САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Фронт-энд разработка

Отчет

Домашняя работа №5

Выполнила:

Красюк Карина

K3441

Проверил:
Добряков Д. И.

Санкт-Петербург

2026 г.

СОДЕРЖАНИЕ

Задача

Вынести все используемые ранее SVG-иконки в общий SVG-спрайт. Если иконок не было, добавьте 3-5 иконок и поместите их в SVG-спрайт.

Вариант “Сайт для поиска работы”.

Ход работы

В рамках лабораторной работы создано одностраничное приложение на Vue.js 3 для управления заметками.

Реализованы:

- Инициализация проекта с использованием Vite и настройка структуры
- Интеграция зависимостей: Vue Router, Pinia, Axios, Bootstrap, pinia-persists
- Архитектура API с применением Dependency Injection
- Управление состоянием через Pinia с персистентностью в localStorage
- Компонентная архитектура: компоненты, представления и лейауты
- Роутинг с ленивой загрузкой компонентов
- Работа с формами через v-model и обработка событий
- Интеграция Bootstrap для стилизации

Приложение позволяет создавать заметки через форму и отображать их в виде карточек, получая данные с backend API.

1 Список реализованных функций с примерами кода

1.1 Инициализация проекта и настройка зависимостей

Файл: package.json

Настроены скрипты для работы с проектом:

- npm start — запуск на порту 8080
- npm run dev — режим разработки
- npm run build — сборка для продакшена

- npm run lint — проверка кода

Установлены зависимости:

- vue, vue-router, pinia — базовые библиотеки Vue
- axios — HTTP-клиент
- bootstrap — CSS-фреймворк
- pinia-persists — персистентность состояния

1.2 Конфигурация API с использованием Dependency Injection

Файл: src/api/instance.js

Создан единый экземпляр Axios для всего приложения:

```
import axios from 'axios'

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

const instance = axios.create({
  baseURL: apiURL
})

export default instance
```

Файл: src/api/notes.js

Реализован класс для работы с API заметок:

```
class NotesApi {
  constructor(instance) {
    this.API = instance
  }

  getAll = async () => {
    return this.API({
      url: '/notes'
    })
  }

  createNote = async (data) => {
```

```
    return this.API({
      method: 'POST',
      url: '/notes',
      data,
      headers: {
        'Content-Type': 'application/json'
      }
    })
  }
}

export default NotesApi
```

Файл: src/api/index.js

Реализован IoC-контейнер для централизованного экспорта API:

```
import instance from "@/api/instance"
import NotesApi from "@/api/notes"

const notesApi = new NotesApi(instance)

export {
  notesApi
}
```

1.3 Управление состоянием через Pinia с персистентностью

Файл: src/stores/index.js

Настроен Pinia с плагином персистентности:

```
import { persist } from 'pinia-persists'
import { createPinia } from 'pinia'

const pinia = createPinia()

pinia.use(persist())

export default pinia
```

Файл: src/stores/notes.js

Создано хранилище для заметок с методами загрузки и создания:

```
import { defineStore } from 'pinia'
import { notesApi } from '@/api'

const useNotesStore = defineStore('notes', {
  state: () => ({
    notes: []
  }),

  actions: {
    async loadNotes() {
      const response = await notesApi.getAll()
      this.notes = response.data
      return response
    },

    async createNote(data) {
      const response = await notesApi.createNote(data)
      this.notes = response.data
      return response
    }
  }
})

export default useNotesStore
```

1.4 Настройка роутинга с ленивой загрузкой

Файл: src/router/index.js

Настроен Vue Router с ленивой загрузкой компонентов:

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

const router = createRouter({
```

```
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'notes',
      component: () => import('../views/NotesPage.vue')
    }
  ]
})
export default router
```

2 Визуализация проделанной работы

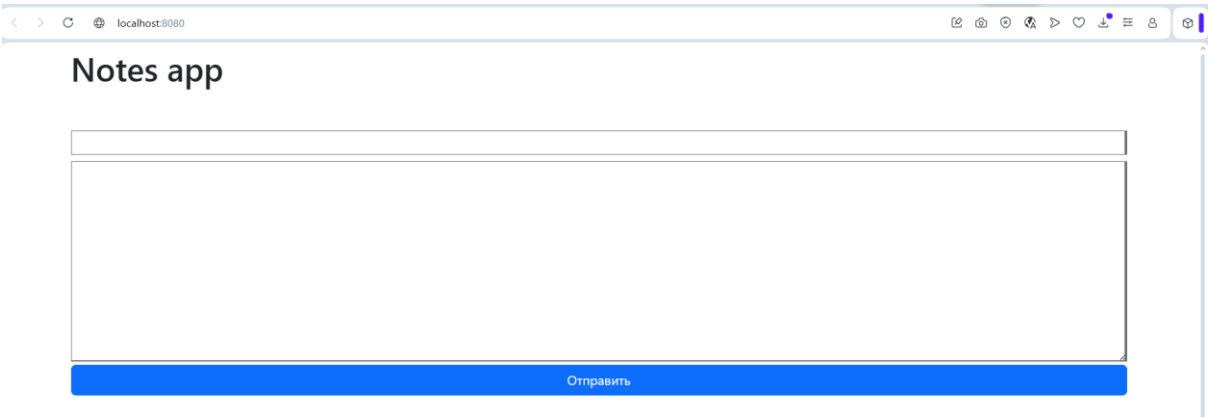

Рисунок 1 – Главная страница приложения (пустое состояние)

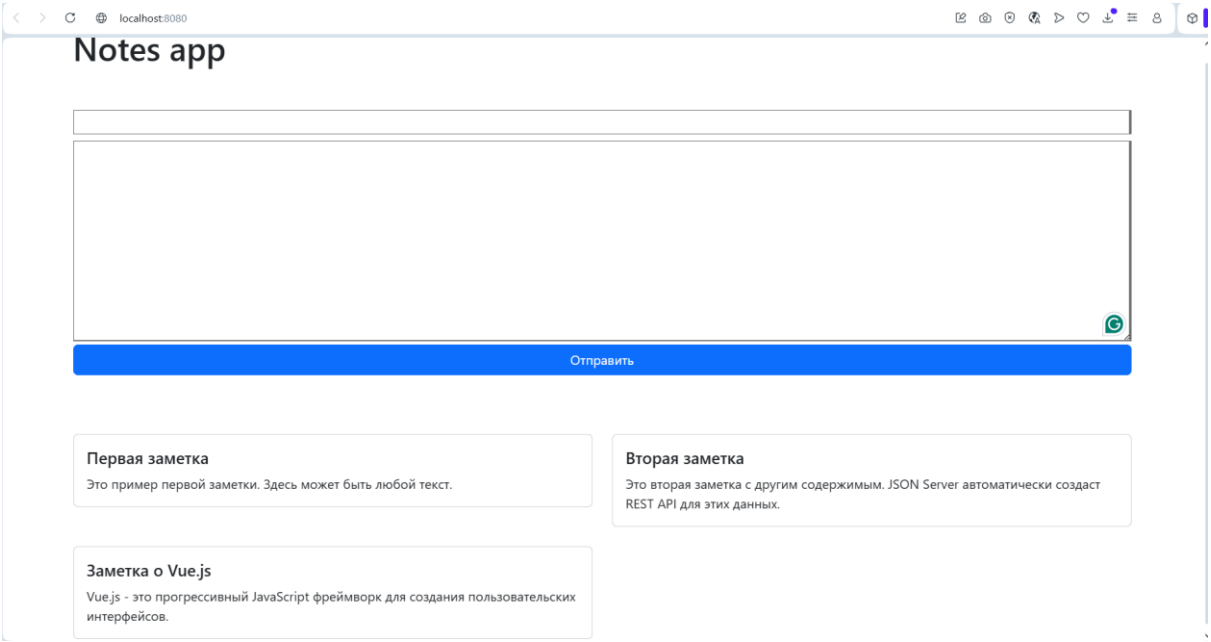

Рисунок 2 – Главная страница с загруженными заметками

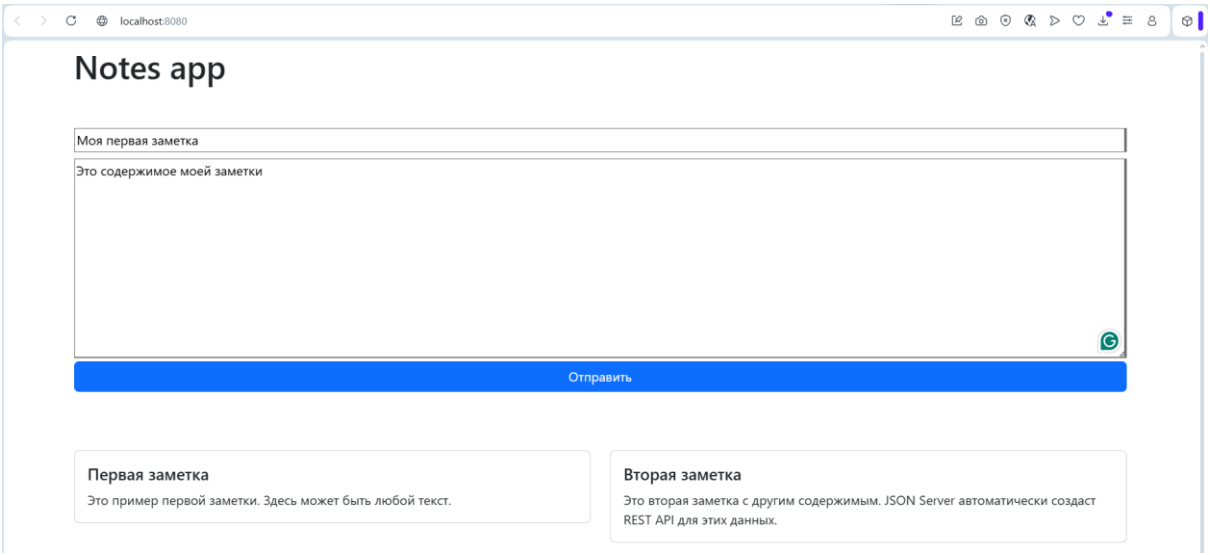

Рисунок 3 – Процесс создания новой заметки (форма заполнена)

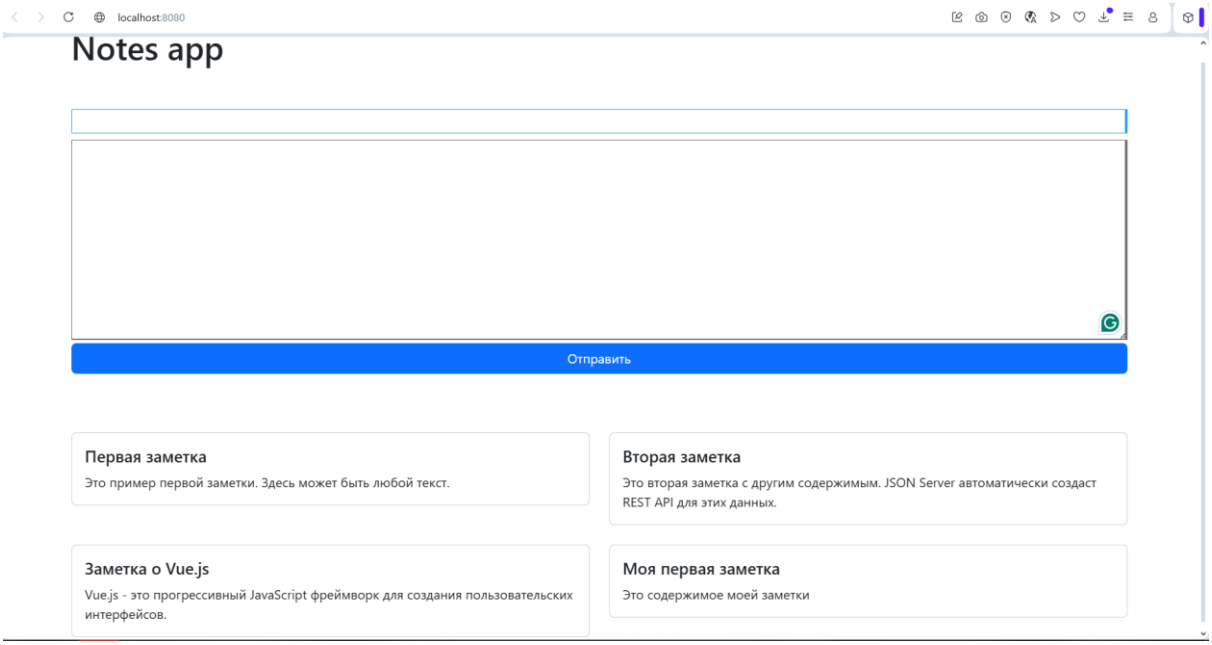

Рисунок 4 – Результат после создания заметки

Вывод

В ходе лабораторной работы изучены и применены:

- Работа с NPM: установка зависимостей, настройка скриптов, управление пакетами
- Vue.js 3: компоненты, props, директивы (v-model, v-for), хуки жизненного цикла, computed properties
- Vue Router: настройка маршрутов, ленивая загрузка компонентов
- Pinia: управление состоянием, actions, интеграция через mapState и mapActions, персистентность
- Axios: создание инстанса, работа с API, обработка запросов
- Архитектурные паттерны: Dependency Injection, IoC-контейнер, разделение на компоненты/представления/лейауты
- Bootstrap: использование классов для стилизации и сетки

Результат: рабочее одностраничное приложение для управления заметками с возможностью создания и отображения. Приложение следует принципам модульности, переиспользования компонентов и разделения ответственности.

Применённые технологии и подходы позволяют масштабировать приложение и добавлять новый функционал без усложнения кодовой базы.