Содержание

[1) Создаем папки: Windows, Pages, Models, Classes, ImagesAndIcons. 2](#_Toc159352820)

[2) Перенос стартового окна и стиль 2](#_Toc159352821)

[3) Добавляем название организации в заголовок окна, иконку приложения и разметку 2](#_Toc159352822)

[4) Загрузка страницы на фрейм 3](#_Toc159352823)

[5) Helper 3](#_Toc159352824)

[6) Скрытие ненужной даты в поле списка 4](#_Toc159352825)

[7) Дополнительные поля в классах моделей 4](#_Toc159352826)

[7) Валидация 7](#_Toc159352827)

[8) Обновление данных одного окна при изменении данных в другом окне 9](#_Toc159352828)

[9) Добавление, удаление картинки 9](#_Toc159352829)

[10) Накопление списка ошибок при сохранении данных 10](#_Toc159352830)

[11) Выбор одного элемента из списка 11](#_Toc159352831)

[12.1) Выбор нескольких элементов из списка (удаление) 12](#_Toc159352832)

[12.2) Выбор нескольких элементов из списка (удаление с зависимостями) 12](#_Toc159352833)

[13) Добавление надписи, если список пуст 13](#_Toc159352834)

[14) Страница со списком и пагинацией 14](#_Toc159352835)

[15) Изменение, добавление, удаление 26](#_Toc159352836)

[16) MessageBox 39](#_Toc159352837)

[17) Форматы вывода дат 39](#_Toc159352838)

# 1) Создаем папки: Windows, Pages, Models, Classes, ImagesAndIcons.

# 2) Перенос стартового окна и стиль

Выносим стартовое окно в папку Windows

После этого в файле «App.xaml» меняем «StartupUri» на: StartupUri="Windows/MainWindow.xaml".

Добавляем в этом же файле стили для проекта – задаем необходимые цвета и шрифт для кнопок и текста в блоке Application.Resources:

<Application.Resources>

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

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

<Setter Property="Background" Value="#FFFF4A6B" />

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

<Setter Property="Cursor" Value="Hand" />

<Setter Property="FontFamily" Value="Tahoma" />

</Style>

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

<Setter Property="FontFamily" Value="Tahoma" />

</Style>

</Application.Resources>

3) Добавляем название организации в заголовок окна, иконку приложения и разметку:

Основные характеристики, которые учитываются: Title="Бархатные бровки" Height="710" Width="1000" MinHeight = "710" MinWidth ="1000" WindowStartupLocation = "CenterScreen" Icon="/ImagesAndIcons/beauty\_logo.ico"

Разметка Main окна:

<Window x:Class="\_20.\_101\_09\_BeautySalon.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:\_20.\_101\_09\_BeautySalon"

mc:Ignorable="d"

Title="Бархатные бровки" Height="710" Width="1000" MinHeight = "710" MinWidth ="1000" WindowStartupLocation = "CenterScreen" Icon="/ImagesAndIcons/beauty\_logo.ico">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="#FFE1E4FE">

<Button x:Name="btnBack" Content="Назад" Margin="0,17,10,17" HorizontalAlignment="Right" Width="46" Click="btnBack\_Click"/>

<Image HorizontalAlignment="Left" Margin="10,7,0,7" Width="45" Source="/ImagesAndIcons/beauty\_logo.png"/>

</Grid>

<Frame x:Name="fr\_content" Grid.Row="1" NavigationUIVisibility="Hidden" LoadCompleted="fr\_content\_LoadCompleted" ContentRendered="fr\_content\_ContentRendered"/>

</Grid>

</Window>

4) Загрузка страницы на фрейм:

* С Main окна:

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

fr\_content.Content = new ClientPage();

}

private void fr\_content\_ContentRendered(object sender, EventArgs e)

{

if (fr\_content.CanGoBack)

{

btnBack.Visibility = Visibility.Visible;

}

else

{

btnBack.Visibility = Visibility.Hidden;

}

}

private void fr\_content\_LoadCompleted(object sender, NavigationEventArgs e)

{

}

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

{

fr\_content.GoBack();

}

}

}

* Со страницы на другую:

private void bt\_guest\_Click(object sender, RoutedEventArgs e)

{

NavigationService.Navigate(new Client(null, null));

}

5) Helper:

internal class Helper

{

public static bool flag = false;

private static Entities BeautySalonEntities;

public static Entities GetContext()

{

if (BeautySalonEntities == null)

{

BeautySalonEntities = new Entities();

}

return BeautySalonEntities;

}

}

6) Скрытие ненужной даты в поле списка:

public class DateTimeToVisibilityConverter : MarkupExtension, IValueConverter

{

private static DateTimeToVisibilityConverter \_converter;

public object Convert(object value, Type targetType, object parameter, CultureInfo culture)

{

if (value is DateTime dateTime)

{

// Проверяем, равно ли значение DateService минимальной дате

if (dateTime == DateTime.MinValue)

{

// Если да, то скрываем дату

return string.Empty;

}

}

// Если значение DateService не минимальное, то отображаем содержимое TextBlock

return value;

}

public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)

{

throw new NotSupportedException();

}

public override object ProvideValue(IServiceProvider serviceProvider)

{

return \_converter ?? (\_converter = new DateTimeToVisibilityConverter());

}

}

Колонка ListView:

<GridViewColumn Header="Дата последнего посещения" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding DateService, Converter={local:DateTimeToVisibilityConverter}}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

7) Дополнительные поля в классах моделей:

public partial class Client

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Client()

{

this.ClientService = new HashSet<ClientService>();

this.Tag = new HashSet<Tag>();

}

public int ID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Patronymic { get; set; }

public string GenderCode { get; set; }

public string Phone { get; set; }

public Nullable<System.DateTime> Birthday { get; set; }

public string Email { get; set; }

public System.DateTime RegistrationDate { get; set; }

public string PhotoPath { get; set; }

public virtual Gender Gender { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ClientService> ClientService { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Tag> Tag { get; set; }

public string CountService

{

get

{

return this.ClientService.Count.ToString();

}

}

public System.DateTime DateService

{

get

{

return Helper.GetContext().ClientService

.Where(c => c.ClientID == this.ID)

.OrderByDescending(c => c.StartTime)

.Select(c => c.StartTime)

.FirstOrDefault();

}

}

public int GenderList

{

get

{

return Convert.ToInt32(this.GenderCode) - 1; ;

}

}

public string ImgPath

{

get

{

var path = "/ImagesAndIcons/Услуги салона красоты/" + this.PhotoPath;

return path;

}

}

public List<Tag> TagList

{

get

{

return this.Tag.ToList();

}

}

public List<ClientService> ServiceList

{

get

{

return this.ClientService.ToList();

}

}

}

public partial class Service

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Service()

{

this.ClientService = new HashSet<ClientService>();

this.ServicePhoto = new HashSet<ServicePhoto>();

}

public int ID { get; set; }

public string Title { get; set; }

public string MainImagePath { get; set; }

public int DurationInSeconds { get; set; }

public decimal Cost { get; set; }

public Nullable<double> Discount { get; set; }

public string Description { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ClientService> ClientService { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ServicePhoto> ServicePhoto { get; set; }

public string ImgPath

{

get

{

var path = "/ImagesAndIcons/Услуги салона красоты/" + this.MainImagePath;

return path;

}

}

public int DurationInMin

{

get

{

return this.DurationInSeconds / 60;

}

}

public double DiscountCost

{

get

{

if (Discount.HasValue)

{

decimal result = Cost / 100;

return (double)(Cost - result\* (decimal)this.Discount);

}

else return (double)Cost;

}

}

public string DiscountText

{

get

{

if (this.Discount.HasValue) return $"\*скидка {Discount}%";

else return "";

}

}

public string CostText

{

get

{

if (Discount.HasValue)

{

double cost = (double)Cost;

return cost.ToString();

}

else return "";

}

}

}

public partial class Tag

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Tag()

{

this.Client = new HashSet<Client>();

}

public int ID { get; set; }

public string Title { get; set; }

public string Color { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Client> Client { get; set; }

public string Background

{

get

{

return "#" + this.Color;

}

}

}

7) Валидация:

public static class ValidatorExtensions

{

public static bool IsValidEmailAddress(this string s)

{

//Regex regex = new Regex(@"^[\w-\.]+@([\w-]+\.)+[\w-]{2,}$");

Regex regex = new Regex(@"^[\w\.-]+@[a-zA-Z\d\.-]+\.[a-zA-Z]{2,}$");

//Regex regex = new Regex(@"^[a-zA-Z][\w\.-]\*[a-zA-Z0-9]@[a-zA-Z0-9][\w\.-]\*[a-zA-Z0-9]\.[a-zA-Z][a-zA-Z\.]\*[a-zA-Z]$");

return !regex.IsMatch(s);

}

public static bool IsValidPhone(this string s)

{

Regex regex = new Regex("[^0-9+() -]+");

return !regex.IsMatch(s);

}

public static bool IsValidFIO(this string s)

{

Regex regex = new Regex("[^a-zа-яА-ЯA-Z -]+");

return !regex.IsMatch(s);

}

public static bool IsValidColor(this string s)

{

Regex regex = new Regex("[^0-9A-Z]+");

return !regex.IsMatch(s);

}

public static bool IsValidDiscount(this string s)

{

Regex regex = new Regex("[^0-9]+");

return !regex.IsMatch(s);

}

public static bool IsValidCost(this string s)

{

////Regex regex = new Regex("^[0-9]+(\\.[0-9]{1,2})?$");

//Regex regex = new Regex("[^0-9.]+");

//return !regex.IsMatch(s);

Regex regex = new Regex(@"^\d+(\.\d{0,2})?$");

return regex.IsMatch(s);

}

public static bool IsValidTitle(this string s)

{

Regex regex = new Regex("[^a-zA-Zа-яА-Я0-9\\s]+");

return !regex.IsMatch(s);

}

}

В классе страницы (окна):

private void tbFirstName\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbFirstName, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbLastName\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbLastName, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbPatronymic\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbPatronymic, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbEmail\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

e.Handled = !ValidatorExtensions.IsValidEmailAddress(e.Text);

}

private void CountAndValidText(TextBox textBox, TextCompositionEventArgs e, int count, bool valid)

{

if (textBox.Text.Length >= count && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = valid;

}

}

private void Phone\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbPhone, e, 18, !ValidatorExtensions.IsValidPhone(e.Text));

//e.Handled = !ValidatorExtensions.IsValidPhone(e.Text);

}

# 8) Обновление данных одного окна при изменении данных в другом окне

Передаем в окно где данные изменяются окно, где данные нужно будет обновить (clientServiceWindow):

public partial class EditService : Window

{

Service service = new Service();

Entities db;

ClientServiceWindow clientServiceWindow;

public EditService(Service service, Entities db, ClientServiceWindow clientServiceWindow)

{

InitializeComponent();

this.service = service;

this.db = db;

this.clientServiceWindow = clientServiceWindow;

DataContext = service;

}

Добавляем метод Closed и в нем к переданной переменной окна вызываем метод .который показывает данные:

private void Window\_Closed(object sender, EventArgs e)

{

try

{

if (clientServiceWindow != null)

{

clientServiceWindow.Load();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

9) Добавление, удаление картинки:

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

{

try

{

var asmName = new AssemblyNameDefinition("DynamicAssembly", new Version(1, 0, 0, 0));

var assembly = AssemblyDefinition.CreateAssembly(asmName, "<Module>", ModuleKind.Dll);

OpenFileDialog GetImageDialog = new OpenFileDialog(); // Открытие диалогового окна

string folderpath = System.IO.Directory.GetParent(Environment.CurrentDirectory).Parent.FullName + "\\ImagesAndIcons\\Услуги салона красоты\\";

GetImageDialog.Title = "Выберите изображение";

GetImageDialog.Filter = "Файлы изображений: (\*.png,\*.jpg,\*.jpeg)| \*.png;\*.jpg;\*.jpeg"; // Фильтр типов файлов

GetImageDialog.InitialDirectory = folderpath;

if (GetImageDialog.ShowDialog() == true)

service.MainImagePath = GetImageDialog.SafeFileName;//Добавление наименования файла в БД

var sourse = new BitmapImage(new Uri(GetImageDialog.FileName));

imgPhoto.Source = sourse;

/\*

string filePath = folderpath + System.IO.Path.GetFileName(GetImageDialog.FileName);

File.Copy(GetImageDialog.FileName, filePath, true);

byte[] imageData = File.ReadAllBytes(filePath);

var imageResource = new EmbeddedResource(GetImageDialog.FileName, ManifestResourceAttributes.Private, imageData);

assembly.MainModule.Resources.Add(imageResource);\*/

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

string defaultImagePath = (string)FindResource("defaultImage"); // Получаем путь к изображению из ресурсов

if (!string.IsNullOrEmpty(defaultImagePath))

{

BitmapImage defaultImage = new BitmapImage(new Uri(defaultImagePath)); // Создаем BitmapImage из пути

imgPhoto.Source = defaultImage; // Устанавливаем изображение по умолчанию

}

service.MainImagePath = null;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

10) Накопление списка ошибок при сохранении данных:

private void Errors(bool expression, StringBuilder errors, string message)

{

if (expression)

{

errors.AppendLine(message);

}

}

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

{

try

{

StringBuilder errors = new StringBuilder();

if (int.TryParse(TbDiscountServ.Text, out int discount))

{

Errors(discount > 100, errors, "Скидка не может быть больше 100%!");

}

Errors(TbTitleServ.Text == ""

|| TbDurationInSecondsServ.Text == ""

|| TbCostServ.Text == "", errors, "Не заполнена важная ифнормация!");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (TbDiscountServ.Text == "") service.Discount = null;

SaveInDB("Обновление информации о сервисе завершено");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

11) Выбор одного элемента из списка:

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

{

try

{

if (LViewService.SelectedItem != null)

{

ClientService clientService = LViewService.SelectedItem as ClientService;

var service = db.Service.Where(s => s.ID == clientService.ServiceID).First();

if (service != null)

{

EditService dlg = new EditService(service, db, this);

dlg.Show();

}

}

else

{

MessageBox.Show("Выберите сервис для изменения", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

12.1) Выбор нескольких элементов из списка (удаление):

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

{

if (LViewService.SelectedItems.Count > 0)

{

if (MessageBox.Show($"Вы действительно хотите удалить {LViewService.SelectedItems.Count} посещение(ий)?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.Yes)

{

try

{

StringBuilder errors = new StringBuilder();

var selected = LViewService.SelectedItems.Cast<ClientService>().ToArray();

int serviceCount = 0;

foreach (var item in selected)

{

db.ClientService.Remove(db.ClientService.Where(cs => cs.ServiceID == item.ID || cs.ClientID == client.ID).First());

db.SaveChanges();

serviceCount++;

}

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

}

if (serviceCount != 0)

{

MessageBox.Show($"Удалено сервисов: {serviceCount}", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

Load();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

else

{

MessageBox.Show("Выберите сервис для удаления", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

12.2) Выбор нескольких элементов из списка (удаление с зависимостями):

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

{

if (LViewClients.SelectedItems.Count > 0)

{

if (MessageBox.Show($"Вы действительно хотите удалить {LViewClients.SelectedItems.Count} агента(ов)?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.Yes)

{

try

{

StringBuilder errors = new StringBuilder();

var selected = LViewClients.SelectedItems.Cast<Client>().ToArray();

int clientCount = 0;

foreach (var item in selected)

{

if (item.ClientService.Count > 0)

{

errors.AppendLine($"Клиент {item.ID} не может быть удален, т.к. есть информация о посещении");

}

else

{

foreach (Tag tag in item.Tag)

{

db.Tag.Remove(tag);

}

db.Client.Remove(item);

db.SaveChanges();

clientCount++;

}

}

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

}

if (clientCount != 0)

{

MessageBox.Show($"Удалено клиентов: {clientCount}", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

Load();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

else

{

MessageBox.Show("Выберите клиента для удаления", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

13) Добавление надписи, если список пуст:

public void Load()

{

TbClientInfo.Text = $"{client.FirstName} {client.LastName} {client.Patronymic}({client.ID})";

if (client.ServiceList.Count > 0)

{

LViewService.ItemsSource = client.ServiceList;

}

else

{

LViewService.Visibility = Visibility.Hidden;

spServiceInfo.Children.Clear();

TextBlock tb = new TextBlock();

tb.Text = "У данного клиента нет посещений";

tb.FontSize = 22;

tb.HorizontalAlignment = HorizontalAlignment.Center;

tb.VerticalAlignment = VerticalAlignment.Center;

spServiceInfo.Children.Add(tb);

}

}

14) Страница со списком и пагинацией:

<Page x:Class="\_20.\_101\_09\_BeautySalon.Pages.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:\_20.\_101\_09\_BeautySalon.Classes" xmlns:sys="clr-namespace:System;assembly=mscorlib"

mc:Ignorable="d"

d:DesignHeight="650" d:DesignWidth="1000"

Title="ClientPage">

<Page.Resources>

<local:DateTimeToVisibilityConverter x:Key="DateTimeToVisibilityConverter"/>

</Page.Resources>

<Grid Background="White">

<Grid.RowDefinitions>

<RowDefinition Height="1\*" />

<RowDefinition Height="8\*" />

<RowDefinition Height="1\*" />

<RowDefinition Height="1\*" />

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="0.7\*"/>

<ColumnDefinition Width="0.3\*"/>

<ColumnDefinition/>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<CheckBox x:Name="CbBirthDateInMonth" VerticalAlignment="Center" HorizontalAlignment="Center" Content="ДР в текущем месяце" Checked="CbBirthDateInMonth\_Checked" Unchecked="CbBirthDateInMonth\_Unchecked"/>

<ComboBox Name="cmbCount" Grid.Column="1" SelectedIndex="0" VerticalAlignment="Center" Height="22" Margin="5, 0, 5, 0" SelectionChanged="cmbCount\_SelectionChanged" />

<TextBox Grid.Column="2" VerticalAlignment="Center" Margin="5, 0, 5, 0" BorderBrush="Black" Height="22" TextChanged="TextBox\_TextChanged"></TextBox>

<ComboBox x:Name="cmbSorting" Grid.Column="3" SelectedIndex="0" VerticalAlignment="Center" SelectionChanged="cmbSorting\_SelectionChanged" Height="22" Margin="5, 0, 5, 0" />

<ComboBox Name="cmbFilter" Grid.Column="4" SelectedIndex="0" VerticalAlignment="Center" Height="22" Margin="5, 0, 5, 0" SelectionChanged="cmbFilter\_SelectionChanged"/>

</Grid>

<ListView x:Name="LViewClients" Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="4" Margin="5, 5, 5, 5" d:ItemsSource="{d:SampleData ItemCount=1}" MouseDoubleClick="LViewClients\_MouseDoubleClick" SelectionChanged="LViewClients\_SelectionChanged">

<ListView.View>

<GridView>

<GridViewColumn Header="Идентификатор" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding ID}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Пол" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Path=Gender.Name}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Фамилия" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding FirstName}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Имя" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding LastName}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Отчество" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Patronymic}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Дата рождения" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Birthday}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Телефон" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Phone}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Электронная почта" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding Email}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Дата добавления" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding RegistrationDate}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Дата последнего посещения" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding DateService, Converter={local:DateTimeToVisibilityConverter}}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Количество посещений" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<TextBlock Text="{Binding CountService}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

<GridViewColumn Header="Список тегов" Width="Auto">

<GridViewColumn.CellTemplate>

<DataTemplate>

<ItemsControl ItemsSource="{Binding TagList}">

<ItemsControl.ItemTemplate>

<DataTemplate>

<TextBlock Text="{Binding Title, StringFormat='\- {0}'}" HorizontalAlignment="Left" VerticalAlignment="Center" Foreground="{Binding Background}"/>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</DataTemplate>

</GridViewColumn.CellTemplate>

</GridViewColumn>

</GridView>

</ListView.View>

<ListView.ItemContainerStyle>

<Style TargetType="ListViewItem">

<Setter Property="HorizontalContentAlignment" Value="Stretch"/>

</Style>

</ListView.ItemContainerStyle>

<ListView.ContextMenu>

<ContextMenu>

<MenuItem x:Name="BtnOpenServices" Header="Список посещений" Click="BtnOpenServices\_Click" />

</ContextMenu>

</ListView.ContextMenu>

</ListView>

<Grid Grid.Row="2">

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0" Orientation="Horizontal" HorizontalAlignment="Stretch" Margin="15, 0, 5, 0">

<TextBlock Text="Записей: " HorizontalAlignment="Left" VerticalAlignment="Center"></TextBlock>

<TextBlock x:Name="txtDataCount" Text="" HorizontalAlignment="Right" VerticalAlignment="Center"></TextBlock>

<TextBlock Text=" из " HorizontalAlignment="Left" VerticalAlignment="Center"></TextBlock>

<TextBlock x:Name="txtFullDataCount" Text="" HorizontalAlignment="Right" VerticalAlignment="Center"></TextBlock>

</StackPanel>

<StackPanel Grid.Column="2" Orientation="Horizontal" HorizontalAlignment="Right" Width="auto" Margin="5, 5, 5, 5">

<Button x:Name="back" Content="&lt;" Click="back\_Click" IsEnabled="False" Height="30" Width="20" Background="White" BorderBrush="White" Foreground="#FFFF4A6B" Cursor="Hand"></Button>

<StackPanel x:Name="pagin" Orientation="Horizontal" HorizontalAlignment="Stretch" Width="auto" ></StackPanel>

<Button x:Name="forward" Content=">" Click="forward\_Click" Height="30" Width="20" Background="White" BorderBrush="White" Foreground="#FFFF4A6B" Cursor="Hand"></Button>

</StackPanel>

</Grid>

<Grid Grid.Row="3">

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

<Button x:Name="deleteButton" Margin="5" Height="30" BorderBrush="White" Background="#FFFF4A6B" Foreground="White" Cursor="Hand" Width="100" Grid.Column="0" Content="Удалить" Click="deleteButton\_Click" />

<Button x:Name="addButton" Margin="5" Height="30" BorderBrush="White" Background="#FFFF4A6B" Foreground="White" Cursor="Hand" Width="100" Grid.Column="1" Content="Добавить" Click="addButton\_Click" />

<Button x:Name="rebButton" Margin="5" Height="30" BorderBrush="White" Background="#FFFF4A6B" Foreground="White" Cursor="Hand" Width="100" Grid.Column="1" Content="Обновить" Click="rebButton\_Click" />

</StackPanel>

</Grid>

</Grid>

</Page>

public partial class ClientPage : Page

{

Entities db = Helper.GetContext();

int page = 0;

int count;

string fnd = "";

public ClientPage()

{

InitializeComponent();

cmbFilter.ItemsSource = FilterList;

cmbCount.ItemsSource = CountList;

cmbSorting.ItemsSource = SortingList;

Load();

}

public string[] FilterList { get; set; } =

{

"Все",

"Женщины",

"Мужчины"

};

public string[] CountList { get; set; } =

{

"Все",

"10",

"50",

"200"

};

public string[] SortingList { get; set; } =

{

"Без сортировки",

"По возрастанию фамилии",

"По убыванию фамилии",

"По возрастанию даты последнего посещения",

"По убыванию даты последнего посещения",

"По возрастанию посещений",

"По убыванию посещений"

};

public void Load()

{

try

{

var result = db.Client.ToList();

txtFullDataCount.Text = result.Count().ToString();

if (cmbSorting.SelectedIndex == 1) result = result.OrderBy(c => c.FirstName).ToList();

if (cmbSorting.SelectedIndex == 2) result = result.OrderByDescending(c => c.FirstName).ToList();

if (cmbSorting.SelectedIndex == 3) result = result.OrderBy(c => c.DateService).ToList();

if (cmbSorting.SelectedIndex == 4) result = result.OrderByDescending(c => c.DateService).ToList();

if (cmbSorting.SelectedIndex == 5) result = result.OrderBy(c => c.CountService).ToList();

if (cmbSorting.SelectedIndex == 6) result = result.OrderByDescending(c => c.CountService).ToList();

if (cmbFilter.SelectedIndex != 0) result = result.Where(c => c.GenderCode == cmbFilter.SelectedIndex.ToString()).ToList();

if (CbBirthDateInMonth.IsChecked == true) result = result.Where(c => c.Birthday.HasValue && c.Birthday.Value.Month == DateTime.Now.Month).ToList();

result = result.Where(c => c.FirstName.ToLower().Contains(fnd.ToLower())

|| c.LastName.ToLower().Contains(fnd.ToLower())

|| c.Patronymic.ToLower().Contains(fnd.ToLower())

|| c.Email.ToLower().Contains(fnd.ToLower())

|| c.Phone.ToLower().Contains(fnd.ToLower())).ToList();

count = result.Count();

if (cmbCount.SelectedIndex != 0 && cmbCount.SelectedItem != null) result = result.Skip(page \* Int32.Parse(cmbCount.SelectedValue.ToString()))

.Take(Int32.Parse(cmbCount.SelectedValue.ToString())).ToList();

LViewClients.ItemsSource = result;

txtDataCount.Text = count.ToString();

int ost = 0;

int pag = 1;

if (cmbCount.SelectedIndex != 0 && cmbCount.SelectedItem != null)

{

ost = count % Int32.Parse(cmbCount.SelectedValue.ToString());

pag = (count - ost) / Int32.Parse(cmbCount.SelectedValue.ToString());

}

if (ost > 0) pag++;

pagin.Children.Clear();

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

{

Button myButton = new Button();

myButton.Height = 30;

myButton.Content = i + 1;

myButton.Width = 20;

myButton.Background = new SolidColorBrush(Colors.White);

myButton.BorderBrush = new SolidColorBrush(Colors.White);

myButton.Foreground = new SolidColorBrush((Color)ColorConverter.ConvertFromString("#FFFF4A6B"));

myButton.Cursor = Cursors.Hand;

myButton.HorizontalAlignment = HorizontalAlignment.Center;

myButton.Tag = i;

myButton.Click += new RoutedEventHandler(paginButto\_Click);

pagin.Children.Add(myButton);

}

turnButton();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void turnButton()

{

try

{

if (page == 0) { back.IsEnabled = false; }

else { back.IsEnabled = true; };

if (cmbCount.SelectedIndex != 0 && cmbCount.SelectedItem != null)

{

if ((page + 1) \* Int32.Parse(cmbCount.SelectedValue.ToString()) >= count) { forward.IsEnabled = false; }

else { forward.IsEnabled = true; }

}

else { forward.IsEnabled = false; }

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

page = Convert.ToInt32(((Button)sender).Tag.ToString());

Load();

}

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

{

page = 0;

Load();

}

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

{

page = 0;

Load();

}

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

{

page = 0;

fnd = ((TextBox)sender).Text;

Load();

}

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

{

Load();

}

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

{

page = 0;

Load();

}

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

{

}

// Метод для проверки, открыто ли окно указанного типа

public bool IsWindowOpen<T>(string name = "") where T : Window

{

return string.IsNullOrEmpty(name)

? Application.Current.Windows.OfType<T>().Any()

: Application.Current.Windows.OfType<T>().Any(w => w.Name.Equals(name));

}

private void LViewClients\_MouseDoubleClick(object sender, MouseButtonEventArgs e)

{

// Проверяем, открыто ли уже окно AddAndEditClient

if (IsWindowOpen<AddAndEditClient>())

{

MessageBox.Show("Окно Редактирования уже открыто.", "Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

}

else

{

// Окно AddAndEditClient не открыто, открываем его

AddAndEditClient dlg = new AddAndEditClient(LViewClients.SelectedItem as Client, db, this);

dlg.Show();

}

}

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

{

try

{

ClientServiceWindow dlg = new ClientServiceWindow(LViewClients.SelectedItem as Client, db, this);

dlg.Show();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

if (page >= 0)

{

page--;

Load();

}

}

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

{

try

{

if (cmbCount.SelectedIndex != 0 && cmbCount.SelectedItem != null)

{

if (page < db.Client.Count() / Int32.Parse(cmbCount.SelectedValue.ToString()) - 1)

{

page++;

Load();

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

if (LViewClients.SelectedItems.Count > 0)

{

if (MessageBox.Show($"Вы действительно хотите удалить {LViewClients.SelectedItems.Count} агента(ов)?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.Yes)

{

try

{

StringBuilder errors = new StringBuilder();

var selected = LViewClients.SelectedItems.Cast<Client>().ToArray();

int clientCount = 0;

foreach (var item in selected)

{

if (item.ClientService.Count > 0)

{

errors.AppendLine($"Клиент {item.ID} не может быть удален, т.к. есть информация о посещении");

}

else

{

foreach (Tag tag in item.Tag)

{

db.Tag.Remove(tag);

}

db.Client.Remove(item);

db.SaveChanges();

clientCount++;

}

}

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

}

if (clientCount != 0)

{

MessageBox.Show($"Удалено клиентов: {clientCount}", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

Load();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

else

{

MessageBox.Show("Выберите клиента для удаления", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

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

{

AddAndEditClient dlg = new AddAndEditClient(null, db, this);

dlg.Show();

}

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

{

Load();

}

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

{

page = 0;

Load();

}

}

15) Изменение, добавление, удаление:

XAML:

<Window x:Class="\_20.\_101\_09\_BeautySalon.Windows.EditService"

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:\_20.\_101\_09\_BeautySalon.Windows" xmlns:sys="clr-namespace:System;assembly=mscorlib"

mc:Ignorable="d"

Title="Редактирование сервиса" Height="650" Width="1000" MinHeight="650" MinWidth="1000" Icon="/ImagesAndIcons/beauty\_logo.ico" WindowStartupLocation = "CenterScreen" Closed="Window\_Closed">

<Window.Resources>

<sys:String x:Key="defaultImage">

pack://application:,,,/ImagesAndIcons/beauty\_logo.png

</sys:String>

</Window.Resources>

<Grid>

<StackPanel HorizontalAlignment="Center" Margin="10" Orientation="Horizontal">

<StackPanel Orientation="Vertical" Margin="0,0,10,0">

<Image x:Name="imgPhoto" Width="165" Height="165" Source="{Binding ImgPath, FallbackValue={StaticResource defaultImage}}" HorizontalAlignment="Left"/>

<Button Name="btnEnterImage" Width="165" Content="Выбрать фото" VerticalAlignment="Center" HorizontalAlignment="Left" Margin="0,5,0,0" Click="btnEnterImage\_Click"/>

<Button Name="btnDelImage" Width="165" Content="Удалить фото" VerticalAlignment="Center" HorizontalAlignment="Left" Margin="0,0,0,5" Click="btnDelImage\_Click"/>

</StackPanel>

<StackPanel Orientation="Vertical" >

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

<TextBlock Text="ID" Margin="0,0,5,0"/>

<TextBox x:Name="TbIDServ" Width="300" Height="16" Text="{Binding ID}" IsEnabled="False"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,5,0,0">

<TextBlock Text="Наименование" Margin="0,0,5,0"/>

<TextBox x:Name="TbTitleServ" Width="300" Height="16" Text="{Binding Title}" PreviewTextInput="TbTitleServ\_PreviewTextInput"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,5,0,0">

<TextBlock Text="Длительность" Margin="0,0,5,0"/>

<TextBox x:Name="TbDurationInSecondsServ" Width="300" Height="16" Text="{Binding DurationInSeconds}" PreviewTextInput="TbDurationInSecondsServ\_PreviewTextInput"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,5,0,0">

<TextBlock Text="Стоимость" Margin="0,0,5,0"/>

<TextBox x:Name="TbCostServ" Width="300" Height="16" Text="{Binding Cost}" PreviewTextInput="TbCostServ\_PreviewTextInput"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,5,0,0">

<TextBlock Text="Действующая скидка" Margin="0,0,5,0"/>

<TextBox x:Name="TbDiscountServ" Width="300" Height="16" Text="{Binding Discount}" PreviewTextInput="TbDiscountServ\_PreviewTextInput"/>

</StackPanel>

<StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,5,0,0">

<TextBlock Text="Описание" Margin="0,0,5,0"/>

<TextBox x:Name="TbDescriptionServ" Width="300" Height="48" Text="{Binding Description}" AcceptsReturn="True" TextWrapping="Wrap" VerticalScrollBarVisibility="Auto" HorizontalScrollBarVisibility="Auto" PreviewTextInput="TbDescriptionServ\_PreviewTextInput"/>

</StackPanel>

<StackPanel Margin="0,13,0,0">

<Button Name="btnRefrSer" Width="165" Content="Изменить" VerticalAlignment="Center" Margin="0,5,0,0" Click="btnRefrSer\_Click"/>

<Button Name="btnDelSer" Width="165" Content="Удалить" VerticalAlignment="Center" Margin="0,0,0,5" Click="btnDelSer\_Click"/>

</StackPanel>

</StackPanel>

</StackPanel>

</Grid>

</Window>

Sharp:

Entities db = Helper.GetContext();

public partial class EditService : Window

{

Service service = new Service();

Entities db;

ClientServiceWindow clientServiceWindow;

public EditService(Service service, Entities db, ClientServiceWindow clientServiceWindow)

{

InitializeComponent();

this.service = service;

this.db = db;

this.clientServiceWindow = clientServiceWindow;

DataContext = service;

}

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

{

try

{

var asmName = new AssemblyNameDefinition("DynamicAssembly", new Version(1, 0, 0, 0));

var assembly = AssemblyDefinition.CreateAssembly(asmName, "<Module>", ModuleKind.Dll);

OpenFileDialog GetImageDialog = new OpenFileDialog(); // Открытие диалогового окна

string folderpath = System.IO.Directory.GetParent(Environment.CurrentDirectory).Parent.FullName + "\\ImagesAndIcons\\Услуги салона красоты\\";

GetImageDialog.Title = "Выберите изображение";

GetImageDialog.Filter = "Файлы изображений: (\*.png,\*.jpg,\*.jpeg)| \*.png;\*.jpg;\*.jpeg"; // Фильтр типов файлов

GetImageDialog.InitialDirectory = folderpath;

if (GetImageDialog.ShowDialog() == true)

service.MainImagePath = GetImageDialog.SafeFileName;//Добавление наименования файла в БД

var sourse = new BitmapImage(new Uri(GetImageDialog.FileName));

imgPhoto.Source = sourse;

/\*

string filePath = folderpath + System.IO.Path.GetFileName(GetImageDialog.FileName);

File.Copy(GetImageDialog.FileName, filePath, true);

byte[] imageData = File.ReadAllBytes(filePath);

var imageResource = new EmbeddedResource(GetImageDialog.FileName, ManifestResourceAttributes.Private, imageData);

assembly.MainModule.Resources.Add(imageResource);\*/

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

string defaultImagePath = (string)FindResource("defaultImage"); // Получаем путь к изображению из ресурсов

if (!string.IsNullOrEmpty(defaultImagePath))

{

BitmapImage defaultImage = new BitmapImage(new Uri(defaultImagePath)); // Создаем BitmapImage из пути

imgPhoto.Source = defaultImage; // Устанавливаем изображение по умолчанию

}

service.MainImagePath = null;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void Errors(bool expression, StringBuilder errors, string message)

{

if (expression)

{

errors.AppendLine(message);

}

}

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

{

try

{

StringBuilder errors = new StringBuilder();

if (int.TryParse(TbDiscountServ.Text, out int discount))

{

Errors(discount > 100, errors, "Скидка не может быть больше 100%!");

}

Errors(TbTitleServ.Text == ""

|| TbDurationInSecondsServ.Text == ""

|| TbCostServ.Text == "", errors, "Не заполнена важная ифнормация!");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (TbDiscountServ.Text == "") service.Discount = null;

SaveInDB("Обновление информации о сервисе завершено");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void SaveInDB(string text)

{

try

{

db.SaveChanges();

this.Close();

MessageBox.Show(text, "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

db.Service.Remove(service);

SaveInDB("Удаление информации о сервесе завешено!");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void TbTitleServ\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

if (TbTitleServ.Text.Length >= 50 && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = !ValidatorExtensions.IsValidTitle(e.Text);

}

}

private void TbDurationInSecondsServ\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

if (TbDurationInSecondsServ.Text.Length >= 9 && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = !ValidatorExtensions.IsValidCost(e.Text);

}

}

private void TbCostServ\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

//if (TbCostServ.Text.Length >= 11 && !e.Text.Equals("\b"))

//{

// e.Handled = true;

//}

//else

//{

// e.Handled = !ValidatorExtensions.IsValidCost(e.Text);

//}

if (TbCostServ.Text.Contains(".") && e.Text == ".")

{

// Запретить ввод более одной точки.

e.Handled = true;

}

else if (TbCostServ.Text.Length >= 11 && !e.Text.Equals("\b"))

{

// Запретить ввод более 11 символов (если нужно).

e.Handled = true;

}

else

{

// Проверить, что новый текст валиден.

string newText = TbCostServ.Text.Insert(TbCostServ.CaretIndex, e.Text);

e.Handled = !ValidatorExtensions.IsValidCost(newText);

}

}

private void TbDiscountServ\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

//if (TbDiscountServ.Text.Length >= 3 && !e.Text.Equals("\b"))

//{

// e.Handled = true;

//}

//else

//{

// e.Handled = !ValidatorExtensions.IsValidDiscount(e.Text);

//}

if (TbDiscountServ.Text.Length >= 3 && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

string newText = TbDiscountServ.Text.Insert(TbDiscountServ.CaretIndex, e.Text);

// Проверьте, что новый текст является допустимым числом и меньше или равен 100.

if (!ValidatorExtensions.IsValidDiscount(newText) || (int.TryParse(newText, out int discount) && discount > 100))

{

e.Handled = true;

}

}

}

private void TbDescriptionServ\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

if (TbDescriptionServ.Text.Length >= 200 && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = !ValidatorExtensions.IsValidTitle(e.Text);

}

}

private void Window\_Closed(object sender, EventArgs e)

{

try

{

if (clientServiceWindow != null)

{

clientServiceWindow.Load();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

public partial class AddAndEditClient : Window

{

Client client = new Client();

private Entities db;

Client currentClient = new Client();

private ClientPage clientPage;

public AddAndEditClient(Client currentClient, Entities db, ClientPage clientPage)

{

InitializeComponent();

this.clientPage = clientPage;

this.db = db;

if (currentClient != null)

{

this.client = currentClient;

LViewTags.ItemsSource = client.TagList;

tbID.Text = client.ID.ToString();

this.Title = "Редактирование инфомрмации о клиенте";

}

else if (currentClient == null)

{

this.currentClient = currentClient;

btnDeleteClient.IsEnabled = false;

TagGrid.Visibility = Visibility.Collapsed;

spID.Visibility = Visibility.Collapsed;

this.Title = "Добавление инфомрмации о клиенте";

}

DataContext = client;

cmbGender.ItemsSource = db.Gender.ToList();

}

private void AddAndEditClientWindow\_Closed(object sender, EventArgs e)

{

try

{

if (clientPage != null)

{

clientPage.Load(); // Вызываем метод на ClientPage

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

var asmName = new AssemblyNameDefinition("DynamicAssembly", new Version(1, 0, 0, 0));

var assembly = AssemblyDefinition.CreateAssembly(asmName, "<Module>", ModuleKind.Dll);

OpenFileDialog GetImageDialog = new OpenFileDialog(); // Открытие диалогового окна

string folderpath = System.IO.Directory.GetParent(Environment.CurrentDirectory).Parent.FullName + "\\ImagesAndIcons\\Услуги салона красоты\\";

GetImageDialog.Title = "Выберите изображение";

GetImageDialog.Filter = "Файлы изображений: (\*.png,\*.jpg,\*.jpeg)| \*.png;\*.jpg;\*.jpeg"; // Фильтр типов файлов

GetImageDialog.InitialDirectory = folderpath;

if (GetImageDialog.ShowDialog() == true)

client.PhotoPath = GetImageDialog.SafeFileName;//Добавление наименования файла в БД

var sourse = new BitmapImage(new Uri(GetImageDialog.FileName));

imgPhoto.Source = sourse;

/\*

string filePath = folderpath + System.IO.Path.GetFileName(GetImageDialog.FileName);

File.Copy(GetImageDialog.FileName, filePath, true);

byte[] imageData = File.ReadAllBytes(filePath);

var imageResource = new EmbeddedResource(GetImageDialog.FileName, ManifestResourceAttributes.Private, imageData);

assembly.MainModule.Resources.Add(imageResource);\*/

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

string defaultImagePath = (string)FindResource("defaultImage"); // Получаем путь к изображению из ресурсов

if (!string.IsNullOrEmpty(defaultImagePath))

{

BitmapImage defaultImage = new BitmapImage(new Uri(defaultImagePath)); // Создаем BitmapImage из пути

imgPhoto.Source = defaultImage; // Устанавливаем изображение по умолчанию

}

client.PhotoPath = null;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void tbFirstName\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbFirstName, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbLastName\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbLastName, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbPatronymic\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbPatronymic, e, 50, !ValidatorExtensions.IsValidFIO(e.Text));

//e.Handled = !ValidatorExtensions.IsValidFIO(e.Text);

}

private void tbEmail\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

e.Handled = !ValidatorExtensions.IsValidEmailAddress(e.Text);

}

private void CountAndValidText(TextBox textBox, TextCompositionEventArgs e, int count, bool valid)

{

if (textBox.Text.Length >= count && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = valid;

}

}

private void Phone\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbPhone, e, 18, !ValidatorExtensions.IsValidPhone(e.Text));

//e.Handled = !ValidatorExtensions.IsValidPhone(e.Text);

}

private void RefrData()

{

client.Birthday = (DateTime)dpBirthDate.SelectedDate;

client.GenderCode = (cmbGender.SelectedIndex + 1).ToString();

}

private void SaveInDB(string text)

{

db.SaveChanges();

MessageBox.Show(text, "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

this.Close();

}

private void Errors(bool expression, StringBuilder errors, string message)

{

if (expression)

{

errors.AppendLine(message);

}

}

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

{

try

{

StringBuilder errors = new StringBuilder();

Errors(cmbGender.SelectedItem == null, errors, "Выберите пол клиента!");

Errors(ValidatorExtensions.IsValidEmailAddress(tbEmail.Text), errors, "Email адрес не действителен");

Errors(tbFirstName.Text == ""

|| tbLastName.Text == ""

|| tbPhone.Text == "", errors, "Не заполнена важная ифнормация!");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (currentClient == null)

{

client.RegistrationDate = DateTime.Now;

RefrData();

db.Client.Add(client);

SaveInDB("Добавление информации о клиенте завершено");

}

else

{

RefrData();

SaveInDB("Обновление информации о клиенте завершено");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

try

{

if (client.ClientService.Count > 0)

{

MessageBox.Show("Удаление не возможно, т.к. есть информация о посещении", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

return;

}

foreach (Tag tag in client.Tag)

{

db.Tag.Remove(tag);

}

db.Client.Remove(client);

db.SaveChanges();

MessageBox.Show("Удаление информации об клиенте завешено!", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

this.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void tbColorTag\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

//e.Handled = !ValidatorExtensions.IsValidColor(e.Text);

if (tbColorTag.Text.Length >= 6 && !e.Text.Equals("\b"))

{

e.Handled = true;

}

else

{

e.Handled = !ValidatorExtensions.IsValidColor(e.Text);

}

}

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

{

try

{

StringBuilder errors = new StringBuilder();

Errors(tbColorTag.Text == "", errors, "Введите цвет!");

Errors(tbTitleTag.Text == "", errors, "Заголовог тега!");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

Tag tag = new Tag

{

ID = db.Tag.Any() ? db.Tag.Max(t => t.ID) + 1 : 1,

Title = tbTitleTag.Text,

Color = tbColorTag.Text

};

client.Tag.Add(tag);

db.SaveChanges();

tbTitleTag.Text = ""; tbColorTag.Text = "";

LViewTags.ItemsSource = client.TagList;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

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

{

int count = LViewTags.SelectedItems.Count;

if (count > 0)

{

if (MessageBox.Show($"Вы действительно хотите удалить {count} запись(ей)?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Warning) == MessageBoxResult.Yes)

{

try

{

var selected = LViewTags.SelectedItems.Cast<Tag>().ToArray();

foreach (var item in selected)

{

db.Tag.Remove(item);

}

db.SaveChanges();

string about;

if (count == 1) about = "Тег удален!";

else about = "Теги удалены!";

MessageBox.Show(about, "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

LViewTags.ItemsSource = client.TagList;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

else

{

MessageBox.Show("Выберите теги для удаления", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

private void tbTitleTag\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

CountAndValidText(tbTitleTag, e, 50, !ValidatorExtensions.IsValidTitle(e.Text));

}

}

16) MessageBox:

MessageBox.Show("Неверный логин или пароль", "Информация", MessageBoxButton.OK, MessageBoxImage.Information);

MessageBox.Show(ex.Message.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

17) Форматы вывода дат:

<TextBlock Text="{Binding Birthday, StringFormat={}{0:dd.MM.yyyy}}" HorizontalAlignment="Center" VerticalAlignment="Center"/>

<TextBlock Text="{Binding DateService, Converter={local:DateTimeToVisibilityConverter}, StringFormat={}{0:dd.MM.yyyy HH:mm}}" HorizontalAlignment="Center" VerticalAlignment="Center"/>