**13 Технологии создания графического пользовательского интерфейса**

Задание 1. Система бронирования отеля

Листинг задачи:

using System.Collections.ObjectModel;

using System.ComponentModel;

using System.Runtime.CompilerServices;

using System.Windows;

using System.Windows.Input;

using WpfApp1.Commands;

using WpfApp1.Models;

using WpfApp1.Views;

namespace WpfApp1.ViewModels

{

public class MainWindowViewModel : INotifyPropertyChanged

{

private ObservableCollection<Booking> \_bookings = new ObservableCollection<Booking>();

private Booking? \_selectedBooking;

private string? \_statusMessage;

private int \_nextBookingId = 1;

public ObservableCollection<Booking> Bookings { get => \_bookings; set { SetField(ref \_bookings, value); OnPropertyChanged(nameof(BookingCount)); } }

public Booking? SelectedBooking

{

get => \_selectedBooking;

set

{

if (SetField(ref \_selectedBooking, value))

{

(EditBookingCommand as RelayCommand)?.RaiseCanExecuteChanged();

(CancelBookingCommand as RelayCommand)?.RaiseCanExecuteChanged();

StatusMessage = \_selectedBooking != null ? "Бронь выбрана" : "Готово";

}

}

}

public string? StatusMessage { get => \_statusMessage; set => SetField(ref \_statusMessage, value); }

public int BookingCount => Bookings?.Count ?? 0;

public ICommand BookRoomCommand { get; }

public ICommand EditBookingCommand { get; }

public ICommand CancelBookingCommand { get; }

public ICommand ExitCommand { get; }

public MainWindowViewModel()

{

LoadSampleData();

BookRoomCommand = new RelayCommand(ExecuteBookRoom);

EditBookingCommand = new RelayCommand(ExecuteEditBooking, CanExecuteEditOrCancel);

CancelBookingCommand = new RelayCommand(ExecuteCancelBooking, CanExecuteEditOrCancel);

ExitCommand = new RelayCommand(ExecuteExit);

StatusMessage = "Система готова.";

}

private void ExecuteBookRoom(object? parameter)

{

var dialogWindow = new BookingDialogWindow();

var dialogViewModel = new BookingDialogViewModel(dialogWindow, newBooking =>

{

newBooking.Id = \_nextBookingId++;

Bookings.Add(newBooking);

SelectedBooking = newBooking;

StatusMessage = $"Добавлена новая бронь: №{newBooking.Id}";

OnPropertyChanged(nameof(BookingCount));

});

dialogWindow.DataContext = dialogViewModel;

dialogWindow.Owner = Application.Current.MainWindow;

dialogWindow.ShowDialog();

StatusMessage = "Готово";

}

private void ExecuteEditBooking(object? parameter)

{

if (SelectedBooking == null) return;

var dialogWindow = new BookingDialogWindow();

var originalBooking = SelectedBooking;

var dialogViewModel = new BookingDialogViewModel(dialogWindow, originalBooking, updatedBooking =>

{

StatusMessage = $"Бронь №{originalBooking.Id} изменена.";

});

dialogWindow.DataContext = dialogViewModel;

dialogWindow.Owner = Application.Current.MainWindow;

dialogWindow.ShowDialog();

StatusMessage = "Готово";

}

private void ExecuteCancelBooking(object? parameter)

{

if (SelectedBooking == null) return;

var result = MessageBox.Show($"Вы уверены, что хотите отменить бронь?\n\n{SelectedBooking.BookingSummary}",

"Подтверждение отмены", MessageBoxButton.YesNo, MessageBoxImage.Warning);

if (result == MessageBoxResult.Yes)

{

int cancelledId = SelectedBooking.Id;

Bookings.Remove(SelectedBooking);

OnPropertyChanged(nameof(BookingCount));

StatusMessage = $"Бронь №{cancelledId} отменена.";

}

else

{

StatusMessage = SelectedBooking != null ? $"Отмена брони №{SelectedBooking.Id} прервана." : "Отмена прервана.";

}

}

private void ExecuteExit(object? parameter)

{

Application.Current.Shutdown();

}

private bool CanExecuteEditOrCancel(object? parameter)

{

return SelectedBooking != null;

}

private void LoadSampleData()

{

Bookings.Add(new Booking { Id = \_nextBookingId++, ClientName = "Иванов И.И.", RoomNumber = "101", StartDate = DateTime.Today.AddDays(1), EndDate = DateTime.Today.AddDays(3) });

Bookings.Add(new Booking { Id = \_nextBookingId++, ClientName = "Петрова А.С.", RoomNumber = "205", StartDate = DateTime.Today.AddDays(5), EndDate = DateTime.Today.AddDays(8) });

OnPropertyChanged(nameof(BookingCount));

}

public event PropertyChangedEventHandler? PropertyChanged;

protected virtual void OnPropertyChanged([CallerMemberName] string? propertyName = null) => PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

protected bool SetField<T>(ref T field, T value, [CallerMemberName] string? propertyName = null)

{

if (EqualityComparer<T>.Default.Equals(field, value)) return false;

field = value;

OnPropertyChanged(propertyName);

return true;

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Данные о номерах, ФИО, даты бронирования | Подтверждение о бронировании, Сообщение об ошибке, Обновление списка |

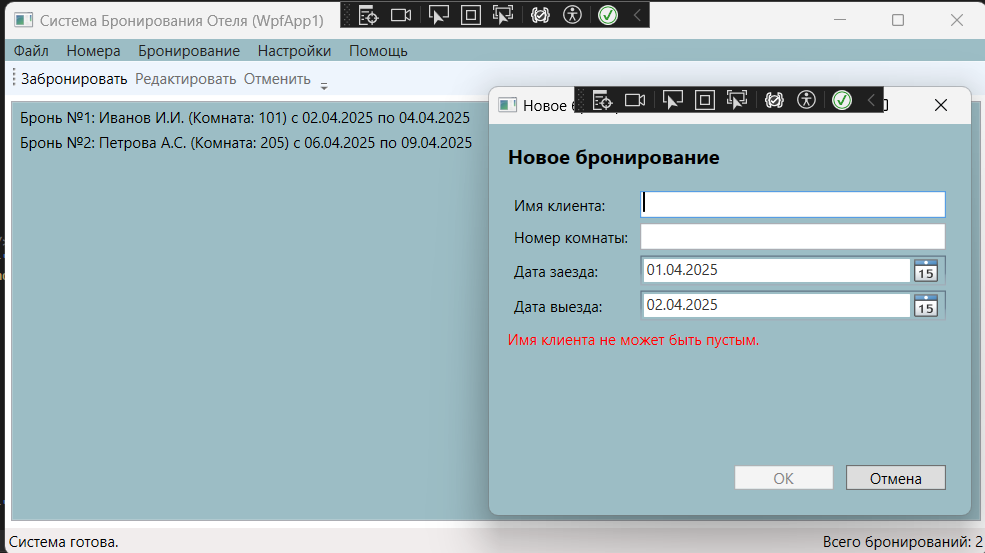
Анализ результатов:

Рисунок 1.1 – Результат работы программы