САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №3

Выполнил:

Игнатьев Алексей

К3440

Проверил:
Добряков Д. И.

Санкт-Петербург 2025 г.

Задача

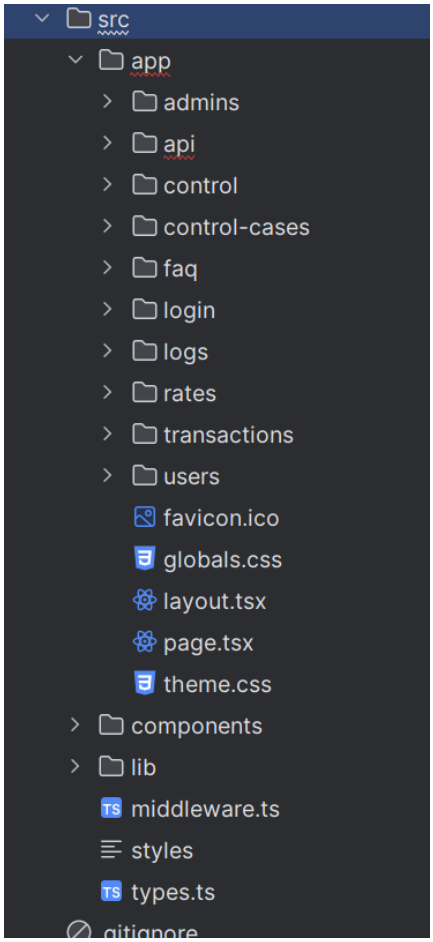
Разработка одностраничного веб-приложения (SPA) с использованием фреймворка Vue.JS

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 composable для выделения повторяющеся функционала в отдельные файлы

Ход работы

В качестве фреймворка для перехода был выбран Next.JS, структура проекта соответствует обще принятой в Next JS:

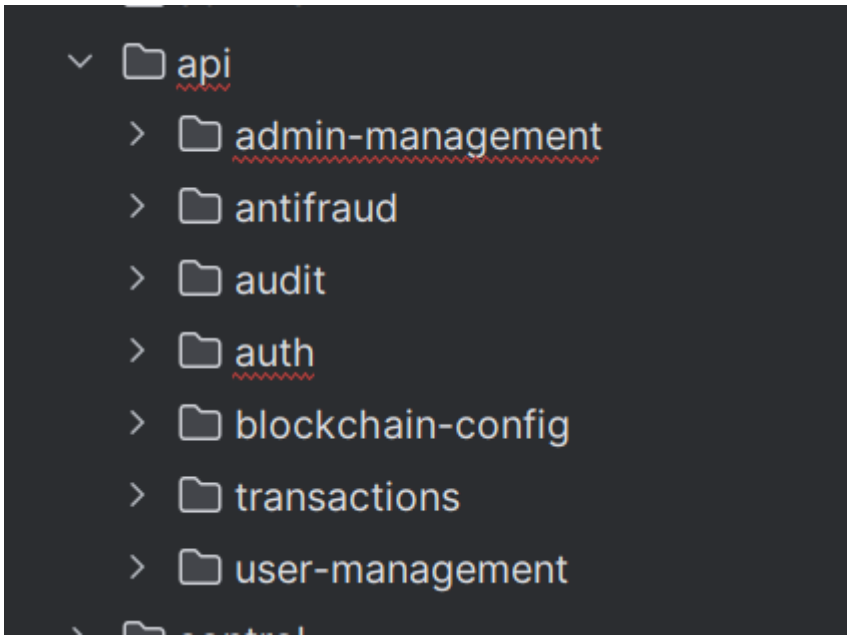


Разработанный ранее код был перенесен в роутеры, следуя официальной документации Next JS
Пример использования роутера после авторизации. Как видно из примера

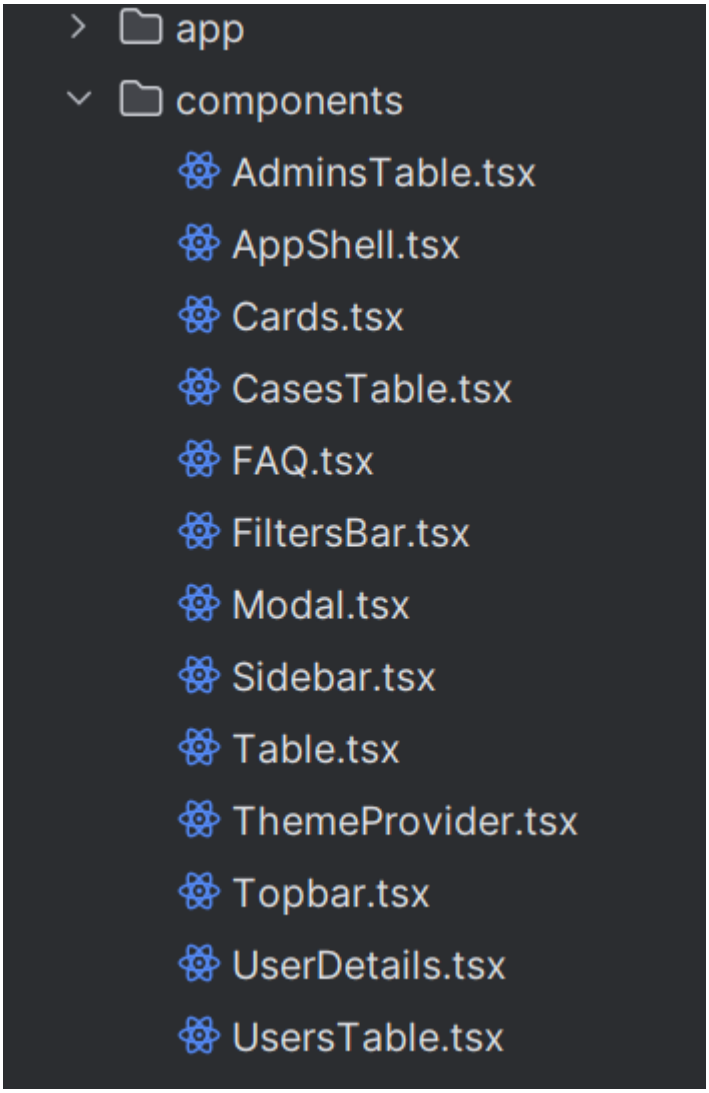
кода, прсое авторизации нас перебрасывает по ссылке указанной в next

```tsx
const handleSubmit : (e: FormEvent<Element>) => Promise<void> = async (e: React.FormEvent) : Promise<void> => { Show usages  openhands +1
  e.preventDefault();
  setError( value: null);
  setLoading( value: true);
  try {
    const res : Response = await fetch( input: "/api/auth/login", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify({ email, password }),
    });
    if (res.ok) {
      const next : string = params.get( name: "next") || "/";
      // Хард-редирект гарантирует, что браузер отправит свежие куки на следующий запрос
      if (typeof window !== "undefined") window.location.assign(next);
      else router.replace(next);
    } else {
      const data : any = await res.json().catch(() : { message: string } => ({ message: "Ошибка авторизации" }));
      setError(data.message || "Ошибка авторизации");
    }
  } catch (err) {
    setError( value: "Сетевая ошибка");
  } finally {
    setLoading( value: false);
  }
};
```

В приложении реализована работа с API, каждый API интерфейс выделен в свой блок.



Код поделен на компоненты, что представлено на скриншоте ниже.



Деление кода на компоненты позволяет переиспользовать в разных частях проекта код. Например, мы можем использовать в разных местах проекта компонент Table, чтобы реализация таблиц была единой.

Вывод

В результате выполнения работы проект был перенесён на Next.JS, соблюдая все принципы построения веб приложений на данном фреймворке