



МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

---

Институт №3

«Системы управления, информатика и электроэнергетика»

Кафедра №304

«Вычислительные машины, системы и сети»

Отчет по курсовой работе

по дисциплине: «**Интернет-технологии**»

Выполнил:

Студент 3-го курса

Гр. М30-307Б-18

Гордеев Н. М.

Принял:

Титов Юрий Павлович

Москва 2021

## Задание

Разработать клиент-серверное AJAX приложение. Серверное BackEnd приложение необходимо разработать на Node.JS (необходимо поставить менеджер пакетов npm) с применением фреймворка Express. Для хранения информации на сервере используется документоориентированная NOSQL база данных MongoDB. Клиентское приложение пишется с помощью фреймворка React.JS с применением CSS для более приятного визуального отображения информации.

В результате клиент-серверное программное приложение должно обеспечивать:

1. Возможность подключения множества клиентов к серверу (в идеале к удаленному серверу)
2. Загрузки информации о разработчиках и назначении программы
3. Возможность добавления новой информации на сервер
4. Возможность удаления с сервера необходимой информации
5. Возможность вывода на экран клиента информации, хранящейся на сервере с возможностью выбора условий вывода (например, вывести все пары на сегодня) и различных вариантов сортировки данных.

Тип хранимой на сервере информации смотрите в соответствии с Вариантом, или можете предложить свой вариант:

Реализовать алгоритма менеджера памяти (FIFO, WS-Clock)

# Серверная часть

Стек технологий:

- Node.js
- Express
- MongoDB

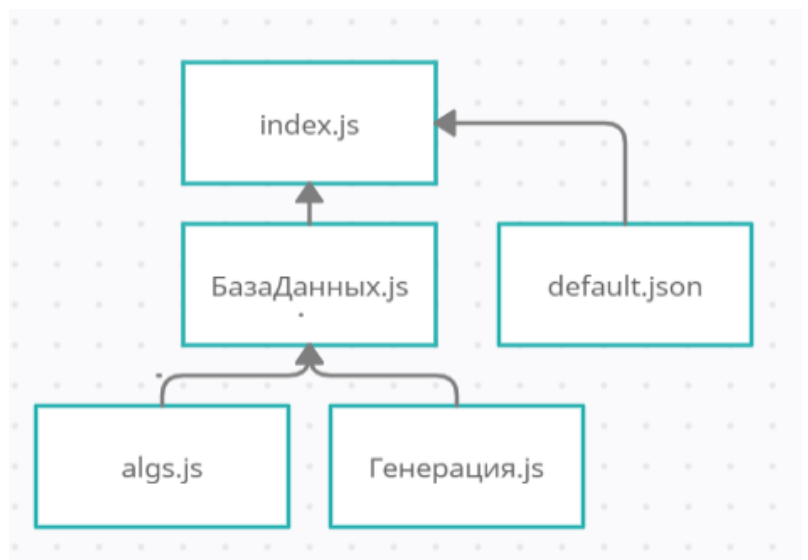
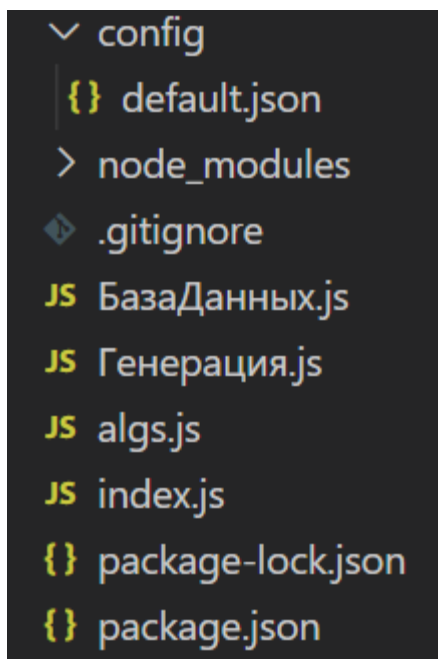
**Node.js** - это серверная платформа для работы с JavaScript через движок V8.

**Express** - это модульный веб - фреймворк для Node.js, который упрощает разработку и облегчает написание безопасных, модульных и быстрых приложений.

**MongoDB** — это документно-ориентированная база данных NoSQL, используемая для хранения больших объемов данных.

Для доступа к базе данных использовалась библиотека **Mongoose**.

## Структура серверной части проекта



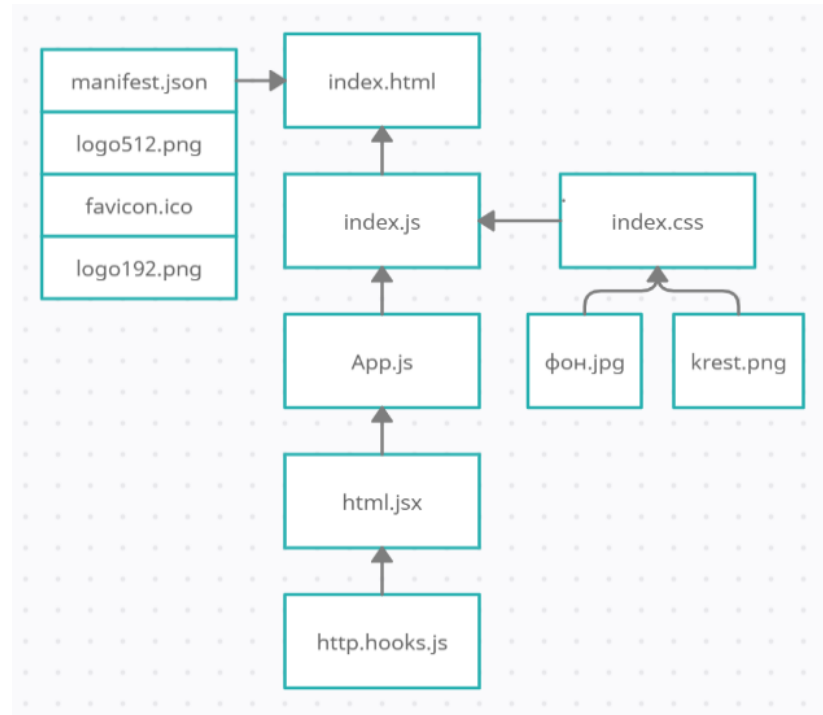
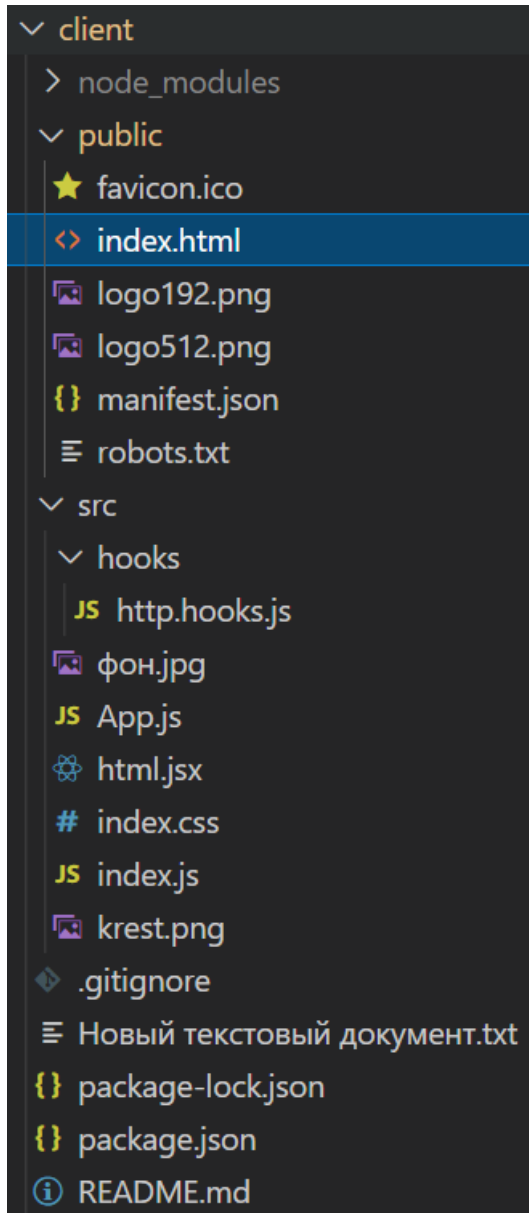
# Клиентская часть

Стек технологий:

- React

**React** — это декларативная, эффективная и гибкая JavaScript библиотека для создания пользовательских интерфейсов.

## Структура клиентской части проекта



## Структура базы данных

String: Название

Number: Размер буфера

Number: Количество элементов

Number: Количество страниц

Number: Рабочее множество

Number: Сброс обращения

Number: Время записи на диск

Array: {  
    Number: Номер страницы  
    Number: Время обращения  
    Bool: Запись  
}

## Структура json соединения

### Отправка данных

```
const response = await fetch(url, {method, body, headers})
```

```
const data = await response.json()
```

Где:

url - url страницы сервера `"/api/start/Get"`

Используется технология проксирования. Итоговый запроса url выглядит так

<http://localhost:7437/api/start/Get> и localhost:7437 меняется в глобальных настройках.

method – тип запроса `"GET"`

body – Произвольный массив данных, сконвертированный в строку командой

```
body = JSON.stringify(body)
```

headers – Содержит информацию о типе соединения

```
headers["Content-Type"] = "application/json"
```

### Ответ

```
res.status(201).json({ message: 'Данные сохранены', fifo: fifo, WS_Clock: WS_Clock })
```

Где:

res – информация об отправителе

status указывает была ли ошибка при соединении.


message – сообщение пользователю

fifo, WS\_Clock – Произвольные массивы данных, преобразованные командой

```
JSON.stringify()
```


## Интерфейс

Загрузить данные

откуда? 

ЗАГРУЗИТЬ

Сгенерировать новые

Название Должно быть уникальным 

Размер буфера

Количество элементов

Количество страниц

Рабочее множество

Время сброса обращения

Время записи на диск

СГЕНЕРИРОВАТЬ

УДАЛИТЬ

fifo

N:

PF:

NPF =

WS\_Clock

ИВО:

ВО:

ОП:

PF:

NPF =

Загрузить данные


откуда?

из файла

из файла

с сервера

ВСЕ ЗАПИСЬ: 1

ТЕСТИМ 


ТЕСТИМ2

ТЕСТИМ3

ТЕСТИМ4

ЗАГРУЗИТЬ

Сгенерировать новые

Название Должно быть уникальным 

Размер буфера

Количество элементов

Количество страниц

Рабочее множество


Время сброса обращения


Время записи на диск

СГЕНЕРИРОВАТЬ


УДАЛИТЬ

Загрузить данные

Тестим 

ЗАГРУЗИТЬ 

Сгенерировать новые

Название Тестим 

Размер буфера

Количество элементов

Количество страниц

Рабочее множество

Время сброса обращения

Время записи на диск

СГЕНЕРИРОВАТЬ

УДАЛИТЬ




## Загрузить данные

Тестим

ЗАГРУЗИТЬ

## Сгенерировать новые

Название	Тестим	
Размер буфера	5	
Количество элементов	30	
Количество страниц	15	
Рабочее множество	5	
Время сброса обращения	10	
Время записи на диск	5	

СГЕНЕРИРОВАТЬ

УДАЛИТЬ

Название: Тестим  
Размер буфера: 5  
Количество элементов: 30  
Количество страниц: 15  
Рабочее множество: 5  
Сброс обращения: 10  
Время записи: 5  
Содержимое: [  
Номер страницы: 12 Время обращения: 2 Запись: 1  
Номер страницы: 0 Время обращения: 3 Запись: 0  
Номер страницы: 14 Время обращения: 4 Запись: 0  
Номер страницы: 12 Время обращения: 6 Запись: 0  
Номер страницы: 0 Время обращения: 7 Запись: 0  
Номер страницы: 13 Время обращения: 8 Запись: 1  
Номер страницы: 1 Время обращения: 11 Запись: 0  
Номер страницы: 5 Время обращения: 14 Запись: 1  
Номер страницы: 7 Время обращения: 16 Запись: 1  
Номер страницы: 8 Время обращения: 17 Запись: 1  
Номер страницы: 6 Время обращения: 23 Запись: 0  
Номер страницы: 12 Время обращения: 24 Запись: 1  
Номер страницы: 0 Время обращения: 25 Запись: 0  
Номер страницы: 10 Время обращения: 26 Запись: 0  
Номер страницы: 4 Время обращения: 27 Запись: 0  
Номер страницы: 14 Время обращения: 28 Запись: 1  
Номер страницы: 14 Время обращения: 29 Запись: 1  
Номер страницы: 11 Время обращения: 31 Запись: 1  
Номер страницы: 10 Время обращения: 32 Запись: 0  
]

fifo

N:	12	0	14	12	0	13	1	5	7	8	6	12	0	10	4	14	14	11	10	0	13	14	0	3	1	3	6	14	5	7
1	12	0	14	14	14	13	1	5	7	8	6	12	0	10	4	14	14	11	11	11	13	13	0	3	1	1	6	14	5	7
2	.	12	0	0	0	14	13	1	5	7	8	6	12	0	10	4	4	14	14	14	11	11	13	0	3	3	1	6	14	5
3	.	.	12	12	12	0	14	13	1	5	7	8	6	12	0	10	10	4	4	4	14	14	11	13	0	0	3	1	6	14
4	.	.	.	.	.	12	0	14	13	1	5	7	8	6	12	0	0	10	10	10	4	4	14	11	13	13	0	3	1	6
5	.	.	.	.	.	12	0	14	13	1	5	7	8	6	12	12	0	0	0	10	10	4	14	11	11	13	0	3	1	
PF:	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	1	0	1	1	1	

NPF = 23

WS\_Clock

ИВО:	2	3	4	6	7	8	10	11	14	14	14	14	14	14	16	17	17	20	23	23	23	23	23	23	23	23	24	24	24	24	25	25	25	25	25	26	26	27	28	28	28	28	28			
ВО:	2	3	4	6	7	8	10	11							14	16		17	20								23		24					25		26	27									
ОП:	12	0	14	12	0	13	1	5	5	5	5	5	5	5	5	7	8	8	8	6	6	6	6	6	6	6	6	12	12	12	12	0	0	0	0	0	10	10	4	14	14	14	14	14		
0	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8		
БО	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
БИ	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
ВПИ	0	0	0	0	0	0	10	10	10	10	10	10	10	10	10	10	10	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20		
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	10	10	10	10	10	10	10		
БО	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1		
БИ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
ВПИ	0	0	0	0	0	0	10	10	10	10	10	10	10	10	10	10	10	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20		
2			14	14	14	14	14	14	14	14	14	14	14	14	14	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7		
БО			1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	
БИ			0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NPF = 23

Загрузить данные

Тестим5

ЗАГРУЗИТЬ

Сгенерировать новые

НазваниеТестим5

Размер буфера1

Количество элементов1

Количество страниц1

Рабочее множество1

Время сброса обращения2

Время записи на диск>= 0

СГЕНЕРИРОВАТЬ

УДАЛИТЬ

Название: Тестим5

Размер буфера: 1

Количество элементов: 1

Количество страниц: 1

Рабочее множество: 1

Сброс обращения: 2

Время записи: 0

Содержимое: [

Номер страницы: 0

Время обращения: 1

Запись: 0

]

fifo

N: 0

1 0

PF: 1

NPF = 1

WS\_Clock

ИВО: 1

ВО: 1

ОП: 0

0 0

БО 1

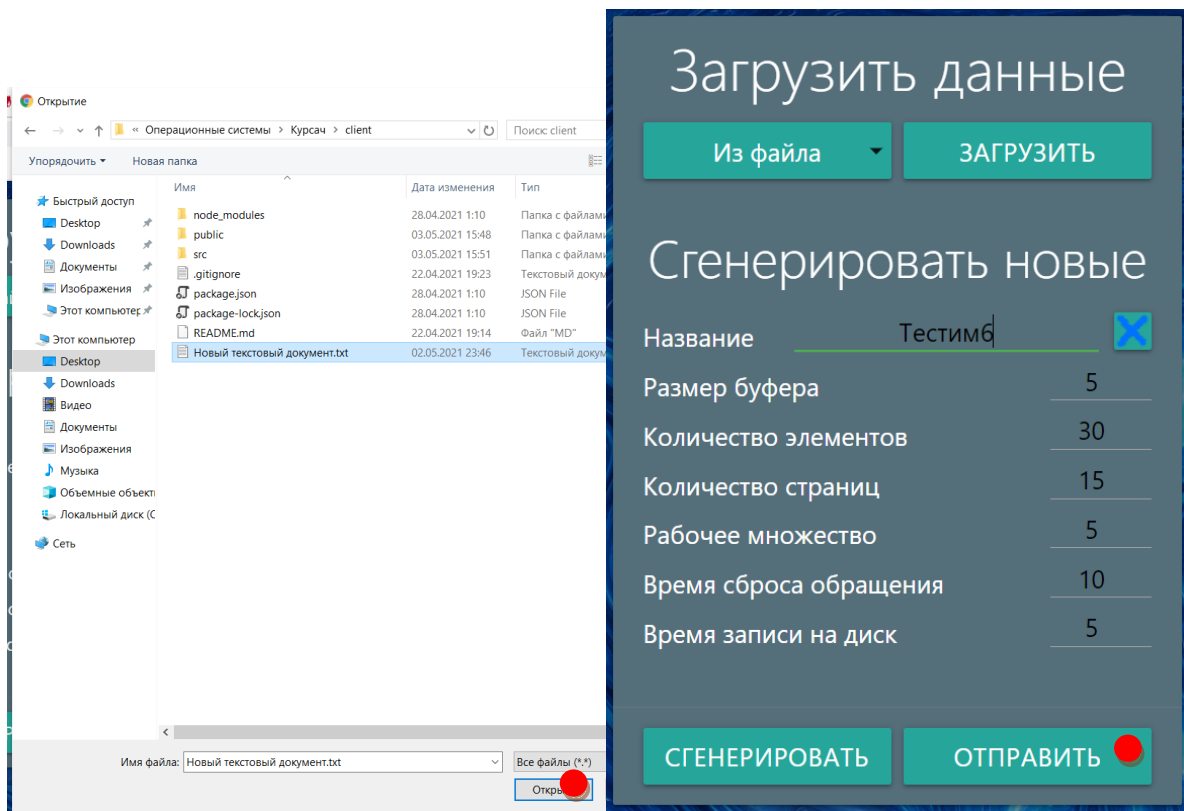
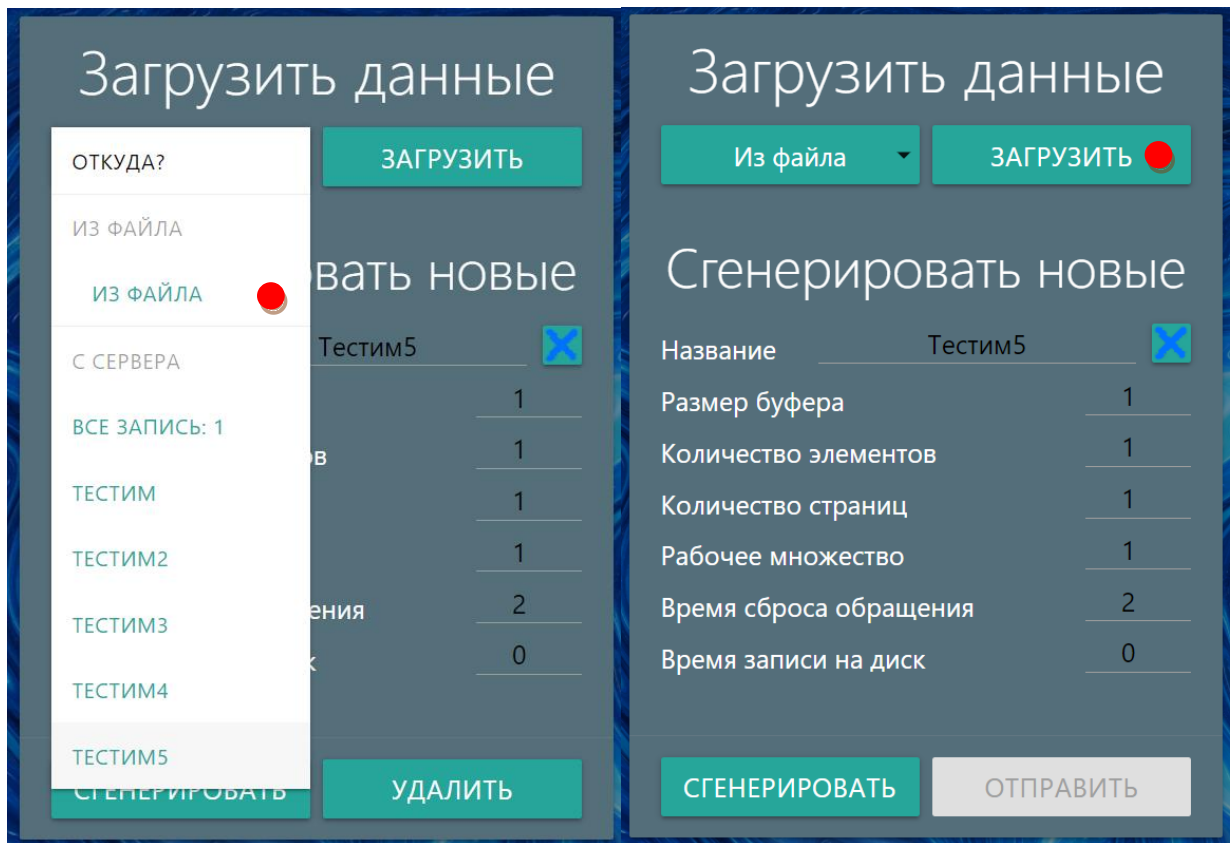
БИ 0

ВПИ 0

PF: 1

NPF = 1





## Содержимое файла :


```
[[5, 30, 15, 5, 10, 5],[12, 2, 1],[0, 3, 2],[14, 4, 1],[12, 6, 0],[0, 7, 0],[13, 8, 1],[1, 11, 0],[5, 14, 1],[7, 16, 1],[8, 17, 1],[6, 23, 0],[12, 24, 1],
[0, 25, 0],[10, 26, 0],[4, 27, 0],[14, 28, 1],[14, 29, 1],[11, 31, 1],[10, 32, 0],[0, 33, 0],[13, 36, 0],[14, 38, 1],[0, 46, 1],[3, 47, 1],[1, 48, 0],[3, 52, 0],
[6, 56, 1],[14, 57, 0],[5, 58, 1],[7, 59, 0]]]
```

## Загрузить данные

Тестим6

ЗАГРУЗИТЬ

## Сгенерировать новые

Название Тестим6 

Размер буфера 5

Количество элементов 30

Количество страниц 15

Рабочее множество 5

Время сброса обращения 10

Время записи на диск 5

СГЕНЕРИРОВАТЬ

УДАЛИТЬ

Название: Тестим6  
Размер буфера: 5  
Количество элементов: 30  
Количество страниц: 15  
Рабочее множество: 5  
Сброс обращения: 10  
Время записи: 5  
Содержимое: [  
Номер страницы: 12 Время обращения: 2 Запись: 1  
Номер страницы: 0 Время обращения: 3 Запись: 2  
Номер страницы: 14 Время обращения: 4 Запись: 1  
Номер страницы: 12 Время обращения: 6 Запись: 0  
Номер страницы: 0 Время обращения: 7 Запись: 0  
Номер страницы: 13 Время обращения: 8 Запись: 1  
Номер страницы: 1 Время обращения: 11 Запись: 0  
Номер страницы: 5 Время обращения: 14 Запись: 1  
Номер страницы: 7 Время обращения: 16 Запись: 1  
Номер страницы: 8 Время обращения: 17 Запись: 1  
Номер страницы: 6 Время обращения: 23 Запись: 0  
Номер страницы: 12 Время обращения: 24 Запись: 1  
Номер страницы: 0 Время обращения: 25 Запись: 0  
Номер страницы: 10 Время обращения: 26 Запись: 0  
Номер страницы: 4 Время обращения: 27 Запись: 0  
Номер страницы: 14 Время обращения: 28 Запись: 1  
Номер страницы: 14 Время обращения: 29 Запись: 1  
Номер страницы: 11 Время обращения: 31 Запись: 1  
Номер страницы: 10 Время обращения: 32 Запись: 0

fifo

№:	12	0	14	12	0	13	1	5	7	8	6	12	0	10	4	14	14	11	10	0	13	14	0	3	1	3	6	14	5	7
1	12	0	14	14	14	13	1	5	7	8	6	12	0	10	4	14	14	11	11	11	13	13	0	3	1	1	6	14	5	7
2	.	12	0	0	0	14	13	1	5	7	8	6	12	0	10	4	4	14	14	14	11	11	13	0	3	3	1	6	14	5
3	.	.	12	12	12	0	14	13	1	5	7	8	6	12	0	10	10	4	4	4	14	14	11	13	0	0	3	1	6	14
4	.	.	.	.	.	12	0	14	13	1	5	7	8	6	12	0	10	10	10	4	4	14	11	13	13	0	3	1	6	
5	.	.	.	.	.	12	0	14	13	1	5	7	8	6	12	12	0	0	0	10	10	4	14	11	11	13	0	3	1	
PF:	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0	1	1	1	0	1	1	1	1

NPF = 23

WS\_Clock

ИВО:	2	3	4	6	7	8	10	11	14	14	14	14	14	14	14	14	14	16	17	20	23	24	25	25	25	25	25	25	25	26	26	27	28	29	30	31	31	31	31	31	31	32	33	33		
ВО:	2	3	4	6	7	8	10	11	.	.	.	.	.	.	.	.	.	14	16	17	20	23	24	.	.	.	.	.	.	25	.	26	27	28	29	30	.	.	.	.	31	32	.	.		
ОП:	12	0	14	12	0	13	1	5	5	5	5	5	5	5	5	5	5	7	8	6	12	0	0	0	0	0	0	0	0	10	10	4	14	14	11	11	11	11	11	11	10	0	0	0		
0	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	7	7	7	7	7	7	7	7	7	7	7	7	4	4	4	4	4	4	4	4	4	4	4	4	4	
БО	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0		
БИ	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
ВПИ	0	0	0	0	0	0	10	10	10	10	10	10	10	10	10	10	10	10	10	20	20	20	20	20	20	20	20	20	20	20	20	20	20	30	30	30	30	30	30	30	30	30	30	30		
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8	8	8	8	8	8	8	14	14	14	14	14	14	14	14	14	14	14	14	14		
БО	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	
БИ	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
ВПИ	0	0	0	0	0	0	10	10	10	10	10	10	10	10	10	10	10	10	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	30	30	30	30	30	30	30	30	30	30	30		
2	.	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	14	6	6	6	6	6	6	6	6	6	6	6	0	0	0	0	0	0	0	0	0	0	11	11	11	11
БО	.	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	
БИ	.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	


NPF = 22

## Загрузить данные

Тестим6

ЗАГРУЗИТЬ

## Сгенерировать новые

Название Тестим6 

Размер буфера 5

Количество элементов 30


Количество страниц 15

Рабочее множество 5

Время сброса обращения 10

Время записи на диск 5

СГЕНЕРИРОВАТЬ

УДАЛИТЬ 




## Загрузить данные

откуда?

ЗАГРУЗИТЬ

## Сгенерировать новые

Название	Тестимб	
Размер буфера	5	
Количество элементов	30	
Количество страниц	15	
Рабочее множество	5	
Время сброса обращения	10	
Время записи на диск	5	

СГЕНЕРИРОВАТЬ

УДАЛИТЬ

Данные удалены.

fifo

N:	
PF:	

NPF = 0

WS\_Clock

ИВО:	
ВО:	
ОП:	
PF:	

NPF = 0

## Код серверной части

index.js\*\*\*\*\*

```
const express = require("express")
const Настройки = require("config")
const Mongoose = require("mongoose")

const сервер = express()
const Port = Настройки.get("port") || 7437

сервер.use(express.json({extended: true}))
сервер.use("/api/start", require("./БазаДанных"))

async function start() {
  try {
    await Mongoose.connect(Настройки.get("mongoUri"), {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      useCreateIndex: true
    })
    сер-
вер.listen(Port, ()=> console.log(`Сервер начал работу на порту ${Port}`))
  } catch(e) {
    console.log("Ошибка сервера", e.message)
    process.exit(1)
  }
}
start()
```

\*\*\*\*\*

## Генерация.js\*\*\*\*\*

```
const Генерация = (КоличествоСтраниц,КоличествоЭлементов,СбросОбращения) =>{
    let Содержимое = []
    let ВозможноеВремяОбращения = []
    for (let i = 1; i <= КоличествоЭлементов*2+1; i++) {
        if(i % СбросОбращения !== 0){
            ВозможноеВремяОбращения.push(i)
        }
    }
    for (let i = 0; i < КоличествоЭлементов; i++) {
        let Обращение = {}
        Обраще-
ние.НомерСтраницы = Math.floor(Math.random() * КоличествоСтраниц)
        let t = Math.floor(Math.random() * ВозможноеВремяОбращения.length)
        Обращение.ВремяОбращения = ВозможноеВремяОбращения[t]
        ВозможноеВремяОбращения.splice(t, 1);
        Обращение.Запись = Math.floor(Math.random()*2)
        Содержимое.push(Обращение)
    }
    Содержимое.sort((a, b) => a.ВремяОбращения - b.ВремяОбращения)
    return Содержимое
}
module.exports = {
    Генерация: Генерация
}
```

\*\*\*\*\*



## БазаДанных.js\*\*\*\*\*

```
const {Router} = require('express')
const {Schema, model} = require('mongoose')
const algoritm = require('./алгс')
const Генерация = require('./Генерация')

const stringSchema = new Schema({
  НомерСтраницы: {type: Number, required: true},
  ВремяОбращения: {type: Number, required: true},
  Запись: {type: Number, required: true},
})

const ШаблонДанных = new Schema({
  Название: {type: String, required: true, unique: true},
  РазмерБуфера: {type: Number, required: true},
  КоличествоЭлементов: {type: Number, required: true},
  КоличествоСтраниц: {type: Number, required: true},
  РабочееМножество: {type: Number, required: true},
  СбросОбращения: {type: Number, required: true},
  ВЗД: {type: Number, required: true},
  Содержимое: [stringSchema]
})

const БазаДанных = Router()
var Данные = model('Данные', ШаблонДанных);

// /api/start/Add
БазаДанных.post(
  "/Add",
  async(req, res) => {
    try {
      const {Название, РазмерБуфера, КоличествоЭлементов,КоличествоСтраниц, РабочееМножество, СбросОбращения, ВЗД, Содержимое} = req.body
      const candidate = await Данные.findOne({ Название: Название })
      if(candidate){
        return res.status(400).json({ message: 'Такое название существует!\nПодумайте другое или переименуйте загружаемый файл.' })
      }
      const данные = new Данные({ Название:Название,
        РазмерБуфера:РазмерБуфера,
        КоличествоЭлементов:КоличествоЭлементов,
        КоличествоСтраниц:КоличествоСтраниц,
        РабочееМножество:РабочееМножество,
        СбросОбращения:СбросОбращения,
        ВЗД:ВЗД,
        Содержимое:Содержимое})

      await данные.save()
      let fifo = JSON.stringify(algoritm.fifo(РазмерБуфера,Содержимое))
      let WS_Clock = JSON.stringify(algoritm.WS_Clock(РазмерБуфера, КоличествоЭлементов,
        ВЗД, РабочееМножество, СбросОбращения, Содержимое))
```

```

        res.status(201).json({ message: 'Данные сохранены', fifo: fifo, WS_Clock:
WS_Clock })
    } catch (e) {
        console.log(e)
        res.status(500).json({ message: 'Что-
то пошло не так, попробуйте снова'})
    }
}
)

// /api/start/getOne
БазаДанных.post(
    "/getOne",
    async(req, res) => {
        try {
            const {Название} = req.body
            const candidate = await Данные.findOne({ Название: Название })
            if(!candidate){
                return res.status(400).json({ message: 'Такого набора данных не сущес
твует.' })
            }
            let fifo = JSON.stringify(algoritm.fifo(candidate.РазмерБуффера,candidate
.Содержимое))
            let WS_Clock = JSON.stringify(algoritm.WS_Clock(candidate.РазмерБуффера,c
andidate.КоличествоЭлементов,
                candi-
date.ВЗД, candidate.РабочееМножество,candidate.СбросОбращения,candidate.Содержимо
е))
            res.status(201).json({ message: 'Данные найдены', source: candidate,
fifo: fifo, WS_Clock: WS_Clock })
        } catch (e) {
            console.log(e)
            res.status(500).json({message: 'Что-
то пошло не так, попробуйте снова'})
        }
    }
)

// /api/data/Get
БазаДанных.get(
    '/Get',
    async(req, res) => {
        try{
            const exists = await Данные.find({}, {"Название":1,"РазмерБуффера":1,
"КоличествоЭлемен-
тов":1,"КоличествоСтраниц":1,"РабочееМножество":1,"СбросОбращения":1, "ВЗД":1, "_
id":0}).lean()
            res.json({exists})
        } catch (e) {
            console.log(e)
            res.status(500).json({message: 'Что-то пошло не так, попробуйте снова'})
        }
    })
})

```

```

// /api/data/Delete
БазаДанных.delete(
  '/Delete',
  async(req, res) => {
    try{
      const {Название} = req.body
      const candidate = await Данные.findOneAndDelete({ Название:Название })
      if(!candidate){
        return res.status(400).json({ message: 'Такого набора данных не существует.' })
      }
      res.status(201).json({message: 'Данные удалены.'})
    } catch (e) {
      console.log(e)
      res.status(500).json({message: 'Что-то пошло не так, попробуйте снова'})
    }
  })

// /api/start/Gen
БазаДанных.post(
  "/Gen",
  async(req, res) => {
    try {
      const {t1,t2,КоличествоЭлементов,КоличествоСтраниц,t3,СбросОбращения,t4}
= req.body
      Содержимое = Генерация.Генерация(КоличествоСтраниц,КоличествоЭлементов,СбросОбращения)
      res.status(201).json({ message: 'Данные сгенерированы', Содержимое:Содержимое })
    } catch (e) {
      console.log(e)
      res.status(500).json( {message: 'Что-то пошло не так, попробуйте снова'})
    }
  }
)

module.exports = БазаДанных

```

\*\*\*\*\*

algs.js\*\*\*\*\*

```
const WS_Clock = (РазмерБуффера, КоличествоЭлементов, ВЗД,
  РабочееМножество, СбросОбращения, Содержимое) =>{    let массив = {
    ИзмененноеВремяОбращения: [],
    ВремяОбращения: [],
    НомерСтраницы: [],
    Буффер: [],
    Pf: [],
    Стили: []
  }

  let РасстояниеДоДырки = 0

  for (let index = 0; index < РазмерБуффера; index++) {
    let Буффер = {
      ТекущаяСтраница: [],
      БитОбращения: [],
      БитИзменения: [],
      ВремяПоследнегоИзменения: []
    }
    массив.Буффер.push(Буффер)
  }
  const добавить = () => {
    массив.ИзмененноеВремяОбращения.push(-1)
    массив.ВремяОбращения.push(-1)
    массив.НомерСтраницы.push(0)
    for (let j = 0; j < РазмерБуффера; j++) {
      массив.Буффер[j].ТекущаяСтраница.push(-1)
      массив.Буффер[j].БитОбращения.push(0)
      массив.Буффер[j].БитИзменения.push(0)
      массив.Буффер[j].ВремяПоследнегоИзменения.push(0)
    }

    массив.Pf.push(-1)
  }
  const скопировать = (столбец,i) => {
    добавить()
    мас-
сив.ИзмененноеВремяОбращения[столбец] = Содержимое[i].ВремяОбращения + Расстояние
ДоДырки
    массив.НомерСтраницы[столбец] = Содержимое[i].НомерСтраницы
    for (let j = 0; j < РазмерБуффера; j++) {
      мас-
сив.Буффер[j].ТекущаяСтраница[столбец] = массив.Буффер[j].ТекущаяСтраница[столбец
-1]
      мас-
сив.Буффер[j].БитОбращения[столбец] = массив.Буффер[j].БитОбращения[столбец-1]
      мас-
сив.Буффер[j].БитИзменения[столбец] = массив.Буффер[j].БитИзменения[столбец-1]
      мас-
сив.Буффер[j].ВремяПоследнегоИзменения[столбец] = массив.Буффер[j].ВремяПоследнег
оИзменения[столбец-1]
    }
  }
```

```

    }
    const Сброс = (столбец, i) => {
        for (let temp = массив.ВремяОбращения[столбец-1]; temp < Содержимое[i].ВремяОбращения; temp++) {
            if(temp % СбросОбращения === 0) {
                скопировать(столбец, i)
                массив.ИзмененноеВремяОбращения[столбец] = temp+РасстояниеДоДырки
                массив.ВремяОбращения[столбец] = temp
                массив.НомерСтраницы[столбец] = -1
                массив.Стили.push([столбец, 2, 2])
                for (let j = 0; j < РазмерБуфера; j++) {
                    if(массив.Буфер[j].БитОбращения[столбец] === 1){
                        массив.Буфер[j].БитОбращения[столбец] = 0
                        массив.Буфер[j].ВремяПоследнегоИзменения[столбец] = temp
                    }
                }
                столбец+=1
                return столбец
            }
        }
        return столбец
    }
    добавить()
    массив.ИзмененноеВремяОбращения[0] = Содержимое[0].ВремяОбращения
    массив.ВремяОбращения[0] = Содержимое[0].ВремяОбращения
    массив.НомерСтраницы[0] = Содержимое[0].НомерСтраницы
    массив.Буфер[0].ТекущаяСтраница[0] = Содержимое[0].НомерСтраницы
    массив.Буфер[0].БитОбращения[0] = 1
    массив.Буфер[0].БитИзменения[0] = Содержимое[0].Запись
    массив.Буфер[0].ВремяПоследнегоИзменения[0] = 0
    массив.Pf[0] = 1
    массив.Стили.push([0, 3, 1])
    let k = 1
    let m = 1
    for (let i = 1; i < РазмерБуфера ; i++) {
        if(КоличествоЭлементов <= m || КоличествоЭлементов === 1)
            break
        k = Сброс(k, m)
        массив.ВремяОбращения[k] = Содержимое[m].ВремяОбращения
        массив.Pf[k] = 0
        скопировать(k, m)
        массив.Стили.push([k, i*4+3, 1])
        let тест = true
        for (let j = 0; j < i ; j++) {
            if (массив.Буфер[j].ТекущаяСтраница[k] === Содержимое[m].НомерСтраницы)
                тест = false
        }
        if (тест){
            массив.Буфер[i].ТекущаяСтраница[k] = Содержимое[m].НомерСтраницы
            массив.Буфер[i].БитОбращения[k] = 1
            массив.Буфер[i].БитИзменения[k] = Содержимое[m].Запись
            массив.Буфер[i].ВремяПоследнегоИзменения[m] = 0
            массив.Pf[k] = 1
        }
    }

```



```

    } else {i-=1}
    k += 1
    m += 1
}

let PF = []
let столбец = k
let Clock = 0
k = 0
for (let i = m; i < КоличествоЭлементов; i++) {
    let круг = true
    столбец = Сброс(столбец,i)
    if(РасстояниеДоДырки<0)РасстояниеДоДырки=0
    if(круг){
        for (let j = 0; j < РазмерБуфера; j++) {
            if(массив.Буфер[j].ТекущаяСтраница[столбец-
1] === Содержимое[i].НомерСтраницы){
                скопировать(столбец,i)
                массив.Стили.push([столбец,Clock*4+3,1])
                массив.ВремяОбращения[столбец] = Содержимое[i].ВремяОбращения
                массив.НомерСтраницы[столбец] = Содержимое[i].НомерСтраницы
                массив.Буфер[j].БитОбращения[столбец] = 1
                массив.Pf[столбец] = 0
                столбец+=1
                круг = false
            }
        }
    }
}

if(круг){
    for (let j = 0; j < РазмерБуфера; j++) {
        (Clock + j < РазмерБуфера)? k = Clock + j : k = (Clock + j) -
РазмерБуфера
        скопировать(столбец,i)
        массив.Стили.push([столбец,k*4+3,1])
        if(массив.Буфер[k].БитОбращения[столбец] === 0 &&
массив.Буфер[k].БитИзменения[столбец] === 0 &&
((массив.ИзмененноеВремяОбращения[столбец] -
массив.Буфер[k].ВремяПоследнегоИзменения[столбец]) > РабочееМножество)){
            массив.ВремяОбращения[столбец] = Содержимое[i].ВремяОбращения
            массив.НомерСтраницы[столбец] = Содержимое[i].НомерСтраницы
            мас-
сив.Буфер[k].ТекущаяСтраница[столбец] = Содержимое[i].НомерСтраницы
            массив.Буфер[k].БитОбращения[столбец] = 1
            массив.Буфер[k].БитИзменения[столбец] = Содержимое[i].Запись
            круг = false
            массив.Pf[столбец] = 1
            столбец +=1
            break
        }
        столбец +=1
    }
    k + 1 < РазмерБуфера? Clock = k + 1 : Clock = 0
    let буд = -1

```

```

        if(круг){
            for (let j = 0; j < РазмерБуффера; j++) {
                (Clock + j < РазмерБуффера)? k = Clock + j : k = (Clock + j)
- РазмерБуффера
                скопировать(столбец,i)
                массив.Стили.push([столбец,k*4+3,1])
                if (буд!==-1){
                    массив.Буффер[буд].БитИзменения[столбец] = 0
                    буд = -1
                }
                if(массив.Буффер[k].БитИзменения[столбец] === 0) {
                    мас-
сив.ВремяОбращения[столбец] = Содержимое[i].ВремяОбращения
                    мас-
сив.НомерСтраницы[столбец] = Содержимое[i].НомерСтраницы
                    мас-
сив.Буффер[k].ТекущаяСтраница[столбец] = Содержимое[i].НомерСтраницы
                    массив.Буффер[k].БитОбращения[столбец] = 1
                    мас-
сив.Буффер[k].БитИзменения[столбец] = Содержимое[i].Запись
                    круг = false
                    массив.Pf[столбец] = 1
                    столбец +=1
                    break
                } else {
                    буд = k
                    массив.Стили.push([столбец,k*4+5,3])
                }
                столбец +=1
            }
        }

        k + 1 < РазмерБуффера? Clock = k + 1 : Clock = 0
        if(круг){
            РасстояниеДоДырки += Number(ВЗД)
            PF.push(столбец)
            скопировать(столбец,i)
            массив.Стили.push([столбец,Clock*4+3,1])
            массив.Стили.push([столбец,1,3])
            if (буд !== -1){
                массив.Буффер[буд].БитИзменения[столбец] = 0
                буд = -1
            }
            массив.ВремяОбращения[столбец] = Содержимое[i].ВремяОбращения
            массив.НомерСтраницы[столбец] = Содержимое[i].НомерСтраницы
            мас-
сив.Буффер[Clock].ТекущаяСтраница[столбец] = Содержимое[i].НомерСтраницы
            массив.Буффер[Clock].БитОбращения[столбец] = 1
            массив.Буффер[Clock].БитИзменения[столбец] = Содержимое[i].Запись
            массив.Pf[столбец] = 1
            столбец +=1
            Clock + 1 < РазмерБуффера? Clock = Clock + 1 : Clock = 0
        }
    }
}

```

```

    }
    for (let i = PF[0]; i < столбец-1; i++) {
        let f = 1
        for (let j = 0; j < PF.length; j++)
            if (i+1===PF[j]) f=0
        if(f)
            if(массив.ВремяОбращения[i]!==-1)
                мас-
сив.ИзмененноеВремяОбращения[i+1] = массив.ИзмененноеВремяОбращения[i] + 1
            else массив.ИзмененноеВремяОбращения[i+1] = массив.ИзмененноеВремяОбращения[i]
        else {
            мас-
сив.ИзмененноеВремяОбращения[i+1] = массив.ИзмененноеВремяОбращения[i] + Number(ВЗД)

            if(массив.ИзмененноеВремяОбращения[i] < массив.ВремяОбращения[i+1]) {
                массив.ИзмененноеВремяОбращения[i] = массив.ВремяОбращения[i+1]
                k = i-1
                while(!f){
                    if (массив.ВремяОбращения[k] !== -1)
                        мас-
сив.ИзмененноеВремяОбращения[k] = массив.ВремяОбращения[k]
                    else
                        мас-
сив.ИзмененноеВремяОбращения[k] = массив.ИзмененноеВремяОбращения[k+1]
                    k--
                    if(массив.ИзмененноеВремяОбращения[k] >= массив.ВремяОбращения[k] &&
                        массив.ВремяОбращения[k] !== -1)
                        f = 1
                }
            }
        }
        if(массив.ИзмененноеВремяОбращения[i+1] < массив.ВремяОбращения[i+1]){
            k = i+1
            while(f){
                if (массив.ВремяОбращения[k] !== -1)
                    массив.ИзмененноеВремяОбращения[k] = массив.ВремяОбращения[k]
                else
                    мас-
сив.ИзмененноеВремяОбращения[k] = массив.ИзмененноеВремяОбращения[k+1]
                k--
                if(массив.ИзмененноеВремяОбращения[k] >= массив.ВремяОбращения[k] &&
                    массив.ВремяОбращения[k] !== -1)
                    f = 0
            }
        }
    }
    return массив
}

const fifo = (РазмерБуффера, Содержимое) =>{
    let f = 0

```

```

let массив = {}
массив.НомерСтраницы = []
массив.Буффер = []
массив.Pf = []

for (let index = 1; index <= РазмерБуффера; index++){
    массив.Буффер.push([index])
массив.Pf.push(1)
for (let j = 0; j < Содержимое.length; j++){
    массив.НомерСтраницы.push(Содержимое[j].НомерСтраницы)
    for (let index = 0; index < РазмерБуффера; index++){
        массив.Буффер[index].push(-1)}
for (let index = 0; index < РазмерБуффера; index++){
    массив.Буффер[0][1]=массив.НомерСтраницы[0]

for (let i = 2; i <= Содержимое.length; i++) {
    f = 0
    for (let j = 0; j < РазмерБуффера; j++){
        if(массив.Буффер[j][i-1] === массив.НомерСтраницы[i-1]){
            f = 1
        }
    }
    if(!f){
        массив.Буффер[0][i] = массив.НомерСтраницы[i-1]
        for (let j = 1; j < РазмерБуффера; j++){
            массив.Буффер[j][i] = массив.Буффер[j-1][i-1]
        }
        массив.Pf.push(1)
    } else{
        for (let j = 0; j < РазмерБуффера; j++){
            массив.Буффер[j][i] = массив.Буффер[j][i-1]
        }
        массив.Pf.push(0)
    }
}
}
return массив
}

module.exports = {
    fifo: fifo,
    WS_Clock: WS_Clock
}

```

**default. json\*\*\*\*\***

```

{
    "port": 7437,
    "mongoUri":
    "mongodb+srv://____@cluster0.bv53p.mongodb.net/app?retryWrites=true&w=majority"
}

```

**package.json\*\*\*\*\***

```
{
  "name": "kursah",
  "version": "1.0.0",
  "description": "kysah",
  "main": "index.js",
  "scripts": {
    "start": "node index.js",
    "server": "nodemon index.js",
    "client": "npm run start --prefix client",
    "dev": "concurrently \"npm run server\" \"npm run client\""
  },
  "author": "Никита Гордеев <gordeev.2017@bk.ru>",
  "license": "ISC",
  "dependencies": {
    "bootstrap": "^4.6.0",
    "classnames": "^2.3.1",
    "config": "^3.3.6",
    "express": "^4.17.1",
    "mongoose": "^5.12.5"
  },
  "devDependencies": {
    "concurrently": "^6.0.2",
    "nodemon": "^2.0.7"
  }
}
```

\*\*\*\*\*



## Код клиентской части

**index.html** \*\*\*\*\*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>Курсовая ИТ</title>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

\*\*\*\*\*

**index.js**\*\*\*\*\*

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

\*\*\*\*\*

**App.js**\*\*\*\*\*

```
import {Rout} from './html';
import {BrowserRouter} from "react-router-dom"

function App() {
  const html = Rout()
  return (
    <BrowserRouter>
      {html}
    </BrowserRouter>
  )
}

export default App;
```

\*\*\*\*\*

**manifest.json\*\*\*\*\***

```
{
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

\*\*\*\*\*

## package.json\*\*\*\*\*

```
{
  "name": "client",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.11.4",
    "@testing-library/react": "^11.1.0",
    "@testing-library/user-event": "^12.1.10",
    "jquery": "^3.6.0",
    "materialize-css": "^1.0.0-rc.2",
    "react": "^17.0.2",
    "react-dom": "^17.0.2",
    "react-router-dom": "^5.2.0",
    "react-scripts": "4.0.3",
    "web-vitals": "^1.0.1"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "proxy": "http://localhost:7437",
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

\*\*\*\*\*

/hooks/http.hooks.js\*\*\*\*\*

```
import {useState, useCallback} from "react"

export const useHttp = () => {
  const [loading, setLoading] = useState(false)
  const [error, setError] = useState(null)

  const request = useCallback( async (url, method = "GET", body = null, headers
= {}) =>{
    setLoading(true)
    try{

      if (body) {
        body = JSON.stringify(body)
        headers["Content-Type"] = "application/json"
      }
      const response = await fetch(url, {method, body, headers})
      const data = await response.json()

      if(!response.ok){
        throw new Error(data.message || "Что-то пошло не так")
      }
      setLoading(false)
      return data
    } catch (e) {
      setLoading(false)
      setError(e.message)
      throw e
    }
  }, [])
  const clearError = useCallback( () => setError(null), [])
  return {loading, request, error, clearError}
}
```

\*\*\*\*\*

html.jsx\*\*\*\*\*

```
import {Redirect, Route} from "react-router-dom"
import React, {useEffect, useState} from 'react';
import cn from "classnames";
import {useHttp} from './hooks/http.hooks';
import materialize from "materialize-css";

export const Rout = () => {
  let [text, setText] = useState("")
  let [fifo, setfifo] = useState({Буффер: [], НомерСтраницы: [], Pf: []})
  let [ЭФ_fifo, setЭФ_fifo] = useState(0)
  let [WS_Clock, setWS_Clock] = useState({
    ИзмененноеВремяОбращения: [],
    ВремяОбращения: [], НомерСтраницы: [], Буффер: [], Pf: [], Стили: []
  })
  let [ЭФ_WS_Clock, setЭФ_WS_Clock] = useState(0)
  let t = ""
  let [Кнопка, setКнопка] = useState(0)
  let [ФайлПрочитан, setФайлПрочитан] = useState(0)
  const {loading, request, error, clearError} = useHttp()
  const [form, setForm] = useState({
    Название: "",
    РазмерБуффера: 0,
    КоличествоЭлементов: 0,
    КоличествоСтраниц: 0,
    РабочееМножество: 0,
    СбросОбращения: 0,
    ВЗД: 0,
    Содержимое: []
  })
  useEffect(() => {
    clearError()
    if (window.M && error) {
      window.M.toast({html: error})
    }
    if (error) {
      setText(error)
    }
  }, [error, clearError])

  const ОбновитьФорму = event => {
    setForm({...form, [event.target.name]: event.target.value})
  }

  const ClearForm = event => {
    document.getElementById("Название").value = ""
    document.getElementById("РазмерБуффера").value = ""
    document.getElementById("КоличествоЭлементов").value = ""
    document.getElementById("КоличествоСтраниц").value = ""
    document.getElementById("РабочееМножество").value = ""
    document.getElementById("СбросОбращения").value = ""
    document.getElementById("ВЗД").value = ""
    form.Название = ""
  }
}
```



```

    form.РазмерБуффера = 0
    form.КоличествоЭлементов = 0
    form.КоличествоСтраниц = 0
    form.РабочееМножество = 0
    form.СбросОбращения = 0
    form.ВЗД = 0
    form.Содержимое = []
    setText("")
    setfifo({Буффер: [], НомерСтраницы: [], Pf: []})
    setWS_Clock({
        ИзмененноеВремяОбращения: [],
        ВремяОбращения: [],
        НомерСтраницы: [],
        Буффер: [],
        Pf: [],
        Стили: []
    })
    setЭФ_fifo("")
    setЭФ_WS_Clock("")
}
let [КогоЗагрузить, setКогоЗагрузить] = useState(-1)
let [ОтУд, setОтУд] = useState("Удалить")
const ОбновитьSelect = event => {
    setКогоЗагрузить(event.target.value)
    setФайлПрочитан(0)
    if (event.target.value !== "0") {
        docu-
ment.getElementById("Название").value = СохранДанные[event.target.value -
1].Название
        docu-
ment.getElementById("РазмерБуффера").value = СохранДанные[event.target.value -
1].РазмерБуффера
        docu-
ment.getElementById("КоличествоЭлементов").value = СохранДанные[event.target.valu
e - 1].КоличествоЭлементов
        docu-
ment.getElementById("КоличествоСтраниц").value = СохранДанные[event.target.value
- 1].КоличествоСтраниц
        docu-
ment.getElementById("РабочееМножество").value = СохранДанные[event.target.value -
1].РабочееМножество
        docu-
ment.getElementById("СбросОбращения").value = СохранДанные[event.target.value -
1].СбросОбращения
        docu-
ment.getElementById("ВЗД").value = СохранДанные[event.target.value - 1].ВЗД
        form.Название = СохранДанные[event.target.value - 1].Название
        form.РазмерБуффера = СохранДанные[event.target.value -
1].РазмерБуффера
        form.КоличествоЭлементов = СохранДанные[event.target.value -
1].КоличествоЭлементов
        form.КоличествоСтраниц = СохранДанные[event.target.value -
1].КоличествоСтраниц

```

```

        form.РабочееМножество = СохранДанные[event.target.value -
1].РабочееМножество
        form.СбросОбращения = СохранДанные[event.target.value -
1].СбросОбращения
        form.ВЗД = СохранДанные[event.target.value - 1].ВЗД
        setОтУд("Удалить")
    } else setОтУд("Отправить")
}
let [СохранДанные, setСохранДанные] = useState([])
useEffect(() => {
    let elems = document.querySelector('select');
    elems.length = 2
    materialize.FormSelect.init(elems, materialize.options);
    for (let index = 0; index < СохранДанные.length; index++) {
        let elems = document.querySelector('select');
        el-
elems.add(new Option(`${СохранДанные[index].Название}`, `${index + 1}`))
        materialize.FormSelect.init(elems, materialize.options);
    }
    if (Кнопка === 0)
        elems.value = -1
    if (Кнопка === 1)
        elems.value = КогоЗагрузить
    if (Кнопка === 2) {
        elems.value = СохранДанные.length
        setКогоЗагрузить(СохранДанные.length)
    }
    if (Кнопка === 3) {
        setКогоЗагрузить(-1)
        elems.value = -1
    }
    materialize.FormSelect.init(elems, materialize.options);
}, [СохранДанные])
const ЧтениеДанных = async () => {
    try {
        const data = await request("/api/start/Get", "get")
        if (window.M && data.message) {
            window.M.toast({html: data.message})
        }
        setСохранДанные(data.exists)
    } catch (e) {
    }
}
let [Загрузить, setЗагрузить] = useState(true)
if (Загрузить) {
    ЧтениеДанных()
    setЗагрузить(!Загрузить)
}

const Печать = (data) => {
    t = ""
    t += "Название: "
    t += data.Название

```

```

t += "\nРазмер буфера: "
t += data.РазмерБуфера
t += "\nКоличество элементов: "
t += data.КоличествоЭлементов
t += "\nКоличество страниц: "
t += data.КоличествоСтраниц
t += "\nРабочее множество: "
t += data.РабочееМножество
t += "\nСброс обращения: "
t += data.СбросОбращения
t += "\nВремя записи: "
t += data.ВЗД
t += "\nСодержимое: ["
for (let index = 0; index < data.Содержимое.length; index++) {
    t += "\nНомер страницы: "
    t += data.Содержимое[index].НомерСтраницы
    t += " Время обращения: "
    t += data.Содержимое[index].ВремяОбращения
    t += " Запись: "
    t += data.Содержимое[index].Запись
}
t += "\n]"
setText(t)
}

const Проверка = () => {
    if (form.Название.length < 3)
        return "Название должно быть > 2 символов"
    if (form.РазмерБуфера < 1)
        return "Размер Буфера должен быть > 0"
    if (form.КоличествоЭлементов < 1)
        return "Количество элементов должно быть > 0"
    if (form.КоличествоСтраниц < 1)
        return "Количество страниц должно быть > 0"
    if (form.РабочееМножество < 1)
        return "Рабочее множество должно быть > 0"
    if (form.СбросОбращения < 2)
        return "Время сброса обращения должно быть > 1"
    if (form.ВЗД < 0)
        return "Время записи на диск должно быть >= 0"
    return 0
}

const ЗагрузитьДанные = async () => {
    try {
        if (КогоЗагрузить > 0) {
            const data = await request("/api/start/getOne", "POST", СохранДан
ные[КогоЗагрузить - 1])
            if (window.M && data.message) {
                window.M.toast({html: data.message})
            }
            data.fifo = await JSON.parse(data.fifo)
            data.WS_Clock = await JSON.parse(data.WS_Clock)
            setКнопка(1)
            Печать(data.source)
        }
    }
}

```

```

setfifo({Буффер: [], НомерСтраницы: [], Pf: []})
setWS_Clock({
    ИзмененноеВремяОбращения: [],
    ВремяОбращения: [],
    НомерСтраницы: [],
    Буффер: [],
    Pf: [],
    Стили: []
})
class(data.WS_Clock.Стили, data.WS_Clock.Pf.length -
1, data.WS_Clock.Буффер.length * 4 + 3)
setfifo(data.fifo)
t = 0
for (let i = 0; i < data.fifo.Pf.length; i++) {
    t += data.fifo.Pf[i]
}
setЭФ_fifo(t)
t = 0
for (let i = 0; i < data.WS_Clock.Pf.length; i++) {
    if (data.WS_Clock.Pf[i] !== -1)
        t += data.WS_Clock.Pf[i]
}
setЭФ_WS_Clock(t)
setЗагрузить(!Загрузить)
setWS_Clock(data.WS_Clock)

} else if (КогоЗагрузить === "0") {
    var input = document.createElement('input')
    input.type = 'file';
    input.onChange = e => {
        var file = e.target.files[0]
        var reader = new FileReader()
        reader.readAsText(file, 'UTF-8')
        reader.onload = async readerEvent => {
            var content = readerEvent.target.result
            content = await JSON.parse(content)
            docu-
ment.getElementById("РазмерБуффера").value = content[0][0]
            docu-
ment.getElementById("КоличествоЭлементов").value = content[0][1]
            docu-
ment.getElementById("КоличествоСтраниц").value = content[0][2]
            docu-
ment.getElementById("РабочееМножество").value = content[0][3]
            docu-
ment.getElementById("СбросОбращения").value = content[0][4]
            document.getElementById("ВЗД").value = content[0][5]
            form.РазмерБуффера = content[0][0]
            form.КоличествоЭлементов = content[0][1]
            form.КоличествоСтраниц = content[0][2]
            form.РабочееМножество = content[0][3]
            form.СбросОбращения = content[0][4]
            form.ВЗД = content[0][5]
            form.Содержимое = []

```

```

        for (let i = 0; i < content[1].length; i++) {
            let Обращение = {}
            Обращение.НомерСтраницы = content[1][i][0]
            Обращение.ВремяОбращения = content[1][i][1]
            Обращение.Запись = content[1][i][2]
            form.Содержимое.push(Обращение)
        }
        Печать(form)
        setФайлПрочитан(1)
    }
}
input.click()
}
} catch (e) {
    console.log(e)
}
}
useEffect(()=>{console.log(Кнопка,КогоЗагрузить,(!!СохранДанные[КогоЗагрузить
- 1]))},[Кнопка])
const Сгенерировать = async () => {
    try {
        if (!Проверка()) {
            if (КогоЗагрузить > 0) {
                console.log(Кнопка)
                form.Содержимое = await request("/api/start/Gen", "POST", {...
.form}))

                form.Содержимое = form.Содержимое.Содержимое
            }
            const data = await request("/api/start/Add", "POST", {...form})
            if (window.M && data.message) {
                window.M.toast({html: data.message})
            }
            setЗагрузить(!Загрузить)
            data.fifo = await JSON.parse(data.fifo)
            data.WS_Clock = await JSON.parse(data.WS_Clock)
            Печать(form)
            setfifo({Буффер: [], НомерСтраницы: [], Pf: []})
            setWS_Clock({ИзмененноеВремяОбращения: [], ВремяОбращения: [], Но
мерСтраницы: [], Буффер: [], Pf: []})
            clacc(data.WS_Clock.Стили, data.WS_Clock.Pf.length, data.WS_Clock
.Буффер.length * 4 + 3)
            setfifo(data.fifo)
            setWS_Clock(data.WS_Clock)
            t = 0
            for (let i = 0; i < data.fifo.Pf.length; i++) {
                t += data.fifo.Pf[i]
            }
            setЭФ_fifo(t)
            t = 0
            for (let i = 0; i < data.WS_Clock.Pf.length; i++) {
                if (data.WS_Clock.Pf[i] !== -1)
                    t += data.WS_Clock.Pf[i]
            }
            setЭФ_WS_Clock(t)

```

```

        setОтУд("Удалить")
        setКнопка(2)
      } else (setText(Проверка()))
    } catch (e) {
      console.log(e)
    }
  }

const УдалениеДанных = async () => {
  try {
    if (КогоЗагрузить > 0) {
      const data = await request("/api/start/Delete", "delete", СохранД
анные[КогоЗагрузить - 1])
      if (window.M && data.message) {
        window.M.toast({html: data.message})
      }
      t = data.message
      setКнопка(3)
      setText(t)
      setЗагрузить(!Загрузить)
      setfifo({Буффер: [], НомерСтраницы: [], Pf: []})
      setWS_Clock({
        ИзмененноеВремяОбращения: [],
        ВремяОбраще-
ния: [], НомерСтраницы: [], Буффер: [], Pf: [], Стили: []
      })
      setЭФ_fifo(0)
      setЭФ_WS_Clock(0)
      if (data.message)
        setText(data.message)
    } else if (КогоЗагрузить === "0") {
      setКнопка(4)
      Сгенерировать()
    }
  } catch (e) {
  }
}

let [Clacc, setClacc] = useState("")
const clacc = async (Стили, x, y) => {
  t = Array.from(Array(x), () => new Array(y))

  for (let i = 0; i < Стили.length; i++) {
    t[Стили[i][0]][Стили[i][1]] = Стили[i][2]
  }
  setClacc(t)
}

return (
  <div>
    <Route path="/" exact>
      <div className="vertical">
        <div style={{height: 20}}/>
        <div className="item">
          <div className="horizontal ">
            <div className="item zag3 zag4">

```

```

<div className="centr catr">
  <div className="card blue-grey darken-
1" style={{margin: 0}}>
    <div className="card-content white-text">
      <span className="card-
title cen zag1">Загрузить данные</span>
      <div style={{height: 5}}/>
      <div className="horizontal input-
field">
        <div className="item">
          <div className="btn zag4 notB
or t">
            <select onChange={Обновит
ьSelect} className="materialSelect"
              id="myDropdown">
                <option value="-
1" disabled defaultValue="Откуда?"
                  name="Названи
е">ОТКУДА?
                </option>
                <optgroup label="Из ф
айла">
                  <op-
tion value="0">Из файла</option>
                </optgroup>
                <optgroup label="с се
рвера">
                  </optgroup>
                </select>
              </div>
            </div>
            <div style={{width: 10}}/>
            <button className="waves-
effect waves-light btn item zag4"
              onClick={ЗагрузитьДанные}
              disabled={!((КогоЗагрузит
ь !== -1) * !loading)}
            >Загрузить
            </button>
          </div>
          <div style={{height: 30}}/>
          <span className="card-
title cen zag2">Сгенерировать новые</span>
          <div style={{height: 5}}/>
          <div>
            <div className="input-field">
              <div className="horizontal">
                <p className="cen "
                  style={{width: 200}}
                >Название</p>
                <input placeholder="Должн
о быть уникальным"

```



```

}}
style={{height: 30

id="Название"
name="Название"
type="text"
class-

Name="validate cen"
onChange={Обновить
Форму}}/>
<div style={{width: 20}}/
>
<button className="waves-
effect waves-light btn zag4 ing"
on-
Click={ClearForm}}/>
</div>
<div className="horizontal">
<p className="cen"
style={{width: 350}}
>Размер буфера</p>
<input placeholder=" > 0"
id="РазмерБуфера"
name="РазмерБуфер
a"
type="Number"
class-
Name="validate rig"
onChange={Обновить
Форму}}/>
</div>
<div className="horizontal">
<p className="cen"
style={{width: 350}}
>Количество элементов</p>
<input placeholder=" > 0
"
id="КоличествоЭлем
ентов"
name="КоличествоЭл
ементов"
type="Number"
class-
Name="validate rig"
onChange={Обновить
Форму}}/>
</div>
<div className="horizontal">
<p className="cen"
style={{width: 350}}

```

```

<input type="text" value="0" />Количество страниц</p>

<input placeholder=" " value="0" />
id="КоличествоСтраниц"
name="КоличествоСтраниц"
type="Number"
class="form-control"
Name="validate rig"
onChange={ОбновитьФорму}/>
</div>

<div className="horizontal">
  <p className="center" style={{width: 350px}}>
    >Рабочее множество</p>

    <input placeholder=" " value="0" />
id="РабочееМножество"
name="РабочееМножество"
type="Number"
class="form-control"
Name="validate rig"
onChange={ОбновитьФорму}/>
</div>

<div className="horizontal">
  <p className="center" style={{width: 350px}}>
    >Время сброса обращения</p>

    <input placeholder=" " value="1" />
id="СбросОбращения"
name="СбросОбращения"
type="Number"
class="form-control"
Name="validate rig"
onChange={ОбновитьФорму}/>
</div>

<div className="horizontal">
  <p className="center" style={{width: 350px}}>
    >Время сброса обращения</p>

```

```

                                style={{width: 350}}
                                >Время записи на диск</p>

                                <in-

put placeholder=" >= 0 "

                                id="ВЗД"
                                name="ВЗД"
                                type="Number"
                                class-

Name="validate rig"

                                onChange={Обновить
Форму}/>

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

                                <div className=" horizontal card-action">
                                <button className="waves-
effect waves-light btn item zag4"

                                style={{}}
                                onClick={Сгенерировать}
                                disabled={loading}
                                >Сгенерировать
                                </button>
                                <div style={{width: 10}}/>
                                <button className="waves-
effect waves-light btn item zag4"

                                style={{}}
                                onClick={УдалениеДанных}
                                disa-
bled={!(((КогоЗагрузить > 0) + (КогоЗагрузить === "0") * ФайлПрочитан) * !loading
)}

                                >{ОтУд}</button>
                                </div>
                                </div>
                                </div>
                                </div>
                                <div className="item">
                                <div className="scrol centr text becc">
                                <pre>{text}</pre>
                                </div>
                                </div>
                                </div>
                                </div>

                                <div className="item ots">
                                <h2>fifo</h2>
                                <div className="tabl becc">
                                <table className="responsive-table1">
                                <thead>
                                <tr>
                                <th>N:</th>
                                {fifo.Буффер.map((Буффер, index) => (

```

```

<th key={`_${index}_${Буффер.length * Math
.random()}`}>{Буффер[0]}</th>
    )))
    <th>PF:</th>
</tr>
</thead>
<tbody>
{fifo.НомерСтраницы.map((НомерСтраницы, index) =>
(
    <tr key={index}>
        <td>{НомерСтраницы}</td>
        {fifo.Буффер.map((Буффер, key) => (
            <td key={`_${key}_${Буффер.length * Ma
th.random()}`}>{Буффер[index + 1] !== -1 ? Буффер[index + 1] : "."} </td>)))}
        <td>{fifo.Pf[index]}</td>
    </tr>)))}
</tbody>
</table>
</div>
<h4>NPF = {ЭФ_fifo}</h4>
<h2>WS_Clock</h2>
<div className="tabl becc">
    <table className="responsive-table1">
        <thead>
            <tr>
                <th>ИВО:</th>
                <th>ВО:</th>
                <th>ОП:</th>
                {WS_Clock.Буффер.map((t, i) => (
                    <tr key={`_${t}_${i * Math.random()}`} sty
le={{padding: 0, border: 0}}>
                        <th>{i}</th>
                        <th>ВО</th>
                        <th>БИ</th>
                        <th>ВПИ</th>
                    </tr>
                )))}
                <th>PF:</th>
            </tr>
        </thead>
        <tbody>
{WS_Clock.НомерСтраницы.map((НомерСтраницы, index
) => (
            <tr key={`_${index}_${НомерСтраницы * Math.ran
dom()}`}>
                <td>{WS_Clock.ИзмененноеВремяОбращения[in
dex] !== -1 ? WS_Clock.ИзмененноеВремяОбращения[index] : "."}</td>
                <td className={cn({
                    ["green"]: Clacc[index][1] === 3
                })}>{WS_Clock.ВремяОбращения[index] !== -
1 ? WS_Clock.ВремяОбращения[index] : "."}</td>
                <td className={cn({

```

```

        ["yellow"]: Clacc[index][2] === 2,
    }}>{НомерСтраницы !== -
1 ? НомерСтраницы : "."}</td>

        {WS_Clock.Буффер.map((Буффер, y) => (
            <td key={`_${y}_${Буффер.length * Math
.random()}`} style={{padding: 0, border: 0}}>
                <td className={cn({
                    ["red jir"]: Clacc[index][y *
4 + 3] === 1,
                    ["jir"]: Clacc[index][y * 4 +
3] !== 1,
                })}>{Буф-
фер.ТекущаяСтраница[index] !== -1 ? Буффер.ТекущаяСтраница[index] : "."} </td>
                <td>{Буффер.ТекущаяСтраница[index
] !== -1 ? Буффер.БитОбращения[index] : "."} </td>
                <td className={cn({
                    ["green"]: Clacc[index][y * 4
+ 5] === 3,
                })}>{Буффер.ТекущаяСтраница[index
] !== -1 ? Буффер.БитИзменения[index] : "."} </td>
                <td>{Буффер.ТекущаяСтраница[index
] !== -1 ? Буффер.ВремяПоследнегоИзменения[index] : "."} </td>
                </td>))}

            <td className="">{WS_Clock.Pf[index] !==
-1 ? WS_Clock.Pf[index] : "."}</td>
        </tr>)))}
    </tbody>
</table>
</div>
<h4>NPF = {ЭФ_WS_Clock}</h4>
<div style={{width: 20}}/>
</div>
<div style={{height: 50}}/>
</div>
</Route>
<Redirect to="/" />
</div>
)
}

```

\*\*\*\*\*

## index.css\*\*\*\*\*

```
@import "~materialize-css/dist/css/materialize.min.css";

.item {
    flex: 1 1 200px;
}
.ots {
    margin-left: 20px;
    margin-right: 20px
}

.horizontal {
    display: flex;
}
body {
    background-image: url(./фон.jpg) ;
    background-position: center ;
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-size: cover;
}
.becc{
    background-color: rgba(95, 136, 175, 0.8);
}

h2, h4 {
    color: #fff;
}

.vertical{
    display: flex;
    flex-flow: column nowrap;
}

.scrol{
    height: 620px;
    overflow-y:auto;
    width: 450px;
}

.centr{
    margin-left: auto;
    margin-right: auto;
}
.catr{
    width: 450px;
}

.cen{
    display: flex;
    align-items: center ;
    text-align:center;
}
```

```

.zag2{
    font-size: 31pt!important;
}
.zag3 input{
    font-size: 15pt!important;
}
.zag4 {
    font-size: 15pt!important;
    height: 45px;
}
.notBor input{
    border-bottom: 0!important;
    color: #fff;
    padding: 0;
    margin-left: auto!important;
    margin-right: auto!important;
    align-items: center!important;
    text-align:center!important;
}
.select-dropdown li.disabled, .select-dropdown li.disabled>span, .select-
dropdown li.optgroup {
    color: rgba(0, 0, 0, 1)!important;
    background-color: transparent!important;
    width: 100%!important;
}
.zag1{
    font-size: 35pt!important;
}
.rig {
    text-align:center;
    width: 80px!important;
    height: 30px!important;
}
.btn, .btn-large, .btn-small, .btn-flat {
    padding: 0!important;
}
.ing{
    background-image: url("../krest.png");
    height: 30px;
    width: 50px
}
.tabl {
    max-height: 500px;
    border:2px solid rgba(0, 0, 0, 1);
    overflow-y:auto;
    overflow-x:auto;
    width:100%!important;
}
.text{
    border:2px solid rgba(0, 0, 0, 1);
    color:rgb(255, 255, 255);
}

```



```

}
.yellow{
    background-color: rgb(255, 255, 0);;
}
.green{
    background-color: rgb(0, 255, 0);;
}
.red{
    background-color: rgb(255, 0, 0);;
}
.jir {font-weight:bolder}
table.responsive-table1{width:100%;display:block;position:relative}
table.responsive-table1 td:empty:before{content:'\00a0'}
table.responsive-table1 th,table.responsive-table td{margin:0;vertical-align:top}
table.responsive-table1 th{text-
align:left; padding:0; color: rgb(255, 255, 255);}
table.responsive-table1 thead{display:block;float:left ;}
table.responsive-table1 thead tr{display:block;}
table.responsive-table1 thead tr th::before{content:"\00a0"}
table.responsive-table1 tbody{display:block;width:auto;position:relative;white-
space:nowrap}
table.responsive-table1 tbody tr{display:inline-block;vertical-align:top}
table.responsive-table1 th{display:block;text-
align:right;border:1px solid rgb(0, 0, 0);padding:5px}
table.responsive-table1 td{display:block;min-height:1.25em;text-
align:left;color: rgb(255, 255, 255); padding:5px;border:1px solid rgb(0, 0, 0)}

```

\*\*\*\*\*

## **Вывод**

Написал клиент-серверное приложение с помощью следующего стека технологий Node.js, Express, MongoDB, React.

Разработал CRUD API (соблюдены принципы REST). Есть возможность добавления данных, их чтения (получения), удаления данных.