Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и

анализа данных

|  |
| --- |
| наименование института |
| **Отчет** по лабораторной работе №1  по дисциплине «Объектно-ориентированные базы данных»  «Databases 4 Objects» | | |  |

наименование темы

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | |
| Выполнил студент |  | ИСМб-19-1 | |  |  |  | Малиновцев И.А. |
| Проверил |  | шифр группы | |  | подпись |  | И.О. Фамилия  В.А. Харахинов |
|  |  |  | |  | подпись |  | И.О. Фамилия |
| Работа защищена с оценкой | | |  | |  | | |

**Содержание**

[**1 Задание 3**](#_Toc119675403)

[**2 Классы 4**](#_Toc119675404)

[**3 Работа программы 34**](#_Toc119675405)

# **1 Задание**

**Магазин**

Предметная область включает в себя Модель, Производителя, Цены, Продукты и Продавцов

Необходимо реализовать CRUD функциональность для данной предметной области.

**Выходные документы:**

* График изменения цены товара у продавца с указанием средней стоимости
* Выдача списка лиц, чей возраст находится в диапазоне от 17 до 65, имеют задолженность выше 10000.

# **2 Классы**

В данной лабораторной работе необходимо было разработать 6 классов: Model, Manufacturer, Price, Product, Saler и DataBase на языке C# c использованием библиотеки db4o.

Код классов представлен ниже.

*Класс Model:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace PriceList.Classes

{

internal class Model

{

[DisplayName("id")]

public string id { get; set; }

[DisplayName("Название")]

public string title { get; set; }

public Model()

{

id = Guid.NewGuid().ToString();

}

public Model(string title)

{

id = Guid.NewGuid().ToString();

this.title = title;

}

public string getInfo()

{

var info = $"id: {id}\nНазвание: {title}";

return info;

}

public override string ToString()

{

return title;

}

}

}

*Класс Manufacturer:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PriceList.Classes

{

internal class Monufacturer

{

[DisplayName("id")]

public string id { get; set; }

[DisplayName("Название")]

public string title { get; set; }

[DisplayName("Страна")]

public string country { get; set; }

[DisplayName("Сайт")]

public string site { get; set; }

public Monufacturer()

{

id = Guid.NewGuid().ToString();

}

public Monufacturer(string title, string country, string site)

{

id = Guid.NewGuid().ToString();

this.title = title;

this.country = country;

this.site = site;

}

public Monufacturer(string id, string title, string country, string site) : this(id, title, country)

{

this.id = id;

this.title = title;

this.country = country;

this.site = site;

}

// override object.Equals

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

else

{

if (id == (obj as Monufacturer).id && title == (obj as Monufacturer).title && country == (obj as Monufacturer).country

&& (obj as Monufacturer).site == site)

{

return true;

}

else

{

return false;

}

}

}

public string getInfo()

{

var info = $"id: {id}\nНазвание: {title}\nСтрана: {country}\nСайт: {site}";

return info;

}

public override string ToString()

{

return title;

}

}

} *Класс Price:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PriceList.Classes

{

internal class Price

{

[DisplayName("id")]

public string id { get; set; }

[DisplayName("Продавец")]

public Saler saler { get; set; }

[DisplayName("Продукт")]

public Product product { get; set; }

[DisplayName("Цена")]

public int price { get; set; }

public DateTime dateTime { get; set; }

public Price()

{

id = Guid.NewGuid().ToString();

this.dateTime = DateTime.Now;

}

public Price(Saler saler, Product product, int price)

{

id = Guid.NewGuid().ToString();

this.saler = saler;

this.product = product;

this.price = price;

this.dateTime = DateTime.Now;

}

public override bool Equals(object obj)

{

return obj is Price price &&

id == price.id &&

(obj as Price).saler.Equals(saler) &&

(obj as Price).product.Equals(product) &&

this.price == price.price;

}

}

}

*Класс Product:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PriceList.Classes

{

internal class Product

{

[DisplayName("id")]

public string id { get; set; }

[DisplayName("Производитель")]

public Monufacturer monufacturer { get; set; }

[DisplayName("Модель")]

public Model model { get; set; }

[DisplayName("Название")]

public string title { get; set; }

public Product()

{

id = Guid.NewGuid().ToString();

}

public Product(Monufacturer monufacturer, Model model, string title)

{

id = Guid.NewGuid().ToString();

this.monufacturer = monufacturer;

this.model = model;

this.title = title;

}

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

if(id == (obj as Product).id && model.Equals((obj as Product).model)

&& monufacturer.Equals((obj as Product).monufacturer)

&& title == (obj as Product).title)

{

return true;

}

else

{

return false;

}

}

public override string ToString()

{

return title;

}

internal string getInfo()

{

var info = $"id: {id}\nНазвание:{title}\nПроизводитель: {monufacturer}\nМодель: {model}";

return info;

}

}

}

*Класс Saler:*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PriceList.Classes

{

internal class Saler

{

[DisplayName("id")]

public string id { get; set; }

[DisplayName("Название")]

public string title { get; set; }

[DisplayName("Адрес")]

public string address { get; set; }

[DisplayName("Телефон")]

public string phone { get; set; }

[DisplayName("Сайт")]

public string site { get; set; }

public Saler()

{

id = Guid.NewGuid().ToString();

}

public Saler(string title, string address, string phone, string site)

{

id = Guid.NewGuid().ToString();

this.title = title;

this.address = address;

this.phone = phone;

this.site = site;

}

public override bool Equals(object obj)

{

if (obj == null || GetType() != obj.GetType())

{

return false;

}

if(this.id == (obj as Saler).id && this.title == (obj as Saler).title

&& this.phone == (obj as Saler).phone

&& this.site == (obj as Saler).site

&& this.address == (obj as Saler).address)

{

return true;

}

else

{

return false;

}

}

public override string ToString()

{

return title;

}

public string getInfo()

{

var info = $"id: {id}\nНазвание {title}\nАдрес: {address}\nТелефон: {phone}\n" +

$"Сайт: {site}";

return info;

}

}

}

*Класс DataBase:*

using Db4objects.Db4o;

using Db4objects.Db4o.Internal.Slots;

using Db4objects.Db4o.Linq;

using Db4objects.Db4o.Query;

using PriceList.Classes;

using Sharpen.Util;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

namespace PriceList

{

internal class DataBase

{

/// <summary>

/// сделать все возможные выводу QueryBeExample+

/// Native(Query)+

/// Soda+

/// Linq+

/// </summary>

const String filename = @"pricelist.yap";

IObjectContainer db;

public void dbConnect()

{

db = Db4oFactory.OpenFile(filename);

db.Ext().Configure().ObjectClass(typeof(Price)).CascadeOnActivate(true);

db.Ext().Configure().ObjectClass(typeof(Product)).CascadeOnActivate(true);

}

public void dbClose()

{

db.Close();

}

#region Работа с моделями

public void addModel(Model model)

{

db.Store(model);

}

public List<Model> getModels()

{

List<Model> list = db.Query<Model>().ToList<Model>();

return list;

}

public List<Model> getModels(string title)

{

List<Model> list = db.Query<Model>(mdl => mdl.title.Contains(title)).ToList<Model>();

return list;

}

public void updateModel(Model model)

{

IEnumerable<Model> result = from Model mdl in db.Cast<Model>()

where mdl.id == model.id

select mdl;

var modl = result.First();

modl.title = model.title;

db.Store(modl);

}

public void deleteModel(string id)

{

try

{

Product product = db.Query<Product>(prd => prd.model.id == id)[0];

db.Delete(product);

}

catch (Exception ex)

{

}

Model model = db.Query<Model>(mdl => mdl.id == id)[0];

db.Delete(model);

}

#endregion

#region Работа с производителем

public void addMonufacturer(Monufacturer monufacturer)

{

db.Store(monufacturer);

}

public List<Monufacturer> GetMonufacturers()

{

IQuery query = db.Query();

query.Constrain(typeof(Monufacturer));

IObjectSet result = query.Execute();

var monufacturers = new List<Monufacturer>();

foreach(var item in result)

{

monufacturers.Add(item as Monufacturer);

}

return monufacturers;

}

public void deleteMonufacturer(string id)

{

try

{

Product product = db.Query<Product>(prd => prd.monufacturer.id == id)[0];

db.Delete(product);

}

catch(Exception ex)

{

}

var Monufacturer = db.Query<Monufacturer>(mdl => mdl.id == id)[0];

db.Delete(Monufacturer);

}

public void updateMonufacturer(Monufacturer monufacturer)

{

var foundMon = db.Query<Monufacturer>(mnf => mnf.id == monufacturer.id)[0];

foundMon.title = monufacturer.title;

foundMon.country = monufacturer.country;

foundMon.site = monufacturer.site;

db.Store(foundMon);

}

#endregion

#region Работа с ценами

public void addPrice(Price priceList)

{

db.Store(priceList);

}

public List<Price> getPrice()

{

return db.Query<Price>().ToList<Price>();

}

internal void deletePrice(string id)

{

var price = db.Query<Price>(prc => prc.id == id)[0];

db.Delete(price);

}

internal void updatePrice(Price price)

{

var pattern = new Price();

pattern.id = price.id;

pattern.saler = null;

pattern.product = null;

pattern.price = 0;

var found = db.QueryByExample(pattern).Next() as Price;

found.price = price.price;

db.Store(found);

}

public bool isPriceClone(int price, Saler saler, Product product)

{

var priceList = new Price();

priceList.id = null;

priceList.price = 0;

priceList.saler = saler;

priceList.product = product;

var found = db.QueryByExample(priceList).Count;

if (found > 0)

{

return true;

}

else

{

return false;

}

}

public List<Price> getPrice(Price price)

{

IEnumerable<Price> priceList = from Price prc in db.Cast<Price>()

where prc.product.id == price.product?.id

&& prc.saler.id == price.saler?.id

select prc;

var list = priceList.ToList();

return list;

}

public List<Product> getPriceProduct()

{

var list = db.Query<Price>().ToList<Price>();

var ans = new List<Product>();

foreach (var p in list)

{

ans.Add(p.product);

}

ans.Distinct().ToList();

return new HashSet<Product>(ans).ToList();

}

public List<Saler> getPriceSaler()

{

var list = db.Query<Price>().ToList<Price>();

var ans = new List<Saler>();

foreach (var p in list)

{

ans.Add(p.saler);

}

return new HashSet<Saler>(ans).ToList();

}

public List<Product> getPriceProduct(string id)

{

var list = db.Query<Price>(prs => prs.saler.id == id).ToList<Price>();

var ans = new List<Product>();

foreach(var p in list)

{

ans.Add(p.product);

}

return new HashSet<Product>(ans).ToList(); ;

}

public List<Saler> getPriceSaler(string id)

{

var list = db.Query<Price>(prs => prs.product.id == id).ToList<Price>();

var ans = new List<Saler>();

foreach (var p in list)

{

ans.Add(p.saler);

}

return new HashSet<Saler>(ans).ToList(); ;

}

public double getAvarangePrice(Saler saler, Product product)

{

var ans = (from Price prc in db.Cast<Price>()

where prc.saler == saler

&& prc.product == product

select prc.price).Average();

return ans;

}

#endregion

#region Работа с продуктом

public void addProduct(Product product)

{

db.Store(product);

}

public List<Product> GetProducts()

{

return db.Query<Product>().ToList<Product>();

}

public List<Product> GetProducts(string id)

{

return db.Query<Product>(prd => prd.id == id).ToList<Product>();

}

internal void deleteProduct(string id)

{

try

{

var price = db.Query<Price>(prs => prs.product.id == id)[0];

db.Delete(price);

}

catch { }

var product = db.Query<Product>(prd => prd.id == id)[0];

db.Delete(product);

}

public bool isProductClone(Model model, Monufacturer monufacturer)

{

var product = new Product();

product.id = null;

product.title = null;

product.model = model;

product.monufacturer = monufacturer;

var found = db.QueryByExample(product).Count;

if (found > 0)

{

return true;

}

else

{

return false;

}

}

public void updateProduct(Product product)

{

var found = db.Query<Product>(prd => prd.id == product.id)[0];

found.title = product.title;

found.model = product.model;

found.monufacturer = product.monufacturer;

db.Store(found);

}

#endregion

#region Работа с Продавцами

public void addSaler(Saler saler)

{

db.Store(saler);

}

public List<Saler> GetSaler()

{

return db.Query<Saler>().ToList<Saler>();

}

public List<Saler> GetSaler(string salerId)

{

return db.Query<Saler>(slr => slr.id == salerId).ToList<Saler>();

}

public void deleteSaler(string id)

{

try

{

Price product = db.Query<Price>(prd => prd.saler.id == id)[0];

db.Delete(product);

}

catch (Exception ex)

{

}

var saler = db.Query<Saler>(slr => slr.id == id)[0];

db.Delete(saler);

}

internal void updateSaler(Saler saler)

{

var found = db.Query<Saler>(slr => slr.id == saler.id)[0];

found.title = saler.title;

found.address = saler.address;

found.phone = saler.phone;

found.site = saler.site;

db.Store(found);

}

#endregion

#region Сода

public void AddSoda(Soda soda)

{

db.Store(soda);

}

public List<Soda> SodaQuery()

{

IQuery query = db.Query();

query.Constrain(typeof(Soda));

IQuery ageQuery = query.Descend("\_age");

IQuery nameQuery = query.Descend("\_name");

IQuery lastNameQuery = query.Descend("\_lastName");

query.Descend("\_money").Constrain(10000).Greater().And(

ageQuery.Constrain(17).Greater().

And(ageQuery.Constrain(65).Smaller())

.And(nameQuery.Constrain("a").StartsWith(true)

.Or(nameQuery.Constrain("u").StartsWith(false)))

.And(lastNameQuery.Constrain("x").EndsWith(false)

.Or(lastNameQuery.Constrain("t").Like())));

IObjectSet result = query.Execute();

var soda = new List<Soda>();

foreach(var r in result)

{

soda.Add(r as Soda);

}

return soda;

}

#endregion

}

}

*Класс Soda:*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PriceList.Classes

{

internal class Soda

{

private int \_age;

//Возраст

public int Age

{

get { return \_age; }

set { \_age = value; }

}

//Имя

private string \_name;

public string Name

{

get { return \_name; }

set { \_name = value; }

}

//Фамилия

private string \_lastName;

public string LastName

{

get { return \_lastName; }

set { \_lastName = value; }

}

//Кол-во денег

private int \_money;

public int MoneyCount

{

get { return \_money; }

set { \_money = value; }

}

public Soda()

{

}

public Soda(int age, string name, string lastName, int moneyCount)

{

Age = age;

Name = name;

LastName = lastName;

MoneyCount = moneyCount;

}

}

}

Также, были разработаны формы: Form1 для общей работы с данными.

Листинг Form1:

using PriceList.Classes;

using System;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http.Headers;

using System.Text.RegularExpressions;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using static System.Net.Mime.MediaTypeNames;

using ComboBox = System.Windows.Forms.ComboBox;

using Random = System.Random;

namespace PriceList

{

public partial class Form1 : Form

{

#region Загрузка и настройка формы

DataBase dataBase;

System.Windows.Forms.ComboBox cmbPriceSalerfound;

System.Windows.Forms.ComboBox cmbPriceProductfound;

private static Random random = new Random();

Chart priceHistory = new Chart();

Series seriesOfprice = new Series("Цена");

string chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789" + "ABCDEFGHIJKLMNOPQRSTUVWXYZ".ToLower();

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

dataBase = new DataBase();

dataBase.dbConnect();

setBindingSource();

dgvSetting();

cmbSetting();

Soda soda;

for (var i = 0; i < 100; i++)

{

soda = new Soda();

soda.Name = generateString(4, 10);

soda.Age = random.Next(1, 100);

soda.LastName = generateString(5, 20);

soda.MoneyCount = random.Next(5000, 100000);

dataBase.AddSoda(soda);

}

var result = dataBase.SodaQuery();

txtProductMonufacturerInfo.Text = (cmbProductMonufacturer.SelectedItem as Monufacturer)?.getInfo();

txtProductModelInfo.Text = (cmbProductModel.SelectedItem as Model)?.getInfo();

txtPriceSalerInfo.Text = (cmbPriceSaler.SelectedItem as Saler)?.getInfo();

txtPticeProductInfo.Text = (cmbPriceProduct.SelectedItem as Product)?.getInfo();

ChartSetting();

labelSerring();

}

public void ChartSetting()

{

priceHistory.Parent = panelChart;

priceHistory.Location = new Point(0, 100);

priceHistory.Width = priceHistory.Parent.Width;

priceHistory.Height = 200;

priceHistory.ChartAreas.Add(new ChartArea("Цена товара"));

seriesOfprice.ChartType = SeriesChartType.FastLine;

seriesOfprice.ChartArea = "Цена товара";

priceHistory.ChartAreas[0].AxisX.MajorGrid.Enabled = false;

priceHistory.ChartAreas[0].AxisY.MajorGrid.Enabled = false;

priceHistory.ChartAreas[0].AxisX2.MajorGrid.Enabled = false;

priceHistory.Legends.Clear();

}

public void dgvSetting()

{

dgvModels.Columns[0].ReadOnly = true;

dgvMonufacturers.Columns[0].ReadOnly = true;

dgvPriceList.Columns[0].ReadOnly = true;

dgvProducts.Columns[0].ReadOnly = true;

dgvSalers.Columns[0].ReadOnly = true;

}

public void labelSerring()

{

Label lblPriceSaler = new Label();

lblPriceSaler.Parent = panelChart;

lblPriceSaler.Text = "Продавец";

lblPriceSaler.Location = new Point(40, 20);

Label lblPriceProduct = new Label();

lblPriceProduct.Parent = panelChart;

lblPriceProduct.Text = "Продукт";

lblPriceProduct.Location = new Point(255, 20);

}

public void cmbSetting()

{

cmbMonufacturerCountry.SelectedIndex = 0;

cmbPriceSalerfound = new System.Windows.Forms.ComboBox();

cmbPriceSalerfound.Parent = panelChart;

cmbPriceSalerfound.Location = new Point(10, 40);

cmbPriceSalerfound.DataSource = dataBase.getPriceSaler();

cmbPriceSalerfound.SelectedIndexChanged += PriceFoundChanged;

cmbPriceSalerfound.SelectedIndex = 0;

cmbPriceSalerfound.DropDownStyle = ComboBoxStyle.DropDownList;

cmbPriceSalerfound.Name = "cmbPriceSalerfound";

cmbPriceProductfound = new System.Windows.Forms.ComboBox();

cmbPriceProductfound.Parent = panelChart;

cmbPriceProductfound.Location = new Point(220, 40);

cmbPriceProductfound.DataSource = dataBase.getPriceProduct();

cmbPriceProductfound.SelectedIndexChanged += PriceFoundChanged;

cmbPriceProductfound.SelectedIndex = 0;

cmbPriceProductfound.DropDownStyle = ComboBoxStyle.DropDownList;

cmbPriceProductfound.Name = "cmbPriceProductfound";

}

public void setBindingSource()

{

foreach (var mon in dataBase.GetMonufacturers())

{

monufacturerBindingSource.Add(mon);

}

foreach (var m in dataBase.getModels())

{

modelBindingSource1.Add(m);

}

foreach (var saler in dataBase.GetSaler())

{

salerBindingSource.Add(saler);

}

foreach (var product in dataBase.GetProducts())

{

productBindingSource.Add(product);

}

foreach (var price in dataBase.getPrice())

{

priceBindingSource.Add(price);

}

}

#endregion

#region Утилиты

void messageBoxSuccessAdd()

{

MessageBox.Show("Запись успешно добавлена",

"Успех", MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

void messageBoxError(string message)

{

MessageBox.Show(message,

"Ошибка", MessageBoxButtons.OK,

MessageBoxIcon.Error);

}

DialogResult messageBoxClickResult(string message)

{

var dialogResult = MessageBox.Show(message,

"Предупреждение",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question);

return dialogResult;

}

string generateString(int from, int to)

{

return new string(Enumerable.Repeat(chars, random.Next(from, to))

.Select(s => s[random.Next(s.Length)]).ToArray());

}

//Отменение изменений

private void tableValidating(object sender, DataGridViewCellValidatingEventArgs e)

{

if (string.IsNullOrEmpty(e.FormattedValue.ToString()))

{

((DataGridView)sender).CancelEdit();

((DataGridView)sender).EndEdit();

}

}

#endregion

#region Работа с моделями

Model selectedModel;

//Удаление строки через встроенную кнопку

private void dgvModels\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

if (dgvModels.Columns[e.ColumnIndex].Index == dgvModels.Columns.Count - 1)

{

if (messageBoxClickResult("Удалить эту запись?") == DialogResult.Yes)

{

var id = dgvModels.Rows[e.RowIndex].Cells[0].Value.ToString();

modelBindingSource1.RemoveAt(e.RowIndex);

dataBase.deleteModel(id);

}

}

}

//Добавление записи

private void btnModelAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtModelTitle.Text))

{

messageBoxError("Вы не ввели название");

return;

}

Model model = new Model(txtModelTitle.Text);

dataBase.addModel(model);

modelBindingSource1.Add(model);

txtModelTitle.Clear();

messageBoxSuccessAdd();

}

//генерация моделей

private void btnModelGenerate\_Click(object sender, EventArgs e)

{

for (int i = 0; i <= nudModelCount.Value; i++)

{

Model randomModel = new Model();

randomModel.title = generateString(5, 15);

modelBindingSource1.Add(randomModel);

dataBase.addModel(randomModel);

}

}

//Конец редатикрования ячейки и изменение данных

private void dgvModels\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

if (selectedModel.title.Equals(dgvModels[1, e.RowIndex].Value.ToString()))

{

return;

}

var dialogResult = messageBoxClickResult("Изменить эту запись?");

if (dialogResult == DialogResult.No)

{

dgvModels[1, e.RowIndex].Value = selectedModel.title;

return;

}

if (dialogResult == DialogResult.Yes)

{

var model = new Model();

model.id = dgvModels.Rows[e.RowIndex].Cells[0].Value.ToString();

model.title = dgvModels.Rows[e.RowIndex].Cells[1].Value.ToString();

dataBase.updateModel(model);

}

}

private void dgvModels\_CellClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

selectedModel = new Model(dgvModels.Rows[e.RowIndex].Cells[1].Value.ToString());

}

private void txtModelfound\_TextChanged(object sender, EventArgs e)

{

modelBindingSource1.Clear();

foreach (var m in dataBase.getModels(txtModelfound.Text))

{

modelBindingSource1.Add(m);

}

}

private void cbModelIsEdit\_CheckedChanged(object sender, EventArgs e)

{

if (cbModelIsEdit.Checked)

{

dgvModels.Columns[1].ReadOnly = false;

dgvModels.Columns[2].Visible = true;

}

else

{

dgvModels.Columns[1].ReadOnly = true;

dgvModels.Columns[2].Visible = false;

}

}

#endregion

#region Работа с производителем

private void dgvMonufacturers\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

if (dgvMonufacturers.Columns[e.ColumnIndex].Index == dgvMonufacturers.Columns.Count - 1)

{

if (messageBoxClickResult("Удалить эту запись?") == DialogResult.Yes)

{

var id = dgvMonufacturers.Rows[e.RowIndex].Cells[0].Value.ToString();

monufacturerBindingSource.RemoveAt(e.RowIndex);

dataBase.deleteMonufacturer(id);

}

}

}

Monufacturer selectedMonufacturer;

private void dgvMonufacturers\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

Monufacturer monufacturer = new Monufacturer();

monufacturer.id = dgvMonufacturers.Rows[e.RowIndex].Cells[0].Value.ToString();

monufacturer.title = dgvMonufacturers.Rows[e.RowIndex].Cells[1].Value.ToString();

monufacturer.country = dgvMonufacturers.Rows[e.RowIndex].Cells[2].Value.ToString();

monufacturer.site = dgvMonufacturers.Rows[e.RowIndex].Cells[3].Value.ToString();

if (selectedMonufacturer.Equals(monufacturer))

{

return;

}

if (selectedMonufacturer.site != monufacturer.site)

{

Regex regex = new Regex(@"\w+[.]\w+");

MatchCollection matches = regex.Matches(monufacturer.site);

if (matches.Count == 0)

{

messageBoxError("Не верный формат сайта");

dgvMonufacturers[3, e.RowIndex].Value = selectedMonufacturer.site;

return;

}

}

var dialogResult = messageBoxClickResult("Изменить эту запись?");

if (dialogResult == DialogResult.No)

{

dgvMonufacturers[1, e.RowIndex].Value = selectedMonufacturer.title;

dgvMonufacturers[2, e.RowIndex].Value = selectedMonufacturer.country;

dgvMonufacturers[3, e.RowIndex].Value = selectedMonufacturer.site;

return;

}

if (dialogResult == DialogResult.Yes)

{

dataBase.updateMonufacturer(dgvMonufacturers.CurrentRow.DataBoundItem as Monufacturer);

}

}

private void dgvMonufacturers\_CellBeginEdit(object sender, DataGridViewCellCancelEventArgs e)

{

selectedMonufacturer = new Monufacturer();

selectedMonufacturer.id = dgvMonufacturers.Rows[e.RowIndex].Cells[0].Value.ToString();

selectedMonufacturer.title = dgvMonufacturers.Rows[e.RowIndex].Cells[1].Value.ToString();

selectedMonufacturer.country = dgvMonufacturers.Rows[e.RowIndex].Cells[2].Value.ToString();

selectedMonufacturer.site = dgvMonufacturers.Rows[e.RowIndex].Cells[3].Value.ToString();

}

private void cbMonufacturerIsEdit\_CheckedChanged(object sender, EventArgs e)

{

if (cbMonufacturerIsEdit.Checked)

{

dgvMonufacturers.Columns[1].ReadOnly = false;

dgvMonufacturers.Columns[2].ReadOnly = false;

dgvMonufacturers.Columns[3].ReadOnly = false;

dgvMonufacturers.Columns[4].Visible = true;

}

else

{

dgvMonufacturers.Columns[1].ReadOnly = true;

dgvMonufacturers.Columns[2].ReadOnly = true;

dgvMonufacturers.Columns[3].ReadOnly = true;

dgvMonufacturers.Columns[4].Visible = false;

}

}

private void btnMonufacturerAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtMonufacturerTitle.Text))

{

messageBoxError("Вы не ввели название");

return;

}

Regex regex = new Regex(@"\w+[.]\w+");

MatchCollection matches = regex.Matches(txtMonufacturerSite.Text);

if (matches.Count > 0)

{

var monufacturer = new Monufacturer(txtMonufacturerTitle.Text,

cmbMonufacturerCountry.Text,

txtMonufacturerSite.Text);

dataBase.addMonufacturer(monufacturer);

monufacturerBindingSource.Add(monufacturer);

txtMonufacturerTitle.Clear();

txtMonufacturerSite.Clear();

messageBoxSuccessAdd();

}

else

{

messageBoxError("Не верный формат сайта");

}

}

string[] countries = new string[]

{

"Россия",

"США",

"Великобритания",

"Китай",

"Япония",

"Нидерланды",

"Германия",

"Франция"

};

private void btnMonufacturerGenerate\_Click(object sender, EventArgs e)

{

for (int i = 0; i < nudMonufacturerCount.Value; i++)

{

Monufacturer monufacturer = new Monufacturer();

monufacturer.title = generateString(5, 15);

monufacturer.country = countries[random.Next(0, countries.Length)];

monufacturer.site = generateString(10, 20) + "." + generateString(2, 5);

dataBase.addMonufacturer(monufacturer);

monufacturerBindingSource.Add(monufacturer);

}

}

#endregion

#region Работа с продавцами

private void cbSalerIsEdit\_CheckedChanged(object sender, EventArgs e)

{

if (cbSalerIsEdit.Checked == true)

{

dgvSalers.Columns[1].ReadOnly = false;

dgvSalers.Columns[2].ReadOnly = false;

dgvSalers.Columns[3].ReadOnly = false;

dgvSalers.Columns[4].ReadOnly = false;

dgvSalers.Columns[5].Visible = true;

}

else

{

dgvSalers.Columns[1].ReadOnly = true;

dgvSalers.Columns[2].ReadOnly = true;

dgvSalers.Columns[3].ReadOnly = true;

dgvSalers.Columns[4].ReadOnly = true;

dgvSalers.Columns[5].Visible = false;

}

}

private void dgvSalers\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

if (dgvSalers.Columns[e.ColumnIndex].Index == dgvSalers.Columns.Count - 1)

{

if (messageBoxClickResult("Удалить эту запись?") == DialogResult.Yes)

{

var id = dgvSalers.Rows[e.RowIndex].Cells[0].Value.ToString();

salerBindingSource.RemoveAt(e.RowIndex);

dataBase.deleteSaler(id);

}

}

}

Saler selectedSaler;

private void dgvSalers\_CellBeginEdit(object sender, DataGridViewCellCancelEventArgs e)

{

selectedSaler = new Saler();

selectedSaler.id = dgvSalers.Rows[e.RowIndex].Cells[0].Value.ToString();

selectedSaler.title = dgvSalers.Rows[e.RowIndex].Cells[1].Value.ToString();

selectedSaler.address = dgvSalers.Rows[e.RowIndex].Cells[2].Value.ToString();

selectedSaler.phone = dgvSalers.Rows[e.RowIndex].Cells[3].Value.ToString();

selectedSaler.site = dgvSalers.Rows[e.RowIndex].Cells[4].Value.ToString();

}

private void dgvSalers\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

Saler saler = new Saler();

saler.id = dgvSalers.Rows[e.RowIndex].Cells[0].Value.ToString();

saler.title = dgvSalers.Rows[e.RowIndex].Cells[1].Value.ToString();

saler.address = dgvSalers.Rows[e.RowIndex].Cells[2].Value.ToString();

saler.phone = dgvSalers.Rows[e.RowIndex].Cells[3].Value.ToString();

saler.site = dgvSalers.Rows[e.RowIndex].Cells[4].Value.ToString();

if (selectedSaler.Equals(saler))

{

return;

}

if (selectedSaler.site != saler.site)

{

Regex regex = new Regex(@"\w+[.]\w+");

MatchCollection matches = regex.Matches(saler.site);

if (matches.Count == 0)

{

messageBoxError("Не верный формат сайта");

dgvSalers[4, e.RowIndex].Value = selectedSaler.site;

return;

}

}

if (selectedSaler.phone != saler.phone)

{

Regex phoneRegex = new Regex(@"^[+]{0,1}[7-8]{1}[(]9{1}\d{2}[)]\d{3}[-]\d{2}[-]\d{2}$");

MatchCollection matches = phoneRegex.Matches(saler.phone);

if (matches.Count == 0)

{

messageBoxError("Не верный формат телефона");

dgvSalers[3, e.RowIndex].Value = selectedSaler.phone;

return;

}

}

var dialogResult = messageBoxClickResult("Изменить эту запись?");

if (dialogResult == DialogResult.No)

{

dgvSalers[1, e.RowIndex].Value = selectedSaler.title;

dgvSalers[2, e.RowIndex].Value = selectedSaler.address;

dgvSalers[3, e.RowIndex].Value = selectedSaler.phone;

dgvSalers[4, e.RowIndex].Value = selectedSaler.site;

return;

}

if (dialogResult == DialogResult.Yes)

{

dataBase.updateSaler(dgvSalers.CurrentRow.DataBoundItem as Saler);

}

}

private void btnSalerAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtSalerTitle.Text))

{

messageBoxError("Вы не ввели название");

return;

}

if (string.IsNullOrEmpty(txtSalerAddress.Text))

{

messageBoxError("Вы не ввели адрес");

return;

}

Regex siteRegex = new Regex(@"\w+[.]\w+");

Regex phoneRegex = new Regex(@"^[+]{0,1}[7-8]{1}[(]9{1}\d{2}[)]\d{3}[-]\d{2}[-]\d{2}$");

MatchCollection matches = phoneRegex.Matches(txtSalerPhone.Text);

if (matches.Count == 0)

{

messageBoxError("Не верный формат телефона");

return;

}

matches = siteRegex.Matches(txtSalerSite.Text);

if (matches.Count == 0)

{

messageBoxError("Не верный формат сайта");

return;

}

Saler saler = new Saler();

saler.phone = txtSalerPhone.Text;

saler.address = txtSalerAddress.Text;

saler.title = txtSalerTitle.Text;

saler.site = txtSalerSite.Text;

dataBase.addSaler(saler);

salerBindingSource.Add(saler);

messageBoxSuccessAdd();

txtSalerPhone.Clear();

txtSalerAddress.Clear();

txtSalerTitle.Clear();

txtSalerSite.Clear();

}

private void txtSalerPhone\_TextChanged(object sender, EventArgs e)

{

//Добавление первой скобки

if (txtSalerPhone.Text.Length == 1 && (txtSalerPhone.Text[0] != '+'))

{

txtSalerPhone.Text += "(9";

txtSalerPhone.SelectionStart = 3;

}

else if (txtSalerPhone.Text.Length == 2 && txtSalerPhone.Text[1] == '7')

{

txtSalerPhone.Text += "(9";

txtSalerPhone.SelectionStart = 4;

}

//Добавление второй скобки

if (txtSalerPhone.Text.Length == 5 && (txtSalerPhone.Text[0] != '+'))

{

txtSalerPhone.Text += ")";

txtSalerPhone.SelectionStart = 6;

}

else if (txtSalerPhone.Text.Length == 6 && txtSalerPhone.Text[0] == '+')

{

txtSalerPhone.Text += ")";

txtSalerPhone.SelectionStart = 7;

}

//Добавление -

if (txtSalerPhone.Text.Length == 9 && (txtSalerPhone.Text[0] != '+'))

{

txtSalerPhone.Text += "-";

txtSalerPhone.SelectionStart = 10;

}

else if (txtSalerPhone.Text.Length == 10 && txtSalerPhone.Text[0] == '+')

{

txtSalerPhone.Text += "-";

txtSalerPhone.SelectionStart = 11;

}

//Добавление -

if (txtSalerPhone.Text.Length == 12 && (txtSalerPhone.Text[0] != '+'))

{

txtSalerPhone.Text += "-";

txtSalerPhone.SelectionStart = 13;

}

else if (txtSalerPhone.Text.Length == 13 && txtSalerPhone.Text[0] == '+')

{

txtSalerPhone.Text += "-";

txtSalerPhone.SelectionStart = 14;

}

}

private void btnSalerGenerate\_Click(object sender, EventArgs e)

{

for (int i = 0; i < nudSalerCount.Value; i++)

{

Saler saler = new Saler();

saler.title = generateString(5, 15);

saler.site = generateString(10, 20) + "." + generateString(2, 5);

saler.address = generateString(7, 16) + " srt";

saler.phone = "7(9" + random.Next(0, 9) + random.Next(0, 9) + ")" +

random.Next(0, 9) + random.Next(0, 9) + random.Next(0, 9) + "-" +

random.Next(0, 9) + random.Next(0, 9) + "-" + random.Next(0, 9) + random.Next(0, 9);

salerBindingSource.Add(saler);

dataBase.addSaler(saler);

}

}

#endregion

#region Работа с продуктами

private void cbProductIsEdit\_CheckedChanged(object sender, EventArgs e)

{

if (cbProductIsEdit.Checked)

{

dgvProducts.Columns[1].ReadOnly = false;

dgvProducts.Columns[4].Visible = true;

}

else

{

dgvProducts.Columns[1].ReadOnly = true;

dgvProducts.Columns[4].Visible = false;

}

}

private void dgvProducts\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

if (dgvProducts.Columns[e.ColumnIndex].Index == dgvProducts.Columns.Count - 1)

{

if (messageBoxClickResult("Удалить эту запись?") == DialogResult.Yes)

{

var id = dgvProducts.Rows[e.RowIndex].Cells[0].Value.ToString();

productBindingSource.RemoveAt(e.RowIndex);

dataBase.deleteProduct(id);

}

}

}

private void cmbProductMonufacturer\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

txtProductMonufacturerInfo.Text = (cmbProductMonufacturer.SelectedItem as Monufacturer)?.getInfo();

}

catch (NullReferenceException) { }

}

private void cmbProductModel\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

txtProductModelInfo.Text = (cmbProductModel.SelectedItem as Model)?.getInfo();

}

catch (NullReferenceException) { }

}

private void btnProductAdd\_Click(object sender, EventArgs e)

{

if (string.IsNullOrEmpty(txtProductTitle.Text))

{

messageBoxError("Вы не ввели название");

return;

}

var model = cmbProductModel.SelectedItem as Model;

var monufacturer = cmbProductMonufacturer.SelectedItem as Monufacturer;

if (dataBase.isProductClone(model, monufacturer))

{

messageBoxError("Такой продукт уже существует");

return;

}

var product = new Product();

product.title = txtProductTitle.Text;

product.model = model;

product.monufacturer = monufacturer;

productBindingSource.Add(product);

dataBase.addProduct(product);

txtProductTitle.Clear();

messageBoxSuccessAdd();

}

Product selectedProduct;

private void dgvProducts\_CellBeginEdit(object sender, DataGridViewCellCancelEventArgs e)

{

selectedProduct = new Product();

selectedProduct.id = dgvProducts.Rows[e.RowIndex].Cells[0].Value.ToString();

selectedProduct.title = dgvProducts.Rows[e.RowIndex].Cells[1].Value.ToString();

selectedProduct.monufacturer = dgvProducts.Rows[e.RowIndex].Cells[2].Value as Monufacturer;

selectedProduct.model = dgvProducts.Rows[e.RowIndex].Cells[3].Value as Model;

}

private void dgvProducts\_CellEndEdit(object sender, DataGridViewCellEventArgs e)

{

var monufacturer = dgvProducts.Rows[e.RowIndex].Cells[2].Value;

var model = dgvProducts.Rows[e.RowIndex].Cells[3].Value;

Product product = new Product();

product.id = dgvProducts.Rows[e.RowIndex].Cells[0].Value.ToString();

product.title = dgvProducts.Rows[e.RowIndex].Cells[1].Value.ToString();

product.monufacturer = monufacturer as Monufacturer;

product.model = model as Model;

if (selectedProduct.Equals(product))

{

return;

}

var dialogResult = messageBoxClickResult("Изменить эту запись?");

if (dialogResult == DialogResult.No)

{

dgvProducts[1, e.RowIndex].Value = selectedProduct.title;

return;

}

if (dialogResult == DialogResult.Yes)

{

dataBase.updateProduct(dgvProducts.CurrentRow.DataBoundItem as Product);

}

}

#endregion

#region Работа с ценами

private void dgvPriceList\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

if (e.RowIndex < 0)

{

return;

}

if (dgvPriceList.Columns[e.ColumnIndex].Index == dgvPriceList.Columns.Count - 1)

{

if (messageBoxClickResult("Удалить эту запись?") == DialogResult.Yes)

{

var id = dgvPriceList.Rows[e.RowIndex].Cells[0].Value.ToString();

priceBindingSource.RemoveAt(e.RowIndex);

dataBase.deletePrice(id);

}

}

}

private void cmbPriceSaler\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

txtPriceSalerInfo.Text = (cmbPriceSaler.SelectedItem as Saler)?.getInfo();

}

catch (Exception) { }

}

private void cmbSalerProduct\_SelectedIndexChanged(object sender, EventArgs e)

{

try

{

txtPticeProductInfo.Text = (cmbPriceProduct.SelectedItem as Product)?.getInfo();

}

catch (Exception) { }

}

private void btnPriceAdd\_Click(object sender, EventArgs e)

{

var product = cmbPriceProduct.SelectedItem as Product;

var saler = cmbPriceSaler.SelectedItem as Saler;

var price = new Price();

price.price = (int)nudPriceCost.Value;

price.saler = saler;

price.product = product;

priceBindingSource.Add(price);

dataBase.addPrice(price);

txtProductTitle.Clear();

messageBoxSuccessAdd();

}

private void cbPriceIsDelelte\_CheckedChanged(object sender, EventArgs e)

{

if (cbPriceIsDelelte.Checked)

{

dgvPriceList.Columns[5].Visible = true;

}

else

{

dgvPriceList.Columns[5].Visible = false;

}

}

private void PriceFoundChanged(object sender, EventArgs e)

{

if (!isClear && isSecondClear >= 2)

{

if ((sender as ComboBox).Name == "cmbPriceProductfound")

{

cmbPriceSalerfound.DataSource = dataBase.getPriceSaler((cmbPriceProductfound.SelectedItem as Product)?.id);

seriesOfprice.Points.Clear();

priceHistory.Series.Clear();

var product = cmbPriceProductfound.SelectedItem as Product;

var saler = cmbPriceSalerfound.SelectedItem as Saler;

var price = new Price();

price.price = 0;

price.saler = saler;

price.product = product;

var priceList = dataBase.getPrice(price);

var count = priceList.Count;

foreach (var p in priceList)

{

seriesOfprice.Points.AddXY(p.dateTime, p.price);

}

priceHistory.Series.Add(seriesOfprice);

txtAvarangePrice.Text = $@"Средняя цена: {dataBase.getAvarangePrice(saler,product)}";

}

else if ((sender as ComboBox).Name == "cmbPriceSalerfound")

{

cmbPriceProductfound.DataSource = dataBase.getPriceProduct((cmbPriceSalerfound.SelectedItem as Saler)?.id);

seriesOfprice.Points.Clear();

priceHistory.Series.Clear();

var product = cmbPriceProductfound.SelectedItem as Product;

var saler = cmbPriceSalerfound.SelectedItem as Saler;

var price = new Price();

price.price = 0;

price.saler = saler;

price.product = product;

var priceList = dataBase.getPrice(price);

foreach (var p in priceList)

{

seriesOfprice.Points.AddXY(p.dateTime, p.price);

}

priceHistory.Series.Add(seriesOfprice);

txtAvarangePrice.Text = $@"Средняя цена: {dataBase.getAvarangePrice(saler, product)}";

}

}

isClear = false;

isSecondClear++;

}

bool isClear = false;

int isSecondClear = 2;

private void btnPriceClear\_Click(object sender, EventArgs e)

{

isClear = true;

isSecondClear = 0;

cmbPriceSalerfound.DataSource = dataBase.getPriceSaler();

cmbPriceProductfound.DataSource = dataBase.getPriceProduct();

seriesOfprice.Points.Clear();

priceHistory.Series.Clear();

}

#endregion

}

}

3 Работа программы

Ниже представлен внешний вид работающей программы (рис. 1 – 7).

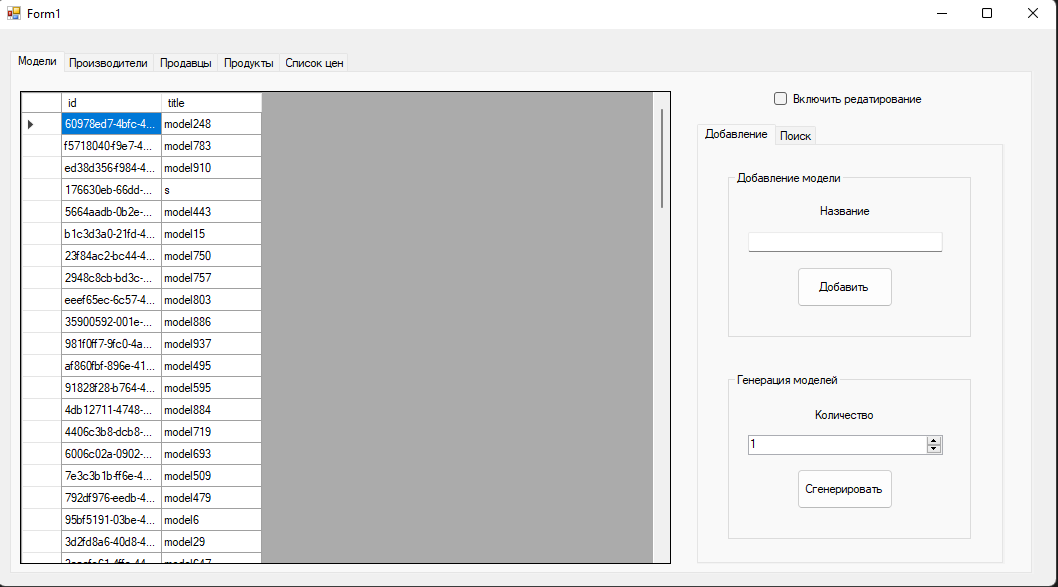


Рисунок 1 – Вкладка «Модели»

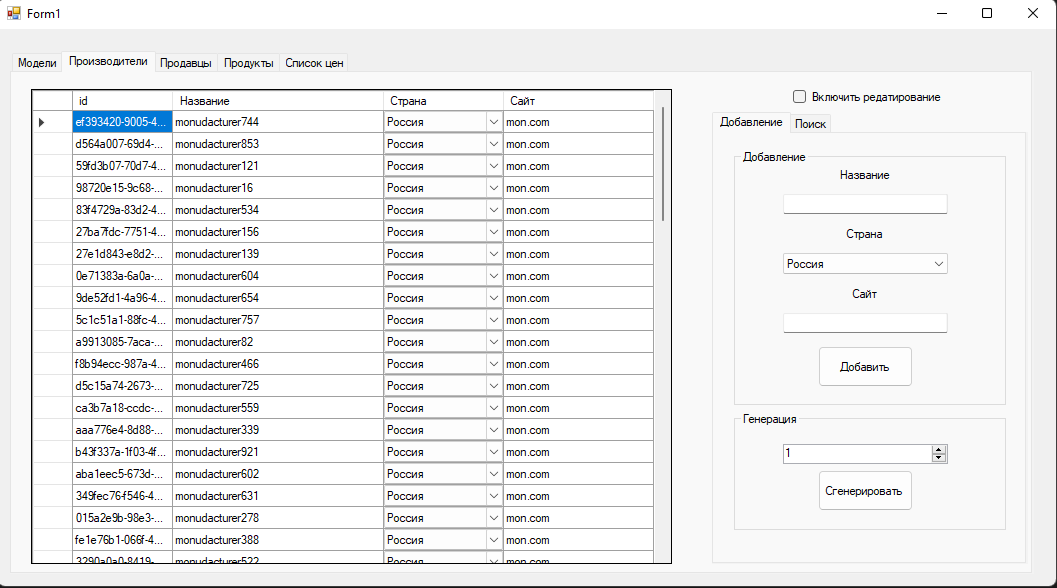


Рисунок 2 – Вкладка «Производители»

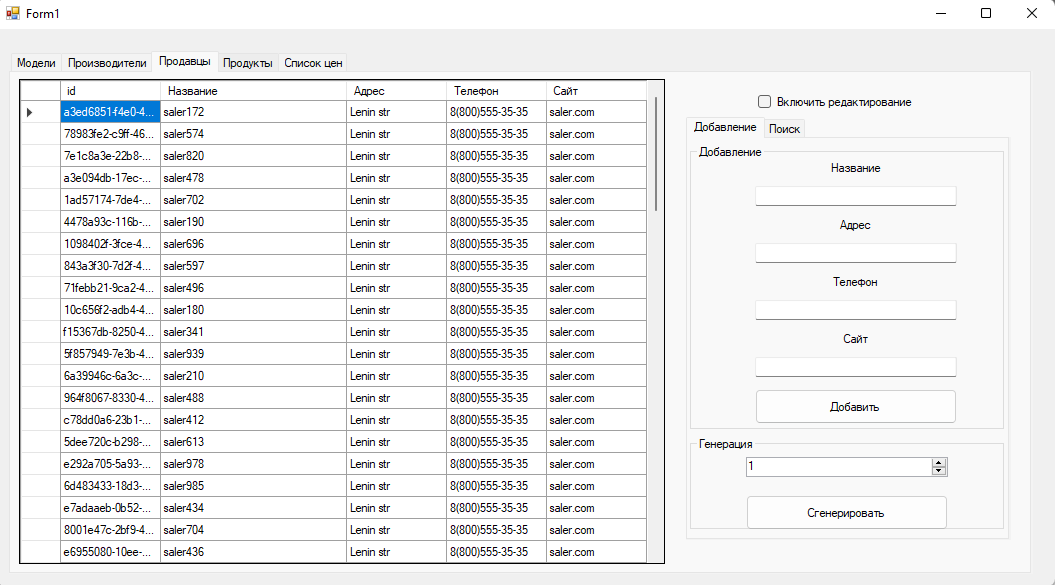


Рисунок 3 – Вкладка «Продавцы»

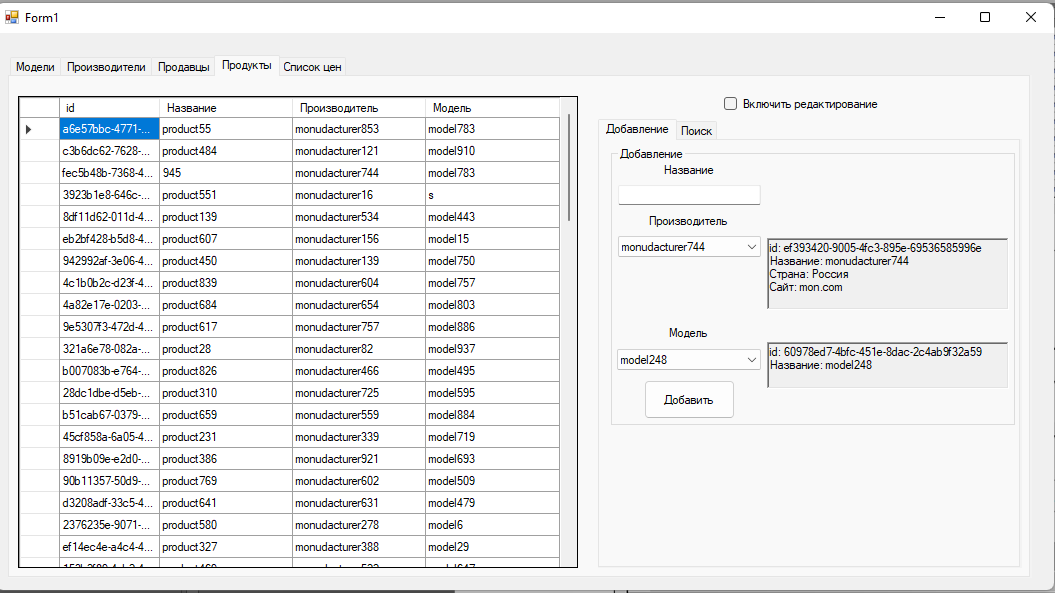


Рисунок 4 – Вкладка «Продукты»

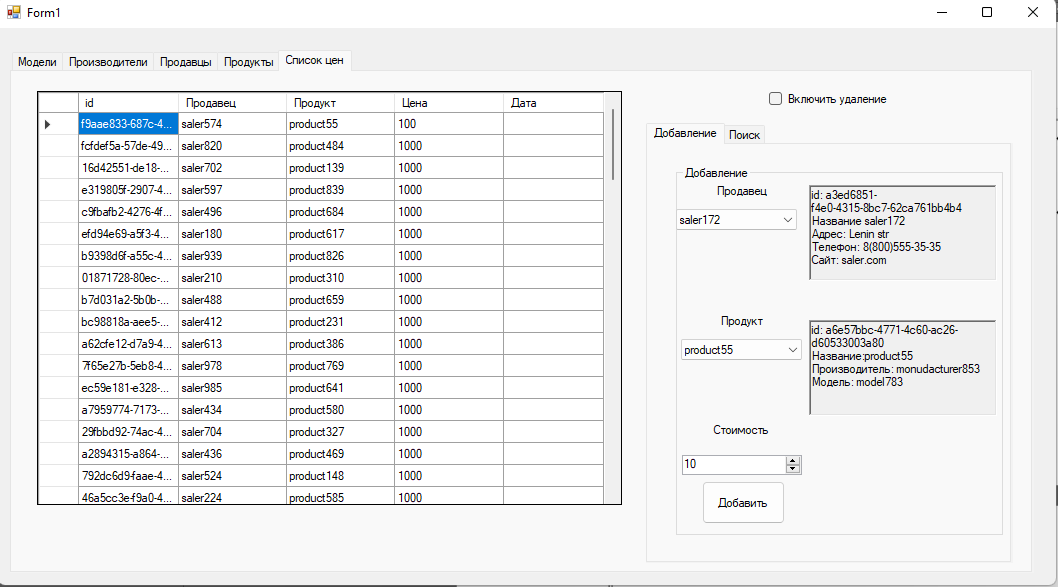


Рисунок 5 – Вкладка «Список цен»

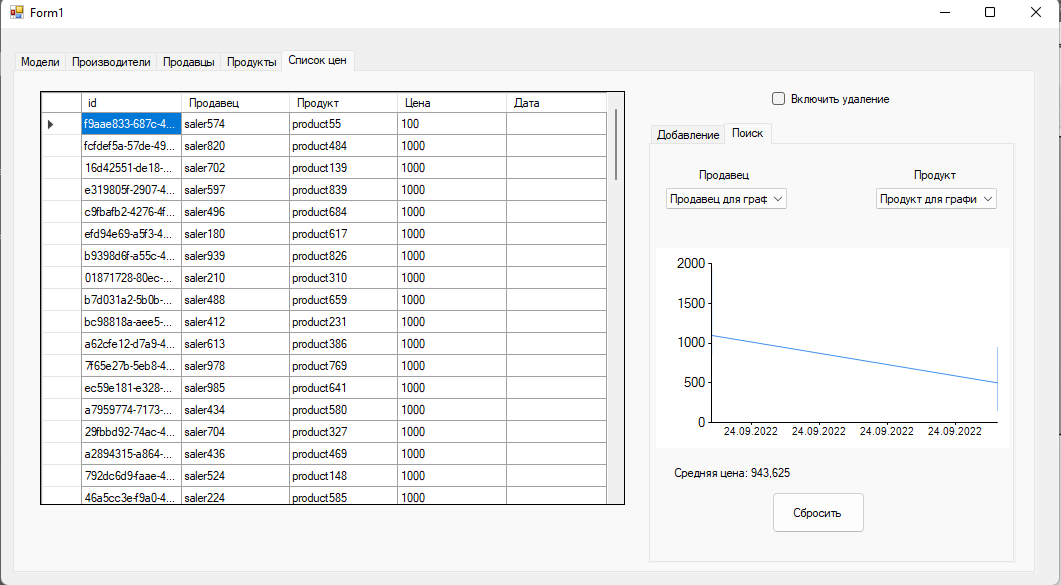


Рисунок 6 – Выдача первого документа

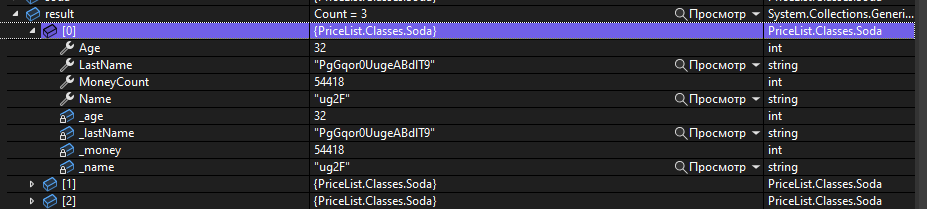


Рисунок 7 – Выдача второго документа