FSD ASSIGNMENT - 4

By-Yashvardhan Tekavade
PB-15 Batch-1
Panel-1 TY-CSF

Aim: Write server-side script in PHP to perform form validation and create database application using PHP and MySQL to perform insert, update, delete and search operations.

Objectives:

1. To understand Server-side Scripting.
2. To learn database connectivity using PHP-MySQL.
3. To perform insert, update, delete and search operations on database.

Theory

1. **PHP Architecture.**

PHP (Hypertext Preprocessor) is a server-side scripting language commonly used for web development. Its architecture can be divided into several key components and layers, each serving a specific purpose in the execution of PHP web applications. Here is an overview of the PHP architecture:

- **Web Server:** PHP applications run on web servers, such as Apache, Nginx, or Microsoft Internet Information Services (IIS). These web servers are responsible for receiving HTTP requests from clients (browsers) and forwarding them to the PHP interpreter for processing.
- **PHP Interpreter:** The PHP interpreter is the core component of the PHP runtime environment. It processes PHP code embedded within HTML or standalone PHP files. When a request is received by the web server and contains PHP code, the web server invokes the PHP interpreter to execute the code.
- **PHP Code:** PHP code can be embedded directly within HTML files using tags like <?php ... ?> or <?= ... ?>, or it can reside in separate .php files. PHP is a server-side language, so it runs on the server, processes the code, and generates HTML or other output to be sent back to the client's browser.
- **PHP Extensions:** PHP supports a wide range of extensions and libraries that provide additional functionality. These extensions can be bundled with PHP or added separately. Common extensions include MySQLi for database connectivity, cURL for making HTTP requests, and GD for image manipulation.
- **Database:** PHP often interacts with databases to store, retrieve, and manipulate data. Popular database management systems used with PHP include MySQL, PostgreSQL, SQLite, and Oracle. PHP provides database-specific extensions and functions for connecting to and working with databases.

2.Steps for Database connectivity in PHP

To establish database connectivity in PHP, you typically need to follow these steps:

- **Choose a Database Management System (DBMS):** Before connecting to a database in PHP, you need to decide which database management system you want to use. Common options include MySQL, PostgreSQL, SQLite, Oracle, and Microsoft SQL Server. Ensure that your chosen DBMS is installed and properly configured on your server.
- **Install PHP:** Make sure that PHP is installed on your web server. PHP is commonly used for server-side scripting and database connectivity.
- **Install Database Extension:** Depending on your chosen DBMS, you may need to install a PHP extension that allows PHP to communicate with the database. For example, if you're using MySQL, you'll need the MySQLi or PDO extension. These extensions are typically included in PHP by default, but you may need to enable them in your PHP configuration.
- **Create a Database:** Use your DBMS's administration tools or command-line interface to create a database. Define tables, fields, and relationships as needed for your application.
- **Gather Database Information:** You will need the following information to connect to the database:
    - Database server hostname or IP address.
    - Database name.
    - Database username and password with appropriate permissions.

FAQs

1. **What are the advantages of Server-side Scripting?**

Server-side scripting offers several advantages when developing web applications and websites. Here are some of the key benefits:

- Security: Server-side scripting allows sensitive logic and data to be processed on the server, keeping it hidden from the client-side code and potentially malicious users. This helps protect against common security threats such as Cross-Site Scripting (XSS) and SQL Injection, as sensitive data and operations are not exposed to the client.
- Data Privacy: With server-side scripting, you can control what data is shared with clients. You can restrict the amount and type of data sent to the client, ensuring that confidential information is kept secure on the server.
- Performance: Server-side scripts can perform resource-intensive operations on the server, reducing the load on the client's device. This can lead to faster page loading times, especially for complex or data-intensive applications, as the server handles tasks like database queries and data processing.

- Scalability: Server-side applications can be more easily scaled by adding more server resources or employing load balancing techniques. This scalability ensures that your application can handle increased traffic and maintain good performance as your user base grows.
- Code Reusability: Server-side scripting promotes code reusability by allowing you to separate server-side logic from the presentation layer. You can reuse server-side code for various parts of your application or even across different projects, leading to more efficient development and easier maintenance.
- Data Manipulation: Server-side scripting languages offer powerful libraries and frameworks for data manipulation, transformation, and validation. This makes it easier to work with databases, process data, and generate dynamic content.
- Cross-Browser Compatibility: Server-side code can generate HTML and other client-side code that is compatible with various web browsers. This helps ensure consistent rendering and functionality across different browsers and platforms.
- Database Interaction: Server-side scripting enables direct interaction with databases, allowing you to store, retrieve, and manipulate data efficiently. This facilitates the creation of dynamic, data-driven web applications.

2. What is XAMPP and phpMyAdmin?

XAMPP (Cross-Platform, Apache, MySQL, PHP, and Perl):

- **Purpose:** XAMPP is a free, open-source web server solution designed to facilitate local web development and testing. It bundles together several key components necessary for web development, including the Apache web server, MySQL (or MariaDB) database server, PHP, and Perl.
- **Features:**
    - **Apache:** XAMPP includes the Apache HTTP Server, which is one of the most widely used web servers in the world. It allows developers to host and serve web content locally.
    - **MySQL/MariaDB:** XAMPP provides a database server (either MySQL or MariaDB) for storing and managing data. This is crucial for web applications that require database functionality.
    - **PHP:** XAMPP includes PHP, a server-side scripting language widely used for web development. Developers can run PHP scripts and create dynamic web applications.
    - **Perl:** Though less common in web development than PHP, XAMPP also includes Perl, a general-purpose scripting language.

phpMyAdmin:

- **Purpose:** phpMyAdmin is a web-based administration tool for managing MySQL or MariaDB databases. It provides a user-friendly interface for performing tasks such as creating databases, tables, and user accounts; importing and exporting data; executing SQL queries; and more.
- **Features:**
    - **Database Management:** phpMyAdmin allows you to create, modify, and delete databases and database objects like tables, views, and stored procedures.
    - **Data Manipulation:** You can insert, update, and delete data records in database tables using the graphical interface.
    - **SQL Querying:** phpMyAdmin provides a SQL query editor, enabling you to write and execute SQL queries directly.

3. What are the two ways to connect to a database in PHP

In PHP, you can connect to a database using two primary methods: the MySQLi extension (MySQL Improved) and the PDO (PHP Data Objects) extension. Both methods allow you to interact with various database management systems, including MySQL, PostgreSQL, SQLite, Oracle, and more. Here's an overview of each approach:

MySQLi (MySQL Improved):

MySQLi is a PHP extension specifically designed for interacting with MySQL databases. It provides a procedural and object-oriented API for database connectivity.

PDO (PHP Data Objects):

PDO is a more generic database abstraction layer for PHP that provides a consistent API for working with various database management systems.

Sample Problem Statements:

PHP CRUD Operations

1.Student can create a PHP form or use existing/ implemented HTML form for Student's Registration System with the fields mentioned: First name,Last name, Roll No/ID, Password, Confirm Password,Contact number and perform following operations

1.Insert student details -First name,Last name, Roll No/ID, Password, Confirm Password,Contact number
2.Delete the Student records based on Roll no/ID
3.Update the Student details based on Roll no/ID- Example students can update their contact details based on searching the record with Roll no.
4.Display the Updated student details or View the Students record in tabular format.

Apply Form Validation on the necessary fields using PHP/Javascript

2. Student can create a PHP form or use existing/ implemented HTML form for Library Management System with the fields mentioned: Book name, ISBN No, Book title, Author name , Publisher name and perform following operations

1.Insert Book details -Book name, ISBN No, Book title, Author name, Publisher name
2.Delete the Book records based on ISBN No
3.Update the Book details based on ISBN No- Example students can update wrong entered book details based on searching the record with ISBN No.
4.Display the Updated Book details or View the Book Details records in tabular format.

Apply Form Validation on the necessary fields using PHP/Javascript

3. Student can create a PHP form or use existing/ implemented HTML form for Employee Management System with the fields mentioned: Employee name, Employee ID, Department_name, Phone number , Joining Date and perform following operations

1.Insert Employee details -Employee name, Employee ID, Department_name, Phone number , Joining Date
2.Delete the Employee records based on Employee ID
3.Update the Employee details based on Employee ID- Example students can update Employee details based on searching the record with Employee ID.
4.Display the Updated Employee details or View the Employee Details records in tabular format.

Apply Form Validation on the necessary fields using PHP/Javascript

4. Student can create a PHP form or use existing/ implemented HTML form for Flight Booking Management System with the fields mentioned: Passenger name, From, to, date,Departure date,Arrival date, Phone number , Email ID and perform following operations

1.Insert Passenger details -Passenger name, From, to, date,Departure date,Arrival date, Phone number , Email ID
2.Delete the Passenger records based on Phone Number

3.Update the Passenger details based on Phone Number - Example students can update Flight Booking details based on searching the record with Phone Number.
4.Display the Updated Flight Booking details or View the Flight Booking Details records in tabular format.

Apply Form Validation on the necessary fields using PHP/Javascript.

Technologies Student Should Use:
XAMPP
PHP for Server-side Scripting
MySQL as a backend Database

Output:

SQL code:

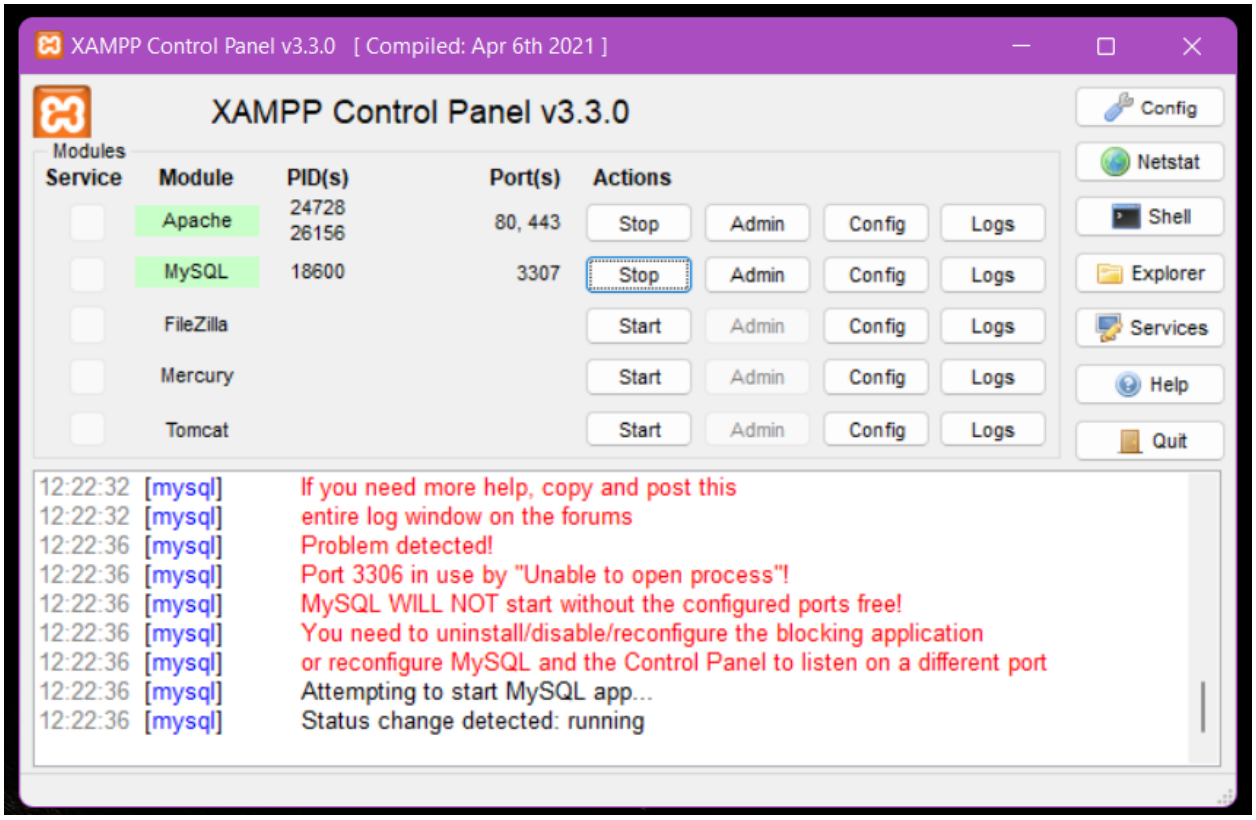
```sql
CREATE TABLE employees (
  id INT AUTO_INCREMENT PRIMARY KEY,
  employee_name VARCHAR(255) NOT NULL,
  employee_id VARCHAR(255) NOT NULL,
  department_name VARCHAR(255) NOT NULL,
  phone_number VARCHAR(15) NOT NULL,
  joining_date DATE NOT NULL
);
```

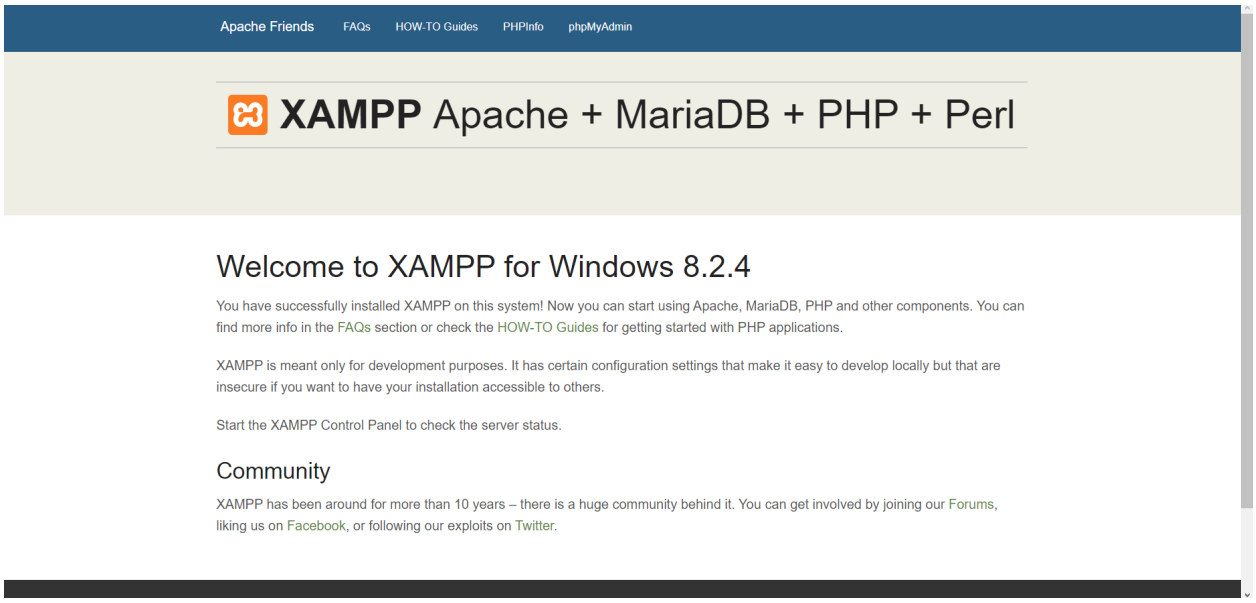
PHP code:

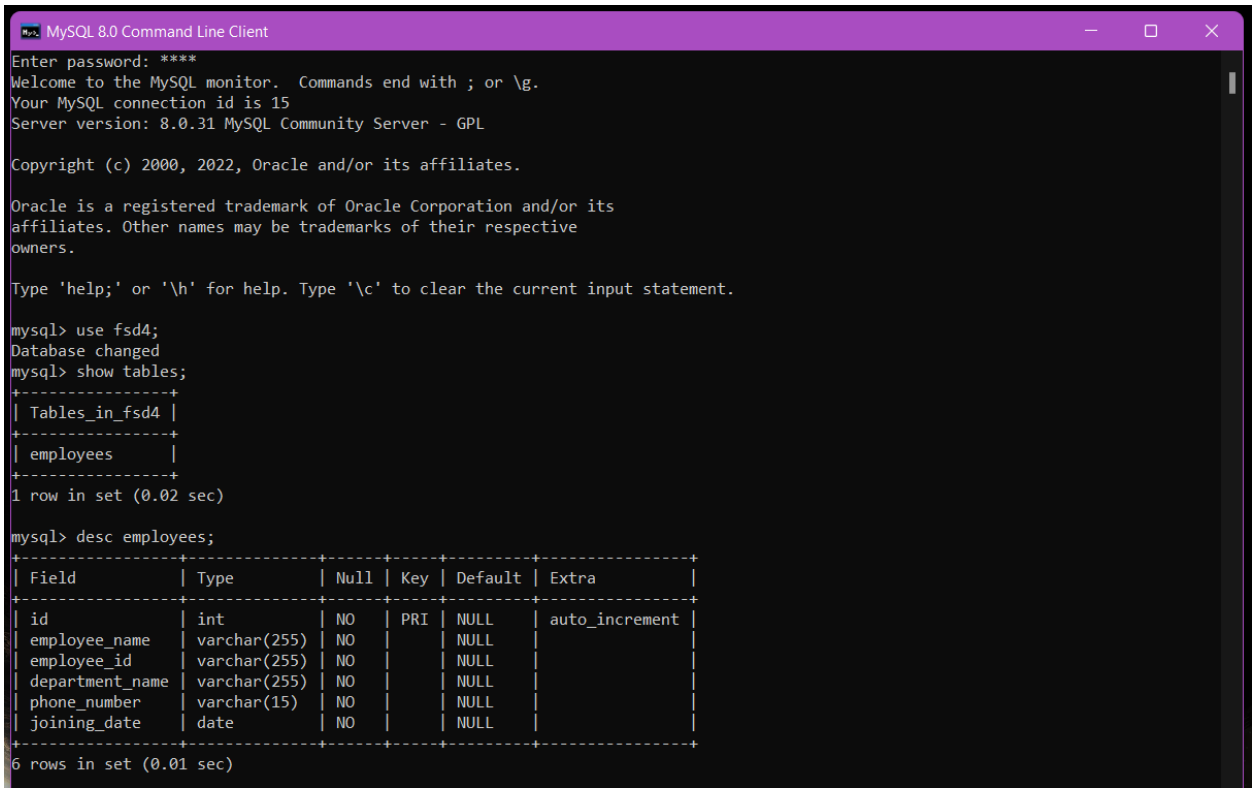
```php
<?php
$conn = new mysqli("localhost", "root", "1234", "fsd4");

if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $employeeName = $_POST["employee_name"];
  $employeeID = $_POST["employee_id"];
```

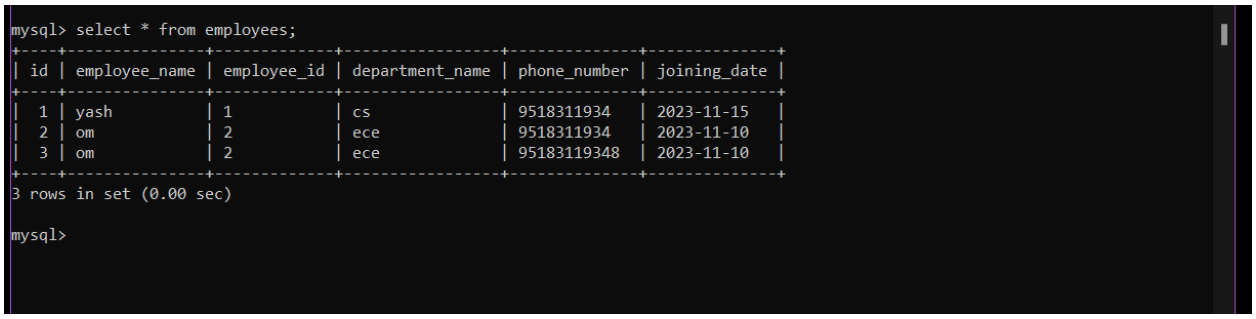
```
  $departmentName = $_POST["department_name"];
  $phoneNumber = $_POST["phone_number"];
  $joiningDate = $_POST["joining_date"];

  $sql = "INSERT INTO employees (employee_name, employee_id,
department_name, phone_number, joining_date) VALUES
('$employeeName', '$employeeID', '$departmentName',
'$phoneNumber', '$joiningDate')";

  if ($conn->query($sql) === TRUE) {
    echo "Data saved successfully";
  } else {
    echo "Error: " . $sql . "<br>" . $conn->error;
  }
}

$conn->close();
?>1234
```

Screenshots of the output to be attached.






MySQL 8.0 Command Line Client
Enter password: ****
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.31 MySQL Community Server - GPL
Copyright (c) 2000, 2022, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> use fsd4;
Database changed
mysql> show tables;
+----------------+
| Tables_in_fsd4 |
+----------------+
| employees |
+----------------+
1 row in set (0.02 sec)
mysql> desc employees;
+-----------------+--------------+------+-----+---------+----------------+
| Field | Type | Null | Key | Default | Extra |
+-----------------+--------------+------+-----+---------+----------------+
id	int	NO	PRI	NULL	auto_increment
employee_name	varchar(255)	NO		NULL	
employee_id	varchar(255)	NO		NULL	
department_name	varchar(255)	NO		NULL	
phone_number	varchar(15)	NO		NULL	
joining_date	date	NO		NULL	
+-----------------+--------------+------+-----+---------+----------------+
6 rows in set (0.01 sec)


mysql> select * from employees;
+----+---------------+-------------+-----------------+--------------+--------------+
| id | employee_name | employee_id | department_name | phone_number | joining_date |
+----+---------------+-------------+-----------------+--------------+--------------+
1	yash	1	cs	9518311934	2023-11-15
2	om	2	ece	9518311934	2023-11-10
3	om	2	ece	95183119348	2023-11-10
+----+---------------+-------------+-----------------+--------------+--------------+
3 rows in set (0.00 sec)
mysql>

Employee Management System

Insert Employee Record

Employee Name:

Employee ID:

Department Name:

Phone Number:

Joining Date:

dd-mm-yyyy

Insert Employee

Delete Employee Record

Employee ID to Delete:

Delete Employee

Update Employee Record

Employee ID to Update:

Updated Department Name:

Updated Phone Number:

Update Employee

Display Employee Records

Display Employee Records