САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №3
Разработка SPA с использованием Vue.js

Выполнил:
Чернышев Михаил Павлович
Группа К3441

Проверил:
Добряков Д. И.

Санкт-Петербург
2026 г.

Содержание

1 Задача **2**

2 Ход работы **2**

2.1 Структура проекта 2
2.2 Настройка Vue Router 2
2.3 Конфигурация Axios 4
2.4 Composable: useAuth 4
2.5 Composable: useProperties 6
2.6 Composable: useFavorites 7
2.7 Компонент: PropertyCard 8
2.8 Компонент: PropertyFilters 9
2.9 Страница: HomeView 11
2.10 Страница: LoginView 13
2.11 Компонент: AppHeader 15

3 Вывод **16**

1 Задача

Мигрировать ранее разработанное приложение сервиса аренды недвижимости на фреймворк Vue.js.

Требования:

- Подключение Vue Router для навигации
- Работа с внешним API через axios
- Разумное деление на компоненты
- Использование composables для переиспользуемой логики

2 Ход работы

2.1 Структура проекта

```
src/
 api/
    axios.js            # Конфигурация axios
    services/           # API сервисы
 components/
    common/             # Общие компоненты (Button, Input, Modal)
    property/           # Компоненты недвижимости
    layout/             # Header, Footer, Sidebar
 composables/
    useAuth.js          # Логика авторизации
    useProperties.js    # Работа с объектами
    useFavorites.js     # Избранное
 router/
    index.js            # Настройка роутера
 views/                  # Страницы приложения
 App.vue
 main.js
```

2.2 Настройка Vue Router

router/index.js

```
import { createRouter, createWebHistory } from ’vue-router’
import { useAuth } from ’@/composables/useAuth’

const routes = [
  { path: ’/’, name: ’Home’, component: () => import(’@/views/HomeView.vue’) },
  { path: ’/login’, name: ’Login’, component: () => import(’@/views/LoginView.vue’) }
  { path: ’/register’, name: ’Register’, component: () => import(’@/views/RegisterVie
  { path: ’/property/:id’, name: ’Property’, component: () => import(’@/views/Propert
  {
    path: ’/profile’,
    name: ’Profile’,
    component: () => import(’@/views/ProfileView.vue’),
    meta: { requiresAuth: true }
  },
  {
    path: ’/messages’,
    name: ’Messages’,
    component: () => import(’@/views/MessagesView.vue’),
    meta: { requiresAuth: true }
  }
]

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

// Защита роутов
router.beforeEach(async (to, from, next) => {
  const { isAuthenticated, checkAuth } = useAuth()

  if (to.meta.requiresAuth) {
    await checkAuth()
    if (!isAuthenticated.value) {
      next({ name: ’Login’, query: { redirect: to.fullPath } })
      return
    }
  }
  next()
})

export default router
```

2.3 Конфигурация Axios

api/axios.js

```
import axios from ’axios’
import router from ’@/router’

const api = axios.create({
  baseURL: ’http://localhost:3001’,
  timeout: 10000
})

api.interceptors.request.use(config => {
  const token = localStorage.getItem(’authToken’)
  if (token) {
    config.headers.Authorization = ‘Bearer ${token}‘
  }
  return config
})

api.interceptors.response.use(
  response => response.data,
  error => {
    if (error.response?.status === 401) {
      localStorage.removeItem(’authToken’)
      router.push(’/login’)
    }
    return Promise.reject(error)
  }
)

export default api
```

2.4 Composable: useAuth

composables/useAuth.js

```
import { ref, computed } from ’vue’
import api from ’@/api/axios’

const user = ref(null)
const loading = ref(false)

export function useAuth() {
  const isAuthenticated = computed(() => !!user.value)

  async function login(email, password) {
    loading.value = true
    try {
      const users = await api.get(‘/users?email=${email}‘)
      if (users.length === 0) throw new Error(’Пользователь не найден’)

      const foundUser = users[0]
      // Проверка пароля и создание токена...
      localStorage.setItem(’authToken’, ’token_’ + Date.now())
      localStorage.setItem(’userId’, foundUser.id)
      user.value = foundUser
      return foundUser
    } finally {
      loading.value = false
    }
  }

  async function logout() {
    localStorage.removeItem(’authToken’)
    localStorage.removeItem(’userId’)
    user.value = null
  }

  async function checkAuth() {
    const token = localStorage.getItem(’authToken’)
    const userId = localStorage.getItem(’userId’)
    if (!token || !userId) return null

    try {
      user.value = await api.get(‘/users/${userId}‘)
      return user.value
    } catch {
      logout()
      return null
    }
  }

  return { user, loading, isAuthenticated, login, logout, checkAuth }
}
```

2.5 Composable: useProperties

composables/useProperties.js

```
import { ref } from ’vue’
import api from ’@/api/axios’

export function useProperties() {
  const properties = ref([])
  const loading = ref(false)
  const error = ref(null)

  async function fetchProperties(filters = {}) {
    loading.value = true
    error.value = null

    try {
      const params = new URLSearchParams()
      params.append(’status’, ’available’)

      if (filters.type) params.append(’type’, filters.type)
      if (filters.rooms) params.append(’rooms’, filters.rooms)
      if (filters.sortBy) {
        params.append(’_sort’, filters.sortBy)
        params.append(’_order’, filters.order || ’asc’)
      }

      let result = await api.get(‘/properties?${params}‘)

      // Фильтрация по цене
      if (filters.priceFrom || filters.priceTo) {
        result = result.filter(p => {
          if (filters.priceFrom && p.price < filters.priceFrom) return false
          if (filters.priceTo && p.price > filters.priceTo) return false
          return true
        })
      }

      properties.value = result
    } catch (e) {
      error.value = e.message
    } finally {
      loading.value = false
    }
  }

  async function fetchPropertyById(id) {
    return api.get(‘/properties/${id}‘)
  }

  return { properties, loading, error, fetchProperties, fetchPropertyById }
}
```

2.6 Composable: useFavorites

composables/useFavorites.js

```
import { ref } from ’vue’
import api from ’@/api/axios’

export function useFavorites() {
  const favorites = ref(new Set())

  async function loadFavorites() {
    const userId = localStorage.getItem(’userId’)
    if (!userId) return

    const data = await api.get(‘/favorites?userId=${userId}‘)
    favorites.value = new Set(data.map(f => f.propertyId))
  }

  async function toggleFavorite(propertyId) {
    const userId = localStorage.getItem(’userId’)
    if (!userId) throw new Error(’Необходима авторизация’)

    if (favorites.value.has(propertyId)) {
      const existing = await api.get(
        ‘/favorites?userId=${userId}&propertyId=${propertyId}‘
      )
      await api.delete(‘/favorites/${existing[0].id}‘)
      favorites.value.delete(propertyId)
    } else {
      await api.post(’/favorites’, { userId: +userId, propertyId })
      favorites.value.add(propertyId)
    }
  }

  function isFavorite(propertyId) {
    return favorites.value.has(propertyId)
  }

  return { favorites, loadFavorites, toggleFavorite, isFavorite }
}
```

2.7 Компонент: PropertyCard

components/property/PropertyCard.vue

```
<template>
  <div class="card h-100 shadow-sm">
    <div class="position-relative">
      <img :src="property.images[0]" class="card-img-top" :alt="property.title">
      <button
        class="btn btn-light btn-sm position-absolute top-0 start-0 m-2"
        @click.prevent="handleFavorite"
      >
        <i :class="isFavorite(property.id) ? ’bi-heart-fill text-danger’ : ’bi-heart’
      </button>
    </div>

    <div class="card-body">
      <h5 class="card-title">{{ property.title }}</h5>
      <p class="text-muted small">
        <i class="bi bi-geo-alt"></i> {{ property.address.city }}
      </p>
    </div>

    <div class="card-footer bg-white">
      <div class="d-flex justify-content-between align-items-center">
        <strong class="text-primary">{{ formatPrice(property.price) }} руб/мес</stron
        <router-link :to="‘/property/${property.id}‘" class="btn btn-outline-primary
          Подробнее
        </router-link>
      </div>
    </div>
  </div>
</template>

<script setup>
import { useFavorites } from ’@/composables/useFavorites’

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

const { isFavorite, toggleFavorite } = useFavorites()

function handleFavorite() {
  toggleFavorite(props.property.id)
}

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

2.8 Компонент: PropertyFilters

components/property/PropertyFilters.vue

```
<template>
  <div class="card shadow-sm">
    <div class="card-header"><h5 class="mb-0">Фильтры</h5></div>
    <div class="card-body">
      <div class="mb-3">
        <label class="form-label">Тип жилья</label>
        <select v-model="filters.type" class="form-select">
          <option value="">Любой</option>
          <option value="apartment">Квартира</option>
          <option value="house">Дом</option>
          <option value="room">Комната</option>
        </select>
      </div>

      <div class="mb-3">
        <label class="form-label">Цена</label>
        <div class="row g-2">
          <div class="col-6">
            <input v-model.number="filters.priceFrom" type="number"
                   class="form-control" placeholder="От">
          </div>
          <div class="col-6">
            <input v-model.number="filters.priceTo" type="number"
                   class="form-control" placeholder="До">
          </div>
        </div>
      </div>

      <div class="mb-3">
        <label class="form-label">Комнат</label>
        <div class="btn-group w-100">
          <button v-for="r in [1, 2, 3]" :key="r"
                  :class="[’btn’, filters.rooms === r ? ’btn-primary’ : ’btn-outline-
                  @click="filters.rooms = filters.rooms === r ? null : r">
            {{ r }}{{ r === 3 ? ’+’ : ’’ }}
          </button>
        </div>
      </div>

      <button class="btn btn-primary w-100 mb-2" @click="$emit(’apply’, filters)">
        Применить
      </button>
      <button class="btn btn-outline-secondary w-100" @click="reset">
        Сбросить
      </button>
    </div>
  </div>
</template>

<script setup>
```

```
import { reactive } from ’vue’

const emit = defineEmits([’apply’])

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

function reset() {
  Object.assign(filters, { type: ’’, priceFrom: null, priceTo: null, rooms: null })
  emit(’apply’, filters)
}
</script>
```

2.9 Страница: HomeView

views/HomeView.vue

```
<template>
  <div class="container py-4">
    <div class="row">
      <div class="col-lg-3">
        <PropertyFilters @apply="handleFilters" />
      </div>

      <div class="col-lg-9">
        <div class="d-flex justify-content-between mb-4">
          <h4>Найдено: {{ properties.length }} объектов</h4>
          <select v-model="sortBy" class="form-select w-auto" @change="loadData">
            <option value="createdAt">По дате</option>
            <option value="price">По цене</option>
          </select>
        </div>

        <div v-if="loading" class="text-center py-5">
          <div class="spinner-border text-primary"></div>
        </div>

        <div v-else class="row">
          <div v-for="property in properties" :key="property.id" class="col-md-6 col-
            <PropertyCard :property="property" />
          </div>
        </div>

        <div v-if="!loading && properties.length === 0" class="text-center py-5 text-
          Объекты не найдены
        </div>
      </div>
    </div>
  </div>
</template>

<script setup>
import { ref, onMounted } from 'vue'
import { useProperties } from '@/composables/useProperties'
import { useFavorites } from '@/composables/useFavorites'
import PropertyCard from '@/components/property/PropertyCard.vue'
import PropertyFilters from '@/components/property/PropertyFilters.vue'

const { properties, loading, fetchProperties } = useProperties()
const { loadFavorites } = useFavorites()

const sortBy = ref('createdAt')
const currentFilters = ref({})

async function loadData() {
  await fetchProperties({ ...currentFilters.value, sortBy: sortBy.value })
}
```

```

function handleFilters(filters) {
  currentFilters.value = filters
  loadData()
}

onMounted(async () => {
  await loadFavorites()
  await loadData()
})
</script>
```

2.10 Страница: LoginView

views/LoginView.vue

```
<template>
  <div class="min-vh-100 d-flex align-items-center bg-light">
    <div class="container">
      <div class="row justify-content-center">
        <div class="col-md-5">
          <div class="card shadow-lg border-0">
            <div class="card-body p-5">
              <h2 class="text-center mb-4">Вход</h2>

              <div v-if="error" class="alert alert-danger">{{ error }}</div>

              <form @submit.prevent="handleSubmit">
                <div class="mb-3">
                  <label class="form-label">Email</label>
                  <input v-model="form.email" type="email" class="form-control" requi
                </div>

                <div class="mb-3">
                  <label class="form-label">Пароль</label>
                  <input v-model="form.password" type="password" class="form-control"
                </div>

                <button type="submit" class="btn btn-primary w-100" :disabled="loadin
                  <span v-if="loading" class="spinner-border spinner-border-sm me-2">
                  Войти
                </button>
              </form>

              <p class="text-center mt-4 mb-0">
                Нет аккаунта? <router-link to="/register">Регистрация</router-link>
              </p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</template>

<script setup>
import { reactive, ref } from ’vue’
import { useRouter, useRoute } from ’vue-router’
import { useAuth } from ’@/composables/useAuth’

const router = useRouter()
const route = useRoute()
const { login, loading } = useAuth()

const form = reactive({ email: ’’, password: ’’ })
const error = ref(null)
```

```
async function handleSubmit() {
  error.value = null
  try {
    await login(form.email, form.password)
    router.push(route.query.redirect || ’/profile’)
  } catch (e) {
    error.value = e.message
  }
}
</script>
```

2.11 Компонент: AppHeader

components/layout/AppHeader.vue

```
<template>
  <nav class="navbar navbar-expand-lg navbar-dark bg-primary sticky-top">
    <div class="container">
      <router-link class="navbar-brand fw-bold" to="/">
        <i class="bi bi-house-door-fill me-2"></i>RentHome
      </router-link>

      <div class="navbar-nav me-auto">
        <router-link class="nav-link" to="/">Поиск</router-link>
        <router-link v-if="isAuthenticated" class="nav-link" to="/profile">
          Кабинет
        </router-link>
        <router-link v-if="isAuthenticated" class="nav-link" to="/messages">
          Сообщения
          <span v-if="unreadCount" class="badge bg-danger">{{ unreadCount }}</span>
        </router-link>
      </div>

      <div class="d-flex gap-2">
        <template v-if="isAuthenticated">
          <span class="text-white me-2">{{ user?.firstName }}</span>
          <button class="btn btn-outline-light" @click="handleLogout">Выйти</button>
        </template>
        <template v-else>
          <router-link to="/login" class="btn btn-outline-light">Вход</router-link>
          <router-link to="/register" class="btn btn-light">Регистрация</router-link>
        </template>
      </div>
    </div>
  </nav>
</template>

<script setup>
import { ref } from 'vue'
import { useRouter } from 'vue-router'
import { useAuth } from '@/composables/useAuth'

const router = useRouter()
const { user, isAuthenticated, logout } = useAuth()
const unreadCount = ref(0)

async function handleLogout() {
  await logout()
  router.push('/')
}
</script>
```

3 Вывод

В ходе выполнения лабораторной работы приложение сервиса аренды недвижимости было успешно мигрировано на фреймворк Vue.js.

Реализованные требования:

- **Vue Router** — настроена навигация с защитой роутов для авторизованных пользователей, lazy-loading компонентов
- **Axios** — настроены интерцепторы для автоматического добавления токена и обработки ошибок авторизации
- **Компонентный подход** — приложение разделено на переиспользуемые компоненты: PropertyCard, PropertyFilters, AppHeader и др.
- **Composables** — выделена переиспользуемая логика: useAuth, useProperties, useFavorites

Структура компонентов:

- `components/common/` — общие UI-компоненты
- `components/property/` — компоненты для работы с недвижимостью
- `components/layout/` — компоненты разметки (Header, Footer)
- `views/` — страницы приложения

Преимущества миграции на Vue.js:

- Реактивность данных без ручного обновления DOM
- Чёткое разделение логики и представления
- Переиспользование кода через composables
- Удобная маршрутизация с защитой роутов
- Лучшая организация и поддерживаемость кода