ICEDEG), 2018 Publication	4%
5	Submitted to Softwarica College Of IT & E-Commerce Student Paper	3%
6	en.wikipedia.org Internet Source	3%

7	www.readbag.com Internet Source	3%
8	Submitted to Universiti Tunku Abdul Rahman Student Paper	2%
9	Medilla Kusriyanto, Agusti Anggara Putra. "Weather Station Design Using IoT Platform Based On Arduino Mega", 2018 International Symposium on Electronics and Smart Devices (ISESD), 2018 Publication	2%
10	Submitted to K. J. Somaiya College of Engineering Vidyavihar, Mumbai Student Paper	2%
11	Submitted to National Institute Of Technology, Tiruchirappalli Student Paper	1%
12	Vivek Hareshbhai Puar, Chintan M. Bhatt, Duong Minh Hoang, Dac-Nhuong Le. "Chapter 28 Communication in Internet of Things", Springer Science and Business Media LLC, 2018 Publication	1%