|  |  |
| --- | --- |
| https://lh7-us.googleusercontent.com/QuBaagabzZYLr7U3ZbYOtMusd5FbWWxIvMUTN8jrLHzSLMg534z9gXRTIG1Us4i_lOwmWlaBxKedNt-SQ26dm4WmyqwjGDmEO6z8GE3QrZosqvHM88J2EFeVf1u0GzyCZQlhWmp1Zeo85tKo4LJVXQ | МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» |

Институт № 3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 «Вычислительные машины, системы и сети»

Отчёт по лабораторной работе №4  
по учебной дисциплине «Web-технологии»

на тему «Сервер на Node.js»

Выполнил  
студент группы М3О-119БВ-24

Нарзиев А.Т.

Принял

Ассистент каф.304 Борисов А.И.

Москва  
2024

# Содержание

[Задание 3](#_Toc194427981)

[Решение 4](#_Toc194427982)

[1. Разработка сервера HTTP 4](#_Toc194427983)

[1.1. Сервер – server.js: 4](#_Toc194427984)

[1.2. Страница с текстом – index.html: 5](#_Toc194427985)

[1.3. Страница с полем ввода – input.html: 5](#_Toc194427986)

[1.4. Страница с картинкой – image.html 6](#_Toc194427987)

[1.5. Страница с таблицей – table.html: 6](#_Toc194427988)

[1.6. Пример работы сервера: 6](#_Toc194427989)

[2. Разработка стилей 8](#_Toc194427990)

[3. Обработка POST запросов 8](#_Toc194427991)

[4. Разработка шаблонов 8](#_Toc194427992)

[5. Логирование 8](#_Toc194427993)

[6. Аутентификация пользователей 8](#_Toc194427994)

[7. Упрощение маршрутизации 8](#_Toc194427995)

[8. Кэширование ресурсов 8](#_Toc194427996)

[Вывод 9](#_Toc194427997)

# Задание

1. Разработать программу клиент, работающий на сокетах на выбранном Вами языке программирования. Данный клиент должен хранить лог файл соединений. В лог файле хранится время подключения и адрес сервера, время отправки сообщения и само сообщение, время получения сообщения и само сообщение. При подключении к серверу клиент через определенный промежуток времени (задается разработчиком клиента) отправляет сообщение серверу: ФИО и группа студента, а также ждет от сервера. Данные о сервере необходимо вводить через файл конфигурации. В отчете привести код клиента и один тестовый лог файл.
2. Разработать программу сервер, которая ожидает строку от клиента. Через промежуток времени (моделирование работы сервера) он отправляет в ответ эту же строку, только зеркально отраженную и добавляет к сообщению ФИО студента, написавшего сервер. (пример ответа: Чиволвап Йирю Вотит. Сервер написан Титовым Ю.П. М3О-1ХХБВ-ХХ).

Сервер должен эмулировать работу путем временной задержки. Отключение клиентов от сервера совершать через определенное время работы клиента. Сервер и клиент должны быть написаны на различных языках программирования.

Сервер ведёт лог-файл, в котором содержится:

* Время запуска сервера,
* Время подсоединения клиента,
* Адрес клиента,
* Время получения сообщения и само сообщение,
* Время отправки сообщения и само сообщение, Время отключения клиента от сервера.

Лог-файл не очищать.

1. Доработать программу сервер для выполнения асинхронных запросов от клиентов. Чётные варианты реализуют многопоточный асинхронный сервер, нечетные – многопроцессный. Номер варианта определяется номером по списку.

При доработке сервера необходимо создать новый проект, т.е. в результате должно получиться две программы сервера.

1. Запустить 5 клиентов в сети кафедры и 1 сервер (5 студентов). В отчете указать настройки для подключения и лог файлы клиентов и серверов. Проводить тестирование одновременного подключения, т.е. второй, третий и т.д. пользователь подключается к серверу пока первый пользователь еще не получил ответ. Желательно провести тестирование как на однопоточном, так и на асинхронном сервере, описать отличия.

# Решение

## Разработка сервера HTTP

## *Сервер – server.js:*

// Импортируем необходимые модули

const http = require('http');// Модуль для создания HTTP сервера

const fs = require('fs');    // Модуль для работы с файловой системой

const path = require('path');// Модуль для работы с путями файловой системы

const PORT = 8080; // Порт, на котором будет работать сервер

// Создаем HTTP сервер

const server = http.createServer((req, res) => {

    let filePath = ''; // Путь к файлу

    // Определяем, какой файл нужно вернуть в зависимости от URL запроса

    switch (req.url) {

        // Главная страница

        case '/':

            filePath = path.join(\_\_dirname, 'public', 'index.html');

            break;

        // Страница ввода

        case '/input':

            filePath = path.join(\_\_dirname, 'public', 'input.html');

            break;

        // Страница таблицы

        case '/table':

            filePath = path.join(\_\_dirname, 'public', 'table.html');

            break;

        // Страница с изображением

        case '/image':

            filePath = path.join(\_\_dirname, 'public', 'image.html');

            break;

        case '/404-image.jpg':

            // Изображение для 404 ошибки

            filePath = path.join(\_\_dirname, 'public', '404-image.jpg');

            break;

        default:

            // Если запрашиваемая страница не найдена, возвращаем 404 ошибку

            res.writeHead(404, { 'Content-Type': 'text/html' });

            res.end('<h1>404 - Страница не найдена</h1><a href="/">Вернуться на главную страницу</a>');

            return;

    }

    // Читаем файл по указанному пути

    fs.readFile(filePath, (err, content) => {

        if (err) {

            // Если произошла ошибка при чтении файла, возвращаем 500 ошибку

            res.writeHead(500, { 'Content-Type': 'text/html' });

            res.end('<h1>500 - Ошибка сервера</h1>');

            return;

        }

        // Если файл успешно прочитан, возвращаем его содержимое с кодом 200

        res.writeHead(200, { 'Content-Type': 'text/html' });

        res.end(content); // Отправляем содержимое файла

    });

});

// Обработчик ошибок

server.on('error', (err) => {

    console.error('Ошибка сервера:', err);

});

// Запуск сервера

server.listen(PORT, '172.17.254.41', () => {

    console.log(`Сервер запущен на http://172.17.254.41:${PORT}/`);

});

## *Страница с текстом – index.html:*

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Главная страница</title>

</head>

<body>

    <header>

        <h1>Добро пожаловать на главную страницу!</h1>

        <nav>

            <ul>

                <li><a href="#about">О сайте</a></li>

                <li><a href="#services">Услуги</a></li>

                <li><a href="#contact">Контакты</a></li>

            </ul>

        </nav>

    </header><main>

        <section id="about">

            <h2>О сайте</h2>

            <p>Это сайт для представления информации и материалов

                по всему курсу предмета Web-технологии</p>

        </section>

        <section id="services">

            <h2>Услуги</h2>

            <p>Здесь мы собрали конспекты всех лекций, практических

                занятий, лабораторных работ и материалов для

                подготовки к экзамену.</p>

        </section>

        <section id="contact">

            <h2>Контакты</h2>

            <p>Свяжитесь с нами!<br><br>

                <a href="https://github.com/ZeroD1vision/Web-tech">Наш репозиторий Git</a><br>

                <a href="https://t.me/+Yku9N\_IK9mEzOTZi">Наш Telegram-канал</a><br>

                <a href="mailto:anatolypozd739@gmail.com">Написать письмо администратору.</a><br>

            </p>

        </section>

    </main>

    <a href="/input">Перейти на страницу с полем ввода</a><br>

    <a href="/table">Перейти на страницу с таблицей</a><br>

    <a href="/image">Перейти на страницу с картинкой</a>

    <footer>

        <p>© 2025 Нарзиев Артемий</p>

    </footer>

</body>

</html>

## *Страница с полем ввода – input.html:*

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Страница с полем ввода</title>

</head>

<body>

    <h1>Страница с полем ввода</h1>

    <form>

        <label for="input">Введите что-нибудь:</label>

        <input type="text" id="input" name="input">

        <input type="submit" value="Отправить">

    </form>

    <a href="/">Вернуться на главную страницу</a>

    </main>

    <footer>

        <p>© 2025 Нарзиев Артемий</p>

    </footer>

</body>

</html>

## *Страница с картинкой – image.html*

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Страница с картинкой</title>

</head>

<body>

    <h1>Страница с картинкой</h1>

    <img src="/404-image.jpg" alt="Ошибка 404" style="width:300px;">

    <a href="/">Вернуться на главную страницу</a>

</body>

</html>

## *Страница с таблицей – table.html:*

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Страница с таблицей</title>

</head>

<body>

    <h1>Страница с таблицей</h1>

    <table border="2">

        <tr>

            <th>Имя</th>

            <th>Фамилия</th>

        </tr>

        <tr>

            <td>Артемий</td>

            <td>Нарзиев</td>

        </tr>

        <tr>

            <td>Артемий</td>

            <td>Кошкин</td>

        </tr>

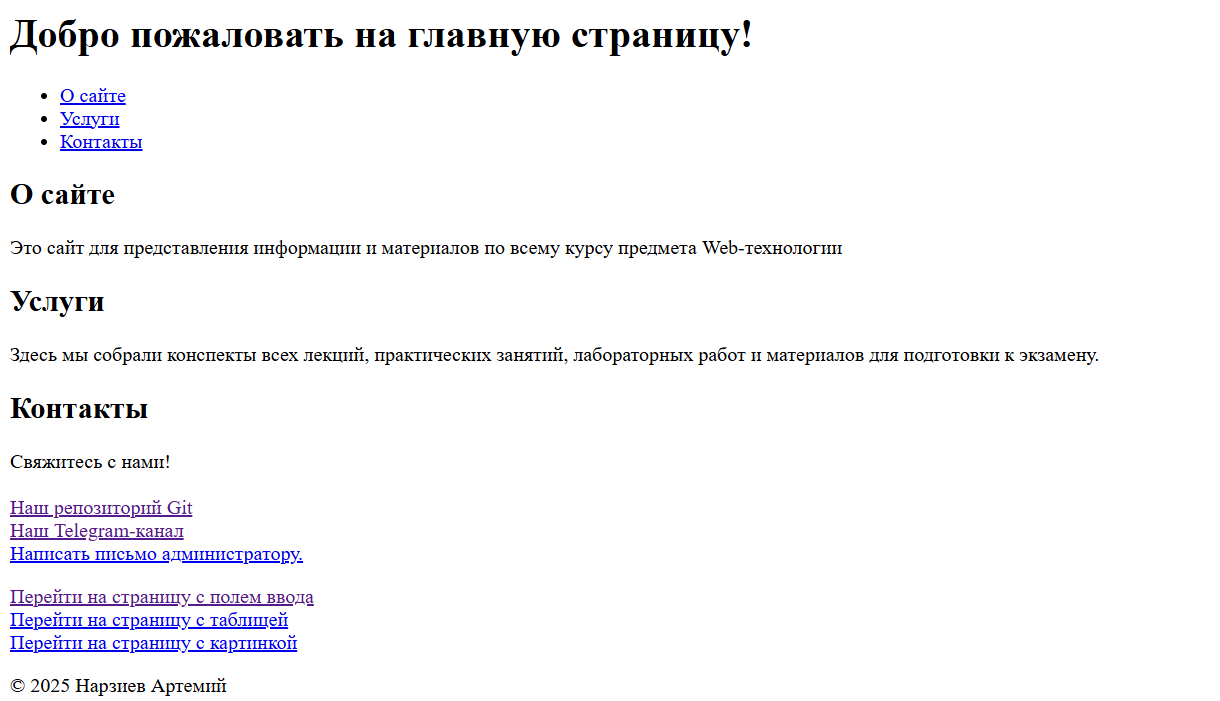
    </table>

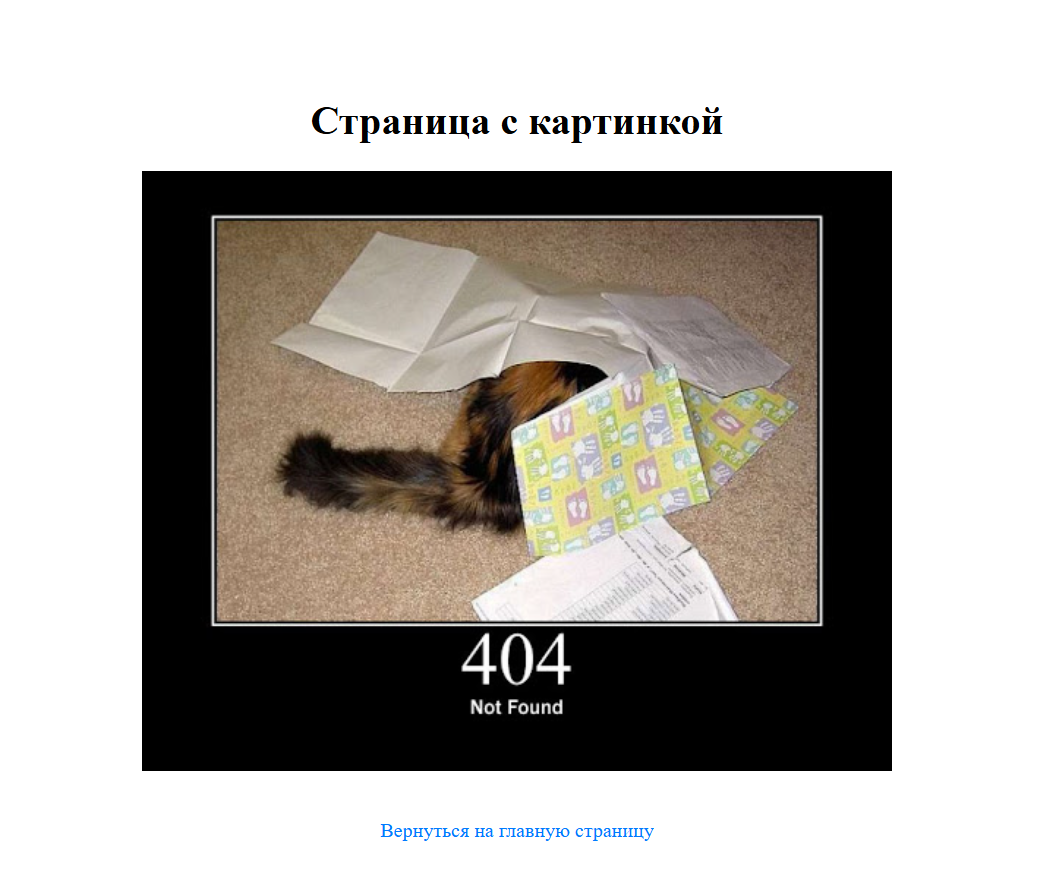
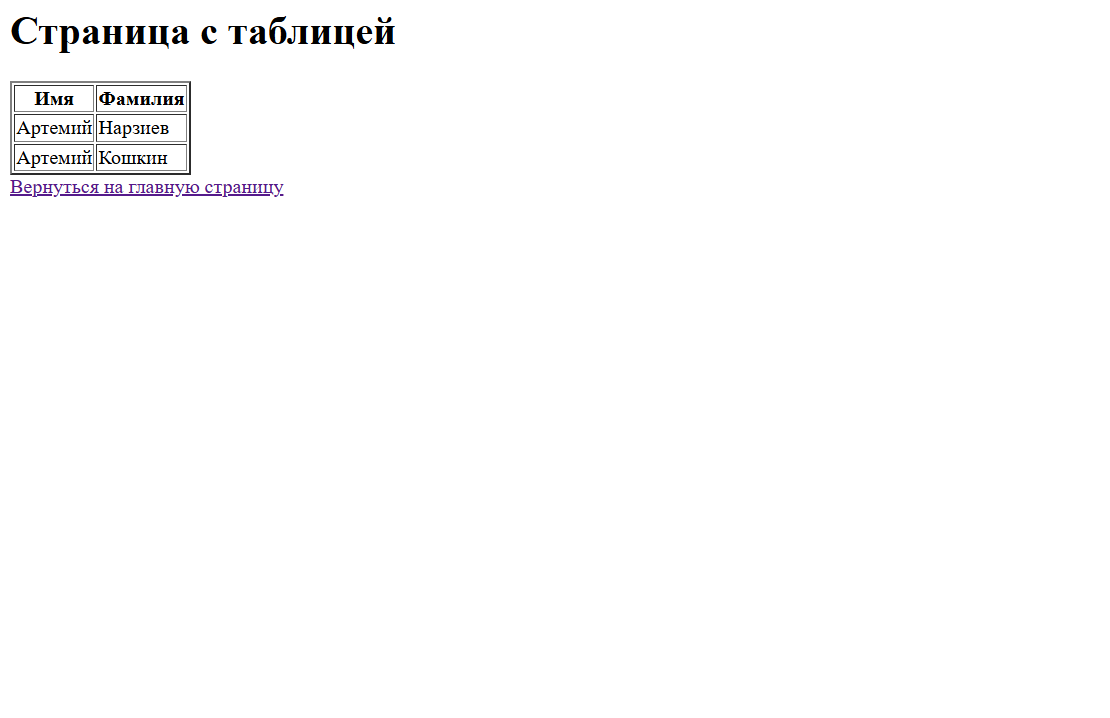
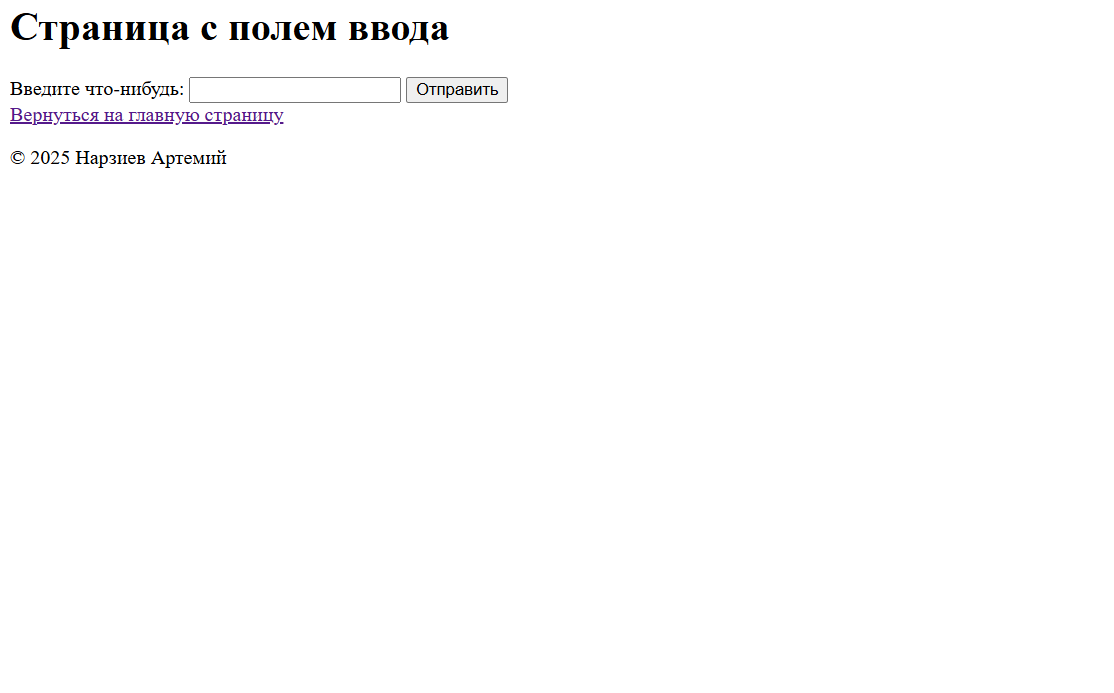
    <a href="/">Вернуться на главную страницу</a>

</body>

</html>

## *Пример работы сервера:*





## Разработка стилей

## Обработка POST запросов

## Разработка шаблонов

## Логирование

## Аутентификация пользователей

## Упрощение маршрутизации

## Кэширование ресурсов

# Вывод

Разработка программы завершена на том основании, что:

1. Полученные результаты совпали с ожидаемыми
2. Считаю набор тестов полным