Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

**Лабораторная работа №7**

**по дисциплине**

**«Графический и веб-дизайн»**

**ХУКИ В REACTJS**

**Выполнил**:

ст. гр. ПРИ-120

Д. А. Грачев

**Принял**:

Шамышев А. А.

Владимир, 2023

ЦЕЛЬ РАБОТЫ

Познакомиться с хуками в библиотеке ReactJS. Научиться создавать React компоненты с внутренним состоянием и сложным поведением

ВЫПОЛНЕНИЕ РАБОТЫ

1. Был создан сервер с помощью фреймворка Express.js
   1. Листинг

*const* goods = require("./data.json");  
*const* express = require("express");  
*const* app = express();  
*const* PORT = 3001;  
  
app.get("/", (request, response) => {  
 *const* page = parseInt(request.query.page);  
 *const* count = parseInt(request.query.count);  
 response.set("Access-Control-Allow-Origin", "http://localhost:5173");  
 *const* responseBody = {  
 paging: {  
 page: page,  
 rowPerPage: count,  
 total: goods.length,  
 },  
 };  
  
 *if* (page && count) {  
 *const* firstIndex = (page - 1) \* count;  
 *const* endIndex = firstIndex + count;  
 responseBody.items = goods.slice(firstIndex, endIndex);  
 response.send(responseBody);  
 } *else* {  
 response.status(400).send("Bad Request");  
 }  
});  
  
app.listen(PORT, () => {  
 console.log(`Back listening on port ${PORT}`);  
});

* 1. Часть данных в виде json

{  
 "id": 12,  
 "name": "Домашняя аудиосистема LG ON88",  
 "manufactureDate": "2021-07-21",  
 "price": 39999  
},  
{  
 "id": 13,  
 "name": "16.2\" Ноутбук Apple MacBook Pro серый",  
 "manufactureDate": "2021-12-12",  
 "price": 319999  
},

* 1. Описание работы: у сервера имеется один эндпоинт, который возвращает список товаров, в него нужно передать страницу и количество элементов на странице, для организации пагинации; данные содержат в себе идентификатор, название, дату производства и цену

1. Был написан хук для получения товаров с сервера
   1. Листинг

*export const* useServerGoods = (initialPage = 1, initialCount = 10) => {  
 *const* [loading, setLoading] = useState(*false*)  
 *const* [page, setPage] = useState(initialPage)  
 *const* [count, setCount] = useState(initialCount)  
 *const* [error, setError] = useState(*false*)  
 *const* [data, setData] = useState<Goods | *null*>(*null*)  
  
 useEffect(() => {  
 setLoading(*true*)  
 fetch(`http://localhost:3001/?page=${page}&count=${count}`)  
 .then((response) => response.json())  
 .then((data) => setData(data))  
 .catch(() => setError(*true*))  
 .finally(() => setLoading(*false*))  
 }, [page, count])  
  
 *return* {  
 loading,  
 data,  
 error,  
 page,  
 count,  
 setPage,  
 setCount,  
 }  
}

* 1. Описание работы: с помощью fetch API получается списко товаров по переданным страницам и количестве элементов, так же организована работа с загрузкой для отображения состояния загрузки в браузере

1. Было реализовано клиентское приложение для отображения данных с сервера о товарах
   1. Листинг

*import* { useServerGoods } *from* './hooks/useServerGoods'  
*import* { DataGrid, GridColDef, GridPaginationModel } *from* '@mui/x-data-grid'  
*import* { useState } *from* 'react'  
  
*function* App() {  
 *const* { error, loading, data, page, count, setCount, setPage } =  
 useServerGoods()  
  
 *const* [columns] = useState<GridColDef[]>([  
 { field: 'id', headerName: 'ID', width: 20 },  
 { field: 'name', headerName: 'Название', width: 500 },  
 {  
 field: 'manufactureDate',  
 headerName: 'Дата производства',  
 width: 200,  
 },  
 { field: 'price', headerName: 'Цена', width: 200 },  
 ])  
  
 *const* handlePageChange = (model: GridPaginationModel) => {  
 setCount(model.pageSize)  
 setPage(model.page + 1)  
 }  
  
 *if* (loading) {  
 *return* <h2>Загрузка</h2>  
 }  
  
 *if* (error) {  
 *return* <h2>Ошибка соединения</h2>  
 }  
  
 *return* (  
 <div style={{ width: 960 }}>  
 {data && (  
 <DataGrid  
 columns={columns}  
 rows={data.items}  
 autoHeight  
 initialState={{  
 pagination: {  
 paginationModel: {  
 pageSize: 10,  
 page: 0,  
 },  
 },  
 }}  
 paginationMode={'server'}  
 paginationModel={{  
 page: page - 1,  
 pageSize: count,  
 }}  
 rowCount={data.paging.total}  
 pageSizeOptions={[5, 10, 25]}  
 onPaginationModelChange={handlePageChange}  
 />  
 )}  
 </div>  
 )  
}  
  
*export default* App

* 1. Описание работы: с помощью хука получается список товаров, далее с помощью библиотеки MUI-X отображается таблица с данными, полученными с сервера, в течении загрузки и в случае ошибки будет отображаться соответствующая надпись

1. В результате работы приложения – в браузере отображается таблица с данными о товарах с сервера

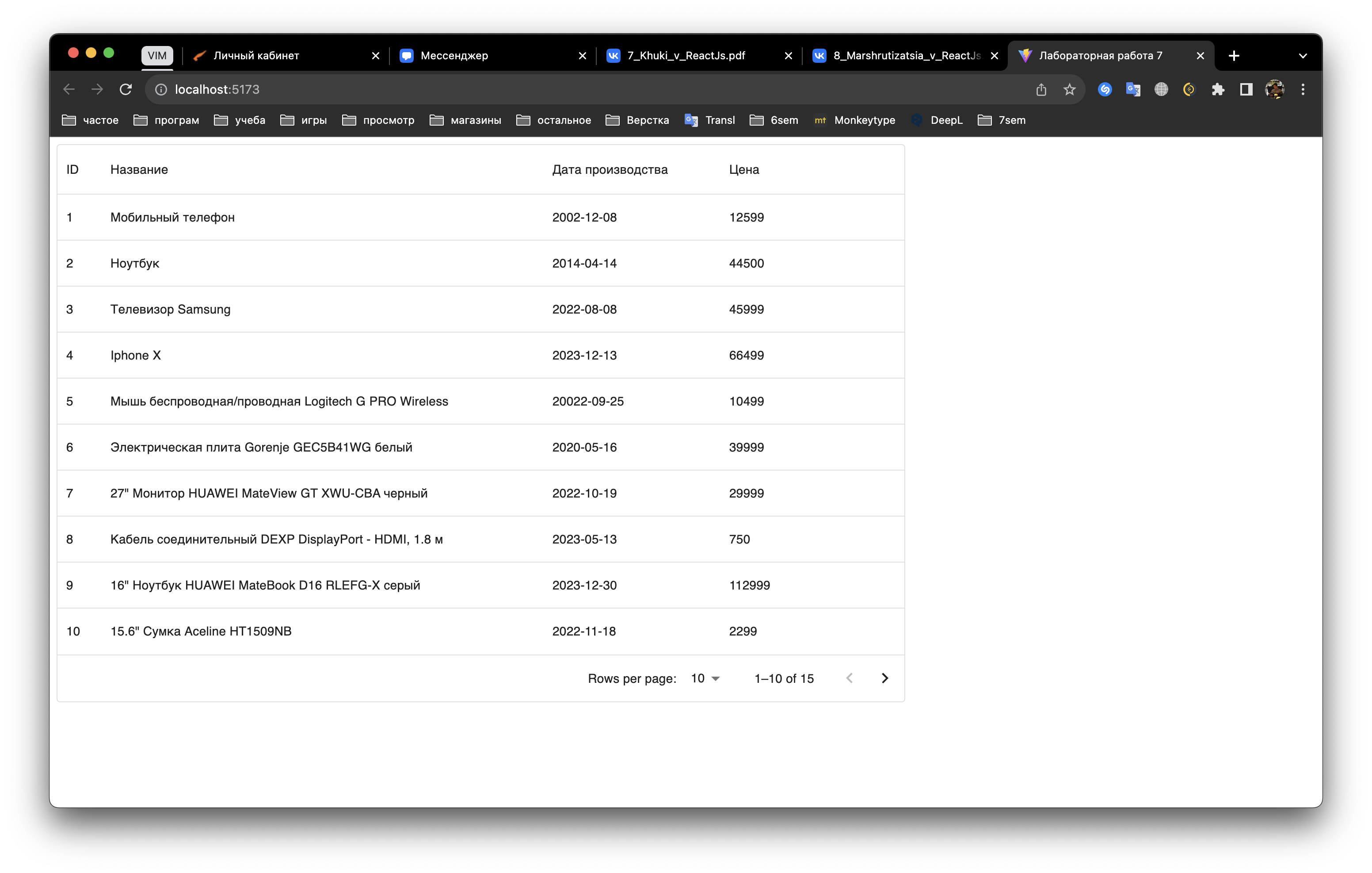


Рисунок . Результат работы приложения

ВЫВОД

В ходе выполнения работы познакомился с хуками в библиотеке ReactJS, научился создавать React компоненты с внутренним состоянием и сложным поведением.