Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский**

**технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

Допускаю к защите

|  |  |
| --- | --- |
| Руководитель |  |
|  | подпись |
|  | В.А. Харахинов |
|  | И.О. Фамилия |

|  |
| --- |
|  |
| Разработка клиент-серверного приложения для работы с  нереляционной базой данных |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе по дисциплине

|  |
| --- |
| Объектно-ориентированные базы данных |

|  |
| --- |
| 1.02.00.00 ПЗ |
| обозначение документа |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСМб-19-1 |  |  |  | Д.Е. Вовиков |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | В.А. Харахинов |
|  |  |  |  | подпись |  | И.О. Фамилия |
| Курсовой проект защищен с оценкой | | | |  | | |

Иркутск 2022 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

ЗАДАНИЕ

на курсовое проектирование

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| По курсу | Объектно-ориентированные базы данных | | | | | | | | | | | | | | | | | |
| Студенту | Вовикову Д.Е. | | | | | | | | | | | | | | | | | |
|  | (фамилия, инициалы) | | | | | | | | | | | | | | | | | |
| Тема проекта | | Разработка клиент-серверного приложения для работы с нереляционной базой данных | | | | | | | | | | | | | | | | |
| Исходные данные | | | Вариант 2 | | | | | | | | | | | | | | | |
| Рекомендуемая литература | | | | | | | |  | | | | | | | | | | |
| 1. Сосинская С.С. «Разработка клиента по технологии MVC»: Методические указания по выполнению курсового проекта» Электронный каталог кафедры вычислительной техники. | | | | | | | | | | | | | | | | | | |
| Графическая часть на | | | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ листах. | | | | | | | | | | | | | | |
| Дата выдачи задания | | | | « | 12 | » | ноября | | | | | 2022 г. | | | | |
|  | | | | | | | | | | | | | | | | | | |
| Задание получил | | | | | | | | |  | | | |  | | Д.Е. Вовиков | | | |
|  | | | | | | | | | подпись | | | |  | | И.О. Фамилия | | | |
|  | | | | | | | | | |  |  | | |  | |  | |  |
| Дата представления проекта руководителю | | | | | | | | | | « | 24 | | | » | | декабря | | 2022 г. |
|  | | | | | | | | | | | | | | | | | | |
| Руководитель курсового проектирования | | | | | | | | |  | | | |  | | В.А. Харахинов | | | |
|  | | | | | | | | | подпись | | | |  | | И.О. Фамилия | | | |

**Содержание**

[Введение 4](#_Toc122459475)

[Индивидуальный вариант 5](#_Toc122459476)

[1. Структура базы данных 6](#_Toc122459477)

[1.1 Концептуальная модель данных 6](#_Toc122459478)

[1.2 Физическая модель данных 6](#_Toc122459479)

[2. Проектирование информационной системы 8](#_Toc122459480)

[3. Контроллеры 9](#_Toc122459481)

[4. Представления 10](#_Toc122459482)

[4.1 Index 10](#_Toc122459483)

[4.2 Juice 10](#_Toc122459484)

[4.3 Vendor 11](#_Toc122459485)

[5 Сценарии 14](#_Toc122459486)

[5.1 juice.js 14](#_Toc122459487)

[5.2 vendor.js 17](#_Toc122459488)

[6. WebApi 24](#_Toc122459489)

[7. Таблица тестов 25](#_Toc122459490)

[8. Результаты тестирования 27](#_Toc122459491)

[Заключение 30](#_Toc122459492)

[Список использованных источников 31](#_Toc122459493)

Введение

Целью курсового проекта является знакомство с фреймворком ASP NET MVC Framework для создания веб-приложения по используемому паттерну Model-view-controller (MVC) и использования NoSQL базы данных MongoDB.

В качестве предметной области был взят торговый автомат, включающий следующие сущности:

1. Соки;
2. Газировки;
3. Алкоголь;
4. Торговые автоматы.

Индивидуальный вариант

Имеются газировки (название, объем, тип, количество пузырьков), соки (название, объем, вкус, наличие мякоти) и алкоголь (название, объем, вкус, крепость). Из этих объектов составляются торговые автоматы (название, газировки, соки, алкоголь).

Выходные документы:

выдать список автоматов, в которых есть газировка;

выдать список автоматов, в которых находятся алкогольные напитки.

1. Структура базы данных

* 1. Концептуальная модель данных

Концептуальная модель данных содержит 4 сущности и 3 связи. Первая модель описывает отношение между сущностями Газировки и Торговый автомат. Торговый автомат не обаятельно должен иметь газировку, но газировку обязательно должна иметь Торговый автомат. Они имеют связь «многие ко многим». Вторая модель описывает отношение между Соками и Торговый автомат. Торговый автомат не обаятельно должен иметь сок, но сок обязательно должна иметь Торговый автомат. Они имеют связь «многие ко многим». Третья модель описывает отношение между сущностями Алкоголь и Торговый автомат. Торговый автомат не обаятельно должен иметь алкоголь, но алкоголь обязательно должен иметь Торговый автомат. Они имеют связь «многие ко многим».



Рисунок 1 – Концептуальная модель данных

1.2 Физическая модель данных

В результате применения инструмента draw.io в отношении разработки логической модели базы данных, основываясь на концептуальной модели (см. рис. 1) была разработана физическая модель, представленная на рисунке 2.

Таблица 1 – Структура модели «Соки»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| Title | String | Нет |  |
| Volume | Int32 | Нет |  |
| Taste | String | Нет |  |
| IsPulp | Boolean | Нет |  |

Таблица 2 – Структура модели «Газировки»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| Title | String | Нет |  |
| Volume | Int32 | Нет |  |
| Type | String | Нет |  |
| BubbleCount | Int32 | Нет |  |

Таблица 3 – Структура модели «Алкоголь»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| Title | String | Нет | FK |
| Volume | Int32 | Нет | FK |
| Type | String | Нет |  |
| Fortress | Int32 | Нет |  |

Таблица 4 – Структура модели «Торговый автомат»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| Title | String | Нет |  |
| Juices | Array< ObjectId > | Да | FK |
| Sodas | Array< ObjectId > | Да | FK |
| Alcohols | Array< ObjectId > | Да | FK |

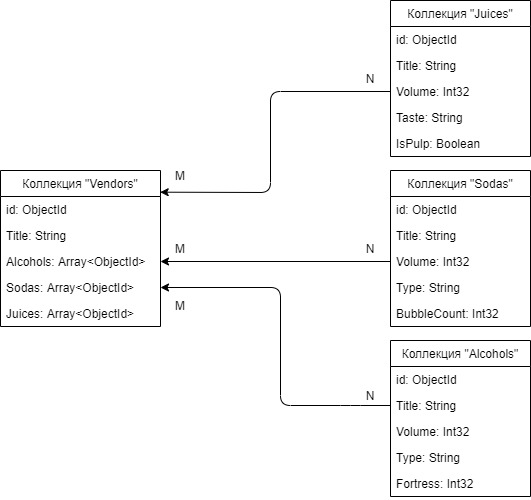


Рисунок 2 – Физическая модель данных «DrinkBD»

2. Проектирование информационной системы

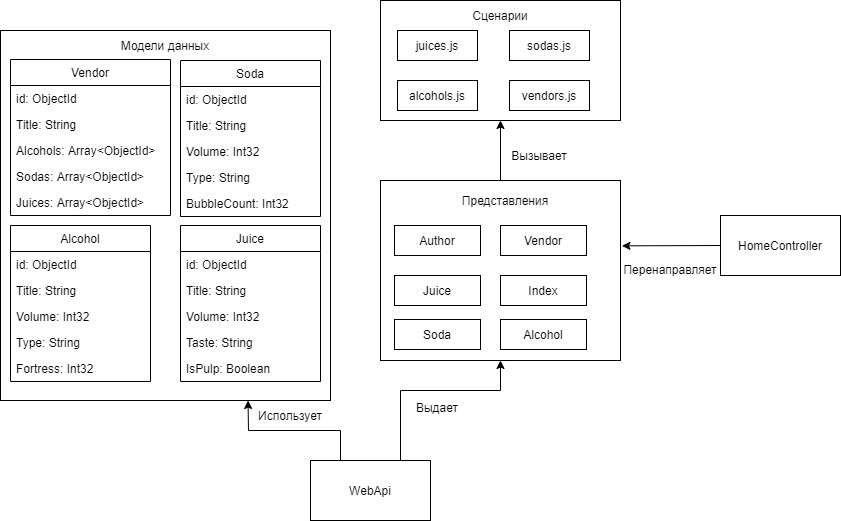


Рисунок 3 – Модель ИС

3. Контроллеры

В программе реализован 1 контроллер: HomeController, который занимается маршрутизацией сайта. Листинг:

public class HomeController : Controller

{

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger)

{

\_logger = logger;

}

public IActionResult Index()

{

return View();

}

public IActionResult Privacy()

{

return View();

}

public IActionResult Author()

{

return View();

}

public IActionResult Juice() => View();

public IActionResult Soda() => View();

public IActionResult Alcohol() => View();

public IActionResult Vendor() => View();

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

4. Представления

Для реализации ИС было создано 6 представлений (см. рис. 3). Представление Index содержит базовую информацию о работе. Представление Author содержит информация об авторе. Juice позволяет взаимодействовать с моделью Juice и выполнять выдачу первого документа. Soda позволяет взаимодействовать с моделью Soda. Alcohol позволяет взаимодействовать с моделью Alcohol. Vendor позволяет взаимодействовать с моделью Vendor и выполнять выдачу второго документа.

4.1 Index

@{

ViewData["Title"] = "Home Page";

}

<div class="text-center">

<h1 class="display-4">Добро пожаловать!</h1>

<h2 class="display-4">Курсованя работа по дисциплине Объектно-Ориентированные Базы Данных</h2>

</div>

<div>

<p>Вариант 2.</p>

<p>Предметная область: Торговый автомат</p>

<p>Модель данных включает в себя:</p>

<ul>

<li>Соки</li>

<li>Газировки</li>

<li>Алкоголь</li>

<li>Торговые автоматы</li>

</ul>

</div>

4.2 Juice

@{

ViewData["Title"] = "Соки";

}

<h1>@ViewData["Title"]</h1>

<div class="flex-container">

<div class="first">

<h2>Список соков</h2>

<div>

<input class="form-control" type="hidden" id="juiceId" />

<p>

Название:<br />

<input class="form-control" id="drinkTitle"/>

</p>

<p>

Объем:<br />

<input class="form-control" id="drinkVolume" type="number" min="250"

max="5000" value="250" step="250" />

</p>

<p>

Вкус:<br />

<select id="juiceTaste" class="form-select">

<option>Яблоко</option>

<option>Апельсин</option>

<option>Банан</option>

<option>Арбуз</option>

<option>Мультифрукт</option>

</select>

</p>

<p class="form-check form-switch">

<input class="form-check-input" id="drinkPulp" type="checkbox" />

<label class="form-check-label" for="flexSwitchCheckDefault">Наличие мякоти</label>

</p>

<p>

<button class="btn btn-primary" id="saveBtn" onclick="addJuice()">Сохранить</button>

<button class="btn btn-warning" id="resetBtn" onclick="reset()">Сбросить</button>

</p>

</div>

<table class="table table-striped">

<thead>

<tr>

<th scope="col">Название</th>

<th scope="col">Объем</th>

<th scope="col">Вкус</th>

<th scope="col">Наличие мякоти</th>

<th></th>

</tr>

</thead>

<tbody>

</tbody>

</table>

<script>

</script>

</div>

<div class="second">

<table id="mainTable" class="table table-striped">

<thead>

<tr>

<th>id</th>

<th>Название</th>

<th></th>

</tr>

</thead>

<tbody id="vendorBody">

</tbody>

</table>

</div>

</div>

<script src="~/js/juice.js" asp-append-version="true"></script>

<link rel="stylesheet" href="~/css/juice.css" asp-append-version="true" />

4.3 Vendor

@{

ViewData["Title"] = "Торговые автоматы";

}

<div>

<h1>@ViewData["Title"]</h1>

<button onclick="Search()" type="button" class="btn btn-success btn-lg btn-block">Показать автоматы с алкогольной продукцией</button>

<button onclick="SearchReset()" type="button" class="btn btn-info btn-lg btn-block">Отменить</button>

</div>

<div class="flex-container">

<input id="vendorId" type="hidden" />

<div class="first">

<select id="cmbJuice" class="form-select" onchange="cmbJuiceChange()"></select>

<input disabled="disabled" class="form-control" id="juiceInfo" />

<button class="btn btn-primary" id="btnJuiceAdd" onclick="AddJuice()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Название</th>

<th>Вкус</th>

<th></th>

</tr>

</thead>

<tbody id="juiceBody"></tbody>

</table>

</div>

<div class="second">

<select id="cmbSoda" class="form-select" onchange="cmbSodaChange()"></select>

<input disabled="disabled" class="form-control" id="sodaInfo" />

<button class="btn btn-primary" id="btnSodaAdd" onclick="AddSoda()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Название</th>

<th>Тип</th>

<th></th>

</tr>

</thead>

<tbody id="sodaBody"></tbody>

</table>

</div>

<div class="third">

<select id="cmbAlcohol" class="form-select" onchange="cmbAlcoholChange()"></select>

<input disabled="disabled" class="form-control" id="alcoholInfo" />

<button class="btn btn-primary" id="btnAlcoholAdd" onclick="AddAlcohol()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Название</th>

<th>Тип</th>

<th></th>

</tr>

</thead>

<tbody id="alcoholBody"></tbody>

</table>

</div>

</div>

<div>

<p>Название автомата:</p>

<input id="title" class="form-control" />

<button class="btn btn-primary" id="btnVendorAdd" onclick="AddVendor()">Сохранить автомат</button>

<button class="btn btn-warning" id="resetBtn" onclick="Reset()">Сбросить</button>

<table id="mainTable" class="table table-striped">

<thead>

<tr>

<th>id</th>

<th>Название</th>

<th></th>

</tr>

</thead>

<tbody id="vendorBody">

</tbody>

</table>

</div>

<link rel="stylesheet" href="~/css/vendor.css" asp-append-version="true" />

<script src="~/js/vendor.js" asp-append-version="true"></script>

5 Сценарии

Для взаимодействия с представлениями были седаны сценарии на языке JavaScript. Каждый сценарий назван также, как и преставление.

5.1 juice.js

async function getJuises() {

const response = await fetch("/api/Juices", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const juices = await response.json();

console.log(juices);

const rows = document.querySelector("tbody");

juices.forEach(juice => rows.append(row(juice)));

}

}

async function getJuice(id) {

const response = await fetch(`/api/Juices/${id}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const juice = await response.json();

document.getElementById("juiceId").value = juice.id;

document.getElementById("drinkTitle").value = juice.title;

document.getElementById("drinkVolume").value = juice.volume;

document.getElementById("juiceTaste").value = juice.taste;

document.getElementById("drinkPulp").value = juice.isPulp;

}

else {

const error = await response.json();

console.log(error);

}

}

async function getVendor(juiceId) {

const response = await fetch(`/api/Vendors/Juices/${juiceId}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok) {

var vendor = await response.json();

return vendor;

}

}

async function createJuice(titleIn, volumeIn, tasteIn, pulpIn) {

const response = await fetch("/api/Juices", {

method: "POST",

headers: { "Accept": "application/json", "Content-Type": "application/json" },

body: JSON.stringify({

title: titleIn,

volume: parseInt(volumeIn, 10),

taste: tasteIn,

isPulp: pulpIn

})

});

if (response.ok == true) {

const juice = await response.json();

document.querySelector("tbody").append(row(juice));

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function editJuice(idIn, titleIn, volumeIn, tasteIn, pulpIn) {

const response = await fetch("/api/Juices", {

method: "PUT",

headers: { "Accept": "application/json", "Content-Type": "application/json" },

body: JSON.stringify({

id: idIn,

title: titleIn,

volume: parseInt(volumeIn, 10),

taste: tasteIn,

pulp: pulpIn

})

});

if (response.ok === true) {

const juice = await response.json();

document.querySelector(`tr[data-rowid='${juice.id}']`).replaceWith(row(juice));

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function deleteJuice(id) {

var vendor = await getVendor(id);

if (vendor.length != 0) {

alert("Этот сок находится в автоматах");

return;

}

const response = await fetch(`/api/Juices/${id}`, {

method: "DELETE",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const juice = await response.json();

document.querySelector(`tr[data-rowid='${juice.id}']`).remove();

}

else {

const error = await response.json();

console.log(error.message);

}

}

function reset() {

document.getElementById("juiceId").value =

document.getElementById("drinkTitle").value = "";

}

function row(juice) {

const tr = document.createElement("tr");

tr.setAttribute("data-rowid", juice.id);

const titleTd = document.createElement("td");

titleTd.append(juice.title);

tr.append(titleTd);

const volumeTd = document.createElement("td");

volumeTd.append(juice.volume);

tr.append(volumeTd);

const tasteTd = document.createElement("td");

tasteTd.append(juice.taste);

tr.append(tasteTd);

const pulpTd = document.createElement("td");

pulpTd.append(juice.isPulp);

tr.append(pulpTd);

const linksTd = document.createElement("td");

const editLink = document.createElement("button");

editLink.append("Изменить");

editLink.className = "btn btn-info";

editLink.addEventListener("click", async () => await getJuice(juice.id));

linksTd.append(editLink);

const removeLink = document.createElement("button");

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await deleteJuice(juice.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

tr.addEventListener("click", async () => await click(juice));

return tr;

}

async function addJuice() {

id = document.getElementById("juiceId").value;

title = document.getElementById("drinkTitle").value;

volume = document.getElementById("drinkVolume").value;

taste = document.getElementById("juiceTaste").value;

pulp = document.getElementById("drinkPulp").checked;

if (title.length == 0) {

alert("Вы не ввели имя");

return;

}

volume = parseInt(volume, 10);

if (id === "")

await createJuice(title, volume, taste, pulp);

else

await editJuice(id, title, volume, taste, pulp);

reset();

}

async function click(juice) {

$("#vendorBody").empty();

var id = juice.id;

var vendors = await getVendor(id);

if (vendors.length === 0) {

alert("Сок нигде не выставлен");

return;

}

console.log(vendors);

var rows = document.querySelectorAll("tbody")[1];

vendors.forEach(v => rows.append(VendorRow(v)));

}

function VendorRow(vendor) {

const tr = document.createElement("tr");

const idId = document.createElement("td");

idId.append(vendor.id);

tr.append(idId);

const titleId = document.createElement("td");

titleId.append(vendor.title);

tr.append(titleId);

return tr;

}

getJuises();

5.2 vendor.js

async function GetVendors() {

const response = await fetch("/api/Vendors", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const vendors = await response.json();

console.log(vendors);

const rows = document.querySelectorAll("tbody")[3];

vendors.forEach(e => rows.append(VendorRow(e)));

await GetVendor(vendors[0].id);

var table = document.getElementById("mainTable");

var row = table.rows;

row[1].bgColor = "green";

var idField = document.getElementById("vendorId");

idField.value = vendors[0].id;

}

}

async function GetVendor(id) {

const response = await fetch(`/api/Vendors/${id}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const vendor = await response.json();

const juiceRows = document.querySelectorAll("tbody")[0];

const sodaRows = document.querySelectorAll("tbody")[1];

const alcoholRows = document.querySelectorAll("tbody")[2];

const juices = [];

for (var i = 0; i < vendor.juices.length; i++) {

juices.push(await GetJuice(vendor.juices[i]));

}

const sodas = [];

for (var i = 0; i < vendor.sodas.length; i++) {

sodas.push(await GetSoda(vendor.sodas[i]));

}

const alcohols = [];

for (var i = 0; i < vendor.alcohols.length; i++) {

alcohols.push(await GetAlcohol(vendor.alcohols[i]));

}

document.getElementById("title").value = vendor.title;

document.getElementById("vendorId").value = vendor.id;

juices.forEach(juice => juiceRows.append(JuiceRow(juice)));

sodas.forEach(soda => sodaRows.append(SodaRow(soda)));

alcohols.forEach(alcohol => alcoholRows.append(AlcoholRow(alcohol)));

}

}

async function GetJuice(id) {

const response = await fetch(`/api/Juices/${id}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const juice = await response.json();

return juice;

}

else {

const error = await response.json();

console.log(error);

}

}

async function GetSoda(id) {

const response = await fetch(`/api/Sodas/${id}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const soda = await response.json();

return soda;

}

else {

const error = await response.json();

console.log(error);

}

}

async function GetAlcohol(id) {

const response = await fetch(`/api/Alcohols/${id}`, {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const alco = await response.json();

return alco;

}

else {

const error = await response.json();

console.log(error);

}

}

//Настройка комбо боксов

async function setJuiceCmb() {

const response = await fetch("/api/Juices", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const juices = await response.json();

var info = document.getElementById("juiceInfo");

if (juices.length < 1) {

info.value = "Не добавлены соки";

return;

}

var cmb = document.getElementById("cmbJuice");

for (var i = 0; i < juices.length; i++) {

var t = document.createElement("option");

t.value = juices[i].id;

t.textContent = juices[i].title;

cmb.appendChild(t);

}

info.value = `Название: ${juices[0].title} | Объем: ${juices[0].volume} |

Вкус: ${juices[0].taste} | Наличие макоти: ${juices[0].isPulp}`;

}

}

async function setSodaCmb() {

const response = await fetch("/api/Sodas", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const sodas = await response.json();

var info = document.getElementById("sodaInfo");

if (sodas.length < 1) {

info.value = "Не добавлена газировка";

return;

}

var cmb = document.getElementById("cmbSoda");

for (var i = 0; i < sodas.length; i++) {

var t = document.createElement("option");

t.value = sodas[i].id;

t.textContent = sodas[i].title;

cmb.appendChild(t);

}

info.value = `Название: ${sodas[0].title} | Объем: ${sodas[0].volume} |

Тип: ${sodas[0].type} |

Количество пузырьков: ${sodas[0].bubbleCount}`;

}

}

async function setAlcoholCmb() {

const response = await fetch("/api/Alcohols", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const alcohols = await response.json();

var info = document.getElementById("alcoholInfo");

if (alcohols.length < 1) {

info.value = "Не добавлен алкоголь";

return;

}

var cmb = document.getElementById("cmbAlcohol");

for (var i = 0; i < alcohols.length; i++) {

var t = document.createElement("option");

t.value = alcohols[i].id;

t.textContent = alcohols[i].title;

cmb.appendChild(t);

}

info.value = `Название: ${alcohols[0].title} | Объем: ${alcohols[0].volume} |

Тип: ${alcohols[0].type} | Крепость: ${alcohols[0].fortress}`;

}

}

async function cmbJuiceChange() {

var e = document.getElementById("cmbJuice");

var id = e.options[e.selectedIndex].value;

var juice = await GetJuice(id);

var info = document.getElementById("juiceInfo");

info.value = `Название: ${juice.title} | Объем: ${juice.volume} |

Вкус: ${juice.taste} | Наличие макоти: ${juice.isPulp}`;

}

async function cmbSodaChange() {

var e = document.getElementById("cmbSoda");

var id = e.options[e.selectedIndex].value;

var soda = await GetSoda(id);

var info = document.getElementById("sodaInfo");

info.value = `Название: ${soda.title} | Объем: ${soda.volume} |

Тип: ${soda.type} |

Количество пузырьков: ${soda.bubbleCount}`;

}

async function cmbAlcoholChange() {

var e = document.getElementById("cmbAlcohol");

var id = e.options[e.selectedIndex].value;

var alcohol = await GetAlcohol(id);

var info = document.getElementById("alcoholInfo");

info.value = `Название: ${alcohol.title} | Объем: ${alcohol.volume} |

Тип: ${alcohol.type} | Крепость: ${alcohol.fortress}`;

}

//Добавление напитков в списки для добавления

function JuiceRow(juice) {

const tr = document.createElement("tr");

tr.setAttribute("data-rowid", juice.id);

const idId = document.createElement("td");

idId.append(juice.id);

tr.append(idId);

const nameId = document.createElement("td");

nameId.append(juice.title);

tr.append(nameId);

const tasteId = document.createElement("td");

tasteId.append(juice.taste);

tr.append(tasteId);

const linksTd = document.createElement("td");

const removeLink = document.createElement("button");

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await DeleteItem(juice.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

return tr;

}

function SodaRow(soda) {

const tr = document.createElement("tr");

tr.setAttribute("data-rowid", soda.id);

const idId = document.createElement("td");

idId.append(soda.id);

tr.append(idId);

const nameId = document.createElement("td");

nameId.append(soda.title);

tr.append(nameId);

const typeId = document.createElement("td");

typeId.append(soda.type);

tr.append(typeId);

const linksTd = document.createElement("td");

const removeLink = document.createElement("button");

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await DeleteItem(soda.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

return tr;

}

function AlcoholRow(alcohol) {

const tr = document.createElement("tr");

tr.setAttribute("data-rowid", alcohol.id);

const idId = document.createElement("td");

idId.append(alcohol.id);

tr.append(idId);

const nameId = document.createElement("td");

nameId.append(alcohol.title);

tr.append(nameId);

const typeId = document.createElement("td");

typeId.append(alcohol.type);

tr.append(typeId);

const linksTd = document.createElement("td");

const removeLink = document.createElement("button");

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await DeleteItem(alcohol.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

return tr;

}

async function DeleteItem(id) {

document.querySelector(`tr[data-rowid='${id}']`).remove();

}

async function AddJuice() {

var e = document.getElementById("cmbJuice");

var id = e.options[e.selectedIndex].value;

var juice = await GetJuice(id);

document.querySelectorAll("tbody")[0].append(JuiceRow(juice));

}

async function AddSoda() {

var e = document.getElementById("cmbSoda");

var id = e.options[e.selectedIndex].value;

var soda = await GetSoda(id);

document.querySelectorAll("tbody")[1].append(SodaRow(soda));

}

async function AddAlcohol() {

var e = document.getElementById("cmbAlcohol");

var id = e.options[e.selectedIndex].value;

var alcohol = await GetAlcohol(id);

document.querySelectorAll("tbody")[2].append(AlcoholRow(alcohol));

}

//Работа с автоматом

function VendorRow(vendor) {

const tr = document.createElement("tr");

tr.setAttribute("data-rowid", vendor.id);

const idId = document.createElement("td");

idId.append(vendor.id);

tr.append(idId);

const nameId = document.createElement("td");

nameId.append(vendor.title);

tr.append(nameId);

const linksTd = document.createElement("td");

const removeLink = document.createElement("button");

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await DeleteVendor(vendor.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

tr.addEventListener("click", async () => await click(vendor, tr));

return tr;

}

async function DeleteVendor(id) {

const response = await fetch(`/api/Vendors/${id}`, {

method: "DELETE",

headers: { "Accept": "application/json" }

});

if (response.ok === true) {

const vendor = await response.json();

document.querySelector(`tr[data-rowid='${vendor.id}']`).remove();

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function CreateVendor(title, juices, sodas, alcohols) {

const response = await fetch("/api/Vendors", {

method: "POST",

headers: { "Accept": "application/json", "Content-Type": "application/json" },

body: JSON.stringify({

title: title,

juices: juices,

sodas: sodas,

alcohols: alcohols

})

});

if (response.ok == true) {

const vendor = await response.json();

document.querySelectorAll("tbody")[3].append(VendorRow(vendor));

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function EditVendor(id, title, juices, sodas, alcohols) {

const response = await fetch("/api/Vendors", {

method: "PUT",

headers: { "Accept": "application/json", "Content-Type": "application/json" },

body: JSON.stringify({

id: id,

title: title,

juices: juices,

sodas: sodas,

alcohols: alcohols

})

});

if (response.ok == true) {

const vendor = await response.json();

document.querySelector(`tr[data-rowid='${vendor.id}']`).replaceWith(VendorRow(vendor));

}

else {

const error = await response.json();

console.log(error.message);

}

}

function Reset() {

$("#juiceBody").empty();

$("#sodaBody").empty();

$("#alcoholBody").empty();

var table = document.getElementById("mainTable");

var row = table.rows;

for (var i = 0; i < row.length; i++) {

row[i].bgColor = "white";

}

document.getElementById("title").value = "";

var idField = document.getElementById("vendorId");

idField.value = "";

}

async function AddVendor() {

var title = document.getElementById("title").value;

let juices = [];

let sodas = [];

let alcohols = [];

const juiceRows = document.querySelectorAll("tbody")[0].rows;

const sodaRows = document.querySelectorAll("tbody")[1].rows;

const alcoholRow = document.querySelectorAll("tbody")[2].rows;

for (var i = 0; i < juiceRows.length; i++) {

var td = juiceRows[i].getElementsByTagName("td");

juices.push(td[0].innerText);

}

for (var i = 0; i < sodaRows.length; i++) {

var td = sodaRows[i].getElementsByTagName("td");

sodas.push(td[0].innerText);

}

for (var i = 0; i < alcoholRow.length; i++) {

var td = alcoholRow[i].getElementsByTagName("td");

alcohols.push(td[0].innerText);

}

if (title.length === 0) {

alert("Вы не ввели название");

return;

}

if (juices.length === 0 && sodas.length === 0 && alcohols.length === 0) {

alert("Вы не выбрали напитки");

return;

}

var id = document.getElementById("vendorId").value;

if (id.length === 0) {

await CreateVendor(title, juices, sodas, alcohols);

}

else {

await EditVendor(id, title, juices, sodas, alcohols);

}

Reset();

}

async function click(vendor, tr) {

Reset();

var idField = document.getElementById("vendorId");

idField.value = vendor.id;

var table = document.getElementById("mainTable");

var row = table.rows;

for (var i = 0; i < row.length; i++) {

row[i].bgColor = "white";

}

var id = vendor.id;

tr.bgColor = "green";

await GetVendor(id);

}

//Поиск

async function Search() {

const response = await fetch("/api/Vendors/Search", {

method: "GET",

headers: { "Accept": "application/json" }

});

if (response.ok) {

var vendors = await response.json();

console.log(vendors);

if (vendors.length == 0) {

alert("Таких автоматов нет");

return;

}

Reset();

$("#vendorBody").empty();

const rows = document.querySelectorAll("tbody")[3];

vendors.forEach(e => rows.append(VendorRow(e)));

await GetVendor(vendors[0].id);

var table = document.getElementById("mainTable");

var row = table.rows;

row[1].bgColor = "green";

var idField = document.getElementById("exhibitionId");

idField.value = vendors[0].id;

}

}

async function SearchReset() {

Reset();

$("#vendorBody").empty();

await GetVendors();

}

GetVendors();

setJuiceCmb();

setSodaCmb();

setAlcoholCmb();

6. WebApi

Для взаимодействия с MongoDB было создано Api средствами ASP.Net MVC. Оно представляет из себя набор из 4 методов, реализующий CRUD. Частичный листинг:

private static void MapJuice(WebApplication app, IMongoDatabase db, string collectionName)

{

//Добавим конечную точку, которая обрабатывает GET по пути

app.MapGet($"/api/{collectionName}", () =>

db.GetCollection<Juice>(collectionName).Find("{}").ToListAsync());

//Если обратиться по id, то получаем другую конечную точку

app.MapGet("/api/" + collectionName + "/{id}", async (string id) =>

{

var juice = await db.GetCollection<Juice>(collectionName)

.Find(r => r.Id == id)

.FirstOrDefaultAsync();

// если не найден, отправляем статусный код и сообщение об ошибке

if (juice == null) return Results.NotFound(new { message = "Пользователь не найден" });

// если пользователь найден, отправляем его

return Results.Json(juice);

});

//Если получаем Delete по этому адресу, вызывается это

app.MapDelete($"/api/" + collectionName + "/{id}", async (string id) =>

{

var juice = await db.GetCollection<Juice>(collectionName).FindOneAndDeleteAsync(r => r.Id == id);

// если не найден, отправляем статусный код и сообщение об ошибке

if (juice is null) return Results.NotFound(new { message = "Пользователь не найден" });

return Results.Json(juice);

});

//При получении запроса с методом POST, идем сюда

app.MapPost($"/api/{collectionName}", async (Juice juice) => {

// добавляем пользователя в список

await db.GetCollection<Juice>(collectionName).InsertOneAsync(juice);

return juice;

});

//Если приходит put по этому адресу, идем сюда

app.MapPut($"/api/{collectionName}", async (Juice juiceData) => {

var juice = await db.GetCollection<Juice>(collectionName)

.FindOneAndReplaceAsync(p => p.Id == juiceData.Id, juiceData, new() { ReturnDocument = ReturnDocument.After });

if (juice == null)

return Results.NotFound(new { message = "Пользователь не найден" });

return Results.Json(juice);

});

}

7. Таблица тестов

Результаты тестирования работоспособности реализованного приложения приведены в таблице 4.

Таблица 4 – Тесты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 1 | Соки | Добавление | Сок, 250, Яблоко, true | Обновление таблицы |
| 2 | Редактирование | Новый сок, 500, Апельсин, false | Обновление таблицы |
| 3 | Удаление | Удаление записи №4 | Обновление таблицы |
| 4 | Газировки | Добавление | Сода, 500, Среднегазированная, 1500 | Обновление таблицы |
| 5 | Редактирование | Новая Сода, 1000, Сильногазированная, 2500 | Обновление таблицы |
| 6 | Удаление | Удаление записи №3 | Обновление таблицы |
| 7 | Алкоголь | Добавление | Алкоголь, 1250, Вино, 11 | Обновление таблицы |
| 8 | Редактирование | Новый Алкоголь, 1750, Виски, 21 | Обновление таблицы |
| 9 | Удаление | Удаление записи №3 | Обновление таблицы |
| 10 | Торговый автомат | Добавление | 638b448dd16476c40b82fa94, 638b486d725808b90b4874b3, 638b4be03255b84528960021,  Автомат | Обновление таблицы |
| 11 | Редактирование | 638b4be03255b84528960021,  Новый Автомат | Обновление таблицы |
| 12 | Удаление | Удаление записи №4 | Обновление таблицы |
| 13 | Первый запрос | Выполнение запроса | 6388816ee72cd74701899f92 | Документ |
| 14 | Второй запрос | Выполнение запроса |  | Документ |

8. Результаты тестирования

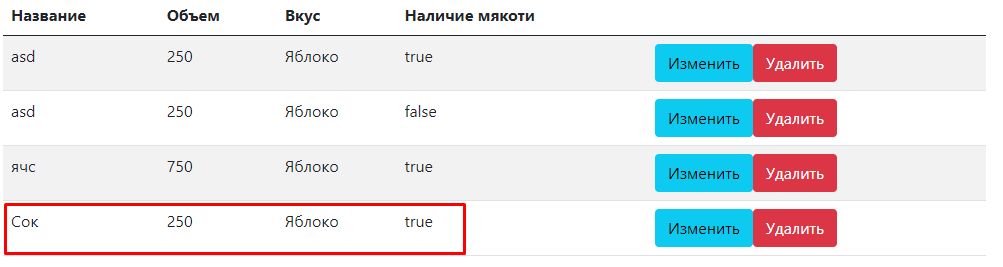


Рисунок 4 – Результат теста №1

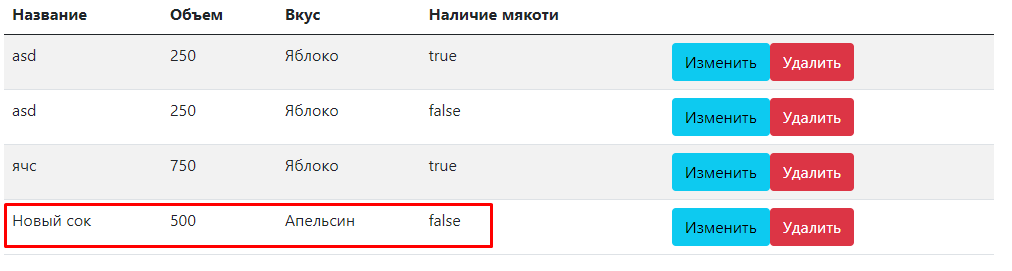


Рисунок 5 – Результат теста №2

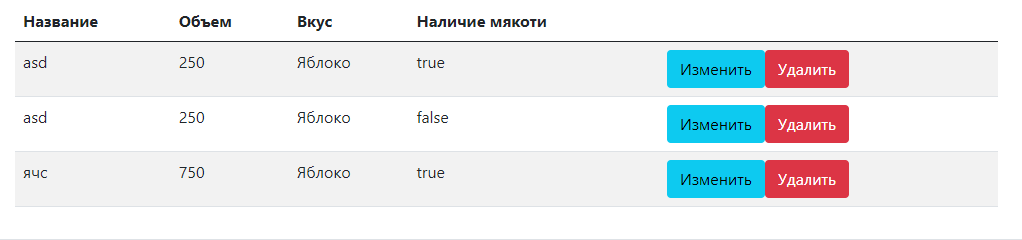


Рисунок 6 – Результат теста №3

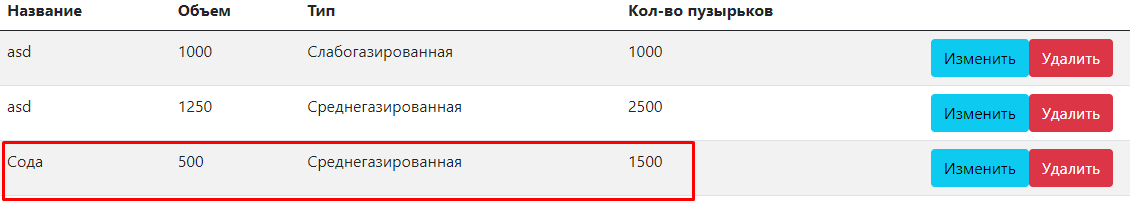


Рисунок 7 – Результат теста №4



Рисунок 8 – Результат теста №5

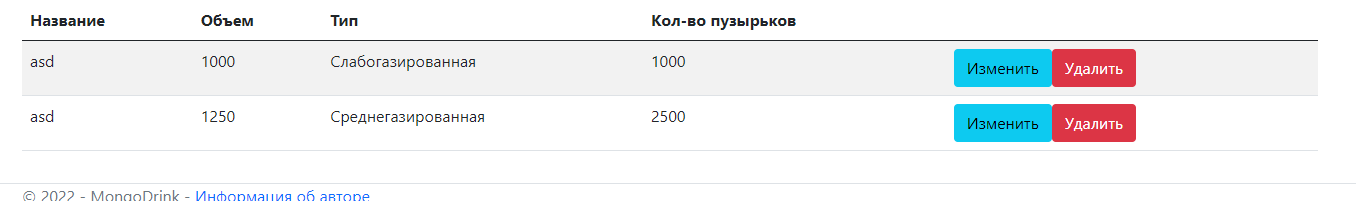


Рисунок 8 – Результат теста №6



Рисунок 9 – Результат теста №7

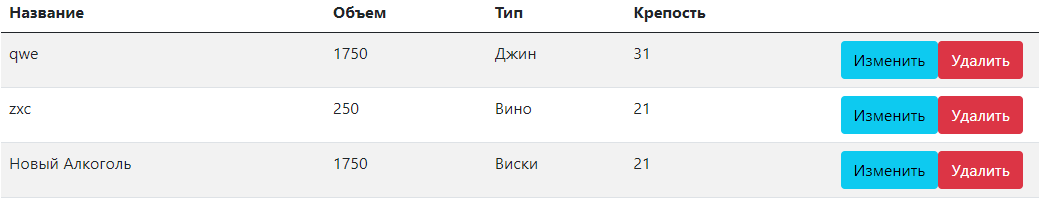


Рисунок 10 – Результат теста №8

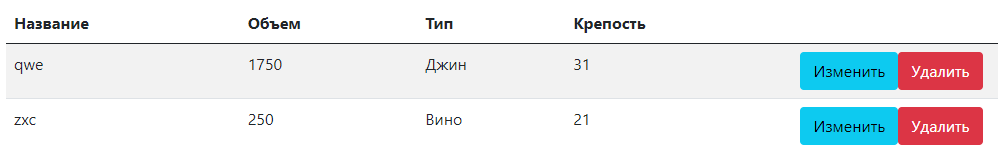


Рисунок 11 – Результат теста №9

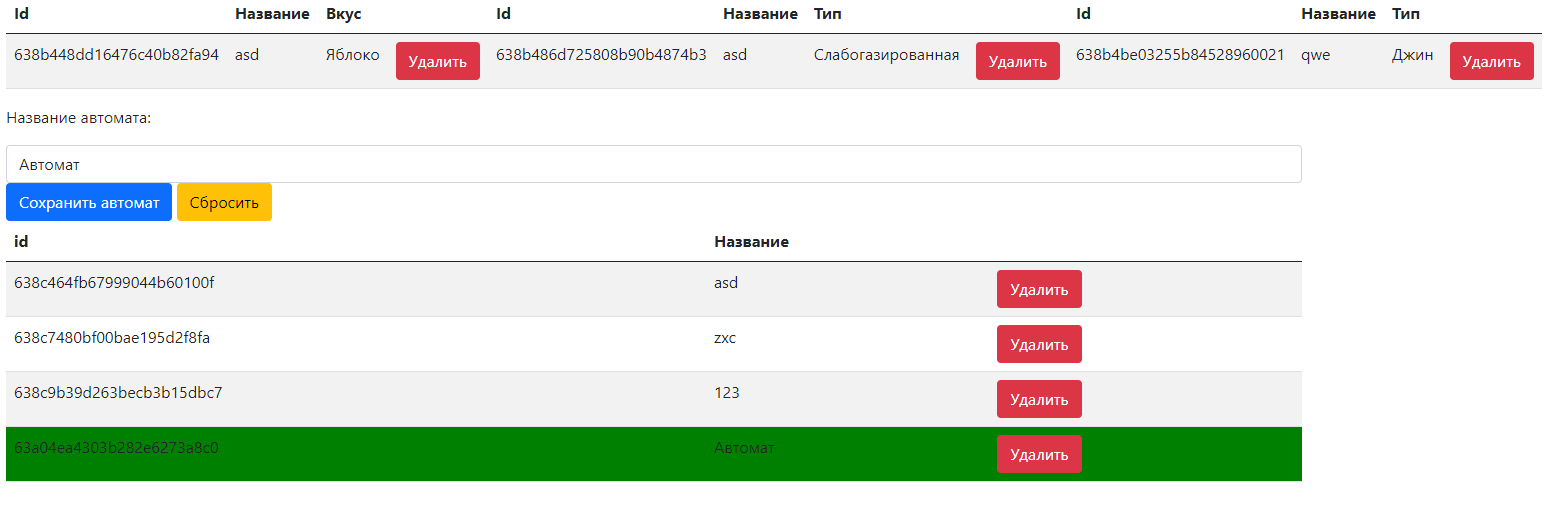


Рисунок 12 – Результат теста №10

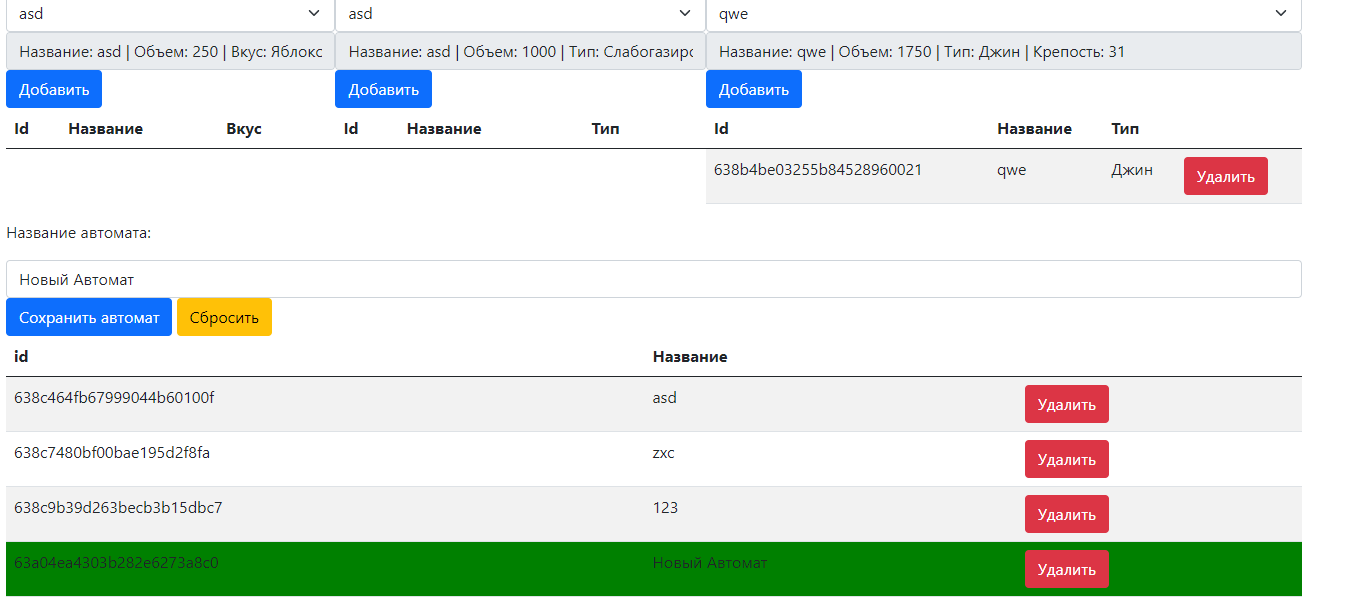


Рисунок 13 – Результат теста №11

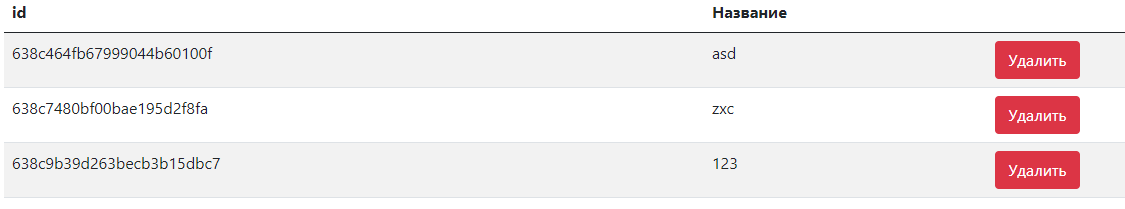


Рисунок 14 – Результат теста №12

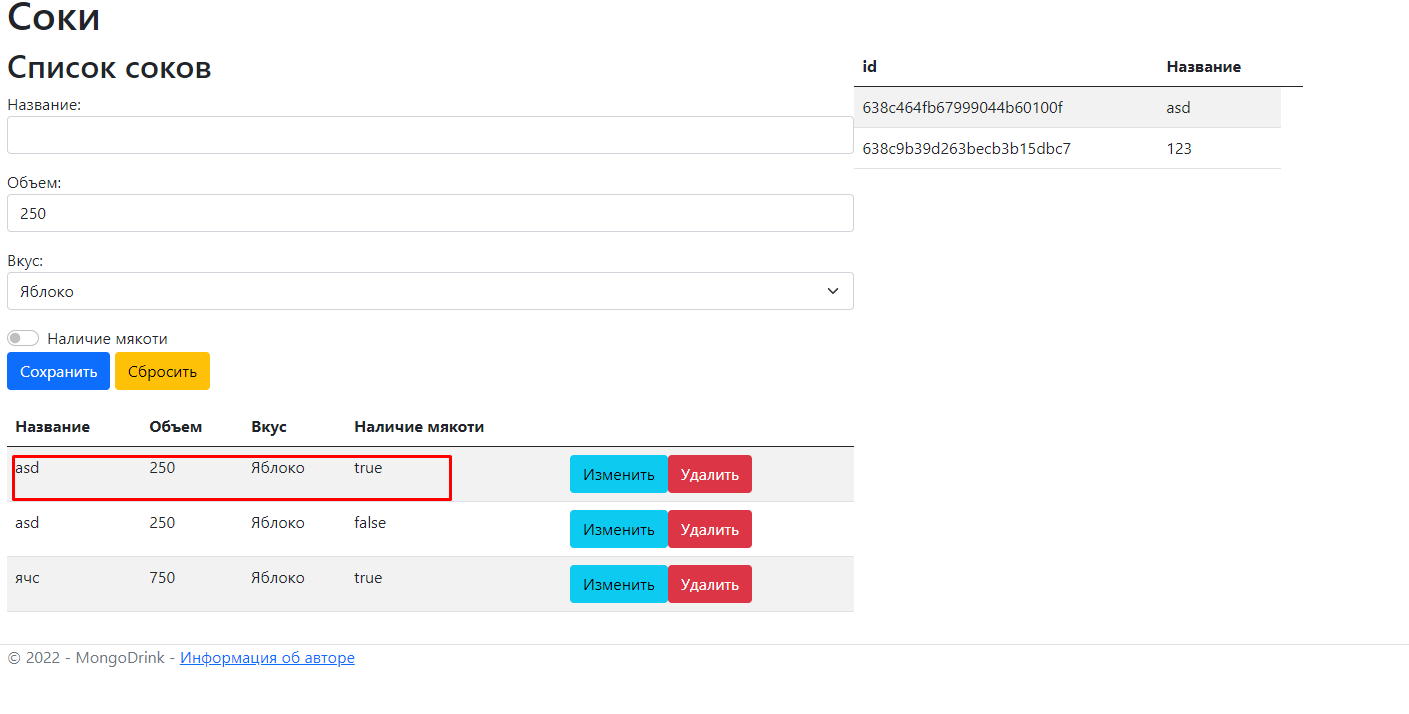


Рисунок 15 – Результат теста №13

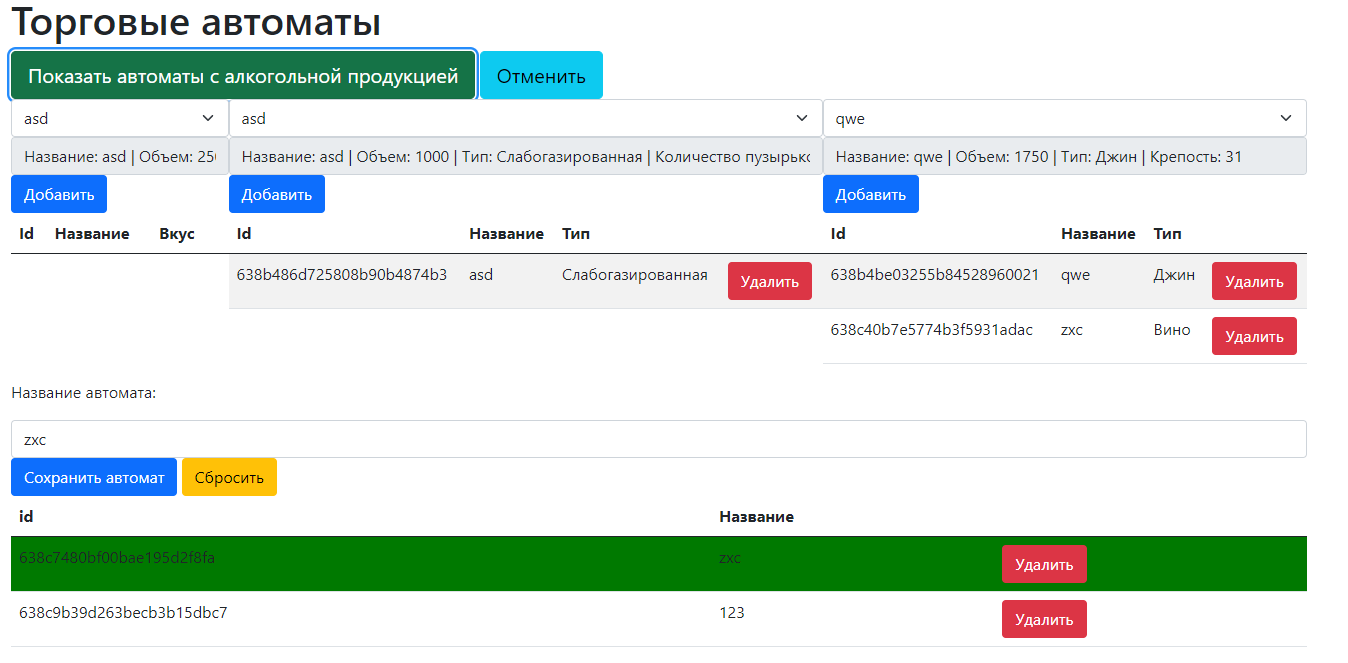


Рисунок 16 – Результат теста №14

Заключение

В процессе разработки веб-клиента были получены и закреплены практические навыки проектирования баз данных, получены новые навыки в использовании фреймворка ASP NET MVC.

В ходе выполнения проекта было разработано 14 тестов, где проверялась работоспособность реализованных операций и запросов.

Из проблем проекта стоит выделить повторяемость кода, которую можно решить путем шаблонизации некоторых участков страниц контроллеров.

Проект может быть продолжен, если исправить недостатки текущей версии приложения, улучшить положительные стороны и добавить новые функции.

Поставленные проектом задачи выполнены, все цели достигнуты.

Список использованных источников

1. Стасышин В. М., Стасышина Т. Л. Базы данных: Технологии доступа 2-е изд., испр. и доп. Учебное пособие для академического бакалавриата, 2019. 164 c. ISBN: 978-5-534-09888-4 Серия: Профессиональное образование. URL: https://biblio-online.ru/viewer/bazy-dannyhtehnologii-dostupa-415342#page/1.
2. Голицына О. Л. Базы данных: учебное пособие для вузов по направлению 230700 "Прикладная информатика" / О. Л. Голицына, Н. В. Максимов, И. И. Попов, 2012. 399 с.
3. Сосинская С. С. Использование языка C# в различных информационных технологиях: учебник для студентов вузов, обучающихся по направлению "Информационные системы и технологии / Сосинская, 2014. 367 с.
4. Шубина М.А. Управление данными: учебное пособие для студентов направлений подготовки 09.03.02 и 09.04.02 «Информационные системы и технологии» / Шубина М.А, 2016. 132 c. ISBN: 978-5-9239-0832-9. URL: https://e.lanbook.com/book/74029#book\_name.
5. Сосинская С.С. «Разработка клиента по технологии MVC»: Методические указания по выполнению курсового проекта» Электронный каталог кафедры вычислительной техники (дата обращения: 07.12.2021).
6. Руководство по ASP NET Core 5 [Электронный ресурс] // Метанит [Сайт], URL: https://metanit.com/sharp/aspnet5/ (Дата обращения: 25.11.2021)
7. MVC framework: большое введение для начинающих [Электронный ресурс] // Хабр [Сайт], URL: https://habr.com/ru/post/49718/ (Дата обращения: 1.12.2021)