CHƯƠNG 4: WEBSITE HƯỚNG DỮ LIfiU

1. Lý thuyết Cốt lõi (Khái niệm)

Để PHP "nói chuyện" với MySQL (hoặc bất kỳ CSDL nào), chúng ta sử dụng một giao diện (interface) chuẩn gọi là **PDO (PHP Data Objects)**.

Luồng làm việc với PDO luôn gồm các bước:

1. **Kết nối (Connect):** Tạo một đối tượng PDO mới, cung cấp cho nó "chuỗi kết nối" (DSN), username và password của CSDL.
2. **Chuẩn bị (Prepare):** Viết câu lệnh SQL (như SELECT * FROM users WHERE id = ?). Dấu ? là một **placeholder** (trình giữ chỗ).
3. **Thực thi (Execute):** "Bind" (gắn) giá trị thật (ví dụ: $id = 5) vào placeholder ? và thực thi câu lệnh.
4. **Lấy kết quả (Fetch):** Nếu là câu SELECT, dùng fetch() (lấy 1 dòng) hoặc fetchAll() (lấy tất cả) để nhận dữ liệu.

Tại sao dùng "Prepared Statements" (dấu ?)? Đây là cách **bắt buộc** để chống lại một kiểu tấn công cực kỳ phổ biến tên là **SQL Injection**. Tuyệt đối **không bao giờ** viết code bằng cách cộng chuỗi trực tiếp như: $sql = "SELECT * FROM users WHERE username = '" . $_POST['user'] . "'"; (CỰC KỲ NGUY HIỂM!)

2. Nhiệm vụ Thực hành (BẮT BUỘC)

Kịch bản: Xây dựng một trang "Danh sách sinh viên" đơn giản. Trang này cho phép bạn:

1. Thêm sinh viên mới vào CSDL (Dùng INSERT).
2. Hiển thị toàn bộ sinh viên đang có trong CSDL (Dùng SELECT).

A. Thiết lập Ban đầu (Bắt buộc)

1. Mở **phpMyAdmin**.
2. Tạo một CSDL mới tên là cse485_web.
3. Chọn CSDL cse485_web, mở tab **SQL** và chạy lệnh sau để tạo bảng:

SQL

```
CREATE TABLE sinhvien (
  id INT AUTO_INCREMENT PRIMARY KEY,
  ten_sinh_vien VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL,
  ngay_tao TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

B. Code Khởi đầu (Starter Code):

Tạo 1 tệp chapter4.php trong thư mục htdocs của XAMPP:

PHP

```
        <button type="submit">Thêm</button>
    </form>

    <h2>Danh Sách Sinh Viên (Chủ đề 4.2)</h2>
    <table>
        <tr>
            <th>ID</th>
            <th>Tên Sinh Viên</th>
            <th>Email</th>
            <th>Ngày Tạo</th>
        </tr>
        <?php
        // TODO 9: Dùng vòng lặp (ví dụ: while) để duyệt qua kết quả
$stmt_select
        // Gợi ý: while ($row = $stmt_select->fetch(PDO::FETCH_ASSOC)) { ... }

            // TODO 10: In (echo) các dòng <tr> và <td> chứa dữ liệu $row
            // Gợi ý: echo "<tr>";
            // Gợi ý: echo "<td>" . htmlspecialchars($row['id']) . "</td>";
            // (htmlspecialchars là để bảo mật, tránh lỗi XSS - sẽ học ở Chffơng
9)

        // Đóng vòng lặp

        ?>
    </table>
</body>
</html>
```

3. Yêu cầu Bằng chứng (Proof of Work)

Bạn phải nộp lại 2 bằng chứng sau:

A. Code đã hoàn thiện: Dán (paste) toàn bộ code của tệp chapter4.php mà bạn đã hoàn thiện.

chapter4.php

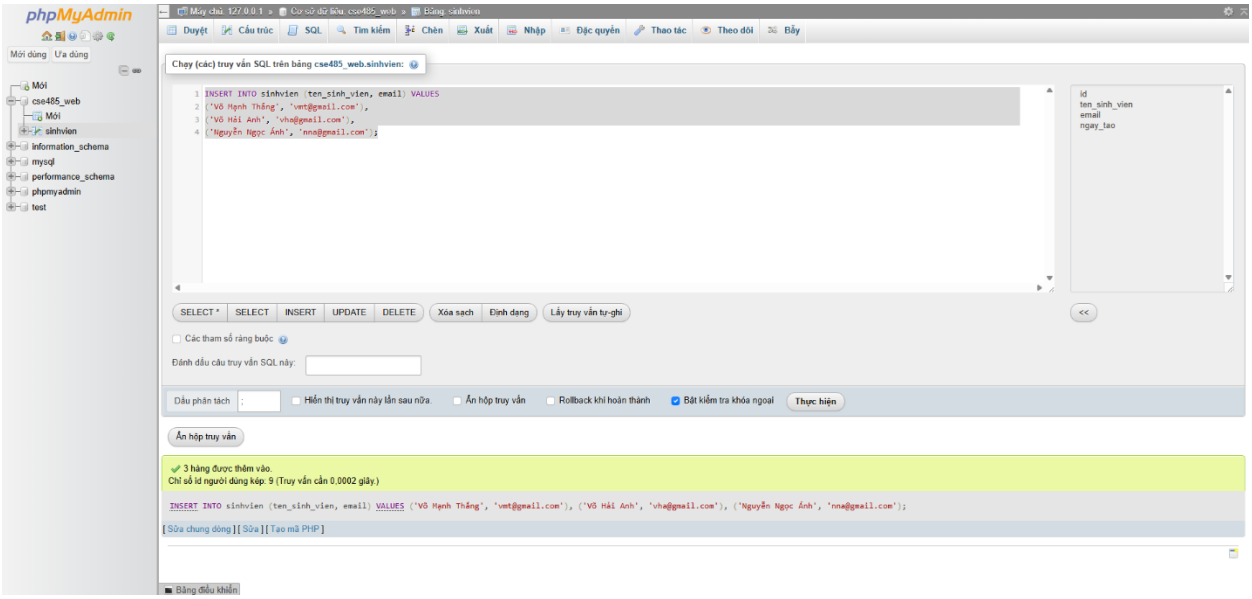
```php
<?php
// === THÔNG SỐ KẾT NỐI CSDL ===
$host = 'localhost'; // Hoặc 127.0.0.1
$db   = 'cse485_web';
$user = 'root';      // Tên user mặc định của XAMPP
$pass = '';          // Mật khẩu mặc định của XAMPP

// === KẾT NỐI PDO ===
try {
    $pdo = new PDO("mysql:host=$host;dbname=$db;charset=utf8", $user, $pass);
    // Thiết lập PDO để báo lỗi (rất quan trọng khi debug)
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    // echo "Kết nối thành công!"; // (Bỏ comment để test)
} catch (PDOException $e) {
    die("Kết nối thất bại: " . $e->getMessage());
}

// === LOGIC THÊM SINH VIÊN (XỬ LÝ FORM POST) ===
// TODO 2: Kiểm tra xem form đã được gửi đi (method POST) và có 'ten_sinh_vien' không
// Chúng ta kiểm tra cả 'ten_sinh_vien' và 'email' để đảm bảo dữ liệu cần thiết đã được gửi.
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['ten_sinh_vien'], $_POST['email'])) {

    // TODO 3: Lấy dữ liệu 'ten_sinh_vien' và 'email' từ $_POST
    // Sử dụng trim() để loại bỏ khoảng trắng dư thừa
    $ten = trim($_POST['ten_sinh_vien']);
    $email = trim($_POST['email']);

    // Kiểm tra tính hợp lệ cơ bản
    if (!empty($ten) && !empty($email)) {
        try {
            // TODO 4: Viết câu lệnh SQL INSERT với Prepared Statement (dùng dấu ?)
            $sql = "INSERT INTO sinhvien (ten_sinh_vien, email) VALUES (?, ?)";

            // TODO 5: Chuẩn bị (prepare) và thực thi (execute) câu lệnh
            $stmt = $pdo->prepare($sql);
            // Thực thi với mảng tham số (giúp ngăn chặn SQL Injection)
            $stmt->execute([$ten, $email]);

            // TODO 6: (Tùy chọn) Chuyển hướng về chính trang này để "làm mới"
            // Việc chuyển hướng POST/Redirect/GET (PRG) giúp tránh việc user F5 sẽ gửi lại form
            header('Location: chapter4.php');
            exit; // Luôn gọi exit sau header('Location: ...')
        } catch (PDOException $e) {
            $error_message = "Lỗi khi thêm sinh viên: " . $e->getMessage();
        }
    } else {
        $error_message = "Vui lòng điền đầy đủ Tên sinh viên và Email.";
    }
}

// === LOGIC LẤY DANH SÁCH SINH VIÊN (SELECT) ===
// TODO 7: Viết câu lệnh SQL SELECT *
$sql_select = "SELECT * FROM sinhvien ORDER BY ngay_tao DESC";

// TODO 8: Thực thi câu lệnh SELECT (không cần prepare vì không có tham số)
$stmt_select = $pdo->query($sql_select);

// TODO 9: Lấy toàn bộ dữ liệu (tùy chọn) hoặc để vòng lặp fetch ở HTML
// $sinh_vien_list = $stmt_select->fetchAll(PDO::FETCH_ASSOC);
// Chúng ta sẽ dùng vòng lặp fetch trong HTML để tối ưu bộ nhớ.
?>
```

```php
<!DOCTYPE html>
<html lang="vi">
<head>
    <meta charset="UTF-8">
    <title>Danh Sách Sinh Viên - CSE485</title>
    <style>
        body { font-family: Arial, sans-serif; margin: 20px; }
        form { margin-bottom: 30px; padding: 15px; border: 1px solid #ccc; border-radius: 5px; }
        input[type="text"], input[type="email"] { padding: 8px; margin-right: 10px; border: 1px solid #ddd; border-radius: 3px; }
        button { padding: 10px 15px; background-color: #007bff; color: white; border: none; border-radius: 5px; cursor: pointer; }
        table { width: 100%; border-collapse: collapse; margin-top: 20px; }
        th, td { border: 1px solid #ddd; padding: 10px; text-align: left; }
        th { background-color: #f2f2f2; }
        .error { color: red; font-weight: bold; margin-bottom: 15px; }
    </style>
</head>
<body>

    <h1>Quản Lý Sinh Viên Cơ Bản (PDO)</h1>

    <?php if (isset($error_message)): ?>
        <p class="error">LỖI: <?php echo $error_message; ?></p>
    <?php endif; ?>

    <h2>➕ Thêm Sinh Viên Mới (Chủ đề 4.3)</h2>
    <form action="" method="post">
        <label for="ten_sinh_vien">Tên sinh viên:</label>
        <input type="text" id="ten_sinh_vien" name="ten_sinh_vien" required>

        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required>

        <button type="submit">Thêm</button>
    </form>

    <hr>
```

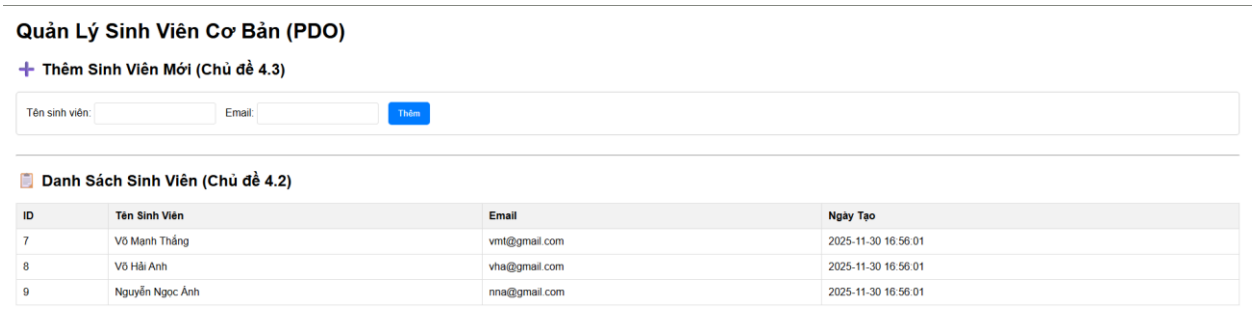
```php
    <h2>📄 Danh Sách Sinh Viên (Chủ đề 4.2)</h2>

    <table>
        <thead>
            <tr>
                <th>ID</th>
                <th>Tên Sinh Viên</th>
                <th>Email</th>
                <th>Ngày Tạo</th>
            </tr>
        </thead>
        <tbody>
            <?php
            // TODO 10: Lặp qua từng dòng dữ liệu và in ra
            // Dùng while để duyệt từng dòng fetch, tiết kiệm bộ nhớ hơn fetchAll()
            while ($row = $stmt_select->fetch(PDO::FETCH_ASSOC)) {
                // $row lúc này chứa dữ liệu của một sinh viên dưới dạng mảng kết hợp (Associative Array)
            ?>
                <tr>
                    <td><?php echo htmlspecialchars($row['id']); ?></td>
                    <td><?php echo htmlspecialchars($row['ten_sinh_vien']); ?></td>
                    <td><?php echo htmlspecialchars($row['email']); ?></td>
                    <td><?php echo htmlspecialchars($row['ngay_tao']); ?></td>
                </tr>
            <?php
            } // Đóng vòng lặp while

            // Tùy chọn: Kiểm tra nếu không có sinh viên nào
            if ($stmt_select->rowCount() === 0) {
                echo '<tr><td colspan="4">Chưa có sinh viên nào trong cơ sở dữ liệu.</td></tr>';
            }
            ?>
        </tbody>
    </table>

</body>
</html>
```

B. Ảnh chụp màn hình Kết quả (BẮT BUỘC CẢ 2 ẢNH):

1. **Ảnh 1 (phpMyAdmin):** Chụp màn hình tab "Browse" (Duyệt) của bảng sinhvien trong phpMyAdmin, cho thấy bạn đã INSERT thành công ít nhất 2-3 sinh viên.



2. **Ảnh 2 (Trình duyệt Web):** Chụp ảnh màn hình trang chapter4.php của bạn, hiển thị đúng 2-3 sinh viên mà bạn vừa thêm (chứng minh SELECT thành công).



(Dán Code A và Ảnh B1, B2 của bạn vào đây)

4. Câu hỏi Phản biện (Bắt buộc)

Sau khi hoàn thành Phần 2 & 3, hãy đặt 01 câu hỏi tư duy.

Câu hỏi của tôi là: Tại sao việc cộng chuỗi INSERT INTO sinhvien (ten) VALUES (':ten') lại nguy hiểm, và tại sao cách dùng execute([:ten]) (Prepared Statement) lại an toàn hơn?

Trả lời:

- Việc cộng chuỗi INSERT INTO sinhvien (ten) VALUES (':ten') nguy hiểm vì nó dễ bị tấn công SQL Injection, cho phép kẻ xấu chèn và thực thi lệnh SQL độc hại thông qua dữ liệu đầu vào
- Dùng Prepared Statement ($stmt->execute([$ten])) an toàn hơn vì:
  - Nó tách biệt hoàn toàn Lệnh SQL (mẫu lệnh có dấu ?) và Dữ liệu đầu vào
  - CSDL luôn coi dữ liệu ($ten), dù có chứa ký tự đặc biệt, là văn bản thuần túy, không bao giờ là một phần của lệnh SQL, từ đó ngăn chặn tấn công

5. Kết nối Đánh giá (Rất quan trọng)

Kỹ năng kết nối CSDL bằng **PDO** (bao gồm INSERT và SELECT) là kỹ năng quan trọng nhất trong khối kiến thức PHP thuần.

Bạn sẽ vận dụng trực tiếp PHT này để hoàn thành **Bài tập trên lớp (Phần PHP)**, chiếm **20%** tổng điểm, dự kiến vào **Tuần 5**. Nắm vững PDO bây giờ cũng sẽ giúp bạn hiểu tại sao **Eloquent ORM** (Chương 8) lại mạnh mẽ và tiện lợi đến vậy.