НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Прикладное программирование

Лабораторная работа №1. "Разработка простого графического приложения WPF на языке C#"

Вариант: 1

Выполнил студент

Вайкус Яков Михайлович

Группа № P3124

Преподаватель: Корж

г. Санкт-Петербург

2022

Оглавление

[**Задание:** 3](#_Toc114565067)

[**Отчет:** 4](#_Toc114565068)

[**Исходный код:** 4](#_Toc114565069)

[**Результат:** 5](#_Toc114565070)

[**Вывод:** 5](#_Toc114565071)

[**Список литературы** 5](#_Toc114565072)

**Задание:**

Разработать простое графическое приложение (WinForms или WPF) на языке C#, в котором должны использоваться основные элементы интерфейса (Label, TextBox, Button, CheckBox, RadioButton, ComboBox, ListBox и др.). Приложение должно демонстрировать умение работать со свойствами элементов, а также реагировать на действия пользователя (обработку событий).

Вариант 1. «Заказ в кафе»

TextBox для ввода имени/ника клиента.

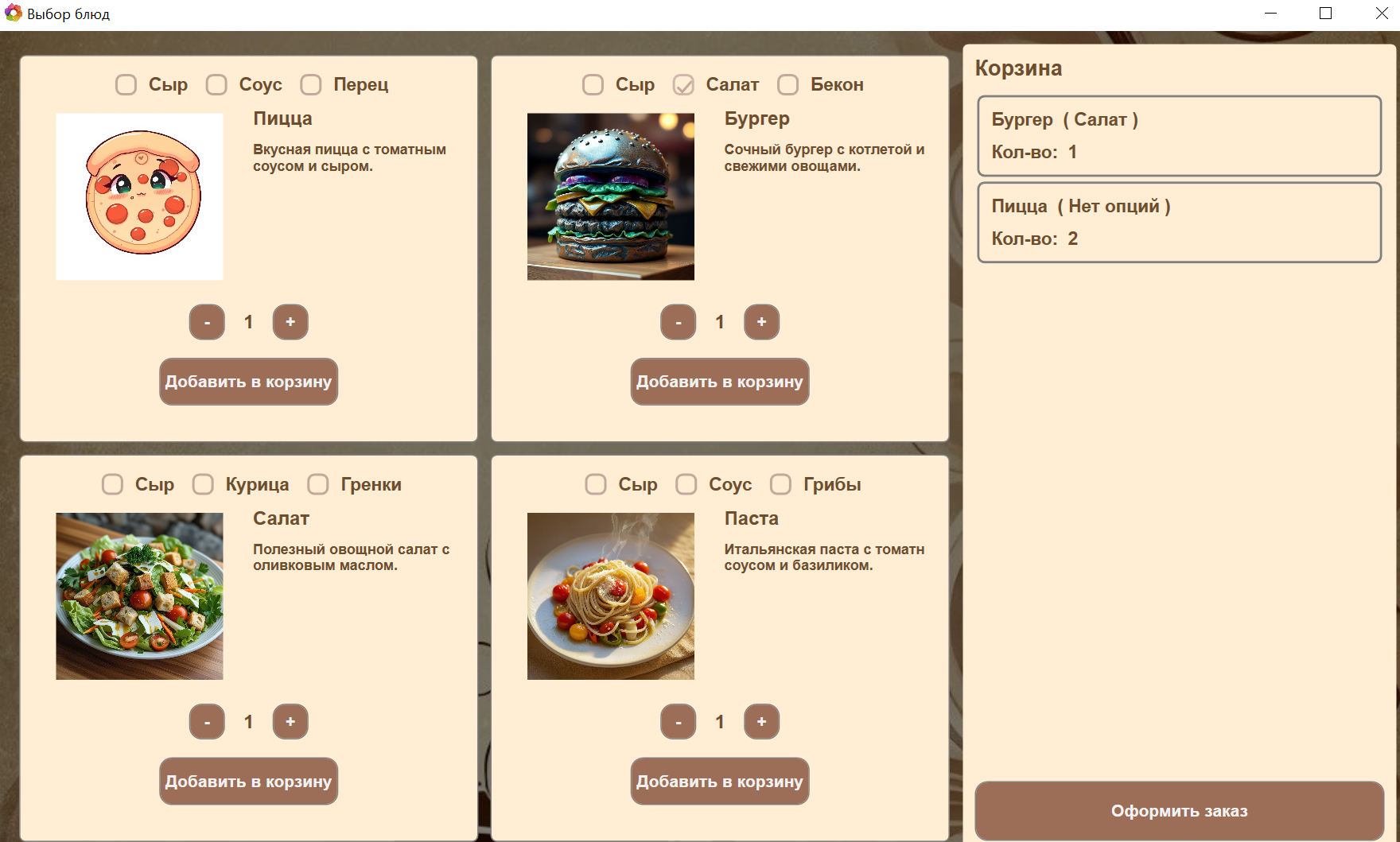
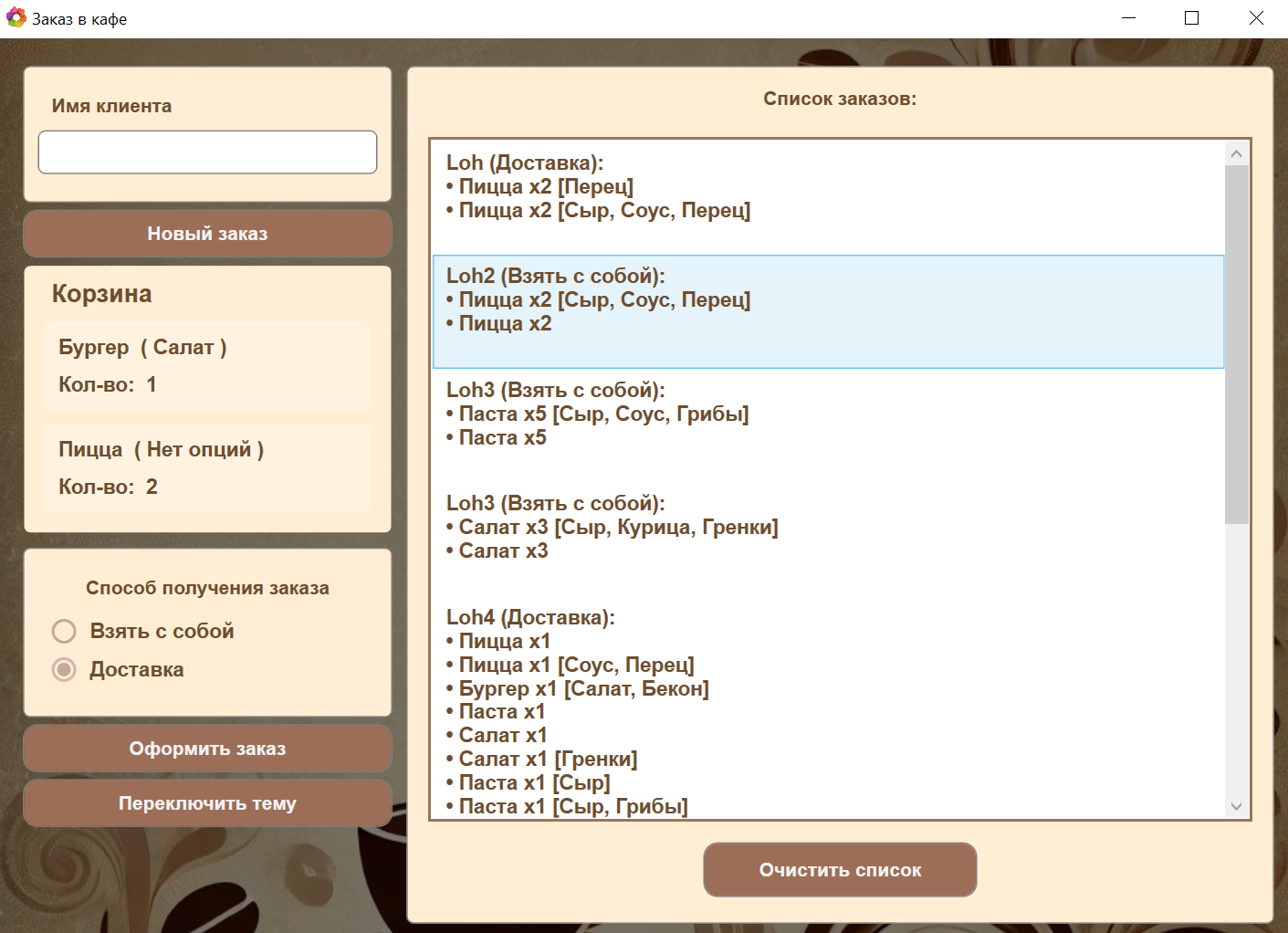
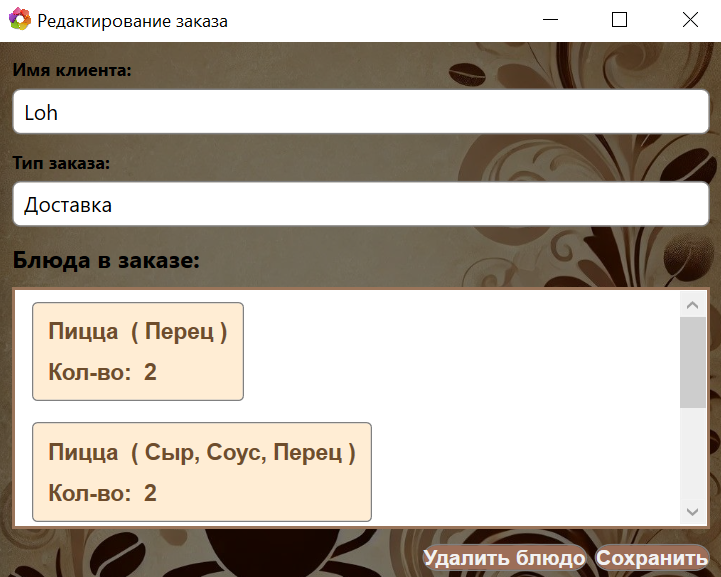
Отдельная форма для выбора основного блюда и дополнительных ингридиентов (пицца, паста, бургер, салат и т.д.).

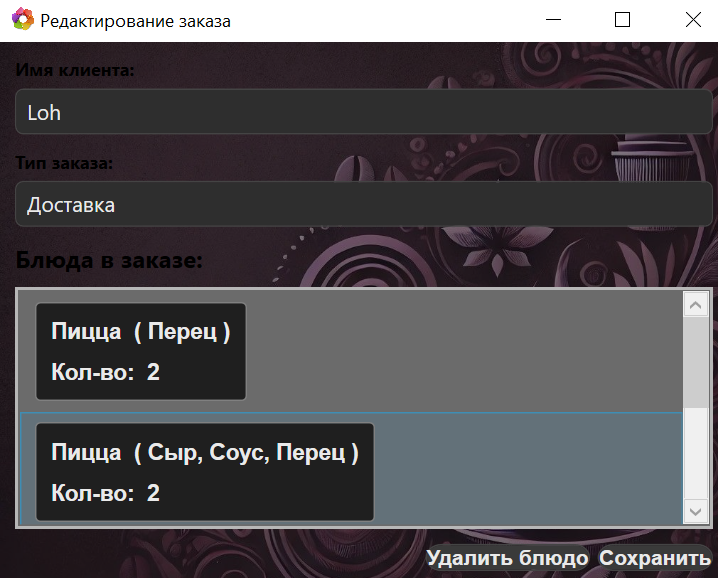
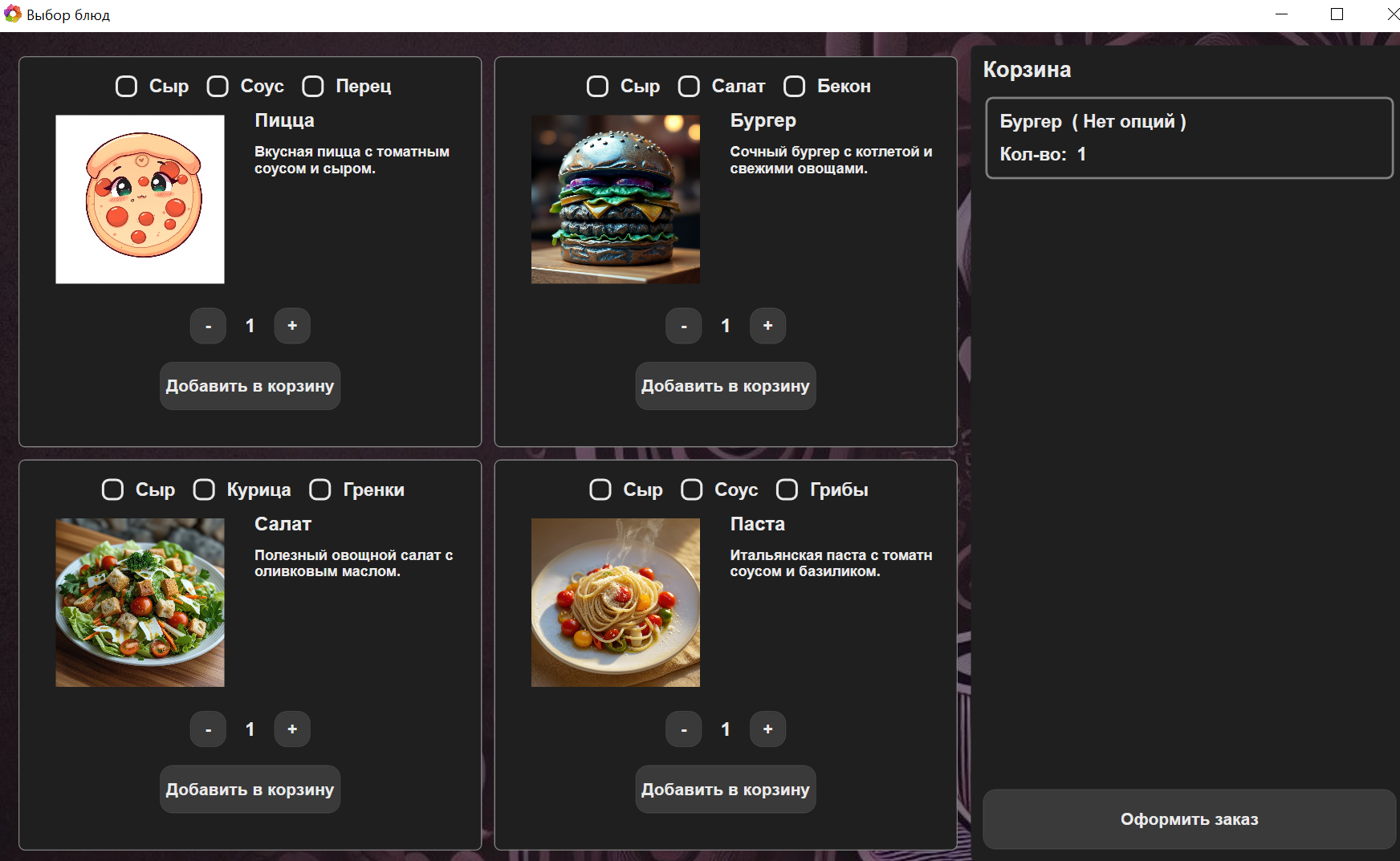
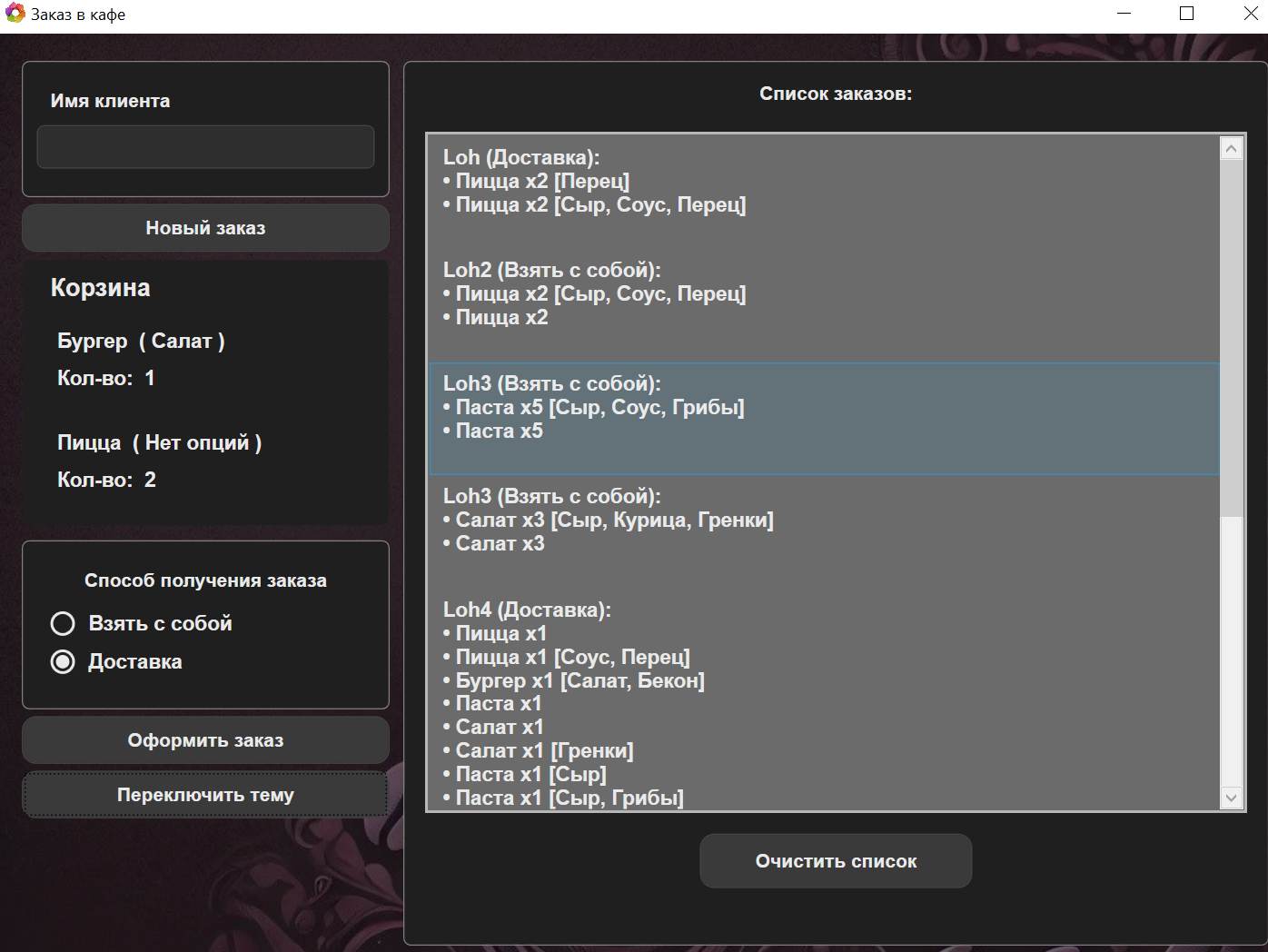
RadioButton для способа получения заказа («Взять с собой» или «Доставка»).

Button «Оформить заказ», по нажатию которого формируется строка-описание и добавляется в ListBox, с возможность редактирования и обновления базы данных.

Label для вывода предупреждений (если имя не введено) и подтверждений.

**Отчет:**

**Приложение в светлой теме :  
  
  
**

**Приложение в темной теме:  
  
  
**

Программа составляет список заказов получая информацию из формы и дополняет список заказов при нажатии кнопки, с возможностью редактирования. Программа взаимодействует с базой данных MSQL и сохраняет все изменения с помощью неё.

# **Исходный код формы:**

<Window x:Class="CafeOrderApp.MainWindow"

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

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

        Title="Заказ в кафе" MinHeight="663" MinWidth="907"

        Icon ="Images\icon.ico"

        Background="{DynamicResource BackgroundBrush}">

    <Grid Margin="10">

        <Grid.ColumnDefinitions>

            <ColumnDefinition Width="280"/> <!-- Левая панель с формами -->

            <ColumnDefinition Width="\*"/> <!-- Список заказов -->

        </Grid.ColumnDefinitions>

        <StackPanel Grid.Column="0">

            <!-- Имя клиента -->

            <Border BorderBrush="Gray" BorderThickness="1" CornerRadius="5" Padding="10" Margin="10,10,0,5">

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

                <Label Content="Имя клиента" HorizontalAlignment="Left"/>

                <TextBox x:Name="NameTextBox" Height="32"/>

            </StackPanel>

            </Border>

            <Button Content="Новый заказ" Click="SelectDishesButton\_Click" Height="35" Width="Auto" Margin="10,0,0,5"/>

            <!-- Корзина -->

<Border x:Name="CartPanel" BorderThickness="1" CornerRadius="5" Padding="10" Grid.Column="1" Visibility="Collapsed">

        <!-- Содержимое корзины с прокруткой -->

        <ScrollViewer VerticalScrollBarVisibility="Auto" MaxHeight="210">

            <StackPanel>

                <TextBlock Text="Корзина" FontSize="18" FontWeight="Bold" Margin="10,0,0,5" Style="{DynamicResource TextBlockInListBoxItemsControl}"/>

                <ItemsControl x:Name="CartItemsControl">

                    <ItemsControl.ItemTemplate>

                        <DataTemplate>

                            <Border Background="#FFF3E0" Margin="5" CornerRadius="5" Padding="5">

                                <StackPanel>

                                    <TextBlock TextWrapping="Wrap" Style="{DynamicResource TextBlockInListBoxItemsControl}">

                                        <Run Text="{Binding Dish.Name}"/>

                                        <Run Text=" ("/>

                                        <Run Text="{Binding OptionsString, Mode=OneWay}"/>

                                        <Run Text=")"/>

                                    </TextBlock>

                                    <TextBlock Style="{DynamicResource TextBlockInListBoxItemsControl}">

                                        <Run Text="Кол-во: "/>

                                        <Run Text="{Binding Quantity}"/>

                                        </TextBlock>

                                </StackPanel>

                            </Border>

                        </DataTemplate>

                    </ItemsControl.ItemTemplate>

                </ItemsControl>

            </StackPanel>

        </ScrollViewer>

</Border>

            <!-- Способ получения -->

            <Border BorderBrush="Gray" BorderThickness="1" CornerRadius="5" Padding="10" Margin="10,5,0,5">

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

                <Label Content="Способ получения заказа"/>

                <RadioButton x:Name="TakeawayRadioButton" Content="Взять с собой" GroupName="DeliveryMethod"/>

                <RadioButton x:Name="DeliveryRadioButton" IsChecked="True" Content="Доставка" GroupName="DeliveryMethod"/>

            </StackPanel>

            </Border>

            <!-- Кнопка "Заказать" -->

            <Button Content="Оформить заказ" Height="35" Width="Auto"

                    HorizontalAlignment="Stretch"

                    Click="PlaceOrderButton\_Click" Margin="10,0,0,5"/>

            <!-- Кнопка переключения темы -->

            <Button Content="Переключить тему" Height="35" Width="Auto" Margin="10,0,0,0"

                    HorizontalAlignment="Stretch"

                    Click="ToggleTheme\_Click"/>

        </StackPanel>

        <!-- Вторая колонка (список заказов) -->

        <Border BorderBrush="Gray" BorderThickness="1" CornerRadius="5" Padding="10" Margin="10,10,0,10" Grid.Column="1">

        <StackPanel Grid.Column="1">

            <Label Content="Список заказов:" Margin="0,0,0,10" FontWeight="Bold"/>

            <ListBox x:Name="OrdersListBox" VerticalAlignment="Stretch" ScrollViewer.VerticalScrollBarVisibility="Auto" MaxHeight="500"

                     HorizontalAlignment="Stretch" Visibility="Collapsed" MouseDoubleClick="OrdersListBox\_DoubleClick">

                    <ListBox.ItemTemplate>

                        <DataTemplate>

                            <TextBlock Style="{DynamicResource TextBlockInListBoxItemsControl}" Text="{Binding FullOrderDetails}" />

                        </DataTemplate>

                    </ListBox.ItemTemplate>

                </ListBox>

            <!-- Кнопка "Очистить список" -->

            <Button Content="Очистить список" Height="40" Width="200"

                    HorizontalAlignment="Center" Margin="0,10,0,0"

                    Click="ClearOrdersButton\_Click"/>

            <!-- Сообщение об ошибке внизу колонки -->

            <Label x:Name="StatusLabel" Foreground="Red"

                   HorizontalAlignment="Left"

                   Visibility="Collapsed"

                   Margin="0,20,0,0"/>

        </StackPanel>

         </Border>

    </Grid>

</Window>

**Исходный код Функций:**

using System.Windows;

using System.Windows.Media.Animation;

using System.Windows.Media.Imaging;

using System.Windows.Media;

using DBManager;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using Lab1;

using System.Windows.Input;

namespace CafeOrderApp

{

    public partial class MainWindow : Window

    {

        private DbManager dbManager;

        private ObservableCollection<OrderItem> selectedDishes = new ObservableCollection<OrderItem>();

        private Order selectedOrder;

        public MainWindow()

        {

            try

    {

        InitializeComponent();

        dbManager = new DbManager("Server=Jacob-book\\SQLEXPRESS;Database=CafeOrderDB;User Id=MyUser4;Password=123;TrustServerCertificate=True;");

        LoadOrders();

    }

    catch (Exception ex)

    {

        MessageBox.Show($"Ошибка при запуске приложения: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

        this.Close();

    }

        }

private void LoadOrders(){

    var orders = dbManager.GetOrders();

    OrdersListBox.ItemsSource = orders;

    OrdersListBox.Items.Refresh();

    OrdersListBox.Visibility = Visibility.Visible;

}

        private string GetSelectedOrderType()

        {

            return TakeawayRadioButton.IsChecked == true ? "Взять с собой" : "Доставка";

        }

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

        {

 try

    {

        dbManager.ClearAllOrders();

        LoadOrders();

        OrdersListBox.Visibility = Visibility.Collapsed;

        StatusLabel.Foreground = Brushes.Green;

        StatusLabel.Content = "Все заказы удалены из базы данных!";

        StatusLabel.Visibility = Visibility.Visible;

        StartFadeOutAnimation();

    }

    catch (Exception ex)

    {

        MessageBox.Show($"Ошибка при очистке базы данных: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

    }

        }

public List<OrderItem> GetSelectedDishes()

{

    return selectedDishes.Select(selectedDish => new OrderItem

    {

        Dish = selectedDish.Dish,  // Теперь передаем сам объект Dish, а не только его название

        Options = GetSelectedOptionsForDish(selectedDish.Dish), // Опции

        Quantity = selectedDish.Quantity

    }).ToList();

}

public List<string> GetSelectedOptionsForDish(Dish dish)

{

    // Здесь должен быть реальный выбор опций, пока заглушка

    return new List<string> { "Option 1", "Option 2" };

}

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

{

    try

    {

        string name = NameTextBox.Text.Trim();

        if (string.IsNullOrEmpty(name))

        {

            StatusLabel.Content = "Пожалуйста, введите имя!";

            StatusLabel.Foreground = Brushes.Red;

            StatusLabel.Visibility = Visibility.Visible;

            StartFadeOutAnimation();

            return;

        }

            string orderType = GetSelectedOrderType(); // Комбо-бокс для типа заказа

            // Отправляем заказ в базу данных

        dbManager.AddOrder(name, orderType, selectedDishes);

            MessageBox.Show("Заказ успешно оформлен!");

        // Обновляем список заказов на форме

        LoadOrders();

        StatusLabel.Content = "Заказ оформлен!";

        StatusLabel.Foreground = Brushes.Green;

        StatusLabel.Visibility = Visibility.Visible;

        StartFadeOutAnimation();

    }

    catch (Exception ex)

    {

        MessageBox.Show($"Произошла ошибка: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

    }

}

        private void StartFadeOutAnimation()

        {

            // Создаем анимацию для исчезновения

            DoubleAnimation fadeOutAnimation = new DoubleAnimation

            {

                From = 1.0,

                To = 0.0,

                Duration = new Duration(TimeSpan.FromSeconds(2))

            };

            // Применяем анимацию к свойству Opacity

            StatusLabel.BeginAnimation(OpacityProperty, fadeOutAnimation);

        }

        private bool isDarkTheme = false;

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

{

    try

    {

        string themeFile = isDarkTheme ? "Styles.xaml" : "DarkTheme.xaml";

        Uri themeUri = new Uri(themeFile, UriKind.Relative);

        ResourceDictionary themeDictionary = new ResourceDictionary { Source = themeUri };

        Application.Current.Resources.MergedDictionaries.Clear();

        Application.Current.Resources.MergedDictionaries.Add(themeDictionary);

        isDarkTheme = !isDarkTheme;

    }

    catch (Exception ex)

    {

        MessageBox.Show($"Произошла ошибка при переключении темы: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

    }

}

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

{

    OrderSelect orderSelect = new OrderSelect(this);

    orderSelect.Owner = this;

    orderSelect.ShowDialog();

}

public void GetDishes(ObservableCollection<OrderItem> dishes)

{

    selectedDishes = dishes;

    CartItemsControl.ItemsSource = null;

    CartItemsControl.ItemsSource = dishes;

    CartPanel.Visibility = Visibility.Visible;

}

        public void RemoveDish(OrderItem item)

        {

            // Если выбранный заказ уже загружен для редактирования

            if (selectedOrder != null && item != null)

            {

                selectedOrder.Items.Remove(item);

            }

            else

            {

                // Если удаляем из корзины до оформления заказа

                if (selectedDishes.Contains(item))

                {

                    selectedDishes.Remove(item);

                }

            }

        }

       private void OrdersListBox\_DoubleClick(object sender, MouseButtonEventArgs e)

{

    if (OrdersListBox.SelectedItem is Order order)

    {

        selectedOrder = order; // сохраняем выбранный заказ для редактирования

        OrderEditWindow editWindow = new OrderEditWindow(order);

        editWindow.Owner = this;

        // После того как окно закрыто и изменения сохранены

        if (editWindow.ShowDialog() == true)

        {

            // Обновляем коллекцию заказов

            //dbManager.UpdateOrder(order);  // Если вы обновляете данные в базе данных

            LoadOrders();  // Перезагружаем список заказов, если это необходимо

            // После сохранения изменений в базе данных или в коллекции, обновляем отображение

            OrdersListBox.Items.Refresh();

        }

    }

}

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

        {

            if (selectedOrder != null)

            {

                // Предположим, что текстовые поля для редактирования существуют: NameTextBox, OrderTypeTextBox

                selectedOrder.ClientName = NameTextBox.Text;

                // Если OrderTypeTextBox существует, иначе используем, например, радиокнопки или ComboBox

                // selectedOrder.OrderType = OrderTypeTextBox.Text;

               // dbManager.UpdateOrder(selectedOrder);

                MessageBox.Show("Изменения сохранены!", "Сохранение", MessageBoxButton.OK, MessageBoxImage.Information);

                LoadOrders();

            }

            else

            {

                MessageBox.Show("Заказ не выбран для редактирования!", "Предупреждение", MessageBoxButton.OK, MessageBoxImage.Warning);

            }

        }

}

    }

**Вторая форма для выбора блюд:**

**<Window x:Class="CafeOrderApp.OrderSelect"**

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

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

**Title="Выбор блюд" Height="800" Width="1200"**

**WindowStartupLocation="CenterOwner"**

**Icon ="Images\icon.ico"**

**Background="{DynamicResource BackgroundBrush}">**

**<Grid Margin="10">**

**<Grid.ColumnDefinitions>**

**<ColumnDefinition Width="800"/>  <!-- Корзина слева -->**

**<ColumnDefinition Width="\*"/>    <!-- Блюда справа -->**

**</Grid.ColumnDefinitions>**

**<!-- Корзина -->**

**<Border x:Name="CartPanel" BorderThickness="1" CornerRadius="5" Padding="10"**

**Grid.Column="1">**

**<Grid>**

**<Grid.RowDefinitions>**

**<RowDefinition Height="\*"/>  <!-- Контент: скроллируемая корзина -->**

**<RowDefinition Height="Auto"/> <!-- Кнопка: фиксирована -->**

**</Grid.RowDefinitions>**

**<!-- Скроллируемая часть -->**

**<ScrollViewer VerticalScrollBarVisibility="Auto" Grid.Row="0">**

**<StackPanel>**

**<TextBlock Text="Корзина" FontSize="18" FontWeight="Bold" Margin="0,0,0,10" Style="{DynamicResource TextBlockInListBoxItemsControl}"/>**

**<ItemsControl x:Name="CartItemsControl">**

**<ItemsControl.ItemTemplate>**

**<DataTemplate>**

**<Border BorderBrush="Gray" BorderThickness="2" CornerRadius="5" Margin="2" Padding="5"**

**>**

**<StackPanel>**

**<TextBlock TextWrapping="Wrap" Style="{DynamicResource TextBlockInListBoxItemsControl}">**

**<Run Text="{Binding Dish.Name}"/>**

**<Run Text=" ("/>**

**<Run Text="{Binding OptionsString, Mode=OneWay}"/>**

**<Run Text=")"/>**

**</TextBlock>**

**<TextBlock TextWrapping="Wrap" Style="{DynamicResource TextBlockInListBoxItemsControl}">**

**<Run Text="Кол-во: "/>**

**<Run Text="{Binding Quantity}"/>**

**</TextBlock>**

**</StackPanel>**

**</Border>**

**</DataTemplate>**

**</ItemsControl.ItemTemplate>**

**</ItemsControl>**

**</StackPanel>**

**</ScrollViewer>**

**<!-- Фиксированная кнопка -->**

**<Button Grid.Row="1" Content="Оформить заказ" Click="ConfirmSelection\_Click" Height="50" Padding="10" Margin="0,10,0,0"/>**

**</Grid>**

**</Border>**

**<!-- Список блюд -->**

**<ScrollViewer Grid.Column="0" VerticalScrollBarVisibility="Hidden">**

**<UniformGrid x:Name="DishesGrid" Columns="2" Margin="5"/>**

**</ScrollViewer>**

**</Grid>**

**</Window>**

**Код второй формы для выбора блюд:  
using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Windows;**

**using System.Windows.Controls;**

**using System.Windows.Input;**

**using System.Windows.Media;**

**using System.Windows.Media.Imaging;**

**using System.Windows.Media.Animation;**

**using System.Collections.ObjectModel;**

**using System.ComponentModel;**

**using System.Runtime.CompilerServices;**

**namespace CafeOrderApp{**

**public partial class OrderSelect : Window**

**{**

**private ObservableCollection<OrderItem> cartItems = new ObservableCollection<OrderItem>();**

**Style style = (Style)Application.Current.FindResource("TextBlockInListBoxItemsControl");**

**private MainWindow mainWindow;**

**public OrderSelect(MainWindow mainWindow)**

**{**

**try**

**{**

**InitializeComponent();**

**this.mainWindow= mainWindow;**

**CartItemsControl.ItemsSource = cartItems;**

**LoadDishes();**

**}**

**catch (Exception ex)**

**{**

**MessageBox.Show($"Ошибка при запуске приложения: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);**

**Application.Current.Shutdown();**

**}**

**}**

**private void LoadDishes()**

**{**

**var dishes = new List<Dish>**

**{**

**new Dish { Name = "Пицца", ImagePath = "Images/pizza.jpg", Description = "Вкусная пицца с томатным соусом и сыром.", Options = new[] { "Сыр", "Соус", "Перец" } },**

**new Dish { Name = "Бургер", ImagePath = "Images/burger.jpg", Description = "Сочный бургер с котлетой и свежими овощами.", Options = new[] { "Сыр", "Салат", "Бекон" } },**

**new Dish { Name = "Салат", ImagePath = "Images/salad.jpg", Description = "Полезный овощной салат с оливковым маслом.", Options = new[] { "Сыр", "Курица", "Гренки" } },**

**new Dish { Name = "Паста", ImagePath = "Images/pasta.jpg", Description = "Итальянская паста с томатным соусом и базиликом.", Options = new[] { "Сыр", "Соус", "Грибы" } }**

**};**

**foreach (var dish in dishes)**

**{**

**var dishBorder = new Border**

**{**

**BorderBrush = Brushes.Gray,**

**BorderThickness = new Thickness(1),**

**CornerRadius = new CornerRadius(5),**

**Padding = new Thickness(10),**

**Margin = new Thickness(5)**

**};**

**var grid = new Grid { Margin = new Thickness(5) };**

**grid.RowDefinitions.Add(new RowDefinition { Height = GridLength.Auto }); // Добавки**

**grid.RowDefinitions.Add(new RowDefinition { Height = GridLength.Auto }); // Картинка + описание**

**grid.RowDefinitions.Add(new RowDefinition { Height = GridLength.Auto }); // Кнопки**

**// Панель с CheckBox для добавок**

**var optionsPanel = new WrapPanel { HorizontalAlignment = HorizontalAlignment.Center };**

**foreach (var option in dish.Options)**

**{**

**optionsPanel.Children.Add(new CheckBox { Content = option, Margin = new Thickness(5, 0, 5, 0) });**

**}**

**Grid.SetRow(optionsPanel, 0);**

**grid.Children.Add(optionsPanel);**

**// Панель с картинкой и описанием**

**var contentPanel = new StackPanel { Orientation = Orientation.Horizontal, Margin = new Thickness(5) };**

**var image = new Image**

**{**

**Source = new BitmapImage(new Uri(dish.ImagePath, UriKind.Relative)),**

**Width = 140,**

**Height = 140,**

**Margin = new Thickness(10)**

**};**

**contentPanel.Children.Add(image);**

**var textPanel = new StackPanel { Margin = new Thickness(10, 0, 0, 0) };**

**textPanel.Children.Add(new TextBlock { Text = dish.Name, FontSize = 16, FontWeight = FontWeights.Bold, Style=style });**

**textPanel.Children.Add(new TextBlock { Text = dish.Description, Style=style, FontSize = 12, TextWrapping = TextWrapping.Wrap, MaxWidth = 200 });**

**contentPanel.Children.Add(textPanel);**

**Grid.SetRow(contentPanel, 1);**

**grid.Children.Add(contentPanel);**

**// Панель для кнопок +/-**

**var buttonPanel = new StackPanel { Orientation = Orientation.Horizontal, HorizontalAlignment = HorizontalAlignment.Center, Margin = new Thickness(5) };**

**var decreaseButton = new Button { Content = "-", Width = 30, Height = 30};**

**var quantityText = new TextBlock { Text = "1", FontSize = 16, Width = 30, Style=style, TextAlignment = TextAlignment.Center, VerticalAlignment = VerticalAlignment.Center };**

**var increaseButton = new Button { Content = "+", Width = 30, Height = 30};**

**int quantity = 1;**

**decreaseButton.Click += (s, e) =>**

**{**

**if (quantity > 1)**

**{**

**quantity--;**

**quantityText.Text = quantity.ToString();**

**}**

**};**

**increaseButton.Click += (s, e) =>**

**{**

**quantity++;**

**quantityText.Text = quantity.ToString();**

**};**

**buttonPanel.Children.Add(decreaseButton);**

**buttonPanel.Children.Add(quantityText);**

**buttonPanel.Children.Add(increaseButton);**

**var addButton = new Button**

**{**

**Content = "Добавить в корзину",**

**Width = 150,**

**Height = 40,**

**Margin = new Thickness(10),**

**HorizontalAlignment = HorizontalAlignment.Center**

**};**

**addButton.Click += (s, e) => AddToCart(dish, optionsPanel, quantity);**

**var buttonStack = new StackPanel { Orientation = Orientation.Vertical };**

**buttonStack.Children.Add(buttonPanel);**

**buttonStack.Children.Add(addButton);**

**Grid.SetRow(buttonStack, 2);**

**grid.Children.Add(buttonStack);**

**dishBorder.Child = grid;**

**DishesGrid.Children.Add(dishBorder);**

**}**

**}**

**private void AddToCart(Dish dish, WrapPanel optionsPanel, int quantity)**

**{**

**// Получаем выбранные опции**

**var selectedOptions = optionsPanel.Children.OfType<CheckBox>()**

**.Where(cb => cb.IsChecked == true)**

**.Select(cb => cb.Content.ToString())**

**.ToList();**

**// Ищем, есть ли уже это блюдо в корзине**

**var existingItem = cartItems.FirstOrDefault(item => item.Dish.Name == dish.Name &&**

**item.Options.SequenceEqual(selectedOptions));**

**if (existingItem != null)**

**{**

**// Если блюдо уже есть в корзине, просто увеличиваем количество**

**existingItem.Quantity += quantity;**

**existingItem.Options = selectedOptions; // Обновляем опции, если они изменились**

**}**

**else**

**{**

**// Добавляем новое блюдо в корзину**

**var orderItem = new OrderItem**

**{**

**Dish = dish,**

**Quantity = quantity,**

**Options = selectedOptions**

**};**

**cartItems.Add(orderItem);**

**}**

**}**

**private void ConfirmSelection\_Click(object sender, RoutedEventArgs e)**

**{**

**MessageBox.Show("Заказ оформлен!");**

**mainWindow.GetDishes(cartItems);**

**Close();**

**}**

**}**

**public class Dish**

**{**

**public string Name { get; set; }**

**public string ImagePath { get; set; }**

**public string Description { get; set; }**

**public string[] Options { get; set; }**

**}**

**public class OrderItem : INotifyPropertyChanged**

**{**

**public Dish Dish { get; set; }**

**private int quantity;**

**public int Quantity**

**{**

**get => quantity;**

**set**

**{**

**quantity = value;**

**OnPropertyChanged();**

**}**

**}**

**private List<string> options;**

**public List<string> Options**

**{**

**get => options;**

**set**

**{**

**options = value;**

**OnPropertyChanged();**

**OnPropertyChanged(nameof(OptionsString)); // Уведомляем и о строке**

**}**

**}**

**public string OptionsString => Options != null && Options.Any() ? string.Join(", ", Options) : "Нет опций";**

**public event PropertyChangedEventHandler PropertyChanged;**

**protected void OnPropertyChanged([CallerMemberName] string name = null)**

**{**

**PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));**

**}**

**}**

**}**

**Форма редактирования заказа:  
<Window x:Class="CafeOrderApp.OrderEditWindow"**

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

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

**Title="Редактирование заказа" Height="400" Width="500"**

**WindowStartupLocation="CenterOwner"**

**Icon ="Images\icon.ico"**

**Background="{DynamicResource BackgroundBrush}">**

**<Grid Margin="10">**

**<Grid.RowDefinitions>**

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

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

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

**</Grid.RowDefinitions>**

**<StackPanel Grid.Row="0">**

**<TextBlock Text="Имя клиента:" FontWeight="Bold"/>**

**<TextBox x:Name="ClientNameTextBox" Text="{Binding Order.ClientName, Mode=TwoWay}" Margin="0,5,0,10"/>**

**<TextBlock Text="Тип заказа:" FontWeight="Bold"/>**

**<TextBox x:Name="OrderTypeTextBox" Text="{Binding Order.OrderType, Mode=TwoWay}" Margin="0,5,0,10"/>**

**</StackPanel>**

**<TextBlock Grid.Row="1" Text="Блюда в заказе:" FontSize="16" FontWeight="Bold" Margin="0,0,0,5"/>**

**<ListBox x:Name="OrderItemsListBox" Grid.Row="1" Margin="0,30,0,0"**

**ItemsSource="{Binding Order.Items}"**

**SelectionMode="Single">**

**<ListBox.ItemTemplate>**

**<DataTemplate>**

**<Border BorderBrush="Gray" BorderThickness="1" CornerRadius="3" Padding="5" Margin="5">**

**<StackPanel>**

**<TextBlock TextWrapping="Wrap" FontWeight="Bold">**

**<Run Text="{Binding Dish.Name}"/>**

**<Run Text=" ("/>**

**<Run Text="{Binding OptionsString, Mode=OneWay}"/>**

**<Run Text=")"/>**

**</TextBlock>**

**<TextBlock TextWrapping="Wrap">**

**<Run Text="Кол-во: "/>**

**<Run Text="{Binding Quantity}"/>**

**</TextBlock>**

**</StackPanel>**

**</Border>**

**</DataTemplate>**

**</ListBox.ItemTemplate>**

**</ListBox>**

**<StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Right" Margin="0,10,0,0">**

**<Button Content="Удалить блюдо" Click="RemoveDish\_Click" Margin="0,0,5,0"/>**

**<Button Content="Сохранить" Click="SaveOrder\_Click"/>**

**</StackPanel>**

**</Grid>**

**</Window>**

**using System.Windows;**

**using DBManager;**

**namespace CafeOrderApp**

**{**

**public partial class OrderEditWindow : Window**

**{**

**// Используем публичное свойство Order для редактируемого заказа**

**public Order Order { get; set; }**

**private DbManager dbManager;**

**public OrderEditWindow(Order order)**

**{**

**InitializeComponent();**

**Order = order;**

**DataContext = this; // Для двусторонней привязки**

**}**

**private void RemoveDish\_Click(object sender, RoutedEventArgs e)**

**{**

**if (OrderItemsListBox.SelectedItem is OrderItem selectedItem)**

**{**

**Order.Items.Remove(selectedItem);**

**OrderItemsListBox.Items.Refresh();**

**}**

**}**

**private void SaveOrder\_Click(object sender, RoutedEventArgs e)**

**{**

**dbManager = new DbManager("Server=Jacob-book\\SQLEXPRESS;Database=CafeOrderDB;User Id=MyUser4;Password=123;TrustServerCertificate=True;");**

**dbManager.UpdateOrder(Order);**

**DialogResult = true;**

**Close();**

**}**

**}**

**}**

**Исходный код Стилей:**

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

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

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

<ImageBrush x:Key="BackgroundBrush" ImageSource="light-theme.png" Stretch ="UniformToFill" Opacity="0.5"/>

<Style TargetType="Border">

<Setter Property=" Background" Value="#ffedd4"/>

</Style>

<Style TargetType="Button">

<Setter Property="Background" Value="#9C6E58"/>

<Setter Property="Foreground" Value="#f2f2f2"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="Padding" Value="10,5"/>

<Setter Property="BorderThickness" Value="0"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Border x:Name="Border"

Background="{TemplateBinding Background}"

CornerRadius="10"

BorderThickness="1"

BorderBrush="Gray">

<ContentPresenter HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</Border>

<ControlTemplate.Triggers>

<!-- Триггер для изменения фона при наведении -->

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="Border" Property="Background" Value="#4322"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Стиль для меток -->

<Style TargetType="Label">

<Setter Property="FontSize" Value="14"/>

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

<Setter Property="Foreground" Value="#6e4d2c"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

<Setter Property="HorizontalAlignment" Value="Center"/>

</Style>

<Style TargetType="TextBox">

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="TextBox">

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}"

CornerRadius="5">

<ScrollViewer x:Name="PART\_ContentHost" />

</Border>

</ControlTemplate>

</Setter.Value>

</Setter>

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

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

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

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

<Setter Property="BorderThickness" Value="1"/>

<Setter Property="FontSize" Value="14"/>

</Style>

<!-- Стиль для ComboBox -->

<Style TargetType="ComboBox">

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

<Setter Property="FontSize" Value="15"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

<Setter Property="BorderBrush" Value="#A18A7F"/>

<Setter Property="Background" Value="#ffc361"/>

<Setter Property="BorderThickness" Value="1"/>

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

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="ComboBox">

<Grid>

<!-- Основной блок -->

<Border Name="Border"

Background="#9C6E58"

BorderBrush="#A18A7F"

BorderThickness="{TemplateBinding BorderThickness}"

CornerRadius="5">

<Grid>

<Grid.ColumnDefinitions>

<!-- Основная область для содержимого -->

<ColumnDefinition Width="\*"/>

<!-- Столбец для стрелки -->

<ColumnDefinition Width="30"/>

</Grid.ColumnDefinitions>

<!-- Текстовое поле для отображения выбранного элемента -->

<ContentPresenter Name="ContentSite"

Grid.Column="0"

Content="{TemplateBinding SelectionBoxItem}"

ContentTemplate="{TemplateBinding SelectionBoxItemTemplate}"

VerticalAlignment="Center"

HorizontalAlignment="Left"

Margin="10,3,30,3"/>

<!-- Кнопка-стрелка -->

<ToggleButton Grid.Column="1"

Name="ToggleButton"

ClickMode="Press"

Background="Transparent"

BorderThickness="0"

IsChecked="{Binding IsDropDownOpen, RelativeSource={RelativeSource TemplatedParent}}">

<Path Data="M 0 0 L 4 4 L 8 0 Z"

Fill="White"

Width="10"

Height="6"

HorizontalAlignment="Center"

VerticalAlignment="Center"/>

</ToggleButton>

</Grid>

</Border>

<!-- Всплывающее окно -->

<Popup Name="Popup"

Placement="Bottom"

IsOpen="{TemplateBinding IsDropDownOpen}"

AllowsTransparency="True"

Focusable="False"

StaysOpen="False">

<Border Background="#7D5B45"

BorderBrush="#C3A89B"

BorderThickness="1"

CornerRadius="10"

Width="{TemplateBinding ActualWidth}">

<ScrollViewer Margin="5">

<StackPanel IsItemsHost="True"/>

</ScrollViewer>

</Border>

</Popup>

</Grid>

<ControlTemplate.Triggers>

<!-- Изменение цвета при наведении -->

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="Border" Property="Background" Value="#83695C"/>

<Setter TargetName="Border" Property="BorderBrush" Value="#C3A89B"/>

</Trigger>

<!-- Изменение цвета при открытии -->

<Trigger Property="IsDropDownOpen" Value="True">

<Setter TargetName="Border" Property="Background" Value="#7D5B45"/>

<Setter TargetName="Border" Property="BorderBrush" Value="#C3A89B"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Стиль для элементов списка -->

<Style TargetType="ComboBoxItem">

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

<Setter Property="Background" Value="Transparent"/>

<Setter Property="Padding" Value="5,2"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="ComboBoxItem">

<Border Name="Border"

Background="{TemplateBinding Background}"

CornerRadius="25"

Padding="{TemplateBinding Padding}">

<ContentPresenter VerticalAlignment="Center" HorizontalAlignment="Left"/>

</Border>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="Border" Property="Background" Value="#83695C"/>

</Trigger>

<Trigger Property="IsSelected" Value="True">

<Setter TargetName="Border" Property="Background" Value="#9D7A6A"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<!-- Стиль для ListBox -->

<Style TargetType="ListBox">

<Setter Property="FontSize" Value="14"/>

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

<Setter Property="BorderBrush" Value="#FF9A755A"/>

<Setter Property="BorderThickness" Value="2"/>

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

</Style>

<!-- Стиль для CheckBox -->

<Style TargetType="CheckBox">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Foreground" Value="#6e4d2c"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="CheckBox">

<StackPanel Orientation="Horizontal">

<!-- Кастомная кнопка -->

<Border Name="CheckBorder"

Width="18" Height="18"

CornerRadius="5"

BorderBrush="#C3A89B"

BorderThickness="2"

Background="Transparent"

Margin="5,0,10,0">

<Path Name="CheckMark"

Data="M2,9 L6,13 L14,4"

Stroke="Transparent"

StrokeThickness="2"/>

</Border>

<!-- Текст -->

<ContentPresenter VerticalAlignment="Center"

HorizontalAlignment="Left"

Content="{TemplateBinding Content}"

TextElement.Foreground="{TemplateBinding Foreground}"/>

</StackPanel>

<ControlTemplate.Triggers>

<!-- Цвет при наведении -->

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="CheckBorder" Property="BorderBrush" Value="#C3A89B"/>

<Setter TargetName="CheckMark" Property="Stroke" Value="#C3A89B"/>

</Trigger>

<!-- Выбранное состояние -->

<Trigger Property="IsChecked" Value="True">

<Setter TargetName="CheckMark" Property="Stroke" Value="#C3A89B"/>

<Setter TargetName="CheckBorder" Property="BorderBrush" Value="#D1B6A9"/>

</Trigger>

<!-- Disabled -->

<Trigger Property="IsEnabled" Value="False">

<Setter TargetName="CheckBorder" Property="Opacity" Value="0.5"/>

<Setter Property="Foreground" Value="#7E6358"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

<Style TargetType="RadioButton">

<Setter Property="FontSize" Value="15"/>

<Setter Property="Foreground" Value="#6e4d2c"/>

<Setter Property="FontWeight" Value="Bold"/>

<Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="RadioButton">

<StackPanel Orientation="Horizontal">

<!-- Кастомная кнопка -->

<Border Name="RadioBorder"

Width="18" Height="18"

CornerRadius="9"

BorderBrush="#C3A89B"

BorderThickness="2"

Background="Transparent"

Margin="5,0,10,0">

<Ellipse Name="RadioDot"

Width="10" Height="10"

Fill="Transparent"/>

</Border>

<!-- Текст -->

<ContentPresenter VerticalAlignment="Center"

HorizontalAlignment="Left"

Content="{TemplateBinding Content}"

TextElement.Foreground="{TemplateBinding Foreground}"/>

</StackPanel>

<ControlTemplate.Triggers>

<!-- Цвет при наведении -->

<Trigger Property="IsMouseOver" Value="True">

<Setter TargetName="RadioBorder" Property="BorderBrush" Value="#C3A89B"/>

<Setter TargetName="RadioDot" Property="Fill" Value="#C3A89B"/>

</Trigger>

<!-- Выбранное состояние -->

<Trigger Property="IsChecked" Value="True">

<Setter TargetName="RadioDot" Property="Fill" Value="#C3A89B"/>

<Setter TargetName="RadioBorder" Property="BorderBrush" Value="#D1B6A9"/>

</Trigger>

<!-- Disabled -->

<Trigger Property="IsEnabled" Value="False">

<Setter TargetName="RadioBorder" Property="Opacity" Value="0.5"/>

<Setter Property="Foreground" Value="#7E6358"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

</ResourceDictionary>

Темная тема:  
  
<ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

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

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

    <!-- Фон и изображения для темной темы -->

    <ImageBrush x:Key="BackgroundBrush" ImageSource="dark-theme.png" Stretch="UniformToFill" Opacity="0.5"/>

    <!-- Стиль для Border -->

    <Style TargetType="Border">

        <Setter Property="Background" Value="#1f1f1f"/>

    </Style>

    <!-- Стиль для Button -->

    <Style TargetType="Button">

        <Setter Property="Background" Value="#3A3A3A"/>

        <Setter Property="Foreground" Value="#EAEAEA"/>

        <Setter Property="FontSize" Value="14"/>

        <Setter Property="Padding" Value="10,5"/>

        <Setter Property="BorderThickness" Value="0"/>

        <Setter Property="FontWeight" Value="Bold"/>

        <Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

        <Setter Property="Template">

            <Setter.Value>

                <ControlTemplate TargetType="Button">

                    <Border x:Name="Border"

                            Background="{TemplateBinding Background}"

                            CornerRadius="10"

                            BorderThickness="1"

                            BorderBrush="#444444">

                        <ContentPresenter HorizontalAlignment="Center"

                                          VerticalAlignment="Center"/>

                    </Border>

                    <ControlTemplate.Triggers>

                        <Trigger Property="IsMouseOver" Value="True">

                            <Setter TargetName="Border" Property="Background" Value="#5C5C5C"/>

                        </Trigger>

                    </ControlTemplate.Triggers>

                </ControlTemplate>

            </Setter.Value>

        </Setter>

    </Style>

    <!-- Стиль для Label -->

    <Style TargetType="Label">

        <Setter Property="FontSize" Value="14"/>

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

        <Setter Property="Foreground" Value="#EAEAEA"/>

        <Setter Property="FontWeight" Value="Bold"/>

        <Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

        <Setter Property="HorizontalAlignment" Value="Center"/>

    </Style>

    <!-- Стиль для TextBox -->

    <Style TargetType="TextBox">

        <Setter Property="Template">

            <Setter.Value>

                <ControlTemplate TargetType="TextBox">

                    <Border Background="{TemplateBinding Background}"

                            BorderBrush="{TemplateBinding BorderBrush}"

                            BorderThickness="{TemplateBinding BorderThickness}"

                            CornerRadius="5">

                        <ScrollViewer x:Name="PART\_ContentHost" />

                    </Border>

                </ControlTemplate>

            </Setter.Value>

        </Setter>

        <Setter Property="Background" Value="#2E2E2E"/>

        <Setter Property="Foreground" Value="#EAEAEA"/>

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

        <Setter Property="BorderBrush" Value="#444444"/>

        <Setter Property="BorderThickness" Value="1"/>

        <Setter Property="FontSize" Value="14"/>

    </Style>

    <Style TargetType="ListBox">

        <Setter Property="FontSize" Value="14"/>

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

        <Setter Property="BorderBrush" Value="#b7b7b7"/>

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

        <Setter Property="BorderThickness" Value="2"/>

        <Setter Property="Background" Value="#6b6b6b"/>

    </Style>

    <!-- Стиль для ComboBox -->

    <Style TargetType="ComboBox">

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

        <Setter Property="FontSize" Value="15"/>

        <Setter Property="FontWeight" Value="Bold"/>

        <Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

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

        <Setter Property="BorderBrush" Value="#5C5C5C"/>

<Setter Property="Background" Value="#3A3A3A"/>

        <Setter Property="BorderThickness" Value="1"/>

        <Setter Property="Margin" Value="0,0,0,5"/>

        <Setter Property="Template">

            <Setter.Value>

                <ControlTemplate TargetType="ComboBox">

                    <Grid>

                        <Border Name="Border"

                                Background="#4C4C4C"

                                BorderBrush="#5C5C5C"

                                BorderThickness="{TemplateBinding BorderThickness}"

                                CornerRadius="5">

                            <Grid>

                                <Grid.ColumnDefinitions>

                                    <ColumnDefinition Width="\*"/>

                                    <ColumnDefinition Width="30"/>

                                </Grid.ColumnDefinitions>

                                <ContentPresenter Name="ContentSite"

                                                  Grid.Column="0"

                                                  Content="{TemplateBinding SelectionBoxItem}"

                                                  ContentTemplate="{TemplateBinding SelectionBoxItemTemplate}"

                                                  VerticalAlignment="Center"

                                                  HorizontalAlignment="Left"

                                                  Margin="10,3,30,3"/>

                                <ToggleButton Grid.Column="1"

                                              Name="ToggleButton"

                                              ClickMode="Press"

                                              Background="Transparent"

                                              BorderThickness="0"

                                              IsChecked="{Binding IsDropDownOpen, RelativeSource={RelativeSource TemplatedParent}}">

                                    <Path Data="M 0 0 L 4 4 L 8 0 Z"

                                          Fill="White"

                                          Width="10"

                                          Height="6"

                                          HorizontalAlignment="Center"

                                          VerticalAlignment="Center"/>

                                </ToggleButton>

                            </Grid>

                        </Border>

                        <Popup Name="Popup"

                               Placement="Bottom"

                               IsOpen="{TemplateBinding IsDropDownOpen}"

                               AllowsTransparency="True"

                               Focusable="False"

                               StaysOpen="False">

                            <Border Background="#333333"

                                    BorderBrush="#5C5C5C"

                                    BorderThickness="1"

                                    CornerRadius="10"

                                    Width="{TemplateBinding ActualWidth}">

                                <ScrollViewer Margin="5">

                                    <StackPanel IsItemsHost="True"/>

                                </ScrollViewer>

                            </Border>

                        </Popup>

                    </Grid>

                    <ControlTemplate.Triggers>

                        <Trigger Property="IsMouseOver" Value="True">

                            <Setter TargetName="Border" Property="Background" Value="#555555"/>

                        </Trigger>

                        <Trigger Property="IsDropDownOpen" Value="True">

                            <Setter TargetName="Border" Property="Background" Value="#666666"/>

                        </Trigger>

                    </ControlTemplate.Triggers>

                </ControlTemplate>

            </Setter.Value>

        </Setter>

    </Style>

<!-- Стиль для ComboBoxItem -->

    <Style TargetType="ComboBoxItem">

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

        <Setter Property="Background" Value="Transparent"/>

        <Setter Property="Padding" Value="5,2"/>

        <Setter Property="Template">

            <Setter.Value>

                <ControlTemplate TargetType="ComboBoxItem">

                    <Border Name="Border"

                            Background="{TemplateBinding Background}"

                            CornerRadius="25"

                            Padding="{TemplateBinding Padding}">

                        <ContentPresenter VerticalAlignment="Center" HorizontalAlignment="Left"/>

                    </Border>

                    <ControlTemplate.Triggers>

                        <Trigger Property="IsMouseOver" Value="True">

                            <Setter TargetName="Border" Property="Background" Value="#5C5C5C"/>

                        </Trigger>

                        <Trigger Property="IsSelected" Value="True">

                            <Setter TargetName="Border" Property="Background" Value="#777777"/>

                        </Trigger>

                    </ControlTemplate.Triggers>

                </ControlTemplate>

            </Setter.Value>

        </Setter>

    </Style>

<!-- Стиль для CheckBox -->

<Style TargetType="CheckBox">

    <Setter Property="FontSize" Value="15"/>

    <Setter Property="FontWeight" Value="Bold"/>

    <Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

    <Setter Property="Foreground" Value="#EAEAEA"/>

    <Setter Property="Template">

        <Setter.Value>

            <ControlTemplate TargetType="CheckBox">

                <StackPanel Orientation="Horizontal">

                    <!-- Кастомная кнопка -->

                    <Border Name="CheckBorder"

                            Width="18" Height="18"

                            CornerRadius="5"

                            BorderBrush="#EAEAEA"

                            BorderThickness="2"

                            Background="Transparent"

                            Margin="5,0,10,0">

                        <Path Name="CheckMark"

                              Data="M2,9 L6,13 L14,4"

                              Stroke="Transparent"

                              StrokeThickness="2"/>

                    </Border>

                    <!-- Текст -->

                    <ContentPresenter VerticalAlignment="Center"

                                      HorizontalAlignment="Left"

                                      Content="{TemplateBinding Content}"

                                      TextElement.Foreground="{TemplateBinding Foreground}"/>

                </StackPanel>

                <ControlTemplate.Triggers>

                    <!-- Цвет при наведении -->

                    <Trigger Property="IsMouseOver" Value="True">

                        <Setter TargetName="CheckBorder" Property="BorderBrush" Value="#EAEAEA"/>

                        <Setter TargetName="CheckMark" Property="Stroke" Value="#EAEAEA"/>

                    </Trigger>

                    <!-- Выбранное состояние -->

                    <Trigger Property="IsChecked" Value="True">

                        <Setter TargetName="CheckMark" Property="Stroke" Value="#EAEAEA"/>

                        <Setter TargetName="CheckBorder" Property="BorderBrush" Value="#EAEAEA"/>

                    </Trigger>

                    <!-- Disabled -->

                    <Trigger Property="IsEnabled" Value="False">

                        <Setter TargetName="CheckBorder" Property="Opacity" Value="0.5"/>

                        <Setter Property="Foreground" Value="#EAEAEA"/>

                    </Trigger>

                </ControlTemplate.Triggers>

            </ControlTemplate>

        </Setter.Value>

    </Setter>

</Style>

<!-- Стиль для RadioButton -->

<Style TargetType="RadioButton">

    <Setter Property="FontSize" Value="15"/>

    <Setter Property="FontWeight" Value="Bold"/>

    <Setter Property="FontFamily" Value="Open Sans, Helvetica Neue, Arial, sans-serif"/>

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

    <Setter Property="Foreground" Value="#EAEAEA"/> <!-- Тот же цвет, что и у лейблов -->

    <Setter Property="Template">

        <Setter.Value>

            <ControlTemplate TargetType="RadioButton">

                <StackPanel Orientation="Horizontal">

                    <!-- Кастомная кнопка -->

                    <Border Name="RadioBorder"

                            Width="18" Height="18"

                            CornerRadius="9"

                            BorderBrush="#EAEAEA"

                            BorderThickness="2"

                            Background="Transparent"

                            Margin="5,0,10,0">

                        <Ellipse Name="RadioDot"

                                 Width="10" Height="10"

                                 Fill="Transparent"/>

                    </Border>

                    <!-- Текст -->

<ContentPresenter VerticalAlignment="Center"

                                      HorizontalAlignment="Left"

                                      Content="{TemplateBinding Content}"

                                      TextElement.Foreground="{TemplateBinding Foreground}"/>

                </StackPanel>

                <ControlTemplate.Triggers>

                    <!-- Цвет при наведении -->

                    <Trigger Property="IsMouseOver" Value="True">

                        <Setter TargetName="RadioBorder" Property="BorderBrush" Value="#EAEAEA"/>

                        <Setter TargetName="RadioDot" Property="Fill" Value="#EAEAEA"/>

                    </Trigger>

                    <!-- Выбранное состояние -->

                    <Trigger Property="IsChecked" Value="True">

                        <Setter TargetName="RadioDot" Property="Fill" Value="#EAEAEA"/>

                        <Setter TargetName="RadioBorder" Property="BorderBrush" Value="#EAEAEA"/>

                    </Trigger>

                    <!-- Disabled -->

                    <Trigger Property="IsEnabled" Value="False">

                        <Setter TargetName="RadioBorder" Property="Opacity" Value="0.5"/>

                        <Setter Property="Foreground" Value="#EAEAEA"/>

                    </Trigger>

                </ControlTemplate.Triggers>

            </ControlTemplate>

        </Setter.Value>

    </Setter>

</Style>

</ResourceDictionary>

# **Вывод:**

В ходе данной лабораторной работы я усовершенствовал графическое приложение WPF на языке C# из прошлой лабораторной работы. Программа взаимодействует с базой данных для хранения, добавления и обновления данных заказов.