

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Фронт-энд разработка

Отчет

Лабораторная работа №3

Выполнил:

Захарчук Александр

Группа К3441

Проверил:

Добряков Д. И.

Санкт-Петербург

2026 г.

## Задача

Мигрировать ранее разработанное приложение (в рамках ЛР1 и ЛР2) на фреймворк Vue.JS.

Каким требованиям ваше приложение должно соответствовать:

- Должен быть подключён роутер
- Должна быть реализована работа с внешним API (желательно посредством axios)
- Разумное деление на компоненты (продемонстрируйте понимание компонентного подхода)
- Использование composables для выделения повторяющегося функционала в отдельные файлы

## 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.

Рисунок 1 - Страница регистрации

Код, обрабатывающий регистрацию, показан на рисунке 2.

```

<script setup>
import FormContainer from "@/components/FormContainer.vue";
import Navbar from "@/components/Navbar.vue";
import FormField from "@/components/FormField.vue";
import { ref } from "vue";
import { useUserStore } from "@/stores/user";

const form = ref({
  username: "",
  email: "",
  imageLink: "",
  bio: "",
  password: "",
  confirmPassword: "",
  termsAccepted: false,
});

const userStore = useUserStore();

const handleRegister = async () => {
  if (form.value.password !== form.value.confirmPassword) {
    return;
  }
  try {
    await userStore.register(form.value.username, form.value.email, form.value.password, form.value.in
  } catch (error) {
    console.error("Login failed:", error);
  }
};
</script>

```

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.

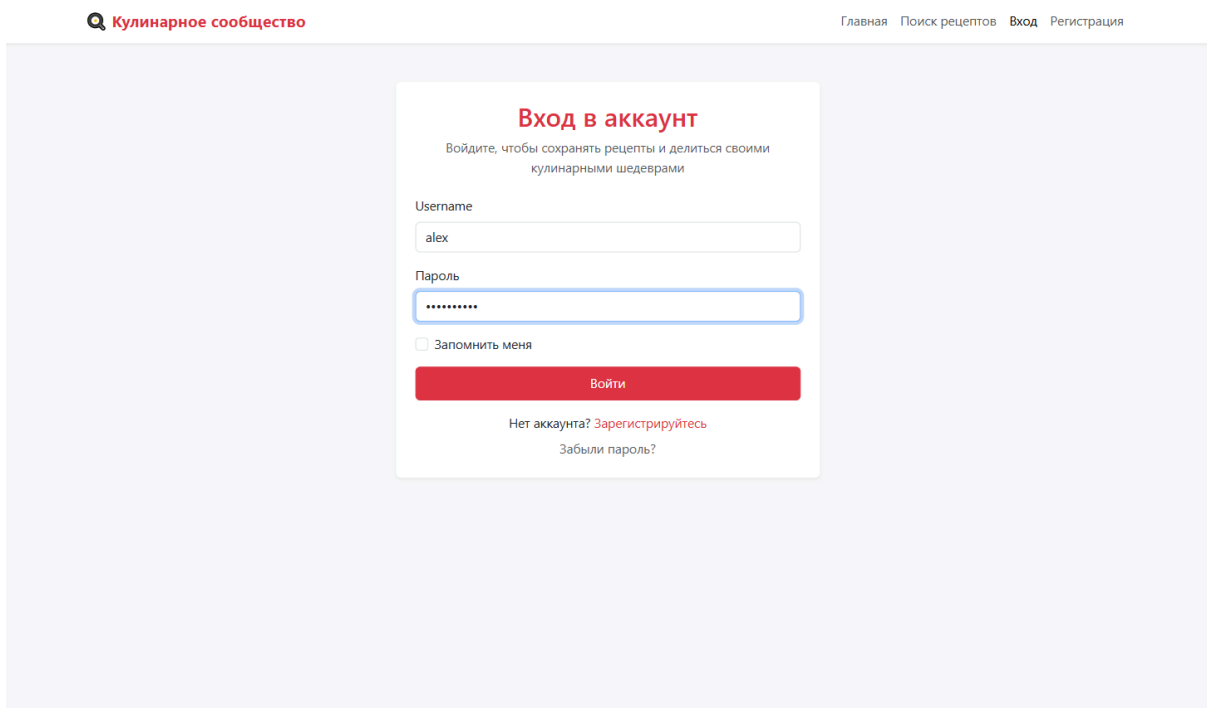


Рисунок 3 - Страница логина

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.

```

<script setup>
import FormContainer from "@/components/FormContainer.vue";
import FormField from "@/components/FormField.vue";
import Navbar from "@/components/Navbar.vue";
import { useUserStore } from "@/stores/user";
import { ref } from "vue";

const form = ref({ username: "", password: "" });

const userStore = useUserStore();

const handleLogin = async () => {
  try {
    await userStore.login(form.value.username, form.value.password);
  } catch (error) {
    console.error("Login failed:", error);
  }
};
</script>

```

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.

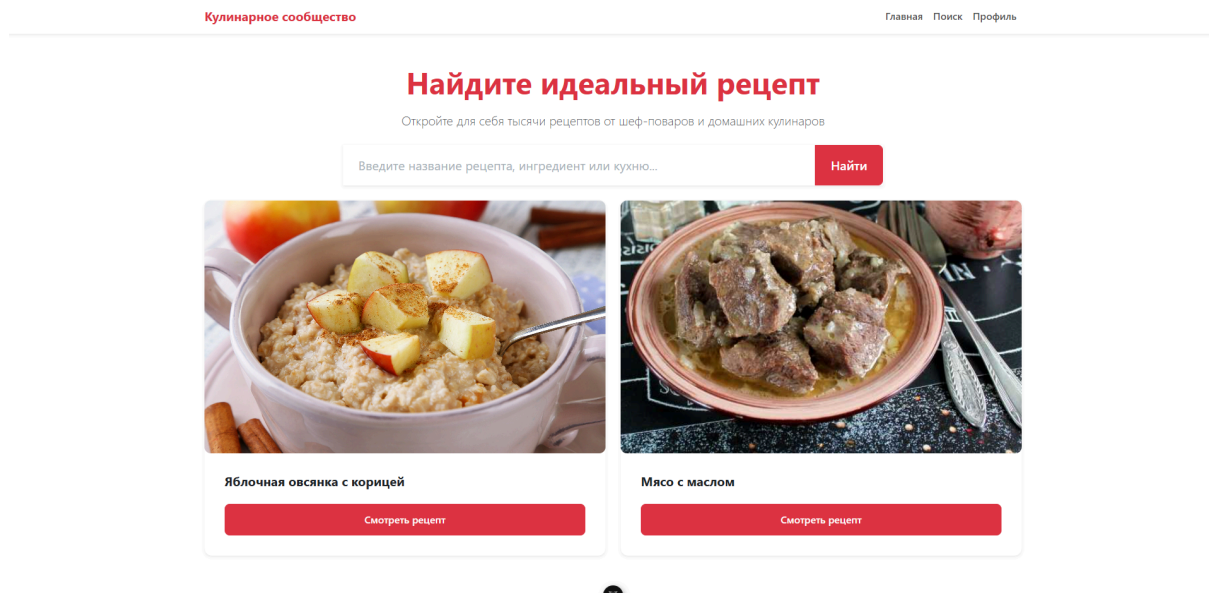


Рисунок 6 - Страница поиска рецептов

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

```
<script setup>
import Navbar from "@/components/Navbar.vue";
import RecipeCard from "@/components/RecipeCard.vue";
import { useRecipeStore } from "@/stores/recipe";
import { ref } from "vue";

const searchText = ref();
const recipeStore = useRecipeStore();

const handleSearch = async () => {
  const filter = searchText.value.trim();

  if (filter) {
    await recipeStore.load(filter);
  }
};
</script>
```

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.

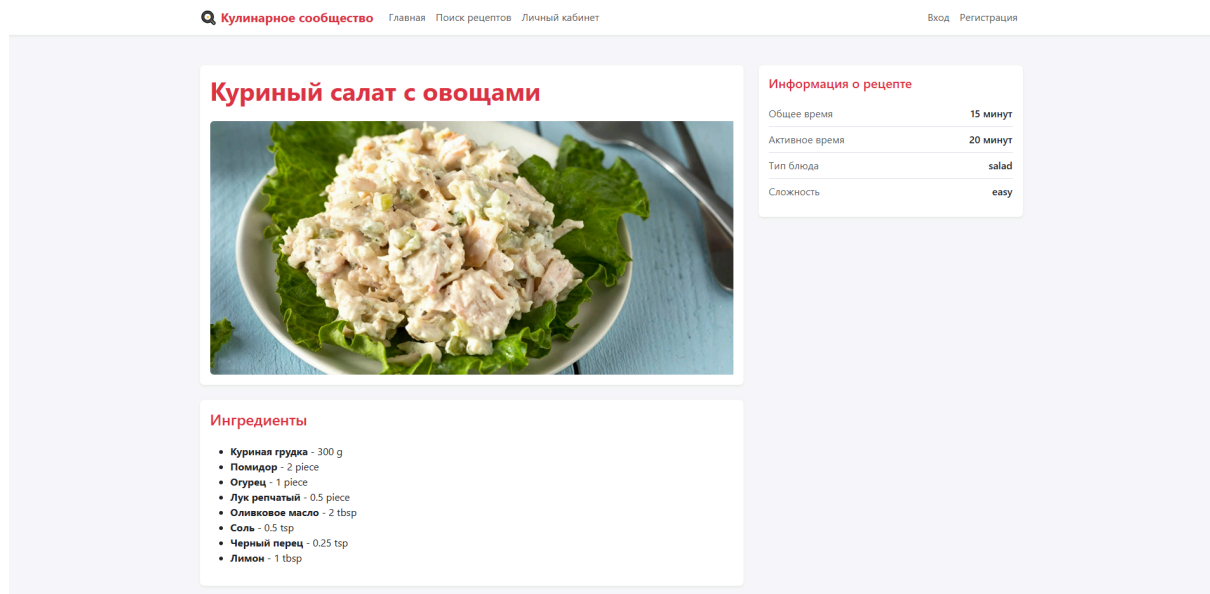


Рисунок 8 - Страница рецепта

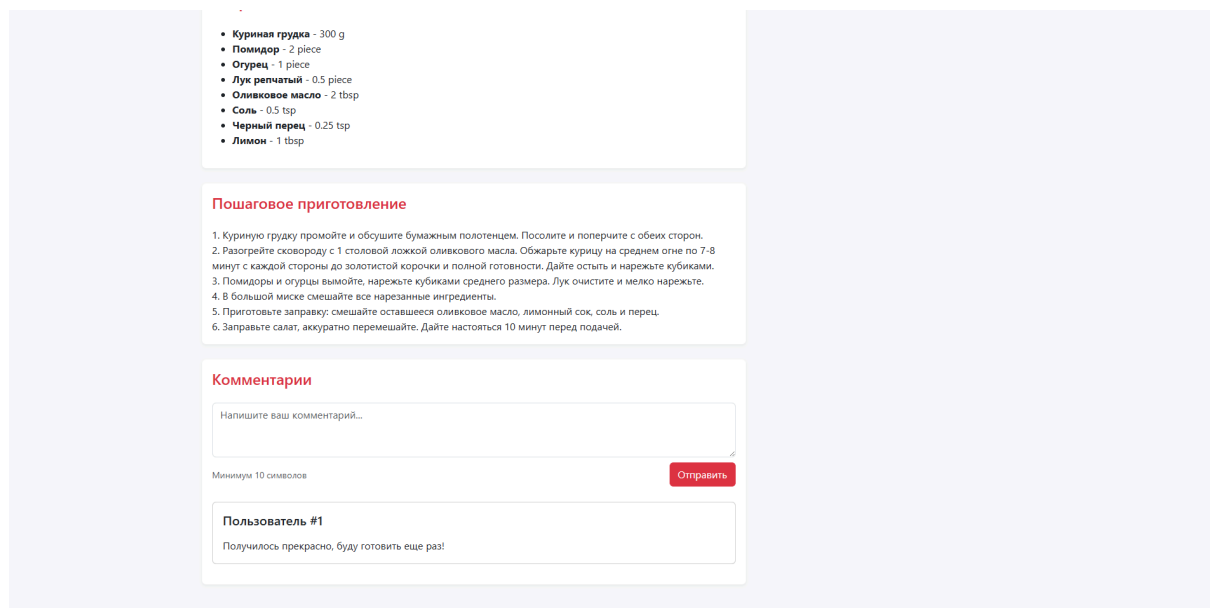


Рисунок 9 - Страница рецепта

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.

```

onMounted(async () => {
  try {
    await recipeStore.loadById(recipeId);
    await commentStore.getByRecipeId(recipeId);
    await userStore.getAll();
  } catch (error) {
    console.error("Ошибка загрузки рецепта:", error);
  }
});

const commentText = ref();

const getCommentAuthor = (userId) => {
  const user = userStore.getById(userId);
  if(user) {
    return user.username;
  }
  return "Guest";
};

const handleCommentCreate = async () => {
  const text = commentText.value.trim();
  if (text && text.length >= 10) {
    await commentStore.create(recipeId, text);
  }
};

```

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.



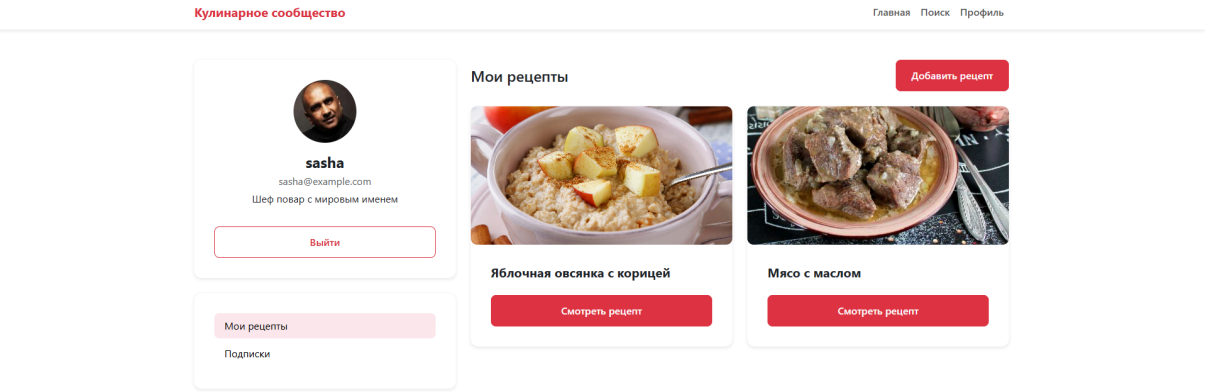


Рисунок 11 - Страница пользователя

Код страницы пользователя показан на рисунке 12.

```

<script setup>
import Navbar from "@/components/Navbar.vue";
import { useUserStore } from "@/stores/user";
import { onMounted } from "vue";
import router from "@/router";
import UserCard from "@/components/UserCard.vue";
import { useRecipeStore } from "@/stores/recipe";
import RecipeCard from "@/components/RecipeCard.vue";

const userStore = useUserStore();
const recipeStore = useRecipeStore();

onMounted(async () => {
  try {
    await userStore.getCurrentUser();
    await userStore.getSubs();
    await recipeStore.load(null, userStore.currentUser.id);
  } catch (error) {
    console.error("Ошибка загрузки пользователя:", error);
    userStore.logout();
  }
});

if (!userStore.isAuthenticated) {
  router.push("/login");
}

```

Рисунок 12 - Код страницы пользователя

## Вывод

В ходе выполнения данной лабораторной работы был изучен фреймворк Vue, а также выполнен перенос реализованного ранее приложения на него.