ENVIRONMENTALMONITORINGIN

PARKS

IOT-PHASE-4

**TEAMMEMBERS:**

KRISHNAPRASATH (71772114303)

SABARIRAJA

(71772114134)

SIVANANDHAM

(71772114310)

# INTRODUCTION

Environmental monitoring in parks using IoT (Internet of Things) is a modernapproach to assess and manage the natural surroundings within these recreational areas. Bydeploying a network of sensors and connected devices, it enables real-time data collectionand analysis. This technology helps track variables like air quality, temperature, humidity,waterquality,andwildlifeactivity.ThesummarizedintroductionunderscoresthesignificanceofIoTinmaintainingandsafeguardingtheecologicalbalanceof ourparks,offeringbetter

insightsandinformeddecision-makingforparkmanagementandpreservation.

# USINGWEBDEVELOPMENTCREATEAPLATFORMTHATDISPLAYSENVIRONMENTALMONITORINGINPARKSDATA:

**HTMLCODE:**

<!DOCTYPEhtml>

<html>

<head>

<linkrel="stylesheet"type="text/css"href="styles.css">

</head>

<body>

<divclass="container">

<h1>EnvironmentalMonitoringinParks</h1>

<divclass="data">

<h2>Temperature</h2>

<pid="temperature">28°C(82.4°F)</p>

</div>

<divclass="data">

<h2>Humidity</h2>

<pid="humidity">30%</p>

</div>

</div>

<scriptsrc="script.js"></script>

</body>

</html>

# CSSCODE:

body{ font-family: Arial,sans-serif; background-color: #f0f0f0;

}

.container{ text-align:center;

background-color:#fff; border-radius:10px; padding:5px; margin: 10px;box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);height:500px; width:350px;

}

h1{ color:#333;

}

.data{

padding:20px;margin:30px;

border: 1px solid#ddd; border-radius: 10px;background-color:rgb(204,255,255);

box-shadow:005px

rgba(0,0,0,0.1);

height:100px;width:250px;

}

p{ color:

#555;

}

h2{ font-size:

24px; font-weight:bold;

}

# JAVASCRIPT:

// Function to update temperature and humidity data functionupdateData(temperature,humidity){ const temperatureElement =document.getElementById("temperature"); consthumidityElement=document.getElementById("humidity");

temperatureElement.textContent=${temperature}°C;humidityElement.textContent=${humidity}%;

}

//FunctiontoconnecttoMQTTbrokerandsubscribetotopicsfunctionconnectToMQTT(){

// Replace the following with your MQTT broker detailsconstbrokerUrl="mqtt://your-mqtt-broker"; consttopicTemperature="environment/temperature"; consttopicHumidity= "environment/humidity";

constclient=newPaho.MQTT.Client(brokerUrl,"web-client-"+parseInt(Math.random()\*1e9,10));

//Setupthecallbackforasuccessfulconnection

client.onConnectionLost=onConnectionLost; client.onMessageArrived

=onMessageArrived;

client.connect({onSuccess: onConnect,onFailure:onFailure,

});

//CallbackforsuccessfulconnectionfunctiononConnect(){

console.log("ConnectedtoMQTTbroker");client.subscribe(topicTemperature);

client.subscribe(topicHumidity);

}

//Callbackforconnectionfailure function onFailure(responseObject) {console.log("FailedtoconnecttoMQTTbroker:"+responseObject.errorMessage);

}

//Callbackforconnectionloss function

onConnectionLost(responseObject){ if(responseObject.errorCode

!==0) { console.log("Connectionlost:"+responseObject.errorMessage);

}

}

//CallbackforreceivedMQTTmessages function

onMessageArrived(message){ if(message.topic===topicTemperature)

{ const temperature = parseFloat(message.payloadString).toFixed(2);updateData(temperature,parseFloat(humidityElement.textContent));

}elseif(message.topic===topicHumidity){ consthumidity=parseFloat(message.payloadString).toFixed(2);updateData(parseFloat(temperatureElement.textContent),humidity);

}

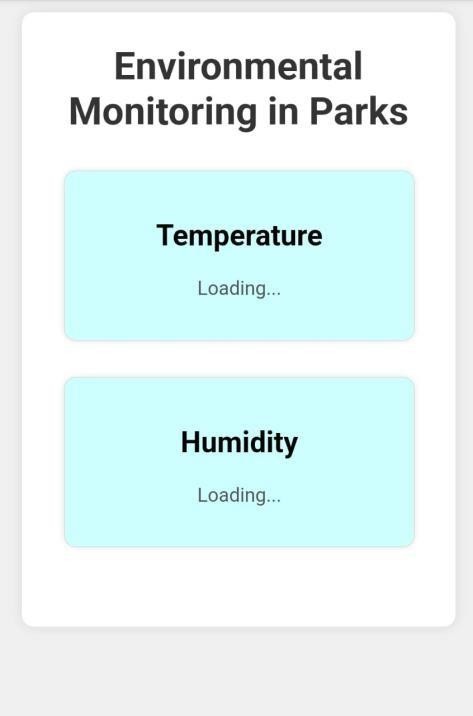
}

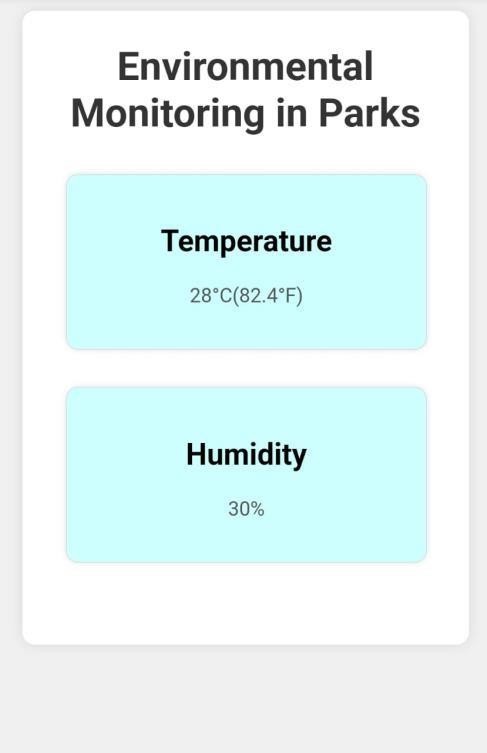
}

//CalltheconnectToMQTTfunctiontoestablishMQTTconnectionconnectToMQTT();

# DESIGNTHEPLATFORMTORECEIVEANDDISPLAYENVIRONMENTALMONITORINGINPARKSDATA

**SENTBYTHEIOTDEVICES**





SensorNodes:Deploysensornodesthroughouttheparktomeasuretemperatureandhumidity.Select appropriate IoT sensors, such as DHT22 or BME280, to capture thedata.Ensurepower sources (e.g.,batteriesor solarpanels)for thenodes.

Communication:Use a communication protocol, such as MQTT or LoRaWAN, totransmitsensordatatoacentralserver.Ensurenetworkconnectivitywithinthepark.Central

Server:Setupacentralservertocollectandstoredata.Implementdatavalidationanderrorhandlingforreliability.Utilizeadatabase(e.g.,MySQLorNoSQL)tostorehistoricaldata.

User Interface:Create a web-based or mobile application for users to access thedata.Designanintuitiveuserinterfacetodisplaytemperatureandhumiditydata.Implementfeatureslikereal-time updates and historicaldataretrieval.

AlertsandNotifications:Implementalertingmechanismsforextremeenvironmentalconditions(e.g.,heatwavesorhighhumidity).Sendnotificationstousersorparkauthoritiesthrough email,SMS,orappnotifications.

Data Analysis:Incorporate data analytics to identify trends or anomalies in theenvironmentaldata.UsetoolslikePythonanddatavisualizationlibrariestocreategraphsand charts.

Security:Ensuredataencryptionandsecureauthenticationtoprotectthesystemfrom unauthorized access.

Scalability:Designtheplatformtoeasilyaddmoresensorsorexpandtocoverlargerareasinthepark.

PowerManagement:Optimizepowerusageinsensornodestoextendbatterylifeorreduce theneed forfrequentmaintenance.

MaintenanceandSupport:Establishamaintenanceplanforregularchecksandsensorcalibration.Providesupportforusersandaddressissuespromptly.

Documentation:Createcomprehensivedocumentationforsetup,maintenance,andtroubleshooting.

RegulatoryCompliance:Ensurecompliancewithlocalregulationsanddataprivacylawswhen handling sensordata.

VisualizationandReporting:Enablethegenerationofreportsforparkmanagementorresearchpurposes.

CommunityEngagement:Considerinvolvingthelocalcommunityorparkvisitorsinthe project,promoting awarenessandengagement.

# Conclusion:

This project demonstrates how to utilize the DHT11 Temperature and HumiditySensorincombinationwithNodeMCUtosendreal-timetemperatureandhumiditydatatotheThingSpeakIoTserver.Bydoingso,itenablesremotemonitoringof

environmentalconditionsovertheinternet,allowinguserstoaccessandvisualizethis

dataontheThingSpeakdashboard.ThisprojectprovidesapracticalexampleofhowIoTtechnologycanbeemployedforglobalenvironmentaldatatrackingand

analysis.