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 клиента по технологии MVC |
| наименование темы |

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине

|  |
| --- |
| Управление данными |

|  |
| --- |
| 1.008.00.00 ПЗ |
| обозначение документа |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ИСТб-19-2 |  |  |  | Комогорцева Ю.В. |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | В.А. Харахинов |
|  |  |  |  | подпись |  | И.О. Фамилия |
| Курсовой проект защищен с оценкой | | | |  | | |

Иркутск 2021 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 | Управление данными | | | | | | | | | | | | | | | | | |
| Студенту | Комогорцевой Ю.В. | | | | | | | | | | | | | | | | | |
|  | (фамилия, инициалы) | | | | | | | | | | | | | | | | | |
| Тема проекта | | Разработка клиента по технологии MVC | | | | | | | | | | | | | | | | |
| Исходные данные | | | Вариант XX | | | | | | | | | | | | | | | |
| 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 | | | | « | 07 | » | октября | | | | | 2021 г. | | | | |
|  | | | | | | | | | | | | | | | | | | |
| Задание получил | | | | | | | | |  | | | |  | | Комогорцева Ю.В. | | | |
|  | | | | | | | | | подпись | | | |  | | И.О. Фамилия | | | |
|  | | | | | | | | | |  |  | | |  | |  | |  |
| Дата представления проекта руководителю | | | | | | | | | | « | 14 | | | » | | декабря | | 2021 г. |
|  | | | | | | | | | | | | | | | | | | |
| Руководитель курсового проектирования | | | | | | | | |  | | | |  | | В.А. Харахинов | | | |
|  | | | | | | | | | подпись | | | |  | | И.О. Фамилия | | | |

**Содержание**

[Введение 5](#_Toc90753369)

[Индивидуальный вариант 6](#_Toc90753370)

[1 Структура базы данных 7](#_Toc90753371)

[1.1 Метод «Объект-Связь» 7](#_Toc90753372)

[1.2 DBDesigner 7](#_Toc90753373)

[2 Структурная схема функционирования моделей, контроллеров и представлений 8](#_Toc90753374)

[3 Модель данных 9](#_Toc90753375)

[4 Контроллеры 12](#_Toc90753376)

[4.1 ClientsController 13](#_Toc90753377)

[4.2 WorksController 18](#_Toc90753378)

[4.3 StatisticsServicesController 22](#_Toc90753379)

[5 Представления 25](#_Toc90753380)

[5.1 Клиенты 25](#_Toc90753381)

[5.1.1 Index 25](#_Toc90753382)

[5.1.2 Edit 28](#_Toc90753383)

[5.1.3 Details 30](#_Toc90753384)

[5.1.4 Delete 32](#_Toc90753385)

[5.1.5 Create 35](#_Toc90753386)

[5.2 Работы 37](#_Toc90753387)

[5.2.1 Index 37](#_Toc90753388)

[5.2.2 Edit 40](#_Toc90753389)

[5.2.3 Details 42](#_Toc90753390)

[5.2.4 Delete 45](#_Toc90753391)

[5.2.5 Create 47](#_Toc90753392)

[5.3 Выходной документ со статистикой по услугам 49](#_Toc90753393)

[5.3.1 Index 49](#_Toc90753394)

[5.3.2 Details 50](#_Toc90753395)

[6 Контроль 53](#_Toc90753396)

[6.1 Services 53](#_Toc90753397)

[6.2 Clients 54](#_Toc90753398)

[6.3Employee 55](#_Toc90753399)

[6.4 Record 56](#_Toc90753400)

[6.5 Work 57](#_Toc90753401)

[7 Таблица тестов 58](#_Toc90753402)

[8 Результаты тестирования 65](#_Toc90753403)

[Заключение 79](#_Toc90753404)

[Список использованных источников 80](#_Toc90753405)

Введение

Целью курсового проекта является закрепление теоретических знаний полученных в процессе изучения курса «Управление данными» и развитие практических навыков работы с базами данных в ходе решения конкретной практической задачи – разработки веб-приложения с использованием паттерна MVC (model - view - controller), а также развитие способностей к обоснованному принятию самостоятельных решений в ходе проектной деятельности и получение базового опыта разработки программного обеспечения.

В рамках курсового проекта решается задача создания веб-приложения — это информационная система для автоматизированного взаимодействия с клиентами и с возможностью хранить и редактировать данные об услугах, клиентах, сотрудниках, выполненных работах и записях. Вся информация хранится в локальной базе данных.

# Индивидуальный вариант

Имеются клиенты (фамилия, имя, отчество, день рождения, почта, телефон, комментарий), сотрудники (фамилия, имя, отчество, день рождения, должность, телефон, комментарий), услуги (название, стоимость, комментарий), записи (услуга, клиент, сотрудник, дата, время, комментарий) и выполненные работы (услуга, клиент, сотрудник, дата, время, комментарий, доход). Запись необходима для хранения информации о том, на какую услугу, к какому сотруднику, во сколько, когда и какой клиент придет в салон красоты. Выполненные работы хранят информацию о том, какая услуга, кому, кем, когда, во сколько и за какую стоимость была выполнена.

Выходные документы:

* + выдать список услуг с количеством выполненных работ за определенный год;
  + выдать список сотрудников, содержащих в имени определенную букву, с указанием максимального и минимального возраста клиентов этих сотрудников, а также с количеством клиентов сотрудников.

1 Структура базы данных

## Метод «Объект-Связь»



Рисунок 1 – ER-диаграмма

Исходя из заданного индивидуального варианта была разработана ER-диаграмма. Объекты «Работа» и «Запись» обязательно содержат такие объекты как «Услуга», «Сотрудник» и «Клиент».

1.2 DBDesigner

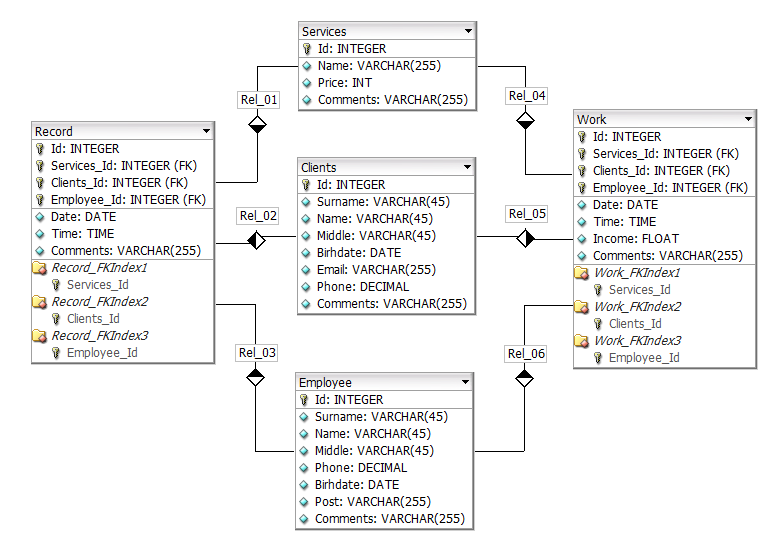


Рисунок 2 – Логическая модель БД «BeautyShop»

# Структурная схема функционирования моделей, контроллеров и представлений

Рисунок 3 – Схема функционирования модели, контроллеров и представлений

# Модель данных

Все сущности в приложении принято выделять в отдельные модели. Данные моделей хранятся в базе данных. Для взаимодействия с базой данных, использовался фреймворк Entity Framework.

Ниже описан процесс добавления модели с помощью визуального конструктора:

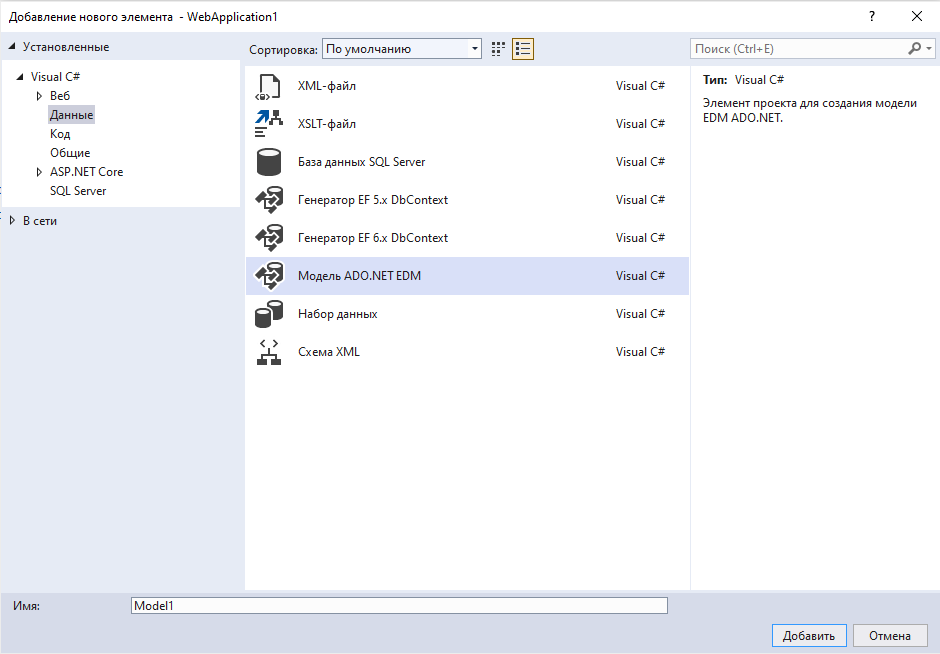


Рисунок 4 – Выбор элемента модели

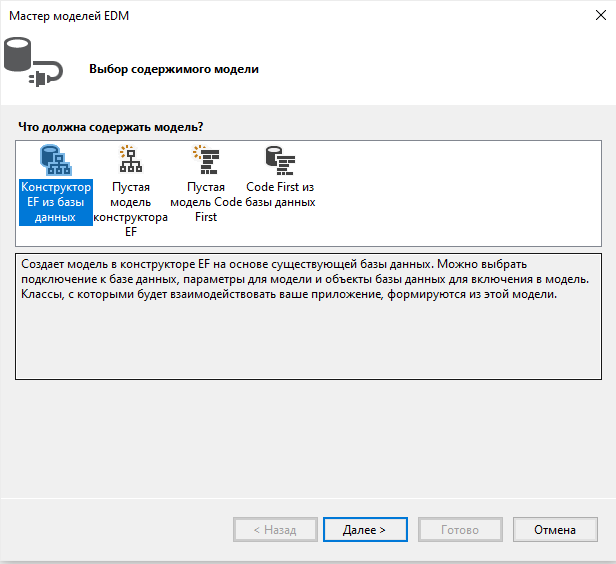


Рисунок 5 – Выбор содержимого модели (из существующей БД)

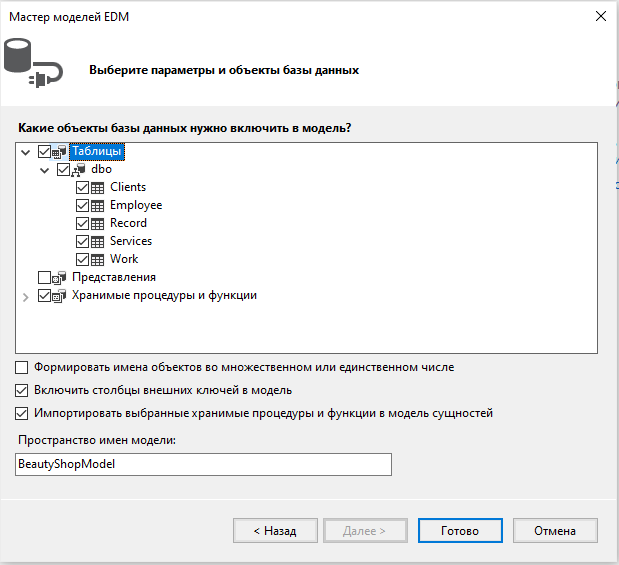


Рисунок 6 – Выбор необходимых объектов

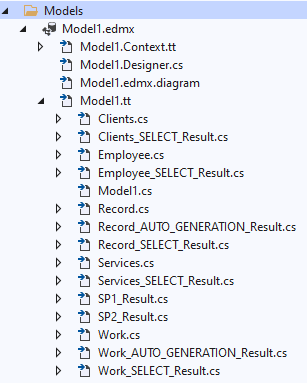


Рисунок 7 – Модель в окне «Обозреватель решений»

# Контроллеры

Контроллер является центральным компонентом в архитектуре MVC. Контроллер получает ввод пользователя, обрабатывает его и посылает обратно результат обработки, например, в виде представления.

Рассмотрим некоторые контроллеры, а начнем с создания контроллера «ClientsController» конструктором.

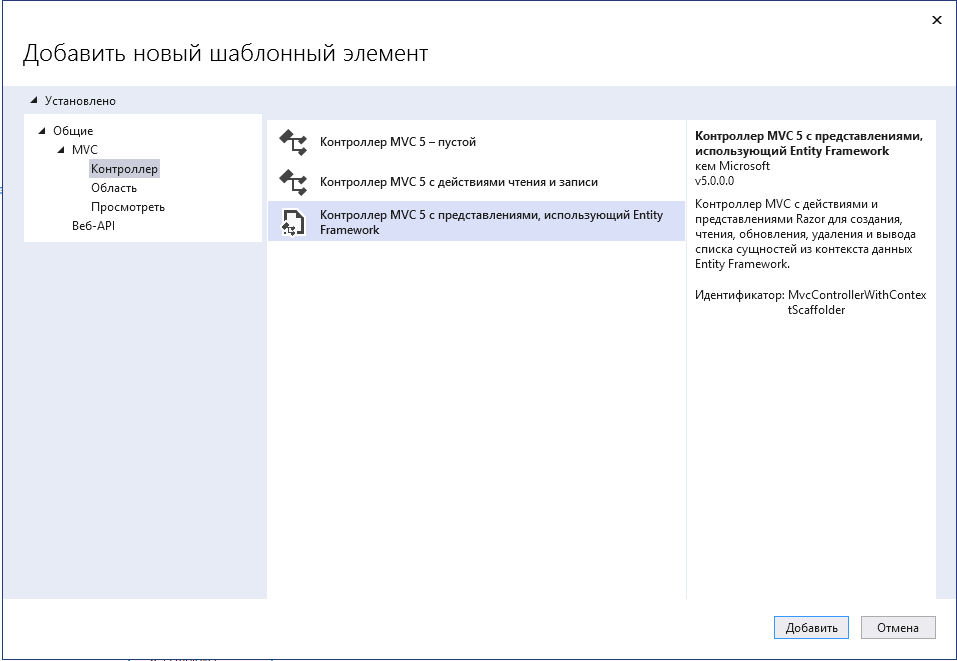


Рисунок 8 – Создание контроллера

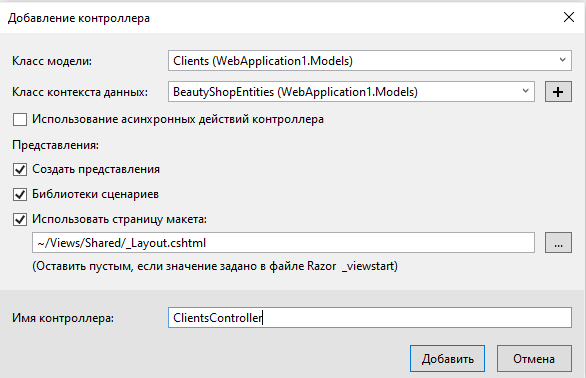


Рисунок 9 – Добавление контроллера

Методы действий (action methods) представляют такие методы контроллера, которые обрабатывают запросы по определенному URL. В данном случае при автоматическом создании контроллера, создаются и методы действий:

* Index – метод для вывода данных всей таблицы
* Details – метод для вывода данных одной записи из таблицы
* Create – метод для создания новой записи
* Edit – метод для редактирования данных одной записи из таблицы
* Delete – метод для удаления данных одной записи из таблицы

После автоматического создания методы контроллеров были отредактированы. Далее приводятся 3 примера контроллеров: ClientsController(таблица без FK), WorksController(таблица с FK), StatisticsServicesController(выходной документ).

## 4.1 ClientsController

using PagedList;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using Web1.Models;

namespace Web1.Controllers

{

public class ClientsController : Controller

{

private BeautyShopEntities1 db = new BeautyShopEntities1();

// GET: Clients

public ActionResult Index(string sortOrder, string currentFilter, string searchString, int? page)

{

ViewBag.CurrentSort = sortOrder;

ViewBag.SurameSortParm = String.IsNullOrEmpty(sortOrder) ? "Surname\_desc" : "";

ViewBag.NameSortParm = sortOrder == "name" ? "name\_desc" : "name";

ViewBag.MiddleSortParm = sortOrder == "middle" ? "middle\_desc" : "middle";

ViewBag.PhoneSortParm = sortOrder == "phone" ? "phone\_desc" : "phone";

ViewBag.BirhdateSortParm = sortOrder == "birhdate" ? "birhdate\_desc" : "birhdate";

ViewBag.PostSortParm = sortOrder == "email" ? "email\_desc" : "email";

ViewBag.CommentSortParm = sortOrder == "comment" ? "comment\_desc" : "comment";

if (searchString != null)

{

page = 1;

}

else

{

searchString = currentFilter;

}

ViewBag.CurrentFilter = searchString;

var clients = from s in db.Clients

select s;

if (!String.IsNullOrEmpty(searchString))

{

clients = clients.Where(s => s.Name.Contains(searchString)

|| s.Surname.Contains(searchString)

|| s.Middle.Contains(searchString));

}

switch (sortOrder)

{

case "Surname\_desc":

clients = clients.OrderByDescending(s => s.Surname);

break;

case "name":

clients = clients.OrderBy(s => s.Name);

break;

case "name\_desc":

clients = clients.OrderByDescending(s => s.Name);

break;

case "middle":

clients = clients.OrderBy(s => s.Middle);

break;

case "middle\_desc":

clients = clients.OrderByDescending(s => s.Middle);

break;

case "phone":

clients = clients.OrderBy(s => s.Phone);

break;

case "phone\_desc":

clients = clients.OrderByDescending(s => s.Phone);

break;

case "email":

clients = clients.OrderBy(s => s.Email);

break;

case "email\_desc":

clients = clients.OrderByDescending(s => s.Email);

break;

case "comment":

clients = clients.OrderBy(s => s.Comment);

break;

case "comment\_desc":

clients = clients.OrderByDescending(s => s.Comment);

break;

case "birhdate":

clients = clients.OrderBy(s => s.Birhdate);

break;

case "birhdate\_desc":

clients = clients.OrderByDescending(s => s.Birhdate);

break;

default: // Name ascending

clients = clients.OrderBy(s => s.Surname);

break;

}

int pageSize =10;

int pageNumber = (page ?? 1);

return View(clients.ToPagedList(pageNumber, pageSize));

}

// GET: Clients/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Clients clients = db.Clients.Find(id);

if (clients == null)

{

return HttpNotFound();

}

return View(clients);

}

// GET: Clients/Create

public ActionResult Create()

{

return View();

}

// POST: Clients/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку.

// Дополнительные сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "Id,Surname,Name,Middle,Birhdate,Email,Phone,Comment")] Clients clients)

{

if (ModelState.IsValid)

{

try

{

var e = db.Clients.Where(s => s.Phone.Equals(((decimal)clients.Phone)));

if (e.Any())

{

ModelState.AddModelError("", "Клиент с таким номером телефона уже существует.");

}

else

{

db.Clients.Add(clients);

db.SaveChanges();

return RedirectToAction("Index");

}

}

catch

{

ModelState.AddModelError("", "Клиент уже существует.");

}

}

return View(clients);

}

// GET: Clients/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Clients clients = db.Clients.Find(id);

if (clients == null)

{

return HttpNotFound();

}

return View(clients);

}

// POST: Clients/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку.

// Дополнительные сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "Id,Surname,Name,Middle,Birhdate,Email,Phone,Comment")] Clients clients)

{

if (ModelState.IsValid)

{

try

{

db.Entry(clients).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

catch

{

ModelState.AddModelError("", "Клиент уже существует.");

}

}

return View(clients);

}

// GET: Clients/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Clients clients = db.Clients.Find(id);

if (clients == null)

{

return HttpNotFound();

}

return View(clients);

}

// POST: Clients/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Clients clients = db.Clients.Find(id);

db.Clients.Remove(clients);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

## 4.2 WorksController

using PagedList;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using Web1.Models;

namespace Web1.Views

{

public class WorksController : Controller

{

private BeautyShopEntities1 db = new BeautyShopEntities1();

// GET: Works

public ActionResult Index(string sortOrder, string currentFilter, int? page)

{

ViewBag.CurrentSort = sortOrder;

ViewBag.DateSortParm = String.IsNullOrEmpty(sortOrder) ? "Date\_desc" : "";

ViewBag.TimeSortParm = sortOrder == "time" ? "time\_desc" : "time";

ViewBag.ClientsSortParm = sortOrder == "clients" ? "clients\_desc" : "clients";

ViewBag.IncomeSortParm = sortOrder == "income" ? "income\_desc" : "income";

ViewBag.EmployeeSortParm = sortOrder == "employee" ? "employee\_desc" : "employee";

ViewBag.ServicesSortParm = sortOrder == "services" ? "services\_desc" : "services";

var work = from s in db.Work

select s;

switch (sortOrder)

{

case "Date\_desc":

work = work.OrderByDescending(s => s.Date);

break;

case "time":

work = work.OrderBy(s => s.Time);

break;

case "time\_desc":

work = work.OrderByDescending(s => s.Time);

break;

case "income":

work = work.OrderBy(s => s.Income);

break;

case "income\_desc":

work = work.OrderByDescending(s => s.Income);

break;

case "clients":

work = work.OrderBy(s => s.Id\_Clients);

break;

case "clients\_desc":

work = work.OrderByDescending(s => s.Id\_Clients);

break;

case "employee":

work = work.OrderBy(s => s.Id\_Employee);

break;

case "employee\_desc":

work = work.OrderByDescending(s => s.Id\_Employee);

break;

case "services":

work = work.OrderBy(s => s.Id\_Services);

break;

case "services\_desc":

work = work.OrderByDescending(s => s.Id\_Services);

break;

default: // Name ascending

work = work.OrderBy(s => s.Date);

break;

}

int pageSize = 10;

int pageNumber = (page ?? 1);

return View(work.ToPagedList(pageNumber, pageSize));

}

// GET: Works/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Work work = db.Work.Find(id);

if (work == null)

{

return HttpNotFound();

}

return View(work);

}

// GET: Works/Create

public ActionResult Create()

{

ViewBag.Id\_Employee = new SelectList(db.Employee, "Id", "FIO");

ViewBag.Id\_Services = new SelectList(db.Services, "Id", "Concat");

return View();

}

// POST: Works/Create

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку.

// Дополнительные сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "Id,Id\_Services,Id\_Employee,Date,Time,Income,Comment")] Work work)

{

var clients = from s in db.Clients select s;

var Name = Request["Clients.Name"].ToString();

var Surname = Request["Clients.Surname"].ToString();

var Middle = Request["Clients.Middle"].ToString();

var c = clients.Where(s => s.Name.Contains(Name)

& s.Surname.Contains(Surname)

& s.Middle.Contains(Middle)).FirstOrDefault();

try

{

if (c != null)

{

if (ModelState.IsValid)

{

work.Id\_Clients = c.Id;

db.Work.Add(work);

db.SaveChanges();

return RedirectToAction("Index");

}

}

else {

ModelState.AddModelError("", "Клиента не существует");

}

}

catch

{

ModelState.AddModelError("", "Работа уже существует.");

}

ViewBag.Id\_Employee = new SelectList(db.Employee, "Id", "FIO", work.Id\_Employee);

ViewBag.Id\_Services = new SelectList(db.Services, "Id", "Concat", work.Id\_Services);

return View(work);

}

// GET: Works/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Work work = db.Work.Find(id);

if (work == null)

{

return HttpNotFound();

}

ViewBag.Id\_Employee = new SelectList(db.Employee, "Id", "FIO", work.Id\_Employee);

ViewBag.Id\_Services = new SelectList(db.Services, "Id", "Concat", work.Id\_Services);

return View(work);

}

// POST: Works/Edit/5

// Чтобы защититься от атак чрезмерной передачи данных, включите определенные свойства, для которых следует установить привязку.

// Дополнительные сведения см. в статье https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "Id,Id\_Services,Id\_Employee,Id\_Clients,Date,Time,Income,Comment")] Work work)

{

var clients = from s in db.Clients select s;

var Name = Request["Clients.Name"].ToString();

var Surname = Request["Clients.Surname"].ToString();

var Middle = Request["Clients.Middle"].ToString();

var c = clients.Where(s => s.Name.Contains(Name)

& s.Surname.Contains(Surname)

& s.Middle.Contains(Middle)).FirstOrDefault();

try

{

if (c != null)

{

if (ModelState.IsValid)

{

work.Id\_Clients = c.Id;

db.Entry(work).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

}

else

{

ModelState.AddModelError("", "Клиента не существует");

}

}

catch

{

ModelState.AddModelError("", "Работа уже существует.");

}

ViewBag.Id\_Employee = new SelectList(db.Employee, "Id", "FIO", work.Id\_Employee);

ViewBag.Id\_Services = new SelectList(db.Services, "Id", "Concat", work.Id\_Services);

return View(work);

}

// GET: Works/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Work work = db.Work.Find(id);

if (work == null)

{

return HttpNotFound();

}

return View(work);

}

// POST: Works/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Work work = db.Work.Find(id);

db.Work.Remove(work);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

## 4.3 StatisticsServicesController

Особенностью выходного документа является отсутствие возможности редактировать, удалять и создавать новые записи. Это связано с тем, что данные для этого контроллера — это статистика, собираемая из других таблиц.

using PagedList;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Web1.Models;

namespace Web1.Controllers

{

public class StatisticsServicesController : Controller

{

private BeautyShopEntities1 db = new BeautyShopEntities1();

public ViewResult Index()

{

return View();

}

// GET: Statistics

public ActionResult Details(string currentFilter, string searchString, int? page)

{

if (searchString != null)

{

page = 1;

}

else

{

searchString = currentFilter;

}

String d = null;

try {

Convert.ToInt32(searchString);

d = searchString;

}

catch {

ModelState.AddModelError("", "Вы ввели не число");

TempData["msg"] = "1";

return View();

}

ViewBag.CurrentFilter = searchString;

if ((String.IsNullOrEmpty(searchString) && page == 1 )|| (Convert.ToInt32(searchString) > 2050) || (Convert.ToInt32(searchString) <1900))

{

TempData["msg"] = "1";

return View();

}

else

{

TempData["msg"] = "";

}

object[] parameters = {

new SqlParameter("@date1",SqlDbType.VarChar) {Value=d}

};

db.Database.CommandTimeout = 360;

IEnumerable<SP1\_Result> query = db.Database.SqlQuery<SP1\_Result>("SP1 @date1", parameters).ToList();

int pageSize = 10;

int pageNumber = (page ?? 1);

return View(query.ToPagedList(pageNumber, pageSize));

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

# 5 Представления

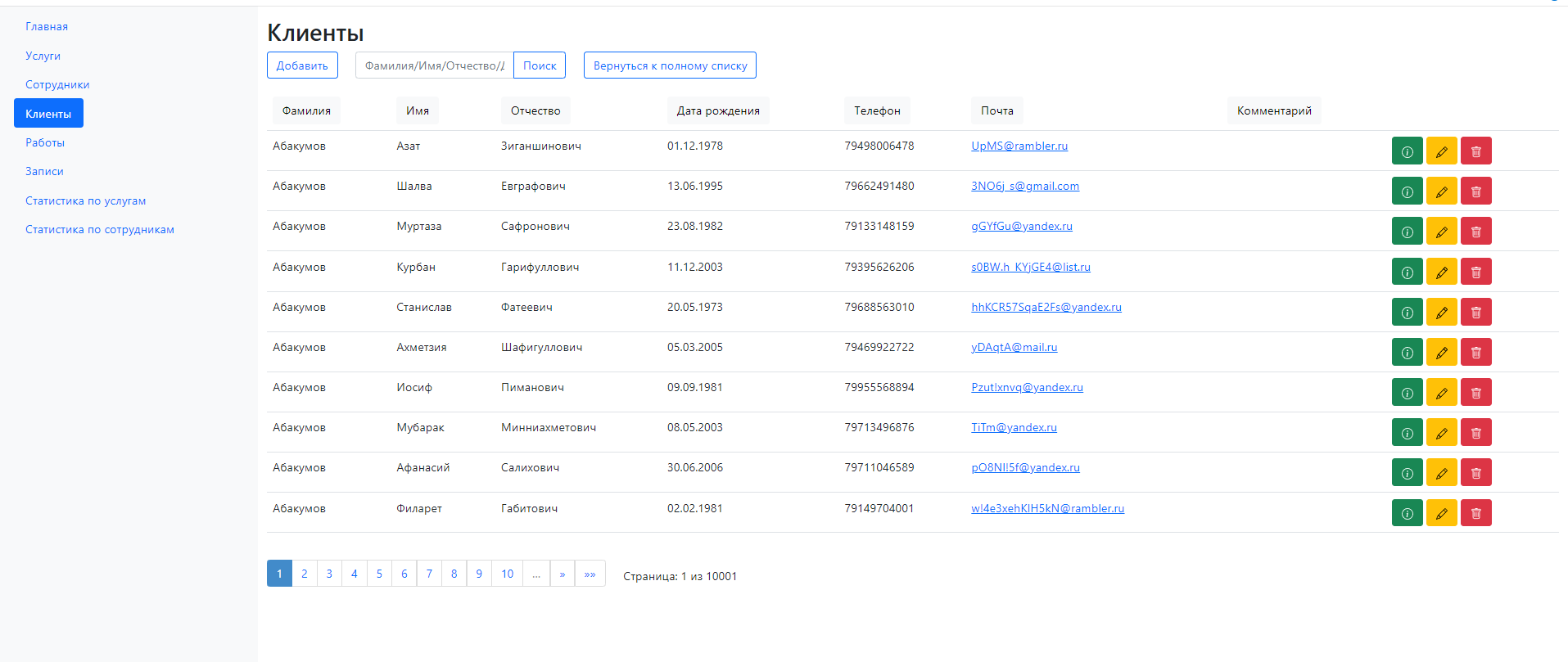
Хотя работа приложения MVC управляется главным образом контроллерами, но непосредственно пользователю приложение доступно в виде представления, которое и формирует внешний вид приложения.

При создании контроллера, автоматически создаются представления Index, Edit, Details, Delete, Create. В ходе разработки проекта представления были изменены, а именно дизайн, верстка и формы для ввода.

Ниже приведены рисунки и листинг кода представлений для каждого из контроллеров раздела 4.

## 5.1 Клиенты

### 5.1.1 Index

Рисунок 10 – Index

@model PagedList.IPagedList<Web1.Models.Clients>

@using PagedList.Mvc;

<link href=”~/Content/PagedList.css” rel=”stylesheet” type=”text/css” />

@{

ViewBag.Title = “Index”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Клиенты</h2>

<div class=”row”>

<div class=”col-auto”>

<a class=”btn btn-outline-primary” href=”@Url.Action(“Create”, “Clients”)”>

Добавить

</a>

</div>

<div class=”col-auto”>

<form asp-action=”Index” method=”get”>

<div class=”form-actions no-color row”>

<div class=”input-group mb-3”>

<input type=”text” id=”SearchString” name=”SearchString” class=”form-control” placeholder=”Фамилия/Имя/Отчество/Должность” aria-label=”Введите ФИО”>

<input type=”submit” value=”Поиск” class=”btn btn-outline-primary” />

</div>

</div>

</form>

</div>

<div class=”col-auto”>

@Html.ActionLink(“Вернуться к полному списку”, “Index”, new { SizePage = @Model.PageSize }, new { @class = “btn btn-outline-primary” })

</div>

</div>

<table class=”table”>

<tr>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.SurameSortParm, currentFilter = ViewBag.CurrentFilter })”>Фамилия</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.NameSortParm, currentFilter = ViewBag.CurrentFilter })”>Имя</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.MiddleSortParm, currentFilter = ViewBag.CurrentFilter })”>Отчество</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.BirhdateSortParm, currentFilter = ViewBag.CurrentFilter })”>Дата рождения</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.PhoneSortParm, currentFilter = ViewBag.CurrentFilter })”>Телефон</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”, new { sortOrder = ViewBag.EmailSortParm, currentFilter = ViewBag.CurrentFilter })”>Почта</a>

</th>

<th>

<a class=”btn btn-light” href=”@Url.Action(“Index”, “Clients”,new { sortOrder = ViewBag.CommentSortParm, currentFilter = ViewBag.CurrentFilter })”>Комментарий</a>

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.DisplayFor(modelItem => item.Surname)

</td>

<td>

@Html.DisplayFor(modelItem => item.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Middle)

</td>

<td>

@Html.DisplayFor(modelItem => item.Birhdate)

</td>

<td>

@Html.DisplayFor(modelItem => item.Phone)

</td>

<td>

@Html.DisplayFor(modelItem => item.Email)

</td>

<td>

@Html.DisplayFor(modelItem => item.Comment)

</td>

<td>

<a class=”btn btn-success” href=”@Url.Action(“Details”, “Clients”, new { id = item.Id })”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-info-circle” viewBox=”0 0 16 16”>

<path d=”M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14zm0 1A8 8 0 1 0 8 0a8 8 0 0 0 0 16z” />

<path d=”m8.93 6.588-2.29.287-.082.38.45.083c.294.07.352.176.288.469l-.738 3.468c-.194.897.105 1.319.808 1.319.545 0 1.178-.252 1.465-.598l.088-.416c-.2.176-.492.246-.686.246-.275 0-.375-.193-.304-.533L8.93 6.588zM9 4.5a1 1 0 1 1-2 0 1 1 0 0 1 2 0z” />

</svg>

</a>

<a class=”btn btn-warning” href=”@Url.Action(“Edit”, “Clients”, new { id = item.Id })”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-pencil” viewBox=”0 0 16 16”>

<path d=”M12.146.146a.5.5 0 0 1 .708 0l3 3a.5.5 0 0 1 0 .708l-10 10a.5.5 0 0 1-.168.11l-5 2a.5.5 0 0 1-.65-.65l2-5a.5.5 0 0 1 .11-.168l10-10zM11.207 2.5 13.5 4.793 14.793 3.5 12.5 1.207 11.207 2.5zm1.586 3L10.5 3.207 4 9.707V10h.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.5h.293l6.5-6.5zm-9.761 5.175-.106.106-1.528 3.821 3.821-1.528.106-.106A.5.5 0 0 1 5 12.5V12h-.5a.5.5 0 0 1-.5-.5V11h-.5a.5.5 0 0 1-.468-.325z” />

</svg>

</a>

<a class=”btn btn-danger” href=”@Url.Action(“Delete”, “Clients”, new { id = item.Id })”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-trash” viewBox=”0 0 16 16”>

<path d=”M5.5 5.5A.5.5 0 0 1 6 6v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm2.5 0a.5.5 0 0 1 .5.5v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm3 .5a.5.5 0 0 0-1 0v6a.5.5 0 0 0 1 0V6z” />

<path fill-rule=”evenodd” d=”M14.5 3a1 1 0 0 1-1 1H13v9a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2V4h-.5a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1H6a1 1 0 0 1 1-1h2a1 1 0 0 1 1 1h3.5a1 1 0 0 1 1 1v1zM4.118 4 4 4.059V13a1 1 0 0 0 1 1h6a1 1 0 0 0 1-1V4.059L11.882 4H4.118zM2.5 3V2h11v1h-11z” />

</svg>

</a>

</td>

</tr>

}

</table>

<div class=”row”>

<div class=”col-auto”>

<form asp-action=”Index” method=”get”>

@Html.PagedListPager(Model, page => Url.Action(“Index”, new { page, SizePage = @Model.PageSize, sortOrder = ViewBag.CurrentSort, currentFilter = ViewBag.CurrentFilter }))

</form>

</div>

<div class=”col-auto align-self-center”>

<span class=”align-middle”>Страница: @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) из @Model.PageCount</span>

</div>

</div>

### 5.1.2 Edit

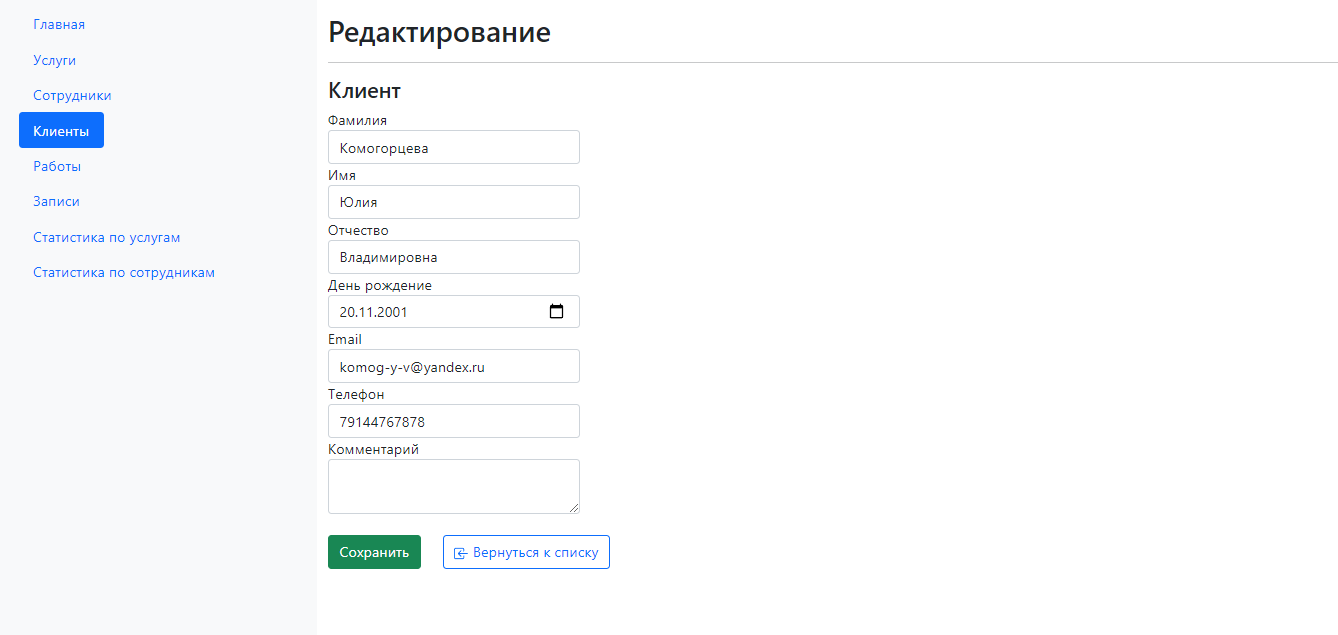
****

Рисунок 11 – Изменение

@model Web1.Models.Clients

@{

ViewBag.Title = “Edit”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class=”form-horizontal”>

<hr />

<h4>Клиент</h4>

@Html.ValidationSummary(true, “”, new { @class = “text-danger” })

@Html.HiddenFor(model => model.Id)

<div class=”form-group”>

@Html.LabelFor(model => model.Surname, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Surname, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Surname, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Name, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Middle, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Middle, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Middle, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Birhdate, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Birhdate, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Birhdate, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Email, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Phone, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Phone, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Comment, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Comment, “”, new { @class = “text-danger” })

</div>

</div>

</div>

<br />

<div class=”row”>

<div class=”col-auto md-6”>

<div class=”col-md-offset-2 col-md-10”>

<input type=”submit” value=”Сохранить” class=”btn btn-success” />

</div>

</div>

<div class=”col-auto md-6”>

<a class=”btn btn-outline-primary” href=”@Url.Action(“Index”, “Employees”,null)”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-box-arrow-in-left” viewBox=”0 0 16 16”>

<path fill-rule=”evenodd” d=”M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z” />

<path fill-rule=”evenodd” d=”M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z” />

</svg>

Вернуться к списку

</a>

</div>

</div>

}

### 5.1.3 Details

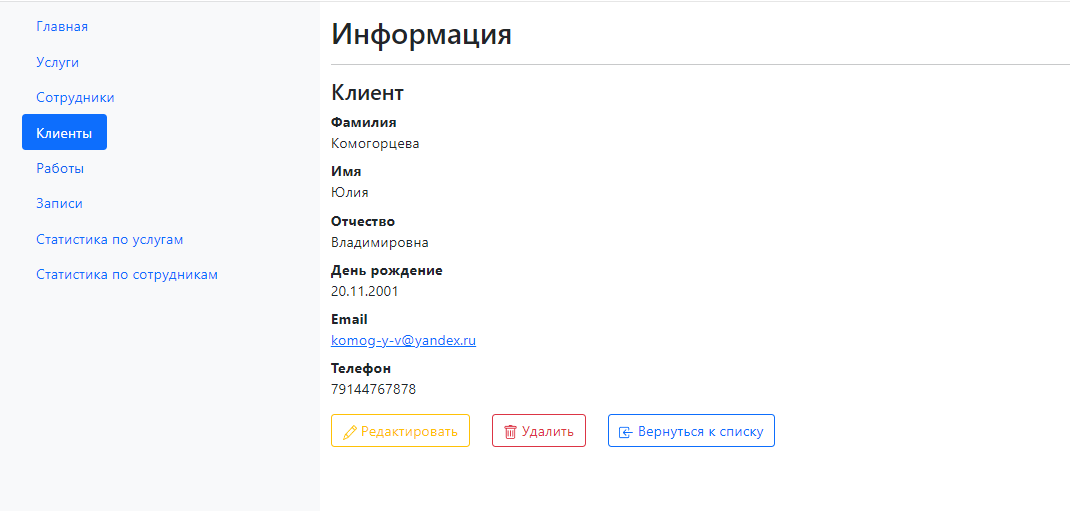
****

Рисунок 12 – Просмотр

@model Web1.Models.Clients

@{

ViewBag.Title = “Details”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Информация</h2>

<div>

<hr />

<h4>Клиент</h4>

<dl class=”dl-horizontal”>

<dt>

@Html.LabelFor(model => model.Surname, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Surname)

</dd>

<dt>

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Name)

</dd>

<dt>

@Html.LabelFor(model => model.Middle, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Middle)

</dd>

<dt>

@Html.LabelFor(model => model.Birhdate, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Birhdate)

</dd>

<dt>

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Email)

</dd>

<dt>

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Phone)

</dd>

<dt>

@Html.DisplayFor(model => model.Comment)

</dt>

<dd class=”mb-2”>

@Html.DisplayFor(model => model.Comment)

</dd>

</dl>

</div>

<div class=”row”>

<div class=”col-auto”>

<a class=”btn btn-outline-warning” href=”@Url.Action(“Edit”, “Clients”, new { id = Model.Id })”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-pencil” viewBox=”0 0 16 16”>

<path d=”M12.146.146a.5.5 0 0 1 .708 0l3 3a.5.5 0 0 1 0 .708l-10 10a.5.5 0 0 1-.168.11l-5 2a.5.5 0 0 1-.65-.65l2-5a.5.5 0 0 1 .11-.168l10-10zM11.207 2.5 13.5 4.793 14.793 3.5 12.5 1.207 11.207 2.5zm1.586 3L10.5 3.207 4 9.707V10h.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.5h.293l6.5-6.5zm-9.761 5.175-.106.106-1.528 3.821 3.821-1.528.106-.106A.5.5 0 0 1 5 12.5V12h-.5a.5.5 0 0 1-.5-.5V11h-.5a.5.5 0 0 1-.468-.325z” />

</svg>

Редактировать

</a>

</div>

<div class=”col-auto”>

<a class=”btn btn-outline-danger” href=”@Url.Action(“Delete”, “Clients”, new { id = Model.Id })”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-trash” viewBox=”0 0 16 16”>

<path d=”M5.5 5.5A.5.5 0 0 1 6 6v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm2.5 0a.5.5 0 0 1 .5.5v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm3 .5a.5.5 0 0 0-1 0v6a.5.5 0 0 0 1 0V6z” />

<path fill-rule=”evenodd” d=”M14.5 3a1 1 0 0 1-1 1H13v9a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2V4h-.5a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1H6a1 1 0 0 1 1-1h2a1 1 0 0 1 1 1h3.5a1 1 0 0 1 1 1v1zM4.118 4 4 4.059V13a1 1 0 0 0 1 1h6a1 1 0 0 0 1-1V4.059L11.882 4H4.118zM2.5 3V2h11v1h-11z” />

</svg>

Удалить

</a>

</div>

<div class=”col-auto”>

<a class=”btn btn-outline-primary “ href=”@Url.Action(“Index”, “Clients”,null)”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-box-arrow-in-left” viewBox=”0 0 16 16”>

<path fill-rule=”evenodd” d=”M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z” />

<path fill-rule=”evenodd” d=”M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z” />

</svg>

Вернуться к списку

</a>

</div>

</div>

### 5.1.4 Delete

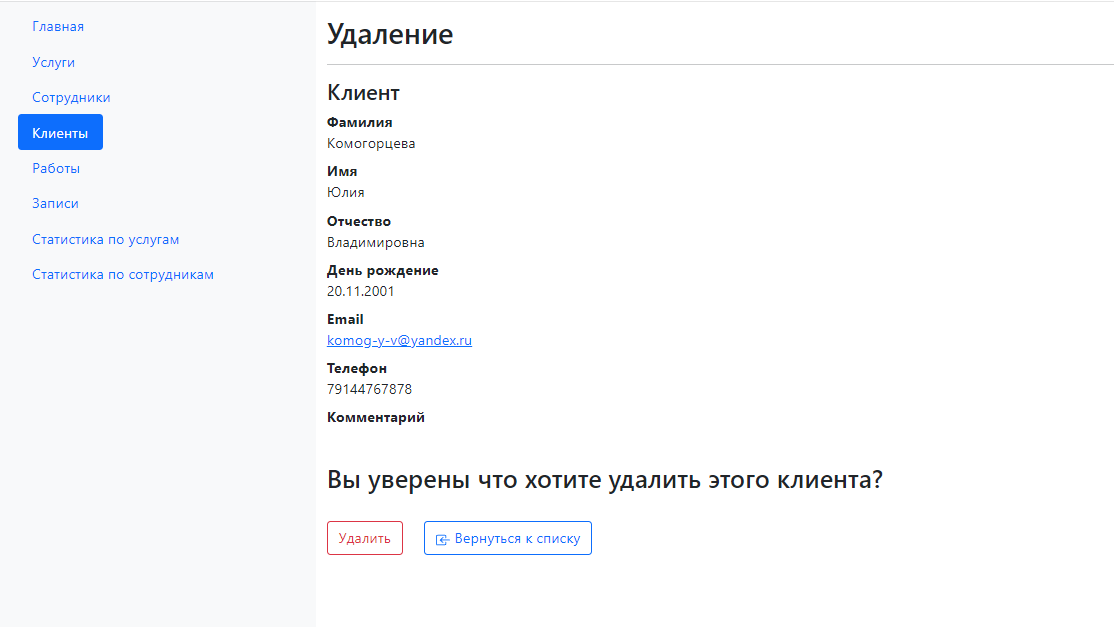
****

Рисунок 13 – Удаление

@model Web1.Models.Clients

@{

ViewBag.Title = “Delete”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Удаление</h2>

<div>

<hr />

<h4>Клиент</h4>

<dl class=”dl-horizontal”>

<dt>

@Html.LabelFor(model => model.Surname, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Surname)

</dd>

<dt>

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Name)

</dd>

<dt>

@Html.LabelFor(model => model.Middle, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Middle)

</dd>

<dt>

@Html.LabelFor(model => model.Birhdate, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Birhdate)

</dd>

<dt>

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Email)

</dd>

<dt>

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Phone)

</dd>

<dt>

@Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = “control-label col-md-2” })

</dt>

<dd>

@Html.DisplayFor(model => model.Comment)

</dd>

</dl>

<br />

<h3>Вы уверены что хотите удалить этого клиента?</h3>

<br />

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class=”form-actions no-color row”>

<div class=”col-auto”>

<input type=”submit” value=”Удалить” class=”btn btn-outline-danger” />

</div>

<div class=”col-auto”>

<a class=”btn btn-outline-primary “ href=”@Url.Action(“Index”, “Clients”,null)”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-box-arrow-in-left” viewBox=”0 0 16 16”>

<path fill-rule=”evenodd” d=”M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z” />

<path fill-rule=”evenodd” d=”M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z” />

</svg>

Вернуться к списку

</a>

</div>

</div>

}

</div>

### 5.1.5 Create

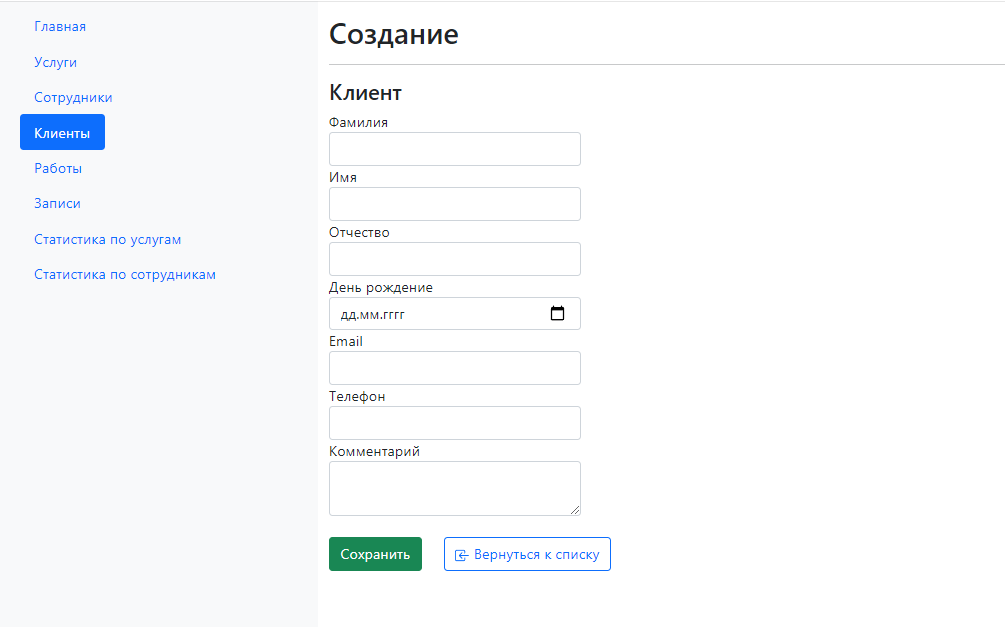


Рисунок 14 – Добавление

@model Web1.Models.Clients

@{

ViewBag.Title = “Create”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class=”form-horizontal”>

<hr />

<h4>Клиент</h4>

@Html.ValidationSummary(true, “”, new { @class = “text-danger” })

<div class=”form-group”>

@Html.LabelFor(model => model.Surname, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Surname, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Surname, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Name, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Middle, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Middle, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Middle, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Birhdate, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Birhdate, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Birhdate, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Email, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Phone, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Phone, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Comment, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Comment, “”, new { @class = “text-danger” })

</div>

</div>

<br />

<div class=”row”>

<div class=”col-auto md-6”>

<div class=”col-md-offset-2 col-md-10”>

<input type=”submit” value=”Сохранить” class=”btn btn-success” />

</div>

</div>

<div class=”col-auto md-6”>

<a class=”btn btn-outline-primary” href=”@Url.Action(“Index”, “Clients”,null)”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-box-arrow-in-left” viewBox=”0 0 16 16”>

<path fill-rule=”evenodd” d=”M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z” />

<path fill-rule=”evenodd” d=”M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z” />

</svg>

Вернуться к списку

</a>

</div>

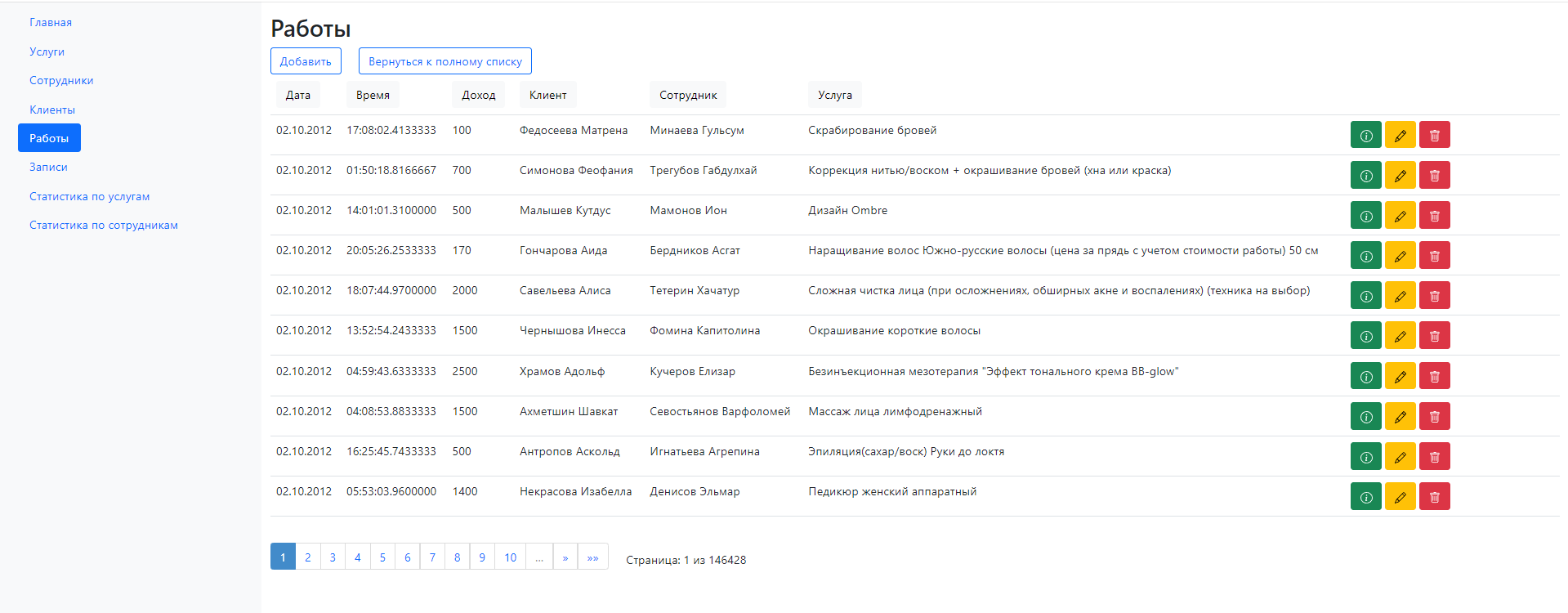
</div>

</div>

}

## 5.2 Работы

### 5.2.1 Index

Рисунок 15 – Index

@model PagedList.IPagedList<Web1.Models.Work>

@using PagedList.Mvc;

<link href="~/Content/PagedList.css" rel="stylesheet" type="text/css" />

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Работы</h2>

<form asp-action="Index" method="get">

<div class="row">

<div class="col-auto">

<a class="btn btn-outline-primary" href="@Url.Action("Create", "Works")">

Добавить

</a>

</div>

<div class="col-auto">

@Html.ActionLink("Вернуться к полному списку", "Index", new { SizePage = @Model.PageSize }, new { @class = "btn btn-outline-primary" })

</div>

</div>

<table class="table">

<tr>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.DateSortParm, currentFilter = ViewBag.CurrentFilter })">Дата</a>

</th>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.TimeSortParm, currentFilter = ViewBag.CurrentFilter })">Время</a>

</th>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.IncomeSortParm, currentFilter = ViewBag.CurrentFilter })">Доход</a>

</th>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.ClientsSortParm, currentFilter = ViewBag.CurrentFilter })">Клиент</a>

</th>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.EmployeeSortParm, currentFilter = ViewBag.CurrentFilter })"> Сотрудник</a>

</th>

<th>

<a class="btn btn-light" href="@Url.Action("Index", "Works", new { sortOrder = ViewBag.ServicesSortParm, currentFilter = ViewBag.CurrentFilter })"> Услуга</a>

</th>

<th></th>

</tr>

@foreach (var item in Model)

{

<tr>

<td>

@Html.ValueFor(modelItem => item.Date, "{0:dd/MM/yyyy}")

</td>

<td>

@Html.DisplayFor(modelItem => item.Time)

</td>

<td>

@Html.DisplayFor(modelItem => item.Income)

</td>

<td>

@Html.DisplayFor(modelItem => item.Clients.Surname) @Html.DisplayFor(modelItem => item.Clients.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Employee.Surname) @Html.DisplayFor(modelItem => item.Employee.Name)

</td>

<td>

@Html.DisplayFor(modelItem => item.Services.Name)

</td>

<td class="col-2">

<a class="btn btn-success" href="@Url.Action("Details", "Works", new { id = item.Id })">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-info-circle" viewBox="0 0 16 16">

<path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14zm0 1A8 8 0 1 0 8 0a8 8 0 0 0 0 16z" />

<path d="m8.93 6.588-2.29.287-.082.38.45.083c.294.07.352.176.288.469l-.738 3.468c-.194.897.105 1.319.808 1.319.545 0 1.178-.252 1.465-.598l.088-.416c-.2.176-.492.246-.686.246-.275 0-.375-.193-.304-.533L8.93 6.588zM9 4.5a1 1 0 1 1-2 0 1 1 0 0 1 2 0z" />

</svg>

</a>

<a class="btn btn-warning" href="@Url.Action("Edit", "Works", new { id = item.Id })">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-pencil" viewBox="0 0 16 16">

<path d="M12.146.146a.5.5 0 0 1 .708 0l3 3a.5.5 0 0 1 0 .708l-10 10a.5.5 0 0 1-.168.11l-5 2a.5.5 0 0 1-.65-.65l2-5a.5.5 0 0 1 .11-.168l10-10zM11.207 2.5 13.5 4.793 14.793 3.5 12.5 1.207 11.207 2.5zm1.586 3L10.5 3.207 4 9.707V10h.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.5h.293l6.5-6.5zm-9.761 5.175-.106.106-1.528 3.821 3.821-1.528.106-.106A.5.5 0 0 1 5 12.5V12h-.5a.5.5 0 0 1-.5-.5V11h-.5a.5.5 0 0 1-.468-.325z" />

</svg>

</a>

<a class="btn btn-danger" href="@Url.Action("Delete", "Works", new { id = item.Id })">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-trash" viewBox="0 0 16 16">

<path d="M5.5 5.5A.5.5 0 0 1 6 6v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm2.5 0a.5.5 0 0 1 .5.5v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm3 .5a.5.5 0 0 0-1 0v6a.5.5 0 0 0 1 0V6z" />

<path fill-rule="evenodd" d="M14.5 3a1 1 0 0 1-1 1H13v9a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2V4h-.5a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1H6a1 1 0 0 1 1-1h2a1 1 0 0 1 1 1h3.5a1 1 0 0 1 1 1v1zM4.118 4 4 4.059V13a1 1 0 0 0 1 1h6a1 1 0 0 0 1-1V4.059L11.882 4H4.118zM2.5 3V2h11v1h-11z" />

</svg>

</a>

</td>

</tr>

}

</table>

<div class="row">

<div class="col-auto">

@Html.PagedListPager(Model, page => Url.Action("Index", new { page, SizePage = @Model.PageSize, sortOrder = ViewBag.CurrentSort, currentFilter = ViewBag.CurrentFilter }))

</div>

<div class="col-auto align-self-center">

<span class="align-middle">Страница: @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) из @Model.PageCount</span>

</div>

</div>

</form>

### 5.2.2 Edit

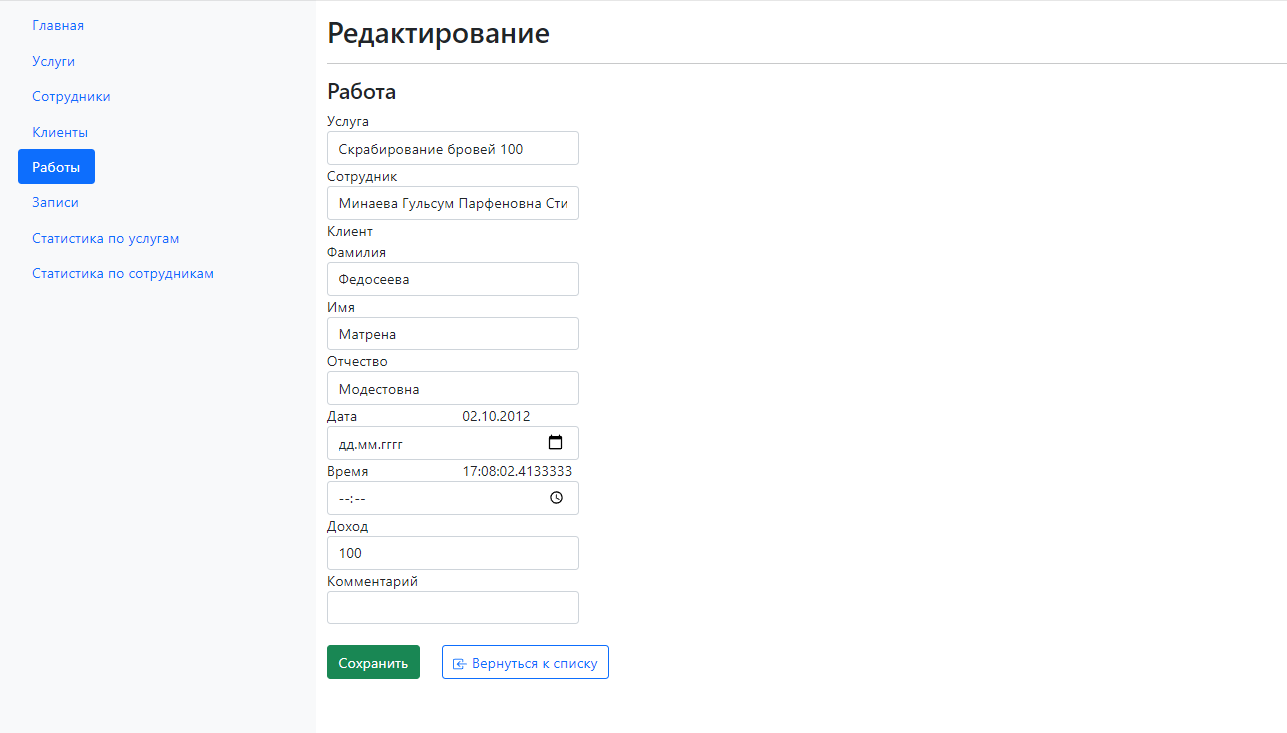
****

Рисунок 16 – Изменение

@model Web1.Models.Work

@{

ViewBag.Title = “Edit”;

Layout = “~/Views/Shared/\_Layout.cshtml”;

}

<h2>Редактирование</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class=”form-horizontal”>

<hr />

<h4>Работа</h4>

@Html.ValidationSummary(true, “”, new { @class = “text-danger” })

@Html.HiddenFor(model => model.Id)

<div class=”form-group”>

@Html.LabelFor(model => model.Id\_Services, null, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.DropDownList(“Id\_Services”, null, htmlAttributes: new { @class = “form-control” })

@Html.ValidationMessageFor(model => model.Id\_Services, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Id\_Employee,null, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.DropDownList(“Id\_Employee”, null, htmlAttributes: new { @class = “form-control” })

@Html.ValidationMessageFor(model => model.Id\_Employee, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Id\_Clients, null, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.LabelFor(model => model.Clients.Surname, null, htmlAttributes: new { @class = “control-label col-md-2” })

@Html.EditorFor(model => model.Clients.Surname, new { htmlAttributes = new { @class = “form-control “ } })

@Html.LabelFor(model => model.Clients.Name, null, htmlAttributes: new { @class = “control-label col-md-2” })

@Html.EditorFor(model => model.Clients.Name, new { htmlAttributes = new { @class = “form-control “ } })

@Html.LabelFor(model => model.Clients.Middle, null, htmlAttributes: new { @class = “control-label col-md-2” })

@Html.EditorFor(model => model.Clients.Middle, new { htmlAttributes = new { @class = “form-control “ } })

@Html.ValidationMessageFor(model => model.Id\_Clients, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Date, htmlAttributes: new { @class = “control-label col-md-1” })

@Html.DisplayFor(model => model.Date)

<div class=”col-md-10”>

@Html.EditorFor(model => model.Date, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Date, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Time, htmlAttributes: new { @class = “control-label col-md-1” })

@Html.DisplayFor(model => model.Time, new { htmlAttributes = new { @class = “form-control col-md-auto” } })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Time, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Time, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Income, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Income, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Income, “”, new { @class = “text-danger” })

</div>

</div>

<div class=”form-group”>

@Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = “control-label col-md-2” })

<div class=”col-md-10”>

@Html.EditorFor(model => model.Comment, new { htmlAttributes = new { @class = “form-control” } })

@Html.ValidationMessageFor(model => model.Comment, “”, new { @class = “text-danger” })

</div>

</div>

<br />

<div class=”row”>

<div class=”col-auto md-6”>

<div class=”col-md-offset-2 col-md-10”>

<input type=”submit” value=”Сохранить” class=”btn btn-success” />

</div>

</div>

<div class=”col-auto md-6”>

<a class=”btn btn-outline-primary” href=”@Url.Action(“Index”, “Works”,null)”>

<svg xmlns=”http://www.w3.org/2000/svg” width=”16” height=”16” fill=”currentColor” class=”bi bi-box-arrow-in-left” viewBox=”0 0 16 16”>

<path fill-rule=”evenodd” d=”M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z” />

<path fill-rule=”evenodd” d=”M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z” />

</svg>

Вернуться к списку

</a>

</div>

</div>

</div>

}

### 5.2.3 Details

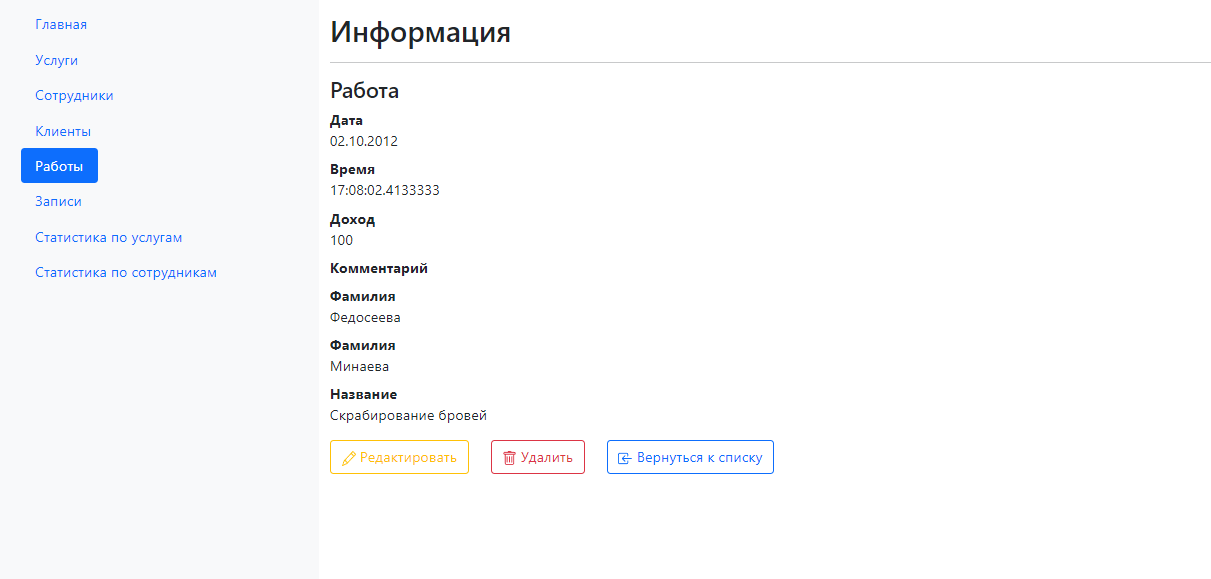
****

Рисунок 17 – Просмотр

@model Web1.Models.Work

@{

ViewBag.Title = "Details";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Информация</h2>

<div>

<hr />

<h4>Работа</h4>

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Date)

</dt>

<dd>

@Html.DisplayFor(model => model.Date)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Time)

</dt>

<dd>

@Html.DisplayFor(model => model.Time)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Income)

</dt>

<dd>

@Html.DisplayFor(model => model.Income)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Comment)

</dt>

<dd>

@Html.DisplayFor(model => model.Comment)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Clients.Surname)

</dt>

<dd>

@Html.DisplayFor(model => model.Clients.Surname)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Employee.Surname)

</dt>

<dd>

@Html.DisplayFor(model => model.Employee.Surname)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Services.Name)

</dt>

<dd>

@Html.DisplayFor(model => model.Services.Name)

</dd>

</dl>

</div>

<div class="row">

<div class="col-auto">

<a class="btn btn-outline-warning" href="@Url.Action("Edit", "Works", new { id = Model.Id })">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-pencil" viewBox="0 0 16 16">

<path d="M12.146.146a.5.5 0 0 1 .708 0l3 3a.5.5 0 0 1 0 .708l-10 10a.5.5 0 0 1-.168.11l-5 2a.5.5 0 0 1-.65-.65l2-5a.5.5 0 0 1 .11-.168l10-10zM11.207 2.5 13.5 4.793 14.793 3.5 12.5 1.207 11.207 2.5zm1.586 3L10.5 3.207 4 9.707V10h.5a.5.5 0 0 1 .5.5v.5h.5a.5.5 0 0 1 .5.5v.5h.293l6.5-6.5zm-9.761 5.175-.106.106-1.528 3.821 3.821-1.528.106-.106A.5.5 0 0 1 5 12.5V12h-.5a.5.5 0 0 1-.5-.5V11h-.5a.5.5 0 0 1-.468-.325z" />

</svg>

Редактировать

</a>

</div>

<div class="col-auto">

<a class="btn btn-outline-danger" href="@Url.Action("Delete", "Works", new { id = Model.Id })">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-trash" viewBox="0 0 16 16">

<path d="M5.5 5.5A.5.5 0 0 1 6 6v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm2.5 0a.5.5 0 0 1 .5.5v6a.5.5 0 0 1-1 0V6a.5.5 0 0 1 .5-.5zm3 .5a.5.5 0 0 0-1 0v6a.5.5 0 0 0 1 0V6z" />

<path fill-rule="evenodd" d="M14.5 3a1 1 0 0 1-1 1H13v9a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2V4h-.5a1 1 0 0 1-1-1V2a1 1 0 0 1 1-1H6a1 1 0 0 1 1-1h2a1 1 0 0 1 1 1h3.5a1 1 0 0 1 1 1v1zM4.118 4 4 4.059V13a1 1 0 0 0 1 1h6a1 1 0 0 0 1-1V4.059L11.882 4H4.118zM2.5 3V2h11v1h-11z" />

</svg>

Удалить

</a>

</div>

<div class="col-auto">

<a class="btn btn-outline-primary " href="@Url.Action("Index", "Works",null)">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-box-arrow-in-left" viewBox="0 0 16 16">

<path fill-rule="evenodd" d="M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z" />

<path fill-rule="evenodd" d="M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z" />

</svg>

Вернуться к списку

</a>

</div>

</div>

### 5.2.4 Delete

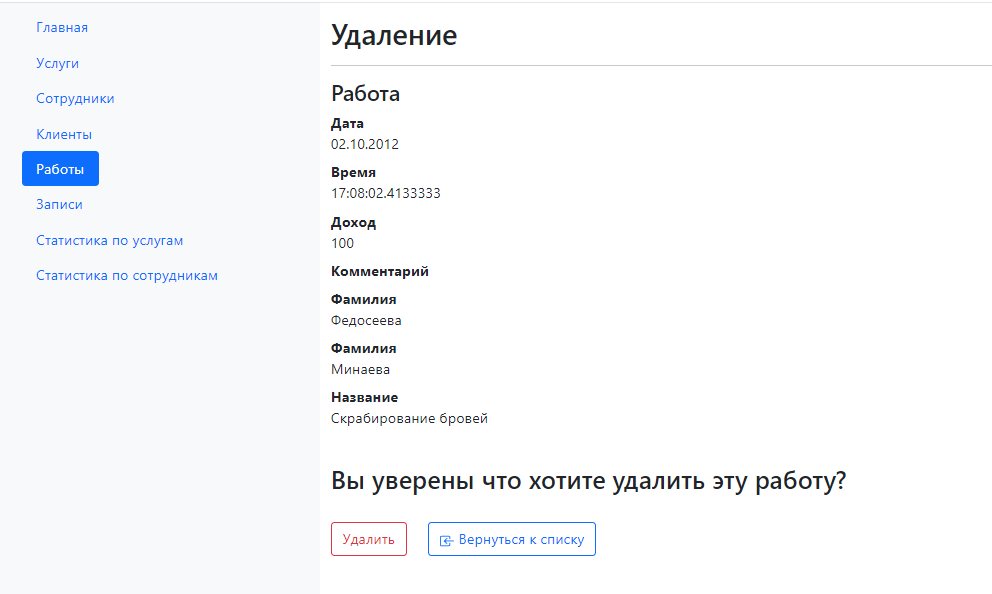
****

Рисунок 18 – Удаление

@model Web1.Models.Work

@{

ViewBag.Title = "Delete";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Удаление</h2>

<div>

<hr />

<h4>Работа</h4>

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.Date)

</dt>

<dd>

@Html.DisplayFor(model => model.Date)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Time)

</dt>

<dd>

@Html.DisplayFor(model => model.Time)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Income)

</dt>

<dd>

@Html.DisplayFor(model => model.Income)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Comment)

</dt>

<dd>

@Html.DisplayFor(model => model.Comment)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Clients.Surname)

</dt>

<dd>

@Html.DisplayFor(model => model.Clients.Surname)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Employee.Surname)

</dt>

<dd>

@Html.DisplayFor(model => model.Employee.Surname)

</dd>

<dt>

@Html.DisplayNameFor(model => model.Services.Name)

</dt>

<dd>

@Html.DisplayFor(model => model.Services.Name)

</dd>

</dl>

<br />

<h3>Вы уверены что хотите удалить эту работу?</h3>

<br />

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-actions no-color row">

<div class="col-auto">

<input type="submit" value="Удалить" class="btn btn-outline-danger" />

</div>

<div class="col-auto">

<a class="btn btn-outline-primary " href="@Url.Action("Index", "Works",null)">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-box-arrow-in-left" viewBox="0 0 16 16">

<path fill-rule="evenodd" d="M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z" />

<path fill-rule="evenodd" d="M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z" />

</svg>

Вернуться к списку

</a>

</div>

</div>

}

</div>

### 5.2.5 Create

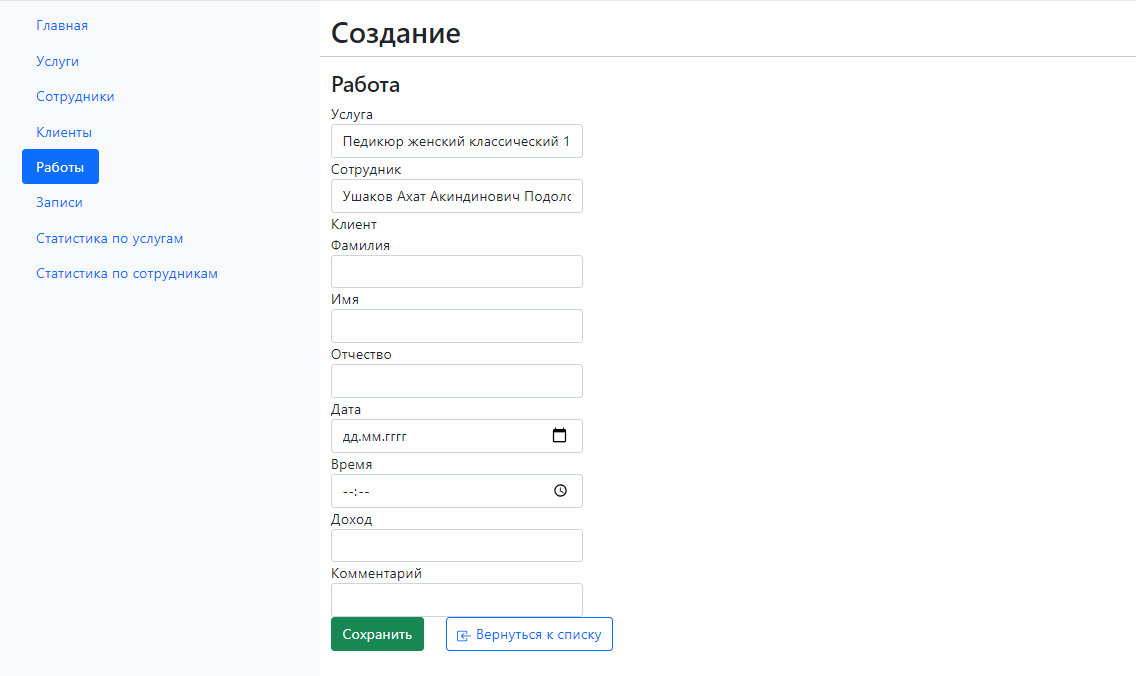


Рисунок 19 – Добавление

@model Web1.Models.Work

@{

ViewBag.Title = "Create";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Создание</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal row">

<hr />

<h4>Работа</h4>

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group col-8">

@Html.LabelFor(model => model.Id\_Services, null, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.DropDownList("Id\_Services", null, htmlAttributes: new { @class = "form-control col-8" })

@Html.ValidationMessageFor(model => model.Id\_Services, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group col-8">

@Html.LabelFor(model => model.Id\_Employee, null, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.DropDownList("Id\_Employee", null, htmlAttributes: new { @class = "form-control col-8" })

@Html.ValidationMessageFor(model => model.Id\_Employee, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group col-8">

@Html.LabelFor(model => model.Id\_Clients, null, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.LabelFor(model => model.Clients.Surname, null, htmlAttributes: new { @class = "control-label col-md-2" })

@Html.EditorFor(model => model.Clients.Surname, new { htmlAttributes = new { @class = "form-control col-8" } })

@Html.LabelFor(model => model.Clients.Name, null, htmlAttributes: new { @class = "control-label col-md-2" })

@Html.EditorFor(model => model.Clients.Name, new { htmlAttributes = new { @class = "form-control col-8" } })

@Html.LabelFor(model => model.Clients.Middle, null, htmlAttributes: new { @class = "control-label col-md-2" })

@Html.EditorFor(model => model.Clients.Middle, new { htmlAttributes = new { @class = "form-control col-8" } })

@Html.ValidationMessageFor(model => model.Id\_Clients, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group col-8">

@Html.LabelFor(model => model.Date, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.EditorFor(model => model.Date, new { htmlAttributes = new { @class = "form-control col-8" } })

@Html.ValidationMessageFor(model => model.Date, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Time, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.EditorFor(model => model.Time, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Time, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Income, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.EditorFor(model => model.Income, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Income, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Comment, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-8">

@Html.EditorFor(model => model.Comment, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Comment, "", new { @class = "text-danger" })

</div>

</div>

<br />

<div class="row">

<div class="col-auto md-6">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Сохранить" class="btn btn-success" />

</div>

</div>

<div class="col-auto md-6">

<a class="btn btn-outline-primary" href="@Url.Action("Index", "Works",null)">

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-box-arrow-in-left" viewBox="0 0 16 16">

<path fill-rule="evenodd" d="M10 3.5a.5.5 0 0 0-.5-.5h-8a.5.5 0 0 0-.5.5v9a.5.5 0 0 0 .5.5h8a.5.5 0 0 0 .5-.5v-2a.5.5 0 0 1 1 0v2A1.5 1.5 0 0 1 9.5 14h-8A1.5 1.5 0 0 1 0 12.5v-9A1.5 1.5 0 0 1 1.5 2h8A1.5 1.5 0 0 1 11 3.5v2a.5.5 0 0 1-1 0v-2z" />

<path fill-rule="evenodd" d="M4.146 8.354a.5.5 0 0 1 0-.708l3-3a.5.5 0 1 1 .708.708L5.707 7.5H14.5a.5.5 0 0 1 0 1H5.707l2.147 2.146a.5.5 0 0 1-.708.708l-3-3z" />

</svg>

Вернуться к списку

</a>

</div>

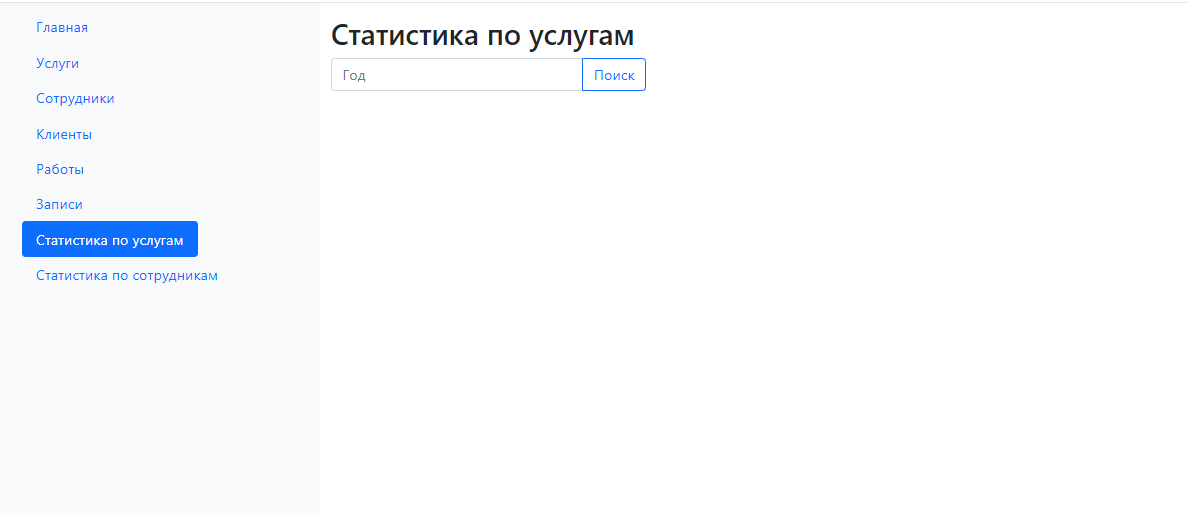
</div>

</div>

}

**5.3 Выходной документ со статистикой по услугам**

**5.3.1 Index**

Рисунок 20 – Index

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<div class="row">

<h2>Статистика по услугам</h2>

@using (Html.BeginForm("Details", "StatisticsServices", FormMethod.Get))

{

<div class="col-auto">

<form asp-action="Index" method="get">

<div class="form-actions no-color row">

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

<input type="text" id="SearchString" name="SearchString" class="form-control" placeholder="Год" aria-label="Введите год">

<input type="submit" value="Поиск" class="btn btn-outline-primary" />

</div>

</div>

</form>

</div>

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

}

</div>

**5.3.2 Details**

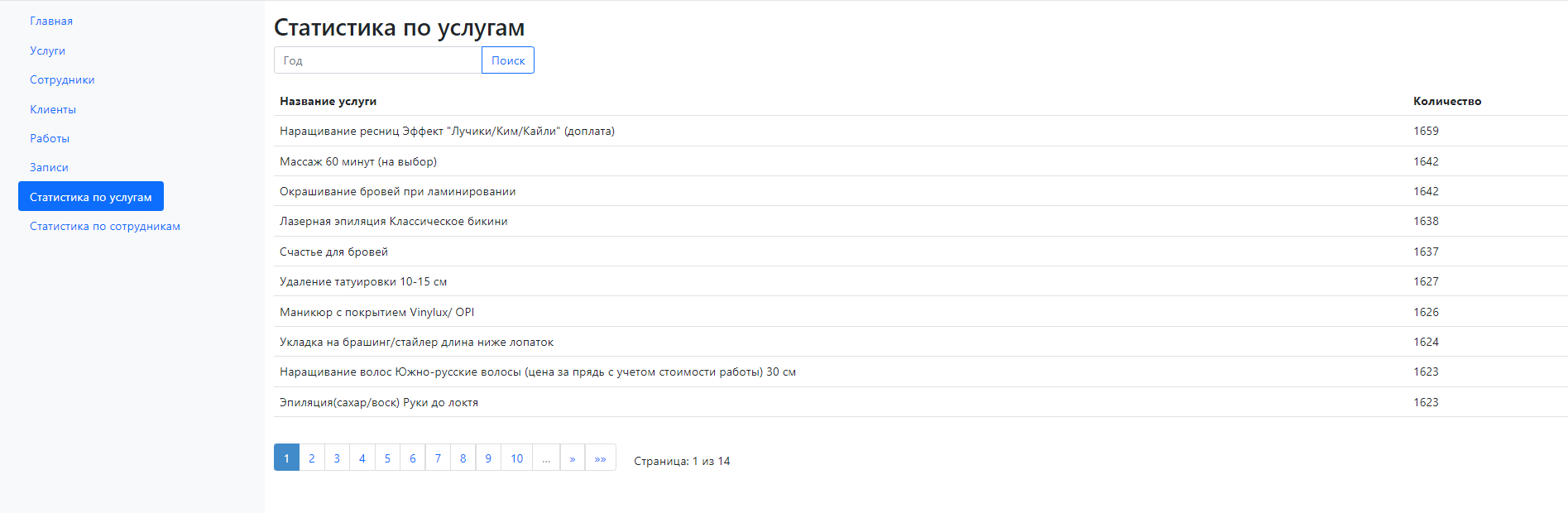
****

Рисунок 21 – Просмотр

@model PagedList.IPagedList<Web1.Models.SP1\_Result>

@using PagedList.Mvc;

<link href="~/Content/PagedList.css" rel="stylesheet" type="text/css" />

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Статистика по услугам</h2>

@using (Html.BeginForm("Details", "StatisticsServices", FormMethod.Get))

{

<div class="col-auto">

<form asp-action="Index" method="get">

<div class="form-actions no-color row">

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

<input type="text" id="SearchString" name="SearchString" class="form-control" placeholder="Год" aria-label="Введите год">

<input type="submit" value="Поиск" class="btn btn-outline-primary" />

</div>

</div>

</form>

</div>

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

}

@if (!TempData["msg"].Equals(""))

{

<h1>Результатов не найдено </h1>

}

@if (TempData["msg"].Equals(""))

{

<table class="table">

<tr>

<th>

Название услуги

</th>

<th>

Количество

</th>

</tr>

@foreach (var item in Model)

{

<tr>

<td> @Html.DisplayFor(modelItem => item.Name)</td>

<td> @Html.DisplayFor(modelItem => item.Column1)</td>

</tr>

}

</table>

<div class="row">

<div class="col-auto">

<form asp-action="Index" method="get">

@Html.PagedListPager(Model, page => Url.Action("Details", new { page, sortOrder = ViewBag.CurrentSort, currentFilter = ViewBag.CurrentFilter, currentFilter1 = ViewBag.CurrentFilter1 }))

</form>

</div>

<div class="col-auto align-self-center">

<span class="align-middle">Страница: @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) из @Model.PageCount</span>

</div>

</div>

}

# 6 Контроль

Большую роль в приложении играет валидация модели или проверка вводимых данных на корректность. Фреймворк .NET предлагает удобный функционал для валидации в виде атрибутов из пространства имен System.ComponentModel.DataAnnotations.

Все правила валидации модели в System.ComponentModel.DataAnnotations определяются в виде атрибутов. В данном случае были использованы атрибуты: Required, Display, DataType, Range, UIHint и RegularExpression.

## 6.1 Services

namespace Web1.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Web.Mvc;

public partial class Services

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Services()

{

this.Record = new HashSet<Record>();

this.Work = new HashSet<Work>();

}

public int Id { get; set; }

[Display(Name = "Название")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Name { get; set; }

[Display(Name = "Стоимость")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

[RegularExpression(@"[z0-9]{0,10}", ErrorMessage = "Некорректная стоимость")]

public int Price { get; set; }

[DataType(DataType.MultilineText)]

[Display(Name = "Комментарий")]

public string Comment { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Record> Record { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Work> Work { get; set; }

public string Concat

{

get

{

return string.Format("{0} {1} ", Name, Price);

}

set { }

}

}

}

## 6.2 Clients

namespace Web1.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using Xunit;

public partial class Clients

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Clients()

{

this.Email = "-";

this.Record = new HashSet<Record>();

this.Work = new HashSet<Work>();

}

public int Id { get; set; }

[Display(Name = "Фамилия")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Surname { get; set; }

[Display(Name = "Имя")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Name { get; set; }

[Display(Name = "Отчество")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Middle { get; set; }

[Display(Name = "День рождение")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

[DataType(DataType.Date)]

[Range(typeof(DateTime), "01.01.1940", "01.01.2008",

ErrorMessage = "Значение для {0} должно быть между {1} и {2}")]

public Nullable<System.DateTime> Birhdate { get ; set; }

[RegularExpression(@"[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}", ErrorMessage = "Некорректный адрес")]

[DataType(DataType.EmailAddress)]

public string Email { get; set; }

[Display(Name = "Телефон")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

[DataType(DataType.PhoneNumber)]

[UIHint("String")]

[RegularExpression(@"^\+?[78][-\(]?\d{3}\)?-?\d{3}-?\d{2}-?\d{2}$", ErrorMessage = "Некорректный номер телефона")]

public decimal Phone { get; set; }

[Display(Name = "Комментарий")]

[DataType(DataType.MultilineText)]

public string Comment { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Record> Record { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Work> Work { get; set; }

}

}

## 6.3Employee

namespace Web1.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

public partial class Employee

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Employee()

{

this.Record = new HashSet<Record>();

this.Work = new HashSet<Work>();

}

public int Id { get; set; }

[Display(Name = "Фамилия")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Surname { get; set; }

[Display(Name = "Имя")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Name { get; set; }

[Display(Name = "Отчество")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Middle { get; set; }

[Display(Name = "Телефон")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

[DataType(DataType.PhoneNumber)]

[UIHint("String")]

[RegularExpression(@"^\+?[78][-\(]?\d{3}\)?-?\d{3}-?\d{2}-?\d{2}$", ErrorMessage = "Некорректный номер телефона")]

public decimal Phone { get; set; }

[Display(Name = "День рождение")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

[DataType(DataType.Date)]

[Range(typeof(DateTime), "01.01.1940", "01.01.2008",

ErrorMessage = "Значение для {0} должно быть между {1} и {2}")]

public Nullable<System.DateTime> Birhdate { get; set; }

[Display(Name = "Должность")]

[Required(ErrorMessage = "Поле должно быть заполнено")]

public string Post { get; set; }

[Display(Name = "Комментарий")]

[DataType(DataType.MultilineText)]

public string Comment { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Record> Record { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Work> Work { get; set; }

public string FIO

{

get

{

return string.Format("{0} {1} {2} {3}", Surname, Name, Middle,Post);

}

set { }

}

}

}

## 6.4 Record

namespace Web1.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

public partial class Record

{

public int Id { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Услуга")]

public int Id\_Services { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Сотрудник")]

public int Id\_Employee { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Клиент")]

public int Id\_Clients { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Дата")]

[DataType(DataType.Date)]

public System.DateTime Date { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Время")]

[DataType(DataType.Time)]

public System.TimeSpan Time { get; set; }

[Display(Name = "Комментарий")]

public string Comment { get; set; }

public virtual Clients Clients { get; set; }

public virtual Employee Employee { get; set; }

public virtual Services Services { get; set; }

}

}

## 6.5 Work

namespace Web1.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

public partial class Work

{

public int Id { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Услуга")]

public int Id\_Services { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Сотрудник")]

public int Id\_Employee { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Клиент")]

public int Id\_Clients { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Дата")]

[DataType(DataType.Date)]

public System.DateTime Date { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Время")]

[DataType(DataType.Time)]

public System.TimeSpan Time { get; set; }

[Required(ErrorMessage = "Поле должно быть заполнено")]

[Display(Name = "Доход")]

[RegularExpression(@"[z0-9]{0,10}", ErrorMessage = "Некорректная стоимость")]

public double Income { get; set; }

[Display(Name = "Комментарий")]

public string Comment { get; set; }

public virtual Clients Clients { get; set; }

public virtual Employee Employee { get; set; }

public virtual Services Services { get; set; }

}

}

# 7 Таблица тестов

Результаты тестирования работоспособности реализованного приложения приведены в таблице 1.

Таблица 1 – Тесты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 1 | Services | Добавление | Название: Анестезия1  Стоимость: 300  Комментарий: | Данные добавлены  (см. рисунок 22) |
| 2 | Редактирование | Замена записи:  Название: Анестезия1  Стоимость:300  Комментарий:  На:  Название: Анестезия1234  Стоимость:10  Комментарий: | Данные изменены  (см. рисунок 23) |
| 3 | Просмотр | Просмотр  записи:  Название: Анестезия1234  Стоимость:10  Комментарий: | Просмотр данных  (см. рисунок 24) |
| 4 | Удаление | Удаление  записи:  Название: Анестезия1234  Стоимость: 10  Комментарий: | Данные удалены  (см. рисунок 25) |
| 5 | Employee | Добавление | Фамилия: ааа  Имя: ааа  Отчество: ааа  Телефон: 79246284302  Дата рождения: 20.11.2001  Должность: ааа  Комментарий: | Данные добавлены  (см. рисунок 26) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 6 | Employee | Редактирование | Замена записи:  Фамилия: ааа  Имя: ааа  Отчество: ааа  Телефон: 79246284302  Дата рождения: 20.11.2001  Должность: ааа  Комментарий:  На:  Фамилия: ааа1  Имя: ааа1  Отчество: ааа1  Телефон: 79246284306  Дата рождения: 20.11.2001  Должность: ааа1  Комментарий: | Данные изменены  (см. рисунок 27) |
| 7 | Просмотр | Просмотр  записи:  Фамилия: ааа1  Имя: ааа1  Отчество: ааа1  Телефон: 79246284306  Дата рождения: 20.11.2001  Должность: ааа1  Комментарий: | Просмотр данных  (см. рисунок 28) |
| 8 | Удаление | Удаление  записи:  Фамилия: ааа1  Имя: ааа1  Отчество: ааа1  Телефон: 79246284306  Дата рождения: 20.11.2001  Должность: ааа1  Комментарий: | Данные удалены  (см. рисунок 29) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 9 | Clients | Добавление | Фамилия: ааа  Имя: ааа  Отчество: ааа  Телефон: 79246284301  Дата рождения:20.11.2001  Почта: komog-y-v@yandex.ru  Комментарий: | Данные добавлены  (см. рисунок 30) |
| 10 | Редактирование | Замена записи:  Фамилия: ааа  Имя: ааа  Отчество: ааа  Телефон: 79246284301  Дата рождения:20.11.2001  Почта: komog-y-v@yandex.ru  Комментарий:  На:  Фамилия: ааа1  Имя: ааа1  Отчество: ааа1  Телефон: 79246284323  Дата рождения:21.11.2001  Почта: komog-y-v1@yandex.ru  Комментарий: | Данные изменены  (см. рисунок 31) |
| 11 | Просмотр | Просмотр  записи:  Фамилия: ааа  Имя: ааа  Отчество: ааа  Телефон: 79246284301  Дата рождения:20.11.2001  Почта: komog-y-v@yandex.ru  Комментарий: | Просмотр данных  (см. рисунок 32) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 12 | Clients | Удаление | Удаление  записи:  Фамилия: ааа1  Имя: ааа1  Отчество: ааа1  Телефон: 79246284323  Дата рождения:21.11.2001  Почта: komog-y-v1@yandex.ru  Комментарий: | Данные удалены  (см. рисунок 33) |
| 13 | Record | Добавление | Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 18.12.2021  Время: 17:34  Комментарий: | Данные добавлены  (см. рисунок 34) |
| 14 | Редактирование | Замена записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 18.12.2021  Время: 17:34  Комментарий:  На:  Услуга: Наращивание ресниц 3D  Сотрудник: Матвеева Ирина Маркияновна  Клиент: Комогорцева Юлия Владимировна  Дата: 19.01.2001  Время: 18:00  Комментарий: | Данные изменены  (см. рисунок 35) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 15 | Record | Просмотр | Просмотр  записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 18.12.2021  Время: 17:34  Комментарий: | Просмотр данных  (см. рисунок 36) |
| 16 | Удаление | Удаление  записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 18.12.2021  Время: 17:34  Комментарий: | Данные удалены  (см. рисунок 37) |
| 17 | Work | Добавление | Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 31.12.2021  Время: 10:30  Доход: 700  Комментарий: | Данные добавлены  (см. рисунок 38) |
| 18 | Просмотр | Просмотр  записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 31.12.2021  Время: 10:30  Доход: 700  Комментарий: | Просмотр данных  (см. рисунок 39) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 19 | Work | Редактирование | Замена записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 31.12.2021  Время: 10:30  Доход: 700  Комментарий:  На:  Услуга: Скрабирование бровей  Сотрудник: Матвеева Ирина Маркияновна  Клиент: Комогорцева Юлия Владимировна  Дата:01.01.2022  Время:10:00  Доход:150  Комментарий: | Данные изменены  (см. рисунок 40) |
| 19 | Просмотр | Просмотр  записи:  Услуга: Снятие ресниц  Сотрудник: Журавлева Ульяна Самсоновна  Клиент: Комогорцева Юлия Владимировна  Дата: 31.12.2021  Время: 10:30  Доход: 700  Комментарий: | Просмотр данных  (см. рисунок 40) |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Название таблицы | Название  теста | Входные  данные | Выходные  данные |
| 20 | Work | Удаление | Удаление  записи:  Услуга: Скрабирование бровей  Сотрудник: Матвеева Ирина Маркияновна  Клиент: Комогорцева Юлия Владимировна  Дата:01.01.2022  Время:10:00  Доход:150  Комментарий: | Данные удалены  (см. рисунок 41) |
| 21 | Запрос статистики по услугам | Запрос | Год: 2020 | Перечень выходных данных  (см. рисунок 42) |
| 22 | Запрос статистики по сотрудникам | Запрос | Буква: б | Перечень выходных данных  (см. рисунок 43) |
| 23 | Контроль данных | Контроль  данных |  | Перечень используемых контролей  (см. рисунок 44) |

# 8 Результаты тестирования

Результат теста №1

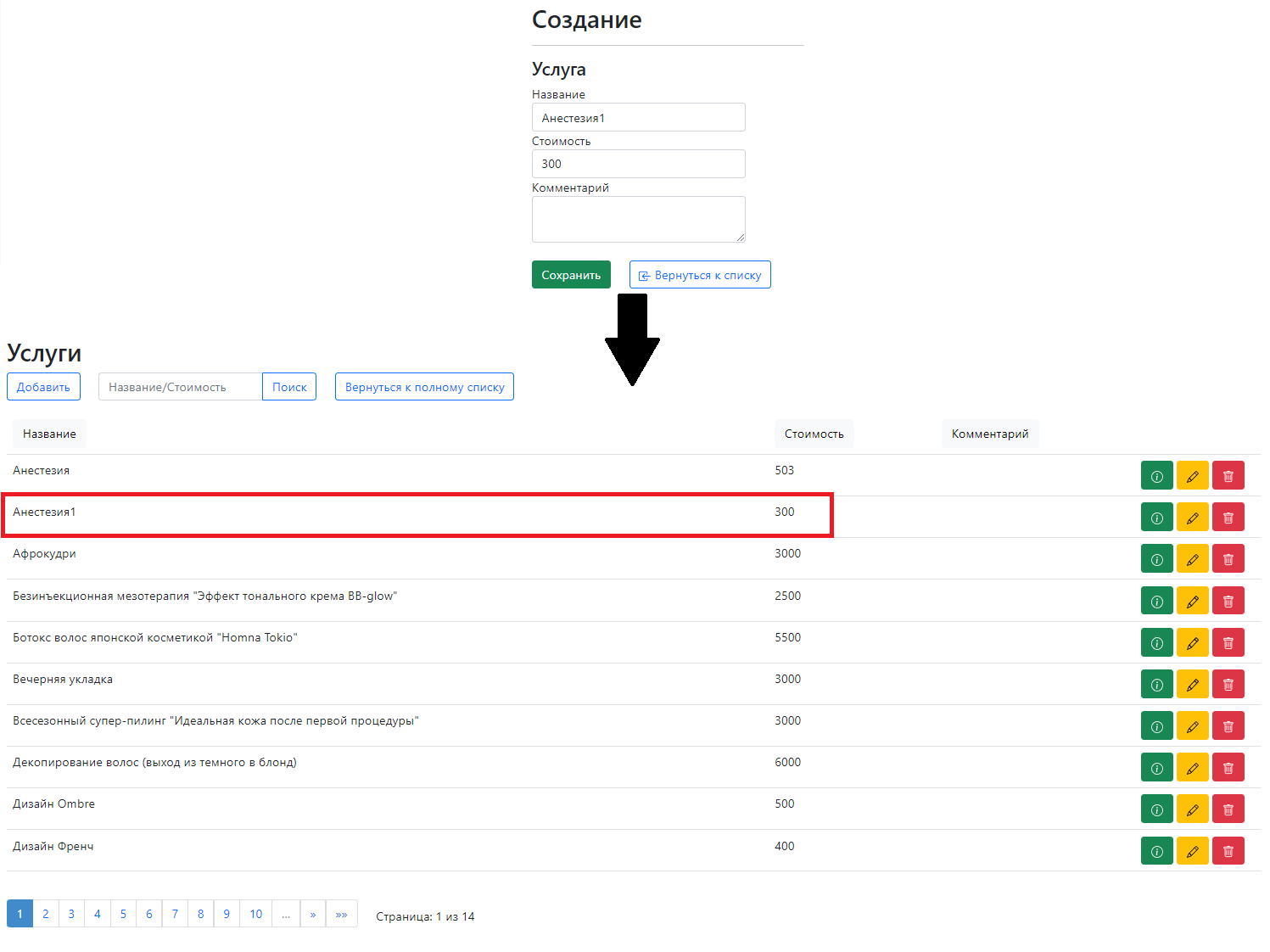


Рисунок 22 – Добавление «Services»

Результат теста №2

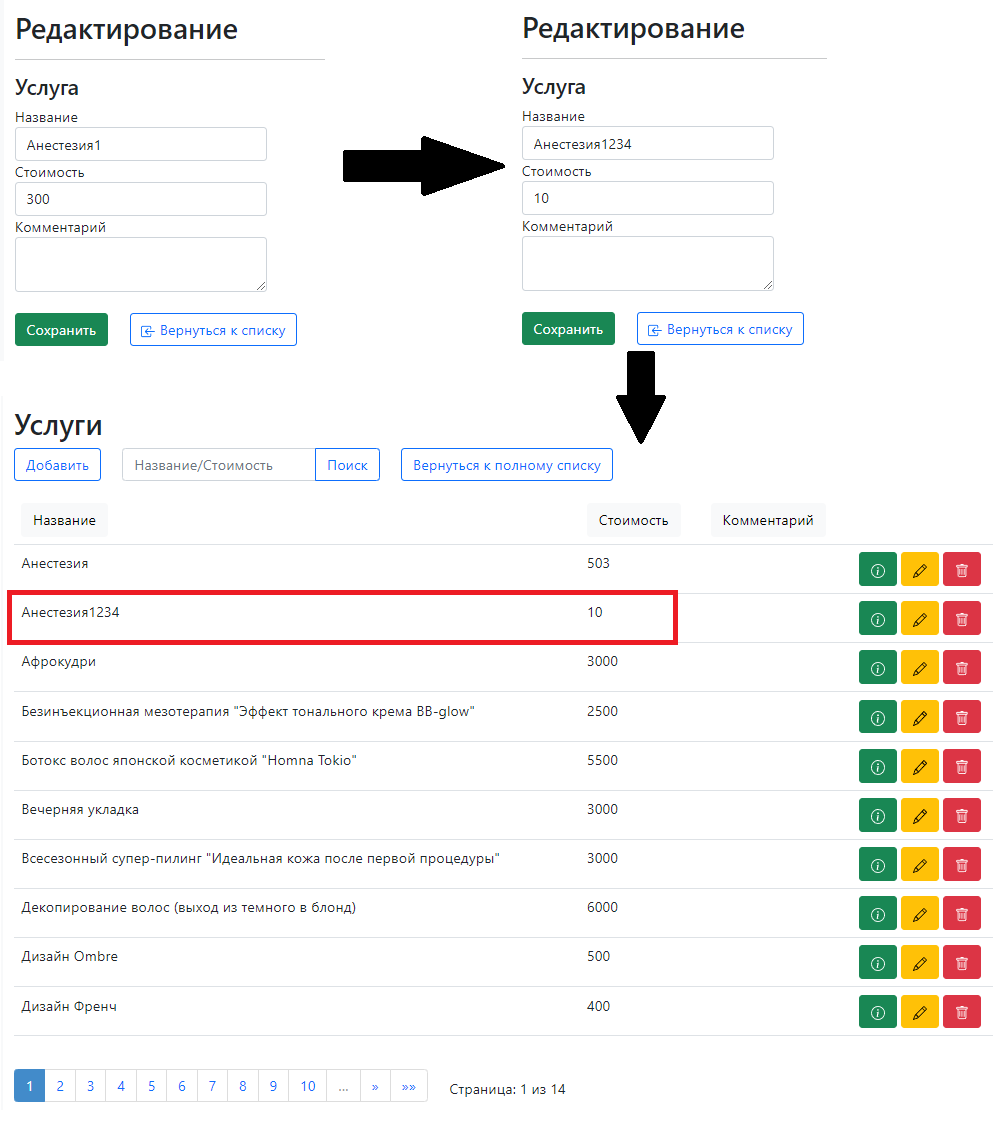


Рисунок 23 – Изменение «Services»

Результат теста №3

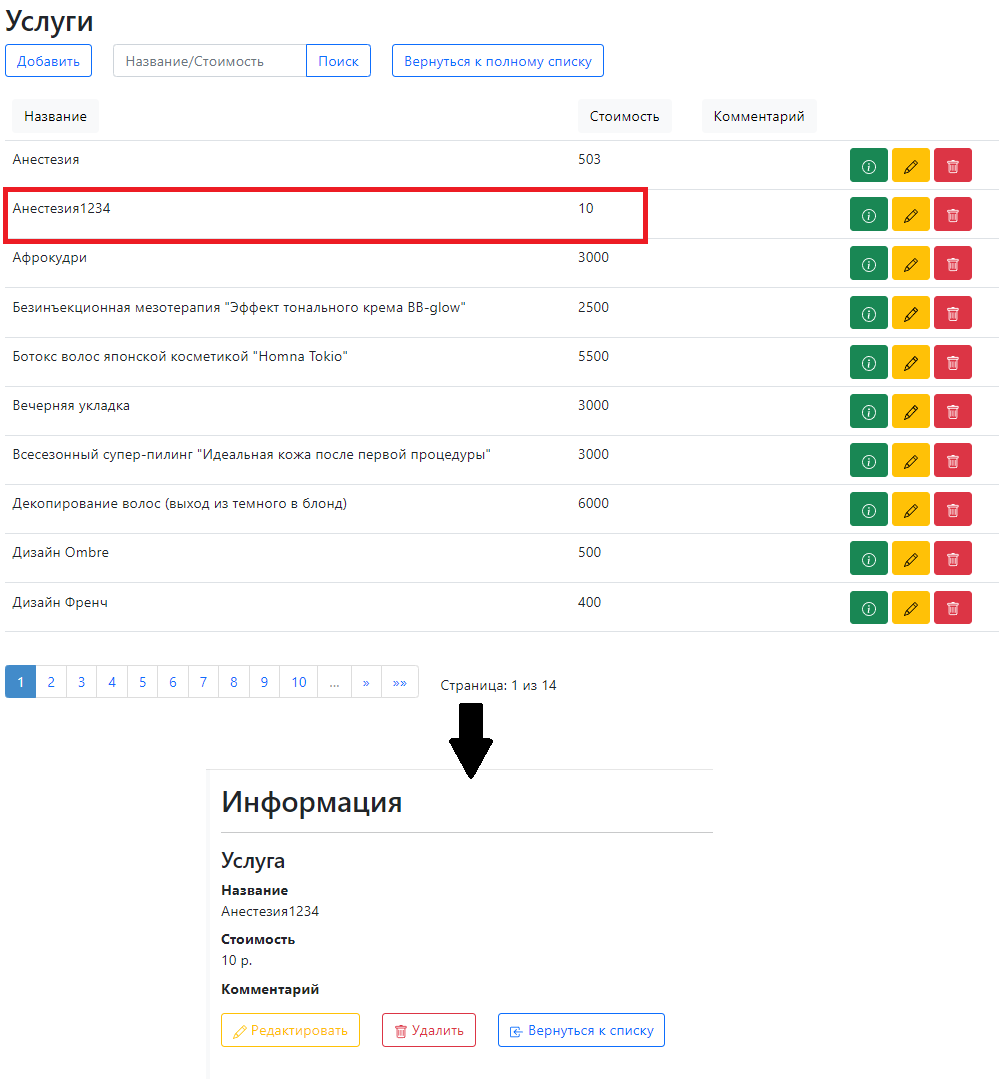


Рисунок 24 –Просмотр «Services»

Результат теста №4

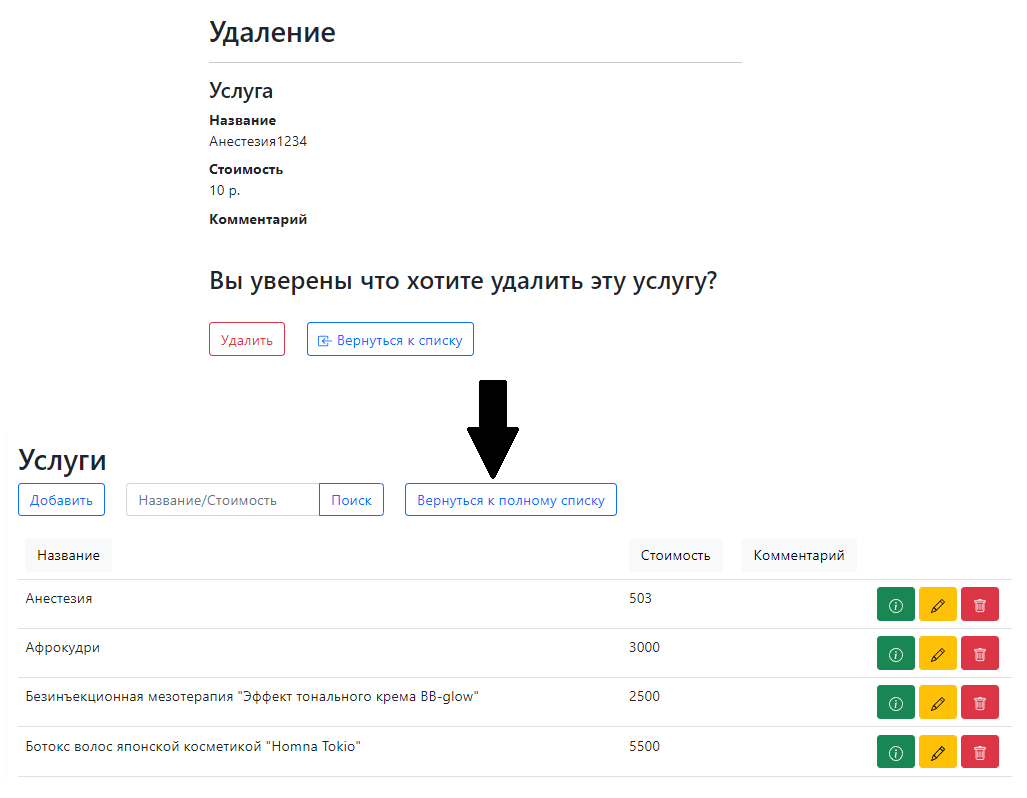


Рисунок 25 – Удаление «Services»

Результат теста №5

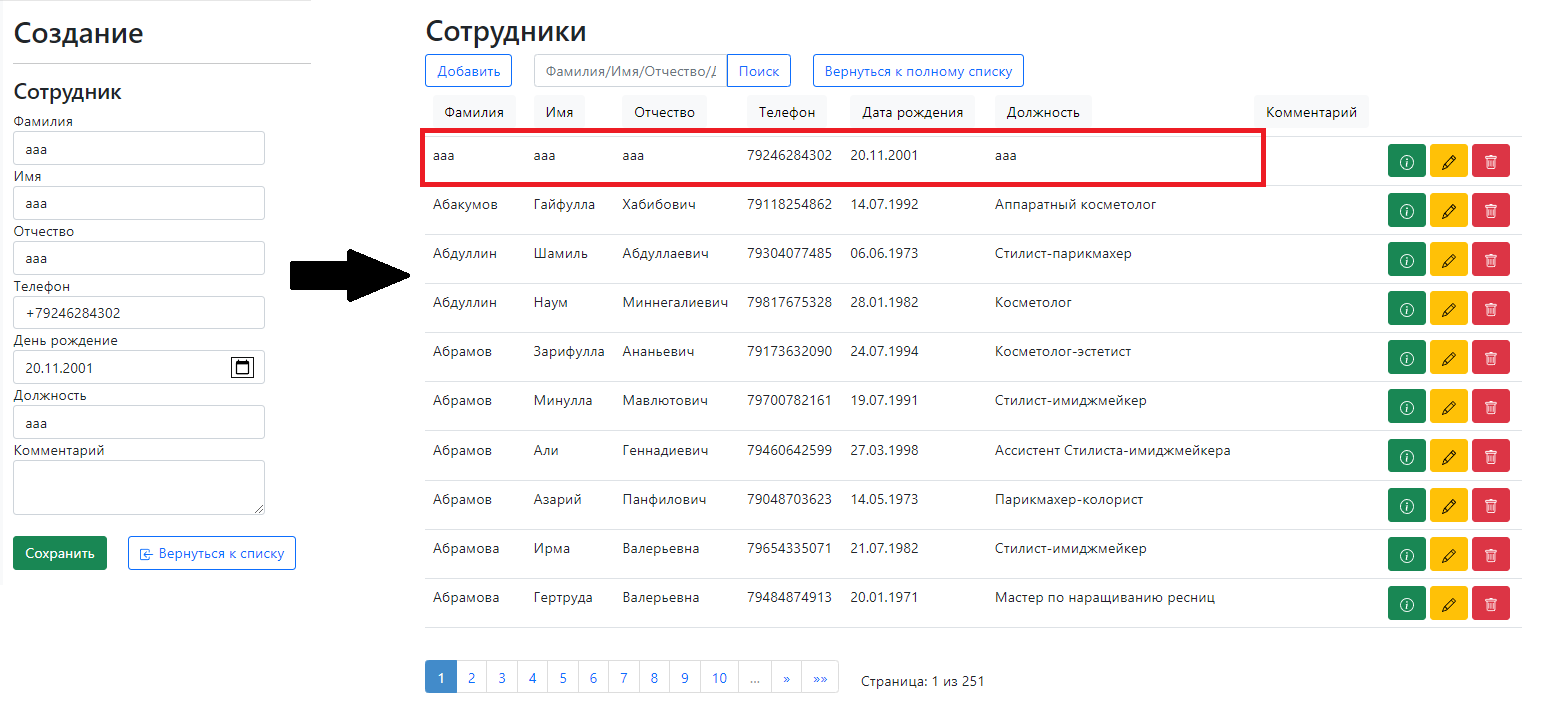


Рисунок 26 – Добавление «Employee»

Результат теста №6

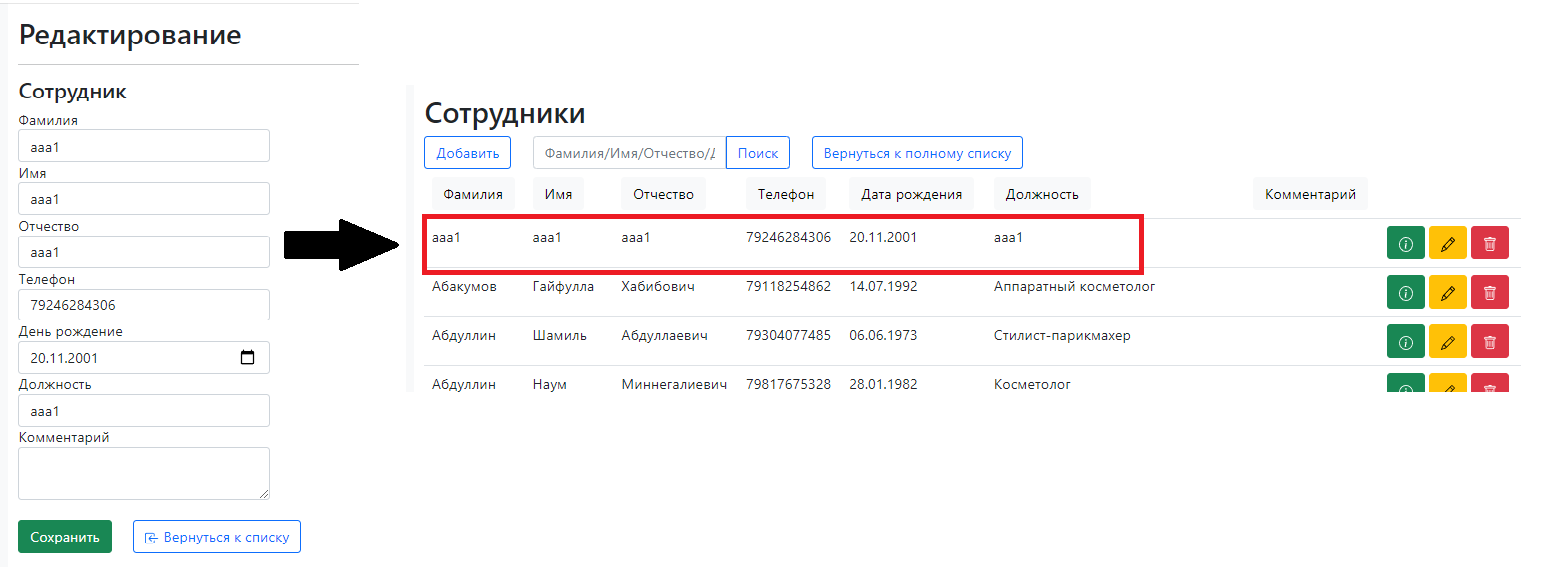


Рисунок 27 – Редактирование «Employee»

Результат теста №7

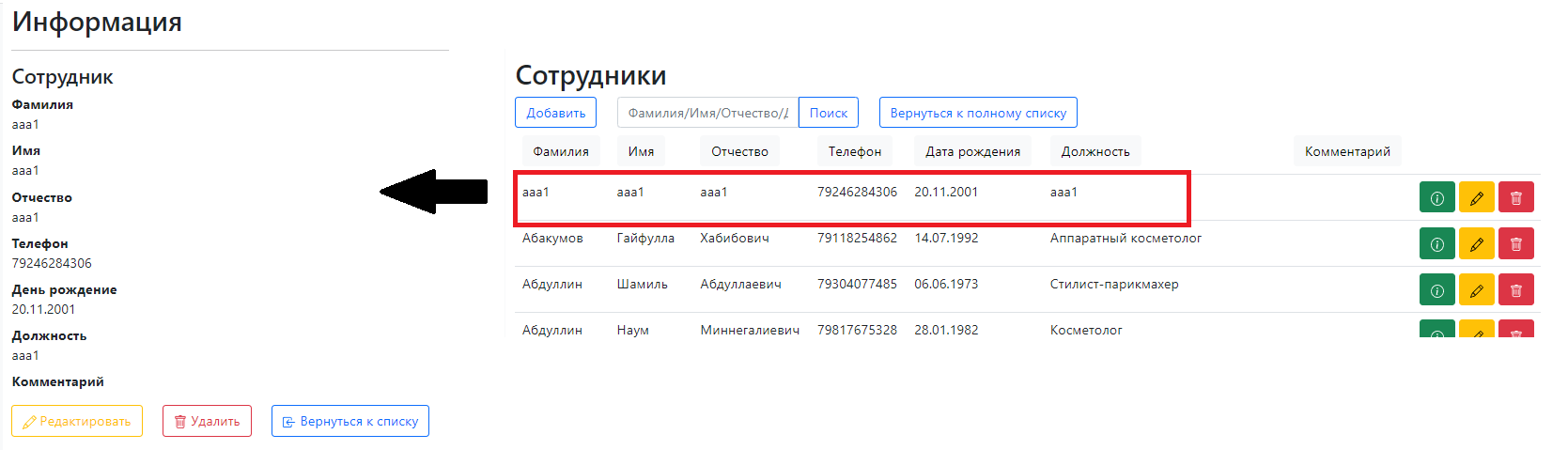


Рисунок 28 – Просмотр «Employee»

Результат теста №8

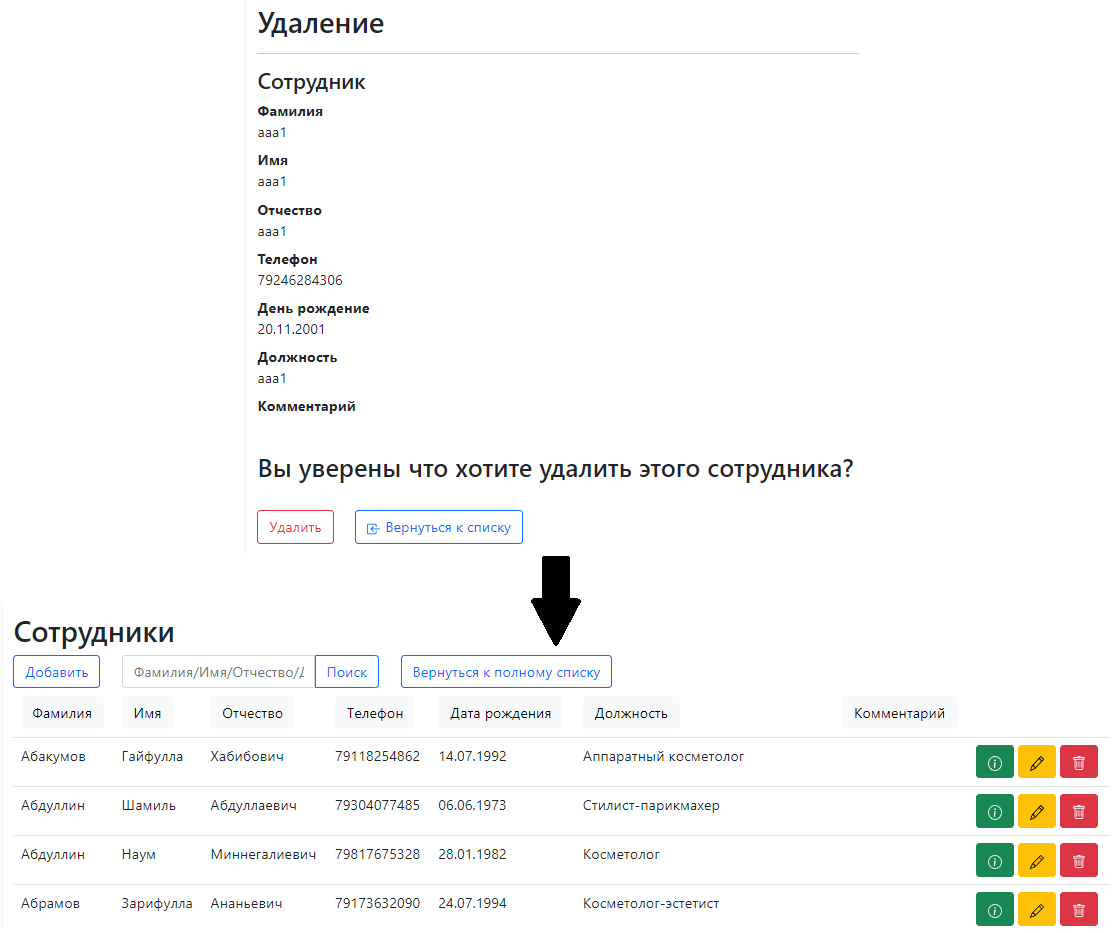


Рисунок 29 – Удаление «Employee»

Результат теста №9

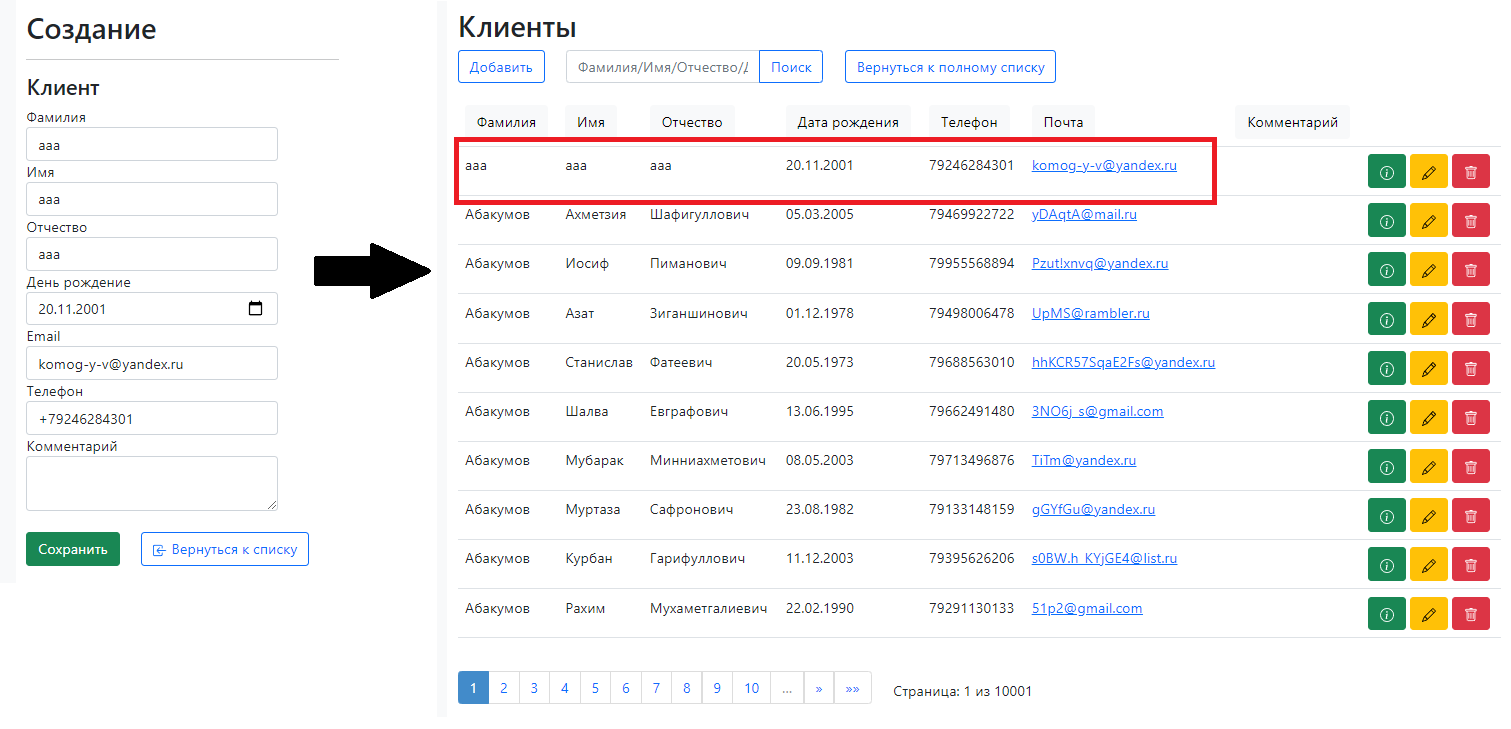


Рисунок 30 – Добавление «Clients»

Результат теста №10

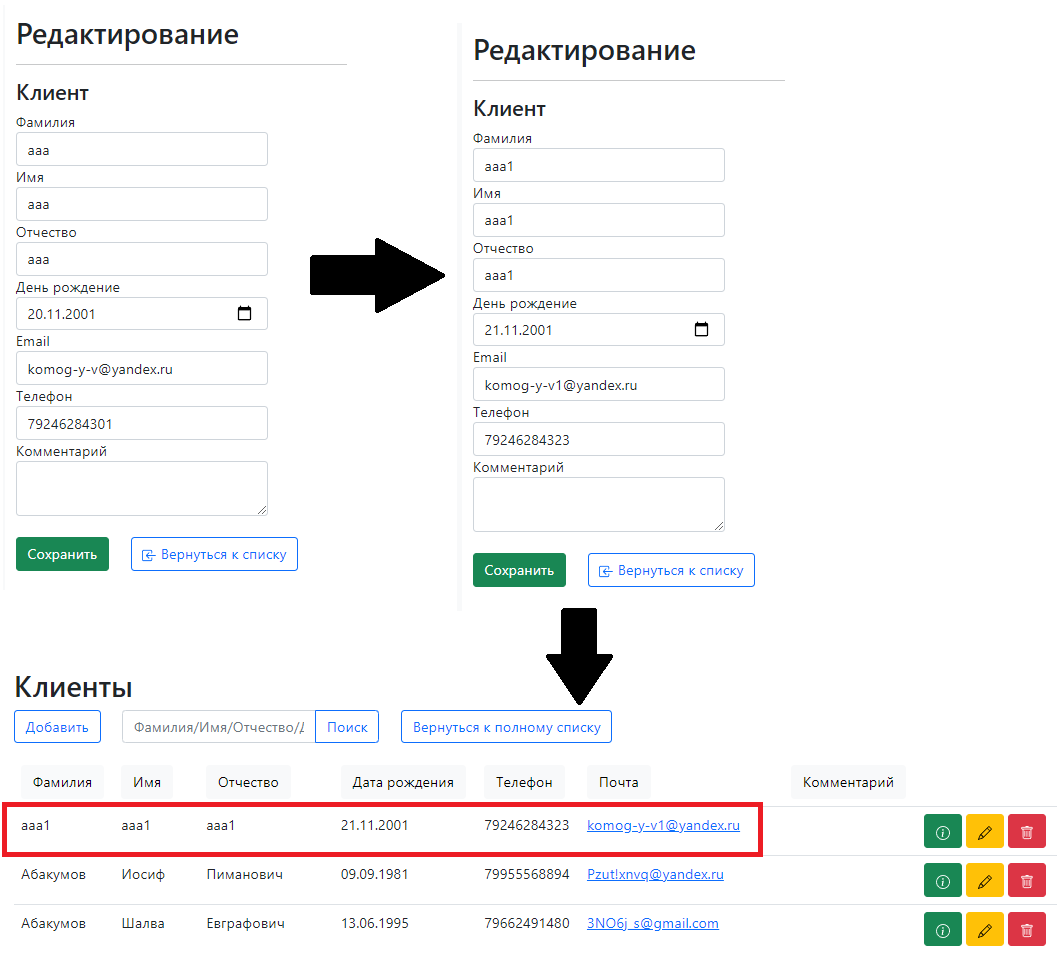


Рисунок 31 – Изменение «Clients»

Результат теста №11

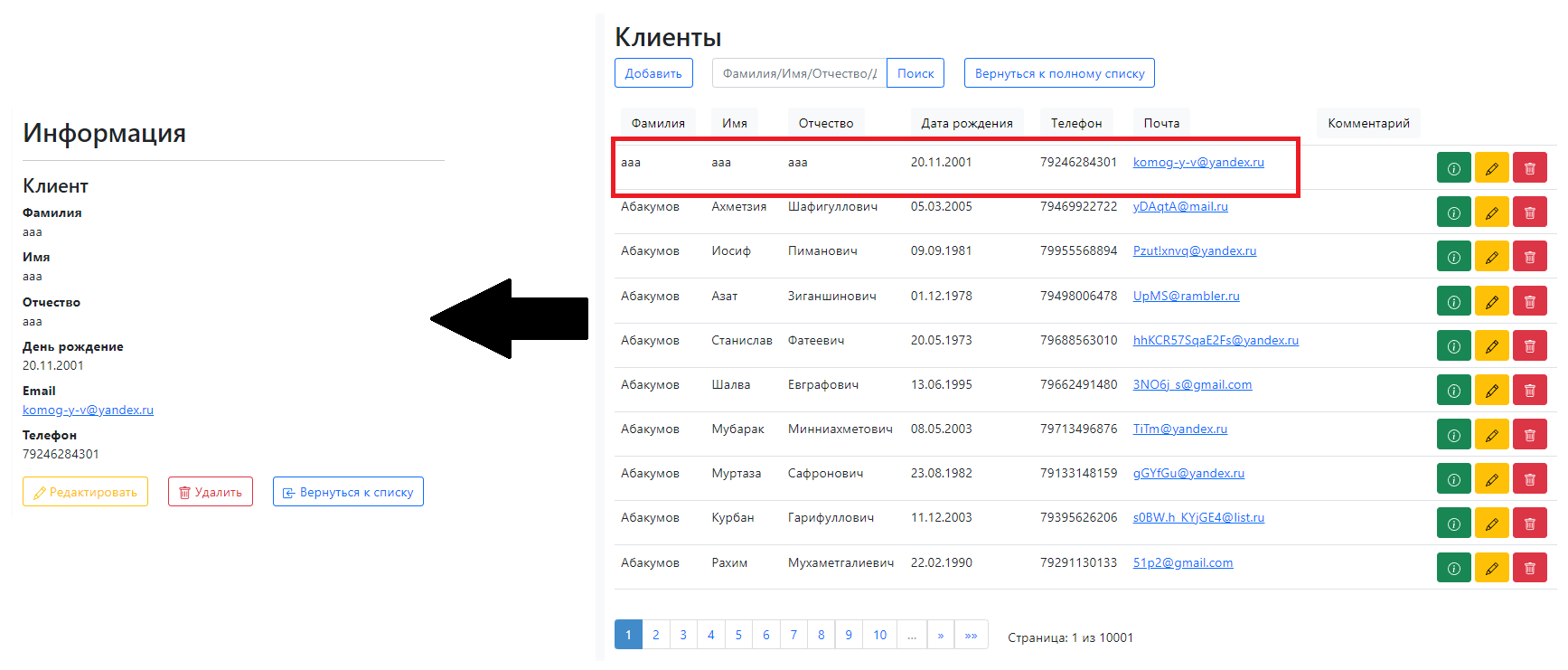


Рисунок 32 –Просмотр «Clients»

Результат теста №12

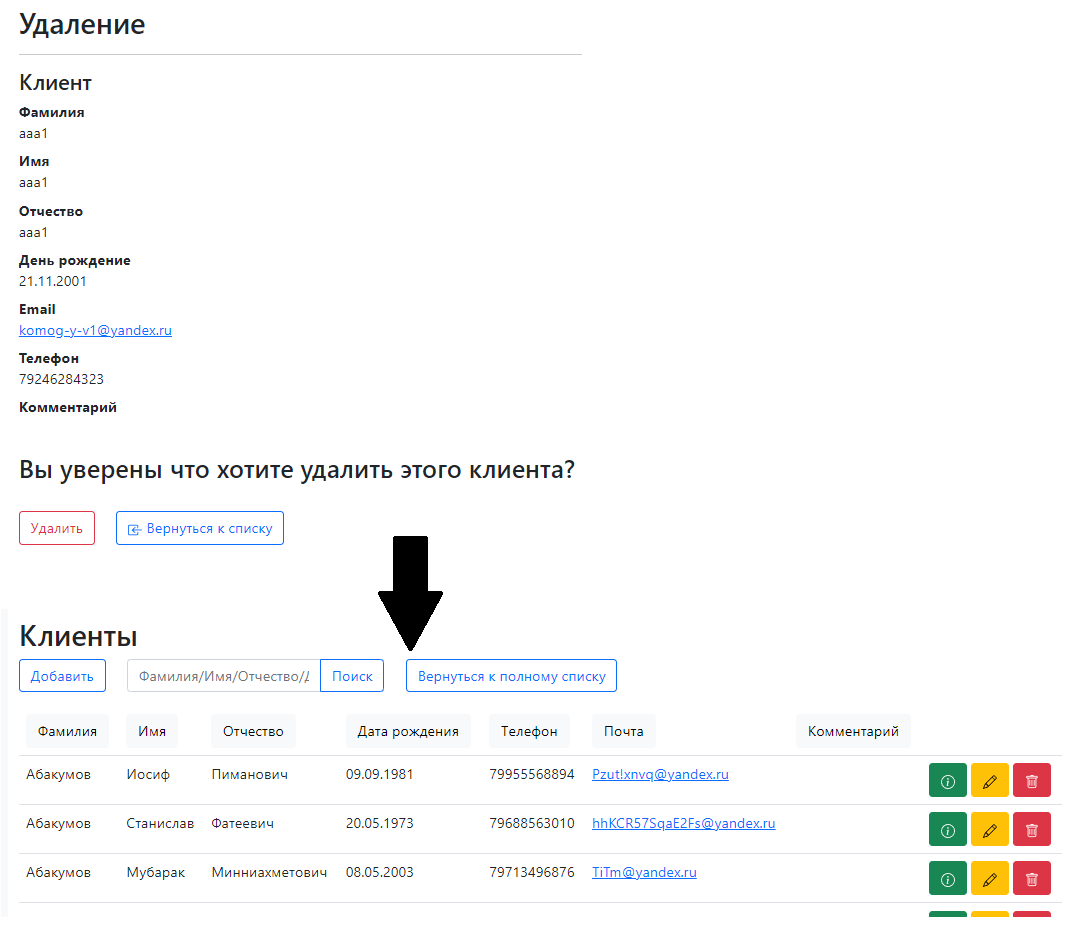


Рисунок 33 – Удаление «Clients»

Результат теста №13

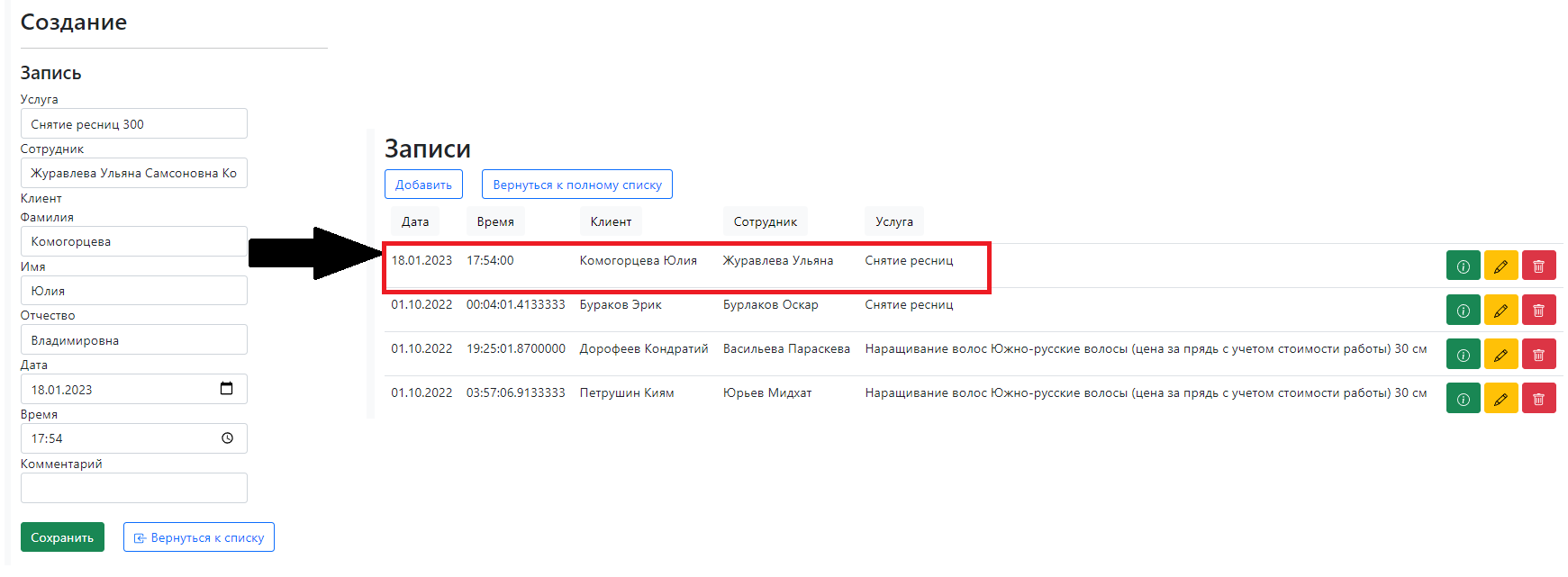


Рисунок 34 – Добавление «Record»

Результат теста №14

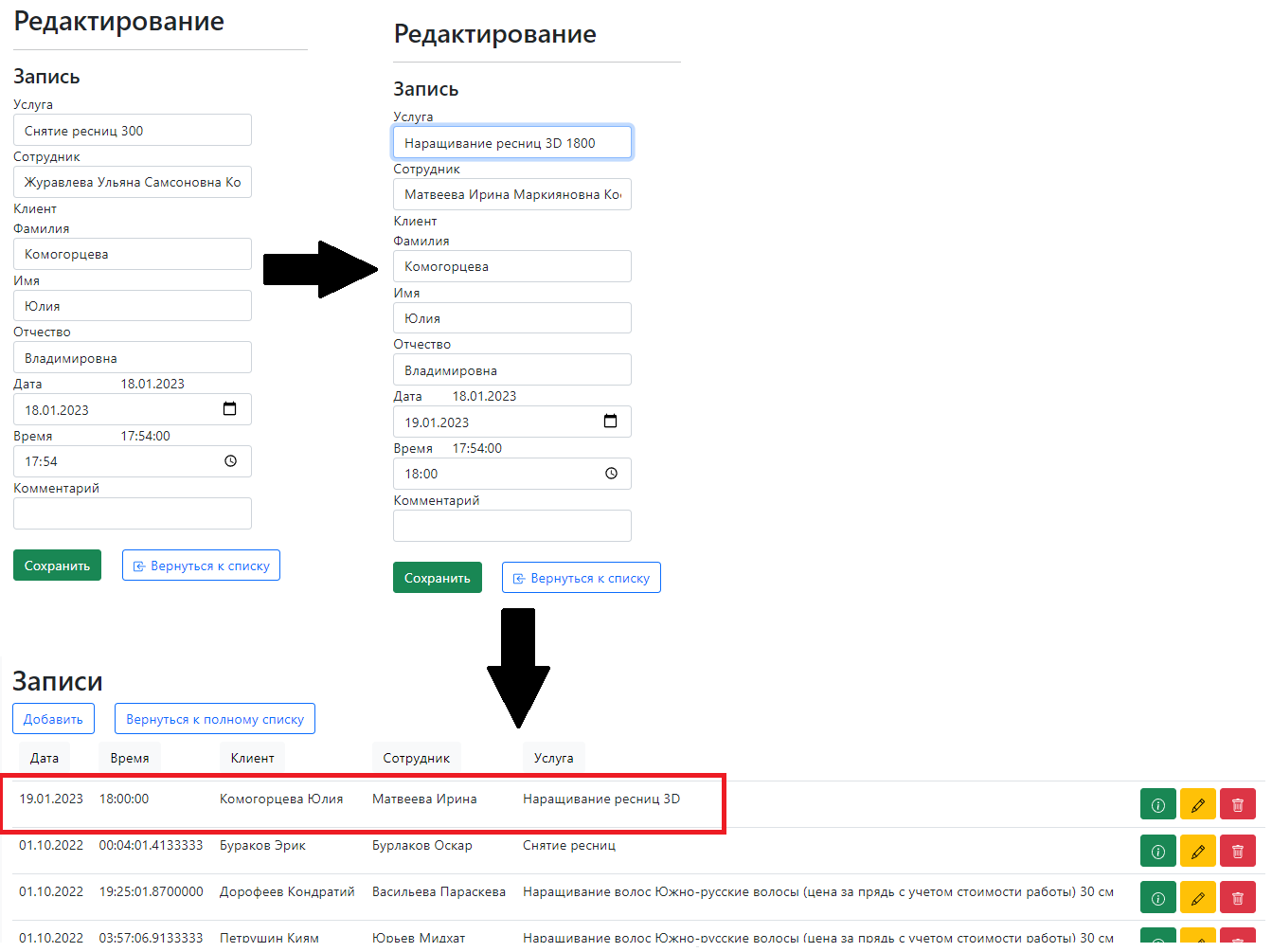


Рисунок 35 – Изменение «Record»

Результат теста №15

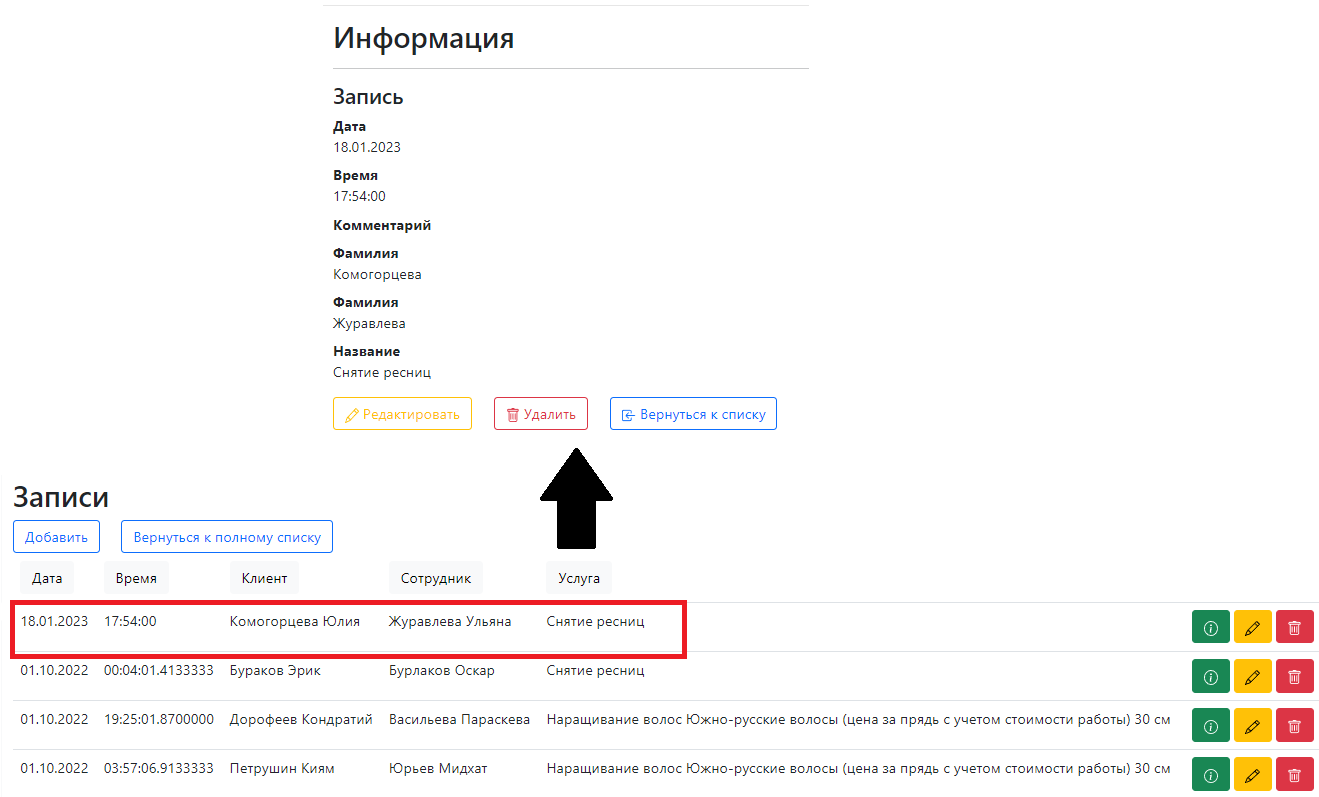


Рисунок 36 –Просмотр «Record»

Результат теста №16

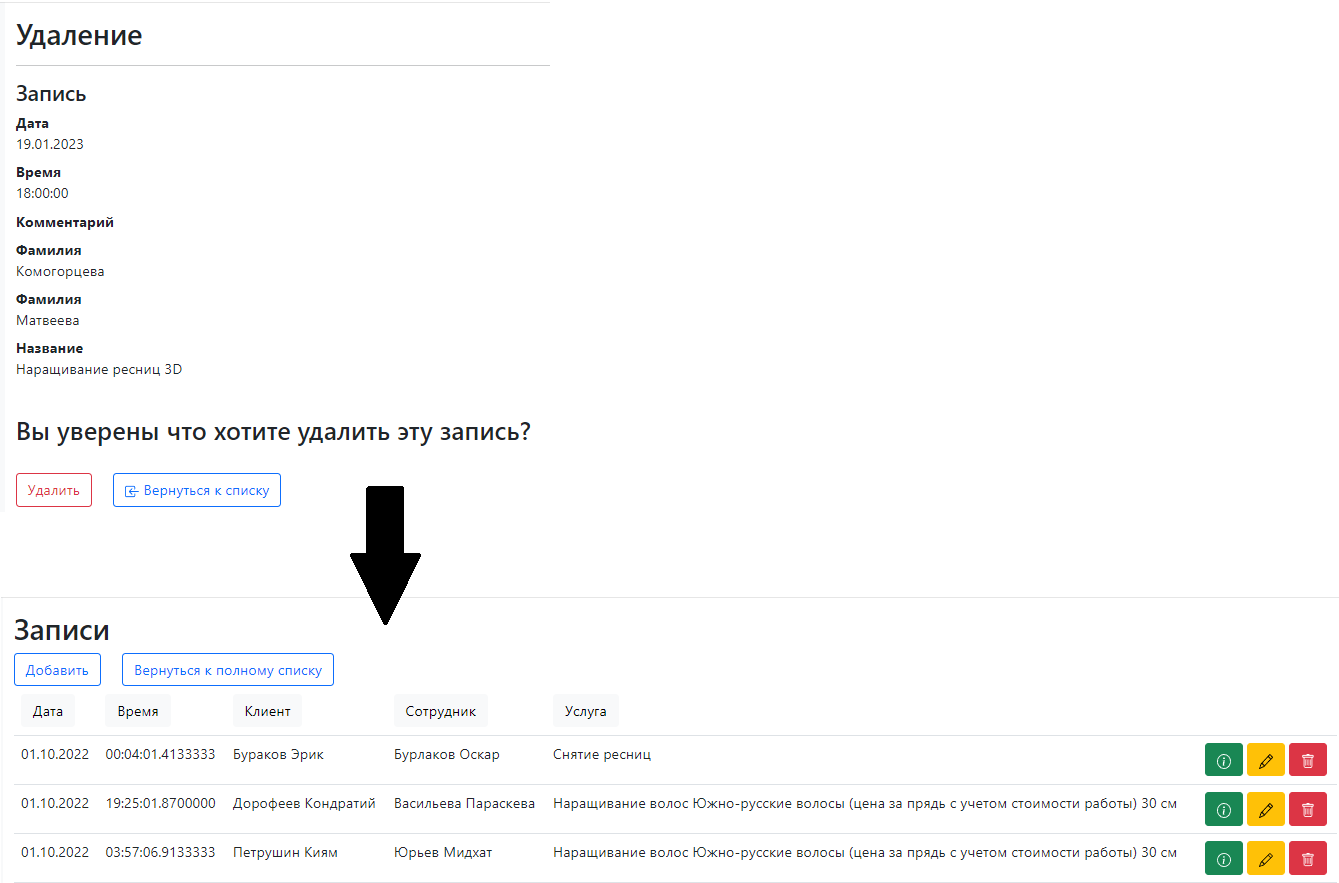


Рисунок 37 – Удаление «Record»

Результат теста №17

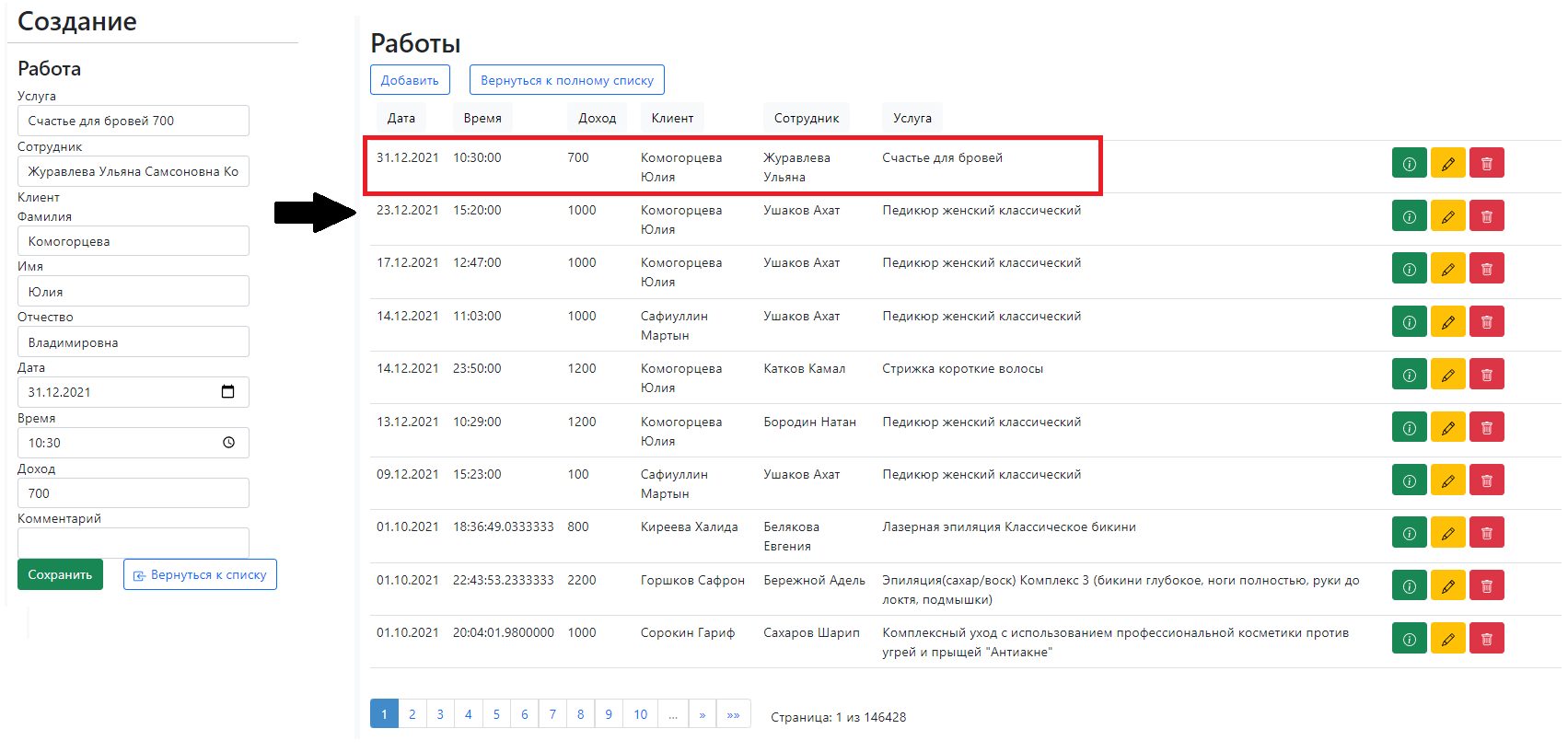


Рисунок 38 – Добавление «Work»

Результат теста №18

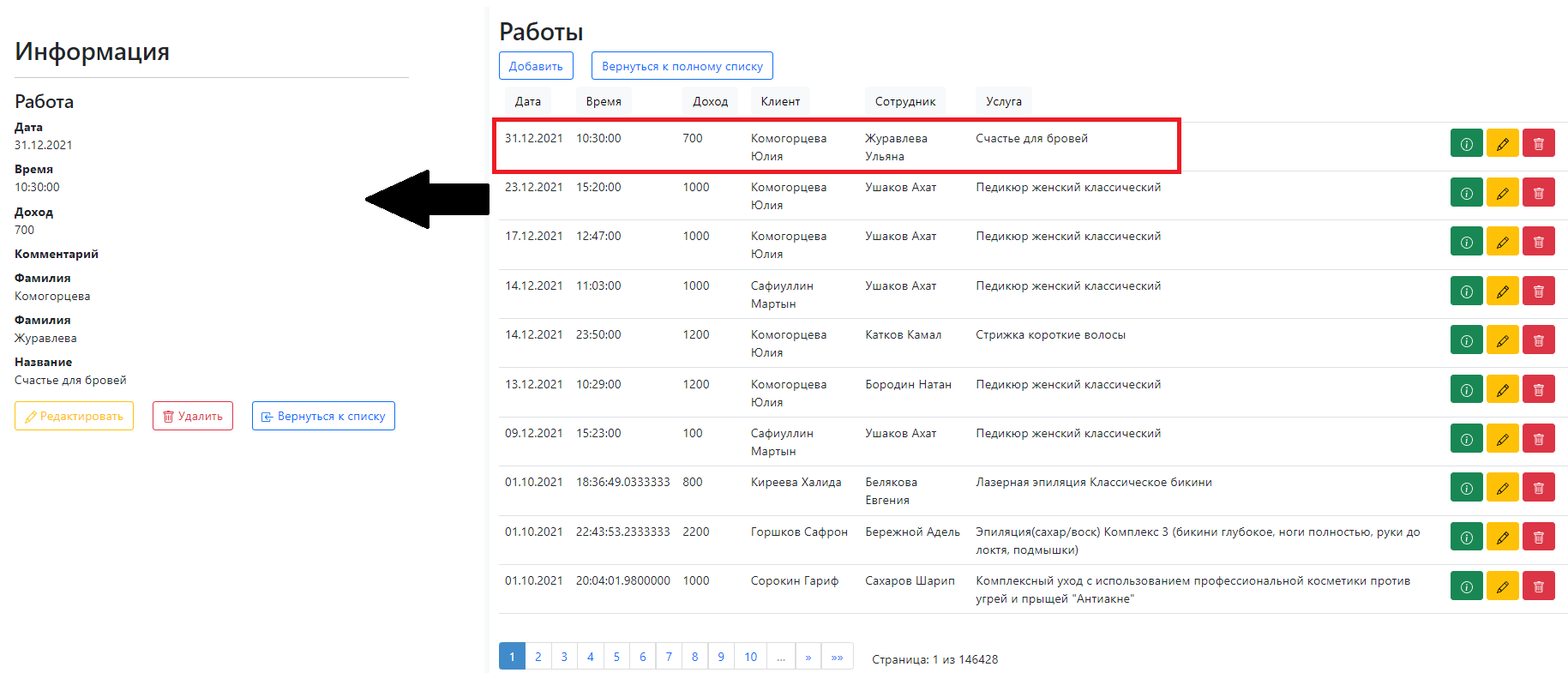


Рисунок 39 – Просмотр «Work»

Результат теста №19

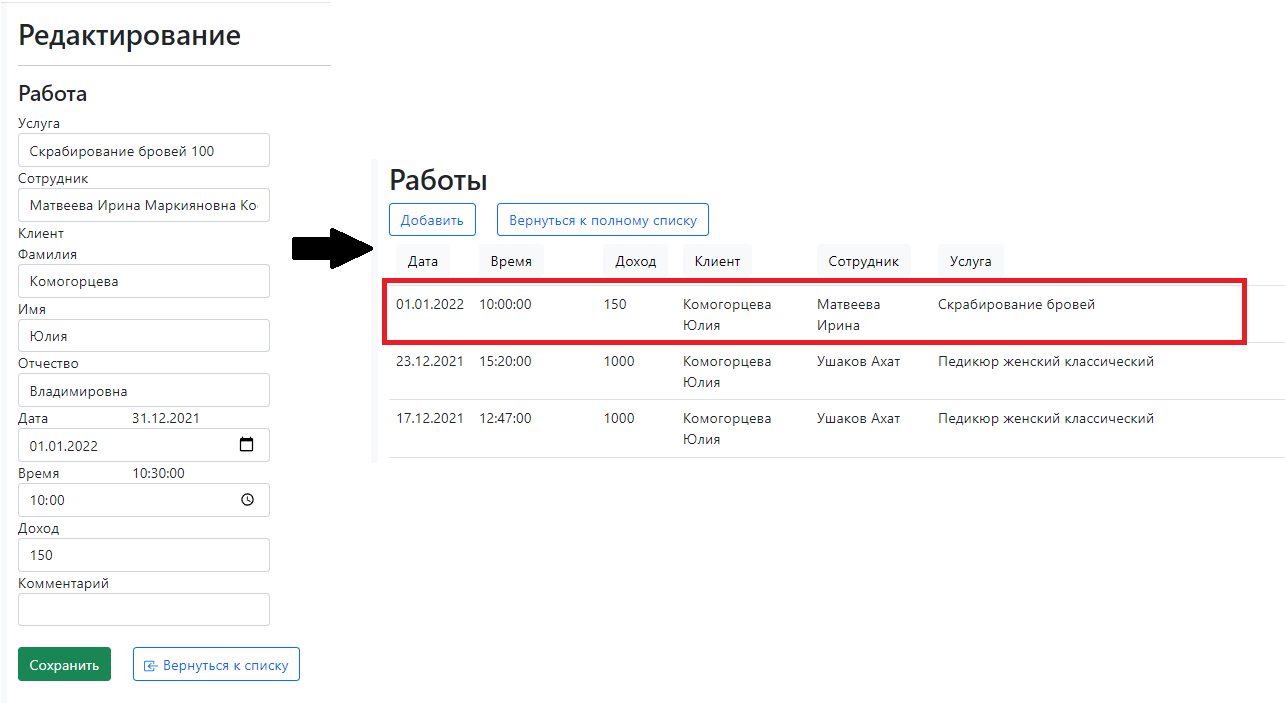


Рисунок 40 – Изменение «Work»

Результат теста №20

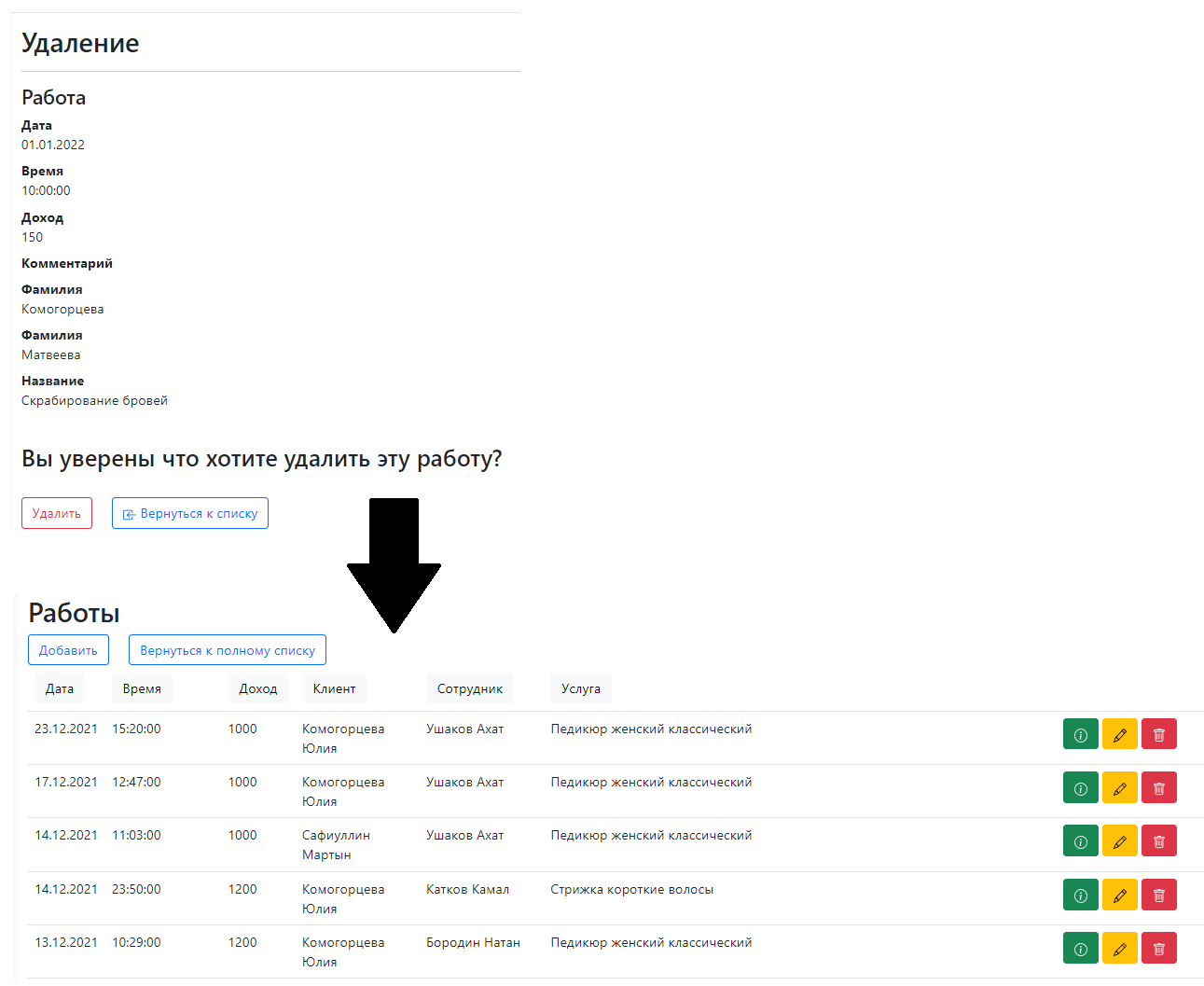


Рисунок 41 –Удаление «Work»

Результат теста №21

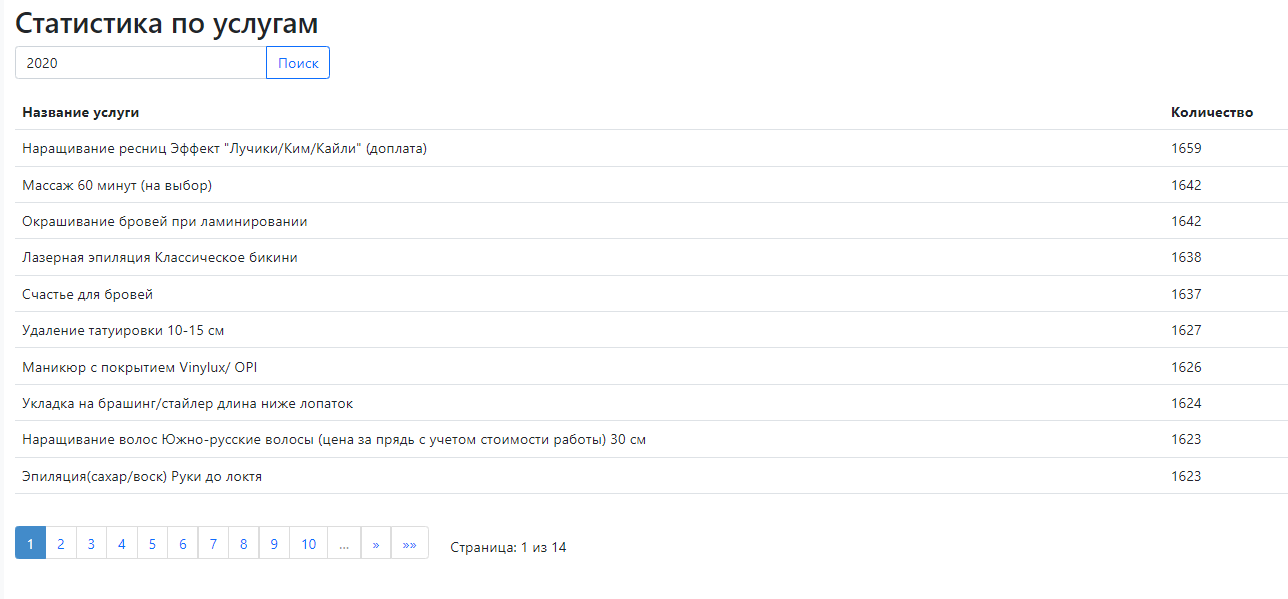


Рисунок 42 – Результат выполнения запроса

Результат теста №22

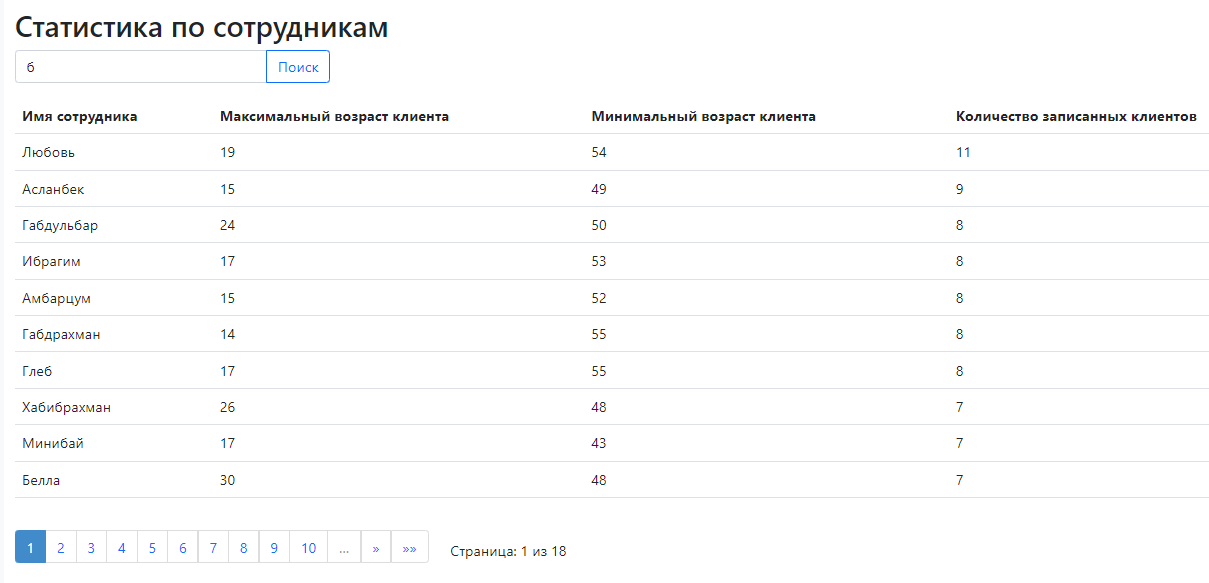


Рисунок 43 – Результат выполнения запроса

Результат теста №23

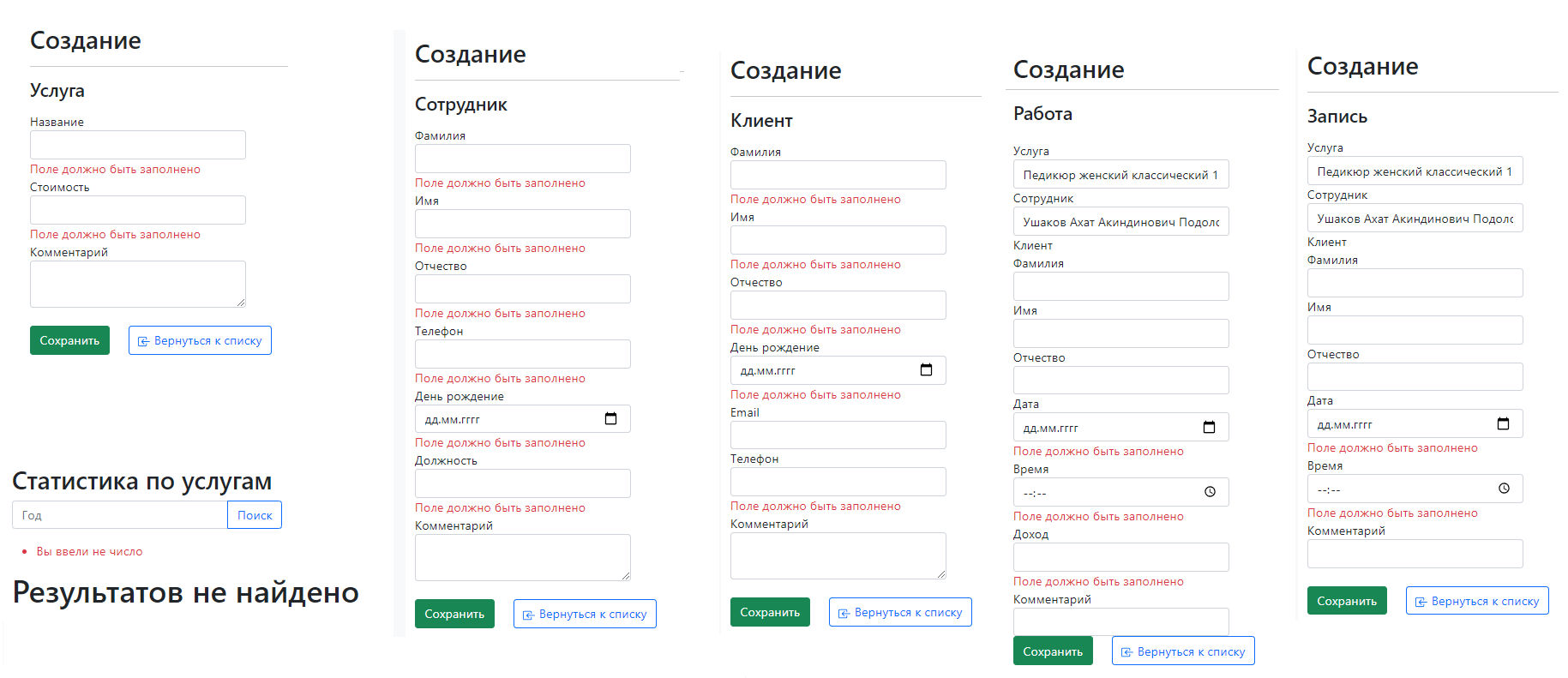


Рисунок 44 – Контроль данных

Заключение

В процессе разработки данного приложения были получены и закреплены следующие навыки:

1. Проектирования и применения паттерна MVC в реализации информационной системы
2. Проектирования пользовательского интерфейса и компоновки элементов взаимодействия с пользователем
3. Создания, проектирования и работы с локальной базы данных с помощью СУБД Microsoft SQL Server
4. Программирование веб-приложения

Были решены следующие задачи:

1. Спроектирована локальная база данных
2. Спроектирована структурная схема функционирования моделей, контроллеров и представлений
3. Развернута модель данных с помощью фреймворка Entity Framework
4. Разработано веб-приложение для автоматизированного взаимодействия с клиентами
5. Тестирование приложения
6. Составлена документация по всей проделанной работе

Исходный код приложения опубликован в публичном репозитории на сервисе GitHub [6].

Цель курсового проекта достигнута.

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. Репозиторий с исходных кодом проекта // GitHub URL: https://github.com/KomogortsevaYulia/CourseWork\_Beauty1.git
7. Введение в ASP.NET MVC // Особенности платформы ASP.NET MVC URL: https://metanit.com/sharp/mvc/1.1.php (дата обращения: 01.01.2021).