**1 Постановка задачи**

ООО «Красотка» - магазин по продаже косметики для женщин.

В рамках выполнения задания по учебной практике необходимо разработать основные модули информационной системы для ООО «Красотка»:

* неавторизованный клиент может просматривать товары;
* авторизованный клиент может просматривать товары и оформлять заказ;
* менеджер может просматривать товары, оформлять и редактировать заказы;
* администратор может добавлять/редактировать/удалять товары, просматривать и редактировать заказы, добавлять/редактировать/удалять категории товара, роли пользователей, добавлять/удалять пользователей БД, осуществлять импорт и экспорт БД;
* программа должна осуществлять экспорт БД в формат csv при каждом новом открытии программы.

Кроме того, разрабатываемая информационная система предполагает установку на терминалах при входе в торговые центры города. На терминале клиент может просмотреть товары, выбрать себе что-то и заказать товар в удобный для него пункт выдачи заказа.

**2 Структура таблиц в БД**

Таблица 2.1 - Структура таблицы «order»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | OrderID | integer | Идентификатор поля товара. Первичный ключ |
| 2 | OrderDeliveryDate | date | Дата доставки товара |
|  | OrderProductID | integer | Идентификатор товара |
| 3 | OrderDate | date | Дата заказа товара |
| 4 | OrderCode | integer | Код получения товара |
| 5 | OrderPickupPoint | integer | Внешний ключ, связанный с таблицей«orderpickuppoint». Пункт получения товара |
| 6 | OrderClient | integer | Внешний ключ, связанный с таблицей«user». Клиент на которого оформлен заказ |
| 7 | OrderStatus | integer | Внешний ключ, связанный с таблицей«orderstatus». Статус заказа |

Таблица 2.2 - Структура таблицы «orderpickuppoint»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idPickupPoint | integer | Идентификатор поля пункта доставки товара. Первичный ключ |
| 2 | PostCode | integer | Почтовый индекс пункта доставки товара |
| 3 | PickupPoint | text | Адрес пункта доставки товара |

Таблица 2.3 - Структура таблицы «orderproduct»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | OrderID | integer | Идентификатор поля товара. Первичный ключ |
| 2 | ProductArticleNumber | varchar(100) | Артикул товара |
| 3 | OrderCount | integer | Количество товара |

Таблица 2.4 - Структура таблицы «orderstatus»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idStatus | integer | Идентификатор поля статуса товара. Первичный ключ |
| 2 | OrderStatus | varchar(60) | Статус заказа товара |

Таблица 2.5 - Структура таблицы «product»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | ProductArticleNumber | varchar(100) | Идентификатор поля артикула товара. Первичный ключ |
| 2 | ProductName | text | Название товара |
| 3 | ProductDescription | text | Описание товара |
| 4 | ProductCategory | integer | Внешний ключ связанный с таблицей«productcategory». Название категории товара |
| 5 | ProductPhoto | text | Фотография товара |
| 6 | ProductManufacturer | integer | Внешний ключ связанный с таблицей«productmanufacturer». Название производителя товара |
| 7 | ProductCost | decimal(19, 4) | Стоимость товара |
| 8 | ProductDiscountAmount | tinyint | Текущая скидка товара |
| 9 | ProductQuantityInStock | integer | Количество на складе |
| 10 | ProductUnit | integer | Внешний ключ связанный с таблицей«productunit». Единицы измерения товара |
| 11 | ProductSupplier | integer | Внешний ключ связанный с таблицей«productsupplier». Название поставщика товара |
| 12 | ProductMaxDiscount | integer | Максимальная скидка товара |

Таблица 2.6 - Структура таблицы «productcategory»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idCategory | integer | Идентификатор поля категории товара. Первичный ключ |
| 2 | Category | varchar(150) | Название категории товара |

Таблица 2.7 - Структура таблицы «productmanufacturer»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idManufacturer | integer | Идентификатор поля производителя товара. Первичный ключ |
| 2 | Manufacturer | varchar(150) | Название производителя товара |

Таблица 2.8 - Структура таблицы «productsupplier»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idSupplier | integer | Идентификатор поля поставщика товара. Первичный ключ |
| 2 | Supplier | varchar(150) | Название поставщика товара |

Таблица 2.9 - Структура таблицы «productunit»

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** |
| 1 | idUnit | integer | Идентификатор поля единицы измерения товара. Первичный ключ |
| 2 | Unit | varchar(150) | Название единицы измерения товара |

Таблица 2.10 - Структура таблицы «role»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** | |
| 1 | RoleID | integer | | Идентификатор поля роли пользователя. Первичный ключ |
| 2 | RoleName | varchar(100) | | Название роли пользователя |

Таблица 2.11- Структура таблицы «user»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** | |
| 1 | UserID | integer | | Идентификатор поля пользователя. Первичный ключ |
| 2 | UserSurname | varchar(100) | | Фамилия пользователя |
| 3 | UserName | varchar(100) | | Имя пользователя |
| 4 | UserPatronymic | varchar(100) | | Отчество пользователя |
| 5 | UserLogin | text | | Логин пользователя |
| 6 | UserPassword | text | | Пароль пользователя |
| 7 | UserRole | integer | | Внешний ключ связанный с таблицей«role». |

Таблица 2.12 - Структура таблицы «cart»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№ п/п** | **Название поля** | **Тип данных** | **Описание поля** | |
| 1 | idCart | integer | | Идентификатор поля корзины. Первичный ключ |
| 2 | idUser | integer | | Идентификатор пользователя |
| 3 | idProduct | varchar(60) | | Идентификатор товара |
| 4 | NameProduct | varchar(150) | | Название товара |
| 5 | Price | decimal(19,2) | | Общая цена корзины |
| 6 | Quantity | integer | | Общее количество в корзине |

**3 Схема базы данных**

На рисунке 3.1 представлена схема базы данных и используемые связи.

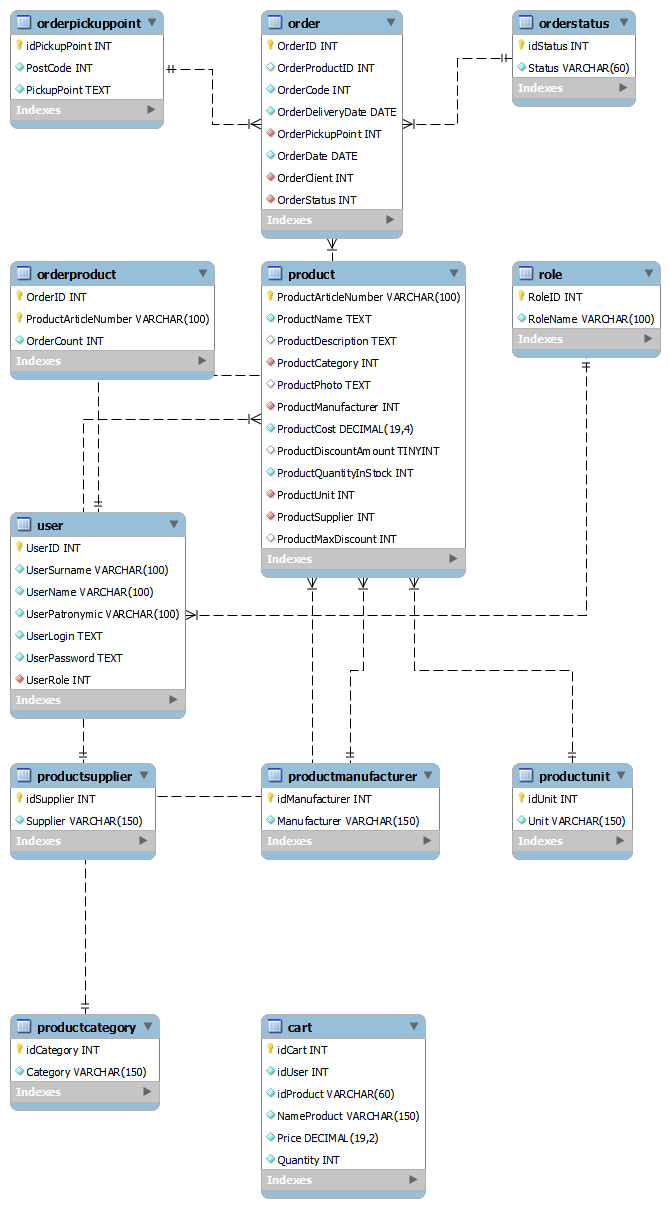


Рисунок 3.1 - Схема базы данных

**4 Описание программных модулей**

На рисунке 4.1 показана форма «Авторизация»

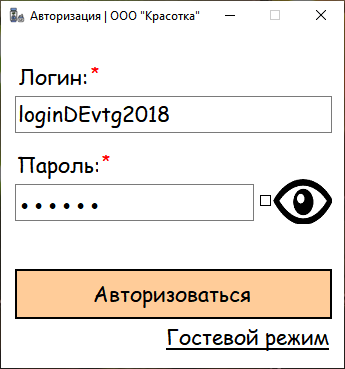


Рисунок 4.1 – Форма«Авторизация»

На рисунке 4.2 показано модальное окно капчи для формы «Авторизация»

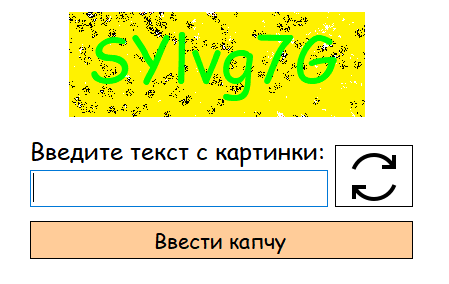


Рисунок 4.2 – Модальное окно капчи для формы «Авторизация»

На рисунке 4.3 показана форма главного меню за администратора

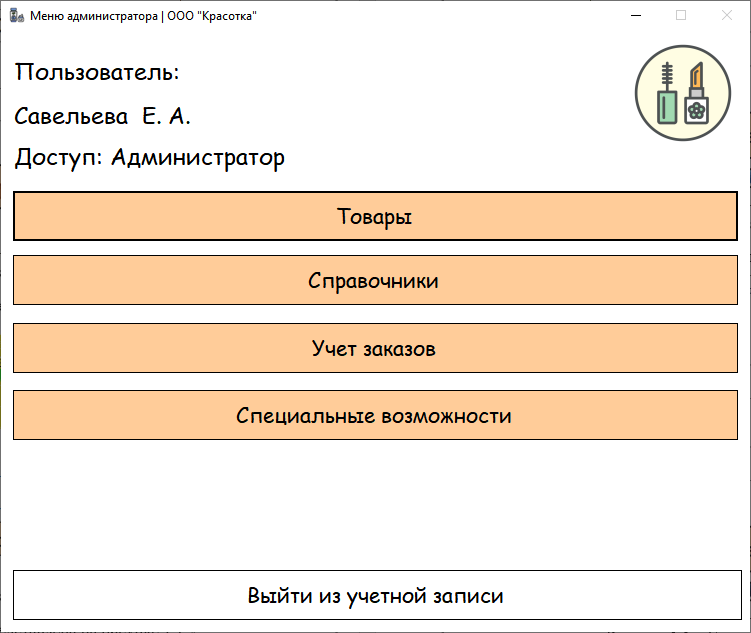


Рисунок 4.3 – Форма меню за администратора

На рисунке 4.4 показана форма главного меню за менеджера

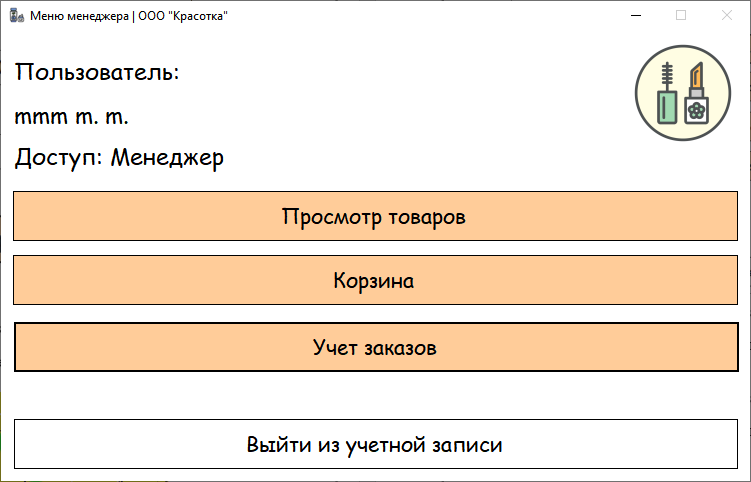


Рисунок 4.4 – Форма меню за менеджера

На рисунке 4.5 показана форма главного меню за клиента



Рисунок 4.5 – Форма меню за пользователя

На рисунке 4.6 показана форма «Товары»под учетной записью неавторизованного пользователя, клиента и менеджера

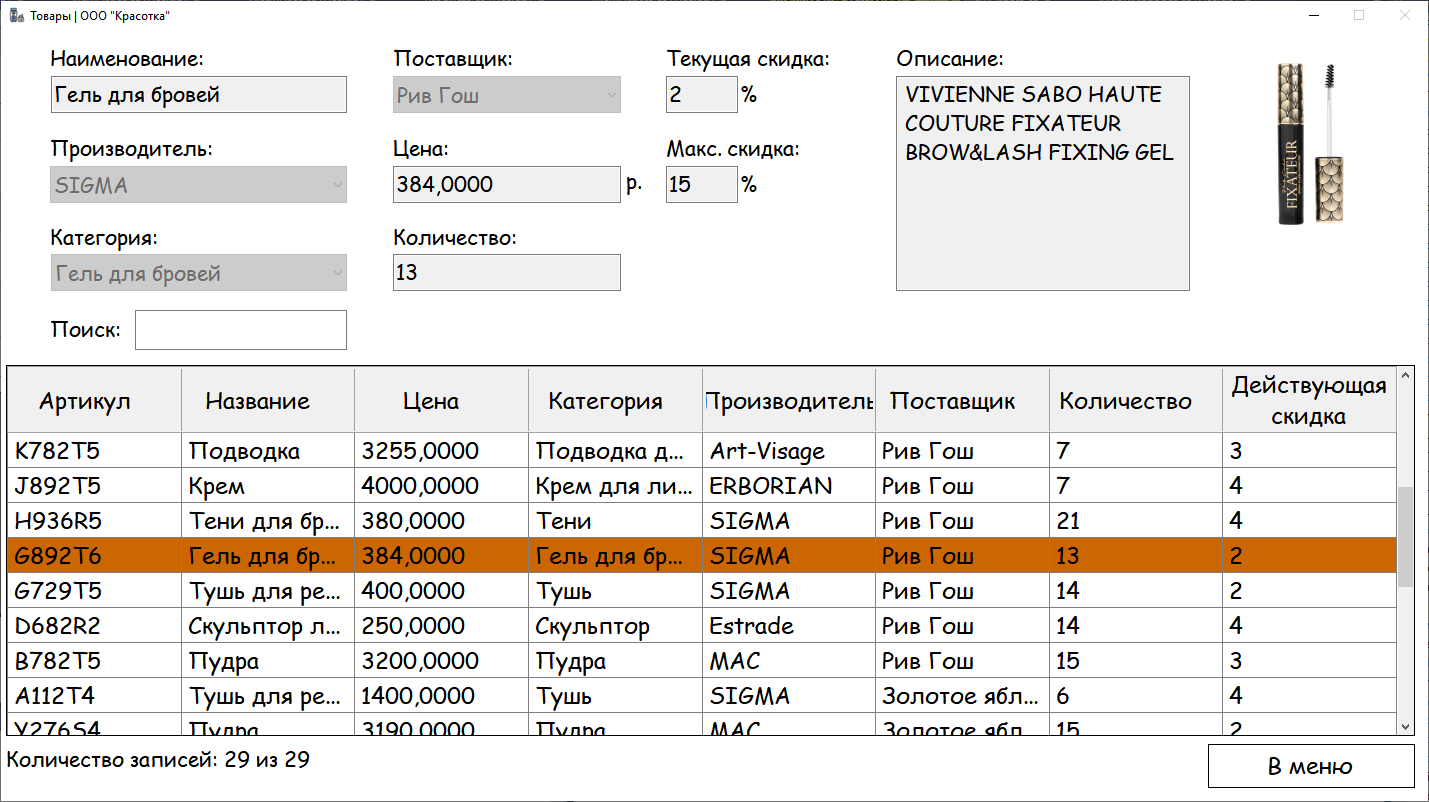


Рисунок 4.6 – Форма просмотра товаров

На рисунке 4.7 показана форма «Заказ товара» под учетной записью клиента и менеджера

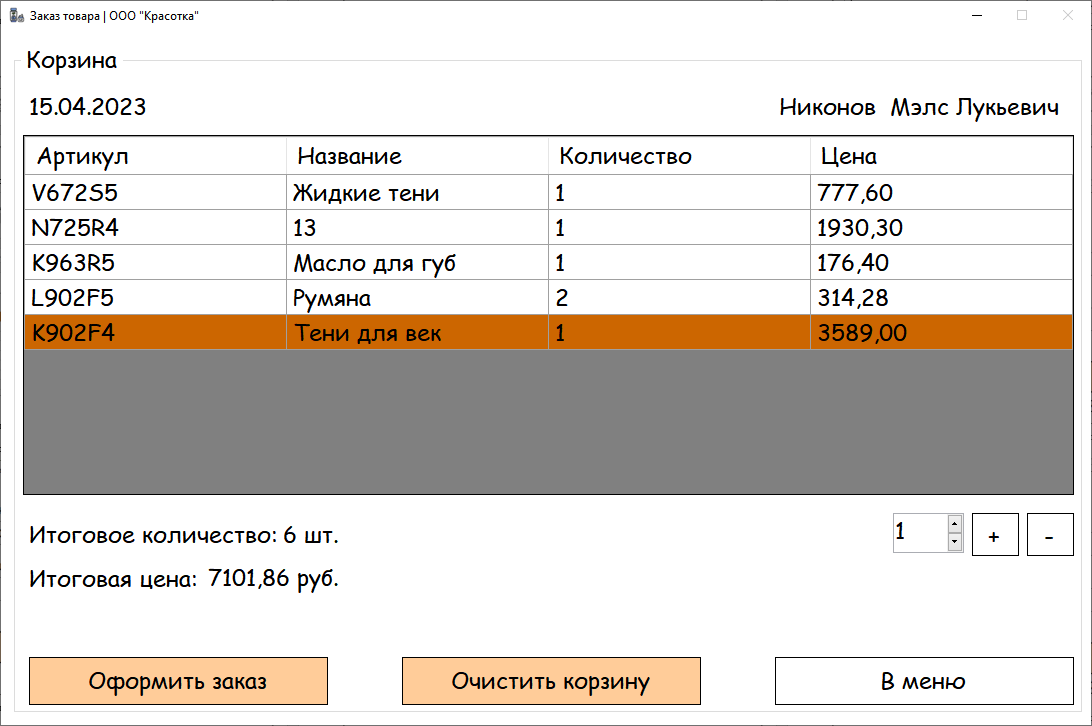


Рисунок 4.7 – Форма заказа товара

На рисунке 4.8 показана форма «Оформление заказа» под учетной записью клиента и менеджера

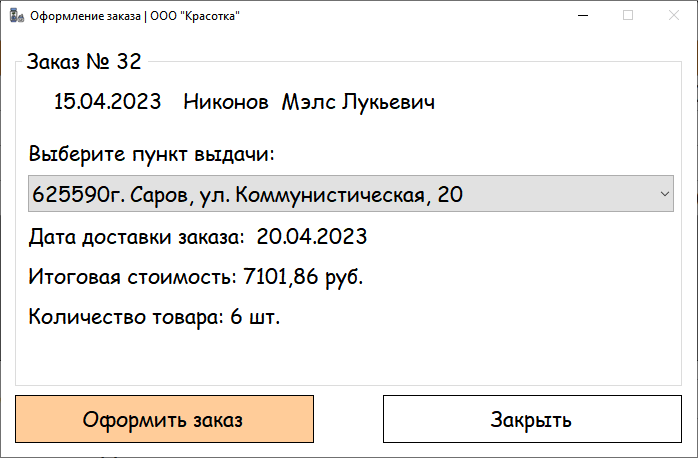


Рисунок 4.8 – Форма оформления заказа

На рисунке 4.9 показана форма «Учет заказов» под учетной записью менеджера и администратора

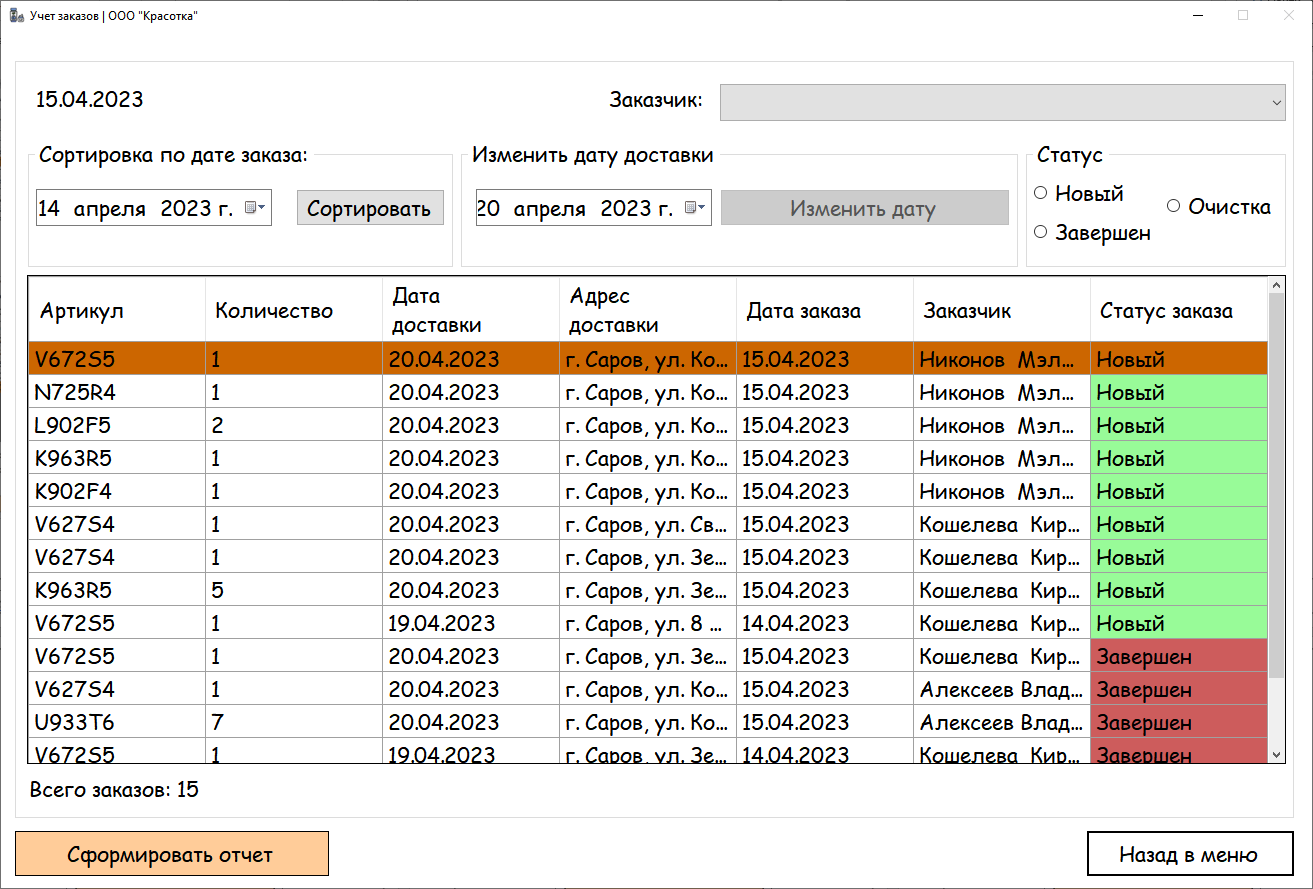


Рисунок 4.9 – Форма учета заказов

На рисунке 4.10 показана форма просмотра товаров с возможностью добавления/редактирования/удаления под учетной записью администратора

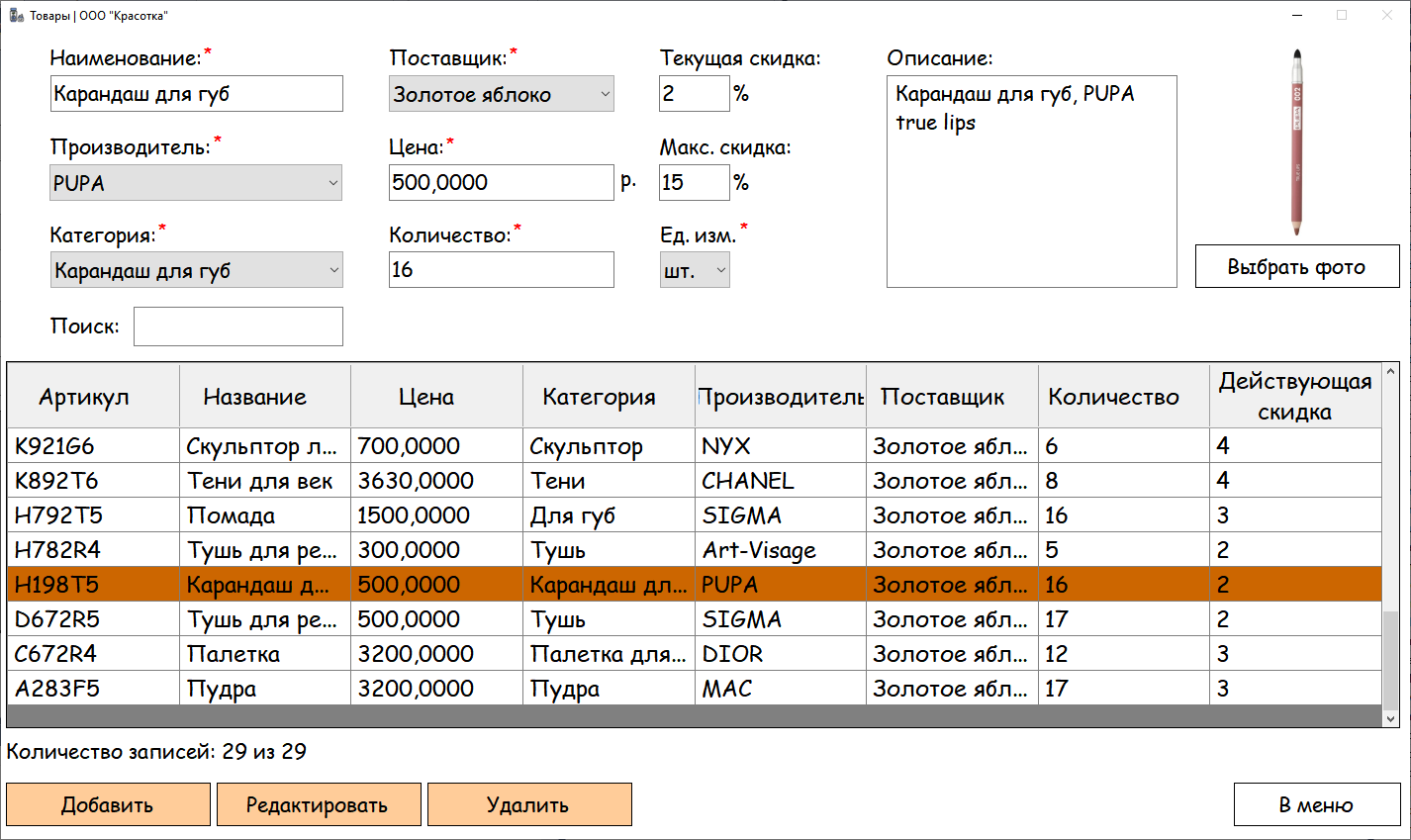


Рисунок 4.10 – Форма просмотра товаров за администратора

На рисунке 4.11 показана форма «Справочники», раздел «Категории»

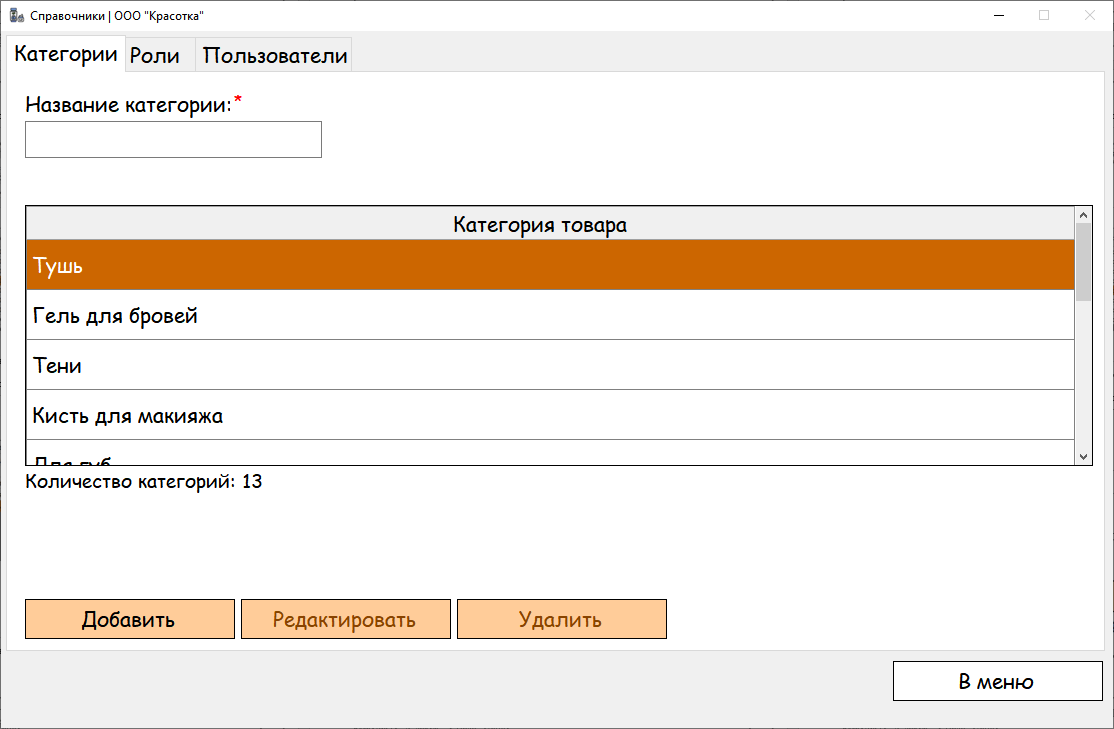


Рисунок 4.11 – форма «Справочники», раздел «Категории»

На рисунке 4.12 показана форма «Справочники», раздел «Роли»

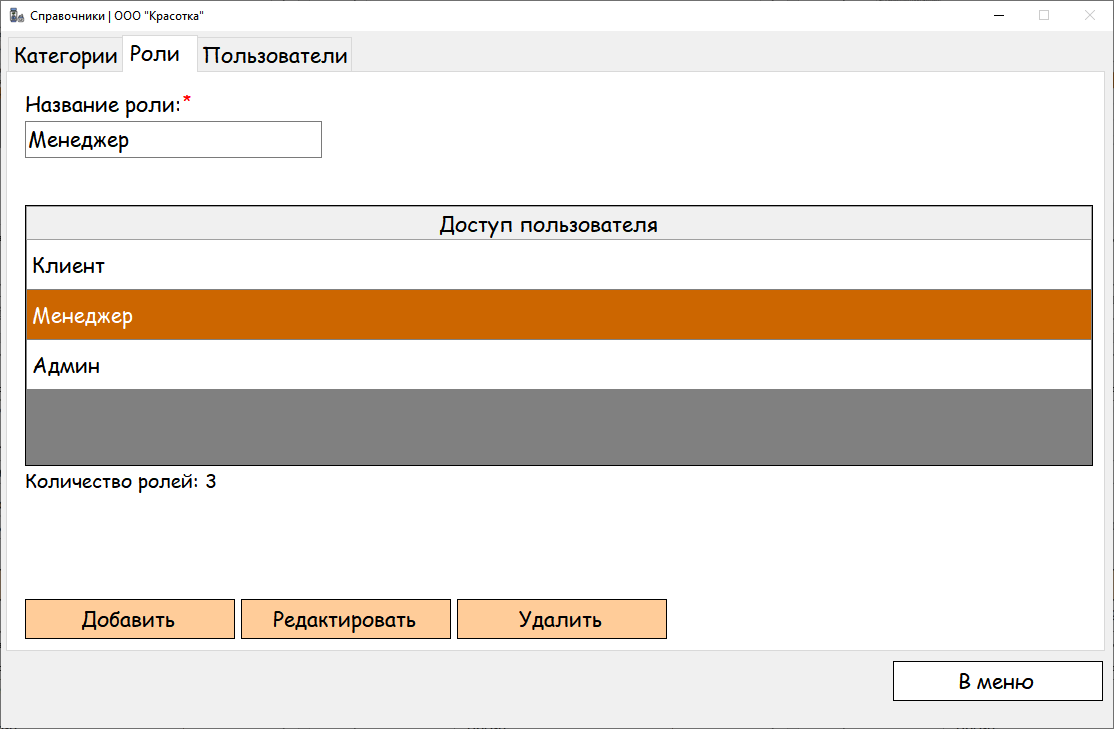


Рисунок 4.12 – форма «Справочники», раздел «Роли»

На рисунке 4.13 показана форма «Справочники», раздел «Пользователи»

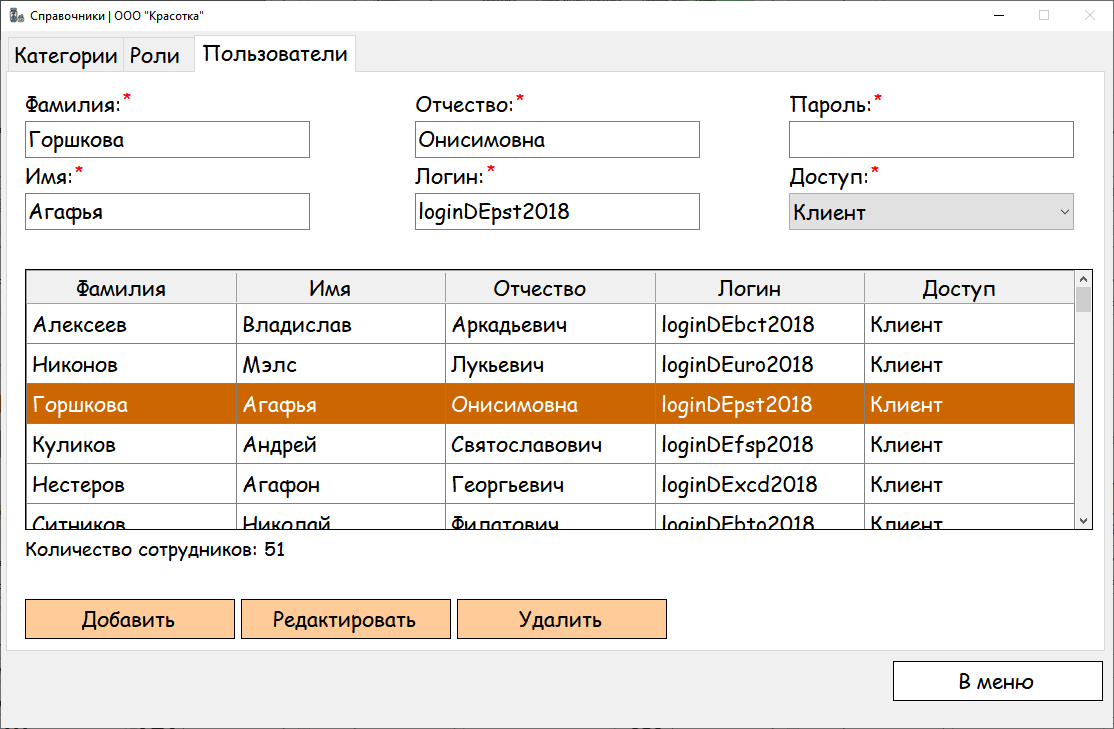


Рисунок 4.13 – форма «Справочники», раздел «Пользователи»

На рисунке 4.14 показана форма «Специальные возможности», раздел «Импорт БД»

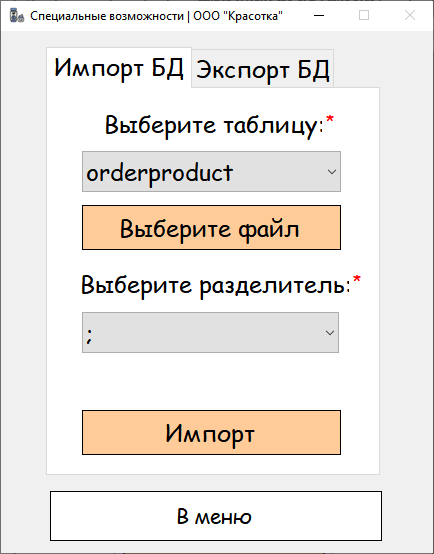


Рисунок 4.14 – форма «Специальные возможности», раздел «Импорт БД»

На рисунке 4.15 показана форма «Специальные возможности», раздел «Экспорт БД»

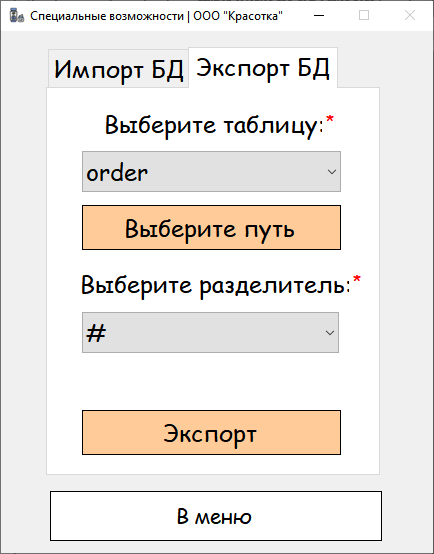


Рисунок 4.15 – форма «Специальные возможности», раздел «Экспорт БД»

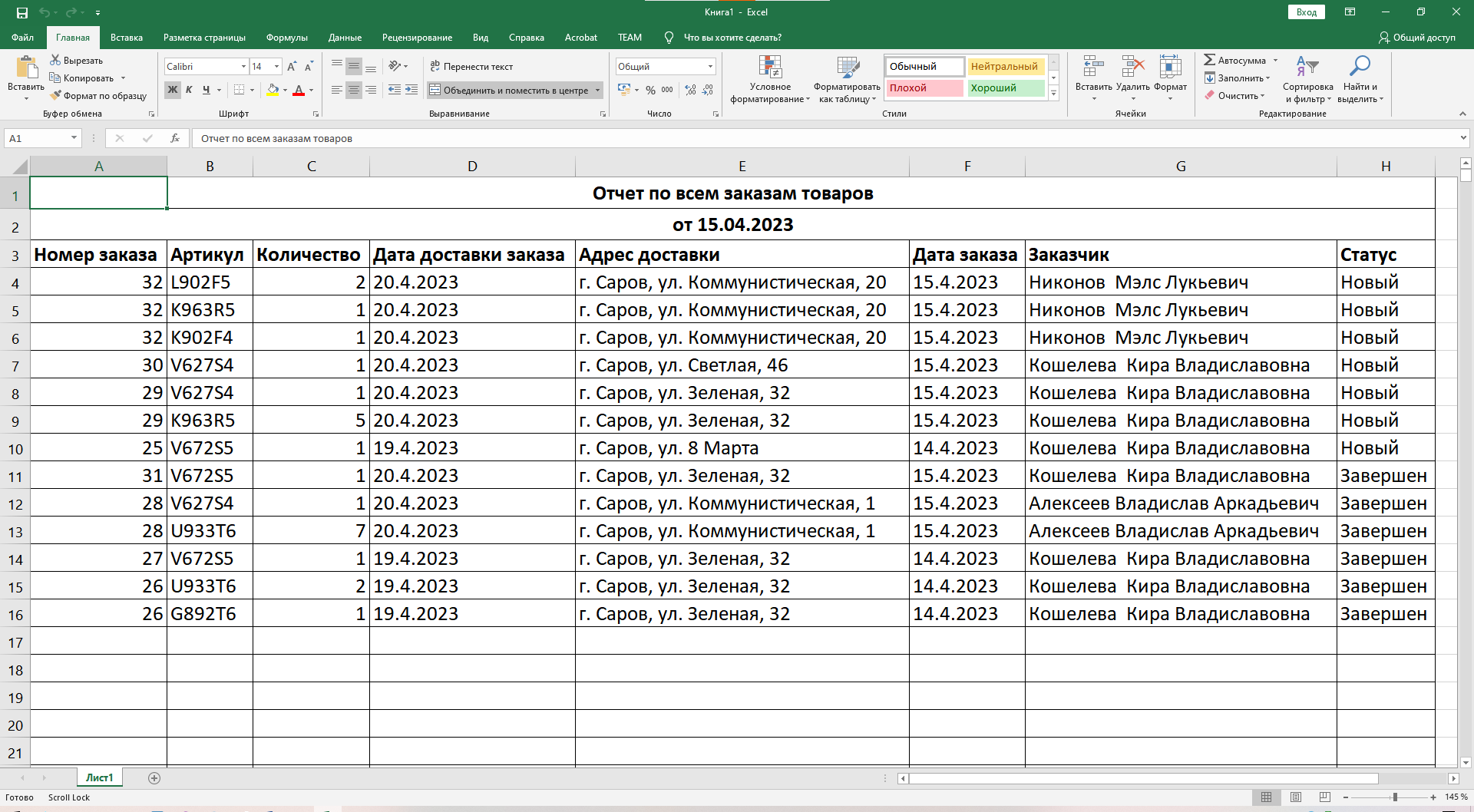
На рисунке 4.16 показана печать отчета по заказам в файл Excel

Рисунок 4.16 – Печать отчета по заказам в файл Excel

На рисунке 4.17 показана печать отчета по заказам в файл Word

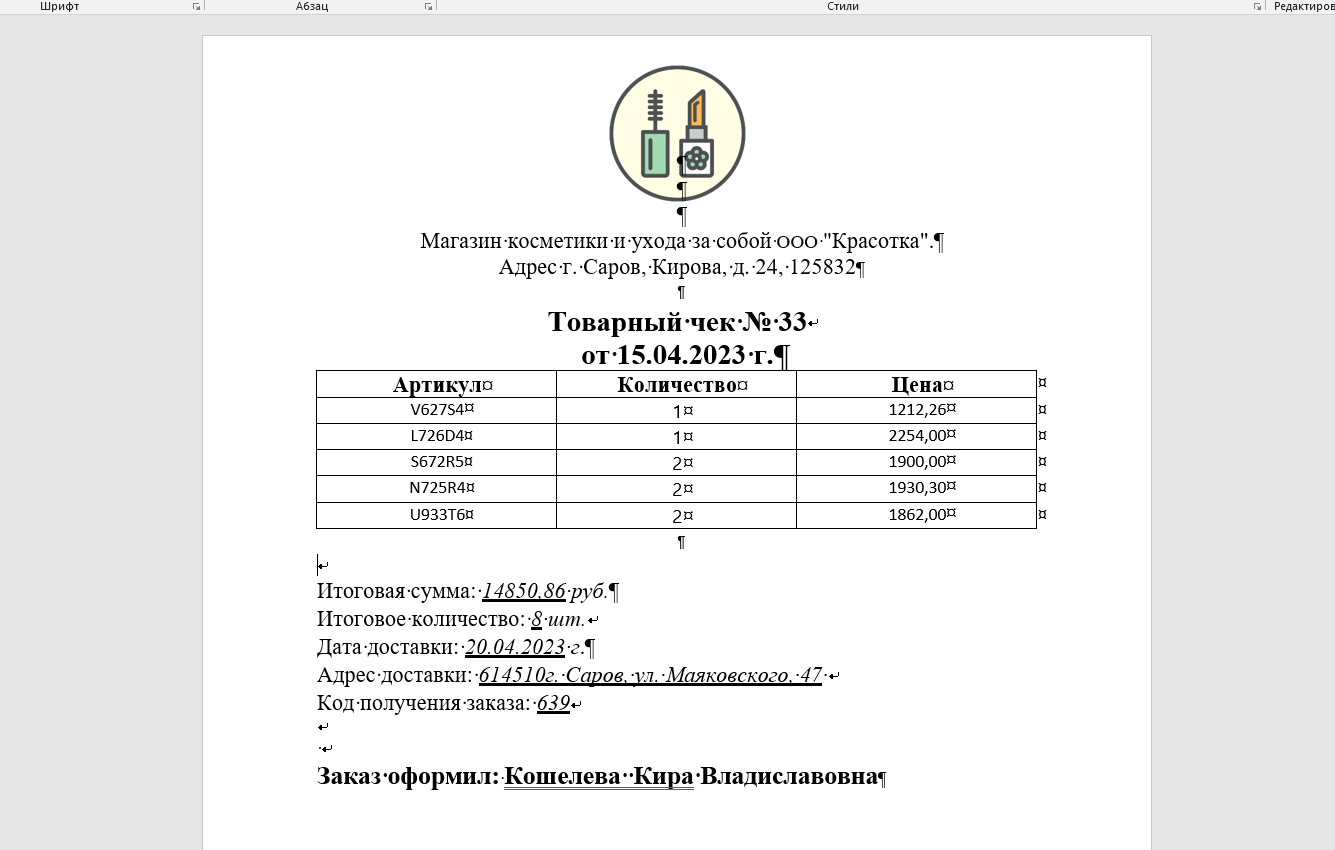


Рисунок 4.17 – Печать чека заказа в файл Word

**5 Листинг программы**

**AuthForm.cs** – главная форма авторизации

using MySql.Data.MySqlClient;

using System;

using System.IO;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Security.Cryptography;

namespace BeautyGlory

{

publicpartialclassAuthForm : Form

{

staticpublicint role\_user = 0;

staticpublicstring id\_user = "";

int count\_auth = 0;

publicstaticint checkform = 0;

Captch captch = newCaptch();

MenuAdmin menuadmin = newMenuAdmin();

MenuManager menumanager = newMenuManager();

MenuUser menuuser = newMenuUser();

ViewProduct vprod = newViewProduct();

public AuthForm()

{

BackupDB bk = newBackupDB();

bk.getBackup();

InitializeComponent();

}

privatevoid Auth\_Load(object sender, EventArgs e)

{

tbPass.UseSystemPasswordChar = true;

}

privatevoid cbPass\_CheckedChanged(object sender, EventArgs e)

{

if (cbPass.Checked == true)

{

tbPass.UseSystemPasswordChar = false;

}

else

{

tbPass.UseSystemPasswordChar = true;

}

}

privatevoid bAuth\_Click(object sender, EventArgs e)

{

try

{

if (count\_auth > 0)

{

MessageBox.Show("Внимание! Теперьдляавторизациинеобходимоввестикапчу.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

captch.ShowDialog();

}

if (tbLog.Text == "" || tbPass.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя!", "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

count\_auth++;

}

else

{

string tblog = tbLog.Text;

string tbpass = tbPass.Text;

string dbpass = "";

bool flag = true;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = "SELECT UserPassword FROM user WHERE UserLogin = '" + tblog + "';";

MySqlCommand command = newMySqlCommand(sql, connection.GetConnect());

try

{

dbpass = command.ExecuteScalar().ToString();

}

catch

{

MessageBox.Show("Ошибка! Такого пользователя не существует. Повторите попытку.", "Возникла ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbLog.Clear();

flag = false;

count\_auth++;

}

if (flag)

{

if (count\_auth > 1)

{

MessageBox.Show("Внимание! Теперьдляавторизациинеобходимоввестикапчу.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

captch.ShowDialog();

}

if (tbpass == dbpass)

{

count\_auth = 0;

var sql\_role = "SELECT UserRole FROM user WHERE UserLogin = @tblog;";

using (var com = newMySqlCommand(sql\_role, connection.GetConnect()))

{

com.Parameters.AddWithValue("@tblog", tblog);

role\_user = Convert.ToInt32(com.ExecuteScalar());

}

var sql\_userid = "SELECT UserID FROM user WHERE UserLogin = @tblog;";

using (var com\_id = newMySqlCommand(sql\_userid, connection.GetConnect()))

{

com\_id.Parameters.AddWithValue("@tblog", tblog);

id\_user = com\_id.ExecuteScalar().ToString();

}

tbPass.Clear();

tbLog.Clear();

switch (role\_user)

{

case 1:

checkform = 0;

this.Visible = false;

menuuser.ShowDialog();

this.Visible = true;

break;

case 2:

checkform = 14741;

this.Visible = false;

menumanager.ShowDialog();

this.Visible = true;

break;

case 3:

checkform = 14741;

this.Visible = false;

menuadmin.ShowDialog();

this.Visible = true;

break;

default:

break;

}

}

else

{

MessageBox.Show("Ошибка! Введенневерныйпароль.\n Повторитепопытку.", "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbPass.Clear();

count\_auth++;

}

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid tbLog\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= 'a'&& e.KeyChar <= 'z' || e.KeyChar >= 'A'&& e.KeyChar <= 'Z' || e.KeyChar >= '0'&& e.KeyChar <= '9' || e.KeyChar == 8 || e.KeyChar == ' ')

{

return;

}

else

{

e.Handled = true;

}

}

privatevoid linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

tbLog.Clear();

tbPass.Clear();

this.Visible = false;

vprod.ShowDialog();

this.Visible = true;

}

privatevoid Auth\_FormClosing(object sender, FormClosingEventArgs e)

{

if (MessageBox.Show("Выуверены, чтохотитевыйтиизпрограммы?", "Выходизпрограммы", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == System.Windows.Forms.DialogResult.No)

e.Cancel = true;

else

e.Cancel = false;

}

}

}

**BackupDB.cs** – класс бэкапа БД при открытии приложения

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using MySql.Data.MySqlClient;

using System.IO;

using System.Diagnostics;

using System.Data.SqlClient;

using System.Data;

namespace BeautyGlory

{

classBackupDB

{

publicvoid getBackup()

{

try

{

string exportPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "BackupDataBaseCSV\_" + DateTime.Now.ToString("yyyy-MM-dd HH-mm-ss"));

if (!Directory.Exists(exportPath))

{

Directory.CreateDirectory(exportPath);

}

using (db\_connect db = newdb\_connect())

{

MySqlConnection connection = db.GetConnect();

DataTable schema = connection.GetSchema("Tables");

foreach (DataRow row in schema.Rows)

{

string tableName = row[2].ToString();

string sql = string.Format("SELECT \* FROM `{0}`", tableName);

using (MySqlDataAdapter adapter = newMySqlDataAdapter(sql, connection))

{

DataTable dataTable = newDataTable();

adapter.Fill(dataTable);

string fileName = Path.Combine(exportPath, tableName + ".csv");

using (StreamWriter writer = newStreamWriter(fileName))

{

foreach (DataRow dataRow in dataTable.Rows)

{

string rowText = "";

for (int i = 0; i < dataTable.Columns.Count; i++)

{

rowText += dataRow[i].ToString();

if (i < dataTable.Columns.Count - 1)

{

rowText += ",";

}

}

writer.WriteLine(rowText);

}

}

}

}

}

}

catch (Exception)

{

;

}

}

}

}

**Captch.cs** – форма капчи на форме авторизации

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassCaptch : Form

{

bool flag = false;

public Captch()

{

InitializeComponent();

GenCaptch();

}

privatevoid bInput\_Click(object sender, EventArgs e)

{

if (tbCaptch.Text == "")

{

MessageBox.Show("Ошибка. Введитекапчу!", "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

bInput.Enabled = false;

bReload.Enabled = false;

Thread.Sleep(10000);

bInput.Enabled = true;

bReload.Enabled = true;

}

else

{

if (lCaptch.Text == tbCaptch.Text)

{

tbCaptch.Clear();

flag = true;

this.Close();

}

else

{

MessageBox.Show("Ошибка. Капчаневерная! Повторитепопыткучерез 10 секунд.", "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

bInput.Enabled = false;

bReload.Enabled = false;

Thread.Sleep(10000);

bInput.Enabled = true;

bReload.Enabled = true;

GenCaptch();

tbCaptch.Clear();

}

}

}

publicvoid GenCaptch()

{

string text = "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm0123456789";

string captch = "";

int len = text.Length;

Random rnd = newRandom();

for (int i = 0; i < 8; i++)

{

captch += text[rnd.Next(1, len)];

}

lCaptch.Text = captch;

}

privatevoid bReload\_Click(object sender, EventArgs e)

{

tbCaptch.Clear();

GenCaptch();

}

privatevoid Captch\_FormClosing(object sender, FormClosingEventArgs e)

{

if (flag == true)

{

e.Cancel = false;

}

else

{

e.Cancel = true;

}

}

privatevoid Captch\_Load(object sender, EventArgs e)

{

GenCaptch();

}

}

}

**dbconnect.cs** – классподключениякБД

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BeautyGlory

{

classdb\_connect : IDisposable

{

privatereadonlyMySqlConnection connection;

public db\_connect()

{

connection = newMySqlConnection("host=localhost;uid=root;pwd=;database=trade1;");

OpenConnect();

}

publicvoid Dispose()

{

CloseConnect();

}

publicvoid OpenConnect()

{

if (connection.State == System.Data.ConnectionState.Closed)

{

connection.Open();

}

}

publicvoid CloseConnect()

{

if (connection.State == System.Data.ConnectionState.Open)

{

connection.Close();

}

}

publicMySqlConnection GetConnect()

{

return connection;

}

}

}

**Guide.cs** – формасправочники

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassGuide : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

string id\_role = "";

string id\_cat = "";

string id\_emp = "";

bool flag\_category = false;

bool flag\_role = false;

public Guide()

{

InitializeComponent();

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

this.Close();

}

privatevoid Guide\_Load(object sender, EventArgs e)

{

FillRole();

FillCat();

FillEmp();

cbRole\_Fill();

CountRowsDgv();

ClearTbEmp();

bDel\_Role.Enabled = false;

bEdit\_Role.Enabled = false;

bDel\_Emp.Enabled = false;

bEdit\_Emp.Enabled = false;

bDel\_Cat.Enabled = false;

bUpd\_Cat.Enabled = false;

}

publicvoid FillRole()

{

using (var connection = newdb\_connect())

{

connection.OpenConnect();

string sql = @"SELECT RoleID AS 'ID', RoleName AS 'Доступпользователя' FROM role;";

using (var com = newMySqlCommand(sql, connection.GetConnect()))

{

using (var adapter = newMySqlDataAdapter(com))

{

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvRole.RowTemplate.Height = 50;

dgvRole.DataSource = dt;

foreach (DataGridViewColumn column in dgvRole.Columns)

{

if (column.Index == 0)

{

column.Visible = false;

}

else

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

}

}

}

ClearTbEmp();

connection.CloseConnect();

}

}

publicvoid FillCat()

{

try

{

using (var connection = newdb\_connect())

{

connection.OpenConnect();

string sql = @"SELECT idCategory AS 'ID', Category AS 'Категориятовара' FROM productcategory;";

using (var com = newMySqlCommand(sql, connection.GetConnect()))

{

using (var adapter = newMySqlDataAdapter(com))

{

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvCat.RowTemplate.Height = 50;

dgvCat.DataSource = dt;

dgvCat.Columns[0].Visible = false;

foreach (DataGridViewColumn column in dgvCat.Columns)

{

if (column.Index != 0)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

}

}

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid FillEmp()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT user.UserID AS 'ID', user.UserSurname AS 'Фамилия', user.USerName AS 'Имя', user.UserPatronymic AS 'Отчество', user.UserLogin AS 'Логин', role.RoleName AS 'Доступ' FROM user INNER JOIN role ON role.RoleID = user.UserRole;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

dgvEmp.RowTemplate.Height = 40;

adapter.Fill(dt);

dgvEmp.DataSource = dt;

dgvEmp.Columns[0].Visible = false;

foreach (DataGridViewColumn column in dgvEmp.Columns)

{

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.AllCells;

column.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

}

connection.CloseConnect();

}

privatevoid bAdd\_Cat\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(tbCat.Text))

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя.", "Добавлениекатегории", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

int categoryID = GetCategory();

if (categoryID != 1012)

{

MessageBox.Show("Ошибка! Такаякатегорияужесуществует.", "Добавлениекатегории", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbCat.Clear();

return;

}

using (var connection = newdb\_connect())

{

connection.OpenConnect();

string sql = string.Format("INSERT INTO productcategory VALUES (null, '{0}')", tbCat.Text);

using (var com = newMySqlCommand(sql, connection.GetConnect()))

{

com.ExecuteNonQuery();

}

}

tbCat.Clear();

FillCat();

CountRowsDgv();

MessageBox.Show("Успешно! Записьдобавлена.", "Добавлениекатегории", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

privatevoid bAdd\_Role\_Click(object sender, EventArgs e)

{

// Очищаем поле tbRole и проверяем, заполнено ли оно

tbRole.Clear();

if (tbRole.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя.", "Добавлениероли", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

else

{

if (GetRole() != 1012)

{

MessageBox.Show("Ошибка! Такая роль уже существует.", "Добавление роли", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbRole.Clear();

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("INSERT INTO role VALUES(null, '{0}') ;", tbRole.Text);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

tbRole.Clear();

FillRole();

CountRowsDgv();

MessageBox.Show("Успешно! Записьдобавлена.", "Добавлениероли", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

privatevoid dgvRole\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

id\_role = dgvRole.Rows[e.RowIndex].Cells[0].Value.ToString();

tbRole.Text = dgvRole.Rows[e.RowIndex].Cells[1].Value.ToString();

bDel\_Role.Enabled = true;

bEdit\_Role.Enabled = true;

}

catch (Exception)

{

;

}

}

privatevoid dgvCat\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

id\_cat = dgvCat.Rows[e.RowIndex].Cells[0].Value.ToString();

tbCat.Text = dgvCat.Rows[e.RowIndex].Cells[1].Value.ToString();

bDel\_Cat.Enabled = true;

bUpd\_Cat.Enabled = true;

}

catch (Exception)

{

;

}

}

privatevoid bDel\_Cat\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитеудалитьвыбраннуюзапись?", "Удалениекатегории", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (DialogResult.Yes == result)

{

Check\_Cat\_Using();

if (flag\_category == true)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"DELETE FROM productcategory WHERE idCategory = " + id\_cat + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноудалена.", "Удалениекатегории", MessageBoxButtons.OK, MessageBoxIcon.Information);

bDel\_Cat.Enabled = false;

bUpd\_Cat.Enabled = false;

tbCat.Clear();

FillCat();

CountRowsDgv();

}

else

{

MessageBox.Show("Ошибка! Даннаякатегорияиспользуется.", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

privatevoid bDel\_Role\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитеудалитьвыбраннуюзапись?", "Удалениероли", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (DialogResult.Yes == result)

{

Check\_Role\_Using();

if (flag\_role == true)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"DELETE FROM role WHERE RoleID = " + id\_role + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноудалена.", "Удалениероли", MessageBoxButtons.OK, MessageBoxIcon.Information);

bDel\_Role.Enabled = false;

bEdit\_Role.Enabled = false;

tbRole.Clear();

FillRole();

CountRowsDgv();

}

else

{

MessageBox.Show("Ошибка! Даннаярольиспользуется.", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

privatevoid bUpd\_Cat\_Click(object sender, EventArgs e)

{

if (GetCategory() != 1012)

{

MessageBox.Show("Ошибка! Такая категория уже существует.", "Добавление категории", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbCat.Clear();

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("UPDATE productcategory SET Category = '{0}' WHERE idCategory = {1};", tbCat.Text, id\_cat);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноотредактирована.", "Редактированиекатегории", MessageBoxButtons.OK, MessageBoxIcon.Information);

bDel\_Cat.Enabled = false;

bUpd\_Cat.Enabled = false;

tbCat.Clear();

FillCat();

CountRowsDgv();

}

}

privatevoid bEdit\_Role\_Click(object sender, EventArgs e)

{

if (GetRole() != 1012)

{

MessageBox.Show("Ошибка! Такая роль уже существует.", "Добавление роли", MessageBoxButtons.OK, MessageBoxIcon.Error);

tbRole.Clear();

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("UPDATE role SET RoleName = '{0}' WHERE RoleID = {1};", tbRole.Text, id\_role);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноотредактирована.", "Редактированиероли", MessageBoxButtons.OK, MessageBoxIcon.Information);

bDel\_Role.Enabled = false;

bEdit\_Role.Enabled = false;

tbRole.Clear();

FillRole();

CountRowsDgv();

}

}

privatevoid tbRole\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= 'а'&& e.KeyChar <= 'я' || e.KeyChar >= 'А'&& e.KeyChar <= 'Я' || e.KeyChar == 8 || e.KeyChar == ' ')

{

return;

}

else

{

e.Handled = true;

}

}

privateint GetCategory()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT idCategory FROM productcategory WHERE Category ='" + tbCat.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

object result = com.ExecuteScalar();

if (result != null)

{

int temp = Convert.ToInt32(result.ToString());

return temp;

}

else

{

// handle the case where the query does not return any results

return 1012;

}

}

catch

{

// handle any other exceptions that may occur

return 1012;

}

}

privateint GetRole()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT RoleID FROM role WHERE RoleName ='" + tbRole.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

object result = com.ExecuteScalar();

if (result != null)

{

int temp = Convert.ToInt32(result.ToString());

return temp;

}

else

{

// handle the case where the query does not return any results

return 1012;

}

}

catch

{

// handle any other exceptions that may occur

return 1012;

}

}

privatevoid Check\_Cat\_Using()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT ProductName FROM product WHERE ProductCategory =" + GetCategory() + ";";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

try

{

string temp = com.ExecuteScalar().ToString();

flag\_category = false;

}

catch

{

flag\_category = true;

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid Check\_Role\_Using()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT RoleName FROM user WHERE UserRole =" + GetRole() + ";";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

try

{

string temp = com.ExecuteScalar().ToString();

flag\_role = false;

}

catch

{

flag\_role = true;

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid CountRowsDgv()

{

label6.Text = "Количествокатегорий: " + dgvCat.Rows.Count;

label8.Text = "Количестворолей: " + dgvRole.Rows.Count;

label9.Text = "Количество сотрудников: " + dgvEmp.Rows.Count;

}

privatevoid cbRole\_Fill()

{

cbRole.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM role;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cbRole.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid bAdd\_Emp\_Click(object sender, EventArgs e)

{

if (tbFirst\_name.Text == "" || tbName.Text == "" || tbMiddle\_name.Text == "" || tbLogin.Text == "" || tbPass.Text == "" || cbRole.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя.", "Добавлениепользователя", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if(CheckNum() != 1012)

{

MessageBox.Show("Предупреждение! Данный логин уже используется другим сотрудником.", "Добавление пользователя", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("INSERT INTO user VALUES(null, '{0}', '{1}', '{2}', '{3}', '{4}', {5}) ;", tbFirst\_name.Text, tbName.Text, tbMiddle\_name.Text, tbLogin.Text, tbPass.Text, role\_id\_u());

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

ClearTbEmp();

FillEmp();

CountRowsDgv();

MessageBox.Show("Успешно! Записьдобавлена.", "Добавлениепользователя", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

publicint role\_id\_u()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = "SELECT RoleID FROM role WHERE RoleName = '" + cbRole.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int role\_int\_id\_u = Convert.ToInt32(com.ExecuteScalar());

connection.CloseConnect();

return role\_int\_id\_u;

}

privatevoid bEdit\_Emp\_Click(object sender, EventArgs e)

{

if (tbFirst\_name.Text == "" || tbName.Text == "" || tbMiddle\_name.Text == "" || tbLogin.Text == "" || tbPass.Text == "" || cbRole.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя.", "Добавлениепользователя", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (CheckNumUpd() != 1012)

{

MessageBox.Show("Предупреждение! Данный логин уже используется другим сотрудником.", "Добавление пользователя ", MessageBoxButtons.OK, MessageBoxIcon.Warning);

tbLogin.Clear();

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("UPDATE user SET UserSurname = '{0}', UserName = '{1}', UserPatronymic = '{2}', UserLogin = '{3}' WHERE UserID = {4};", tbFirst\_name.Text, tbName.Text, tbMiddle\_name.Text, tbLogin.Text, id\_emp);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

ClearTbEmp();

FillEmp();

CountRowsDgv();

MessageBox.Show("Успешно! Записьотредактирована.", "Редактированиепользователя", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

privatevoid bDel\_Emp\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитеудалитьвыбраннуюзапись?", "Удалениепользователя", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (DialogResult.Yes == result)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"DELETE FROM user WHERE UserID = " + id\_emp + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноудалена.", "Удалениепользователя", MessageBoxButtons.OK, MessageBoxIcon.Information);

bEdit\_Emp.Enabled = false;

bDel\_Emp.Enabled = false;

ClearTbEmp();

FillEmp();

CountRowsDgv();

}

}

privateint CheckNum()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT UserID FROM user WHERE UserLogin ='" + tbLogin.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int temp = Convert.ToInt32(com.ExecuteScalar().ToString());

return temp;

}

catch

{

return 1012;

}

}

privatevoid dgvEmp\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

if (e.RowIndex >= 0 && e.RowIndex < dgvEmp.Rows.Count)

{

DataGridViewRow row = dgvEmp.Rows[e.RowIndex];

id\_emp = row.Cells[0].Value != null ? row.Cells[0].Value.ToString() : "";

tbFirst\_name.Text = row.Cells[1].Value != null ? row.Cells[1].Value.ToString() : "";

tbName.Text = row.Cells[2].Value != null ? row.Cells[2].Value.ToString() : "";

tbMiddle\_name.Text = row.Cells[3].Value != null ? row.Cells[3].Value.ToString() : "";

tbLogin.Text = row.Cells[4].Value != null ? row.Cells[4].Value.ToString() : "";

cbRole.Text = row.Cells[5].Value != null ? row.Cells[5].Value.ToString() : "";

bEdit\_Emp.Enabled = true;

bDel\_Emp.Enabled = true;

}

}

catch (Exception )

{

;

}

}

privatevoid ClearTbEmp()

{

tbFirst\_name.Clear();

tbName.Clear();

tbMiddle\_name.Clear();

tbLogin.Clear();

tbPass.Clear();

cbRole.SelectedIndex = -1;

}

privateint CheckNumUpd()

{

try

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT UserID FROM user WHERE UserLogin ='" + tbLogin.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int temp = Convert.ToInt32(com.ExecuteScalar().ToString());

if (temp == Convert.ToInt32(id\_emp))

{

return 1012;

}

else

{

return temp;

}

}

catch

{

return 1012;

}

}

privatevoid label12\_Click(object sender, EventArgs e)

{

}

privatevoid tabCat\_Click(object sender, EventArgs e)

{

}

}

}

**MenuAdmin.cs** – формаменюадминистратора

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassMenuAdmin : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

Products prod = newProducts();

Guide guide = newGuide();

Special spec = newSpecial();

ViewOrders vieword = newViewOrders();

public MenuAdmin()

{

InitializeComponent();

}

privatevoid MenuAdmin\_Load(object sender, EventArgs e)

{

FillLabel();

}

publicvoid FillLabel()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_query = "SELECT UserSurname, UserName, UserPatronymic FROM user WHERE UserID = @id\_user;";

MySqlCommand command = newMySqlCommand(sql\_query, connection.GetConnect());

command.Parameters.AddWithValue("@id\_user", AuthForm.id\_user);

using (MySqlDataReader reader = command.ExecuteReader())

{

if (reader.Read())

{

string first\_name = reader.GetString(0);

string name = reader.GetString(1);

string middle\_name = reader.GetString(2);

lFIO.Text = string.Format("{0} {1}. {2}.", first\_name, name[0], middle\_name[0]);

}

}

connection.CloseConnect();

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитевыйтиизучетнойзаписи?", "Выход", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

this.Close();

}

}

privatevoid bProducts\_Click(object sender, EventArgs e)

{

this.Visible = false;

prod.ShowDialog();

this.Visible = true;

}

privatevoid bGuide\_Click(object sender, EventArgs e)

{

this.Visible = false;

guide.ShowDialog();

this.Visible = true;

}

privatevoid bSpecial\_Click(object sender, EventArgs e)

{

this.Visible = false;

spec.ShowDialog();

this.Visible = true;

}

privatevoid bOrders\_Click(object sender, EventArgs e)

{

this.Visible = false;

vieword.ShowDialog();

this.Visible = true;

}

}

}

**MenuManager.cs** – формаменюменеджера

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassMenuManager : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

ViewProduct viewprod = newViewProduct();

NewOrder neworder = newNewOrder();

ViewOrders vieword = newViewOrders();

public MenuManager()

{

InitializeComponent();

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитевыйтиизучетнойзаписи?", "Выход", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

this.Close();

}

}

privatevoid bProducts\_Click(object sender, EventArgs e)

{

this.Visible = false;

viewprod.ShowDialog();

this.Visible = true;

}

privatevoid MenuManager\_Load(object sender, EventArgs e)

{

FillLabel();

}

publicvoid FillLabel()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_first\_name = "SELECT UserSurname FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com1 = newMySqlCommand(sql\_first\_name, connection.GetConnect());

string first\_name = com1.ExecuteScalar().ToString();

string sql\_name = "SELECT UserName FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com2 = newMySqlCommand(sql\_name, connection.GetConnect());

string name = com2.ExecuteScalar().ToString();

string sql\_middle\_name = "SELECT UserPatronymic FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com3 = newMySqlCommand(sql\_middle\_name, connection.GetConnect());

string middle\_name = com3.ExecuteScalar().ToString();

connection.CloseConnect();

lFIO.Text = "";

lFIO.Text = first\_name + " " + name[0] + ". " + middle\_name[0] + ". ";

}

privatevoid bNewOrder\_Click(object sender, EventArgs e)

{

this.Visible = false;

neworder.ShowDialog();

this.Visible = true;

}

privatevoid bOrders\_Click(object sender, EventArgs e)

{

this.Visible = false;

vieword.ShowDialog();

this.Visible = true;

}

}

}

**MenuUser.cs** – формаменюпользователя

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassMenuUser : Form

{

Products prod = newProducts();

ViewProduct vprod = newViewProduct();

NewOrder neword = newNewOrder();

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

public MenuUser()

{

InitializeComponent();

}

privatevoid MenuUser\_Load(object sender, EventArgs e)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_first\_name = "SELECT UserSurname FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com1 = newMySqlCommand(sql\_first\_name, connection.GetConnect());

string first\_name = com1.ExecuteScalar().ToString();

string sql\_name = "SELECT UserName FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com2 = newMySqlCommand(sql\_name, connection.GetConnect());

string name = com2.ExecuteScalar().ToString();

string sql\_middle\_name = "SELECT UserPatronymic FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com3 = newMySqlCommand(sql\_middle\_name, connection.GetConnect());

string middle\_name = com3.ExecuteScalar().ToString();

string sql\_role = "SELECT RoleName FROM role WHERE RoleID = " + AuthForm.role\_user + ";";

MySqlCommand com\_role = newMySqlCommand(sql\_role, connection.GetConnect());

lRole.Text = com\_role.ExecuteScalar().ToString();

connection.CloseConnect();

lFIO.Text = "";

lFIO.Text = first\_name + " " + name + " " + middle\_name + " ";

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

this.Close();

}

privatevoid button1\_Click(object sender, EventArgs e)

{

if (AuthForm.role\_user == 3)

{

this.Visible = false;

prod.ShowDialog();

this.Visible = true;

}

else

{

this.Visible = false;

vprod.ShowDialog();

this.Visible = true;

}

}

privatevoid pictureBox1\_Click(object sender, EventArgs e)

{

}

privatevoid bOrders\_Click(object sender, EventArgs e)

{

this.Visible = false;

neword.ShowDialog();

this.Visible = true;

}

}

}

**NewOrder.cs** – формазаказатовара

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassNewOrder : Form

{

privatestring cellErrorText;

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

OrderView orderview = newOrderView();

string id\_prod = "";

string id\_cart = "";

string id\_prod\_cart = "";

publicstaticstring[] art\_global;

publicstaticstring[] quan\_global;

publicstaticstring[] price\_global;

publicstaticint N\_global;

public NewOrder()

{

InitializeComponent();

}

privatevoid NewOrder\_Load(object sender, EventArgs e)

{

#region FIO

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_first\_name = "SELECT UserSurname FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com1 = newMySqlCommand(sql\_first\_name, connection.GetConnect());

string first\_name = com1.ExecuteScalar().ToString();

string sql\_name = "SELECT UserName FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com2 = newMySqlCommand(sql\_name, connection.GetConnect());

string name = com2.ExecuteScalar().ToString();

string sql\_middle\_name = "SELECT UserPatronymic FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com3 = newMySqlCommand(sql\_middle\_name, connection.GetConnect());

string middle\_name = com3.ExecuteScalar().ToString();

connection.CloseConnect();

lFIO.Text = first\_name + " " + name + " " + middle\_name + " ";

#endregion

lDate.Text = DateTime.Now.ToString("dd.MM.yyyy");

bAdd.Enabled = false;

bDel.Enabled = false;

CartFill();

labelFill();

}

publicvoid CartFill()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT idCart AS 'ИД', idUser AS 'useer', idProduct AS 'Артикул',

NameProduct AS 'Название', Quantity AS 'Количество', Price AS 'Цена' FROM cart WHERE idUser = " + AuthForm.id\_user + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvCart.DataSource = dt;

dgvCart.Columns[0].Visible = false;

dgvCart.Columns[1].Visible = false;

connection.CloseConnect();

}

publicstaticstring Count()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT SUM(Quantity) FROM cart WHERE idUser = " + AuthForm.id\_user + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

string count = com.ExecuteScalar().ToString();

connection.CloseConnect();

return count;

}

publicstaticstring Price()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT SUM(Quantity \* Price) FROM cart WHERE idUser = " + AuthForm.id\_user + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

string price = com.ExecuteScalar().ToString();

connection.CloseConnect();

return price;

}

privatevoid bAdd\_Click(object sender, EventArgs e)

{

try

{

if (CheckQun() == 1)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

MySqlTransaction trans = connection.GetConnect().BeginTransaction();

string sql1 = String.Format("UPDATE cart SET Quantity = (Quantity + {0}) WHERE idProduct = '{1}' AND idUser = {2};", tbCount.Text, id\_prod\_cart, AuthForm.id\_user);

MySqlCommand com1 = newMySqlCommand(sql1, connection.GetConnect());

com1.Transaction = trans;

string sql2 = String.Format("UPDATE product SET ProductQuantityInStock = (ProductQuantityInStock - {0}) WHERE ProductArticleNumber = '{1}';", tbCount.Text, id\_prod\_cart);

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

com2.Transaction = trans;

try

{

com1.ExecuteNonQuery();

com2.ExecuteNonQuery();

trans.Commit();

CartFill();

labelFill();

tbCount.Value = 1;

bDel.Enabled = false;

bAdd.Enabled = false;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

trans.Rollback();

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid butBack\_Click(object sender, EventArgs e)

{

this.Close();

}

privatebool CheckCartTovar()

{

try

{

bool flag = false;

string count = "";

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT quantity FROM cart WHERE id\_user = " + AuthForm.id\_user + " AND article = " + id\_prod + ";";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

try

{

count = com.ExecuteScalar().ToString();

flag = false;

}

catch

{

flag = true;

return flag;

}

return flag;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

returnfalse;

}

}

privatevoid labelFill()

{

if (dgvCart.Rows.Count == 0)

{

lCount.Text = "";

lPrice.Text = "";

}

else

{

lCount.Text = Count() + " шт.";

lPrice.Text = Price() + " руб.";

}

}

privateint CartAmount()

{

int ecx = 0;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("SELECT quantity FROM cart WHERE article = {0} AND id\_user = {1};", id\_prod, AuthForm.id\_user);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

ecx = Convert.ToInt32(com.ExecuteScalar().ToString());

connection.CloseConnect();

return ecx;

}

privateint ProdAmount()

{

int ecx = 0;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("SELECT quantity FROM products WHERE article = {0};", id\_prod);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

ecx = Convert.ToInt32(com.ExecuteScalar().ToString());

connection.CloseConnect();

return ecx;

}

privateint ProdAmountCart()

{

int ecx = 0;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("SELECT quantity FROM products WHERE article = {0};", id\_prod\_cart);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

ecx = Convert.ToInt32(com.ExecuteScalar().ToString());

connection.CloseConnect();

return ecx;

}

privatevoid bDel\_Click(object sender, EventArgs e)

{

if (dgvCart.Rows.Count == 0)

{

MessageBox.Show("Корзинапустая!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

int ecx = Convert.ToInt32(dgvCart.CurrentRow.Cells[4].Value) - Convert.ToInt32(tbCount.Value);

if (ecx < 0)

{

MessageBox.Show("Вкорзиненетстолькотовара!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

tbCount.Value = 1;

}

else

{

//int pRow = dgvCart.CurrentCell.RowIndex;

//int pColumn = dgvCart.CurrentCell.ColumnIndex;

//dgvCart.CurrentCell = dgvCart[pColumn, pRow];

if (ecx == 0)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

MySqlTransaction trans = connection.GetConnect().BeginTransaction();

string sql1 = String.Format("DELETE FROM cart WHERE idCart = {0};", id\_cart);

MySqlCommand com1 = newMySqlCommand(sql1, connection.GetConnect());

com1.Transaction = trans;

string sql2 = String.Format("UPDATE product SET ProductQuantityInStock = (ProductQuantityInStock + {0}) WHERE ProductArticleNumber = '{1}';", tbCount.Text, id\_prod\_cart);

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

com2.Transaction = trans;

try

{

com1.ExecuteNonQuery();

com2.ExecuteNonQuery();

trans.Commit();

CartFill();

labelFill();

tbCount.Value = 1;

bDel.Enabled = false;

bAdd.Enabled = false;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

trans.Rollback();

}

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

MySqlTransaction trans = connection.GetConnect().BeginTransaction();

string sql1 = String.Format("UPDATE cart SET Quantity = (Quantity - {0}) WHERE idProduct = '{1}' AND idUser = {2};", tbCount.Text, id\_prod\_cart, AuthForm.id\_user);

MySqlCommand com1 = newMySqlCommand(sql1, connection.GetConnect());

com1.Transaction = trans;

string sql2 = String.Format("UPDATE product SET ProductQuantityInStock = (ProductQuantityInStock + {0}) WHERE ProductArticleNumber = '{1}';", tbCount.Text, id\_prod\_cart);

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

com2.Transaction = trans;

try

{

com1.ExecuteNonQuery();

com2.ExecuteNonQuery();

trans.Commit();

CartFill();

labelFill();

tbCount.Value = 1;

bDel.Enabled = false;

bAdd.Enabled = false;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

trans.Rollback();

}

}

}

}

}

privatevoid dgvCart\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

id\_cart = dgvCart.Rows[e.RowIndex].Cells[0].Value.ToString();

bDel.Enabled = true;

bAdd.Enabled = true;

id\_prod\_cart = dgvCart.Rows[e.RowIndex].Cells[2].Value.ToString();

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid bClear\_Click(object sender, EventArgs e)

{

if (dgvCart.Rows.Count == 0)

{

MessageBox.Show("Корзинапустая!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

DialogResult dg = MessageBox.Show("Выдействительнохотитеполностьюочиститькорзину?", "Корзина", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dg == DialogResult.Yes)

{

try

{

DelAll();

CartFill();

labelFill();

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

privatevoid DelAll()

{

#region GetCount

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("SELECT COUNT(\*) FROM cart WHERE idUser = {0};", AuthForm.id\_user);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int N = Convert.ToInt32(com.ExecuteScalar());

#endregion

int[] vIdCart = newint[N];

string art = "";

int quan = 0;

int quan\_prod = 0;

int ecx = 0;

string sql\_id = "SELECT idCart FROM cart WHERE idUser = " + AuthForm.id\_user;

MySqlCommand com\_id = newMySqlCommand(sql\_id, connection.GetConnect());

MySqlDataReader reader\_id = com\_id.ExecuteReader();

while (reader\_id.Read())

{

vIdCart[ecx] = Convert.ToInt32(reader\_id[0].ToString());

ecx++;

}

connection.CloseConnect();

for(int i = 0; i < N; i++)

{

connection.OpenConnect();

string sql\_upd = "SELECT Quantity, idProduct FROM cart WHERE idCart = " + vIdCart[i];

MySqlCommand com\_upd = newMySqlCommand(sql\_upd, connection.GetConnect());

MySqlDataReader reader = com\_upd.ExecuteReader();

while (reader.Read())

{

quan = Convert.ToInt32(reader[0].ToString());

art = reader[1].ToString();

}

connection.CloseConnect();

connection.OpenConnect();

string sql\_prod = String.Format("SELECT ProductQuantityInStock FROM product WHERE ProductArticleNumber = '{0}';", art);

MySqlCommand com\_prod = newMySqlCommand(sql\_prod, connection.GetConnect());

quan\_prod = Convert.ToInt32(com\_prod.ExecuteScalar().ToString());

string sql\_prod\_upd = String.Format("UPDATE product SET ProductQuantityInStock = {0} WHERE ProductArticleNumber = '{1}';", quan + quan\_prod, art);

MySqlCommand com\_prod\_upd = newMySqlCommand(sql\_prod\_upd, connection.GetConnect());

com\_prod\_upd.ExecuteNonQuery();

string sql\_del = "DELETE FROM cart WHERE idCart = " + vIdCart[i];

MySqlCommand com\_del = newMySqlCommand(sql\_del, connection.GetConnect());

com\_del.ExecuteNonQuery();

connection.CloseConnect();

}

MessageBox.Show("Успешно! Теперькорзинапустая.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

privatevoid bOrder\_Click(object sender, EventArgs e)

{

if (dgvCart.Rows.Count == 0)

{

MessageBox.Show("Вкорзиненетниодноготовара! Дляоформлениязаказадобавьтехотябы 1 товарвкорзину.", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

orderview.ShowDialog();

CartFill();

labelFill();

}

}

privateint CheckQun()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = "SELECT ProductQuantityInStock FROM product WHERE ProductArticleNumber = '" + id\_prod\_cart + "';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int n = Convert.ToInt32(com.ExecuteScalar().ToString());

if (n <Convert.ToInt32(tbCount.Text))

{

MessageBox.Show("На складе не достаточно товара. Остаток товара: " + n, "Корзина", MessageBoxButtons.OK, MessageBoxIcon.Warning);

tbCount.Value = 1;

return 0;

}

else

{

return 1;

}

}

publicvoid AddProductOrder()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = "SELECT MAX(OrderID) FROM orderproduct;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int num = Convert.ToInt32(com.ExecuteScalar()) + 1;

string sql2 = "SELECT COUNT(\*) FROM cart WHERE idUser = " + AuthForm.id\_user + ";";

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

int N = Convert.ToInt32(com2.ExecuteScalar());

N\_global = N;

string[] art = newstring[N];

string[] count = newstring[N];

string[] price = newstring[N];

int i = 0;

string sql3 = "SELECT idProduct, Quantity, Price FROM cart WHERE idUser = " + AuthForm.id\_user + ";";

MySqlCommand com3 = newMySqlCommand(sql3, connection.GetConnect());

MySqlDataReader reader = com3.ExecuteReader();

while (reader.Read())

{

art[i] = reader[0].ToString();

count[i] = reader[1].ToString();

price[i] = reader[2].ToString();

i++;

}

art\_global = art;

quan\_global = count;

price\_global = price;

//Array.Copy(art, art\_global, N);

//Array.Copy(count, quan\_global, N);

//Array.Copy(price, price\_global, N);

connection.CloseConnect();

for (int j = 0; j < N; j++)

{

connection.OpenConnect();

string sql4 = String.Format("INSERT INTO orderproduct VALUES({0}, '{1}', {2});", num, art[j], count[j]);

MySqlCommand com4 = newMySqlCommand(sql4, connection.GetConnect());

com4.ExecuteNonQuery();

connection.CloseConnect();

}

for (int j = 0; j < N; j++)

{

connection.OpenConnect();

string sql4 = String.Format("DELETE FROM cart WHERE idUser = {0} AND idProduct = '{1}';", AuthForm.id\_user, art[j]);

MySqlCommand com4 = newMySqlCommand(sql4, connection.GetConnect());

com4.ExecuteNonQuery();

connection.CloseConnect();

}

bOrder.Enabled = false;

}

}

}

**OrderView.cs** – формаоформлениязаказа

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using Word = Microsoft.Office.Interop.Word;

namespace BeautyGlory

{

publicpartialclassOrderView : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

privatereadonlystring FileName = Directory.GetCurrentDirectory() + @"\check.docx";

int num\_order;

publicbool flagorder = false;

public OrderView()

{

InitializeComponent();

bAddOrder.Enabled = false;

}

privatevoid OrderView\_Load(object sender, EventArgs e)

{

#region FIO

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_first\_name = "SELECT UserSurname FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com1 = newMySqlCommand(sql\_first\_name, connection.GetConnect());

string first\_name = com1.ExecuteScalar().ToString();

string sql\_name = "SELECT UserName FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com2 = newMySqlCommand(sql\_name, connection.GetConnect());

string name = com2.ExecuteScalar().ToString();

string sql\_middle\_name = "SELECT UserPatronymic FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com3 = newMySqlCommand(sql\_middle\_name, connection.GetConnect());

string middle\_name = com3.ExecuteScalar().ToString();

connection.CloseConnect();

lFIO.Text = first\_name + " " + name + " " + middle\_name + " ";

#endregion

lDate.Text = DateTime.Now.ToString("dd.MM.yyyy");

FillLabels();

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

this.Close();

}

privatevoid FillLabels()

{

#region cmPPFill

cmPP.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM `orderpickuppoint`;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmPP.Items.Add(reader[1].ToString() + reader[2].ToString());

}

connect.CloseConnect();

#endregion

#region NumOrder

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql2 = @"SELECT MAX(OrderID) FROM orderproduct;";

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

string num = com2.ExecuteScalar().ToString();

connection.CloseConnect();

if (num == "")

{

num\_order = 1;

}

else

{

num\_order = Convert.ToInt32(num) + 1;

}

#endregion

groupBox1.Text = "Заказ № " + num\_order;

lDateD.Text = DateTime.Now.AddDays(5).ToString("dd.MM.yyyy");

lPrice.Text = "Итоговаястоимость: " + NewOrder.Price() + " руб.";

lCount.Text = "Количествотовара: " + NewOrder.Count() + " шт.";

}

privatevoid bAddOrder\_Click(object sender, EventArgs e)

{

try

{

DialogResult dg = MessageBox.Show("Выдействительнохотитеоформитьзаказномер - " + num\_order + "?", "Оформлениезаказа", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dg == DialogResult.Yes)

{

flagorder = true;

NewOrder fillorder = newNewOrder();

fillorder.AddProductOrder();

string date = DateTime.Now.ToString("yyyy-MM-dd");

string dated = DateTime.Now.AddDays(5).ToString("yyyy-MM-dd");

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("INSERT INTO `order` VALUES(null, {0}, {1}, '{2}', {3}, '{4}', {5}, 1);", num\_order, GetCode(), dated, GetPP(), date, AuthForm.id\_user);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MessageBox.Show("Успешнозаказоформлен!\nКоддляполучениязаказа: " + Convert.ToString(GetCode() - 1) + "\nЗаказбудетдоставлен - " + DateTime.Now.AddDays(5).ToString("dd.MM.yyyy г.") + "\nПоадресу: " + cmPP.Text, "Оформлениезаказа", MessageBoxButtons.OK, MessageBoxIcon.Information);

DialogResult result = MessageBox.Show("Желаетераспечататьчек? Длязаказа №" + num\_order + "?", "Оформлениезаказа", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (DialogResult.Yes == result)

{

CheckPrint();

this.Close();

}

else

{

this.Close();

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid CheckPrint()

{

try

{

#region Word

var wordApp = new Word.Application();

wordApp.Visible = false;

var wordd = wordApp.Documents.Open(FileName);

ReplaceWordStub("{num}", num\_order.ToString(), wordd);

ReplaceWordStub("{date}", DateTime.Now.ToString("dd.MM.yyyy"), wordd);

ReplaceWordStub("{date\_d}", DateTime.Now.AddDays(5).ToString("dd.MM.yyyy"), wordd);

for (int i = 0; i <NewOrder.N\_global; i++)

{

wordd.Bookmarks["info"].Range.Text = NewOrder.art\_global[i] + "; " + NewOrder.quan\_global[i] + "; " + NewOrder.price\_global[i] + "\n";

}

#region FIO

string FIO = "";

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = "SELECT UserSurname, UserName, UserPatronymic FROM user WHERE UserID = " + AuthForm.id\_user + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

FIO = reader[0].ToString() + " " + reader[1].ToString() + " " + reader[2].ToString();

}

#endregion

string[] sum\_itog = lPrice.Text.Split(' ');

string[] count\_itog = lCount.Text.Split(' ');

ReplaceWordStub("{sum\_itog}", sum\_itog[2], wordd);

ReplaceWordStub("{count\_itog}", count\_itog[2], wordd);

ReplaceWordStub("{address}", cmPP.Text, wordd);

ReplaceWordStub("{code}", Convert.ToString(GetCode() - 1), wordd);

ReplaceWordStub("{FIO}", FIO, wordd);

wordApp.Visible = true;

#endregion

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid ReplaceWordStub(string stubToReplace, string text, Word.Document wordDocument)

{

var range = wordDocument.Content;

range.Find.ClearFormatting();

range.Find.Execute(FindText: stubToReplace, ReplaceWith: text);

}

privatevoid cmPP\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cmPP.SelectedIndex > -1)

{

bAddOrder.Enabled = true;

}

}

privateint GetCode()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql2 = @"SELECT MAX(OrderCode) FROM `order`;";

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

int code = Convert.ToInt32(com2.ExecuteScalar()) + 1;

connection.CloseConnect();

return code;

}

privateint GetPP()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql2 = @"SELECT idPickupPoint FROM `orderpickuppoint` WHERE PostCode = '" + cmPP.Text + "';";

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

int idpp = Convert.ToInt32(com2.ExecuteScalar());

connection.CloseConnect();

return idpp;

}

}

}

**Products.cs** – формапросмотрапродуктов

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassProducts : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

string id = "";

bool flag\_product = false;

string art = "";

public Products()

{

InitializeComponent();

}

privatevoid Products\_Load(object sender, EventArgs e)

{

ProductFill();

Count();

cmCat\_Fill();

cmMan\_Fill();

cmSup\_Fill();

cmUnit\_Fill();

bEdit.Enabled = false;

bDel.Enabled = false;

pbPhoto.ImageLocation = "Photo/picture.png";

}

privatevoid ProductFillNew()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format(@"SELECT product.ProductArticleNumber AS 'Артикул',

product.ProductName AS 'Название', product.ProductDescription AS 'Описание',

product.ProductCost AS 'Цена', productcategory.Category AS 'Категория',

productmanufacturer.Manufacturer AS 'Производитель', productsupplier.Supplier AS 'Поставщик',

product.ProductQuantityInStock AS 'Количество', productunit.unit AS 'Ед.измерения',

product.ProductDiscountAmount AS 'Действующаяскидка', ProductMaxDiscount AS 'Максимальнаяскидка',

product.ProductPhoto AS 'Изображение' FROM product INNER JOIN productmanufacturer ON product.ProductManufacturer = productmanufacturer.idManufacturer

INNER JOIN productsupplier ON product.ProductSupplier = productsupplier.idSupplier INNER JOIN productunit ON product.ProductUnit = productunit.idUnit

INNER JOIN productcategory ON product.ProductCategory = productcategory.idCategory

ORDER BY CASE WHEN ProductArticleNumber = '{0}' THEN 0 ELSE 1 END, ProductArticleNumber;", art);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvProduct.DataSource = dt;

dgvProduct.Columns[2].Visible = false;

dgvProduct.Columns[8].Visible = false;

dgvProduct.Columns[10].Visible = false;

dgvProduct.Columns[11].Visible = false;

connection.CloseConnect();

FillCell();

}

publicvoid ProductFill()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT product.ProductArticleNumber AS 'Артикул',

product.ProductName AS 'Название', product.ProductDescription AS 'Описание',

product.ProductCost AS 'Цена', productcategory.Category AS 'Категория',

productmanufacturer.Manufacturer AS 'Производитель', productsupplier.Supplier AS 'Поставщик',

product.ProductQuantityInStock AS 'Количество', productunit.unit AS 'Ед.измерения',

product.ProductDiscountAmount AS 'Действующаяскидка', ProductMaxDiscount AS 'Максимальнаяскидка',

product.ProductPhoto AS 'Изображение' FROM product INNER JOIN productmanufacturer ON product.ProductManufacturer = productmanufacturer.idManufacturer

INNER JOIN productsupplier ON product.ProductSupplier = productsupplier.idSupplier INNER JOIN productunit ON product.ProductUnit = productunit.idUnit

INNER JOIN productcategory ON product.ProductCategory = productcategory.idCategory;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvProduct.DataSource = dt;

dgvProduct.Columns[2].Visible = false;

dgvProduct.Columns[8].Visible = false;

dgvProduct.Columns[10].Visible = false;

dgvProduct.Columns[11].Visible = false;

connection.CloseConnect();

FillCell();

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

ClearInfo();

this.Close();

}

privatevoid bAdd\_Click(object sender, EventArgs e)

{

try

{

if (tbName.Text == "" || tbDesc.Text == "" || cmMan.Text == "" || tbPrice.Text == "" || tbQuantly.Text == "" || cmCategory.Text == "" || tbPhoto.Text == "" || tbDiscMax.Text == "" || tbDisc.Text == "" || cmUnit.Text == "" || cmSupplier.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя!", "Добавление", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

if (Convert.ToInt32(tbDiscMax.Text) <Convert.ToInt32(tbDisc.Text))

{

MessageBox.Show("Ошибка! Максимальнаяскидканедолжнабытьменьшедействующей!", "Добавление", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

CheckProduct();

if (flag\_product == false)

{

MessageBox.Show("Ошибка! Такойтоваружезаведеннаскладе.", "Добавление", MessageBoxButtons.OK, MessageBoxIcon.Error);

ClearInfo();

}

else

{

#region genart

string abc = "QWERTYUIOPASDFGHJKLZXCVBNM";

Random rnd = newRandom();

art = "";

for (int i = 0; i < 6; i++)

{

if (i == 0 || i == 4)

{

art += abc[rnd.Next(1, abc.Length)];

}

else

{

art += rnd.Next(0, 9);

}

}

#endregion

int cat = get\_cmCat();

int man = get\_cmMan();

int sup = get\_cmSup();

int unit = get\_cmUnit();

string name = tbName.Text;

string desc = tbDesc.Text;

string price = tbPrice.Text.Replace(",", ".");

string quan = tbQuantly.Text;

string mdisc = tbDiscMax.Text;

string disc = tbDisc.Text;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format(@"INSERT INTO `product` VALUES ('{0}', '{1}', '{2}', {3}, '{4}', {5}, {6}, {7}, {8}, {9}, {10}, {11});", art, name, desc, cat, tbPhoto.Text, man, price, disc, quan, unit, sup, mdisc);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Успешно! Записьдобавлена.", "Добавление", MessageBoxButtons.OK, MessageBoxIcon.Information);

ClearInfo();

ProductFillNew();

Count();

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid Count()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT COUNT(\*) FROM product;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int count\_db = Convert.ToInt32(com.ExecuteScalar());

connection.CloseConnect();

int count\_dgv = dgvProduct.Rows.Count;

lCount.Text = String.Format("Количествозаписей: {0} из {1}", count\_dgv, count\_db);

}

privatevoid bDel\_Click(object sender, EventArgs e)

{

DialogResult result = MessageBox.Show("Выдействительнохотитеудалитьвыбраннуюзапись?", "Удаление", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (DialogResult.Yes == result)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"DELETE FROM product WHERE ProductArticleNumber = '" + id + "';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Записьуспешноудалена.", "Удаление", MessageBoxButtons.OK, MessageBoxIcon.Information);

ClearInfo();

ProductFill();

Count();

}

}

publicvoid ClearInfo()

{

tbDisc.Clear();

tbDiscMax.Clear();

cmUnit.SelectedIndex = -1;

tbName.Clear();

tbDesc.Clear();

cmMan.SelectedIndex = -1;

tbPrice.Clear();

tbQuantly.Clear();

tbPhoto.Clear();

cmCategory.SelectedIndex = -1;

cmSupplier.SelectedIndex = -1;

pbPhoto.ImageLocation = "Photo/picture.png";

}

privatevoid dgvProduct\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

bDel.Enabled = true;

bEdit.Enabled = true;

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

art = dgvProduct.Rows[e.RowIndex].Cells[0].Value.ToString();

tbName.Text = dgvProduct.Rows[e.RowIndex].Cells[1].Value.ToString();

tbDesc.Text = dgvProduct.Rows[e.RowIndex].Cells[2].Value.ToString();

tbPrice.Text = dgvProduct.Rows[e.RowIndex].Cells[3].Value.ToString();

cmMan.Text = dgvProduct.Rows[e.RowIndex].Cells[5].Value.ToString();

cmCategory.Text = dgvProduct.Rows[e.RowIndex].Cells[4].Value.ToString();

cmSupplier.Text = dgvProduct.Rows[e.RowIndex].Cells[6].Value.ToString();

tbQuantly.Text = dgvProduct.Rows[e.RowIndex].Cells[7].Value.ToString();

tbPhoto.Text = dgvProduct.Rows[e.RowIndex].Cells[11].Value.ToString();

cmUnit.Text = dgvProduct.Rows[e.RowIndex].Cells[8].Value.ToString();

tbDisc.Text = dgvProduct.Rows[e.RowIndex].Cells[9].Value.ToString();

tbDiscMax.Text = dgvProduct.Rows[e.RowIndex].Cells[10].Value.ToString();

pbPhoto.ImageLocation = "Photo/" + Convert.ToString(dgvProduct.Rows[e.RowIndex].Cells[11].Value.ToString());

}

catch (Exception)

{

;

}

}

privatevoid cmCat\_Fill()

{

cmCategory.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductCategory;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmCategory.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid cmUnit\_Fill()

{

cmUnit.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductUnit;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmUnit.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid cmMan\_Fill()

{

cmMan.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductManufacturer;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmMan.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid cmSup\_Fill()

{

cmSupplier.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductSupplier;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmSupplier.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

publicint get\_cmCat()

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT idCategory FROM productcategory WHERE Category = '" + cmCategory.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int id = Convert.ToInt32(com.ExecuteScalar());

connect.CloseConnect();

return id;

}

publicint get\_cmMan()

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT idManufacturer FROM productmanufacturer WHERE Manufacturer = '" + cmMan.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int id = Convert.ToInt32(com.ExecuteScalar());

connect.CloseConnect();

return id;

}

publicint get\_cmSup()

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT idSupplier FROM productsupplier WHERE Supplier = '" + cmSupplier.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int id = Convert.ToInt32(com.ExecuteScalar());

connect.CloseConnect();

return id;

}

publicint get\_cmUnit()

{

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT idUnit FROM productunit WHERE Unit = '" + cmUnit.Text + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

int id = Convert.ToInt32(com.ExecuteScalar());

connect.CloseConnect();

return id;

}

privatevoid bEdit\_Click(object sender, EventArgs e)

{

if (tbName.Text == "" || tbDesc.Text == "" || cmMan.Text == "" || tbPrice.Text == "" || tbQuantly.Text == "" || cmCategory.Text == "" || tbPhoto.Text == "" || tbDiscMax.Text == "" || tbDisc.Text == "" || cmUnit.Text == "" || cmSupplier.Text == "")

{

MessageBox.Show("Ошибка! Сначалазаполнитевсеполя!", "Редактирование", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

if (Convert.ToInt32(tbDiscMax.Text) <Convert.ToInt32(tbDisc.Text))

{

MessageBox.Show("Ошибка! Максимальнаяскидканедолжнабытьменьшедействующей!", "Добавление", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

int cat = get\_cmCat();

int man = get\_cmMan();

int sup = get\_cmSup();

int unit = get\_cmUnit();

string name = tbName.Text;

string desc = tbDesc.Text;

string price = tbPrice.Text.Replace(",", ".");

string quan = tbQuantly.Text;

string mdisc = tbDiscMax.Text;

string disc = tbDisc.Text;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format(@"UPDATE `product` SET ProductName = '{0}', ProductDescription = '{1}', ProductCategory = {2}, ProductPhoto = '{3}', ProductManufacturer = {4}, ProductCost = {5}, ProductDiscountAmount = {6}, ProductQuantityInStock = {7}, ProductUnit = {8}, ProductSupplier = {9}, ProductMaxDiscount = {10} WHERE ProductArticleNumber = '{11}';", name, desc, cat, tbPhoto.Text, man, price, disc, quan, unit, sup, mdisc, id);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

MessageBox.Show("Успешно! Записьобновлена.", "Редактирование", MessageBoxButtons.OK, MessageBoxIcon.Information);

ClearInfo();

ProductFillNew();

Count();

}

}

privatevoid bPhoto\_Click(object sender, EventArgs e)

{

try

{

using (OpenFileDialog upload = newOpenFileDialog())

{

upload.Filter = ".jpg, .png, .jpeg|\*.jpg;\*.png;\*.jpeg";

if (upload.ShowDialog() != DialogResult.OK)

{

return;

}

string path1 = upload.FileName;

string Filename = System.IO.Path.GetFileName(path1);

string path2 = Environment.CurrentDirectory + "\\Photo\\" + Filename;

File.Copy(path1, path2, true);

tbPhoto.Text = Filename;

pbPhoto.ImageLocation = Environment.CurrentDirectory + "\\Photo\\" + Filename;

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

publicvoid CheckProduct()

{

try

{

string name = tbName.Text;

string temp = "";

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT ProductArticleNumber FROM product WHERE ProductName ='" + name + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

try

{

temp = com.ExecuteScalar().ToString();

flag\_product = false;

}

catch

{

flag\_product = true;

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid tbName\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= 'a'&& e.KeyChar <= 'z' || e.KeyChar >= 'A'&& e.KeyChar <= 'Z' || e.KeyChar >= 'а'&& e.KeyChar <= 'я' || e.KeyChar >= 'А'&& e.KeyChar <= 'Я' || e.KeyChar == 8 || e.KeyChar == ' ')

{

return;

}

else

{

e.Handled = true;

}

}

privatevoid tbMan\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= 'a'&& e.KeyChar <= 'z' || e.KeyChar >= 'A'&& e.KeyChar <= 'Z' || e.KeyChar >= 'а'&& e.KeyChar <= 'я' || e.KeyChar >= 'А'&& e.KeyChar <= 'Я' || e.KeyChar == 8 || e.KeyChar == ' ')

{

return;

}

else

{

e.Handled = true;

}

}

privatevoid tbPrice\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= '0'&& e.KeyChar <= '9' || e.KeyChar == 8 || e.KeyChar == ' ' || e.KeyChar == ',')

{

return;

}

else

{

e.Handled = true;

}

}

privatevoid tbQuantly\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar >= '0'&& e.KeyChar <= '9' || e.KeyChar == 8 || e.KeyChar == ' ')

{

return;

}

else

{

e.Handled = true;

}

}

privatevoid FillCell()

{

for (int i = 0; i < dgvProduct.Rows.Count; i++)

{

if (Convert.ToInt32(dgvProduct.Rows[i].Cells[7].Value) == 0)

{

dgvProduct.Rows[i].Cells[7].Style.BackColor = Color.IndianRed;

}

}

}

privatevoid tbDiscMax\_TextChanged(object sender, EventArgs e)

{

try

{

if (Convert.ToInt32(tbDiscMax.Text) > 100 && tbDiscMax.Text.Length > 1)

{

tbDiscMax.Text = "100";

}

}

catch

{

}

}

privatevoid tbDisc\_TextChanged(object sender, EventArgs e)

{

try

{

if (Convert.ToInt32(tbDisc.Text) > 100 && tbDisc.Text.Length > 1)

{

tbDisc.Text = "100";

}

}

catch

{

}

}

privatevoid textBox1\_TextChanged(object sender, EventArgs e)

{

if (tbFind.Text.Length > 2)

{

//cmFiltr.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT product.ProductArticleNumber AS 'Артикул',

product.ProductName AS 'Название', product.ProductDescription AS 'Описание',

product.ProductCost AS 'Цена', productcategory.Category AS 'Категория',

productmanufacturer.Manufacturer AS 'Производитель', productsupplier.Supplier AS 'Поставщик',

product.ProductQuantityInStock AS 'Количество', productunit.unit AS 'Ед.измерения',

product.ProductDiscountAmount AS 'Действующаяскидка', ProductMaxDiscount AS 'Максимальнаяскидка',

product.ProductPhoto AS 'Изображение' FROM product INNER JOIN productmanufacturer ON product.ProductManufacturer = productmanufacturer.idManufacturer

INNER JOIN productsupplier ON product.ProductSupplier = productsupplier.idSupplier INNER JOIN productunit ON product.ProductUnit = productunit.idUnit

INNER JOIN productcategory ON product.ProductCategory = productcategory.idCategory WHERE product.ProductName LIKE '%" + tbFind.Text + "%' OR product.ProductArticleNumber LIKE '%" + tbFind.Text + "%';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvProduct.DataSource = dt;

dgvProduct.Columns[7].Visible = false;

connection.CloseConnect();

Count();

}

else

{

//cmFiltr.Enabled = true;

ProductFill();

Count();

}

}

privatevoid label14\_Click(object sender, EventArgs e)

{

}

privatevoid dgvProduct\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

**Program.cs** – главныйкласспрограммы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

staticclassProgram

{

///<summary>

/// The main entry point for the application.

///</summary>

[STAThread]

staticvoid Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(newAuthForm());

}

}

}

**Special.cs** – форма специальных возможностей

using Microsoft.Office.Interop.Word;

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassSpecial : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

publicstring path\_csv = "";

public Special()

{

InitializeComponent();

}

publicvoid Fill\_Import()

{

cmImport.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SHOW TABLES;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = null;

try

{

reader = com.ExecuteReader();

while (reader.Read())

{

cmImport.Items.Add(reader[0].ToString());

}

}

finally

{

if (reader != null)

{

reader.Close();

}

connect.CloseConnect();

}

}

publicvoid Fill\_Export()

{

cmExport.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SHOW TABLES;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = null;

try

{

reader = com.ExecuteReader();

while (reader.Read())

{

cmExport.Items.Add(reader[0].ToString());

}

}

finally

{

if (reader != null)

{

reader.Close();

}

connect.CloseConnect();

}

}

privatevoid Form1\_Load(object sender, EventArgs e)

{

Fill\_Export();

Fill\_Import();

}

privatevoid bPath\_Click(object sender, EventArgs e)

{

if (this.folderBrowserDialog1.ShowDialog() == DialogResult.OK)

{

tbExport.Text = this.folderBrowserDialog1.SelectedPath;

}

}

privatevoid bExport\_Click(object sender, EventArgs e)

{

if (tbExport.Text == "")

{

MessageBox.Show("Предупреждение! Выберите таблицу для экспорта.", "Экспорт БД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

if (cmExport.Text == "")

{

MessageBox.Show("Предупреждение! Выберите путь для экспорта.", "Экспорт БД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

if (cm1.Text == "")

{

MessageBox.Show("Предупреждение! Выберитеразделительдляэкспорта.", "ЭкспортБД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

try

{

string temp = tbExport.Text;

tbExport.Text = temp.Replace("\\", "//");

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM " + cmExport.Text + " INTO OUTFILE @file\_path FIELDS TERMINATED BY @delimiter;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

com.Parameters.AddWithValue("@file\_path", tbExport.Text + "//" + cmExport.Text + ".csv");

com.Parameters.AddWithValue("@delimiter", cm1.Text);

com.ExecuteNonQuery();

connect.CloseConnect();

cmExport.SelectedIndex = -1;

tbExport.Clear();

cm1.SelectedIndex = -1;

MessageBox.Show("Успешно! Экспортзавершен. Файлсохраненповыбранномупути.", "ЭкспортБД", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception msg)

{

MessageBox.Show("Возниклаошибка!" + msg.Message, "Ошибкапрограммы", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

privatevoid bFile\_Click(object sender, EventArgs e)

{

using (OpenFileDialog upload = newOpenFileDialog())

{

upload.Filter = "csv Files |\*.csv";

if (upload.ShowDialog() != DialogResult.OK)

{

return;

}

path\_csv = upload.FileName;

tbImport.Text = path\_csv;

string[] temp = path\_csv.Split('.');

string[] temp2 = temp[0].Split('\\');

cmImport.Text = temp2[1];

}

}

privatevoid bImport\_Click(object sender, EventArgs e)

{

try

{

if (string.IsNullOrEmpty(tbImport.Text))

{

MessageBox.Show("Предупреждение! Выберитефайлдляимпорта.", "ИмпортБД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (string.IsNullOrEmpty(cmImport.Text))

{

MessageBox.Show("Предупреждение! Выберитетаблицудляимпорта.", "ИмпортБД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (string.IsNullOrEmpty(cm2.Text))

{

MessageBox.Show("Предупреждение! Выберитеразделительдляимпорта.", "ИмпортБД", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

string temp = tbImport.Text;

tbImport.Text = temp.Replace("\\", "//");

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = string.Format("LOAD DATA INFILE '{0}' IGNORE INTO TABLE {1} FIELDS TERMINATED BY '{2}';", tbImport.Text, cmImport.Text, cm2.Text);

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

com.ExecuteNonQuery();

connect.CloseConnect();

cmImport.SelectedIndex = -1;

tbImport.Clear();

cm2.SelectedIndex = -1;

MessageBox.Show("Успешно! Импортзавершен.", "ИмпортБД", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception msg)

{

MessageBox.Show("Возниклаошибка! " + msg.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

ClearInfo();

this.Close();

}

publicvoid ClearInfo()

{

cmImport.SelectedIndex = -1;

cm2.SelectedIndex = -1;

cmExport.SelectedIndex = -1;

cm1.SelectedIndex = -1;

}

}

}

**ViewOrders.cs** – форма учета заказов

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Runtime.InteropServices;

namespace BeautyGlory

{

publicpartialclassViewOrders : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

int[] idUser;

publicstaticstring id\_order = "";

public ViewOrders()

{

InitializeComponent();

bDateOrderNew.Enabled = false;

var date = DateTime.Now.AddDays(5);

dtpDateUpd.MinDate = date;

}

privatevoid ViewOrders\_Load(object sender, EventArgs e)

{

try

{

lDate.Text = DateTime.Now.ToString("dd.MM.yyyy");

if (AuthForm.checkform == 14741)

{

bExcels.Visible = true;

}

GetRole();

dgvFillOrder();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

cmFillEmp();

FillCell();

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid dgvFillOrder()

{

try

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid FillCell()

{

for (int i = 0; i < dgvOrderAll.Rows.Count; i++)

{

if (Convert.ToString(dgvOrderAll.Rows[i].Cells[8].Value) == "Новый")

{

dgvOrderAll.Rows[i].Cells[8].Style.BackColor = Color.PaleGreen;

}

else

{

dgvOrderAll.Rows[i].Cells[8].Style.BackColor = Color.IndianRed;

}

}

}

privatevoid cbYes\_CheckedChanged(object sender, EventArgs e)

{

if (rbNew.Checked == true)

{

cmUser.Enabled = false;

bSort.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE orderstatus.Status = 'Новый';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

cmUser.Enabled = true;

bSort.Enabled = true;

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid bBack\_Click(object sender, EventArgs e)

{

rbNew.Checked = false;

rbFinish.Checked = false;

bSort.Enabled = false;

this.Close();

}

privatevoid cmFillEmp()

{

cmUser.Items.Clear();

cmUser.Items.Add("");

for (int i = 0; i < idUser.Length; i++)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT \* FROM user WHERE UserID = " + idUser[i] + ";";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

string s = reader[0].ToString() + " - " + reader[1].ToString() + " " + reader[2].ToString() + " " + reader[3].ToString() + " ";

cmUser.Items.Add(s);

}

connection.CloseConnect();

}

}

privatevoid cmUser\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cmUser.SelectedIndex == 0)

{

bSort.Enabled = true;

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

string n = cmUser.Text[0].ToString();

rbNew.Checked = false;

rbFinish.Checked = false;

bSort.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format(@"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE user.UserID = {0};", n);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid GetRole()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql\_count = @"SELECT COUNT(\*) FROM user WHERE UserRole = 2 OR UserRole = 1;";

MySqlCommand com\_count = newMySqlCommand(sql\_count, connection.GetConnect());

int N = Convert.ToInt32(com\_count.ExecuteScalar());

idUser = newint[N];

string sql\_id = @"SELECT UserID FROM user WHERE UserRole = 2 OR UserRole = 1;";

MySqlCommand com\_id = newMySqlCommand(sql\_id, connection.GetConnect());

MySqlDataReader reader = com\_id.ExecuteReader();

int i = 0;

while (reader.Read())

{

idUser[i] = Convert.ToInt32(reader[0].ToString());

i++;

}

connection.CloseConnect();

}

privatevoid dgvOrderAll\_CellMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

if (dgvOrderAll.Rows[e.RowIndex].Cells[8].Value.ToString() == "Новый")

{

DialogResult dg = MessageBox.Show("Выдействительнохотитезавершитьданныйзаказ?\n", "Завершениезаказа", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dg == DialogResult.Yes)

{

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

privatevoid button1\_Click(object sender, EventArgs e)

{

if (bSort.Text == "Сортировать")

{

rbNew.Checked = false;

rbFinish.Checked = false;

cmUser.Enabled = false;

bSort.Text = "Отменить";

string date = dtpDateSort.Value.ToString();

string[] date\_split = date.Split(' ');

var newdate = Convert.ToDateTime(date\_split[0]).ToString("yyyy.MM.dd");

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format(@"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE `order`.OrderDate = '{0}' ORDER BY `order`.OrderID;", newdate);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

cmUser.Enabled = true;

bSort.Text = "Сортировать";

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid dgvOrderAll\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

if (e.RowIndex >= 0 && e.RowIndex < dgvOrderAll.Rows.Count)

{

bDateOrderNew.Enabled = true;

id\_order = dgvOrderAll.Rows[e.RowIndex].Cells[0].Value.ToString();

}

}

catch (Exception)

{

;

}

bDateOrderNew.Enabled = true;

}

privatevoid bDateOrderNew\_Click(object sender, EventArgs e)

{

DialogResult dg = MessageBox.Show("Выдействительнохотитеизменитьдатудоставки?", "Информацияодоставке", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (dg == DialogResult.Yes)

{

string date = dtpDateUpd.Value.ToString();

string[] date\_split = date.Split(' ');

var newdate = Convert.ToDateTime(date\_split[0]).ToString("yyyy.MM.dd");

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = String.Format("UPDATE `order` SET OrderDeliveryDate = '{0}' WHERE OrderID = {1}", newdate, id\_order);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

connection.CloseConnect();

dgvFillOrderNew();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

bDateOrderNew.Enabled = false;

MessageBox.Show("Успешно! Дата доставки заказа перенесена на выбранную дату.", "Информация о доставке", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

privatevoid dgvFillOrderNew()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql =String.Format(@"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus

ORDER BY CASE WHEN `order`.OrderID = {0} THEN 0 ELSE 1 END, `order`.OrderID;", id\_order);

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

}

privatevoid radioButton1\_CheckedChanged(object sender, EventArgs e)

{

if (rbNew.Checked == true)

{

cmUser.Enabled = false;

bSort.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE orderstatus.Status = 'Новый';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

cmUser.Enabled = true;

bSort.Enabled = true;

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid radioButton2\_CheckedChanged(object sender, EventArgs e)

{

if (rbFinish.Checked == true)

{

cmUser.Enabled = false;

bSort.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE orderstatus.Status = 'Завершен';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

FillCell();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

cmUser.Enabled = true;

bSort.Enabled = true;

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid rbAll\_CheckedChanged(object sender, EventArgs e)

{

if (rbAll.Checked == true)

{

cmUser.Enabled = false;

bSort.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'OPD', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество', `order`.OrderCode AS 'Кодполучения',

`order`.OrderDeliveryDate AS 'Датадоставки', orderpickuppoint.PickupPoint AS 'Адресдоставки', `order`.OrderDate AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus WHERE orderstatus.Status IN ('Завершен', 'Новый');";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvOrderAll.DataSource = dt;

connection.CloseConnect();

FillCell();

dgvOrderAll.Columns[0].Visible = false;

dgvOrderAll.Columns[3].Visible = false;

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

else

{

cmUser.Enabled = true;

bSort.Enabled = true;

dgvFillOrder();

FillCell();

lCountOrder.Text = "Всегозаказов: " + dgvOrderAll.Rows.Count;

}

}

privatevoid bExcels\_Click(object sender, EventArgs e)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT `order`.OrderID AS 'Номерзаказа', orderproduct.ProductArticleNumber AS 'Артикул', orderproduct.OrderCount AS 'Количество',

CONCAT(EXTRACT(DAY FROM `order`.OrderDeliveryDate), '.', EXTRACT(MONTH FROM `order`.OrderDeliveryDate), '.', EXTRACT(YEAR FROM `order`.OrderDeliveryDate)) AS 'Датадоставки',

orderpickuppoint.PickupPoint AS 'Адресдоставки', CONCAT(EXTRACT(DAY FROM `order`.OrderDate), '.', EXTRACT(MONTH FROM `order`.OrderDate), '.', EXTRACT(YEAR FROM `order`.OrderDate)) AS 'Датазаказа',

CONCAT(user.UserSurname, ' ', user.UserName, ' ', user.UserPatronymic) AS 'Заказчик', orderstatus.Status AS 'Статусзаказа' FROM `order`

INNER JOIN orderproduct ON `order`.OrderProductID = orderproduct.OrderID INNER JOIN orderpickuppoint ON orderpickuppoint.idPickupPoint = `order`.OrderPickupPoint

INNER JOIN user ON user.UserID = `order`.OrderClient INNER JOIN orderstatus ON orderstatus.idStatus = `order`.OrderStatus;";

MySqlCommand cmd = newMySqlCommand(sql, connection.GetConnect());

cmd.ExecuteNonQuery();

MySqlDataAdapter da = newMySqlDataAdapter(cmd);

DataTable dt = newDataTable();

da.Fill(dt);

Microsoft.Office.Interop.Excel.Application ExcelApp = new Microsoft.Office.Interop.Excel.Application();

Microsoft.Office.Interop.Excel.Workbook ExcelWorkBook;

Microsoft.Office.Interop.Excel.Worksheet ExcelWorkSheet;

ExcelWorkBook = ExcelApp.Workbooks.Add(System.Reflection.Missing.Value);

ExcelWorkSheet = (Microsoft.Office.Interop.Excel.Worksheet)ExcelWorkBook.Worksheets.get\_Item(1);

Microsoft.Office.Interop.Excel.Range \_excelCells1 = (Microsoft.Office.Interop.Excel.Range)ExcelWorkSheet.get\_Range("A1", "H1").Cells;

\_excelCells1.Merge(Type.Missing);

ExcelApp.Cells[1, 1].EntireRow.Font.Size = 16;

ExcelApp.Cells[1, 1].EntireRow.Font.Bold = true;

ExcelApp.Cells[1, 1] = "Отчет по всем заказам товаров";

ExcelApp.Cells[1, 1].HorizontalAlignment = -4108;

ExcelApp.Cells[1, 1].VerticalAlignment = -4108;

Microsoft.Office.Interop.Excel.Range \_excelCells2 = (Microsoft.Office.Interop.Excel.Range)ExcelWorkSheet.get\_Range("A2", "H2").Cells;

\_excelCells2.Merge(Type.Missing);

ExcelApp.Cells[2, 1].EntireRow.Font.Bold = true;

ExcelApp.Cells[2, 1].EntireRow.Font.Size = 16;

ExcelApp.Cells[2, 1] = "от " + DateTime.Now.ToString("dd.MM.yyyy");

ExcelApp.Cells[2, 1].HorizontalAlignment = -4108;

ExcelApp.Cells[2, 1].VerticalAlignment = -4108;

ExcelApp.Range["A3:H3"].EntireRow.Font.Bold = true;

ExcelApp.Cells[3, 1] = "Номерзаказа";

ExcelApp.Cells[3, 2] = "Артикул";

ExcelApp.Cells[3, 3] = "Количество";

ExcelApp.Cells[3, 4] = "Дата доставки заказа";

ExcelApp.Cells[3, 5] = "Адрес доставки";

ExcelApp.Cells[3, 6] = "Дата заказа";

ExcelApp.Cells[3, 7] = "Заказчик";

ExcelApp.Cells[3, 8] = "Статус";

ExcelApp.Range["A1:H100" + (dt.Rows.Count + 2)].Borders.LineStyle = Microsoft.Office.Interop.Excel.XlLineStyle.xlContinuous;

ExcelApp.Range["A1:H100" + (dt.Rows.Count + 2)].Borders.Weight = Microsoft.Office.Interop.Excel.XlBorderWeight.xlThin;

ExcelApp.Range["A1:H100"].Font.Size = 14;

for (int i = 3; i < dt.Rows.Count + 1; i++)

{

for (int j = 0; j < dt.Columns.Count; j++)

{

if (i == 0)

{

ExcelApp.Cells[i + 1, j + 1] = dt.Rows[0].ItemArray[j].ToString();

}

else

{

ExcelApp.Cells[i + 1, j + 1] = dt.Rows[i - 1].ItemArray[j].ToString();

}

}

}

ExcelWorkSheet.Columns.AutoFit();

ExcelApp.Visible = true;

ExcelApp.UserControl = true;

ExcelApp.UserControl = true;

}

privatevoid dgvOrderAll\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

**ViewProduct.cs** – форма просмотра товаров

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace BeautyGlory

{

publicpartialclassViewProduct : Form

{

privateconstint CP\_NOCLOSE\_BUTTON = 0x200;

protectedoverrideCreateParams CreateParams

{

get

{

CreateParams myCp = base.CreateParams;

myCp.ClassStyle = myCp.ClassStyle | CP\_NOCLOSE\_BUTTON;

return myCp;

}

}

string id\_prod;

public ViewProduct()

{

InitializeComponent();

}

privatevoid ViewProduct\_Load(object sender, EventArgs e)

{

ProductFill();

Count();

cmCat\_Fill();

cmSup\_Fill();

cmMan\_Fill();

FillFilter();

pbPhoto.ImageLocation = "Photo/img\_holder.png";

bPhoto.Visible = false;

}

publicvoid ProductFill()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT product.ProductArticleNumber AS 'Артикул',

product.ProductName AS 'Название', product.ProductDescription AS 'Описание',

product.ProductCost AS 'Цена', productcategory.Category AS 'Категория',

productmanufacturer.Manufacturer AS 'Производитель', productsupplier.Supplier AS 'Поставщик',

product.ProductQuantityInStock AS 'Количество', productunit.unit AS 'Ед.измерения',

product.ProductDiscountAmount AS 'Действующаяскидка', ProductMaxDiscount AS 'Максимальнаяскидка',

product.ProductPhoto AS 'Изображение' FROM product INNER JOIN productmanufacturer ON product.ProductManufacturer = productmanufacturer.idManufacturer

INNER JOIN productsupplier ON product.ProductSupplier = productsupplier.idSupplier INNER JOIN productunit ON product.ProductUnit = productunit.idUnit

INNER JOIN productcategory ON product.ProductCategory = productcategory.idCategory;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvProduct.DataSource = dt;

dgvProduct.Columns[2].Visible = false;

dgvProduct.Columns[8].Visible = false;

dgvProduct.Columns[10].Visible = false;

dgvProduct.Columns[11].Visible = false;

connection.CloseConnect();

FillCell();

}

privatevoid Count()

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT COUNT(\*) FROM product;";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

int count\_db = Convert.ToInt32(com.ExecuteScalar());

connection.CloseConnect();

int count\_dgv = dgvProduct.Rows.Count;

lCount.Text = String.Format("Количествозаписей: {0} из {1}", count\_dgv, count\_db);

}

privatevoid cmCat\_Fill()

{

cmCategory.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductCategory;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmCategory.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

publicvoid ClearInfo()

{

tbDisc.Clear();

tbDiscMax.Clear();

tbName.Clear();

tbDesc.Clear();

cmMan.SelectedIndex = -1;

tbPrice.Clear();

tbQuantly.Clear();

tbPhoto.Clear();

cmCategory.SelectedIndex = -1;

cmSupplier.SelectedIndex = -1;

pbPhoto.ImageLocation = "Photo/img\_holder.png";

}

privatevoid cmMan\_Fill()

{

cmMan.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductManufacturer;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmMan.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid cmSup\_Fill()

{

cmSupplier.Items.Clear();

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT \* FROM ProductSupplier;";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

MySqlDataReader reader = com.ExecuteReader();

while (reader.Read())

{

cmSupplier.Items.Add(reader[1].ToString());

}

connect.CloseConnect();

}

privatevoid dgvProduct\_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

try

{

id\_prod = dgvProduct.Rows[e.RowIndex].Cells[0].Value.ToString();

tbName.Text = dgvProduct.Rows[e.RowIndex].Cells[1].Value.ToString();

tbDesc.Text = dgvProduct.Rows[e.RowIndex].Cells[2].Value.ToString();

tbPrice.Text = dgvProduct.Rows[e.RowIndex].Cells[3].Value.ToString();

cmMan.Text = dgvProduct.Rows[e.RowIndex].Cells[5].Value.ToString();

cmCategory.Text = dgvProduct.Rows[e.RowIndex].Cells[4].Value.ToString();

cmSupplier.Text = dgvProduct.Rows[e.RowIndex].Cells[6].Value.ToString();

tbQuantly.Text = dgvProduct.Rows[e.RowIndex].Cells[7].Value.ToString();

tbPhoto.Text = dgvProduct.Rows[e.RowIndex].Cells[11].Value.ToString();

tbDisc.Text = dgvProduct.Rows[e.RowIndex].Cells[9].Value.ToString();

tbDiscMax.Text = dgvProduct.Rows[e.RowIndex].Cells[10].Value.ToString();

pbPhoto.ImageLocation = "Photo/" + Convert.ToString(dgvProduct.Rows[e.RowIndex].Cells[11].Value.ToString());

// Путькбазеданных

string pathToDB = "host=localhost;uid=root;pwd=root;database=trade1;";

if (string.IsNullOrEmpty(pathToDB))

{

// Если путь к базе данных пустой, то подставляем фото-заглушку

pbPhoto.Image = Image.FromFile("Photo/img\_holder.png");

}

else

{

// Иначе загружаем данные из базы данных

}

}

catch (Exception)

{

;

}

}

privatevoid tbFind\_TextChanged(object sender, EventArgs e)

{

privatevoid bBack\_Click(object sender, EventArgs e)

{

ClearInfo();

this.Close();

}

privatevoid FillFilter()

{

privatevoid cmFiltr\_SelectedIndexChanged(object sender, EventArgs e)

{

privatevoid FillCell()

{

for (int i = 0; i < dgvProduct.Rows.Count; i++)

{

if (Convert.ToInt32(dgvProduct.Rows[i].Cells[7].Value) == 0)

{

dgvProduct.Rows[i].Cells[7].Style.BackColor = Color.IndianRed;

}

}

}

privatevoid dgvProduct\_ColumnHeaderMouseClick(object sender, DataGridViewCellMouseEventArgs e)

{

if (e.ColumnIndex >= 0 && e.ColumnIndex < dgvProduct.Columns.Count)

{

// код обработки щелчка по заголовку столбца

}

}

privatevoid label3\_Click(object sender, EventArgs e)

{

}

privatevoid dgvProduct\_CellMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

if (AuthForm.role\_user == 2 || AuthForm.role\_user == 1)

{

DialogResult result = MessageBox.Show("Выдействительнохотитедобавитьтоварвкорзину?", "Корзина", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes)

{

try

{

if (id\_prod == "")

{

MessageBox.Show("Ошибка! Сначалавыберитетовар!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

else

{

if (Convert.ToInt32(dgvProduct.CurrentRow.Cells[7].Value) == 0)

{

MessageBox.Show("Ксожалению, товарзакончилсянаскладе!", "Информация", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

else

{

if (CheckCartTovar() == false)

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

MySqlTransaction trans = connection.GetConnect().BeginTransaction();

string sql1 = String.Format("UPDATE cart SET Quantity = (Quantity + 1) WHERE idProduct = '{0}' AND idUser = {1};", id\_prod, AuthForm.id\_user);

MySqlCommand com1 = newMySqlCommand(sql1, connection.GetConnect());

com1.Transaction = trans;

string sql2 = String.Format("UPDATE product SET ProductQuantityInStock = (ProductQuantityInStock - 1) WHERE ProductArticleNumber = '{0}';", id\_prod);

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

com2.Transaction = trans;

try

{

com1.ExecuteNonQuery();

com2.ExecuteNonQuery();

trans.Commit();

ProductFill();

MessageBox.Show("Успешно! Товарсартикулом - " + id\_prod + ". Добавленвкорзинувколичестве 1-ойпозиции.", "Корзина", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

trans.Rollback();

}

}

else

{

db\_connect connection = newdb\_connect();

connection.OpenConnect();

MySqlTransaction trans = connection.GetConnect().BeginTransaction();

double price = Convert.ToDouble(tbPrice.Text);

double disc = Convert.ToInt32(tbDisc.Text) / 100.0;

price = price - (price \* disc);

string price2 = Convert.ToString((Math.Round(price, 2))).Replace(",", ".");

string sql1 = String.Format("INSERT INTO cart VALUES(null, {0}, '{1}', '{2}', {3}, 1);", AuthForm.id\_user, id\_prod, tbName.Text, price2);

MySqlCommand com1 = newMySqlCommand(sql1, connection.GetConnect());

com1.Transaction = trans;

string sql2 = String.Format("UPDATE product SET ProductQuantityInStock = (ProductQuantityInStock - 1) WHERE ProductArticleNumber = '{0}';", id\_prod);

MySqlCommand com2 = newMySqlCommand(sql2, connection.GetConnect());

com2.Transaction = trans;

try

{

com1.ExecuteNonQuery();

com2.ExecuteNonQuery();

trans.Commit();

ProductFill();

MessageBox.Show("Успешно! Товарсартикулом - " + id\_prod + ". Добавленвкорзинувколичестве 1-ойпозиции.", "Корзина", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

trans.Rollback();

}

}

}

}

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

privatebool CheckCartTovar()

{

try

{

bool flag = false;

string count = "";

db\_connect connect = newdb\_connect();

connect.OpenConnect();

string sql = "SELECT Quantity FROM cart WHERE idUser = " + AuthForm.id\_user + " AND idProduct = '" + id\_prod + "';";

MySqlCommand com = newMySqlCommand(sql, connect.GetConnect());

try

{

count = com.ExecuteScalar().ToString();

flag = false;

}

catch

{

flag = true;

return flag;

}

return flag;

}

catch (Exception msg)

{

MessageBox.Show("Ошибка! " + msg.Message, "Возниклаошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

returnfalse;

}

}

privatevoid textBox1\_TextChanged(object sender, EventArgs e)

{

if (tbFind.Text.Length > 2)

{

//cmFiltr.Enabled = false;

db\_connect connection = newdb\_connect();

connection.OpenConnect();

string sql = @"SELECT product.ProductArticleNumber AS 'Артикул',

product.ProductName AS 'Название', product.ProductDescription AS 'Описание',

product.ProductCost AS 'Цена', productcategory.Category AS 'Категория',

productmanufacturer.Manufacturer AS 'Производитель', productsupplier.Supplier AS 'Поставщик',

product.ProductQuantityInStock AS 'Количество', productunit.unit AS 'Ед.измерения',

product.ProductDiscountAmount AS 'Действующаяскидка', ProductMaxDiscount AS 'Максимальнаяскидка',

product.ProductPhoto AS 'Изображение' FROM product INNER JOIN productmanufacturer ON product.ProductManufacturer = productmanufacturer.idManufacturer

INNER JOIN productsupplier ON product.ProductSupplier = productsupplier.idSupplier INNER JOIN productunit ON product.ProductUnit = productunit.idUnit

INNER JOIN productcategory ON product.ProductCategory = productcategory.idCategory WHERE product.ProductName LIKE '%" + tbFind.Text + "%' OR product.ProductArticleNumber LIKE '%" + tbFind.Text + "%';";

MySqlCommand com = newMySqlCommand(sql, connection.GetConnect());

com.ExecuteNonQuery();

MySqlDataAdapter adapter = newMySqlDataAdapter(com);

DataTable dt = newDataTable();

adapter.Fill(dt);

dgvProduct.DataSource = dt;

dgvProduct.Columns[7].Visible = false;

connection.CloseConnect();

Count();

}

else

{

//cmFiltr.Enabled = true;

ProductFill();

Count();

}

}

}

}