САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Домашняя работа №5

Основы работы с менеджером зависимостей npm

Выполнил:
Чернышев Михаил Павлович
Группа К3441

Проверил:
Добряков Д. И.

Санкт-Петербург
2026 г.

Содержание

1 Задача

Изучить основные команды пакетного менеджера npm и научиться создавать проект на Vue.js.

2 Основы npm

2.1 Что такое npm

npm (Node Package Manager) — это менеджер пакетов для JavaScript, который позволяет:

- Устанавливать сторонние библиотеки
- Управлять версиями зависимостей
- Запускать скрипты для сборки и разработки
- Публиковать собственные пакеты

2.2 Основные команды npm

Инициализация проекта

```
# Создание package.json с интерактивным режимом
npm init

# Создание package.json с настройками по умолчанию
npm init -y
```

Установка зависимостей

```
# Установка всех зависимостей из package.json
npm install
npm i

# Установка конкретного пакета
npm install vue
npm i vue

# Установка с указанием версии
npm install vue@3.4.0

# Установка как dev-зависимость
npm install --save-dev vite
npm i -D vite

# Глобальная установка
npm install -g @vue/cli
```

Удаление и обновление

```
# Удаление пакета
npm uninstall lodash

# Обновление всех пакетов
npm update

# Обновление конкретного пакета
npm update vue

# Проверка устаревших пакетов
npm outdated
```

Запуск скриптов

```
# Запуск скрипта из package.json
npm run dev
npm run build
npm run test

# Сокращённые команды
npm start    # эквивалент npm run start
npm test     # эквивалент npm run test
```

Информация о пакетах

```
# Просмотр установленных пакетов
npm list
npm ls

# Только верхний уровень
npm list --depth=0

# Информация о пакете
npm info vue

# Поиск пакетов
npm search vue router
```

2.3 Файл package.json

Структура package.json

```
{
  "name": "renthome-vue",
  "version": "1.0.0",
  "description": "Сервис аренды недвижимости на Vue.js",
  "author": "Чернышев Михаил",
  "license": "MIT",

  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview",
    "lint": "eslint . --ext .vue,.js,.ts"
  },

  "dependencies": {
    "vue": "^3.4.0",
    "vue-router": "^4.2.0",
    "pinia": "^2.1.0",
    "axios": "^1.6.0"
  },

  "devDependencies": {
    "@vitejs/plugin-vue": "^5.0.0",
    "vite": "^5.0.0",
    "eslint": "^8.56.0",
    "eslint-plugin-vue": "^9.20.0"
  }
}
```

2.4 Разница между dependencies и devDependencies

- **dependencies** — пакеты, необходимые для работы приложения в production (Vue, axios, vue-router)
- **devDependencies** — пакеты для разработки (Vite, ESLint, тестовые фреймворки)

2.5 Файл package-lock.json

Файл `package-lock.json` фиксирует точные версии всех установленных пакетов и их зависимостей. Это гарантирует, что на разных машинах будут установлены одинаковые версии.

Важно: этот файл нужно коммитить в git!

3 Создание проекта Vue.js

3.1 Создание проекта через Vite

Создание нового проекта

```
# Создание проекта
npm create vite@latest renthome-vue -- --template vue

# Переход в папку проекта
cd renthome-vue

# Установка зависимостей
npm install

# Запуск dev-сервера
npm run dev
```

3.2 Структура проекта Vue

```
renthome-vue/
 node_modules/          # Установленные пакеты
 public/                # Статические файлы
    favicon.ico
 src/
    assets/            # Ресурсы (стили, изображения)
       main.css
    components/        # Компоненты
       PropertyCard.vue
       PropertyList.vue
       SearchFilters.vue
       AppHeader.vue
    views/             # Страницы
       HomeView.vue
       PropertyView.vue
    router/            # Маршрутизация
       index.js
    stores/            # Состояние (Pinia)
       properties.js
    App.vue            # Корневой компонент
    main.js            # Точка входа
 index.html
 package.json
 vite.config.js
 README.md
```

4 Код приложения

4.1 Точка входа (main.js)

src/main.js

```
import { createApp } from 'vue'
import { createPinia } from 'pinia'
import router from './router'
import App from './App.vue'
import './assets/main.css'

const app = createApp(App)

app.use(createPinia())
app.use(router)

app.mount('#app')
```

4.2 Корневой компонент (App.vue)

src/App.vue

```
<template>
    <div id="app">
        <AppHeader />
        <main class="container">
            <RouterView />
        </main>
        <AppFooter />
    </div>
</template>

<script setup>
import AppHeader from './components/AppHeader.vue'
import AppFooter from './components/AppFooter.vue'
</script>

<style>
#app {
    min-height: 100vh;
    display: flex;
    flex-direction: column;
}

main {
    flex: 1;
    padding: 20px;
}
</style>
```

4.3 Компонент AppHeader

src/components/AppHeader.vue

```
<template>
    <header class="header">
        <div class="container">
            <RouterLink to="/" class="logo">
                 RentHome
            </RouterLink>

            <nav class="nav">
                <RouterLink to="/" class="nav-link">Главная</RouterLink>
                <RouterLink to="/favorites" class="nav-link">Избранное</RouterLink>
                <RouterLink to="/profile" class="nav-link">Профиль</RouterLink>
            </nav>

            <button class="theme-toggle" @click="toggleTheme">
                {{ isDark ? '' : '' }}
            </button>
        </div>
    </header>
</template>

<script setup>
import { ref } from 'vue'

const isDark = ref(false)

function toggleTheme() {
    isDark.value = !isDark.value
    document.documentElement.setAttribute(
        'data-theme',
        isDark.value ? 'dark' : 'light'
    )
}
</script>

<style scoped>
.header {
    background: var(--primary-color);
    color: white;
    padding: 16px 0;
}

.header .container {
    display: flex;
    align-items: center;
    justify-content: space-between;
}

.logo {
    font-size: 24px;
    font-weight: bold;
```

```
    color: white;
    text-decoration: none;
}

.nav {
    display: flex;
    gap: 24px;
}

.nav-link {
    color: white;
    text-decoration: none;
    opacity: 0.8;
    transition: opacity 0.2s;
}

.nav-link:hover,
.nav-link.router-link-active {
    opacity: 1;
}

.theme-toggle {
    background: none;
    border: none;
    font-size: 20px;
    cursor: pointer;
}
</style>
```

4.4 Компонент PropertyCard

src/components/PropertyCard.vue

```
<template>
    <article class="property-card">
        <div class="property-image">
            <img :src="property.image" :alt="property.title">
            <button
                class="favorite-btn"
                :class="{ active: isFavorite }"
                @click="toggleFavorite"
            >
                {{ isFavorite ? '' : '' }}
            </button>
        </div>

        <div class="property-content">
            <h3 class="property-title">{{ property.title }}</h3>

            <p class="property-location">
                 {{ property.location }}
            </p>

            <div class="property-features">
                <span> {{ property.rooms }} комн.</span>
                <span> {{ property.area }} м²</span>
            </div>

            <div class="property-footer">
                <span class="property-price">
                    {{ formatPrice(property.price) }} /мес
                </span>
                <RouterLink
                    :to="`/property/${property.id}`"
                    class="btn btn-primary"
                >
                    Подробнее
                </RouterLink>
            </div>
        </div>
    </article>
</template>

<script setup>
import { computed } from 'vue'
import { useFavoritesStore } from '@/stores/favorites'

const props = defineProps({
    property: {
        type: Object,
        required: true
    }
})
```

```
const favoritesStore = useFavoritesStore()

const isFavorite = computed(() =>
    favoritesStore.isFavorite(props.property.id)
)

function toggleFavorite() {
    favoritesStore.toggle(props.property.id)
}

function formatPrice(price) {
    return price.toLocaleString('ru-RU')
}
</script>

<style scoped>
.property-card {
    background: var(--card-bg);
    border-radius: 12px;
    overflow: hidden;
    box-shadow: var(--shadow-md);
    transition: transform 0.2s, box-shadow 0.2s;
}

.property-card:hover {
    transform: translateY(-4px);
    box-shadow: var(--shadow-lg);
}

.property-image {
    position: relative;
    height: 200px;
}

.property-image img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

.favorite-btn {
    position: absolute;
    top: 12px;
    right: 12px;
    background: white;
    border: none;
    border-radius: 50%;
    width: 40px;
    height: 40px;
    cursor: pointer;
    font-size: 18px;
```

```
    transition: transform 0.2s;
}

.favorite-btn:hover {
    transform: scale(1.1);
}

.property-content {
    padding: 16px;
}

.property-title {
    margin: 0 0 8px;
    font-size: 18px;
}

.property-location {
    color: var(--text-secondary);
    margin: 0 0 12px;
}

.property-features {
    display: flex;
    gap: 16px;
    margin-bottom: 16px;
    color: var(--text-secondary);
}

.property-footer {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.property-price {
    font-size: 20px;
    font-weight: bold;
    color: var(--primary-color);
}
</style>
```

4.5 Компонент SearchFilters

src/components/SearchFilters.vue

```
<template>
    <aside class="filters">
        <h2>Фильтры</h2>

        <div class="filter-group">
            <label>Тип жилья</label>
            <select v-model="filters.type">
                <option value="">Любой</option>
                <option value="apartment">Квартира</option>
                <option value="house">Дом</option>
                <option value="room">Комната</option>
            </select>
        </div>

        <div class="filter-group">
            <label>Цена</label>
            <div class="price-inputs">
                <input
                    v-model.number="filters.priceFrom"
                    type="number"
                    placeholder="От"
                >
                <input
                    v-model.number="filters.priceTo"
                    type="number"
                    placeholder="До"
                >
            </div>
        </div>

        <div class="filter-group">
            <label>Комнат</label>
            <div class="room-buttons">
                <button
                    v-for="n in 4"
                    :key="n"
                    :class="{ active: filters.rooms === n }"
                    @click="filters.rooms = filters.rooms === n ? null : n"
                >
                    {{ n === 4 ? '4+' : n }}
                </button>
            </div>
        </div>

        <button class="btn btn-primary" @click="applyFilters">
            Применить
        </button>
        <button class="btn btn-secondary" @click="resetFilters">
            Сбросить
        </button>
```

```
    </aside>
</template>

<script setup>
import { reactive } from 'vue'

const emit = defineEmits(['filter'])

const filters = reactive({
    type: '',
    priceFrom: null,
    priceTo: null,
    rooms: null
})

function applyFilters() {
    emit('filter', { ...filters })
}

function resetFilters() {
    Object.assign(filters, {
        type: '',
        priceFrom: null,
        priceTo: null,
        rooms: null
    })
    emit('filter', {})
}
</script>

<style scoped>
.filters {
    background: var(--card-bg);
    padding: 20px;
    border-radius: 12px;
    box-shadow: var(--shadow-sm);
}

.filter-group {
    margin-bottom: 20px;
}

.filter-group label {
    display: block;
    margin-bottom: 8px;
    font-weight: 600;
}

.filter-group select,
.filter-group input {
    width: 100%;
    padding: 10px;
```

```
    border: 1px solid var(--border-color);
    border-radius: 8px;
}

.price-inputs {
    display: flex;
    gap: 10px;
}

.price-inputs input {
    flex: 1;
}

.room-buttons {
    display: flex;
    gap: 8px;
}

.room-buttons button {
    flex: 1;
    padding: 10px;
    border: 1px solid var(--border-color);
    background: var(--bg-secondary);
    border-radius: 8px;
    cursor: pointer;
}

.room-buttons button.active {
    background: var(--primary-color);
    color: white;
    border-color: var(--primary-color);
}

.btn {
    width: 100%;
    padding: 12px;
    border: none;
    border-radius: 8px;
    cursor: pointer;
    margin-top: 10px;
}

.btn-primary {
    background: var(--primary-color);
    color: white;
}

.btn-secondary {
    background: var(--bg-secondary);
}
</style>
```

4.6 Страница HomeView

src/views/HomeView.vue

```
<template>
    <div class="home">
        <h1>Найдите идеальное жильё</h1>

        <div class="home-layout">
            <SearchFilters @filter="handleFilter" />

            <section class="properties">
                <div class="properties-header">
                    <span>Найдено: {{ filteredProperties.length }}</span>
                    <select v-model="sortBy">
                        <option value="date">По дате</option>
                        <option value="price_asc">Сначала дешёвые</option>
                        <option value="price_desc">Сначала дорогие</option>
                    </select>
                </div>

                <div v-if="loading" class="loading">
                    Загрузка...
                </div>

                <div v-else class="properties-grid">
                    <PropertyCard
                        v-for="property in filteredProperties"
                        :key="property.id"
                        :property="property"
                    />
                </div>
            </section>
        </div>
    </div>
</template>

<script setup>
import { ref, computed, onMounted } from 'vue'
import PropertyCard from '@/components/PropertyCard.vue'
import SearchFilters from '@/components/SearchFilters.vue'
import { usePropertiesStore } from '@/stores/properties'

const propertiesStore = usePropertiesStore()

const loading = ref(true)
const sortBy = ref('date')
const currentFilters = ref({})

const filteredProperties = computed(() => {
    let result = [...propertiesStore.properties]

    const f = currentFilters.value
    if (f.type) result = result.filter(p => p.type === f.type)
```

```
    if (f.priceFrom) result = result.filter(p => p.price >= f.priceFrom)
    if (f.priceTo) result = result.filter(p => p.price <= f.priceTo)
    if (f.rooms) result = result.filter(p => p.rooms >= f.rooms)

    if (sortBy.value === 'price_asc') {
        result.sort((a, b) => a.price - b.price)
    } else if (sortBy.value === 'price_desc') {
        result.sort((a, b) => b.price - a.price)
    }

    return result
})

function handleFilter(filters) {
    currentFilters.value = filters
}

onMounted(async () => {
    await propertiesStore.fetchProperties()
    loading.value = false
})
</script>

<style scoped>
.home h1 {
    margin-bottom: 24px;
}

.home-layout {
    display: grid;
    grid-template-columns: 280px 1fr;
    gap: 24px;
}

.properties-header {
    display: flex;
    justify-content: space-between;
    margin-bottom: 20px;
}

.properties-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(300px, 1fr));
    gap: 24px;
}

.loading {
    text-align: center;
    padding: 40px;
    color: var(--text-secondary);
}
```

```
</style>
```

4.7 Store (Pinia)

src/stores/properties.js

```
import { defineStore } from 'pinia'
import axios from 'axios'

export const usePropertiesStore = defineStore('properties', {
    state: () => ({
        properties: [],
        loading: false,
        error: null
    }),

    actions: {
        async fetchProperties() {
            this.loading = true
            try {
                const response = await axios.get('/api/properties')
                this.properties = response.data
            } catch (error) {
                this.error = error.message
            } finally {
                this.loading = false
            }
        }
    }
})
```

src/stores/favorites.js

```
import { defineStore } from 'pinia'

export const useFavoritesStore = defineStore('favorites', {
    state: () => ({
        ids: JSON.parse(localStorage.getItem('favorites') || '[]')
    }),

    getters: {
        isFavorite: (state) => (id) => state.ids.includes(id),
        count: (state) => state.ids.length
    },

    actions: {
        toggle(id) {
            const index = this.ids.indexOf(id)
            if (index > -1) {
                this.ids.splice(index, 1)
            } else {
                this.ids.push(id)
            }
            localStorage.setItem('favorites', JSON.stringify(this.ids))
        }
    }
})
```

4.8 Router

src/router/index.js

```
import { createRouter, createWebHistory } from 'vue-router'

const routes = [
    {
        path: '/',
        name: 'Home',
        component: () => import('@/views/HomeView.vue')
    },
    {
        path: '/property/:id',
        name: 'Property',
        component: () => import('@/views/PropertyView.vue')
    },
    {
        path: '/favorites',
        name: 'Favorites',
        component: () => import('@/views/FavoritesView.vue')
    },
    {
        path: '/profile',
        name: 'Profile',
        component: () => import('@/views/ProfileView.vue')
    }
]

const router = createRouter({
    history: createWebHistory(),
    routes
})

export default router
```

5 Вывод

В ходе выполнения домашней работы были изучены основы работы с npm и создан проект на Vue.js.

Изученные команды npm:

- `npm init` — инициализация проекта
- `npm install` — установка зависимостей
- `npm run dev` — запуск dev-сервера
- `npm run build` — сборка для production
- `npm update`, `npm outdated` — обновление пакетов

Созданные компоненты Vue:

- `AppHeader.vue` — шапка с навигацией
- `PropertyCard.vue` — карточка объекта
- `SearchFilters.vue` — фильтры поиска
- `HomeView.vue` — главная страница

Использованные технологии:

- Vue 3 с Composition API (`<script setup>`)
- Vue Router для маршрутизации
- Pinia для управления состоянием
- Vite для сборки проекта
- Axios для HTTP-запросов

Результат: создано работающее Vue-приложение сервиса аренды недвижимости с компонентами, роутингом и state management.

6 Скриншоты приложения

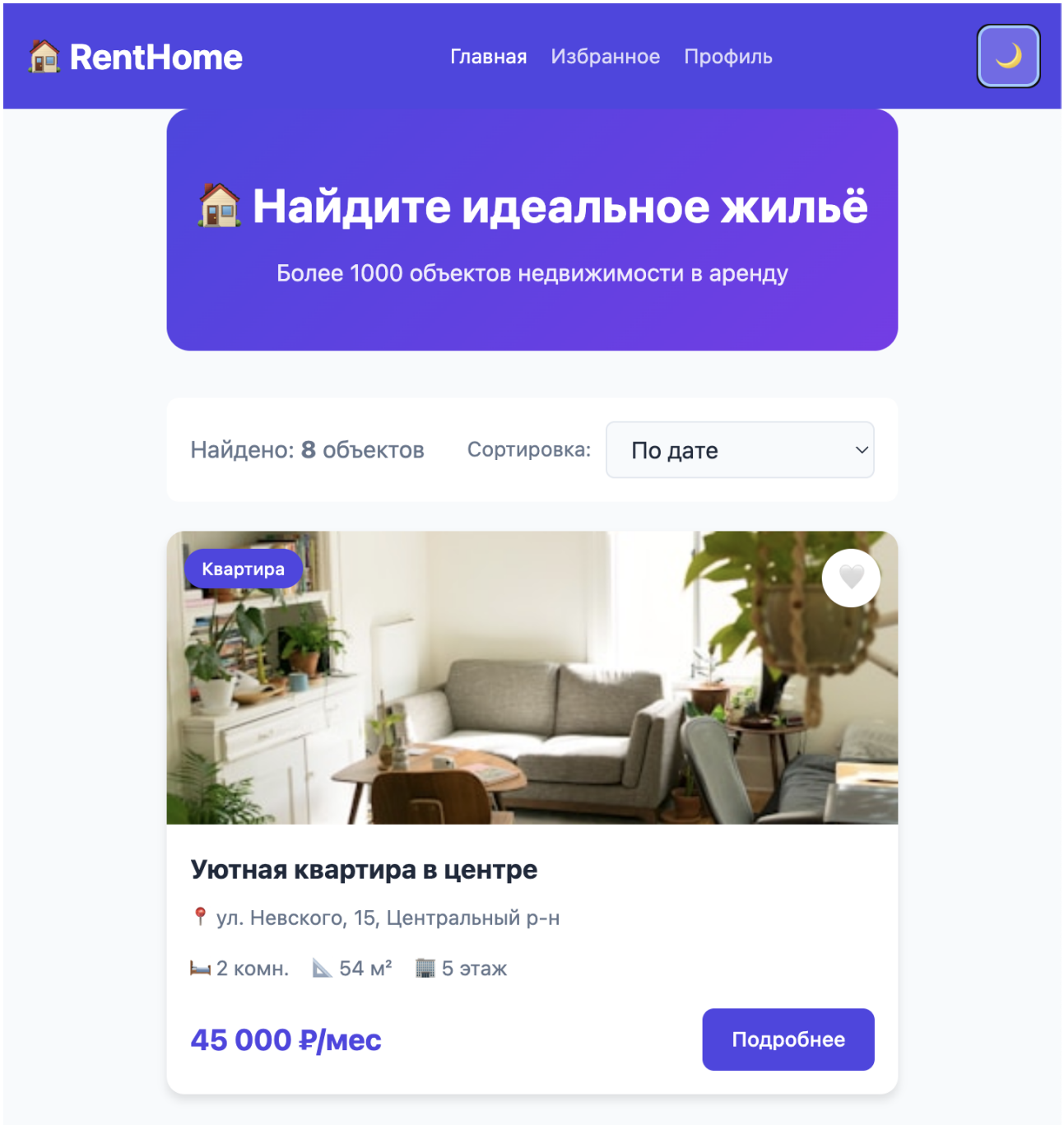

Рис. 1: Главная страница с карточками объектов и фильтрами

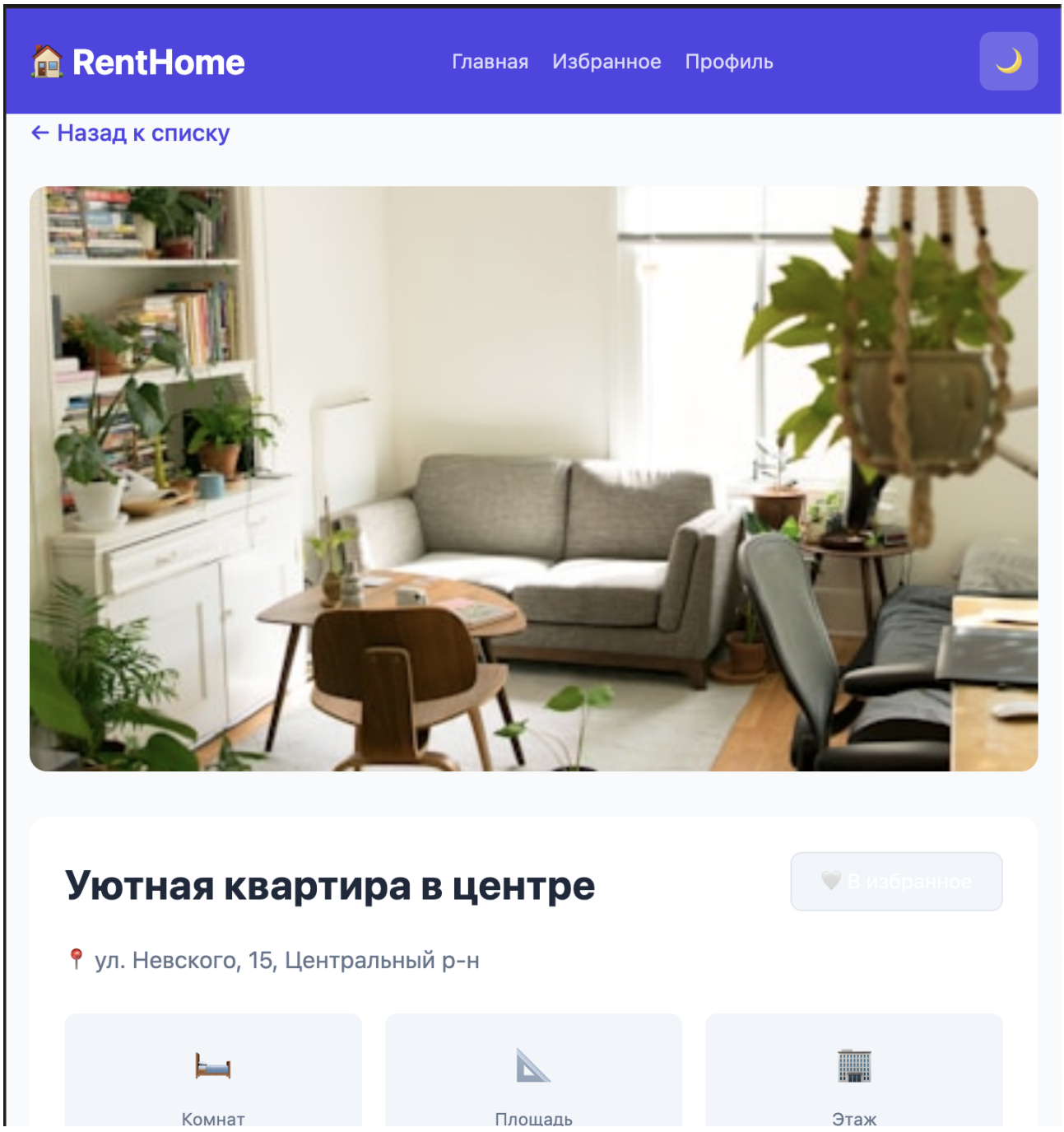

Рис. 2: Детальная страница объекта недвижимости

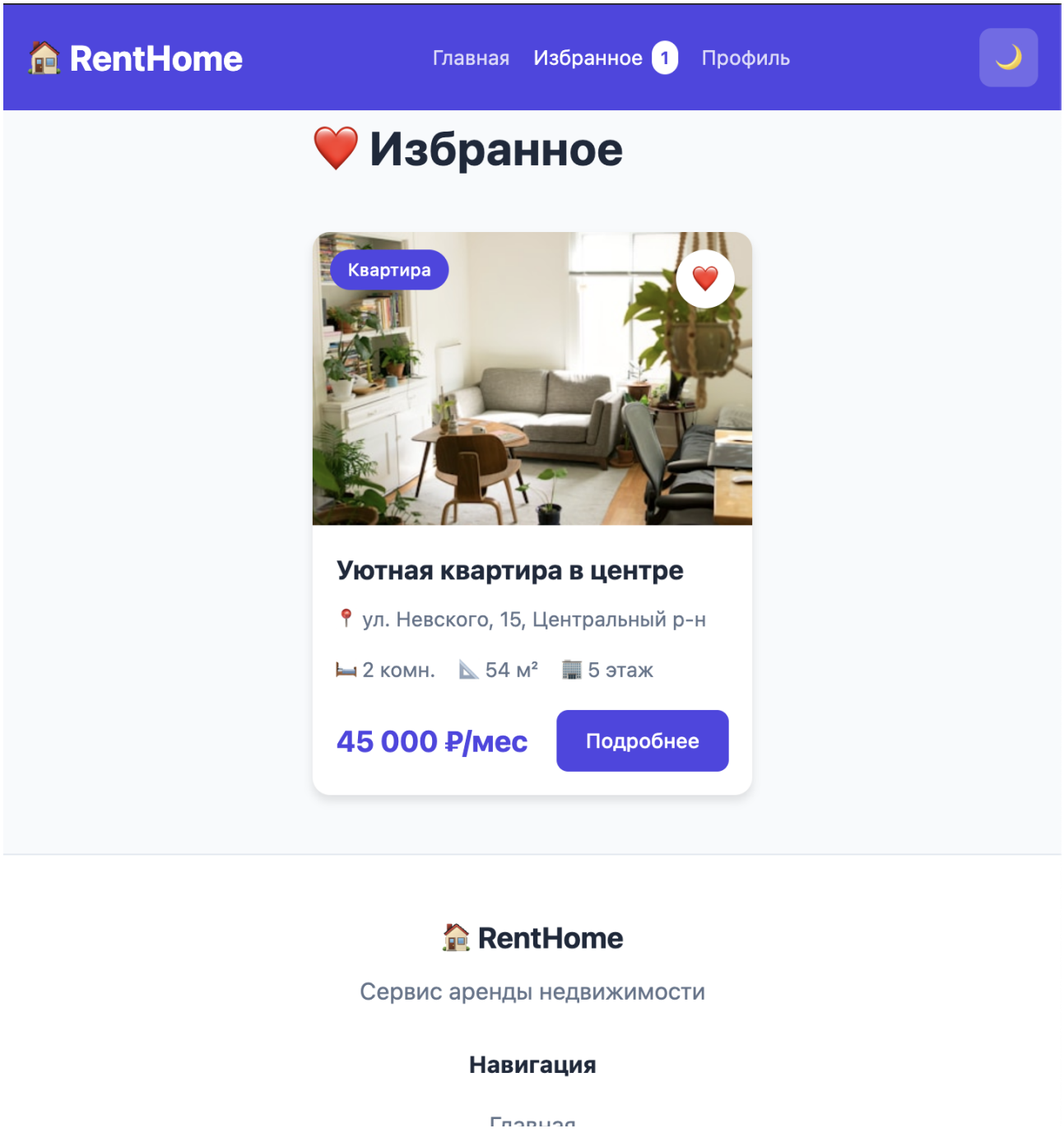

Рис. 3: Страница избранных объектов

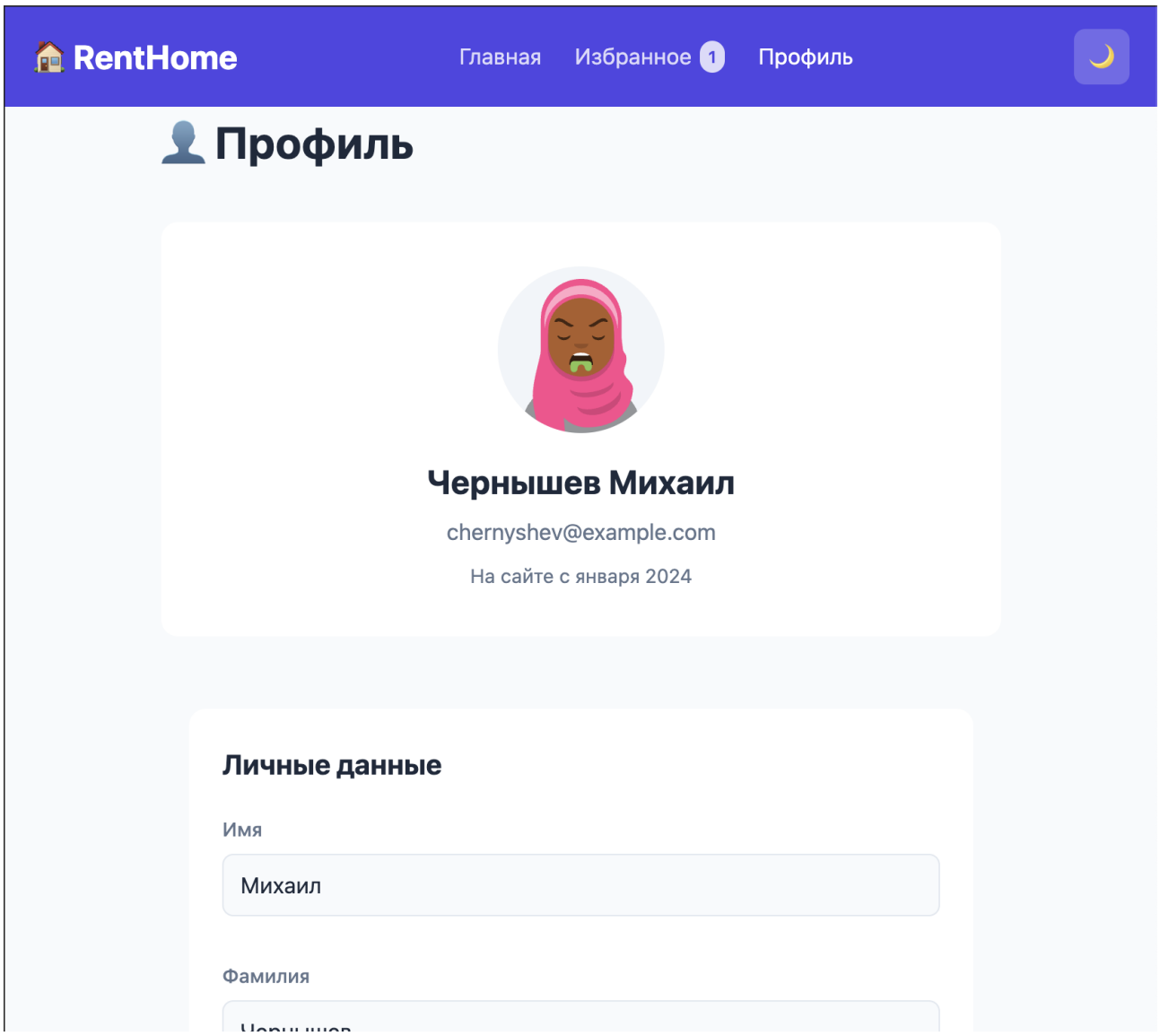

Рис. 4: Личный кабинет пользователя