САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет Лабораторной работы №2

Взаимодействие с внешним API

Выполнил:
Бункута Натан Селест
Группа К33412

Проверила:
Давид Добияков

Санкт-Петербург 2022

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

Ход работы:

Для начала использовал внешний API сайта unsplash.com чтобы получить доступ к фотографии. Требовалось ключ что для доступа, я тогда зарегистрировался потом генерировали для меня личный ключ.

- AppService.js

мы создали файл ApiService.js в котором мы вызывали внешний API

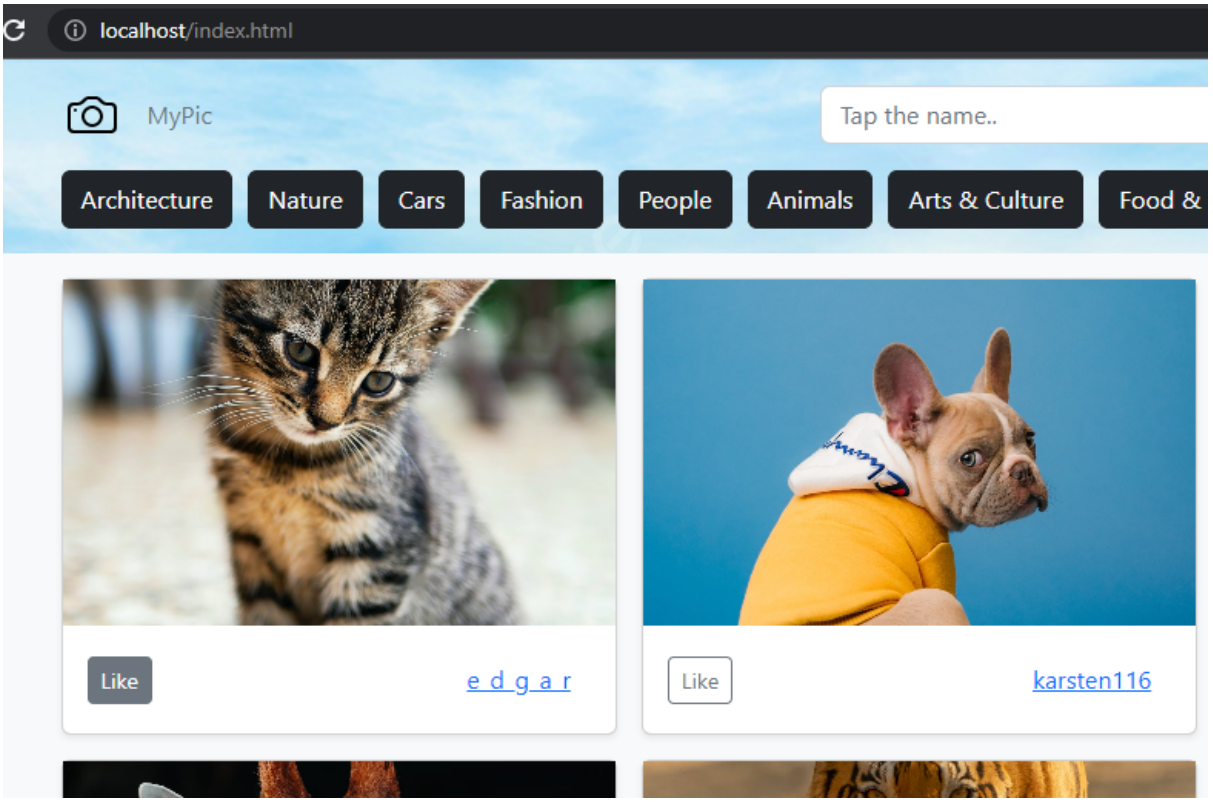
```
import {fetchHandler} from "./utils.js";

export class ApiService{
    constructor() {
        this.api_key = "BOkeOE9NuQ8QFpfoIXnu0zC7gD-FgG7ocyZynYSC4hA"
        this.base_url = "https://api.unsplash.com"
    }

    url(path, params= {}){
        let query = "?";
        params["client_id"]=this.api_key
        Object.entries(params).forEach(e=>{
            query+=`${e[0]}=${e[1]}&`
        })
        query = query.slice(0,-1)
        return `${this.base_url}${path}/${query}`
    }

    async fetchRandomPhotos(){
        return await fetchHandler(this.url("/photos/random", {count:27}))
    }

    async fetchPhotosByQuery(query){
        const response = await fetchHandler(this.url("/search/photos", {query:query, per_page:27}))
        return response.results
    }
}

```

- App.js

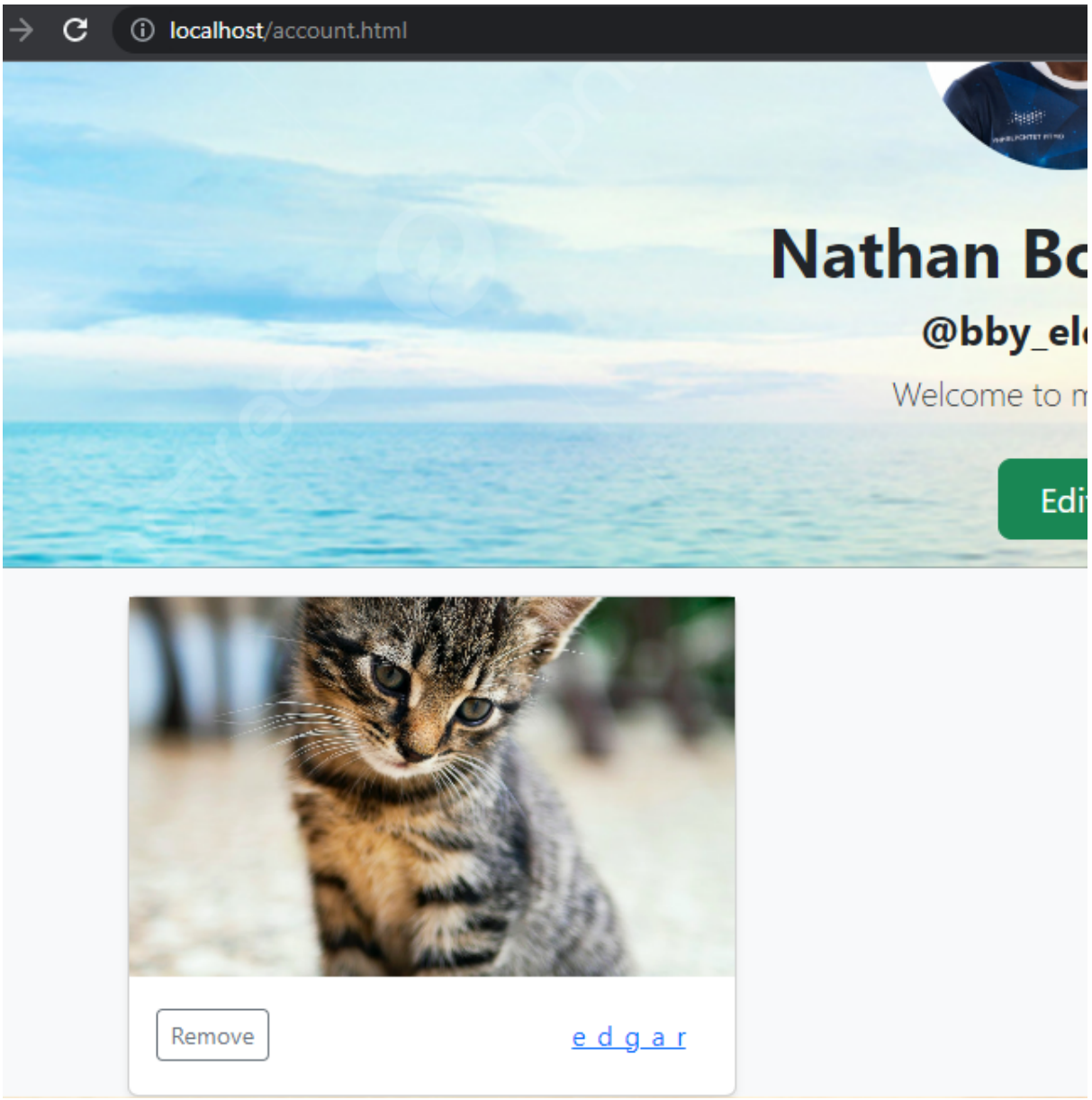
Именно этот файл (приложение) позволяет нам сохранять где-нибудь те фотографии, которые мы лайкаем и даже можем их удалить оттуда.

```js
import {ApiService} from "./ApiService.js";

if(!localStorage.getItem('likes')){
    localStorage.setItem('likes', [])
}

const api = new ApiService()
export let photos = await api.fetchRandomPhotos()

export const likes = () => {
    let data = localStorage.getItem('likes')
    if(data){
        return JSON.parse(data)
    }else{
        return []
    }
}

export const handleLike = (photo) => {
    let data = likes()
    if(!data.find(e => e.id === photo.id)){
        data.push(photo)
        localStorage.setItem('likes', JSON.stringify(data))
    }
}

export const removeLike = (photo) => {
    localStorage.setItem('likes', JSON.stringify((likes().filter(e => e.id !== photo.id))))
    clearAlbum()
    renderAlbum(likes(), removeButton)
}
```

Давайте пример..
Лайкнем одно фото на странице index.html



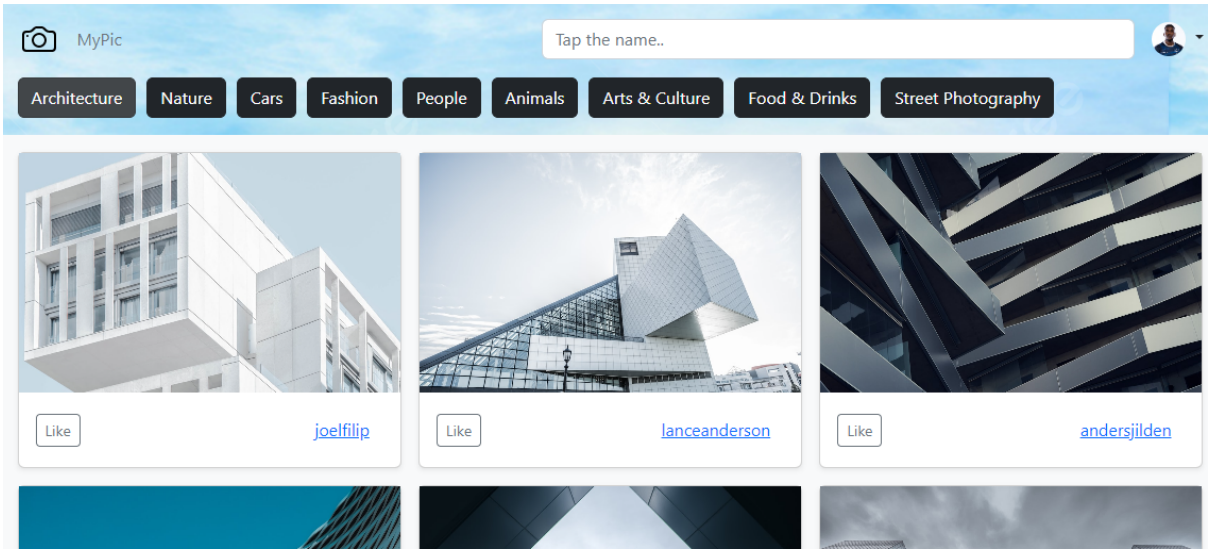
Переходим на странице Account.html и смотрим сохранялось ли фото



В этом же файле, использовал функцию renderFilters для того, чтобы фильтровать поиски. Предлагает готовые типы фотографии для быстрого поиска.

```
export const renderFilters = () => {
    const filters = ["Cities", "Nature", "Health", "Fashion & style", "People", "Animals", "Arts & Culture", "Sport", "Houses"]
    const filtersField = document.getElementsByClassName("categories_filter")
    filters.forEach(filter => {
        const btn = document.createElement('button')
        btn.className = "btn btn-dark"
        btn.textContent = filter
        btn.addEventListener("click", async () => {
            await queryFilter(filter.toLowerCase())
        })
        filtersField[0].appendChild(btn)
    })
}

export const clearAlbum = () => {
    const album = document.getElementById('album')
    while (album.firstChild) {
        album.firstChild.remove();
    }
}

export const likeButton = (photo) => {
    const like_button = document.createElement('button')
    like_button.type="button"
    like_button.className="btn btn-sm btn-outline-secondary"
    like_button.textContent="Like"
    like_button.onclick = handleLike.bind(null, photo)
    return like_button
}

export const removeButton = (photo) => {
    const removeButton = document.createElement('button')
    removeButton.type="button"
    removeButton.className="btn btn-sm btn-outline-secondary"
    removeButton.textContent="Remove"
    removeButton.onclick = removeLike.bind(null, photo)
    return removeButton
}

export const viewPhotoModal = (photo) => {

```

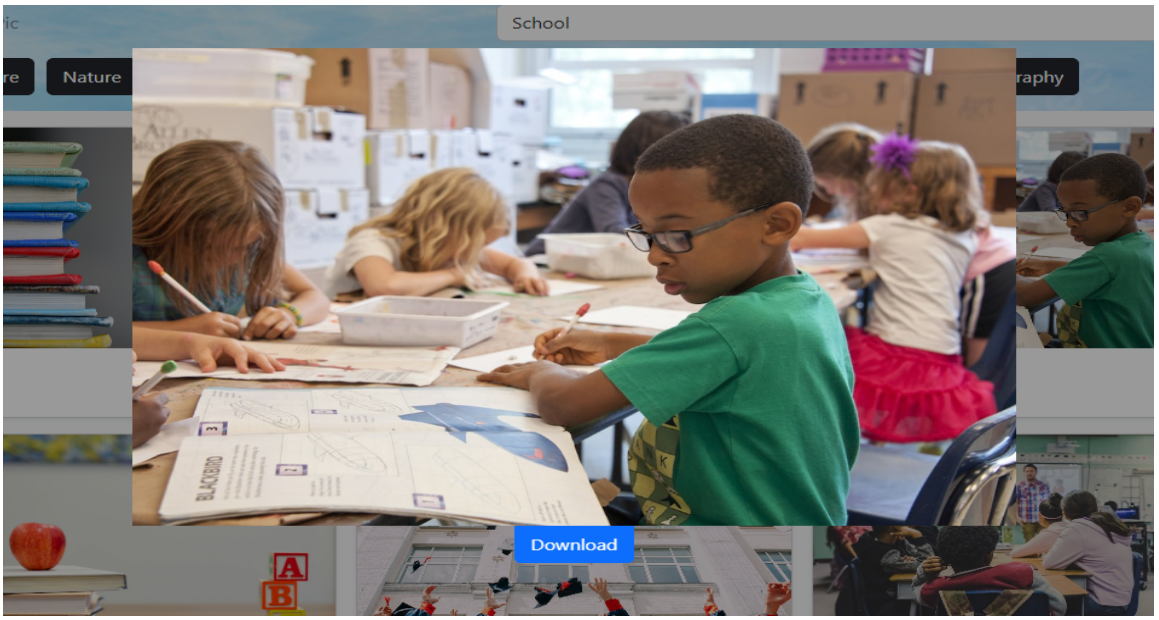
Например: Хотим смотреть только фото красивых зданий то можно нажать на кнопке “Architecture”



- viewPhotoModal

```
export const viewPhotoModal = (photo) => {


    const modal = document.querySelector(".myModal")
    modal.style.display = "flex";

    const content = document.createElement("div")
    content.className="myContent"

    const img = document.createElement("img")
    img.src=photo.urls.regular;
    img.className="content"


    const download = document.createElement("a")
    download.href=photo.links.download
    download.className="rounded-0 rounded-bottom btn btn-success"
    download.textContent = "Download"

    window.onclick = function(event) {
        if (event.target === content || event.target === modal) {
            modal.style.display = "none";
            while (modal.firstChild) {
                modal.firstChild.remove();
            }
        }

    }

    modal.appendChild(content)
    content.appendChild(img)
    content.appendChild(download)

}
```

Одна из интересных функций приложения у меня является функцией `viewPhotoModal`. Она нам нужно для предпросмотра фотографии нажимая на неё и возможность скачать



- main.js

```
import {renderAlbum, renderFilters, photos, searchBar, likeButton} from "./app.js";

renderAlbum(photos, likeButton)
renderFilters()
searchBar()
```

main.js импортирует функции из app.js для того чтобы поиске рандомно отобразились фотографии.

Вывод:

При выполнении данной лабораторной работе получили доступ к API внешнего сайта потом применили его в нашем с помощью метода fetch. Узнали как работает API и как сайты взаимодействуют друг с другом.