

Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Факультет Инфокоммуникационных Технологий

Дисциплина: Web-программирование
Лабораторная работа №4

Выполнил:
Теребов М.А.
Проверила:
Марченко Е.В.

Санкт-Петербург, 2024

Оглавление

Цель Работы	3
Ход работы	4
1. Форма с заказом	4
2. Работа с авторизацией через WordPress	7
3. Веб-сервер с выделенным портом	9
Заключение	11

Цель Работы

Научиться обрабатывать запросы от клиента с помощью PHP и сохранять данные в MySQL.

Ход работы

1. Форма с заказом

В данном задании была разработана веб-страница, на которой пользователь сможет оставить свои данные, выбрать товар и оставить комментарий к заказу (по желанию). Для начала была создана HTML страница с использованием CSS.

```
<body>
  <div class="form-container">
    <h1>Order Form</h1>
    <form action="process_order.php" method="POST">
      <div class="form-group">
        <label for="last_name">Last Name:</label>
        <input type="text" id="last_name" name="last_name" required>
      </div>
      <div class="form-group">
        <label for="first_name">First Name:</label>
        <input type="text" id="first_name" name="first_name" required>
      </div>
      <div class="form-group">
        <label for="middle_name">Middle Name:</label>
        <input type="text" id="middle_name" name="middle_name">
      </div>
      <div class="form-group">
        <label for="address">Delivery Address:</label>
        <input type="text" id="address" name="address" required>
      </div>
      <div class="form-group">
        <label for="phone">Phone:</label>
        <input type="tel" id="phone" name="phone" required>
      </div>
      <div class="form-group">
        <label for="email">Email Address:</label>
        <input type="email" id="email" name="email" required>
      </div>
      <div class="form-group">
        <label for="product">Choose Product:</label>
        <select id="product" name="product" required>
          <option value="product1">Product 1</option>
          <option value="product2">Product 2</option>
          <option value="product3">Product 3</option>
          <option value="product4">Product 4</option>
        </select>
      </div>
      <div class="form-group">
        <label for="comments">Comments:</label>
        <textarea id="comments" name="comments" rows="4"></textarea>
      </div>
      <button type="submit">Submit Order</button>
    </form>
  </div>
</body>
</html>
```

Рисунок 1 – Код HTML страницы

После был создан php скрипт, обрабатывающий введенные данные и сохраняющий их в базу данных.

```

process_order.php
1  <?php
2  $host = 'localhost';
3  $dbname = 'orders_db';
4  $user = 'root';
5  $pass = '';
6
7  // Подключение к БД
8  try {
9      $pdo = new PDO("mysql:host=$host;dbname=$dbname", $user, $pass);
10     $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
11 } catch (PDOException $e) {
12     die("Database connection failed: " . $e->getMessage());
13 }
14
15 // Получение данных из формы
16 $lastName = $_POST['last_name'];
17 $firstName = $_POST['first_name'];
18 $middleName = $_POST['middle_name'];
19 $address = $_POST['address'];
20 $phone = $_POST['phone'];
21 $email = $_POST['email'];
22 $product = $_POST['product'];
23 $comments = $_POST['comments'];
24
25 // SQL-запрос
26 $sql = "INSERT INTO orders (last_name, first_name, middle_name, address, phone, email, product, comments)
27     VALUES (:last_name, :first_name, :middle_name, :address, :phone, :email, :product, :comments)";
28
29 $stmt = $pdo->prepare($sql);
30 $stmt->execute([
31     ':last_name' => $lastName,
32     ':first_name' => $firstName,
33     ':middle_name' => $middleName,
34     ':address' => $address,
35     ':phone' => $phone,
36     ':email' => $email,
37     ':product' => $product,
38     ':comments' => $comments
39 ]);
40
41 echo "Order successfully submitted!";
42 ?>
43

```

Рисунок 2 – Код PHP скрипта

С помощью phpMyAdmin и SQL запроса была создана база данных.

```

CREATE DATABASE IF NOT EXISTS orders_db;
USE orders_db;

CREATE TABLE IF NOT EXISTS orders (
    id INT AUTO_INCREMENT PRIMARY KEY,
    lastname VARCHAR(100) NOT NULL,
    firstname VARCHAR(100) NOT NULL,
    middlename VARCHAR(100),
    address TEXT NOT NULL,
    phone VARCHAR(15) NOT NULL,
    email VARCHAR(100) NOT NULL,
    product VARCHAR(100) NOT NULL,
    comment TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

```

Рисунок 3 – SQL запрос

Форма, результат ее обработки php скриптом и доказательство сохранения данных в БД представлены на рисунках ниже.

Order Form

Last Name:
fqw

First Name:
qwe

Middle Name:
qwe

Delivery Address:
qweqweqwe

Phone:
670850967840

Email Address:
mterebov@gmail.com

Choose Product:
Product 3

Comments:
324r5234

Submit Order

Рисунок 4 – Форма

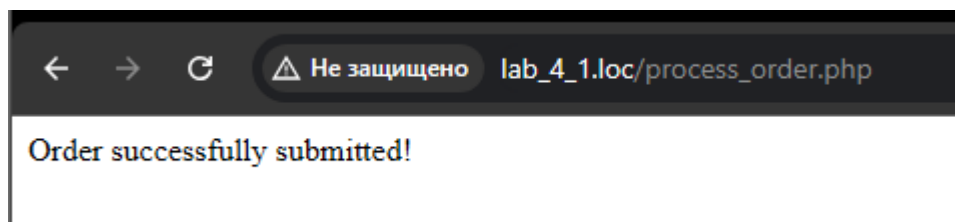


Рисунок 5 – Результат обработки PHP скриптом







		id	last_name	first_name	middle_name	address	phone	email	product	comments		
<input type="checkbox"/>	 Изменить	 Копировать	 Удалить	1	Теребов	Максим	Андреевич	маями	+78005553535	example@gmail.com	product3	faster
<input type="checkbox"/>	 Изменить	 Копировать	 Удалить	2	fqw	qwe	qwe	qweqweqwe	670850967840	mterebov@gmail.com	product3	324r5234

Рисунок 6 – Данные в базе данных

Все данные переданы успешно, значит все работает.

2. Работа с авторизацией через WordPress

В данном упражнении с помощью написания плагина для WordPress была реализована возможность сохранять в базу данных пароль в прямом и инвертированном виде. Сначала была создана новая таблица.

```
1 CREATE TABLE wp_user_password(  
2     id INT AUTO_INCREMENT PRIMARY KEY,  
3     username VARCHAR(255) NOT NULL,  
4     password VARCHAR(255) NOT NULL,  
5     password_inverted VARCHAR(255) NOT NULL  
6 );  
7
```

Рисунок 7 – SQL запрос для создания новой таблицы

Далее был создан PHP скрипт “custom-auth-logger” и сохранен в папке wp-content/plugins. Код скрипта представлен на рисунке ниже.

```
custom-auth-logger.php X  
C:\xampp\htdocs\test_site\wp-content\plugins\custom-auth-logger> custom-auth-logger.php  
1 <?php  
2 /**  
3  * Plugin Name: User Password Logger  
4  * Description: Сохраняет пароли пользователей при их входе в систему.  
5  * Version: 1.0  
6  * Author: TMA  
7  */  
8  
9 // Функция для обработки данных при успешном входе пользователя  
10 function log_user_password($username, $user) {  
11     global $wpdb;  
12  
13     // Получаем пароль пользователя из поля $_POST['pwd'], который был введен при логине  
14     $password = $_POST['pwd']; // Пароль, введенный при логине (из POST-запроса)  
15  
16     // Хэшируем пароль  
17     $hashed_password = password_hash($password, PASSWORD_BCRYPT);  
18  
19     // Инвертируем пароль  
20     $inverted_password = invert_password_bits($password);  
21  
22     // Вставляем данные в таблицу  
23     $table_name = $wpdb->prefix . 'user_password'; // Используем уже существующую таблицу  
24     $wpdb->insert(  
25         $table_name,  
26         array(  
27             'username' => $username,  
28             'password' => $hashed_password, // Хэшированный пароль  
29             'password_inverted' => $inverted_password // Инвертированный пароль  
30         )  
31     );  
32 }  
33  
34 // Функция для инвертирования битов каждого символа пароля  
35 function invert_password_bits($password) {  
36     $inverted_password = '';  
37  
38     // Проходим по каждому символу в пароле  
39     foreach (str_split($password) as $char) {  
40         // Получаем ASCII-код символа  
41         $ascii = ord($char);  
42  
43         // Преобразуем его в 8-битное бинарное представление  
44         $binary = str_pad(decbin($ascii), 8, '0', STR_PAD_LEFT);  
45  
46         // Инвертируем биты (меняем 1 на 0, а 0 на 1)  
47         $inverted_binary = '';  
48         for ($i = 0; $i < strlen($binary); $i++) {  
49             $inverted_binary .= ($binary[$i] == '0') ? '1' : '0';  
50         }  
51  
52         // Добавляем инвертированный символ в итоговую строку  
53         $inverted_password .= $inverted_binary;  
54     }  
55  
56     return $inverted_password;  
57 }  
58  
59 // Подключаем функцию к хуку "wp_login", который срабатывает при успешном входе пользователя  
60 add_action('wp_login', 'log_user_password', 10, 2);  
61
```

Рисунок 8 – PHP скрипт плагина

После плагин был активирован в админке wordpress и была проверена его работа, результат на рисунке ниже.

Сервер: 127.0.0.1 » База данных: test_site_db » Таблица: wp_user_password

Обзор Структура SQL Поиск Вставить Экспорт Импорт Привилегии Операции Слежение Триггеры

Отображение строк 0 - 0 (1 всего, Запрос занял 0.0002 сек.)

SELECT * FROM `wp_user_password`

Профилирование [Построчное редактирование] [Изменить] [Анализ SQL запроса] [Создать PHP-код] [Обновить]

Показать все | Количество строк: 25 | Фильтровать строки: Поиск в таблице

Extra options

	id	username	password	password_inverted
<input type="checkbox"/> Изменить <input type="button" value="Копировать"/> <input type="button" value="Удалить"/>	1	admin	\$2y\$10\$V7RwI6L40uYdSh4luRo98OhKc.g6x3J4NHezl8EdpwN...	1001111010011011100100101001011010010001

↑ ☐ Отметить все | С отмеченными:

Показать все | Количество строк: 25 | Фильтровать строки: Поиск в таблице

Рисунок 9 – Результат работы плагина

Видно, что был сохранен кэшированный пароль и его инвертированная версия, работоспособность плагина подтверждена.

3. Веб-сервер с выделенным портом

В данном задании нужно было создать программу, которая поднимает веб-сервер на заданном пользователем порте и открывает HTML-страницу. Для реализации был выбран язык программирования Python.

The image shows a code editor with two tabs: 'server.py' and 'index.html'. The 'server.py' tab is active, displaying Python code for a web server. The code includes imports for http.server, socketserver, and os. It defines a CustomHTTPRequestHandler class with a do_GET method that serves 'index.html'. The server prompts the user for a port and then starts listening on that port, printing the address to open in a browser.

```
1 import http.server
2 import socketserver
3 import os
4
5 # Создаем обработчик для сервера
6 class CustomHTTPRequestHandler(http.server.SimpleHTTPRequestHandler):
7     def do_GET(self):
8         if self.path == "/":
9             self.path = "index.html"
10        return super().do_GET()
11
12
13 # Запрос порта у пользователя
14 port = int(input("Введите номер порта для запуска сервера: "))
15
16
17 # Запускаем сервер
18 with socketserver.TCPServer(("", port), CustomHTTPRequestHandler) as httpd:
19     print(f"Сервер запущен на порту {port}. Откройте в браузере: http://127.0.0.1:{port}/")
20     httpd.serve_forever()
21
```

Рисунок 10 – Код программы

После запуска сервера переходим по адресу, указанному при его запуске, чтобы проверить его работоспособность.

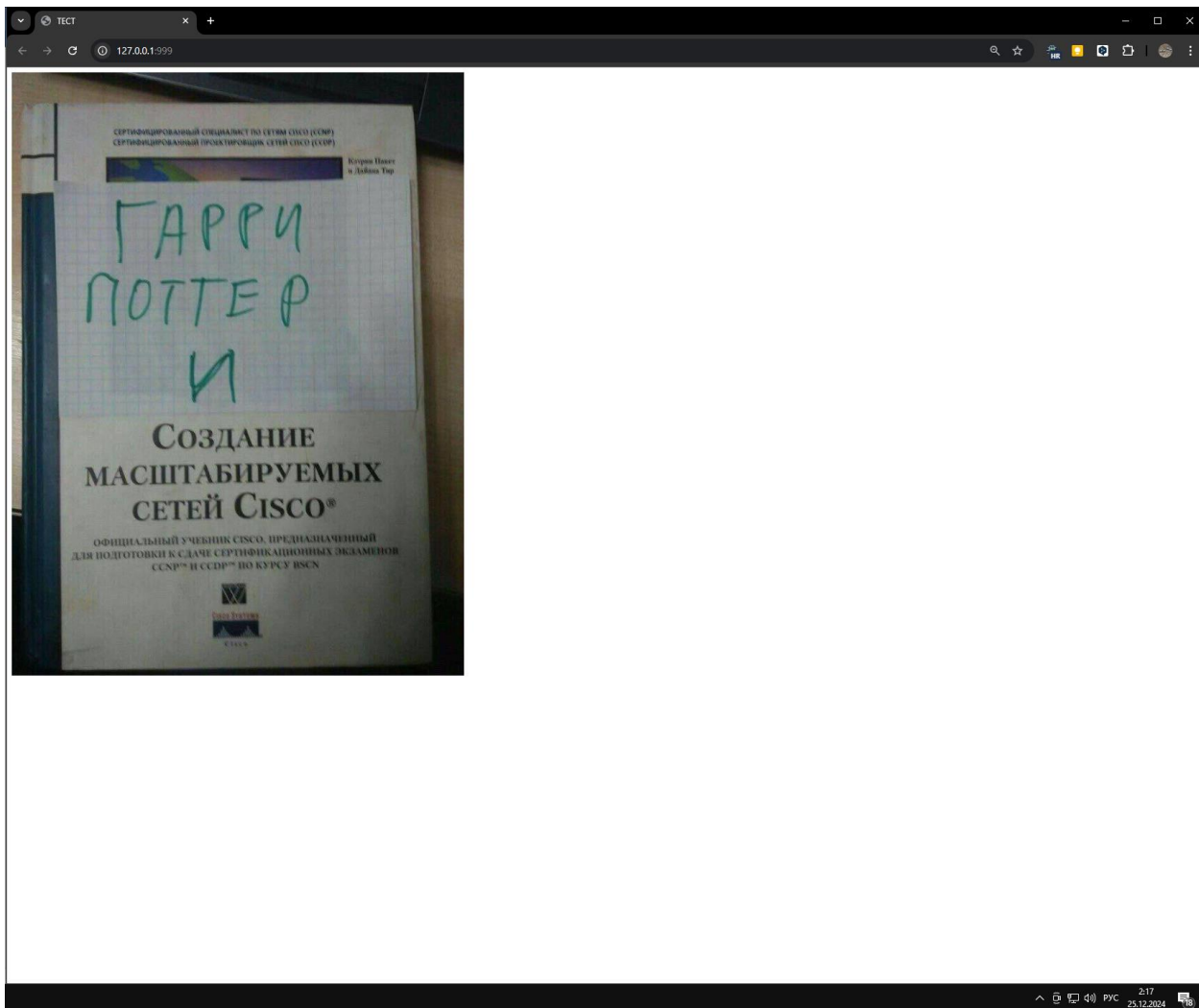


Рисунок 11 - Работоспособность

Заключение

Благодаря данной лабораторной работе научились работать с базами данных через php-скрипты. Разработали программу для запуска веб-серверов на собственном порте