

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа 3

Выполнил:

Григорян Самвел

Группа К3440

**Проверил:
Добряков Д. И.**

Санкт-Петербург

2025 г.

Задача

Мигрировать ранее написанный сайт на фреймворк Vue.JS.

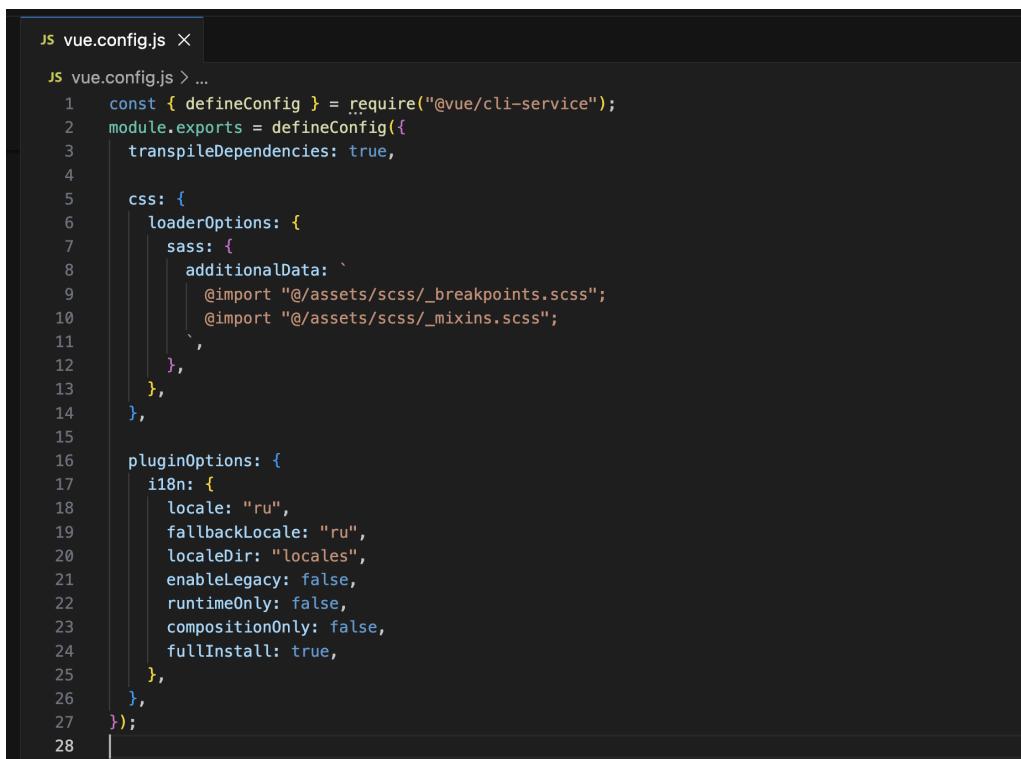
Минимальные требования:

- Должен быть подключен роутер
- Должна быть реализована работа с внешним API
- Разумное деление на компоненты
- Использование composable

Ход работы

В рамках работы был выполнен перенос ранее созданного сайта на фреймворк Vue.JS. Сначала подготовлено окружение (npm, Vite, плагин Vue), затем создана папка src и базовые точки входа (main.js, App.vue).

Структура проекта после миграции стала более организованной. В корне находятся package.json, vue.config.js, vite.config.js и index.html, который используется как точка монтирования всего SPA-приложения. Внутри папки src лежат main.js и App.vue, а также несколько ключевых директорий.



```
JS vue.config.js ×
JS vue.config.js > ...
1  const { defineConfig } = require("@vue/cli-service");
2  module.exports = defineConfig({
3    transpileDependencies: true,
4
5    css: {
6      loaderOptions: {
7        sass: {
8          additionalData: `
9            @import "@/assets/scss/_breakpoints.scss";
10           @import "@/assets/scss/_mixins.scss";
11           `,
12        },
13      },
14    },
15
16    pluginOptions: {
17      i18n: {
18        locale: "ru",
19        fallbackLocale: "ru",
20        localeDir: "locales",
21        enableLegacy: false,
22        runtimeOnly: false,
23        compositionOnly: false,
24        fullInstall: true,
25      },
26    },
27  });
28 |
```

Рисунок 1 - Файл vue.config.js

Папка components содержит переиспользуемые элементы интерфейса: кнопки, инпуты, навигацию, карточки и прочее. Папка views хранит страницы, которые отображаются через роутер. Папка layouts используется для компоновок страниц и частей страниц. Папка data предназначена для различных методов работы с данными. Там находятся функции для работы с API, с localStorage и конфигурация глобального состояния Vuex. Папка router содержит конфигурацию маршрутов и проверку доступа. Папка services содержит вспомогательные функции. В папке assets находятся стили, в том числе переменные тем, а также статические ресурсы.

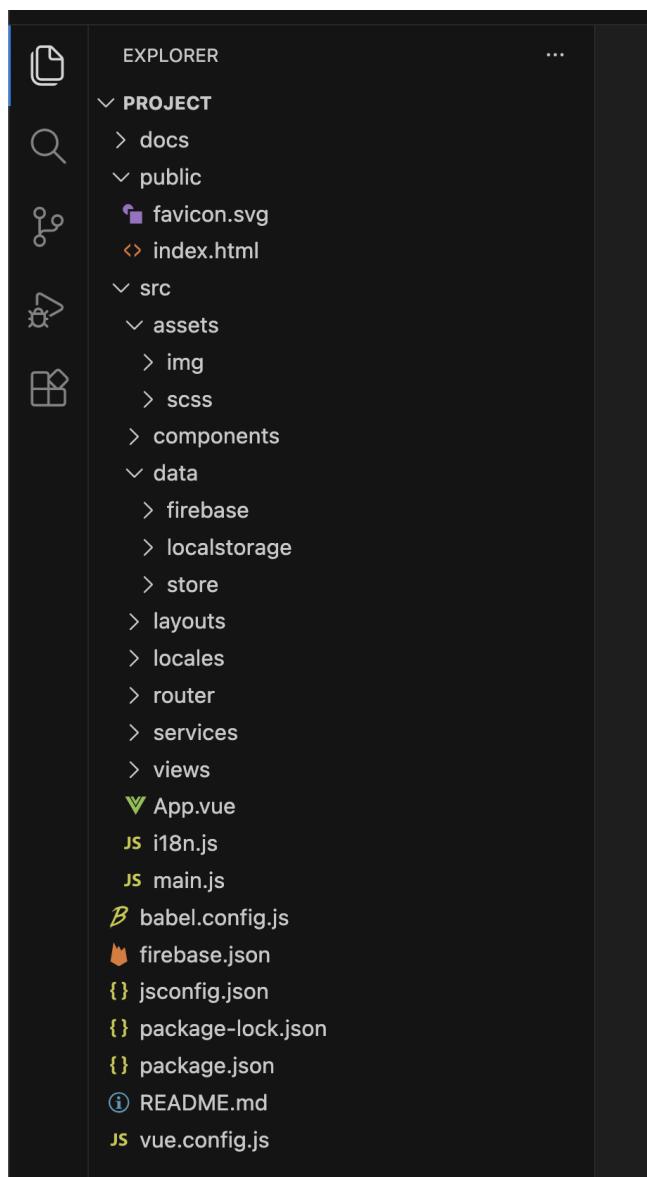
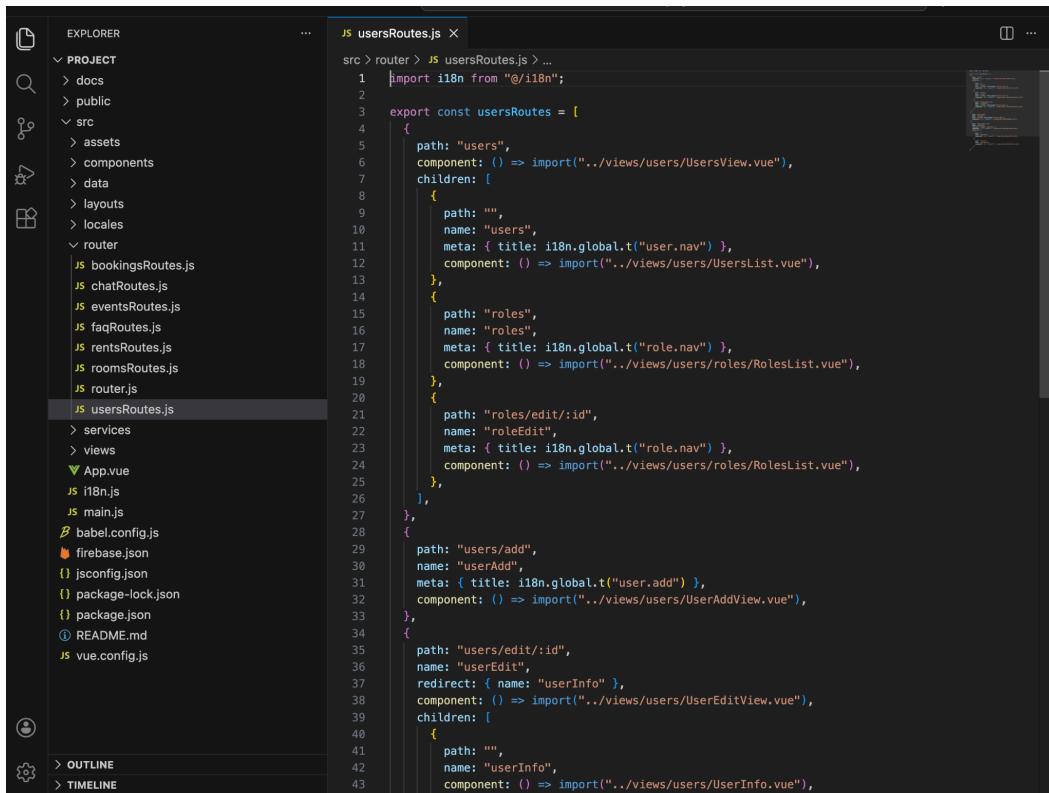


Рисунок 2 - Структура проекта

Работа страниц организована следующим образом. Например, при переходе пользователя на адрес наподобие /users/edit/:id роутер передает значение параметра id в компонент UserEditView. Компонент делает запрос к API и получает данные пользователя. Затем эти данные отображаются на странице.

Работа фильтрации происходит на странице общего списка. Пользователь указывает фильтры, форма связана с реактивными переменными. После отправки формы создаются параметры запроса, затем выполняется обращение к API. Сервер возвращает отфильтрованный список рецептов, который отображается на странице.

Роутер настроен так, что каждая страница подгружается по требованию. В нем также есть проверки авторизации. Если маршрут требует авторизации, а пользователь не вошел в систему, он будет перенаправлен на страницу входа. Компоненты отвечают за внешний вид и взаимодействие с пользователем, а повторяющиеся операции вынесены в отдельные функции. Такой подход делает код чище и удобнее.



The screenshot shows a code editor with the file `usersRoutes.js` open in the center pane. The left sidebar displays a project structure with various files like `bookingsRoutes.js`, `chatRoutes.js`, `eventsRoutes.js`, etc. The right sidebar shows a preview of the application's interface with several cards or cards.

```
src > router > JS usersRoutes.js > ...
1  import i18n from "@i18n";
2
3  export const usersRoutes = [
4    {
5      path: "users",
6      component: () => import("../views/users/UsersView.vue"),
7      children: [
8        {
9          path: "",
10         name: "users",
11         meta: { title: i18n.global.t("user.nav") },
12         component: () => import("../views/users/UsersList.vue"),
13       },
14       {
15         path: "roles",
16         name: "roles",
17         meta: { title: i18n.global.t("role.nav") },
18         component: () => import("../views/users/roles/RolesList.vue"),
19       },
20       {
21         path: "roles/edit/:id",
22         name: "roleEdit",
23         meta: { title: i18n.global.t("role.nav") },
24         component: () => import("../views/users/roles/RolesList.vue"),
25       },
26     ],
27   },
28   {
29     path: "users/add",
30     name: "userAdd",
31     meta: { title: i18n.global.t("user.add") },
32     component: () => import("../views/users/UserAddView.vue"),
33   },
34   {
35     path: "users/edit/:id",
36     name: "userEdit",
37     redirect: { name: "userInfo" },
38     component: () => import("../views/users/UserEditView.vue"),
39     children: [
40       {
41         path: "",
42         name: "userInfo",
43         component: () => import("../views/users/UserInfo.vue"),
44       },
45     ],
46   },
47 ];
```

Рисунок 3 - Файл роутинга пользователей

Вывод

В результате работы было выполнено перенесение проекта на Vue, подключение и настройка Vue Router, реализация централизованной работы с внешним API, разделение интерфейса на небольшие переиспользуемые компоненты. Все требования лабораторной работы выполнены, а структура приложения стала более гибкой и удобной для дальнейшего развития.