

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №2

Выполнил:

Захарчук Александр

Группа К3441

Проверил:

Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

Варианты остаются прежними. Теперь Вам нужно привязать то, что Вы делали в ЛР1 к внешнему API средствами fetch/axios/xhr. Реализуйте моковое API средствами JSON-сервера и подключите к нему авторизацию, как в примерах, которые мы рассматривали в рамках тем "Имитация работы с API".

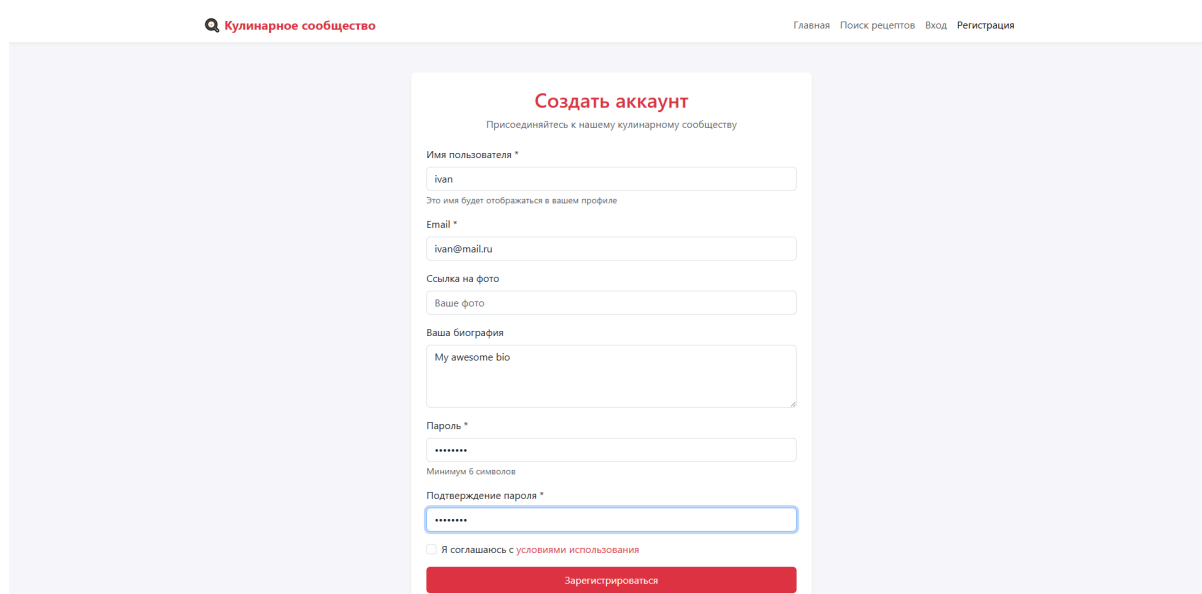
Например, для приложения для просмотра прогнозов погоды задание выглядит следующим образом:

Реализовать получение погоды (прогноз на ближайшие 7 дней) из открытого API OpenWeatherMap, в зависимости от геолокации пользователя. Реализовать вывод полученного прогноза в виде 7 карточек в три ряда (первый ряд - крупная карточка, второй ряд - три карточки в меньшем размере, третий ряд - четыре карточки в маленьком размере).

Ход работы

В качестве API для данной работы использовался сервис рецептов, разработанный в прошлом семестре в рамках бэкенд разработки.

На странице регистрации была добавлена отправка данных из формы на бэкенд по нажатию кнопки. После успешной регистрации происходит редирект на страницу логина.



The screenshot shows a web browser window with the title "Кулинарное сообщество" and navigation links: Главная, Поиск рецептов, Вход, and Регистрация. The main content is a registration form titled "Создать аккаунт" with the subtitle "Присоединяйтесь к нашему кулинарному сообществу". The form contains the following fields and elements:

- Имя пользователя ***: Input field with "ivan" entered. Below it, a note: "Это имя будет отображаться в вашем профиле".
- Email ***: Input field with "ivan@mail.ru" entered.
- Ссылка на фото**: Input field with "Ваше фото" entered.
- Ваша биография**: Text area with "My awesome bio" entered.
- Пароль ***: Password input field with masked characters. Below it, a note: "Минимум 5 символов".
- Подтверждение пароля ***: Password input field with masked characters.
- ☐ **Я соглашаюсь с условиями использования**
- Зарегистрироваться**: Red button at the bottom of the form.

Рисунок 1 - Страница регистрации

Код, обрабатывающий регистрацию, показан на рисунке 2.

```
const registerForm = document.querySelector("#registerForm");

registerForm.addEventListener("submit", async (e) => {
  e.preventDefault();

  const username = registerForm.elements["username"].value;
  const email = registerForm.elements["email"].value;
  const profile_picture = registerForm.elements["profilePicture"].value;
  const bio = registerForm.elements["bio"].value;
  const password = registerForm.elements["password"].value;

  const response = await fetch("http://localhost/api/users/register", {
    method: "POST",
    body: JSON.stringify({ username, email, password, bio, profile_picture }),
    headers: {
      accept: "application/json",
      "Content-Type": "application/json",
    },
  });

  if (response.status === 201) {
    window.location.href = "login.html";
  }
});
```

Рисунок 2 - Код регистрации

На странице логина была добавлена обработка формы. После ввода логина и пароля отправляется запрос на бэкенд. В случае успешного входа полученный токен сохраняется в localStorage.

Страница логина показана на рисунке 3.

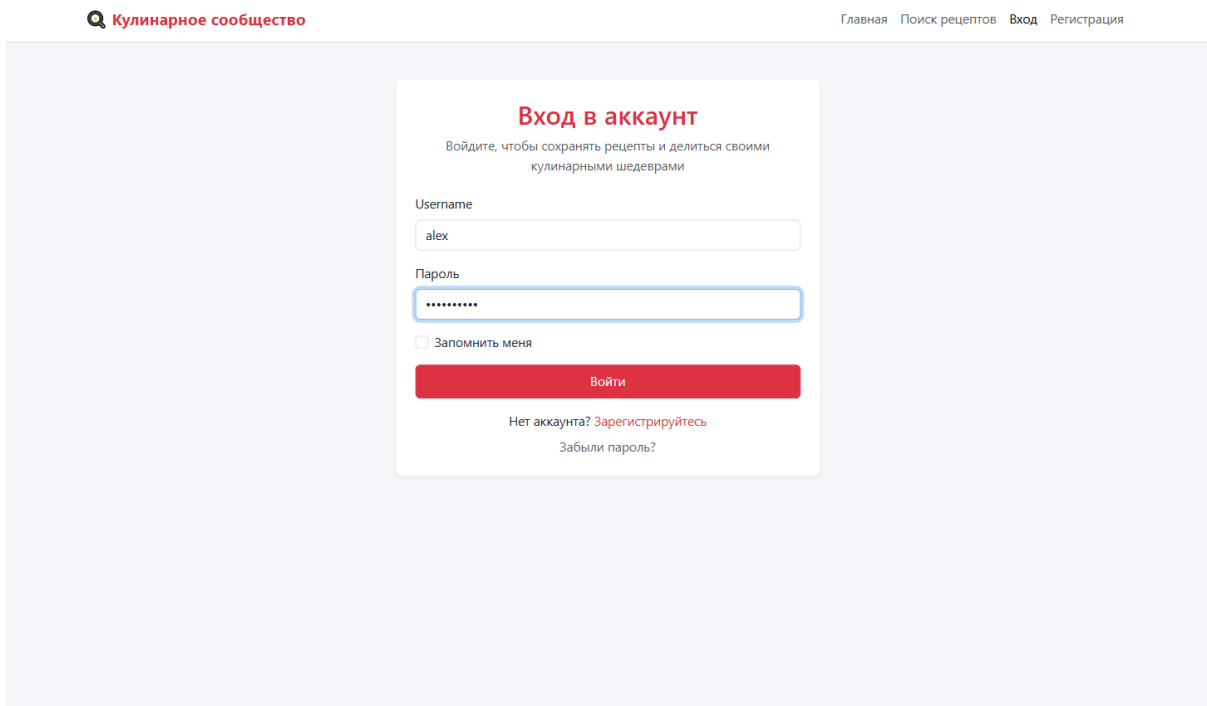


Рисунок 3 - Страница логина

Токен доступа, сохраненный в localStorage, показан на рисунке 4.

Application		
	↻	Filter
Manifest	Key	Value
Service...	access_token	eyJhbGciOiJIUzI1Ni...
Storage		

Рисунок 4 - Токен в localStorage

Код, обрабатывающий форму логина показан на рисунке 5.

```

loginForm.addEventListener("submit", async (e) => {
  e.preventDefault();

  const username = loginForm.elements["username"].value;
  const password = loginForm.elements["password"].value;

  const response = await fetch("http://localhost/api/users/login", {
    method: "POST",
    body: JSON.stringify({ username, password }),
    headers: {
      accept: "application/json",
      "Content-Type": "application/json",
    },
  });

  const json = await response.json();
  const token = json["token"];

  if (token !== undefined) {
    localStorage.setItem("access_token", token);
    window.location.href = "index.html";
  }
});

```

Рисунок 5 - Код логина

На странице поиска рецептов была реализована загрузка информации с сервера. При открытии страницы на бэкенд отправляется запрос, данные из которого отрисовываются на странице. Страница поиска рецептов показана на рисунке 6.

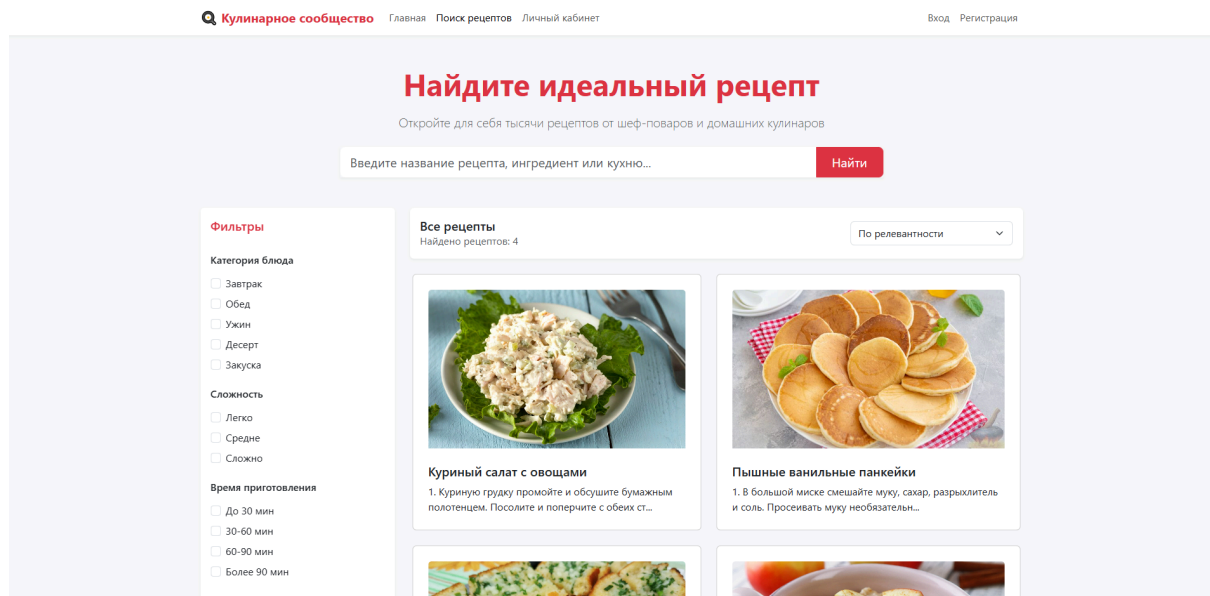


Рисунок 6 - Страница поиска рецептов

Код, который загружает рецепты с сервера и отрисовывает их, показан на рисунке 7.

```
fetch("http://localhost/api/recipes", {
  method: "GET",
  headers: {
    accept: "application/json",
    "Content-Type": "application/json",
  },
}).then(r => r.json()).then(json => {
  resultsCount.innerHTML = `Найдено рецептов: ${json.length}`;
  recipesContainer.innerHTML = "";
  json.forEach((recipe, index) => {
    recipesContainer.innerHTML += `
    <a class="nav-link col-6 mb-4" href="recipe.html?id=${recipe.id}">
      <div class="card p-4 ${index % 2 === 0 ? 'me-4' : ''}>
        
          <h5 class="card-title">${recipe.title}</h5>
          <p class="card-text">${recipe.description.slice(0, 90)}...</p>
        </div>
      </div>
    </a>
    `;
  });
});
```

Рисунок 7 - Код поиска рецептов

По нажатию на какой-либо рецепт происходит переход на страницу с информацией об этом рецепте. Информация о нем также запрашивается с сервера. Идентификатор рецепта извлекается из адреса страницы.

Страница рецепта показана на рисунках 8 и 9.

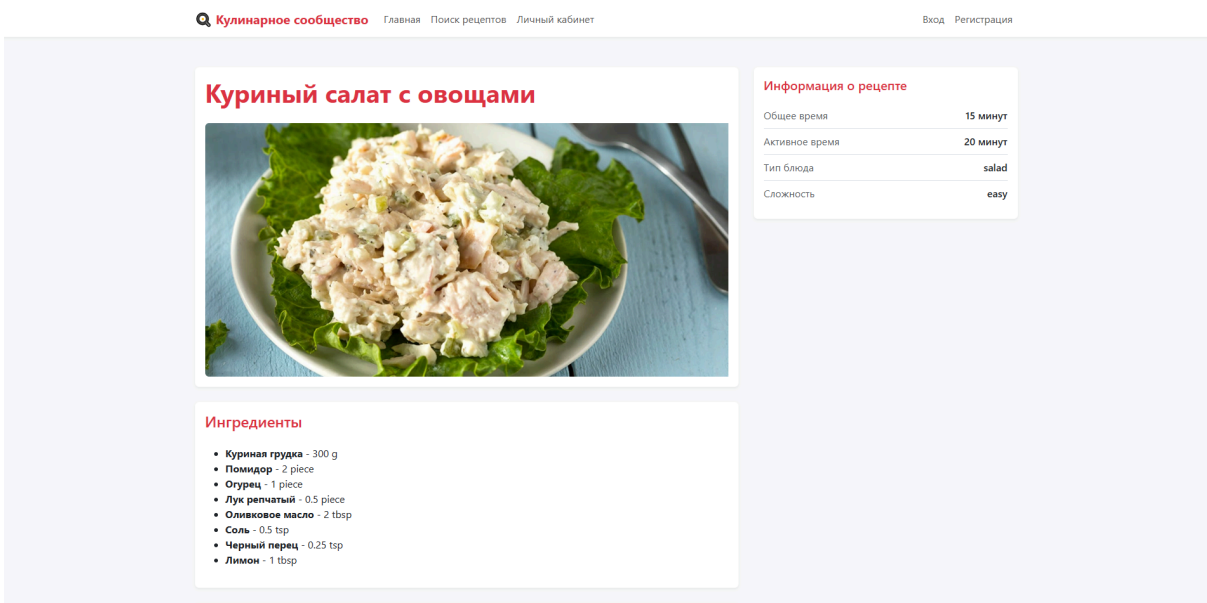


Рисунок 8 - Страница рецепта

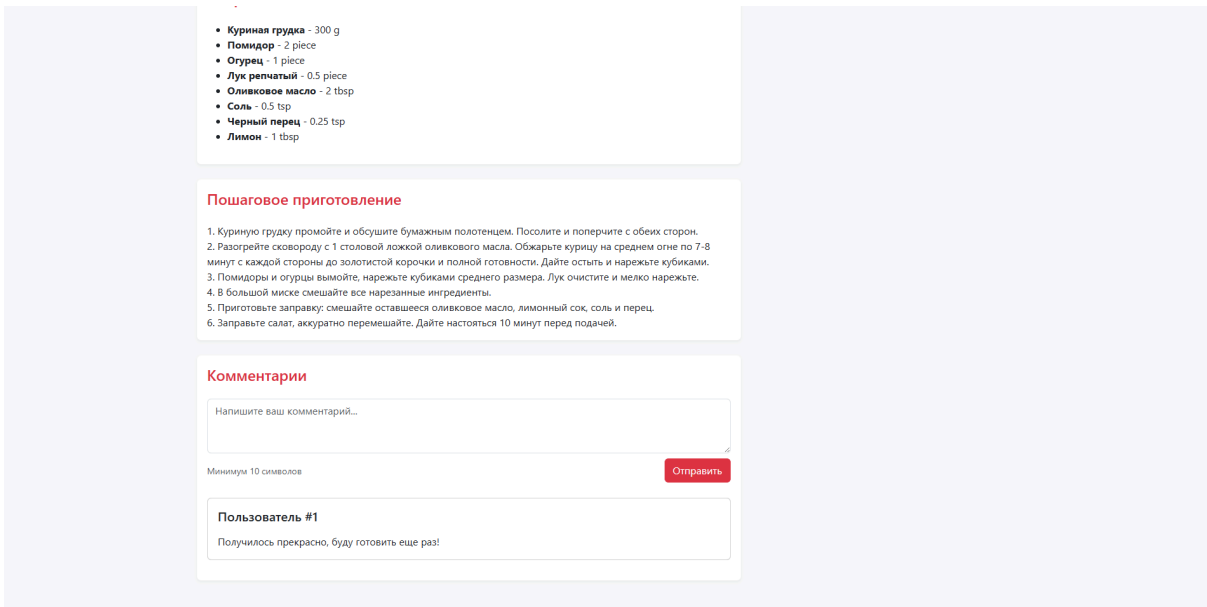


Рисунок 9 - Страница рецепта

В нижней части страницы отображаются комментарии к рецепту, которые также загружаются с сервера. Есть возможность добавить собственный

комментарий, который сразу же отобразится на странице и сохранится в базе данных на бэкенде.

Код страницы рецепта показан на рисунке 10.

```
const recipeTitle = document.querySelector("#recipeTitle");
const mainPicture = document.querySelector("#mainPicture");
const ingredientsList = document.querySelector("#ingredientsList");
const recipeStepsContainer = document.querySelector("#recipeSteps");
const totalTime = document.querySelector("#totalTime");
const activeTime = document.querySelector("#activeTime");
const dishType = document.querySelector("#dishType");
const difficulty = document.querySelector("#difficulty");
const commentText = document.querySelector("#commentText");
const commentsList = document.querySelector("#commentsList");
const submitCommentButton = document.querySelector("#submitCommentBtn");

const urlParams = new URLSearchParams(window.location.search);
const recipeId = parseInt(urlParams.get("id") || "1");

fetch(`http://localhost/api/recipes/${recipeId}`, {
  method: "GET",
  headers: {
    accept: "application/json",
    "Content-Type": "application/json",
  },
}).then((r) => {
  r.json().then((json) => {
    recipeTitle.innerHTML = json.title;
    mainPicture.src = json.picture;

    ingredientsList.innerHTML = "";
    json.ingredients.forEach((ingredient) => {
      ingredientsList.innerHTML += `<li><strong>${ingredient.ingredient.name}</strong> - ${ingredient.quantity} ${ingredient.unit}</li>`;
    });

    recipeStepsContainer.innerHTML = json.description;
    totalTime.innerHTML = `${json.cooking_time} минут`;
    activeTime.innerHTML = `${json.preparation_time} минут`;
    dishType.innerHTML = json.dish_type;
    difficulty.innerHTML = json.difficulty_level;
  })
});
```

Рисунок 10 - Код страницы рецепта

Авторизованный пользователь может перейти на страницу своего профиля, на которой будет отображаться информация о нем. Если пользователь не авторизован, то он будет перенаправлен на страницу логина. Страница профиля показана на рисунке 11.

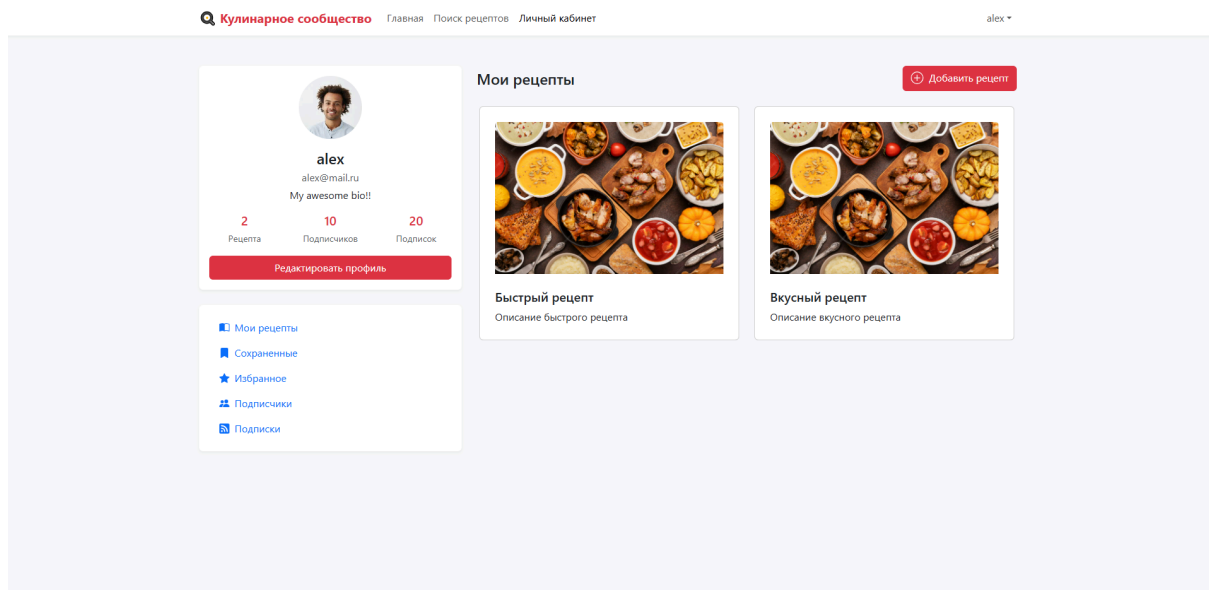


Рисунок 11 - Страница пользователя

Код страницы пользователя показан на рисунке 12.

```
const usernameField = document.querySelector("#userName");
const emailField = document.querySelector("#userEmail");
const userImageField = document.querySelector("#userImage");
const userBioField = document.querySelector("#userBio");

fetch("http://localhost/api/users/me", {
  method: "GET",
  headers: {
    accept: "application/json",
    "Content-Type": "application/json",
    authorization: `Bearer ${access_token}`,
  },
}).then(r => r.json()).then(json => {
  usernameField.innerHTML = json["username"];
  emailField.innerHTML = json["email"];
  userImageField.src = json["profile_picture"];
  userBioField.innerHTML = json["bio"];
})))
```

Рисунок 12 - Код страницы пользователя

Вывод

В ходе выполнения данной лабораторной работы был изучен и реализован механизм передачи информации от фронтенда к бэкенду. Были реализованы страницы входа, регистрации, поиска рецептов, профиля пользователя, а также рецепта.