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 |

|  |
| --- |
| 1.07.00.00 ПЗ |
| обозначение документа |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСМб-19-1 |  |  |  | Ю.А. Михейко |
|  |  | 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 | Михейко Ю.А. | | | | | | | | | | | | | | | | | |
|  | (фамилия, инициалы) | | | | | | | | | | | | | | | | | |
| Тема проекта | | Разработка клиент-серверного приложения для работы с нереляционной базой данных | | | | | | | | | | | | | | | | |
| Исходные данные | | | Вариант 7 | | | | | | | | | | | | | | | |
| 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 | | | | | | | | |  | | | |  | | Ю.А. Михейко | | | |
|  | | | | | | | | | 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](#_Toc122369769)

[Индивидуальный вариант 5](#_Toc122369770)

[1. Структура базы данных 6](#_Toc122369771)

[1.1 Концептуальная модель данных 6](#_Toc122369772)

[1.2 Физическая модель данных 6](#_Toc122369773)

[2. Проектирование информационной системы 8](#_Toc122369774)

[3. Контроллеры 9](#_Toc122369775)

[4. Представления 10](#_Toc122369776)

[4.1 Index 10](#_Toc122369777)

[4.2 Cat 10](#_Toc122369778)

[4.3 Exhibition 11](#_Toc122369779)

[5 Сценарии 14](#_Toc122369780)

[5.1 cat.js 14](#_Toc122369781)

[5.2 exhibition.js 17](#_Toc122369782)

[6. WebApi 24](#_Toc122369783)

[7. Таблица тестов 25](#_Toc122369784)

[8. Результаты тестирования 27](#_Toc122369785)

[Заключение 31](#_Toc122369786)

[Список использованных источников 32](#_Toc122369787)

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 была взята выставка животных, включающий следующие сущности:

1. Коты;
2. Собаки;
3. Птицы;
4. Выставки.

Индивидуальный вариант

Имеются коты (кличка, возраст, цвет, пора), собаки (кличка, возраст, цвет, пора) и птицы (кличка, возраст, цвет, пора). Из этих объектов составляются выставки (названия, коты, собаки, птицы)

Выходные документы:

выдать список выставок, в которых учувствует кошка;

выдать список выставок, в которых находится определенное количество котов, собак и птиц.

1. Структура базы данных

* 1. Концептуальная модель данных

Концептуальная модель данных содержит 4 сущности и 3 связи. Первая модель описывает отношение между сущностями Кошки и Выставка. Выставка не обаятельно должна иметь кошку, но кошка обязательно должна иметь выставку. Они имеют связь «один ко многим». Вторая модель описывает отношение между сущностями Собаки и Выставка. Выставка не обаятельно должна иметь собаку, но собака обязательно должна иметь выставку. Они имеют связь «один ко многим». Третья модель описывает отношение между сущностями Птицы и Выставка. Выставка не обаятельно должна иметь птицу, но птица обязательно должна иметь выставку. Они имеют связь «один ко многим».

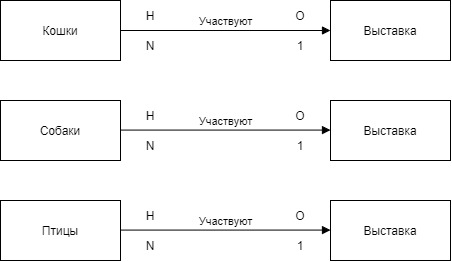


Рисунок 1 – Концептуальная модель данных

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 «Кошки»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| name | String | Нет |  |
| age | Int32 | Нет |  |
| color | String | Нет |  |
| breed | String | Нет |  |

Таблица 2 – Структура модели «Собаки»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| name | String | Нет |  |
| age | Int32 | Нет |  |
| color | String | Нет |  |
| breed | String | Нет |  |

Таблица 3 – Структура модели «Птицы»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| name | String | Нет | FK |
| age | Int32 | Нет | FK |
| color | String | Нет |  |
| breed | String | Нет |  |

Таблица 4 – Структура модели «Выставки»

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип | Null | Примечание |
| \_id | ObjectId | Нет | PK, Index |
| Title | String | Нет |  |
| Cats | Array< ObjectId > | Да | FK |
| Dogs | Array< ObjectId > | Да | FK |
| Birds | Array< ObjectId > | Да | FK |

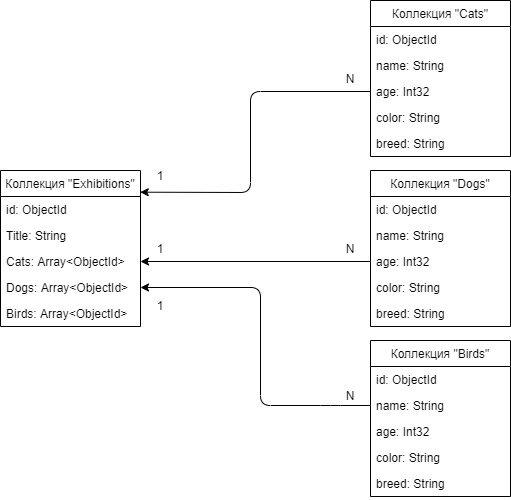


Рисунок 2 – Физическая модель данных «buildDB»

2. Проектирование информационной системы

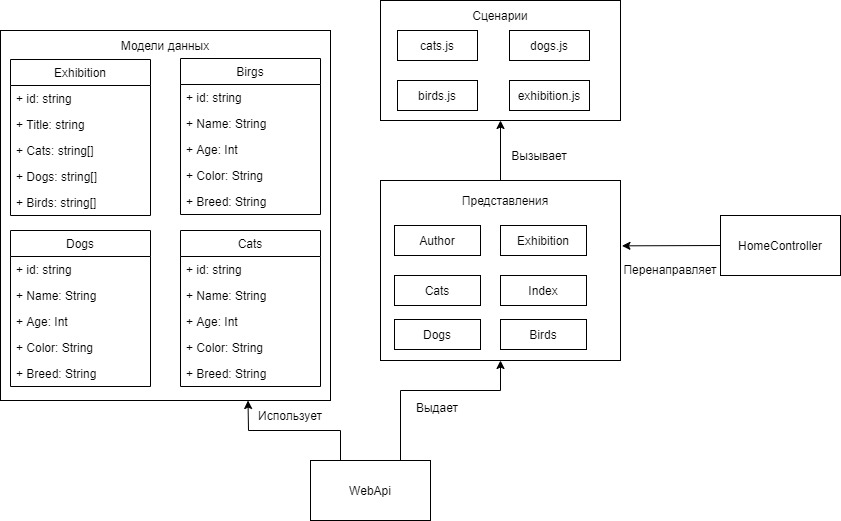


Рисунок 3 – Модель ИС

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 Cat()

{

return View();

}

public IActionResult Author()

{

return View();

}

public IActionResult Dog() => View();

public IActionResult Bird() => View();

public IActionResult Exhibition() => 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. Cat позволяет взаимодействовать с моделью Cat и выполнять выдачу первого документа. Dog позволяет взаимодействовать с моделью Dog. Bird позволяет взаимодействовать с моделью Bird. Exhibition позволяет взаимодействовать с моделью Exhibition 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 7.</p>

<p>Предметная область: Выставка животных</p>

<p>Модель данных включает в себя:</p>

<ul>

<li>Котов</li>

<li>Собак</li>

<li>Птиц</li>

<li>Выставки</li>

</ul>

</div>

4.2 Cat

@{

ViewData["Title"] = "Кошки";

}

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

<div class="flex-container">

<div class="first">

<div>

<h2>Список кошек</h2>

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

<p>

Имя:<br />

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

</p>

<p>

Возраст:<br />

<input class="form-control" id="catAge" type="number" min="1"

max="15" value="3" step="2" />

</p>

<p>

Цвет:<br />

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

<option>Серый</option>

<option>Черный</option>

<option>Белый</option>

<option>Рыжий</option>

</select>

</p>

<p>

Порода:<br />

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

<option>Азиатская</option>

<option>Бенгальская</option>

<option>Чаузи</option>

<option>Кипреская</option>

<option>Фолдекс</option>

<option>Канаани</option>

<option>Корат</option>

<option>Менуэт</option>

<option>Раас</option>

<option>Сококе</option>

</select>

</p>

<p>

<button class="btn btn-primary" id="saveBtn" onclick="addCat()">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">Имя</th>

<th scope="col">Возраст</th>

<th scope="col">Цвет</th>

<th scope="col">Порода</th>

<th></th>

</tr>

</thead>

<tbody>

</tbody>

</table>

</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="exhibitionBody">

</tbody>

</table>

</div>

</div>

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

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

4.3 Exhibition

@{

ViewData["Title"] = "Выставки";

}

<h1>Поиск</h1>

<div class="input-group mb-3">

<input id="catCount" type="number" class="form-control" placeholder="Количетсво кошек" aria-label="Количетсво кошек" aria-describedby="button-addon2">

<input id="dogCount" type="number" class="form-control" placeholder="Количетсво собак" aria-label="Количетсво собак" aria-describedby="button-addon2">

<input id="birdCount" type="number" class="form-control" placeholder="Количетсво птиц" aria-label="Количетсво птиц" aria-describedby="button-addon2">

<button onclick="Search()" class="btn btn-success" type="button" id="button-addon2">Найти</button>

<button onclick="SearchClear()" class="btn btn-info" type="button" id="button-addon2">Сбросить</button>

</div>

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

<div class="flex-container">

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

<div class="first">

<select id="cmbCat" class="form-select" onchange="cmbCatChange()"></select>

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

<button class="btn btn-primary" id="btnCatAdd" onclick="AddCat()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Порода</th>

<th>Кличка</th>

<th></th>

</tr>

</thead>

<tbody id="catBody"></tbody>

</table>

</div>

<div class="second">

<select id="cmbDog" class="form-select" onchange="cmbDogChange()"></select>

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

<button class="btn btn-primary" id="btnDogAdd" onclick="AddDog()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Порода</th>

<th>Кличка</th>

<th></th>

</tr>

</thead>

<tbody id="dogBody"></tbody>

</table>

</div>

<div class="third">

<select id="cmbBird" class="form-select" onchange="cmbBirdChange()"></select>

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

<button class="btn btn-primary" id="btnBirdAdd" onclick="AddBird()">Добавить</button>

<table class="table table-striped">

<thead>

<tr>

<th>Id</th>

<th>Порода</th>

<th>Кличка</th>

<th></th>

</tr>

</thead>

<tbody id="birdBody"></tbody>

</table>

</div>

</div>

<div>

<p>Название выставки:</p>

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

<button class="btn btn-primary" id="btnExhibitionAdd" onclick="AddExhibition()">Сохранить выставку</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="exhibitionBody">

</tbody>

</table>

</div>

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

<script src="~/js/exhibition.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 cat.js

async function getCats() {

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

method: "GET",

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

});

if (response.ok === true) {

const cats = await response.json();

console.log(cats);

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

cats.forEach(cat => rows.append(row(cat)));

}

}

async function getCat(id) {

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

method: "GET",

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

});

if (response.ok === true) {

const cat = await response.json();

document.getElementById("catId").value = cat.id;

document.getElementById("catName").value = cat.name;

document.getElementById("catAge").value = cat.age;

document.getElementById("catColor").value = cat.color;

document.getElementById("catBreed").value = cat.breed;

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function getExhibition(catId) {

const response = await fetch(`/api/Exhibitions/Cats/${catId}`, {

method: "GET",

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

});

if (response.ok) {

var exbs = await response.json();

console.log(exbs);

return exbs;

}

}

async function createCat(nameIn, ageIn, colorIn, breedIn) {

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

method: "POST",

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

body: JSON.stringify({

name: nameIn,

age: ageIn,

color: colorIn,

breed: breedIn

})

});

if (response.ok == true) {

const cat = await response.json();

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

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function editCat(idIn, nameIn, ageIn, colorIn, breedIn) {

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

method: "PUT",

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

body: JSON.stringify({

id: idIn,

name: nameIn,

age: ageIn,

color: colorIn,

breed: breedIn

})

});

if (response.ok === true) {

const cat = await response.json();

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

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function deleteCat(id) {

var exb = await getExhibition(id);

if (exb.length > 0) {

alert("Этот кот участвует в выставках");

return;

}

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

method: "DELETE",

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

});

if (response.ok === true) {

const cat = await response.json();

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

}

else {

const error = await response.json();

console.log(error.message);

}

}

function reset() {

document.getElementById("catId").value =

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

}

function row(cat) {

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

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

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

nameTd.append(cat.name);

tr.append(nameTd);

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

ageTd.append(cat.age);

tr.append(ageTd);

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

colorTd.append(cat.color);

tr.append(colorTd);

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

breedTd.append(cat.breed);

tr.append(breedTd);

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

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

editLink.append("Изменить");

editLink.className = "btn btn-info";

editLink.addEventListener("click", async () => await getCat(cat.id));

linksTd.append(editLink);

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

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await deleteCat(cat.id));

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

linksTd.append(removeLink);

tr.appendChild(linksTd);

return tr;

}

async function addCat() {

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

name = document.getElementById("catName").value;

age = document.getElementById("catAge").value;

color = document.getElementById("catColor").value

breed = document.getElementById("catBreed").value;

if (name.length == 0) {

alert("Вы не ввели имя");

return;

}

if (age.length == 0) {

alert("Вы не ввели возраст");

return;

}

if (id === "")

await createCat(name, age, color, breed);

else

await editCat(id, name, age, color, breed);

reset();

}

async function click(cat) {

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

var id = cat.id;

var exb = await getExhibition(id);

if (exb.length === 0) {

alert("Кот нигде не учавствует");

return;

}

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

exb.forEach(ex => rows.append(ExhibitionsRow(ex)));

}

function ExhibitionsRow(exhibition) {

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

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

idId.append(exhibition.id);

tr.append(idId);

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

titleId.append(exhibition.title);

tr.append(titleId);

return tr;

}

getCats();

5.2 exhibition.js

async function GetExhibitions() {

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

method: "GET",

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

});

if (response.ok === true) {

const exhibitions = await response.json();

console.log(exhibitions);

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

exhibitions.forEach(e => rows.append(ExhibitionsRow(e)));

await GetExhibition(exhibitions[0].id);

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

var row = table.rows;

row[1].bgColor = "green";

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

idField.value = exhibitions[0].id;

}

}

async function GetExhibition(id) {

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

method: "GET",

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

});

if (response.ok === true) {

const exhibition = await response.json();

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

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

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

const cats = [];

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

cats.push(await GetCat(exhibition.cats[i]));

}

const dogs = [];

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

dogs.push(await GetDog(exhibition.dogs[i]));

}

const birds = [];

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

birds.push(await GetBird(exhibition.birds[i]));

}

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

cats.forEach(cat => catRows.append(AnimalRow(cat)));

dogs.forEach(dog => dogRows.append(AnimalRow(dog)));

birds.forEach(bird => birdRows.append(AnimalRow(bird)));

}

}

async function GetCat(id) {

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

method: "GET",

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

});

if (response.ok === true) {

const cat = await response.json();

return cat;

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function GetDog(id) {

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

method: "GET",

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

});

if (response.ok === true) {

const dog = await response.json();

return dog;

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function GetBird(id) {

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

method: "GET",

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

});

if (response.ok === true) {

const bird = await response.json();

return bird;

}

else {

const error = await response.json();

console.log(error.message);

}

}

//Настройка комбо боксов

async function setCatCmb() {

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

method: "GET",

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

});

if (response.ok === true) {

const cats = await response.json();

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

if (cats.length < 1) {

info.value = "Не добавлены коты";

return;

}

var cmbCat = document.getElementById("cmbCat");

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

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

t.value = cats[i].id;

t.textContent = cats[i].name;

cmbCat.appendChild(t);

}

info.value = `Кличка: ${cats[0].name} | Возраст: ${cats[0].age} | Цвет: ${cats[0].color} |

Порода: ${cats[0].breed}`

}

}

async function setDogCmb() {

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

method: "GET",

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

});

if (response.ok === true) {

const dogs = await response.json();

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

if (dogs.length < 1) {

info.value = "Не добавлены собаки";

return;

}

var cmbDog = document.getElementById("cmbDog");

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

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

t.value = dogs[i].id;

t.textContent = dogs[i].name;

cmbDog.appendChild(t);

}

info.value = `Кличка: ${dogs[0].name} | Возраст: ${dogs[0].age} | Цвет: ${dogs[0].color} |

Порода: ${dogs[0].breed}`

}

}

async function setBirdCmb() {

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

method: "GET",

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

});

if (response.ok === true) {

const birds = await response.json();

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

if (birds.length < 1) {

info.value = "Не добавлены собаки";

return;

}

var cmbDog = document.getElementById("cmbBird");

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

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

t.value = birds[i].id;

t.textContent = birds[i].name;

cmbDog.appendChild(t);

}

info.value = `Кличка: ${birds[0].name} | Возраст: ${birds[0].age} | Цвет: ${birds[0].color} |

Порода: ${birds[0].breed}`

}

}

async function cmbCatChange() {

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

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

var cat = await GetCat(id);

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

info.value = `Кличка: ${cat.name} | Возраст: ${cat.age} | Цвет: ${cat.color} |

Порода: ${cat.breed}`;

}

async function cmbDogChange() {

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

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

var dog = await GetDog(id);

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

info.value = `Кличка: ${dog.name} | Возраст: ${dog.age} | Цвет: ${dog.color} |

Порода: ${dog.breed}`;

}

async function cmbBirdChange() {

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

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

var bird = await GetBird(id);

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

info.value = `Кличка: ${bird.name} | Возраст: ${bird.age} | Цвет: ${bird.color} |

Порода: ${bird.breed}`;

}

//Добавление животных в списоки для добавления

function AnimalRow(animal) {

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

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

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

idId.append(animal.id);

tr.append(idId);

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

nameId.append(animal.name);

tr.append(nameId);

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

breedId.append(animal.breed);

tr.append(breedId);

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

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

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

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

linksTd.append(removeLink);

tr.appendChild(linksTd);

return tr;

}

async function DeleteItem(id) {

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

}

async function AddCat() {

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

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

var rowCat = document.getElementsByTagName("tbody")[0].rows;

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

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

if (id === td[0].innerText) {

alert("Этот кот уже участвует в выставке");

return;

}

}

var cat = await GetCat(id);

document.querySelectorAll("tbody")[0].append(AnimalRow(cat));

}

async function AddDog() {

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

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

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

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

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

if (id === td[0].innerText) {

alert("Эта собака уже участвует в выставке");

return;

}

}

var dog = await GetDog(id);

document.querySelectorAll("tbody")[1].append(AnimalRow(dog));

}

async function AddBird() {

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

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

var rowBird = document.getElementsByTagName("tbody")[2].rows;

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

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

if (id === td[0].innerText) {

alert("Эта птица уже участвует в выставке");

return;

}

}

var bird = await GetBird(id);

document.querySelectorAll("tbody")[2].append(AnimalRow(bird));

}

//Работа с выставкой

function ExhibitionsRow(exhibition) {

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

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

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

idId.append(exhibition.id);

tr.append(idId);

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

titleId.append(exhibition.title);

tr.append(titleId);

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

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

removeLink.append("Удалить");

removeLink.className = "btn btn-danger";

removeLink.addEventListener("click", async () => await DeleteExhibition(exhibition.id));

linksTd.append(removeLink);

tr.appendChild(linksTd);

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

return tr;

}

async function DeleteExhibition(id) {

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

method: "DELETE",

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

});

if (response.ok === true) {

const exhibition = await response.json();

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

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function CreateExhibition(title, cats, dogs, birds) {

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

method: "POST",

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

body: JSON.stringify({

Title: title,

cats: cats,

dogs: dogs,

birds: birds

})

});

if (response.ok == true) {

const exhibition = await response.json();

document.querySelectorAll("tbody")[3].append(ExhibitionsRow(exhibition));

}

else {

const error = await response.json();

console.log(error.message);

}

}

async function EditExhibition(id, title, cats, dogs, birds) {

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

method: "PUT",

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

body: JSON.stringify({

id: id,

Title: title,

cats: cats,

dogs: dogs,

birds: birds

})

});

if (response.ok == true) {

const exhibition = await response.json();

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

}

else {

const error = await response.json();

console.log(error.message);

}

}

function Reset() {

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

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

$("#birdBody").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("exhibitionId");

idField.value = "";

}

async function AddExhibition() {

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

let cats = [];

let dogs = [];

let birds = [];

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

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

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

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

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

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

}

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

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

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

}

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

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

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

}

if (title.length === 0) {

alert("Вы не ввели название");

return;

}

if (cats.length === 0 && dogs.length === 0 && bird.length === 0) {

alert("Вы не выбрали животных");

return;

}

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

if (id === "") {

await CreateExhibition(title, cats, dogs, birds);

}

else {

await EditExhibition(id, title, cats, dogs, birds);

}

Reset();

}

async function click(exhibition, tr) {

Reset();

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

idField.value = exhibition.id;

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

var row = table.rows;

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

row[i].bgColor = "white";

}

var id = exhibition.id;

tr.bgColor = "green";

await GetExhibition(id);

}

//Поиск

async function Search() {

var cat = document.getElementById("catCount").value;

var dog = document.getElementById("dogCount").value;

var bird = document.getElementById("birdCount").value;

if (cat.length === 0) {

cat = 0;

}

else {

cat = parseInt(cat, 10);

}

if (dog.length === 0) {

dog = 0;

}

else {

dog = parseInt(dog, 10);

}

if (bird.length === 0) {

bird = 0;

}

else {

bird = parseInt(bird, 10);

}

const response = await fetch(`/api/Exhibitions/Search/${cat}-${dog}-${bird}`, {

method: "GET",

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

});

if (response.ok) {

var exhibitions = await response.json();

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

Reset();

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

exhibitions.forEach(e => rows.append(ExhibitionsRow(e)));

await GetExhibition(exhibitions[0].id);

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

var row = table.rows;

row[1].bgColor = "green";

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

idField.value = exhibitions[0].id;

}

}

async function SearchClear() {

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

Reset();

GetExhibitions();

}

setCatCmb();

setDogCmb();

setBirdCmb();

GetExhibitions();

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 MapCat(WebApplication app, IMongoDatabase db, string collectionName)

{

//Добавим конечную точку, которая обрабатывает GET по пути

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

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

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

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

{

var cat = await db.GetCollection<Cat>(collectionName)

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

.FirstOrDefaultAsync();

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

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

// если пользователь найден, отправляем его

return Results.Json(cat);

});

//Если получаем Delete по этому адресу, вызывается это

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

{

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

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

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

return Results.Json(cat);

});

//При получении запроса с методом POST, идем сюда

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

// добавляем пользователя в список

await db.GetCollection<Cat>(collectionName).InsertOneAsync(cat);

return cat;

});

//Если приходит put по этому адресу, идем сюда

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

var cat = await db.GetCollection<Cat>(collectionName)

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

if (cat == null)

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

return Results.Json(cat);

});

}

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 | Кошки | Добавление | Кот, 5, Черный, Чаузи | Обновление таблицы |
| 2 | Редактирование | Новый Кот, 6, Серый, Раас | Обновление таблицы |
| 3 | Удаление | Удаление записи №4 | Обновление таблицы |
| 4 | Собаки | Добавление | Собака, 4, Белый, Айди | Обновление таблицы |
| 5 | Редактирование | Новая Собака, 5, Серый, Тази | Обновление таблицы |
| 6 | Удаление | Удаление записи №4 | Обновление таблицы |
| 7 | Птицы | Добавление | Птица, 7, Рыжий, Каик | Обновление таблицы |
| 8 | Редактирование | Новая Птица, 9, Серый, Уиллет | Обновление таблицы |
| 9 | Удаление | Удаление записи №4 | Обновление таблицы |
| 10 | Выставки | Добавление | 638af85280440f97b2681b28, 637c8763c363add151cee701, 637c89cfcb01a1a3fed9db3e,  Выставка | Обновление таблицы |
| 11 | Редактирование | 638af85280440f97b2681b28,  Новая Выставка | Обновление таблицы |
| 12 | Удаление | Удаление записи №3 | Обновление таблицы |
| 13 | Первый запрос | Выполнение запроса | 6388816ee72cd74701899f92 | Документ |
| 14 | Второй запрос | Выполнение запроса | 3, 1, 1 | Документ |

8. Результаты тестирования

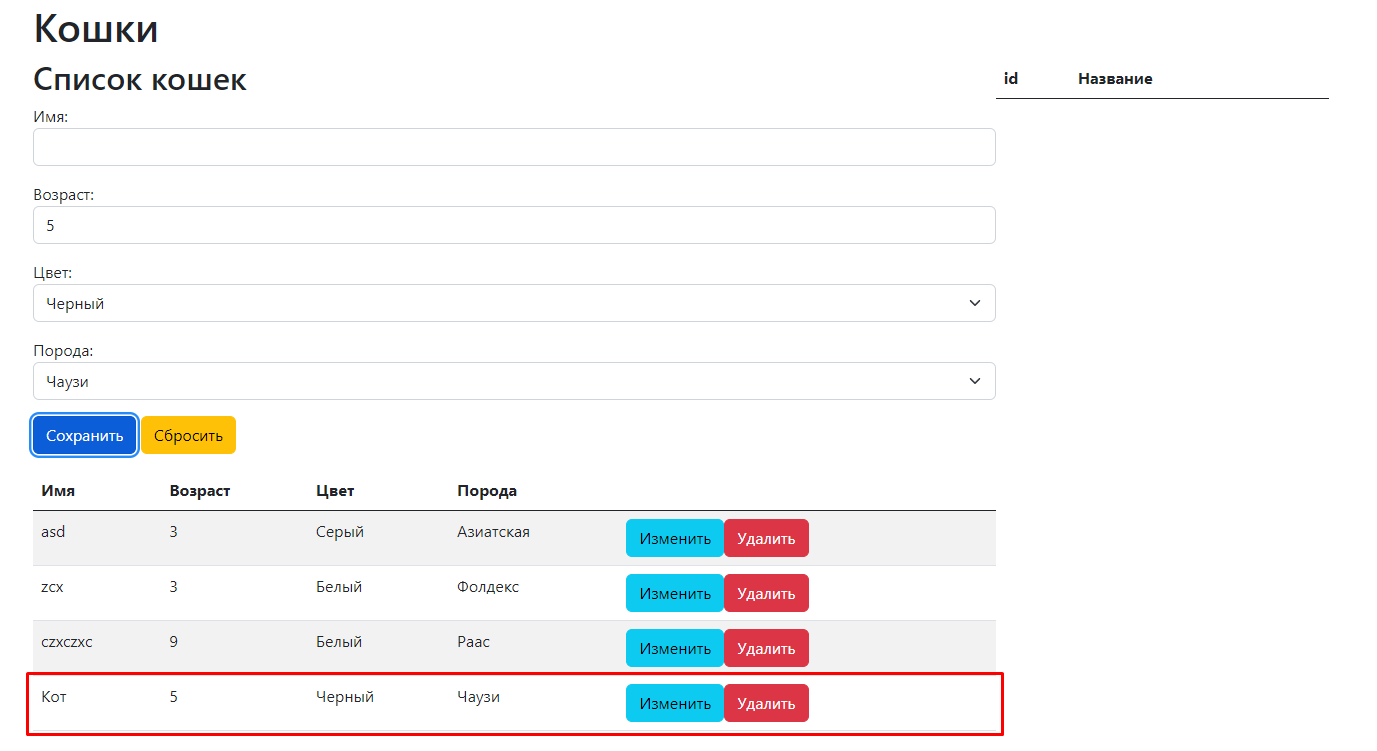


Рисунок 4 – Результат теста №1

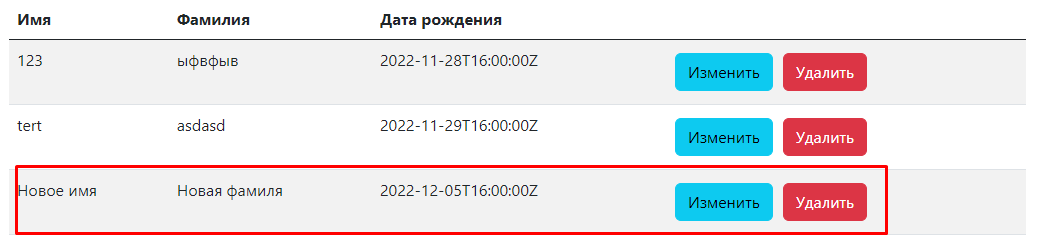


Рисунок 5 – Результат теста №2

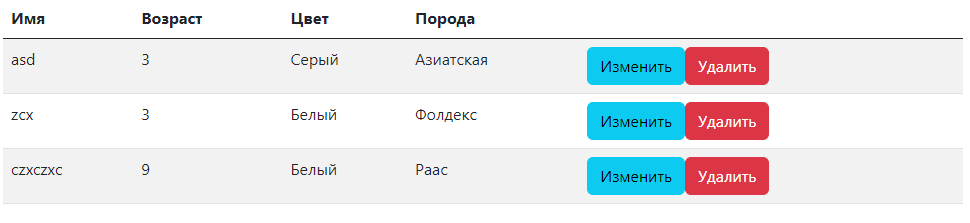


Рисунок 6 – Результат теста №3

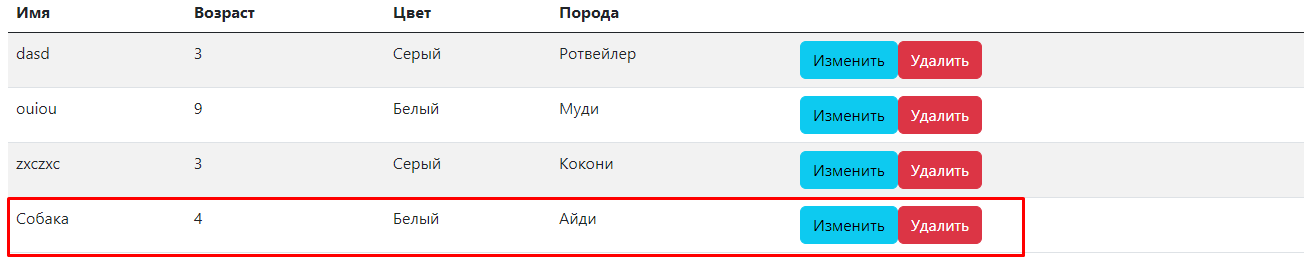


Рисунок 7 – Результат теста №4

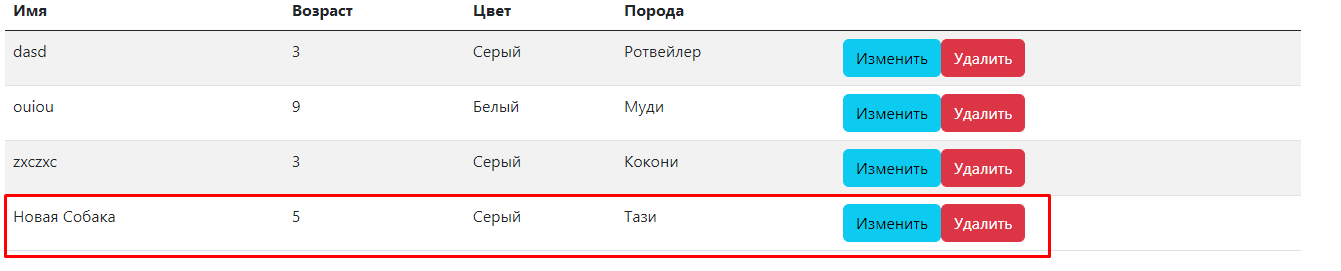


Рисунок 8 – Результат теста №5

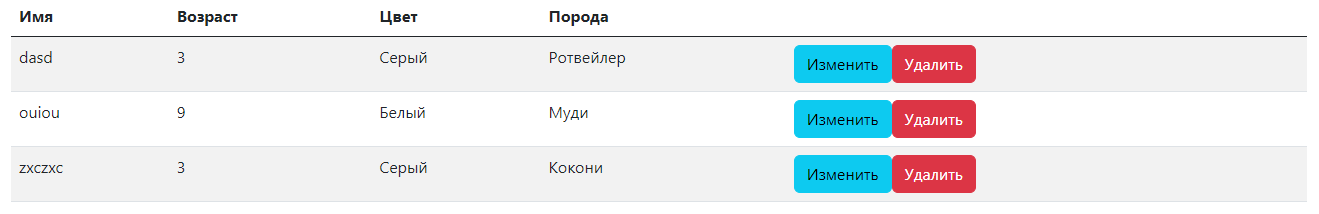


Рисунок 8 – Результат теста №6



Рисунок 9 – Результат теста №7



Рисунок 10 – Результат теста №8



Рисунок 11 – Результат теста №9

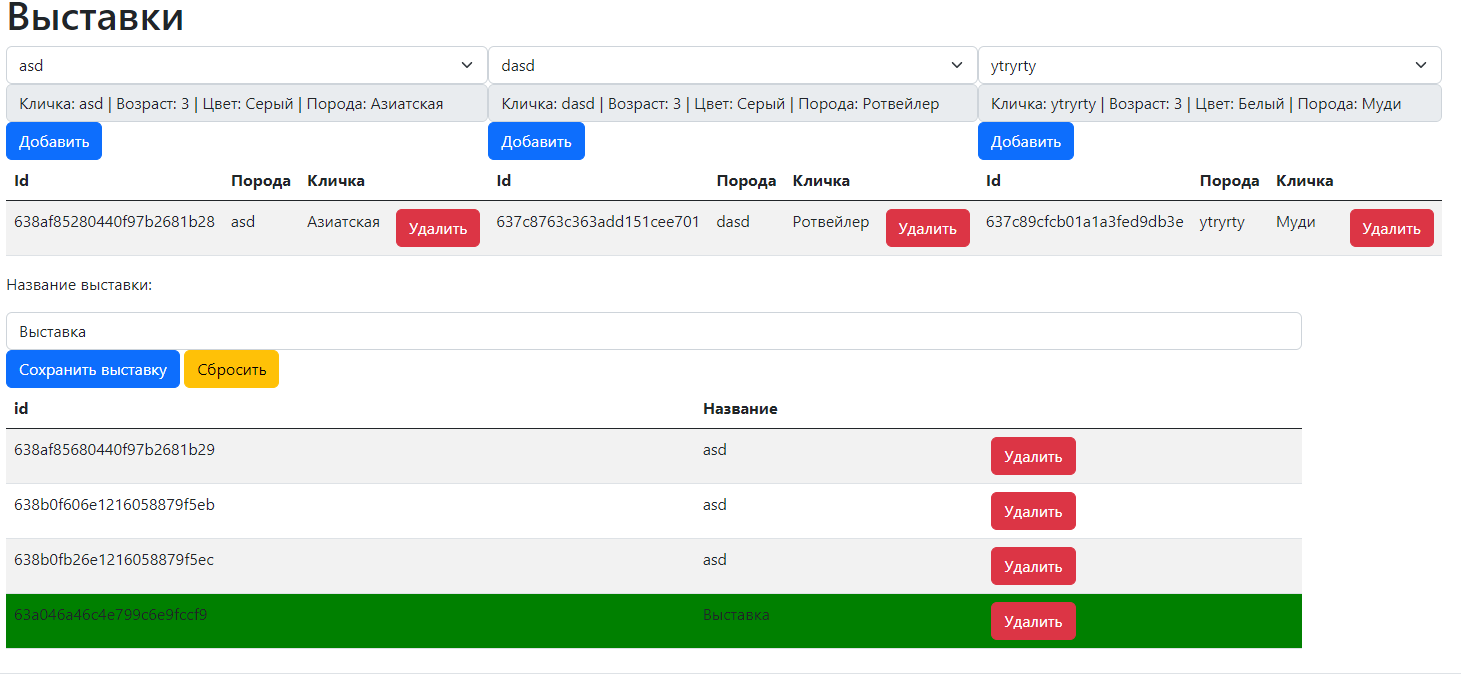


Рисунок 12 – Результат теста №10

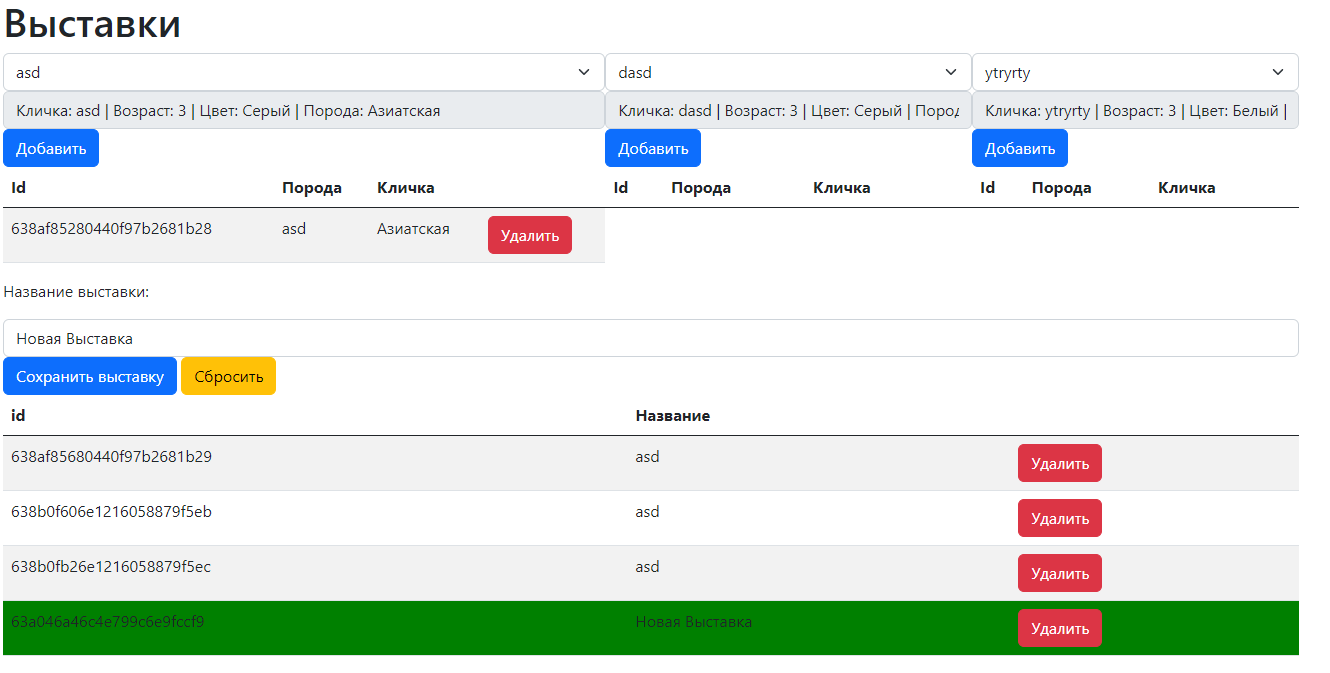


Рисунок 13 – Результат теста №11

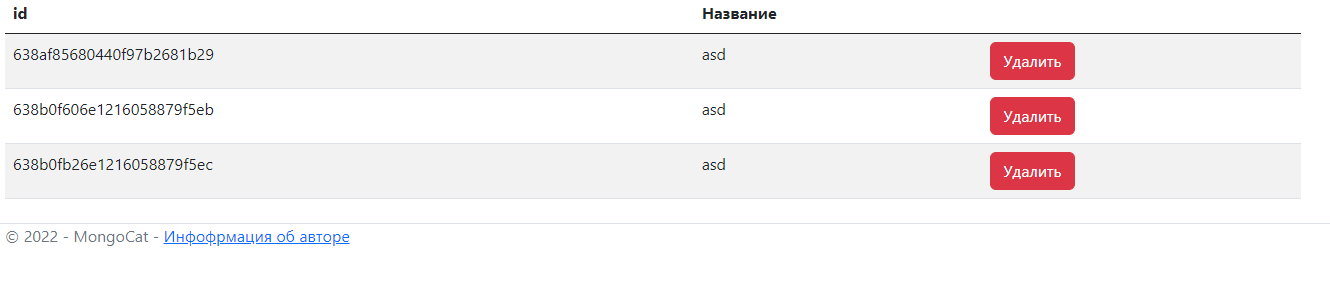


Рисунок 14 – Результат теста №12

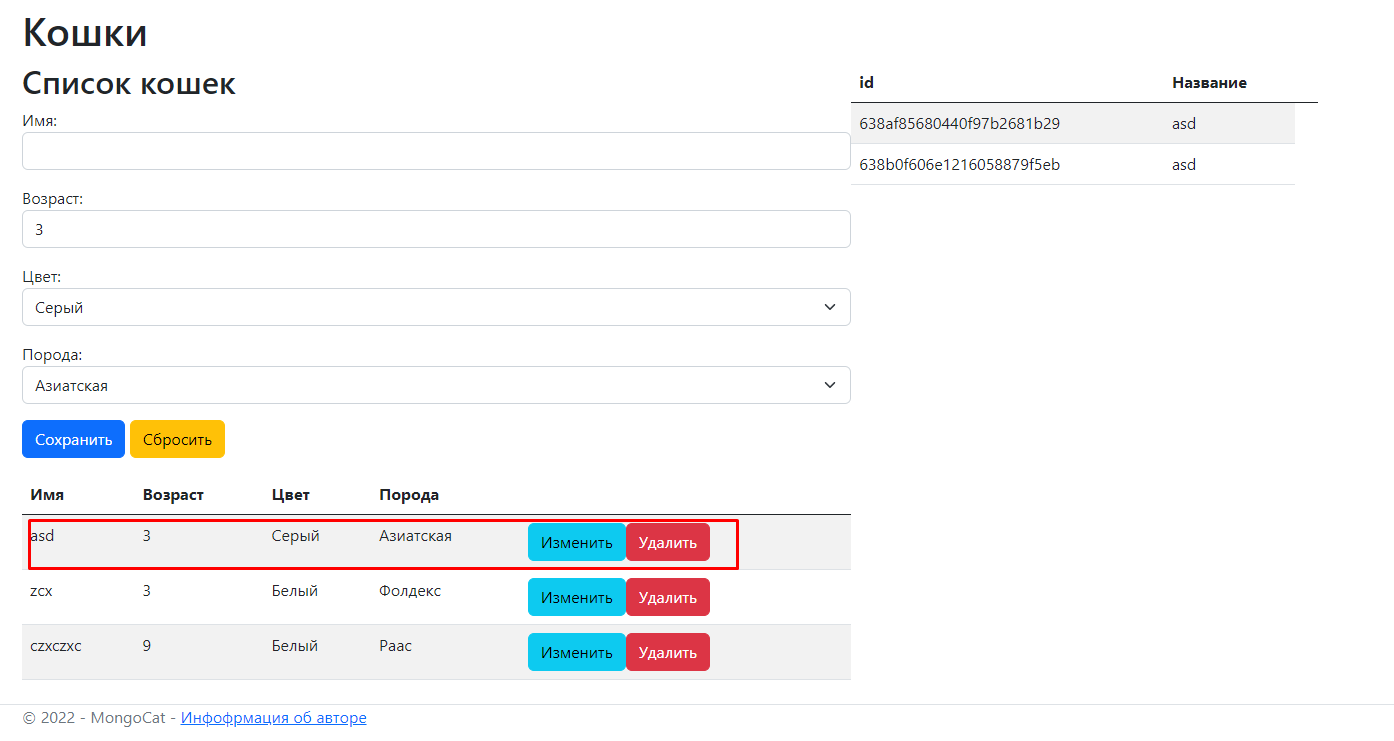


Рисунок 15 – Результат теста №13

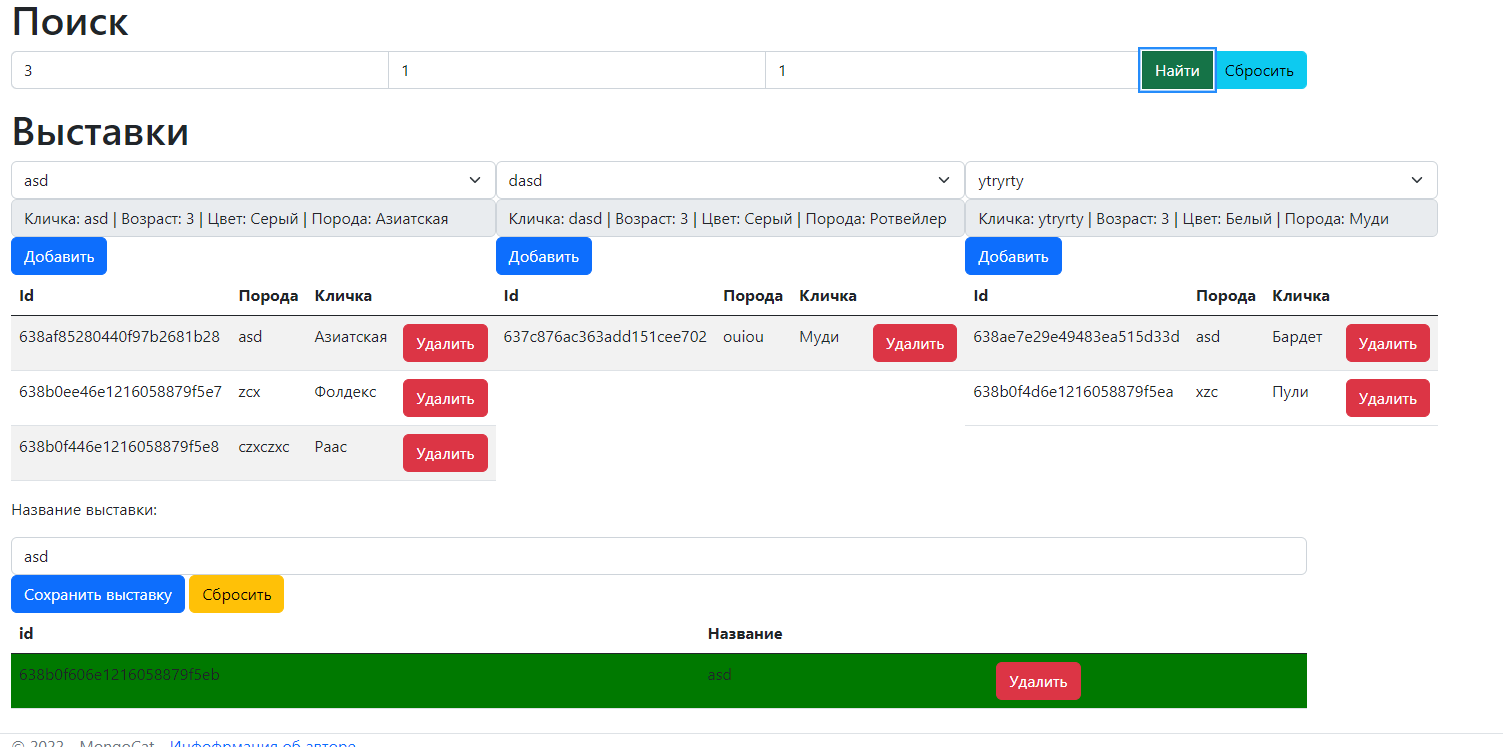


Рисунок 16 – Результат теста №14

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)