PHP Fundamentals & CRUD Application Implementation

A Practical Demonstration for Dynamic Web Development

Course: Web Technologies | **Instructor:** Dr. Amjad Farooq

Team: Ahmad Hassan, Huzaifa Arshad, Ahmad Sajjad, Atif Awan

Introduction to PHP: The Server-Side Powerhouse

Our objective is to explain the core concepts of PHP and demonstrate how these fundamentals enable the creation of a functional CRUD web application integrated with MySQL.

Hypertext Preprocessor

PHP stands for Hypertext Preprocessor. It is a robust server-side scripting language.

Dynamic Web Content

It is primarily used for building dynamic and interactive web pages and applications.

Seamless Integration

PHP code is typically embedded directly within HTML, allowing for easy mix of static and dynamic elements.

Why PHP Powers the Web: Key Advantages

PHP remains one of the most popular languages for web development due to its flexibility and ecosystem.

Integration Capability

Excellent compatibility with various databases (especially MySQL) and web servers.

Open Source & Community

Large, active community support and a vast library of reusable functions.

Ecosystem Adoption

It powers major Content Management Systems (CMS) like WordPress and frameworks like Laravel, simplifying large-scale development.

Low Learning Curve

Relatively easy for beginners to learn and deploy applications quickly.

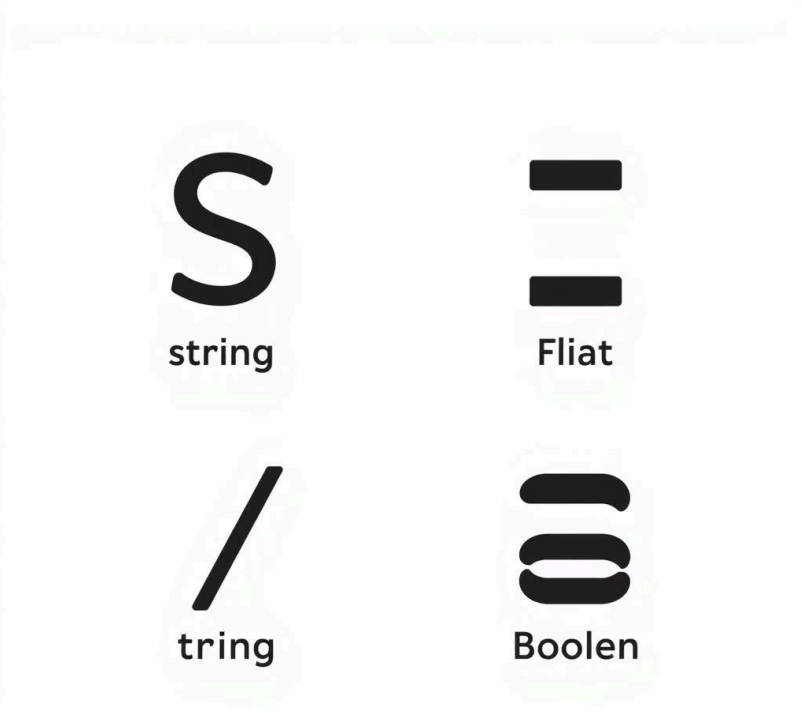
Fundamental Building Blocks: Syntax, Variables, & Types

Understanding how PHP handles data is crucial for writing any functional script.

Variables & Syntax

Variables in PHP are declared using the dollar sign ($) and are case-sensitive.

```
$name = "Huzaifa"; // String
$total = 150.50; // Float
```



Core Data Types

- **String:** Sequences of characters (e.g., names, sentences).
- **Integer:** Whole numbers (e.g., counts, IDs).
- **Float:** Numbers with decimal points (e.g., prices, averages).
- **Boolean:** True or False values for logic and decision-making.
- **Array:** Stores multiple values in a single variable.
- **NULL:** A variable with no assigned value.

> **Example Use of Operators:**
>
> `$total = $price * $qty;`
>
> This demonstrates the use of the `*` (Arithmetic) and `=` (Assignment) operators.

Control Structures & Loops

Manage program flow and automate repetition in your code.

Control Statements

Direct program flow based on conditions.

- **If/Elseif/Else:** Conditional execution of code.
- **Switch:** Tests a variable against multiple values.

Loop Statements

Repeat blocks of code efficiently.

- **For/While:** Iterate based on conditions or counts.
- **Foreach:** Iterate over array elements.

> **Loop Example:** Iterate through student data.
>
> ```
> foreach($students as $student) {
>  echo "Student Name: " . $student;
> }
> ```

Advanced Structures: Arrays, Functions, and OOP Basics

These concepts provide organization, reusability, and scalability for larger applications.

1

Arrays for Data Organization

Arrays manage multiple data points under a single variable name.

- **Indexed:** `$fruits = ["Apple", "Banana"];`
- **Associative:** `$marks = ["Ali"=>90, "Sara"=>85];`

2

Functions for Reusability

Functions encapsulate logic, making code clearer, easier to maintain, and highly reusable.

```
function greet($name) {
    return "Hello $name!";
}
```

3

Object-Oriented Programming (OOP)

Organizes code using **Classes** (blueprints) and **Objects** (instances).

Example: Class *Student* with *Properties* (Name, Email) and *Methods* (displayInfo()).

Database Setup and Connectivity

Our demonstration application uses MySQL to store and manage student data. A reliable connection is the gateway to data persistence.

1

Database Structure

- The database is named student_management.
- The primary table is **students**, with columns for `id`, `name`, and `email`.

2

PHP Connection Details

PHP uses the `mysqli` extension to establish the connection to the database.

Connection Code Snippet (db_connection.php):

```
$conn = new mysqli("localhost", "root", "", "student_management");
```

This command links our PHP application to the MySQL server.

3

Error Handling Best Practices

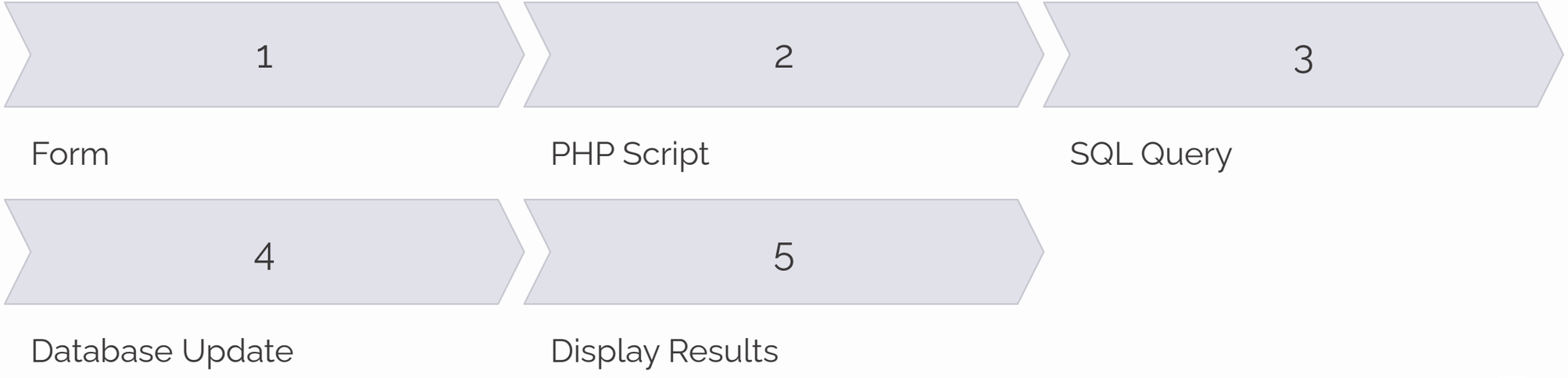
Always include error handling logic to ensure the application gracefully manages connection failures.

- Prevents data loss.
- Avoids security issues.
- Improves application robustness.

CRUD Operations: The Foundation of Web Applications

CRUD (Create, Read, Update, Delete) defines the four basic functions necessary for persistence in most applications.

Create	add_new.php	Allows users to **add** a new student record to the database (INSERT query).
Read	index.php	**Displays** a list of all current student records retrieved from the database (SELECT query).
Update	edit.php	Provides a form to **modify** details of an existing record (UPDATE query).
Delete	delete.php	**Removes** a specific student record from the table (DELETE query).

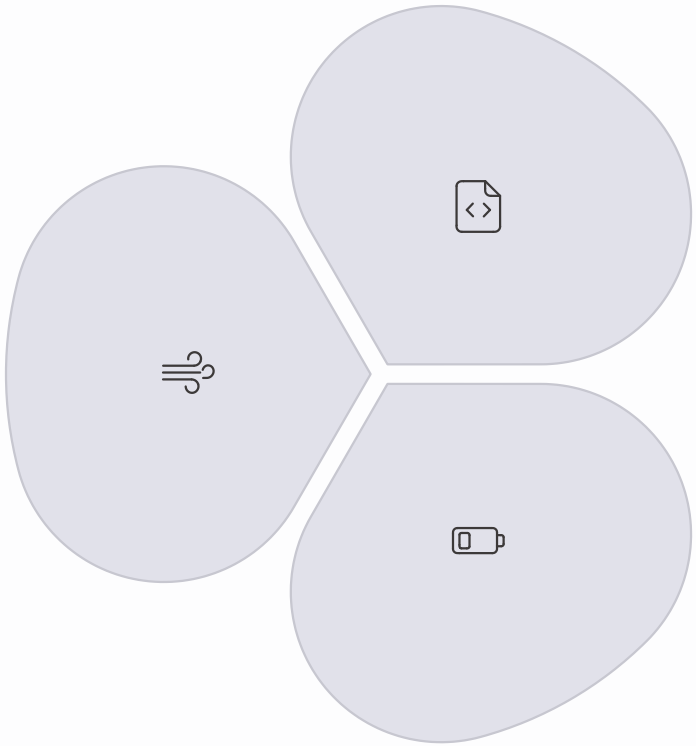


Conclusion & Live Demonstration Preview

The CRUD application integrates HTML forms, PHP processing scripts, and MySQL database management, showcasing essential full-stack development skills.

Dynamic Websites

PHP makes the creation of dynamic, database-driven sites accessible and efficient.



Core Application Foundation

CRUD operations are the fundamental requirements for any data-driven web application.

Full-Stack Integration

This project demonstrates the critical linkage between front-end (HTML), back-end (PHP), and database (MySQL).

Live Demo Plan: Seeing CRUD in Action

1. Demonstrate the "Create" process by adding a new student.
2. Verify the "Read" function by displaying the updated list of records.
3. Show the "Update" and "Delete" functionality on an existing student entry.

Thank You!

We are now open for questions and discussion.