САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Выполнила: Лапшина Екатерина

Группа: К3440

Проверил: Добряков Д. И.

Санкт-Петербург

2025 г.

Задание

Добавить к проекту, реализованному в ЛР1 подключение к внешнему API средствами fetch/axios/xhr. Реализовать моковое API средствами JSON-сервера и подключить к нему авторизацию.

Проект: Сервис для аренды недвижимости.

Ход работы

В предыдущей лабораторной работе был реализован сайт, оперирующий статическими данными. Для того, чтобы получать данные динамически был создан моковый сервер с использованием инструмента JSON-server. Для хранения данных был сформирован файл db.json, содержащий информацию о пользователях, объектах недвижимости и бронированиях. JSON-server автоматически предоставляет REST API на основе структуры данного файла. Для запуска сервера использовалась команда json-server --watch db.json --port 3000. В результате был получен локальный API по адресу http://localhost:3000.

Для взаимодействия с сервером был создан класс ApiService. Основные методы для работы с API представлены в листинге ниже:

```
/* js/api.js */
const BASE_URL = 'http://localhost:3000';
class ApiService {
    static async login(email, password) {
        const response = await fetch(
            `${BASE_URL}/users?email=${email}&password=${password}`
        );
        const users = await response.json();
        if (users.length > 0) {
            return users[0];
        }
        throw new Error('Неверный email или пароль');
    }

    static async getProperties(filters = {}) {
        let url = `${BASE_URL}/properties`;
        const params = new URLSearchParams();

        if (filters.type) params.append('type', filters.type);
        if (filters.minPrice) params.append('price_gte',filters.minPrice);
        if (filters.maxPrice) params.append('price_lte',filters.maxPrice);
```

```
        if (params.toString()) {
            url += `?${params.toString()}`;
        }

        const response = await fetch(url);
        return await response.json();
    }

    static async getUserBookings(userId) {
        const response = await fetch(
            `${BASE_URL}/bookings?userId=${userId}&_expand=property`
        );
      return await response.json();
    }
}
```

Листинг 1 – Класс ApiService для работы с API

Авторизация

Для реализации авторизации использовалась форма входа, которая отправляет введенные данные через ApiService.login. При успешном входе данные пользователя сохраняются в localStorage:

```
/* login.html (script) */

document.getElementById('loginForm').addEventListener('submit',
    async (e) => {
          e.preventDefault();
          const email = document.getElementById('email').value;
          const password = document.getElementById('password').value;

          try {
                const user = await ApiService.login(email, password);
                localStorage.setItem('user', JSON.stringify(user));
                window.location.href = 'profile.html';
          } catch (error) {
                // Обработка ошибки
          }
    }
);
```

Листинг 2 – Обработка формы авторизации

Поиск недвижимости

На странице поиска реализована загрузка списка объектов с сервера с учетом фильтров:

```
/* search.html (script) */
```

```
async function loadProperties(filters = {}) { try
    {
           const properties = await ApiService.getProperties(filters);
           displayProperties(properties);
     } catch (error) {
           // Отображение ошибки
    }
}

searchForm.addEventListener('submit', (e) => {
     e.preventDefault();
     const filters = {
           type: document.getElementById('inputType').value,
           minPrice: document.getElementById('inputMinPrice').value,
           maxPrice: document.getElementById('inputMaxPrice').value
    };
     loadProperties(filters);
});
```

Листинг 3 – Загрузка объектов недвижимости с фильтрацией

Личный кабинет

В личном кабинете отображаются данные текущего пользователя и его бронирования, полученные через API:

```
/* profile.html (script) */

async function loadBookings() {
const bookings = await ApiService.getUserBookings(user.id);
     bookings.forEach(booking => {
           // Отображение бронирования и связанного объекта недвижимости
           const property = booking.property;
           // ... создание HTML элементов ...
     });
}
```

Листинг 4 – Загрузка бронирований пользователя

Вывод

В ходе выполнения лабораторной работы статические данные были заменены на данные с мокового json-сервера с помощью Fetch API. Была реализована авторизация, регистрация пользователей, поиск объектов с фильтрацией и отображение данных в личном кабинете.