

**Санкт-Петербургский Национальный Исследовательский Университет
Информационных технологий, механики и оптики**

Отчет

Дисциплина: Web-программирование

Лабораторная работа 4.

Выполнил: Сарычев С.И.

Группа № K3321

Проверила: Марченко Е.В.

Санкт-Петербург

2024

Оглавление

Введение	3
Ход работы	4
1. Работа с Gulp.....	Ошибка! Закладка не определена.
2. Создание формы.	Ошибка! Закладка не определена.
3. Установка движка.	Ошибка! Закладка не определена.
Вывод	12

Введение

Цель работы: научиться обрабатывать запросы от клиента с помощью PHP и сохранять данные в MySQL.

Ход работы

1. Форма с заказом.

В данном упражнении необходимо было разработать веб-страницу, на которой пользователь может оставить данные о себе для начала был создан html код формы (рисунок 1)

The image shows a code editor with three tabs: 'order.html', 'process_order.php', and 'styles.css'. The 'order.html' tab is active, displaying the following HTML code:

```
<?DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Order Form</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="form-container">
    <h1>Place Your Order</h1>
    <form action="process_order.php" method="POST">
      <label for="surname">Surname:</label>
      <input type="text" id="surname" name="surname" required><br><br>
      <label for="name">Name:</label>
      <input type="text" id="name" name="name" required><br><br>
      <label for="patronymic">Patronymic:</label>
      <input type="text" id="patronymic" name="patronymic"><br><br>
      <label for="address">Address:</label>
      <textarea id="address" name="address" required></textarea><br><br>
      <label for="phone">Phone:</label>
      <input type="tel" id="phone" name="phone" required><br><br>
      <label for="email">Email:</label>
      <input type="email" id="email" name="email" required><br><br>
      <label for="product">Select 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="other">Other</option>
      </select><br><br>
      <label for="comments">Comments:</label>
      <textarea id="comments" name="comments"></textarea><br><br>
      <button type="submit">Submit</button>
    </form>
  </div>
</body>
</html>
```

Рисунок 1 – Order.html

Далее был также создан css файл чтобы форма выглядела опрятно, после был создан php скрипт для сохранения данных в базу данных. (рисунок 2)

```

<?php
// Database connection
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "Shop py.

$conn = new mysqli($servername, $username, $password, $dbname);

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

// Retrieve form data
$surname = $_POST['surname'];
$name = $_POST['name'];
$patronymic = $_POST['patronymic'];
$address = $_POST['address'];
$phone = $_POST['phone'];
$email = $_POST['email'];
$product = $_POST['product'];
$comments = $_POST['comments'];

// Prepare and execute SQL statement
$sql = "INSERT INTO Orders (surname, name, patronymic, address, phone, email, product, comments)
VALUES (?, ?, ?, ?, ?, ?, ?, ?)";
$stmt = $conn->prepare($sql);
$stmt->bind_param("sssssss", $surname, $name, $patronymic, $address, $phone, $email, $product, $comments);

if ($stmt->execute()) {
    echo "Order successfully placed!";
} else {
    echo "Error: " . $stmt->error;
}

$stmt->close();
$conn->close();
?>

```

Рисунок 2 – Php скрипт

Также с помощью phpMyAdmin была создана таблица для сохранения данных. Вид формы представлен на рисунке 3.

2. Работа с авторизацией через WordPress

В данном упражнении требуется реализовать сохранение пароля в прямом и инвертированном(по битам) виде в базе данных при авторизации в wordpress. Создадим новую таблицу `wp_user_passwords` с помощью phpMyAdmin (рисунок 5).

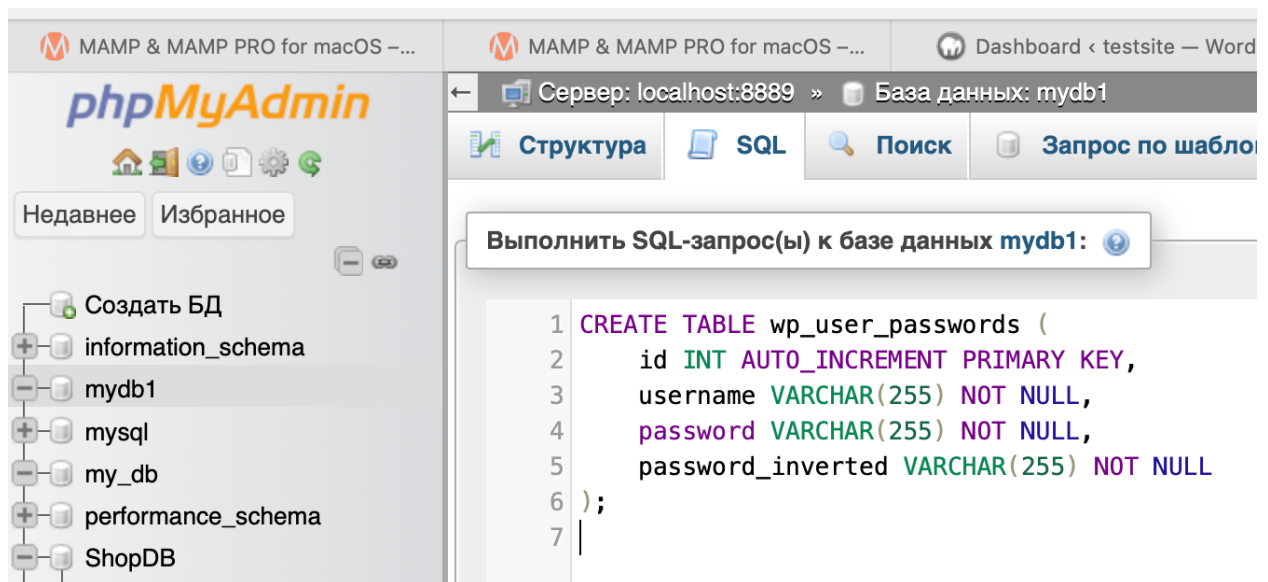


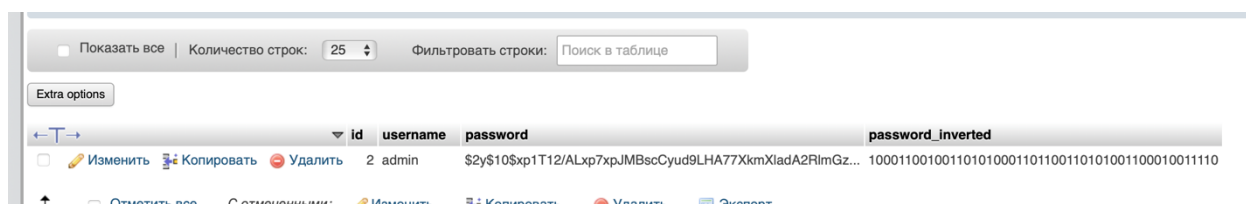
Рисунок 5 – Создание новой таблицы

После этого создадим скрипт `user-password-logger`, в папке `plugins`, внутри директории `wordpress`, текст скрипта представлен на рисунке 6.

```
Welcome user-password-logger.php X
user-password-logger.php
2  /**
4   * Description: Сохраняет пароли пользователей при их входе в систему.
5   * Version: 1.0
6   * Author: serg
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     // Хешируем 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_passwords'; // Используем уже существующую таблицу
24     $wpdb->insert(
25         $table_name,
26         array(
27             'username' => $username,
28             'password' => $hashed_password, // Хешированный 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 хук "wp_login", который срабатывает при успешном входе пользователя
60 add_action('wp_login', 'log_user_password', 10, 2);
61
```

Рисунок 6 – Скрипт для сохранения и инвертирования пароля

После активируем плагин в админ панели wordpress, и для проверки работоспособности плагина переходим на страницу wp-login, вводим имя пользователя и пароль, и проверяем таблицу wp_user_passwords (рисунок 7).



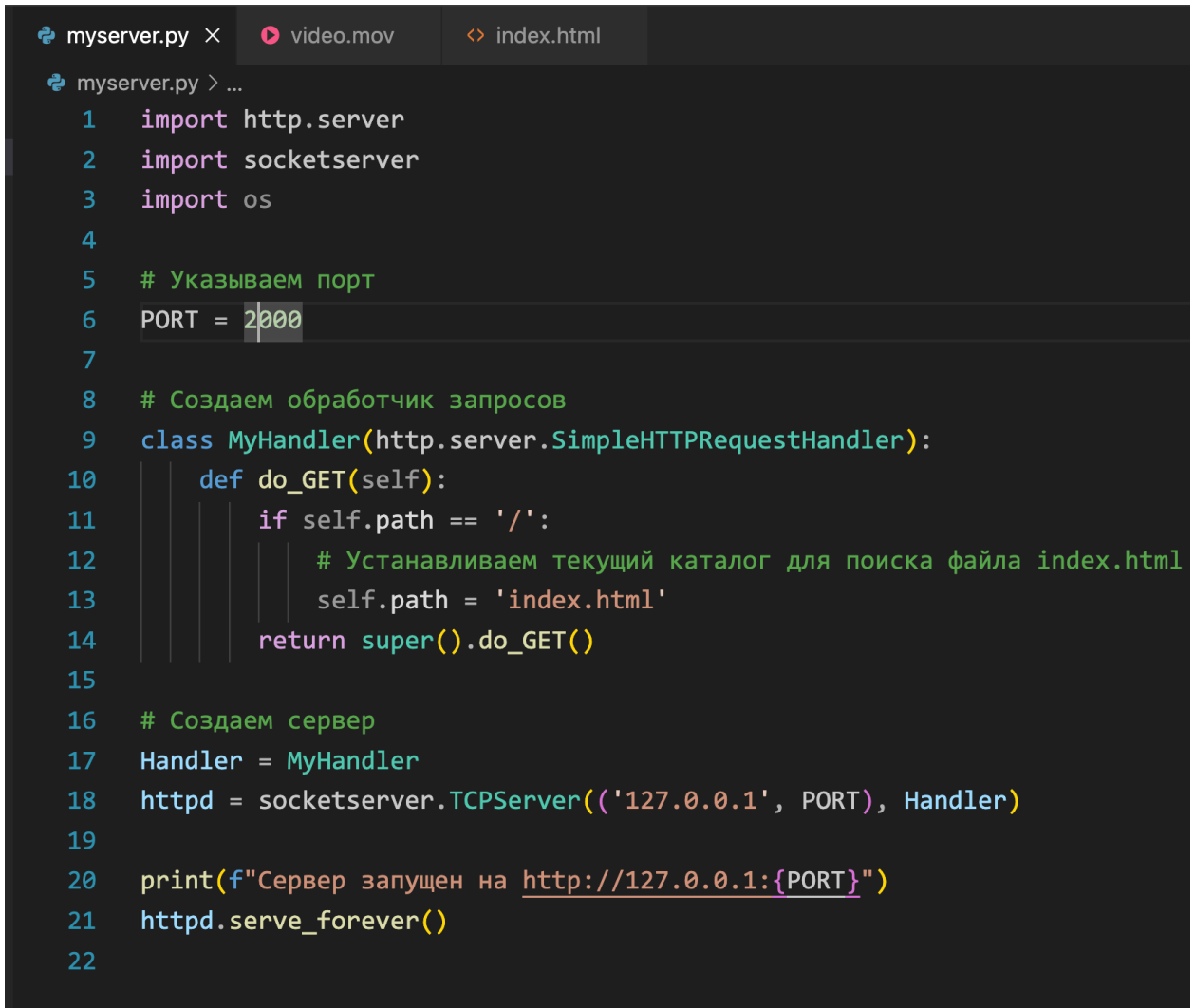
	id	username	password	password_inverted
<input type="checkbox"/>	2	admin	\$2y\$10\$xp1T12/ALxp7xpJMBscCyud9LHA77XkmXladA2RlmGz...	1000110010011010100011011001101010011000100111110

Рисунок 7 – Таблица wp_user_passwords

Видно что сохранился кэшированный пароль и его инвертированная версия, это подтверждает работоспособность плагина.

3. Веб-сервер с собственным портом

В данном упражнении требовалось написать программу, которая позволяет поднимать сервер на указанном порте, а также открывать html-страницу. Для этого было решено использовать python. Код представлен на рисунке 8.

The image shows a screenshot of a code editor with a dark theme. At the top, there are three tabs: 'myserver.py' (active), 'video.mov', and 'index.html'. The 'myserver.py' tab shows a Python script. The script starts with imports for 'http.server', 'socketserver', and 'os'. It then defines a port 'PORT' as 2000. A class 'MyHandler' is defined, inheriting from 'http.server.SimpleHTTPRequestHandler'. The 'do_GET' method of 'MyHandler' checks if the path is '/', and if so, it sets 'self.path' to 'index.html' before calling 'super().do_GET()'. The script then creates a 'Handler' instance, a 'TCPServer' object 'httpd' listening on '127.0.0.1' and 'PORT', and prints a message indicating the server is running on 'http://127.0.0.1:{PORT}'. Finally, it calls 'httpd.serve_forever()' to start the server.

```
myserver.py > ...
1  import http.server
2  import socketserver
3  import os
4
5  # Указываем порт
6  PORT = 2000
7
8  # Создаем обработчик запросов
9  class MyHandler(http.server.SimpleHTTPRequestHandler):
10     def do_GET(self):
11         if self.path == '/':
12             # Устанавливаем текущий каталог для поиска файла index.html
13             self.path = 'index.html'
14             return super().do_GET()
15
16 # Создаем сервер
17 Handler = MyHandler
18 httpd = socketserver.TCPServer(('127.0.0.1', PORT), Handler)
19
20 print(f"Сервер запущен на http://127.0.0.1:{PORT}")
21 httpd.serve_forever()
22
```

Рисунок 8 – Веб сервер с помощью python

Перейдем по адресу на котором запущен веб сервер, и проверим его работоспособность (рисунок 9)

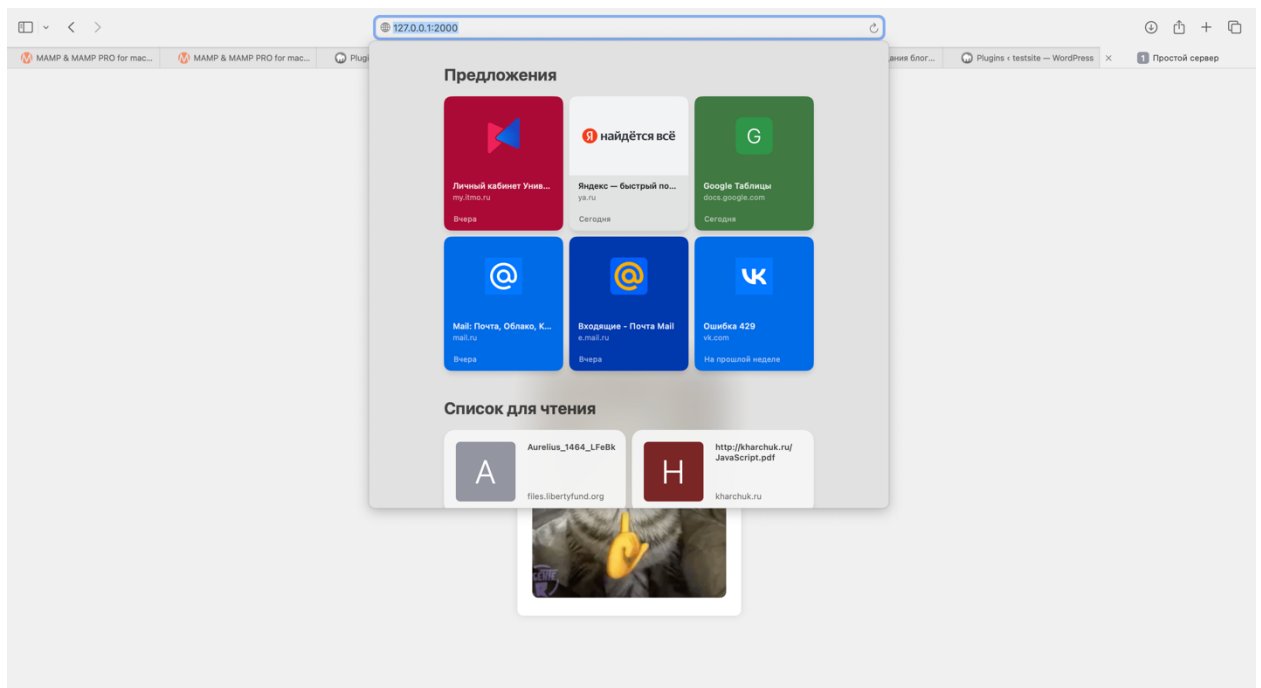


Рисунок 9 – Веб сервер

Вывод

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