|  |
| --- |
| Финансово-промышленный университет «Синергия»  Факультет дополнительного профессионального образования |
|  |
|  |
|  |
|  |
|  |
|  |
| Леонтьев Б. П. |
|  |
| **Программа дополнительного профессионального образования «Верстка CSS и программирование на языке JavaScript»** |
| Отчет о выполнении задания по теме 6 «Фреймворк Vue для языка JavaScript» |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| Москва |
| 2024 |

Оглавление

[Задание 3](#_Toc179980930)

[1 Выполнение задания 3](#_Toc179980931)

[2 Код index.html 4](#_Toc179980932)

[3 Код /src/main.js 4](#_Toc179980933)

[4 Код ./src/script.js 4](#_Toc179980934)

[5 Код ./src/assets/style.css 4](#_Toc179980935)

[6 Код ./src/App\_1.vue 5](#_Toc179980936)

[7 Код ./src/shablon\_dlya\_elementa\_spiska\_1.vue 13](#_Toc179980937)

[8 Реализация функциональных требований 13](#_Toc179980938)

[9 Просмотр списка задач 13](#_Toc179980939)

[9.1 Сортировка списка задач 14](#_Toc179980940)

[9.2 Добавление задач 15](#_Toc179980941)

[9.3 Удаление задач 15](#_Toc179980942)

[9.4 Фильтрация задач 16](#_Toc179980943)

[9.5 Снятие фильтра 17](#_Toc179980944)

Задание

Изучить основы Vue.js и создать простое приложение для составления списка дел.

1. Создать новый проект Vue.js.
2. Подключиться и получить данные с помощью открытого API к https://jsonplaceholder.typicode.com/ (или к любому свободному API).
3. Создать новый компонент для отображения списка дел.
4. Использовать директиву v-for для вывода списка дел из массива.
5. Добавить компонент формы для добавления новых пунктов в список дел.
6. С помощью директивы v-model привязать поле ввода к свойству data.
7. С помощью директивы v-on обработать событие отправки формы и добавить новый элемент в список.
8. Добавить кнопку для удаления элементов из списка.
9. С помощью директивы v-on обработать событие нажатия на кнопку и удалить элемент из списка.
10. Добавить функцию сортировки и фильтрации списка.
11. Оформить приложение с помощью CSS, чтобы сделать его визуально привлекательным.
12. Реализовать приложение с помощью чистого и читабельного кода на JavaScript.
13. Тщательно протестировать приложение, чтобы убедиться, что оно работает так, как ожидалось.
14. Результат работы выложить в виде ссылки на git.

Выполнение задания

Рисунок 1 показывает отладку и работу приложения в среде Visual Studio Code.

Проект включает следующие файлы.

1. index.html.
2. module /src/main.js – монтирует модуль.
3. text/javascript" ./src/script.js – содержит обычный JavaScript-код.
4. stylesheet ./src/assets/style.css – содержит определения стилей.
5. ./src/App\_1.vue – главный модуль приложения.
6. ./src/shablon\_dlya\_elementa\_spiska\_1.vue

HTML-, CSS-, JavaScript-коды приведены ниже в соответствующих разделах.

Материалы по выполнению задания помещены в хранилище, доступное по ссылке <https://github.com/bleontyev/calculator>.

Рисунок 2 показывает ветку main в удаленном репозитории.

Код index.html

<!--

Vue.js. Задание.

-->

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script type="module" src="/src/main.js"></script>

<script defer type="text/javascript" src="./src/script.js"></script>

<link rel="stylesheet" href="./src/assets/style.css">

<title>Vite App - Vue - project-5</title>

</head>

<body>

<div id="app\_1"></div>

<script>

</script>

<div id="app"></div>

</body>

</html>

Код /src/main.js

import './assets/main.css'

import { createApp } from 'vue'

import app\_xxx from './App\_1.vue'

createApp( app\_xxx ).mount('#app\_1');

Код ./src/script.js

let vneshnyaya\_peremennaya = "Xxxxxxx";

Код ./src/assets/style.css

.kornevoj-kontejner{

width: 1015px;

height: 1010px;

border: 1px solid black;

border-radius: 5px;

}

.sticky-forma{

position: sticky;

border: 1px solid blue;

/\* Это чтобы панель приобрела свойство sticky. \*/

top: 20px;

bottom: 10px;

/\* \*/

margin-left: 10px;

margin-top: 10px;

/\* \*/

padding-left: 5px;

padding-bottom: 5px;

padding-right: 5px;

color: black;

border-radius: 5px;

width: 240px;

float: left;

}

.app-1{

position: relative;

margin-left: 10px;

margin-top: 400px;

width: 800px;

}

.blok-del{

color: black;

border: 1px solid black;

border-radius: 5px;

height: 985px;

overflow: scroll;

width: 730px;

float: left;

margin-left: 10px;

margin-top: 10px;

margin-bottom: 10px;

}

.knopka{

margin-top: 5px;

}

Код ./src/App\_1.vue

<!--

Это компонент для списка дел.

-->

<script>

import shablon\_dlya\_elementa\_spiska\_1 from './shablon\_dlya\_elementa\_spiska\_1.vue'

let peremennaya\_app\_1\_vue = 1234567;

let massiv\_vne\_export = [1, 2, 3];

export default {

props: ['prop\_1'],

components: {

shablon\_dlya\_elementa\_spiska\_1,

},

data() {

return {

massiv\_zadach\_seansa: [],

massiv\_zadach\_seansa\_sorted: [],

massiv\_zadach\_seansa\_filtered: [],

massiv\_zadach\_istochnik: [],

massiv\_id\_zadach: [],

massiv\_id\_zadach\_sorted: [],

massiv\_userId: [],

massiv\_title: [],

massiv\_completed: [],

tek\_id\_sorted: 0,

tek\_userId\_sorted: 0,

tek\_title\_sorted: "",

tek\_completed\_sorted: false,

tek\_id\_filtered: 0,

tek\_userId\_filtered: 0,

tek\_title\_filtered: "",

tek\_completed\_filtered: false,

v\_sostoyanii\_filtratsii: false,

index\_dlya\_udaleniya: -1,

massiv\_zadach\_seansa\_rab: [],

massiv\_zadach: [],

dlina\_massiva\_zadach: 0,

peremennaya\_vnutri: 0,

// Добавляем переменные-маркеры состояния, отслеживающие выполнение блоков кода.

sost\_zadacha\_dobavlena: false,

sost\_zadacha\_udalena: false,

sost\_zadachi\_otsortirovany: false,

sost\_zadachi\_otfiltrovany: false,

sost\_filtr\_snyat: false,

// Массив своих опеределенных локально задач.

massiv\_svoikh\_zadach: [

{

userId: 1,

id: 1,

title: "Протестировать подключение к Vue.js через CDN.",

completed: true,

},

{

userId: 1,

id: 2,

title: "Протестировать подключение к Vue.js локально через ./vue/vue.global.js.",

completed: true,

},

{

userId: 1,

id: 3,

title: "Исследовать работу директивы v-model.",

completed: true,

},

{

userId: 1,

id: 4,

title: "Исследовать работу директивы v-if.",

completed: true,

},

],

formulirovka\_tekushchej\_zadachi: "",

tek\_id: 0,

tek\_userId: 0,

tek\_title: "",

tek\_completed: false,

}

},

methods:{

test(){

this.list.push(this.name);

},

sort(){

this.list.sort();

},

increaseCounter(){

this.counter++ ;

},

decreaseCounter(){

this.counter-- ;

},

dobavit\_zadachu(){

if( this.tek\_title.trim() == "") {

return ;

}

// Предействия.

// Вычислить следующий по порядку id для задачи.

this.massiv\_id\_zadach = this.massiv\_id\_zadach.splice( 0, 0);

for( let i = 0; i<= this.massiv\_zadach\_istochnik.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_istochnik[i].id;

}

console.log( "this.massiv\_id\_zadach", this.massiv\_id\_zadach );

this.massiv\_id\_zadach\_sorted = this.massiv\_id\_zadach.toSorted( (a, b) => a - b );

console.log( "this.massiv\_id\_zadach\_sorted", this.massiv\_id\_zadach\_sorted );

// +

this.dlina\_massiva\_zadach = this.massiv\_id\_zadach\_sorted.length;

this.tek\_id = this.massiv\_id\_zadach\_sorted[this.dlina\_massiva\_zadach - 1] + 1;

console.log( "this.tek\_id", this.tek\_id );

// Теперь можно вдвигать добавляемый элемент в массив сеанса и в массив-источник.

// this.massiv\_zadach\_seansa.unshift({

// id: this.tek\_id,

// userId: 1,

// title: this.tek\_title,

// completed: true,

// });

this.massiv\_zadach\_istochnik.unshift({

id: this.tek\_id,

userId: 1,

title: this.tek\_title,

completed: false,

});

// this.massiv\_id\_zadach.unshift(

// this.tek\_id,

// );

// Последействия.

this.tek\_title = "" ;

this.tek\_id = 0 ;

fetch('https://jsonplaceholder.typicode.com/todos', {

method: 'POST',

body: JSON.stringify({

id: this.tek\_id,

userId: this.tek\_userId,

title: this.tek\_title,

completed: this.tek\_completed,

}),

headers: {

'Content-type': 'application/json; charset=UTF-8',

},

})

.then((response) => response.json())

.then((json) => console.log(json))

.catch( (error) => console.log(error))

this.massiv\_zadach\_seansa = this.massiv\_zadach\_istochnik;

this.sost\_zadacha\_dobavlena = true;

this.sost\_zadachi\_otsortirovany = false;

},

//

udalit\_zadachu(){

// Преддействия - отсутствуют.

if( this.tek\_id == 0) {

return ;

}

// Удаляем задачу из массива текущих задач.

for( let i = 0; i<= this.massiv\_zadach\_seansa.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_seansa[i].id;

}

this.index\_dlya\_udaleniya = this.massiv\_id\_zadach.indexOf( this.tek\_id );

this.massiv\_zadach\_seansa.splice( this.index\_dlya\_udaleniya, 1 ) ;

this.massiv\_id\_zadach.splice( this.index\_dlya\_udaleniya, 1 ) ;

this.tek\_id = 0;

// Удаляем задачу из массива-источника.

for( let i = 0; i<= this.massiv\_zadach\_istochnik.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_istochnik[i].id;

}

this.index\_dlya\_udaleniya = this.massiv\_id\_zadach.indexOf( this.tek\_id );

this.massiv\_zadach\_istochnik.splice( this.index\_dlya\_udaleniya, 1 ) ;

this.massiv\_id\_zadach.splice( this.index\_dlya\_udaleniya, 1 ) ;

this.tek\_id = 0;

// Удаляем задачу в удаленном Web API сервисе.

fetch('https://jsonplaceholder.typicode.com/todos/1', {

method: 'DELETE',

})

.then((response) => response.json())

.then((json) => console.log(json))

.catch( (error) => console.log(error));

// Последействия.

this.sost\_zadacha\_udalena = true;

this.sost\_zadachi\_otsortirovany = false;

this.sost\_zadachi\_otfiltrovany = false;

},

//

otsortirovat\_zadachi(){

if( !this.sost\_zadachi\_otsortirovany ){

console.log( "otsortirovat\_zadachi" );

// Преддествия.

// Очистка массива \_sorted.

this.massiv\_zadach\_seansa\_sorted = this.massiv\_zadach\_seansa\_sorted.splice( 0, 0);

// Из массива задач сеанса создаем отдельные массивы с той же поледовательностью элементов.

for( let i = 0; i<= this.massiv\_zadach\_seansa.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_seansa[i].id;

this.massiv\_userId[i] = this.massiv\_zadach\_seansa[i].userId;

this.massiv\_title[i] = this.massiv\_zadach\_seansa[i].title;

this.massiv\_completed[i] = this.massiv\_zadach\_seansa[i].completed;

}

// Сортировка массива id задач с сипользованием функции сортировки. Ниже получается восходящий порядок.

this.massiv\_id\_zadach\_sorted = this.massiv\_id\_zadach.toSorted( (a, b) => b - a );

// Для нисходящего порядка ниадо как ниже.

// this.massiv\_id\_zadach\_sorted = this.massiv\_id\_zadach.toSorted( (a, b) => b - a );

for( let i = 0; i<= this.massiv\_id\_zadach\_sorted.length - 1; i++ ) {

this.indeks\_poiska = this.massiv\_id\_zadach\_sorted[i];

this.indeks\_v\_massive\_zadach = this.massiv\_id\_zadach.indexOf(this.indeks\_poiska);

// Получаем объект, пригодный для того чтобы вставлять в отсортированный массив.

this.tek\_id\_sorted = this.indeks\_poiska;

this.tek\_iserId\_sorted = this.massiv\_userId[this.indeks\_v\_massive\_zadach];

this.tek\_title\_sorted = this.massiv\_title[this.indeks\_v\_massive\_zadach];

this.tek\_completed\_sorted = this.massiv\_completed[this.indeks\_v\_massive\_zadach];

// Вдвигаем объект в массив \_sorted.

this.massiv\_zadach\_seansa\_sorted.unshift({

id: this.tek\_id\_sorted,

userId: this.tek\_iserId\_sorted,

title: this.tek\_title\_sorted,

completed: this.tek\_completed\_sorted,

});

}

console.log("this.massiv\_zadach\_seansa\_sorted", this.massiv\_zadach\_seansa\_sorted );

// Источниковый массив не изменился, изменился только массив задач сеанса.

this.massiv\_zadach\_seansa = this.massiv\_zadach\_seansa\_sorted;

this.sost\_zadachi\_otsortirovany = true;

}

},

//

otfiltrovat\_zadachi(){

console.log( "otfiltrovat\_zadachi" );

this.massiv\_zadach\_seansa = this.massiv\_zadach\_istochnik;

this.massiv\_id\_zadach = this.massiv\_id\_zadach.splice( 0, 0 );

this.massiv\_zadach\_seansa\_filtered = this.massiv\_zadach\_seansa\_filtered.splice( 0, 0 );

console.log( this.massiv\_zadach\_istochnik );

for( let i = 0; i<= this.massiv\_zadach\_istochnik.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_istochnik[i].id;

this.massiv\_userId[i] = this.massiv\_zadach\_istochnik[i].userId;

this.massiv\_title[i] = this.massiv\_zadach\_istochnik[i].title;

this.massiv\_completed[i] = this.massiv\_zadach\_istochnik[i].completed;

}

for( let i = 0; i<= this.massiv\_id\_zadach.length - 1; i++ ) {

this.indeks\_poiska = this.massiv\_id\_zadach[i];

if( this.indeks\_poiska >= 5 && this.indeks\_poiska <= 21 ){

this.indeks\_v\_massive\_zadach = this.massiv\_id\_zadach.indexOf(this.indeks\_poiska);

this.tek\_id\_filtered = this.indeks\_poiska;

this.tek\_iserId\_filtered = this.massiv\_userId[this.indeks\_v\_massive\_zadach];

this.tek\_title\_filtered = this.massiv\_title[this.indeks\_v\_massive\_zadach];

this.tek\_completed\_filtered = this.massiv\_completed[this.indeks\_v\_massive\_zadach];

this.massiv\_zadach\_seansa\_filtered.unshift({

id: this.tek\_id\_filtered,

userId: this.tek\_iserId\_filtered,

title: this.tek\_title\_filtered,

completed: this.tek\_completed\_filtered,

});

}

}

console.log("this.massiv\_zadach\_seansa\_filtered", this.massiv\_zadach\_seansa\_filtered );

this.massiv\_zadach\_seansa = this.massiv\_zadach\_seansa\_filtered;

this.v\_sostoyanii\_filtratsii = true;

// Последействия.

this.sost\_zadachi\_otfiltrovany = true;

},

//

snyat\_filtr(){

if( this.sost\_zadachi\_otfiltrovany ){

console.log( "snyat\_filtr" );

this.massiv\_zadach\_seansa = this.massiv\_zadach\_istochnik;

}

this.sost\_zadachi\_otfiltrovany = false;

}

},

//

mounted() {

fetch('https://jsonplaceholder.typicode.com/todos/')

.then(response => response.json())

.then( (json) => {

this.massiv\_zadach\_seansa = json;

console.log( "vneshnyaya\_peremennaya", vneshnyaya\_peremennaya );

for( let i = 0; i<= this.massiv\_zadach\_seansa.length - 1; i++ ) {

this.massiv\_id\_zadach[i] = this.massiv\_zadach\_seansa[i].id;

}

this.massiv\_id\_zadach\_sorted = this.massiv\_id\_zadach.toSorted( (a, b) => a - b );

this.dlina\_massiva\_zadach = this.massiv\_id\_zadach\_sorted.length;

//

// \* Цикл добавления задач из локального массива.

//

this.tek\_id = this.massiv\_id\_zadach\_sorted[this.dlina\_massiva\_zadach - 1] + 1;

for( let i = 0; i <= this.massiv\_svoikh\_zadach.length - 1; i++ ){

this.tek\_userId = this.massiv\_svoikh\_zadach[i].userId;

this.tek\_title = this.massiv\_svoikh\_zadach[i].title;

this.tek\_completed = this.massiv\_svoikh\_zadach[i].completed;

this.massiv\_zadach\_seansa.unshift({ id: this.tek\_id, userId: this.tek\_userId, title: this.tek\_title, completed: this.tek\_completed });

this.massiv\_id\_zadach.unshift( this.tek\_id );

this.tek\_id = this.tek\_id + 1;

}

this.massiv\_zadach\_istochnik = this.massiv\_zadach\_seansa;

// \*

this.tek\_id = 0;

this.tek\_userId = 0;

this.tek\_title = "" ;

this.tek\_completed = false;

})

.catch( (error) => console.log(error)) ;

for( let i = 0; i<= this.massiv\_zadach\_seansa.length - 1; i++ ) {

massiv\_vne\_export[i] = this.massiv\_zadach\_seansa[i];

}

},

};

</script>

<template>

<div class="kornevoj-kontejner">

<div class="blok-del">

<div>

<p>Список дел.</p>

<ul>

<shablon\_dlya\_elementa\_spiska\_1

v-for="item in this.massiv\_zadach\_seansa" :todo="item" :key="item.id"

>

</shablon\_dlya\_elementa\_spiska\_1>

</ul>

</div>

</div>

<!-- block\_del -->

<div class="sticky-forma">

<br>

<label for="tek\_id">id:</label>

<input type="number" v-model="tek\_id">

<br>

<label for="tek\_userId">userId:</label>

<input class="knopka" type="number" v-model="tek\_userId">

<br>

<label for="tek\_title">Имя задачи:</label>

<input class="knopka" type="text" v-model="tek\_title" style="width: 230px;">

<br>

<label for="tek\_completed">Завершено:</label>

<input class="knopka" type="checkbox" v-model="tek\_completed" value="false">

<br>

<button class="knopka" v-on:click="dobavit\_zadachu">Добавить задачу</button>

<br>

<button class="knopka" v-on:click="udalit\_zadachu">Удалить задачу</button>

<br>

<button class="knopka" v-on:click="otsortirovat\_zadachi">Отсортировать задачи в восходящем порядке по id</button>

<br>

<button class="knopka" v-on:click="otfiltrovat\_zadachi">Отфильтровать задачи id >= 5 AND id <= 21 </button>

<br>

<button class="knopka" v-on:click="snyat\_filtr">Снять фильтр</button>

</div>

</div>

<!-- kornevoj-kontejner -->

</template>

Код ./src/shablon\_dlya\_elementa\_spiska\_1.vue

<script>

export default {

props: {

todo: Object

},

Data(){

return{

}

},

}

</script>

<template>

<li>id: {{ todo.id }}; userId: {{ todo.userId }}; title: {{ todo.title }}; <input type="checkbox" :checked = "todo.completed" /></li>

</template>

Реализация функциональных требований и тестирование

Сценарий тестирования включает следующие шаги.

1. Просмотреть весь список задач в исходном состоянии.
2. Отсортировать задачи.
3. Добавить задачи со следующими id 205, 206, 207. Они должны появиться в списке задач.
4. Удалить задачи с id 5, 7, 17, 206. Они должны исчезнуть из списка задач.
5. Накладываем фильтр с id от 5 до 21 включительно. Задачи 5, 7, 17 должна исчезнуть.
6. Снять фильтр. Должны увидеть задачи 205, 207 и не увидеть задачи с id 5, 7, 17.

Ниже приводятся снимки экрана при соответствующих шагах тестирования.

Просмотр списка задач

Рисунок 1 показывает вид экрана при просмотре списка задач.

Видим, что список задач включает подсписок локально определенных задач с id от 201 до 204 и подсписок удалаенно определенных задач из 200 элементов с

id от 1 до 200.

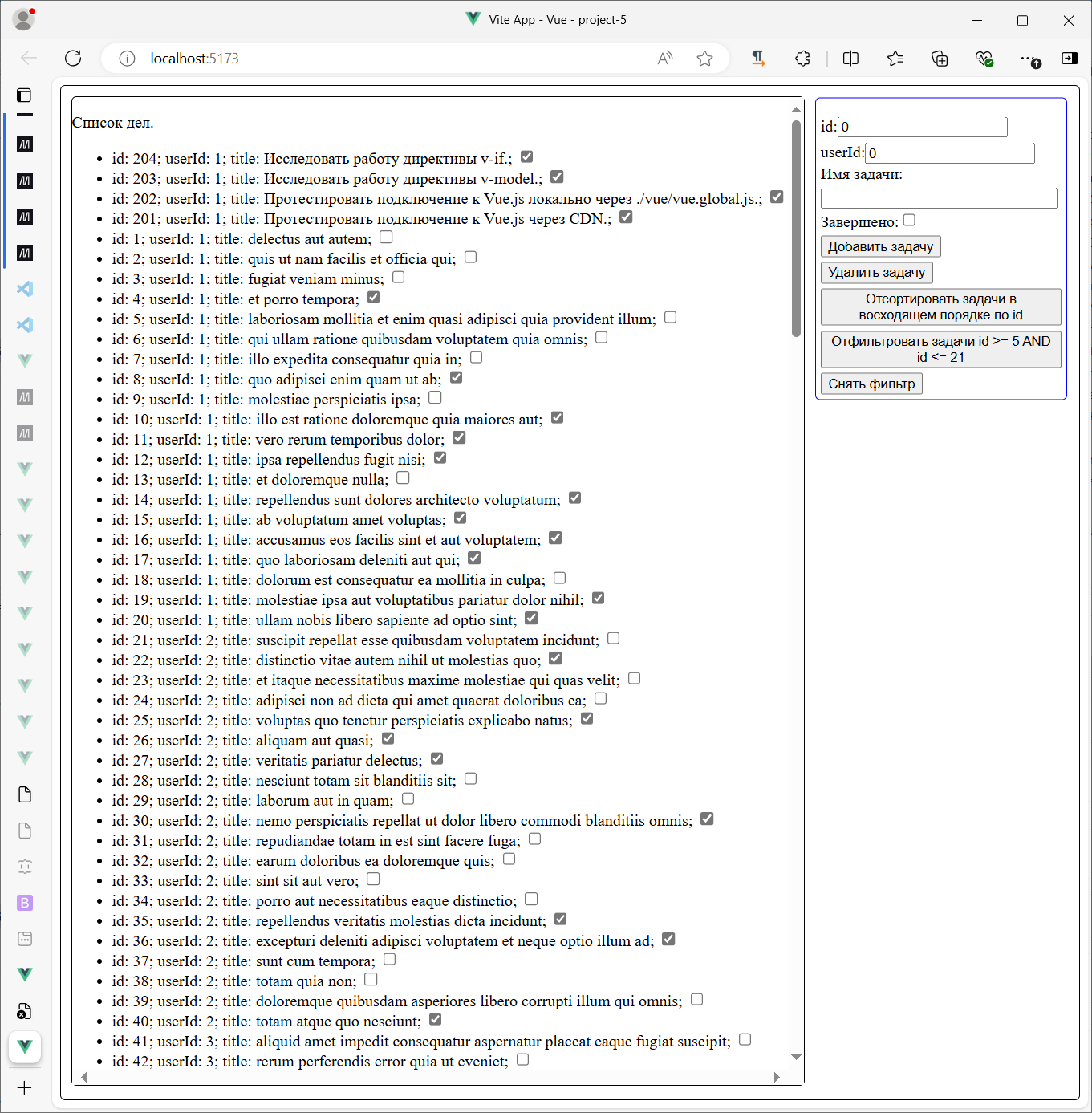


Рисунок 1. Просмотр списка задач в исходном состоянии

Сортировка списка задач

Рисунок 2 показывает вид экрана после сортировки списка задач.

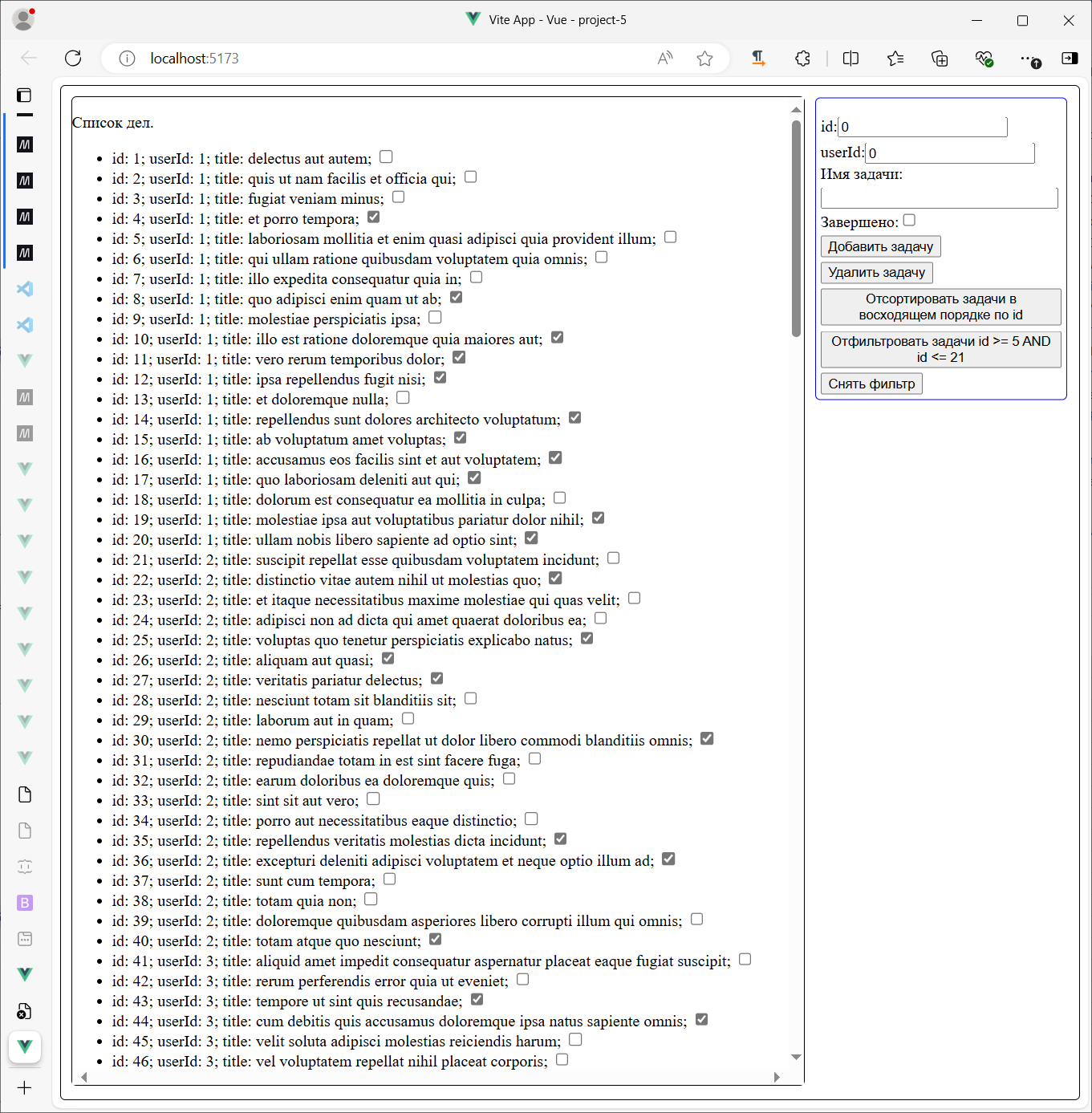


Рисунок 2. Вид экрана после сортировки списка задач

Добавление задач

Рисунок 3 показывает добавление задач с id 205, 206, 207.

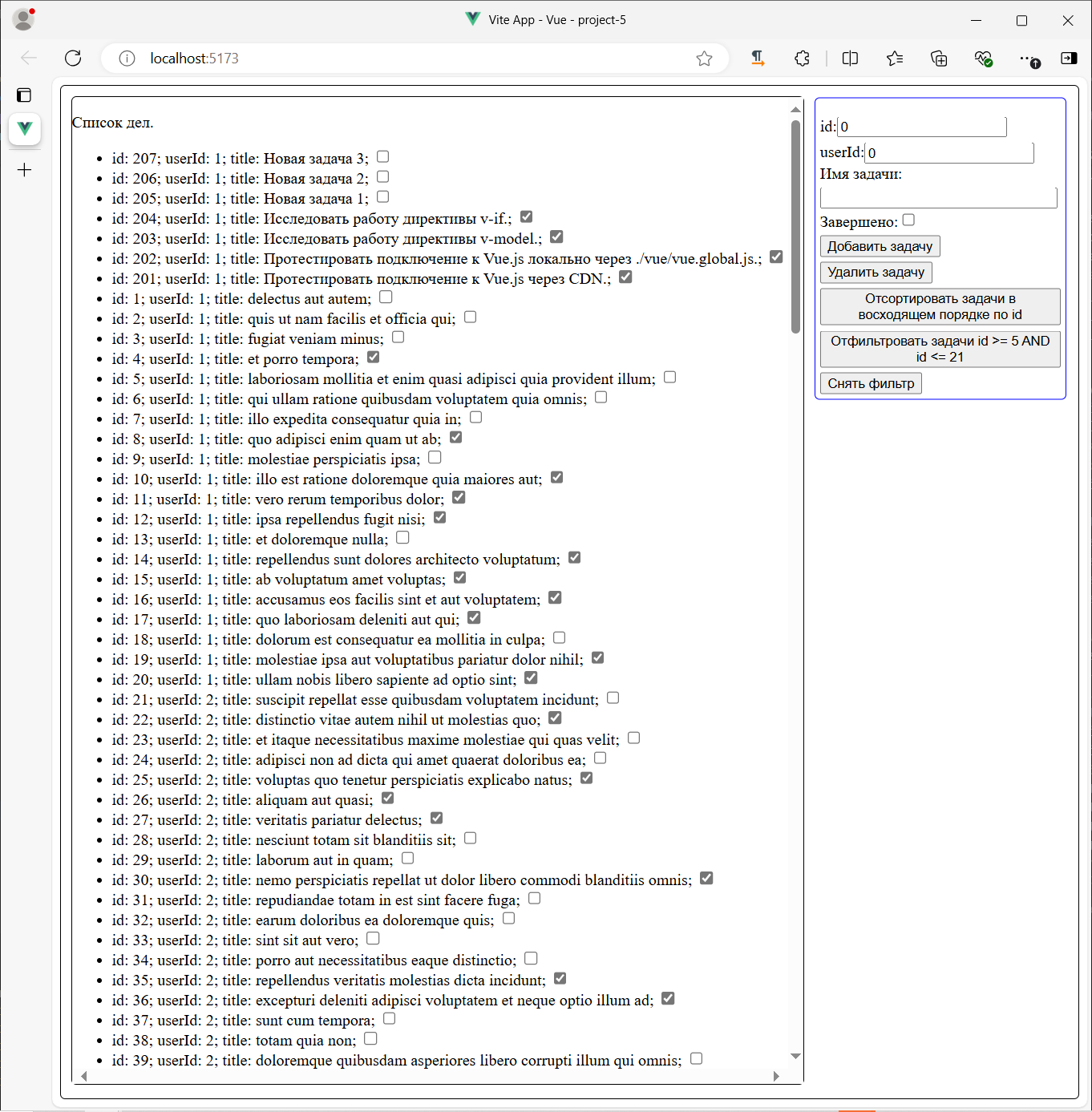


Рисунок 3. Добавлены задачи с id 205, 206, 207

Удаление задач

Рисунок 3 показывает вид экрана после удаления задач с id .

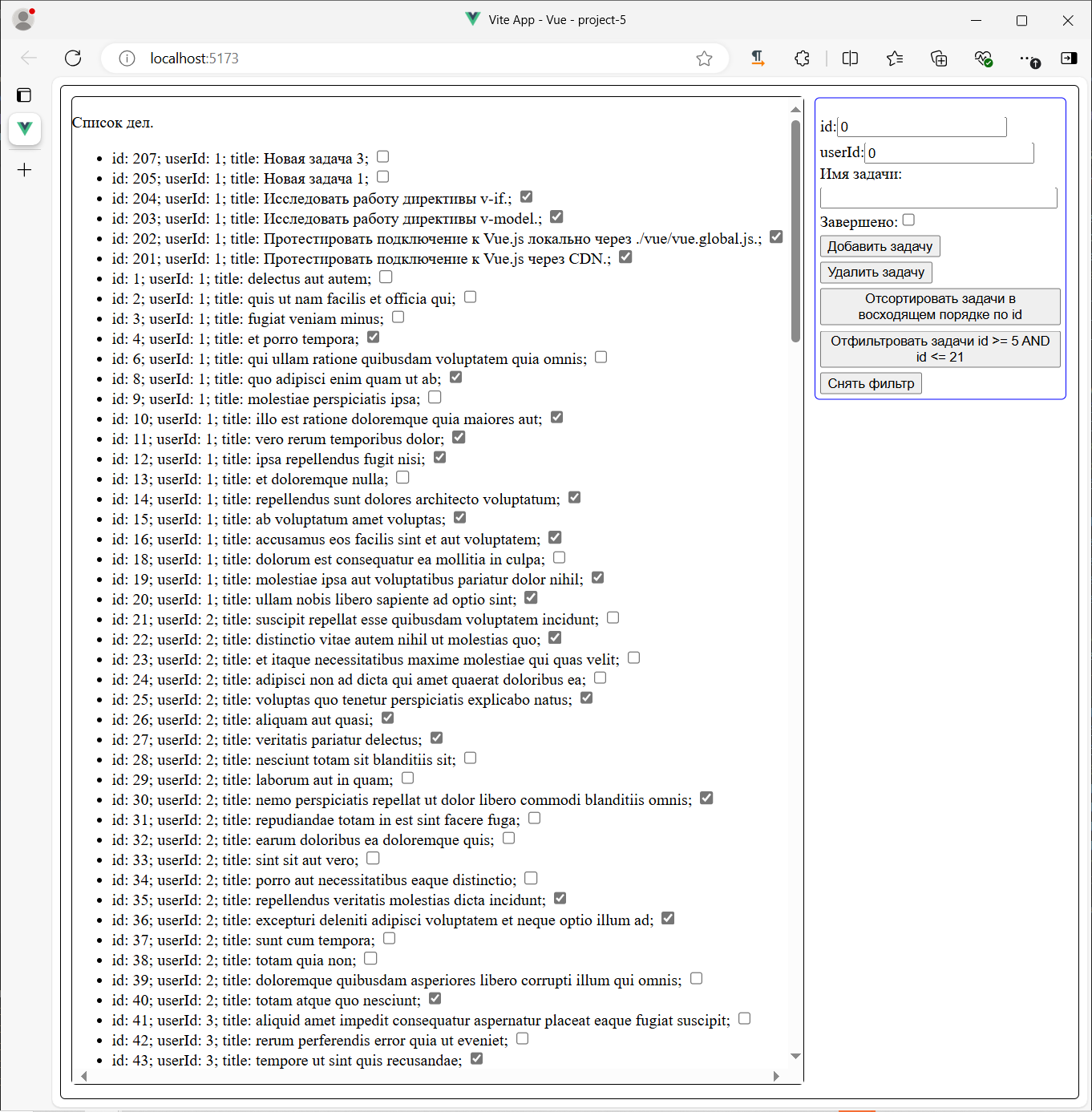


Рисунок 4. Удалены задачи 5, 7, 17, 206

Фильтрация задач

Рисунок 5 показывает снимок экрана после фильтрации, задачи 5, 7, 17 отсутствуют.

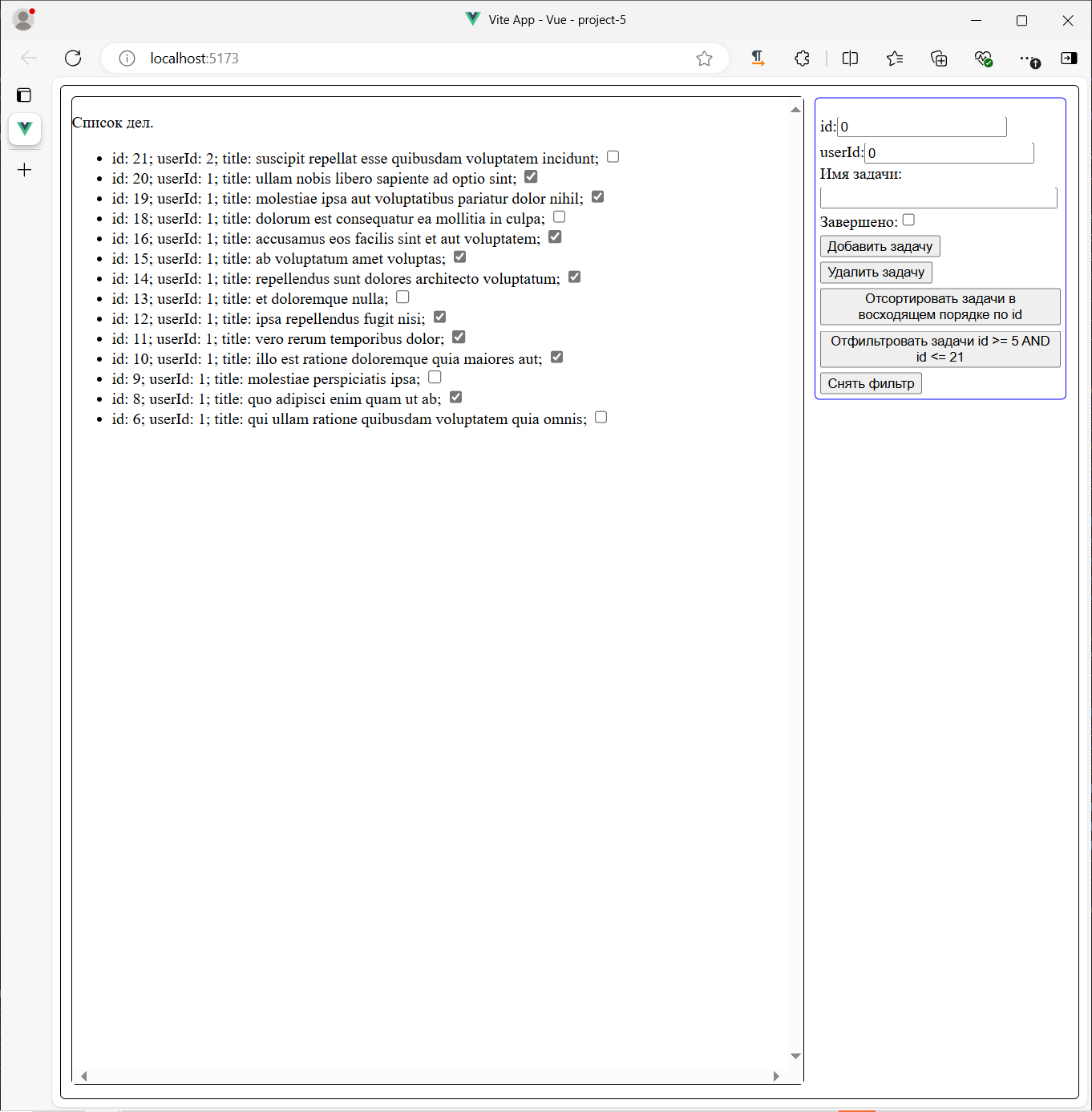


Рисунок 5. Наложен фильтр на задачи с 5 по 21 включительно

Снятие фильтра

Рисунок 6 показывает вид экрана после добавдения и удаления задач и сняти фильтра. Задач 5, 7, 17, 206 нет.

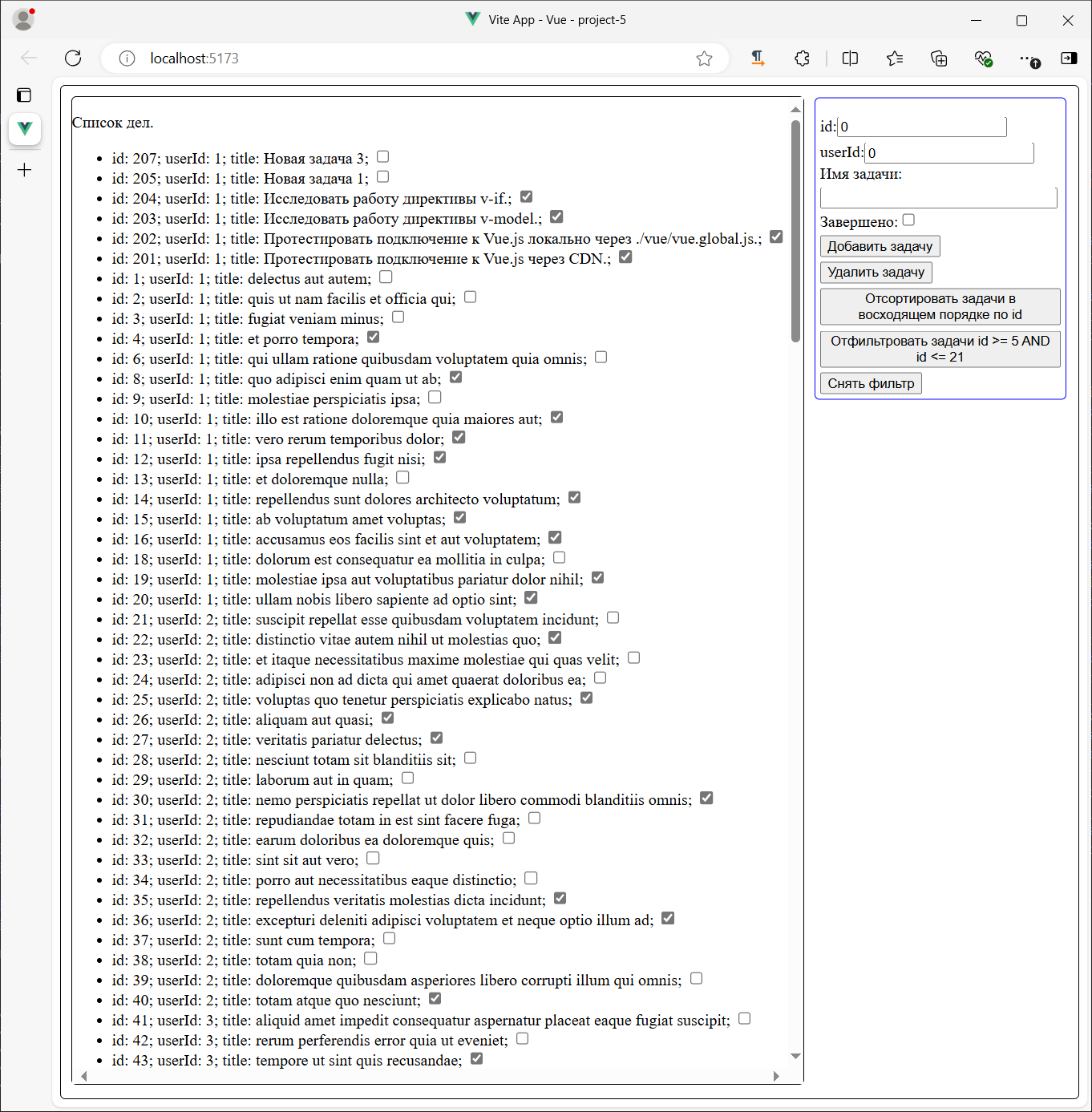


Рисунок 6. После добавления и удаления задач фильтр снят

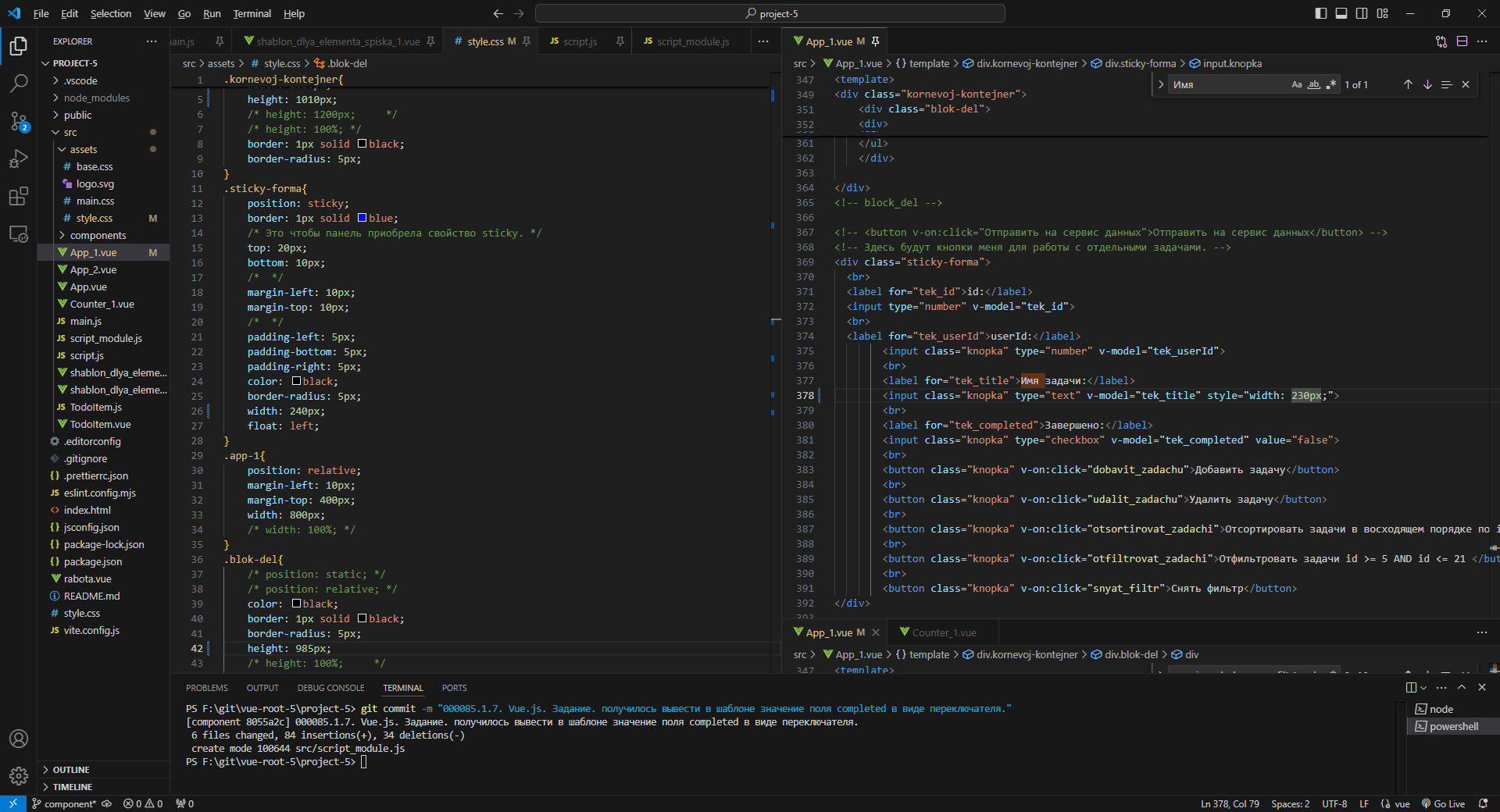


Рисунок 7. Разработка приложения в среде Visual Studio Code

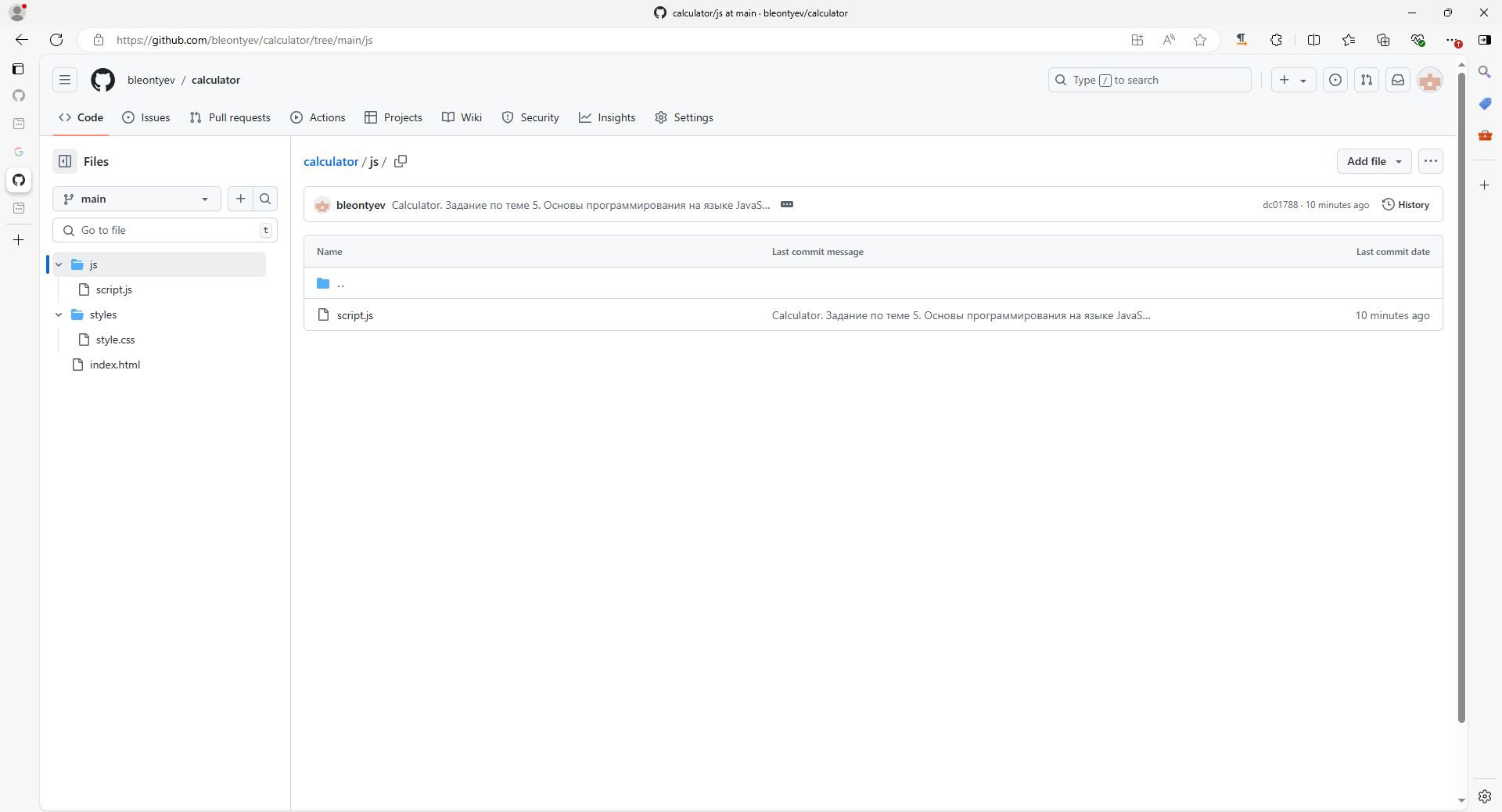


Рисунок 9. Ветка main репозитория