

## The Big Picture (Server-Side Development)


Every web app generally needs 3 main parts:

Role	JavaScript Stack	PHP Stack
Server (backend)	Express.js	PHP
Database	MongoDB (NoSQL)	MySQL (SQL)
Runtime + Tools	Node.js	XAMPP (Apache + MySQL + PHP)

## XAMPP = Web Server Environment for PHP

XAMPP stands for:

- **X** = Cross-platform
- **A** = Apache (Web server)
- **M** = MySQL (Relational DB)
- **P** = PHP
- **P** = Perl

 It's a bundle that gives you:

1. A **web server** (Apache),
2. A **database system** (MySQL),
3. A **PHP engine** to run `.php` files locally.

## What's Used for What?

Component	What it does	PHP Stack	Express Stack
PHP	Backend language (process forms, logic)	<code>&lt;?php ... ?&gt;</code> code	Not used
MySQL	SQL database for storing data	<code>SELECT</code> , <code>INSERT</code> queries	Possible, but rarely used
MongoDB	NoSQL database	Not commonly used	Stores data as JSON (BSON)
XAMPP	Development environment (includes Apache & MySQL)	Runs PHP & MySQL locally	Not used
Express.js	Backend server (JS-based)	Not used	Processes requests
phpMyAdmin	GUI for MySQL (bundled with XAMPP)	View/manage database	Not relevant

## Real-Life Analogy (Restaurant)



Role	Real World Example	In PHP + MySQL	In Express + MongoDB
Client	Customer (your browser)	Browser sends requests	Same
Waiter	PHP or Express	PHP fetches data and sends HTML	Express fetches JSON
Kitchen	Database	MySQL stores tables	MongoDB stores JSON documents
Restaurant	XAMPP (Local Host)	Apache+PHP+MySQL all run locally	Node.js runs Express

## What is `fetch_assoc()`?

`fetch_assoc()` is a method used in PHP with MySQLi (MySQL Improved Extension) to fetch a single row from a result set as an associative array.

### Basic Example:

php

 Copy  Edit

```
$result = $conn->query("SELECT * FROM users");

while ($row = $result->fetch_assoc()) {
    echo $row['name'] . "<br>";
}
```

### What's happening?



1. `$result` contains all rows returned from the query.
2. `fetch_assoc()` takes **one row at a time** and returns it as an **associative array**, where:
  - **keys** = column names ( `'id'` , `'name'` , `'email'` , etc.)
  - **values** = data in that row.

### Example Table (users)

id	name	email
1	Chayti	<a href="mailto:chayti@mail.com">chayti@mail.com</a>
2	Tanvir	<a href="mailto:tanvir@mail.com">tanvir@mail.com</a>

On first call to `fetch_assoc()` :



php

 Copy  Edit

```
$row = [
    "id" => 1,
    "name" => "Chayti",
    "email" => "chayti@mail.com"
];
```

Second call:

php

 Copy  Edit

```
$row = [
    "id" => 2,
    "name" => "Tanvir",
    "email" => "tanvir@mail.com"
];
```

After that, it returns `false` , and the loop ends.

## When to Use `fetch_assoc()`



- When you want **column names as array keys**
- It's easy to access: `$row['email']` , `$row['name']`
- Common in `while` loops when reading multiple rows

## Compared to Other Fetch Methods

Method	Returns	Example
<code>fetch_assoc()</code>	Associative array (column names)	<code>\$row['name']</code>
<code>fetch_row()</code>	Numeric array (index-based)	<code>\$row[1]</code>
<code>fetch_array()</code>	Both assoc and numeric	<code>\$row['name']</code> or <code>\$row[1]</code>
<code>fetch_object()</code>	Object with properties	<code>\$row-&gt;name</code>

### Final Mini Example:

php

 Copy  Edit

```
$sql = "SELECT id, name FROM users";
$result = $conn->query($sql);

while ($row = $result->fetch_assoc()) {
    echo "ID: " . $row['id'] . ", Name: " . $row['name'] . "<br>";
}
```

## Exercise:

=====

1. Create a form that calculates the **sum of two numbers** using PHP.
2. Write a PHP program that counts **vowels** in a string.
3. Create a PHP array of student grades and find the **highest score**.
4. Write a function that **checks if a number is prime**.
5. Take user input for **email** and validate format using regex.

3.	You are creating a "Device Registration" form for a company's new product line. The form requires a device name, a unique serial number, serial number confirmation and other necessary information. You must implement client-side validation to guide the user before submission.			
	a)	Write the HTML code for the registration form. Analyze and <b>apply</b> necessary information to the structure.	[5]	CO4
	b)	Write the CSS rules needed to provide visual feedback for the validation. Your CSS must include a rule to provide a <u>basic, readable layout</u> , error status with color to any input field that fails validation.	[4]	
	c)	<b>Describe</b> the logic and write the function structure for validating the form upon submission. You must ensure- <ul style="list-style-type: none"> <li>i. On form submit, prevent default and validate</li> <li>ii. Device Name: required → else show "Device Name is required."</li> <li>iii. Serial Format: must be <b>XXXX-YYYY</b> → else show "Invalid Serial Number format. Must be XXXX-YYYY"</li> <li>iv. Match Serial: confirm matches → else show "Serial Numbers do not match."</li> </ul>	[6]	

6.