Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**КУРСОВОЙ ПРОЕКТ**

Тема: Разработка программного модуля для учета продукции фармацевтической компании

МДК 01.01 «Разработка программных модулей»

**Выполнил:**

Стародубцев Максим Сергеевич

студент группы ИСП.20А

09.02.07 Информационные системы и программирование

очной формы обучения

**Руководитель:**

Кузьмина Елена Евгеньевна­­­­­­­­­­­­­­­­­­

Оценка

\_\_\_\_\_\_\_\_\_/\_\_\_­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

Подпись руководителя

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оглавление

[**Введение** 3](#_Toc131524741)

[1. Назначение разработки 4](#_Toc131524742)

[2. Требования к программному модулю 5](#_Toc131524743)

[2.1 Требования к функциональным характеристикам 5](#_Toc131524744)

[2.2 Требования к составу и параметрам технических средств 5](#_Toc131524745)

[2.3 Требования к информационной и программной совместимости 6](#_Toc131524746)

[3. Руководство программиста 6](#_Toc131524747)

[3.1 Разработка базы данных 6](#_Toc131524748)

[3.2 Разработка программного модуля 15](#_Toc131524749)

[4. Руководство пользователя 27](#_Toc131524750)

[4.1 Выполнение программного модуля 27](#_Toc131524751)

[4.2 Сообщения пользователю 32](#_Toc131524752)

[5.Тестирование программного модуля 33](#_Toc131524753)

[5.1Модульное тестирование 33](#_Toc131524754)

[**5.2Тестовые сценарии** 35](#_Toc131524755)

[**Заключение** 40](#_Toc131524756)

[Список литературы: 41](#_Toc131524757)

**Введение**

Важной частью здравоохранения является фармацевтическая служба, призванная обеспечить население своевременной и высококачественной лекарственной помощью. Задачей фармацевтической службы является более полное удовлетворение потребностей населения и ЛПУ в лекарственных средствах, развитии сети аптечных учреждений, повышении уровня подготовки и совершенствовании фармацевтических кадров.

Общедоступность лекарственной помощи обеспечивается развитием сети аптек и аптечных пунктов их максимальное приближении к населению.

Совместная деятельность врачей и провизоров по определению потребности в лекарственных средствах и распределению их ресурсов, систематическое информирование врачей о наличии медикаментов и перспективах снабжения ими улучшает обеспечение лекарствами населения.

Для обеспечения аптек ЛС существует сеть фармацевтический дистрибьюторов, которые имеют современно организованные аптечные склады, для хранения медицинских товаров.

Современный аптечный склад - это сложное техническое сооружение, которое имеет определенную структуру. Складские помещения располагают таким образом, чтобы обеспечивалась функциональная взаимосвязь между всеми участками склада.

1. Назначение разработки

Разработка проводится на основании приказа о курсовом проектировании. Аптечный склад РУП "Аптечный мир" осуществляет оптовую продажу лекарственных препаратов различным аптекам республики. аптечного склада, получающей запросы на лекарства от аптеки. Запросы рассматриваются администрацией аптечного склада с использованием информации об аптеках и лекарствах. При этом проверяется и обновляется список лекарств, проверяются записи об аптеках. Если требуемое лекарство имеется в наличии, администрация информирует клиента о стоимости. Аптечный склад получает от производителей новые лекарства. При их поступлении в аптечный склад, необходимая информация о новых лекарствах фиксируется. Информация об аптеках содержится отдельно от записей об лекарстве. Администрация склада регулярно готовит отчеты за определенный период времени об аптеках, производителях, лекарствах, о продаже лекарств и об их приобретении. Информационная система может быть описана данными, сгруппированными следующим образом:

* Лекарства (код лекарства, название лекарства, код производителя, код типа лекарства, цена(руб.)),
* Аптека (номер аптеки, название аптеки, адрес аптеки, номер телефона),
* Заявка (номер заявки, дата составления заявки, номер аптеки, дата выполнения заявки),
* Закупка лекарственных препаратов (номер заявки, код лекарства, количество (шт.)),
* Производитель (код производителя, ФИО, номер телефона, адрес),
* Тип лекарства (код типа лекарства, тип),
* Корзина (код корзины, код лекарства, количество, сессия).

1. Требования к программному модулю
   1. Требования к функциональным характеристикам

Приложение должно выполнять следующие функции:

1) Поиск данных (Поиск товара по типу).

2) Фильтрацию данных (Фильтрация лекарства по типу).

3) Добавление данных (Добавление нового товара).

4) Удаление данных (Удаление данных о лекарствах).

5) Редактирование данных (Редактирование данных о товаре).

6) Сортировку данных (Сортировка товара от А до Я).

7) Вывод на печать в Excel (Полный список товаров).

8)Кнопка возвращение на прошлую страницу.

9)Загрузка и отчистка картинки (лекарства).

* 1. Требования к составу и параметрам технических средств

Таблица№1 «Минимальные характеристики компьютера»

|  |  |
| --- | --- |
| Процессор | Ryzen 5 2600 |
| ОЗУ | 4 |
| Разрешение экрана | От 1920x1080 |
| Устройства ввода | Клавиатура, мышь |
| Операционная система | Windows 10 |
| Видеокарта | GIGABYTE GeForce GTX 760 |
| Место на жестком диске | От 30 Гб |

Таблица№2 «Оптимальный характеристики компьютера»

|  |  |
| --- | --- |
| Процессор | Ryzen 5 3600 |
| ОЗУ | 8 |
| Разрешение экрана | От 1920x1080 |
| Устройства ввода | Клавиатура, мышь |
| Операционная система | Windows 10 |
| Видеокарта | GIGABYTE GeForce GTX 1660 |
| Место на жестком диске | До 80 Гб |

2.3 Требования к информационной и программной совместимости

Для корректной работы программы необходимо: Иметь установленные приложения и программы: Visual Studio 2019-2022, Microsoft Excel, Microsoft SQL Server Management Studio а также доступ к интернету.

1. Руководство программиста
   1. Разработка базы данных

Для проекта база данных была разработана в СУБД Microsoft SQL Server Management Studio. В процессе проектирования ERD модели БД были выделены следующий сущности: Application, Basket, Manufacturer, Medicament, Order, Pharmacy, Purchase, Type.

Описание сущностей:

Application – хранит данные о заявке.

Basket – хранит данные о корзине товаров.

Manufacturer – хранит данные о производителе.

Medicament – хранит данные о лекарствах.

Order – хранит данные о покупке лекарства.

Pharmacy – хранит данные о аптеках.

Purchase– хранит данные о заявлении.

Type – хранит данные о виде лекарства.

Таблица№3 «Application»

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDapplication | INT | PK NN AI | Уникальный номер заявки |
| DateApplication | NVARCHAR (50) | NN | Дата составления заявки |
| IDpharmacy | INT | NN | Номер аптеки |
| DateCompletion | NVARCHAR (50) | NN | Дата выполнения заявки |

Таблица№4 «Basket»

|  |  |  |  |
| --- | --- | --- | --- |
| **Basket** | | | |
| Поле | Тип | Атрибуты | Описание |
| ID | INT | PK NN AI | Уникальный номер заявки |
| IDmedicament | INT | NN | Номер лекарства |
| Kolvo | INT | NN | Количество лекарства |
| Sessoin | INT | NN | Сессия |

Таблица№5 «Manufacturer»

|  |  |  |  |
| --- | --- | --- | --- |
| **Manufacturer** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDmanufacturer | INT | PK NN AI | Уникальный номер производители |
| FIO | NVARCHAR (MAX) | NN | ФИО |
| Phone | NVARCHAR (50) | NN | Номер телефона |
| Address | NVARCHAR (MAX) | NN | Адрес производителя |

Таблица№6 «Medicament»

|  |  |  |  |
| --- | --- | --- | --- |
| **Medicament** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDmedicament | INT | PK NN AI | Уникальный номер лекарства |
| NazvanieMedicament | NVARCHAR(50) | NN | Название лекарства |
| IDmanufacturer | INT | NN | Номер производителя |
| IDtype | INT | NN | Номер типа лекарства |
| Price | INT | NN | Цена |
| Image | NVARCHAR(MAX) | NN | Картинка лекарства |

Таблица№7 «Order»

|  |  |  |  |
| --- | --- | --- | --- |
| **Order** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDOrder | INT | PK NN AI | Уникальный номер аптеки |
| IDBasket | INT | NN | Номер корзины |
| DateOrder | DATE | NN | Дата заказа |

Таблица№8 «Pharmacy»

|  |  |  |  |
| --- | --- | --- | --- |
| **Pharmacy** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDpharmacy | INT | PK NN AI | Уникальный номер аптеки |
| NazvaniePharmacy | NVARCHAR(MAX) | NN | Название аптеки |
| PharmacyAddress | NVARCHAR(MAX) | NN | Адрес аптеки |
| PhonePharmacy | NVARCHAR(50) | NN | Номер телефона |

Таблица№9 «Purchase»

|  |  |  |  |
| --- | --- | --- | --- |
| **Purchase** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDpurchase | INT | PK NN AI | Уникальный номер закупки лекарственных препаратов |
| IDapplication | INT | NN | Номер заявки |
| IDmedicament | INT | NN | Номер лекарства |
| Quantity | INT | NN | Количество лекарств |

Таблица№10 «Type»

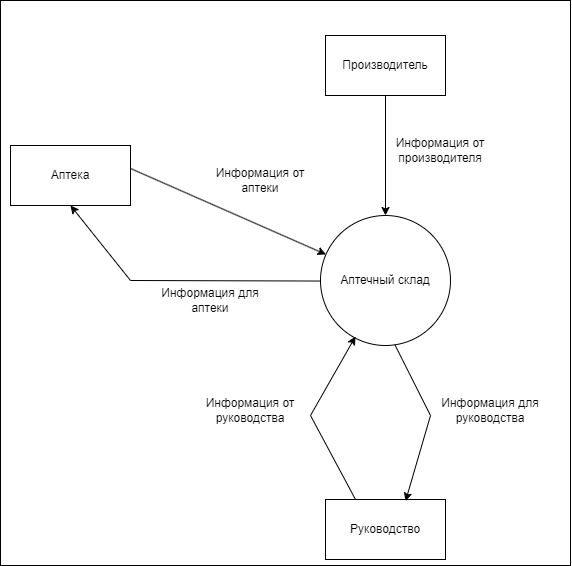
|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | | | |
| Поле | Тип | Атрибуты | Описание |
| IDtype | INT | PK NN AI | Уникальный номер типа лекарств |
| NazvanieType | NVARCHAR(50) | NN | Тип лекарства |

Для соединения таблиц использовались следующие типы связей:

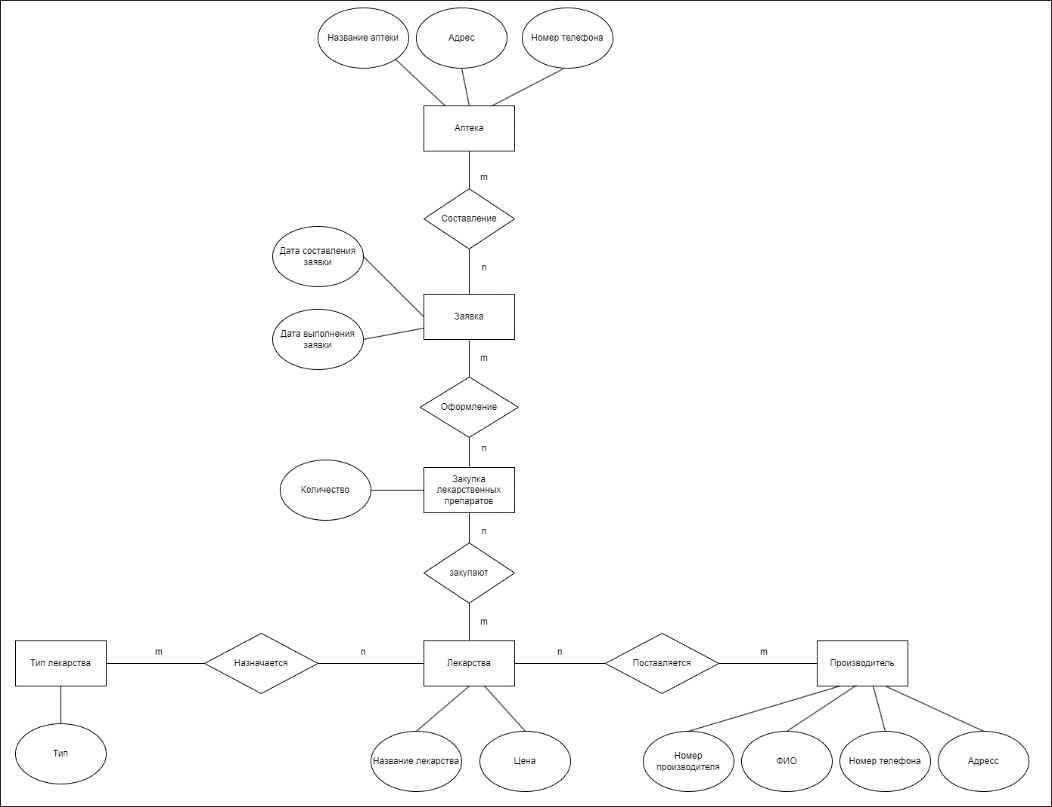
 - один ко многим вида: not-**identifying relationship**

Заполнение базы данных

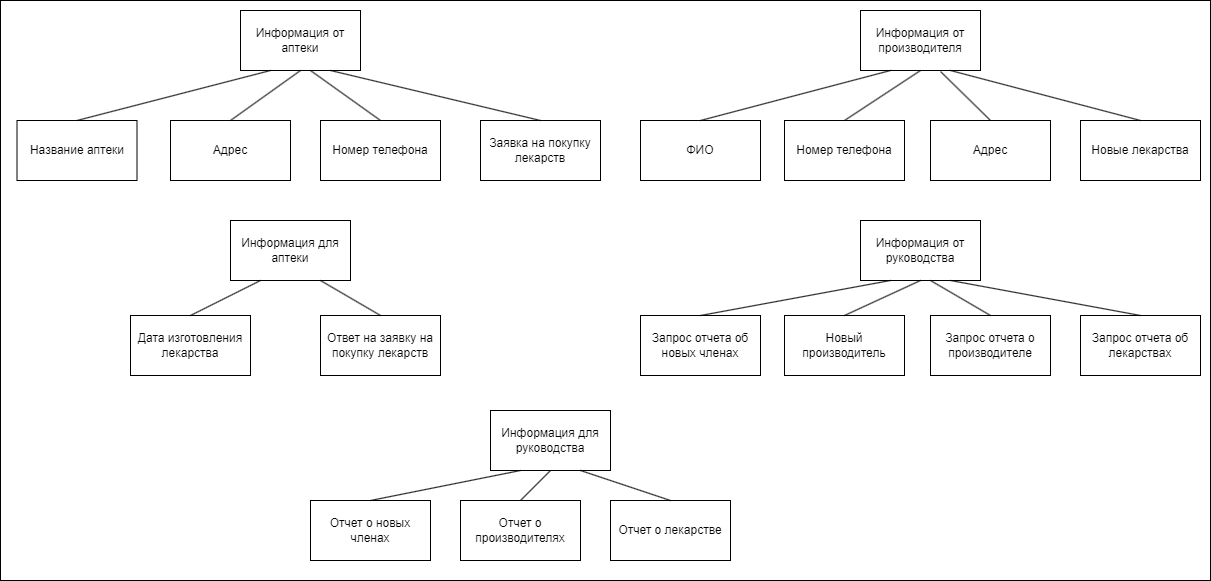
Диаграммы:

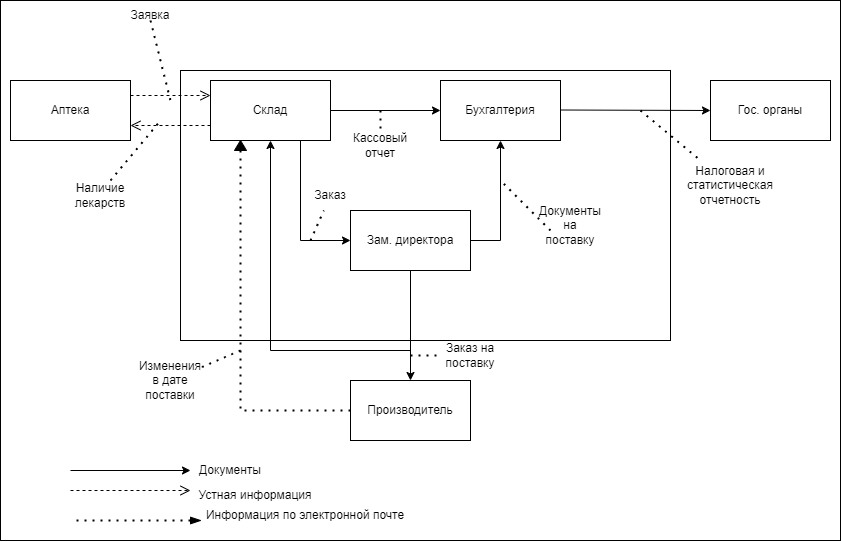


Рис№1 «Начальная контекстная диаграмма»

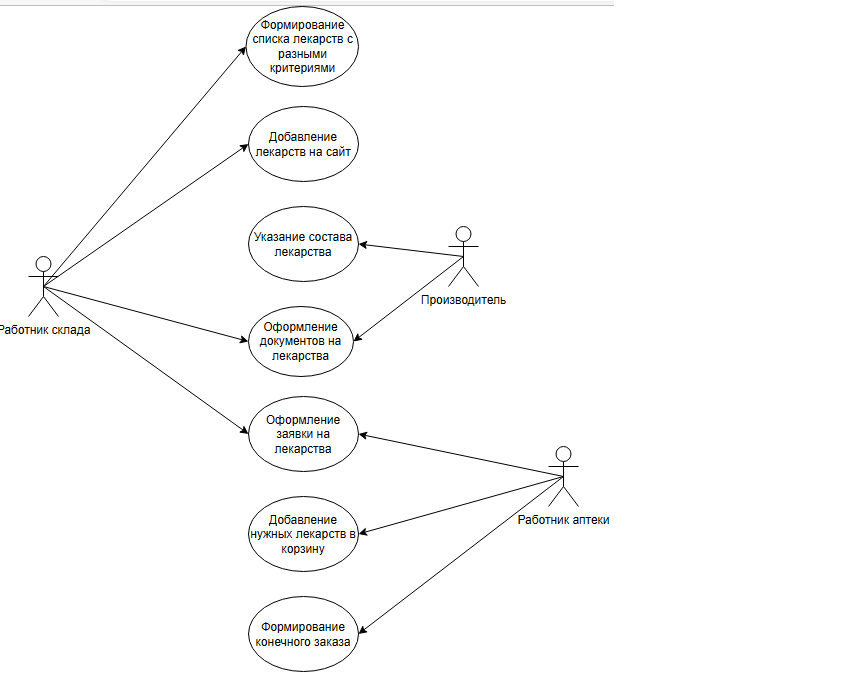


Рис№2 «Диаграмма "сущность-связь"»

 Рис№3 «Диаграмма структур данных»

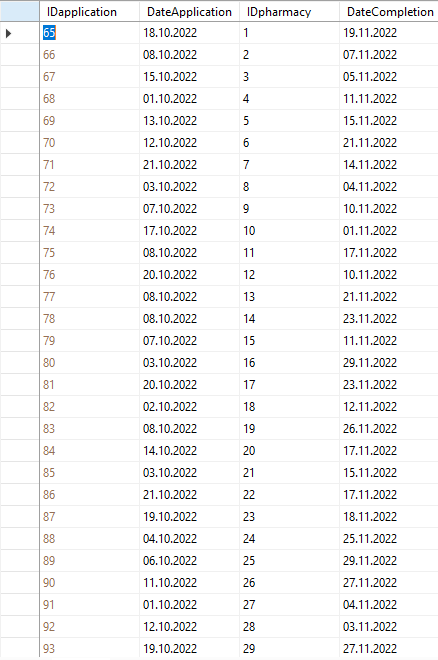


Рис№4 «Информационная модель»



Рис№5 «UML диаграмма»

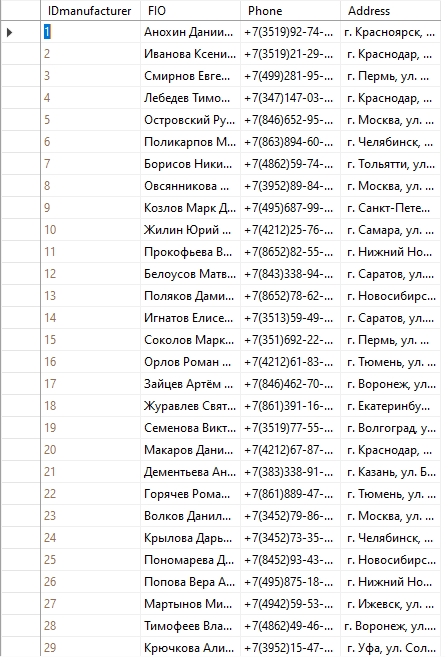
Заполнение базы данных



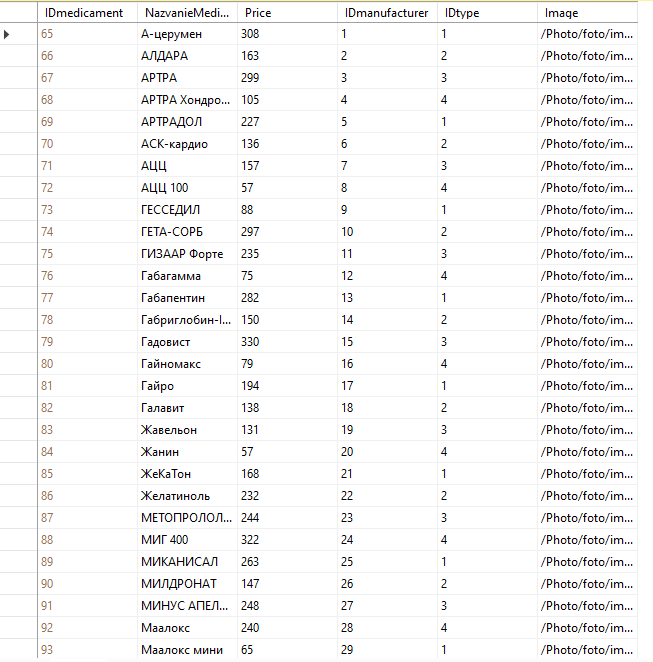
Рис№6 «Application»



Рис№7 «Basket»



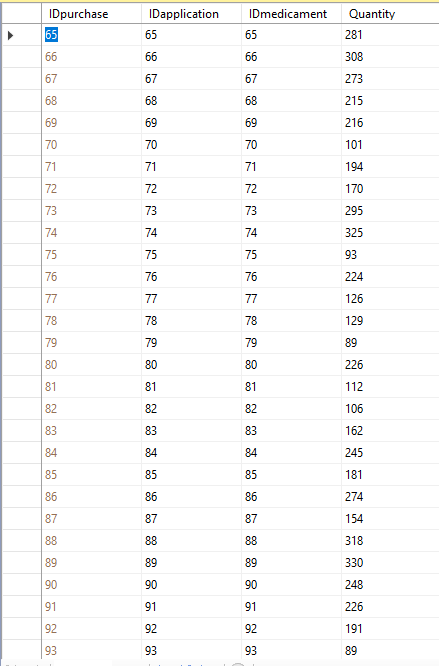
Рис№8 «Manufacturer»



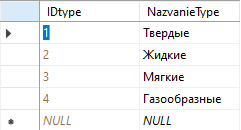
Рис№9 «Medicament»



Рис№10 «Pharmacy»



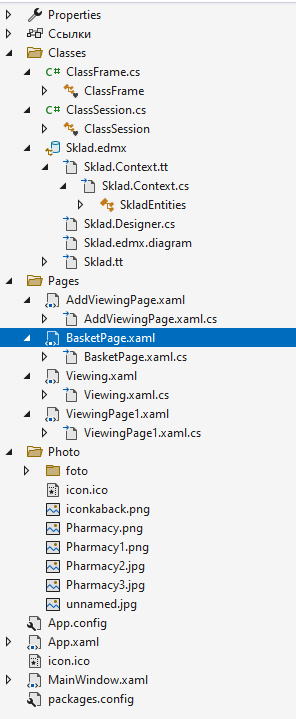
Рис№11 «Purchase»



Рис№12 «Type»

* 1. Разработка программного модуля

Структура проекта для приложения выглядит так:



Рис№13 «Структура проекта»

**Main Window.xaml (Содержит интерфейс главной страницы)**

<Window x:Class="sklad.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:sklad"

mc:Ignorable="d"

Title="Аптечный склад" d:DesignHeight="800" d:DesignWidth="1500">

<Grid>

<Frame Grid.Row="1"

Name="frmMain"

NavigationUIVisibility="Hidden">

</Frame>

</Grid>

</Window>

**Main Window.xaml.cs**

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

//MessageBox.Show( SkladEntities.GetContext().Basket.Max(x=>x.Sessoin).ToString());

ClassSession.idSession = (int)SkladEntities.GetContext().Basket.Max(x => x.Sessoin);

ClassFrame.frmObj = frmMain;

frmMain.Navigate(new Pages.Viewing());

}

}

В приложении было создано три основных папки: Classes, Pages, Photo.

Папка Classes содержит в себе:

**ClassFrame.cs (Для загрузки страниц)**

namespace sklad.Classes

{

internal class ClassFrame

{

public static Frame frmObj;

}

}

**ClassSession.cs(Для сохранения сессии в БД)**

namespace sklad.Classes

{

internal class ClassSession

{

public static int idSession;

public static int countMed;

}

}

**Sklad (Подключенная база данных).** **EntityFramework — это набор технологий в ADO.NET, которые поддерживают разработку программных приложений, ориентированных на данные.**

public partial class SkladEntities : DbContext

{

private static SkladEntities \_Context;

public SkladEntities()

: base("name=SkladEntities")

{

}

public static SkladEntities GetContext()

{

if (\_Context == null)

\_Context = new SkladEntities();

return \_Context;

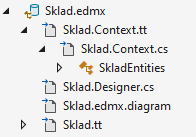
}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

****

Рисунок№14 «Sklad»

Папка Photo содержит в себе: (Для хранения изображении)

**Icon.icon**



Рисунок№15 «Логотип приложения»

Папка Pages содержит в себе: (Для хранения страниц)

**ViewingPage1.xaml**

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="1\*"/>

<RowDefinition Height="8\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="15\*"/>

</Grid.ColumnDefinitions>

<Image Source="/Photo/Pharmacy1.png" Margin="0,-8,0,-8" Stretch="Fill" Grid.ColumnSpan="3" Grid.RowSpan="3"/>

<Button x:Name="Back" Click="Back\_Click" Width="45" Height="45">

<Button.Background>

<ImageBrush ImageSource="/Photo/iconkaback.png"/>

</Button.Background>

</Button>

<StackPanel Orientation="Horizontal" Grid.ColumnSpan="3"

HorizontalAlignment="Center">

<TextBlock Text="Фильтр по типу"

FontSize="20"

Height="30"

Width="150"

Margin="5"/>

<ComboBox Name="CmbFiltr"

SelectionChanged="CmbFiltr\_SelectionChanged" Width="200" Height="40">

<ComboBoxItem Content="Твёрдый"/>

<ComboBoxItem Content="Жидкий"/>

<ComboBoxItem Content="Мягкий"/>

<ComboBoxItem Content="Газообразный"/>

</ComboBox>

<Button Name="BtnResetAll"

Height="40"

Width="200"

Margin="5"

Content="Сбросить"

FontSize="20"

Click="BtnResetAll\_Click"/>

<TextBlock Text="Поиск"

FontSize="20"

Width="70"

Height="40"

Margin="5"/>

<TextBox Width="200"

Height="40"

Name="TxtSearch"

TextChanged="TxtSearch\_TextChanged"/>

<TextBlock Text="Сортировка"

Height="40"

Width="200"

Margin="5"

FontSize="20"

HorizontalAlignment="Center"/>

<StackPanel Orientation="Vertical">

<RadioButton Content="От А до Я"

Height="40"

Width="200"

Margin="5"

FontSize="20"

x:Name="RbUp"

Checked="RbUp\_Checked"/>

<RadioButton Content="От Я до А"

Height="40"

Width="200"

Margin="5"

FontSize="20"

x:Name="RbDown"

Checked="RbDown\_Checked"/>

</StackPanel>

</StackPanel>

<Border Height="600" Width="1300" CornerRadius="30" Margin="15" Background="White" VerticalAlignment="Center" Padding="30" Grid.Row="1" Grid.ColumnSpan="3">

<Border.Effect>

<DropShadowEffect BlurRadius="30" Color="Purple" ShadowDepth="0">

</DropShadowEffect>

</Border.Effect>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="auto"></RowDefinition>

<RowDefinition Height="\*"></RowDefinition>

</Grid.RowDefinitions>

<ListView Grid.Row="1" Name="LviewMed">

<ListView.ContextMenu>

<ContextMenu >

<MenuItem Header="Удалить" Name="BtnDelete" Click="BtnDelete\_Click" />

<MenuItem Header="Добавить" Name="BtnAdd" Click="BtnAdd\_Click" />

<MenuItem Header="Печать" Name="BtnSaveToExcel" Click="BtnSaveToExcel\_Click" />

</ContextMenu>

</ListView.ContextMenu>

<ListView.ItemTemplate>

<DataTemplate>

<Grid Margin="20" Width="1150">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="2\*"/>

<ColumnDefinition Width="5\*"/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0" HorizontalAlignment="Left">

<Border BorderBrush="Orange" BorderThickness="2" CornerRadius="4">

<Image Width="300" Height="300" Grid.Row="1" Stretch="Fill" HorizontalAlignment="Center" Margin="5">

<Image.Source>

<Binding Path="Image">

<Binding.TargetNullValue>

<ImageSource>/Photo/unnamed.jpg</ImageSource>

</Binding.TargetNullValue>

</Binding>

</Image.Source>

</Image>

</Border>

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Column="1" VerticalAlignment="Top" HorizontalAlignment="Center">

<TextBlock Text="{Binding NazvanieMedicament}" FontSize="30" FontWeight="Bold" Foreground="Orange"></TextBlock>

<TextBlock Text=": " FontSize="30" FontWeight="Bold" Foreground="Black"></TextBlock>

<TextBlock Text="{Binding Type.NazvanieType}" FontSize="30" FontWeight="Bold" Foreground="Black"></TextBlock>

<TextBlock Text=" " FontSize="30" FontWeight="Bold"></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Column="1" HorizontalAlignment="Left" Margin="0,50,0,0">

<Ellipse Fill="Black" Width="10" Height="10" Margin="0,-220,0,0"/>

<TextBlock Text=" Цена: " FontSize="30" Foreground="Black"></TextBlock>

<TextBlock Text="{Binding Price}" FontSize="30" Foreground="Black"></TextBlock>

<TextBlock Text=" руб." FontSize="30" Foreground="Black"></TextBlock>

<Ellipse Fill="Black" Width="10" Height="10" Margin="-421,-130,0,0"/>

<TextBlock Text=" Количество: " FontSize="30" Foreground="Black" Margin="-205 40 0 0"></TextBlock>

<StackPanel>

<TextBox Height="30" HorizontalAlignment="Left" Margin="-20,47,0,0" x:Name="textBox1"

VerticalAlignment="Center" Width="150" FontSize="23" />

</StackPanel>

<StackPanel>

<Button Content="Купить" Name="buybtn" Click="buybtn\_Click" Margin="-300,200,0,0" Width="150" Height="35" />

</StackPanel>

</StackPanel>

</Grid>

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</Border>

<StackPanel Grid.Row="2"

Grid.Column="1"

Orientation="Horizontal"

HorizontalAlignment="Center">

</StackPanel>

</Grid>

**Фильтр по виду:**

private void CmbFiltr\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (CmbFiltr.SelectedIndex == 0)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.Where(x => x.IDtype == 1).ToList();

}

else

if (CmbFiltr.SelectedIndex == 1)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.Where(x => x.IDtype == 2).ToList();

}

else

if (CmbFiltr.SelectedIndex == 2)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.Where(x => x.IDtype == 3).ToList();

}

else

if (CmbFiltr.SelectedIndex == 3)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.Where(x => x.IDtype == 4).ToList();

}

}

**Сбросить данные:**

private void BtnResetAll\_Click(object sender, RoutedEventArgs e)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.ToList();

}

**Поиск по названия лекарства:**

private void TxtSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.Where(x => x.NazvanieMedicament.ToLower().Contains(TxtSearch.Text.ToLower())).ToList();

}

**Сортировка по названию лекарства От до Я и От Я до А:**

private void RbDown\_Checked(object sender, RoutedEventArgs e)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.

OrderByDescending(x => x.NazvanieMedicament).ToList();

}

private void RbUp\_Checked(object sender, RoutedEventArgs e)

{

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.

OrderBy(x => x.NazvanieMedicament).ToList();

}

**Удаление лекарства:**

private void BtnDelete\_Click(object sender, RoutedEventArgs e)

{

var usersForRemoving = LviewMed.SelectedItems.Cast<Medicament>().ToList();

if (MessageBox.Show($"Удалить {usersForRemoving.Count()} пользователей?",

"Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

try

{

SkladEntities.GetContext().Medicament.RemoveRange(usersForRemoving);

SkladEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

LviewMed.ItemsSource = SkladEntities.GetContext().Medicament.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

**Переход на страницу добавление лекарства:**

private void BtnAdd\_Click(object sender, RoutedEventArgs e)

{

ClassFrame.frmObj.Navigate(new AddViewingPage(null));

}

**Вывод на печать в Excel:**

private void BtnSaveToExcel\_Click(object sender, RoutedEventArgs e)

{

var app = new Excel.Application();

Excel.Workbook wb = app.Workbooks.Add();

Excel.Worksheet worksheet = app.Worksheets.Item[1];

int indexRows = 1;

worksheet.Cells[1][indexRows] = "Номер";

worksheet.Cells[2][indexRows] = "Название лекарства";

worksheet.Cells[3][indexRows] = "Цена";

worksheet.Cells[4][indexRows] = "Тип";

var printItems = LviewMed.Items;

foreach (Medicament item in printItems)

{

worksheet.Cells[1][indexRows + 1] = indexRows;

worksheet.Cells[2][indexRows + 1] = item.NazvanieMedicament;

worksheet.Cells[3][indexRows + 1].Value = item.Price.ToString(); ;

worksheet.Cells[4][indexRows + 1].Value = item.IDtype.ToString(); ;

indexRows++;

}

Excel.Range range = worksheet.Range[worksheet.Cells[2][indexRows + 1],

worksheet.Cells[5][indexRows + 1]];

range.ColumnWidth = 30;

range.HorizontalAlignment = Excel.XlHAlign.xlHAlignLeft;

app.Visible = true;

}

**Переход на страницу корзина:**

private void buybtn\_Click(object sender, RoutedEventArgs e)

{

this.NavigationService.Navigate(new Pages.BasketPage((sender as Button).DataContext as Medicament));

}

**Viewing.xaml**

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Image Source="/Photo/Pharmacy.png" Margin="0,0,4,0" Stretch="Fill" Grid.ColumnSpan="2"/>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center" Grid.Column="1">

<Button x:Name="Next" Content="Перейти" Click="Next\_Click" HorizontalAlignment="Right" VerticalAlignment="Center"

Width="200" Height="30" Background="#a3dadf" >

</Button>

</StackPanel>

</Grid>

**Переход на страницу товаров:**

private void Next\_Click(object sender, RoutedEventArgs e)

{

this.NavigationService.Navigate(new ViewingPage1());

}

**AddViewingPage.xaml**

<Grid>

<Grid.RowDefinitions>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1\*"/>

</Grid.ColumnDefinitions>

<Image Source="/Photo/Pharmacy3.jpg" Margin="0,0,4,0" Stretch="Fill" Grid.RowSpan="2"/>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Left" Margin="324,150,0,-150">

<StackPanel Orientation="Vertical" HorizontalAlignment="Center" Margin="15 15 0 0">

<TextBlock Text="Фото" FontSize="30"/>

<StackPanel Name="FillNull" HorizontalAlignment="Left" VerticalAlignment="Center">

<Rectangle Width="300" Height="300" Fill="White" Stroke="Black" StrokeThickness="3"/>

<Path Data="M0 0L300 300M300 0L0 300" Stroke="Black" Margin="0 -300 0 0" StrokeThickness="1"/>

</StackPanel>

<Image Name="imageMedicament" Width="300" Height="300" Margin="0 -300 0 0" Source="{Binding Image}" Stretch="UniformToFill" HorizontalAlignment="Left"/>

<Button Content="Загрузить" Width="200" Height="35" FontSize="25" Click="ImageLoad\_Click" HorizontalAlignment="Center" Margin="0 10 0 10"/>

<Button Content="Очистить" Width="200" Height="35" FontSize="25" Click="ImageDel\_Click"/>

</StackPanel>

</StackPanel>

<StackPanel Orientation="Vertical"

HorizontalAlignment="Left" Width="400" Height="300" Margin="730,240,0,260">

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">

<TextBlock Text="Название лекарстава:" Width="200" Height="35" FontSize="19"/>

<TextBox Name="TxtNazvanieMedicament" Text="{Binding NazvanieMedicament}" Width="200" Height="35" FontSize="25" TextChanged="TxtNazvanieMedicament\_TextChanged"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">

<TextBlock Text="Цена:" Width="200" Height="35" FontSize="19"/>

<TextBox Name="TxtPrice" Text="{Binding Price}" Width="200" Height="35" FontSize="25" TextChanged="TxtPrice\_TextChanged"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Center">

<TextBlock Text="Вид лекарства:" Width="200" Height="35" FontSize="19"/>

<ComboBox Name="CMBtype" Width="200" Height="35" FontSize="25" SelectedValue="{Binding IDtype}"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Row="2">

<Button Content="Добавить" Name="BtnAddOrEdit" Click="BtnAddOrEdit\_Click" Width="200" Height="35" FontSize="25" Margin="100 25 0 0"/>

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Row="2" >

<Button Content="Отмена" Click="BtnCancel\_Click" Width="200" Height="35" FontSize="25" Margin="100 10 0 0"/>

</StackPanel>

</StackPanel>

</Grid>

**Загрузка картинки:**

private void ImageLoad\_Click(object sender, RoutedEventArgs e)

{

try

{

OpenFileDialog dlg = new OpenFileDialog

{

Filter = "JPG Files (\*.jpg)|\*.jpg|PNG Files (\*.png)|\*.png|GIF Files (\*.gif)|\*.gif|All Files (\*.\*)|\*.\*",

Title = "Выберите фото/изображение лекарства"

};

if (dlg.ShowDialog() == true)

{

imgLoc = dlg.FileName.ToString();

imageMedicament.Source = new BitmapImage(new Uri(imgLoc));

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

**Очистить картинку:**

private void ImageDel\_Click(object sender, RoutedEventArgs e)

{

imageMedicament.Source = new BitmapImage(new Uri(".\\Photo\\foto\\im.jpg"));

imgLoc = "пусто";

}

**Добавление товара:**

private void BtnAddOrEdit\_Click(object sender, RoutedEventArgs e)

{

StringBuilder error = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_currentMedicament.NazvanieMedicament)) error.AppendLine("Укажите название лекарства");

if (string.IsNullOrWhiteSpace(\_currentMedicament.Price.ToString())) error.AppendLine("Укажите цену лекарства");

if (string.IsNullOrWhiteSpace(\_currentMedicament.IDtype.ToString())) error.AppendLine("Укажите вид лекарства");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentMedicament.IDmedicament == 0)

{

SkladEntities.GetContext().Medicament.Add(\_currentMedicament);

try

{

if (imgLoc != null && imgLoc != "пусто")

{

byte[] img = null;

FileStream fs = new FileStream(imgLoc, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

img = br.ReadBytes((int)fs.Length);

string filename = imgLoc.Substring(imgLoc.LastIndexOf('\\')+1);

\_currentMedicament.Image = String.Concat("/Photo/foto/", filename);

}

if (imgLoc == "пусто") \_currentMedicament.Image = null;

SkladEntities.GetContext().SaveChanges();

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

MessageBox.Show("Новое лекарство успешно добавлено!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

else

{

try

{

if (imgLoc != null && imgLoc != "пусто")

{

byte[] img = null;

FileStream fs = new FileStream(imgLoc, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

img = br.ReadBytes((int)fs.Length);

\_currentMedicament.Image = imgLoc;

}

if (imgLoc == "пусто") \_currentMedicament.Image = null;

SkladEntities.GetContext().SaveChanges();

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

MessageBox.Show("Лекарство успешно изменено!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

**Отмена добавление:**

private void BtnCancel\_Click(object sender, RoutedEventArgs e)

{

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

}

**BasketPage.xaml**

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="1.3\*"/>

<RowDefinition Height="7\*"/>

<RowDefinition Height="1.3\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="2\*"/>

<ColumnDefinition Width="10\*"/>

<ColumnDefinition Width="10\*"/>

</Grid.ColumnDefinitions>

<Image Source="/Photo/Pharmacy1.png" Margin="0,-8,0,-8" Stretch="Fill" Grid.ColumnSpan="3" Grid.RowSpan="3"/>

<StackPanel Grid.ColumnSpan="1">

<Button x:Name="Back" Click="Back\_Click" Width="45" Height="45">

<Button.Background>

<ImageBrush ImageSource="/Photo/iconkaback.png"/>

</Button.Background>

</Button>

</StackPanel>

<StackPanel Grid.ColumnSpan="3" Margin="0, 0, 0, 0">

<TextBlock Text="Корзина заказов"

Height="30"

Width="200"

Margin="5"

FontSize="25"/>

</StackPanel>

<ListView Name="ListOrder" Grid.Row="1" Grid.Column="1" SelectionChanged="ListOrder\_SelectionChanged" >

<ListView.ContextMenu>

<ContextMenu >

<MenuItem Header="Удалить" Name="DeleteMed" Click="DeleteMed\_Click" />

</ContextMenu>

</ListView.ContextMenu>

<ListView.View>

<GridView AllowsColumnReorder="true" ColumnHeaderToolTip="Employee Information">

<GridViewColumn DisplayMemberBinding="{Binding Path=Medicament.NazvanieMedicament}" Header="Название" Width="100"/>

<GridViewColumn DisplayMemberBinding="{Binding Path=Medicament.Price}" Width="100">

<GridViewColumnHeader>Цена

</GridViewColumnHeader>

</GridViewColumn>

<GridViewColumn DisplayMemberBinding="{Binding Path=Kolvo}" Header="Количество" Width="100"/>

</GridView>

</ListView.View>

</ListView>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center" Grid.Column="0" Grid.Row="0">

</StackPanel>

<StackPanel Grid.Row="2"

Grid.Column="1"

Orientation="Horizontal"

HorizontalAlignment="Center">

</StackPanel>

<StackPanel Grid.Row="1" Grid.Column="2">

<TextBlock Text="Количество"/>

<TextBlock Text="0" Name="CountItems" FontSize="35"/>

</StackPanel>

</Grid>

**Удаление товара из корзины:**

private void DeleteMed\_Click(object sender, RoutedEventArgs e)

{

var personForRemoving = ListOrder.SelectedItems.Cast<Basket>().ToList();

var resMessage = MessageBox.Show("Удалить запись?", "Подтверждение",

MessageBoxButton.YesNo, MessageBoxImage.Question);

if (resMessage == MessageBoxResult.Yes)

{

try

{

SkladEntities.GetContext().Basket.RemoveRange(personForRemoving);

SkladEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

ListOrder.ItemsSource = SkladEntities.GetContext().Basket.ToList();

}

catch (Exception ex)

{

}

}

}

**Вернуться назад:**

private void Back\_Click(object sender, RoutedEventArgs e)

{

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

}

# Руководство пользователя

## 4.1 Выполнение программного модуля

Для использования приложение необходимо:

1) Запустить exe файл приложения (sklad.exe)



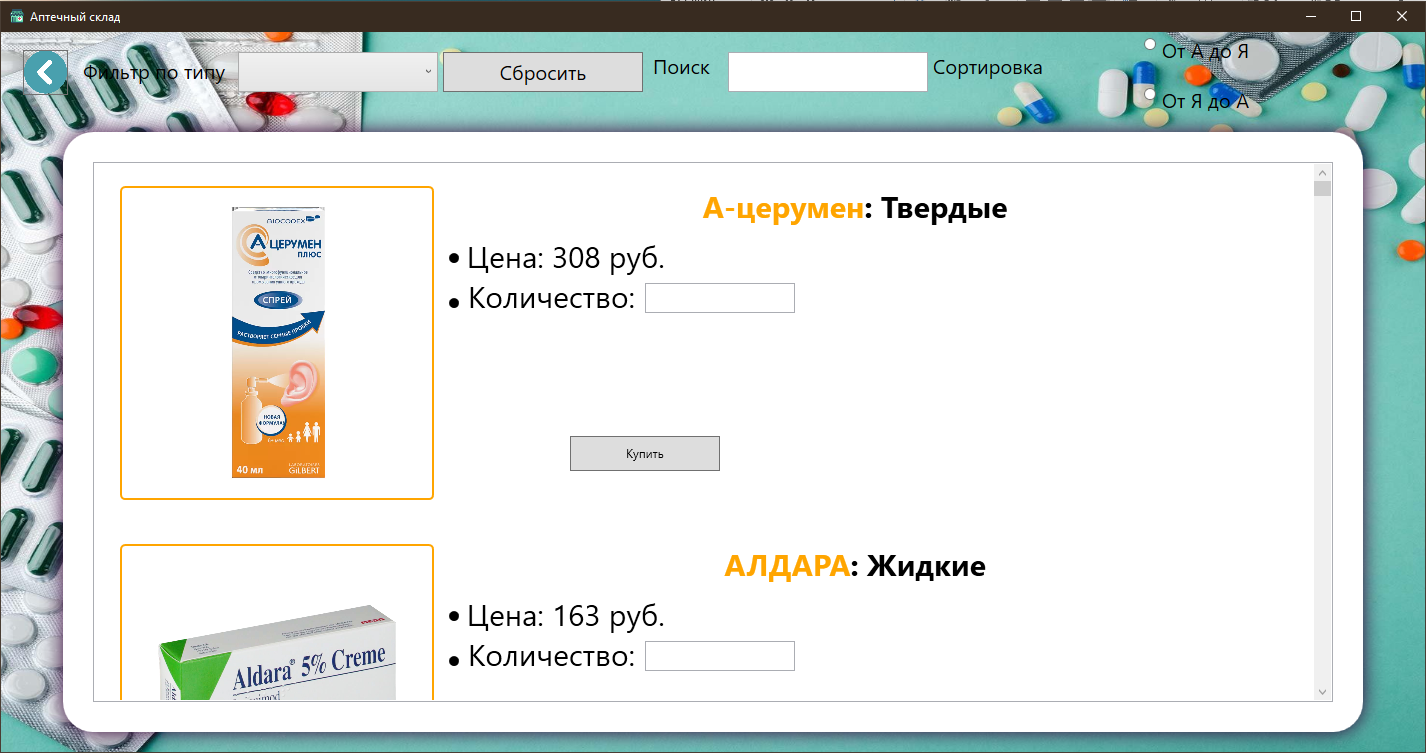
Рисунок№16 «exe файл»

2) После запуска exe файла пользователь попадает на «Главное Меню» приложения, где возможен переход на основную страницу, а именно: «Лекарства».



Рисунок№17 «Главное Меню»

3) При нажатии на кнопку «Перейти» (в главном меню приложения) пользователь попадает на основную страницу для работы с медикаментами. На этой страницы пользователь может выполнять следующие функции:



Рисунок№18 «Страница Медикаменты»

3.1) Фильтр по типу лекарства.



Рисунок№19 «Фильтр»

3.2) Кнопка «Сбросить» данные.



Рисунок№20 «Кнопка сбросить данные»

3.3) Поиск лекарства по названию.



Рисунок№21 «Поиск»

3.4) Сортировка лекарства От А до Я и От Я до А.



Рисунок№22 «Сортировка»

3.5) Удаление лекарства.

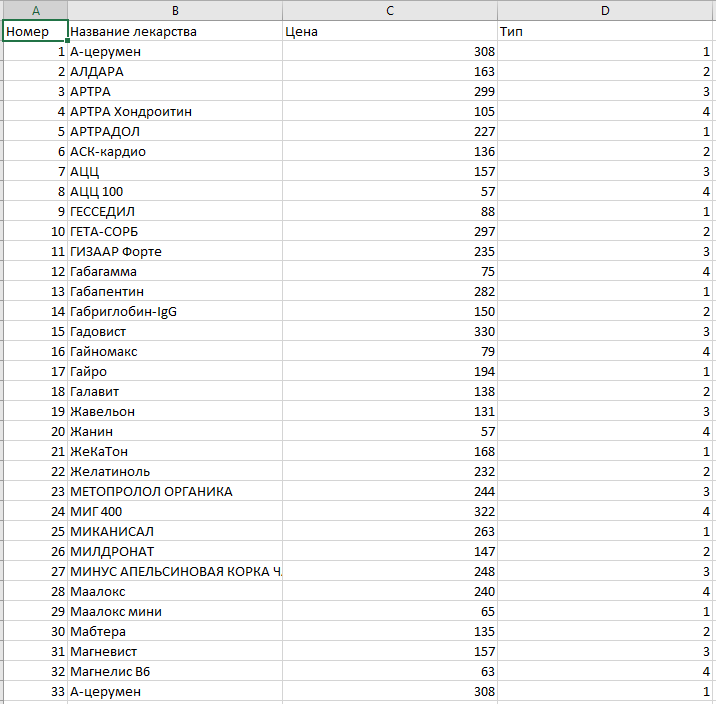


Рисунок№23 «Удаление»

3.6) Вывод на печать всех лекарств.



Рисунок№24 «Вывод на печать»



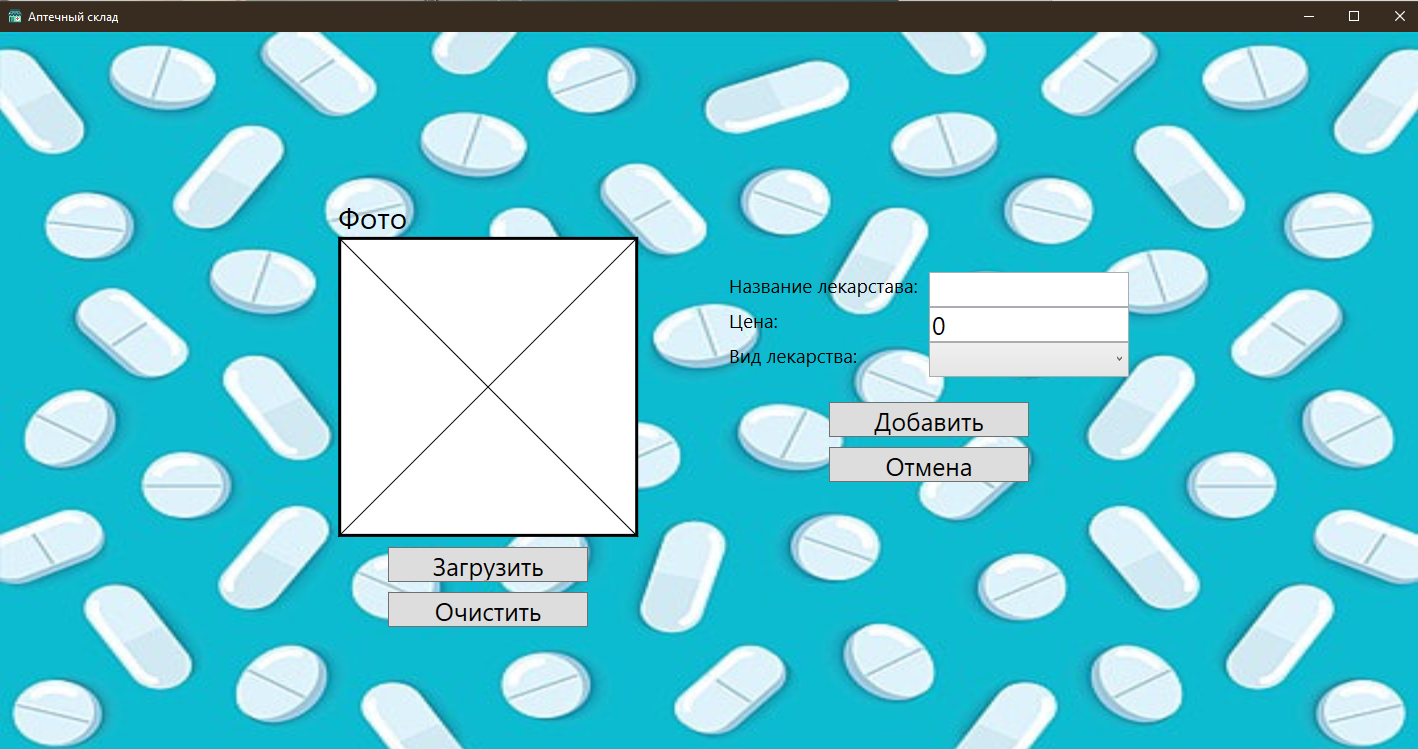
Рисунок№25 «Excel файл»

3.7) Возможен переход на предыдущую страницу.



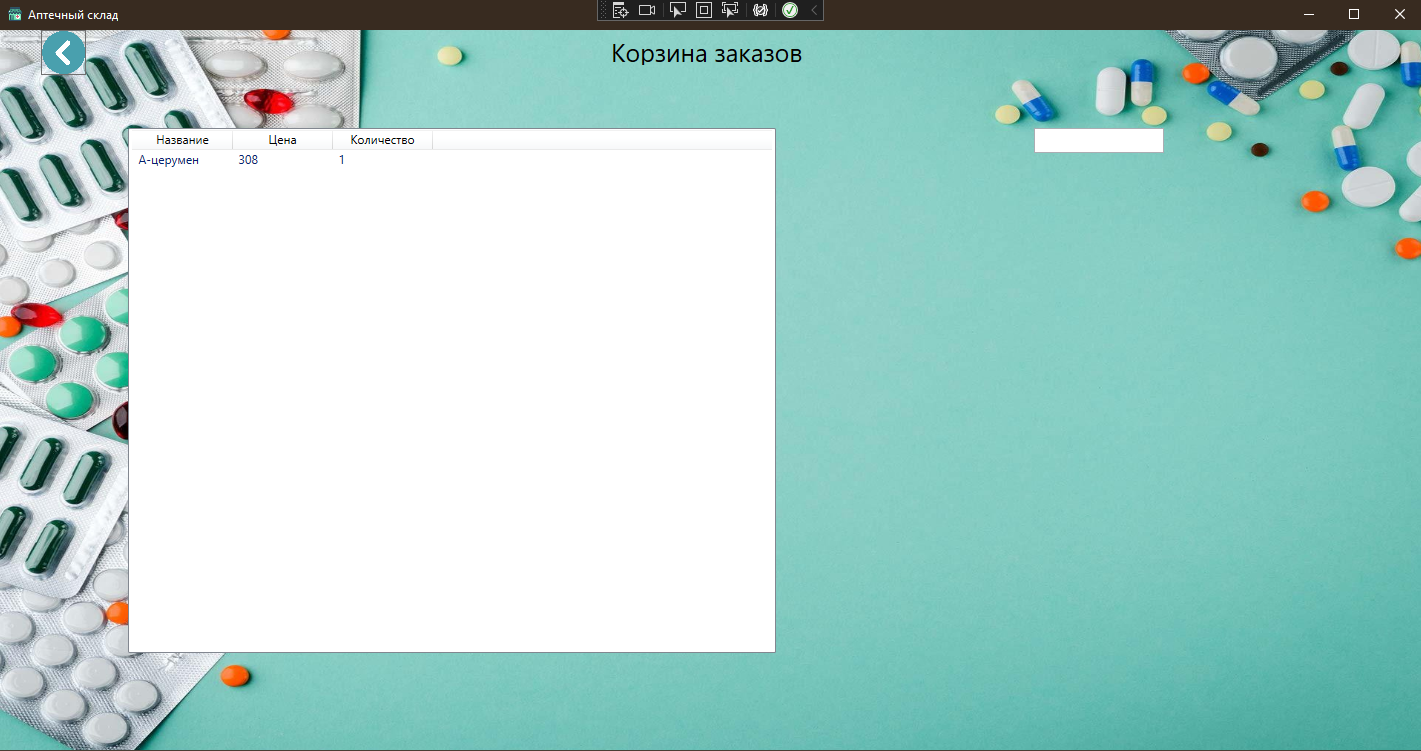
Рисунок№26 «Кнопка назад»

3.8) Добавление нового медикамента (Необходимо заполнить поля данными, и добавить при необходимости картинку и для добавления нажать кнопку «Добавить», если же добавление ненужно нажать кнопку «Отмена»).



Рисунок№27 «Добавление лекарства»

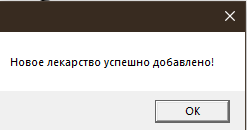
3.9) Переход на страницу корзина (при нажатии на кнопку купить переходим на страницу корзина там можно удалить данные и вернуться на предыдущую страницу).



Рисунок№28 «Корзина»

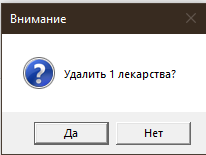
# 4.2 Сообщения пользователю

1.При добавлении нового лекарства пользователь получает сообщение: «Новое лекарство успешно добавлено!»

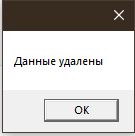


Рисунок№29 «Сообщение при добавлении»

2. При удалении лекарства пользователь получает сообщение: «Удалить 1 лекарства?». После чего если пользователь намерен удалить запись нужно нажать «Да», если же удаление ненужно необходимо нажать «Нет». Если же пользователь удалил лекарства то, он получает сообщение «Данные удалены».

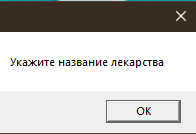


Рисунок№30 «Предупреждение при удалении»



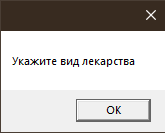
Рисунок№31 «Сообщение об удалении»

3. Если при добавлении нового лекарства не заполнить поле «Название лекарства», то пользователь получает сообщение: «Укажите название лекарства».



Рисунок№32 «Сообщение об указание данных»

4. Если при добавлении нового лекарства не выбрать данные из поле «Вид лекарства», то пользователь получает сообщение: «Укажите вид лекарства».



Рисунок№33 «Сообщение об указание данных»

# 5.Тестирование программного модуля

## 5.1Модульное тестирование

1)Код С# модульных тестов

В качестве образца возьмем поле с ценной лекарства. В данной ситуации тест пройдет успешо

1. Данный тест проверяет чтобы цена не содержала буквы.

[TestMethod]

public static int GetPriceLetter(string Price)

{

int expected = 1;

string Price = “aa”;

int result = 0;

if (!Regex.Match((Price).ToString(), "[a-z]").Success)

{

result += 1;

}

}

double actual = result;

Assert.AreEqual(expected, actual);

return result;

}

В качестве образца возьмем поле с ценной лекарства. В данной ситуации тест пройдет успешо

1. Данный тест проверяет чтобы цена не была отрицательной.

[TestMethod]

public static int GetPriceLess(int Price)

{

int expected = 1;

int Price = -10;

int result = 0;

if (Price < 0)

{

result += 1;

}

double actual = result;

Assert.AreEqual(expected, actual);

return result;

}

В качестве образца возьмем поле с ценной лекарства. В данной ситуации тест пройдет успешо

3. Данный тест проверяет чтобы цена не была больше 1500.

[TestMethod]

public static int GetPriceLittle(int Price)

{

int expected = 1;

PhoneNumber = "1600";

int result = 0;

if (Price > 1500)

{

result += 1;

}

double actual = result;

Assert.AreEqual(expected, actual);

return result;

}

Итоги тестов:

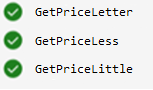


Рис. 34 «Итог тестов»

## **5.2Тестовые сценарии**

Таблица№11 «Аннотация теста»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Название проекта | TestPrice |
| Рабочая версия | 1.0 |
| Имя тестирующего | Стародубцев Максим Сергеевич |
| Дата(ы) теста | 03.04.2023 |

Таблица№12 «GetPriceLetter»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | GetPriceLetter. |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Проверка наличия букв. |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка наличия букв в строке. Цена не должна содержать буквы. |
| Этапы теста | Зайти на страницу добавления, указать следующие значения: Название лекарства, цену, вид лекарства. |
| Тестовые данные | Название товара: АЦЦ. Цена: 157. Тип: Твёрдый. |
| Ожидаемый результат | Сообщение отсутствует. |
| Фактический результат | Запись сохранена. |
| Предварительное условие | Должна быть кнопка для перехода на страницу добавления и сохранения данных, элементы управления куда будут заполняться данные. |
| Постусловие | Данные были добавлены в таблицу. |
| Статус(Зачет/Незачет) | Зачёт. |

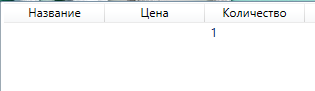
Таблица№13 «GetPriceLittle»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | GetPriceLess. |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Проверка на отрицательность цены. |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка наличия отрицательности цены. Цена не должна быть отрицательной. |
| Этапы теста | Зайти на страницу добавления, указать следующие значения: Название лекарства, цену, вид лекарства. |
| Тестовые данные | Название товара: АЦЦ. Цена: 157. Тип: Твёрдый. |
| Ожидаемый результат | Сообщение отсутствует. |
| Фактический результат | Запись сохранена. |
| Предварительное условие | Должна быть кнопка для перехода на страницу добавления и сохранения данных, элементы управления куда будут заполняться данные. |
| Постусловие | Данные были добавлены в таблицу. |
| Статус(Зачет/Незачет) | Зачёт. |

Таблица№14 «GetPriceLess»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | GetPriceLess. |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Проверка на высокую цену. |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка наличия высокой цены. Цена не должна быть слишком высокой. |
| Этапы теста | Зайти на страницу добавления, указать следующие значения: Название лекарства, цену, вид лекарства. |
| Тестовые данные | Название товара: АЦЦ. Цена: 157. Тип: Твёрдый. |
| Ожидаемый результат | Сообщение отсутствует |
| Фактический результат | Запись сохранена |
| Предварительное условие | Должна быть кнопка для перехода на страницу добавления и сохранения данных, элементы управления куда будут заполняться данные. |
| Постусловие | Данные были добавлены в таблицу. |
| Статус(Зачет/Незачет) | Зачёт |

1. Отладка программного модуля
2. Не видит данные в таблице.



Рисунок№34 «Не видит данные»

* 1. Необходимо дописать код.

<GridView AllowsColumnReorder="true" ColumnHeaderToolTip="Employee Information">

<GridViewColumn DisplayMemberBinding="{Binding Path=Medicament.NazvanieMedicament}" Header="Название" Width="100"/>

<GridViewColumn DisplayMemberBinding="{Binding Path=Medicament.Price}" Width="100">

<GridViewColumnHeader>Цена

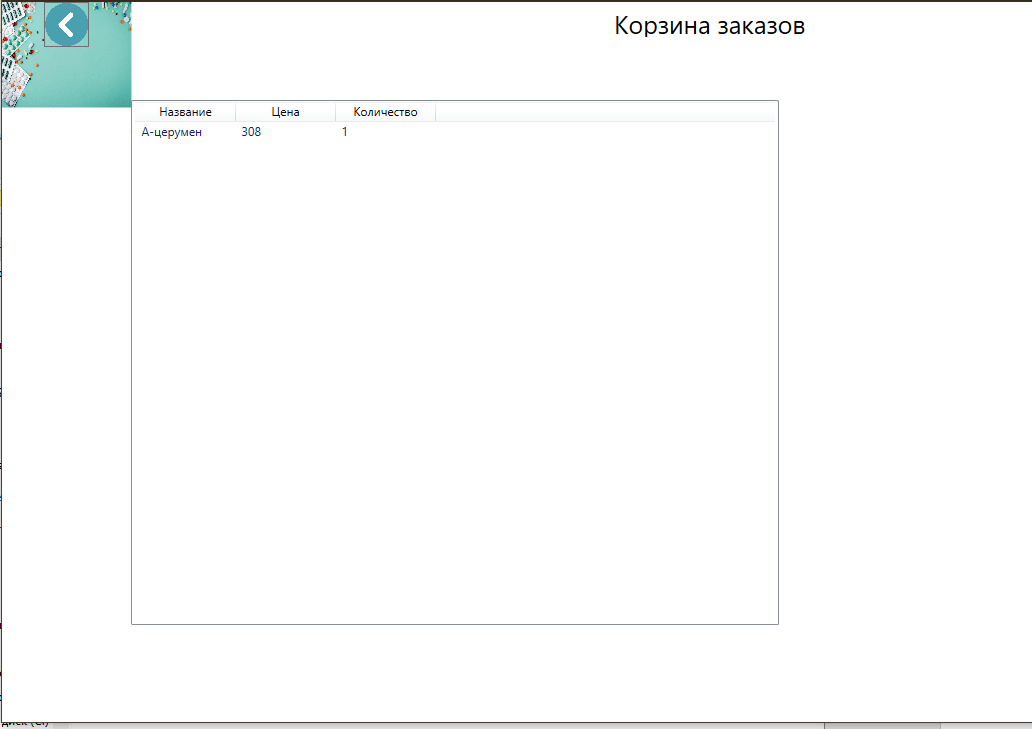
</GridViewColumnHeader>

</GridViewColumn>

<GridViewColumn DisplayMemberBinding="{Binding Path=Kolvo}" Header="Количество" Width="100"/>

</GridView>

1. Некорректная отображение картинки.



Рисунок№35 «Некорректная отображение картинки»

2.1) Необходимо дописать код.

<Image Source="/Photo/Pharmacy1.png" Margin="0,-8,0,-8" Stretch="Fill" Grid.ColumnSpan="3" Grid.RowSpan="3"/>

1. Не работает функция добавления картинок.

3.1) Необходимо дописать код

private void BtnAddOrEdit\_Click(object sender, RoutedEventArgs e)

{

StringBuilder error = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_currentMedicament.NazvanieMedicament)) error.AppendLine("Укажите название лекарства");

if (string.IsNullOrWhiteSpace(\_currentMedicament.Price.ToString())) error.AppendLine("Укажите цену лекарства");

if (string.IsNullOrWhiteSpace(\_currentMedicament.IDtype.ToString())) error.AppendLine("Укажите вид лекарства");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentMedicament.IDmedicament == 0)

{

SkladEntities.GetContext().Medicament.Add(\_currentMedicament);

try

{

if (imgLoc != null && imgLoc != "пусто")

{

byte[] img = null;

FileStream fs = new FileStream(imgLoc, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

img = br.ReadBytes((int)fs.Length);

string filename = imgLoc.Substring(imgLoc.LastIndexOf('\\')+1);

\_currentMedicament.Image = String.Concat("/Photo/foto/", filename);

}

if (imgLoc == "пусто") \_currentMedicament.Image = null;

SkladEntities.GetContext().SaveChanges();

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

MessageBox.Show("Новое лекарство успешно добавлено!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

else

{

try

{

if (imgLoc != null && imgLoc != "пусто")

{

byte[] img = null;

FileStream fs = new FileStream(imgLoc, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

img = br.ReadBytes((int)fs.Length);

\_currentMedicament.Image = imgLoc;

}

if (imgLoc == "пусто") \_currentMedicament.Image = null;

SkladEntities.GetContext().SaveChanges();

Classes.ClassFrame.frmObj.Navigate(new ViewingPage1());

MessageBox.Show("Лекарство успешно изменено!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

# **Заключение**

Целью данного курсового проекта была создание программы складского учета в "Аптечный мир". В результате проделанной работы была разработана программа на языке c#, которая позволяет:

- добавлять и удалять данные из меню;

- просматривать хранимые данные при поиске соответствующей информации;

- позволяет сортировать информацию по конкретным категориям.

- формирует, выводит на экран и на печать все лекарства, благодаря которым осуществляется контроль над складским учетом;

- позволяет добавлять данные в корзину покупок.

Данная программа позволяет существенно облегчить труд работников склада, так как благодаря ей значительно сокращается время обработки документов, расчета данных.

В данном проекте была разработана автоматизированная система складского учета, которая позволяет быстрое отслеживание товара на складе, так же она позволяет быстро составлять на основе информации о товаре бланк заказа.

Благодаря внедрению данной системы появилась возможность, по требованию руководства, оперативно выдавать информацию о совершенных заказах, о выдаче товара сотрудникам детского дома, о том какие товары имеются на складе в данный момент за любой промежуток времени.

Так же данная программа может быть использована в любом другом похожем учреждении. Требуется, лишь внести новые справочные данные взамен старой информации удалив её или откорректировав.

# Список литературы:

1. Петгольц, Ч. Программирование для Microsoft Windows на C#. В 3-х томах. Том 2. Пер. с англ./ Ч. Петгольц - М. : Издательско-торговый дом «Русская редакция», 2002. - 576 с.

2. Петгольц, Ч. Программирование для Microsoft Windows на C#. В 3-х томах. Том 3 Пер. с англ./ Ч. Петгольц - М. : Издательско-торговый дом «Русская редакция», 2002. - 624 с.

3. Жилинский, А. Самоучитель Misrosoft SQL Server 2008 / А. Жилинский. - М.: БХВ-Петербург, **2018**. - 240 c.

4. Кляйн, Д. SQL. Справочник / Д. Кляйн. - М.: Символ-плюс, 2014. - **935** c.