Министерство образования Республики Беларусь

Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Объектно-ориентированное программирование (ООП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему

«Приложение для управления заказами, сервисом и персоналом отеля на основе платформы .NET с использованием технологии WPF»

Выполнил: студент гр. 653505 Хлопцев А.А.

Руководитель: ассистент кафедры информатики Рогов М.Г.

Минск 2018

**Оглавление**

[Введение 3](#_Toc515309212)

[1.Постановка задачи…………………………………………………………5](#_Toc515309213)

[1.1 Описание предметной области 5](#_Toc515309214)

[1.2 Задачи проектирования 5](#_Toc515309215)

[1.3 Требования к разрабатываемому программному обеспечению 6](#_Toc515309216)

[2. Проектирование приложения 8](#_Toc515309217)

[2.1 Архитектура приложения 8](#_Toc515309218)

[2.2 Используемые технологии 8](#_Toc515309219)

[2.3 Среды реализации задачи 8](#_Toc515309220)

[2.3.1 Microsoft Visual Studio 2017 8](#_Toc515309221)

[2.3.2 .Net Framework 9](#_Toc515309222)

[2.3.3 C# 10](#_Toc515309223)

[2.3.4 XAML 10](#_Toc515309224)

[2.3.5 JSON 11](#_Toc515309225)

[3. Программная реализация 12](#_Toc515309226)

[3.1 Структура приложения 12](#_Toc515309227)

[4. Описания применения 19](#_Toc515309228)

[4.1 USER 19](#_Toc515309229)

[4.2 Client 21](#_Toc515309230)

[4.3 Admin 22](#_Toc515309231)

[Заключение 25](#_Toc515309232)

[Список использованных источников 26](#_Toc515309233)

# Введение

Приложение для управления отелем позволит вам значительно упростить контроль над всеми процессами жизни отеля. Быстро и удобно осуществлять кадровые перестановки, добавление нового сервиса и функционала вашего отеля.

Выбрав различный уровень вашего статуса, вы в полной мере получите всю необходимую для вас информацию.

Пользователь - если вы ещё не пользовались нашим отелем, и решаете, где же остановить на ночь или провести уик-енд это ваш статус. Воспользовавшись данным статусом, вам будет предоставлена вся информация о нашем отеле: виды комнат развлечений, услуги наших инструкторов и тренеров, меню прекраснейшего ресторана французской кухни, а также винная карта, состоящая из более чем 200 видов благородных напитков со всех сторон света. При условии, что наше убранство полностью удовлетворит все ваши прихоти, вам будет доступна страница для резервирования себе места в отеле. На выбор гостей представлено 10 великолепных номеров, оборудованных всем необходимым для комфортного отдыха, ориентированные на разные типы людей.

Гость – статус, который получает человек, зарезервировавший номер в гостинице. Данный тип посетителей получает все привилегии статуса Пользователь + возможность просмотра информации о своём номере, бронировании столика в нашем замечательном ресторане, а также использование дополнительных услуг в соответствии с зарезервированным номером.

Администратор – статус получаемый только верхушкой отеля. Данный вид привилегий позволяет полностью управлять всей жизнью отеля, собирать необходимую информацию и статистику, нанимать/увольнять работников, повышать и понижать в должностях. В дополнении ко всему администраторы утверждают введение-удаление новых услуг и сервиса в гостиницу, корректируют меню ресторана и бара. Им доступно изменение всех данных о гостях и работниках отеля, а также управление заселением-выселением посетителей в случае непредвиденной ситуации.

Данное приложение избавит вас от кучи ручного труда, использования труда третьих лиц, горы документов и бумаг. Сделает более наглядными все основные процессы менеджмента заведения, значительно их ускорит и автоматизирует, что позволит вам получать больше свободного времени, которое вы потратите с удовольствием.

# Постановка задачи

## 1.1 Описание предметной области

* Гости́ница, оте́ль — средство размещения, состоящее из определённого количества номеров, имеющее единое руководство, предоставляющее набор услуг (минимум — заправка постелей, уборка номера и санузла). Гостиницы подразделяются на различные классы в соответствии с предоставляемыми услугами и оборудованием номеров.
* Персонал – специальные люди, нанятые для поддержания функционирования отеля. Каждый член персонала отвечает за свою конкретную область, на которую он был назначен в соответствии с его навыками.
* Клиенты – приобретатель услуг гостиницы. (в виде бронирования номера, заказа столика в ресторане, пользование баром и т.п.)

## 1.2 Задачи проектирования

Чтобы приступить к непосредственному проектированию базы данных, необходимо выделить минимальный набор основных сущностей и указать типы связи между ними.

* Человек (People) – содержит информацию о человеке: ФИО, дату рождения, фотографию и комментарии.
* Гость (Guest) – наследуется от «Человека», дополнительно содержит информацию о комнате, в которую был заселён, дату въезда и выезда из отеля.
* Персонал (Personal) – наследуется от «Человека», дополнительно содержит информацию о зарплате, месте работы и статусе (главный или нет).
* Комната (Room) – содержит информацию о помещении: фотография и этаж расположения.
* Апартаменты (Apartament) – наследуется от «Комната», дополнительно содержит информацию: цена за ночь, номер, кол-во комнат, статус номера, используется ли в данный момент, жилец, список услуг номера.
* Комната развлечений (EntertaimentRoom) – наследуется от «Комната», дополнительно содержит: описание, время работы, рентабельность, рейтинг, стоимость содержания, название, список услуг, перечень персонала, информацию о главном, статус (только для Администратора или нет).

Выделим дополнительные сущности:

* Кухня – наследуется от «Комнаты развлечений»
* Спортивная комната (бассейн и тренажёрный зал) – наследуется от «Комнаты развлечений»
* Комната услуг (бар и ресторан) - наследуется от «Комнаты развлечений»,
* дополнительно включает в себя список меню.

## 1.3 Требования к разрабатываемому программному обеспечению

* Иметь простой и понятный оконный интерфейс (WPF).
* База данных на основе файловой системы в формате .json (Возможность сериализации и десериализации объектов).
* Для пользователя – возможности просмотра информации о комнатах развлечений и их услугах, а также возможность заказа номера.
* Для гостя – возможности пользователя + просмотр информации о своём номере, и возможность использования услуг своих апартаментов.
* Для администратора:

1. возможность управления всеми сущностями отеля
2. добавлении/удалении новых работников
3. выселении жильцов
4. изменение данных о всех людях в отеле
5. добавление новых услуг в комнаты
6. корректирование меню
7. просмотр всей информации об отеле

Таким образом, задача данного курсового проекта сводится к разработке приложения для обеспечения работы отеля. Готовое приложение должно иметь понятный интерфейс и быть удобным в использовании.

# Проектирование приложения

## 2.1 Архитектура приложения

На рис. 1 представлена UML-диаграмма архитектуры данного приложения.

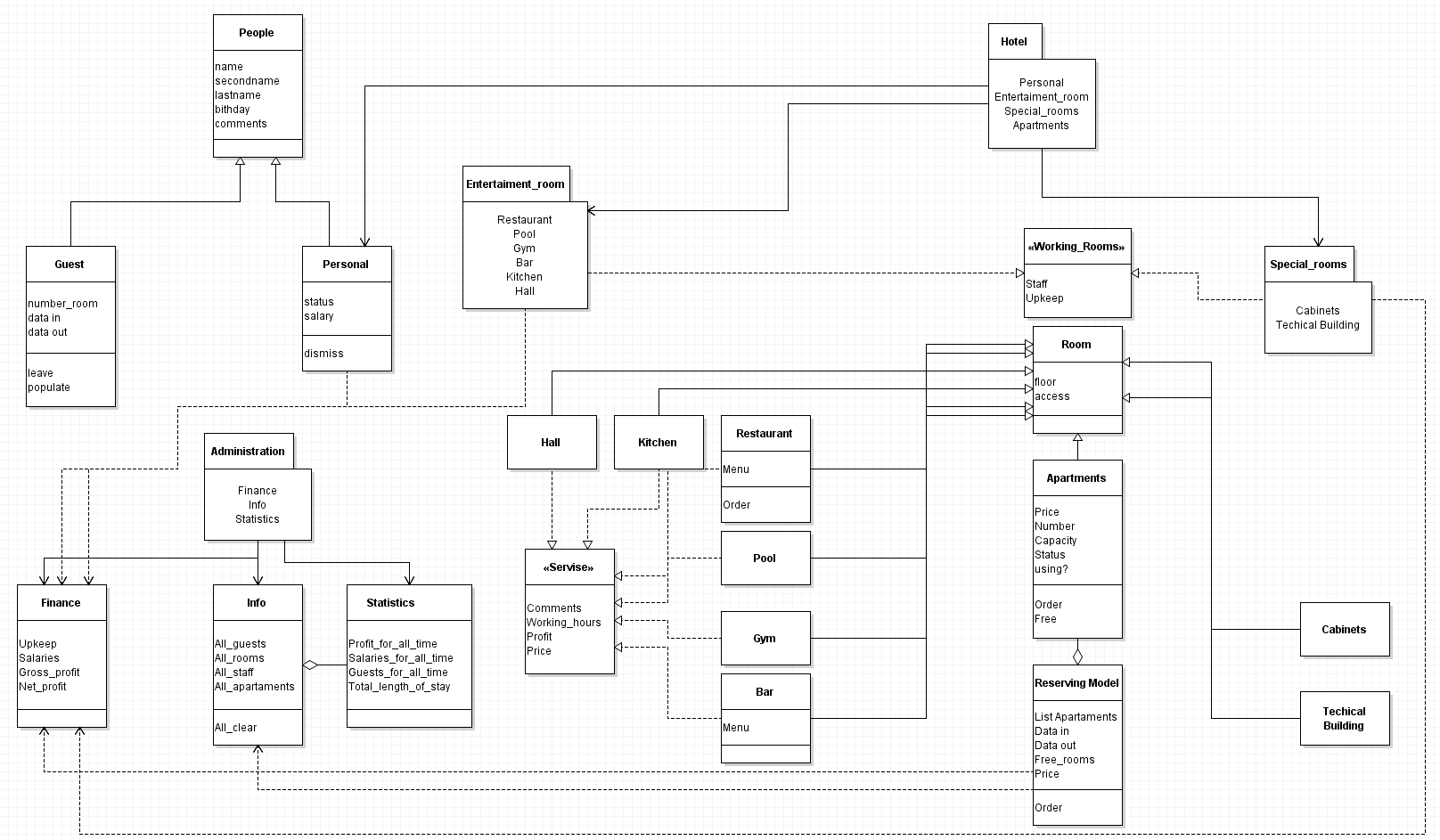


Рис. 1. Архитектура приложения

## 2.2 Используемые технологии

Приложения реализовано на языке C# 7.0 на платформе .Net с использованием языка разметки XAML. В качестве базы данных использовались файлы JSON формата. Среда разработки Microsoft Visual Studio 2017 Community.

## 2.3 Среды реализации задачи

### 2.3.1 Microsoft Visual Studio 2017

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

### 2.3.2 .Net Framework

.NET Framework — программная платформа, выпущенная компанией Microsoft в 2002 году. Основой платформы является общеязыковая среда исполнения Common Language Runtime (CLR), которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду.

Считается, что платформа .NET Framework явилась ответом компании Microsoft на набравшую к тому времени большую популярность платформу Java компании Sun Microsystems (ныне принадлежит Oracle).

Хотя .NET является патентованной технологией корпорации Microsoft и официально рассчитана на работу под операционными системами семейства Microsoft Windows, существуют независимые проекты (прежде всего это Mono и Portable.NET), позволяющие запускать программы .NET на некоторых других операционных системах. В настоящее время .NET Framework получает развитие в виде .NET Core, изначально предполагающей кроссплатформенную разработку и эксплуатацию.

### 2.3.3 C#

C# (произносится си шарп) — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота[4] как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

### XAML

XAML (англ. eXtensible Application Markup Language) — расширяемый язык разметки для приложений (произносится [замл] или [зэмл]) — основанный на XML язык разметки для декларативного программирования приложений, разработанный Microsoft.

Модель приложений Vista включает объект Application. Его набор свойств, методов и событий позволяет объединить веб-документы в связанное приложение. Объект Application контролирует выполнение программы и генерирует события для пользовательского кода. Документы приложения пишутся на XAML. Впрочем, с помощью XAML описывается, прежде всего, пользовательский интерфейс. Логика приложения по-прежнему управляется процедурным кодом (С#, VB, JavaScript и т. д.). XAML может использоваться как для браузер-базированных приложений, так и для настольных приложений.

### JSON

JSON (англ. JavaScript Object Notation, обычно произносится как /ˈdʒeɪsən/ JAY-sən [2]) — текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. Формат JSON был разработан Дугласом Крокфордом[3].

Несмотря на происхождение от JavaScript (точнее, от подмножества языка стандарта ECMA-262 1999 года), формат считается независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON.

# Программная реализация

В данном разделе будет рассмотрена архитектура приложения, а также будут приведены примеры некоторых пользовательских функций и хранимых процедур.

## 3.1 Структура приложения

В основе приложения лежит паттерн MVVM(рис. 2)

Паттерн MVVM (Model-View-ViewModel) позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения.

MVVM состоит из трех компонентов: модели (Model), модели представления (ViewModel) и представления (View).

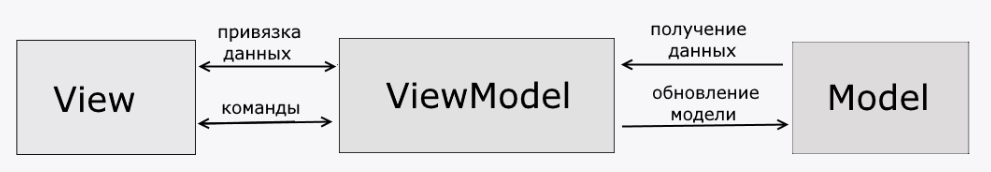


Рис. 2 Паттерн MVVM

**Model**

Модель описывает используемые в приложении данные. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления.

**View**

View или представление определяет визуальный интерфейс, через который пользователь взаимодействует с приложением. Применительно к WPF представление - это код в xaml, который определяет интерфейс в виде кнопок, текстовых полей и прочих визуальных элементов.

**ViewModel**

ViewModel или модель представления связывает модель и представление через механизм привязки данных. Если в модели изменяются значения свойств, при реализации моделью интерфейса INotifyPropertyChanged автоматически идет изменение отображаемых данных в представлении, хотя напрямую модель и представление не связаны.

ViewModel также содержит логику по получению данных из модели, которые потом передаются в представление. И также VewModel определяет логику по обновлению данных в модели.

Поскольку элементы представления, то есть визуальные компоненты типа кнопок, не используют события, то представление взаимодействует с ViewModel посредством команд.

Итогом применения паттерна MVVM является функциональное разделение приложения на три компонента, которые проще разрабатывать и тестировать, а также в дальнейшем модифицировать и поддерживать.

В качестве примера проиллюстрируем функцию добавления нового клиента.

**Model**

Сам класс клиента:

[Serializable]

public abstract class Peoples

{

protected string name;

protected string bithday;

protected string comments;

protected string photo;

}

[Serializable]

public class Guest: Peoples, INotifyPropertyChanged

{

private int number\_room;

private string data\_in;

private string data\_out;

}

Функция добавления клиента:

public void CreateNewGuest(string din, string dout)

{

NewHost.Number\_Room = IsFreeApart.Number;

NewHost.Data\_in = din;

NewHost.Data\_out = dout;

List\_Of\_Order.Add(NewHost);

DateTime start = DateTime.Parse(din);

if (DateTime.Now.Date >= start)

{

IsFreeApart.Host = NewHost;

IsFreeApart.use = true;

}

AddNewHost();

}

**View**

Код окна для добавления клиента:

<Window.Resources>

<ItemsPanelTemplate x:Key="listPanelTemplate">

<WrapPanel />

</ItemsPanelTemplate>

<Style x:Key="FieldInfo" TargetType="TextBlock">

<Setter Property="Margin" Value="0, 3"/>

<Setter Property="FontSize" Value="18"/>

</Style>

<Style x:Key="FieldInput" TargetType="TextBox">

<Setter Property="Margin" Value="30, 5"/>

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

<Setter Property="Width" Value="300"/>

<Style.Triggers>

<Trigger Property="Validation.HasError" Value="True">

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

<Setter Property="ToolTip"

Value="{Binding RelativeSource={RelativeSource Self}, Path=(Validation.Errors)[0].ErrorContent}"/>

</Trigger>

</Style.Triggers>

</Style>

</Window.Resources>

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="150"/>

<ColumnDefinition Width="600"/>

</Grid.ColumnDefinitions>

<Grid Grid.Column = "0">

<Grid.RowDefinitions>

<RowDefinition Height="850"/>

<RowDefinition Height="auto"/>

</Grid.RowDefinitions>

<ListBox x:Name="AllRooms" Grid.Row="0" ItemsSource="{Binding Living\_Rooms}" SelectedItem="{Binding IsApartament}">

<ListBox.ItemTemplate>

<DataTemplate>

<StackPanel>

<Image Source="{Binding Path=Image}" HorizontalAlignment="Center" Width="120" Height="100"/>

<TextBlock Text="{Binding Path=Number, ValidatesOnExceptions=True, StringFormat={}Комната № {0}}"/>

<TextBlock Text="{Binding Path=Status, ValidatesOnExceptions=True, StringFormat={}Статус: {0}}"/>

<TextBlock Text="{Binding Path=Price, ValidatesOnExceptions=True, StringFormat={}Цена: {0}$}"/>

<TextBlock/>

</StackPanel>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

<Button Grid.Row="1" Height="50" Width="50" Margin="50,10,50,0" Content="!" FontSize="30" Click="InfoAboutApartment" VerticalAlignment="Top" Background="PaleGoldenrod"/>

</Grid>

<Grid Grid.Column="1">

<Grid.RowDefinitions>

<RowDefinition Height="20"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="20"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="auto"/>

<RowDefinition Height="auto"/>

</Grid.RowDefinitions>

<TextBlock Grid.Row="0" Text="Выберите дату" HorizontalAlignment="Center"/>

<Grid Grid.Row="1">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="260"/>

<ColumnDefinition Width="260"/>

<ColumnDefinition Width="70"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="20"/>

<RowDefinition Height="auto"/>

</Grid.RowDefinitions>

<DatePicker x:Name="StartDate" Grid.Row="1" Grid.Column="0"/>

<DatePicker x:Name="EndDate" Grid.Row="1" Grid.Column="1"/>

<Button Content="Find" Grid.Row="1" Grid.Column="2" Margin="10,0,1,0" Width="60" Click="Find" Background="PaleGoldenrod" Height="24" VerticalAlignment="Bottom"/>

</Grid>

<TextBlock Text="Выберите пустой номер на эти дни" Grid.Row="2" HorizontalAlignment="Center"/>

<ListBox Grid.Row="3" ItemsSource="{Binding Free\_Rooms, UpdateSourceTrigger=PropertyChanged}" ItemsPanel="{StaticResource listPanelTemplate}" SelectedItem="{Binding IsFreeApart, UpdateSourceTrigger=PropertyChanged, Mode=TwoWay}" HorizontalAlignment="Center">

<ListBox.ItemContainerStyle>

<Style>

<Setter Property="ListBoxItem.IsEnabled" Value="True"/>

<Style.Triggers>

<Trigger Property="ListBoxItem.IsEnabled" Value="True">

<Setter Property="ListBoxItem.Background" Value="Black"/>

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

<Setter Property="ListBoxItem.Margin" Value="0.5"/>

</Trigger>

</Style.Triggers>

</Style>

</ListBox.ItemContainerStyle>

<ListBox.ItemTemplate>

<DataTemplate>

<Canvas Height="40" Width="40">

<TextBlock Text="{Binding Path=Number}" Canvas.Top="11" Canvas.Left="10"/>

</Canvas>

</DataTemplate>

</ListBox.ItemTemplate>

</ListBox>

<Grid Grid.Row="4">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="150"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<StackPanel Grid.Column="0">

<TextBlock Text="ФИО:" Style="{StaticResource FieldInfo}"/>

<TextBlock Text="Фото:" Style="{StaticResource FieldInfo}"/>

<TextBlock Text="Дата рождения:" Style="{StaticResource FieldInfo}"/>

<!--<TextBlock Text="Фото:" Style="{StaticResource FieldInfo}"/>-->

</StackPanel>

<StackPanel Grid.Column="1" DataContext="{Binding NewHost, Mode=TwoWay}">

<TextBox x:Name="Name" Style="{StaticResource FieldInput}">

<Binding Path="Name" NotifyOnValidationError="True" UpdateSourceTrigger="PropertyChanged" Mode="TwoWay">

<Binding.ValidationRules>

<local:RulesForName/>

<ExceptionValidationRule/>

</Binding.ValidationRules>

</Binding>

</TextBox>

<TextBox x:Name="Photo" Style="{StaticResource FieldInput}">

<Binding Path="Photo" NotifyOnValidationError="True" UpdateSourceTrigger="PropertyChanged" Mode="TwoWay"/>

</TextBox>

<TextBox x:Name="Date" Style="{StaticResource FieldInput}">

<Binding Path="Bithday" NotifyOnValidationError="True" UpdateSourceTrigger="PropertyChanged" Mode="TwoWay">

<Binding.ValidationRules>

<local:RulesForDate/>

<ExceptionValidationRule/>

</Binding.ValidationRules>

</Binding>

</TextBox>

</StackPanel>

</Grid>

<StackPanel Grid.Row="5">

<Grid DataContext="{Binding IsFreeApart, UpdateSourceTrigger=PropertyChanged}">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="150"/>

<ColumnDefinition Width="auto"/>

</Grid.ColumnDefinitions>

<TextBlock Text="Стоимость:" Style="{StaticResource FieldInfo}" Grid.Column="0"/>

<TextBlock Text="{Binding Path=Price, UpdateSourceTrigger=PropertyChanged}" Style="{StaticResource FieldInfo}" Grid.Column="1"/>

</Grid>

<TextBlock Text="Что мы должны знать о вас?" HorizontalAlignment="Center" />

<TextBox x:Name="Comments" TextWrapping="Wrap" Text="{Binding NewHost.Comments, UpdateSourceTrigger=PropertyChanged}"/>

<Button Height="40" Width="100" Margin="10" HorizontalAlignment="Center" Click="AddNewGuest" Content="Добавить" FontSize="16" Background="PaleGoldenrod"/>

</StackPanel>

</Grid>

</Grid>

</Window>

**ViewModel**

Элементы связывания Model и View

public string Name

{

get

{

return name;

}

set

{

name = value;

OnPropertyChanged("Name");

}

}

public int Number\_Room

{

get

{

return number\_room;

}

set

{

number\_room = value;

OnPropertyChanged("Number\_Room");

}

}

public string Bithday

{

get

{

return bithday;

}

set

{

bithday = value;

OnPropertyChanged("Bithday");

}

}

public string Comments

{

get

{

return comments;

}

set

{

comments = value;

OnPropertyChanged("Comments");

}

}

public string Photo

{

get

{

return photo;

}

set

{

photo = value;

OnPropertyChanged("Photo");

}

}

public string Data\_in

{

get

{

return data\_in;

}

set

{

data\_in = value;

OnPropertyChanged("Data\_in");

}

}

public string Data\_out

{

get

{

return data\_out;

}

set

{

data\_out = value;

OnPropertyChanged("Data\_out");

}

}

public Guest(string name, int number, string date, string dout, string din, string photo, string comments)

{

Name = name;

Number\_Room = number;

Bithday = date;

Data\_in = din;

Data\_out = dout;

Photo = photo;

Comments = comments;

}

public event PropertyChangedEventHandler PropertyChanged;

public void OnPropertyChanged([CallerMemberName]string prop = "")

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(prop));

}

# Описания применения

Данное приложение содержит 3 основных окна, для каждого типа пользователей (рис. 3)

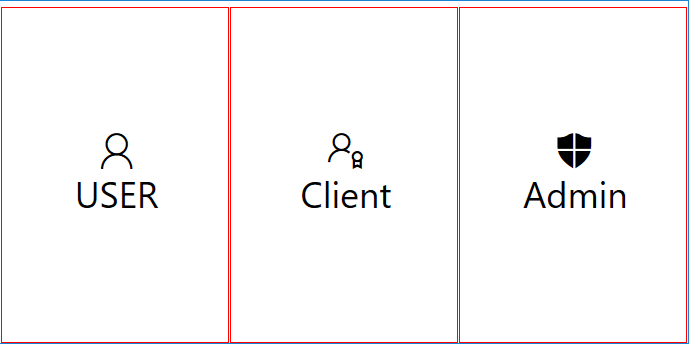


Рис. 3. Окно перехода между пользователями

USER, Clint, Admin – покажем пример работы каждого из них

## 4.1 USER

При нажатии на вкладку USER мы попадаем на окно информации об отеле (рис. 4). Через данное окно можно перейти к любой выбранной вами комнате, и прочесть доступную о ней информацию (прим. рис. 5)

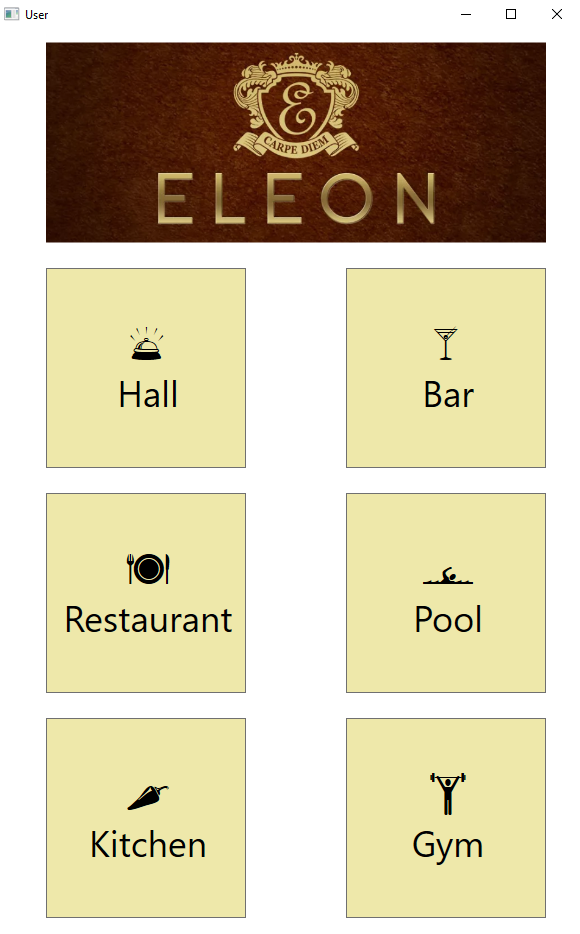


Рис. 4. Панель пользователя



Рис. 5. Пример вывода информации об ресторане

## 4.2 Client

При нажатии на вкладку клиент мы попадаем в окно выбора действий (рис.6)

В данном окне можно либо перейти к окну заказа (рис. 7), либо посмотреть информацию о своём номере (рис. 8)

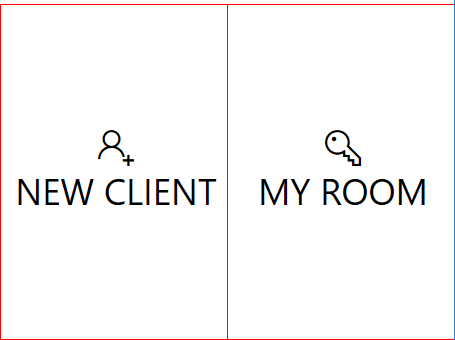


Рис. 6. Окно выбора клиента

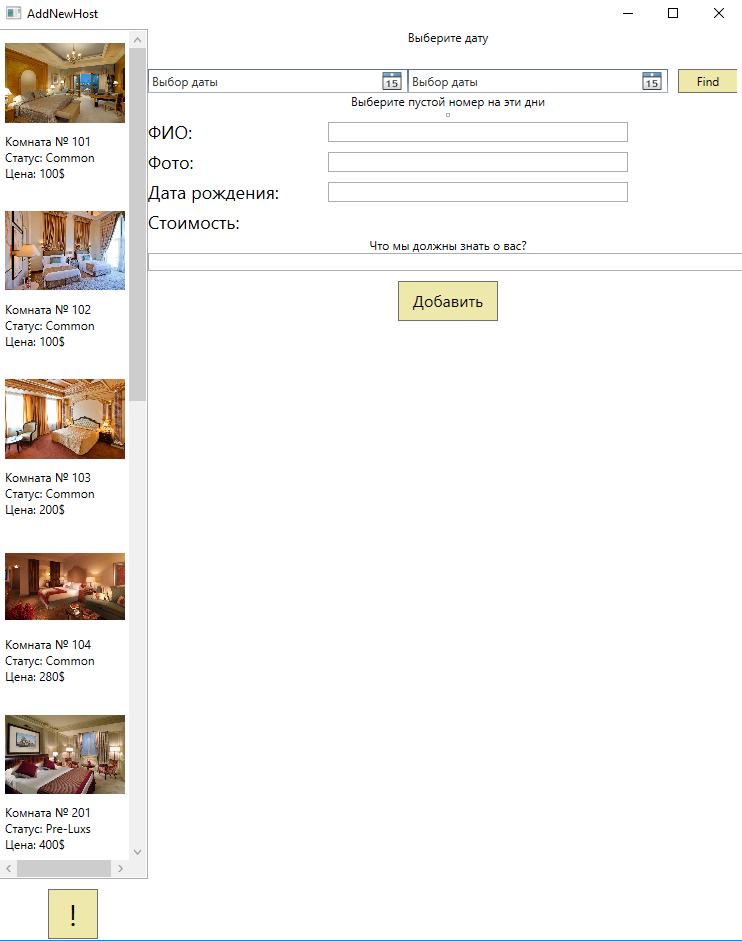


Рис. 7. Окно бронирования номера

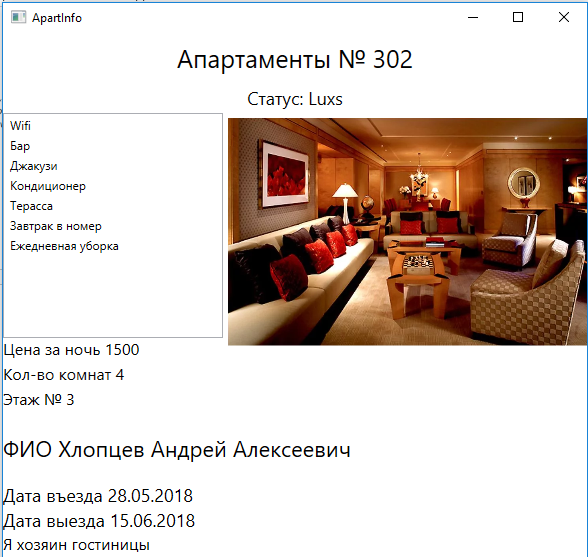


Рис. 8. Окно информации о своём номере

## Admin

При нажатии на вкладку Admin и успешного выполнения аутентификации администратора (рис. 9), мы попадаем в окно администрирования (рис. 10).

Здесь вы можете просмотреть информацию обо всех комнатах в отеле (прим. рис 11), всех гостях и всём персонале, и, при желании, редактировать их данные (прим. рис 12)

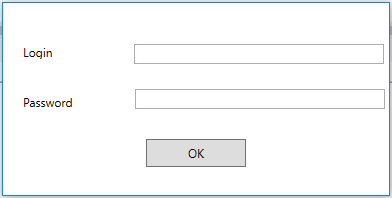


Рис. 9. Окно аутентификации администратора

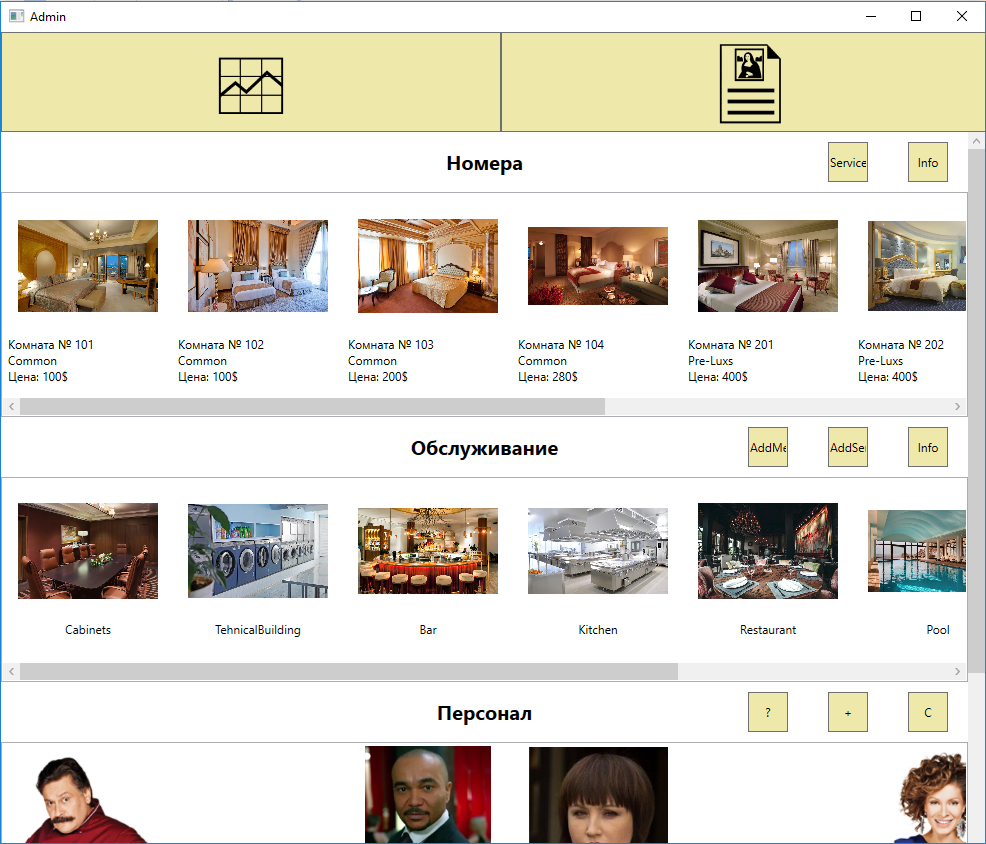


Рис. 10. Окно администрации



Рис. 11. Расширенное окно информации о кухне

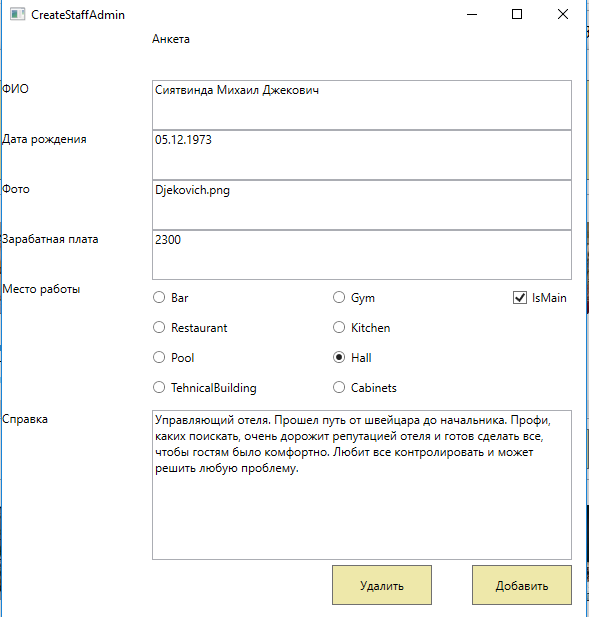


Рис. 12. Окно изменения параметров персонала

# Заключение

В рамках работы над курсовым проектом был разработан программный продукт для управления работой отеля организации.

За время написания проекта была изучена информация о принципах работы отелей и подходы решения основных задач, возлагаемых на администрацию гостиниц.

В плане разработки приложения были углублены и закреплены знания по разработке веб-приложений на платформе .NET и языку C#.

Были проанализированы различные подходы и технологии, из которых были выбраны те, которые автор хотела бы видеть в своем программном продукте.

Разработанное программное средство представляет собой законченный программный продукт, готовый к использованию. Но при желании приложение можно доработать: расширить функциональность приложения, изменить дизайн и др.

# Список использованных источников

1. Фленов М.Е. Библия C#. – 3-e издание, перераб. и доп. – СПб:: БХВ-Петербург, 2016. – 554 с.:ил.
2. <https://metanit.com> – помощь по MVVM, C# и XAML
3. <https://ru.wikipedia.org> – получение дополнительной вспомогательной информации
4. <http://hotel-eleon-serial.net> – изображения и информация для объектов работы