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 обмена валют в пунктах обмена валют

МДК 01.01 «Разработка программных модулей»

**Выполнил:**

Абуталыблы Ильяс Ядигар оглы

студент группы ИСП.20А

09.02.07 Информационные системы и программирование

очной формы обучения

**Руководитель:**

Кузьмина Елена Евгеньевна­­­­­­­­­­­­­­­­­­

Оценка

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

Подпись руководителя

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

Оглавление

[**Введение** 3](#_Toc131591138)

[1. Назначение разработки 4](#_Toc131591139)

[2. Требования к программному модулю 5](#_Toc131591140)

[2.1 Требования к функциональным характеристикам 5](#_Toc131591141)

[2.2 Требования к составу и параметрам технических средств 5](#_Toc131591142)

[2.3 Требования к информационной и программной совместимости 6](#_Toc131591143)

[3. Руководство программиста 6](#_Toc131591144)

[3.1 Разработка базы данных 6](#_Toc131591145)

[3.2 Разработка программного модуля 16](#_Toc131591146)

[4. Руководство пользователя 28](#_Toc131591147)

[4.1 Выполнение программного модуля 28](#_Toc131591148)

[4.2 Сообщения пользователю 33](#_Toc131591149)

[5.Тестирование программного модуля 34](#_Toc131591150)

[5.1Модульное тестирование 34](#_Toc131591151)

[**5.2Тестовые сценарии** 36](#_Toc131591152)

[**Заключение** 41](#_Toc131591153)

[Список литературы: 42](#_Toc131591154)

**Введение**

Важной частью экономики является обмен валют по актуальным курсам, призванная обеспечить население возможностью обменивать валюты между собой по актуальным курсам. Задачей пункта обмена валют является обмен валют по самым актуальным курсам

Общедоступность возможности обменивать валюты обеспечивается развитием количества построенных пунктов.

Обменный пункт – это защищенная кабина (стационарная или передвижная) предназначенная для обмена валют и других операций с денежными средствами.

1. Назначение разработки

Разработка проводится на основании приказа о курсовом проектировании. Многофункционального центра "МФЦ" 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. Информационная система может быть описана данными, сгруппированными следующим образом:

* FullAdress (idFullAdress, NameCity, NameStreet, HomeNumber, Apartment),
* Passport (IdPassport, Type\_of\_passport, Issued\_by, PassNumber)
* Registration (IdRegistration, FIO, Date\_of\_Birth, Place\_of\_Birth, Family\_status, Where\_did\_come\_from, Passport, Place\_of\_discharge, Registration\_date, Date\_of\_discharge, idWorker)
* Worker (IdWorker, FIO, DateOfBirth, PhoneNumber).

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

Приложение должно выполнять следующие функции:

1) Поиск данных (Поиск Регистрации по названию).

2) Фильтрацию данных (Фильтрация Регистрации по типу).

3) Добавление данных (Добавление новой Регистрации).

4) Удаление данных (Удаление данных о Регистрации).

5) Редактирование данных (Редактирование данных о Регистрации).

6)Кнопка возвращение на прошлую страницу.

* 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, Windows 11 |
| Видеокарта | GIGABYTE GeForce GTX 550 |
| Место на жестком диске | От 30 Гб |

Таблица№2 «Оптимальный характеристики компьютера»

|  |  |
| --- | --- |
| Процессор | Intel Core i3-7100 |
| 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: Cashier, Currency, Deal, Klient.

Описание сущностей:

FullAdress – хранит данные о Адресе.

Passport – хранит данные о Пасспорте.

Registration – хранит данные о Регистрции.

Worker – хранит данные Работниках.

Таблица№3 «Cashier»

|  |  |  |  |
| --- | --- | --- | --- |
| **FullAdress** | | | |
| Поле | Тип | Атрибуты | Описание |
| idFullAdress | INT | PK NN AI | Уникальный номер адреса |
| NameCity | NVARCHAR (50) | NN | Название города |
| NameStreet | NVARCHAR (50) | NN | Название улицы |
| HomeNumber | NVARCHAR (50) | NN | Номер дома |
| Apartment | NVARCHAR (50) | NN | Квартира |

Таблица№4 «Currency»

|  |  |  |  |
| --- | --- | --- | --- |
| **Currency** | | | |
| Поле | Тип | Атрибуты | Описание |
| IdPassport | INT | PK NN AI | Уникальный номер Валюты |
| Type\_of\_passport | NVARCHAR (50) | NN | Название Валюты |
| Issued\_by | FLOAT | NN | Курс покупки |
| PassNumber | FLOAT | NN | Курс продажи |

Таблица№5 «Deal»

|  |  |  |  |
| --- | --- | --- | --- |
| **Deal** | | | |
| Поле | Тип | Атрибуты | Описание |
| Id\_Deal | INT | PK NN AI | Уникальный номер сделки |
| id\_Cashier | INT | NN | Номер кассира |
| id\_Client | INT | NN | Номер клиента |
| Deal\_Date | date | NN | Дата сделки |
| Deal\_Time | time(7) | NN | Время сделки |
| Sum\_sold\_currency | FLOAT | NN | Сумма проданной валюты |
| Sum\_buy\_currency | FLOAT | NN | Сумма купленной валюты |
| id\_sold\_currency | INT | NN | Номер проданной валюты |
| id\_buy\_currency | INT | NN | Номер купленной валюты |

Таблица№6 «Klient»

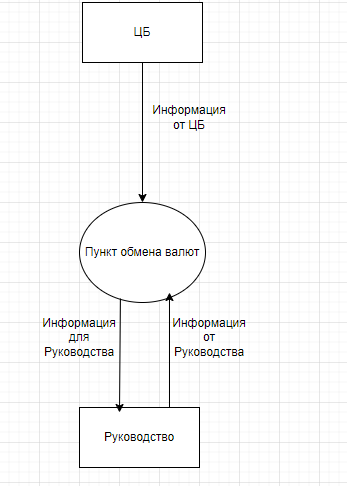
|  |  |  |  |
| --- | --- | --- | --- |
| **Klient** | | | |
| Поле | Тип | Атрибуты | Описание |
| Id\_Client | INT | PK NN AI | Уникальный номер клиента |
| FIO | NVARCHAR(50) | NN | ФИО клиента |
| KlientPassport | INT | NN | Номер паспорта клиента |

Для соединения таблиц использовались следующие типы связей:

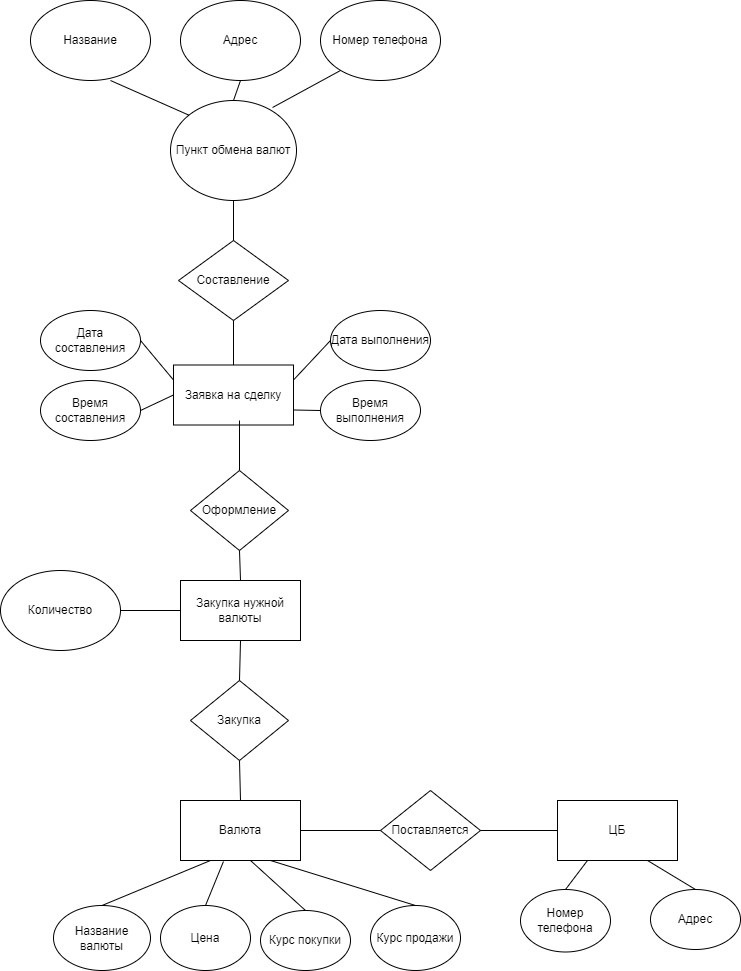
 - один ко многим вида: not-**identifying relationship**

Заполнение базы данных

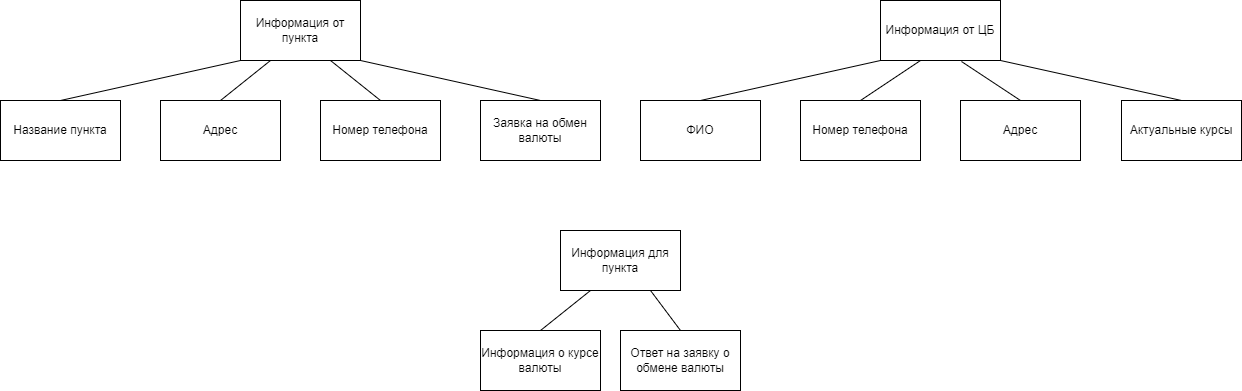
Диаграммы:

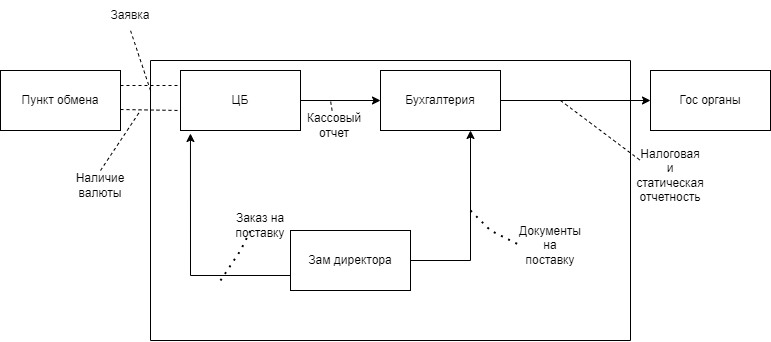


Рис№1 «Начальная контекстная диаграмма»

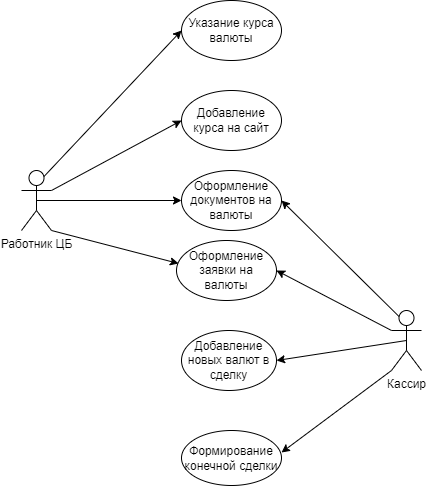


Рис№2 «Диаграмма "сущность-связь"»

Рис№3 «Диаграмма структур данных»

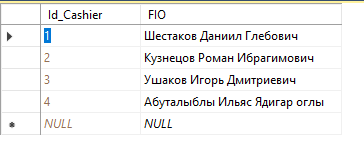


Рис№4 «Информационная модель»

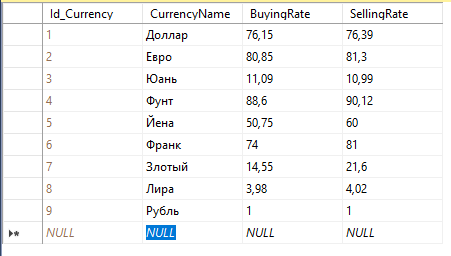
****

Рис№5 «UML диаграмма»

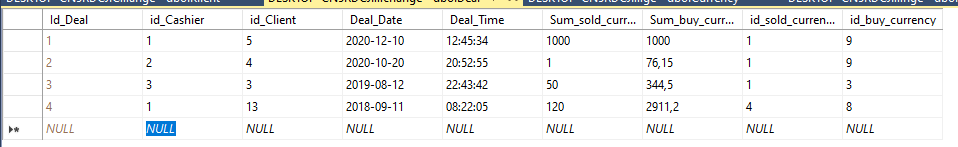
Заполнение базы данных



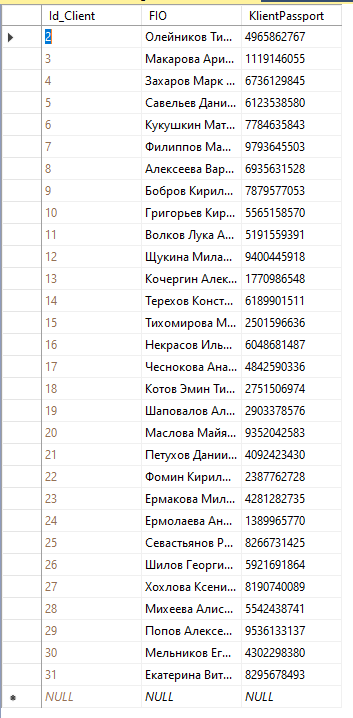
Рис№6 «Cashier»



Рис№7 «Currency»



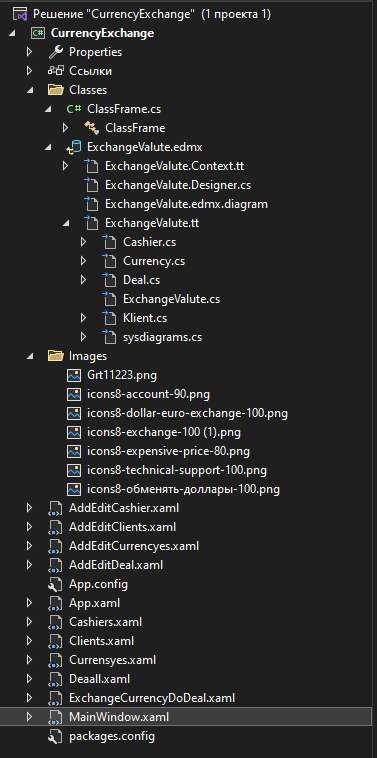
Рис№8 «Deal»



Рис№9 «Klient»

* 1. Разработка программного модуля

Структура проекта для приложения выглядит так:



Рис№10 «Структура проекта»

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

<Window x:Class="CurrencyExchange.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:CurrencyExchange"

mc:Ignorable="d"

Title="MainWindow" Height="650" Width="1000">

<Grid Background="#293133">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="100"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="100"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Image Source="/Images/icons8-обменять-доллары-100.png" Grid.Row="0" Grid.Column="0" Height="90" Width="90"/>

<!--Функции-->

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

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

<StackPanel Width="200">

<Button Height="70" Width="70" Margin="10" Click="Button\_Click\_ExchangeValueee">

<Image Source="/Images/icons8-exchange-100 (1).png" />

</Button>

</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 Height="70" Width="70" Click="Button\_Click\_Klient">

<Image Source="/Images/icons8-account-90.png" />

</Button>

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

<Image Source="/Images/icons8-technical-support-100.png" />

</Button>

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

<Image Source="/Images/icons8-expensive-price-80.png" />

</Button>

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

<Image Source="/Images/icons8-dollar-euro-exchange-100.png" />

</Button>

</StackPanel>

</Border>

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

</Grid>

</Window>

**Main Window.xaml.cs**

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

ClassFrame.frmObj = FrmCurExc;

}

В приложении было создано три основных папки: Classes, images.

Папка Classes содержит в себе:

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

namespace CurrencyExchange.Classes

{

internal class ClassFrame

{

public static Frame frmObj;

}

}

**ExchangeValute (Подключенная база данных).**

EntityFramework — это набор технологий в ADO.NET, которые поддерживают разработку программных приложений, ориентированных на данные.

public partial class CurrencyExchangeEntities : DbContext

{

private static CurrencyExchangeEntities \_context;

public CurrencyExchangeEntities()

: base("name=CurrencyExchangeEntities")

{

}

public static CurrencyExchangeEntities GetContext()

{

if (\_context == null)

\_context = new CurrencyExchangeEntities();

return \_context;

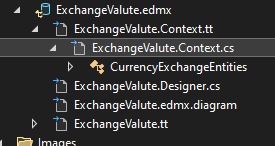
}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

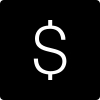
}

****

Рисунок№11 «CurrencyExchange »

Папка Photo содержит в себе: (Для хранения изображении)

**Icon.icon**



Рисунок№12 «Логотип приложения»

Папка Pages содержит в себе: (Для хранения страниц)

**Cashier.xaml(см.** **Рисунок№18)**

<Page x:Class="CurrencyExchange.Cashiers"

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

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

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

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

xmlns:local="clr-namespace:CurrencyExchange"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="900"

Title="Cashiers" IsVisibleChanged="Page\_IsVisibleChanged">

<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="idCashier" Binding ="{Binding Id\_Сashier}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="name" Binding ="{Binding FIO}" 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"></Button>

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

<TextBlock Text="Фильтрация по названию"/>

<ComboBox x:Name="CmbFilterCashier"

Height="30"

SelectionChanged="CmbFilterCashier\_SelectionChanged"/>

</StackPanel>

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

<TextBlock Text="Поиск по названию"/>

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

</StackPanel>

</Grid>

</Page>

**Фильтр по имени:**

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

{

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

DGridCashiers.ItemsSource = CurrencyExchangeEntities.GetContext().Cashier.Where(x => x.Id\_Сashier == id).ToList();

}

**Поиск по имени:**

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

{

if (DGridCashiers.ItemsSource != null)

{

DGridCashiers.ItemsSource = CurrencyExchangeEntities.GetContext().Cashier.Where(x => x.FIO.ToLower().Contains(SearchCashier.Text.ToLower())).ToList();

}

if (SearchCashier.Text.Count() == 0) DGridCashiers.ItemsSource = CurrencyExchangeEntities.GetContext().Cashier.ToList();

}

**Удаление кассира:**

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

{

var cashierForRemoving = DGridCashiers.SelectedItems.Cast<Cashier>().ToList();

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

{

try

{

CurrencyExchangeEntities.GetContext().Cashier.RemoveRange(cashierForRemoving);

CurrencyExchangeEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены!");

DGridCashiers.ItemsSource = CurrencyExchangeEntities.GetContext().Cashier.ToList();

}

catch (Exception ex)

{

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

}

}

}

**Переход на страницу добавление кассира:**

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

{

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

}

**Clients.xaml(см.** **Рисунок№17)**

<Grid Background="#293133">

<Grid.RowDefinitions>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

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

<DataGrid.Columns>

<DataGridTextColumn Header="idKlient" Binding ="{Binding Id\_Client}" Width="\*"></DataGridTextColumn>

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

<DataGridTextColumn Header="Passport" Binding ="{Binding KlientPassport}" 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"></Button>

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

<TextBlock Text="Фильтрация по названию"/>

<ComboBox x:Name="CmbFilterKlients"

Height="30"

SelectionChanged="CmbFilterKlients\_SelectionChanged"/>

</StackPanel>

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

<TextBlock Text="Поиск по названию"/>

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

</StackPanel>

</Grid>

**Переход на страницу клиенты:**

private void Button\_Click\_Klient(object sender, RoutedEventArgs e)

{

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

}

**Currensyes.xaml(см.** **Рисунок№17)**

<Grid Background="#293133">

<Grid.RowDefinitions>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

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

<DataGrid.Columns>

<DataGridTextColumn Header="idCurrency" Binding ="{Binding Id\_Сurrency}" Width="\*"></DataGridTextColumn>

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

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

<DataGridTextColumn Header="SellingRate" Binding ="{Binding SellingRate}" 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="CmbFilterCurrency"

Height="30"

SelectionChanged="CmbFilterCurrency\_SelectionChanged"/>

</StackPanel>

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

<TextBlock Text="Поиск по названию"/>

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

</StackPanel>

</Grid>

**Переход на страницу валюты:**

private void Button\_Click\_Currency(object sender, RoutedEventArgs e)

{

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

}

**Currensyes.xaml(см.** **Рисунок№17)**

<Grid Background="#293133">

<Grid.RowDefinitions>

<RowDefinition Height="5\*"/>

<RowDefinition Height="1\*"/>

</Grid.RowDefinitions>

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

<DataGrid.Columns>

<DataGridTextColumn Header="IdDeal" Binding ="{Binding Id\_Deal}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Date" Binding ="{Binding Deal\_Date}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Time" Binding ="{Binding Deal\_Time}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="SumSoldCurrency" Binding ="{Binding Sum\_sold\_currency}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="SumBuyCurrency" Binding ="{Binding Sum\_buy\_currency}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="SoldCurrency" Binding ="{Binding id\_sold\_currency}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="BuyCurrency" Binding ="{Binding id\_buy\_currency}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Cashier" Binding ="{Binding id\_Cashier}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Client" Binding ="{Binding id\_Client}" Width="\*"></DataGridTextColumn>

</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"/>

</Grid>

**Переход на страницу сделки:**

private void Button\_Click\_ExchangeCurrency(object sender, RoutedEventArgs e)

{

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

}

# Руководство пользователя

## 4.1 Выполнение программного модуля

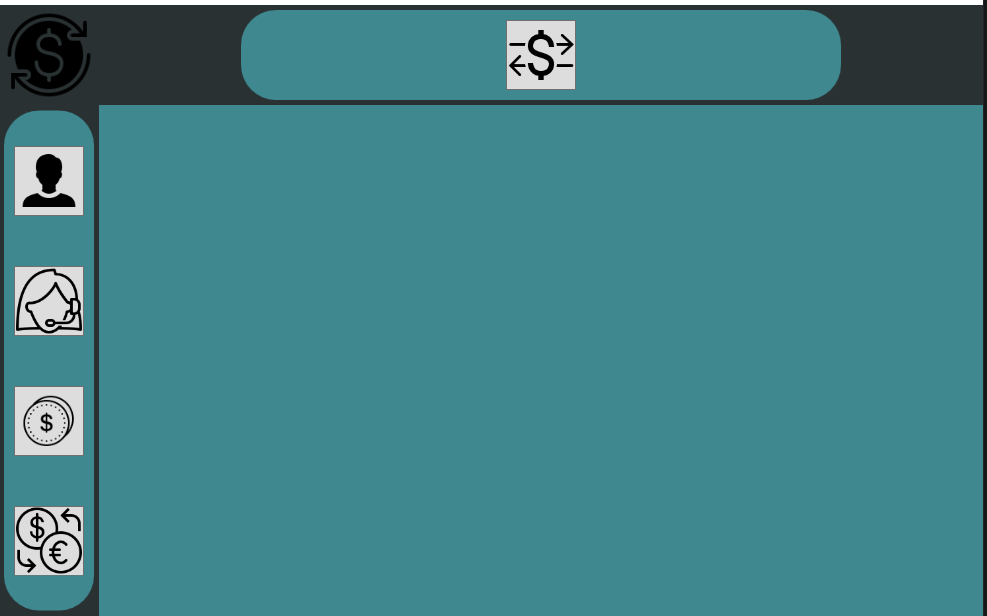
Для использования приложение необходимо:

1) Запустить exe файл приложения (CurrencyExchange.exe)



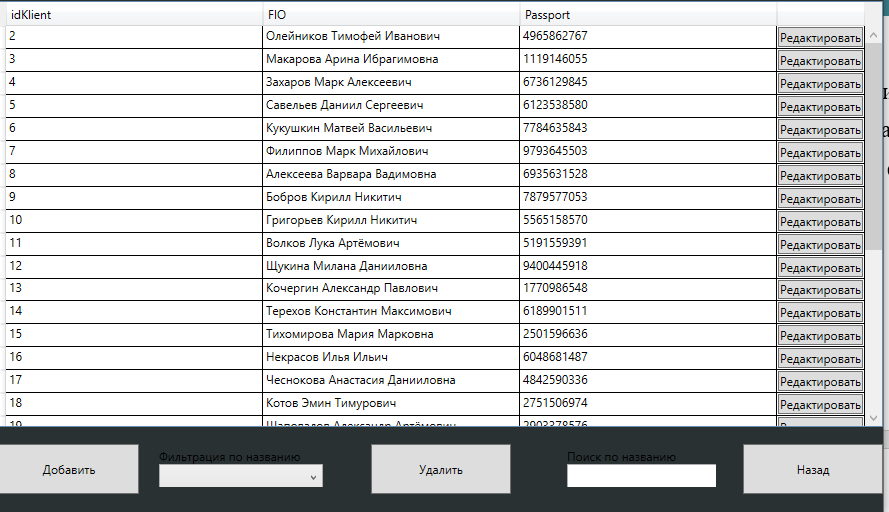
Рисунок№16 «exe файл»

2) После запуска exe файла пользователь попадает на «Главное Меню» приложения, где возможен переход на разные страницы



Рисунок№13 «Главное Меню»

3) При нажатии на кнопку с изображением мужчины «Клиенты» (в главном меню приложения) пользователь попадает на страницу Клиенты. На этой страницы пользователь может выполнять следующие функции:



Рисунок№14 «Страница Клиенты»

3.1) Фильтр по имени.



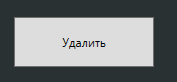
Рисунок№15 «Фильтр»

3.2) Поиск клиента по имени.



Рисунок№16 «Поиск»

3.3) Удаление клиента.



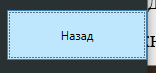
Рисунок№17 «Удаление»

3.4) Редактирование.



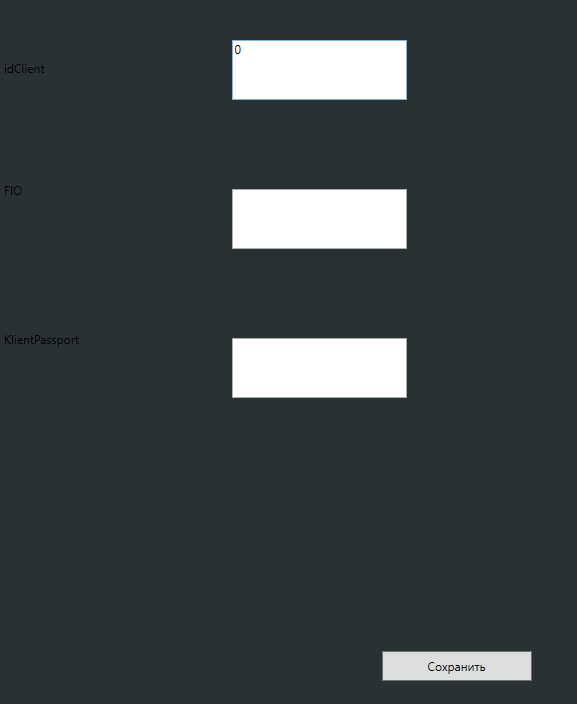
Рисунок№18 «Редактирование»

3.5) Возможен переход на предыдущую страницу.



Рисунок№19 «Кнопка назад»

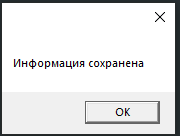
3.6) Добавление нового Клиента (Необходимо заполнить поля данными и для добавления нажать кнопку «Добавить»).



Рисунок№20 «Добавление клиента»

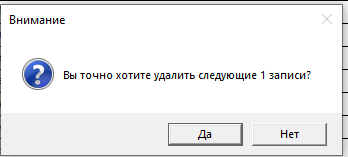
# 4.2 Сообщения пользователю

1.При добавлении нового клиента пользователь получает сообщение: «Информация сохранена»

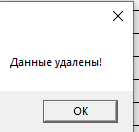


Рисунок№21 «Сообщение при добавлении»

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



Рисунок№22 «Предупреждение при удалении»



Рисунок№23 «Сообщение об удалении»

# 5.Тестирование программного модуля

## 5.1Модульное тестирование

1)Код С# модульных тестов

В качестве образца возьмем поле с курсом. В данной ситуации тест пройдет успешно

1. Данный тест проверяет чтобы курс определялся точно

[TestMethod]

public void TestMethod1()

{

double current\_buy\_rate = 20;

double current\_sell\_rate = 10;

double current\_trade\_rate = 2;

MyClLab c = new MyClLab();

double actual = c.division(current\_buy\_rate, current\_sell\_rate);

Assert.AreEqual(current\_trade\_rate, actual);

}

В качестве образца возьмем поле с курсом. В данной ситуации тест пройдет успешно

1. Данный тест проверяет чтобы курс не была отрицательной.

[TestMethod]

public void Negative()

{

double expected = 1;

double kyrsValute = -10;

double current\_trade\_rate = 0;

if (kyrsValute < 0)

{

current\_trade\_rate += 1;

}

double actual = current\_trade\_rate;

Assert.AreEqual(current\_trade\_rate, actual);

}

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

3. Данный тест проверяет чтобы сумма обмениваемой валюты не была больше 2000.

[TestMethod]

public void KyrsLittle()

{

int expected = 1;

double Kyrs = 2100;

int result = 0;

if (Kyrs > 2000)

{

result += 1;

}

double actual = result;

Assert.AreEqual(expected, actual);

}

Итоги тестов:

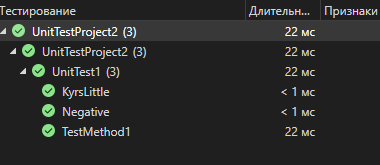


Рис. 24 «Итог тестов»

## **5.2Тестовые сценарии**

Таблица№11 «Аннотация теста»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Название проекта | CurrencyExchange |
| Рабочая версия | 1.0 |
| Имя тестирующего | Абуталыблы Ильяс Ядигар оглы |
| Дата(ы) теста | 03.04.2023 |

Таблица№12 «TestMethod1»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | TestMethod1 |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Определяет точность курса |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка точности курса данной валюты. |
| Этапы теста | Зайти на страницу обмена, указать следующие значения: Продаваемую валюту, покупаемую валюту |
| Тестовые данные | Название валюты: Рубль, Курс:76,15 |
| 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. |

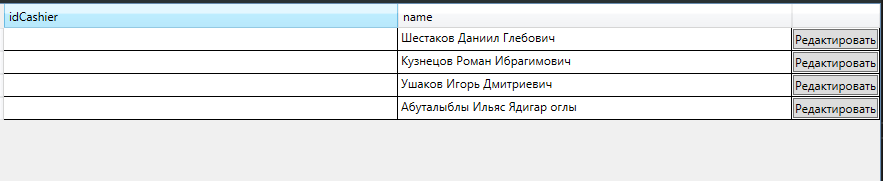
Таблица№13 «Negative»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | Negative |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Проверка на отрицательность суммы. |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка наличия отрицательности суммы сделки. Сумма не должна быть отрицательной. |
| Этапы теста | Зайти на страницу обмена, указать следующие значения: Продаваемую валюту, покупаемую валюту. |
| Тестовые данные | Название валюты: Рубль, Курс:76,15. |
| 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 «KyrsLittle»

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| Тестовый пример # | KyrsLittle |
| Приоритет тестирования  (Низкий/Средний/Высокий) | Средний. |
| Заголовок/название теста | Проверка на высокую сумму сделку. |
| Краткое изложение теста | При сохранении информации цены, должна быть проверка наличия высокой суммы сделки. Сумма сделки не должна быть слишком высокой. |
| Этапы теста | Зайти на страницу обмена, указать следующие значения: Продаваемую валюту, покупаемую валюту. |
| Тестовые данные | Название товара: Рубль, Курс:76,15. |
| 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 выводит данные в таблице.



Рисунок№25 «Не выводит данные»

* 1. Необходимо дописать код.

DataGridTextColumn Header="idCashier" Binding ="{Binding Id\_Сashier}" Width="\*"></DataGridTextColumn>

1. Некорректная отображение картинки.



Рисунок№26 «Некорректная отображение картинки»

2.1) Необходимо добавить изображение в кнопку.

<Button Height="70" Width="70" Margin="10" Click="Button\_Click\_ExchangeValueee">

<Image Source="/Images/icons8-exchange-100 (1).png"/>

</Button>

1. Не работает функция добавление Кассира.

3.1) Необходимо дописать код

private void BtnSave\_Click\_1(object sender, RoutedEventArgs e)

{

StringBuilder errors = new StringBuilder();

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (\_cashiers.Id\_Сashier == 0)

CurrencyExchangeEntities.GetContext().Cashier.Add(\_cashiers);

try

{

CurrencyExchangeEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена");

}

catch (Exception ex)

{

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

}

}

# **Заключение**

Целью данного курсового проекта была создание программы обмена валют учета в "Пункт обмена валюты". 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 валюты;

- позволяет добавлять данные в сделку.

Данная программа позволяет исполнять работу в пункте обмена валюты.

В данном проекте была разработана автоматизированная система обмена валют, которая позволяет быстро обменивать валюты по актуальному курсу, так же она позволяет быстро составлять на основе информации о валюту бланк сделки.

Благодаря внедрению данной системы появилась возможность, по требованию руководства, оперативно выдавать информацию о совершенных сделках, о обмене валюты клиентам, о том какие валюты имеются 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. Агуров, Павел C#. Сборник рецептов / Павел Агуров. - М.: "БХВ-Петербург", 2012. - 432 c.

2. Троелсен, Эндрю Язык программирования С# 2008 и платформа .NET 3.5 / Эндрю Троелсен. - М.: Вильямс, 2010. - 370 c

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

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