

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Попов Никита

К3344

**Проверил:
Добряков Д. И.**

Санкт-Петербург

2025 г.

Тема: Взаимодействие с внешним API (JSON-server)

Вариант: Приложение для бронирования столов в ресторанах.

Цель работы

- Научиться работать с внешним API с помощью fetch/axios.
- Реализовать моковое REST-API через JSON-server и json-server-auth для авторизации.
- Перевести функционал ЛР1 на получение данных из API.

Ход работы:

Взаимодействие с внешним API:

Создан файл db.json

db.json

```

1  {
2      "restaurants": [
3          {
4              "id": "1",
5                  "name": "Итальянский Ресторан",
6                  "cuisine": "Итальянская",
7                  "location": "Центр",
8                  "price": "₽₽",
9                  "images": [
10                      "assets/images/italian.jpg",
11                      "assets/images/italian2.jpg"
12                  ],
13                  "menu": [
14                      "Паста Карбонара – 450 ₽",
15                      "Маргарита – 350 ₽",
16                      "Тирамису – 250 ₽"
17                  ],
18                  "reviews": [
19                      {
20                          "name": "Иван",
21                          "text": "Отличная паста и уютная атмосфера!"
22                      },
23                      {
24                          "name": "Мария",
25                          "text": "Очень понравились десерты."
26                      }
27                  ]
28              },
29              {
30                  "id": "2",
31                  "name": "Суши Бар",
32                  "cuisine": "Японская",
33                  "location": "Север",
34                  "price": "₽₽₽",
35                  "images": [
36                      "assets/images/sushi.jpg"
37                  ],
38                  "menu": [
39                      "Суши".

```

В script.js реализована вся логика работы с сервером json

```

> assets           135
> node_modules    136
js api.js          137
() db.json         138
< index.html       139
js middleware.js   140
() package-lock.json 141
() package.json     142
< profile.html     143
< README.md        144
< restaurant.html 145
js script.js        146
# style.css         147
< lab3 /restaurant-b... 148
> .bin             149
> .vite            150
                           151
                           152
                           153

```

```

async function loadRestaurants(filters = {}) {
    try {
        let restaurants;
        if (Object.keys(filters).length === 0 ||
            (!filters.cuisine && !filters.location && !filters.price)) {
            restaurants = await restaurantsAPI.getAll();
        } else {
            restaurants = await restaurantsAPI.search(filters);
        }
        render(restaurants);
    } catch (error) {
        showToast(error.message || "Ошибка загрузки ресторанов");
        console.error(error);
        list.innerHTML = '<div class="col-12"><p class="text-center text-danger">Ошибка загрузки ресторанов</p></div>';
    }
}

```

Сам сервер поднимается через npm с помощью команды npm run server

```
> restaurant-booking-api@1.0.0 server
> json-server --watch db.json --port 3000 --middlewares ./middleware.js

\{^_^\}/ hi!

Loading db.json
Loading ./middleware.js
Done

Resources
http://localhost:3000/restaurants
http://localhost:3000/users
http://localhost:3000/bookings

Home
http://localhost:3000

Type s + enter at any time to create a snapshot of the database
Watching...

GET /restaurants 200 5.290 ms - -
```

Заключение

В рамках данной лабораторной работы была реализована интеграция ранее разработанного интерфейса с внешним API. Для имитации серверной части использовался JSON-сервер с настроенной авторизацией. Взаимодействие с API осуществлялось с помощью средств работы с HTTP-запросами (fetch/axios/xhr). В результате была отработана логика получения и отправки данных, а также закреплены навыки работы с клиент-серверным взаимодействием и имитацией реального API.