МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РФ Федеральное государственное бюджетное образовательное учреждение высшего образования

**«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 учебной дисциплине: «Интернет технологии»

На тему

**«Разработка клиент-серверного AJAX приложения.»**

**Выполнил:**

Студент группы №M30-324Бк-17

Архинчеев С.Б.

**Проверил:**

Титов Ю.П.

1. Постановка задачи

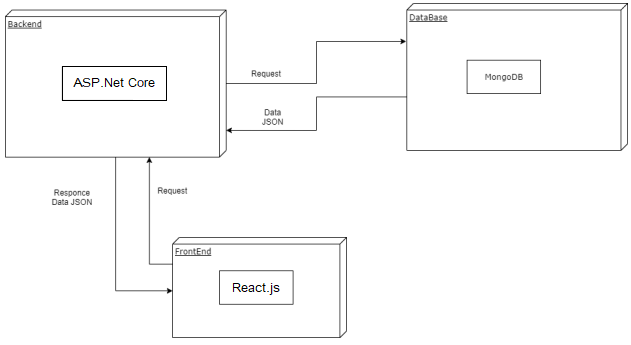
Разработать клиент-серверное AJAX приложение. Серверное BackEnd приложение необходимо разработать на Node.JS (необходимо поставить менеджер пакетов npm) с применением фреймворка Express. Для хранения информации на сервере используется документоориентированная NOSQL база данных MongoDB. Клиентское приложение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

В результате клиент-серверное программное приложение должно обеспечивать:

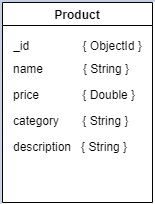
* Возможность подключения множества клиентов к серверу (в идеале к удаленному серверу)
* Загрузки информации о разработчиках и назначении программы
* Возможность добавления новой информации на сервер
* Возможность удаления с сервера необходимой информации
* Возможность вывода на экран клиента информации,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

Тип хранимой на сервере информации: База данных продуктов

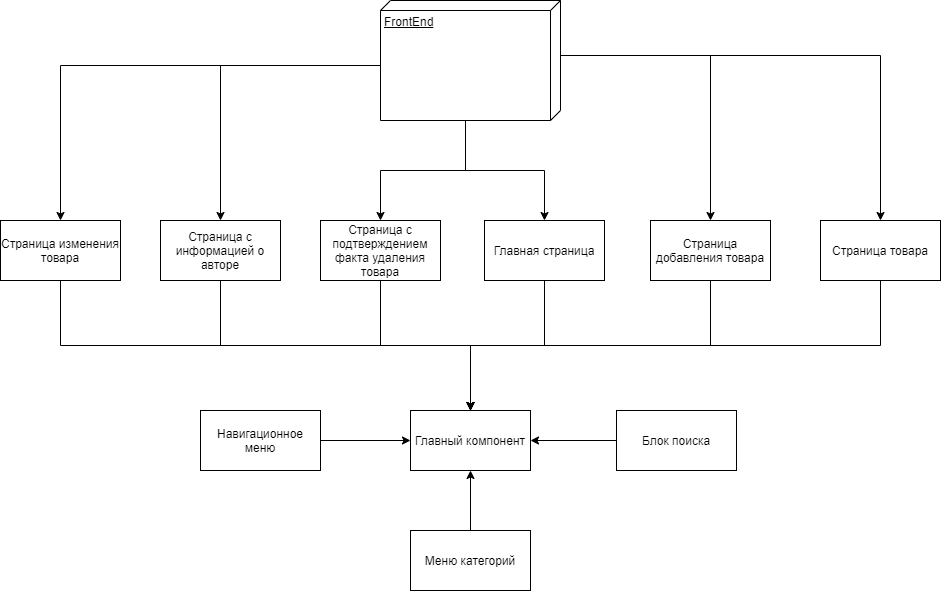
2. Структурная схема приложения.



3. Структурная схема Базы Данных.



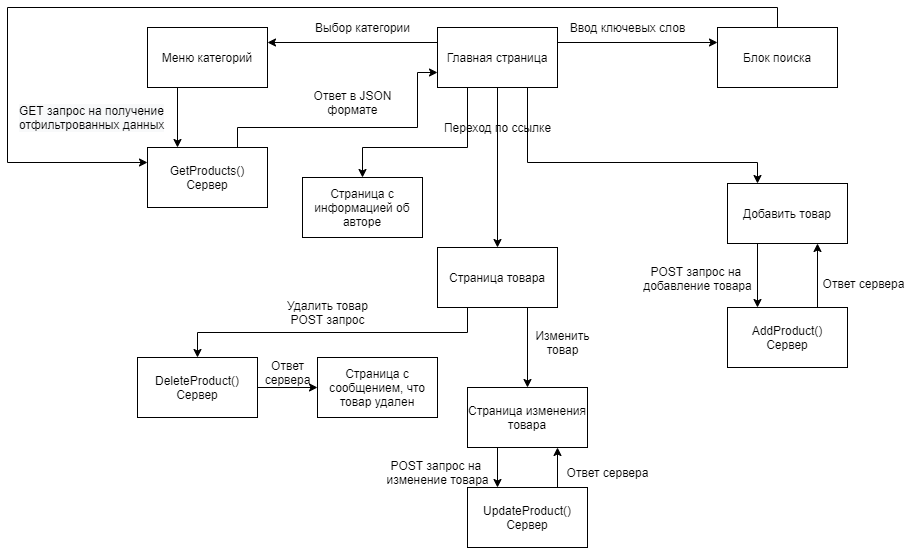
4. Схема Front-end модулей.



В главный компонент приложения всегда встроены навигационное меню, блок поиска и меню категорий, а остальные компоненты встраиваются в зависимости от ссылки, на которую переходит пользователь.

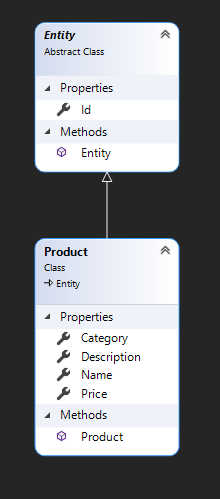
Каждый модуль включает в себя js и css файлы.

5. Схемы взаимодействия модулей с BackEnd.

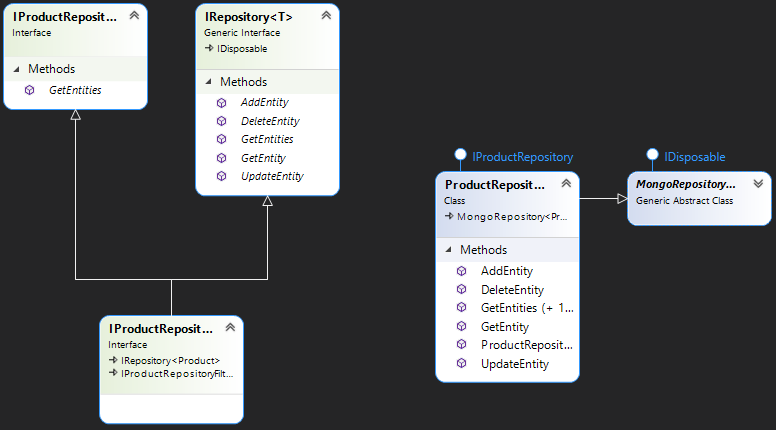


6. Диаграммы классов BackEnd’a

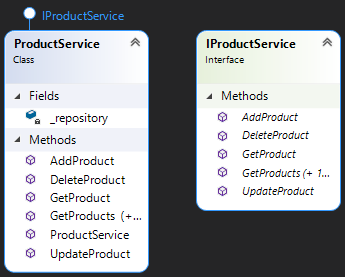
Модели данных:



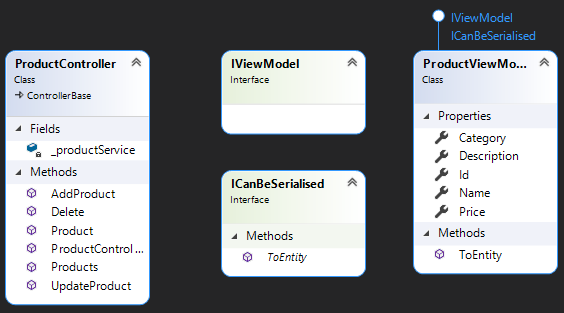
Репозитории:



Сервисы:



Контроллеры и видимые модели позволяющих избавиться от прямых зависимостей между контроллером и моделями данных:



7. Код программ.

FrontEnd:

Файл App.js

import React, { Component } from 'react';

import { Route } from 'react-router';

import { Layout } from './components/Layout';

import { MainPage } from './components/MainPage';

import { AddProduct } from './components/AddProduct';

import { ProductPage } from './components/ProductPage';

import { Deleted } from './components/Deleted'

import { UpdateProduct } from './components/UpdateProduct'

import { Info } from './components/Info'

import './custom.css'

export default class App extends Component {

static displayName = App.name;

render () {

return (

<Layout>

<Route exact path='/' component={MainPage} />

<Route path='/search-:name' component={MainPage} />

<Route path='/category/:category' component={MainPage} />

<Route path='/add\_product' component={AddProduct} />

<Route path='/info' component={Info} />

<Route path='/product:id' component={ProductPage} />

<Route path='/deleted:id' component={Deleted} />

<Route path='/update\_product:id' component={UpdateProduct} />

</Layout>

);

}

}

Файл Layout.js

import React, { Component } from 'react';

import { Container } from 'reactstrap';

import { NavMenu } from './NavMenu';

import { SearchTab } from './SearchTab';

import { Categories } from './Categories';

import './Layout.css';

export class Layout extends Component {

static displayName = Layout.name;

render() {

return (

<div>

<Categories />

<div id="top" >

<NavMenu />

<SearchTab />

<div id="container">

<Container>

{this.props.children}

</Container>

</div>

</div>

</div>

);

}

}

Файл Layout.css

body {

}

#top {

width: 92%;

position: relative;

right: -8%;

height: 100vh;

}

#top header {

display: block;

width: 100%;

z-index: 0;

}

#container {

position: relative;

top: 80px;

}

#first {

z-index: 1 !important;

}

Файл AddProduct.js

import React, { Component } from 'react';

import axios from "C:/Users/ArhiS/node\_modules/axios";

import "./AddProduct.css";

function onClick(e) {

let product = {

name: document.getElementById("name").value,

price: Number(document.getElementById("price").value),

category: document.getElementById("category").value,

description: document.getElementById("description").value,

id: Number(document.getElementById("id").value),

};

axios.post('https://localhost:5001/api/Product/AddProduct', product);

}

export class AddProduct extends Component {

static displayName = AddProduct.name;

constructor(props) {

super(props);

this.onClick = onClick;

}

componentDidMount() {

var btn = document.getElementById("addProduct");

btn.addEventListener("click", onClick);

}

render() {

return (

<form name="add">

<input type="text" id="name" placeholder="Name"></input>

<input type="number" id="price" placeholder="Price"></input>

<input type="text" id="category" placeholder="Category"></input>

<input type="text" id="description" placeholder="Description"></input>

<input type="number" id="id" placeholder="Id"></input>

<input type="button" id="addProduct" value="Add" />

</form>

);

}

}

Файл AddProduct.css

body {

}

form {

display: flex;

flex-direction: column;

justify-content: space-around;

width: 200px;

height: 30vh;

}

#container {

padding-top: 20px;

}

#description {

height: 100px;

}

Файл Categories.js

import React, { Component } from 'react';

import { Collapse, Container, Navbar, NavbarBrand, NavbarToggler, NavItem, NavLink } from 'reactstrap';

import { Link } from 'react-router-dom';

import './Categories.css';

export class Categories extends Component {

static displayName = Categories.name;

constructor(props) {

super(props);

this.toggleNavbar = this.toggleNavbar.bind(this);

this.onClick = this.onClick.bind(this);

this.state = {

collapsed: true

};

}

toggleNavbar() {

this.setState({

collapsed: !this.state.collapsed

});

}

onClick(e) {

let url = "https://localhost:5011/category/" + e.target.innerHTML;

window.location.href = url;

}

render() {

return (

<header>

<Navbar id="categories" className="navbar-expand-sm navbar-toggleable-sm ng-white border-bottom box-shadow mb-3" light>

<Container>

<NavbarToggler onClick={this.toggleNavbar} className="mr-2" />

<Collapse className="d-sm-inline-flex flex-sm-row-reverse" isOpen={!this.state.collapsed} navbar>

<ul className="navbar-nav flex-grow">

<NavItem>

<NavLink tag={Link} className="text-dark" to="/">All</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" onClick={this.onClick}>Smartphone</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" onClick={this.onClick}>Laptop</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" onClick={this.onClick}>Tablet</NavLink>

</NavItem>

</ul>

</Collapse>

</Container>

</Navbar>

</header>

);

}

}

Файл Categories.css

body {

}

a.navbar-brand {

white-space: normal;

text-align: center;

word-break: break-all;

}

html {

font-size: 14px;

}

@media (min-width: 768px) {

html {

font-size: 16px;

}

}

.box-shadow {

box-shadow: 0 .25rem .75rem rgba(0, 0, 0, .05);

}

#categories ul {

flex-direction: column !important;

}

#categories {

width: 100%;

padding-left: 5px;

height: 100vh;

box-shadow: none;

border: solid lightgray 1px;

margin-bottom: 0 !important;

}

header {

width: 8%;

display: inline-block;

position: absolute;

top: 0;

left: 0;

}

Файл Deleted.js

import React, { Component } from 'react';

import axios from "C:/Users/ArhiS/node\_modules/axios";

export class Deleted extends Component {

static displayName = Deleted.name;

constructor(props) {

super(props);

this.state = {

product: {},

IsLoaded: false,

};

}

async componentDidMount() {

let url = `https://localhost:5001/api/Product/GetProduct/` + `${this.props.match.params.id}`

await axios.get(url)

.then(res => res.data)

.then((data) => {

this.setState({ IsLoaded: true, product: data })

})

.catch(console.log);

if (this.state.IsLoaded)

axios.post('https://localhost:5001/api/Product/DeleteProduct', this.state.product);

}

render() {

return (

<div>

<strong>Deleted.</strong>

</div>

);

}

}

Файл Info.js

import React, { Component } from 'react';

export class Info extends Component {

static displayName = Info.name;

constructor(props) {

super(props);

}

render() {

return (

<div>

<div>

<p><strong>Author: </strong></p><p>Arkhincheev S.B.</p>

</div>

<div>

<p><strong>Email: </strong></p><p>IDKnife@mail.ru</p>

</div>

<div>

<p>

<strong>Fraimworks: </strong><br></br>

<i>Client</i> - React.js;<br></br>

<i>Server</i> - ASP.Net Core 3.1;<br></br>

<i>Database</i> - MongoDB;<br></br>

</p>

</div>

</div>

);

}

}

Файл MainPage.js

import React, { Component } from 'react';

import axios from "C:/Users/ArhiS/node\_modules/axios";

import { Link } from 'react-router-dom';

import { NavLink } from 'reactstrap';

import './MainPage.css';

export class MainPage extends Component {

static displayName = MainPage.name;

constructor(props) {

super(props);

this.onClickAscendingName = this.onClickAscendingName.bind(this);

this.onClickDescendingName = this.onClickDescendingName.bind(this);

this.onClickAscendingPrice = this.onClickAscendingPrice.bind(this);

this.onClickDescendingPrice = this.onClickDescendingPrice.bind(this);

this.state = {

products: [],

IsLoaded: false,

};

}

async componentDidMount() {

let url;

if (this.props.match.params.name != null)

url = `https://localhost:5001/api/Product/Products/` + `${this.props.match.params.name}`;

else if (this.props.match.params.category != null)

url = `https://localhost:5001/api/Product/Products/-/` + `${this.props.match.params.category}`;

else

url = `https://localhost:5001/api/Product/Products`;

await axios.get(url)

.then(res => res.data)

.then((data) => {

this.setState({ IsLoaded: true, products: data })

})

.catch(console.log);

}

onClickAscendingName() {

this.setState({

IsLoaded: true, products: this.state.products.sort(function (a, b) {

if (a.name > b.name) return 1;

if (a.name == b.name) return 0;

if (a.name < b.name) return -1;

})

})

}

onClickDescendingName() {

this.setState({

IsLoaded: true, products: this.state.products.sort(function (a, b) {

if (a.name > b.name) return -1;

if (a.name == b.name) return 0;

if (a.name < b.name) return 1;

})

})

}

onClickAscendingPrice() {

this.setState({

IsLoaded: true, products: this.state.products.sort(function (a, b) {

if (a.price > b.price) return 1;

if (a.price == b.price) return 0;

if (a.price < b.price) return -1;

})

})

}

onClickDescendingPrice() {

this.setState({

IsLoaded: true, products: this.state.products.sort(function (a, b) {

if (a.price > b.price) return -1;

if (a.price == b.price) return 0;

if (a.price < b.price) return 1;

})

})

}

render() {

var { IsLoaded, products } = this.state;

let i = 0;

let k = 0;

let parts = [];

if (IsLoaded)

while (i <= products.length) {

parts.push(products.slice(i, i + 4));

i += 4;

k++;

}

if (!IsLoaded) {

return (

<p>

Loading.

</p>

);

}

else {

return (

<table>

<tr>

<p><b>Name:</b></p><th onClick={this.onClickAscendingName}>Ascending</th><th onClick={this.onClickDescendingName}>Descending</th>

<p><b>Price:</b></p><th onClick={this.onClickAscendingPrice}>Ascending</th><th onClick={this.onClickDescendingPrice}>Descending</th>

</tr>

{parts.map((products) => (

<tr>

{products.map((product) => (

<NavLink tag={Link} to={`/product${product.id}`} >

<td>

<div>{product.name}</div>

<div>{product.description}</div>

<div>{product.price}</div>

</td>

</NavLink>

))}

</tr>

))}

</table>

);

}

}

}

Файл MainPage.css

body {

}

table {

width: 100%;

}

tr {

margin-top: 10px;

display: flex;

}

td {

margin-right: 70px;

height: 150px;

width: 150px;

display: flex;

flex-direction: column;

align-items: center;

justify-content: space-between;

border: solid lightgray 1px;

}

th {

border: solid lightgray 1px;

margin: 0 5px 0 5px;

height: 30px;

}

Файл NavMenu.js

import React, { Component } from 'react';

import { Collapse, Container, Navbar, NavbarBrand, NavbarToggler, NavItem, NavLink } from 'reactstrap';

import { Link } from 'react-router-dom';

import './NavMenu.css';

export class NavMenu extends Component {

static displayName = NavMenu.name;

constructor(props) {

super(props);

this.toggleNavbar = this.toggleNavbar.bind(this);

this.state = {

collapsed: true

};

}

toggleNavbar() {

this.setState({

collapsed: !this.state.collapsed

});

}

render() {

return (

<header id="first">

<Navbar id="first-nav" className="navbar-expand-sm navbar-toggleable-sm ng-white border-bottom box-shadow mb-3" light>

<Container>

<NavbarBrand>8(495)\*\*\*\*\*\*\*</NavbarBrand>

<NavbarToggler onClick={this.toggleNavbar} className="mr-2" />

<Collapse className="d-sm-inline-flex flex-sm-row-reverse" isOpen={!this.state.collapsed} navbar>

<ul className="navbar-nav flex-grow">

<NavItem>

<NavLink tag={Link} className="text-dark" to="/info">Info</NavLink>

</NavItem>

{ /\* <NavItem>

<NavLink tag={Link} className="text-dark" to="#">Delivery</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" to="#">Store addresses</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" to="#">Feedback</NavLink>

</NavItem> \*/

}

</ul>

</Collapse>

</Container>

</Navbar>

</header>

);

}

}

Файл NavMenu.css

a.navbar-brand {

white-space: normal;

text-align: center;

word-break: break-all;

}

html {

font-size: 14px;

}

@media (min-width: 768px) {

html {

font-size: 16px;

}

}

.box-shadow {

box-shadow: 0 .25rem .75rem rgba(0, 0, 0, .05);

}

#first-nav {

box-shadow: none;

margin-bottom: 0 !important;

height: 40px;

}

Файл ProductPage.js

import React, { Component } from 'react';

import axios from "C:/Users/ArhiS/node\_modules/axios";

import { Link } from 'react-router-dom';

import { NavLink } from 'reactstrap';

import './ProductPage.css';

export class ProductPage extends Component {

static displayName = ProductPage.name;

constructor(props) {

super(props);

this.state = {

product: {},

IsLoaded: false,

};

}

async componentDidMount() {

let url = `https://localhost:5001/api/Product/GetProduct/` + `${this.props.match.params.id}`

await axios.get(url)

.then(res => res.data)

.then((data) => {

this.setState({ IsLoaded: true, product: data })

})

.catch(console.log);

}

render() {

var { IsLoaded, product } = this.state;

if (!IsLoaded) {

return (

<p>

Loading.

</p>

);

}

else {

return (

<div>

<h1>{product.name}</h1>

<div><strong>Price</strong>:{product.price}</div>

<div><strong>Category</strong>:{product.category}</div>

<div><strong>Description</strong>:{product.description}</div>

<NavLink tag={Link} to={`/deleted${product.id}`}>Delete</NavLink><p>|</p><NavLink tag={Link} to={`/update\_product${product.id}` }>Change</NavLink>

</div>

);

}

}

}

Файл ProductPage.css

div, h1 {

color: black;

}

a {

display: inline-block !important;

padding: 0 !important;

}

p {

display: inline-block !important;

padding: 0 5px 0 5px;

}

Файл SearchTab.js

import React, { Component } from 'react';

import { Collapse, Container, Navbar, NavbarBrand, NavbarToggler, NavItem, NavLink } from 'reactstrap';

import { Link } from 'react-router-dom';

import './SearchTab.css';

export class SearchTab extends Component {

static displayName = SearchTab.name;

constructor(props) {

super(props);

this.onClick = this.onClick.bind(this);

this.toggleNavbar = this.toggleNavbar.bind(this);

this.state = {

collapsed: true

};

}

onClick() {

let elem = document.getElementById("search");

let url;

if (elem.value != "")

url = "https://localhost:5011/search-" + elem.value;

else

url = "https://localhost:5011/"

window.location.href = url;

}

toggleNavbar() {

this.setState({

collapsed: !this.state.collapsed

});

}

render() {

return (

<header>

<Navbar id="second-nav" className="navbar-expand-sm navbar-toggleable-sm ng-white border-bottom box-shadow mb-3" light>

<Container>

<NavbarBrand><input type="text" placeholder="Search" id="search" autofocus /><p onClick=

{this.onClick} id="btn"><b>Go!</b></p></NavbarBrand>

<NavbarToggler onClick={this.toggleNavbar} className="mr-2" />

<Collapse className="d-sm-inline-flex flex-sm-row-reverse" isOpen={!this.state.collapsed} navbar>

<ul className="navbar-nav flex-grow">

{ /\* <NavItem>

<NavLink tag={Link} className="text-dark" to="#">User</NavLink>

</NavItem>

<NavItem>

<NavLink tag={Link} className="text-dark" to="#">Busket</NavLink>

</NavItem> \*/

}

<NavItem>

<NavLink tag={Link} className="text-dark" to="/add\_product">Add Product</NavLink>

</NavItem>

</ul>

</Collapse>

</Container>

</Navbar>

</header>

);

}

}

Файл SearchTab.css

a.navbar-brand {

white-space: normal;

text-align: center;

word-break: break-all;

}

html {

font-size: 14px;

}

@media (min-width: 768px) {

html {

font-size: 16px;

}

}

.box-shadow {

box-shadow: 0 .25rem .75rem rgba(0, 0, 0, .05);

}

header {

margin-top: 0;

}

#second-nav {

height: 40px;

position: relative;

top: 40px;

}

#search {

height: 25px;

}

#btn {

display: inline-block;

margin: 0;

}

Файл UpdateProduct.js

import React, { Component } from 'react';

import axios from "C:/Users/ArhiS/node\_modules/axios";

function onClick(e) {

let product = {

name: document.getElementById("name").value,

price: Number(document.getElementById("price").value),

category: document.getElementById("category").value,

description: document.getElementById("description").value,

id: Number(document.getElementById("id").value),

};

axios.post('https://localhost:5001/api/Product/UpdateProduct', product);

}

export class UpdateProduct extends Component {

static displayName = UpdateProduct.name;

constructor(props) {

super(props);

this.onClick = onClick;

this.state = {

product: {},

IsLoaded: false,

};

}

async componentDidMount() {

let url = `https://localhost:5001/api/Product/GetProduct/` + `${this.props.match.params.id}`

await axios.get(url)

.then(res => res.data)

.then((data) => {

this.setState({ IsLoaded: true, product: data })

})

.catch(console.log);

var btn = document.getElementById("updateProduct");

btn.addEventListener("click", onClick);

}

render() {

return (

<form name="update">

<input type="text" id="name" value={this.state.product.name}></input>

<input type="number" id="price" value={this.state.product.price}></input>

<input type="text" id="category" value={this.state.product.category}></input>

<input type="text" id="description" value={this.state.product.description}></input>

<input type="hidden" id="id" value={this.state.product.id}></input>

<input type="button" id="updateProduct" value="Update" />

</form>

);

}

}

BackEnd:

Файл Entity.cs

namespace CourseWork.Models

{

public abstract class Entity

{

public int Id { get; set; }

public Entity(int id)

{

Id = id;

}

}

}

Файл Product.cs

namespace CourseWork.Models

{

public class Product : Entity

{

public string Name { get; set; }

public double Price { get; set; }

//ToDo: Сделать категорию сущностью

public string Category { get; set; }

public string Description { get; set; }

public Product(int id, string name, double price, string category, string description) : base(id)

{

Name = name;

Price = price;

Category = category;

Description = description;

}

}

}

Файл IRepository.cs

using System;

using System.Collections.Generic;

using System.Reflection.Metadata;

using System.Threading.Tasks;

using CourseWork.Models;

namespace CourseWork.Repositories.Interfaces

{

public interface IRepository<T> : IDisposable

where T : Entity

{

Task<IList<T>> GetEntities();

Task AddEntity(T entity);

Task DeleteEntity(T entity);

Task UpdateEntity(T entity);

Task<T> GetEntity(int id);

}

}

Файл IProductRepositoryFilters.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using CourseWork.Models;

namespace CourseWork.Repositories.Interfaces

{

public interface IProductRepositoryFilters

{

Task<IList<Product>> GetEntities(string name, string category);

}

}

Файл IProductRepository.cs

using CourseWork.Models;

namespace CourseWork.Repositories.Interfaces

{

public interface IProductRepository : IRepository<Product>, IProductRepositoryFilters

{

}

}

Файл ProductRepository.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using CourseWork.Models;

using CourseWork.Repositories.Interfaces;

using MongoDB.Bson;

using MongoDB.Driver;

namespace CourseWork.Repositories.Implementations

{

public class ProductRepository : MongoRepository<Product>, IProductRepository

{

public ProductRepository(IMongoDatabase database) : base(database)

{

Entities = database.GetCollection<Product>("Products");

}

public async Task<IList<Product>> GetEntities()

{

return await Entities.Find(FilterDefinition<Product>.Empty).ToListAsync();

}

public async Task<IList<Product>> GetEntities(string name, string category)

{

var filter1 = Builders<Product>.Filter.Regex("Name", new BsonRegularExpression(name));

var filter2 = Builders<Product>.Filter.Eq("Category", category);

if (category == null)

return await Entities.Find(filter1).ToListAsync();

else

return await Entities.Find(filter2).ToListAsync();

}

public async Task AddEntity(Product entity)

{

await Entities.InsertOneAsync(entity);

}

public async Task DeleteEntity(Product entity)

{

var filter = Builders<Product>.Filter.Eq("\_id", entity.Id);

await Entities.DeleteOneAsync(filter);

}

public async Task UpdateEntity(Product entity)

{

var filter = Builders<Product>.Filter.Eq("\_id", entity.Id);

await Entities.ReplaceOneAsync(filter, entity);

}

public async Task<Product> GetEntity(int id)

{

var filter = Builders<Product>.Filter.Eq("\_id", id);

return await Entities.Find(filter).FirstOrDefaultAsync();

}

}

}

Файл MongoRepository.cs

using System;

using CourseWork.Models;

using MongoDB.Driver;

namespace CourseWork.Repositories.Implementations

{

public abstract class MongoRepository<T> : IDisposable

where T : Entity

{

protected IMongoDatabase Database { get; set; }

protected IMongoCollection<T> Entities { get; set; }

public MongoRepository(IMongoDatabase database)

{

Database = database;

}

public void Dispose()

{

Entities = null;

Database = null;

}

}

}

Файл IProductService.cs

using System.Collections.Generic;

using System.Threading.Tasks;

using CourseWork.Models;

namespace CourseWork.Services.Interfaces

{

public interface IProductService

{

Task<IList<Product>> GetProducts();

Task<IList<Product>> GetProducts(string name, string category);

Task AddProduct(Product entity);

Task DeleteProduct(Product entity);

Task UpdateProduct(Product entity);

Task<Product> GetProduct(int id);

}

}

Файл ProductService.cs

using System;

using System.Collections.Generic;

using System.Threading.Tasks;

using CourseWork.Models;

using CourseWork.Repositories.Interfaces;

using CourseWork.Services.Interfaces;

namespace CourseWork.Services.Implementations

{

public class ProductService : IProductService

{

private readonly IProductRepository \_repository;

public ProductService(IProductRepository repository)

{

\_repository = repository;

}

public async Task<IList<Product>> GetProducts()

{

try

{

return await \_repository.GetEntities();

}

catch (Exception e)

{

//ToDo: логирование

throw;

}

}

public async Task<IList<Product>> GetProducts(string name, string category)

{

try

{

return await \_repository.GetEntities(name, category);

}

catch (Exception e)

{

//ToDo: логирование

throw;

}

}

public async Task AddProduct(Product entity)

{

try

{

await \_repository.AddEntity(entity);

}

catch (Exception e)

{

//ToDo: логирование

throw;

}

}

public async Task DeleteProduct(Product entity)

{

try

{

await \_repository.DeleteEntity(entity);

}

catch (Exception e)

{

//ToDo: логирование

throw;

}

}

public async Task UpdateProduct(Product entity)

{

try

{

await \_repository.UpdateEntity(entity);

}

catch (Exception e)

{

//ToDo: логирование

throw;

}

}

public async Task<Product> GetProduct(int id)

{

try

{

return await \_repository.GetEntity(id);

}

catch (Exception)

{

//ToDo: логирование

throw;

}

}

}

}

Файл ICanBeSerialised.cs

using CourseWork.Models;

namespace CourseWork.WebApi.Interfaces

{

public interface ICanBeSerialised

{

Entity ToEntity();

}

}

Файл IViewModel.cs

namespace CourseWork.WebApi.Interfaces

{

public interface IViewModel

{

}

}

Файл ProductViewModel.cs

using CourseWork.Models;

using CourseWork.WebApi.Interfaces;

namespace CourseWork.WebApi.ViewModels

{

/// <summary>

/// Модель данных товара.

/// </summary>

public class ProductViewModel : IViewModel, ICanBeSerialised

{

/// <summary>

/// Уникальный идентификатор.

/// </summary>

public int Id { get; set; }

/// <summary>

/// Наименование товара.

/// </summary>

public string Name { get; set; }

/// <summary>

/// Стоимость товара.

/// </summary>

public double Price { get; set; }

/// <summary>

/// Категория товара.

/// </summary>

public string Category { get; set; }

/// <summary>

/// Описание товара.

/// </summary>

public string Description { get; set; }

/// <summary>

/// Преобразовать модель в сущность.

/// </summary>

/// <returns>Сущность.</returns>

public Entity ToEntity()

{

return new Product(Id, Name, Price, Category, Description);

}

}

}

Файл ProductController.cs

using System;

using System.Collections.Generic;

using System.Threading.Tasks;

using CourseWork.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using CourseWork.Services.Interfaces;

using CourseWork.WebApi.ViewModels;

using Microsoft.AspNetCore.Cors;

namespace CourseWork.WebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class ProductController : ControllerBase

{

private readonly IProductService \_productService;

public ProductController(IProductService productService)

{

\_productService = productService;

}

/// <summary>

/// Получить список товаров.

/// </summary>

/// <returns>Список товаров.</returns>

[HttpGet]

[Route("Products/{name?}/{category?}")]

[EnableCors("DefaultPolicy")]

[ProducesResponseType(typeof(List<Product>), StatusCodes.Status200OK)]

[ProducesResponseType(typeof(string), StatusCodes.Status400BadRequest)]

public async Task<IActionResult> Products(string name = null, string category = null)

{

try

{

IList<Product> products;

if ((name != null) || (category != null))

products = await \_productService.GetProducts(name, category);

else

products = await \_productService.GetProducts();

return Ok(products);

}

catch (Exception e)

{

return BadRequest(e.ToString());

}

}

/// <summary>

/// Добавить товар.

/// </summary>

/// <param name="product">Товар.</param>

/// <returns>Ответ сервера.</returns>

[HttpPost]

[Route("AddProduct")]

[EnableCors("DefaultPolicy")]

[ProducesResponseType(typeof(void), StatusCodes.Status200OK)]

[ProducesResponseType(typeof(string), StatusCodes.Status400BadRequest)]

public async Task<IActionResult> AddProduct([FromBody]ProductViewModel product)

{

try

{

await \_productService.AddProduct(product.ToEntity() as Product);

return Ok();

}

catch (Exception e)

{

return BadRequest(e.ToString());

}

}

/// <summary>

/// Получить товар по Id.

/// </summary>

/// <returns>Товар.</returns>

[HttpGet]

[Route("GetProduct/{id?}")]

[EnableCors("DefaultPolicy")]

[ProducesResponseType(typeof(Product), StatusCodes.Status200OK)]

[ProducesResponseType(typeof(string), StatusCodes.Status400BadRequest)]

public async Task<IActionResult> Product(int id)

{

try

{

var product = await \_productService.GetProduct(id);

return Ok(product);

}

catch (Exception e)

{

return BadRequest(e.ToString());

}

}

/// <summary>

/// Удалить товар.

/// </summary>

/// <returns>Ответ сервера.</returns>

[HttpPost]

[Route("DeleteProduct")]

[EnableCors("DefaultPolicy")]

[ProducesResponseType(typeof(void), StatusCodes.Status200OK)]

[ProducesResponseType(typeof(string), StatusCodes.Status400BadRequest)]

public async Task<IActionResult> Delete([FromBody]ProductViewModel product)

{

try

{

await \_productService.DeleteProduct(product.ToEntity() as Product);

return Ok();

}

catch (Exception e)

{

return BadRequest(e.ToString());

}

}

/// <summary>

/// Обновить товар.

/// </summary>

/// <param name="product">Товар.</param>

/// <returns>Ответ сервера.</returns>

[HttpPost]

[Route("UpdateProduct")]

[EnableCors("DefaultPolicy")]

[ProducesResponseType(typeof(void), StatusCodes.Status200OK)]

[ProducesResponseType(typeof(string), StatusCodes.Status400BadRequest)]

public async Task<IActionResult> UpdateProduct([FromBody]ProductViewModel product)

{

try

{

await \_productService.UpdateProduct(product.ToEntity() as Product);

return Ok();

}

catch (Exception e)

{

return BadRequest(e.ToString());

}

}

}

}

Файл Program.cs

using Microsoft.AspNetCore;

using Microsoft.AspNetCore.Hosting;

namespace CourseWork.WebApi

{

public class Program

{

public static void Main(string[] args)

{

CreateWebHostBuilder(args).Build().Run();

}

public static IWebHostBuilder CreateWebHostBuilder(string[] args) =>

WebHost.CreateDefaultBuilder(args)

.UseStartup<Startup>();

}

}

Файл Startup.cs

using CourseWork.Repositories.Implementations;

using CourseWork.Repositories.Interfaces;

using CourseWork.Services.Implementations;

using CourseWork.Services.Interfaces;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using MongoDB.Driver;

using Microsoft.OpenApi.Models;

namespace CourseWork.WebApi

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

public void ConfigureServices(IServiceCollection services)

{

var connectionString = Configuration.GetConnectionString("Mongo");

var connection = new MongoUrlBuilder(connectionString);

var client = new MongoClient(connectionString);

services.AddControllers();

services.AddTransient<IProductRepository, ProductRepository>();

services.AddTransient<IProductService, ProductService>();

services.AddSingleton<IMongoDatabase>(client.GetDatabase(connection.DatabaseName));

services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo { Title = "My API", Version = "v1" });

});

services.AddCors(o => o.AddPolicy("DefaultPolicy", builder =>

{

builder

.WithMethods("Access-Control-Allow-Methods,GET,POST,PUT,DELETE,PATCH,OPTIONS")

.WithHeaders("Access-Control-Allow-Origin,Access-Control-Allow-Methods")

.AllowAnyHeader()

.WithOrigins("https://localhost:5011", "http://localhost:5010");

}));

}

public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

app.UseHsts();

}

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "My API V1");

});

app.UseRouting();

app.UseCors();

app.UseEndpoints(endPoints => endPoints.MapControllers());

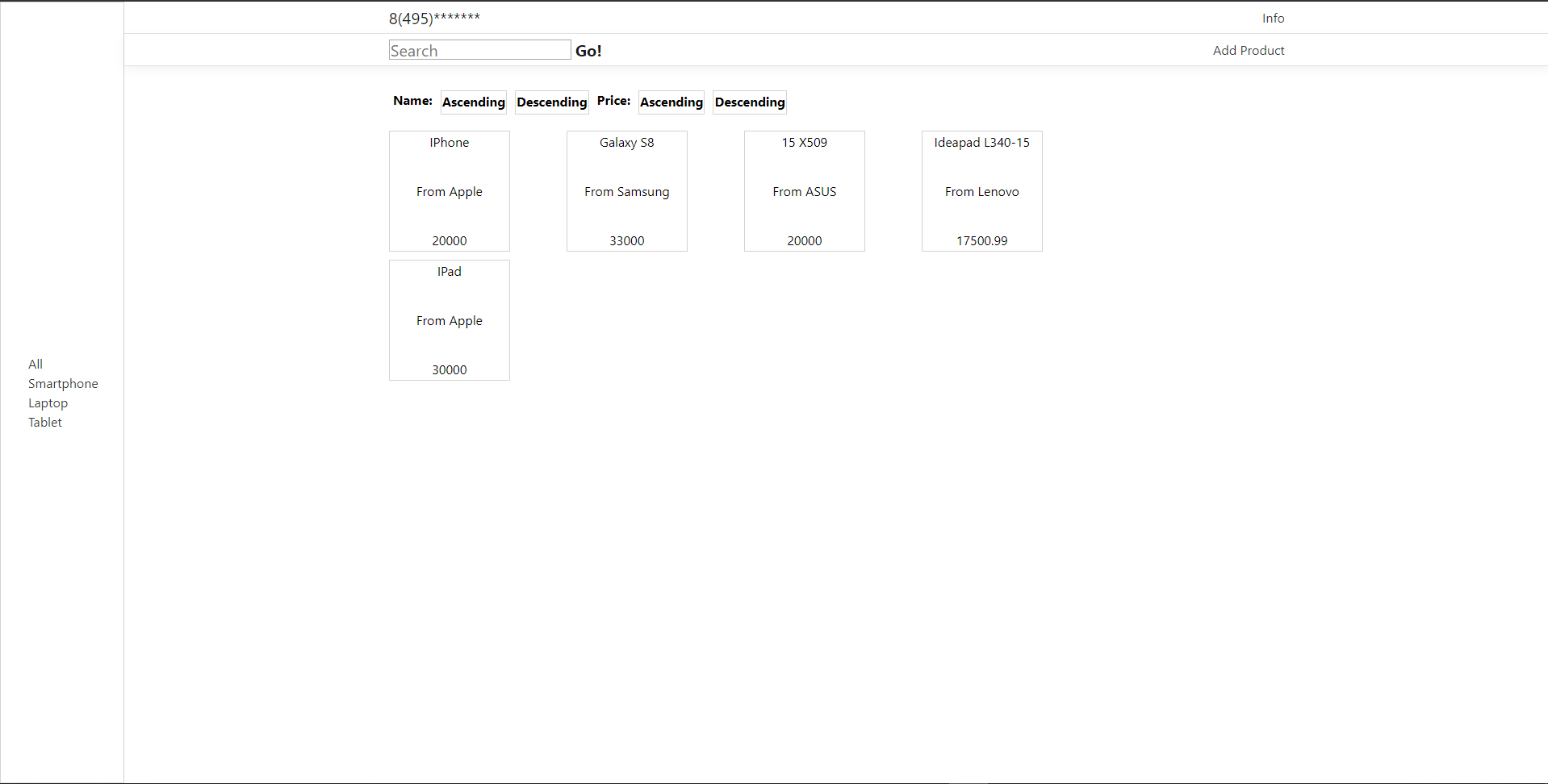
}

}

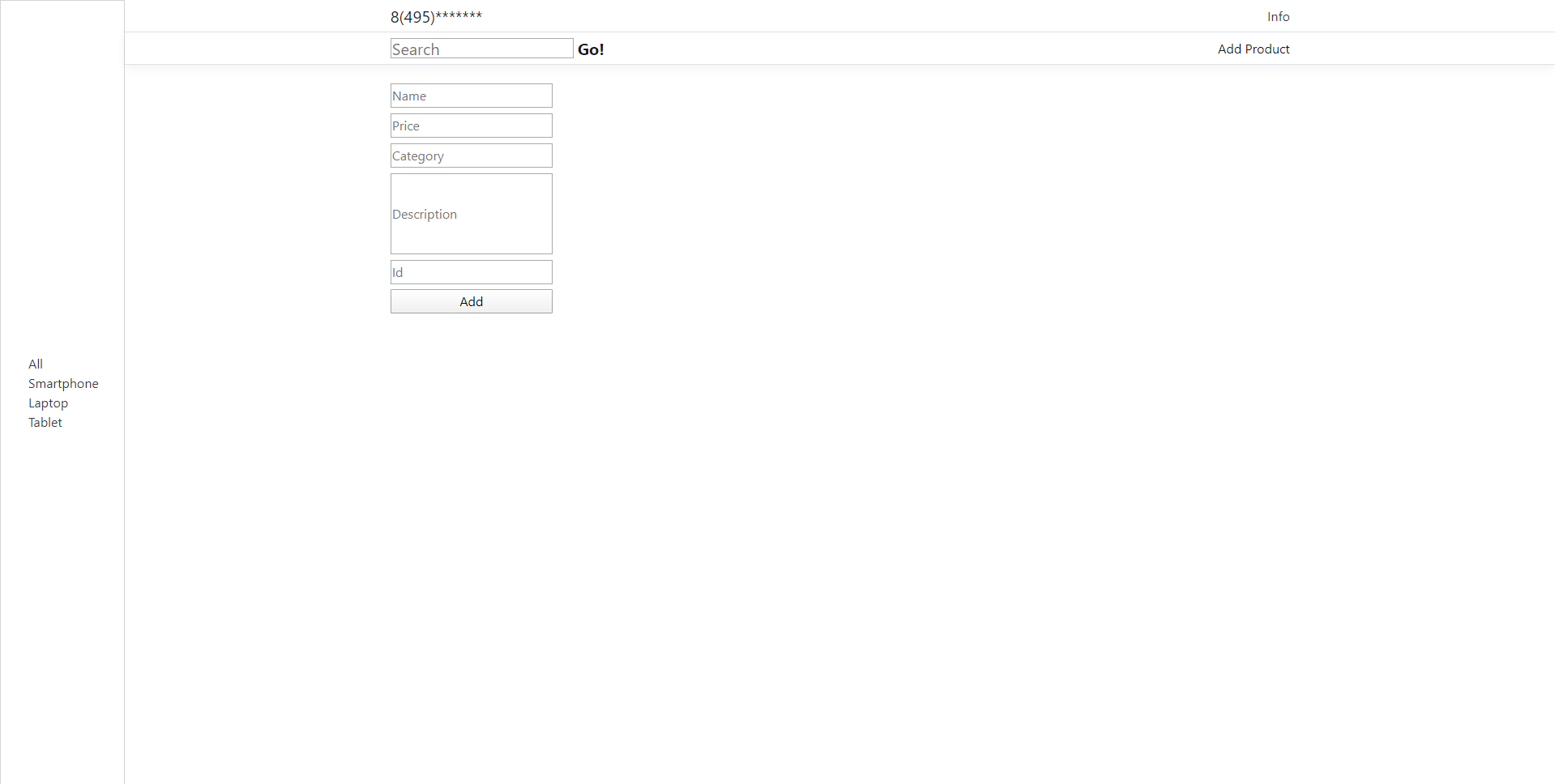
}

8. Результаты работы программы.

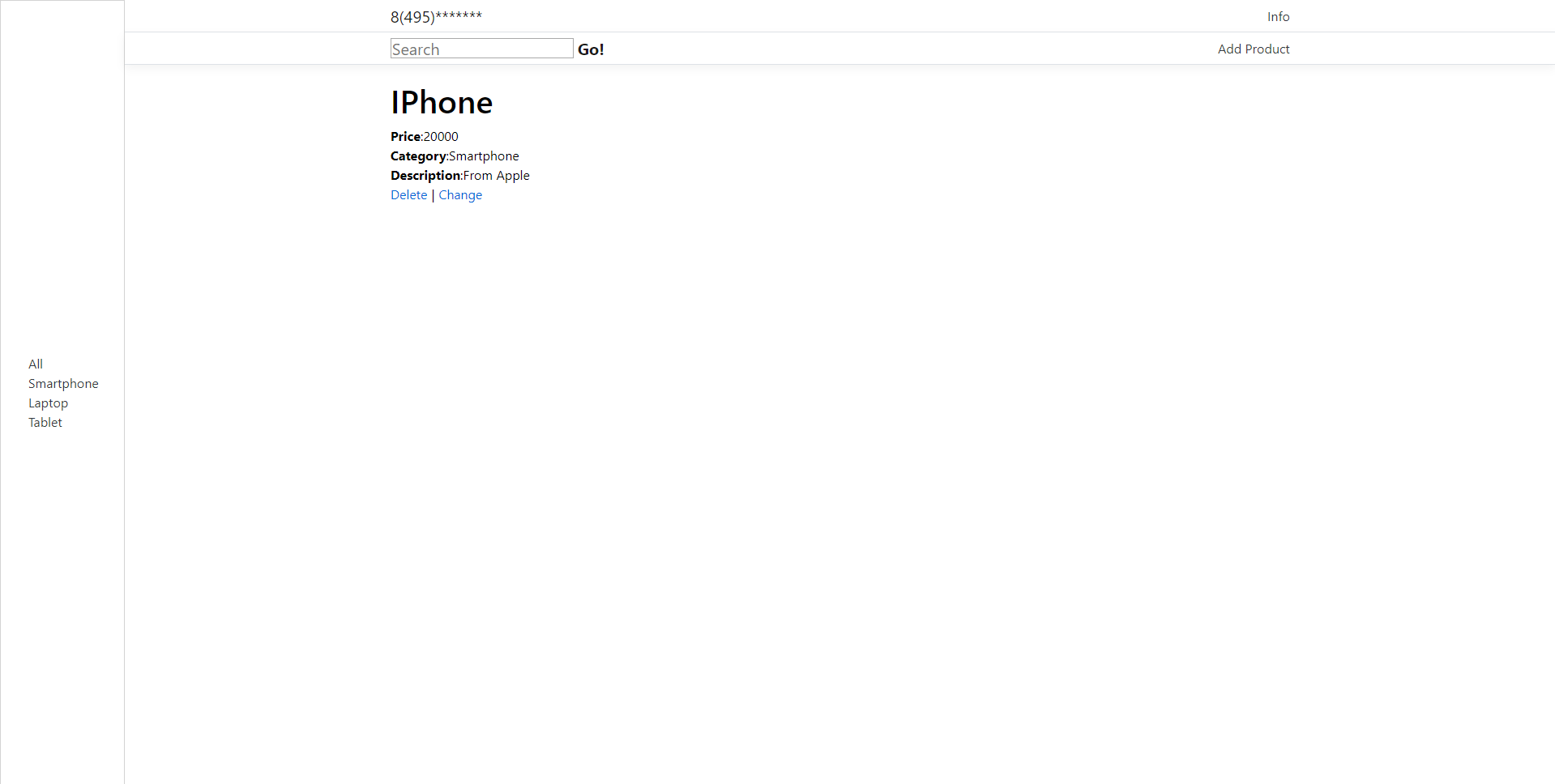
Главная страница



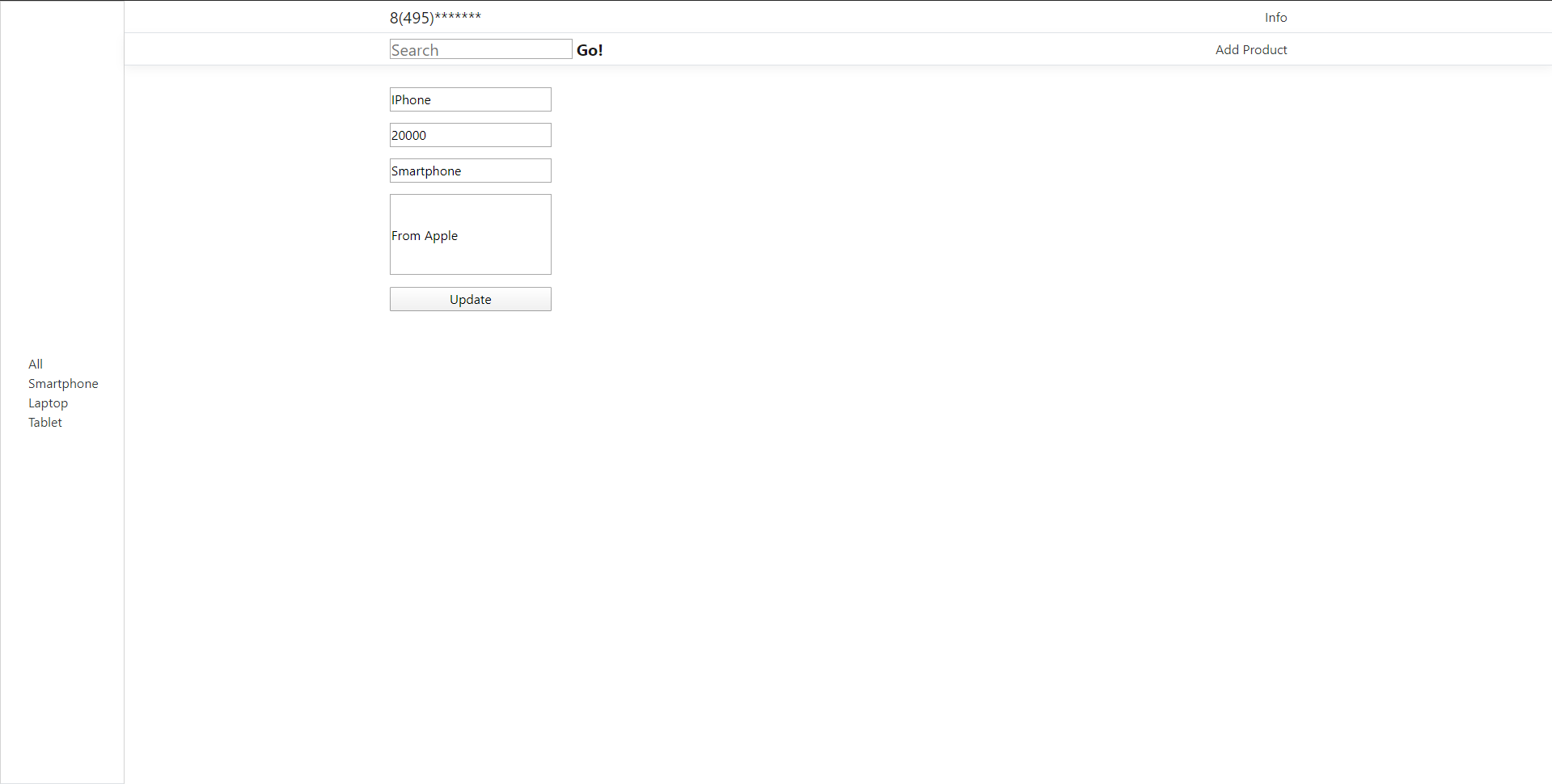
Страница добавления товара



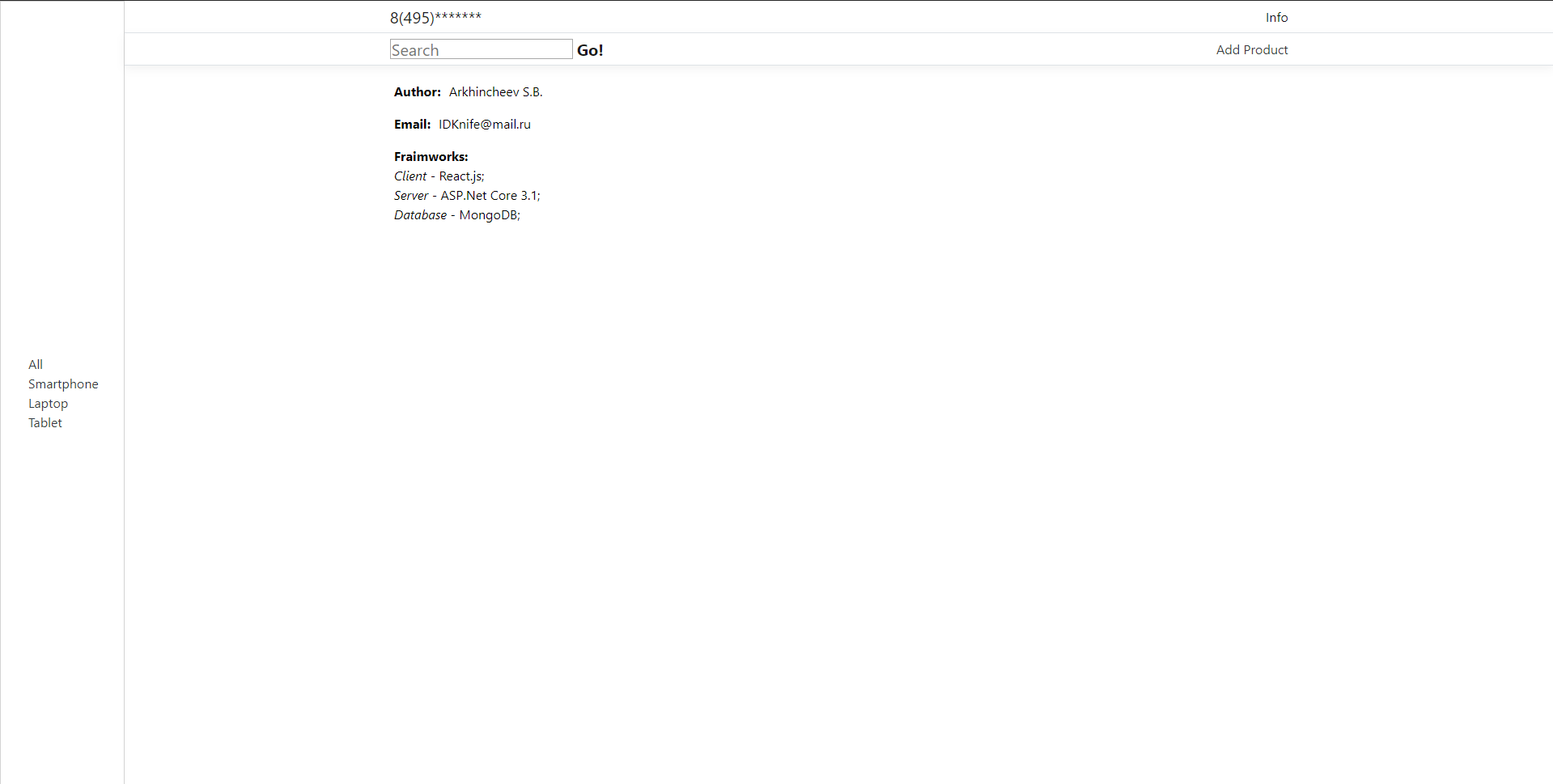
Страница товара



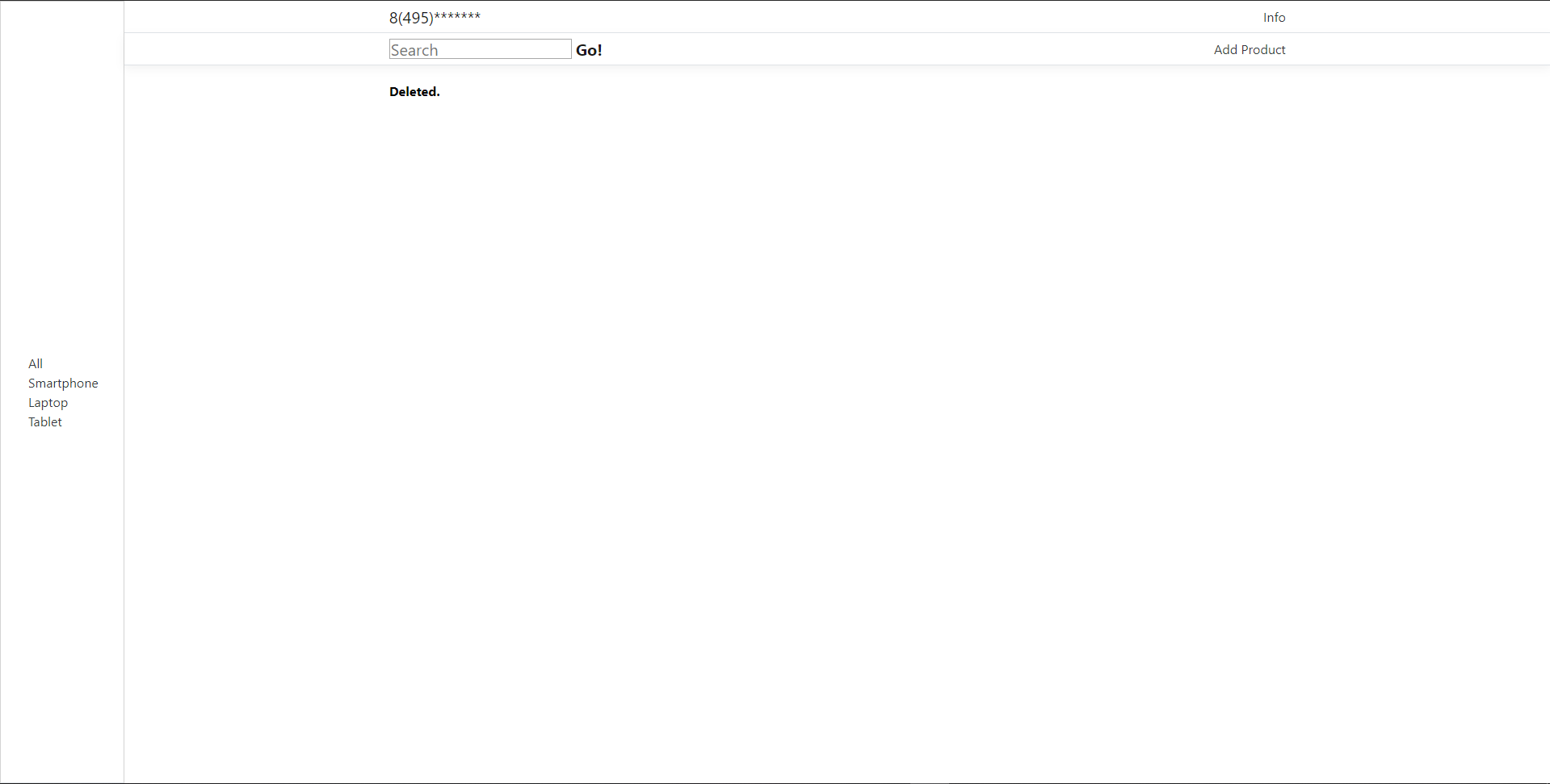
Страница изменения товара



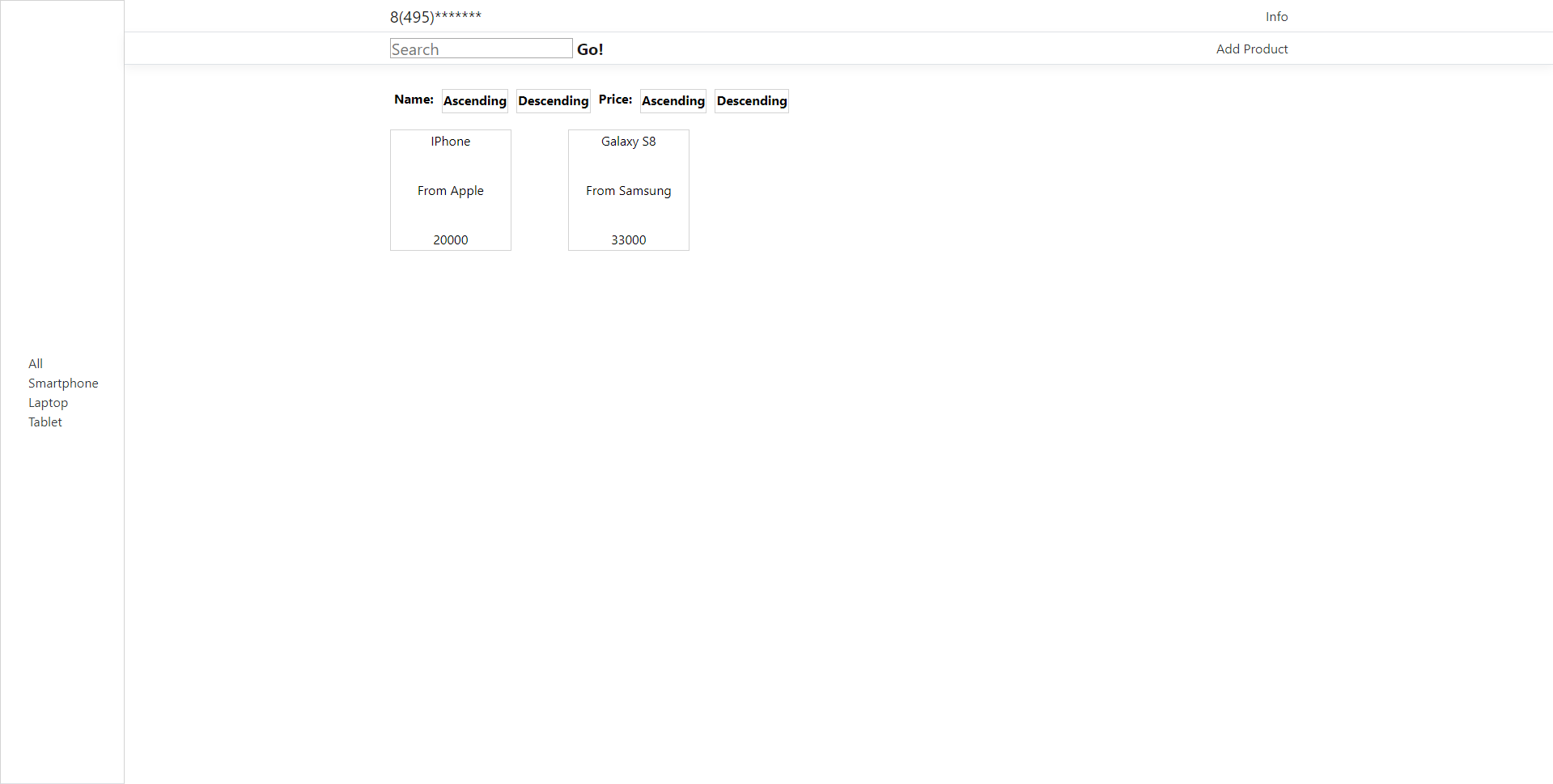
Страница с информацией об авторе



Страница с сообщением об удалении товара



Пример результата выбора одной из категорий (Smartphone)



Пример результата поиска (15)

