

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Цой Степан

Группа
К3440

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

Задачи и цели

Цель работы — освоить принципы взаимодействия фронтенда с backend-сервисом через REST API. Изучить методы HTTP, форматы передачи данных, асинхронные запросы с помощью `fetch` и `async/await`.

- Развернуть локальный REST API с помощью `json-server`
- Реализовать CRUD-операции: получение, создание и удаление данных
- Применить асинхронные функции `async/await` для работы с промисами
- Реализовать фильтрацию и поиск через параметры запроса
- Разработать систему аутентификации на `localStorage`

Ход работы

Структура базы данных (db.json)

```
{
  "recipes": [
    {
      "id": 1,
      "title": "Борщ украинский",
      "type": "soup",
      "difficulty": "medium",
      "cookTime": 90,
      "ingredients": ["свёкла", "капуста", "морковь"],
      "likes": 42
    }
  ],
  "comments": [
    { "id": 1, "recipeId": 1, "author": "Анна", "text": "Очень вкусно!" }
  ]
}
```

Запуск сервера

```
// package.json
"scripts": {
  "api": "json-server --watch db.json --port 3000"
}

// Запуск в терминале:
npm run api
// Сервер доступен: http://localhost:3000
```

Фильтрация рецептов

```
async function filterRecipes(type, query) {
  let url = 'http://localhost:3000/recipes?';
  const params = new URLSearchParams();
  if (type && type !== 'all') params.append('type', type);
  if (query) params.append('q', query);
  const data = await loadRecipes(url + params.toString());
  renderCards(data);
}

// Привязка к элементам управления:
document.getElementById('filter-type').addEventListener('change', e => {
  filterRecipes(e.target.value, searchInput.value);
});
```

Добавление комментария (POST)

```
async function addComment(recipeId, text) {
  const user = JSON.parse(localStorage.getItem('user'));
  const response = await fetch('http://localhost:3000/comments', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({
      recipeId,
      text,
      author: user?.name || 'Аноним',
      createdAt: new Date().toISOString()
    })
  });
  if (!response.ok) throw new Error('Ошибка сохранения');
  return response.json();
}
```

Вывод

В ходе работы освоен полный цикл взаимодействия фронтенда с REST API. json-server позволил эмулировать backend без написания серверного кода, что значительно ускорило разработку. Паттерн `async/await` обеспечил читабельный асинхронный код с явной обработкой ошибок. Параметры запроса (`?type`, `?q`) позволили реализовать фильтрацию и поиск без дополнительной логики на сервере. Понимание HTTP-методов и кодов ответа является фундаментом для работы с любыми реальными API.