Приложение А. Текст программы

АННОТАЦИЯ

В данном программном документе приведен текст программы для информационной системы автосервиса

1. ТЕКСТ ПРОГРАММЫ
   1. Наименование программы

Наименование – «AutoService».

* 1. Область применения программы

Программа предназначена для управления базой данных автосервиса.

* 1. Модули

###### Таблица 10 - Модули

| № | Название модуля | Описание модуля | Размер модуля в Кб | Кол-во строк |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 1 | AdminDataBaseControlPage.xaml | Отображение страницы управления БД | 1.6 kB | 33 |
| 2 | AdminDataBaseControlPage.xaml.cs | Создание резервных копий и восстановление данных | 2.2 kB | 68 |
| 3 | AdminPage.xaml | Отображение страницы администратора | 5.5 kB | 86 |
| 4 | AdminPage.xaml.cs | Управление пользователями | 12.0 kB | 210 |
| 5 | ClientOrderCreationPage.xaml | Отображение страницы оформления заказа клиентом | 4.9 kB | 71 |
| 6 | ClientOrderCreationPage.xaml.cs | Оформление заказа клиентом | 3.7 kB | 84 |
| 7 | ClientOrderPage.xaml | Отображения страницы заказов клиента | 3.0 kB | 61 |
| 8 | ClientOrderPage.xaml.cs | Отображение оформленных заказов клиентом | 4.7 kB | 99 |
| 9 | ClientPage.xaml | Отображение страницы автомобилей клиента | 4.7 kB | 73 |
| 10 | ClientPage.xaml.cs | Управление машинами клиента | 4.1 kB | 108 |
| 11 | DBHelper.cs | Формирование запросов в БД | 13.9 kB | 404 |
| 12 | DiagramAdmin.xaml | Отображение графика статистики пользователей | 3.2 kB | 69 |
| 13 | DiagramAdmin.xaml.cs | Формирование графиков данных | 4.2 kB | 85 |
| 14 | EmailHelper.cs | Работа с почтовым клиентом | 3.0 kB | 75 |
| 15 | FrameManager.cs | Маршрутизация приложения | 264 B | 14 |
| 16 | MainWindow.xaml | Главное окно | 4.0 kB | 60 |
| 17 | MainWindow.xaml.cs | Инициализация главного окна | 3.4 kB | 95 |
| 18 | ModalWindowAddClient.xaml | Модальное окно для дополнительных данных | 2.0 kB | 40 |
| 19 | ModalWindowAddClient.xaml.cs | Заполнение дополнительных данных | 2.7 kB | 80 |
| 20 | PurchasePage.xaml | Отображение страницы отдела закупок | 14.2 kB | 263 |
| 21 | PurchasePage.xaml.cs | Управление данными связанными с поставкой | 8.6 kB | 162 |
| 22 | RecoveryPage.xaml | Отображение страницы восстановления пароля | 3.7 kB | 58 |
| 23 | RecoveryPage.xaml.cs | Восстановление пароля | 2.6 kB | 76 |
| 24 | RepairPage.xaml | Отображение страницы ремонтного отдела | 5.4 kB | 92 |
| 25 | RepairPage.xaml.cs | Управление данными связанными с ремонтом | 5.4 kB | 105 |
| 26 | SalesDepartmentPage.xaml | Отображение страницы ремонтного отдела | 7.0 kB | 128 |
| 27 | SalesDepartmentPage.xaml.cs | Управление данными связанными с продажами | 5.5 kB | 108 |
| 28 | SignInPage.xaml | Отображение страницы авторизации | 2.8 kB | 42 |
| 29 | SignInPage.xaml.cs | Авторизация | 3.2 kB | 84 |
| 30 | SignUpPage.xaml | Отображение страницы регистрации | 4.7 kB | 73 |
| 31 | SignUpPage.xaml.cs | Регистрация | 2.0 kB | 53 |
| 32 | ValidationData.cs | Валидация данных | 8.7 kB | 242 |
| 33 | WarehousePage.xaml | Отображение страницы кладовщика | 6.7 kB | 125 |
| 34 | WarehousePage.xaml.cs | Управление данными связанными со складским учетом | 5.7 kB | 117 |

* 1. Код программы

# AdminDataBaseControlPage.xaml

<Page x:Class="AutoService.AdminDataBaseControlPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="AdminDataBaseControlPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="80\*"/>

<ColumnDefinition Width="80\*"/>

<ColumnDefinition Width="80\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

<RowDefinition Height="70\*"/>

</Grid.RowDefinitions>

<Button Grid.Column="1" Grid.Row="1" Content="Создать резервную копию" Name="btnRezCopy" Click="btnRezCopy\_Click"/>

<Button Grid.Column="1" Grid.Row="3" Content="Выполнить Backup данных" Name="btnBackup" Click="btnBackup\_Click"/>

<Button Grid.Column="1" Grid.Row="6" Content="Выполнить откат данных по выбранной дате" Name="btnRollback" Click="btnRollback\_Click"/>

<ComboBox Grid.Column="1" Grid.Row="5" Margin="10,10,10,10" Name="cbDateRollback"/>

</Grid>

</Page>

# AdminDataBaseControlPage.xaml.cs

using Microsoft.Win32;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для AdminDataBaseControlPage.xaml

/// </summary>

public partial class AdminDataBaseControlPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

/// <summary>

/// Инициализация компонентов страницы управления БД

/// </summary>

public AdminDataBaseControlPage()

{

InitializeComponent();

DirectoryInfo dir = new DirectoryInfo((System.IO.Directory.GetCurrentDirectory() + @"\BackupFiles").Replace(@"\bin", "").Replace(@"\Debug", ""));

FileInfo[] Files = dir.GetFiles();

foreach (FileInfo file in Files)

cbDateRollback.Items.Add(file.Name.Insert(2, ".").Insert(5, ".").Replace(".backup",""));

}

/// <summary>

/// Обработчик кнопки резерного копирования

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

SaveFileDialog dialog = new SaveFileDialog()

{

Filter = "BACKUP Files|\*.backup\*"

};

dialog.Title = "Save as backup file";

if (dialog.ShowDialog() == true)

{

dbhelper.Backup(dialog.FileName + ".backup");

}

}

/// <summary>

/// Обработчик кнопки бэкапа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

OpenFileDialog dialog = new OpenFileDialog()

{

Filter = "BACKUP Files|\*.backup\*"

};

dialog.Title = "Choose backup file";

if (dialog.ShowDialog() == true)

{

dbhelper.Restore(dialog.FileName);

}

}

/// <summary>

/// Обработчик отката системы по дате

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.Restore((System.IO.Directory.GetCurrentDirectory() + @"\BackupFiles").Replace(@"\bin", "").Replace(@"\Debug", "")+@"\"+cbDateRollback.SelectedItem.ToString().Replace(".","")+".backup");

}

}

}

# AdminPage.xaml

<Page x:Class="AutoService.AdminPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="AdminPage">

<Grid Name="grdAdmin">

<Grid.RowDefinitions>

<RowDefinition Height="14\*"/>

<RowDefinition Height="211\*"/>

<RowDefinition Height="50\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="21\*"/>

<RowDefinition Height="29\*"/>

<RowDefinition Height="20\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="101\*"/>

<ColumnDefinition Width="62\*"/>

<ColumnDefinition Width="106\*"/>

<ColumnDefinition Width="70\*"/>

<ColumnDefinition Width="56\*"/>

<ColumnDefinition Width="50\*"/>

<ColumnDefinition Width="69\*"/>

<ColumnDefinition Width="47\*"/>

<ColumnDefinition Width="61\*"/>

<ColumnDefinition Width="108\*"/>

<ColumnDefinition Width="66\*"/>

</Grid.ColumnDefinitions>

<TextBox TextChanged="tbSearch\_TextChanged" Name="tbSearch" BorderBrush="#445c93" Grid.Row="0" Background="LightGray" Grid.ColumnSpan="5" Margin="0,0,0.4,0.6"/>

<ComboBox Grid.Row="0" Grid.Column="6" Name="cbYearSearch" SelectionChanged="cbYearSearch\_SelectionChanged" Grid.ColumnSpan="3" Margin="0.2,0,60.6,0.6"/>

<Button Grid.Row="0" Grid.Column="9" Name="btnClearSearch" Content="Убрать фильтр" FontSize="10" Click="btnClearSearch\_Click" />

<Label Content="Фильтр:" Grid.Row="0" Grid.Column="5" FontSize="10" />

<DataGrid Grid.Row="1" Name="dgUsers" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True" Grid.ColumnSpan="11" Margin="0,0.4,0,1" SelectionChanged="Data\_SelectionChanged">

<DataGrid.Columns>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Скрыть" Name="btnHideUser" Click="btnHideUser\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Button Grid.Row="2" VerticalAlignment="Top" Content="Импортировать" Name="btnImport" Click="btnImport\_Click" Grid.Column="1" Grid.ColumnSpan="2" />

<Button Grid.Row="2" Grid.Column="8" VerticalAlignment="Top" Content="Экспортировать" Name="btnEport" Click="btnEport\_Click" Grid.ColumnSpan="2" />

<Label Content="Email:" Grid.Row="3" HorizontalAlignment="Center" />

<Label Content="Пароль:" Grid.Row="5" HorizontalAlignment="Center" />

<Label Content="Должность:" Grid.Row="7" HorizontalAlignment="Center" />

<TextBox Grid.Row="3" Grid.Column="1" Name="tbEmail" Grid.ColumnSpan="2" PreviewTextInput= "tb1\_PreviewTextInputEngNum" DataObject.Pasting="tb1\_PastingEngNum"/>

<TextBox Grid.Row="5" Grid.Column="1" Name="tbPassword" Grid.ColumnSpan="2" PreviewTextInput= "tb1\_PreviewTextInputEngNum" DataObject.Pasting="tb1\_PastingEngNum"/>

<ComboBox Grid.Row="7" Grid.Column="1" Name="cbPost" Grid.ColumnSpan="2">

<ComboBoxItem>Администратор</ComboBoxItem>

<ComboBoxItem>Сотрудник отдела закупок</ComboBoxItem>

<ComboBoxItem>Клиент</ComboBoxItem>

<ComboBoxItem>Сотрудник склада</ComboBoxItem>

<ComboBoxItem>Сотрудник отдела продаж</ComboBoxItem>

<ComboBoxItem>Сотрудник ремонтного отдела</ComboBoxItem>

</ComboBox>

<Label Content="Фамилия:" Grid.Row="3" Grid.Column="5" HorizontalAlignment="Left" />

<Label Content="Имя:" Grid.Row="5" Grid.Column="5" HorizontalAlignment="Left"/>

<Label Content="Отчество:" Grid.Row="7" Grid.Column="5" HorizontalAlignment="Left" />

<TextBox Grid.Row="3" Grid.Column="6" Grid.ColumnSpan="3" Name="tbLastName" PreviewTextInput= "tb1\_PreviewTextInputRus" DataObject.Pasting="tb1\_PastingRus"/>

<TextBox Grid.Row="5" Grid.Column="6" Grid.ColumnSpan="3" Name="tbFirstName" />

<TextBox Grid.Row="7" Grid.Column="6" Grid.ColumnSpan="3" Name="tbMiddleName" />

<Button Grid.Row="9" Grid.Column="2" Content="Добавить" Name="btnAdd" Click="btnAdd\_Click"/>

<Button Grid.Row="9" Grid.Column="4" Content="Изменить" Name="btnUpd" Grid.ColumnSpan="2" Click="btnUpd\_Click"/>

<Button Grid.Row="9" Grid.Column="7" Content="Удалить" Name="btnDel" Grid.ColumnSpan="2" Click="btnDel\_Click"/>

<Button Grid.Row="2" Grid.Column="4" Grid.ColumnSpan="3" VerticalAlignment="Top" Content="Скрытые" Name="btnVisibleUsers" Click="btnVisibleUsers\_Click"/>

<Button Name="btnClear" Grid.Row="5" Grid.Column="9" Grid.ColumnSpan="2" Margin="20,0,20,0" Content="Очистить" Click="btnClear\_Click"/>

</Grid>

</Page>

# AdminPage.xaml.cs

using Microsoft.Win32;

using System;

using System.Collections;

using System.Collections.Generic;

using System.Data;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для AdminPage.xaml

/// </summary>

public partial class AdminPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

//Коллекция годов регистрации

List<string> dateRegistration = new List<string>();

/// <summary>

/// Обработчик выбра значения в DataGrid

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgUsers.SelectedItem == null)

{

return;

}

DataRowView row = (DataRowView)dgUsers.SelectedItem;

tbEmail.Text = row[0].ToString();

tbPassword.Text = row[1].ToString();

tbFirstName.Text = row[2].ToString();

tbLastName.Text = row[3].ToString();

tbMiddleName.Text = row[4].ToString();

cbPost.Text = row[6].ToString();

}

/// <summary>

/// Инициализация компонентов страницы администратора

/// </summary>

public AdminPage()

{

InitializeComponent();

BackupData();

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

dateRegistration = dbhelper.EecuteQueryReader("select distinct extract(year from registrationdate) from Users;", dateRegistration, "date\_part");

foreach(string years in dateRegistration)

cbYearSearch.Items.Add(years.Replace("0:00:00",""));

}

/// <summary>

/// Backup по определенной дате

/// </summary>

private void BackupData()

{

if (DateTime.Now.ToString("dd") == "28")

{

dbhelper.Backup((System.IO.Directory.GetCurrentDirectory() + @"\BackupFiles").Replace(@"\bin", "").Replace(@"\Debug", "")+$@"\{DateTime.Now.ToString("d").Replace(".", "")}.backup");

}

}

/// <summary>

/// Обработчик кнопки импорта

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

try

{

OpenFileDialog op = new OpenFileDialog();

op.Title = "Select as csv file";

op.Filter = "CSV Files|\*.csv";

if (op.ShowDialog() == true)

{

dbhelper.EecuteQuery($@"COPY Users FROM '{op.FileName}' DELIMITER ';' CSV HEADER;");

valid.Show("Данные импортированы").GetAwaiter();

}

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

catch (NotSupportedException)

{

valid.Show("Ошибка загрузки файла").GetAwaiter() ;

}

}

/// <summary>

/// Обработчик кнопки экспорта

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

try

{

SaveFileDialog dialog = new SaveFileDialog()

{

Filter = "DQY Files|\*.dqy\*"

};

dialog.Title = "Save as dqy file";

if (dialog.ShowDialog() == true)

{

var buffer = Encoding.UTF8.GetBytes("XLODBC\n1\nDRIVER={PostgreSQL Unicode(x64)};DATABASE=AutoService;SERVER=192.168.231.204;PORT=5432;UID=postgres;PASSWORD=1;SSLmode=disable;ReadOnly=0;Protocol=7.4;FakeOidIndex=0;ShowOidColumn=0;RowVersioning=0;ShowSystemTables=0;ConnSettings=;Fetch=100;Socket=4096;UnknownSizes=0;MaxVarcharSize=255;MaxLongVarcharSize=8190;Debug=0;CommLog=0;Optimizer=0;Ksqo=1;UseDeclareFetch=0;TextAsLongVarchar=1;UnknownsAsLongVarchar=0;BoolsAsChar=1;Parse=0;CancelAsFreeStmt=0;ExtraSysTablePrefixes=dd\_;LFConversion=1;UpdatableCursors=1;DisallowPremature=0;TrueIsMinus1=0;BI=0;ByteaAsLongVarBinary=0;UseServerSidePrepare=0;LowerCaseIdentifier=0;GssAuthUseGSS=0;XaOpt=1\nselect \* from Users");

using (var fs = new FileStream(dialog.FileName+".dqy", FileMode.OpenOrCreate,

FileAccess.Write, FileShare.None, buffer.Length, true))

{

fs.Write(buffer, 0, buffer.Length);

}

valid.Show("Данные экспортированы").GetAwaiter();

}

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

catch (NotSupportedException)

{

valid.Show("Ошибка сохранения файла").GetAwaiter();

}

}

/// <summary>

/// Обработчик выбора даты регистрации

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if(tbSearch.Text == "" && cbYearSearch.SelectedIndex != -1)

dbhelper.Refresh(dgUsers, "select email as \"email\", password as \"Пароль\", lastname as \"Фамилия\",firstname as \"Имя\",middlename as \"Отчество\", RegistrationDate as \"Дата регистрации\", rolename as \"Роль\", extract(year from registrationdate) as \"Год\"from Users where extract(year from registrationdate) = " + cbYearSearch.SelectedItem.ToString(),null);

else if(cbYearSearch.SelectedIndex != -1)

dbhelper.Refresh(dgUsers, "select email as \"email\", password as \"Пароль\", lastname as \"Фамилия\",firstname as \"Имя\",middlename as \"Отчество\", RegistrationDate as \"Дата регистрации\", rolename as \"Роль\", extract(year from registrationdate) as \"Год\"from Users where extract(year from registrationdate) = " + cbYearSearch.SelectedItem.ToString() + $@" and (email LIKE '%{tbSearch.Text}%' or lastname LIKE '%{tbSearch.Text}%' or firstname LIKE'%{tbSearch.Text}%' or middlename LIKE'%{tbSearch.Text}%' or rolename LIKE'%{tbSearch.Text}%')", null);

}

/// <summary>

/// Обработчик кнопки очищения параметров поиска

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

tbSearch.Text = "";

cbYearSearch.SelectedIndex = -1;

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

/// <summary>

/// Обработчик ввода данных в строку поиска

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (tbSearch.Text == "" && cbYearSearch.SelectedIndex == -1)

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", null);

else if (cbYearSearch.SelectedIndex != -1)

dbhelper.Refresh(dgUsers, "select email as \"email\", password as \"Пароль\", lastname as \"Фамилия\",firstname as \"Имя\",middlename as \"Отчество\", RegistrationDate as \"Дата регистрации\", rolename as \"Роль\", extract(year from registrationdate) as \"Год\"from Users where extract(year from registrationdate) = " + cbYearSearch.SelectedItem.ToString() + $@" and (email LIKE '%{tbSearch.Text}%' or lastname LIKE '%{tbSearch.Text}%' or firstname LIKE'%{tbSearch.Text}%' or middlename LIKE'%{tbSearch.Text}%' or rolename LIKE'%{tbSearch.Text}%')", null);

else if(cbYearSearch.SelectedIndex == -1)

dbhelper.Refresh(dgUsers, "select email as \"email\", password as \"Пароль\", lastname as \"Фамилия\",firstname as \"Имя\",middlename as \"Отчество\", RegistrationDate as \"Дата регистрации\", rolename as \"Роль\", extract(year from registrationdate) as \"Год\"from Users where" + $@" email LIKE '%{tbSearch.Text}%' or lastname LIKE '%{tbSearch.Text}%' or firstname LIKE'%{tbSearch.Text}%' or middlename LIKE'%{tbSearch.Text}%' or rolename LIKE'%{tbSearch.Text}%'", null);

else if(tbSearch.Text == "" && cbYearSearch.SelectedIndex != -1)

dbhelper.Refresh(dgUsers, "select email as \"email\", password as \"Пароль\", lastname as \"Фамилия\",firstname as \"Имя\",middlename as \"Отчество\", RegistrationDate as \"Дата регистрации\", rolename as \"Роль\", extract(year from registrationdate) as \"Год\"from Users where extract(year from registrationdate) = " + cbYearSearch.SelectedItem.ToString(), null);

}

/// <summary>

/// Обработчик кнопки скрытия пользователей

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.EecuteQuery($@"UPDATE Users SET life = false WHERE email = '{dbhelper.GetValueData(dgUsers,"Email")}';");

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

/// <summary>

/// Обработчик кнопки раскрытия пользователей

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.EecuteQuery($@"UPDATE Users SET life = true WHERE life = false;");

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

/// <summary>

/// Обработчик кнопки добавления пользователей в БД

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (valid.IsValidEmail(tbEmail.Text) != false && valid.IsValidPassword(tbPassword.Text) != false && tbLastName.Text != "" && tbFirstName.Text != "" && tbMiddleName.Text != "" && cbPost.Text !="")

{

string[] atributes = { tbEmail.Text, valid.Encrypt(tbPassword.Text), tbLastName.Text, tbFirstName.Text, tbMiddleName.Text, cbPost.Text };

dbhelper.InsertInto("Users", atributes);

if (cbPost.SelectedIndex == 2)

{

ModalWindowAddClient appointment = new ModalWindowAddClient(tbEmail.Text,0,"");

appointment.ShowDialog();

}

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

}

/// <summary>

/// Обработчик кнопки удаления пользователей из БД

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgUsers.SelectedIndex == -1)

return;

if (MessageBox.Show("Вы точно хотите удалить выбранного пользователя?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

if (dbhelper.GetValueData(dgUsers, "Роль") == "Клиент")

{

dbhelper.EecuteQuery($@"DELETE FROM Client WHERE Email = '{dbhelper.GetValueData(dgUsers, "Email")}';");

}

dbhelper.DeleteInto("Users", $@"{dbhelper.GetValueData(dgUsers, "Email")}");

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

}

/// <summary>

/// Обработчик кнопки обновления данных пользователя в БД

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

string password = tbPassword.Text;

if (dbhelper.EecuteQueryReaderOne($@"select count(password) as aaa from users where password = '{tbPassword.Text}'", "aaa") == "0")

password = valid.Encrypt(password);

if (dgUsers.SelectedIndex == -1)

return;

if (valid.IsValidEmail(tbEmail.Text) != false && valid.IsValidPassword(tbPassword.Text) != false && tbLastName.Text != "" && tbFirstName.Text != "" && tbMiddleName.Text != "" && cbPost.Text != "")

{

string[] atributes = { dbhelper.GetValueData(dgUsers, "Email"), password, tbLastName.Text, tbFirstName.Text, tbMiddleName.Text, cbPost.Text, tbEmail.Text};

dbhelper.UpdateInto("Users", atributes);

}

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

/// <summary>

/// Обработчик кнопки очищения текстовых полей

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.Refresh(dgUsers, "select \* from Users\_View; ", grdAdmin);

}

/// <summary>

/// Обработчик вводимых и вставляемых значений в текстовое поле

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void tb1\_PreviewTextInputEngNum(object sender, TextCompositionEventArgs e) => new ValidationData().tb1\_PreviewTextInputAngNum(sender, e);

private void tb1\_PastingEngNum(object sender, DataObjectPastingEventArgs e) => new ValidationData().tb1\_PastingAngNum(sender, e);

private void tb1\_PreviewTextInputRus(object sender, TextCompositionEventArgs e) => new ValidationData().tb1\_PreviewTextInputRus(sender, e);

private void tb1\_PastingRus(object sender, DataObjectPastingEventArgs e) => new ValidationData().tb1\_PastingRus(sender, e);

}

}

# App.xaml

<Application x:Class="AutoService.App"

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

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

xmlns:local="clr-namespace:AutoService"

StartupUri="MainWindow.xaml">

<Application.Resources>

<Style TargetType="Button">

<Setter Property="Background" Value="#445c93"/>

<!--<Setter Property="Width" Value="175"/>

<Setter Property="Height" Value="30"/>-->

<Setter Property="Margin" Value="2.5"/>

<Setter Property="Foreground" Value="White"/>

</Style>

</Application.Resources>

</Application>

# App.xaml.cs

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Linq;

using System.Threading.Tasks;

using System.Windows;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для App.xaml

/// </summary>

public partial class App : Application

{

}

}

# ClientOrderCreationPage.xaml

<Page x:Class="AutoService.ClientOrderCreationPage"

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:AutoService"

xmlns:System="clr-namespace:System;assembly=mscorlib"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="ClientOrderCreationPage">

<Grid Name="grdOrderCreate">

<StackPanel Width="800" VerticalAlignment="Stretch">

<StackPanel HorizontalAlignment="Center" VerticalAlignment="Stretch">

<WrapPanel >

<StackPanel HorizontalAlignment="Center" VerticalAlignment="Stretch">

<Label Content="Выберите услуги" HorizontalContentAlignment="Center" Width="400"/>

<DataGrid Name="dgService" Width="400" HorizontalScrollBarVisibility="Hidden" CanUserReorderColumns="False" CanUserResizeColumns="False" CanUserSortColumns="False" VerticalGridLinesBrush="Transparent" HorizontalGridLinesBrush="Transparent" AlternatingRowBackground="AliceBlue" BorderBrush="Transparent" HeadersVisibility="Column" AutoGenerateColumns="False" ItemsSource="{Binding service}" CanUserAddRows="False" CanUserDeleteRows="False" HorizontalAlignment="Center" VerticalAlignment="Stretch" ScrollViewer.CanContentScroll="False" SelectionChanged="dgService\_SelectionChanged">

<DataGrid.Columns>

<DataGridTemplateColumn Header="Услуги" Width="180">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=servicename}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Header="Стоимость" Width="180">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=price}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Width="40">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<CheckBox Name="chDG" HorizontalAlignment="Left" VerticalAlignment="Center" Checked="chDG\_Checked" Unchecked="chDG\_Unchecked"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<TextBlock Name="tblDescription"/>

</StackPanel>

<Rectangle Width="20"/>

<StackPanel>

<Label Content="Укажите дату" HorizontalAlignment="Center"/>

<DatePicker Width="150" DisplayDateStart="{Binding Mode=OneWay, Source={x:Static System:DateTime.Today}}" Name="dpOrder"/>

</StackPanel>

<Rectangle Width="20"/>

<StackPanel>

<Label Content="Укажите автомобиль" HorizontalAlignment="Center"/>

<ComboBox Width="150" Name="cbCars" />

</StackPanel>

</WrapPanel>

<RichTextBox Name="lbDescription" HorizontalContentAlignment="Center" VerticalContentAlignment="Center"/>

</StackPanel>

</StackPanel>

<StackPanel Width="800" HorizontalAlignment="Stretch" VerticalAlignment="Bottom" Margin="0, 0, 10, 10">

<WrapPanel HorizontalAlignment="Right" >

<Label Content="Итого: " HorizontalAlignment="Right"/>

<Label Name="lbPriceEnding" Content="0" HorizontalAlignment="Right"/>

<Label Content=" руб." HorizontalAlignment="Right"/>

</WrapPanel>

<Button Grid.Row="9" Grid.Column="4" Content="Оформить заказ" Name="btnAddOrder" Click="btnAddOrder\_Click" Grid.ColumnSpan="2" Width="800"/>

</StackPanel>

</Grid>

</Page>

# ClientOrderCreationPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для ClientOrderCreationPage.xaml

/// </summary>

public partial class ClientOrderCreationPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

//Коллекция выбранных услуг

List<string> addService = new List<string>();

/// <summary>

/// Инициализация компонентов страницы оформления заказа

/// </summary>

/// <param name="clientNumber">уникальный индетификатор клиента</param>

public ClientOrderCreationPage(String clientNumber)

{

InitializeComponent();

dbhelper.Refresh(dgService, "select \* from service", grdOrderCreate);

List<string> cars = new List<string>();

dbhelper.EecuteQueryReader($@"select (CarBrand || ' ' || CarModel) as cr from Car where ClientNumber = '{clientNumber}'", cars, "cr");

foreach(string car in cars)

{

cbCars.Items.Add(car);

}

}

/// <summary>

/// Обработчик сheckbox выбранных услуг

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

lbPriceEnding.Content = Convert.ToDouble(lbPriceEnding.Content.ToString()) + Convert.ToDouble(dbhelper.GetValueData(dgService, "price"));

addService.Add(dbhelper.GetValueData(dgService, "servicename"));

}

/// <summary>

/// ООбработчик вheckbox отмененных услуг

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void chDG\_Unchecked(object sender, RoutedEventArgs e)

{

lbPriceEnding.Content = Convert.ToDouble(lbPriceEnding.Content.ToString()) - Convert.ToDouble(dbhelper.GetValueData(dgService, "price"));

addService.Remove(dbhelper.GetValueData(dgService, "servicename"));

}

/// <summary>

/// Обработчик выбора значения в DataGrid

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

lbDescription.Document.Blocks.Clear();

lbDescription.AppendText(dbhelper.EecuteQueryReaderOne($@"select ServiceDescription from Service where ServiceName = '{dbhelper.GetValueData(dgService, "servicename")}'","ServiceDescription"));

}

/// <summary>

/// ООбработчик кнопки оформления заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (lbPriceEnding.Content.ToString() == "0")

valid.Show("Выберете услугу").GetAwaiter();

if(dpOrder.SelectedDate.ToString()!="" && cbCars.SelectedIndex != -1)

{

string ContractNumber = "";

do

{

Random rnd = new Random();

ContractNumber = rnd.Next(111111111, 999999999).ToString();

}

while (dbhelper.EecuteQueryReaderOne($@"SELECT count(ordernumber) FROM OrderClient where ordernumber = '{ContractNumber}'", "count") == "1");

string[] atributes = {ContractNumber, lbPriceEnding.Content.ToString(), dpOrder.SelectedDate.Value.Date.ToString("yyyy.MM.dd"),

dbhelper.EecuteQueryReaderOne($@"select carnumber from car where (CarBrand || ' ' || CarModel) = '{cbCars.SelectedItem.ToString()}'","carnumber") };

dbhelper.InsertInto("orderclient", atributes);

foreach(string service in addService)

{

string[] atributes2 = {service, ContractNumber};

dbhelper.InsertInto("ServiceOrder", atributes2);

}

valid.Show("Заказ оформлен").GetAwaiter();

FrameManager.MainFrame.GoBack();

}

}

}

}

# ClientOrderPage.xaml

<Page x:Class="AutoService.ClientOrderPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="ClientOrderPage">

<Grid Name="grdOrder">

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<DataGrid Name="dgHistory" ItemsSource="{Binding orderclient}" AutoGenerateColumns="False" SelectionChanged="dgHistory\_SelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер заказа" Width="\*" Binding="{Binding ordernumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTemplateColumn Header="Дата" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=orderdate}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Header="Машина" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=carnumber}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Header="Стоимость" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=ordersum}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<StackPanel Grid.Row="1" HorizontalAlignment="Center">

<Label Content="Выбранные услуги:" FontSize="25"/>

<ListBox Name="lbServices"/>

<Label/>

</StackPanel>

<StackPanel Grid.Row="2" HorizontalAlignment="Center" VerticalAlignment="Center">

<Label Content="Статсус:" FontSize="25"/>

<ListBox Name="labelStatus" />

</StackPanel>

<Button Grid.Row="2" Content="Отменить заказ" VerticalAlignment="Bottom" Name="btnCancelOrder" Click="btnCancelOrder\_Click"/>

</Grid>

</Page>

# ClientOrderPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для ClientOrderPage.xaml

/// </summary>

public partial class ClientOrderPage : Page

{

//Экземпляр класса работы с БД

DBHelper dbhelper = new DBHelper();

//Коллекция номеров машин

List<string> carNumbersClient = new List<string>();

//хранения запроса

string Query = "";

//коллекция статусов

List<string> statuses = new List<string>();

//коллекция номеров диагностиуи

List<string> diagnostics = new List<string>();

/// <summary>

/// Инициализация компонентов страницы

/// </summary>

/// <param name="clientNumber">уникальный индетификатор клиента</param>

public ClientOrderPage(String clientNumber)

{

InitializeComponent();

dbhelper.EecuteQueryReader($@"select carnumber from car where clientnumber = '{clientNumber}'", carNumbersClient, "carnumber");

string query = "select \* from orderclient where ";

foreach(string car in carNumbersClient)

{

query += "carnumber = '" + car + "' or ";

}

query += "carnumber = ' '";

dbhelper.Refresh(dgHistory, query, grdOrder);

Query = query;

}

/// <summary>

/// Обработчик выбора значения DataGrid заказов

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

diagnostics.Clear();

statuses.Clear();

lbServices.Items.Clear();

carNumbersClient.Clear();

labelStatus.Items.Clear();

dbhelper.EecuteQueryReader($@"select \* from serviceorder where ordernumber = '{dbhelper.GetValueData(dgHistory, "ordernumber")}'",carNumbersClient, "servicename");

foreach(string service in carNumbersClient)

{

lbServices.Items.Add(service);

}

if (dbhelper.EecuteQueryReaderOne($@"select count(ordernumber) from contract where ordernumber = '{dbhelper.GetValueData(dgHistory, "ordernumber")}'", "count") == "0")

labelStatus.Items.Clear();

else

{

string contractnumber = dbhelper.EecuteQueryReaderOne($@"select contractnumber from contract where ordernumber = '{dbhelper.GetValueData(dgHistory, "ordernumber")}'", "contractnumber");

if (dbhelper.EecuteQueryReaderOne($@"select count(diagnosticsnumber) from diagnostics where contractnumber = '{contractnumber}'", "count") == "0")

labelStatus.Items.Add("Заказ оформлен чек выслан на поту");

else

{

labelStatus.Items.Clear();

diagnostics = dbhelper.EecuteQueryReader($@"select diagnosticsnumber from diagnostics where contractnumber = '{contractnumber}'", diagnostics, "diagnosticsnumber");

foreach (string diag in diagnostics)

{

statuses.Add(dbhelper.EecuteQueryReaderOne($@"select (statustime || ': ' || statuscar) as aaa from statuscar where diagnosticsnumber = '{diag}'", "aaa"));

}

//statuses = dbhelper.EecuteQueryReader($@"select (statustime || ': ' || statuscar) as aaa from statuscar where diagnosticsnumber = '{dbhelper.EecuteQueryReaderOne($@"select diagnosticsnumber from diagnostics where contractnumber = '{contractnumber}'", "diagnosticsnumber")}'", statuses,"aaa");

foreach (string st in statuses)

{

labelStatus.Items.Add(st);

}

}

}

}

/// <summary>

/// Обработчик кнопки отмены заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if(dgHistory.SelectedIndex != -1)

{

if (dbhelper.EecuteQueryReaderOne("select count(ordernumber) from contract", "count") == "0")

{

if (MessageBox.Show("Вы точно хотите удалить выбранный заказ?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("orderclient", $@"{dbhelper.GetValueData(dgHistory, "ordernumber")}");

dbhelper.Refresh(dgHistory, Query, grdOrder);

}

}

else

{

MessageBox.Show("Данный заказ нельзя отменить, так как уже была произведена оплата, обратитесь в отдел продаж");

}

}

}

}

}

# ClientPage.xaml

<Page x:Class="AutoService.ClientPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="ClientPage">

<Grid Name="grdClientCar">

<Grid.RowDefinitions>

<RowDefinition Height="40\*"/>

<RowDefinition Height="216\*"/>

<RowDefinition Height="23\*"/>

<RowDefinition Height="23\*"/>

<RowDefinition Height="23\*"/>

<RowDefinition Height="22\*"/>

<RowDefinition Height="22\*"/>

<RowDefinition Height="24\*"/>

<RowDefinition Height="33\*"/>

<RowDefinition Height="22\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="101\*"/>

<ColumnDefinition Width="62\*"/>

<ColumnDefinition Width="106\*"/>

<ColumnDefinition Width="70\*"/>

<ColumnDefinition Width="56\*"/>

<ColumnDefinition Width="50\*"/>

<ColumnDefinition Width="69\*"/>

<ColumnDefinition Width="47\*"/>

<ColumnDefinition Width="61\*"/>

<ColumnDefinition Width="108\*"/>

<ColumnDefinition Width="66\*"/>

</Grid.ColumnDefinitions>

<DataGrid Grid.Row="1" Name="dgCars" CanUserAddRows="False" CanUserDeleteRows="False" IsReadOnly="True" Grid.ColumnSpan="11" Margin="0,2,0,1" SelectionChanged="Data\_SelectionChanged" ItemsSource="{Binding car}" AutoGenerateColumns="False">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер" MinWidth="25" Width="\*" Binding="{Binding carnumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Марка" MinWidth="25" Width="\*" Binding="{Binding carbrand, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Модель" MinWidth="25" Width="\*" Binding="{Binding carmodel, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="СТС" MinWidth="25" Width="\*" Binding="{Binding sts, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="VIN" MinWidth="25" Width="\*" Binding="{Binding vin, UpdateSourceTrigger=PropertyChanged}"/>

</DataGrid.Columns>

</DataGrid>

<Label Content="Номер машины" Grid.Row="2" HorizontalAlignment="Center" Margin="2,23,1.6,0.4" Grid.RowSpan="2" Width="98"/>

<Label Content="Марка машины:" Grid.Row="5" HorizontalAlignment="Center" Margin="1,0.4,1.6,0" Width="99" />

<Label Content="Модель:" Grid.Row="6" HorizontalAlignment="Center" Margin="23,22,22.6,0.4" Grid.RowSpan="2" Width="56" />

<TextBox Grid.Row="2" Grid.Column="1" Name="tbNumber" Grid.ColumnSpan="2" MaxLength="6" Margin="0.4,23,0.4,0.4" Grid.RowSpan="2"/>

<TextBox Grid.Row="5" Grid.Column="1" Name="tbBrand" Grid.ColumnSpan="2" MaxLength="30" Margin="0.4,0.4,0.4,0"/>

<TextBox Grid.Row="6" Grid.Column="1" Name="tbModel" Grid.ColumnSpan="2" MaxLength="50" Margin="0.4,22,0.4,0.4" Grid.RowSpan="2"/>

<Label Content="VIN:" Grid.Row="2" Grid.Column="5" HorizontalAlignment="Left" Margin="0.2,23,0,0.4" Grid.RowSpan="2" Width="32" />

<Label Content="STS:" Grid.Row="5" Grid.Column="5" HorizontalAlignment="Left" Margin="0.2,0.4,0,0" Width="32"/>

<TextBox Grid.Row="2" Grid.Column="5" Grid.ColumnSpan="5" Name="tbVIN" Margin="50.2,23,108.6,0.4" Grid.RowSpan="2" MaxLength="17"/>

<TextBox Grid.Row="5" Grid.Column="5" Grid.ColumnSpan="5" Name="tbSTS" Margin="50.2,0.4,108.6,0" MaxLength="10"/>

<Button Grid.Row="9" Grid.Column="2" Content="Добавить" Name="btnAdd" Click="btnAdd\_Click" Margin="2,2.8,2.4,2.6"/>

<Button Grid.Row="9" Grid.Column="4" Content="Изменить" Name="btnUpd" Grid.ColumnSpan="2" Click="btnUpd\_Click" Margin="2.2,2.8,2.2,2.6"/>

<Button Grid.Row="9" Grid.Column="7" Content="Удалить" Name="btnDel" Grid.ColumnSpan="2" Click="btnDel\_Click" Margin="2.2,2.8,2.8,2.6"/>

<Button Name="btnClear" Grid.Row="7" Grid.Column="6" Grid.ColumnSpan="2" Content="Очистить" Click="btnClear\_Click"/>

<WrapPanel Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="11" HorizontalAlignment="Center">

<Button Content="Мои заказы" Name="btnGoOrder" Click="btnGoOrder\_Click" Width="150" />

<Button Content="Оформить заказ" Name="btnGetOrder" Click="btnGetOrder\_Click" Width="150"/>

</WrapPanel>

</Grid>

</Page>

# ClientPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для ClientPage.xaml

/// </summary>

public partial class ClientPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

//Уникалтьный индетификатор клиента

string Client = "";

/// <summary>

/// Обработчик выбора значения в DataGrid автомобилей

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgCars.SelectedItem == null)

{

return;

}

DataRowView row = (DataRowView)dgCars.SelectedItem;

tbNumber.Text = row[0].ToString();

tbBrand.Text = row[1].ToString();

tbModel.Text = row[2].ToString();

tbSTS.Text = row[3].ToString();

tbVIN.Text = row[4].ToString();

}

/// <summary>

/// Инциализация копонентов страницы клиента

/// </summary>

/// <param name="clientNumber">уникальный индетификатор клиента</param>

public ClientPage(string clientNumber)

{

InitializeComponent();

dbhelper.Refresh(dgCars, $@"select \* from Car where clientnumber = '{clientNumber}'", grdClientCar);

Client = clientNumber;

}

/// <summary>

/// Обработчик кнопки очищения текстовых полей

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.Refresh(dgCars, $@"select \* from Car where clientnumber = '{Client}'", grdClientCar);

}

/// <summary>

/// Обработчик кнопки добавления машины

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (tbNumber.Text.Length == 6 && tbSTS.Text.Length == 10 && tbVIN.Text.Length == 17 && tbBrand.Text != "" && tbModel.Text != "")

{

string[] atributes = { tbNumber.Text, tbBrand.Text, tbModel.Text, tbSTS.Text, tbVIN.Text, Client};

dbhelper.InsertInto("Car", atributes);

dbhelper.Refresh(dgCars, $@"select \* from Car where clientnumber = '{Client}'", grdClientCar);

}

else

{

valid.Show("Все поля должны быть заполнены: длина номера - 6, длина стс - 10, длина VIN - 17").GetAwaiter();

}

}

/// <summary>

/// Обработчик обновления данных по машине

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgCars.SelectedIndex == -1)

return;

if (tbNumber.Text.Length == 6 && tbSTS.Text.Length == 10 && tbVIN.Text.Length == 17 && tbBrand.Text != "" && tbModel.Text != "")

{

string[] atributes = { dbhelper.GetValueData(dgCars, "carnumber"), tbBrand.Text, tbModel.Text, tbSTS.Text, tbVIN.Text, Client, tbNumber.Text };

dbhelper.UpdateInto("Car", atributes);

dbhelper.Refresh(dgCars, $@"select \* from Car where clientnumber = '{Client}'", grdClientCar);

}

else

{

valid.Show("Все поля должны быть заполнены: длина номера - 6, длина стс - 10, длина VIN - 17").GetAwaiter();

}

}

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

{

if (dgCars.SelectedIndex == -1)

return;

if (MessageBox.Show("Вы точно хотите удалить данную машину?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Car", $@"{dbhelper.GetValueData(dgCars, "carnumber")}");

dbhelper.Refresh(dgCars, $@"select \* from Car where clientnumber = '{Client}'", grdClientCar);

}

}

/// <summary>

/// Обработчик перехода на страницу оформления заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

FrameManager.MainFrame.Navigate(new ClientOrderPage(Client));

}

/// <summary>

/// Обработчик кнопки перехода на страницу истории заказов

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

FrameManager.MainFrame.Navigate(new ClientOrderCreationPage(Client));

}

}

}

# DiagramAdmin.xaml

<Page x:Class="AutoService.DiagramAdmin"

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:AutoService"

xmlns:lvc="clr-namespace:LiveCharts.Wpf;assembly=LiveCharts.Wpf"

xmlns:charts="clr-namespace:System.Windows.Forms.DataVisualization.Charting;assembly=System.Windows.Forms.DataVisualization"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="DiagramAdmin">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="35\*"/>

<RowDefinition Height="427\*"/>

</Grid.RowDefinitions>

<WindowsFormsHost Grid.ColumnSpan="3" Grid.Row="1">

<charts:Chart x:Name="chart" >

<charts:Chart.Legends>

<charts:Legend>

</charts:Legend>

</charts:Chart.Legends>

</charts:Chart>

</WindowsFormsHost>

<ComboBox SelectionChanged="ComboBox\_SelectionChanged" Name="typeDiagram" VerticalAlignment="Center" Grid.Column="1" >

<ComboBoxItem>График</ComboBoxItem>

<ComboBoxItem>Круговая диаграмма</ComboBoxItem>

<ComboBoxItem>Cтолбчатая диаграмма</ComboBoxItem>

</ComboBox>

</Grid>

</Page>

<!--<UserControl x:Class="AutoService.DiagramAdmin"

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:AutoService"

xmlns:lvc="clr-namespace:LiveCharts.Wpf;assembly=LiveCharts.Wpf"

xmlns:charts="clr-namespace:System.Windows.Forms.DataVisualization.Charting;assembly=System.Windows.Forms.DataVisualization"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

d:DataContext="{d:DesignInstance local:PieChartExample}">

<Grid>

<lvc:PieChart LegendLocation="Bottom" DataClick="Chart\_OnDataClick" Hoverable="False" DataTooltip="{x:Null}">

<lvc:PieChart.Series>

<lvc:PieSeries Title="Maria" Values="3" DataLabels="True"

LabelPoint="{Binding PointLabel}"/>

<lvc:PieSeries Title="Charles" Values="4" DataLabels="True"

LabelPoint="{Binding PointLabel}"/>

<lvc:PieSeries Title="Frida" Values="6" DataLabels="True"

LabelPoint="{Binding PointLabel}"/>

<lvc:PieSeries Title="Frederic" Values="2" DataLabels="True"

LabelPoint="{Binding PointLabel}"/>

</lvc:PieChart.Series>

</lvc:PieChart>

</Grid>

</UserControl>-->

# DiagramAdmin.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Forms.DataVisualization.Charting;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using LiveCharts;

//using LiveCharts.Wpf;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для DiagramAdmin.xaml

/// </summary>

public partial class DiagramAdmin : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//Коллекция значений по оси X

List<string> axisXData = new List<string>();

//Коллекция значений по оси Y

List<string> axisYData = new List<string>();

/// <summary>

/// Инициализация компонентов страницы графиков

/// </summary>

public DiagramAdmin()

{

InitializeComponent();

// добавим данные линии

axisXData = dbhelper.EecuteQueryReader("select distinct extract(year from registrationdate) from Users;", axisXData, "date\_part");

foreach (string year in axisXData)

{

axisYData.Add(dbhelper.EecuteQueryReaderOne($@"select count(email) from users where extract(year from registrationdate) = {year};", "count"));

}

typeDiagram.SelectedIndex = 0;

}

/// <summary>

/// Обработчик выбра значения в ComboBox вида графика

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

chart.ChartAreas.Clear();

chart.Series.Clear();

if(typeDiagram.SelectedIndex == 0)

{

// Все графики находятся в пределах области построения ChartArea, создадим ее

chart.ChartAreas.Add(new ChartArea("Default"));

// Добавим линию, и назначим ее в ранее созданную область "Default"

chart.Series.Add(new Series("Количество пользователей"));

chart.Series["Количество пользователей"].ChartArea = "Default";

chart.Series["Количество пользователей"].ChartType = SeriesChartType.Line;

chart.Series["Количество пользователей"].Points.DataBindXY(axisXData, axisYData);

}

if (typeDiagram.SelectedIndex == 1)

{

// Все графики находятся в пределах области построения ChartArea, создадим ее

chart.ChartAreas.Add(new ChartArea("Default"));

// Добавим линию, и назначим ее в ранее созданную область "Default"

chart.Series.Add(new Series("Количество пользователей"));

chart.Series["Количество пользователей"].ChartArea = "Default";

chart.Series["Количество пользователей"].ChartType = SeriesChartType.Pie;

chart.Series["Количество пользователей"].IsValueShownAsLabel = true;

chart.Series["Количество пользователей"].Points.DataBindXY(axisXData, axisYData);

}

if (typeDiagram.SelectedIndex == 2)

{

// Все графики находятся в пределах области построения ChartArea, создадим ее

chart.ChartAreas.Add(new ChartArea("Default"));

// Добавим линию, и назначим ее в ранее созданную область "Default"

chart.Series.Add(new Series("Количество пользователей"));

chart.Series["Количество пользователей"].ChartArea = "Default";

chart.Series["Количество пользователей"].ChartType = SeriesChartType.Column;

chart.Series["Количество пользователей"].IsValueShownAsLabel = true;

chart.Series["Количество пользователей"].Points.DataBindXY(axisXData, axisYData);

}

}

}

}

# MainWindow.xaml

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

mc:Ignorable="d"

Title="Сервис-ПРО" WindowState="Maximized" Height="1080" Width="1920">

<Grid x:Name="MainGrid">

<Grid.RowDefinitions>

<RowDefinition Height="75"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="50"/>

<RowDefinition Height="30" x:Name="ErrorRow"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Image Grid.Row="0" Name="MPTImg" Source="Resources/AutoServiceLogo.png" HorizontalAlignment="Left" Margin="0,0,0,0.2" Width="75"/>

<Button Grid.Row="0" HorizontalAlignment="Center" Width="175" Content="Диаграмма" Visibility="Hidden" Name="btnDiagram" Click="btnDiagram\_Click" Grid.Column="3" Margin="134,23,60.6,22.2"/>

<Button Grid.Row="0" HorizontalAlignment="Center" Width="175" Content="БД" Visibility="Hidden" Name="btnDControl" Click="btnDControl\_Click" Grid.Column="1" Margin="135.2,23,60.2,22.2"/>

<!--<Button Grid.Row="0" Name="btnProfile" HorizontalAlignment="Left" Width="175" Height="30" Content="Профиль" Visibility="Hidden" Click="btnProfile\_Click"/>-->

<TextBlock Name="PageName" Text="Сервис-ПРО" FontSize="20" Grid.Row="0" Grid.Column="2" VerticalAlignment="Center" HorizontalAlignment="Center"/>

<!--<Button Name="btnAditionally" Content="История" Width="175" Height="30" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="5" Click="Additional\_Click" Visibility="Hidden"/>

<Button Name="btnTaskBook" Content="Заметки" Width="175" Height="30" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="5" Click="btnTaskBook\_Click" Visibility="Hidden"/>-->

<Grid Background="#bae3e8" Panel.ZIndex="-2" Grid.ColumnSpan="5"/>

<Grid Background="#445c93" Grid.Row="2" Grid.RowSpan="2" Grid.ColumnSpan="5" Margin="0,0.2,-0.4,-0.2" x:Name="ErrorGrid" VerticalAlignment="Bottom">

<Grid.Style>

<Style TargetType="{x:Type Grid}">

<Style.Triggers>

<EventTrigger RoutedEvent="SizeChanged">

<EventTrigger.Actions>

<BeginStoryboard>

<Storyboard>

<DoubleAnimation

Storyboard.TargetProperty="Opacity"

From="0.4"

To="1"

Duration="0:0:1.5" />

<!--Duration менять время анимации, From (от значения) To (до значения)-->

</Storyboard>

</BeginStoryboard>

</EventTrigger.Actions>

</EventTrigger>

</Style.Triggers>

</Style>

</Grid.Style>

<Label Content="Лол" HorizontalAlignment="Center" FontSize="20" Visibility="Hidden" Foreground="White" VerticalAlignment="Center"/>

</Grid>

<Frame Name="MainFrame" ContentRendered="MainFrame\_ContentRendered" NavigationUIVisibility="Hidden" Grid.Row="1" Grid.ColumnSpan="5"/>

<Button Name="btnGoBack" Content="Назад" Width="175" Height="30" HorizontalAlignment="Right" VerticalAlignment="Center" Margin="0,23,4.6,22.2" Click="GoBack\_Click" Grid.Column="4"/>

</Grid>

</Window>

# MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

/// <summary>

/// Инициализация компонентов главного окна

/// </summary>

public MainWindow()

{

InitializeComponent();

FrameManager.MainFrame = MainFrame;

MainFrame.Navigate(new SignInPage());//ClientPage("4567865467"));//SalesDepartmentPage("diagram14@mail.ru"));//ClientPage("4567865467"));//new PurchasePage());//SignInPage());//new SignInPage());new DiagramAdmin() SalesDepartmentPage("diagram14@mail.ru"));

}

/// <summary>

/// Обработчик кнопки перехода на прошлую страницу

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

FrameManager.MainFrame.GoBack();

}

/// <summary>

/// Отрисовка кнопок перехода в зависимости от страницы

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void MainFrame\_ContentRendered(object sender, EventArgs e)

{

if (((Page)MainFrame.Content).Title.ToString() == "DiagramAdmin")

{

PageName.Text = "Статистика пользователей";

btnDiagram.Visibility = Visibility.Hidden;

PageName.Visibility = Visibility.Visible;

btnGoBack.Content = "Назад";

}

else if(((Page)MainFrame.Content).Title.ToString() == "AdminPage")

{

btnDiagram.Visibility = Visibility.Visible;

btnDControl.Visibility = Visibility.Visible;

PageName.Text = "Сервис-ПРО";

btnGoBack.Content = "Выйти";

}

else if (((Page)MainFrame.Content).Title.ToString() == "AdminDataBaseControlPage")

{

PageName.Text = "Управление базой данных";

btnDControl.Visibility = Visibility.Hidden;

PageName.Visibility = Visibility.Visible;

btnGoBack.Content = "Назад";

}

else if (((Page)MainFrame.Content).Title.ToString() == "ClientPage")

{

PageName.Text = "Мои машины";

btnDControl.Visibility = Visibility.Hidden;

PageName.Visibility = Visibility.Visible;

btnGoBack.Content = "выйти";

}

else

{

PageName.Text = "Сервис-ПРО";

btnDiagram.Visibility = Visibility.Hidden;

btnDControl.Visibility = Visibility.Hidden;

PageName.Visibility = Visibility.Visible;

btnGoBack.Content = "Выйти";

}

if (MainFrame.CanGoBack)

{

btnGoBack.Visibility = Visibility.Visible;

}

else

{

btnGoBack.Visibility = Visibility.Hidden;

}

}

/// <summary>

/// Переход на страницу диаграмм

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

MainFrame.Navigate(new DiagramAdmin());

}

/// <summary>

/// Переход на страницу управления БД

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

MainFrame.Navigate(new AdminDataBaseControlPage());

}

}

}

# ModalWindowAddClient.xaml

<Window x:Class="AutoService.ModalWindowAddClient"

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:xctk="http://schemas.xceed.com/wpf/xaml/toolkit"

xmlns:local="clr-namespace:AutoService"

mc:Ignorable="d"

Title="Добавление клиента" SizeToContent="WidthAndHeight" WindowStartupLocation="CenterScreen" Closing="Window\_Closing">

<Grid Margin="5">

<Grid.RowDefinitions>

<RowDefinition Height="30" />

<RowDefinition Height="30" />

<RowDefinition Height="30" />

<RowDefinition Height="30" />

<RowDefinition Height="Auto" />

</Grid.RowDefinitions>

<TextBlock Grid.Row="0" Name="tbTitle">Введите данные клиента:</TextBlock>

<WrapPanel Grid.Row="1">

<Label Content="Серия паспорта:"/>

<xctk:MaskedTextBox Mask="0000" Name="passportSeries" MinWidth="250"/>

</WrapPanel>

<WrapPanel Grid.Row="2">

<Label Content="Номер паспорта:"/>

<xctk:MaskedTextBox Mask="000000" Name="passportNumber" MinWidth="250"/>

</WrapPanel>

<WrapPanel Grid.Row="3">

<Label Content="Номер телефона:"/>

<xctk:MaskedTextBox Mask="+70000000000" Name="ClientNumber" MinWidth="250"/>

</WrapPanel>

<WrapPanel Grid.Row="5" HorizontalAlignment="Right" Margin="0,15,0,0">

<Button Name="AddClient" IsDefault="True" MinWidth="60" Margin="0,0,10,0" Click="AddClient\_Click">OK</Button>

<Button IsCancel="True" MinWidth="60">Отмена</Button>

</WrapPanel>

<RichTextBox Grid.Row="1" Grid.RowSpan="3" Visibility="Hidden" Name="rtMessage"/>

</Grid>

</Window>

# ModalWindowAddClient.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для ModalWindowAddClient.xaml

/// </summary>

public partial class ModalWindowAddClient : Window

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса работы с электронной почтой

EmailHelper emailHelper = new EmailHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

static string email = "";

int Who = 0;

string id = "";

/// <summary>

/// Инициализация компонентов модального окна

/// </summary>

/// <param name="Email">электронная почта клиента</param>

/// <param name="who">разграничение на администратора и клиента</param>

/// <param name="ID">уникальный индетификатор пользователя</param>

public ModalWindowAddClient(string Email, int who, string ID)

{

InitializeComponent();

email = Email;

if (who == 1)

{

rtMessage.Visibility = Visibility.Visible;

tbTitle.Text = "Введите обоснование отмены";

}

Who = who;

id = ID;

}

/// <summary>

/// Обработчик сохранения данных

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (Who == 0)

{

if (!ClientNumber.Text.Contains("\_") && !passportSeries.Text.Contains("\_") && !passportNumber.Text.Contains("\_"))

{

string[] atributes = { ClientNumber.Text.Replace("+7", ""), passportSeries.Text, passportNumber.Text, email };

dbhelper.InsertInto("Client", atributes);

this.Close();

}

else

{

valid.Show("Заполните все поля").GetAwaiter();

}

}

else

{

if(new TextRange(rtMessage.Document.ContentStart, rtMessage.Document.ContentEnd).Text != "")

{

dbhelper.DeleteInto("orderclient", id);

emailHelper.SendMailOrder(email, $@"Заказ отменен", $@"<h1>Ваш заказ № {id} отменен.<h1><p>{new TextRange(rtMessage.Document.ContentStart, rtMessage.Document.ContentEnd).Text}</p>").GetAwaiter();

this.Close();

}

}

}

/// <summary>

/// Обработчик закрытия модального окна

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e)

{

if (Who == 0)

{

if (dbhelper.EecuteQueryReaderOne($@"select count(Email) FROM Client where Email = '{email}'", "count") == "0")

{

dbhelper.DeleteInto("Users", email);

}

}

}

}

}

# PurchasePage.xaml

<Page x:Class="AutoService.PurchasePage"

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:xctk="http://schemas.xceed.com/wpf/xaml/toolkit"

xmlns:System="clr-namespace:System;assembly=mscorlib"

xmlns:local="clr-namespace:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="PurchasePage">

<Grid Name="grdPurchase">

<Grid.RowDefinitions>

<RowDefinition Height="20\*"/>

<RowDefinition Height="111\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="111\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<DataGrid Grid.Column="0" Grid.Row="1" AutoGenerateColumns="False" ItemsSource="{Binding purchaseOrder}" x:Name="dgPurchaseOrder" RowEditEnding="dgPurchaseOrder\_RowEditEnding" BorderBrush="Black" Margin="0,0,1,0">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер заказа на поставку" MinWidth="25" Width="\*" Binding="{Binding purchaseordernumber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto" Header="Дата">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<DatePicker SelectedDate="{Binding purchaseorderdate, UpdateSourceTrigger=PropertyChanged}" DisplayDateStart="{Binding Mode=OneWay, Source={x:Static System:DateTime.Today}}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTextColumn Header="Количество" MinWidth="60" Width="\*" Binding="{Binding quantity, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="7"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Итоговая цена" MinWidth="100" Width="\*" Binding="{Binding rotalprice, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="9"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Артикул товара" MinWidth="140" Width="\*" Binding="{Binding article, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelPurchase\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid Grid.Column="1" Grid.Row="1" AutoGenerateColumns="False" ItemsSource="{Binding suplier}" x:Name="dgSuplier" RowEditEnding="dgSuplier\_RowEditEnding" BorderBrush="Black" Margin="1,0,0,0">

<DataGrid.Columns>

<DataGridTemplateColumn Header="ИНН поставщика">

<DataGridTemplateColumn.CellEditingTemplate>

<DataTemplate>

<TextBox Text="{Binding Path=inn, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}" PreviewTextInput= "tb1\_PreviewTextInputNum" DataObject.Pasting="tb1\_PastingNum" MaxLength="10"/>

</DataTemplate>

</DataGridTemplateColumn.CellEditingTemplate>

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=inn, Mode=TwoWay,UpdateSourceTrigger=PropertyChanged}" />

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<!--<DataGridTextColumn Header="ИНН поставщика" MinWidth="25" Width="\*" Binding="{Binding inn, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="10"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>-->

<DataGridTextColumn Header="Наименование" MinWidth="60" Binding="{Binding supliername, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<!--<DataGridTextColumn Header="Расчетный счет" MinWidth="100" Width="\*" Binding="{Binding paymentaccount, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="20"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>-->

<DataGridTemplateColumn Header="Расчетный счет">

<DataGridTemplateColumn.CellEditingTemplate>

<DataTemplate>

<TextBox Text="{Binding Path=paymentaccount, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}" PreviewTextInput= "tb1\_PreviewTextInputNum" DataObject.Pasting="tb1\_PastingNum" MaxLength="20"/>

</DataTemplate>

</DataGridTemplateColumn.CellEditingTemplate>

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=paymentaccount, Mode=TwoWay,UpdateSourceTrigger=PropertyChanged}" />

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelSuplier\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid Grid.Column="0" Grid.Row="3" AutoGenerateColumns="False" ItemsSource="{Binding purchasecontract}" x:Name="dgPurchaseContract" RowEditEnding="dgPurchaseContract\_RowEditEnding" BorderBrush="Black" Margin="0,0,1,0">

<DataGrid.Columns>

<DataGridTemplateColumn Width="auto" Header="Дата">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<DatePicker SelectedDate="{Binding purchasecontractdate, UpdateSourceTrigger=PropertyChanged}" DisplayDateStart="{Binding Mode=OneWay, Source={x:Static System:DateTime.Today}}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridComboBoxColumn x:Name="cbPurchaseOrder" Header="Номер заказа на поставку" MinWidth="60" />

<DataGridComboBoxColumn x:Name="cbSuplier" Header="Поставщик" MinWidth="60" />

<DataGridComboBoxColumn x:Name="cbEmployee" Header="Сотрудник" MinWidth="60"/>

<DataGridTextColumn Header="Номер контракта на поставку" MinWidth="25" Binding="{Binding purchasecontractnumber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelPurchaseContract\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid Grid.Column="1" Grid.Row="3" AutoGenerateColumns="False" ItemsSource="{Binding disposable}" x:Name="dgDisposable" RowEditEnding="dgDisposable\_RowEditEnding" BorderBrush="Black" Margin="1,0,0,0" >

<DataGrid.Columns>

<DataGridTextColumn Header="Серийный номер расходника" MinWidth="25" Width="\*" Binding="{Binding seriesnumber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridComboBoxColumn x:Name="cbPurchaseContractNumber" Header="Номер контракта на поставку" MinWidth="60" />

<DataGridTextColumn Header="Название расходника" MinWidth="60" Width="\*" Binding="{Binding disposablename, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="7"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Вес" MinWidth="100" Width="\*" Binding="{Binding disposableweight, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="9"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Цвет" MinWidth="140" Width="\*" Binding="{Binding disposablecolor, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="20"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Метериал" MinWidth="140" Width="\*" Binding="{Binding disposablematerial, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Номер сертификата" MinWidth="140" Width="\*" Binding="{Binding disposablesertificate, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelDisposable\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Grid.Row="0" Grid.Column="0" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Заказы на поставку" />

<Label Grid.Row="0" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Поставщики" />

<Label Grid.Row="2" Grid.Column="0" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Контракты на поставку" />

<Label Grid.Row="2" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Расходники" />

</Grid>

</Page>

# PurchasePage.xaml.cs

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для PurchasePage.xaml

/// </summary>

public partial class PurchasePage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

/// <summary>

/// Инициализация компонентов страницы Закупок

/// </summary>

public PurchasePage()

{

InitializeComponent();

DataGridsRefresh();

}

/// <summary>

/// Обработчик кнопки удаления закупки

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данную Поставку?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("PurchaseOrder", $@"{dbhelper.GetValueData(dgPurchaseOrder, "purchaseordernumber")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик изменения значений в DataGrid закупок

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgPurchaseOrder\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

if(e.EditAction == DataGridEditAction.Commit)

{

string[] atributes = { dbhelper.GetValueData(dgPurchaseOrder, "purchaseordernumber"), dbhelper.GetValueData(dgPurchaseOrder, "purchaseorderdate"), dbhelper.GetValueData(dgPurchaseOrder, "quantity"), dbhelper.GetValueData(dgPurchaseOrder, "rotalprice").Replace(",", "."), dbhelper.GetValueData(dgPurchaseOrder, "article"), dbhelper.GetValueData(dgPurchaseOrder, "purchaseordernumber") };

if (dbhelper.EecuteQueryReaderOne($@"select count(purchaseordernumber) from purchaseorder where purchaseordernumber = '{dbhelper.GetValueData(dgPurchaseOrder, "purchaseordernumber")}'","count")=="1")

{

dbhelper.UpdateInto("PurchaseOrder", atributes);

}

else

{

dbhelper.InsertInto("PurchaseOrder", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик изменения значений в DataGrid поставщиков

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgSuplier\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

if (e.EditAction == DataGridEditAction.Commit)

{

string[] atributes = { dbhelper.GetValueData(dgSuplier, "inn"), dbhelper.GetValueData(dgSuplier, "supliername"), dbhelper.GetValueData(dgSuplier, "paymentaccount"), dbhelper.GetValueData(dgSuplier, "inn") };

if (dbhelper.EecuteQueryReaderOne($@"select count(inn) from suplier where inn = '{dbhelper.GetValueData(dgSuplier, "inn")}'", "count") == "1")

{

dbhelper.UpdateInto("Suplier", atributes);

}

else

{

dbhelper.InsertInto("Suplier", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик удаления поставщика

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данную Поставку?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Suplier", $@"{dbhelper.GetValueData(dgSuplier, "inn")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обработчики добавления и вставления значений в тестовые поля

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void tb1\_PreviewTextInputNum(object sender, TextCompositionEventArgs e) => new ValidationData().tb1\_PreviewTextInputNum(sender, e);

private void tb1\_PastingNum(object sender, DataObjectPastingEventArgs e) => new ValidationData().tb1\_PastingNum(sender, e);

/// <summary>

/// Обработчик изменения значений в DataGrid контракта

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgPurchaseContract\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

if (e.EditAction == DataGridEditAction.Commit)

{

string[] atributes = { dbhelper.GetValueData(dgPurchaseContract, "purchasecontractnumber"), dbhelper.GetValueData(dgPurchaseContract, "purchasecontractdate"),

dbhelper.GetTextComboBox(dgPurchaseContract,1),

dbhelper.EecuteQueryReaderOne($@"select \* from Suplier where supliername = '{dbhelper.GetTextComboBox(dgPurchaseContract, 2)}'", "inn"),

dbhelper.EecuteQueryReaderOne($@"select \* from Users where (lastname ||' '|| firstname||' '|| middlename) = '{dbhelper.GetTextComboBox(dgPurchaseContract, 3)}'", "email"),

dbhelper.GetValueData(dgPurchaseContract, "purchasecontractnumber") };

if (dbhelper.EecuteQueryReaderOne($@"select count(purchasecontractnumber) from purchasecontract where purchasecontractnumber = '{dbhelper.GetValueData(dgPurchaseOrder, "purchasecontractnumber")}'", "count") == "1")

{

dbhelper.UpdateInto("purchasecontract", atributes);

}

else

{

dbhelper.InsertInto("purchasecontract", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик удаления контракта

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данный контракт на Поставку?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("purchasecontract", $@"{dbhelper.GetValueData(dgPurchaseContract, "purchasecontractnumber")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик изменения значений в DataGrid расходников

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgDisposable\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

if (e.EditAction == DataGridEditAction.Commit)

{

string[] atributes = {

dbhelper.GetValueData(dgDisposable, "seriesnumber"),

dbhelper.GetValueData(dgDisposable, "disposablename"),

dbhelper.GetValueData(dgDisposable, "disposableweight"),

dbhelper.GetValueData(dgDisposable, "disposablecolor"),

dbhelper.GetValueData(dgDisposable, "disposablematerial"),

dbhelper.GetValueData(dgDisposable, "disposablesertificate"),

dbhelper.GetTextComboBox(dgDisposable,1),

dbhelper.GetValueData(dgDisposable, "seriesnumber"),

};

if (dbhelper.EecuteQueryReaderOne($@"select count(seriesnumber) from Disposable where seriesnumber = '{dbhelper.GetValueData(dgDisposable, "seriesnumber")}'", "count") == "1")

{

dbhelper.UpdateInto("Disposable", atributes);

}

else

{

dbhelper.InsertInto("Disposable", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик удаления расходников

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данный Расходник?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Disposable", $@"{dbhelper.GetValueData(dgDisposable, "seriesnumber")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обновление DataGrid-ов всей страницы

/// </summary>

private void DataGridsRefresh()

{

dbhelper.Refresh(dgDisposable, "select \* from Disposable", grdPurchase);

dbhelper.Refresh(dgPurchaseContract, "select \* from PurchaseContract", grdPurchase);

dbhelper.Refresh(dgSuplier, "select \* from suplier; ", grdPurchase);

dbhelper.Refresh(dgPurchaseOrder, "select \* from purchaseorder; ", grdPurchase);

dbhelper.BindComboBox(cbPurchaseOrder, "select \* from purchaseorder", "purchaseordernumber", "purchaseordernumber");

dbhelper.BindComboBox(cbSuplier, "select \* from Suplier", "supliername", "inn");

dbhelper.BindComboBox(cbEmployee, "select lastname ||' '|| firstname||' '|| middlename as colum, email from Users where RoleName = 'Сотрудник отдела закупок'", "colum", "email");

dbhelper.BindComboBox(cbPurchaseContractNumber, "select \* from PurchaseContract", "purchasecontractnumber", "purchasecontractnumber");

}

}

}

# RecoveryPage.xaml

<Page x:Class="AutoService.RecoveryPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="RecoveryPage">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Row="0" Grid.ColumnSpan="2" FontFamily="Ink Free" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="30" Text="Восстановление пароля" Grid.Column="1"/>

<TextBox Grid.Row="1" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="tbEmail"/>

<TextBlock Grid.Row="0" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Почта " FontSize="20"/>

<TextBlock Grid.Row="4" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Код восстановления " FontSize="20"/>

<TextBox Grid.Row="5" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="tbCode"/>

<TextBlock Grid.Row="8" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Новый пароль" FontSize="20"/>

<PasswordBox Grid.Row="9" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="pbPassword"/>

<TextBlock Grid.Row="11" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Повторите пароль" FontSize="20"/>

<PasswordBox Grid.Row="12" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="pbPassword2"/>

<Button Name="btnSendMessage" FontSize="20" Background="#445c93" Grid.Row="2" Grid.Column="1" Grid.ColumnSpan="2" Content="Отправить письмо" FontFamily="Ink Free" Click="btnSendMessage\_Click"/>

<Button Name="btnCheckCode" Click="btnCheckCode\_Click" Margin="70 0 70 0" FontSize="20" Background="#445c93" Grid.Row="6" Grid.ColumnSpan="2" Grid.Column="1" Content="Проверить" FontFamily="Ink Free"/>

<Button Name="btnRecovery" Click="btnRecovery\_Click" Margin="70 10 70 60" FontSize="20" Background="#445c93" Grid.Row="14" Grid.ColumnSpan="2" Grid.Column="1" Content="Восстановить пароль" FontFamily="Ink Free"/>

</Grid>

</Page>

# RecoveryPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для RecoveryPage.xaml

/// </summary>

public partial class RecoveryPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса работы с электронной почтой

EmailHelper emailHelper = new EmailHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

//экземпляр класса генерации рандомных чисел

Random r = new Random();

//Сгенерированный код

int code;

/// <summary>

/// Инициализация страницы востанновления пароля

/// </summary>

public RecoveryPage()

{

InitializeComponent();

}

/// <summary>

/// Обработчик кнопки отправки электронного письма

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

code = r.Next(100000, 999999);

if (dbhelper.EecuteQueryReaderOne($@"select count(Email) FROM Users where Email = '{tbEmail.Text}'", "count") == "1")

{

emailHelper.SendMail(tbEmail.Text, code).GetAwaiter();

tbEmail.IsEnabled = false;

btnSendMessage.IsEnabled = false;

}

else

valid.Show("Такой почты нет в системе").GetAwaiter();

}

/// <summary>

/// Обработчик кнопки проверки кода

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (tbCode.Text == code.ToString())

{

tbCode.IsEnabled = false;

btnCheckCode.IsEnabled = false;

}

else

valid.Show("Невверный код для восстанвления").GetAwaiter();

}

/// <summary>

/// Обработчик кнопки восстановления пароля

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (tbCode.IsEnabled == true)

{

valid.Show("Пройдите все этапы восстановления пароля").GetAwaiter();

return;

}

if(pbPassword.Password == pbPassword2.Password)

{

dbhelper.EecuteQuery($@"UPDATE Users SET Password = '{valid.Encrypt(pbPassword.Password)}' WHERE Email = '{tbEmail.Text}';");

valid.Show("Успешное восстановление пароля").GetAwaiter();

FrameManager.MainFrame.GoBack();

}

else

{

valid.Show("Пароли не совпадают").GetAwaiter();

}

}

}

}

# RepairPage.xaml

<Page x:Class="AutoService.RepairPage"

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:System="clr-namespace:System;assembly=mscorlib"

xmlns:local="clr-namespace:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="RepairPage">

<Grid Name="grdRepair">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<DataGrid Grid.ColumnSpan="2" Name="dgContracts" AutoGenerateColumns="False" ItemsSource="{Binding contract}" SelectionChanged="dgContracts\_SelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер контракта" Width="\*" Binding="{Binding contractnumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Дата заклюения" Width="\*" Binding="{Binding datecomclusion, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Клиент" Width="\*" Binding="{Binding clientnumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Номер заказа" Width="\*" Binding="{Binding ordernumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Номер машины" Width="\*" Binding="{Binding carnumber, UpdateSourceTrigger=PropertyChanged}"/>

</DataGrid.Columns>

</DataGrid>

<Label Grid.Row="1" Content="Диагностика" FontSize="25" HorizontalAlignment="Center" />

<DataGrid RowEditEnding="dgDiagnostics\_RowEditEnding" Grid.Row="2" AutoGenerateColumns="False" ItemsSource="{Binding diagnostics}" x:Name="dgDiagnostics" BorderBrush="Black" Margin="1,0,0,0" SelectionChanged="dgDiagnostics\_SelectionChanged">

<DataGrid.Columns>

<DataGridTextColumn Header="Уникальный номер диагностики" MinWidth="60" Width="\*" Binding="{Binding diagnosticsnumber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto" Header="Дата">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<DatePicker SelectedDate="{Binding diagnosticsdate, UpdateSourceTrigger=PropertyChanged}" DisplayDateStart="{Binding Mode=OneWay, Source={x:Static System:DateTime.Today}}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTextColumn Header="Результат диагностики" MinWidth="60" Width="\*" Binding="{Binding diagnosticsresults, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Описание" MinWidth="60" Width="\*" Binding="{Binding diagnosticsdescription, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="500"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Grid.Row="1" Grid.Column="1" Content="Статус машины" FontSize="25" HorizontalAlignment="Center" />

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

<Label FontSize="15" Content="Статус манины" HorizontalContentAlignment="Center"/>

<TextBox Name="tbStatusName" TextAlignment="Center"/>

<Label FontSize="15" Content="Время" HorizontalContentAlignment="Center"/>

<Label FontSize="20" Content="00:00" HorizontalContentAlignment="Center" Name="lbTimeStatus"/>

<Button Name="btnAddStatus" Content="Обозначить статус" Click="btnAddStatus\_Click"/>

<Button Name="btnDelOrder" Content="Закрыть ремонт" Click="btnDelOrder\_Click"/>

</StackPanel>

</Grid>

</Page>

# RepairPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для RepairPage.xaml

/// </summary>

public partial class RepairPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса валидацции

ValidationData validation = new ValidationData();

//уникальный индетификатор сотрудника

string email = "";

/// <summary>

/// Инициализаии компонентов страницы ремонтного отдела

/// </summary>

/// <param name="employeeEmail">уникальный индетификатор сотрудника</param>

public RepairPage(string employeeEmail)

{

InitializeComponent();

dbhelper.Refresh(dgContracts, "select \* from orderclient, contract where orderclient.ordernumber = contract.ordernumber", grdRepair);

email = employeeEmail;

}

/// <summary>

/// Обработчик выбора значения DataGrid контрактов

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

dbhelper.Refresh(dgDiagnostics, $@"select \* from diagnostics where contractnumber = '{dbhelper.GetValueData(dgContracts, "contractnumber")}'", grdRepair);

}

/// <summary>

/// Обработчик изменения значения DataGrid диагностики

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgDiagnostics\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

string[] atributes = { dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber"),

dbhelper.GetValueData(dgDiagnostics, "diagnosticsdate"),

dbhelper.GetValueData(dgDiagnostics, "diagnosticsresults"),

dbhelper.GetValueData(dgDiagnostics, "diagnosticsdescription"),

dbhelper.GetValueData(dgContracts, "carnumber"),

dbhelper.GetValueData(dgContracts, "contractnumber"),

email,

dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber"),

};

if (dbhelper.EecuteQueryReaderOne($@"select count(diagnosticsnumber) from diagnostics where diagnosticsnumber = '{dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber")}'", "count") == "1")

{

dbhelper.UpdateInto("diagnostics", atributes);

}

else

{

dbhelper.InsertInto("diagnostics", atributes.Take(atributes.Length - 1).ToArray());

}

dbhelper.Refresh(dgDiagnostics, $@"select \* from diagnostics where contractnumber = '{dbhelper.GetValueData(dgContracts, "contractnumber")}'", grdRepair);

}

/// <summary>

/// Обработчик выбора значения DataGrid диагностики

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

tbStatusName.Text = dbhelper.EecuteQueryReaderOne($@"select statuscar from statuscar where diagnosticsnumber = '{dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber")}'", "statuscar");

lbTimeStatus.Content = dbhelper.EecuteQueryReaderOne($@"select statustime from statuscar where diagnosticsnumber = '{dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber")}'", "statustime");

if (lbTimeStatus.Content.ToString() == "" || lbTimeStatus.Content.ToString() == "00:00")

lbTimeStatus.Content = DateTime.Now.ToString();

}

/// <summary>

/// Обработчик кнопки обновления статуса

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgDiagnostics.SelectedIndex != -1 && tbStatusName.Text.Replace(" ","")!="")

{

if (dbhelper.EecuteQueryReaderOne($@"SELECT count(diagnosticsnumber) FROM statuscar where diagnosticsnumber = '{dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber")}'", "count") == "1")

{

string[] atributes = { dbhelper.EecuteQueryReaderOne($@"select id\_statuscar from statuscar where diagnosticsnumber = '{dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber")}'", "id\_statuscar"), tbStatusName.Text, DateTime.Now.ToString(), dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber") };

dbhelper.UpdateInto("statuscar", atributes);

}

else

{

string[] atributes = { tbStatusName.Text, DateTime.Now.ToString(), dbhelper.GetValueData(dgDiagnostics, "diagnosticsnumber") };

dbhelper.InsertInto("statuscar", atributes);

}

lbTimeStatus.Content = DateTime.Now.ToString();

validation.Show("Успешное обновление статуса").GetAwaiter();

}

}

/// <summary>

/// Обработчик кнопки завершения заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (dgDiagnostics.SelectedIndex != -1)

{

if (MessageBox.Show("Вы точно хотите закрыть ремонт?!?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("orderclient", dbhelper.GetValueData(dgContracts, "ordernumber"));

}

}

dbhelper.Refresh(dgContracts, "select \* from orderclient, contract where orderclient.ordernumber = contract.ordernumber", grdRepair);

dbhelper.Refresh(dgDiagnostics, $@"select \* from diagnostics where contractnumber = ' '", grdRepair);

tbStatusName.Text = "";

lbTimeStatus.Content = DateTime.Now.ToString();

}

}

}

# SalesDepartmentPage.xaml

<Page x:Class="AutoService.SalesDepartmentPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="SalesDepartmentPage">

<Grid Name="grdSalesDepartament">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Label Content="Заказы ожидающие оформления" FontSize="25" HorizontalAlignment="Center"/>

<Label Content="Оформленные" FontSize="25" Grid.Column="1" HorizontalAlignment="Center"/>

<DataGrid Name="dgOrders" ItemsSource="{Binding orderclient}" AutoGenerateColumns="False" Grid.Row="1">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер заказа" Width="\*" Binding="{Binding ordernumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTemplateColumn Header="Дата" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=orderdate}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Header="Машина" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=carnumber}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Header="Стоимость" Width="\*">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=ordersum}" TextWrapping="Wrap" HorizontalAlignment="Center" VerticalAlignment="Center"></TextBlock>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Оформить" Click="btnAddOrder\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Отменить" Click="btnDelOrder\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid Grid.Row="1" Grid.Column="1" Name="dgContracts" AutoGenerateColumns="False" ItemsSource="{Binding contract}" >

<DataGrid.Columns>

<DataGridTextColumn Header="Номер контракта" Width="\*" Binding="{Binding contractnumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Дата заклюения" Width="\*" Binding="{Binding datecomclusion, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Клиент" Width="\*" Binding="{Binding clientnumber, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTextColumn Header="Оформил" Width="\*" Binding="{Binding email, UpdateSourceTrigger=PropertyChanged}"/>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Отменить" Click="btnDelContract\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Grid.Row="2" Grid.ColumnSpan="2" Content="Предлогаемые услуги" FontSize="25" HorizontalAlignment="Center" />

<DataGrid RowEditEnding="dgServices\_RowEditEnding" Grid.ColumnSpan="2" Grid.Row="3" AutoGenerateColumns="False" ItemsSource="{Binding warehouse}" x:Name="dgServices" BorderBrush="Black" Margin="1,0,0,0" >

<DataGrid.Columns>

<DataGridTextColumn Header="Название услуги" MinWidth="25" Width="\*" Binding="{Binding servicename, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Цена" MinWidth="60" Width="\*" Binding="{Binding price, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Описание" MinWidth="60" Width="\*" Binding="{Binding servicedescription, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="500"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelService\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Page>

# SalesDepartmentPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для SalesDepartmentPage.xaml

/// </summary>

public partial class SalesDepartmentPage : Page

{

//экземпляр класса кправления БД

DBHelper dbhelper = new DBHelper();

//Электронная почта пользователя

string employeeMail = "";

//Экземпляр класса управления почтовым клиентом

EmailHelper emailHelper = new EmailHelper();

//Коллекция заказов

List<string> orders = new List<string>();

//Запрос

string Query = "";

/// <summary>

/// Инициализация компонентов страницы Отдела продаж

/// </summary>

/// <param name="employeeEmail">Уникальный индетификатор сотрудника</param>

public SalesDepartmentPage(string employeeEmail)

{

InitializeComponent();

employeeMail = employeeEmail;

Refresh();

}

/// <summary>

/// Обновление DataGrid-ов страницы

/// </summary>

private void Refresh()

{

orders.Clear();

orders = dbhelper.EecuteQueryReader("select \* from contract", orders, "ordernumber");

Query = "select \* from orderclient where ";

foreach(string order in orders)

{

Query += $@" ordernumber != '{order}' and ";

}

Query += "ordernumber != '{order}'";

dbhelper.Refresh(dgOrders, Query, grdSalesDepartament);

dbhelper.Refresh(dgContracts, "select \* from Contract", grdSalesDepartament);

dbhelper.Refresh(dgServices, "select \* from service", grdSalesDepartament);

}

/// <summary>

/// Обработчик добавления заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

string contractNumber = "";

string clientEmail = dbhelper.EecuteQueryReaderOne($@"select \* from car where carnumber = '{dbhelper.GetValueData(dgOrders, "carnumber")}'", "clientnumber");

do

{

Random rnd = new Random();

contractNumber = rnd.Next(111111111, 999999999).ToString();

}

while (dbhelper.EecuteQueryReaderOne($@"SELECT count(ContractNumber) FROM Contract where ContractNumber = '{contractNumber}'", "count") == "1");

string[] atributes = {contractNumber, DateTime.Now.ToString("yyyy.MM.dd"), dbhelper.GetValueData(dgOrders, "ordernumber"),

clientEmail,

employeeMail};

dbhelper.InsertInto("Contract", atributes);

emailHelper.SendMailOrder(dbhelper.EecuteQueryReaderOne($@"select email from client where clientnumber = '{clientEmail}'", "email"), $@"Успешное оформление заказа № {dbhelper.GetValueData(dgOrders, "ordernumber")}", "<h1>Ваш заказ оформлен, ожидаем вас в нашем автосервисе<h1>").GetAwaiter();

Refresh();

}

/// <summary>

/// Обработчик кнопки удаления заказа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

string clientEmail = dbhelper.EecuteQueryReaderOne($@"select \* from car where carnumber = '{dbhelper.GetValueData(dgOrders, "carnumber")}'", "clientnumber");

ModalWindowAddClient appointment = new ModalWindowAddClient(dbhelper.EecuteQueryReaderOne($@"select email from client where clientnumber = '{clientEmail}'", "email"), 1, dbhelper.GetValueData(dgOrders, "ordernumber"));

appointment.ShowDialog();

Refresh();

//dbhelper.DeleteInto("orderclient", $@"{dbhelper.EecuteQueryReaderOne($@"select \* from contract where contractnumber = '{dbhelper.GetValueData(dgOrders, "contractnumber")}'", "ordernumber")}");

//dbhelper.EecuteQueryReaderOne($@"select \* from contract where contractnumber = '{dbhelper.GetValueData(dgOrders, "contractnumber")}'", "ordernumber")

}

/// <summary>

/// Обработчик кнопки удаления контракта

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

ModalWindowAddClient appointment = new ModalWindowAddClient(dbhelper.EecuteQueryReaderOne($@"select email from client where clientnumber = '{dbhelper.GetValueData(dgContracts, "clientnumber")}'", "email"), 1, dbhelper.EecuteQueryReaderOne($@"select \* from contract where contractnumber = '{dbhelper.GetValueData(dgContracts, "contractnumber")}'", "ordernumber"));

appointment.ShowDialog();

Refresh();

}

/// <summary>

/// Обработчик кнопки удаления услуги

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данную услугу?!?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Service", $@"{dbhelper.GetValueData(dgServices, "servicename")}");

Refresh();

}

}

/// <summary>

/// Обработчик изменения значений в DataGrid услуг

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgServices\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

string[] atributes = { dbhelper.GetValueData(dgServices, "servicename"), dbhelper.GetValueData(dgServices, "price"), dbhelper.GetValueData(dgServices, "servicedescription"), dbhelper.GetValueData(dgServices, "servicename") };

if (dbhelper.EecuteQueryReaderOne($@"select count(servicename) from service where servicename = '{dbhelper.GetValueData(dgServices, "servicename")}'", "count") == "1")

{

dbhelper.UpdateInto("Service", atributes);

}

else

{

dbhelper.InsertInto("Service", atributes.Take(atributes.Length - 1).ToArray());

}

Refresh();

}

}

}

# SignInPage.xaml

<Page x:Class="AutoService.SignInPage"

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:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="SignInPage" >

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.1\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.1\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Row="0" Grid.ColumnSpan="2" FontFamily="Ink Free" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="30" Text="Авторизация" Grid.Column="1"/>

<TextBox Grid.Row="1" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="tbLogin" PreviewTextInput= "tb1\_PreviewTextInput" DataObject.Pasting="tb1\_Pasting"/>

<PasswordBox Grid.Row="3" HorizontalContentAlignment="Center" Grid.ColumnSpan="2" FontSize="20" VerticalContentAlignment="Center" FontFamily="Ink Free" Grid.Column="1" Name="tbPassword" PreviewTextInput= "tb1\_PreviewTextInput" DataObject.Pasting="tb1\_Pasting"/>

<TextBlock Grid.Row="0" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Логин " FontSize="20"/>

<TextBlock Grid.Row="2" Grid.Column="1" Grid.ColumnSpan="2" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Пароль " FontSize="20"/>

<Button Name="btnAuth" FontSize="20" Background="#445c93" Grid.Row="5" Grid.Column="1" Content="Войти" Click="btnAuth\_Click" FontFamily="Ink Free"/>

<Button Name="btnReg" FontSize="20" Grid.Row="5" Grid.Column="2" Content="---Регистрация---" Click="btnReg\_Click" FontFamily="Ink Free" Background="Transparent" Foreground="Blue"/>

<Button Name="btnRecovery" Margin="70 0 70 0" FontSize="20" Background="#445c93" Grid.Row="7" Grid.ColumnSpan="2" Grid.Column="1" Content="Восстановить пароль" Click="btnRecovery\_Click" FontFamily="Ink Free"/>

</Grid>

</Page>

# SignInPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для SignInPage.xaml

/// </summary>

public partial class SignInPage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

//экземпляр класса валидации

ValidationData valid = new ValidationData();

/// <summary>

/// Инициализая компонентов страницы авторизации

/// </summary>

public SignInPage()

{

InitializeComponent();

}

/// <summary>

/// Обработчик вводимых значений в текстовое поле

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void tb1\_PreviewTextInput(object sender, TextCompositionEventArgs e) => new ValidationData().tb1\_PreviewTextInputAngNum(sender, e);

/// <summary>

/// Обработчик вставки значений в текстовое поле

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void tb1\_Pasting(object sender, DataObjectPastingEventArgs e) => new ValidationData().tb1\_PastingAngNum(sender, e);

/// <summary>

/// Обработчик кнопки входа

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

string role = dbhelper.Authorization(tbLogin.Text, valid.Encrypt(tbPassword.Password));

if (role !=null)

{

if(role == "Администратор")

{

FrameManager.MainFrame.Navigate(new AdminPage());

}

if (role == "Сотрудник отдела закупок")

{

FrameManager.MainFrame.Navigate(new PurchasePage());

}

if (role == "Сотрудник склада")

{

FrameManager.MainFrame.Navigate(new WarehousePage());

}

if (role == "Сотрудник отдела продаж")

{

FrameManager.MainFrame.Navigate(new SalesDepartmentPage(tbLogin.Text));

}

if (role == "Сотрудник ремонтного отдела")

{

FrameManager.MainFrame.Navigate(new RepairPage(tbLogin.Text));

}

if (role == "Клиент")

{

FrameManager.MainFrame.Navigate(new ClientPage(dbhelper.EecuteQueryReaderOne($@"select ClientNumber from client where email = '{tbLogin.Text}'", "ClientNumber")));

}

}

}

/// <summary>

/// Обработчик кнопки перехода на страницу регистрации

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

FrameManager.MainFrame.Navigate(new SignUpPage());

}

/// <summary>

/// Обработчик кнопки перехода на страницу восстановления пароля

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

FrameManager.MainFrame.Navigate(new RecoveryPage());

}

}

}

# SignUpPage.xaml

<Page x:Class="AutoService.SignUpPage"

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:xctk="http://schemas.xceed.com/wpf/xaml/toolkit"

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

xmlns:local="clr-namespace:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="SignUpPage">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="0.3\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

<RowDefinition Height="0.2\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<TextBlock Grid.Row="0" FontFamily="Ink Free" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="30" Text="Регистрация" Grid.Column="1"/>

<TextBox Grid.Row="15" HorizontalContentAlignment="Center" VerticalContentAlignment="Center" FontSize="20" Grid.Column="1" Name="tbLogin" />

<PasswordBox Grid.Row="17" HorizontalContentAlignment="Center" FontSize="20" VerticalContentAlignment="Center" FontFamily="Ink Free" Grid.Column="1" Name="tbPassword" />

<TextBlock Grid.Row="14" Grid.Column="1" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Email " FontSize="20"/>

<TextBlock Grid.Row="18" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Повторите пароль" FontSize="20"/>

<PasswordBox Grid.Row="19" Grid.Column="1" Name="pbPassword" FontSize="20" HorizontalContentAlignment="Center"/>

<TextBlock Grid.Row="16" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="пароль" FontSize="20"/>

<Button Name="btnReg" FontSize="20" Background="#445c93" Grid.Row="21" Grid.Column="1" Content="Регистрация" FontFamily="Ink Free" Click="btnReg\_Click"/>

<TextBlock Grid.Row="1" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Фамилия" FontSize="20"/>

<TextBox Grid.Row="2" Grid.Column="1" x:Name="tbLastName" FontSize="20" TextAlignment="Center"/>

<TextBlock Grid.Row="3" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Имя" FontSize="20"/>

<TextBox Grid.Row="4" Grid.Column="1" x:Name="tbFirstName" FontSize="20" TextAlignment="Center"/>

<TextBlock Grid.Row="5" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Отчество" FontSize="20"/>

<TextBox Grid.Row="6" Grid.Column="1" x:Name="tbMiddleName" FontSize="20" TextAlignment="Center"/>

<TextBlock Grid.Row="7" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Серия паспорта" FontSize="20"/>

<xctk:MaskedTextBox Grid.Row="8" Grid.Column="1" x:Name="passportSeries" Mask="0000" TextAlignment="Center" FontSize="20"/>

<TextBlock Grid.Row="9" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Номер паспорта" FontSize="20"/>

<xctk:MaskedTextBox Grid.Row="10" Grid.Column="1" x:Name="passportNumber" Mask="000000" FontSize="20" TextAlignment="Center" />

<TextBlock Grid.Row="11" Grid.Column="1" FontFamily="Ink Free" VerticalAlignment="Bottom" HorizontalAlignment="Center" Text="Номер телефона" FontSize="20"/>

<xctk:MaskedTextBox Grid.Row="12" Grid.Column="1" x:Name="ClientNumber" Mask="+70000000000" FontSize="20" TextAlignment="Center"/>

</Grid>

</Page>

# SignUpPage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для SignUpPage.xaml

/// </summary>

public partial class SignUpPage : Page

{

/// <summary>

/// Инициализация компонентов страницы регистрации

/// </summary>

public SignUpPage()

{

InitializeComponent();

}

/// <summary>

/// Обработчик кнопки регистрации

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

DBHelper dbhelper = new DBHelper();

ValidationData valid = new ValidationData();

if (pbPassword.Password == tbPassword.Password )

{

if (valid.IsValidEmail(tbLogin.Text) != false && valid.IsValidPassword(tbPassword.Password) != false && tbLastName.Text != "" && tbFirstName.Text != "" && tbMiddleName.Text != "" && !ClientNumber.Text.Contains("\_") && !passportSeries.Text.Contains("\_") && !passportNumber.Text.Contains("\_"))

{

string[] atributes = { tbLogin.Text, valid.Encrypt(tbPassword.Password), tbLastName.Text, tbFirstName.Text, tbMiddleName.Text, "Клиент" };

dbhelper.InsertInto("Users", atributes);

string[] atributes2 = { ClientNumber.Text.Replace("+7", ""), passportSeries.Text, passportNumber.Text, tbLogin.Text };

dbhelper.InsertInto("Client", atributes2);

FrameManager.MainFrame.GoBack();

}

else

{

valid.Show("Все поля должны быть правильно заполнены").GetAwaiter(); ;

}

}

else

{

valid.Show("Пароли не совпадают").GetAwaiter();

}

}

}

}

# WarehousePage.xaml

<Page x:Class="AutoService.WarehousePage"

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:System="clr-namespace:System;assembly=mscorlib"

xmlns:local="clr-namespace:AutoService"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="WarehousePage">

<Grid Name="grdWarehousePage">

<Grid.RowDefinitions>

<RowDefinition Height="20\*"/>

<RowDefinition Height="111\*"/>

<RowDefinition Height="20\*"/>

<RowDefinition Height="111\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Label Grid.Row="0" Grid.Column="0" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Склады" />

<Label Grid.Row="2" Grid.ColumnSpan="2" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Накладная" />

<DataGrid Grid.Column="0" Grid.Row="1" AutoGenerateColumns="False" ItemsSource="{Binding warehouse}" x:Name="dgWarehouse" RowEditEnding="dgWarehouse\_RowEditEnding" BorderBrush="Black" Margin="1,0,0,0" >

<DataGrid.Columns>

<DataGridTextColumn Header="Название склада" MinWidth="25" Width="\*" Binding="{Binding warehousename, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="30"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTextColumn Header="Адрес" MinWidth="60" Width="\*" Binding="{Binding warehouseaddress, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="60"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelDisposable\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Grid.Row="0" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="25" Content="Ячейки" />

<DataGrid Grid.Column="1" Grid.Row="1" AutoGenerateColumns="False" ItemsSource="{Binding cell}" x:Name="dgCells" RowEditEnding="dgCells\_RowEditEnding" BorderBrush="Black" Margin="1,0,0,0" >

<DataGrid.Columns>

<DataGridComboBoxColumn x:Name="cbWarehouseName" Header="Склад" MinWidth="60" />

<DataGridTextColumn Header="Номер ячейки" MinWidth="25" Width="\*" Binding="{Binding cellnumber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="5"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelCell\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid Grid.Row="3" Grid.ColumnSpan="2" AutoGenerateColumns="False" ItemsSource="{Binding invoice}" x:Name="dgInvoice" RowEditEnding="dgInvoice\_RowEditEnding" BorderBrush="Black" Margin="1,0,0,0" >

<DataGrid.Columns>

<DataGridComboBoxColumn x:Name="cbseriesnumber" Header="Серийный номер расходника" MinWidth="60" />

<DataGridComboBoxColumn x:Name="cbcellnumber" Header="Ячейка" MinWidth="60" />

<DataGridComboBoxColumn x:Name="cbcellemail" Header="Сотрудник" MinWidth="60" />

<DataGridTextColumn Header="Номер накладной" MinWidth="25" Width="\*" Binding="{Binding invoicenmber, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="35"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto" Header="Дата">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<DatePicker SelectedDate="{Binding invoicedate, UpdateSourceTrigger=PropertyChanged}" DisplayDateStart="{Binding Mode=OneWay, Source={x:Static System:DateTime.Today}}"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

<DataGridTextColumn Header="Количество" MinWidth="25" Width="\*" Binding="{Binding quantity, UpdateSourceTrigger=PropertyChanged}">

<DataGridTextColumn.EditingElementStyle>

<Style TargetType="TextBox">

<Setter Property="MaxLength" Value="8"/>

</Style>

</DataGridTextColumn.EditingElementStyle>

</DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Удалить" Click="btnDelInvoice\_Click"/>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

</Grid>

</Page>

# WarehousePage.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace AutoService

{

/// <summary>

/// Логика взаимодействия для WarehousePage.xaml

/// </summary>

public partial class WarehousePage : Page

{

//экземпляр класса управления БД

DBHelper dbhelper = new DBHelper();

/// <summary>

/// Инциализация компонентов страницы Складского отдела

/// </summary>

public WarehousePage()

{

InitializeComponent();

dbhelper.Refresh(dgWarehouse, "select \* from warehouse; ", grdWarehousePage);

DataGridsRefresh();

}

/// <summary>

/// Обработчик обновления данных DataGrid скаладов

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgWarehouse\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

string[] atributes = { dbhelper.GetValueData(dgWarehouse, "warehousename"), dbhelper.GetValueData(dgWarehouse, "warehouseaddress"),dbhelper.GetValueData(dgWarehouse, "warehousename") };

if (dbhelper.EecuteQueryReaderOne($@"select count(warehousename) from warehouse where warehousename = '{dbhelper.GetValueData(dgWarehouse, "warehousename")}'", "count") == "1")

{

dbhelper.UpdateInto("Warehouse", atributes);

}

else

{

dbhelper.InsertInto("Warehouse", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

/// <summary>

/// Обработчик кнопки удаления склада

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данный Склад?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Warehouse", $@"{dbhelper.GetValueData(dgWarehouse, "warehousename")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обработчик обновления данных DataGrid ячеек

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgCells\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

string[] atributes = { dbhelper.GetValueData(dgCells, "cellnumber"), dbhelper.GetTextComboBox(dgCells, 0), dbhelper.GetValueData(dgCells, "cellnumber") };

if (dbhelper.EecuteQueryReaderOne($@"select count(cellnumber) from cell where cellnumber = '{dbhelper.GetValueData(dgCells, "cellnumber")}'", "count") == "1")

{

dbhelper.UpdateInto("Cell", atributes);

}

else

{

dbhelper.InsertInto("Cell", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

/// <summary>

/// Обработчик кнопки удаления ячейки склада

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данную ячейку?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Cell", $@"{dbhelper.GetValueData(dgCells, "cellnumber")}");

DataGridsRefresh();

}

}

/// <summary>

/// Обновление всех DataGrid-ов страницы

/// </summary>

private void DataGridsRefresh()

{

dbhelper.Refresh(dgWarehouse, "select \* from warehouse; ", grdWarehousePage);

dbhelper.Refresh(dgCells, "select \* from Cell; ", grdWarehousePage);

dbhelper.Refresh(dgInvoice, "select \* from Invoice; ", grdWarehousePage);

dbhelper.BindComboBox(cbWarehouseName, "select \* from warehouse", "warehousename", "warehousename");

dbhelper.BindComboBox(cbseriesnumber, "select \* from Disposable", "seriesnumber", "seriesnumber");

dbhelper.BindComboBox(cbcellnumber, "select (Cell.warehousename || ' - '|| Cell.cellnumber) as cl, cellnumber from cell", "cl", "cellnumber");

dbhelper.BindComboBox(cbcellemail, "select lastname ||' '|| firstname||' '|| middlename as colum, email from Users where RoleName = 'Сотрудник склада'", "colum", "email");

}

/// <summary>

/// Обработчик обновления данных DataGrid наклодной

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

private void dgInvoice\_RowEditEnding(object sender, DataGridRowEditEndingEventArgs e)

{

string[] atributes = { dbhelper.GetValueData(dgInvoice, "invoicenmber"),

dbhelper.GetValueData(dgInvoice, "invoicedate"),

dbhelper.GetValueData(dgInvoice,"quantity"),

dbhelper.GetTextComboBox(dgInvoice, 0),

dbhelper.EecuteQueryReaderOne($@"select \* from Cell where (Cell.warehousename || ' - '|| Cell.cellnumber) = '{dbhelper.GetTextComboBox(dgInvoice, 1)}'", "cellnumber"),

dbhelper.EecuteQueryReaderOne($@"select \* from Users where (lastname ||' '|| firstname||' '|| middlename) = '{dbhelper.GetTextComboBox(dgInvoice, 2)}'", "email"),

dbhelper.GetValueData(dgInvoice, "invoicenmber") };

if (dbhelper.EecuteQueryReaderOne($@"select count(invoicenmber) from invoice where invoicenmber = '{dbhelper.GetValueData(dgInvoice, "invoicenmber")}'", "count") == "1")

{

dbhelper.UpdateInto("Invoice", atributes);

}

else

{

dbhelper.InsertInto("Invoice", atributes.Take(atributes.Length - 1).ToArray());

}

DataGridsRefresh();

}

/// <summary>

/// Обработчик кнопки удаления накладной

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

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

{

if (MessageBox.Show("Вы точно хотите удалить данную накладную?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

dbhelper.DeleteInto("Invoice", $@"{dbhelper.GetValueData(dgInvoice, "invoicenmber")}");

DataGridsRefresh();

}

}

}

}

# EmailHelper.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Mail;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

namespace AutoService

{

/// <summary>

/// Управление почтовым клиеном

/// </summary>

class EmailHelper

{

/// <summary>

/// Отправка электронного письма

/// </summary>

/// <param name="email">Электронная почта</param>

/// <param name="r">Код для восстановления</param>

public async Task SendMail(string email, int r)

{

// отправитель - устанавливаем адрес и отображаемое в письме имя

MailAddress from = new MailAddress("vepritskiy1999@mail.ru", "Сервис-Про");

// кому отправляем

MailAddress to = new MailAddress(email);

// создаем объект сообщения

MailMessage m = new MailMessage(from, to);

// тема письма

m.Subject = "Код для восстановления пароля";

// текст письма

m.Body = $@"<h1>Ваш код для восстановления пароля: <b>{r}</b><p>";

// письмо представляет код html

m.IsBodyHtml = true;

// адрес smtp-сервера и порт, с которого будем отправлять письмо

SmtpClient smtp = new SmtpClient("smpt.mail.ru", 25);

// логин и пароль

smtp.Credentials = new NetworkCredential("vepritskiy1999@mail.ru", "NAFzSL6mwayXkA5sd6bW");

smtp.EnableSsl = true;

try

{

await smtp.SendMailAsync(m);

}

catch (Exception exc)

{

MessageBox.Show(exc.Message);

}

Console.Read();

}

/// <summary>

/// Отправка электронного письма

/// </summary>

/// <param name="email">электронная почта</param>

/// <param name="title">заголовок</param>

/// <param name="message">сообщение</param>

public async Task SendMailOrder(string email, string title, string message)

{

// отправитель - устанавливаем адрес и отображаемое в письме имя

MailAddress from = new MailAddress("vepritskiy1999@mail.ru", "Сервис-Про");

// кому отправляем

MailAddress to = new MailAddress(email);

// создаем объект сообщения

MailMessage m = new MailMessage(from, to);

// тема письма

m.Subject = title;

// текст письма

m.Body = message;

// письмо представляет код html

m.IsBodyHtml = true;

// адрес smtp-сервера и порт, с которого будем отправлять письмо

SmtpClient smtp = new SmtpClient("smpt.mail.ru", 25);

// логин и пароль

smtp.Credentials = new NetworkCredential("vepritskiy1999@mail.ru", "NAFzSL6mwayXkA5sd6bW");

smtp.EnableSsl = true;

try

{

await smtp.SendMailAsync(m);

}

catch (Exception exc)

{

MessageBox.Show(exc.Message);

}

Console.Read();

}

}

}

# DBHelpertests.cs

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using AutoService;

namespace AutoServiceTests

{

[TestClass]

public class DBHelperTests

{

string email = "";

string password = "";

string lastName = "";

string firstName = "";

string middleName = "";

string post = "";

DBHelper dBHelper = new DBHelper();

[TestMethod]

public void Authorization\_AdminandAdmin\_Administratorreturned()

{

//arange

email = "Admin";

password = "Admin";

string expected = "Администратор";

//act

string actual = dBHelper.Authorization(email, password);

//assert

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void UsersInserInto\_EmailandPasswordandFIandPost\_1\_returned()

{

//arange

email = "user@mail.ru";

password = "User!123";

lastName = "LastName";

firstName = "FirstName";

middleName = "MiddleName";

post = "Администратор";

string expected = "1";

string[] atributes = { email, password, lastName, firstName, middleName, post };

//act

dBHelper.InsertInto("Users", atributes);

string actual = dBHelper.EecuteQueryReaderOne($@"select count(Email) from Users where Email = '{email}'", "count");

//assert

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void UsersUpdateInto\_EmailandPasswordandFIandPost\_1\_returned()

{

//arange

string newEmail = "newuser@mail.ru";

email = "user@mail.ru";

password = "User!123";

lastName = "LastName2";

firstName = "FirstName2";

middleName = "MiddleName2";

post = "Клиент";

string expected = "1";

string[] atributes = { email, password, lastName, firstName, middleName, post, newEmail };

//act

dBHelper.UpdateInto("Users", atributes);

string actual = dBHelper.EecuteQueryReaderOne($@"select count(Email) from Users where Email = '{newEmail}' and Password = '{password}' and LastName = '{lastName}' and FirstName = '{firstName}' and MiddleName = '{middleName}' and RoleName = '{post}'", "count");

//assert

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void UsersZeleteInto\_Email\_1\_returned()

{

//arange

email = "newuser@mail.ru";

string expected = "0";

//act

dBHelper.DeleteInto("Users", email);

string actual = dBHelper.EecuteQueryReaderOne($@"select count(Email) from Users where Email = '{email}'", "count");

//assert

Assert.AreEqual(expected, actual);

}

}

}

# Framemanager

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace AutoService

{

/// <summary>

/// Навигация страниц

/// </summary>

class FrameManager

{

/// <summary>

/// Получение страниц

/// </summary>

public static Frame MainFrame { get; set; }

}

}

# Validationdata.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using System.Windows.Media;

namespace AutoService

{

/// <summary>

/// Валидация данных

/// </summary>

class ValidationData

{

/// <summary>

/// Валидация на цифры и английские буквы

/// </summary>

/// <param name="c"></param>

/// <returns>bool значение</returns>

bool IsValidEngNum(char c)

{

if (c >= '0' && c <= '9')

return true;

if (c >= 'A' && c <= 'Z')

return true;

if (c >= 'a' && c <= 'z')

return true;

string spec = "!\"#$%&'()\*+,-./::<=>?@[\\]:\_{|}";

if (c.ToString().IndexOfAny(spec.ToCharArray()) > -1)

return true;

return false;

}

/// <summary>

/// Валидация на цифры и русские буквы

/// </summary>

/// <param name="c"></param>

/// <returns>bool значение</returns>

bool IsValidRusNum(char c)

{

if (c >= '0' && c <= '9')

return true;

if (c >= 'А' && c <= 'Я')

return true;

if (c >= 'а' && c <= 'я')

return true;

string spec = "!\"#$%&'()\*+,-./::<=>?@[\\]:\_{|}";

if (c.ToString().IndexOfAny(spec.ToCharArray()) > -1)

return true;

return false;

}

/// <summary>

/// Валидация на русские буквы

/// </summary>

/// <param name="c"></param>

/// <returns>bool значение</returns>

bool IsValidRus(char c)

{

if (c >= '0' && c <= '9')

return true;

if (c >= 'А' && c <= 'Я')

return true;

if (c >= 'а' && c <= 'я')

return true;

string spec = "!\"#$%&'()\*+,-./::<=>?@[\\]:\_{|}";

if (c.ToString().IndexOfAny(spec.ToCharArray()) > -1)

return true;

return false;

}

/// <summary>

/// Валидация на цифры и английские буквы и русские буквы

/// </summary>

/// <param name="c"></param>

/// <returns>bool значение</returns>

bool IsValidRusEngNum(char c)

{

if (c >= '0' && c <= '9')

return true;

if (c >= 'A' && c <= 'Z')

return true;

if (c >= 'a' && c <= 'z')

return true;

if (c >= 'А' && c <= 'Я')

return true;

if (c >= 'а' && c <= 'я')

return true;

string spec = "!\"#$%&'()\*+,-./::<=>?@[\\]:\_{|}";

if (c.ToString().IndexOfAny(spec.ToCharArray()) > -1)

return true;

return false;

}

/// <summary>

/// Валидация на цифры

/// </summary>

/// <param name="c"></param>

/// <returns>bool значение</returns>

bool IsValidNum(char c)

{

if (c >= '0' && c <= '9')

return true;

return false;

}

/// <summary>

/// Обработчики заполнения и вставки данных в textbox

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="e">данные события</param>

public void tb1\_PreviewTextInputAngNum(object sender, TextCompositionEventArgs e)

{

e.Handled = !e.Text.All(IsValidEngNum);

}

public void tb1\_PastingAngNum(object sender, DataObjectPastingEventArgs e)

{

var stringData = (string)e.DataObject.GetData(typeof(string));

if (stringData == null || !stringData.All(IsValidEngNum))

e.CancelCommand();

}

public void tb1\_PreviewTextInputRusNum(object sender, TextCompositionEventArgs e)

{

e.Handled = !e.Text.All(IsValidRusNum);

}

public void tb1\_PastingRusNum(object sender, DataObjectPastingEventArgs e)

{

var stringData = (string)e.DataObject.GetData(typeof(string));

if (stringData == null || !stringData.All(IsValidRusNum))

e.CancelCommand();

}

public void tb1\_PreviewTextInputRus(object sender, TextCompositionEventArgs e)

{

e.Handled = !e.Text.All(IsValidRus);

}

public void tb1\_PastingRus(object sender, DataObjectPastingEventArgs e)

{

var stringData = (string)e.DataObject.GetData(typeof(string));

if (stringData == null || !stringData.All(IsValidRus))

e.CancelCommand();

}

public void tb1\_PreviewTextInputRusEngNum(object sender, TextCompositionEventArgs e)

{

e.Handled = !e.Text.All(IsValidRusEngNum);

}

public void tb1\_PastingRusEngNum(object sender, DataObjectPastingEventArgs e)

{

var stringData = (string)e.DataObject.GetData(typeof(string));

if (stringData == null || !stringData.All(IsValidRusEngNum))

e.CancelCommand();

}

public void tb1\_PreviewTextInputNum(object sender, TextCompositionEventArgs e)

{

e.Handled = !e.Text.All(IsValidNum);

}

public void tb1\_PastingNum(object sender, DataObjectPastingEventArgs e)

{

var stringData = (string)e.DataObject.GetData(typeof(string));

if (stringData == null || !stringData.All(IsValidNum))

e.CancelCommand();

}

/// <summary>

/// Валидация электронной почты

/// </summary>

/// <param name="email">Электронная почта</param>

/// <returns>bool значение</returns>

public bool IsValidEmail(string email)

{

var trimmedEmail = email.Trim();

if (trimmedEmail.EndsWith("."))

{

Show("Введен некорректный адрес электронной почты").GetAwaiter(); ;

return false; // suggested by @TK-421

}

try

{

var addr = new System.Net.Mail.MailAddress(email);

return addr.Address == trimmedEmail;

}

catch

{

Show("Введен некорректный адрес электронной почты").GetAwaiter();

return false;

}

}

/// <summary>

/// Валидация пароля

/// </summary>

/// <param name="s">пароль</param>

/// <returns>bool значение</returns>

public bool IsValidPassword(string s)

{

int contain = 0;

if (s.Length >= 7)

contain += 1;

string zag = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

if (s.IndexOfAny(zag.ToCharArray()) > -1)

contain += 1;

string small = "abcdefghijklmnopqrstuvwxyz";

if (s.IndexOfAny(small.ToCharArray()) > -1)

contain += 1;

string numb = "123456789";

if (s.IndexOfAny(numb.ToCharArray()) > -1)

contain += 1;

string spec = "!\"#$%&'()\*+,-./::<=>?@[\\]:\_{|}";

if (s.IndexOfAny(spec.ToCharArray()) > -1)

contain += 1;

if (contain == 5)

return true;

else

{

Show("Пароль должен содержать не менее 7-ми символов, хотя бы одну заглавную букву и хотя бы один спец символ, Буквы латинские").GetAwaiter();

return false;

}

}

/// <summary>

/// хэширование MD5

/// </summary>

/// <param name="decrypted"></param>

/// <returns></returns>

public string Encrypt(string decrypted)

{

byte[] data = UTF8Encoding.UTF8.GetBytes(decrypted);

MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();

TripleDESCryptoServiceProvider tripDes = new TripleDESCryptoServiceProvider();

tripDes.Key = md5.ComputeHash(UTF8Encoding.UTF8.GetBytes("123"));

tripDes.Mode = CipherMode.ECB;

ICryptoTransform transform = tripDes.CreateEncryptor();

byte[] result = transform.TransformFinalBlock(data, 0, data.Length);

return Convert.ToBase64String(result);

}

public string Decrypt(string encrypted)

{

byte[] data = Convert.FromBase64String(encrypted);

MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();

TripleDESCryptoServiceProvider tripDes = new TripleDESCryptoServiceProvider();

tripDes.Key = md5.ComputeHash(UTF8Encoding.UTF8.GetBytes("123"));

tripDes.Mode = CipherMode.ECB;

ICryptoTransform transform = tripDes.CreateDecryptor();

byte[] result = transform.TransformFinalBlock(data, 0, data.Length);

return UTF8Encoding.UTF8.GetString(result);

}

/// <summary>

/// Вывод ошибок

/// </summary>

/// <param name="errorMessage">сообщение об ошибке</param>

/// <returns></returns>

public async Task Show(string errorMessage)

{

Grid gridError = Application.Current.MainWindow.FindName("ErrorGrid") as Grid;

RowDefinition errorRow = Application.Current.MainWindow.FindName("ErrorRow") as RowDefinition;

gridError.Children[0].Visibility = Visibility.Visible;

Label label = gridError.Children[0] as Label;

label.Content = errorMessage;

gridError.Background = new SolidColorBrush(Colors.Black);

gridError.VerticalAlignment = VerticalAlignment.Stretch;

await Task.Delay(3000);

gridError.Background = new SolidColorBrush((Color)ColorConverter.ConvertFromString("#445c93"));

gridError.VerticalAlignment = VerticalAlignment.Bottom;

label.Visibility = Visibility.Hidden;

}

}

}

# DBHelper.cs

using Npgsql;

using System;

using System.Collections;

using System.Collections.Generic;

using System.Data;

using System.Diagnostics;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

namespace AutoService

{

/// <summary>

/// Класс формирования запросов в БД

/// </summary>

public class DBHelper

{

//Строка подключения

public static string conn\_param = "Server=192.168.0.12; Port=5432;User Id=postgres;Password=1;Database=AutoService; Timeout=300; CommandTimeout=300"; //25.41.59.168 192.168.169.204 192.168.0.16

//Перенаправление подключения pgsql

public NpgsqlConnection connection = new NpgsqlConnection(conn\_param);

//Считыватель значения БД

NpgsqlDataReader dataReader = null;

// инструкция SQL для выполнения в базе данных PostgreSQL

NpgsqlCommand command = null;

//Экземпляр класса валидации

ValidationData valid = new ValidationData();

/// <summary>

/// формирования запроса на авторизацию

/// </summary>

/// <param name="login">логин пользователя</param>

/// <param name="password">пароль авторизация</param>

/// <returns>роль пользователя</returns>

public string Authorization(string login, string password)

{

try

{

connection.Open();

command = new NpgsqlCommand($@"select Sign\_In('{login}','{password}')", connection);

dataReader = command.ExecuteReader();

string roleName = "";

while (dataReader.Read())

{

roleName = dataReader["sign\_in"].ToString();

}

if (roleName == "false")

{

valid.Show("Неправильный логин или пароль").GetAwaiter();

return null;

}

else

return roleName;

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

return null;

}

finally

{

dataReader.Close();

connection.Close();

}

}

/// <summary>

/// Обновление всей страницы

/// </summary>

/// <param name="sender">ссылка на элемент управления</param>

/// <param name="query">запрос</param>

/// <param name="grd">Grid</param>

public void Refresh(object sender, string query, Grid grd)

{

try

{

connection.Close();

connection.Open();

command = new NpgsqlCommand(query, connection);

DataTable datatbl = new DataTable();

datatbl.Load(command.ExecuteReader());

(sender as DataGrid).ItemsSource = datatbl.DefaultView;

if (grd != null)

{

try

{

foreach (Control ctl in grd.Children)

{

if (ctl.GetType() == typeof(DatePicker))

((DatePicker)ctl).SelectedDate = null;

if (ctl.GetType() == typeof(RichTextBox))

{

FlowDocument document = new FlowDocument();

Paragraph paragraph = new Paragraph();

paragraph.Inlines.Add(new Bold(new Run("")));

document.Blocks.Add(paragraph);

((RichTextBox)ctl).Document = document;

}

if (ctl.GetType() == typeof(TextBox))

{

((TextBox)ctl).Text = String.Empty;

}

if (ctl.GetType() == typeof(ComboBox))

{

((ComboBox)ctl).SelectedIndex = -1;

}

}

}

catch { }

}

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

}

finally

{

connection.Close();

}

}

/// <summary>

/// Выполнение запроса

/// </summary>

/// <param name="query">запрос</param>

public void EecuteQuery(string query)

{

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

dataReader = command.ExecuteReader();

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

}

finally

{

connection.Close();

}

}

public List<string> EecuteQueryReader(string query, List<string> data, string tablename)

{

if (query == "select distinct extract(year from registrationdate) from Users;")

{

List<int> dataint = new List<int>();

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

dataReader = command.ExecuteReader();

while (dataReader.Read())

{

dataint.Add(Convert.ToInt32(dataReader[tablename].ToString()));

}

dataint.Sort();

foreach(int a in dataint)

{

data.Add(a.ToString());

}

return data;

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

return data;

}

finally

{

connection.Close();

dataReader.Close();

}

}

else

{

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

dataReader = command.ExecuteReader();

while (dataReader.Read())

{

data.Add(dataReader[tablename].ToString());

}

return data;

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

return data;

}

finally

{

connection.Close();

dataReader.Close();

}

}

}

/// <summary>

/// Считывание определенного значения из БД

/// </summary>

/// <param name="query">запрос</param>

/// <param name="tablename">название таблицы</param>

/// <returns>значение из БД</returns>

public string EecuteQueryReaderOne(string query, string tablename)

{

string data = "";

try

{

connection.Close();

connection.Open();

command = new NpgsqlCommand(query, connection);

dataReader = command.ExecuteReader();

while (dataReader.Read())

{

data = dataReader[tablename].ToString();

}

return data;

}

catch (NpgsqlException ex)

{

System.Windows.MessageBox.Show(ex.Message);

return data;

}

finally

{

dataReader.Close();

connection.Close();

}

}

//Путь к dump pgsql

string strPG\_dumpPath = "SET PGPASSWORD=1\r\n\r\ncd /D C:\\Program Files\r\n\r\ncd PostgreSQL\r\n\r\ncd 13\r\n\r\ncd bin\r\n\r\n";

//Ip сервера

string strServer = "192.168.0.12";

//Порт

string strPort = "5432";

//Название ДБ

string strDatabaseName = "AutoService";

/// <summary>

/// Создание файла Backup

/// </summary>

/// <param name="pathSave">Путь сохранения</param>

public void Backup(string pathSave)

{

try

{

StreamWriter sw = new StreamWriter("DBBackup.bat");

// Do not change lines / spaces b/w words.

StringBuilder strSB = new StringBuilder(strPG\_dumpPath);

if (strSB.Length != 0)

{

//strSB.Append("SET PGPASSWORD=1");

strSB.Append("pg\_dump.exe --host " + strServer + " --port " + strPort + " --username postgres --format custom --blobs --verbose --file ");

strSB.Append("\"" + pathSave + "\"");

strSB.Append(" \"" + strDatabaseName+"\"" + "\r\n\r\n");

sw.WriteLine(strSB);

sw.Dispose();

sw.Close();

Process processDB = Process.Start("DBBackup.bat");

MessageBox.Show("Резервная копия успешно сохранена");

}

else

{

MessageBox.Show("Пожалуйста, укажите место для создания резервной копии");

}

}

catch

{ }

}

/// <summary>

/// Восстановление БД

/// </summary>

/// <param name="pathFile">путь к резервной копии</param>

public void Restore(string pathFile)

{

strDatabaseName = "ProbaRestore";

connection.Open();

try

{

command = new NpgsqlCommand("DROP DATABASE \"ProbaRestore\";", connection);

command.ExecuteNonQuery();

}

catch { }

command = new NpgsqlCommand("Create DATABASE \"ProbaRestore\";", connection);

command.ExecuteNonQuery();

connection.Close();

try

{

if (strDatabaseName != "")

{

if (pathFile != "")

{

StreamWriter sw = new StreamWriter("DBRestore.bat");

// Do not change lines / spaces b/w words.

StringBuilder strSB = new StringBuilder(strPG\_dumpPath);

if (strSB.Length != 0)

{

strSB.Append("pg\_restore.exe --host " + strServer +

" --port " + strPort + " --username postgres --dbname");

strSB.Append(" \"" + strDatabaseName + "\"");

strSB.Append(" --verbose ");

strSB.Append("\"" + pathFile + "\"");

sw.WriteLine(strSB);

sw.Dispose();

sw.Close();

Process processDB = Process.Start("DBRestore.bat");

MessageBox.Show("Успешный backup данных");

}

else

{

MessageBox.Show("Пожалуйста, введите путь сохранения, чтобы получить резервную копию");

}

}

}

else

{

MessageBox.Show("Please enter the Database name to Restore!");

}

}

catch

{ }

}

/// <summary>

/// Выборка Данных из DataGrid

/// </summary>

/// <param name="grd">DataGrid</param>

/// <param name="value">Наименование столбца</param>

/// <returns></returns>

public string GetValueData(DataGrid grd, string value)

{

try

{

IList rows = grd.SelectedItems;

DataRowView row = (DataRowView)grd.SelectedItems[0];

return row[value].ToString();

}

catch

{

return "";

}

}

public void InsertInto(string tableName, string[] atributes)

{

string query = "";

query = $@"call {tableName}\_insert(";

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

{

if (i != atributes.Length - 1)

query += $@"'{atributes[i]}',";

else

query += $@"'{atributes[i]}');";

}

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

command.ExecuteNonQuery();

}

catch (NpgsqlException ex)

{

valid.Show(ex.Message).GetAwaiter();

}

finally

{

connection.Close();

}

}

/// <summary>

/// Формирование запроса на удаление

/// </summary>

/// <param name="tableName">Название таблицы</param>

/// <param name="ID">Уникальный индетификатор</param>

public void DeleteInto(string tableName, string ID)

{

string query = $@"call {tableName}\_Delete('{ID}')";

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

command.ExecuteNonQuery();

}

catch (NpgsqlException ex)

{

valid.Show(ex.Message).GetAwaiter();

}

finally

{

connection.Close();

}

}

/// <summary>

/// Формирование запроса на обновление данных в БД

/// </summary>

/// <param name="tableName">название таблицы</param>

/// <param name="atributes">данные для добавления</param>

public void UpdateInto(string tableName, string[] atributes)

{

string query = $@"call {tableName}\_Update(";

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

{

if (i != atributes.Length - 1)

query += $@"'{atributes[i]}',";

else

query += $@"'{atributes[i]}');";

}

try

{

connection.Open();

command = new NpgsqlCommand(query, connection);

command.ExecuteNonQuery();

}

catch (NpgsqlException ex)

{

valid.Show(ex.Message).GetAwaiter();

}

finally

{

connection.Close();

}

}

/// <summary>

/// Биндинг Combobox

/// </summary>

/// <param name="comboBox">ComboBox</param>

/// <param name="query">Запрос</param>

/// <param name="tableNameForDisplay">Выводиме значения</param>

/// <param name="tableNameForBinding">Значения выводимых значений</param>

public void BindComboBox(DataGridComboBoxColumn comboBox,string query, string tableNameForDisplay ,string tableNameForBinding)

{

command = new NpgsqlCommand(query, connection);

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter();

adapter.SelectCommand = command;

DataTable table = new DataTable();

adapter.Fill(table);

comboBox.ItemsSource = table.DefaultView;

comboBox.DisplayMemberPath = tableNameForDisplay;

comboBox.SelectedValuePath = tableNameForBinding;

comboBox.SelectedValueBinding = new Binding(tableNameForBinding);

}

/// <summary>

/// Получение значения ячеки DataGrid

/// </summary>

/// <param name="grd">DataGrid</param>

/// <param name="column">Наименование столбца DataGrid</param>

/// <returns>значение ячейки</returns>

public string GetTextComboBox(DataGrid grd, int column)

{

DataGridRow row = grd.ItemContainerGenerator.ContainerFromIndex(grd.SelectedIndex) as DataGridRow;

ComboBox ele = grd.Columns[column].GetCellContent(row) as ComboBox;

return ele.Text;

}

}

}