**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

# **Дисциплина:** Фронт-энд разработка

Отчет

### Лабораторная работа 2: взаимодействие с внешним API

Выполнила: Пронина Александра

Группа К33392

Проверил:

Добряков Д. И.

Санкт-Петербург

2023 г.

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

Что было добавлено?

1. Имитация закрытого API для регистрации:

Скрипт: <script>  
 *// Функция для имитации отправки запроса на сервер* function *sendRequest*(url, method, data) {  
 return new *Promise*((resolve, reject) => {  
 *setTimeout*(() => {  
 *// Имитация успешного ответа сервера* resolve({ status: 200, message: 'Success' });  
  
 *// Имитация ошибки сервера  
 // reject({ status: 500, message: 'Internal Server Error' });* }, 1000);  
 });  
 }  
  
 *// Функция для добавления пользователя в файл users.json* function *addUser*(username, password) {  
 *// Отправка POST запроса на сервер для добавления пользователя  
 sendRequest*('http://localhost:3000/users', 'POST', { username, password })  
 .then(response => {  
 if (response.status === 200) {  
 console.log('Пользователь успешно добавлен');  
 *// Дополнительные действия после успешного добавления пользователя* } else {  
 console.log('Не удалось добавить пользователя');  
 *// Дополнительные действия при ошибке добавления пользователя* }  
 })  
 .catch(error => {  
 console.log('Произошла ошибка при отправке запроса');  
 *// Дополнительные действия при ошибке отправки запроса* });  
 }  
  
 *// Обработчик события отправки формы* function *handleSubmit*(event) {  
 event.preventDefault();  
 const username = document.getElementById('email').value;  
 const password = document.getElementById('password').value;  
 const confirmPassword = document.getElementById('confirm\_password').value;  
 if (password !== confirmPassword) {  
 console.log('Пароли не совпадают');  
 return;  
 }  
 *addUser*(username, password);  
 }  
  
 *// Добавление обработчика события на форму* document.getElementById('registrationForm').addEventListener('submit', *handleSubmit*);  
</script>  
</body>  
</html>

1. Добавление API для работы с картами в buy и way:  
    <div id="map"><script type="text/javascript" charset="utf-8" async src="https://api-maps.yandex.ru/services/constructor/1.0/js/?um=constructor%3Abbcefd6f30234b7cc77a40445c5c4c129e767cb8c046aaa6b4933fb9346391d0&amp;width=500&amp;height=400&amp;lang=ru\_RU&amp;scroll=true"></script>

<script>  
 function *initMap*() {  
 var map = new google.maps.Map(document.getElementById('map'), {  
 center: { lat: 55.7558, lng: 37.6176 },  
 zoom: 12  
 });  
  
 var locations = [  
 {lat: 55.7558, lng: 37.6176},  
 {lat: 55.7522, lng: 37.6156},  
 {lat: 55.7517, lng: 37.6109},  
 {lat: 55.7495, lng: 37.6051},  
 {lat: 55.7539, lng: 37.6226}  
 ];  
  
 locations.forEach(function (location) {  
 new google.maps.Marker({  
 position: location,  
 map: map  
 });  
 });  
  
 var path = new google.maps.Polyline({  
 path: locations,  
 geodesic: true,  
 strokeColor: '#ff0000',  
 strokeOpacity: 1.0,  
 strokeWeight: 2  
 });  
  
 path.setMap(map);  
 }  
</script>

1. Axios используется в скрипте входа в аккаунт:
2. <<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>  
   <script>  
    const *checkAuth* = () => {  
    if (sessionStorage.accessToken) {  
    window.location.pathname = "http://localhost:63342/labs/K33392/%D0%9F%D1%80%D0%BE%D0%BD%D0%B8%D0%BD%D0%B0%20%D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%B0/lab\_2/travel\_site/html/account.html?\_ijt=uc67dmfq1dhijcsjurlaaj5ich";  
    }  
    };  
     
    const *login* = async (event) => {  
    event.preventDefault();  
    const inputs = Array.from(event.target.querySelectorAll('input'));  
    const loginData = Object.fromEntries(inputs.map(input =>  
    [input.name, input.value]));  
     
    try {  
    const response = await axios.get('./users.json');  
    const users = response.data.users;  
    const authenticatedUser = users.find(user => user.email === loginData.email && user.password === loginData.password);  
    if (authenticatedUser) {  
    sessionStorage.accessToken = authenticatedUser.email;  
    window.location.href = "http://localhost:63342/labs/K33392/%D0%9F%D1%80%D0%BE%D0%BD%D0%B8%D0%BD%D0%B0%20%D0%90%D0%BB%D0%B5%D0%BA%D1%81%D0%B0%D0%BD%D0%B4%D1%80%D0%B0/lab\_2/travel\_site/html/account.html?\_ijt=uc67dmfq1dhijcsjurlaaj5ich";  
    } else {  
    *alert*("Неверный email или пароль");  
    }  
    } catch (error) {  
    console.error(error);  
    }  
    };  
     
    *checkAuth*();  
   </script>

И в finding используется json для хранения данных о поездках: <script>  
 $.getJSON("travel.json", function(data) {  
 const travels = data;  
  
 *// Обработчик события изменения фильтров* function handleFilterChange() {  
 const upcomingFilter = document.getElementById("upcomingFilter").checked;  
 const ongoingFilter = document.getElementById("ongoingFilter").checked;  
 const completedFilter = document.getElementById("completedFilter").checked;  
 const postponedFilter = document.getElementById("postponedFilter").checked;  
 const cancelledFilter = document.getElementById("cancelledFilter").checked;  
  
 const filteredTravels = travels.filter(travel => {  
 return (upcomingFilter && travel.status === "upcoming") ||  
 (ongoingFilter && travel.status === "ongoing") ||  
 (completedFilter && travel.status === "completed") ||  
 (postponedFilter && travel.status === "postponed") ||  
 (cancelledFilter && travel.status === "cancelled");  
});  
  
 renderTravelList(filteredTravels);  
}  
  
 function renderTravelList(travels) {  
 const travelListElement = document.getElementById("travelList");  
 travelListElement.innerHTML = "";  
  
 travels.forEach(travel => {  
 const travelItem = document.createElement("div");  
 travelItem.classList.add("travel-item");  
  
 switch (travel.status) {  
 case "upcoming":  
 travelItem.classList.add("upcoming");  
 break;  
 case "ongoing":  
 travelItem.classList.add("ongoing");  
 break;  
 case "completed":  
 travelItem.classList.add("completed");  
 break;  
 case "postponed":  
 travelItem.classList.add("postponed");  
 break;  
 case "cancelled":  
 travelItem.classList.add("cancelled");  
break;  
}  
  
const travelName = document.createElement("h5");  
travelName.textContent = travel.destination;  
  
const travelStatus = document.createElement("p");  
travelStatus.textContent = travel.status;  
  
const travelDate = document.createElement("p");  
travelDate.textContent = travel.date;  
  
const travelImg = document.createElement("img");  
travelImg.src = travel.image;  
  
travelItem.appendChild(travelName);  
travelItem.appendChild(travelStatus);  
travelItem.appendChild(travelDate);  
travelItem.appendChild(travelImg);  
  
travelListElement.appendChild(travelItem);  
});  
}  
  
  
  
document.getElementById("upcomingFilter").addEventListener("change", handleFilterChange);  
document.getElementById("ongoingFilter").addEventListener("change", handleFilterChange);  
document.getElementById("completedFilter").addEventListener("change", handleFilterChange);  
document.getElementById("postponedFilter").addEventListener("change", handleFilterChange);  
document.getElementById("cancelledFilter").addEventListener("change", handleFilterChange);  
  
*// Инициализация отображения путешествий при загрузке страницы*renderTravelList(handleFilterChange);  
  
});  
  
  
</script>

Вывод: В данной лабораторной работе мы научились взаимодействовать с внешним API средствами fetch/axios/xhr.