

**Асинхронные запросы   
(Fetch API, AJAX)**

**Лаборатория 9**

**Технологии программирования**

**Алимова Малика ИС 24-21**

**🎯 Цель работы**

Освоить работу с асинхронными запросами в веб-приложениях, научиться получать и обрабатывать данные из внешнего REST API и отображать их в интерфейсе пользователя.

**Краткий теоретический обзор**

Fetch API — это современный способ выполнения HTTP-запросов в JavaScript.

Он заменяет старые технологии вроде XMLHttpRequest (используемого в AJAX).

Главная особенность Fetch — это работа с Promise и асинхронностью (async/await).

**Основные этапы работы:**

1. Отправка запроса с помощью fetch(url).

2. Проверка ответа response.ok.

3. Преобразование ответа в JSON: response.json().

4. Работа с полученными данными.

5. Обработка ошибок через try...catch.

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Лабораторная №9 — Fetch API</title>

<style>

body { font-family: Arial, sans-serif; margin: 40px; }

h2 { color: #2a4d8f; }

table { border-collapse: collapse; width: 100%; margin-top: 10px; }

th, td { border: 1px solid #ccc; padding: 8px; text-align: left; }

th { background-color: #f2f2f2; }

#loader { color: blue; font-weight: bold; margin-bottom: 10px; }

button { margin-top: 10px; padding: 8px 12px; background: #2a4d8f; color: white; border: none; border-radius: 5px; cursor: pointer; }

button:hover { background: #1f3768; }

</style>

</head>

<body>

<h2>Список пользователей</h2>

<div id="loader"></div>

<table id="userTable">

<thead>

<tr><th>ID</th><th>Имя</th><th>Email</th><th>Город</th></tr>

</thead>

<tbody></tbody>

</table>

<button id="reload">Обновить данные</button>

<script>

async function loadUsers() {

const loader = document.getElementById('loader');

const tbody = document.querySelector('#userTable tbody');

loader.textContent = 'Загрузка данных...';

tbody.innerHTML = '';

try {

const response = await fetch('https://jsonplaceholder.typicode.com/users');

if (!response.ok) throw new Error('Ошибка сети');

const users = await response.json();

users.forEach(user => {

const row = `

<tr>

<td>${user.id}</td>

<td>${user.name}</td>

<td>${user.email}</td>

<td>${user.address.city}</td>

</tr>`;

tbody.insertAdjacentHTML('beforeend', row);

});

loader.textContent = '';

} catch (error) {

loader.textContent = 'Ошибка при получении данных.';

console.error(error);

}

}

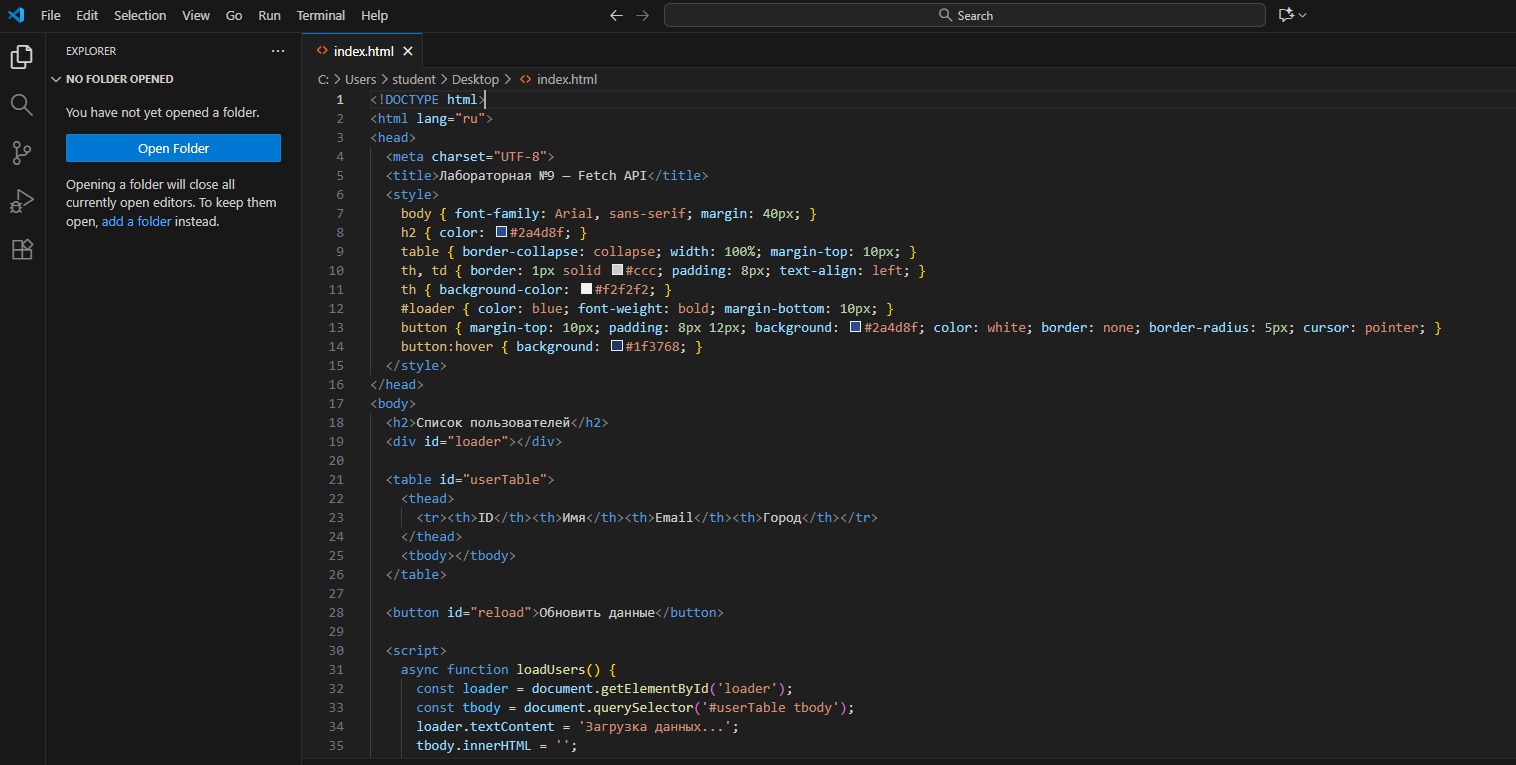
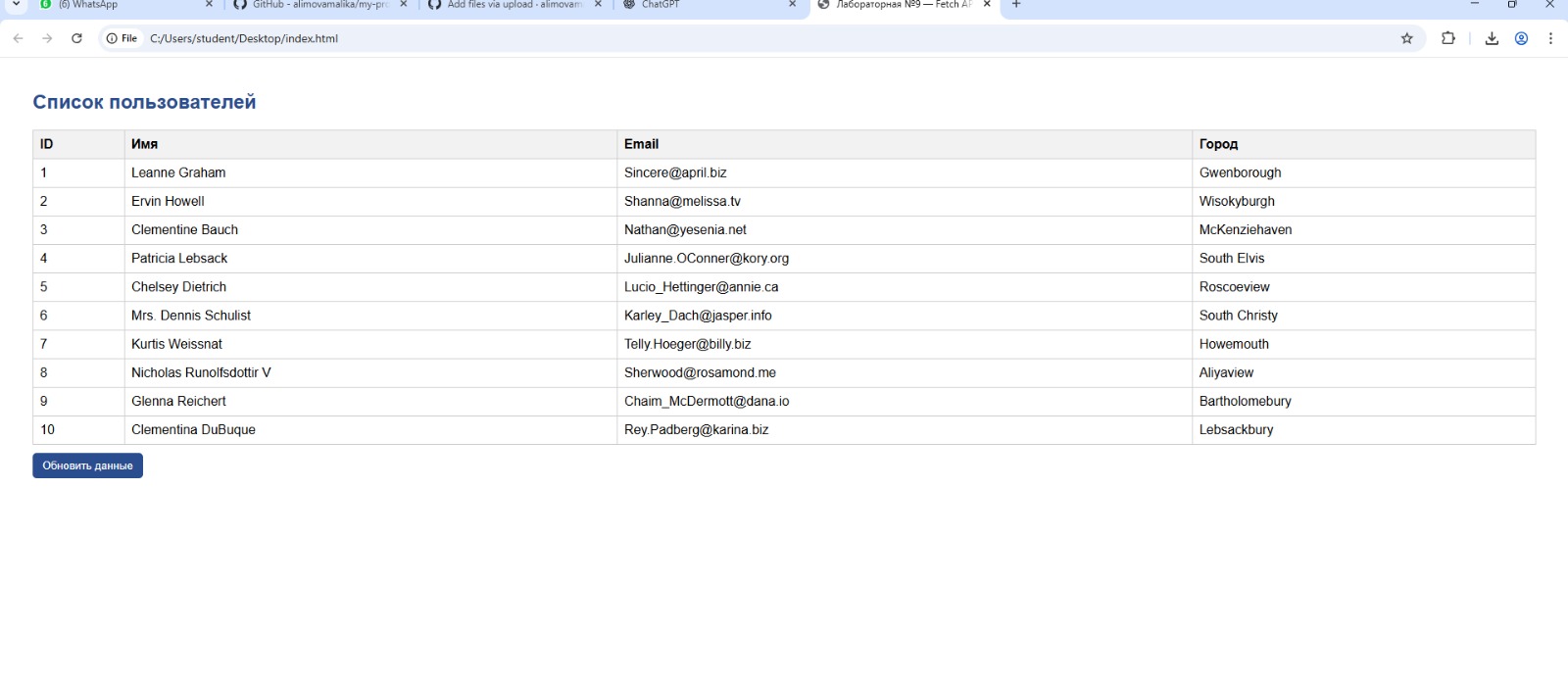
document.getElementById('reload').addEventListener('click', loadUsers);

loadUsers();

</script>

</body>

</html>



**Вывод**

В ходе выполнения лабораторной работы было освоено использование Fetch API для выполнения асинхронных запросов и получения данных из REST API. Полученные данные отображались в виде таблицы, а также реализована обработка ошибок и отображение состояния загрузки. Работа позволила понять принципы работы асинхронности и взаимодействия клиентской части с внешними ресурсами.