САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа 2

Выполнил: Пластун Елизавета

Группа: K33402

Проверил: Добряков Д. И.

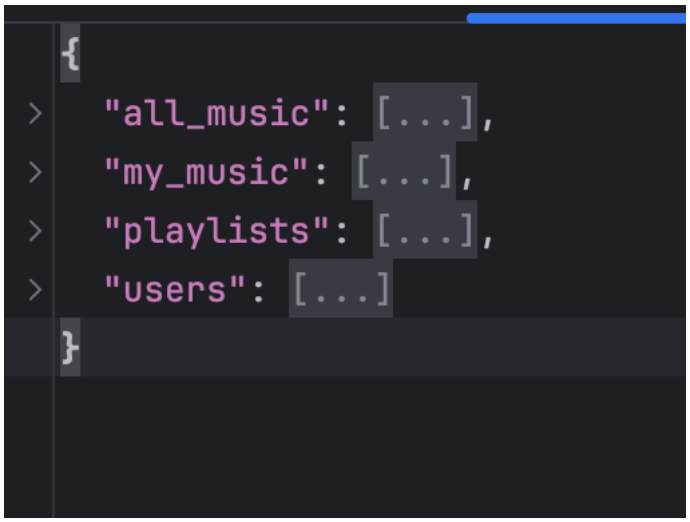
Санкт-Петербург

2023 г.

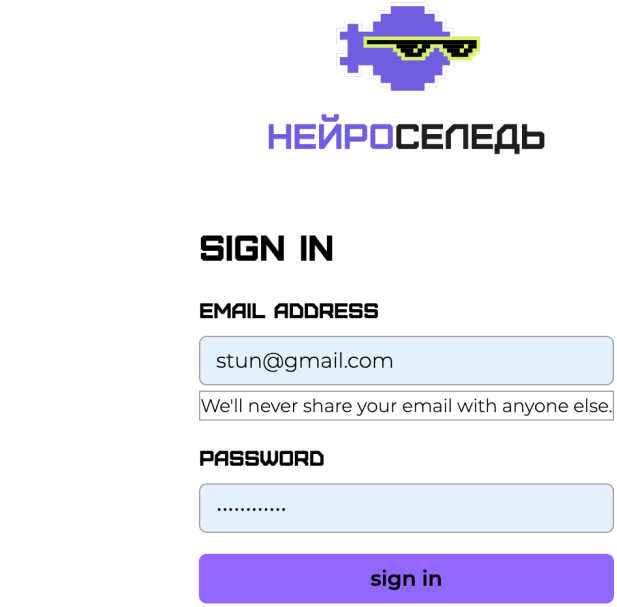
Задача Варианты остаются прежними. Теперь Вам нужно привязать то, что Вы делали в ЛР1 к внешнему API средствами fetch/ axios/xhr.

Ход работы

Сначала был реализован json сервер



Далее была реализована авторизация и регистрация пользователей




```
POST /users 201 11.884 ms - 72
GET /users 200 13.920 ms - 367
GET /all_music 200 13.779 ms - 744
GET /my_music 200 18.171 ms - 381
GET /playlists 200 6.385 ms - -
GET /all_music 200 16.896 ms - 744
GET /my_music 200 15.030 ms - 381
GET /playlists 200 28.494 ms - -
GET /all_music 200 10.135 ms - 744
GET /my_music 200 11.855 ms - 381
GET /playlists 200 13.717 ms - -
GET /all_music 200 5.822 ms - 744
```

Как видно из скринов авторизация прошла успешно и уже подключился скрипт подгрузки музыки, но про это позже. Была реализована регистрация

НЕЙРОСЕЛЕДЬ

REGISTER

EMAIL ADDRESS

new@gmail.com

We'll never share your email with anyone else.

PASSWORD

Sign up

```
GET /my_music 200 5.278 ms - 381
GET /playlists 200 5.121 ms - -
POST /users 201 11.921 ms - 69
GET /users 200 3.924 ms - 448
GET /all_music 200 5.740 ms - 744
GET /my_music 200 6.658 ms - 381
GET /playlists 200 7.539 ms - -
POST /users 201 11.251 ms - 73
```


```
  },
  {
    "email": "new@gmail.com",
    "password": "1234567890",
    "id": 5
  },
  {
    "email": "newww@gmail.com",
    "password": "fdghjk56dFfe",
    "id": 6
  }
]
}
```

Как видно из скринов новые пользователи успешно сохраняются на сервере.

Скрипт авторизации:

```
const SERVER_URL = 'http://localhost:3000';

async function login(event) {
    try {
        event.preventDefault();
        const formData = new FormData(event.target);
        const email = formData.get('email');
        const password = formData.get('password');

        const response = await fetch(`${SERVER_URL}/users`);
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        const users = await response.json();
        const user = users.find(user => user.email === email && user.password === password);
        if (user) {
            window.location.href = 'music.html';
            console.log('Аутентификация успешна');
        } else {
            console.error('Ошибка аутентификации');
        }
    } catch (error) {
        console.error('Произошла ошибка:', error);
    }
}
```

Скрипт регистрации:

```
async function register(event) {
    try {
        event.preventDefault();
        const formData = new FormData(event.target);
        const email = formData.get('email');
        const password = formData.get('password');

        if (!email || !password) {
            throw new Error('Email и пароль обязательны для регистрации');
        }

        const response = await fetch(`${SERVER_URL}/users`, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ email, password })
        });

        if (response.ok) {
            window.location.href = 'sign_in.html';
            console.log('Регистрация успешна');
        } else {
            console.error('Ошибка регистрации');
        }
    } catch (error) {
        console.error('Произошла ошибка:', error);
    }
}
```

Далее была реализована подгрузка песен с сервера и генерация их на странице:

Хранение музыки:

```
{
  "all_music": [
    {
      "id": "1",
      "name": "конечности",
      "artist": "8th",
      "filename": "8th-konechnosti.mp3"
    },
    {
      "id": "2",
      "name": "loveshit",
      "artist": "8th",
      "filename": "8th-loveshit.mp3"
    },
    {
      "id": "3",
      "name": "Пыяла",
      "artist": "АИГЕЛ",
      "filename": "aigel-pyyala-mp3.mp3"
    },
    {
      "id": "4",
      "name": "Slavic Blood",
      "artist": "Kraenkova",
      "filename": "kraenkova-slavic-blood-mp3.mp3"
    },
    {
      "id": "5",
      "name": "Zaritsa",
      "artist": "Kraenkova",
      "filename": "kraenkova-zaritsa-mp3.mp3"
    },
```

Подгрузка всей музыки:

```
async function getDataAllMusic() {
    try {
        const response = await fetch(`${SERVER_URL}/all_music`);
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        const data = await response.json();
        return Array.isArray(data) ? data : [];
    } catch (error) {
        console.error('Произошла ошибка при получении данных:', error);
        return [];
    }
}
```

Подгрузка моей музыки:

```
async function getDataMyMusic() {
    try {
        const response = await fetch(`${SERVER_URL}/my_music`);
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        const data = await response.json();
        return Array.isArray(data) ? data : [];
    } catch (error) {
        console.error('Произошла ошибка при получении данных:', error);
        return [];
    }
```

Подгрузка моих плэйлистов:

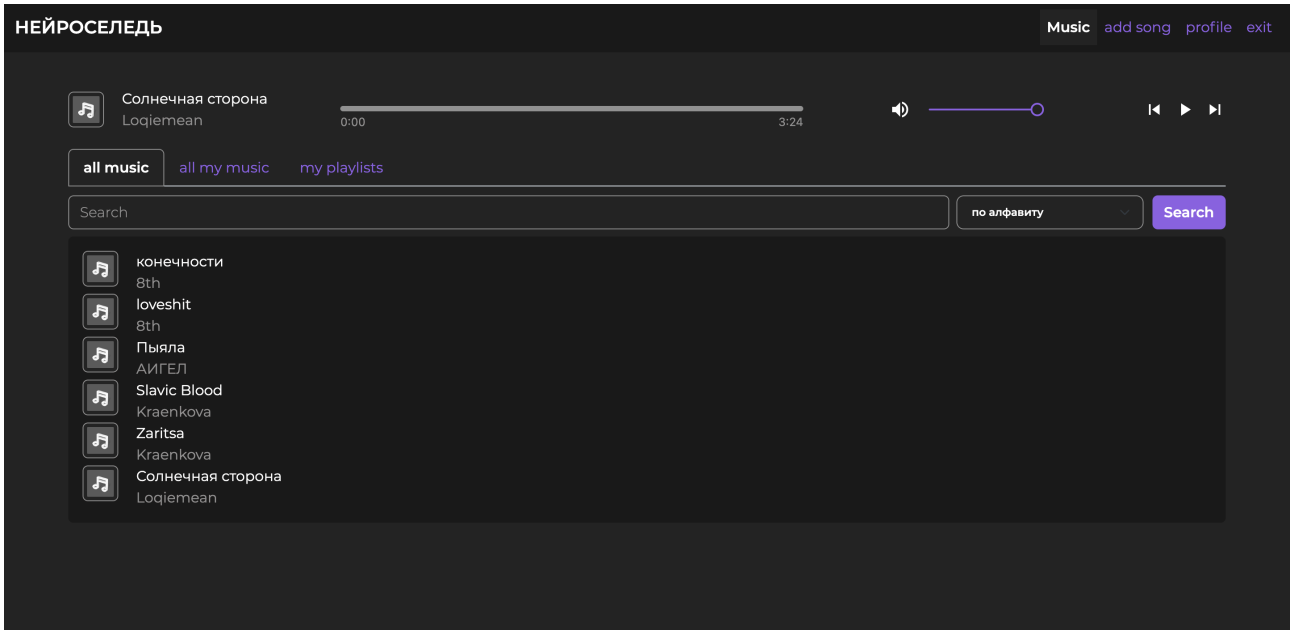
```
async function getDataMyPlaylists() {
    try {
        const response = await fetch(`${SERVER_URL}/playlists`);
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        const data = await response.json();
        return Array.isArray(data) ? data : [];
    } catch (error) {
        console.error('Произошла ошибка при получении данных:', error);
        return [];
    }
}
```

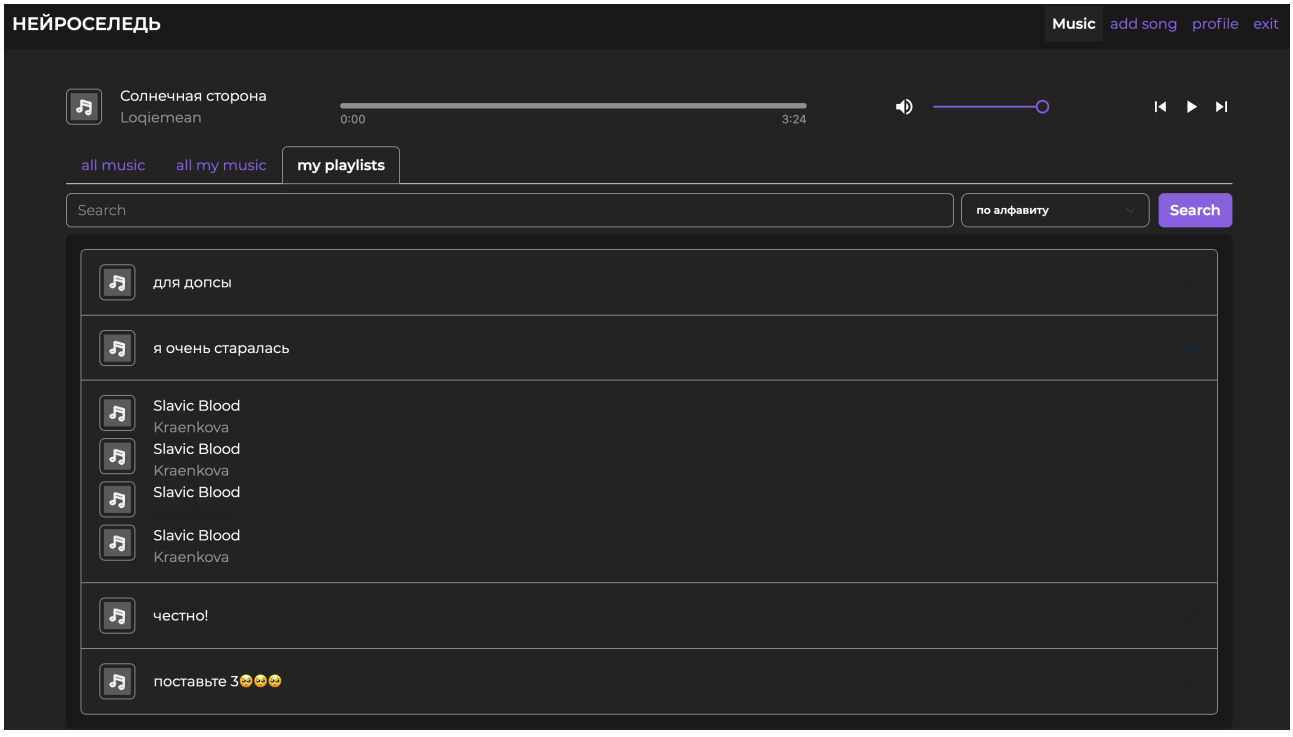
Отображение списка музыки:

```
function createHtmlSongList(allMusic, parent) {
    allMusic.forEach((song) => {
        const songDiv = document.createElement('div');
        songDiv.classList.add('song-container-main');
        const songPhotoDiv = document.createElement('div');
        songPhotoDiv.classList.add('container-song-photo');
        const songInfoDiv = document.createElement('div');
        songInfoDiv.classList.add('song-container');
        const songImage = document.createElement('img');
        songImage.classList.add('img-thumbnail', 'resize')
        songImage.src = `../../src/img/1.png`;
        songImage.setAttribute('alt', 'изображение песни');
        const songName = document.createElement('div');
        songName.classList.add('nav-link', 'black');
        songName.textContent = song.name;
        const songArtist = document.createElement('div');
        songArtist.classList.add('nav-link', 'grey');
        songArtist.textContent = song.artist;

        songInfoDiv.appendChild(songName);
        songInfoDiv.appendChild(songArtist);
        songPhotoDiv.appendChild(songImage);
        songPhotoDiv.appendChild(songInfoDiv);
        songDiv.appendChild(songPhotoDiv);
        parent.appendChild(songDiv);

        songDiv.addEventListener('click', function() {
            const songId = song.id;
            loadMusic(songId, allMusic);
        });
    });
}
```

И еще много много логики плела…

Результат:

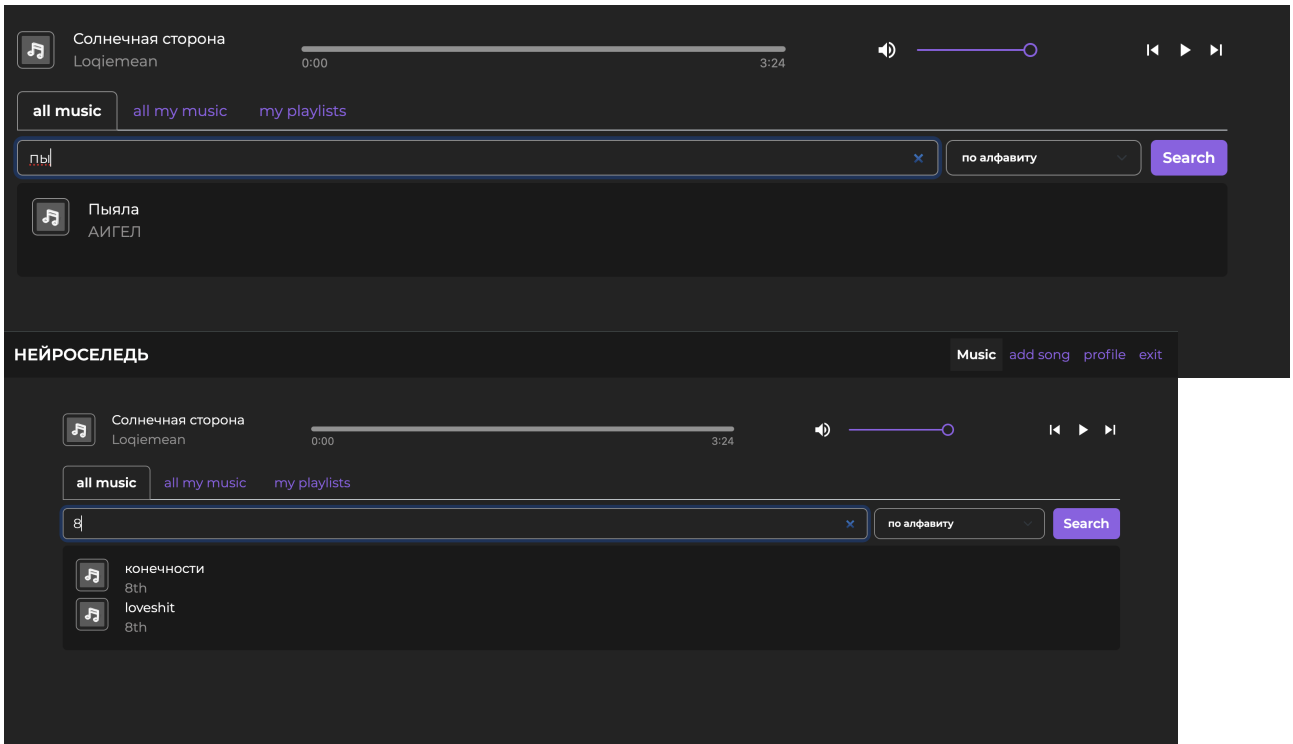

НЕЙРОСЕЛЕДЬ
Music
add song
profile
exit
Солнечная сторона
Loqiemean
0:00
3:24
all music
all my music
my playlists
Search
по алфавиту
Search
конечности
8th
loveshit
8th
Пыяла
АИГЕЛ
Slavic Blood
Kraenkova
Zaritsa
Kraenkova
Солнечная сторона
Loqiemean


НЕЙРОСЕЛЕДЬ
Music
add song
profile
exit
Солнечная сторона
Loqiemean
0:00
3:24
all music
all my music
my playlists
Search
по алфавиту
Search
для допсы
я очень старалась
Slavic Blood
Kraenkova
Slavic Blood
Kraenkova
Slavic Blood
Slavic Blood
Kraenkova
честно!
поставьте 3

Так же был реализован поиск:

```
async function searchSongs(query) {
    try {
        const response = await fetch(`${SERVER_URL}/all_music?q=${encodeURIComponent(query)}`);
        if (!response.ok) {
            throw new Error('Network response was not ok');
        }
        const data = await response.json();
        return Array.isArray(data) ? data : [];
    } catch (error) {
        console.error('Произошла ошибка при поиске песен:', error);
        return [];
    }
}
searchInput.addEventListener('input', async () => {
    const query = searchInput.value.trim();
    const parentContainer = songContainerAll;
    if (query === '') {
        while (parentContainer.firstChild) {
            parentContainer.removeChild(parentContainer.firstChild);
        }
        const allMusic = await getDataAllMusic();
        createHtmlSongList(allMusic, parentContainer);
    } else {
        while (parentContainer.firstChild) {
            parentContainer.removeChild(parentContainer.firstChild);
        }
        const searchResults = await searchSongs(query);
        createHtmlSongList(searchResults, parentContainer);
    }
});
```

Результат работы:



Вывод: в ходе выполнения данной лабораторной работы я освоила работу с api и сервером. Реализовала логику веб приложения и знатно намучалась с js.