САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Платонова Александра

Группа K3439

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

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: Прикладное программное обеспечение деятельности отдела заселения муниципальных общежитий администрации города.

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, общежитиях, комнатах и жильцах. JSON-server автоматически предоставляет REST API на основе структуры данного файла. Для запуска сервера использовалась команда npx json-server --watch db.json --port 3000. В результате был получен локальный API по адресу http://localhost:3000.

Далее в структуре мокового API были определены основные сущности системы. Сущность users используется для хранения данных пользователей и реализации авторизации. Сущности dormitories и rooms предназначены для хранения информации об общежитиях и комнатах с использованием связей по идентификаторам. Сущность residents содержит данные о студентах, проживающих в конкретных комнатах. Взаимодействие с моковым сервисом осуществляется посредством axios.

```
const api = axios.create({
baseURL: 'http://localhost:3000'
});
```

Пример функции для загрузки общежитий и комнат.

```
async function loadDormitories() {
try {
const response = await api.get('/dormitories');
const dormitories = response.data;
```

```
      const                          container                          =
document.getElementById('dormitoriesContainer');

      container.innerHTML = '';



      dormitories.forEach(dorm => {

      container.innerHTML += `

             <div class="col-md-6">

               <div class="card">

                 <div class="card-header fw-bold">${dorm.name}</div>

                              <ul  class="list-group  list-group-flush"
id="rooms-${dorm.id}"></ul>

               </div>

             </div>

           `;

      loadRooms(dorm.id);

      });

      } catch (error) {

      console.error(error);

      }

      }
```

Код для поиска комнат:

```


      async function searchRooms() {

      const dormId = document.getElementById('dormSelect').value;

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

      roomSelect.innerHTML       =       '<option       value="">Выберите
комнату</option>';
```

```
    if (!dormId) return;

    const response = await api.get('/rooms', {

    params: { dormitoryId: dormId }

    });

    response.data.forEach(room => {

    roomSelect.innerHTML += `

        <option value="${room.id}">Комната ${room.id}</option>

      `;

    });

    }
```

Для реализации авторизации и регистрации была реализована функция для хэширования паролей, чтобы не хранить пароли в открытом виде в файле.

```
    async function hashPassword(password) {

      const encoder = new TextEncoder();

      const data = encoder.encode(password);

        const hashBuffer = await crypto.subtle.digest('SHA-256',
data);

      return Array.from(new Uint8Array(hashBuffer))

      .map(b => b.toString(16).padStart(2, '0'))

      .join('');

    }
```

С использованием хэширования код для авторизации будет выглядеть следующим образом:

```
    async function login(e) {

      e.preventDefault();

            const email = document.querySelector('#loginScreen
input[type="email"]').value;
```

```
            const password = document.querySelector('#loginScreen
input[type="password"]').value;

        try {

        const passwordHash = await hashPassword(password);

        const response = await api.get('/users', {

        params: {

        email: email,

        password: passwordHash

        }

        });

        if (response.data.length > 0) {

                                        localStorage.setItem('user',
JSON.stringify(response.data[0]));

        showScreen('dashboardScreen');

        loadDormitories();

        loadPayments();

        } else {

        alert('Неверный логин или пароль');

        }

        } catch (error) {

        console.error(error);

        alert('Ошибка авторизации');

        }

    }
```

Вывод

В ходе выполнения лабораторной работы статические данные были заменены на данные с мокового json-сервера с помощью axios. Также была реализована авторизация и хэширование паролей.