САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Саунин Антон

Группа

K33402

Проверил:
Добряков Д. И.

Санкт-Петербург

2023 г.

Задача

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

Вариант: магазин одежды

Ход работы

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

Для начала необходимо было реализовать внешнее API при помощи json-server, а также наполнить его данными

db.son:


```
{
  "users": [
    {
      "email": "anton@gmail.com",
      "password": "$2a$10$j9RBokA4cZFRKQwewAJXvO7prQ3kRveqLXSRT.POjlgaqmzsixDHa",
      "id": "snusp@gmail.com",
      "phonenumber": "+79610258944"
    }
  ],
  "products": [
    {
      "name": "Baggy pants",
      "price": "199$",
      "imgSrc": "img/catalog/boys1.jpeg",
      "type": "1",
      "id": 1,
      "sex": "boys"
    },
    {
      "name": "Classic Boston pants",
      "price": "299$",
      "imgSrc": "img/catalog/boys2.png",
      "type": "1",
      "id": 2,
      "sex": "boys"
    },
    {
      "name": "Classic Gooliver pants",
      "price": "149$",
      "imgSrc": "img/catalog/boys3.jpeg",
      "type": "1",
      "id": 3,
      "sex": "boys"
    },
    {
      "name": "Classic President pants",
      "price": "499$",
      "imgSrc": "img/catalog/boys4.jpeg",
      "type": "1",
      "id": 4,
      "sex": "boys"
    },
    {
      "name": "Classic Insbruc costume",
      "price": "799$",
      "imgSrc": "img/catalog/boys5.jpeg",
      "type": "2",
```

Далее был создан скрипт products.js, в котором написаны две функции:

- getProduct, которая возвращает html-код карточки товара,

- loadProducts, которая делает запрос на сервер по параметрам поиска и добавляет карточку услуги по заданным параметрам на страницу

```
function getProduct({ name, price, imgSrc }) {
  return `

    <div class="d-flex flex-column align-items-center cat-item">
        <button onclick="OpenModal('${name}','${imgSrc}')" style="border: none;">
        <img src="${imgSrc}" class="cat-img">
        </button>
        <h4>${name}</h4>
        <div class="">
            ${price}
        </div>

    </div>
    `;
}
async function loadProducts(searchString = "", sex = "") {
  document.getElementById("productsContainer").innerHTML = "";
  let url = "http://localhost:3000/products?";

  if (sex == "boys") {
    url += `&sex=boys`;
  }
  if (sex == "girls") {
    url += `&sex=girls`;
  }
  if (sex == "acs") {
    url += `&sex=acs`;
  }
  url += `${searchString}`;
  const response = await fetch(url);

  const responseJson = await response.json();

  for (const product of responseJson) {
    document.getElementById("productsContainer").innerHTML +=
      getProduct(product);
    document.getElementsByClassName;
  }
}
```

Использование на разных страницах различно, потому что в разных случаях изначально нужно загружать разные типы товаров, поэтому было решено добавить параметр sex вызывать функцию загрузки товаров относительно этого параметра.

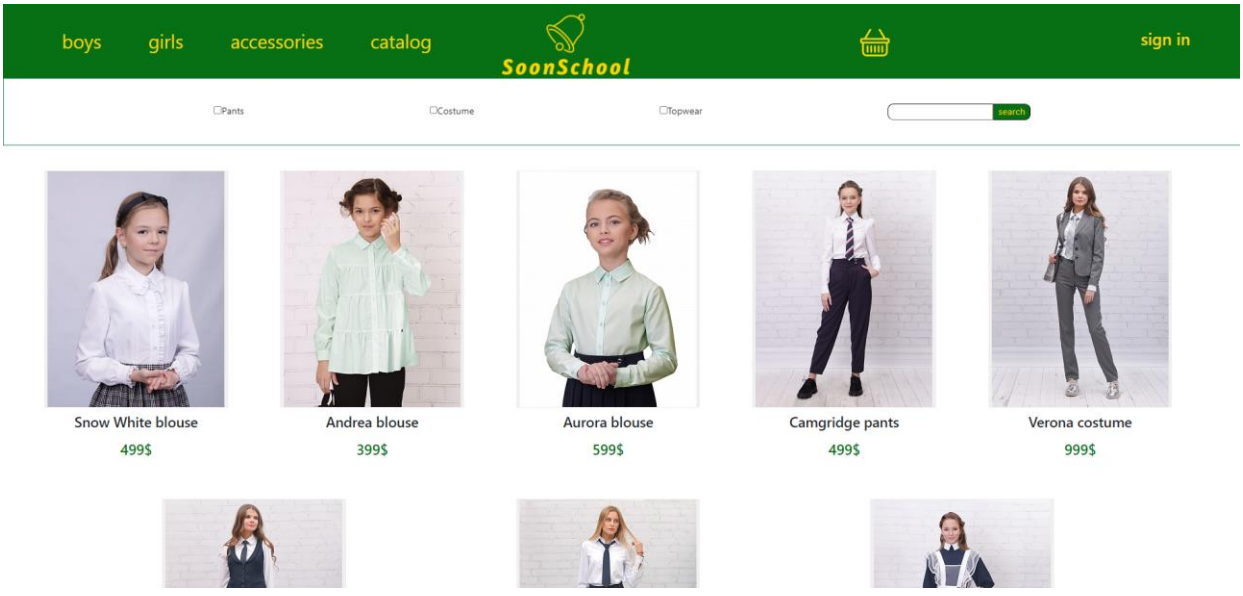
Пример вызова на странице с мужскими товарами:

```
<script>
    document.addEventListener('DOMContentLoaded', () => {
        loadProducts("", "boys")
    })
</script>
```

Пример вызова на странице с женскими товарами:

```
<script>
    document.addEventListener('DOMContentLoaded', () => {
        loadProducts("", "girls")
    })
</script>
```

Результат(в случае с женскими товарами):

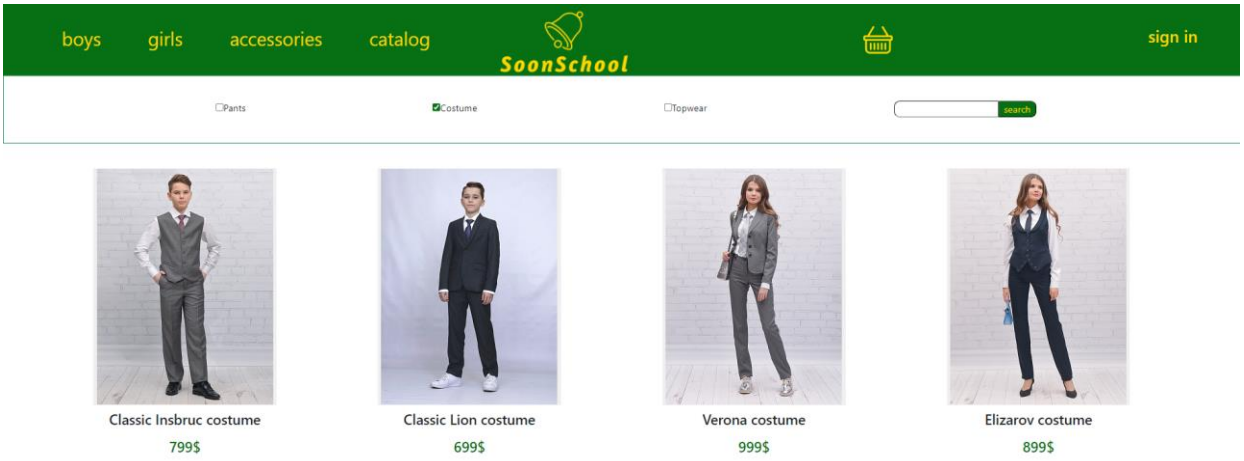


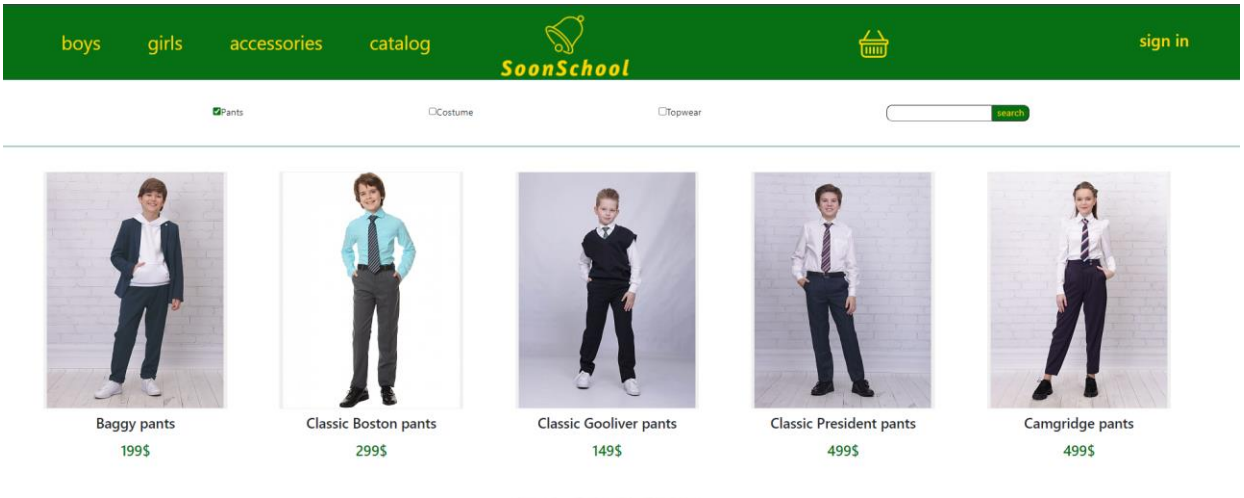
Сортировка карточек по категориям и поиск:

Создан скрипт filters.js, который формирует параметры поиска и вызывает loadProducts с этими параметрами

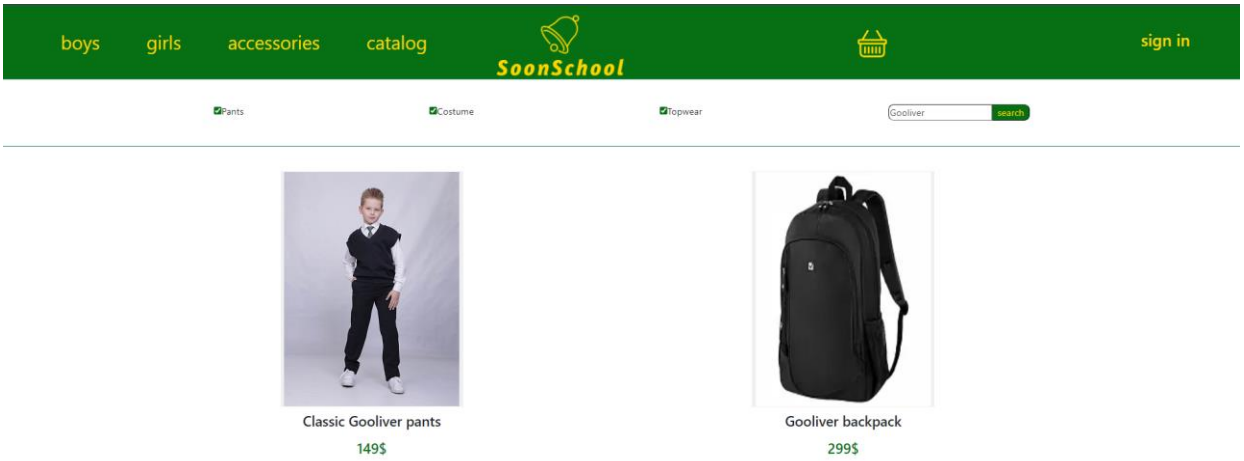
```
JS filters.js > ...
1   async function search(inputValue, checkbox, sex) {
2     for (i = 0; i < checkbox.length; i++) {
3       if (checkbox[i]) {
4         loadProducts(`&type=${i + 1}&q=${inputValue}`, sex);
5       }
6     }
7   }
8   async function Search(sex = "") {
9     checks = [];
0     for (i = 1; i < 4; i++)
1       checks.push(document.getElementById(`checkbox${i}`).checked);
2     search(document.getElementById("searchbox").value, checks, sex);
3   }
4
```

Примеры фильтрации по категориям:





Пример поиска:



Также json-server реализованы логин и регистрация:

Регистрация:

```
async function registration(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",
    },
  });
```

Логин:

```
async function login(event) {
  event.preventDefault();
  let reload = true;
  const inputs = Array.from(event.target.querySelectorAll("input"));
  const loginData = {};

  for (const input of inputs) {
    loginData[input.name] = input.value;
  }

  const response = await fetch("http://localhost:3000/login", {
    method: "POST",
    body: JSON.stringify(loginData),
    headers: {
      "Content-Type": "application/json",
    },
  });
  const responseJson = await response.json();

  const { accessToken, user } = responseJson;

  console.log("response", responseJson);
  if (responseJson == "Cannot find user") {
    alert("User not found");
    reload = false;
  } else {
    if (responseJson == "Incorrect password") {
      alert("password is incorrect");
      reload = false;
    }
  }
  if (reload) {
    reload = true;
    localStorage.accessToken = accessToken;
    localStorage.user = JSON.stringify(user);
    document.getElementById("header").innerHTML = addProfile();
    modalLogin.hide();
  }
}
```

Вывод

В данной лабораторной работе я познакомился json-server, на практике реализовал внешнее API при помощи json server и взаимодействие с ним при помощи fetch. Реализовал поиск и фильтрацию, а также авторизацию на своем сайте