**Documentation on backend of the edusphere**

### **\*Backend Functionalities**

**You’ll need to create PHP scripts for:**

**✅ User Authentication**

* **Register (register.php): Store user info securely with password hashing.**
* **Login (login.php): Verify credentials and start a session.**
* **Logout (logout.php): Destroy session and redirect to home.**

**✅ Course & Lesson Management**

* **List Courses (courses.php): Fetch from the courses table.**
* **List Lessons (lessons.php?course\_id=1): Fetch from lessons.**
* **Fetch Lesson Details (lesson.php?id=1).**
* **Enroll a user into a course.**

**✅ Quizzes**

* **Get Questions (quiz.php?lesson\_id=1).**
* **Submit Answers (submit\_quiz.php): Validate responses and store scores.**

**✅ Test**

* **Get Questions (test.php?lesson\_id=1).**
* **Submit Answers (submit\_quiz.php): Validate responses and store scores.**

**✅ Tracking Progress**

* **Mark Lessons as Complete (progress.php).**
* **Check Progress (get\_progress.php?user\_id=1).**

**✅ Certificate Generation**

* **If a user passes the final test (generate\_certificate.php), generate a PDF certificate.**

**\* Directory**

**eduSphere/**

**│── backend/**

**│ |----db.php (Database connection)**

**│ |---- auth/**

**│ │ |----register.php**

**│ │ |---- login.php**

**│ │ |----logout.php**

**│ |---- courses/**

**│ │ |---- get\_courses.php**

**│ │ |---- get\_lessons.php**

**| | |---- enroll.php (Enroll a user in a course)**

**│ |----quizzes/**

**│ │ |---- get\_quiz.php**

**│ │ |---- submit\_quiz.php**

**│ |---- tests/**

**│ │ |---- get\_test.php (Fetch test questions)**

**│ │ |---- submit\_test.php (Submit answers & check pass/fail)**

**│ |----progress/**

**│ │ |---- save\_progress.php**

**│ │ |---- get\_progress.php**

**│ |---- certificates/**

**│ │ |---- generate\_certificate.php**

**\*Database schema**

### **1. users (Stores user data)**

| **id** | **name** | **email** | **password** | **created\_at** |
| --- | --- | --- | --- | --- |
| **1** | **Abel Teferi** | **abel@example.com** | **hashed\_pass1** | **2025-03-19 12:00:00** |

### **2. courses (Stores available courses)**

| **id** | **title** | **description** | **thumbnail** | **created\_at** |
| --- | --- | --- | --- | --- |
| **1** | **CSS Basics** | **Learn CSS from scratch** | **css.jpg** | **2025-03-19 12:30:00** |

### **3. lessons (Lessons in each course)**

| **id** | **course\_id** | **title** | **content** |
| --- | --- | --- | --- |
| **1** | **1** | **Introduction to CSS** | **Basic CSS syntax** |

### **lesson\_videos (Stores video URLs for lessons)**

| **id** | **course\_id** | **lesson\_id** | **video\_url** |
| --- | --- | --- | --- |
| **1** | **1** | **1** | **css\_intro.mp4** |
| **2** | **1** | **2** | **css\_flexbox.mp4** |
| **3** | **2** | **3** | **git\_basics.mp4** |

### **5. enrollments (Tracks which users are enrolled in which courses)**

| **id** | **user\_id** | **course\_id** | **enrolled\_at** |
| --- | --- | --- | --- |
| **1** | **1** | **1** | **2025-03-19 13:00:00** |

### **6. quizzes (Stores quiz questions per lesson)**

| **id** | **lesson\_id** | **question** | **option\_a** | **option\_b** | **option\_c** | **option\_d** | **correct\_option** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **1** | **What does CSS stand for?** | **Cascading Style Syntax** | **Creative Style Sheets** | **Cascading Style Sheets** | **Computer Style Syntax** | **C** |
| **2** | **1** | **What property is used for Flexbox?** | **display** | **flex** | **justify-content** | **align-items** | **B** |

**✅ Each lesson can have multiple quiz questions (at least two).**

### **7. quiz\_results (Stores user quiz scores)**

| **id** | **user\_id** | **lesson\_id** | **score** | **completed\_at** |
| --- | --- | --- | --- | --- |
| **1** | **1** | **1** | **90** | **2025-03-19 14:00:00** |

### **8. tests (Final test questions for courses)**

| **id** | **course\_id** | **question** | **option\_a** | **option\_b** | **option\_c** | **option\_d** | **correct\_option** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **1** | **Which unit is relative to parent?** | **em** | **rem** | **px** | **%** | **A** |

### **9. test\_results (Stores final test results)**

| **id** | **user\_id** | **course\_id** | **score** | **passed** | **completed\_at** |
| --- | --- | --- | --- | --- | --- |
| **1** | **1** | **1** | **85** | **TRUE** | **2025-03-19 15:00:00** |

### **10. lesson\_progress (Tracks lesson completion)**

| **id** | **user\_id** | **course\_id** | **lesson\_id** | **completed\_at** |
| --- | --- | --- | --- | --- |
| **1** | **1** | **1** | **1** | **2025-03-19 15:30:00** |

### **11. certificates (Tracks certificate issuance)**

| **id** | **user\_id** | **course\_id** | **issued\_at** |
| --- | --- | --- | --- |
| **1** | **1** | **1** | **2025-03-19 16:00:00** |

1. **Comment table( to store comments about edusphere and fetch it to the hompage for the what students say about us section)**

| **id** | **user\_id** | **course\_id** | **issued\_at** | **Comment** |
| --- | --- | --- | --- | --- |
| **1** | **1** | **1** | **2025-03-19 16:00:00** | **My experience in edusphere was amazing……** |

1. **Course progress(tracks course completion)**

| **id** | **user\_id** | **course\_id** | **completed\_at** |
| --- | --- | --- | --- |
| **1** | **1** | **1** | **2025-03-19 15:30:00** |

what i didnt understand is that how do we integrate the php or the backend with the frontend? for let me give you a scenario: our homepage have a section called "What the Students Say About Us?" section this section initially have a static comments like the comment is always the same after each load but to make more authentic i thought to to create a table in the database to store all the comment that is given by the students and randomly display the comments on the "What the Students Say About Us?" section. so what is done in this situation i have the html, css and js of the home page but as i recall the php also returns an html document so how am i supposed to achieve that like tell me what is happening the files?

###### ChatGPT said:

Great question! This is where **backend and frontend integration** happens. Let's break it down **step by step** using your scenario:

**Scenario:**  
The homepage has a "What the Students Say About Us?" section. Currently, the comments are static, but you want to fetch **random testimonials from the database** and display them dynamically.

## ****How Backend and Frontend Work Together****

The frontend (**HTML, CSS, JS**) **requests** data from the backend (**PHP, MySQL**), and the backend **responds** with the requested data. This is done using **AJAX (JavaScript Fetch API)**.

## ****Step 1: Create a Table for Testimonials****

First, you need a database table to store student testimonials.

### ****Table:**** testimonials

| **id** | **user\_id** | **comment** | **created\_at** |
| --- | --- | --- | --- |
| 1 | 2 | "This platform is amazing!" | 2025-03-19 10:00 |
| 2 | 5 | "I learned so much, thanks!" | 2025-03-19 12:30 |
| 3 | 8 | "Great courses and easy to use" | 2025-03-19 14:00 |

## ****Step 2: Create a PHP File to Fetch Random Testimonials****

This PHP file will **retrieve random comments** from the database and return them as JSON.

### ****📄**** backend/get\_testimonials.php

php

CopyEdit

<?phprequire\_once '../db.php'; // Include the database connection

header('Content-Type: application/json');

$sql = "SELECT comment FROM testimonials ORDER BY RAND() LIMIT 3"; // Get 3 random comments$result = $conn->query($sql);

$testimonials = [];while ($row = $result->fetch\_assoc()) {

$testimonials[] = $row;

}

echo json\_encode($testimonials); // Send JSON response?>

✔ **Connects to the database (**db.php**)**  
✔ **Fetches 3 random comments**  
✔ **Returns them as JSON**

## ****Step 3: Fetch Testimonials in JavaScript****

Now, modify the **JavaScript file of your homepage** to **fetch testimonials from the backend**.

### ****📄**** frontend/homeP.js

js

CopyEdit

document.addEventListener("DOMContentLoaded", () => {

fetch("backend/get\_testimonials.php") // Fetch data from PHP

.then(response => response.json()) // Convert response to JSON

.then(data => {

const testimonialsContainer = document.getElementById("testimonials");

testimonialsContainer.innerHTML = ""; // Clear old comments

data.forEach(item => {

const p = document.createElement("p");

p.textContent = `"${item.comment}"`;

testimonialsContainer.appendChild(p);

});

})

.catch(error => console.error("Error fetching testimonials:", error));

});

✔ **Makes an AJAX request to** get\_testimonials.php  
✔ **Receives JSON data and updates the DOM dynamically**

## ****Step 4: Update the Homepage HTML****

Modify the **homepage HTML** to have a container where testimonials will be inserted.

### ****📄**** frontend/homeP.html

html

CopyEdit

<section id="testimonials">

<h2>What the Students Say About Us?</h2>

<div id="testimonials-container">

<!-- Testimonials will be added here dynamically -->

</div></section>

<script src="homeP.js"></script> <!-- Ensure this script is included -->

## ****How Everything Works Together****

1️⃣ **User opens the homepage (**homeP.html**).**  
2️⃣ homeP.js **runs and fetches testimonials** from backend/get\_testimonials.php.  
3️⃣ **PHP fetches random comments from MySQL** and returns them as **JSON**.  
4️⃣ **JavaScript receives the JSON**, processes it, and updates the **HTML dynamically**.  
5️⃣ **Testimonials now change randomly** each time the page loads! 🚀

## ****Summary****

✅ **Frontend (**homeP.html**,** homeP.js**)** → Sends request to backend and updates UI.  
✅ **Backend (**get\_testimonials.php**)** → Retrieves random data and returns JSON.  
✅ **Database (**testimonials **table)** → Stores student comments.

Now, your testimonials will be **dynamic and randomized** instead of being **static**! 🎉

Let me know if you need any adjustments! 🚀🔥

4o

##### You said:

so what makes it complex to me is that when you say echo json\_encode($testimonials); // Send JSON resp doesnt it change the homepage and dont we connect it to the html document using like <script src="">

###### ChatGPT said:

​

##### You said:

so what makes it complex to me is that when you say echo json\_encode($testimonials); // Send JSON resp doesnt it change the homepage and dont we connect it to the html document using like <script src="">

###### ChatGPT said:

I get what you mean! The confusion comes from how **PHP and JavaScript** work together. Let me explain it in a simple way.

### ****How Does**** echo json\_encode($testimonials); ****Work?****

No, it **does not change the homepage** directly.  
Instead, it **only sends data** (in JSON format) when another script (JavaScript) requests it.

Think of it like this:

* **PHP (**get\_testimonials.php**)** is like a **waiter** at a restaurant.
* **JavaScript (**homeP.js**)** is the **customer** who asks for a menu.
* **PHP does NOT create a whole new webpage**; it just gives **data (JSON)** to JavaScript when asked.
* **JavaScript then takes that data and updates the existing page.**

## ****Common Misconception: "Shouldn't PHP Be Connected Like a**** <script src=''>****?"****

No, because PHP is **not** a JavaScript file.  
We don't link get\_testimonials.php in HTML like this:

html

CopyEdit

<script src="backend/get\_testimonials.php"></script> ❌ (WRONG)

PHP **isn't JavaScript**—it's a backend script that **returns data**, not a script that runs in the browser.

## ****So, How Do We Use It?****

We use **JavaScript (AJAX or Fetch API)** to get the data from PHP **without reloading the page**.

### ****What Actually Happens****

1️⃣ **User loads** homeP.html → Homepage is displayed.  
2️⃣ **JavaScript (**homeP.js**) makes a request to** get\_testimonials.php.  
3️⃣ **PHP (**get\_testimonials.php**) fetches 3 random testimonials from the database**.  
4️⃣ **PHP sends the data back in JSON format**.  
5️⃣ **JavaScript receives the data and updates the testimonials section dynamically**.

### ****Example in Action****

Let’s say we visit backend/get\_testimonials.php directly in the browser.  
We might see something like this:

json

CopyEdit

[

{"comment": "This platform is amazing!"},

{"comment": "I learned so much, thanks!"},

{"comment": "Great courses and easy to use"}]

That’s **just raw data**—not a webpage.

But **JavaScript fetches that data and adds it to your homepage dynamically**.

## ****Final Code Recap****

### ****1️⃣**** backend/get\_testimonials.php ****(Backend - Returns JSON)****

php

CopyEdit

<?phprequire\_once '../db.php'; // Connect to DB

header('Content-Type: application/json'); // Set response as JSON

$sql = "SELECT comment FROM testimonials ORDER BY RAND() LIMIT 3"; // Get 3 random comments$result = $conn->query($sql);

$testimonials = [];while ($row = $result->fetch\_assoc()) {

$testimonials[] = $row;

}

echo json\_encode($testimonials); // Output JSON data?>

### ****2️⃣**** frontend/homeP.js ****(Frontend - Fetches & Updates UI)****

js

CopyEdit

document.addEventListener("DOMContentLoaded", () => {

fetch("backend/get\_testimonials.php") // Request data from PHP

.then(response => response.json()) // Convert response to JSON

.then(data => {

const testimonialsContainer = document.getElementById("testimonials-container");

testimonialsContainer.innerHTML = ""; // Clear old comments

data.forEach(item => {

const p = document.createElement("p");

p.textContent = `"${item.comment}"`;

testimonialsContainer.appendChild(p);

});

})

.catch(error => console.error("Error fetching testimonials:", error));

});

### ****3️⃣**** frontend/homeP.html ****(Frontend - Contains the Section)****

html

CopyEdit

<section id="testimonials">

<h2>What the Students Say About Us?</h2>

<div id="testimonials-container">

<!-- JavaScript will insert testimonials here -->

</div></section>

<script src="homeP.js"></script> <!-- Include the JavaScript file -->

## ****Final Thoughts****

* **PHP does NOT change the homepage**—it **only returns data** when asked.
* **JavaScript fetches that data** and updates the page **without reloading**.
* We **don't** use <script src="backend/get\_testimonials.php"> because PHP is **not** JavaScript.

This is how **backend and frontend communicate in real web apps!** 🚀🔥

Let me know if anything is still unclear! 😊

but when i learn database system i have learned it by microsoft sql management system, doesnt it affect me

###### ChatGPT said:

Since you learned **database systems** using **Microsoft SQL Server Management Studio (SSMS)**, switching to **MySQL** might feel a bit different at first. However, the core database concepts (tables, queries, relationships, normalization, indexing, transactions, etc.) **remain the same** across both systems.

You might need to adjust to **some syntax differences** (like LIMIT vs. TOP, AUTO\_INCREMENT vs. IDENTITY, etc.), but if you understand how databases work in general, you'll adapt quickly.

### ****Does Your Prior MSSQL Knowledge Affect You?****

✅ **No, it won’t be a big problem!**  
✅ **Your SQL skills transfer easily to MySQL** with only small adjustments.  
✅ **MySQL is simpler to set up** and works directly with XAMPP and PHP.

**What You Can Do to Transition Smoothly:**

1. **Use MySQL Workbench** – It provides a visual tool similar to SSMS.
2. **Practice MySQL Queries** – Try converting basic MSSQL queries into MySQL.
3. **Experiment with phpMyAdmin** – It's included in XAMPP and makes managing databases easier.
4. **Use Online Converters** – If you ever need to convert complex MSSQL queries to MySQL.

Since you're building your **eduSphere** project in **PHP**, **MySQL is the best choice** because:

* **PHP and MySQL work seamlessly together.**
* **XAMPP already includes MySQL.**
* **Hosting providers for web apps usually support MySQL, not MSSQL.**

So don’t worry! You’ll get used to MySQL **very quickly**, and I can help you along the way. 🚀💡

4o

##### You said:

ok so how can i use mysql for mssql there is microsoft sql management app can i use mysql on vs code to afffect the backend of my project

###### ChatGPT said:

Yes! You **don’t need Microsoft SQL Server Management Studio (SSMS)** anymore since you're switching to **MySQL**. Instead, you'll use **MySQL tools** with **VS Code** and **XAMPP** to manage your database and integrate it with PHP.

### ****How to Use MySQL for Your Backend in VS Code****

1️⃣ **Start MySQL Server in XAMPP**

* Open **XAMPP Control Panel**
* Start **Apache** (for PHP) and **MySQL** (for the database)

2️⃣ **Access MySQL Database**  
You can use **phpMyAdmin** (a graphical tool) or directly run MySQL queries in **VS Code**.

* Open a browser and go to **http://localhost/phpmyadmin/**
* Create a new database: eduSphere
* Inside the database, create tables (e.g., users, courses, progress, etc.).

3️⃣ **Install MySQL Extension in VS Code**  
If you want to run SQL queries inside **VS Code**, install the **"MySQL" extension**:

* Go to **Extensions** (Ctrl + Shift + X)
* Search **"MySQL"** and install a MySQL extension
* Use it to connect to your **localhost MySQL server** (host: localhost, user: root, password: "" by default).

4️⃣ **Write PHP Code to Connect to MySQL**  
Inside your backend/db.php file, write:

php

CopyEdit

<?php$host = "localhost";$user = "root";$password = "";$dbname = "eduSphere";

// Create connection$conn = new mysqli($host, $user, $password, $dbname);

// Check connectionif ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}?>

5️⃣ **Query the Database in PHP**  
Example: Fetching testimonials from the testimonials table in backend/get\_testimonials.php:

php

CopyEdit

<?phpinclude "db.php";

$sql = "SELECT name, comment FROM testimonials ORDER BY RAND() LIMIT 3";$result = $conn->query($sql);

$testimonials = [];

while ($row = $result->fetch\_assoc()) {

$testimonials[] = $row;

}

echo json\_encode($testimonials);?>

6️⃣ **Fetch Data in JavaScript (Frontend)** Inside your **home page JavaScript (homeP.js)**, fetch and display testimonials dynamically:

js

CopyEdit

fetch("http://localhost/eduSphere/backend/get\_testimonials.php")

.then(response => response.json())

.then(data => {

let testimonialSection = document.getElementById("testimonials");

testimonialSection.innerHTML = "";

data.forEach(testimonial => {

testimonialSection.innerHTML += `

<p><strong>${testimonial.name}</strong>: ${testimonial.comment}</p>

`;

});

})

.catch(error => console.error("Error:", error));

### ****What Happens When You Run the Project?****

1️⃣ **User visits the homepage** (homeP.html).  
2️⃣ **JavaScript (homeP.js) requests testimonials** from get\_testimonials.php.  
3️⃣ **PHP fetches data from MySQL and returns it as JSON**.  
4️⃣ **JavaScript updates the HTML dynamically** with real testimonials.

This is how PHP (backend), MySQL (database), and JavaScript (frontend) **work together** in your project! 🚀🔥

Would you like help with setting up database tables in **phpMyAdmin**?

how is it supposed can i just create the backend folder anywhere and just use vs code to everything

###### ChatGPT said:

Yes, you can create the **backend folder** anywhere on your computer, but there are some best practices when working with **XAMPP** and **VS Code** to make things easier.

Here's a general guideline:

### ****1. Folder Structure****

It's a good idea to organize your project folder in a way that's easy to manage and consistent with how XAMPP handles local server files. The recommended structure is to put your **backend** inside the **htdocs** folder (which is the root folder for web applications in XAMPP).

### ****Steps to Set Up Your Backend:****

#### ****a. Place Your Project in the**** htdocs ****Folder****

In your XAMPP directory, navigate to:

* 1. **Windows**: C:\xampp\htdocs\
  2. **Mac/Linux**: /Applications/XAMPP/htdocs/

Inside the htdocs folder, create a folder for your project (e.g., eduSphere).

Inside the eduSphere folder, create subdirectories for your backend:

plaintext

CopyEdit

eduSphere/

├── backend/

├── frontend/

└── index.php

#### ****b. Backend Directory****

* Your **backend** directory will contain all your PHP files for handling database queries, authentication, etc.
* You can organize your backend into separate folders like **auth**, **courses**, **quizzes**, etc. (as you already mentioned).

plaintext

CopyEdit

eduSphere/

├── backend/

│ ├── db.php

│ ├── auth/

│ │ ├── register.php

│ │ ├── login.php

│ ├── courses/

│ │ ├── get\_courses.php

│ │ ├── get\_lessons.php

│ └── quizzes/

│ ├── get\_quiz.php

│ └── submit\_quiz.php

└── frontend/

├── index.html

├── styles.css

└── scripts.js

#### ****c. Use VS Code for Development****

* Open the **entire project folder** (i.e., eduSphere/) in **VS Code**.
* You can edit both the **backend** (PHP) and **frontend** (HTML, CSS, JS) files inside VS Code.

#### ****d. Start Your Local Server****

* After setting up your files, you can start the **Apache** and **MySQL** services using the XAMPP control panel.
  + Open **XAMPP Control Panel**.
  + Click **Start** next to **Apache** (for serving your site) and **MySQL** (for database).
* Your project should now be accessible in your browser by visiting http://localhost/eduSphere (assuming you named the folder eduSphere).

#### ****e. Example: Accessing PHP Files****

**Backend Files** (e.g., db.php, get\_courses.php) can be accessed using URLs like:

* + http://localhost/eduSphere/backend/db.php
  + http://localhost/eduSphere/backend/courses/get\_courses.php

**Frontend Files** (e.g., index.html, styles.css) can be accessed as usual through the browser:

* + http://localhost/eduSphere/frontend/index.html

### ****2. Using VS Code for Everything****

**Edit Frontend and Backend**: You can edit both **frontend** (HTML, CSS, JS) and **backend** (PHP) files within **VS Code**. It provides syntax highlighting, autocompletion, and error-checking for both PHP and JavaScript, making development smooth.

**Use Extensions**: There are several VS Code extensions to help with PHP, MySQL, and front-end development.

* + For PHP: Install extensions like **PHP Intelephense**, **PHP Debug**, and **PHP Server**.
  + For MySQL: Use **SQL Server (mssql)** or **MySQL** extensions to manage and run queries directly from VS Code.

### ****3. How Does the Backend and Frontend Communicate?****

* The **frontend** (HTML, CSS, JS) makes HTTP requests (e.g., via AJAX or forms) to the **backend** PHP files.
* PHP will handle requests like fetching data from the database, processing it, and then sending the data back to the frontend as a response.
* In the case of the testimonials section, your frontend (JavaScript) would request data from the backend (PHP), and the backend would fetch it from the database, returning it to the frontend as JSON.

### ****Final Thoughts****

* **Yes, you can create the backend folder anywhere**, but for convenience with XAMPP, place it inside the **htdocs** directory so your server can easily access it.
* **VS Code** is perfect for this workflow, and it will help you manage both the backend and frontend seamlessly.
* The backend PHP files will handle database queries, while the frontend JavaScript will make requests to these files and display the data.