

<https://www.upsite.com/blog/sealing-rack-need/>

MODULE 2d

Centralized Temperature & Humidity Monitoring System for Server Racks

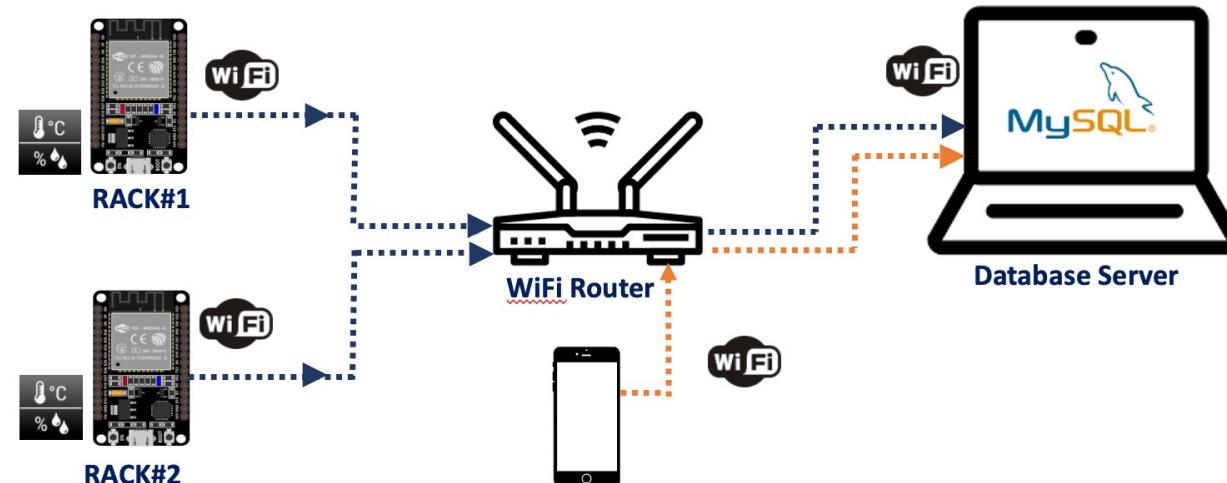
safyzan salim
019 622 0575

Project Overview

This tutorial will teach how to save data from ESP32s (or any ethernet microcontrollers) into MySQL.

Scenario:

You are required to monitor 2 server racks in a data center; i.e., **Rack#1** & **Rack#2** by using 2 ESP32s and 2 DHT11 sensors. Both readings need to be saved into a single table. The temperature & humidity of both rooms can be monitored thru webpage.



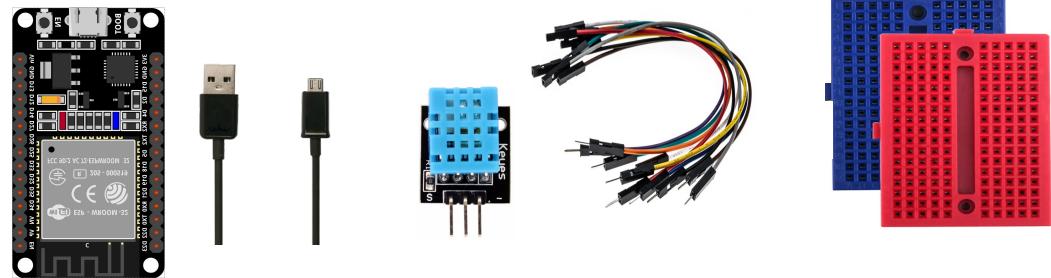
ZXZXZ

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Components

Hardware:

- 1 x ESP-32 Wifi+Bluetooth 2-In-1 Development Board for Arduino (30 pin) + Cable.
- 1 x Temperature & Humidity Sensor, DHT11.
- Jumpers (male to female).
- 2 x Mini Breadboard.



Software:

- XAMPP ver 8.0.1(PHP 8.0.1) for Windows / OS X.



What is XAMPP?

XAMPP is the most popular PHP development environment

XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.

Solution

→ **Step 1: Propose block diagram & flowcharts**

A story board of the whole project which will give clear picture what you need to do.

→ **Step 2: Install Apache Web Server**

- i. Install XAMPP (once), start web server service
- ii. Start XAMPP service
- iii. Test XAMPP page & PHPMyAdmin page

→ **Step 3: Creating MySQL Database**

- i. Create database + test
 - ii. Create a PHP file for medium of transferring information from ESP32 to MySQL + Test
- **Test with dummy data / manually inject data to database

→ **Step 4: Preparing PHP script to insert data to MySQL database**

- i. Configure *.php scripts.
- ii. PHP script to display database.
- iii. HTTP GET & HTTP POST.

Solution

→ Step 5: Upload sketch to ESP32

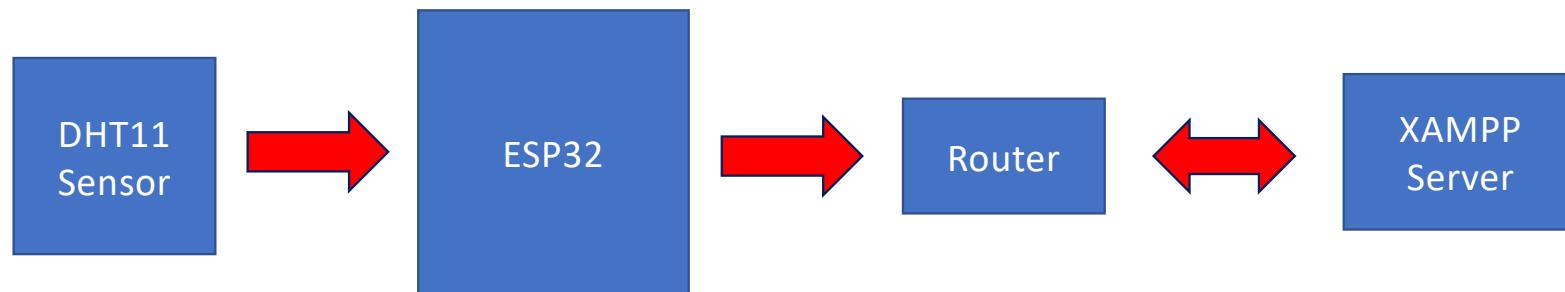
- i. Connect DHT11 to ESP32.
- ii. Configure ***sketch-vii_esp32-dht11-http-post.ino***.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 1

PROPOSED BLOCK DIAGRAM & FLOWCHART

Project Block Diagram

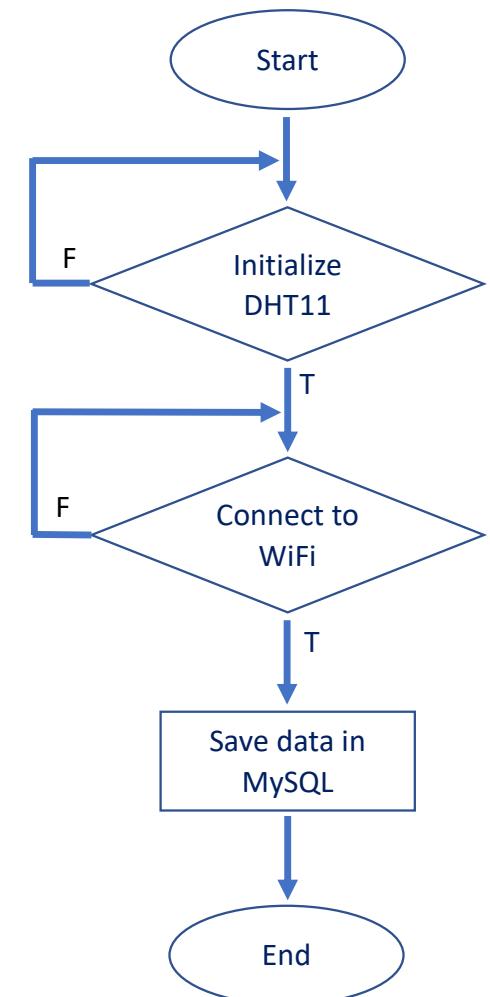
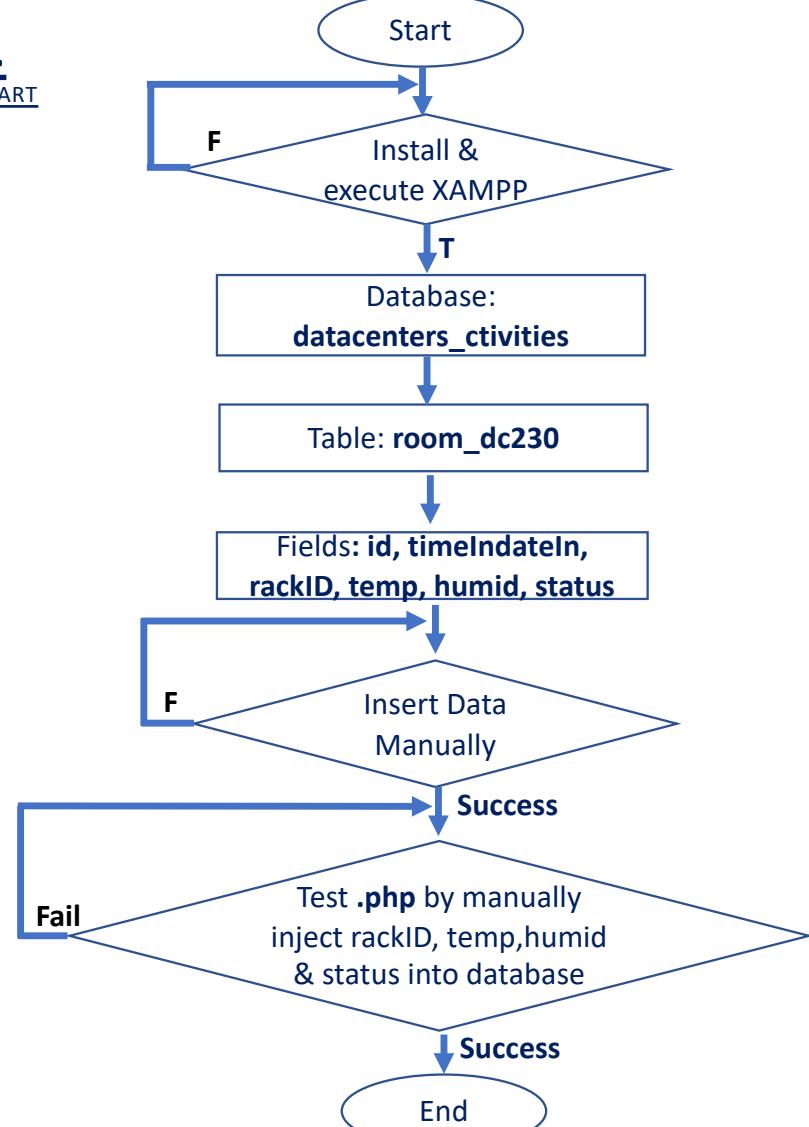


2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 1

PROPOSED BLOCK DIAGRAM & FLOWCHART

Flowchart



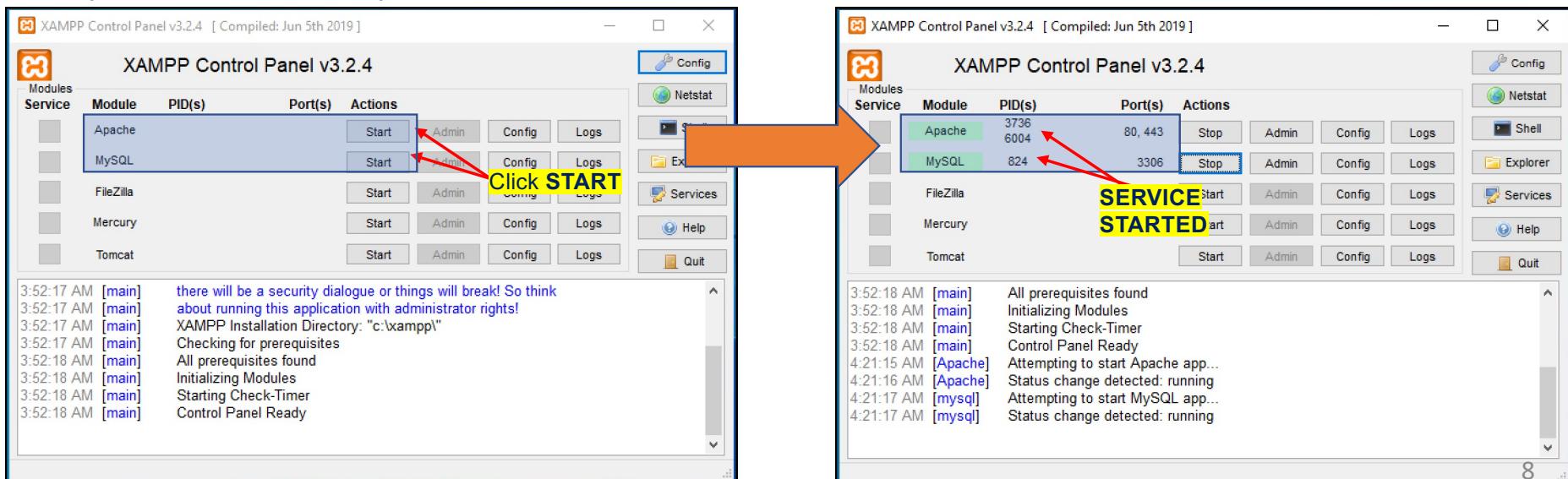
2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 2 INSTALL, START & TEST XAMPP SERVICES

- Download and install latest version of XAMPP from <https://www.apachefriends.org/download.html>
- This link shows full tutorial on installing XAMPP on windows <https://blog.templatetoaster.com/install-xampp-on-windows/>
- Once installed, launch XAMPP Control Panel (7.3.26)

For Windows (v7.3.26):

- to run, go to **c://xampp/control-control.exe** → Start **MySQL Database & Apache Web Service**
- open your browser and type <http://localhost/>

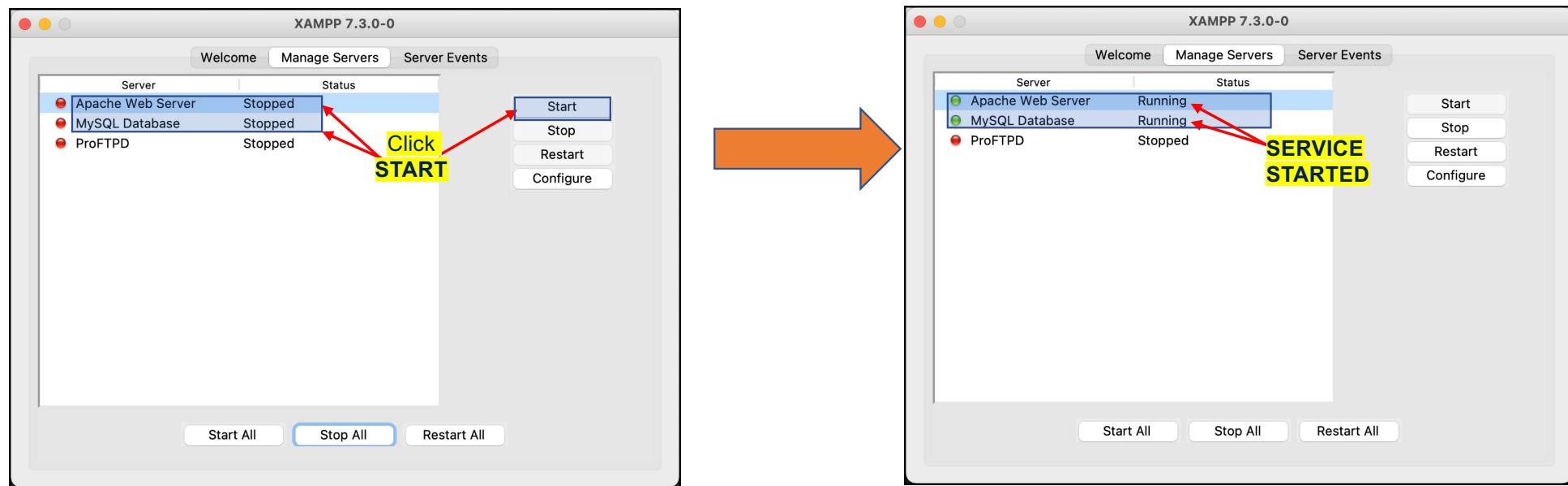


2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 2 INSTALL, START & TEST XAMPP SERVICES

For OS X (v7.3.0):

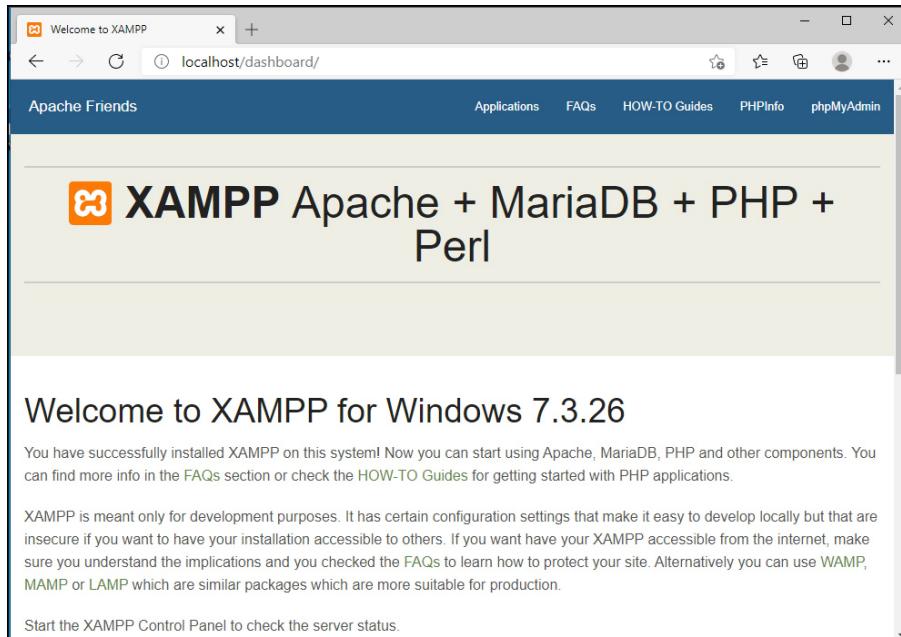
- go to **Applications>XAMPP>manager-osx** → Start **MySQL Database & Apache Web Service**
- open your browser and type <http://localhost/>



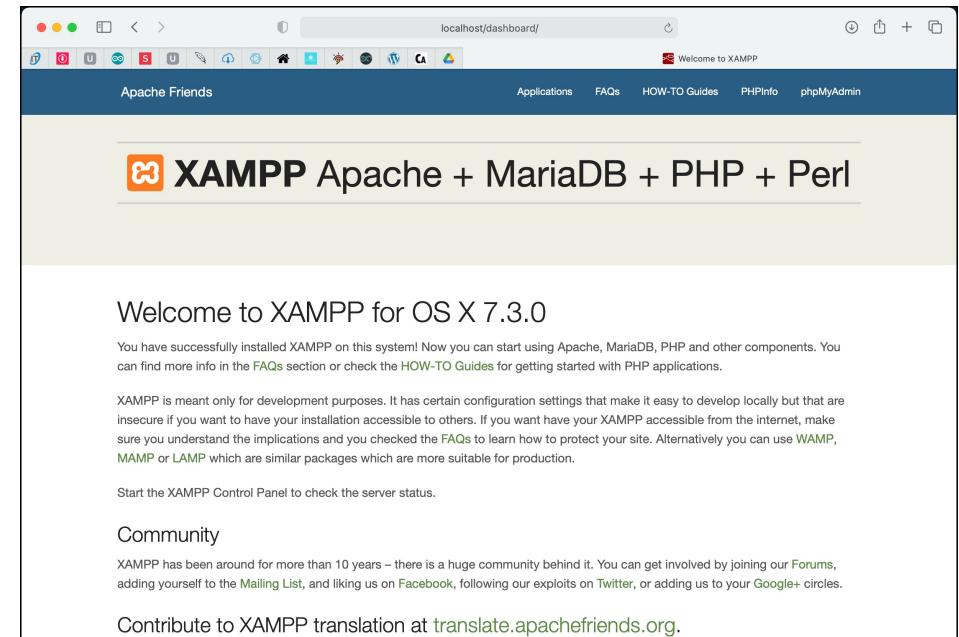
2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 2 INSTALL, START & TEST XAMPP SERVICES

→ XAMPP landing page can be found at <http://localhost/>



Windows system

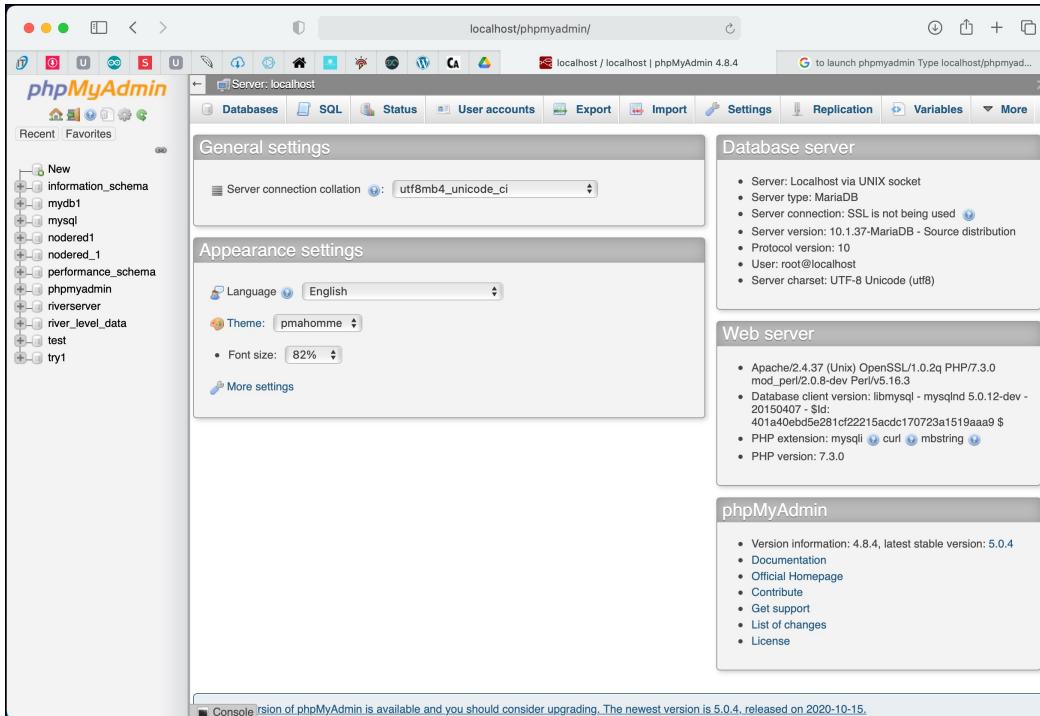


OS X system

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 2 INSTALL, START & TEST XAMPP SERVICES

- To use MySQL database, point your browser to <http://localhost/phpmyadmin>
- You can also click phpMyAdmin link at XAMPP dashboard.
- PHPMyAdmin is a third party tool to manage data inside the database.



2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Set up database with the following properties:

Database Name: **datacenter_activities**

Table Name: **room_dc230**

Columns: **6**

- The number of characters used should be equal to or less than 64.
- The name should comprise of letters, numbers and underscore.
- The DB name should not start with a number.
- It should be relevant to the topic for which it is being created.

<https://www.javatpoint.com/creating-mysql-database-with-xampp>

| NAME | TYPE | LENGTH | ADDITIONAL SETTING |
|-------------|----------|--------|---|
| id | INT | 11 | Index: PRIMARY AI: YES |
| logDateTime | DATETIME | - | Default: CURRENT_TIMESTAMP |
| rackID | VARCHAR | 255 | - |
| tempValue | VARCHAR | 255 | - |
| humidValue | VARCHAR | 255 | - |
| rackStatus | VARCHAR | 255 | - |

AI = Auto Increment

- = empty

#use appropriate name for variables (dbname, tablename...) that reflect with the task/activity

12

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Database Name: **datacenter_activities**

5

6

The screenshot shows the phpMyAdmin interface for creating a new database. A red arrow labeled '1' points from the 'New' button in the sidebar to the 'Create database' input field. Another red arrow labeled '2' points from the 'datacenters_activities' input field to the 'Create' button. A third red arrow labeled '3' points from the 'Create' button to the 'Create' button in the top right corner of the main panel. The 'Collation' dropdown is set to 'latin1_swedish_ci'. The 'Existing databases' sidebar lists several existing databases: information_schema, mydb1, mysql, nodered1, nodered_1, performance_schema, phpmyadmin, riverserver, river_level_data, test, and try1.

| Database | Collation | Action |
|--------------------|-------------------|----------------------------------|
| information_schema | utf8_general_ci | Check privileges |
| mydb1 | latin1_swedish_ci | Check privileges |
| mysql | latin1_swedish_ci | Check privileges |
| nodered1 | latin1_swedish_ci | Check privileges |
| nodered_1 | latin1_swedish_ci | Check privileges |
| performance_schema | utf8_general_ci | Check privileges |
| Console | admin | Check privileges |

13

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Table Name: **room_ds230** Number of columns: **6**

The screenshot shows the phpMyAdmin interface. The left sidebar lists databases: New, datacenters_activities (highlighted with a yellow box labeled 'Just created'), information_schema, mydb1, mysql, nodered1, nodered_1, performance_schema, phpmyadmin, riverserver, river_level_data, test, and try1. The main area shows the 'datacenters_activities' database selected. A message says 'No tables found in database.' Below it, a 'Create table' form is open. The 'Name:' field contains 'room_dc230'. To its right, a red arrow points to the 'Number of columns:' dropdown, which has '6' selected. Another red arrow points to the dropdown's current value '5'. A third red arrow points to the 'Go' button, which is highlighted with a blue box and has the number '6' above it. At the bottom of the page, there is a 'Console' tab.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Table Name: room_ds230 Number of columns: 6

The screenshot shows the 'Structure' tab of the phpMyAdmin interface for the 'room_dc230' table. The table has six columns:

- id**: Type INT, Length/Values 11, Default NULL, Null, Index PRIMARY, A_I checked, Comments 12.
- logDateTime**: Type DATETIME, Length/Values CURRENT_TIME, Default NULL.
- rackID**: Type VARCHAR(255), Length/Values 18, Default None.
- tempValue**: Type VARCHAR(255), Length/Values 21, Default None.
- humidValue**: Type VARCHAR(255), Length/Values 24, Default None.
- rackStatus**: Type VARCHAR(255), Length/Values 27, Default None.

A modal window titled 'Add index' is open, showing an index named 'PRIMARY' with 'Column' set to 'id'. A red arrow labeled 11 points from the 'Index' button in the modal to the 'Save' button at the bottom of the main table structure screen.

Red arrows numbered 1 through 28 highlight various elements:

- Table name: room_dc230
- Name: id
- Type: INT
- Length/Values: 11
- Default: NULL
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 10
- Name: logDateTime
- Type: DATETIME
- Length/Values: CURRENT_TIME
- Default: NULL
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 12
- Name: rackID
- Type: VARCHAR
- Length/Values: 18
- Default: None
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 13
- Name: tempValue
- Type: VARCHAR
- Length/Values: 21
- Default: None
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 14
- Name: humidValue
- Type: VARCHAR
- Length/Values: 24
- Default: None
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 15
- Name: rackStatus
- Type: VARCHAR
- Length/Values: 27
- Default: None
- Collation: NULL
- Attributes: Null, Index PRIMARY
- Comments: 16
- Table comments:
- Storage Engine: InnoDB
- PARTITION definition:
- Partition by: Expression or column list
- Partitions:
- Save button: 28

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ What you should know on PHPMyAdmin panel (at least).

The screenshot shows the PHPMyAdmin interface with the following details:

- Left sidebar:** Shows the database tree with the 'datacenters_activities' database selected.
- Top menu:** Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, Events, More.
- Structure tab:** Active (highlighted in orange). Contains a 'Filters' section and a search bar.
- Table list:** Shows one table: 'room_dc230'. The table has 4 columns and 0 rows. It is defined as InnoDB, latin1_swedish_ci, 16 KiB.
- Action buttons:** Sum, a, b, c, d, e. Buttons 'a' and 'b' are highlighted in orange, while 'c', 'd', and 'e' are grey.
- Create table form:** Name: [] Number of columns: 4 Go button.
- Console:** A small text input field at the bottom.

- a **Browse** → to view / edit / delete data stored in room_dc230, increment counter will proceed to next number.
- b **Structure** → to view / edit / delete / add table structure.
- c **Insert** → to manually insert data into room_dc230.
- d **Empty** → to empty room_dc230 database, any increment counter will start from 1.
- e **Drop** → will delete room_dc230 table.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

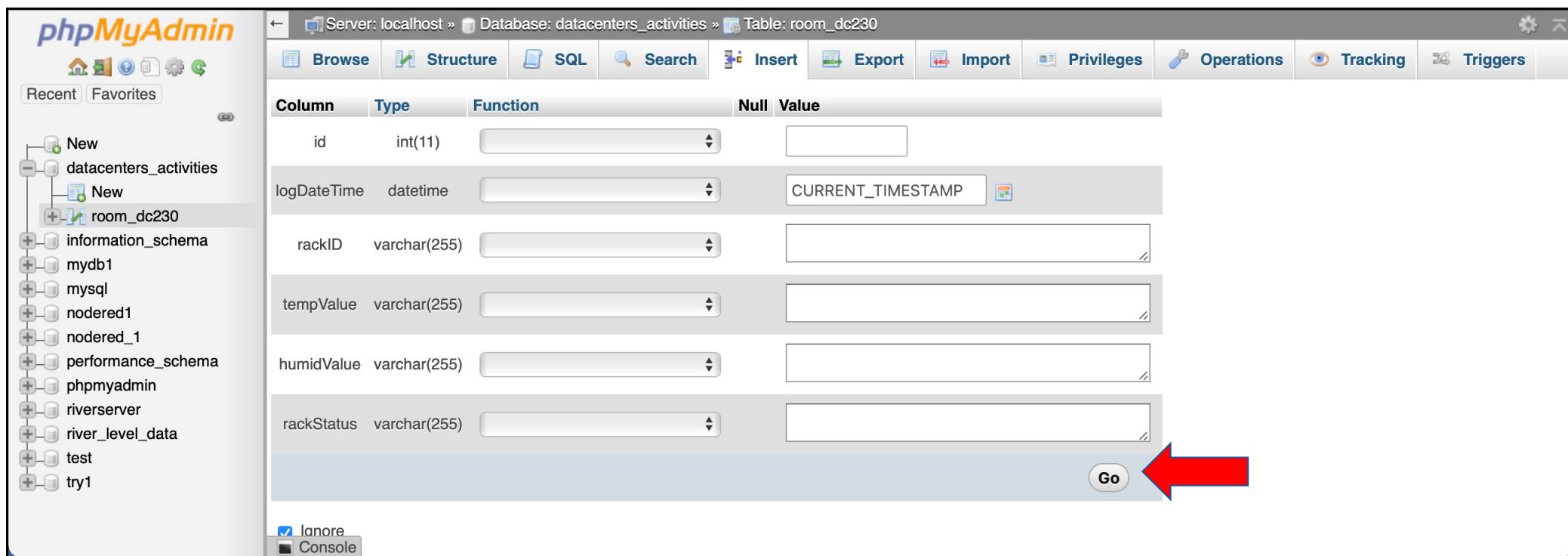
Solution: Step 3 CREATING MySQL DATABASE

→ Click **a** **Browse** to view room_dc230 database. It is empty since no data in it.

The screenshot shows the phpMyAdmin interface. On the left, the database tree is visible with the 'datacenters_activities' database selected. Inside it, the 'room_dc230' table is selected. The main area shows the table structure with columns: id, logDateTime, rackID, tempValue, humidValue, and rackStatus. A message at the top states: "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)". Below the message is the SQL query: "SELECT * FROM `room_dc230`". The table body displays the message "An empty database". At the bottom, there are options for "Query results operations" like "Create view" and "Bookmark this SQL query". The "Console" tab is also visible at the bottom.

Solution: Step 3 CREATING MySQL DATABASE

- In order to insert data manually into room_dc230, select  **Insert**.
- For this time, leave **ALL** text box empty. Then click **Go** to save into room_dc230 database.



The screenshot shows the phpMyAdmin interface for the 'room_dc230' table. The table structure is as follows:

| Column | Type | Function | Null | Value |
|-------------|--------------|----------|------|-------------------|
| id | int(11) | | | |
| logDateTime | datetime | | | CURRENT_TIMESTAMP |
| rackID | varchar(255) | | | |
| tempValue | varchar(255) | | | |
| humidValue | varchar(255) | | | |
| rackStatus | varchar(255) | | | |

A red arrow points to the 'Go' button at the bottom right of the form.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Observe the red arrow. What do you think?

→ a Click **Browse** link to view the input.

```
INSERT INTO `room_dc230` (`id`, `logDateTime`, `rackID`, `tempValue`, `humidValue`, `rackStatus`) VALUES (NULL, CURRENT_TIMESTAMP, '', '', '', ''');
```

The screenshot shows the phpMyAdmin interface for the 'datacenters_activities' database. The 'room_dc230' table is selected. The 'Browse' tab is active. A success message '1 row inserted.' is displayed above the query log. The query log shows the executed SQL statement:

```
INSERT INTO `room_dc230` (`id`, `logDateTime`, `rackID`, `tempValue`, `humidValue`, `rackStatus`) VALUES (NULL, CURRENT_TIMESTAMP, '', '', '', '');
```

A yellow box highlights the message 'Blank data inserted into rackID, tempValue, humidValue & rackStatus'. A red arrow points from the text 'What do you think?' in the previous slide to the 'rackID' column in the 'Columns' list on the right, which lists the columns: id, logDateTime, rackID, tempValue, humidValue, and rackStatus. Another red arrow points to the 'rackID' column in the query log.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

- Congrats! You managed to add your first data into room_dc230 database.
- Click **Edit** link if you wish to edit the data.
- **Delete** will not reset the *id* into **zero**. Next data entered will get the following running number.

The screenshot shows the phpMyAdmin interface for the 'room_dc230' table. The table has columns: id, logDateTime, rackID, tempValue, humidValue, and rackStatus. There is one row with the following values:

| | Edit | Copy | Delete | id | logDateTime | rackID | tempValue | humidValue | rackStatus |
|--|----------------------|----------------------|------------------------|----|---------------------|--------|-----------|------------|------------|
| | | | | 1 | 2021-02-09 01:21:19 | | | | |

A red arrow points to the 'Delete' link in the row header. The status bar at the bottom right of the table area also contains a red arrow pointing to the same 'Delete' link.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Go to **Insert** link and add the information as show by red arrow & press **Go** to save the data.

The screenshot shows the phpMyAdmin interface for the 'room_dc230' table. The table structure is as follows:

| Column | Type | Function | Null | Value |
|-------------|--------------|----------|------|-------------------|
| id | int(11) | | | |
| logDateTime | datetime | | | CURRENT_TIMESTAMP |
| rackID | varchar(255) | | | r1 |
| tempValue | varchar(255) | | | 24 |
| humidValue | varchar(255) | | | 70 |
| rackStatus | varchar(255) | | | Normal |

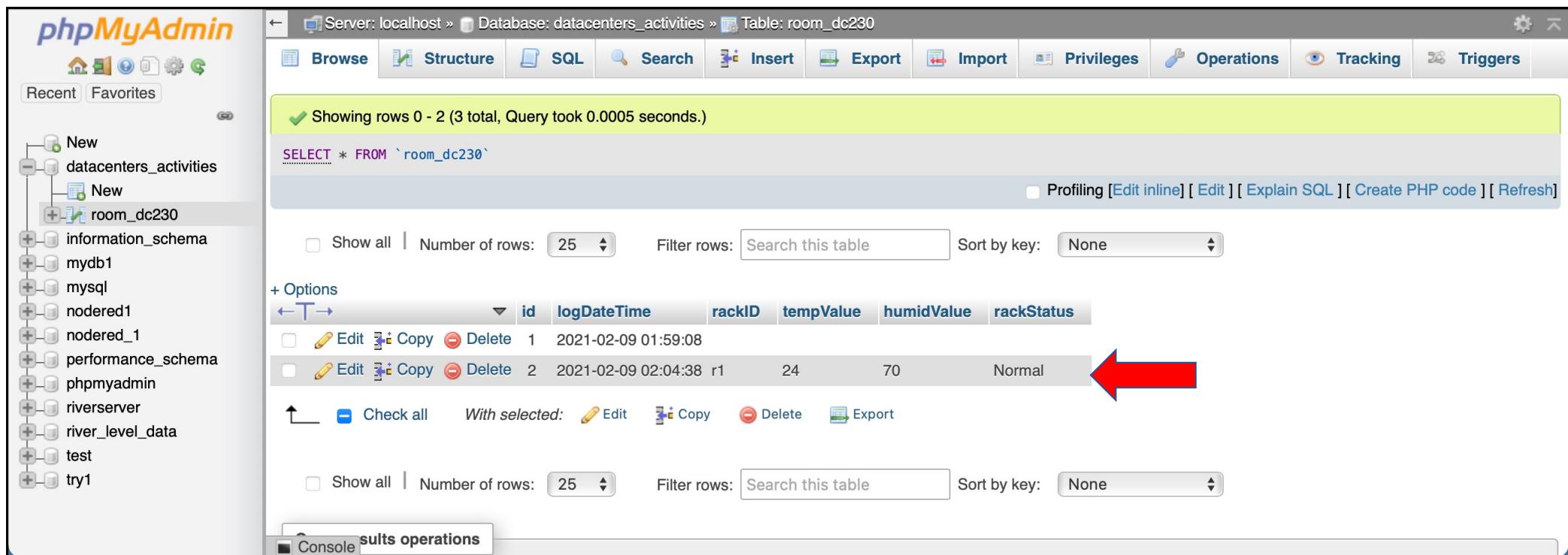
Red arrows numbered 1 through 5 point to the following elements:

- Arrow 1 points to the 'rackID' input field containing 'r1'.
- Arrow 2 points to the 'tempValue' input field containing '24'.
- Arrow 3 points to the 'humidValue' input field containing '70'.
- Arrow 4 points to the 'rackStatus' input field containing 'Normal'.
- Arrow 5 points to the 'Go' button at the bottom right.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 3 CREATING MySQL DATABASE

→ Red arrow shows the new data that have been entered.



The screenshot shows the phpMyAdmin interface for the 'room_dc230' table in the 'datacenters_activities' database. The table has columns: id, logDateTime, rackID, tempValue, humidValue, and rackStatus. There are two rows of data:

| | 1 | 2021-02-09 01:59:08 | r1 | 24 | 70 |
|--|---|---------------------|----|----|----|
| | 2 | 2021-02-09 02:04:38 | r1 | 24 | 70 |

A red arrow points to the second row of data, indicating the newly entered data.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

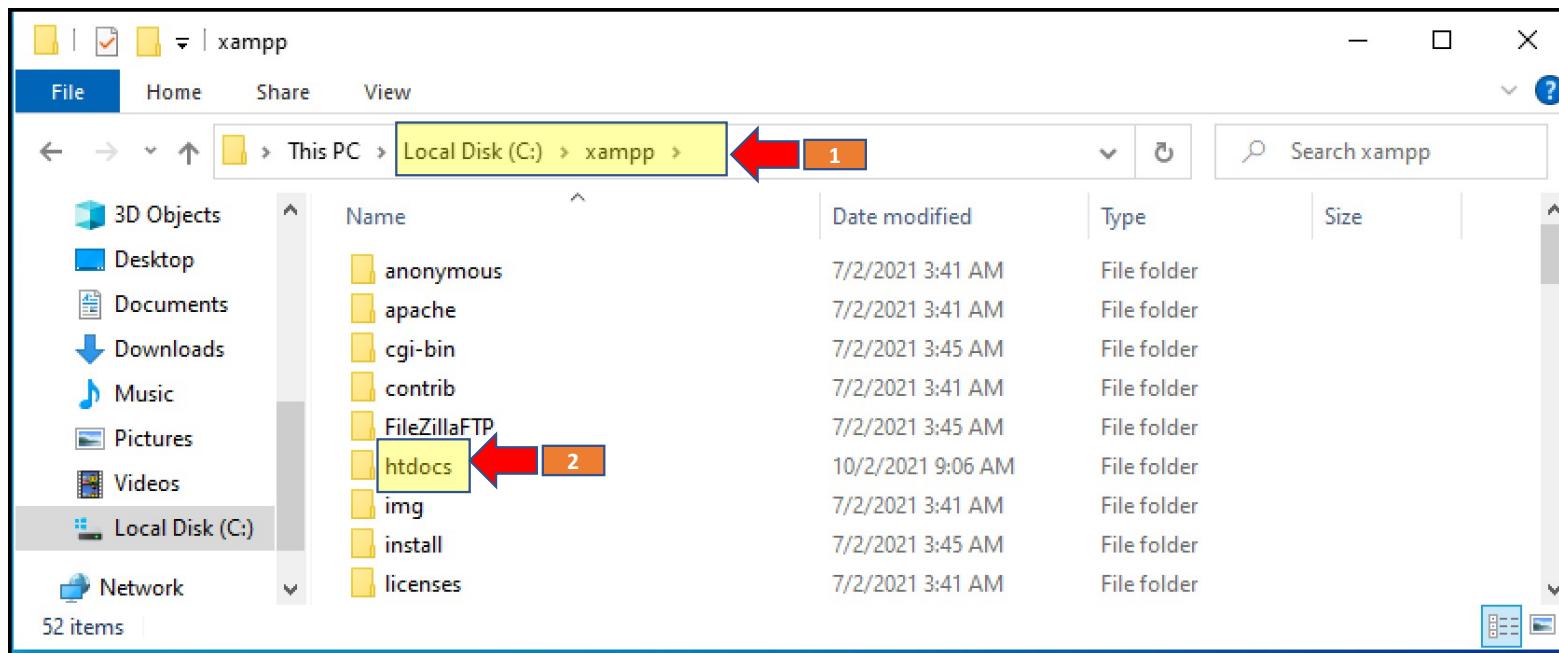
Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

→ Two PHP files are needed; esptodb.php & webview.php.



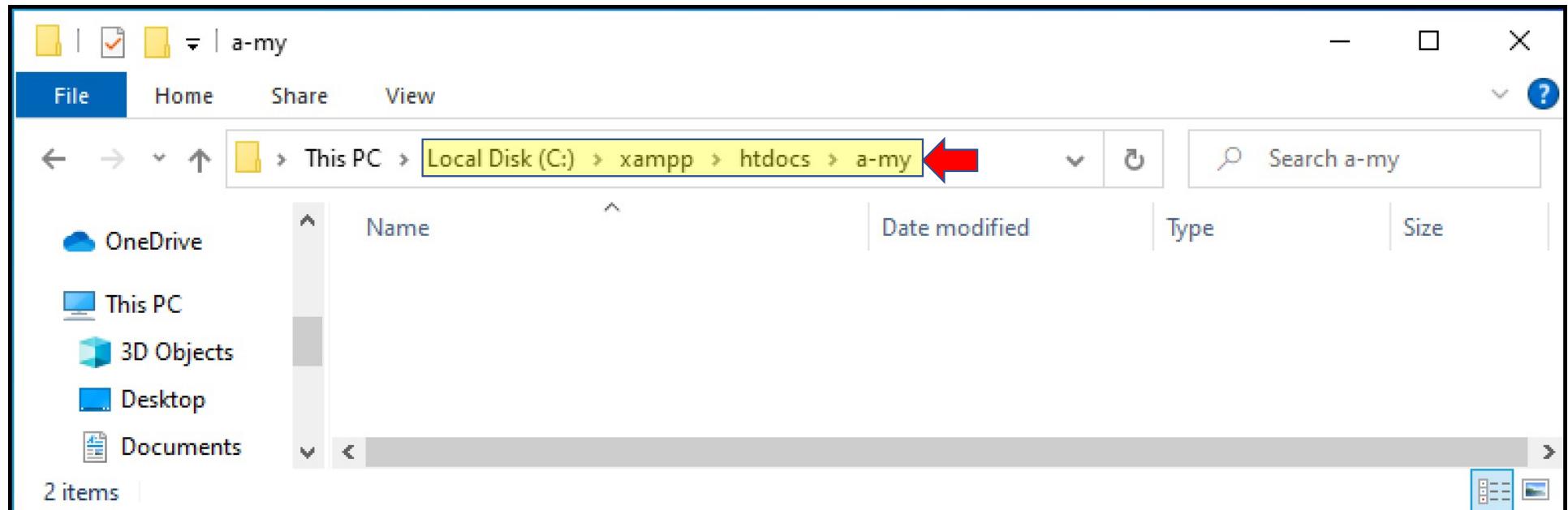
Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- In order to publish any website either PHP or HTML, you need to save the files in a **specific folder**, named **htdocs**. It can be found in XAMPP folder.
- It is advised that each project has its own folder. This may avoid you to override files that carries same filename.



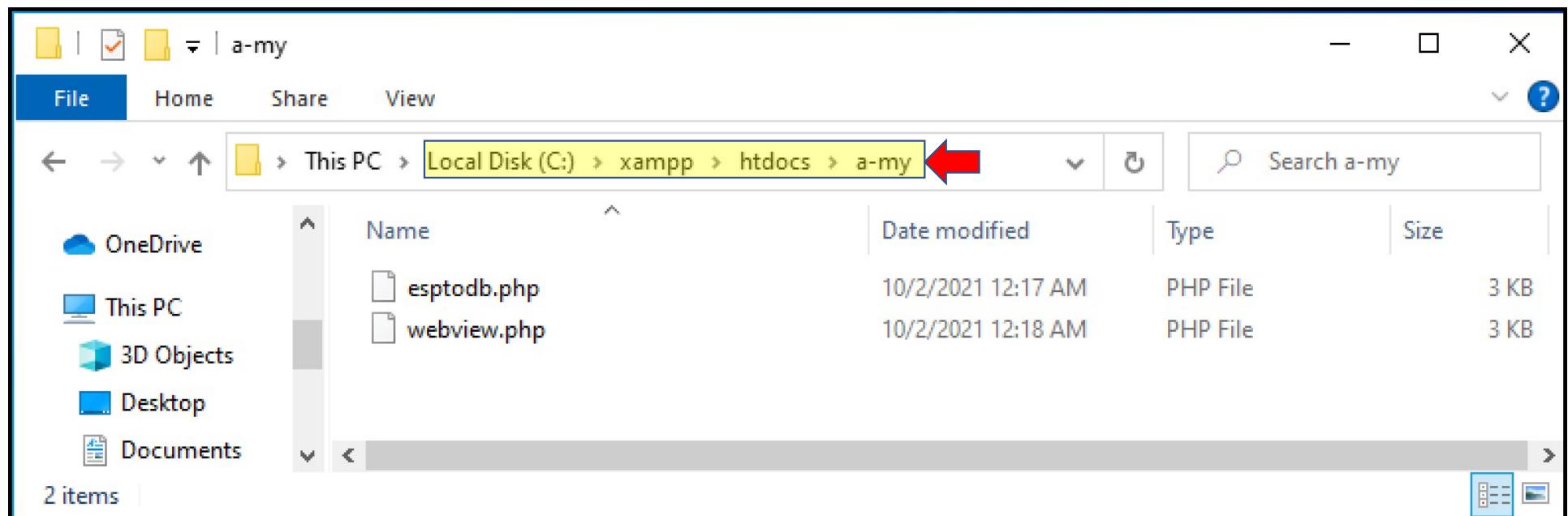
Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

→ Navigate to **htdocs** & create new folder named **a-my**.



Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Download “**esptodb.php**” & “**webview.php**” from <http://bit.ly/3p2pmNd>.
- Extract & paste the files into **a-my** folder.
- ESP32 will use “**esptodb.php**” as a medium to save information into MySQL while client will use “**webview.php**” to view the information thru web browsers.



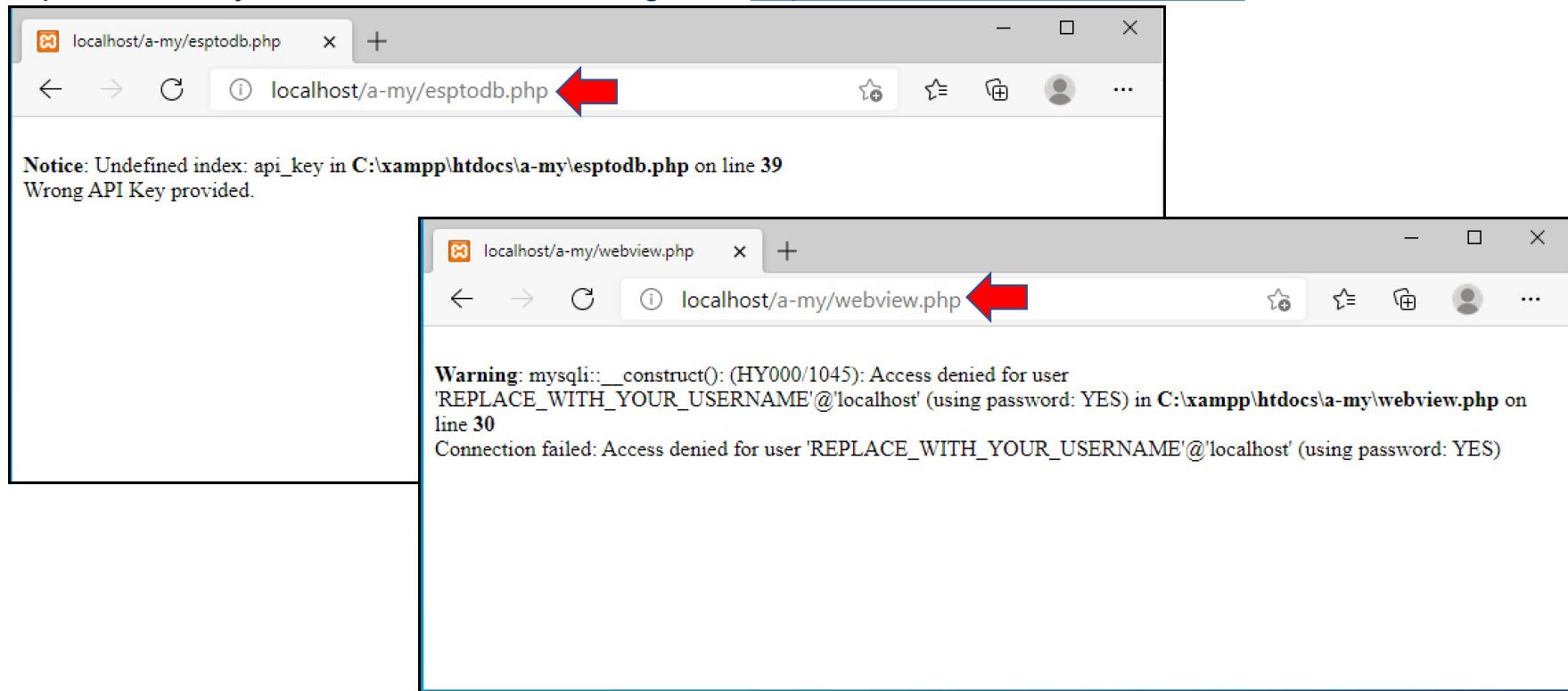
Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

→ Open up your web browser & type **localhost/a-my/** in the address bar. You should get the similar output. If you failed to see this, go to your XAMPP C-Panel & check whether the server services has been deployed or not.



Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Click at the file one at a time. There are errors/warnings since the files were not properly configured.
- You need a text editor to edit both files & we are going to use Sublime Text as the editor for PHP scripts. You may download it from the following link: <https://www.sublimetext.com/3>



Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Open ***webview.php*** with **sublimetext** or any preferable text editor. The credentials are to establish connection to the database.
- By default, **\$servername** is “**localhost**” & **\$username** is “**root**”.

```

15 $servername = "localhost";
16
17 // REPLACE with your Database name
18 $dbname = "REPLACE_WITH_YOUR_DATABASE_NAME";
19
20 // REPLACE with table name
21 $tablename = "REPLACE_WITH_YOUR_TABLE_NAME";
22
23 // REPLACE with Database user
24 $username = "REPLACE_WITH_YOUR_USERNAME";
25
26 // REPLACE with Database user password
27 $password = "REPLACE_WITH_YOUR_PASSWORD";

```

→ Set up database with the following properties:
 Database Name: **datacenter_activities**
 Table Name: **room_dc230**



```

15 $servername = "localhost";
16
17 // REPLACE with your Database name
18 $dbname = "datacenters_activities";
19
20 // REPLACE with table name
21 $tablename = "room_dc230";
22
23 // REPLACE with Database user
24 $username = "root";
25
26 // REPLACE with Database user password
27 $password = "";

```

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Before getting access to MySQL database, the client must establish a connection to the server as shown in **Line#30**. If any of the credentials incorrect, the client don't have the permission to access the server. The script stops & will display error message at the browser. (MySQLi = MySQL improved)
- **Line#36** indicates selection of 5 columns from **room_dc230** table with descending order of ***id***. The syntax is put into a variable called **\$sql**.

```
29 // Create connection
30 $conn = new mysqli($servername, $username, $password, $dbname);
31 // Check connection
32 if ($conn->connect_error) {
33     die("Connection failed: " . $conn->connect_error);
34 }
35
36 $sql = "SELECT logDateTime, rackID, tempValue, humidValue, rackStatus FROM $tablename ORDER BY id DESC";
37
```

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

→ A mix of PHP and HTML syntax in creating table.

```

38 echo '<table cellspacing="5" cellpadding="5">
39     <tr>
40         <td>Rack ID</td>
41         <td>Timestamp</td>
42         <td>Temparature (C)</td>
43         <td>Humidity (%)</td>
44         <td>Rack Status</td>
45     </tr>';

```

→ Line#47 – Line#69 is hosting asking, fetch data from database & display them in table manner.

→ "query()" is a function that fetch data that was assigned in \$sql variable. The results were put into a variable called \$results.

→ each data from "\$results" will be split into an array form according to its respected field.

```

47 if ($result = $conn->query($sql)) {
48     while ($row = $result->fetch_assoc()) {
49         $row_id = $row["rackID"];
50         $row_log = $row["logDateTime"];
51         $row_temp = $row["tempValue"];
52         $row_humid = $row["humidValue"];
53         $row_stats = $row["rackStatus"];

```

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Script to display the information fetched from database in table form. Each row represent 1 set data / reading. New row carries another set of reading/data.

```

60      echo '<tr>
61          <td>' . $row_id . '</td>
62          <td>' . $row_log . '</td>
63          <td>' . $row_temp . '</td>
64          <td>' . $row_humid . '</td>
65          <td>' . $row_stats . '</td>
66      '</tr>';
67  }

```

- The fetched data in **\$result** are emptied, thus the variable carries no information.
- "**close()**" is a function that closes previously open database connection.
- End of HTML script.

```

68      $result->free();
69  }
70
71 $conn->close();
72 ?>
73 </table>
74 </body>
75 </html>

```

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- Open **esptodb.php** with **sublimetext** or any preferable text editor. Fill up with your database credentials.
- If **webview.php** is meant for client to view the temperature & humidity of the racks, **esptodb.php** is a script for EPS32 saving the racks data into the database.
- **\$api_key_value** is a kind of password to except information send from the **ESP32**. **ESP32** must send same key & the information will only be entertained when the key is matched. Otherwise, the information will be rejected.

```
28 // Keep this API Key value to be compatible with the ESP32 code provided in the project page.  
29 // If you change this value, the ESP32 sketch needs to match  
30 $api_key_value = "tPmAT5Ab3j7F9";  
31
```

- **Line#33** will empty the content of the handlers. The task of the handlers are to keep for information temporarily when received from the ESP32 thru HTTP POST or HTTP GET request.

```
32 // empty the variables  
33 $api_key= $rackID = $tempVal = $humidVal = $rackStat = "";  
34
```

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- There are 2 types of Hyper Text Transfer Protocol (HTTP) request methods: **POST & GET**.
- What is HTTP? HTTP works as request-response protocol between a client & a server.
- **HTTP GET** is used to request data from a specified resource. It is often used to get values from APIs.

- eg., ***http://localhost/a-my/esptodb.php?key=tPmAT5Ab3jF9&temp=10***
- Name = **Key** & value = **tPmAT5Ab3jF9** + name = **temp** & value = **10** are send together in URL
- With HTPP GET, data is **visible** to everyone

- **HTTP POST** is used to send information to a server to create/update a resource.

- eg., publish sensor readings to a server
- The data sent to the server with POST is stored in the request body of the HTTP request
- ***String serverPath = serverName "?temperature=24.37"; // sample in sketch***
- With HTTP POST, data is **not visible** in the URL request. However, if it's not encrypted, it's still visible in the request body.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

http://localhost/a-my/esptodbbase.php?api_key=tPmAT5Ab3j7F9&rID=v112&temp=22&humid=40¤tStat=NORMAL

a

```
30 // DESC TEST FOR PRODUCTION
31
32 if ($_SERVER["REQUEST_METHOD"] == "GET") { //GET or POST
33     //echo "test";
34     $api_key = test_input($_GET["api_key"]); //$_GET or $_POST
35     if($api_key == $api_key_value) { //$_GET or $_POST
36         $rackID = test_input($_GET["rID"]); //$_GET or $_POST
37         $tempVal = test_input($_GET["temp"]); //$_GET or $_POST
38         $humidVal = test_input($_GET["humid"]); //$_GET or $_POST
39         $rackStat = test_input($_GET["currentStat"]); //$_GET or $_POST
40
41         // Create connection
42         $conn = new mysqli($servername, $username, $password, $dbname);
43         // Check connection
44         if ($conn->connect_error) {
45             die("Connection failed: " . $conn->connect_error);
46         }
47
48         $sql = "INSERT INTO tablename (rackID, tempValue, humidValue, rackStatus)
49             VALUES ('" . $rackID . "', '" . $tempVal . "', '" . $humidVal . "', '" . $rackStat . "')";
50
51         if ($conn->query($sql) === TRUE) {
52             echo "New record created successfully";
53         } else {
54             echo "Error: " . $sql . "<br>" . $conn->error;
55         }
56
57         $conn->close();
58     } else {
59         echo "Wrong API Key provided.";
60     }
61
62 }
63
64 }
65
66 }
67
68 }
69
70 else {
71     echo "No data posted with HTTP POST.";
72 }
73
74 function test_input($data) {
75     $data = trim($data); //remove whitespace
76     $data = stripslashes($data); //removes backslashes
77     $data = htmlspecialchars($data); //to converts special characters
78     //& (ampersand), " (double quote), ' (single quote), < (less than), > (greater than)
79     //to HTML entities ( i.e. & (ampersand) becomes &amp;, ' (single quote) becomes &#039,
80     //< (less than) becomes &lt; (greater than) becomes &gt; ). //< (less than) becomes &lt; (greater than) becomes &gt; .
81     return $data;
82 }
```

35

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

- It is now the time to test both PHP scripts. Make sure that the database credentials have set accordingly.
- No information print out since the database is still empty.

The screenshot shows a desktop environment with two windows open. The top window is a web browser displaying a table with columns: Rack ID, Timestamp, Temperatute (C), Humidity (%), and Rack Status. The table is currently empty. The bottom window is a phpMyAdmin interface showing the results of a SQL query: "SELECT * FROM `room_dc230`". The results pane displays the following message: "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)". The query itself is: "SELECT * FROM `room_dc230`".

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 4 PHP SCRIPTS & HTDOCS & CODE EXPLANATION & RUN TEST

→ Open a browser, then copy & paste the following link & press enter.

http://localhost/a-my/esptodb.php?api_key=tPmAT5Ab3j7F9&rID=v112&temp=22&humid=40¤tStat=NORMAL

New record created successfully

| Rack ID | Timestamp | Temparature (C) | Humidity (%) | Rack Status |
|---------|---------------------|-----------------|--------------|-------------|
| v112 | 2021-02-12 13:11:59 | 22 | 40 | NORMAL |
| v112 | 2021-02-12 13:11:40 | 22 | 40 | NORMAL |
| r1 | 2021-02-09 02:04:38 | 24 | 70 | Normal |
| | 2021-02-09 01:59:08 | | | |

Server: localhost » Database: datacenters_activities » Table: room_dc230

| | id | logDateTime | rackID | tempValue | humidValue | rackStatus |
|---|----|---------------------|--------|-----------|------------|------------|
| 1 | 1 | 2021-02-09 01:59:08 | | | | |
| 2 | 2 | 2021-02-09 02:04:38 | r1 | 24 | 70 | Normal |
| 4 | 4 | 2021-02-12 13:11:40 | v112 | 22 | 40 | NORMAL |
| 5 | 5 | 2021-02-12 13:11:59 | v112 | 22 | 40 | NORMAL |

→ Try this:

1- <http://localhost/a-my/>

2- http://localhost/a-my/esptodb.php?api_key=tpMAT5Ab3j7F9&rID=v111&temp=22.4&humid=40%¤tStat=999

Solution: Step 5 ESP32 POST INFORMATION TO DBASE: SKETCH WALKTHROUGH

- Open **sketch-vii_esp32-dht11-http-post.ino** from <http://bit.ly/3p2pmNd> downloaded in Step 4. Do not upload the sketch since you need to make some changes.
- This will inform the compiler to compile the correct library no matter if what board you are currently uploading into. In this case, either ESP32 or NodeMCU & ESP8266.

```
13 #ifdef ESP32
14 // this will compile for ESP32 board
15 #include <WiFi.h>
16 #include <HTTPClient.h>
17 #else
18 // this will compile for ESP8266, NodeMCU boards
19 #include <ESP8266WiFi.h>
20 #include <ESP8266HTTPClient.h>
21 #include <WiFiClient.h>
22#endif
```

Solution: Step 5 ESP32 POST INFORMATION TO DBASE: SKETCH WALKTHROUGH

→ Set the followings to the WiFi network connection.

```
24 // Replace with your network credentials  
25 const char* ssid      = "REPLACE_WITH_YOUR_SSID";  
26 const char* password = "REPLACE_WITH_YOUR_PASSWORD";
```

→ Change the “192.168.1.102” with your server’s IP address. How to find IP address of your system ☺?

→ The file path “/a-my/esptodb.php” reflects the “drive/xampp/htdocs/” & set in STEP xx.

```
28 // REPLACE with your Domain name and URL path or server's IP address with path  
29 const char* serverName = "http://192.168.1.102/a-my/esptodb.php";
```

→ Both ESP32 has the same API key with esptodb.php to avoid fake request. If not match, the information will not save into database. The API key is manually generated by the developer/user.

```
31 // Keep this API Key value to be compatible with the PHP code provided in the project page.  
32 // If you change the apiKeyValue value, the PHP file "/esptodb.php" also needs to have the same key  
33 String apiKeyValue = "tPmAT5Ab3j7F9";
```

→ Set the sensorID accordingly – e.g. rackID. This ID must be different from one another & must be documented, easier to detect & locate.

```
35 String sensorID = "a1";
```

Solution: Step 5 ESP32 POST INFORMATION TO DBASE: SKETCH WALKTHROUGH

→ The board will connect to the WiFi router. If any of the SSID or Password is incorrect, the board will not join the network. Open Serial Monitor to view the process. IP address will be published if the board connected to the router.

```
40 void setup() {  
41   Serial.begin(115200);  
42  
43   WiFi.begin(ssid, password);  
44   Serial.println("Connecting");  
45   while(WiFi.status() != WL_CONNECTED) {  
46     delay(500);  
47     Serial.print(".");  
48   }  
49   Serial.println("");  
50   Serial.print("Connected to WiFi network with IP Address: ");  
51   Serial.println(WiFi.localIP());  
52 }
```

→ Fail to connect to router.

```
Connecting  
.....
```

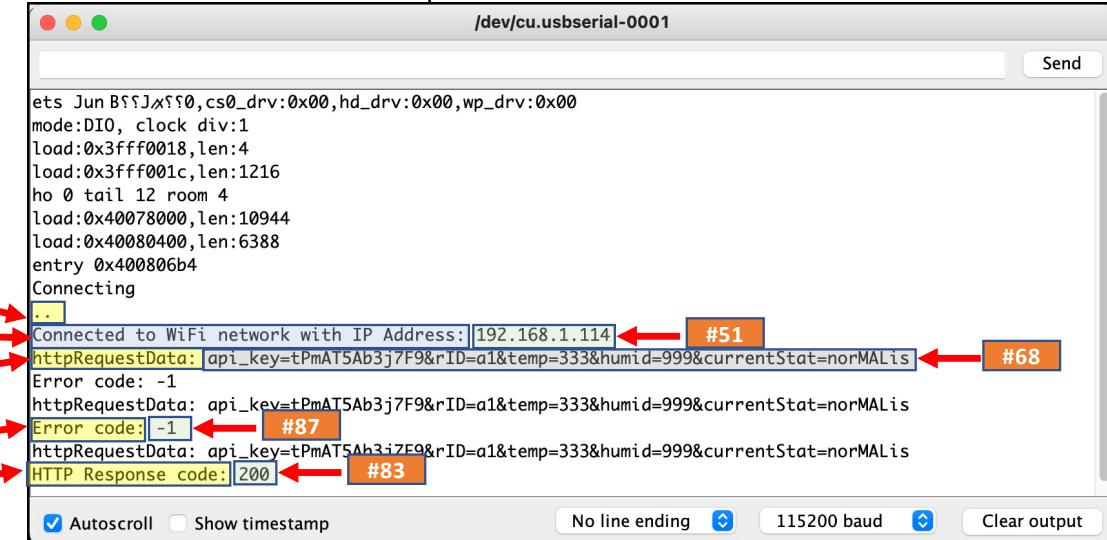
2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 5 ESP32 POST INFORMATION TO DBASE: SKETCH WALKTHROUGH

```

54 void loop() {
55 //Check WiFi connection status
56 if(WiFi.status()== WL_CONNECTED){
57   HTTPClient http;
58
59   // Your Domain name with URL path or IP address with path
60   http.begin(serverName);
61
62   // Specify content-type header
63   http.addHeader("Content-Type", "application/x-www-form-urlencoded");
64
65   // Prepare your HTTP POST request data
66   String httpRequestData = "api_key=" + apiKeyValue + "&rID=" + sensorID + "&temp=" + 333 + "&humid=" + 999 + "&currentStat=" + "norMALis";
67   Serial.print("httpRequestData: ");
68   Serial.println(httpRequestData);
69
70   // You can comment the httpRequestData variable above
71   // then, use the httpRequestData variable below (for testing purposes without the DHT11 sensor) -->Similar with Line#66
72   //String httpRequestData = "api_key=tPmAT5Ab3j7F9&rID=abc&temp=123&humid=24.75&currentStat=norMALis";
73
74   // Send HTTP POST request
75   int httpResponseCode = http.POST(httpRequestData);
76
77   // If you need an HTTP request with a content type: text/plain
78   //http.addHeader("Content-Type", "text/plain");
79   //int httpResponseCode = http.POST("Hello, World!");
80
81   if (httpResponseCode>0) {
82     Serial.print("HTTP Response code: ");
83     Serial.println(httpResponseCode);
84   }
85   else {
86     Serial.print("Error code: ");
87     Serial.println(httpResponseCode);
88   }
89   // Free resources
90   http.end();
91
92   else {
93     Serial.println("WiFi Disconnected");
94   }
95   //Send an HTTP POST request every 30 seconds
96   delay(30000);
97 }
```

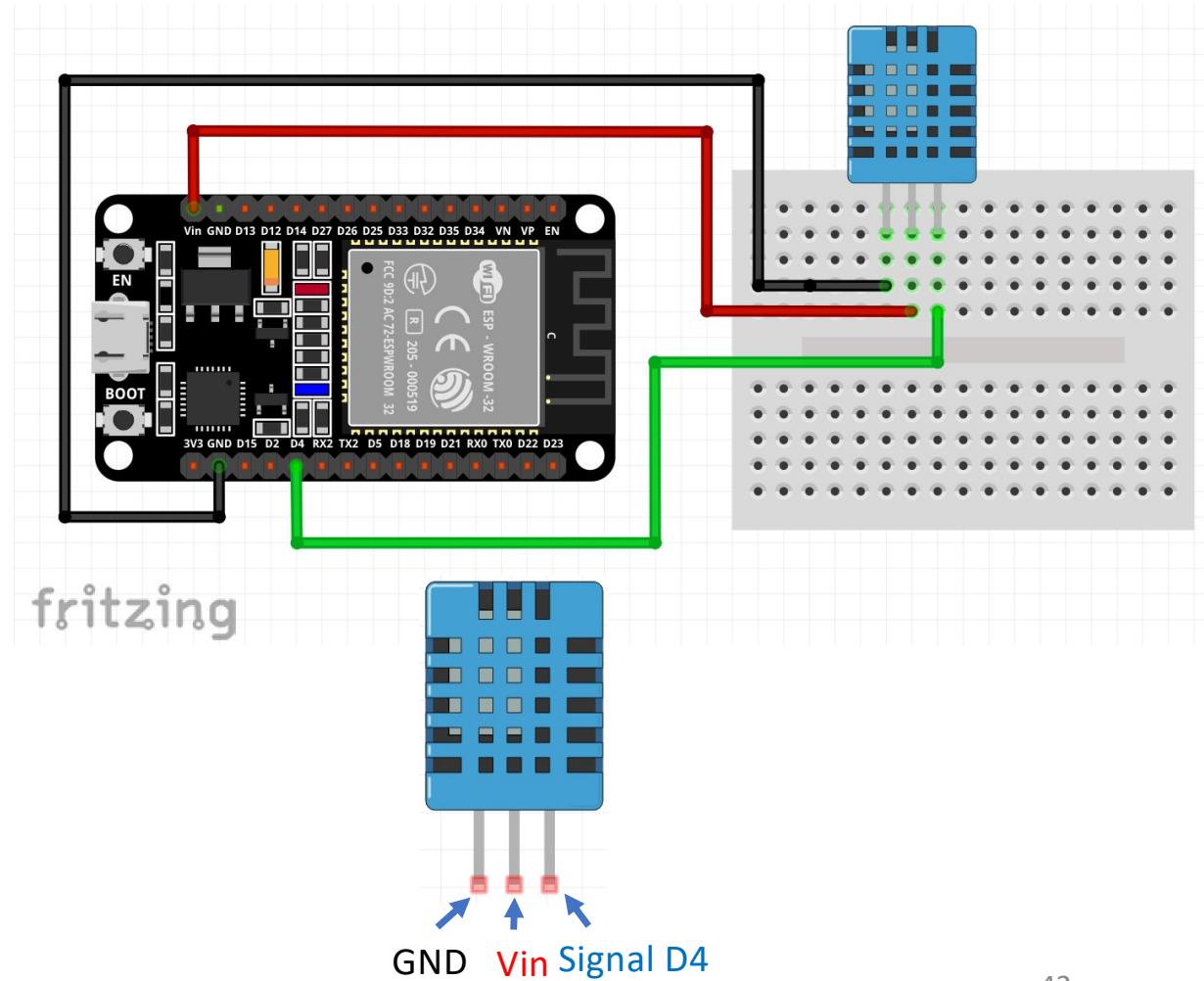
- #66: Test send data to esptodb.php (IP add at #29)
- #81: Success to communicate with server
- #85: Unable to communicate with server --> not in the same network
- #96: Update every 30 seconds



2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 6 SCHEMATIC DIAGRAM

- After success tested the sketch, its time to combine the DHT part with ESP32.
- Before we go any further, construct the circuit as in the figure.
- Check a few times, the connections that have been made before plugging ESP32 to USB port. This might prevent short circuit to DHT11. Watch out on sensors' pin. Same type, doesn't mean same pin configurations.



Solution: Step 7 THE WORKING SKETCH

→ Open **sketch-vii_esp32-dht11-http-post.ino** from <http://bit.ly/3p2pmNd> downloaded in Step 4. Do not upload the sketch since you need to make some changes.

```

13 #ifdef ESP32
14 // this will compile for ESP32 board
15 #include <WiFi.h>
16 #include <HTTPClient.h>
17 #else
18 // this will compile for ESP8266, NodeMCU boards
19 #include <ESP8266WiFi.h>
20 #include <ESP8266HTTPClient.h>
21 #include <WiFiClient.h>
22 #endif
23
24 #include <dht11.h> ←
25 dht11 DHT;
26 #define DHT11_PIN 4
27
28 // Replace with your network credentials
29 const char* ssid      = "air24";//REPLACE_WITH_YOUR_SSID";
30 const char* password = "polis12345";//REPLACE_WITH_YOUR_PASSWORD";
31
32 // REPLACE with your Domain name and URL path or server's IP address with path
33 const char* serverName = "http://192.168.1.102/a-my/esptodb.php";
34
35 // Keep this API Key value to be compatible with the PHP code provided in the project page.
36 // If you change the apiKeyValue value, the PHP file "/esptodb.php" also needs to have the same key
37 String apiKeyValue = "tPmAT5Ab3j7F9";
38
39 String sensorID = "a1";
40 //// latihan
41 ////String sensorLocation = "Office";

```

Solution: Step 7 THE WORKING SKETCH

```
44 void setup() {  
45   Serial.begin(115200);  
46   WiFi.begin(ssid, password);  
47   Serial.println("Connecting");  
48   while(WiFi.status() != WL_CONNECTED) {  
49     delay(500);  
50     Serial.print(".");  
51   }  
52   Serial.println("");  
53   Serial.print("Connected to WiFi network with IP Address: ");  
54   Serial.println(WiFi.localIP());  
55 }
```

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 7 THE WORKING SKETCH

```
57 void loop() {  
58     int chk = DHT.read(DHT11_PIN);    //Read humid & temp ←  
59     //Check WiFi connection status  
60     if(WiFi.status()== WL_CONNECTED){  
61         HTTPClient http;  
62  
63         // Your Domain name with URL path or IP address with path  
64         http.begin(serverName);  
65  
66         // Specify content-type header  
67         http.addHeader("Content-Type", "application/x-www-form-urlencoded");  
68  
69         // Prepare your HTTP POST request data  
70         String httpRequestData = "api_key=" + apiKeyValue + "&rID=" + sensorID  
71             + "&temp=" + DHT.temperature + "&humid=" + DHT.humidity ←  
72             + "&currentStat=" + norMALis;  
73         Serial.print("httpRequestData: ");  
74         Serial.println(httpRequestData);  
75  
76         // You can comment the httpRequestData variable above  
77         // then, use the httpRequestData variable below (for testing purposes without the DHT11 sensor) -->Similar with Line#66  
78         //String httpRequestData = "api_key=tPmAT5Ab3j7F9&rID=abc&temp=123&humid=24.75&currentStat=norMALis";  
79  
80         // Send HTTP POST request  
81         int httpResponseCode = http.POST(httpRequestData);  
82  
83         // If you need an HTTP request with a content type: text/plain  
84         //http.addHeader("Content-Type", "text/plain");  
85         //int httpResponseCode = http.POST("Hello, World!");  
86         if (httpResponseCode>0) {  
87             Serial.print("HTTP Response code: ");  
88             Serial.println(httpResponseCode);  
89         }  
90         else {  
91             Serial.print("Error code: ");  
92             Serial.println(httpResponseCode);  
93         }  
94         // Free resources  
95         http.end();  
96     }  
97     else {  
98         Serial.println("WiFi Disconnected");  
99     }  
100    //Send an HTTP POST request every 30 seconds  
101    delay(30000);  
102 }
```

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

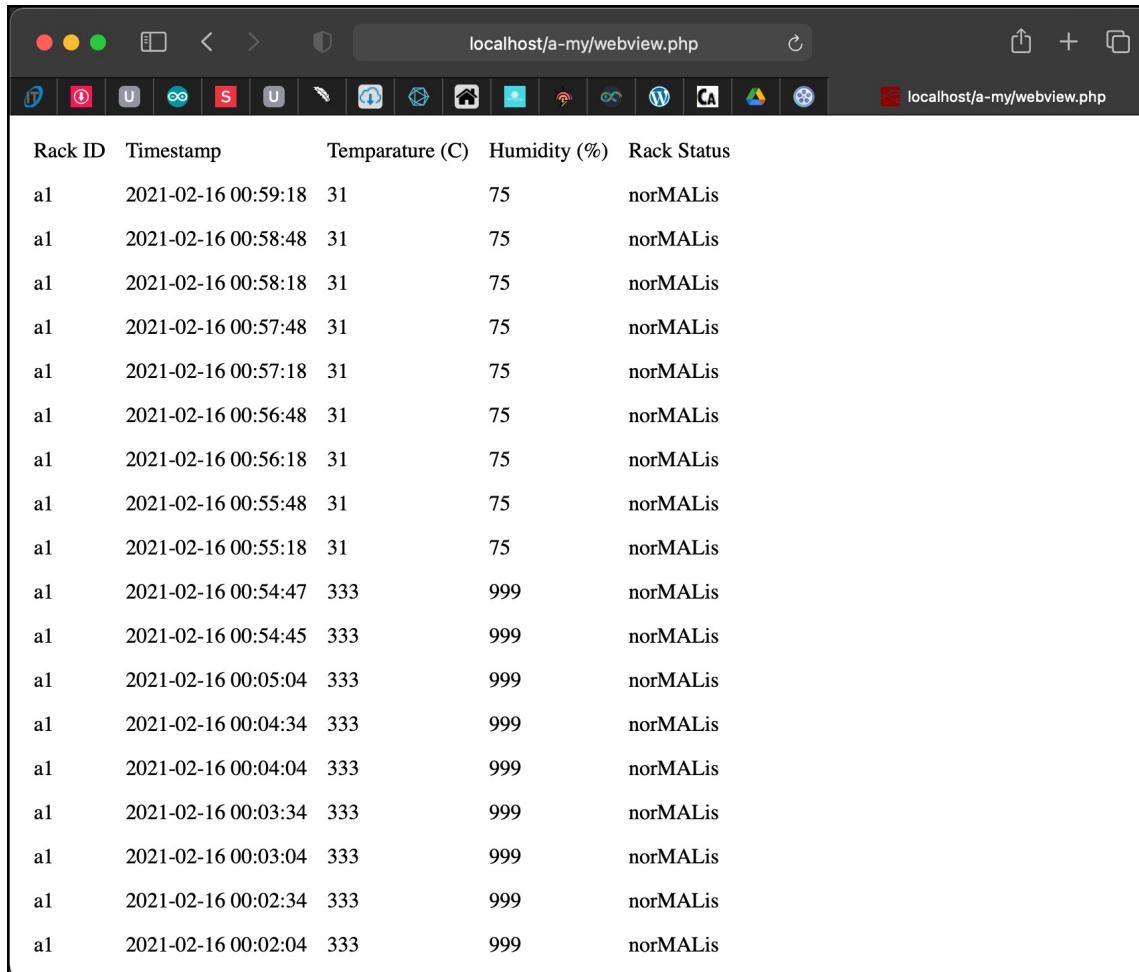
Solution: Step 7 THE WORKING SKETCH

The image displays three windows illustrating the working sketch of a centralized monitoring system:

- Terminal Log:** Shows logs from device `/dev/cu.usbserial-0001`. The logs indicate successful HTTP requests to a server, with timestamps ranging from 00:56:48.461 to 00:58:18.596. Each log entry includes the API key (`tPmAT5Ab3j7F9`), rack ID (`a1`), temperature (`31`), and humidity (`75`).
- MySQL Database:** A screenshot of a MySQL table named `rack_data`. The table has columns: `+ Options`, `id`, `logDateTime`, `rackID`, `tempValue`, `humidValue`, and `rackStatus`. The data shows 35 rows of historical and real-time monitoring data.
- Web Browser:** A screenshot of a web browser window titled `localhost/a-my/webview.php`. It displays a real-time monitoring interface with columns: `Rack ID`, `Timestamp`, `Temparature (C)`, `Humidity (%)`, and `Rack Status`. The data is identical to the MySQL table, showing values for rack `a1`.

2d. Centralized Temperature & Humidity Monitoring System for Server Racks

Solution: Step 7 THE WORKING SKETCH



A screenshot of a web browser window titled "localhost/a-my/webview.php". The browser interface includes standard controls like back/forward, search, and refresh, along with a toolbar containing various icons. The main content area displays a table with the following data:

| Rack ID | Timestamp | Temparature (C) | Humidity (%) | Rack Status |
|---------|---------------------|-----------------|--------------|-------------|
| a1 | 2021-02-16 00:59:18 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:58:48 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:58:18 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:57:48 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:57:18 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:56:48 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:56:18 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:55:48 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:55:18 | 31 | 75 | norMALis |
| a1 | 2021-02-16 00:54:47 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:54:45 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:05:04 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:04:34 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:04:04 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:03:34 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:03:04 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:02:34 | 333 | 999 | norMALis |
| a1 | 2021-02-16 00:02:04 | 333 | 999 | norMALis |

QUESTIONS?

Questions?

EXERCISE

Modify this project by doing the followings:

- a. Add LED statement in ESP32 sketch – if connected to WiFi, built in led (GPIO2) will switch on.
- b. Write condition statements at PHP files:
 - i. If temp $\geq 28^{\circ}\text{C}$ && $\leq 39^{\circ}\text{C}$ status is **ALERT**
 - ii. If temp $\geq 40^{\circ}\text{C}$ status is **DANGER**

The status need to be saved in the database together with other information.

HINT: you need to omit the status information from ESP32 sketch.