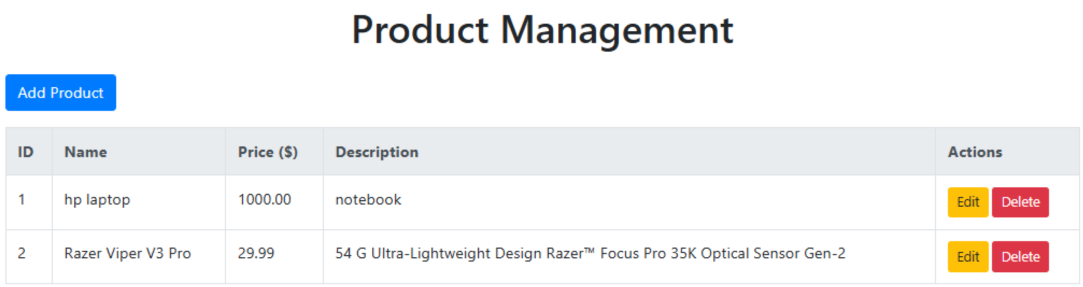
Implementing a CRUD (Create, Read, Update, Delete) application using PHP with PDO

Result:

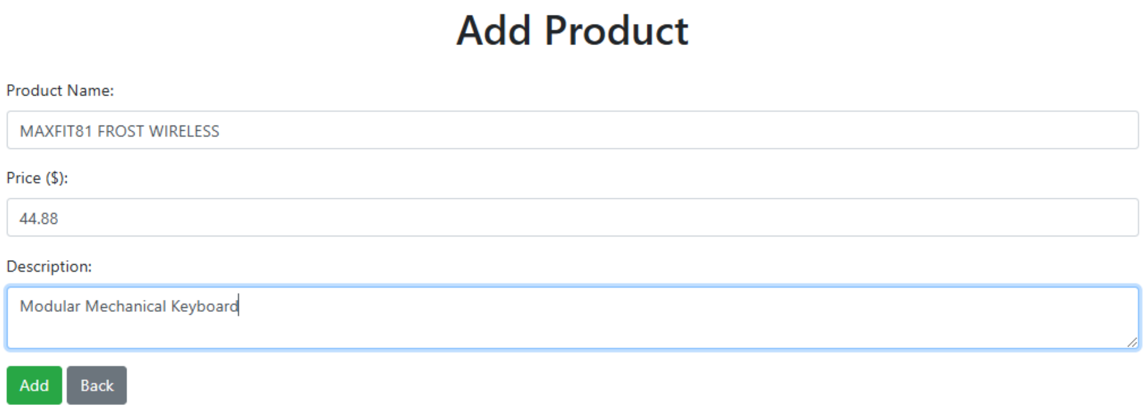
Main display page:



Product Management

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	notebook	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete

Adding product (Create):



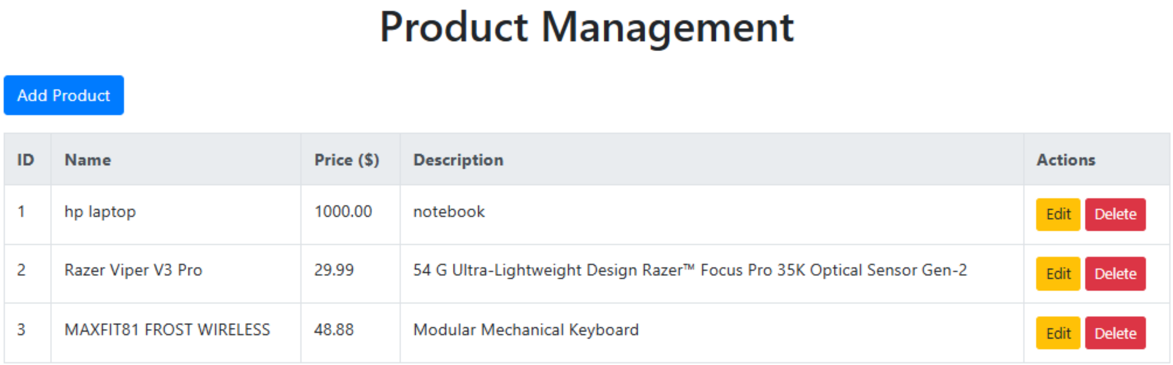
Add Product

Product Name:
MAXFIT81 FROST WIRELESS

Price ($):
44.88

Description:
Modular Mechanical Keyboard

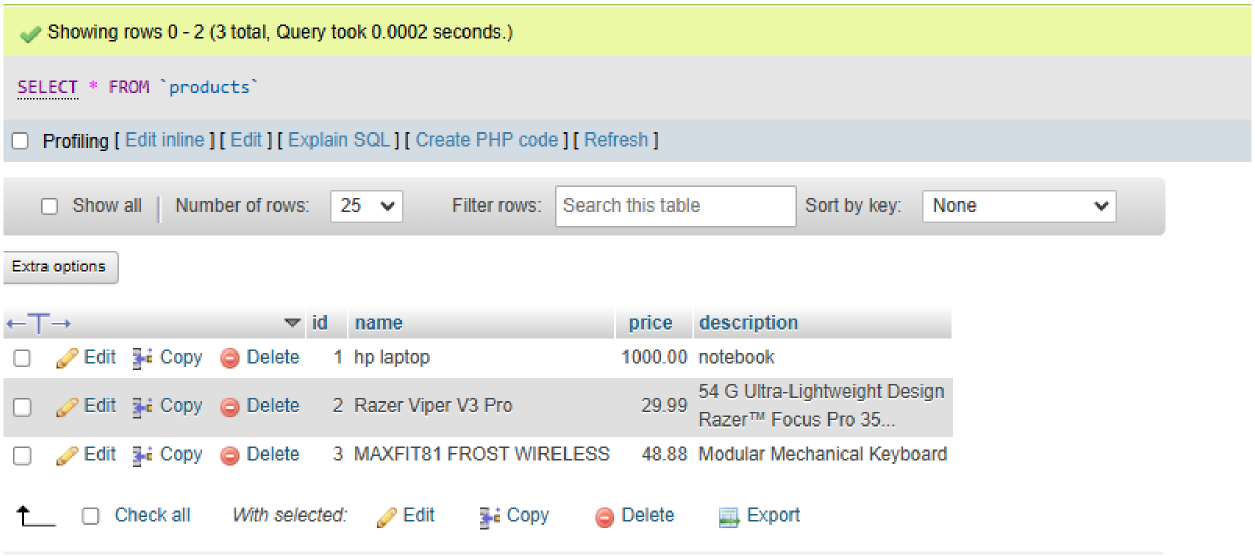
Product added:



Product Management

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	notebook	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete
3	MAXFIT81 FROST WIRELESS	48.88	Modular Mechanical Keyboard	Edit Delete

Reading data in database (Read) :



✓ Showing rows 0 - 2 (3 total, Query took 0.0002 seconds.)

```
SELECT * FROM `products`
```

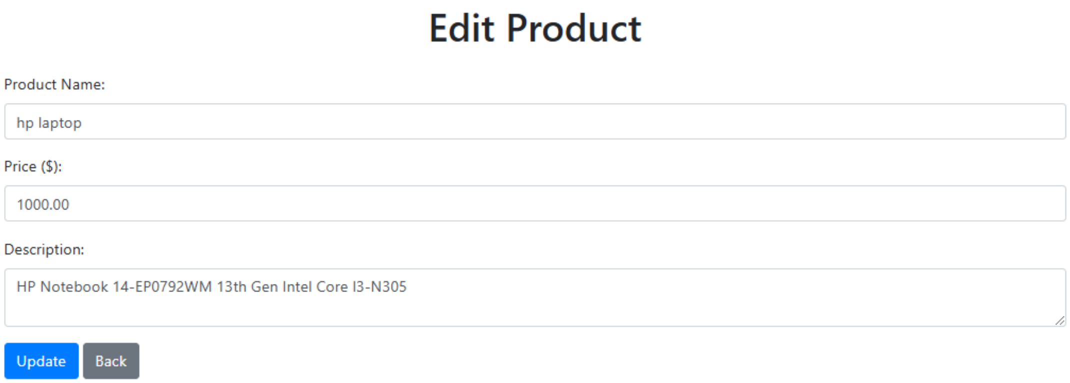
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options

	id	name	price	description
□ Edit Copy Delete	1	hp laptop	1000.00	notebook
□ Edit Copy Delete	2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35...
□ Edit Copy Delete	3	MAXFIT81 FROST WIRELESS	48.88	Modular Mechanical Keyboard

□ Check all *With selected:* Edit Copy Delete Export

Editing the products (Update):

editing description of product 1: hp laptop



Edit Product

Product Name:

hp laptop

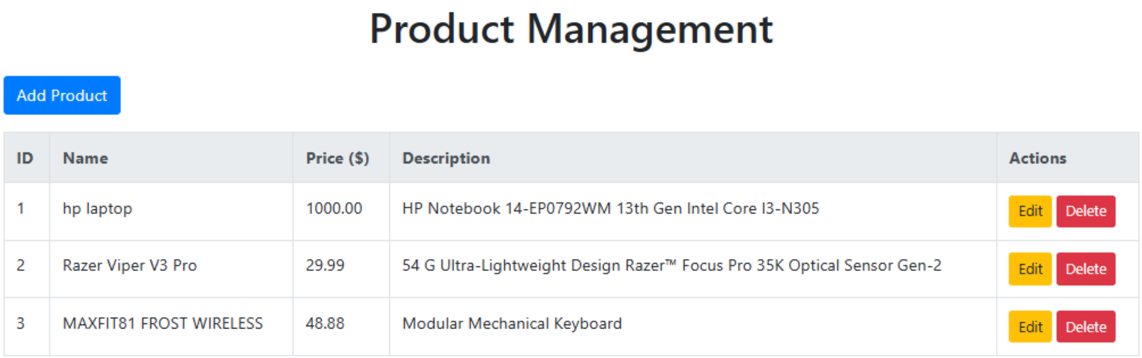
Price ($):

1000.00

Description:

HP Notebook 14-EP0792WM 13th Gen Intel Core I3-N305

Updated:

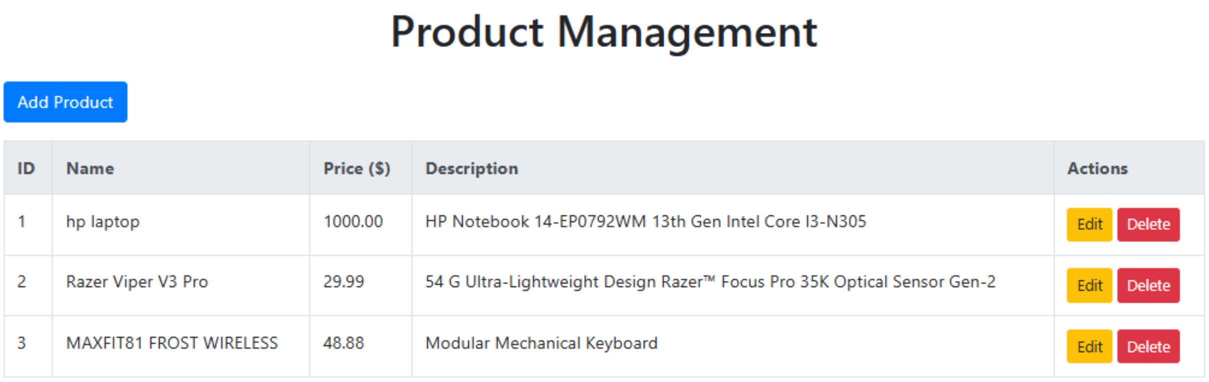


Product Management

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	HP Notebook 14-EP0792WM 13th Gen Intel Core I3-N305	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete
3	MAXFIT81 FROST WIRELESS	48.88	Modular Mechanical Keyboard	Edit Delete

Removing a product: (Delete)

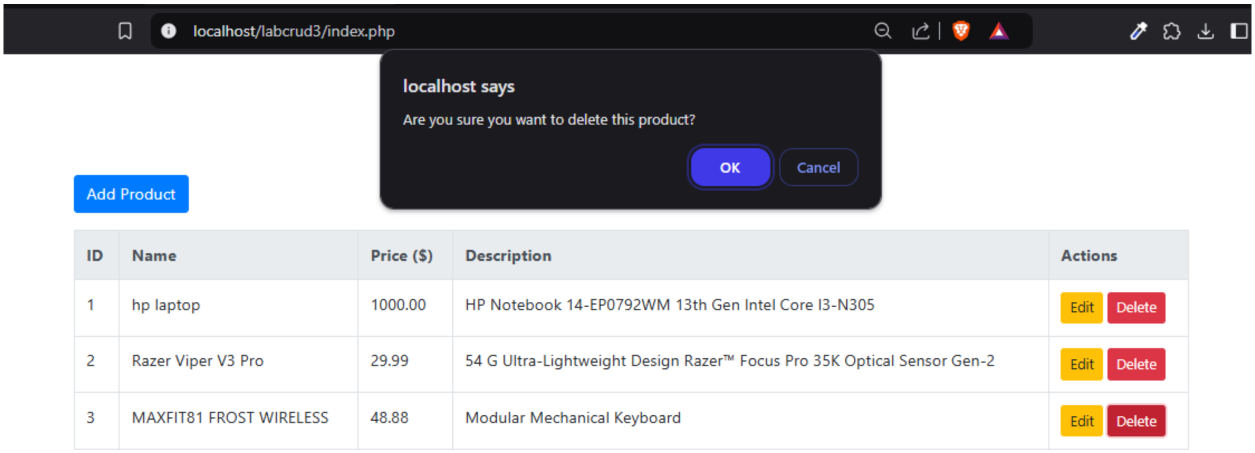
product 3:



Product Management

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	HP Notebook 14-EP0792WM 13th Gen Intel Core I3-N305	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete
3	MAXFIT81 FROST WIRELESS	48.88	Modular Mechanical Keyboard	Edit Delete

Alert message:

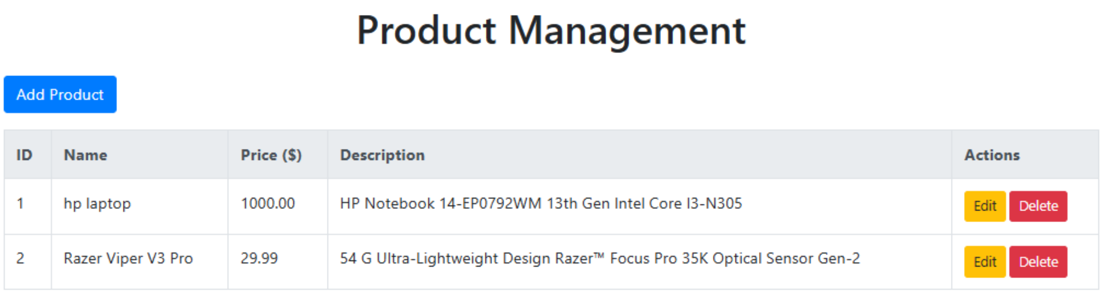


localhost says

Are you sure you want to delete this product?

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	HP Notebook 14-EP0792WM 13th Gen Intel Core I3-N305	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete
3	MAXFIT81 FROST WIRELESS	48.88	Modular Mechanical Keyboard	Edit Delete

Product removed:

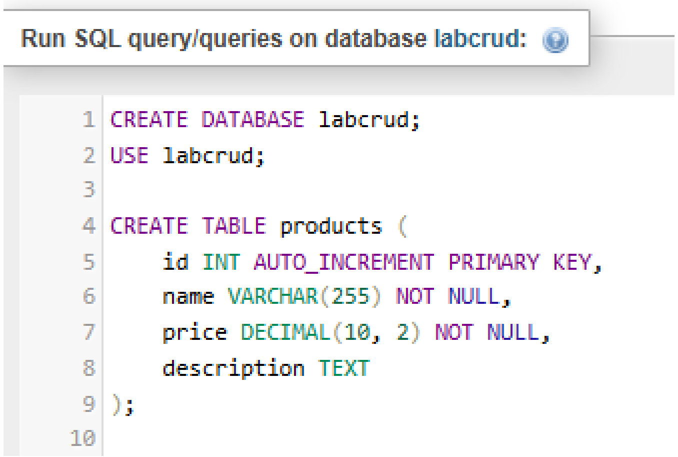


Product Management

ID	Name	Price ($)	Description	Actions
1	hp laptop	1000.00	HP Notebook 14-EP0792WM 13th Gen Intel Core I3-N305	Edit Delete
2	Razer Viper V3 Pro	29.99	54 G Ultra-Lightweight Design Razer™ Focus Pro 35K Optical Sensor Gen-2	Edit Delete

SQL:

Creating a database named `labcrud` and a table with columns id, name, price, and description (which can be nullable).

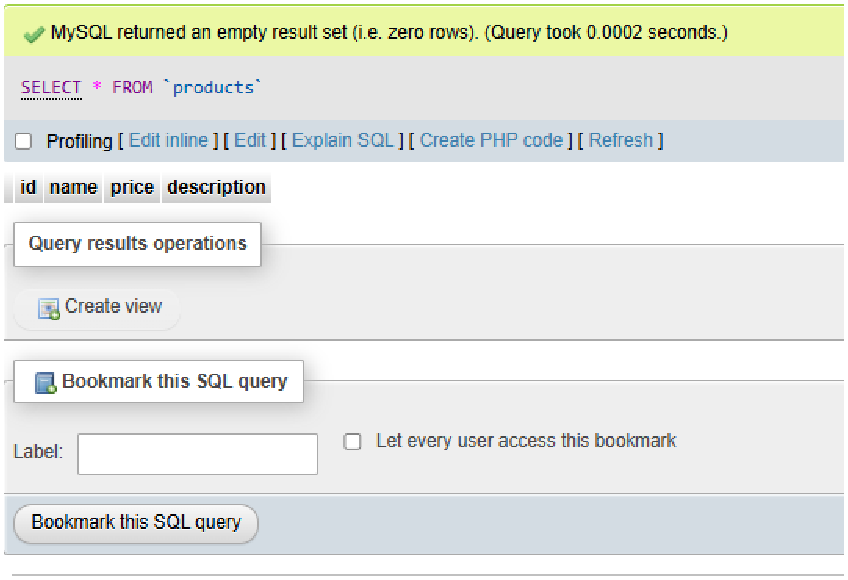


Run SQL query/queries on database labcrud:

```
CREATE DATABASE labcrud;
USE labcrud;

CREATE TABLE products (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    price DECIMAL(10, 2) NOT NULL,
    description TEXT
);

```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)

```
SELECT * FROM `products`
```

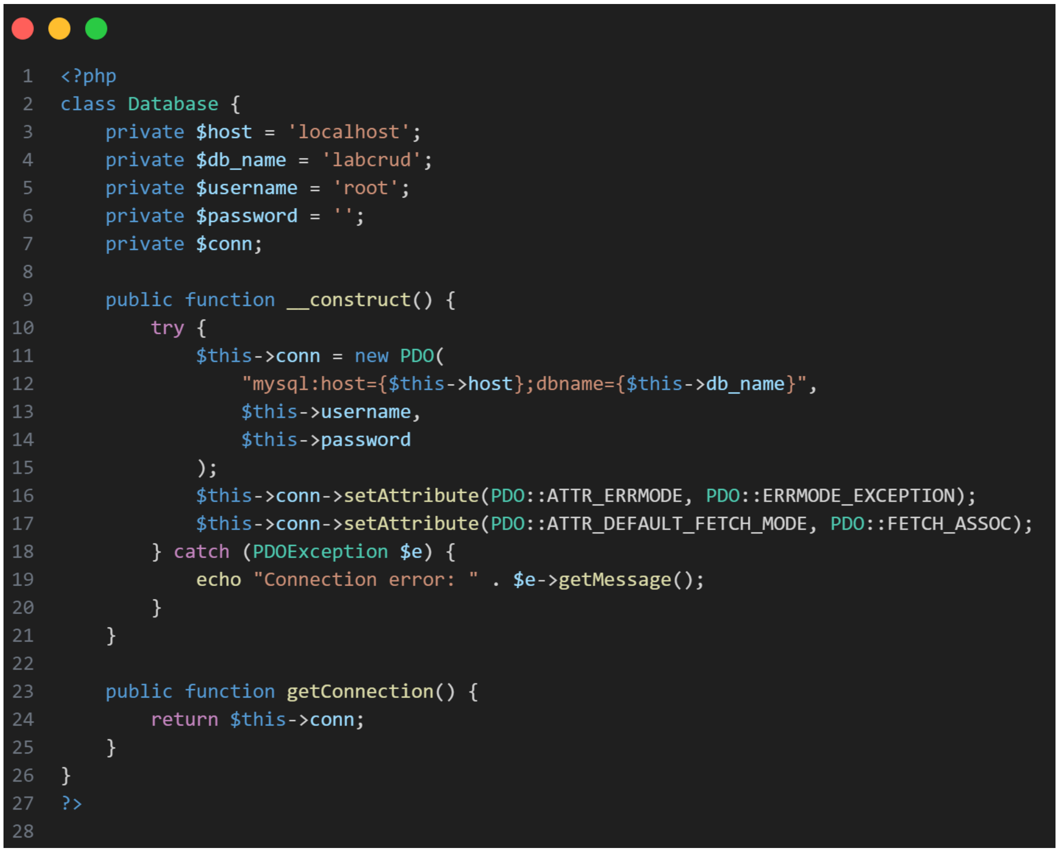
id	name	price	description

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Query results operations
Create view
Bookmark this SQL query
Label:
□ Let every user access this bookmark
Bookmark this SQL query

Database Configuration (database.php):

This file establishes a connection to the MySQL database using PDO (PHP Data Objects). It defines a Database class that manages the connection and provides methods to interact with the database.

database.php:



```php
<?php
class Database {
    private $host = 'localhost';
    private $db_name = 'labcrud';
    private $username = 'root';
    private $password = '';
    private $conn;

    public function __construct() {
        try {
            $this->conn = new PDO(
                "mysql:host={$this->host};dbname={$this->db_name}",
                $this->username,
                $this->password
            );
            $this->conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            $this->conn->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
        } catch (PDOException $e) {
            echo "Connection error: " . $e->getMessage();
        }
    }

    public function getConnection() {
        return $this->conn;
    }
}
?>

```

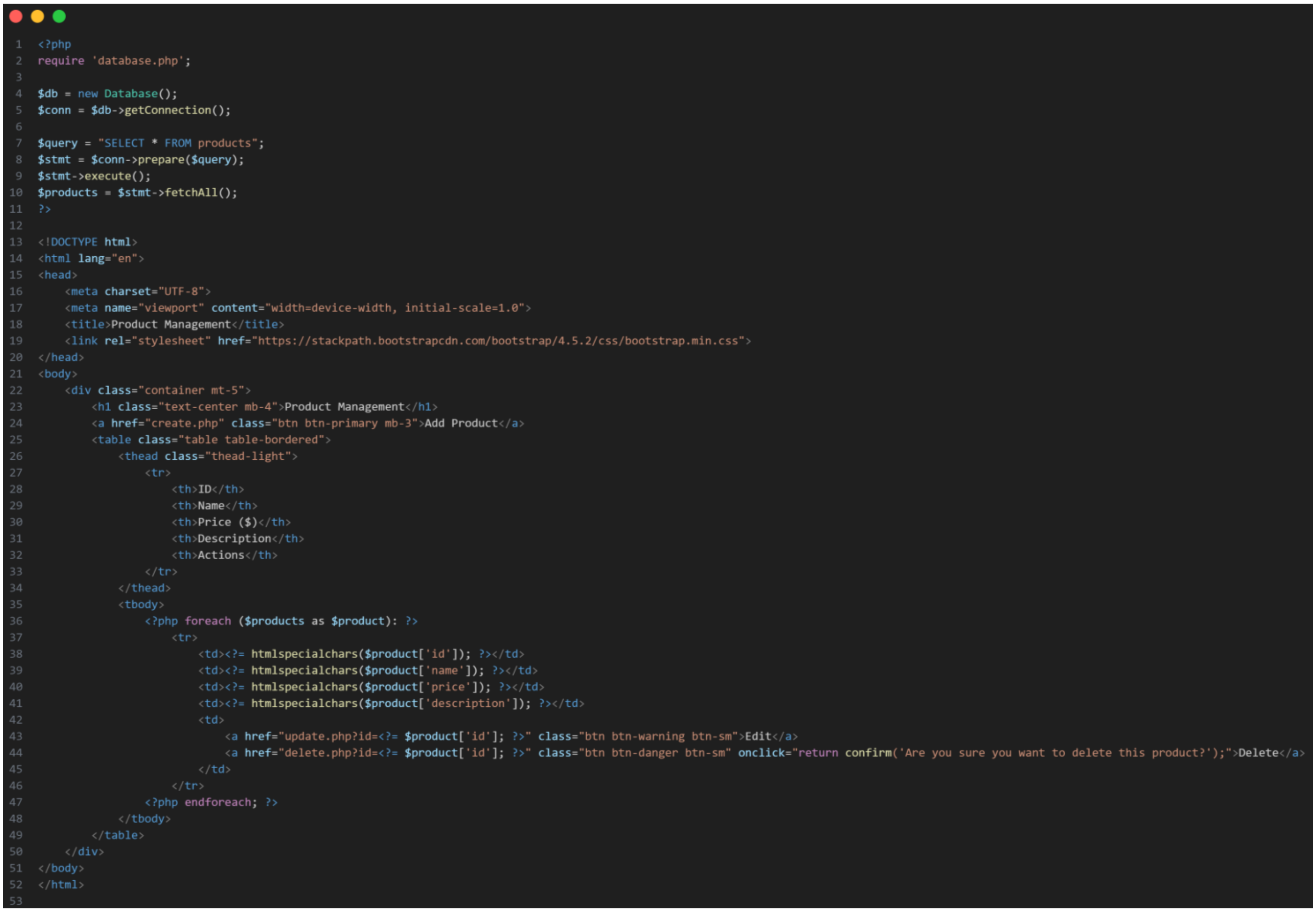
Explanation:

- **Class Properties:** Define the database connection parameters.
- **getConnection Method:** Attempts to establish a PDO connection to the MySQL database and returns the connection object.

Display Products (index.php):

This file retrieves and displays all products from the database in a tabular format. It also provides options to add, edit, or delete products.

index.php:



```php
<?php
require 'database.php';

$db = new Database();
$conn = $db->getConnection();

$query = "SELECT * FROM products";
$stmt = $conn->prepare($query);
$stmt->execute();
$products = $stmt->fetchAll();
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Product Management</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>
    <div class="container mt-5">
        <h1 class="text-center mb-4">Product Management</h1>
        <a href="create.php" class="btn btn-primary mb-3">Add Product</a>
        <table class="table table-bordered">
            <thead class="thead-light">
                <tr>
                    <th>ID</th>
                    <th>Name</th>
                    <th>Price ($)</th>
                    <th>Description</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                <?php foreach ($products as $product): ?>
                    <tr>
                        <td><?= htmlspecialchars($product['id']); ?></td>
                        <td><?= htmlspecialchars($product['name']); ?></td>
                        <td><?= htmlspecialchars($product['price']); ?></td>
                        <td><?= htmlspecialchars($product['description']); ?></td>
                        <td>
                            <a href="update.php?id=<?= $product['id']; ?>" class="btn btn-warning btn-sm">Edit</a>
                            <a href="delete.php?id=<?= $product['id']; ?>" class="btn btn-danger btn-sm" onclick="return confirm('Are you sure you want to delete this product?');">Delete</a>
                        </td>
                    </tr>
                <?php endforeach; ?>
            </tbody>
        </table>
    </div>
</body>
</html>

```

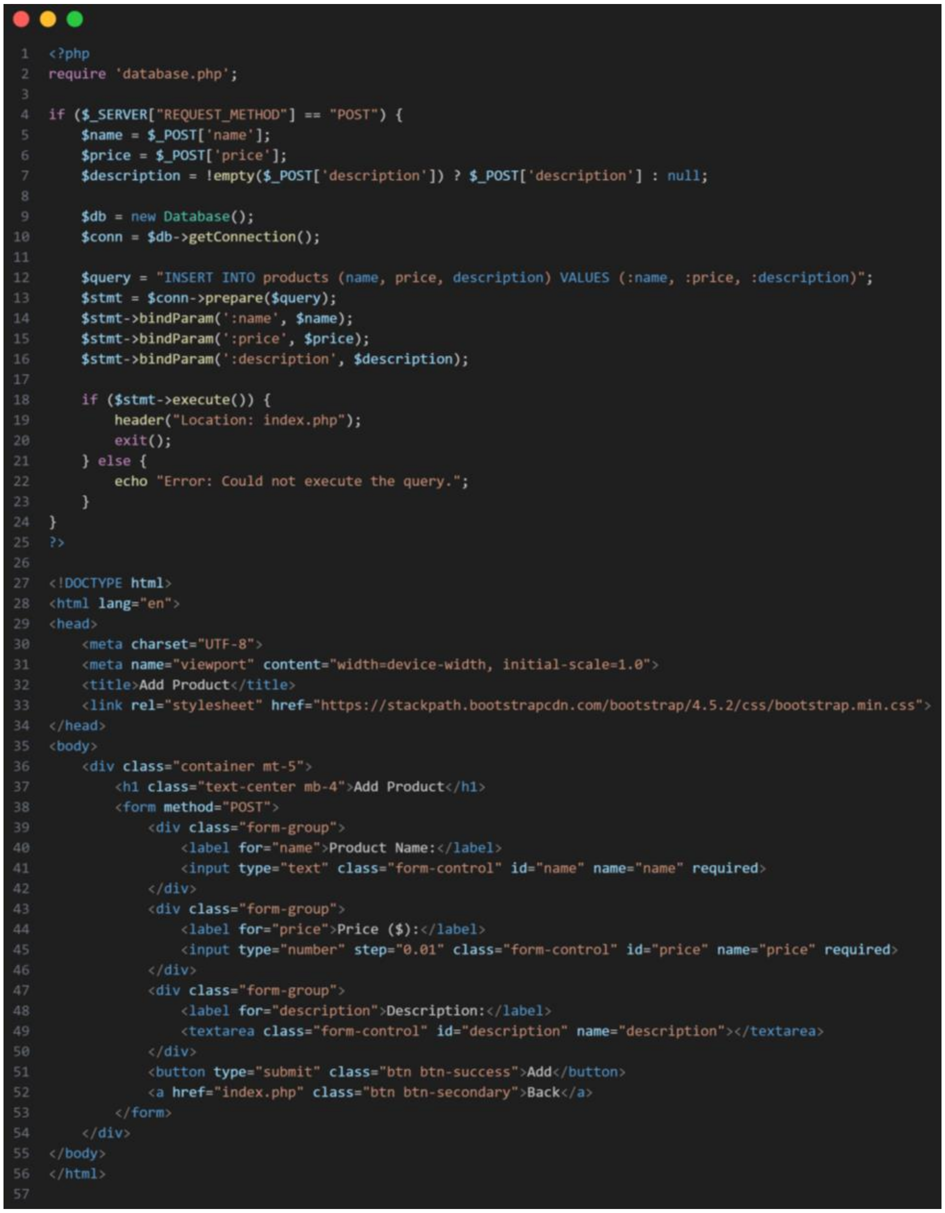
Explanation:

- **Database Retrieval:** Fetches all products from the products table.
- **HTML Structure:** Uses Bootstrap for styling and displays the products in a table with options to add, edit, or delete.

Create Product (create.php):

This file provides a form to add a new product to the database.

create.php:



```php
<?php
require 'database.php';

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['name'];
    $price = $_POST['price'];
    $description = !empty($_POST['description']) ? $_POST['description'] : null;

    $db = new Database();
    $conn = $db->getConnection();

    $query = "INSERT INTO products (name, price, description) VALUES (:name, :price, :description)";
    $stmt = $conn->prepare($query);
    $stmt->bindParam(':name', $name);
    $stmt->bindParam(':price', $price);
    $stmt->bindParam(':description', $description);

    if ($stmt->execute()) {
        header("Location: index.php");
        exit();
    } else {
        echo "Error: Could not execute the query.";
    }
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add Product</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>
    <div class="container mt-5">
        <h1 class="text-center mb-4">Add Product</h1>
        <form method="POST">
            <div class="form-group">
                <label for="name">Product Name:</label>
                <input type="text" class="form-control" id="name" name="name" required>
            </div>
            <div class="form-group">
                <label for="price">Price ($):</label>
                <input type="number" step="0.01" class="form-control" id="price" name="price" required>
            </div>
            <div class="form-group">
                <label for="description">Description:</label>
                <textarea class="form-control" id="description" name="description"></textarea>
            </div>
            <button type="submit" class="btn btn-success">Add</button>
            <a href="index.php" class="btn btn-secondary">Back</a>
        </form>
    </div>
</body>
</html>
```

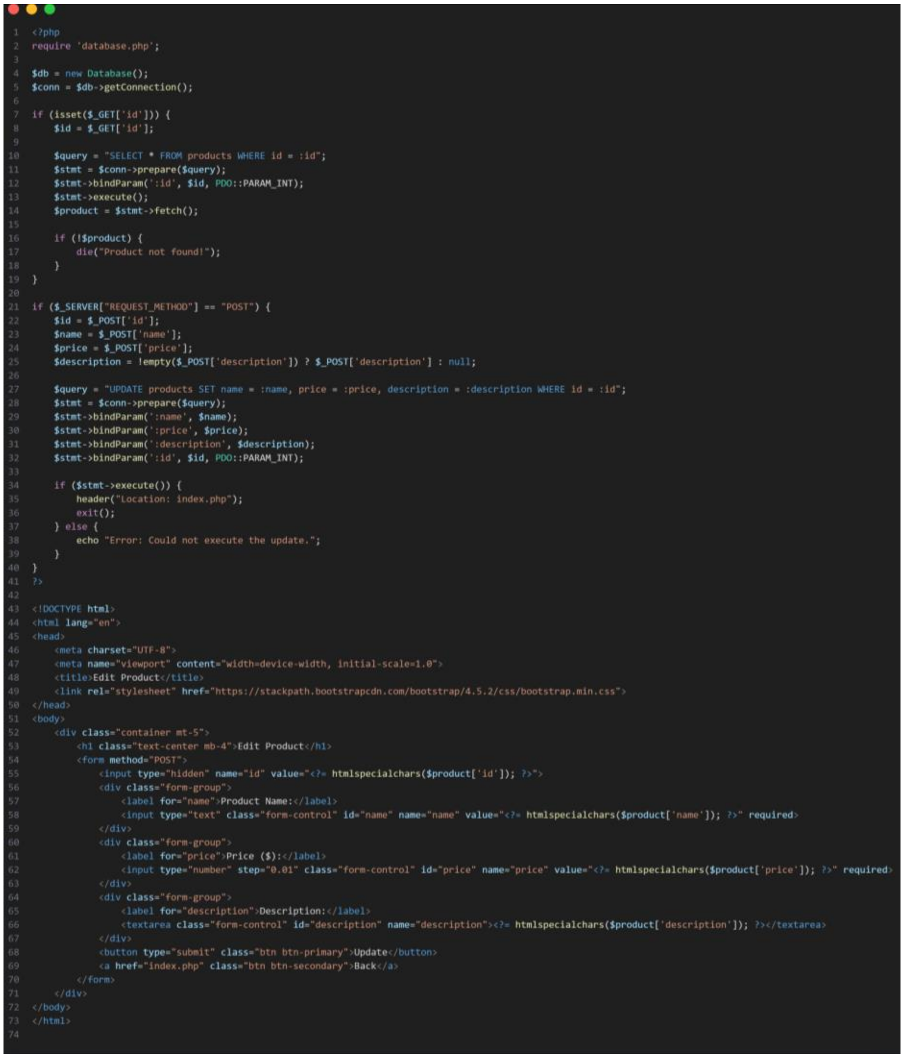
Explanation:

- **Form Handling:** Processes the form submission to insert a new product into the database.
- **HTML Form:** Provides input fields for the product's name, price, and description.

Update Product (update.php):

This file allows users to edit an existing product's details. It retrieves the current product information based on the provided id and updates the database with the new data submitted through the form.

update.php:



```php
<?php
require 'database.php';

$db = new Database();
$conn = $db->getConnection();

if (isset($_GET['id'])) {
    $id = $_GET['id'];

    $query = "SELECT * FROM products WHERE id = :id";
    $stmt = $conn->prepare($query);
    $stmt->bindParam(':id', $id, PDO::PARAM_INT);
    $stmt->execute();
    $product = $stmt->fetch();

    if (!$product) {
        die("Product not found!");
    }
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $id = $_POST['id'];
    $name = $_POST['name'];
    $price = $_POST['price'];
    $description = !empty($_POST['description']) ? $_POST['description'] : null;

    $query = "UPDATE products SET name = :name, price = :price, description = :description WHERE id = :id";
    $stmt = $conn->prepare($query);
    $stmt->bindParam(':name', $name);
    $stmt->bindParam(':price', $price);
    $stmt->bindParam(':description', $description);
    $stmt->bindParam(':id', $id, PDO::PARAM_INT);

    if ($stmt->execute()) {
        header("Location: index.php");
        exit();
    } else {
        echo "Error: Could not execute the update.";
    }
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit Product</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>
    <div class="container mt-5">
        <h1 class="text-center mb-4">Edit Product</h1>
        <form method="POST">
            <input type="hidden" name="id" value="<?= htmlspecialchars($product['id']); ?>">
            <div class="form-group">
                <label for="name">Product Name:</label>
                <input type="text" class="form-control" id="name" name="name" value="<?= htmlspecialchars($product['name']); ?>" required>
            </div>
            <div class="form-group">
                <label for="price">Price ($):</label>
                <input type="number" step="0.01" class="form-control" id="price" name="price" value="<?= htmlspecialchars($product['price']); ?>" required>
            </div>
            <div class="form-group">
                <label for="description">Description:</label>
                <textarea class="form-control" id="description" name="description"><?= htmlspecialchars($product['description']); ?></textarea>
            </div>
            <button type="submit" class="btn btn-primary">Update</button>
            <a href="index.php" class="btn btn-secondary">Back</a>
        </form>
    </div>
</body>
</html>
```

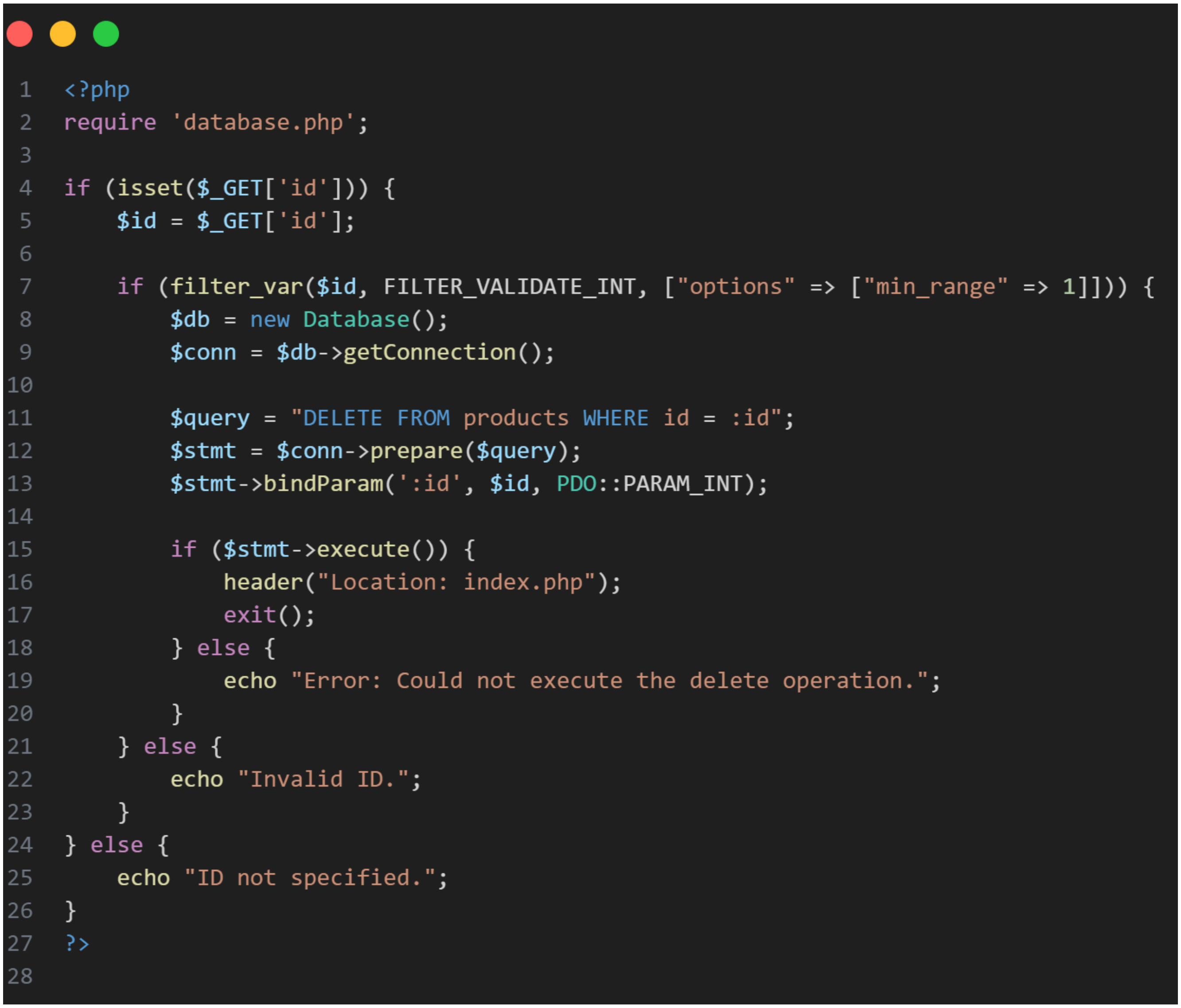
Explanation:

- **Retrieve Product:** Fetches the product details based on the id from the URL parameter.
- **Form Handling:** Updates the product information in the database with the data submitted through the form.
- **HTML Form:** Pre-fills the form fields with the current product details for editing.

Delete Product (delete.php):

This file handles the deletion of a product from the database based on the provided id.

delete.php:



```php
<?php
require 'database.php';

if (isset($_GET['id'])) {
    $id = $_GET['id'];

    if (filter_var($id, FILTER_VALIDATE_INT, ["options" => ["min_range" => 1]])) {
        $db = new Database();
        $conn = $db->getConnection();

        $query = "DELETE FROM products WHERE id = :id";
        $stmt = $conn->prepare($query);
        $stmt->bindParam(':id', $id, PDO::PARAM_INT);

        if ($stmt->execute()) {
            header("Location: index.php");
            exit();
        } else {
            echo "Error: Could not execute the delete operation.";
        }
    } else {
        echo "Invalid ID.";
    }
} else {
    echo "ID not specified.";
}
?>

```

Explanation:

- **Delete Operation:** Deletes the product from the database where the id matches the provided id from the URL parameter.