|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ**

**Отчет по лабораторной работе № 6**

**по дисциплине Программирование сетевых приложений**

#### Тема работы: "Реализация собственного API на Node.js, подключение к интерфейсу."

**Выполнил:**

Студент группы ИУ5Ц-54Б

Перевощиков Н.Д.

(дата, подпись)

**Проверила:**

Преподаватель

Аксёнова М.В.

(дата, подпись)

Москва, 2023

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ ОТЧЕТА**

[1. Цель лабораторной работы 3](#_Toc115088672)

[2. Описание задания 3](#_Toc115088673)

[3. Текст программы 4](#_Toc115088674)

[4. Результат 25](#_Toc115088675)

## Цель лабораторной работы

## Описание задания

1. Что такое Node.js?
2. Возможности Node.js и способы применения в production-среде
3. Небольшие примеры скриптов
4. Пишем "голую" простую API
5. Пишем API на express
6. Добавляем в проект структуру (рефакторим код)
7. Добавляем внешний клиент (то самое BFF)
8. Что еще умеет Node.js

## Текст программы

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Simple App</title>

    <link rel="stylesheet" href="node\_modules/bootstrap/dist/css/bootstrap.min.css">

    <link rel="stylesheet" type="text/css" href="main/styles.css" />

</head>

<body>

<div id="root"></div>

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

<script src="node\_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>

</body>

</html>

**Main.js**

import {MainPage} from "./main/index.js";

const root = document.getElementById('root');

const mainPage = new MainPage(root);

mainPage.render();

**Server.js**

const express = require('express');

const stocks = require('./internal/stocks');

const app = express();

const host = 'localhost';

const port = 8000;

app.use(express.json());

app.use('/stocks', stocks);

app.listen(port, host, () => {

    console.log(`Сервер запущен по адресу http://${host}:${port}`);

});

**Папка components**

**Папка back-button**

**Index.js**

export class BackButtonComponent {

    constructor(parent) {

        this.parent = parent;

    }

    addListeners(listener) {

        document

            .getElementById("back-button")

            .addEventListener("click", listener)

    }

    getHTML() {

        return (

            `

                <button id="back-button" class="btn btn-primary" type="button">Назад</button>

            `

        )

    }

    render(listener) {

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(listener)

    }

}

**Папка button**

**Add.js**

export class AddButtonComponent {

    constructor(parent) {

        this.parent = parent;

    }

    addListeners(listener) {

        document

            .getElementById("add-button")

            .addEventListener("click", listener)

    }

    getHTML() {

        return (

            `

                <button id="add-button" class="btn btn-primary" type="button">Добавить</button>

            `

        )

    }

    render(listener) {

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(listener)

    }

}

**Delete.js**

export class DeleteButtonComponent {

    constructor(parent) {

        this.parent = parent;

    }

    addListeners(listener) {

        document

            .getElementById("delete-button")

            .addEventListener("click", listener)

    }

    getHTML() {

        return (

            `

                <button id="delete-button" class="btn btn-primary" type="button">Удалить</button>

            `

        )

    }

    render(listener) {

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(listener)

    }

}

**Index.js**

export class ButtonComponent {

    constructor(parent) {

        this.parent = parent;

    }

    render() {

        this.parent.insertAdjacentHTML('beforeend', '<button type="button" class="btn btn-primary">Hello world 4!</button>');

    }

}

**Update.js**

export class UpdateButtonComponent {

    constructor(parent) {

        this.parent = parent;

    }

    addListeners(listener) {

        document

            .getElementById("update-button")

            .addEventListener("click", listener)

    }

    getHTML() {

        return (

            `

                <button id="update-button" class="btn btn-primary" type="button">Обновить</button>

            `

        )

    }

    render(listener) {

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(listener)

    }

}

**Папка input**

**Input.js**

export class InputComponent {

    constructor(parent) {

        this.parent = parent;

    }

    addListeners(listener) {

        document

            .getElementById("input-component")

    }

    getHTML() {

        return (

            `

                <input id="input-component" class="btn btn-primary" type="input">

            `

        )

    }

    render(listener) {

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(listener)

    }

}

**Папка product**

**Index.js**

export class ProductComponent {

    constructor(parent) {

        this.parent = parent

        console.log('project/components/product/index.js; ProductComponent, this.parent:', this.parent)

    }

    getHTML(data) {

        return (

            `

                <div class="card mb-3" style="width: 540px;">

                    <div class="row g-0">

                        <div class="col-md-4">

                            <img src="${data.src}" class="img-fluid" alt="картинка">

                        </div>

                        <div class="col-md-8">

                            <div class="card-body">

                                <h5 class="card-title">${data.title}</h5>

                                <p class="card-text">${data.text}</p>

                            </div>

                        </div>

                    </div>

                </div>

            `

        )

    }

    render(data) {

        const html = this.getHTML(data)

        this.parent.insertAdjacentHTML('beforeend', html)

    }

}

**Папка product-card**

**Index.js**

export class ProductCardComponent {

    constructor(parent) {

        this.parent = parent;

    }

    getHTML(data) {

        return (

            `

            <div class="card" id="id-card-block-${data.id}" style="width: 300px;">

                <img class="card-img-top" src="${data.src}" alt="картинка">

                <div class="card-body">

                    <h5 class="card-title">${data.title}</h5>

                    <p class="card-text">${data.text}</p>

                    <button class="btn btn-primary" id="click-card-${data.id}" data-id="${data.id}">Купить</button>

                </div>

                <div id="liveAlertPlaceholder-${data.id}"></div>

            </div>

            `

        )

    }

    addListeners(data, listener) {

        // debugger

        document

            .getElementById(`click-card-${data.id}`)

            .addEventListener("click", listener)

    }

    render(data, listener) {

        const html = this.getHTML(data)

        this.parent.insertAdjacentHTML('beforeend', html)

        this.addListeners(data, listener)

    }

}

**Папка db**

**Stocks.json**

[{

    "id":1,

    "src":"milk1.jpg",

    "title":"Молоко",

    "text":"Состав на 100 г продукта Энергетическая ценность: 60 ккал  Вода: 88 г Белки: 3,2 г  Жиры: 3,25 г Углеводы: 5,2 г"

}, {

    "id": 2,

    "src": "myaso1.jpg",

    "title": "Мясо",

    "text": "Состав на 100 г продукта Энергетическая ценность: 218 ккал  Вода: 60,1 г Белки: 18,6 г  Жиры: 16,0 г Углеводы: 0 г"

}, {

    "id": 3,

    "src": "ryby1.jpg",

    "title": "Рыба",

    "text": "Состав на 100 г продукта Энергетическая ценность: 148 ккал  Вода: 71,42 г Белки: 20,77 г  Жиры: 6,61 г Углеводы: 0 г"

}, {

    "id": 4,

    "src": "http://klublady.ru/uploads/posts/2022-02/thumbs/1644704305\_10-klublady-ru-p-pechenki-s-shokoladom-foto-10.jpg",

    "title": "Печенье",

    "text": "Такой акции вы еще не видели"

}]

**Папка Internal**

**Папка stocks**

**Index.js**

const express = require('express');

const {StocksController} = require('./StocksController');

const router = express.Router();

router.get('/', StocksController.findStocks);

router.get('/:id', StocksController.findStockById);

router.post('/', StocksController.addStock);

router.delete('/:id', StocksController.deleteStock);

module.exports = router;

**StocksController.js**

const {StocksService} = require('./StocksService');

class StocksController {

    static findStocks(req, res) {

        try {

            res.send(StocksService.findStocks());

        } catch (err) {

            res.status(400).send({status: 'Bad Request', message: err.message})

        }

    }

    static findStockById(req, res) {

        try {

            const id = Number.parseInt(req.params.id);

            res.send(StocksService.findStocks(id))

        } catch (err) {

            res.status(400).send({status: 'Bad Request', message: err.message})

        }

    }

    static addStock(req, res) {

        try {

            res.send(StocksService.addStock(req.body));

        } catch (err) {

            res.status(400).send({status: 'Bad Request', message: err.message})

        }

    }

    static deleteStock(req, res) {

        try {

            const id = Number.parseInt(req.params.id);

            res.send(StocksService.deleteStock(id));

        } catch (err) {

            res.status(400).send({status: 'Bad Request', message: err.message})

        }

    }

}

module.exports = {

    StocksController,

};

**StocksDAO.js**

const {StocksRepository} = require('./StocksRespository');

class StockDAO {

    constructor(id, src, title, text) {

        this.id = id;

        this.src = src;

        this.title = title;

        this.text = text;

    }

    static \_validateId(id) {

        const numberId = Number.parseInt(id);

        if (Number.isNaN(numberId)) {

            throw new Error('invalidate id');

        }

    }

    static \_validate(stock) {

        if (

            stock.id === undefined ||

            stock.src === undefined ||

            stock.title === undefined ||

            stock.text === undefined

        ) {

            throw new Error('invalidate stock data');

        }

        this.\_validateId(stock.id);

    }

    static find() {

        const stocks = StocksRepository.read();

        return stocks.map(({id, src, title, text}) => {

            return new this(id, src, title, text);

        });

    }

    static findById(id) {

        this.\_validateId(id);

        const stocks = StocksRepository.read();

        const stock = stocks.find((s) => s.id === id);

        return new this(stock.id, stock.src, stock.title, stock.text);

    }

    static insert(stock) {

        this.\_validate(stock);

        const stocks = StocksRepository.read();

        StocksRepository.write([...stocks, stock]);

        return new this(stock.id, stock.src, stock.title, stock.text);

    }

    static delete(id) {

        this.\_validateId(id);

        const stocks = StocksRepository.read();

        const filteredStocks = stocks.filter((s) => s.id !== id);

        StocksRepository.write(filteredStocks);

        return filteredStocks.map(({id, src, title, text}) => {

            return new this(id, src, title, text);

        });

    }

    toJSON() {

        return {

            id: this.id,

            src: this.src,

            title: this.title,

            text: this.text,

        }

    }

}

module.exports = {

    StockDAO,

}

**StocksRespository.js**

const {DBConnector} = require('../../modules/DBConnector');

class StocksRepository {

    static db = new DBConnector('stocks.json');

    static read() {

        const file = this.db.readFile();

        return JSON.parse(file);

    }

    static write(json) {

        this.db.writeFile(JSON.stringify(json));

    }

}

module.exports = {

    StocksRepository,

}

**StocksService.js**

const {StockDAO} = require('./StocksDAO');

class StocksService {

    static findStocks(id) {

        if (id !== undefined) {

            return StockDAO.findById(id).toJSON();

        }

        return StockDAO.find().map((stock) => stock.toJSON());

    }

    static addStock(stock) {

        return StockDAO.insert(stock).toJSON();

    }

    static deleteStock(id) {

        return StockDAO.delete(id).map((stock) => stock.toJSON());

    }

}

module.exports = {

    StocksService,

}

**Папка main**

**Папка page**

**AddPage.js**

import {ProductCardComponent} from "../../components/product-card/index.js";

import {ProductPage} from "./CardPage.js";

import {ajax} from "../../modules/ajax.js";

export class AddPage {

    constructor(parent, data, max\_length) {

        this.parent = parent;

        this.data = data;

        this.max\_length = max\_length;

        console.log(this.data)

        console.log(this.length)

    }

    get pageRoot() {

        return document.getElementById('tovars-page')

    }

    getHTML() {

        return (

            `

                <div id="tovars-page" class="d-flex flex-wrap">

                    <input id="input-component1" class="btn btn-primary" type="text" placeholder="ссылка на изображение" value="">

                    <input id="input-component2" class="btn btn-primary" type="text" placeholder="название продукта" value="">

                    <input id="input-component3" class="btn btn-primary" type="text" placeholder="описание" value="">

                    <button id="simply\_button" class="btn btn-primary">Сохранить</button>

                <div/>

            `

        )

    }

    clickCard(e) {

        // debugger

        const cardId = e.target.dataset.id

        console.log('project/main/index.js; clickCard, this.parent:', this.parent)

        console.log('project/main/index.js; clickCard, cardId:', cardId)

        const productPage = new ProductPage(this.parent, cardId)

        productPage.render()

    }

    render() {

        // debugger;

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        document.getElementById('simply\_button').onclick = function() {

            console.log('work');

            let src = document.getElementById('input-component1').value;

            let title = document.getElementById('input-component2').value;

            let text = document.getElementById('input-component3').value;

            let obj = new Object();

            obj = {

                id: Date.now(),

                src: src,

                title: title,

                text: text

            };

            console.log('obj: ', obj)

            // debugger;

            ajax.post('http://localhost:8000/stocks/', obj)

        };

    }

}

**CardPage.js**

import {ProductComponent} from "../../components/product/index.js";

import {BackButtonComponent} from "../../components/back-button/index.js";

import {MainPage} from "../index.js";

export class ProductPage {

    constructor(parent, id) {

        this.parent = parent

        this.id = id

        console.log('ProductPage; parent:', parent)

    }

    getData() {

        return {

            id: 1,

            src: "https://images.squarespace-cdn.com/content/v1/59f9f933914e6b4f6e5fba1f/1521559098360-YTC45ORL6AQM6DB8FMQ5/COMPRA-02.jpg",

            title: `ID ${this.id}`,

            text: "Заказ оформлен. Ожидайте курьера."

        }

    }

    get pageRoot() {

        return document.getElementById('product-page')

    }

    getHTML() {

        return (

            `

                <div id="product-page"></div>

            `

        )

    }

    clickBack() {

        const mainPage = new MainPage(this.parent)

        mainPage.render()

    }

    render() {

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        const backButton = new BackButtonComponent(this.pageRoot)

        backButton.render(this.clickBack.bind(this))

        const data = this.getData()

        const stock = new ProductComponent(this.pageRoot)

        stock.render(data)

        console.log('main/product/index.js; render:', data)

    }

}

**DeletePage.js**

import {ProductCardComponent} from "../../components/product-card/index.js";

import {ProductPage} from "./CardPage.js";

import {ajax} from "../../modules/ajax.js";

export class DeletePage {

    constructor(parent, data) {

        this.parent = parent;

        this.data = data;

    }

    get pageRoot() {

        return document.getElementById('tovars-page')

    }

    getHTML() {

        return (

            `

                <div id="tovars-page" class="d-flex flex-wrap">

                    <input id="input-component" class="btn btn-primary" type="text" placeholder="ID продукта" value="">

                    <button id="simply\_button" class="btn btn-primary">Убрать с данных</button>

                <div/>

            `

        )

    }

    clickCard(e) {

        // debugger

        const cardId = e.target.dataset.id

        console.log('project/main/index.js; clickCard, this.parent:', this.parent)

        console.log('project/main/index.js; clickCard, cardId:', cardId)

        const productPage = new ProductPage(this.parent, cardId)

        productPage.render()

    }

    render() {

        // debugger;

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        document.getElementById('simply\_button').onclick = function() {

            console.log('work');

            let ID = document.getElementById('input-component').value;

            console.log('ID: ', ID)

            debugger;

            ajax.delete('http://localhost:8000/stocks/' + ID, function(ID) {

                console.log('Data deleted successfully');

              }, function(xhr, status, error) {

                console.log(`Error deleting data \nerror: ${error}\nstatus: ${status} \nxhr: ${xhr}`);

              });

        };

    }

}

**TovarsPage.js**

import {ProductCardComponent} from "../../components/product-card/index.js";

import {ProductPage} from "./CardPage.js";

import {ajax} from "../../modules/ajax.js";

export class DogsPage {

    constructor(parent, data) {

        this.parent = parent;

        this.data = data;

    }

    get pageRoot() {

        return document.getElementById('tovars-page')

    }

    getHTML() {

        return (

            `

                <div id="tovars-page" class="d-flex flex-wrap"><div/>

            `

        )

    }

    clickCard(e) {

        const cardId = e.target.dataset.id

        console.log('project/main/index.js; clickCard, this.parent:', this.parent)

        console.log('project/main/index.js; clickCard, cardId:', cardId)

        const productPage = new ProductPage(this.parent, cardId)

        productPage.render()

    }

    render() {

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        this.data.forEach((item) => {

            const productCard = new ProductCardComponent(this.pageRoot)

            productCard.render(item, this.clickCard.bind(this))

            console.log('main/index.js; render, productCard:', productCard)

            console.log('main/index.js; render, item:', item)

        })

    }

}

**Index.js**

import {ProductCardComponent} from "../components/product-card/index.js";

import {ProductPage} from "./page/CardPage.js";

import {ajax} from "../modules/ajax.js";

import {UpdateButtonComponent} from "../components/button/update.js"

import {AddButtonComponent} from "../components/button/add.js"

import {DeleteButtonComponent} from "../components/button/delete.js"

import {DogsPage} from "./page/TovarsPage.js"

import {AddPage} from "./page/AddPage.js"

import {DeletePage} from "./page/DeletePage.js"

// Подключение CSS

export class MainPage {

    constructor(parent, data) {

        this.parent = parent;

        this.data = data;

    }

    getDataInfoAll() {

        // debugger;

        ajax.get('http://localhost:8000/stocks/', (data) => {

            this.data = data

            console.log('data', data)

        })

    }

    get pageRoot() {

        return document.getElementById('main-page')

    }

    getHTML() {

        return (

            `

                <div id="main-page" class="d-flex flex-wrap"><div/>

            `

        )

    }

    clickCard(e) {

        // debugger

        const cardId = e.target.dataset.id

        console.log('project/main/index.js; clickCard, this.parent:', this.parent)

        console.log('project/main/index.js; clickCard, cardId:', cardId)

        const productPage = new ProductPage(this.parent, cardId)

        productPage.render()

    }

    clickBackUpdate() {

        const dogspage = new DogsPage(this.parent, this.data)

        dogspage.render()

    }

    clickBackAdd() {

        // debugger;

        const addpage = new AddPage(this.parent, this.data, Object.keys(this.data).length)

        addpage.render()

    }

    clickBackDelete() {

        // debugger;

        const addpage = new DeletePage(this.parent, this.data, Object.keys(this.data).length)

        addpage.render()

    }

    render() {

        // debugger;

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        const UpdateButton = new UpdateButtonComponent(this.pageRoot)

        UpdateButton.render(this.clickBackUpdate.bind(this))

        const AddButton = new AddButtonComponent(this.pageRoot)

        AddButton.render(this.clickBackAdd.bind(this))

        const DeleteButton = new DeleteButtonComponent(this.pageRoot)

        DeleteButton.render(this.clickBackDelete.bind(this))

        const data = this.getDataInfoAll()

        console.log(data)

    }

}

**Папка modules**

**Ajax.js**

class Ajax {

    debugger;

    post(url, data, callback) {

        console.log('url:', url);

        fetch(url, {

            method: 'POST',

            body: JSON.stringify(data),

            headers: {

                'Content-Type': 'application/json'

            }

        })

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

            .then(data => {

                callback(data);

                // console.log(data);

            });

    }

    get(url, callback) {

        // console.log('url:', url);

        fetch(url, {method: 'GET'})

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

            .then(data => {

                callback(data);

                // console.log(data);

            });

    }

    delete(url, callback) {

        // console.log('url:', url);

        fetch(url, {method: 'DELETE'})

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

            .then(data => {

                callback(data);

                // console.log(data);

            });

    }

}

export const ajax = new Ajax();

**DBConnector.js**

const fs = require('fs');

const path = require('path');

class DBConnector {

    constructor(filename) {

        this.filename = filename;

    }

    readFile() {

        return fs.readFileSync(path.join(process.cwd(), 'db', this.filename), "utf8");

    }

    writeFile(data) {

        fs.writeFileSync(path.join(process.cwd(), 'db', this.filename), data, "utf8");

    }

}

module.exports = {

     DBConnector,

};

**Папка pages**

**Папка main**

**Index.js**

import {ButtonComponent} from "../../components/button/index.js";

import {ProductCardComponent} from "../../components/product-card/index.js";

import {ProductComponent} from "../../components/product/index.js";

import {ProductPage} from "./product/index.js";

import {BackButtonComponent} from "../../components/back-button/index.js";

export class MainPage {

    constructor(parent) {

        this.parent = parent;

    }

    getData() {

        return [

            {

                id: 1,

                src: "https://i.pinimg.com/originals/c9/ea/65/c9ea654eb3a7398b1f702c758c1c4206.jpg",

                title: "Акция",

                text: "Такой акции вы еще не видели 1"

            },

            {

                id: 2,

                src: "https://i.pinimg.com/originals/c9/ea/65/c9ea654eb3a7398b1f702c758c1c4206.jpg",

                title: "Акция",

                text: "Такой акции вы еще не видели 2"

            },

            {

                id: 3,

                src: "https://i.pinimg.com/originals/c9/ea/65/c9ea654eb3a7398b1f702c758c1c4206.jpg",

                title: "Акция",

                text: "Такой акции вы еще не видели 3"

            }

        ]

    }

    get pageRoot() {

        return document.getElementById('main-page')

    }

    getHTML() {

        return (

            `

                <div id="main-page" class="d-flex flex-wrap"><div/>

            `

        )

    }

    clickCard(e) {

        const cardId = e.target.dataset.id

        const productPage = new ProductPage(this.parent, cardId)

        productPage.render()

    }

    render() {

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        const data = this.getData()

        data.forEach((item) => {

            const productCard = new ProductCardComponent(this.pageRoot)

            productCard.render(item, this.clickCard.bind(this))

            console.log('render, item:', item)

        })

    }

}

**Папка product**

**Index.js**

import {ProductComponent} from "../../../components/product/index.js";

import {BackButtonComponent} from "../../../components/back-button/index.js";

import {MainPage} from "../index.js";

export class ProductPage {

    constructor(parent, id) {

        this.parent = parent

        this.id = id

    }

    getData() {

        return {

            id: 1,

            src: "https://i.pinimg.com/originals/c9/ea/65/c9ea654eb3a7398b1f702c758c1c4206.jpg",

            title: `Акция ${this.id}`,

            text: "Такой акции вы еще не видели"

        }

    }

    get pageRoot() {

        return document.getElementById('product-page')

    }

    getHTML() {

        return (

            `

                <div id="product-page"></div>

            `

        )

    }

    clickBack() {

        const mainPage = new MainPage(this.parent)

        mainPage.render()

    }

    render() {

        this.parent.innerHTML = ''

        const html = this.getHTML()

        this.parent.insertAdjacentHTML('beforeend', html)

        const backButton = new BackButtonComponent(this.pageRoot)

        backButton.render(this.clickBack.bind(this))

        const data = this.getData()

        const stock = new ProductComponent(this.pageRoot)

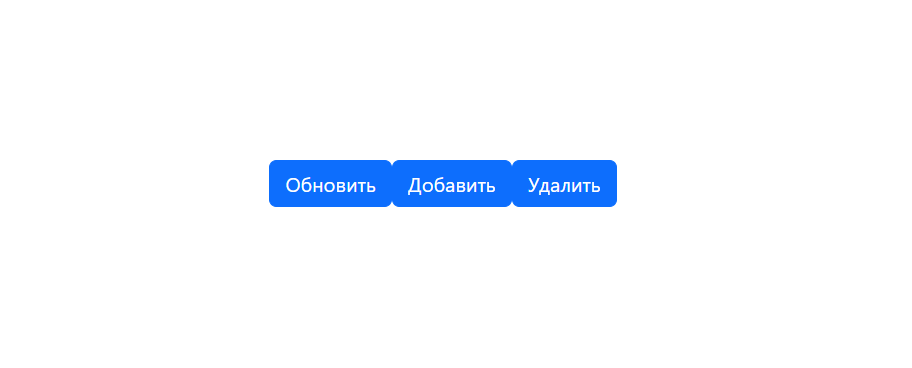
        stock.render(data)

    }

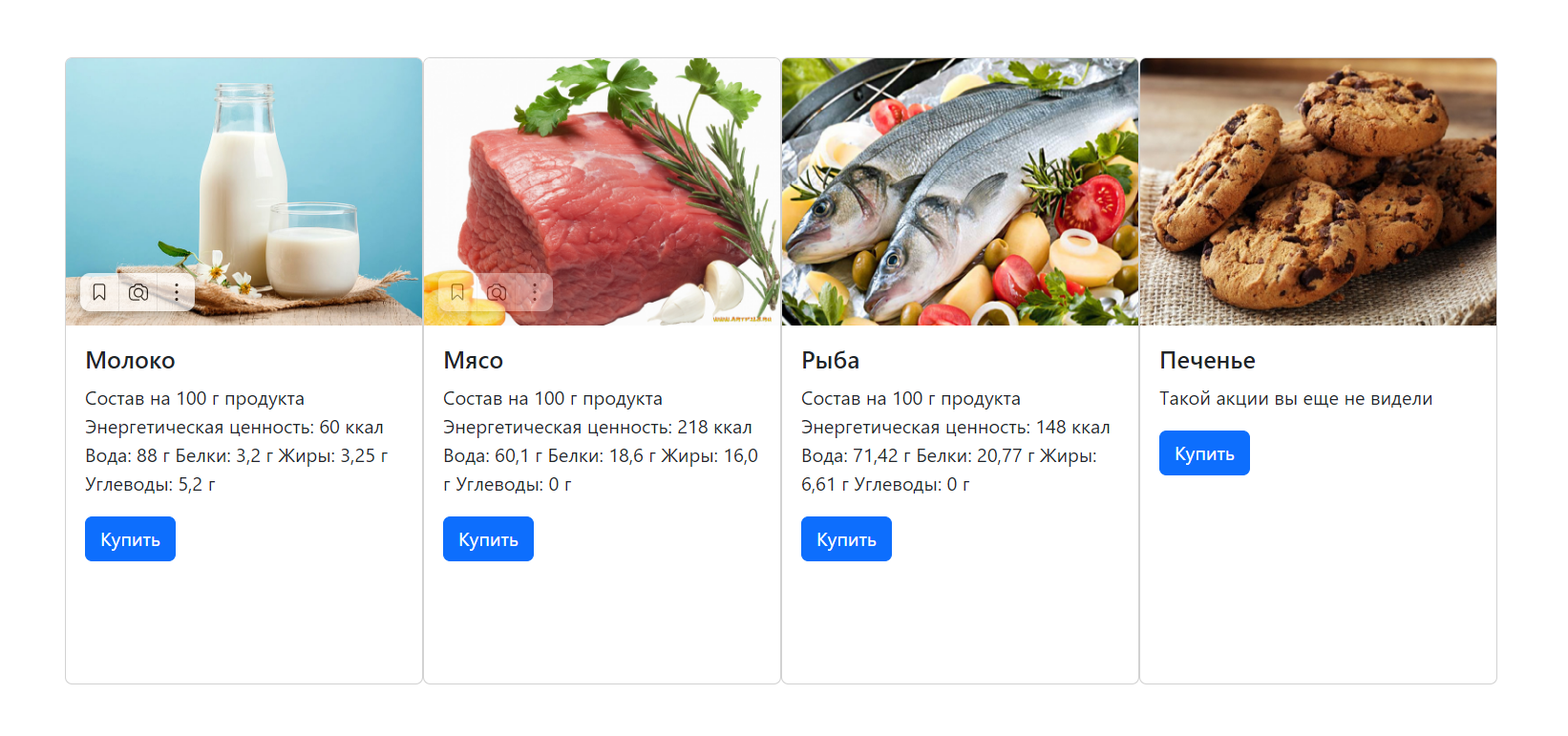
}

## Результат

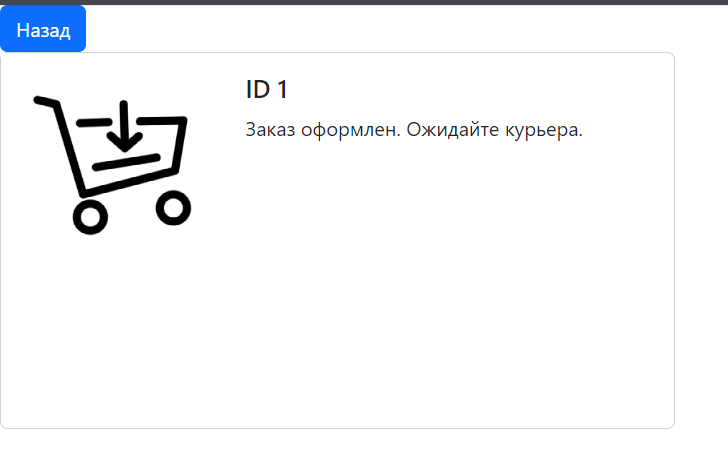
На окне браузера показаны 3 кнопки



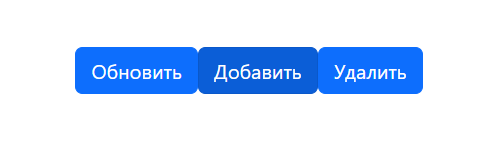
Нажмем на кнопку Обновить, показывает 4 карточки



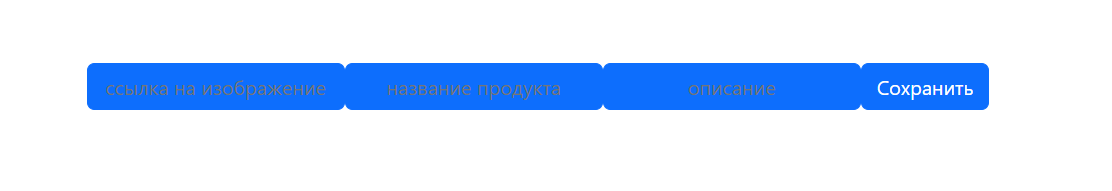
Нажмем на кнопку Купить Молоко, показывает окно результата о том, что заказ оформлен



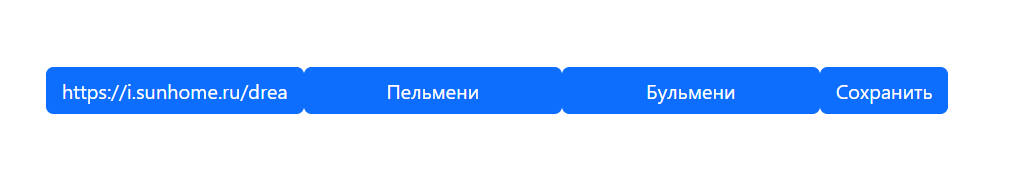
Нажмем на кнопку добавить



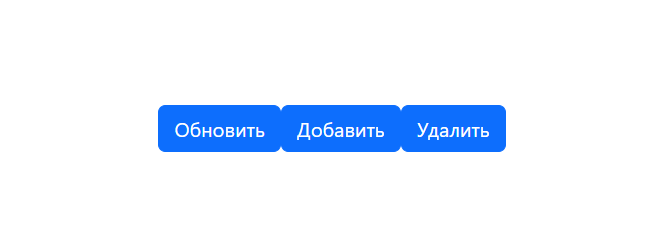
Появляется 4 кнопки, нужно будет заполнять все данные, для того, чтобы отобразилось на карточках



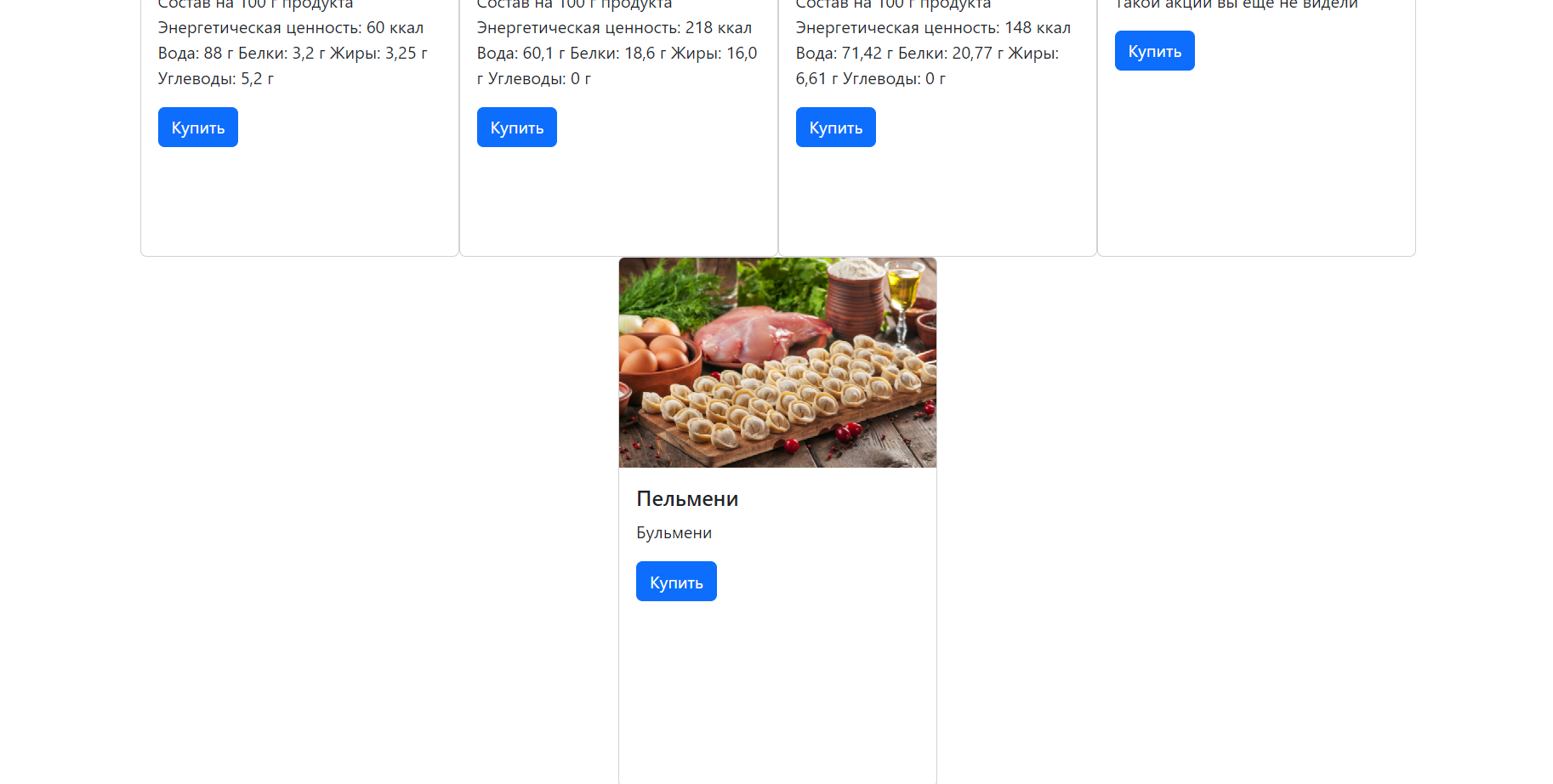
Заполнили поля и нажмем на кнопку сохранить



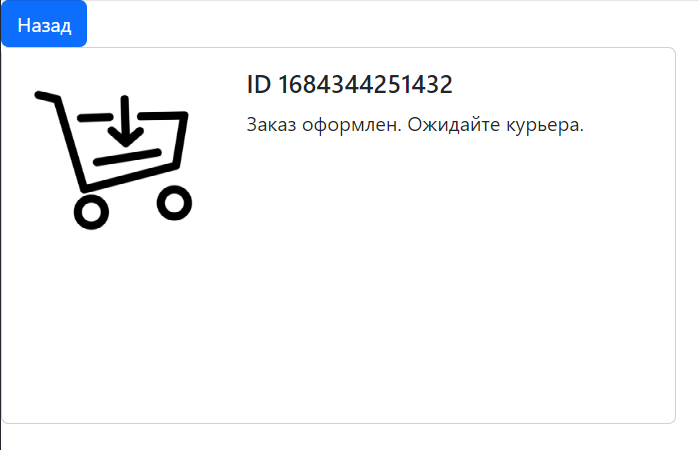
Нажмем на кнопку Обновить



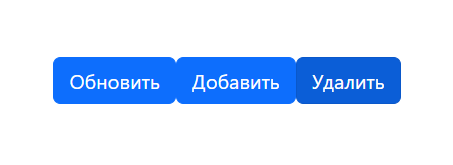
Появляется карточка Пельмени



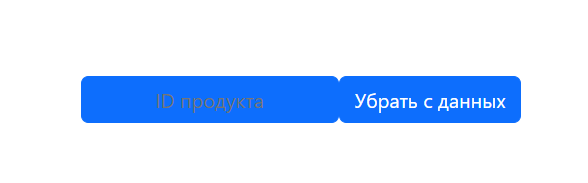
Нажмем на кнопку Купить Пельмени и скопируем ID для того, чтобы потом можно было удалить карточку



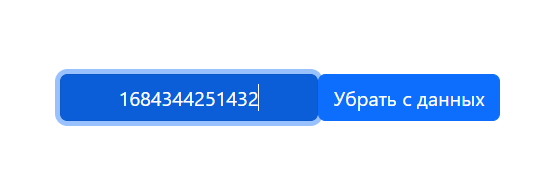
Нажмем на кнопку удалить



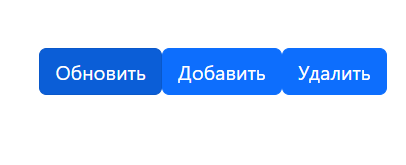
Появляется 2 кнопки, вставим ID карточки для того, чтобы удалить



Вставили ID карточки и удаляем его



Нажмем на кнопку Обновить, чтобы проверить удалилась ли карточка



Карточка удалена

