Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

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

**КУРСОВОЙ ПРОЕКТ**

«Разработка приложения по автоматизации учёта поступлений и распределений материалов на склады»

МДК 02.01 «Технология разработки программного обеспечения»

**Выполнил:**

Абуталыблы Ильяс Ядигар оглы

Студент 4 курса группы ИСП.20А

09.02.07 Информационные системы и программирование

очной формы обучения

**Руководитель:**

Селиверстова Ольга Михайловна

Оценка

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

Подпись руководителя

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

Ликино-Дулево

2023 год

**Оглавление**

[Введение 2](#_Toc154041397)

[1. Назначение разработки 5](#_Toc154041398)

[1.2 Требования к программе 5](#_Toc154041399)

[1.2.1 Требования к функциональным характеристикам 5](#_Toc154041400)

[1.2.2 Требования к надежности и безопасности 6](#_Toc154041401)

[1.2.3 Требования к составу и параметрам технических средств 6](#_Toc154041402)

[1.2.4 Требования к информационной и программной совместимости 6](#_Toc154041403)

[2. Разработка технического проекта 7](#_Toc154041404)

[2.1. Обоснование выбора CASE – средств 7](#_Toc154041405)

[2.2. Проектирование модели данных 7](#_Toc154041406)

[2.2.1 CASE – средство MS Visio 7](#_Toc154041407)

[3. Реализация 11](#_Toc154041408)

[3.1 Обоснование выбора средств разработки 11](#_Toc154041409)

[3.2 Руководство программиста 11](#_Toc154041410)

[3.3 Руководство пользователя 14](#_Toc154041411)

[4. Тестирование и откладка 20](#_Toc154041412)

[5. Методы и средства проведения расчётов оценки трудоёмкости разработки проекта (или Методы и средства защиты БД) 24](#_Toc154041413)

[Заключение 24](#_Toc154041414)

[Приложение 25](#_Toc154041415)

[Список литературы 27](#_Toc154041416)

**Введение**

Многофункциональный центр (МФЦ), полное название —Многофункциональный центр предоставления государственных и муниципальных услуг) — категория бюджетных учреждений в России, предоставляющих государственные и муниципальные услуги по принципу «одного окна» после однократного обращения заявителя с соответствующим запросом. При этом взаимодействие с органами, предоставляющими государственные услуги, или органами, предоставляющими муниципальные услуги, осуществляется многофункциональным центром без участия заявителя.

Заявленные цели:

* повышение качества и доступности государственных услуг;
* снижение издержек бизнеса на преодоление административных барьеров;
* повышение эффективности деятельности органов исполнительной власти и межведомственной координации;
* повышение открытости и прозрачности для общества.

25 октября 2005 года в России была принята концепция административной реформы и план мероприятий по её проведению в 2006—2010 годах. Одной из основных целей реформы было провозглашено повышение доступности и качества госуслуг.

В декабре 2013 года Председатель Правительства РФ поручил создать новый общий бренд сети МФЦ. Идея общего бренда системы МФЦ заключается в том, что документы нужны людям в важные и значимые моменты их жизни. Первые МФЦ под новым брендом были открыты в 2014 году. Товарный знак «Мои документы | Государственные и муниципальные услуги» зарегистрирован в Федеральной службе по интеллектуальной собственности. В 2015 году на XIX национальном фестивале рекламы «Идея!» бренд «Мои Документы» получил второе место в номинации «Товарный знак, фирменный стиль».

В центрах «Мои документы» по месту жительства можно воспользоваться следующими услугами по трудоустройству: составление резюме, подбор вакансий из базы данных, присвоение статуса безработного, выплата пособий и выдача направления на профессиональное обучение. Для получения расширенного комплекса услуг нужно обратиться во Флагманский центр занятости «Моя работа» на улице Щепкина. Список услуг включает: карьерную консультацию; профориентацию; психологическую помощь; тренинги по поиску работы; информирование о положении на рынке труда; определение стратегии обучения новой профессии; организацию стажировок и практик; организацию временного трудоустройства; осуществление социальных выплат гражданам, признанным в установленном порядке безработными; организацию ярмарок вакансий и учебных рабочих мест; организацию сопровождения при содействии занятости инвалидов; содействие безработным гражданам при переезде и безработным гражданам и членам их семей в переселении в другую местность для трудоустройства по направлению органов службы занятости.

В соответствии с правилами организации деятельности многофункциональных центров предоставления государственных и муниципальных услуг МФЦ использует автоматизированную информационную систему (АИС МФЦ). Основными принципами построения АИС МФЦ являются:

* процессно-ориентированная модель предоставления услуги;
* обеспечение информационной безопасности и защиты персональных данных;
* взаимодействие с внешними информационными системами в соответствии с требованиями действующего законодательства.

**1. Назначение разработки**

Автоматизированная информационная система «MFC» предназначена для оказания услуг клиенту по регистрации пользователя по указанному адресу. Пользователями программы выступает сотрудник центра. Перерегистрация заказчика осуществляется на основании договоров Регистрация гражданина Российской Федерации по месту жительства, в которых оговариваются условия регистрации. Акте перерегистрации указываются: первый адрес, где заказичк прописан на данный момент и конечный адрес куда он будет прописываться.

Данные первичных документов фиксируются в карточках учета, которые выполняют роль регистров регистрационного учета.

# 1.2 Требования к программе

## 1.2.1 Требования к функциональным характеристикам

Функциональные требования:

* Регистрация пользователя.
* Авторизация пользователя по логину и паролю.
* Добавление, редактирование и удаление данных из таблиц:
  + - «FullAdress»;
    - «Passport»;
    - «Registration»;
    - «Worker»;
* Поиск данных по основным атрибутам таблиц.
* Сортировка: пользователей по логину и ФИО, остальных данных по всем доступным атрибутам.
* Фильтрация данных по различным критериям: названию города, типу пасспорта, ФИО(Регистрации), ФИО(Работника)
* Вывод приходной накладной в Word.

## 1.2.2 Требования к надежности и безопасности

Программа должна быть в достаточной степени надёжна от сбоев. На крайний случай предусмотрено сохранение данных БД в приложении «Microsoft SQL» или восстановление данных в случае завершения работы.

### 1.2.3 Требования к составу и параметрам технических средств

Таблица №1. «Состав технических средств и их характеристики»

|  |  |
| --- | --- |
| Процессор | Intel® Core™ i5-6400 CPU @ 2.70GHz |
| Оперативная Память | 2 x 4 ГБ, DDR4, DIMM, 2133 МГц |
| Разрешение экрана | 1920x1080 |
| Размер экрана | 20 дюйма |
| Устройства ввода | Мышь, клавиатура |
| Дисковое пространство | 1Гб |
| Подключаемые устройства | Принтер |
| Операционная система | Windows 10 11 |

### 1.2.4 Требования к информационной и программной совместимости

Для корректной работы программы необходимо:

ОС Windows 10/11 – операционная система, сделанная корпорацией Microsoft;

Visual Studio 2022 – лучшая интегрированная среда разработки для создания многофункциональных, привлекательных кроссплатформенных приложений для Windows;

Microsoft SQL Server Management Studio 18 – интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до баз данных;

Microsoft Excel - программа для работы с электронными таблицами, созданная корпорацией Microsoft для Microsoft Windows, Windows NT и Mac OS, а также Android, iOS и Windows Phone. Она предоставляет возможности экономико-статистических расчетов, графические инструменты и, за исключением Excel 2008 под Mac OS X, язык макропрограммирования VBA. Microsoft Excel входит в состав Microsoft Office.

# 2. Разработка технического проекта

## Обоснование выбора CASE – средств

Выбор CASE-средства во многом зависит от конкретного подхода к проектированию ИС. Важнейшими из подходов являются структурный (функциональный), объектно-ориентированный, также отдельно выделяется методология ARIS.

Сущность структурного подхода к разработке ИС заключается в ее декомпозиции на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. На сегодняшний момент широкое распространение получили:

* CA ERwin Process Modeler (ранее: BPwin)
* CA ERwin Data Modeler (ранее: ERwin)
* Vantage Team Builder
* Oracle Designer

Исходя из выбранного подхода к проектированию было выбрано CASE – средство MS Visio, полностью удовлетворяющее запросам разработки программного средства.

## Проектирование модели данных

### 2.2.1 CASE – средство MS Visio

Microsoft Visio — программа для создания всевозможных видов схем. К их числу относятся блок-схемы, органиграммы, планы зданий и этажей, диаграммы DFD, схемы технологических процессов, модели бизнес-процессов, диаграммы плавательных дорожек, трехмерные карты и так далее. Основные возможности Visio:

* Диаграмма прецедентов - это графическое представление взаимодействия между системой и ее окружением, фокусирующееся на функциональности, которую система предоставляет своим пользователям (актерам). Она описывает различные прецеденты (сценарии использования) системы, используемые для взаимодействия с внешними сущностями. Диаграммы прецедентов часто используются в методологии Unified Modeling Language (UML) для моделирования требований к системе.
* Диаграмма действий – это графическое представление последовательности шагов или действий в конкретном процессе или сценарии. Она помогает визуально отобразить последовательность выполнения операций, решения проблем или других действий.
* Таблица операций обычно используется для документирования различных операций, процедур или действий, которые выполняются в рамках определенной системы или процесса.
* Таблицы описания документов могут быть использованы для организации и управления информацией о документах в организации или проекте. Эти таблицы могут включать информацию о названии документа, его типе, авторе, дате создания, версии, изменениях, статусе и другие сведения, которые помогают эффективно управлять документами. Такие таблицы облегчают отслеживание и доступ к документам в организации

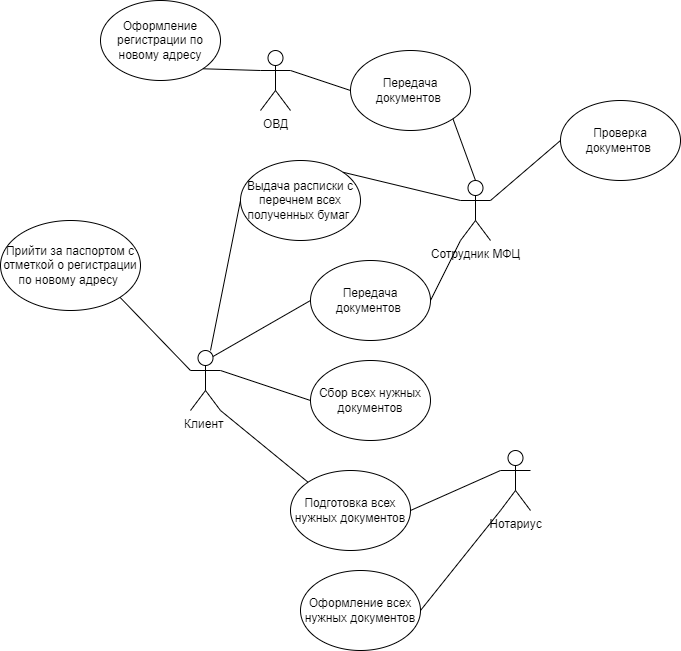


Рис. 2 «Диаграмма прецедентов»

Определение бизнес-процессов:

Таблица №2 «Таблица Бизнес-процессов»

|  |  |
| --- | --- |
| **Номер бизнес-процесса** | **Название бизнес-процесса** |
| 1-Подготовка | Подготовка необходимых документ |
| 2-Прием заявления | Прием заявления от клиента |
| 3-Регистрация | Регистрация заявления в системе |
| 4-Очередь | Ожидание очереди |
| 5-Выполнение услуги | Выполнение запроса клиента |
| 6-Оплата | Оплата услуги |
| 7-Выдача документов | Выдача документов клиенту |

Словесный алгоритм бизнес-процесса «Выполнение услуги к» осуществляется следующим образом:

1. Подготовка: - Определите необходимые документы для выполнения услуги. - Соберите и подготовьте все необходимые документы по списку.

2. Прием заявления: - Примите заявление от клиента. - Проверьте правильность заполнения заявления. - Запишите данные клиента и заявления в систему.

3. Регистрация: - Зарегистрируйте заявление клиента в системе. - Присвойте заявлению уникальный номер. - Сохраните все данные и документы, связанные с заявлением.

4. Очередь: - Установите клиента в очередь ожидания обслуживания. - Предоставьте клиенту информацию о его текущем положении в очереди и примерное время ожидания.

5. Выполнение услуги: - Обработайте заявление клиента согласно установленным правилам и процедурам. - Проведите необходимые проверки и проверьте наличие всех необходимых документов. - Выполните запрошенную услугу согласно установленным процессам.

6. Оплата: - Предоставьте клиенту информацию о необходимости оплаты услуги (если применимо). - Примите платеж от клиента в соответствии с указанными условиями.

7. Выдача документов: - Подготовьте и сформируйте необходимые документы для клиента в соответствии с выполненной услугой. - Проверьте правильность и полноту выдаваемых документов. - Выдайте документы клиенту.

Диаграмма действий:

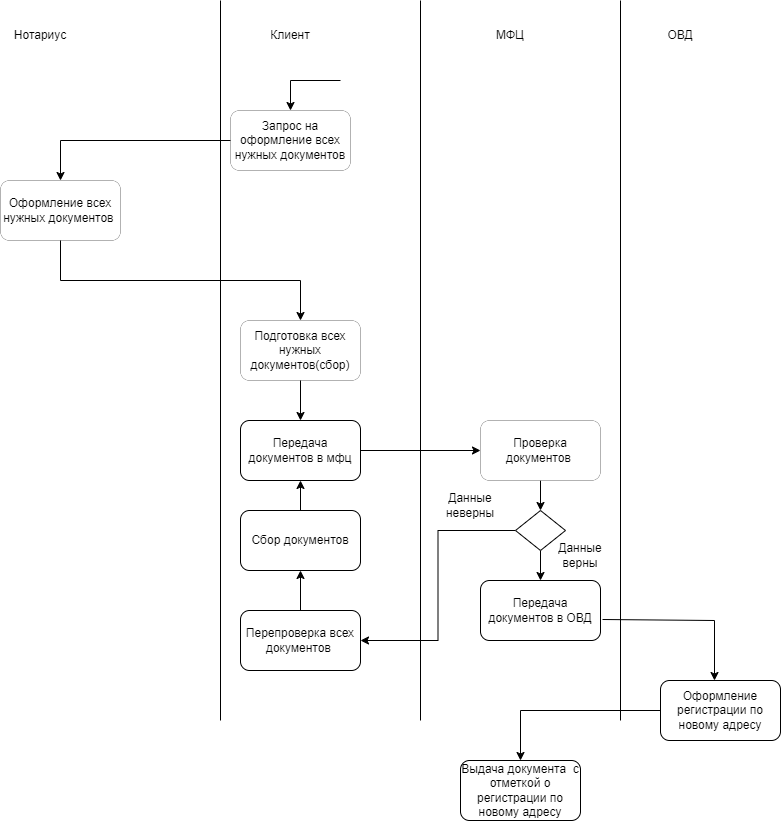


Рис. 3 «Бизнес-процесс «Выполнение услуги»»

Таблица №3 «Таблица описания операций

«Приходование\_товаров\_1Пр\_Т»»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Диаграмма и номер операции на  диаграмме | Составляемый документ  (Исходящий документ) | Операция | Исполнитель | Как часто | Входящие  документы  (документы-  основания) | Реестр, в котором регистрируется документ |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 Пр\_Т\_1 | 1.Заявление на услугу | Подача заявления | Клиент | По мере необходимости | - | Реестр регистрации документов |
| 1 Пр\_Т\_2 | 2.Регистрация заявления в системе | Регистрация заявления | Сотрудник регистрации | По мере необходимости | Заявление на услугу | Реестр заявлений |
| 1 Пр\_Т\_3 | 3.Уведомление о принятии заявления | Уведомление о принятии заявления | Сотрудник регистрации | Ежедневно | Регистрация заявления в системе | Реестр уведомлений |
| 1 Пр\_Т\_4 | 4.Заявление об отказе в услуге | Отказ в предоставлении услуги | Сотрудник регистрации | По мере необходимости | Регистрация заявления в системе | Реестр отказов |
| 1 Пр\_Т\_5 | 5.Уведомление о готовности услуги | Уведомление о готовности услуги | Сотрудник регистрации | По мере необходимости | Заявление на услугу | Реестр уведомлений |

# 3. Реализация

## 3.1 Обоснование выбора средств разработки

В качестве среды разработки выбран мощнейший инструмент - Visual Studio 2022.

Разработчики часто сталкиваются с многочисленными проблемами по обработке информации; например, с потребностью в более быстрых и основанных на управлении данными решениях, с потребностью увеличить производительность и мобильность штата разработчиков, а также с настойчивыми требованиями уменьшить общий бюджет отдела информационных технологий, увеличивая инфраструктуру в соответствии со строго возрастающими требованиями. Microsoft SQL Server 2022 призван помочь справиться с этими проблемами. MS SQL Server 2018 представляет собой интегрированное решение по управлению и анализу данных, которое помогает:

- строить, развертывать и управлять промышленными приложениями, которые являются более безопасными, масштабируемыми и надежными.

- увеличивать продуктивность информационных технологий, уменьшая сложность построения, развертывания и управления приложениями.

- разделять данные между платформами, приложениями и устройствами для облегчения соединения внутренних и внешних систем.

- контролировать стоимость, не жертвуя качеством выполнения, доступностью, масштабируемостью и безопасностью.

Исходя с вышеперечисленных достоинств MS SQL Server 2018, в качестве сервера базы данных был выбран именно этот продукт.

## 3.2 Руководство программиста

Разработана схема данных в среде в Visual Studio.

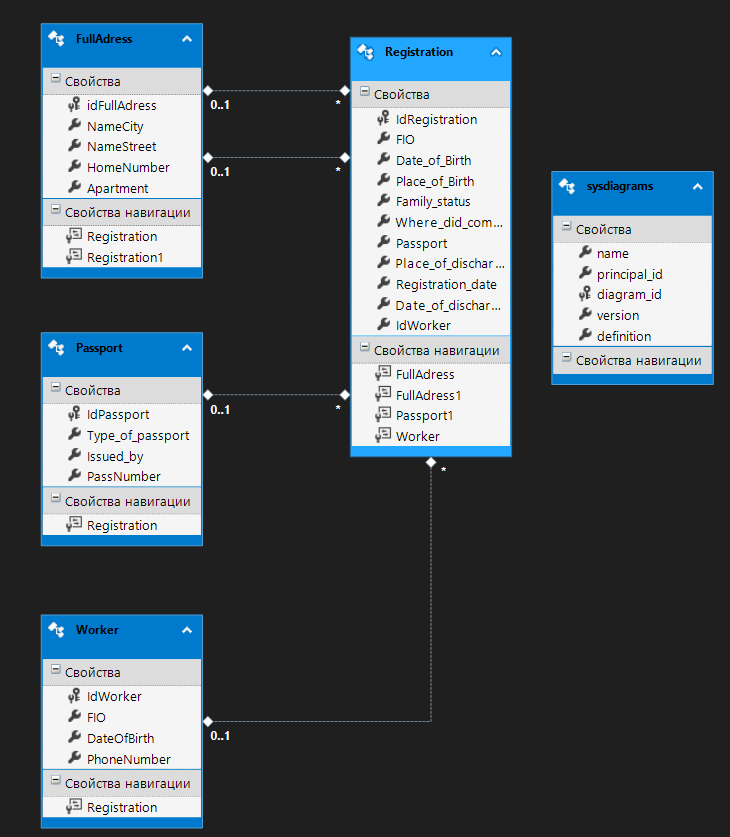


Рис. 3 «Модель данных»

**Структура приложения в обозревателе решений:**

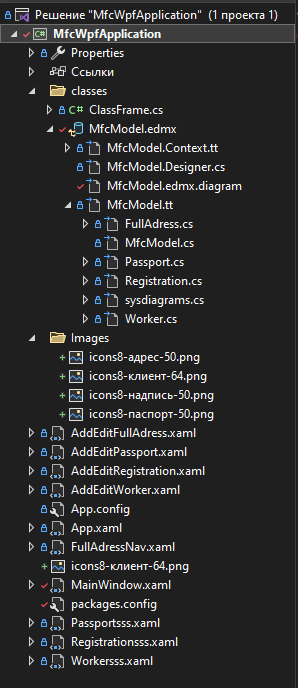


Рис. 4 «Структура приложения»

Класс «Frame», служащий для вывода данных с базы данных:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace SalesOfMaterials.Classes

{

public class ClassFrame

{

public static Frame frmObj;

}

}

Класс «MFCDBEntities», служащий для связи с базой данных:

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class MFCDBEntities : DbContext

{

private static MFCDBEntities \_context;

public MFCDBEntities()

: base("name=MFCDBEntities")

{

}

public static MFCDBEntities GetContext()

{

if (\_context == null)

\_context = new MFCDBEntities();

return \_context;

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public virtual DbSet<FullAdress> FullAdress { get; set; }

public virtual DbSet<Passport> Passport { get; set; }

public virtual DbSet<Registration> Registration { get; set; }

public virtual DbSet<sysdiagrams> sysdiagrams { get; set; }

public virtual DbSet<Worker> Worker { get; set; }

}

Страница регистрации:

Registrationsss.xaml:

<Grid Background="#293133">

<Grid.RowDefinitions>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

<DataGrid x:Name="DGridCashiers" AutoGenerateColumns="False" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="IdRegistration" Binding ="{Binding IdRegistration}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="FIO" Binding ="{Binding FIO}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Date\_of\_Birth" Binding ="{Binding Date\_of\_Birth}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Place\_of\_Birth" Binding ="{Binding Place\_of\_Birth}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Family\_status" Binding ="{Binding Family\_status}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Where\_did\_come\_from" Binding ="{Binding Where\_did\_come\_from}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Passport" Binding ="{Binding Passport}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Place\_of\_discharge" Binding ="{Binding Place\_of\_discharge}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Registration\_date" Binding ="{Binding Registration\_date}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Date\_of\_discharge" Binding ="{Binding Date\_of\_discharge}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="IdWorker" Binding ="{Binding IdWorker}" Width="\*"></DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Редактировать" Name="BtnEdit" Click="BtnEdit\_Click"></Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Button Content="Добавить" HorizontalAlignment="Left" Grid.Row="1" Width="140" Height="50" Name="BtnAdd" Click="BtnAdd\_Click"/>

<Button Content="Удалить" HorizontalAlignment="Center" Grid.Row="1" Width="140" Height="50" Name="BtnDelete" Click="BtnDelete\_Click"/>

<Button Content="Назад" HorizontalAlignment="Right" Grid.Row="1" Width="140" Height="50" Name="BtnBack" Click="BtnBack\_Click"/>

<StackPanel Grid.Row="1" Margin="160,21,560,25">

<TextBlock Text="Фильтрация по названию"/>

<ComboBox x:Name="CmbFilterRegistration"

Height="30"

SelectionChanged="CmbFilterRegistration\_SelectionChanged"/>

</StackPanel>

<StackPanel Grid.Row="1" Margin="568,21,167,25">

<TextBlock Text="Поиск по названию"/>

<TextBox Height="30" x:Name="SearchRegistration" TextChanged="SearchRegistration\_TextChanged" Width="165"/>

</StackPanel>

</Grid>

Registrationsss.xaml.cs:

public partial class Registrationsss : Page

{

public Registrationsss()

{

InitializeComponent();

DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.ToList();

CmbFilterRegistration.ItemsSource = MFCDBEntities.GetContext().Registration.ToList();

CmbFilterRegistration.SelectedValuePath = "IdRegistration";

CmbFilterRegistration.DisplayMemberPath = "FIO";

}

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

{

ClassFrame.frmObj.Navigate(new AddEditRegistration((sender as Button).DataContext as Registration));

}

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

{

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

}

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

{

var registrationForRemoving = DGridCashiers.SelectedItems.Cast<Registration>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить следующие {registrationForRemoving.Count()} записи?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

MFCDBEntities.GetContext().Registration.RemoveRange(registrationForRemoving);

MFCDBEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены!");

DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.ToList();

}

catch (Exception ex)

{

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

}

}

}

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

{

}

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

{

if (DGridCashiers.ItemsSource != null)

{

DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.Where(x => x.FIO.ToLower().Contains(SearchRegistration.Text.ToLower())).ToList();

}

if (SearchRegistration.Text.Count() == 0) DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.ToList();

}

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

{

int id = Convert.ToInt32(CmbFilterRegistration.SelectedValue);

DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.Where(x => x.IdRegistration == id).ToList();

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

if (Visibility == Visibility.Visible)

{

MFCDBEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

DGridCashiers.ItemsSource = MFCDBEntities.GetContext().Registration.ToList();

}

}

}

Страница добавления/редактирования Регистрации:

AddEditRegistration.xaml:

<Grid Background="#293133">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="50\*"/>

<ColumnDefinition Width="350\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<TextBlock Text="FIO" Grid.Row="1" Height="20"/>

<TextBlock Text="Date\_of\_Birth" Grid.Row="2" Height="20"/>

<TextBlock Text="Place\_of\_Birth" Grid.Row="3" Height="20"/>

<TextBlock Text="Family\_status" Grid.Row="4" Height="20"/>

<TextBlock Text="Where\_did\_come\_from" Grid.Row="5" Height="20"/>

<TextBlock Text="Passport" Grid.Row="6" Height="20"/>

<TextBlock Text="Place\_of\_discharge" Grid.Row="7" Height="20"/>

<TextBlock Text="Registration\_date" Grid.Row="8" Height="20"/>

<TextBlock Text="Date\_of\_discharge" Grid.Row="9" Height="20"/>

<TextBlock Text="IdWorker" Grid.Row="10" Height="20"/>

<TextBox Text="{Binding FIO}" Grid.Column="1" Grid.Row="1" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Date\_of\_Birth}" Grid.Column="1" Grid.Row="2" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Place\_of\_Birth}" Grid.Column="1" Grid.Row="3" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Family\_status}" Grid.Column="1" Grid.Row="4" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Where\_did\_come\_from}" Grid.Column="1" Grid.Row="5" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Passport}" Grid.Column="1" Grid.Row="6" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Place\_of\_discharge}" Grid.Column="1" Grid.Row="7" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Registration\_date}" Grid.Column="1" Grid.Row="8" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding Date\_of\_discharge}" Grid.Column="1" Grid.Row="9" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<TextBox Text="{Binding IdWorker}" Grid.Column="1" Grid.Row="10" Width="175" Height="40" HorizontalAlignment="Left"></TextBox>

<Button Grid.ColumnSpan="1" Content="Сохранить" HorizontalAlignment="Left" Margin="150,0,0,0" Grid.Row="11" Name="BtnSave" Click="BtnSave\_Click" VerticalAlignment="Center" Grid.Column="1" Height="30" Width="150"/>

</Grid>

AddEditRegistration.xaml.cs:

public partial class AddEditRegistration : Page

{

private Registration \_registration = new Registration();

public AddEditRegistration(Registration selectedRegistration)

{

InitializeComponent();

if (selectedRegistration != null)

\_registration = selectedRegistration;

DataContext = \_registration;

}

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

{

StringBuilder errors = new StringBuilder();

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (\_registration.IdRegistration == 0)

MFCDBEntities.GetContext().Registration.Add(\_registration);

try

{

MFCDBEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена");

}

catch (Exception ex)

{

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

}

}

}

Главное меню MainWindow.xaml:

<Grid Background="#293133">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="100"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="100"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!--Функции-->

<Border Background="#3F888F" Grid.Row="0" Grid.Column="1" Height="90" Width="600" CornerRadius="35">

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

<StackPanel Width="200">

</StackPanel>

</StackPanel>

</Border>

<!--Данные с БД-->

<Border Grid.Row="1" Grid.Column="0" Background="#3F888F" CornerRadius="35" Height="500" Width="90">

<StackPanel Height="430">

<Button Name="FullAdress" Height="70" Width="70" Click="FullAdress\_Click" >

<Image Source="/Images/icons8-адрес-50.png" />

</Button>

<Button Name="Passport" Height="70" Width="70" Margin="0,50" Click="Passport\_Click" >

<Image Source="/Images/icons8-паспорт-50.png" />

</Button>

<Button Name="Registration" Height="70" Width="70" Margin="0,0" Click="Registration\_Click" >

<Image Source="/Images/icons8-надпись-50.png" />

</Button>

<Button Name="Worker" Height="70" Width="70" Margin="0,50" Click="Worker\_Click">

<Image Source="/Images/icons8-клиент-64.png" />

</Button>

</StackPanel>

</Border>

<Frame Grid.Row="1" Grid.Column="1" Name="FrmCurExc" Background="#3F888F"/>

</Grid>

MainWindow.xaml.cs:

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

ClassFrame.frmObj = FrmCurExc;

}

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

{

ClassFrame.frmObj.Navigate(new FullAdressNav());

}

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

{

ClassFrame.frmObj.Navigate(new Passportsss());

}

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

{

ClassFrame.frmObj.Navigate(new Registrationsss());

}

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

{

ClassFrame.frmObj.Navigate(new Workersss());

}

}

**Входные и выходные данные**

Таблица №2 «Словарь данных»

|  |  |  |  |
| --- | --- | --- | --- |
| **ключ** | **поле** | **обязательное** | **примечание** |
| **Адрес** | | | |
| Первичный | IdExpenseIvoices | Да | Идентификатор |
|  | NameCity | Да | Название города |
|  | NameStreet | Да | Название улицы |
|  | HomeNumber | Да | Номер дома |
|  | Apartment | Да | Номер квартиры |
| **Паспорт** | | | |
| Первичный | IdPassport | Да | Идентификатор |
|  | Type\_of\_passport | Да | Тип Паспорта |
|  | Issued\_by | Да | Кем выдан |
|  | PassNumber | Да | Номер Паспорта |
| **Работники** | | | |
| Первичный | IdWorker | Да | Идентификатор |
|  | FIO | Да | ФИО |
|  | DateOfBirth | Да | Дата рождения |
|  | PhoneNumber | Да | Номер телефона |
| **Тип пользователей** | | | |
| Первичный | IdRegistration | Да | Идентификатор |
|  | FIO | Да | ФИО |
|  | Date\_of\_Birth | Да | Дата рождения |
| Внешний | Place\_of\_Birth | Да | Внешний ключ к таблице адрес |
|  | Family\_status | Да | Семейное положение |
| Внешний | Where\_did\_come\_from | Да | Внешний ключ к таблице адрес |
| Внешний | Passport | Да | Внешний ключ к таблице паспорт |
| Внешний | Place\_of\_discharge | Да | Внешний ключ к таблице адрес |
|  | Registration\_date | Да | Дата регистрации |
|  | Date\_of\_discharge | Да | Дата когда выбыл |
| Внешний | IdWorker | Да | Внешний ключ к таблице работник |

## 3.3 Руководство пользователя

При открытии приложения вылезает главное окно “MainWindow”

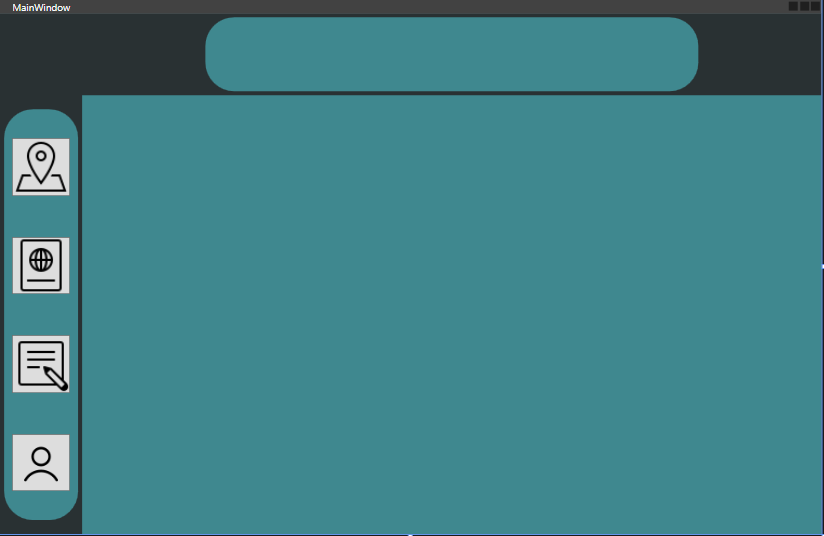


Рис. 4 «Главное меню»

С кнопками переходов на следующие страницы:

Страница Адресов

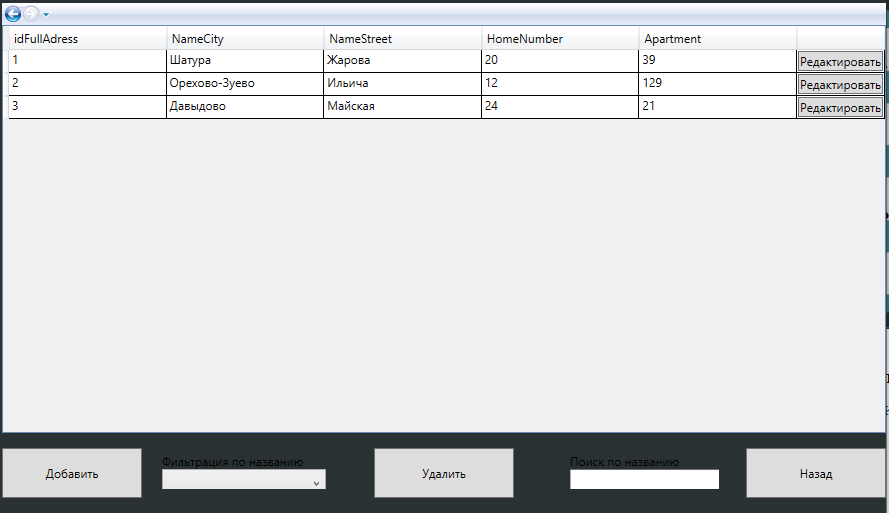


Рис. 5 «Страница адреса»

Страница Паспортов

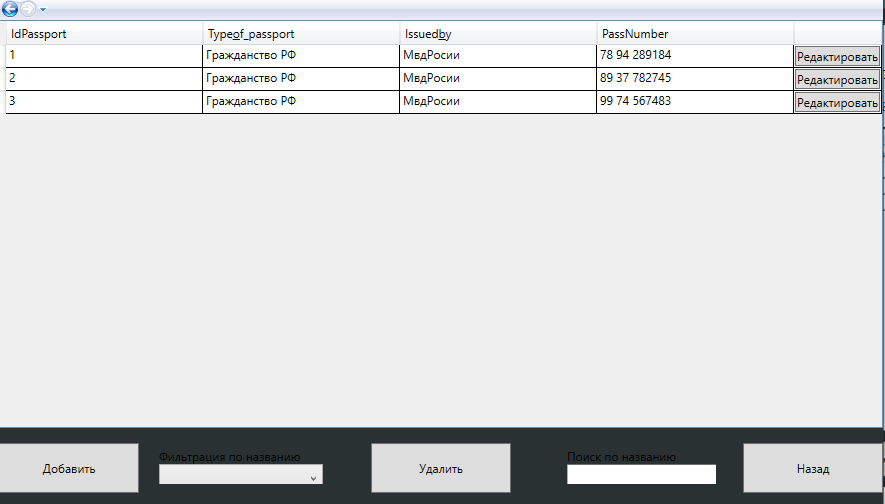


Рис. 6 «Страница паспорта»

Страница регистраций

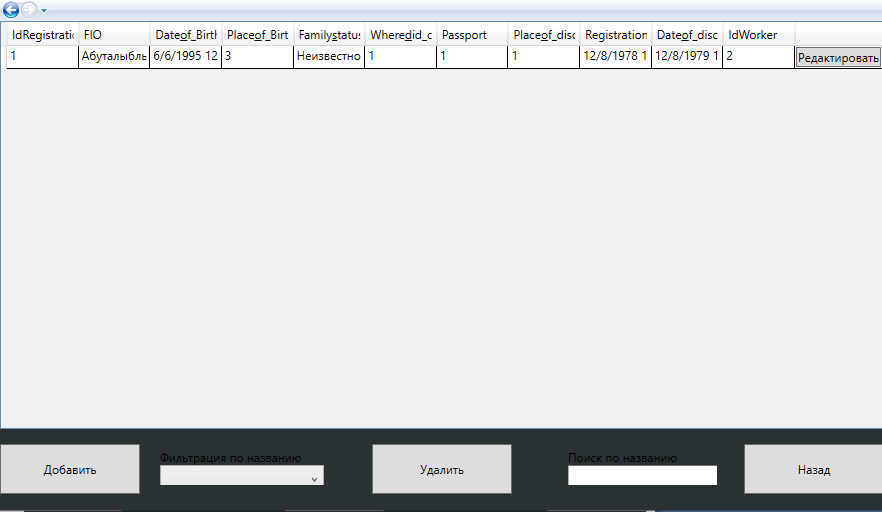


Рис. 7 «Страница Регистрации»

Страница Работников

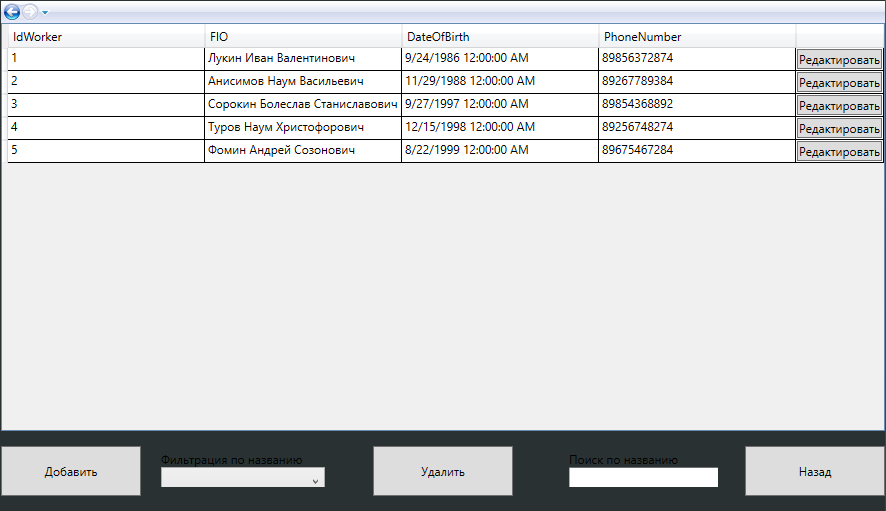


Рис. 8 «Страница Работников»

Страница добавления на примере Работников:



Рис. 9 «Страница Добавления»

Страница добавления на примере работников:

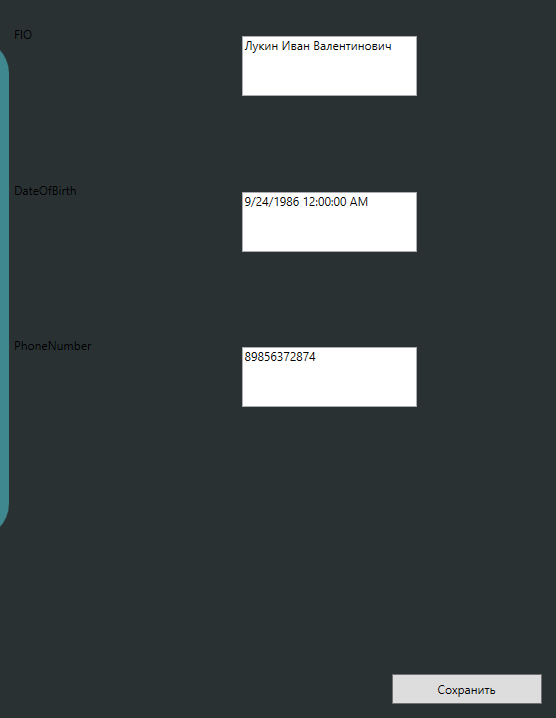


Рис. 10 «Страница Редактирования»

Функция Фильтрации по названию:

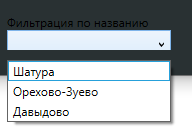


Рис. 11 «Функция фильтрации»

Функция Поиска по названию:

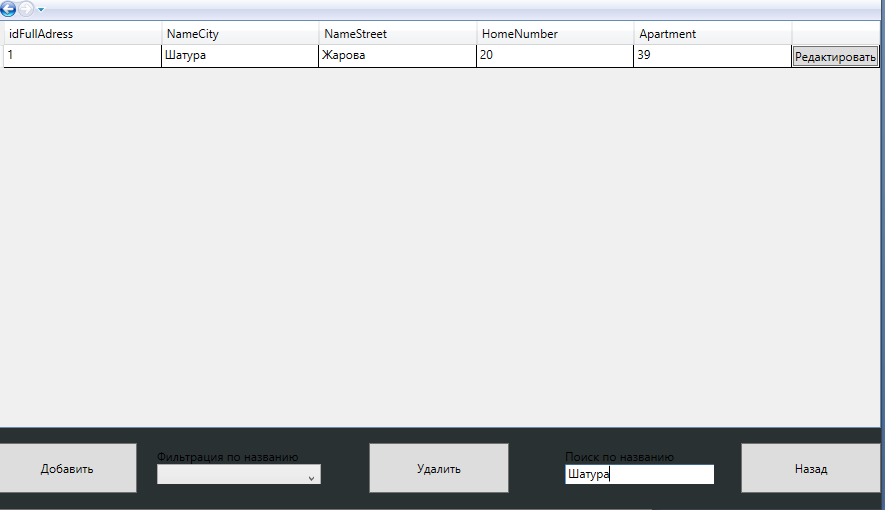


Рис. 12 «Функция поиска по названию»

Функция удаления на примере Работника:

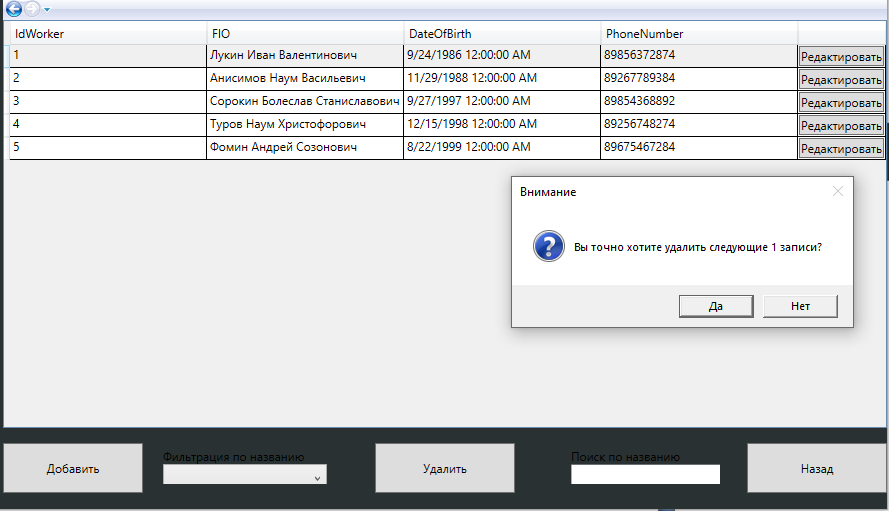


Рис. 13 «Функция удаления до удаления»

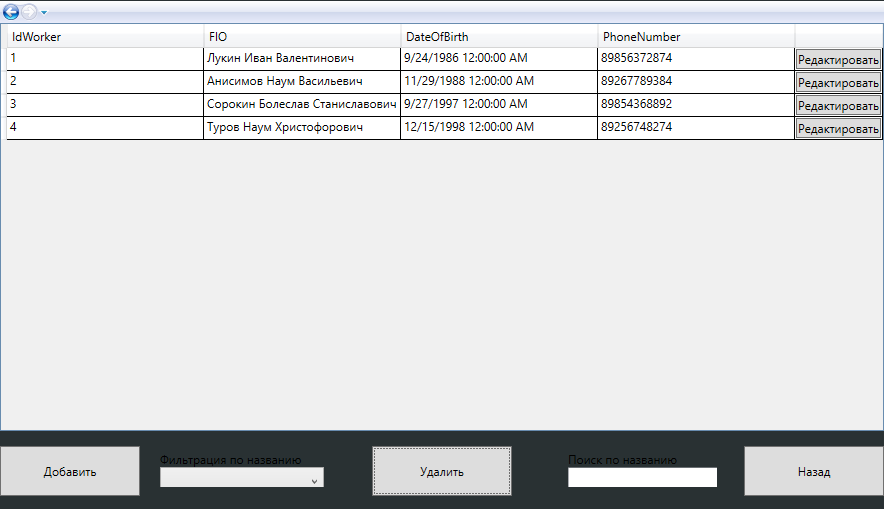


Рис. 14 «Функция удаления после удаления»

Диалоговые окна для корректной работы пользователя с приложением:

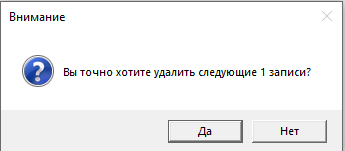


Рис. 15 «Диалоговое окно удаления»

# 4. Тестирование и откладка

Жизненный цикл тестирования программного обеспечения — это процесс выполнения различных действий в ходе проведения тестирования. Как пример, сюда относится составление тест-плана, анализ требований, поиск дефектов, поддержка после релиза и так далее.

Тестирование ПО — процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом (ISO/IEC TR 19759:2005).

Цель тестирования — проверка соответствия ПО предъявляемым требованиям, обеспечение уверенности в качестве ПО, поиск очевидных ошибок в программном обеспечении, которые должны быть выявлены до того, как их обнаружат пользователи программы.

Тестовый сценарий (test case) — это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.

Методы тестирования:

* Тестирование белого ящика — метод тестирования ПО, который предполагает, что внутренняя структура/устройство/реализация системы известны тестировщику.
* Тестирование серого ящика — метод тестирования ПО, который предполагает комбинацию White Box и Black Box подходов. То есть, внутреннее устройство программы нам известно лишь частично.
* Тестирование чёрного ящика — также известное как тестирование, основанное на спецификации или тестирование поведения — техника тестирования, основанная на работе исключительно с внешними интерфейсами тестируемой системы.

Тестовые сценарии, выполненные по методу белого ящика:

Тестовый сценарий № 1:

|  |  |
| --- | --- |
| **Тестовый пример #** | 1 |
| **Приоритет тестирования** | Низкий |
| **Заголовок/название теста** | Перейдите на страницу «FullAdress» и нажмите на пункт «Фильтр по названию» и выберете любой параметр |
| **Краткое изложение теста** | Должно поменяться количество записей при использовании фильтрации |
| **Этапы теста** | 1. Запустить программу через Visual Studio  2. Попасть в главное меню. Оттуда перейти на страницу «FullAdress»  3. Затем нажать в контекстном меню у пункта «Фильтр по названию» любой параметр |
| **Тестовые данные** | После использования «Фильтра по названию» с любым параметром все записи должны отфильтроваться по выбранному параметру |
| **Ожидаемый результат** | Если будет использоваться фильтрация, то количество записей должно изменяться |
| **Фактический результат** | Количество записей не изменилось |
| **Статус** | Незачет |
| **Предварительное условие** | Исправьте код, чтобы фильтр по типу материала была выполнена корректно |
| **Постусловие** | Подсчет количества записей работает совместно с фильтрацией |
| **Примечания/комментарии** | Подсчёт количества осуществляется через присвоение значения, полученного от метода «Count» у коллекции таблицы, текстовому элементу |

Тестовый сценарий № 2:

|  |  |
| --- | --- |
| **Тестовый пример #** | 2 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Перейдите на страницу «FullAdress» и введите текст в «Поиск» |
| **Краткое изложение теста** | Должен выводиться список контрагентов c поиском по всем параметрам |
| **Этапы теста** | 1. Запустить программу через Visual Studio.  2. Попасть в главное меню. Оттуда перейти на страницу «FullAdress».  3. Затем нажать ввести в поиск любой текст |
| **Тестовые данные** | После ввода текста, находящегося в одном из полей строки, данные выводятся правильно |
| **Ожидаемый результат** | Если будет использоваться поиск, то он будет выводить данные по всем полям |
| **Фактический результат** | Список контрагентов выводится корректно |
| **Статус** | Зачет |
| **Предварительное условие** | - |
| **Постусловие** | Вывод данных с поиском по всем полям |
| **Примечания/комментарии** | Поиск осуществляется через выражение LINQ |

Участок кода с навигацией на страницу FullAdress работающий некорректно:

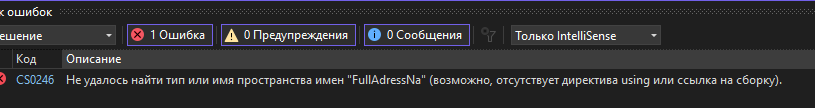


Рис. 16 «Неправильный тип или имя пространства имен»

Участок кода с навигацией на страницу FullAdress работающий корректно:

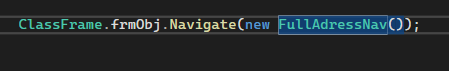


Рис. 17 «Навигация на страницу FullAdressNav»

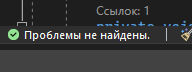


Рис. 18 «Исправленная функция»

Участок кода с присвоением переменной данных из БД работающий некорректно:



Рис. 19 «Неправильное название функции Binding»

Участок кода с присвоением переменной данных из БД работающий корректно:

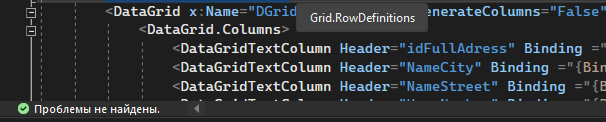


Рис. 20 «Исправленное название функции Binding»

# 5. Методы и средства проведения расчётов оценки трудоёмкости разработки проекта (или Методы и средства защиты БД)

## Заключение

В рамках данной курсовой работы разработана программа для управления предприятием, которая успешно соответствует всем требованиям пользователя. Это было достигнуто благодаря анализу и учету существующей готовой продукции.

В первую очередь рассмотрена общая информация о внутренней структуре организации, а также её бизнес-процессы. Это позволило глубже понять специфику работы предприятия и корректно отразить её в разрабатываемой программе.

В ходе выполнения работы создана база данных, предназначенная для учета товара и последующих продаж. Эта база данных обеспечивает полную информацию о продаже товаров, наличии на складах, а также о поставщиках и покупателях.

Важным аспектом является наличие приложенных документов: приходная накладная, расходная накладная, расчетная ведомость. Эти документы являются неотъемлемой частью разработанной программы и обеспечивают полноценное взаимодействие с ней.

Следует отметить, что данная программа создана с использованием платформы 1С: Предприятие. Выбор этой платформы обусловлен её отечественным происхождением, своевременным обновлением функционала и способностью создавать решения для удобного использования информацией. Таким образом, программа полностью удовлетворяет поставленным требованиям в курсовой работе.

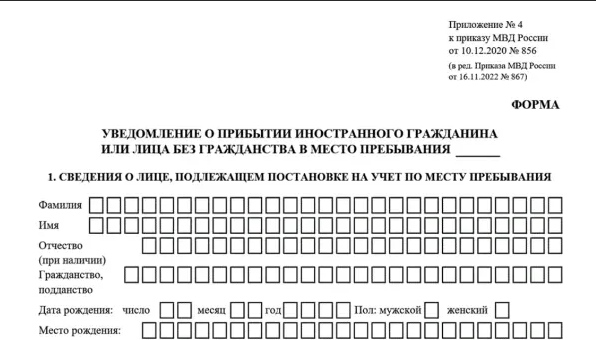
Руководство программиста позволит сопровождать и модифицировать программное решение другим специалистам.

В результате, разработанная программа представляет собой эффективное решение для управления предприятием, реализованы основные этапы жизненного цикла программного обеспечения

## Приложение

Документы

Приложение 1 «Форма Регистрации»



Приложение 2 «Свидетельство»



## Список литературы

1. Joseph, J. Bambara SQL Server® Developer's Guide / Joseph J. Bambara, Paul R. Allen. - Москва: Мир, 2016. - 235 c.

2. Kalen, Delaney Inside Microsoft® SQL Server(TM) 2005: Query Tuning and Optimization / Kalen Delaney и др. - М.: Microsoft Press, 2014. - 448 c.

3. Аллен, Г. Тейлор SQL для чайников / Аллен Г. Тейлор. - М.: Диалектика, Вильямс, 2015. - 416 c.

4. Грабер, Мартин SQL для простых смертных / Мартин Грабер. - М.: ЛОРИ, 2014. - 378 c.

5. Кузнецов, Владимир Новейший самоучитель по 1С:Бухгалтерии 8 / Владимир Кузнецов , Сергей Засорин. - М.: БХВ-Петербург, 2017. - 336 c.

6. Чистов, Д. В. Практикум по программе "1С:Упрощенка 8" / Д.В. Чистов, С.А. Харитонов. - М.: 1С-Паблишинг, 2020. - 436 c.

7. Гудсон, Джон Практическое руководство по доступу к данным (+ DVD-ROM) / Джон Гудсон , Роб Стюард. - М.: БХВ-Петербург, 2013. - 304 c.

8. Филатова, В. 1С для начинающих. Понятный самоучитель / В. Филатова. - М.: Питер, 2018. - 256 c.

9. Рудаков А.В., Федорова Г.Н. – Технология разработки программных продуктов. - М.: Академия, 2018. - 206 с.

10. Рудаков А.В., Федорова Г.Н. – Технология разработки программных продуктов. Практикум. - М.: Академия, 2014. - 189 с.

11. Бьюли, А. Изучаем SQL / А. Бьюли. - М.: Символ-плюс, 2014. - 108 c.