

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Фронт-энд разработка

Отчет  
Лабораторная работа 3

Выполнил:

Хурс Павел

Группа  
К3440

Проверил:  
Добряков Д. И.

Санкт-Петербург  
2026 г.

## Задача

Мигрировать ранее написанный сайт на фреймворк для разработки одностраничных веб-приложений (SPA).

Минимальные требования: - Должен быть подключён роутер - Должна быть реализована работа с внешним API - Разумное деление на компоненты - Использование composable

Вариант: Сервис для аренды недвижимости

## Ход работы

### Выбор технологий

Для реализации проекта был выбран фреймворк Next.js версии 16 - современный React-фреймворк для создания SPA. В качестве языка программирования использован TypeScript, обеспечивающий статическую типизацию. Для стилизации применён TailwindCSS в связке с библиотекой компонентов shadcn/ui на базе Radix UI.

### Реализация роутинга

В проекте используется App Router - встроенная система маршрутизации Next.js на основе файловой структуры. Каждая папка в директории app соответствует маршруту, а файл page.tsx определяет компонент страницы.

Реализованы следующие страницы: - Главная страница (/) - лендинг с преимуществами сервиса и популярными предложениями - Поиск недвижимости (/properties) - каталог с фильтрацией по типу, цене и расположению - Страница объекта (/properties/[id]) - детальная информация об объекте недвижимости - Вход (/login) - форма авторизации - Регистрация (/register) - форма создания аккаунта - Профиль (/profile) - личный кабинет с информацией и избранным - Сообщения (/messages) - система обмена сообщениями

### Работа с API

Серверная часть реализована через API Routes в директории app/api. Данные хранятся в JSON-файлах, имитируя работу с базой данных.

Реализованные эндпоинты: - GET/POST /api/properties - получение списка объектов с фильтрацией - GET /api/properties/[id] - получение информации об

объекте - POST /api/auth/login - авторизация пользователя - POST /api/auth/register - регистрация нового пользователя - POST /api/auth/logout - выход из системы - GET/POST/DELETE /api/favorites - управление избранным - GET/POST /api/messages - система сообщений - GET/POST /api/rentals - управление арендой

## **Компонентная архитектура**

Проект разделён на компоненты трёх уровней:

UI-компоненты (components/ui/) - базовые элементы интерфейса: Button, Card, Input, Select, Dialog, Badge, Avatar, Skeleton, Tabs, Slider, DropdownMenu, Label, Form.

Layout-компоненты (components/layout/) - структурные элементы: Header с навигацией и меню пользователя, Footer с информацией о проекте, ThemeToggle для переключения темы.

Business-компоненты (components/properties/) - бизнес-логика: PropertyCard для отображения карточки объекта недвижимости с информацией о цене, площади, расположении и возможностью добавления в избранное.

## **Composable (React Hooks)**

Вместо Vue composable в React используются custom hooks, выполняющие аналогичную функцию инкапсуляции логики:

useProperties - получение списка объектов с поддержкой фильтрации по типу, цене, расположению и другим параметрам.

useProperty - получение детальной информации об одном объекте по его идентификатору.

useLocalStorage - универсальный hook для работы с localStorage, обеспечивающий персистентность данных между сессиями.

## **Контексты**

Для управления глобальным состоянием используется Context API:

AuthContext - хранит информацию о текущем пользователе, состоянии авторизации и предоставляет методы login, register, logout.

FavoritesContext - управляет списком избранных объектов, синхронизирует данные между вкладками браузера через событие storage.

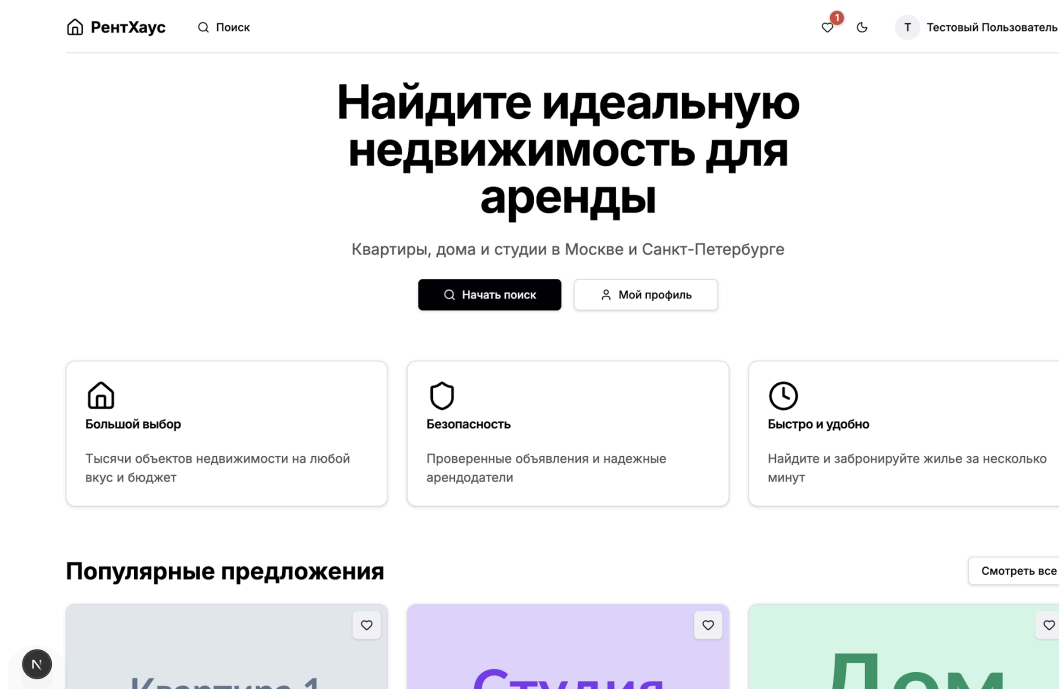
ThemeContext - отвечает за переключение между светлой и тёмной темой, учитывает системные настройки пользователя.

## Дополнительные возможности

Реализована поддержка тёмной и светлой темы с переключателем в шапке сайта. Интерфейс адаптирован под различные размеры экранов. Система фильтрации позволяет искать объекты по типу недвижимости, диапазону цены и местоположению.

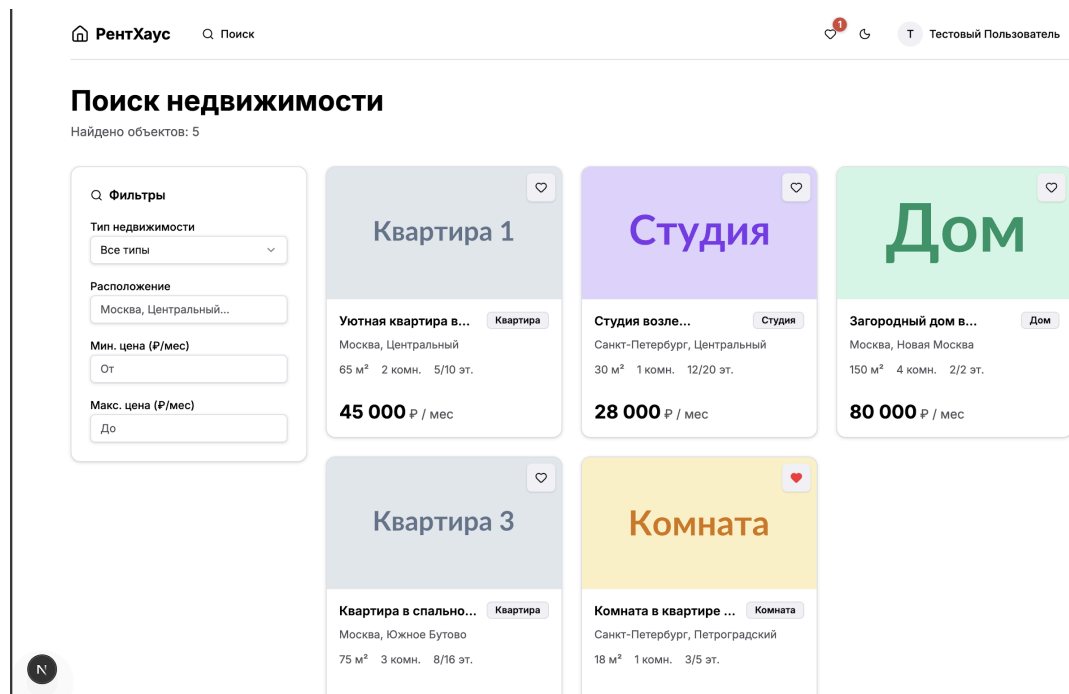
## Скриншоты

Главная страница в светлой теме



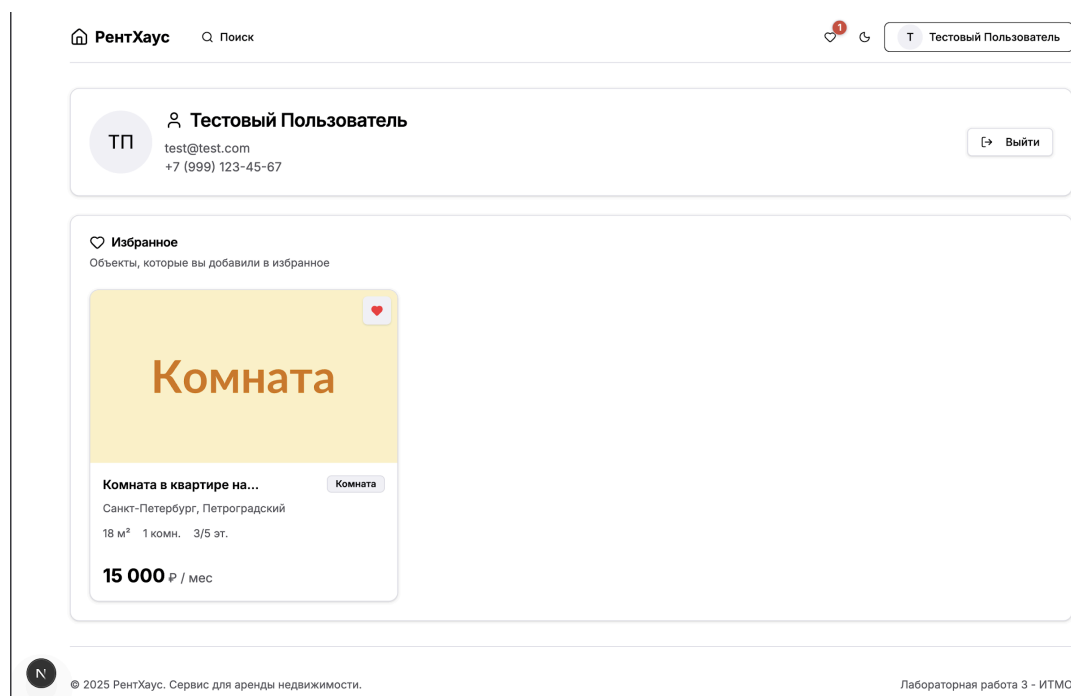
Главная страница (светлая тема)

Страница поиска недвижимости с фильтрами в светлой теме



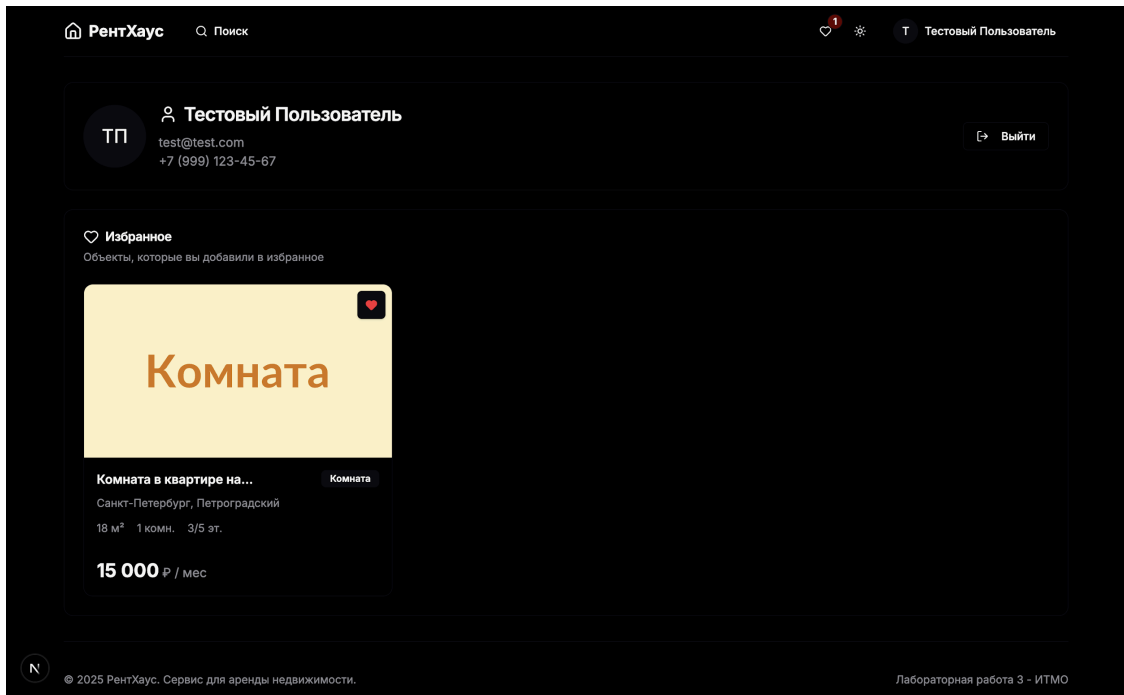
Поиск недвижимости (светлая тема)

Профиль пользователя с разделом избранного в светлой теме



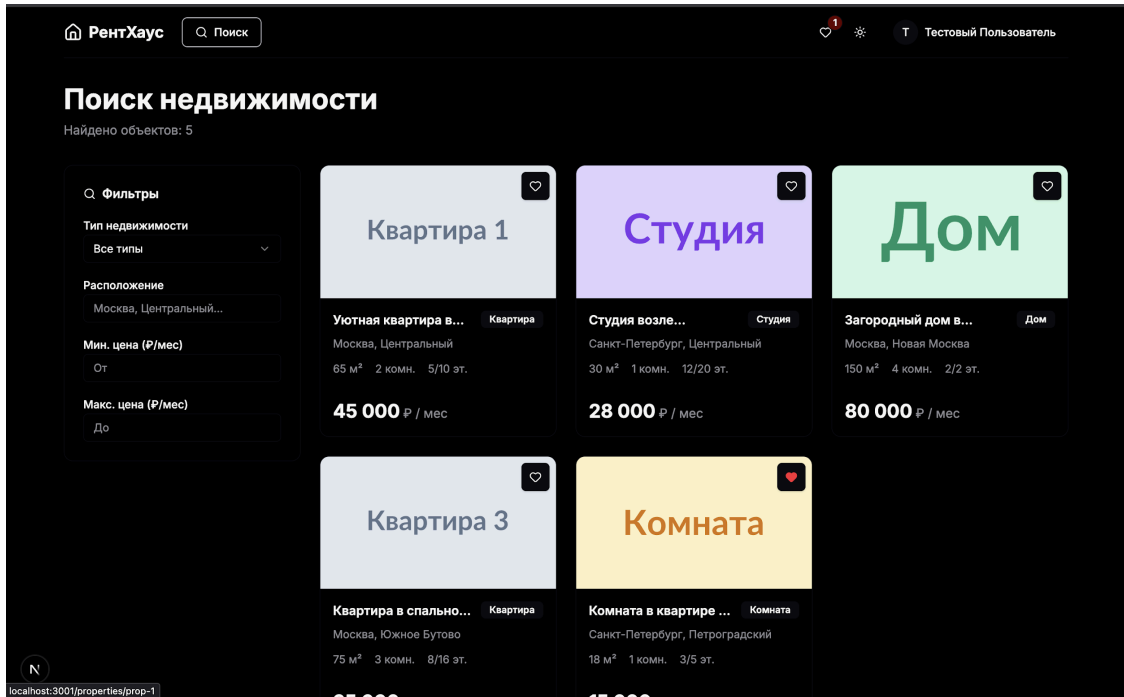
Профиль с избранным (светлая тема)

Профиль пользователя в тёмной теме



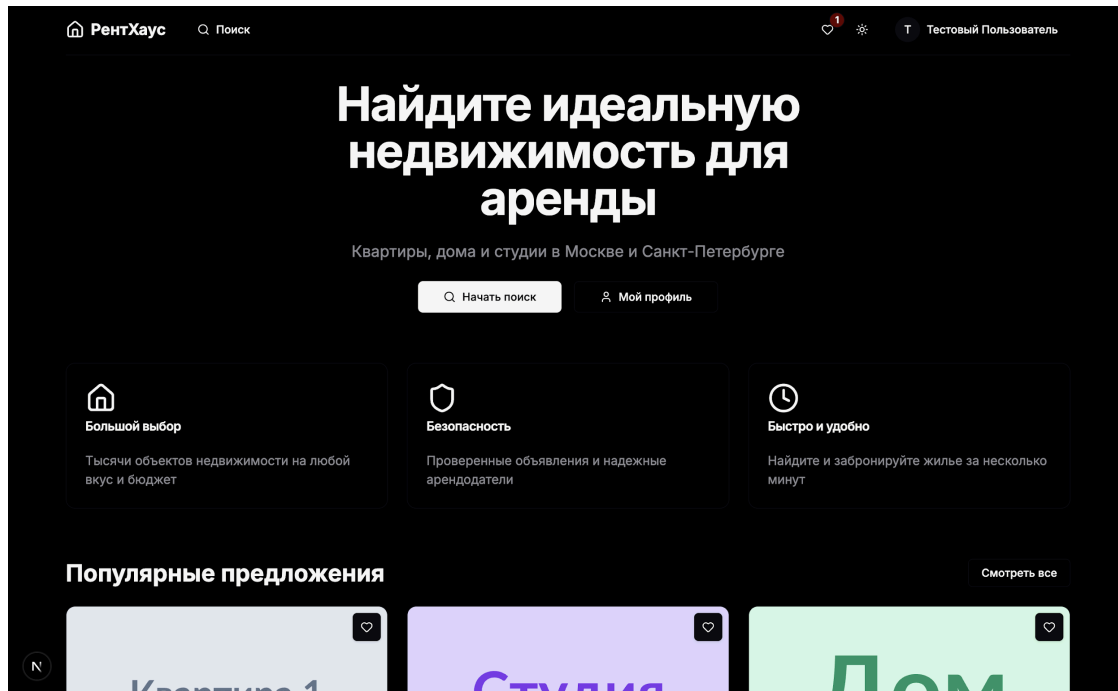
Профиль с избранным (тёмная тема)

Страница поиска недвижимости в тёмной теме



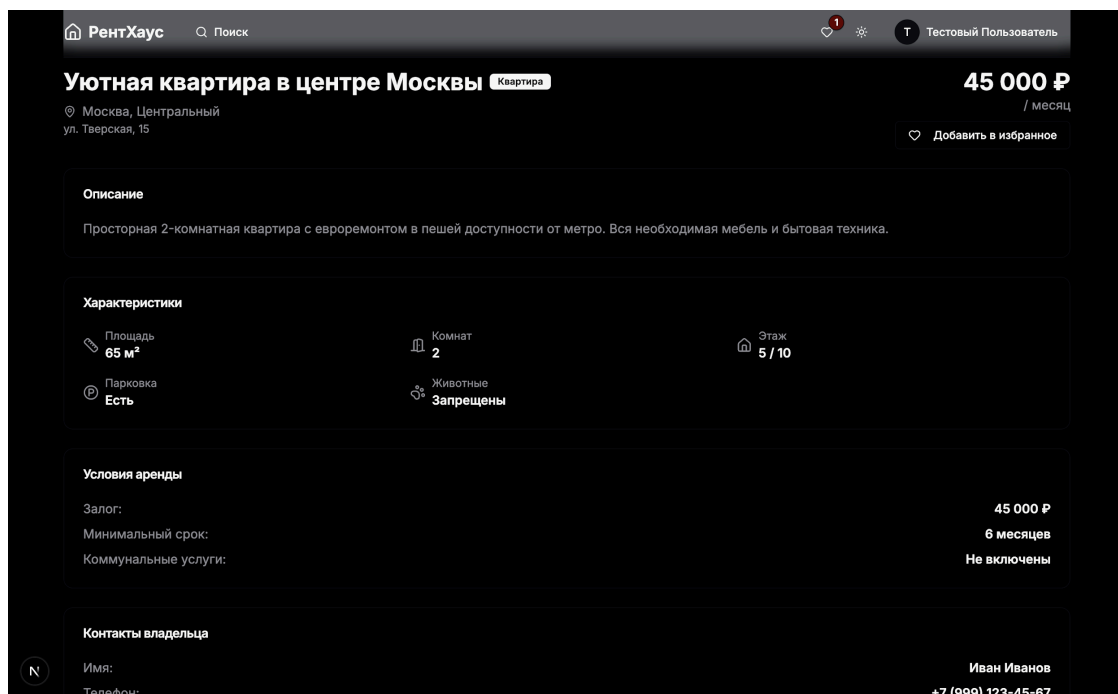
Поиск недвижимости (тёмная тема)

Главная страница в тёмной теме



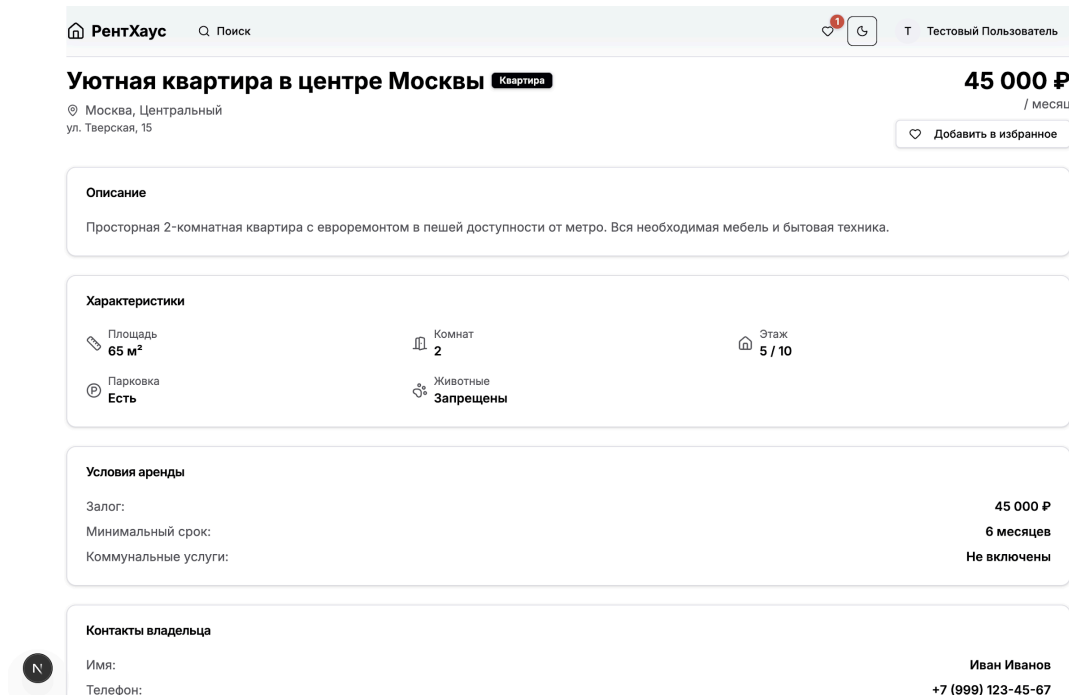
Главная страница (тёмная тема)

Детальная страница объекта недвижимости в тёмной теме



Страница объекта (тёмная тема)

Детальная страница объекта недвижимости в светлой теме



Страница объекта (светлая тема)

## Вывод

В ходе выполнения лабораторной работы было разработано одностраничное веб-приложение «РентХаус» - сервис для аренды недвижимости. Приложение реализовано на фреймворке Next.js с использованием TypeScript.

Выполнены все минимальные требования к работе. Подключён роутер с файловой маршрутизацией, реализовано семь страниц приложения. Создан набор API endpoints для работы с данными об объектах недвижимости, пользователях, избранном и сообщениях. Проект разделён на компоненты трёх уровней: UI, layout и business. Логика вынесена в переиспользуемые hooks, выполняющие роль composable.

Дополнительно реализована темизация с переключением между светлой и тёмной темой, система избранного с синхронизацией между вкладками, а также расширенная фильтрация объектов по различным параметрам.