**PHASE – 4: DEVELOPMENT PART 2**

**INTRODUCTION:**

It sounds like you've implemented an IoT (Internet of Things) project using a Raspberry Pi and a custom web application to monitor and display water tank levels. This is a common and practical use of IoT technology. Here's a breakdown of the components and processes involved:

**1. Hardware Setup:**

- You have an ultrasonic sensor mounted on top of a water tank.

- The ultrasonic sensor is connected to a Raspberry Pi.

**2. Data Collection:**

- The Raspberry Pi collects water level data from the ultrasonic sensor.

- This data is gathered at regular intervals, in your case, every minute.

**3. Database Integration:**

- The collected water level data is sent to a remote database.

- This database stores the historical data of water level readings.

**4. Web Application:**

- You've developed a web application that is hosted on a remote server.

- The web application fetches data from the remote database to update its user interface.

**5. User Interface:**

- The web application provides a user-friendly interface.

- It displays an animation representing the water tank and the current water level.

- It also includes a thin bar line graph that likely shows the historical trends of water level over time.

**6. Data Visualization:**

- The web application uses the collected data to update the water level animation and the thin bar line graph.

This system provides a real-time and historical view of water levels in the tank, allowing users to monitor and manage their water resources effectively. It's a practical example of how IoT technology can be used for remote monitoring and data visualization. If you have any specific questions or need assistance with any aspect of your project, please feel free to ask!

**DEVELOPMENT CODE WITH EXPLANATION:**

<!DOCTYPE html>

<html lang="en">

<head>

<title>Smart Water Tank</title>

<style>

.outer {

border: 1px solid grey;

margin: 0%;

width: 100%;

float: left;

}

</style>

</head>

<body>

<div align='center'>

<h3>Real Time Tank Level Monitoring</h3>

<p style='font-size: 12px; color: grey'>

The dashboard displays real-time water level in a tank. Tank level is updated every minute on arrival of sensor data from Raspberry Pi.

<a href='tank\_simulation' target='\_blank'>Simulation</a>

</p>

</div>

<div class='outer'>

<div style='width: 20%; float: left; margin-left: 20%;'>

<iframe height='410px' width='100%' src='basic\_page/' name='iframe\_a'>

</iframe>

</div>

<div style='width: 30%; float: left; margin-left: 5%;'>

<iframe height='410px' width='100%' src='tank\_animation/' name='iframe\_a'>

</iframe>

</div>

</div>

<div class='outer'>

<div style='width: 70%; margin-left: 20%; float: left'>

<iframe height='420px' width='100%' src='graph/' name='iframe\_a'></iframe>

<p style='margin: 1%; color: grey; font-size: 12px'>

The graph shows the water level readings for a selected date. Y-axis indicates the volume of water in the tank, and X-axis denotes the hour of the day.

</p>

</div>

</div>

</body>

</html>

**INSET-DATA-PHP code:**

The PHP code starts by setting the default time zone to "Asia/Kolkata" to ensure that date and time calculations are consistent with the specified time zone.

It includes an external file named 'util.php,' which likely contains utility functions related to database operations. The purpose of this file is to encapsulate commonly used functions for better code organization and reusability.

The code retrieves the 'dist' parameter from the HTTP request using the $\_GET super global. This parameter appears to represent a distance value.

The code then calls the insert\_in\_db function, passing the retrieved 'val' parameter to this function for database insertion.

The insert\_in\_db function is defined to insert data into a database table. It performs the following steps:

Displays a message indicating that a value is being inserted into the database, showing the 'val' parameter.

Creates a database connection using a function create\_db\_connection() from the 'util.php' file.

Gets the current timestamp using the time() function and formats it as a human-readable date and time string.

Displays the formatted date and time.

Defines an SQL query to insert data into the 'level\_log' table, including the 'val,' timestamp, and formatted date and time.

Executes the SQL query and stores the result in the $result variable.

Checks the result of the query execution. If unsuccessful, it displays an error message and returns 0 (although the return statements are commented out and not executed).

If the query execution is successful, it displays a success message and returns 1 (commented out).

Finally, it closes the database connection using a function close\_db\_connection() from the 'util.php' file.

The purpose of this code is to insert water level data into a database, recording the level, timestamp, and human-readable date and time. It provides diagnostic messages and error handling for database operations.

<?php

// Change the time zone as per your country

date\_default\_timezone\_set("Asia/Kolkata");

// Include the 'util.php' file, which likely contains functions for database operations

include\_once 'util.php';

// Get the 'dist' parameter from the HTTP request

$val = $\_GET["dist"];

// Call the 'insert\_in\_db' function and pass the 'val' parameter to it

insert\_in\_db($val);

// 'insert\_in\_db' function definition

function insert\_in\_db($val) {

// Display a message indicating that a value is being inserted into the database

echo "<p>Inserting value <b>$val</b> in the database</p>";

// Create a database connection

$connection = create\_db\_connection();

// Get the current timestamp

$t = time();

// Format the timestamp into a human-readable date and time format

$date\_time = date('Y-m-d H:i:s', $t);

// Display the formatted date and time

echo "<p>$date\_time</p>";

// Define an SQL query to insert data into the 'level\_log' table

$sql = "INSERT INTO `level\_log`(`level`, `timestamp`, `date\_time`)

VALUES ('$val', '$t', '$date\_time')";

// Execute the SQL query and store the result

$result = mysqli\_query($connection, $sql);

// Check if the query execution was unsuccessful

if (!$result) {

die(mysql\_error() . "\n" . $sql);

echo " !!!<br> ";

// Return 0; (This is commented out and not executed)

}

// Check if the query execution was successful

if ($result) {

echo " <p>Successfully inserted >>>> </p> ";

// Return 1; (This is commented out and not executed)

}

// Close the database connection

close\_db\_connection($connection);

}

?>

**UTILE.PHP:**

<?php

// Define global database connection parameters

global $db\_server, $db\_username, $db\_pwd, $db\_name;

$db\_server = 'localhost';

$db\_username = 'root';

$db\_pwd = '';

$db\_name = 'water\_level';

// Define global tank dimensions

global $dia, $height;

$dia = 104; // Diameter of the tank in centimeters

$height = 116; // Height of the tank in centimeters

// Utility functions for calculating volume, time ago, and database operations

function calculate\_volume($dia, $water\_level) { /\* ... \*/ }

function time\_ago($timestamp) { /\* ... \*/ }

function fetch\_arr($result) { /\* ... \*/ }

function create\_db\_connection() { /\* ... \*/ }

function close\_db\_connection($connection) { /\* ... \*/ }

function db\_select\_all($connection, $sql) { /\* ... \*/ }

function db\_select\_row($connection, $sql) { /\* ... \*/ }

?>

The code snippet begins by defining global variables for database connection parameters and tank dimensions. The database parameters include the server, username, password, and database name. The tank dimensions represent the diameter and height of the water tank.

**BASE PAGE INDEX.HTML:**

<!doctype html>

<html lang="en">

<head>

<script type="text/javascript" src='jquery.min.js'></script>

<script>

function data\_request\_timer(){

window.setInterval(get\_data, 2000); //timer for running get\_data() function every 2 seconds

}

function get\_data(){

console.log("fetching data from server");

$.get("read\_data1.php",

function(data, status){

document.getElementById("info").innerHTML = data;

});

}

</script>

</head>

<body onload="data\_request\_timer()">

<!-- This is where the data is displayed -->

<div style='border:0px solid blue; padding:0%;'>

<p id='info'></p>

</div>

<!-- Dummy value insertion link -->

<button>

<a style='text-decoration:none;font-size:12px' href='../insert\_data.php?dist=65' target='\_blank'>Insert dummy value</a>

</button>

</body>

</html>

**DATA READ FUNCTION USING PHP:**

<?php

ini\_set('display\_errors', '1');

//----------database access----------------------------

include\_once '../util.php';

//-----------------------------------------------------

$con=create\_db\_connection();

/\* select the latest entry in the table \*/

$sql="SELECT \* FROM `level\_log` WHERE 1 order by `timestamp` desc limit 1";

$row=db\_select\_row($con,$sql);

close\_db\_connection($con);

$x\_cm=floatval($row['level']); //this is the sensor reading i.e. the distance of empty part of tank

global $dia, $height; //these are defined in 'util.php' file

$water\_level= $height-$x\_cm;

$water\_level= round($water\_level,1);

//$x\_cm=round($x\_cm,2);

//$x\_inches=$x\_cm \* 0.393701;

$timestamp=$row['timestamp'];

$currentTime = time();

$diff=$currentTime-$timestamp; //how old the entry is

$seconds\_left=60-$diff; //seconds left for the next value to arrive

if($diff<5){

$style="background-color:lightgreen";

$msg="<h2>Data Arrived !!!!</h2>";

$img="<img src='data.gif' alt='Data Arrived !!!' width='80%' height='100'>";

}

else{

$style="background-color:white;color:grey";

$msg="Data expected in:<br> <b style='color:brown;font-size:40px'>$seconds\_left</b> seconds";

$img="";

}

if($diff>60){

$str=time\_ago($timestamp);

$ago="(<txt style='color:brown'>$str</txt>)";

}

else{

$ago="";

}

if($seconds\_left < -4){

$msg="<p style='color:red;font-size:20px'>Data from sensor not available temporarily. Pl check after some time.</p>";

}

$html="<div style='$style'>

$msg $img

<br><br>

Last value: $x\_cm <br>$ago

<hr>

Water Level:<br>

<b style='color:blue'>$water\_level cm</b>

<br>

</div>

";

echo $html;

?>

**Conclusion:**

The Smart Water Tank project is a successful implementation of an IoT (Internet of Things) system that monitors and manages water levels in a tank. The project combines hardware, software, and web technology to provide real-time and historical insights into water levels.