Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ

УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированной обработки информации (АОИ)

**ИНТЕРНЕТ-ПРИЛОЖЕНИЕ «СПИСОК ДЕЛ»**

Отчет о выполнении лабораторной работы

по дисциплине «Разработка интернет-приложений»

Студент гр. 429-3

\_\_\_\_\_\_\_\_\_\_ Бабец А. А.

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2021 г.

Ассистент каф. АОИ

\_\_\_\_\_\_\_\_\_\_ Владимиров М. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2021 г.

Томск 2021

**Оглавление**

[1 Введение 3](#_Toc83149604)

[2 Основная часть 4](#_Toc83149605)

[2.1 Описание используемой среды программирования 4](#_Toc83149606)

[2.2 Описание метода решения задачи 4](#_Toc83149607)

[2.3 Описание фрагментов кода 5](#_Toc83149608)

[2.4 Описание функциональных возможностей разработанного приложения. 6](#_Toc83149609)

[3 Заключение 7](#_Toc83149611)

[Приложения 8](#_Toc83149612)

# Введение

Целью лабораторной работы является интеграция frontend приложения с backend

Задачи лабораторной работы следующие:

* Отправка запросов с фронтенда на бекенд;
* Сохранение изменений на сервере.

# Основная часть

## Описание стека технологий

Для разработки frontend-приложения было принято решение воспользоваться библиотекой ReactJS для Javascript. React – это инструмент для создания пользовательских интерфейсов. Его главная задача – обеспечение вывода на экран того, что можно видеть на веб-страницах. React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты. Мы называем эти фрагменты компонентами.

Также писать код будем в Visual Studio Code, так как это бесплатная IDE с широким функционалом.

## Описание метода решения задачи

Для реализации приложения необходимо реализовать api для взаимодействия с сервером .

## Описание фрагментов кода

Для реализации api необходимо реализовать CRUD функционал.

Для написания запросов импортируем библиотеку axios.

Для создания дел была написана такая функция:

const createTodo **=** useCallback**(async** **()** **=>** **{**

**if** **(!**text**)** **return** **null**

**try** **{**

**await** axios**.**post**(**'/api/todo/add'**,** **{**text**},** **{**

headers**:** **{**

'Content-Type'**:** 'application/json'

**}**

**}).**then**(**response **=>** **{**

setText**(**''**)**

setTodos**([...**todos**],** response**.**data**)**

getTodo**()**

**})**

**}**

**catch** **(**error**)** **{**

console**.**log**(**error**)**

**}**

**},** **[**text**,** todos**,** getTodo**])**

Для удаления дел создана следующая функция:

const removeTodo **=** useCallback**(async** **(**id**)** **=>** **{**

**try** **{**

**await** axios**.**delete**(**`**/api/todo/**delete**/**$**{**id**}**`**,** **{**id**},** **{**

headers**:** **{**

'Content-Type'**:** 'application/json'

**}**

**})**

**.**then**(()** **=>** getTodo**())**

**}**

**catch** **(**error**)** **{**

console**.**log**(**error**)**

**}**

**},** **[**getTodo**])**

Для обновления флага о выполнении была создана следующая функция:

const completeTodo **=** useCallback**(async** **(**id**)** **=>** **{**

**try** **{**

**await** axios**.**put**(**`**/api/todo/**complete**/**$**{**id**}**`**,** **{**id**},** **{**

headers**:** **{**

'Content-Type'**:** 'application/json'

**}**

**})**

**.**then**((**response**)** **=>** **{**

setTodos**([...**todos**],** response**.**data**)**

getTodo**()**

**})**

**}**

**catch** **(**error**)** **{**

console**.**log**(**error**)**

**}**

**},** **[**getTodo**,** todos**])**

Для получения списка дел была написана следующая функция:

const getTodo **=** useCallback**(async** **()** **=>** **{**

**try** **{**

**await** axios**.get(**'/api/todo'**,{**

headers**:** **{**

'Content-Type'**:** 'application/json'

**}**

**})**

**.**then**((**response**)** **=>** setTodos**(**response**.**data**))**

**}**

**catch** **(**error**)** **{**

console**.**log**(**error**)**

**}**

**},** **[])**

Для получения списка дел при загрузке страницы используется ReactHook «useEffect»:

useEffect**(()** **=>** **{**

getTodo**()**

**},** **[**getTodo**])**

Функция создания привязана к обработчику события формы:

**<**form onSubmit **=** **{**e **=>** **{**

e**.**preventDefault**()**

createTodo**()**

**}**

**}>**

Функция удаления привязана к обработчику события иконки:

**<**IconButton

aria**-**label**=**"Delete"

**>**

**<**DeleteIcon onClick**={()** **=>** removeTodo**(**todo**.**\_id**)}** **/>**

**</**IconButton**>**

Функция изменения флага о выполнении привязана к обработчику события checkbox:  
 **<**Checkbox tabIndex**={-**1**}** onClick**={()** **=>** completeTodo**(**todo**.**\_id**)}** **checked={**todo**.**completed**}** **/>**

Листинг кода представлен в приложении Б.

## Описание функциональных возможностей разработанного приложения

При загрузке страницы видим окно для ввода названия (добавление осуществляется по нажатию клавиши Enter), ниже расположен сам список дел., в котором можно удалять дела и изменять флаг о выполнении

Скриншоты приложения представлен в приложении Б.

# Заключение

В практической работе были достигнуты поставленные задачи. Мы познакомились с библиотекой ReactJS и изучили её основы. Разработали проект для каталога электроники.

# Приложения

# Приложение А

# (обязательное)

# Листинг кода

index.js

import React **from** 'react'

import ReactDOM **from** 'react-dom'

import './index.scss'

import App **from** './App'

ReactDOM**.**render**(**

**<**React**.**StrictMode**>**

**<**App **/>**

**</**React**.**StrictMode**>,**

document**.**getElementById**(**'root'**)**

**);**

App.js

import React **from** 'react'

import MainPage **from** './pages/MainPage/MainPage'

import './App.scss'

function App**()** **{**

**return** **(**

**<**div className**=**"app"**>**

**<**MainPage **/>**

**</**div**>**

**)**

**}**

export **default** App

MainPage.jsx

import React, {useState, useCallback, useEffect} from 'react'

import axios from 'axios'

import List from '@material-ui/core/List';

import ListItem from '@material-ui/core/ListItem';

import ListItemSecondaryAction from '@material-ui/core/ListItemSecondaryAction';

import ListItemText from '@material-ui/core/ListItemText';

import Checkbox from '@material-ui/core/Checkbox';

import IconButton from '@material-ui/core/IconButton';

import DeleteIcon from '@material-ui/icons/Delete';

import TextField from '@material-ui/core/TextField';

import Typography from '@material-ui/core/Typography';

import "./MainPage.scss"

const Mainpage = () => {

const [text, setText] = useState('')

const [todos, setTodos] = useState([])

const getTodo = useCallback(async () => {

try {

await axios.get('/api/todo',{

headers: {

'Content-Type': 'application/json'

}

})

.then((response) => setTodos(response.data))

}

catch (error) {

console.log(error)

}

}, [])

const removeTodo = useCallback(async (id) => {

try {

await axios.delete(`/api/todo/delete/${id}`, {id}, {

headers: {

'Content-Type': 'application/json'

}

})

.then(() => getTodo())

}

catch (error) {

console.log(error)

}

}, [getTodo])

const completeTodo = useCallback(async (id) => {

try {

await axios.put(`/api/todo/complete/${id}`, {id}, {

headers: {

'Content-Type': 'application/json'

}

})

.then((response) => {

setTodos([...todos], response.data)

getTodo()

})

}

catch (error) {

console.log(error)

}

}, [getTodo, todos])

const createTodo = useCallback(async () => {

if (!text) return null

try {

await axios.post('/api/todo/add', {text}, {

headers: {

'Content-Type': 'application/json'

}

}).then(response => {

setText('')

setTodos([...todos], response.data)

getTodo()

})

}

catch (error) {

console.log(error)

}

}, [text, todos, getTodo])

useEffect(() => {

getTodo()

}, [getTodo])

return (

<div className="App">

<Typography component="h1" variant="h2">

Todos

</Typography>

<form onSubmit = {e => {

e.preventDefault()

createTodo()

}

}>

<TextField

variant="outlined"

placeholder="Add todo"

margin="normal"

onChange={event => {

setText(event.target.value);

}}

value = {text}

/>

</form>

<List>

{

todos.map((todo, index) => {

let cls = ['']

if (todo.completed) {

cls.push('completed')

}

return (

<ListItem key={index+1} dense button>

<Checkbox tabIndex={-1} onClick={() => completeTodo(todo.\_id)} checked={todo.completed} />

<ListItemText primary={todo.text} className={cls.join(' ')}/>

<ListItemSecondaryAction>

<IconButton

aria-label="Delete"

>

<DeleteIcon onClick={() => removeTodo(todo.\_id)} />

</IconButton>

</ListItemSecondaryAction>

</ListItem>

)

})

}

</List>

</div>

)

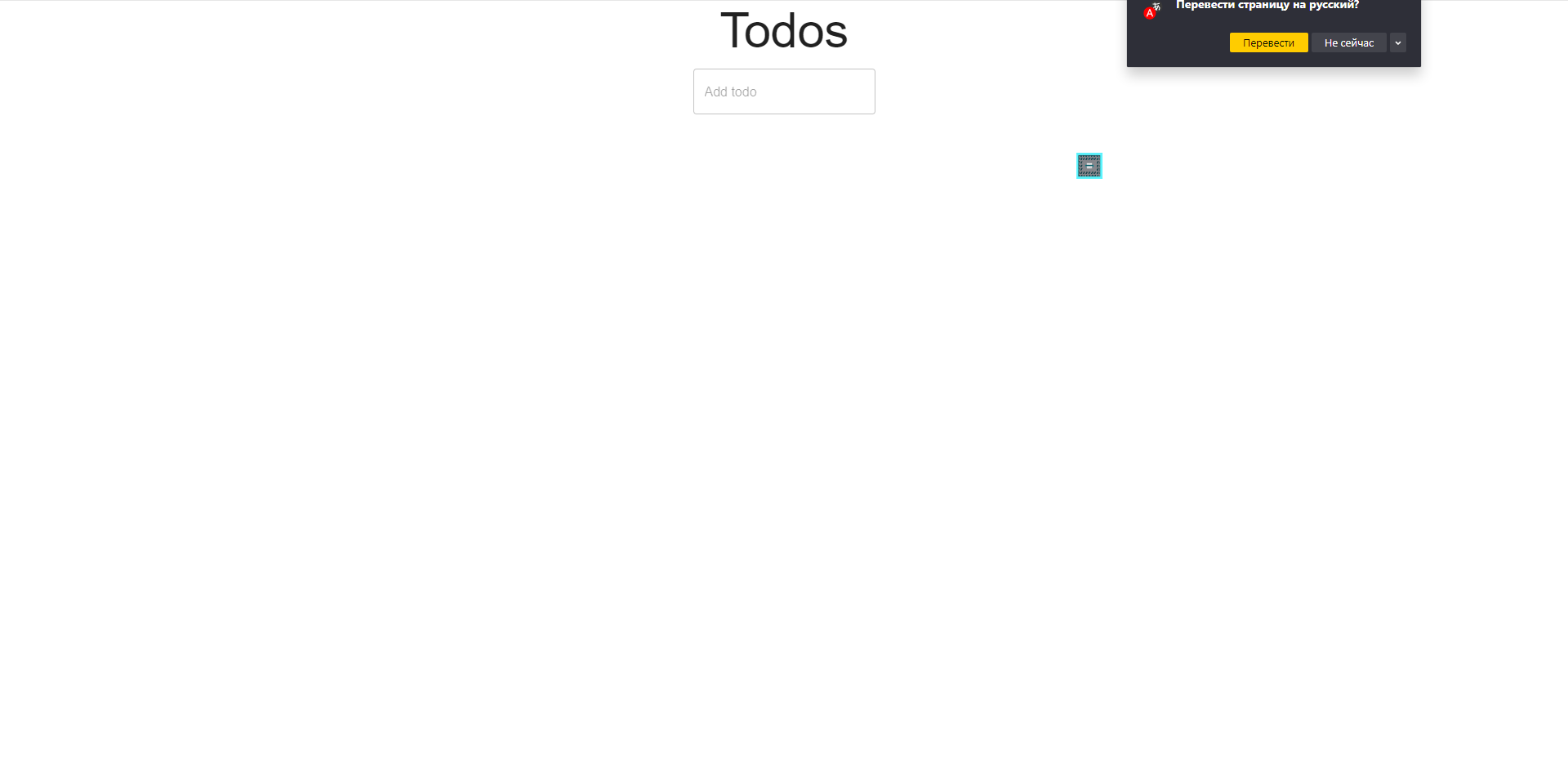
}

export default Mainpage

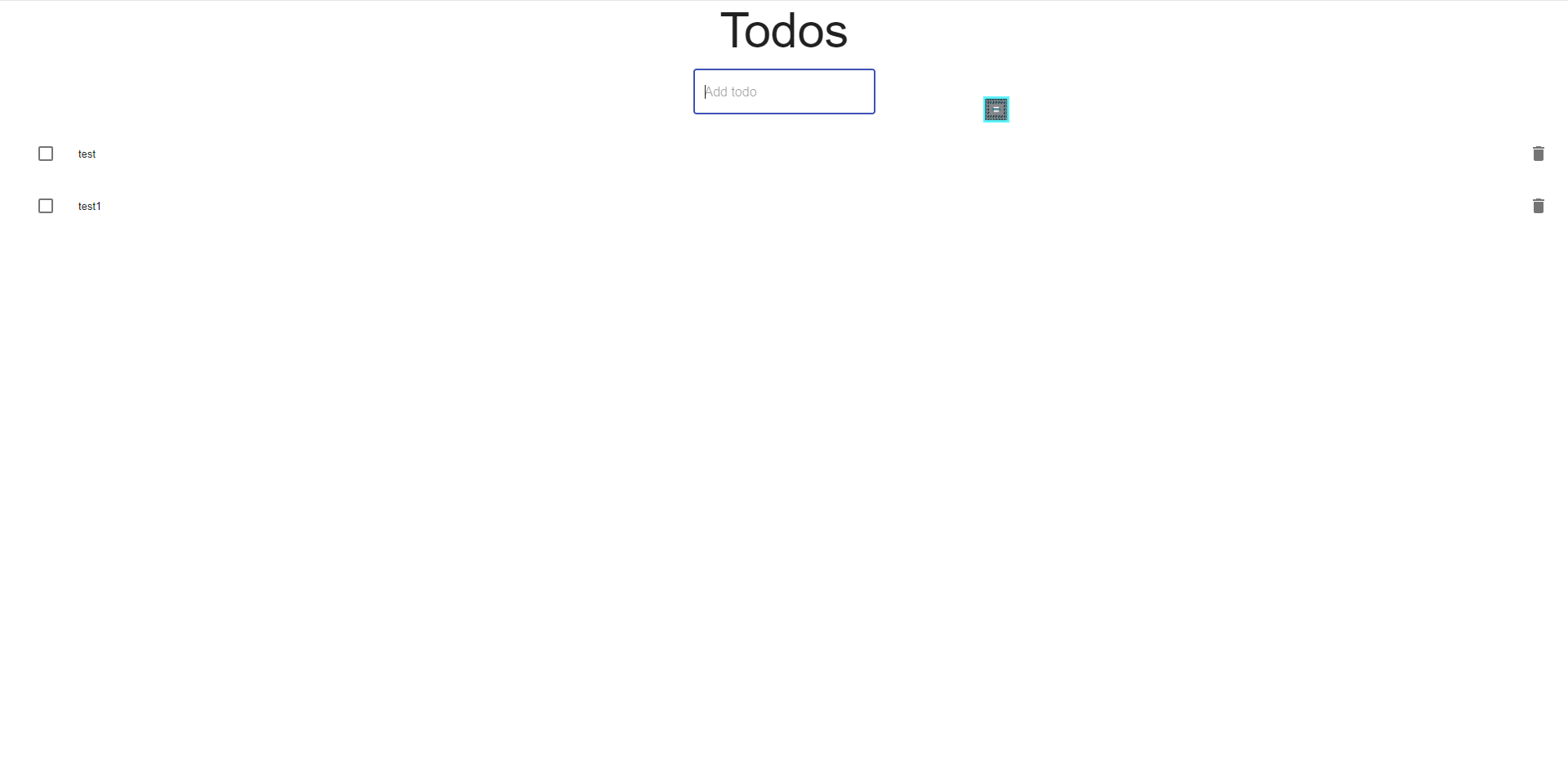
# Приложение Б

# (обязательное)

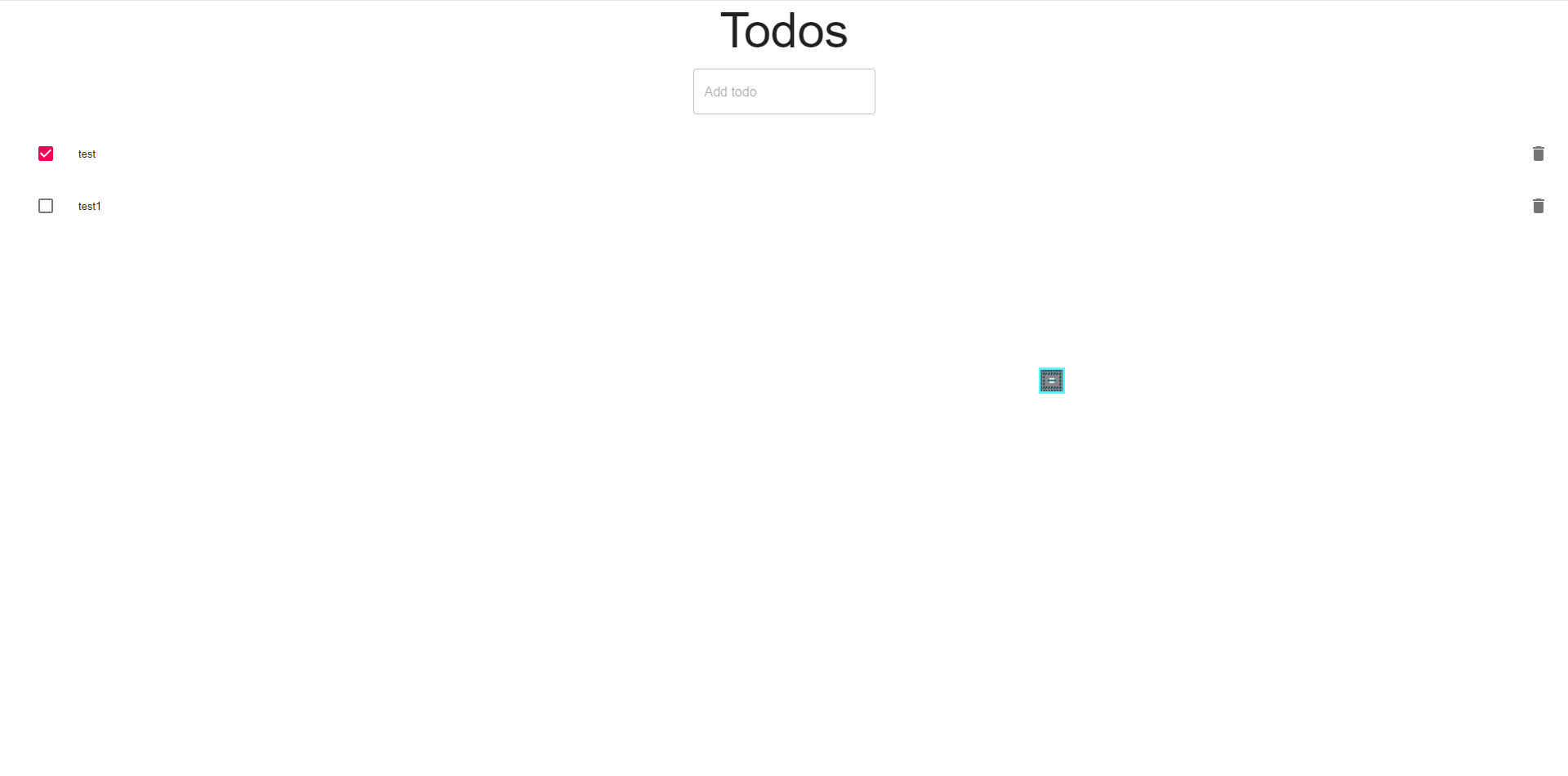
# Начальное окно приложения



# Проверка функционала приложения



# Проверка функционала приложения



# Проверка функционала приложения

