САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронтенд-разработка

Отчёт

Лабораторная работа №3.
Разработка одностраничного веб-приложения (SPA) с использованием фреймворка Vue.JS

Выполнил:
Колмогорова А.С.
K3342

Проверил:
Добряков Д. И.

Санкт-Петербург
2025 г.

СОДЕРЖАНИЕ

1 Задача

В рамках третьей лабораторной работы необходимо было мигрировать ранее разработанное приложение на фреймворк Vue.JS.

Ключевые требования приложения:

- должен быть подключён роутер
- должна быть реализована работа с внешним API (посредством axios)
- сделано разумное деление на компоненты (для демонстрации понимания компонентного подхода)
- использование composable для выделения повторяющегося функционала в отдельные файлы

Вариант: **Сервис для обмена рецептами и кулинарных блогов**

Основные составляющие приложения:

- Вход
- Регистрация
- Личный кабинет пользователя (сохраненные рецепты, публикации)
- Поиск рецептов с фильтрацией по типу блюда, сложности, поиск по названию
- Страница рецепта с фото, пошаговыми инструкциями и видео
- Социальные функции (комментарии, отметки в избранное)

2 Ход работы

Первым делом необходимо было создать и стартовать приложение на Vue.JS, для этого пригодились наработки из ДЗ 5, где мы овладели знанием базового функционала и требований для работы с фреймворком.

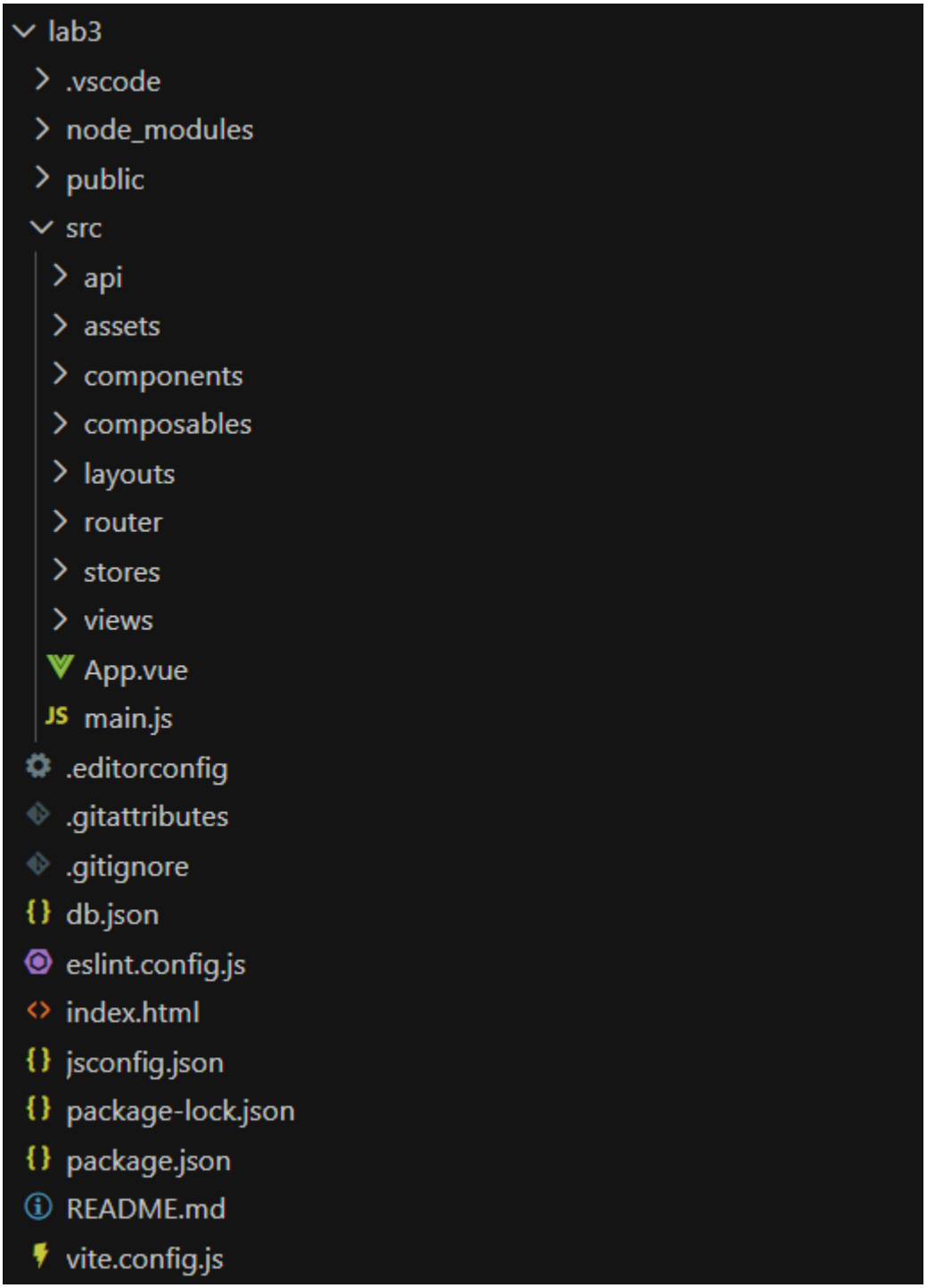
2.1 Предварительные действия, настройка Vue.JS

Алгоритм действий:

- устанавливаем node, npm и vue
- инициализация проекта, задаем ему параметры (подключаем pinia, router)
- изменяем часть скрипта package.json, чтобы корректно запускался сервер и localhost
- устанавливаем дополнительные пакеты:
  - axios для взаимодействия с внешним API;
  - pinia-persist для сохранения стейта проекта в localStorage;
  - bootstrap для базовых стилей.

Теперь можно приступать к переносу ЛР2 на Vue.

2.2 Структура проекта



Теперь последовательно рассмотрим содержимое всех директорий, с пояснениями, для чего тот или иной подход используется в работе.

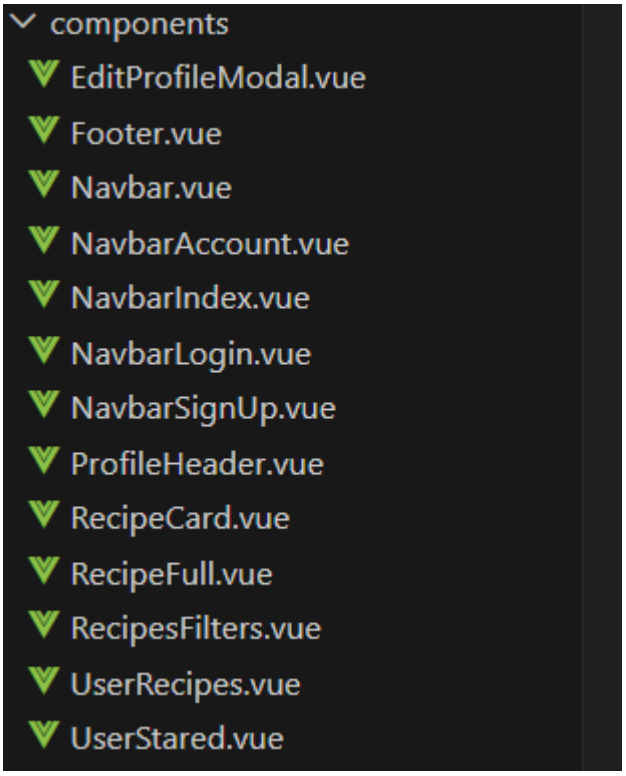
В папке **src** находятся все ключевые составляющие проекта:

- **api/instance.js:** инициализирует инстанс axios для отлавливания и обработки ошибок

- **api/recipes.js:** импортирует созданный инстанс и экспортирует RecipesApi с методами для работы с рецептами посредством GET/POST/PATCH запросов

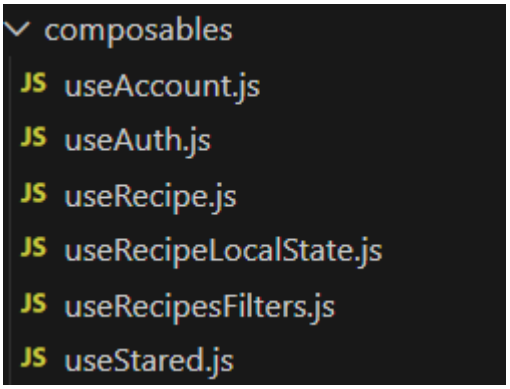
В **assets** хранятся базовые стили, а также две директории с images+videos.

В директории **components** расположены все компоненты, главная задача которых создавать отдельные логические части, который затем импортируются в представления (то есть страницы, которые видит пользователь). Для своей работы я выделила достаточно много отдельных компонентов, чтобы учесть разрозненность применяемых методов.



Они содержат как модальные окна и navbar (различные для различных страниц), так и карточки, меню с фильтрами для рецептов и т.д.). Их базовая структура – html-template и script (иногда setup – для использования composables).

Далее директория **composables** с js файлами, описывающими отдельную, более сложную логику взаимодействия объектов. Они импортируются в компоненты как методы или модули и используются явно.



В файле конфигурации роутера – **router/index.js** – есть массив со всеми путями, которые используются в приложении:

/ – index

/main

/login

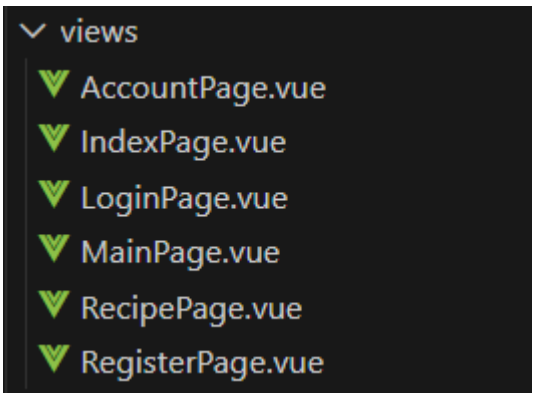
/register

/recipe/:id=<...>

/account (`meta: { requiresAuth: true }`)

При необходимости в роутинге используются так называемые хуки, ориентированные на различные события, например beforeEach, для выполнения проверок. А также meta-информация для расширения данных (ограничение доступа при авторизации, например).

В папке **views** расположены все представления или, иными словами, страницы, которые видит пользователь и на которые направлен роутинг.



В представления импортируются, как компоненты, так и composables объекты через setup-метод.

Еще одна важная составляющая проекта – db.json, подключенный сервер, хранящий в себе данные о пользователях, публичные рецепты, а также динамически изменяющийся при изменение/дополнении данных о пользователе и его рецептах.

3 Вывод

В рамках выполнения лабораторной работы были изучены возможности и особенности фреймворка Vue.JS, его основные методы и подходы. Освоены ключевые концепции: компонентный подход, клиентскую маршрутизацию и взаимодействие с REST API. Применение Vue.JS обеспечило создание интуитивно понятного интерфейса, подтвердив эффективность SPA-архитектуры для построения приложений.

Реализовано приложение, позволяющее пользователям смотреть выложенные рецепты, помечать их в папку «starred», добавлять свои, редактировать данные профиля.