

**САНКТ-ПЕТЕРБУРГСКИЙ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд

разработка Отчет

Лабораторная работа
№2

Выполнил: Казанков
Илья K33402

Провер

ил:

Добряков

Д. И

Санкт-Петербурб

ург 2024 г.

Задача

Привязать сверстаный сайт к внешнему API средствами fetch/axios/xhr.

Вариант: сайт-портфолио

Ход работы

Было реализовано получение карточек работ (их наполнения) с внешнего API. Выполнена реализация сортировки этих карточек по категориям и поиск по ним

Создан скрипт addCards.js, в котором написаны две функции:

- getCardHtml, которая возвращает html-код карточки услуги
- loadServices, которая делает запрос на сервер по параметрам поиска и добавляет карточку услуги по заданным параметрам на страницу

```
function getCardHtml(work) {  
  return `  
    <div class="col-md-4 mb-4">  
      <div class="card">  
          
        <div class="card-body">  
          <h5 class="card-title">${work.title}</h5>  
          <p class="card-text">${work.description}</p>  
        </div>  
      </div>  
    </div>  
  `;  
}
```

```

async function loadServices(searchTerm = '', category = '') {
  try {
    const response = await fetch('http://localhost:3000/services');
    const works = await response.json();
    const cardsContainer = document.querySelector('.row');
    cardsContainer.innerHTML = '';

    let filteredWorks = works;

    if (category) {
      filteredWorks = works.filter(work => work.category === category);
    }

    if (searchTerm) {
      const searchRegex = new RegExp(searchTerm, 'i');
      filteredWorks = filteredWorks.filter(work => searchRegex.test(work.title) || searchRegex.test(work.description));
    }

    if (filteredWorks.length === 0) {
      cardsContainer.innerHTML = '<p>Работы не найдены по выбранным критериям</p>';
      return;
    }

    filteredWorks.forEach((work) => {
      const cardHtml = getCardHtml(work);
      cardsContainer.innerHTML += cardHtml;
    });
  } catch (error) {
    console.error('Ошибка при загрузке работ:', error);
  }
}

document.addEventListener('DOMContentLoaded', () => {
  loadServices();
});

const searchForm = document.querySelector('.form-inline');
searchForm.addEventListener('submit', (event) => {
  event.preventDefault();

  const searchTerm = document.querySelector('input[type="search"]').value;
  const category = document.getElementById('filterCategory').value;

  loadServices(searchTerm, category);
});

```

Сортировка карточек по категориям и поиск:

Создан скрипт search.js, который формирует параметры поиска и вызывает loadServices с этими параметрами

```
// Функция для выполнения поиска и фильтрации
async function search(inputValue, checkbox) {
  const category = checkbox.findIndex((isChecked) => isChecked) + 1;
  await loadServices(inputValue, category);
}

// Функция для инициирования поиска и фильтрации
async function makeSearch() {
  const checks = [];
  for (let i = 1; i <= 3; i++) {
    checks.push(document.getElementById(`checkbox${i}`).checked);
  }

  const inputValue = document.getElementById('searchbox').value;
  await search(inputValue, checks);
}
```


Пример фильтрации по категориям:

Портфолио Ильи Казанкова


Профиль
Сменить тему
Главная
Вход

СбросUnityUESQL


Поиск




Игра "Лабиринт"
Group: Unity
Требуется авторизация
для добавления в
избранное



Аркадная игра
"Пространственный
защитник"
Group: UE
Требуется авторизация
для добавления в
избранное



Блоговая платформа
Group: SQL
Требуется авторизация
для добавления в
избранное



Виртуальная экскурсия
по музею

СбросUnityUESQL

Поиск

Пример поиска

игра



Игра "Лабиринт"

Group: Unity

Требуется авторизация
для добавления в
избранное



UNREAL
ENGINE

Аркадная игра
"Пространственный
защитник"

Group: UE

Требуется авторизация
для добавления в
избранное

Через json-server реализованы логин и регистрация:

```
async function login() {  
  let reload = true;  
  const form = document.getElementById('loginForm');  
  const inputs = Array.from(form.querySelectorAll('input'));  
  const loginData = {};  
  
  for (const input of inputs) {  
    loginData[input.name] = input.value;  
  }  
  
  console.log('login data', loginData);  
  
  const response = await fetch('http://localhost:3000/login', {  
    method: "POST",  
    body: JSON.stringify(loginData),  
    headers: {  
      'Content-Type': 'application/json'  
    }  
  });  
  
  console.log(response);  
  
  if (reload) {  
    const responseJson = await response.json();  
    const { accessToken, user } = responseJson;  
  
    console.log('response', responseJson);  
  
    localStorage.accessToken = accessToken;  
    localStorage.user = JSON.stringify(user);  
    console.log("reloading");  
    location.reload();  
  }  
}
```

```
async function register(event) {  
  event.preventDefault()  
  
  const inputs = Array.from(event.target.querySelectorAll('input'))  
  const loginData = {}  
  
  for (const input of inputs) {  
    loginData[input.name] = input.value  
  }  
  
  console.log('login data', loginData)  
  
  const response = await fetch('http://localhost:3000/users', {  
    method: "POST",  
    body: JSON.stringify(loginData),  
    headers: {  
      'Content-Type': 'application/json'  
    }  
  })  
  
  window.location.href = "http://127.0.0.1:5500/services.html"  
}
```

Контейнер с карточками:


```
✓ "services": [  
✓ {  
  "id": 0,  
  "title": "Работа 1",  
  "description": "Описание работы 1",  
  "image": "work1.jpg",  
  "category": "Unity"  
},  
✓ {  
  "id": 1,  
  "title": "Работа 2",  
  "description": "Описание работы 2",  
  "image": "work2.jpg",  
  "category": "UE"  
},  
✓ {  
  "id": 2,  
  "title": "Работа 3",  
  "description": "Описание работы 3",  
  "image": "work3.jpg",  
  "category": "SQL"  
}  
]  
}
```

Вывод

В данной лабораторной работе я познакомился json-server, на практике изучил работу с API (его подключение к сайту) и fetch. Реализовал поиск и фильтрацию, а также авторизацию на своем сайте