

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Фронт-энд разработка

Отчет

Лабораторная работа №3

Выполнила:

Красюк Карина

К3441

Проверил:

Добряков Д. И.

Санкт-Петербург

2026 г.

СОДЕРЖАНИЕ

Задача.....	3
Ход работы.....	3
1 Краткое резюме выполненного	3
2 Список реализованных функций с примерами кода	3
2.1 Vue Router с защищёнными маршрутами.....	3
2.2 Работа с API через Axios	5
3 Демонстрация проделанной работы в виде визуализаций	6
Вывод.....	7

Задача

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 повторяющегося функционала в отдельные файлы

Вариант “Сайт для поиска работы”.

Ход работы

1 Краткое резюме выполненного

Выполнена миграция приложения JobFinder с нативного JavaScript на Vue.js 3. Реализованы:

- Vue Router для навигации с защищёнными маршрутами
- Axios для работы с REST API
- Компонентная архитектура (12 компонентов)
- Composable функции для переиспользуемой логики (7 composables)
- Система уведомлений
- Управление темой (светлая/тёмная)

2 Список реализованных функций с примерами кода

2.1 Vue Router с защищёнными маршрутами

Реализован роутинг с проверкой авторизации и ролей.

Файл: src/router/index.js

```
router.beforeEach((to, from, next) => {
```

```

    const { isAuthenticated, isCandidate, isEmployer } =
    useAuth()

    // Если маршрут требует авторизации
    if (to.meta.requiresAuth && !isAuthenticated.value) {
        next({ name: 'Login', query: { redirect: to.fullPath } })
        return
    }

    // Если маршрут требует роль соискателя
    if (to.meta.requiresRole === 'candidate' &&
    !isCandidate.value) {
        next({ name: 'Home' })
        return
    }

    // Если маршрут требует роль работодателя
    if (to.meta.requiresRole === 'employer' &&
    !isEmployer.value) {
        next({ name: 'Home' })
        return
    }

    next()
})

```

Маршруты:

/ — главная страница (публичная)
 /login — вход (только для гостей)
 /register — регистрация (только для гостей)
 /jobs/:id — детали вакансии (публичная)
 /dashboard/user — личный кабинет соискателя (требуется авторизации и роль candidate)

/dashboard/employer — личный кабинет работодателя (требуется авторизации и роль employer)

2.2 Работа с API через Axios

Создан централизованный API-клиент с интерцепторами.

Файл: src/composables/useApi.js

```
import axios from 'axios'

const API_BASE_URL = 'http://localhost:3000'

const apiClient = axios.create({
  baseURL: API_BASE_URL,
  headers: {
    'Content-Type': 'application/json'
  }
})

// Интерцептор для добавления токена авторизации
apiClient.interceptors.request.use((config) => {
  const token = localStorage.getItem('token')
  if (token) {
    config.headers.Authorization = `Bearer ${token}`
  }
  return config
})

// Интерцептор для обработки ошибок
apiClient.interceptors.response.use(
  (response) => response,
  (

```

Использование в компонентах:

```
import { useApi } from '@composables/useApi'

const { apiClient } = useApi()
```

```
const response = await apiClient.get('/jobs')
```

3 Демонстрация проделанной работы в виде визуализаций

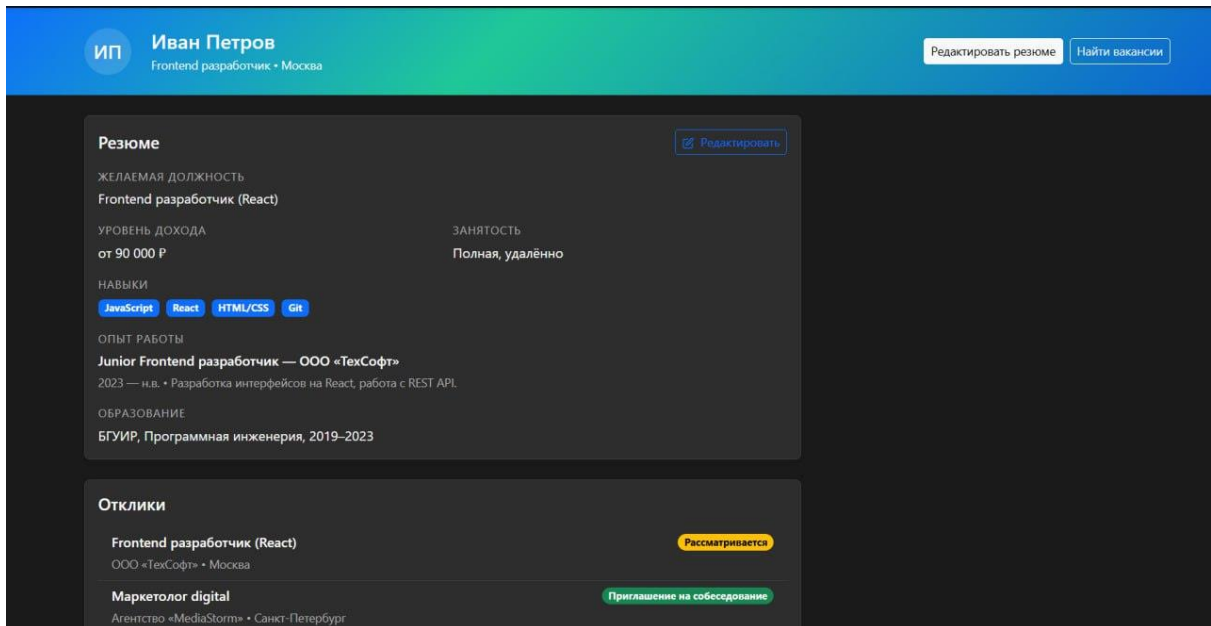


Рисунок 1 – Пример личного кабинета соискателя на vue

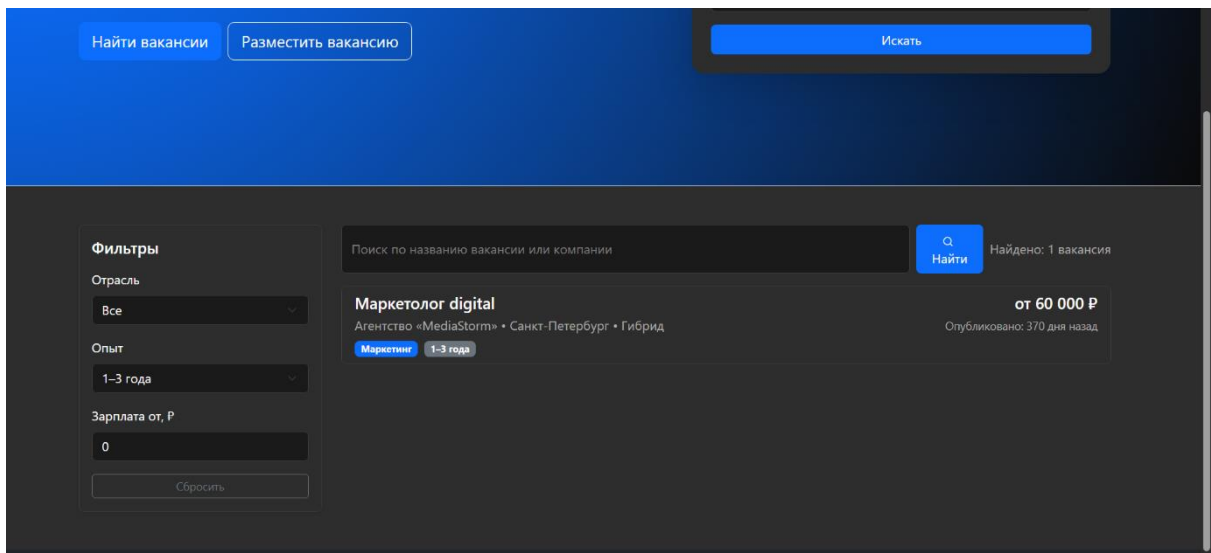


Рисунок 2 – Пример главной страницы с фильтрами для вакансий на vue

Вывод

Миграция завершена. Приложение соответствует требованиям Vue.js, использует современные практики и готово к дальнейшему развитию. Архитектура обеспечивает масштабируемость, поддерживаемость и производительность.