**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

***Факультет Информационных технологий***

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА №** 16

**Дисциплина: «Backend»**

**Тема:** *Добавление скриптов JS и стилей CSS в веб-приложение на основе ASP.NET Core для использования на стороне фронтенда*

**Выполнил: студент группы: 231-339**

Карапетян Нвер Каренович\_\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)



**Дата, подпись:** 07.05.25 \_\_\_\_\_\_\_\_\_\_\_\_

(Дата)(Подпись)

**Проверил:** \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_*

(Фамилия И.О., степень, звание) (Оценка)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *\_\_\_\_\_\_\_\_\_\_\_\_\_*

(Дата)(Подпись)

**Москва**

**2025**

Цель:

Освоить процесс интеграции и использования скриптов JavaScript и стилей CSS в веб-приложении на основе ASP.NET Core для обеспечения их доступности на стороне фронтенда.

Задачи:

* Создать структуру каталогов для хранения стилей CSS и скриптов JavaScript в проекте ASP.NET Core.
* Добавить несколько файлов стилей CSS и скриптов JavaScript в соответствующие каталоги.
* Использовать теги <link> и <script> для подключения стилей и скриптов в HTML-страницах.
* Убедиться в корректной загрузке и применении добавленных стилей и скриптов на веб-страницах.

Ход работы

Структура проекта и раздача статических файлов

В корне проекта была создана папка **wwwroot**, куда помещены все ресурсы, доступные клиенту без участия контроллеров. Внутри **wwwroot** организованы два подкаталога: **css** и **js**.



Рисунок . Структура папки wwwroot.

ASP.NET Core по умолчанию раздаёт содержимое этой папки благодаря вызовам app.UseDefaultFiles() и app.UseStaticFiles() в конвейере middleware. В файле Program.cs эти методы расположены после app.UseSession() и перед app.UseHttpsRedirection(), что обеспечивает первоочередную отдачу статического контента:

Листинг 1. Фрагмент из Program.cs.

app.UseSession();

app.UseDefaultFiles();

app.UseStaticFiles();

Такой порядок позволяет при обращении к корню приложения автоматически отдавать файл index.html, а затем обслуживать все запросы вида «/css/...» и «/js/...».

Добавление файлов CSS и JavaScript

В папке **wwwroot/css** был создан файл **site.css**, в котором определены базовые стили для фона страницы, типографики и оформления кнопок:

Листинг 2. site.css.

body {

    background-color: #eef;

    font-family: Arial, sans-serif;

}

h1 {

    color: #225;

}

button {

    padding: 0.5rem 1rem;

    font-size: 1rem;

    margin: 0.5rem;

    cursor: pointer;

}

В папке **wwwroot/js** лежит файл **app.js**, реализующий логику работы с localStorage и sessionStorage. При нажатии на кнопки «Сохранить» и «Прочитать» скрипт сохраняет или извлекает строки, выводя результаты в элемент <h3 id="out">:

Листинг 3. app.js.

const out = document.getElementById("out");

document.getElementById("saveLocal").onclick = () => {

    const value = prompt("Значение для localStorage: ");

    localStorage.setItem("localStorageKey", value);

    out.textContent = "localStorage успешно сохранен"

};

document.getElementById("loadLocal").onclick = () => {

    out.textContent = "localStorage: " + localStorage.getItem("localStorageKey");

};

document.getElementById("saveSession").onclick = () => {

    const value = prompt("Значение для sessionStorage: ");

    sessionStorage.setItem("sessionStorageKey", value);

    out.textContent = "sessionStorage успешно сохранен";

};

document.getElementById("loadSession").onclick = () => {

    out.textContent = "sessionStorage: " + sessionStorage.getItem("sessionStorageKey");

};

Оба файла размещены в своих каталогах и доступны по URL /css/site.css и /js/app.js.

Подключение ресурсов в HTML

HTML-страница **index.html**, которая находится в **wwwroot**, содержит разметку для взаимодействия с пользователем. В теге <head> к ней подключаются стили через <link rel="stylesheet" href="/css/site.css" />, а перед закрывающимся тегом </head> — скрипт <script src="/js/app.js" defer></script>. Это гарантирует, что сначала загрузятся все стили, а затем — сам скрипт после парсинга разметки:

Листинг 4. index.html.

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8" />

    <title>Клиентские состояния</title>

    <link rel="stylesheet" href="/css/site.css" />

    <script src="/js/app.js" defer></script>

</head>

<body>

    <h1>Клиентские состояния</h1>

    <button id="saveLocal">Сохранить в localStorage</button>

    <button id="loadLocal">Прочитать из localStorage</button>

    <button id="saveSession">Сохранить в sessionStorage</button>

    <button id="loadSession">Прочитать из sessionStorage</button>

    <h3 id="out"></h3>

</body>

</html>

При запуске приложения и переходе по корневому адресу браузер получает **index.html** благодаря UseDefaultFiles(). В инструментах разработчика на вкладке Network видно, что запросы к /css/site.css и /js/app.js возвращают код 200 и правильные MIME-типы. После загрузки страницы фон становится светло-голубым, заголовок окрашивается в тёмно-синий цвет, а кнопки получают отступы и изменённый курсор. При нажатии на кнопки «Сохранить» появляются системные окна prompt, а результаты отображаются в элементе <h3>, демонстрируя корректную работу скрипта.

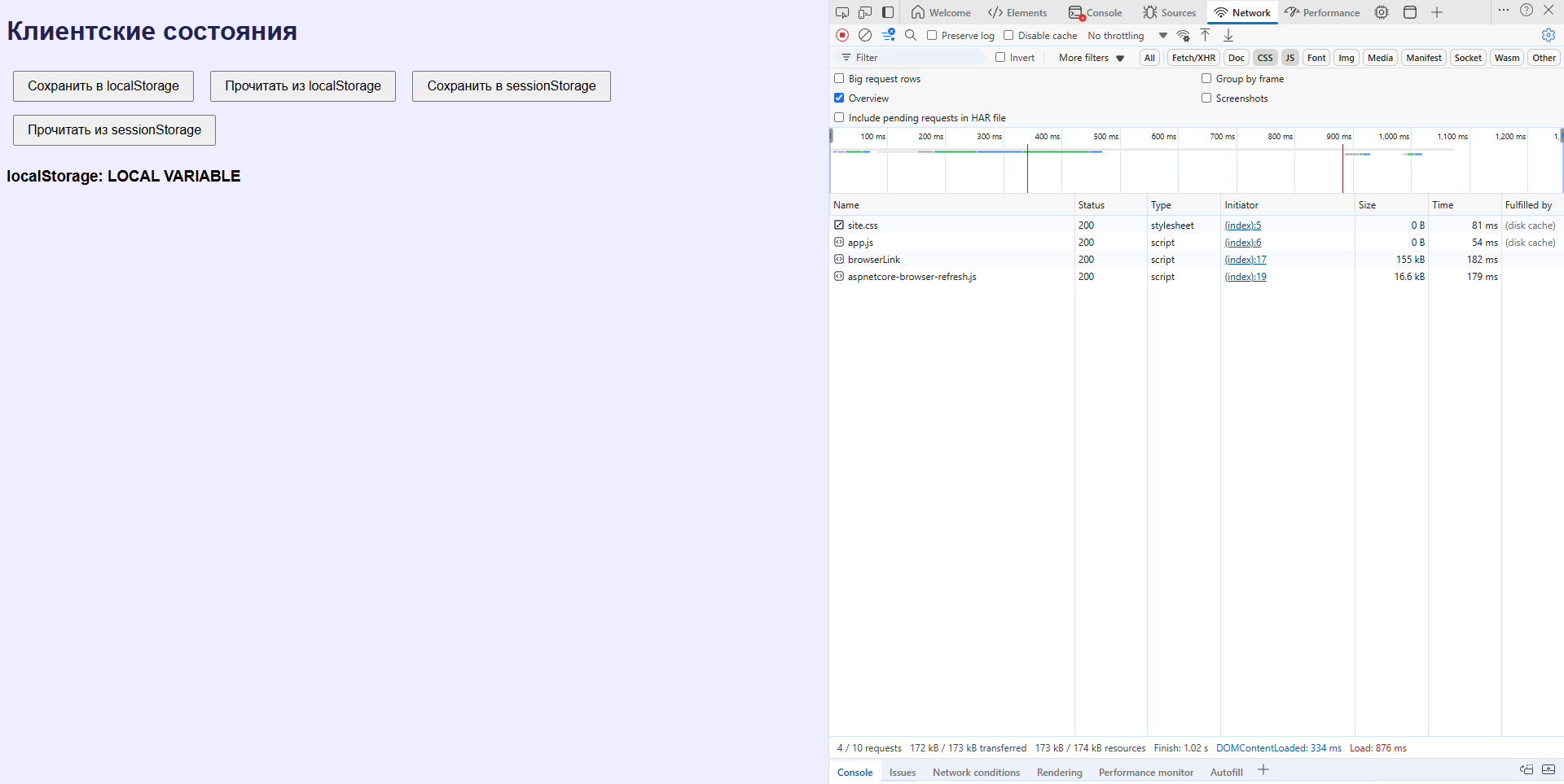


Рисунок . Результат страницы index.html.