

Частное учреждение образования  
«Колледж бизнеса и права»

ДОПУЩЕН К ЗАЩИТЕ  
Заместитель директора  
по учебной работе  
\_\_\_\_\_ И.В.Малафеев  
«\_\_» \_\_\_\_\_ 2023

СОЗДАНИЕ ПРОГРАММНОГО СРЕДСТВА ДЛЯ АВТОМАТИЗАЦИИ  
РАБОТЫ АДМИНИСТРАТОРА ГОСТИНИЦЫ «БОНА»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ДП Т.993144.401

Председатель цикловой комиссии	(Т.Г. Багласова )
Руководитель проекта	(Е.Н. Коропа)
Консультант по экономической части	(Е.А. Андреева )
Консультант по охране труда	(В.С. Кудласевич)
Обучающийся	(А.А.Яроцкий)
Рецензент	( )

2023

## СОДЕРЖАНИЕ

Введение	4
1 Описание задачи	6
1.1 Анализ предметной области	6
1.2 Постановка задачи	6
2 Проектирование системы	8
2.1 Требования к приложению	8
2.2 Проектирование модели	8
2.3 Проектирование структуры базы данных	10
2.4 Концептуальный прототип	12
3 Описание реализации программного средства	16
3.1 Инструменты разработки и применяемые технологии	16
3.2 Организация данных	17
3.3 Функции: логическая и физическая организации	19
3.4 Входные и выходные данные	23
3.5 Функциональное тестирование	24
3.6 Описание справочной системы	26
4 Применение	28
4.1 Назначение программного средства	28
4.2 Условия применения	28
5 Охрана труда и окружающей среды	29
5.1 Правовые, нормативные, социально-экономические и организационные вопросы охраны труда	29
5.2 Предупреждение пожаров на производстве	30
5.3 Обеспечение пожарной безопасности	32
5.4 Охрана окружающей среды	35
6 Экономический раздел	37
6.1 Техничко-экономическое обоснование разработки программного средства	37
6.2 Составление плана по разработке программного средства	37
6.3 Расчет затрат на разработку программного продукта	38
6.4 Экономическая эффективность разработки программного продукта	44
6.5 Экономическая эффективность у пользователя программного продукта	45
6.6 Экономическая эффективность разработки	47
Заключение	48
Список использованных источников	49
ПРИЛОЖЕНИЕ А	50
ПРИЛОЖЕНИЕ Б	67

					ДП Т.993144.401ПЗ				
Изм.	Лист	№ докум.	Подпись	Дата					
Разраб.		Яроцкий А.А			Создание программного средства для автоматизации работы администратора гостиницы «Бона»	Лит.	Лист	Листов	
Провер.		Коропа Е.Н				у	3	69	
Т. контр.		Коропа Е.Н				КБП 3			
Н. контр.		Багласова Е.В.							
Утверд.		Багласова Т.Г.							

## **Введение**

Один из наиболее важных аспектов работы отеля – это прием, размещение и обслуживание гостей. Это включает работу персонала, который занимается бронированием номеров, приемом гостей, проведением регистрации, предоставлением дополнительных услуг в виде питания и услуг отеля. Важным аспектом является также уровень комфорта, который предоставляют номера и общественные зоны отеля. Гостиничный комплекс должен обеспечивать высокий уровень сервиса и чистоты, а также соответствовать стандартам безопасности.

Управление и эксплуатация отеля также является важным аспектом предметной области. Отель должен иметь эффективную систему управления запасами и ресурсами, а также гибкую систему расписания работы персонала и учета финансовых потоков. Это также включает в себя управление инфраструктурой отеля, такими как здания, оборудование, коммуникации, техническое обслуживание и уборка.

Также важным аспектом предметной области отеля является маркетинг. Это включает в себя разработку ценовой политики, рекламную кампанию, и планирование мероприятий и развлечений для гостей. В современных условиях особенно важен онлайн-маркетинг, так как большинство потенциальных клиентов ищут информацию о гостиницах через интернет.

Изучив предметную область работы администратора отеля, можно сделать вывод, что это сложный и многогранный бизнес, который требует высокой организационной и маркетинговой культуры. Отель должен обеспечивать максимальный уровень комфорта и безопасности для своих гостей, а также предоставлять различные услуги, чтобы гости получили удовольствие от пребывания в нем.

Поэтому разработка дипломного проекта на тему «Создание программного средства для автоматизации работы администратора гостиницы «Бона» является актуальной задачей. К разрабатываемому программному средству составлена пояснительная записка, которая состоит из шести разделов.

В первом разделе «Описание задачи» подробно раскрывается её сущность, а именно цели создания приложения и постановка задачи.

Во втором разделе «Проектирование системы» перечисляются требования к десктоп приложению, его структура, проектирование макета, программно-технические средства необходимые для разработки, описывается защита и сохранность данных.

В третьем разделе «Описание реализации программного средства» отражает общие сведения о программе, функциональное назначение, структуру входных и выходных данных, приводится описание справочной системы, а также описывается организация и ведение базы данных.

Четвёртый раздел «Применение» содержит информацию, необходимую в процессе эксплуатации программного средства: его назначение, условия применения, а также приводится описание справочной системы.

В пятом разделе «Охрана труда и окружающей среды» описываются вопросы охраны труда в соответствии с заданием, полученным от консультанта по разделу охраны труда и техники безопасности.

В шестом разделе «Экономический раздел» описывается технико-экономическое обоснование разработки программного средства, составляется план по разработке программного средства, и экономическая эффективность разработки.

В заключении отражается степень соответствия проектных решений заданию.

В списке использованных источников приводится список книг и интернет-ресурсов, использование которых помогло в процессе разработки программного средства.

В приложении А представлен текст программных модулей.

В приложении Б приведены результаты работы программы.

Графическая часть содержит диаграммы вариантов использования, последовательности, классов и деятельности.

## **1 Описание задачи**

### **1.1 Анализ предметной области**

Отель является бизнесом, который зависит от своей репутации. Если у гостей сложилось положительное впечатление о предоставляемом сервисе, то они будут склонны рекомендовать отель своим друзьям и знакомым, а также вернуться снова и снова. Важно наладить скорость предоставляемых услуг, чтобы гости оставались довольными и посещали отель не единожды. Поэтому важна автоматизация процесса работы администратора отеля. Сотруднику необходимо вести учет бронирований, предоставления дополнительных услуг.

Целью решаемой задачи является упрощение ведения учета данных отеля. Для налаженной работы необходимо участие администратора, который будет владеть всей общей информацией и вести базу данных. Также программное средство облегчает использование приложения, предлагая пользователю удобный и лаконичный интерфейс.

Для того, чтобы достичь наибольшей эффективности, нужно автоматизировать следующие процессы:

- регистрация новых гостей;
- учет номеров;
- предоставление дополнительных услуг.

Программное средство позволяет администратору вести учет бронирований, учет гостей и номеров, а также предоставлять дополнительные услуги гостям отеля. Таким образом, программное средство для автоматизации работы администратора гостиницы «Бона» должно помочь улучшить эффективность работы гостиницы, снизить количество ошибок при учете.

### **1.2 Постановка задачи**

Исходя из анализа предметной области можно выделить следующие задачи, подлежащие автоматизации:

- организация ведения базы данных, содержащей информацию о номерах гостиницы и предоставляемых услугах;
- регистрация новых бронирований;
- сохранение информации о госте отеля;
- поиск информации по различным критериям;
- формирование отчётов по различным критериям;
- отображение списка номеров и прайс-листа на проживание в гостинице.

В ходе исследования предметной области были выявлены аналоги данного программного средства, которые широко распространены в интернете, в то время как разрабатываемое приложение десктоп версии, включило в себя все необходимые функции.

## **2 Проектирование системы**

### **2.1 Требования к приложению**

Требования к интерфейсу программного продукта:

- приложение должно быть легко и понятно в использовании;
- дизайн приложения должен быть лаконичным, но в то же время и современным;

Система управления контентом приложения должна обеспечить пользователю возможность следующих действий:

- возможность просматривать, добавлять и изменять бронирования
- хранить данные о бронированиях номеров;
- реализовать просмотр бронирований;
- обеспечить поиск, фильтрацию записей по различным категориям;
- реализовать формирование чека и отчетов по интересующей пользователя информации.

Требования к аппаратным и операционным ресурсам:

- среда разработки Visual Studio 2022;
- мышь и клавиатура;
- оперативная память 4,00 ГБ;
- видеокарта NVIDIA GeForce GTX 680;
- пакет программ Microsoft Office;
- MS SQL Server 2019.

### **2.2 Проектирование модели**

На основании проведенного анализа предметной области и выявленных функциональных возможностей, необходимо будет построить модель, которая бы отображала функциональную структуру объектов программного средства, действия, производимые им и связь между этими действиями, а также определить инструменты разработки, которые будут использоваться в ходе написания программного средства.

В качестве инструмента для построения такой модели программного средства будет выбран унифицированный язык моделирования. Данный язык моделирования предназначен для специфицирования, визуализации, конструирования и документирования программного средства.

В рамках языка UML все представления о модели сложной системы фиксируются в виде специальных графических конструкций – диаграмм. В терминах языка UML определены следующие виды диаграмм: диаграмма вариантов использования, диаграмма последовательности, диаграмма классов, диаграмма деятельности.

Перечень этих диаграмм представляет собой неотъемлемую часть графической нотации языка UML, сам процесс объектно-ориентированного программирования (ООП) неразрывно связан с процессом построения этих диаграмм.

Суть диаграммы вариантов использования заключается в том, что проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью, так называемых вариантов использования.

Варианты использования описывают как взаимодействие между пользователем и сущностью, так и реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами сущности. Варианты использования могут включать в себя описание особенностей способов реализации сервиса и различных исключительных ситуаций, таких как корректная обработка ошибок системы. Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы.

В данной проектируемой системе в качестве актера выступает администратор, который управляет данными программного средства. На диаграмме представлены все возможные действия актеров к программному средству.

Диаграмма вариантов использования, отражающая варианты приложения для пользователя системы представлена в графической части на 1 листе.

При моделировании поведения проектируемой или анализируемой системы возникает необходимость детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности.

Диаграмма последовательности – основная цель этих диаграмм это визуализация как операции происходят во времени, для более детального описания логики сценариев использования. Диаграмма последовательности для формирования нового бронирования представлена на листе 2 графической части.

Диаграммы деятельности – частный случай диаграмм состояний. Основная цель использования таких диаграмм – визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. Диаграмма деятельности представлена на листе 4 графической части.

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывать их внутреннюю структуру и типы отношений.

Диаграмма классов представлена на листе 3 графической части.



## 2.3 Проектирование структуры базы данных

Цель моделирования данных состоит в обеспечении разработчика информационной системы концептуальной схемой базы данных в форме одной или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему без данных.

Наиболее распространенным средством моделирования данных является диаграмма «Сущность-связи» (ERD). С их помощью определяется важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных.

Исходя из предметной области можно выделить следующие сущности разработки: «Клиент», «Номер», «Бронирование», «Услуги».

Для сущности «Клиент» атрибутами будут являться:

- номер паспорта;
- имя;
- фамилия;
- номер телефона;
- e-mail адрес;
- пол.

Для сущности «Номер» атрибутами будут являться:

- тип комнаты;
- статус комнаты;
- цена за ночь;
- этаж.

Для сущности «Бронирование» атрибутами будут являться:

- дата заезда;
- дата отъезда;
- клиент;
- номер;
- дата бронирования.

Для сущности «Услуги» атрибутами будут являться:

- название;
- стоимость.

Диаграмма «Сущность-связь» представлена на рисунке 2.1.

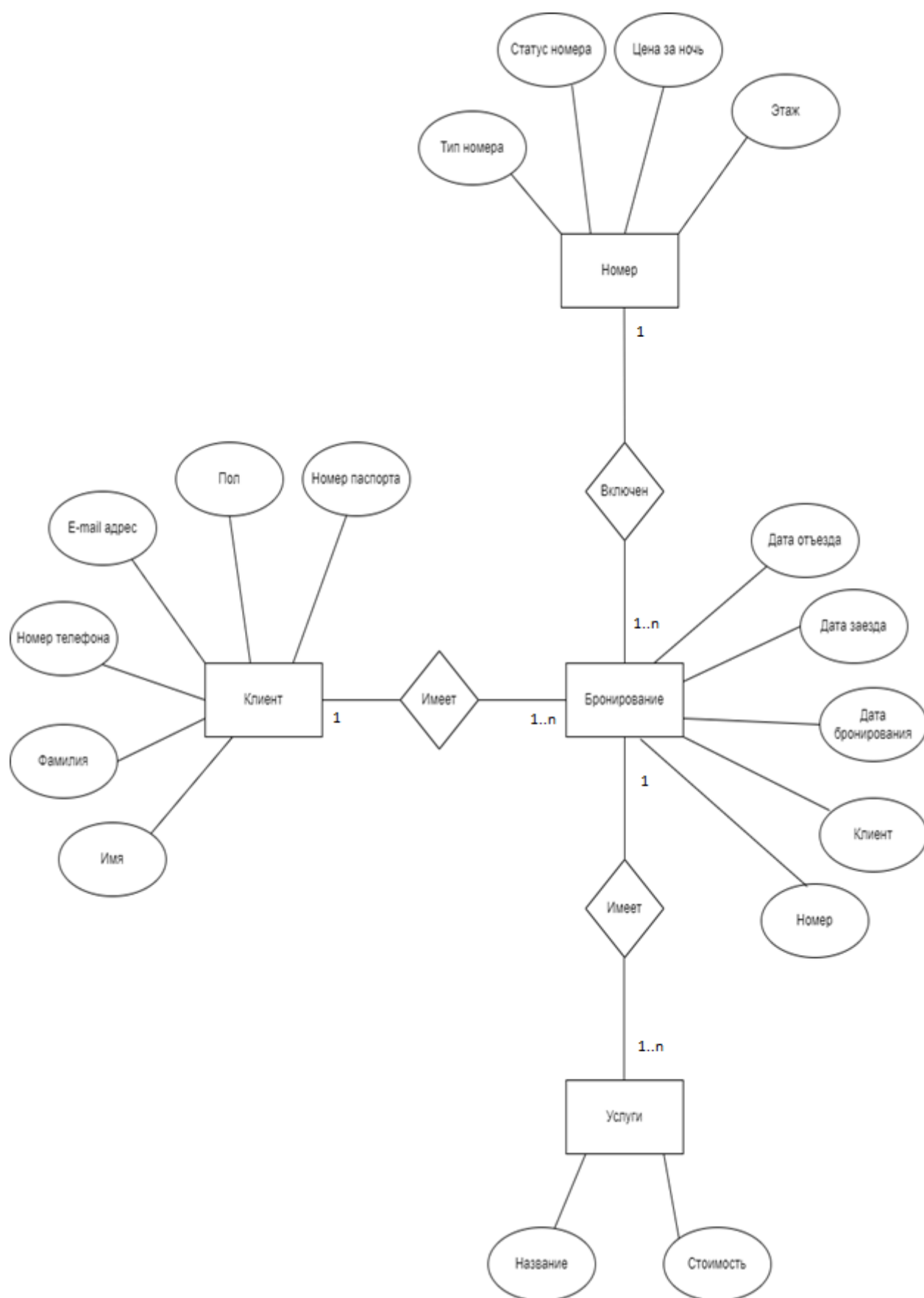


Рисунок 2.1 – Диаграмма «Сущность-связь»

## 2.4 Концептуальный прототип

Концептуальный прототип состоит из описания внешнего пользовательского интерфейса, а именно, элементов управления.

При создании данного приложения важную роль играют формы, так как они являются основным средством работы с пользователем. Разрабатываемое приложение будет содержать несколько форм.

Все формы будут содержать стандартные пользовательские элементы управления.

В рабочем режиме программы, пользователю, для удобной навигации, будет предоставлено меню.

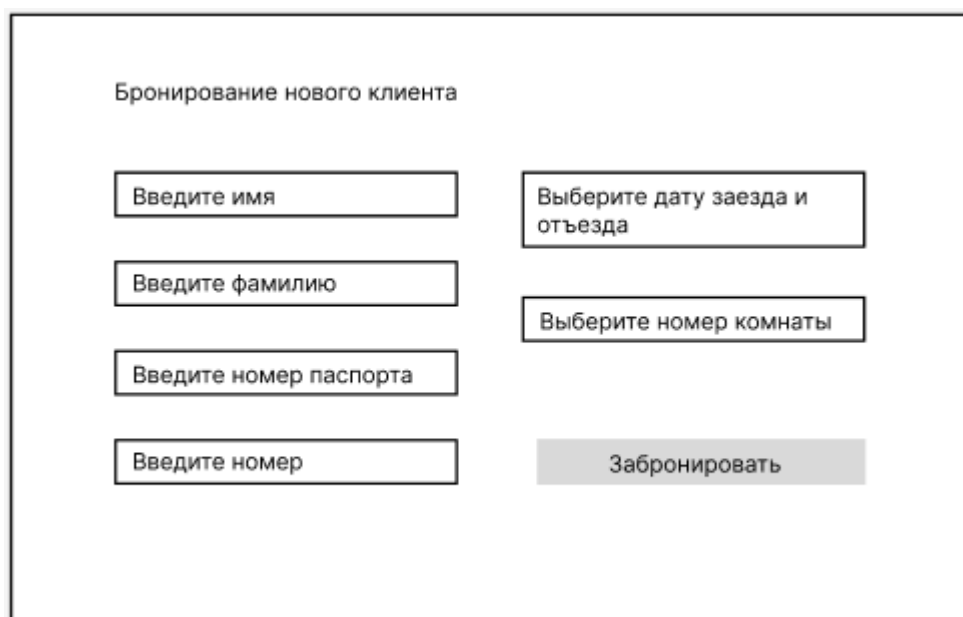
На главной странице расположен весь необходимый для пользователя функционал.

Все страницы главной формы будут иметь схожий интерфейс. Макет главной формы представлен на рисунке 2.2.



Рисунок 2.2 – Макет главной формы

При нажатии на кнопку «Новое бронирование» будет открыта форма для регистрации нового клиента. Макет формы представлен на рисунке 2.3.

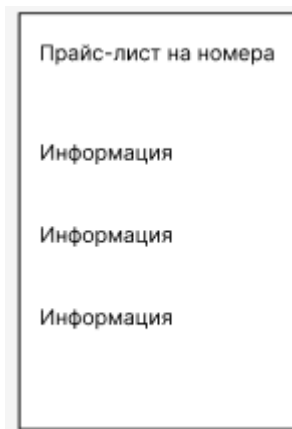


Бронирование нового клиента

Введите имя	Выберите дату заезда и отъезда
Введите фамилию	Выберите номер комнаты
Введите номер паспорта	
Введите номер	Забронировать

Рисунок 2.3 – Макет формы бронирования

При нажатии на кнопку «Прайс-лист» будет открыта форма с информацией о стоимости номеров. Макет формы представлен на рисунке 2.4.



Прайс-лист на номера

Информация
Информация
Информация

Рисунок 2.4 – Макет формы прайс-листа

При нажатии на кнопку «Услуги» будет открыта форма для добавления услуг к номеру. Макет формы представлен на рисунке 2.5.

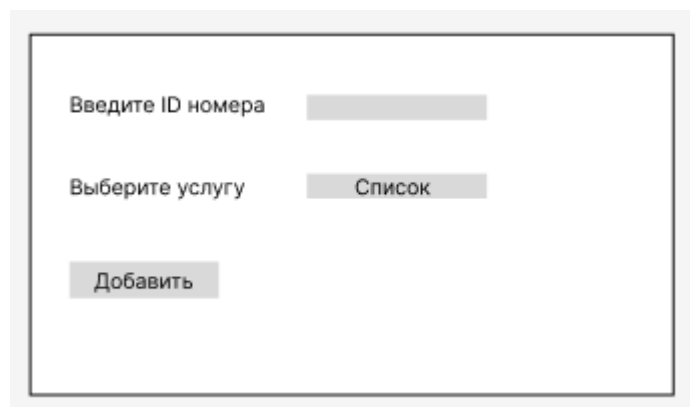
A rectangular form with a thin black border. Inside, there are three elements: a label 'Введите ID номера' followed by a light gray rectangular input field; a label 'Выберите услугу' followed by a light gray button labeled 'Список'; and a light gray button labeled 'Добавить' at the bottom left.

Рисунок 2.5 – Макет формы услуг

При нажатии на кнопку «Транзакции» в меню будет открыта форма с последними осуществленными транзакциями клиента. Макет формы представлен на рисунке 2.6.

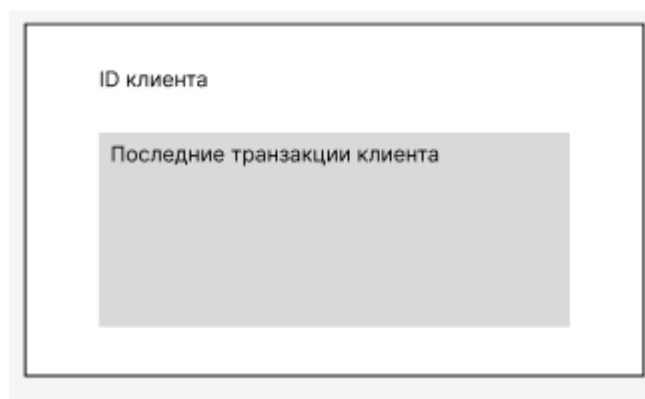
A rectangular form with a thin black border. It contains two elements: a label 'ID клиента' at the top left, and a large light gray rectangular area below it with the label 'Последние транзакции клиента' at its top left corner.

Рисунок 2.6 – Макет формы просмотра транзакций

При нажатии на кнопку «Работа с БД гостей» в меню будет открыта форма с базой данных гостей. Макет формы представлен на рисунке 2.7.

A rectangular form with a thin black border. It contains three elements: a label 'БД гостей отеля' at the top left; a large light gray rectangular area below it; and three light gray buttons labeled 'Изменить', 'Удалить', and 'Добавить' at the bottom, spaced out horizontally.

Рисунок 2.7 – Макет формы данных гостей

При нажатии на кнопку «Работа с БД номеров» в меню будет открыто окно с базой данных номеров. Макет формы представлен на рисунке 2.8.



Рисунок 2.8 – Макет формы данных номеров

### 3 Описание реализации программного средства

#### 3.1 Инструменты разработки и применяемые технологии

Программное обеспечение, используемое для разработки программного средства:

- среда разработки Visual Studio 2019;
- операционная система Windows 10;
- пакет программ Microsoft Office;
- программа для создания файлов справки DR.Explain;
- программа для разработки диаграмм Draw.io;
- система управления базами данных (СУБД) Microsoft SQL Server;
- объектно-ориентируемый язык программирования C#.

Visual Studio 2022 – это лучший инструмент для разработки любого приложения под любую платформу. Система управления версиями в этом выпуске делает разработку гибкой, а совместную работу – эффективной.

Операционная система – это набор управляющих программ, предназначенных для управления ресурсами вычислительной системы как единого комплекса, другими словами, операционная система – это набор программного обеспечения, который обеспечивает работу компьютера.

Microsoft Office 2021 – это совокупность программных средств автоматизации офисной деятельности. В состав пакета входит множество приложений, каждое из которых предназначено для выполнения определенных функций и может быть использовано автономно и независимо от остальных[12].

Dr. Explain – это приложение для быстрого создания файлов справки, справочных систем, онлайн-руководств пользователя, пособий и технической документации к программному обеспечению и техническим системам [11].

Программа «draw.io» - это интернет-ресурс, позволяющий разрабатывать различные диаграммы. Имеет огромный функционал и все необходимые инструменты, необходимые для разработки диаграмм [8].

SQL - это структурированный язык запросов, созданный для того, чтобы получать из базы данных необходимую информацию. Если описать схему работы SQL простыми словами, то специалист формирует запрос и направляет его в базу. Та в свою очередь обрабатывает эту информацию, «понимает», что именно нужно специалисту, и отправляет ответ[7].

SQL Server Management Studio (SSMS) - это интегрированная среда для управления любой инфраструктурой SQL. Используйте SSMS для доступа, настройки, администрирования и разработки всех компонентов SQL Server [9].

C# - объектно-ориентированный, ориентированный на компоненты язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. [10].

### 3.2 Организация данных

Организация данных подразумевает создание модели данных, главными элементами которой являются сущности и их связи.

На основании ER-модели была создана база данных. Структура базы данных разрабатываемого программного средства включает четыре таблицы. Описание таблиц приводится в таблицах 3.1-3.4.

Таблица «Клиент» хранит информацию о всех посетителях, структура которой представлена в таблице 3.1.

Таблица 3.1 – Структура таблицы «Клиент»

Имя поля	Тип поля	Размер поля, байт	Описание поля
ID_клиента	int	4	Идентификатор
Имя	varchar	20	Имя клиента
Фамилия	varchar	20	Фамилия клиента
Номер_телефон	varchar	20	Номер телефона клиента
Номер_паспорта	varchar	20	Номер паспорта клиента
E-mail_адрес	varchar	20	E-mail адрес клента
Пол	varchar	2	Пол клиента

Таблица «Номер» хранит информацию о номерах, структура которой представлена в таблице 3.2.

Таблица 3.2 – Структура таблицы «Номер»

Имя поля	Тип поля	Размер поля, байт	Описание поля
ID_номера	int	4	Идентификатор
Тип_номера	int	2	Тип номера
Статус_комнаты	varchar	20	Статус комнаты
Цена_за_ночь	varchar	10	Цена за ночь проживания
Этаж	varchar	3	Этаж номера

Таблица «Бронирование» хранит информацию о бронированиях, структура которой представлена в таблице 3.3.



Таблица 3.3 – Структура таблицы «Бронирование»

Имя поля	Тип поля	Размер поля, байт	Описание поля
ID_брони	int	4	Идентификатор
ID_клиента	varchar	3	Идентификатор_клиента
ID_номера	varchar	3	Идентификатор_номера
Дата_заезда	varchar	12	Дата заезда клиента
Дата_отъезда	varchar	12	Дата отъезда клиента
Дата_бронирования	varchar	12	Дата бронирования

Таблица «Услуги» хранит информацию о дополнительных услугах отеля, структура которой представлена в таблице 3.4.

Таблица 3.4 – Структура таблицы «Услуги»

Имя поля	Тип поля	Размер поля, байт	Описание поля
ID_услуги	int	4	Идентификатор
Название	varchar	20	Название услуги
Стоимость	varchar	20	Стоимость услуги

Схема базы данных представлена на рисунке 3.1.

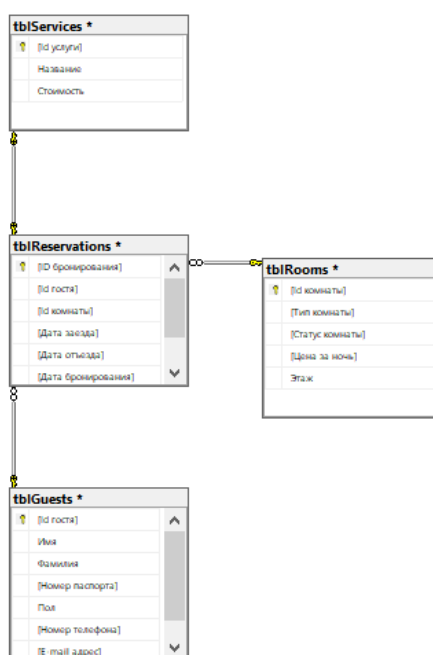


Рисунок 3.1 – Схема базы данных

### 3.3 Функции: логическая и физическая организации

Диаграмма вариантов использования является концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними.

На основании диаграммы вариантов использования в приложении будут реализованы следующие основные функции:

- новое бронирование;
- просмотр бронирований.

Для добавления нового бронирования необходимо нажать на кнопку «Новое бронирования» и ввести необходимые данные.

Код функции обработки запроса представлен ниже:

```
private void btnSubmit_Click(object sender, RoutedEventArgs e)
{
    if (rdbExistingGuest.IsChecked == true) // проверка был ли гость раньше
    {
        string error = "";
        conn.Open();
        if (txtGuestID.Text != "")
        {
            if (isUserExist(txtGuestID.Text) > 0)
            {
                if (cmbRoomType.SelectedItem == null ||
cmbRoomNumber.SelectedItem == null)
                {
                    error += "\n Нужно выбрать комнату!!";
                }
                if (dpiStartDate.Text == "" || dpiEndDate.Text == "")
                {
                    error += "\n Нужно выбрать дату!!";
                }

                if (error == "")
                {
                    SqlCommand insertReservation = new SqlCommand("INSERT INTO
tblReservations (GuestID,RoomID,EmployeeID,DateMade,
ReservationStartDate,ReservationEndDate) VALUES (@guest,@room,@emp,CONVERT (date,
SYSDATETIME()),@start,@end)", conn);
                    insertReservation.Parameters.Add(new SqlParameter("guest",
txtGuestID.Text));
                    insertReservation.Parameters.Add(new SqlParameter("room",
cmbRoomNumber.Text));
                    insertReservation.Parameters.Add(new SqlParameter("emp",
txtEmployeeID.Text));
                    insertReservation.Parameters.Add(new SqlParameter("start",
dpiStartDate.Text));
                    insertReservation.Parameters.Add(new SqlParameter("end",
dpiEndDate.Text));

                    int r = insertReservation.ExecuteNonQuery();
                    if (r == 0)
                    {
                        MessageBox.Show("Невозможно забронировать!");
                    }
                    else
                    {
                        MessageBox.Show("Успешно забронировано!!");
                        NewReservation rs = new NewReservation();
                        rs.Show();
                        this.Close();
                    }
                }
            }
        }
    }
}
```

```

        }
    }
    else
    {
        MessageBox.Show(error);
    }
    if (conn != null)
    {
        conn.Close();
    }
}
else
{
    MessageBox.Show("\nГость с ID - " + txtGuestID.Text + " не
найден\n");
    txtGuestID.Focus();
    txtGuestID.SelectAll();
}
}
else
{
    MessageBox.Show("Такого гостя не существует!");
}

conn.Close();
}
else if(rdbNewGuest.IsChecked == true) // действие если гость в отеле
{
    string error = "";
    conn.Open();
    if (cmbRoomType.SelectedItem == null || cmbRoomNumber.SelectedItem ==
null)
    {
        error += "\n Номер должен быть выбран!";
    }
    if (dpiStartDate.Text == "" || dpiEndDate.Text == "")
    {
        error += "\n Дата должна быть выбрана!";
    }
    if (txtFirstName.Text == "" || txtLastName.Text == "")
    {
        error += "\nИмя и Фамилия обязательны к заполнению!";
    }
    if (txtPhone.Text == "")
    {
        error += "\nВведите номер телефона!";
    }
    if (txtPhone.Text.Length != 12)
    {
        error += "\nТелефон номер должен состоять из 12 символов!";
    }
    if (txtAddress.Text == "")
    {
        error += "\nАдрес не заполнен!";
    }
    if (txtPostalCode.Text == "")
    {
        error += "\nИндекс не заполнен!";
    }
    if (txtPostalCode.Text.Length != 6)
    {
        error += "\nИндекс должен состоять из 6 символов!";
    }
}

```

```

        if (error == "")
        {
            SqlCommand insertGuest = new SqlCommand("INSERT INTO tblGuests
(GuestID,FirstName,LastName,GuestAddress, PostalCode,Phone,EmailAddress) VALUES
(@guest,@first,@last,@address,@postal,@phone,@email)", conn);

            insertGuest.Parameters.Add(new SqlParameter("guest",
txtGuestID.Text));
            insertGuest.Parameters.Add(new SqlParameter("first",
txtFirstName.Text));
            insertGuest.Parameters.Add(new SqlParameter("last",
txtLastName.Text));
            insertGuest.Parameters.Add(new SqlParameter("address",
txtAddress.Text));
            insertGuest.Parameters.Add(new SqlParameter("postal",
(txtPostalCode.Text).ToUpper()));
            insertGuest.Parameters.Add(new SqlParameter("phone",
txtPhone.Text));
            insertGuest.Parameters.Add(new SqlParameter("email",
txtEmail.Text));
            int result = insertGuest.ExecuteNonQuery();
            if (result == 0)
            {
                MessageBox.Show("Невозможно добавить гостя!");
            }
            else
            {
                SqlCommand insertReservation = new SqlCommand("INSERT INTO
tblReservations (GuestID,RoomID,EmployeeID,DateMade,
ReservationStartDate,ReservationEndDate) VALUES (@guest,@room,@emp,CONVERT (date,
SYSDATETIME()),@start,@end)", conn);
                insertReservation.Parameters.Add(new SqlParameter("guest",
txtGuestID.Text));
                insertReservation.Parameters.Add(new SqlParameter("room",
cmbRoomNumber.Text));
                insertReservation.Parameters.Add(new SqlParameter("emp",
txtEmployeeID.Text));
                insertReservation.Parameters.Add(new SqlParameter("start",
dpiStartDate.Text));
                insertReservation.Parameters.Add(new SqlParameter("end",
dpiEndDate.Text));
                int r = insertReservation.ExecuteNonQuery();
                if (r == 0)
                {
                    MessageBox.Show("Невозможно забронировать!");
                }
                else
                {
                    MessageBox.Show("Успешно забронировано!");
                    NewReservation rs = new NewReservation();
                    rs.Show();
                    this.Close();
                }
            }
        }
        else
        {
            MessageBox.Show(error);
        }
        if (conn != null)
        {
            conn.Close();
        }
    }
}

```

```
conn.Close();
```

```
}
```

Для получения информации о существующих бронированиях необходимо нажать на кнопку «Бронирования» на главной странице приложения. Код функции обработки запроса представлен ниже:

```
public Reserv(string id, string type)
{
    sql = "select serviceTransactionID as'Transaction ID', servicedescription as
'Description', servicedate as Date,amount as Amount from tblServicesTransactions inner
join tblServices on tblServicesTransactions.ServiceID = tblServices.ServiceId where
ReservationID = '" + 1 + "'";

    //sql = "select orderID as'Transaction ID', Transactiondate as Date,Cost as
Amount from tblRestaurantTransactions where ReservationID = '" + 1 + "'";

    InitializeComponent();
    this.id = id;
    SqlCommand tr = new SqlCommand(sql, conn);
    SqlDataAdapter dataAdapter = new SqlDataAdapter(tr); //c.con is the
connection string

    DataTable dtRecord = new DataTable();
    dataAdapter.Fill(dtRecord);
}
```

Для просмотра прайс-листа на услуги и номера отеля необходимо нажать на кнопку «Прайс-листы» на главной странице приложения. Код функции обработки запроса представлен ниже:

```
private void btnUpdate_Click(object sender, RoutedEventArgs e)
{
    double price = 0;
    if(double.TryParse(txtPrice.Text,out price) && price > 0)
    {
        conn.Open();

        if (cmbRoomType.SelectedValue.ToString() == "Suite")
        {
            SqlCommand update = new SqlCommand("UPDATE tblRooms SET Cost='" +
txtPrice.Text + "' where RoomID = '" + cmbRoomNumber.Text + "'", conn);
            int r = update.ExecuteNonQuery();

            if (r > 0)
            {
                Int r;
                MainWindow main = new MainWindow();
                main.Show();
                this.Close();
            }
        }
        else
        {
            SqlCommand update = new SqlCommand("Update tblrooms set cost = '" +
txtPrice.Text + "' where RoomTypeID = ( select top 1 tblrooms.roomtypeid from tblrooms
inner join tblroomtype on tblrooms.RoomTypeID = tblroomtype.RoomTypeID where
Typedescription = '" + cmbRoomType.Text + "')", conn);

            int r = update.ExecuteNonQuery();
            SqlCommand update = new SqlCommand("Update tblrooms set cost = '" + txtPrice.Text + "'
where RoomTypeID = ( select top 1 tblrooms.roomtypeid from tblrooms where
Typedescription = '" + cmbRoomType.Text + "')", conn);
            if (r > 0)
            {
                MainWindow main = new MainWindow();
            }
        }
    }
}
```

```

        main.Show();
        this.Close();
    }
}
else
{
    MainWindow main = new MainWindow();
    main.Show();
    this.Close();
}
}

```

Полный текст программы представлен в приложении А.

### 3.4 Входные и выходные данные

Входными являются данные, введенные администратором в таблицы базы данных: «Клиенты», «Номера», «Бронирования», «Услуги».

Входными данными при добавлении данных о новом клиенте будут являться:

- имя;
- фамилия;
- номер телефона;
- номер паспорта;
- e-mail адрес;
- пол.

Входными данными при добавлении данных о новом номере являются:

- тип комнаты;
- статус комнаты;
- цена за ночь;
- этаж.

Входными данными при добавлении данных о новом бронировании будут являться:

- дата заезда;
- дата бронирования;
- клиент;
- комната;
- дата отъезда.

Входными данными при добавлении данных о новой услуги будут являться:

- название;
- стоимость.

Выходными данными будет являться чек, сформированный для выбранного клиента по введенным данным, сформированный чек представлен в Приложении Б на рисунке Б.6.

### 3.5 Функциональное тестирование

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям и проводится для выявления неполадок и недочетов программы на этапе ее сдачи в эксплуатацию.

Проведем тестирование проверки каждого пункта с целью проверки всех функций.

Тестирование программы будет производиться последовательно, переходя из одной части программы в другую. Во время теста будут проверяться все действия с программой, навигация пунктам меню, которые может произвести пользователь. После чего, все собранные и найденные ошибки будут исправлены.

В таблицах 3.5 – 3.9 представлены тест-кейсы, подготовленные для проведения функционального тестирования.

Таблица 3.5 – Функция тестирования нового бронирования

№	Модуль / Функция	Шаги выполнения	Ожидаемый результат	Фактический результат
1	Новое бронирование.	1. Зайти в приложение. 2. Нажать на кнопку «Новое бронирование». 3. Выбрать «Новый гость». 4. Заполнить поле «Имя» данными «Артем». 5. Заполнить поле «Фамилия» данными «Яроцкий». 6. Заполнить поле «Номер паспорта» данными «AB133722». 7. Заполнить поле «Номер телефона» данными «+375445570550». 8. Заполнить поле «E-mail адрес» данными «college@diplom.by». 9. Заполнить поле «Пол» данными «Мужской». 10. Заполнить поле «Дата заезда» данными «27.09.2023». 11. Заполнить поле «Дата отезда» данными «05.10.2023». 12. Заполнить поле «Тип номера» данными «Luxe». 13. Заполнить поле «Номер комнаты» данными «U201». 14. Нажать на кнопку «Забронировать».	Добавление в базу данных нового бронирования.	Добавление в базу данных нового бронирования. Результат работы отображен в приложении Б на рисунке Б.1.

Таблица 3.6 – Функция удаления записи из базы данных

№	Модуль / Функция	Шаги выполнения	Ожидаемый результат	Фактический результат
2	Проверка возможности удаления записи из базы данных.	1. Зайти в приложение. 2. Нажать на кнопку «Бронирования». 3. Нажать на кнопку «Редактирование». 3. Заполнить поле «ID бронирования» данными «1004». 4. Нажать на кнопку «Удалить бронирование».	Удаление записи из базы данных.	После нажатия на кнопку «Удалить» запись пропала из списка и была удалена из базы данных. Результат работы представлен в приложении на рисунке Б.2

Таблица 3.7 – Функция изменения записи в базе данных

№	Модуль / Функция	Шаги выполнения	Ожидаемый результат	Фактический результат
3	Изменение записи в базе данных.	1. Зайти в приложение. 2. Нажать на кнопку «Бронирования». 3. Нажать на кнопку «Редактирование». 3. Заполнить поле «Номер бронирования» данными «1009». 4. Заполнить поле «Тип комнаты» данными «Double». 5. Заполнить поле «Номер комнаты» данными «D203» 6. Нажать на кнопку «Сохранить».	Изменение записи базе данных приложения.	После нажатия на кнопку в базе данных изменяется запись. Результат работы представлен в приложении на рисунке Б.3

Таблица 3.8 – Функция просмотра прайс-листа

№	Модуль / Функция	Шаги выполнения	Ожидаемый результат	Фактический результат
4	Просмотр прайс-листа.	1. Зайти в приложение. 2. Нажать на кнопку «Прайс-листы». 3. Выбрать в меню пункт прайс-листа «Luxe».	Отображение прайс-листа на выбранную услугу.	После выбора услуги цена была отображена. Результат работы представлен в приложении Б на рисунке Б.4



Таблица 3.9 – Функция выдачи чека

№	Модуль / Функция	Шаги выполнения	Ожидаемый результат	Фактический результат
5	Выдача чека.	1. Зайти в приложение. 2. Нажать на кнопку «Новое бронирование». 3. Выбрать «Новый гость». 4. Заполнить поле «Имя» данными «Артем». 5. Заполнить поле «Фамилия» данными «Яроцкий». 6. Заполнить поле «Номер паспорта» данными «AB133722». 7. Заполнить поле «Номер телефона» данными «+375445570550». 8. Заполнить поле «E-mail адрес» данными «college@diplom.by». 9. Заполнить поле «Пол» данными «Мужской». 10. Заполнить поле «Дата заезда» данными «27.09.2023». 11. Заполнить поле «Дата отезда» данными «05.10.2023». 12. Заполнить поле «Тип номера» данными «Luxe». 13. Заполнить поле «Номер комнаты» данными «U201». 14. Нажать на кнопку «Забронировать». 15. Нажать на кнопку «Ок»	Отображение чека по введенным данным.	Отображение чека по введенным данным. Результат работы отображен в приложении Б на рисунке Б.5.

### 3.6 Описание справочной системы

Справочная система — это программный компонент, выполняющий роль документации для программного пакета. В первую очередь он описывает и объясняет каждую функцию программы, кнопку, параметр панели инструментов или другой элемент в пользовательском интерфейсе.

В программном средстве разработана справочная система в программе «Dr.Explain».

Dr.Explain — это программное обеспечение для создания профессиональных документаций, инструкций и справочных материалов. С

помощью Dr.Explain вы можете легко создавать скриншоты, рисовать на изображениях и создавать диаграммы, чтобы проиллюстрировать процессы и функции вашего продукта или приложения.

Иерархия разделов справочной системы представлена на рисунке 3.1.

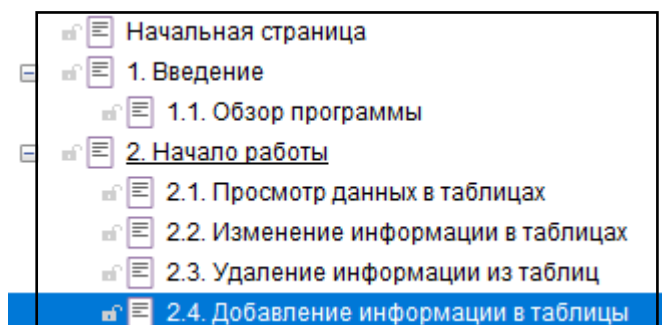


Рисунок 3.1 – Иерархия разделов справочной системы

При выборе необходимо раздела пользователю выдается соответствующая информация.

Например, при выборе «Просмотр данных в таблицах» пользователю предоставляется информация о данных в таблице.

Данный раздел представлен на рисунке 3.2.

## 2.1. Просмотр данных в таблицах

Для просмотра данных в таблицах необходимо выбрать нужную таблицу и нажать на нее.

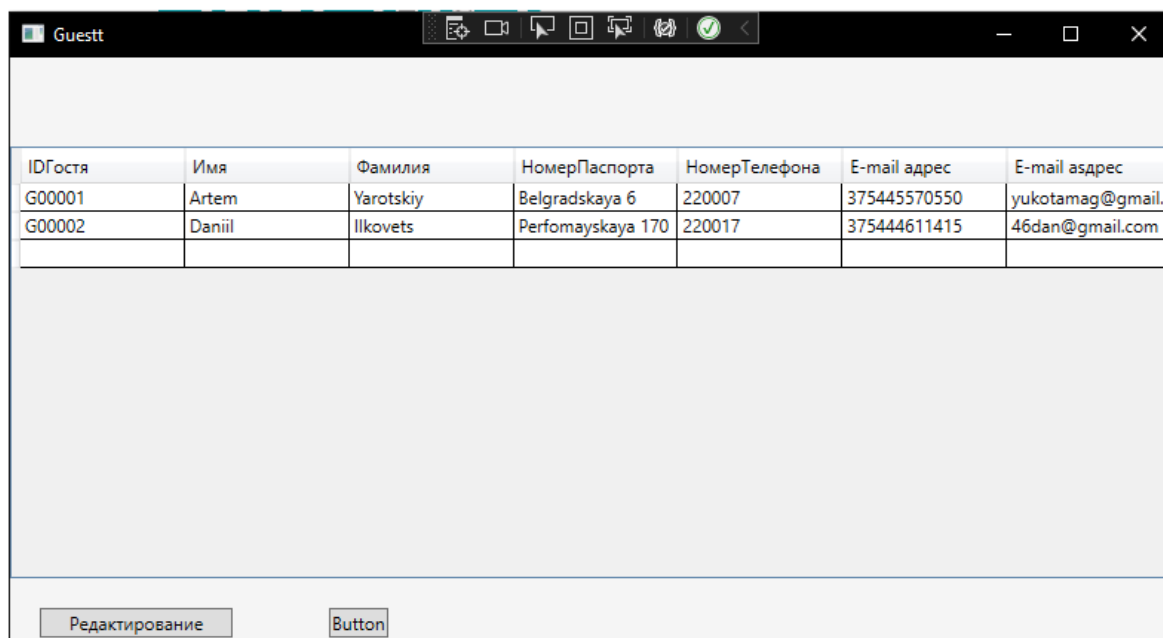


Рисунок 3.2 – Просмотр данных в справочной системе

## **4 Применение**

### **4.1 Назначение программного средства**

Программное средство для автоматизации работы отеля позволяет эффективно управлять работой отеля, оптимизировать процессы бронирования и сократить время на подготовку отчетности. С помощью данного программного средства ведется учет гостей и бронирований, а также формирование отчетности.

### **4.2 Условия применения**

Для работы с программой необходимо наличие программного обеспечения:

- операционная система, начиная с Windows 11;
- система управления базами данных Microsoft SQL Server Express 2019;
- .NET 6.0;
- офисный пакет приложений Microsoft Office (MS Office) 2021;

На случай редактирования проекта программы необходимо наличие программного обеспечения:

- интегрированная среда Microsoft SQL Server Management Studio 2018;
- среда разработки приложений Microsoft Visual Studio 2022.

## **5 Охрана труда и окружающей среды**

### **5.1 Правовые, нормативные, социально-экономические и организационные вопросы охраны труда**

В Гостинице «Бона» при производстве необходимо соблюдение правил и норм техники безопасности и производственной санитарии. На предприятии разработана и функционирует Система управления охраной труда (СУОТ). СУОТ разработана и в соответствии с требованиями нормативных правовых актов Республики Беларусь в области охраны труда и «Системы управления охраной труда».

Обучение, инструктаж и проверка знаний работников по вопросам охраны труда являются важным элементом системы мер по предупреждению аварий и травматизма на производстве, обеспечению конституционного права граждан на здоровые и безопасные условия труда и носят непрерывный многоуровневый характер.

Организация обучения работников по вопросам охраны труда возлагается на службы подготовки кадров или технического обучения предприятий связи (отдел, бюро, инженера по обучению или лица, на которое возложены эти обязанности).

Контроль за своевременностью и качеством обучения, инструктажа и проверки знаний работников по вопросам охраны труда осуществляют службы охраны труда (отдел, бюро, инженеры или лица, на которых возложены эти обязанности). Также производится инструктаж работников по охране труда.

Охрана труда является одним из важных направлений в обеспечении безопасности труда. В Гостинице «Бона» данная система имеет свои нормативные документы.

Основными задачами «Службы охраны труда» в Гостинице «Бона» являются:

- внедрение научных разработок по безопасности и гигиене труда, активная пропаганда охраны труда;
- совершенствование Системы управления охраной труда.
- консультирование работников предприятия по вопросам охраны труда;
- контроль соблюдения всех норм и требований охраны труда на предприятии.

Координация работы охраны труда, а также управление «Службы охраны труда» осуществляется главным инженером Гостиницы «Бона», ответственным за реализацию всех функций СУОТ является также главный инженер.

Функции главного инженера:

- подготовка отчётности по охране труда;
- организация обучения всех работников по вопросам охраны труда;

- участие в организации улучшения условий охраны труда на предприятии;
- проведение проверок рабочих мест, в том числе с привлечением специалистов по охране труда;
- проведение вводного инструктажа.

Контроль охраны труда в Гостинице «Бона» проводится по следующим критериям:

- проверка состояния рабочих мест и оборудования;
- проверка соблюдения правил работы с техническими устройствами;
- проведение инструктажей по охране труда и мероприятий связанных с охраной труда.

Все эти мероприятия необходимы для установления эффективного контроля за соблюдением требований охраны труда на работе, предотвращения возникновения производственного травматизма и заболеваний, а также повышения уровня безопасности труда в организации.

Социально-экономические вопросы охраны труда в Гостинице «Бона» связаны с обеспечением безопасности труда на производстве, предоставлением работникам соответствующей специальной оборудования, а также обучением персонала.

## **5.2 Предупреждение пожаров на производстве**

На предприятиях в соответствии с законодательством организация работ по обеспечению пожарной безопасности возложена на их руководителей. Они обязаны:

- организовать изучение и выполнение на подведомственных объектах действующих законодательных и нормативных документов в области пожарной безопасности с проведением противопожарного инструктажа и занятий по пожарно-техническому минимуму;
- создавать и контролировать боеспособность добровольных пожарных дружин (пожарно-сторожевой охраны);
- устанавливать на объектах строгий противопожарный режим и постоянно контролировать его соблюдение;
- обеспечивать разработку плана действий работников на случай возникновения пожара и проведение практических тренировок по его отработке;
- периодически проверять состояние пожарной безопасности объектов, исправность технических средств борьбы с пожарами и устранять выявленные недостатки;
- принимать дополнительные меры по усилению противопожарной защиты объектов в пожароопасные периоды года;
- применять меры воздействия к лицам, нарушающим правила пожарной безопасности.

Руководители организаций приказом устанавливают порядок и сроки прохождения противопожарного инструктажа и занятий по пожарно-техническому минимуму, перечень объектов или профессий, работники которых должны проходить обучение по программе этих занятий, перечень должностных лиц, на которых возлагается проведение противопожарного инструктажа и занятий, а также место их проведения, порядок учета лиц, прошедших противопожарный инструктаж и обученных по программе пожарно-технического минимума.

Приказом должны создаваться постоянно действующие пожарно-технические комиссии (ПТК) в составе технического руководителя, главных специалистов объекта и других лиц по усмотрению руководителя. Комиссия не реже одного раза в полугодие проводит детальную проверку всех производственных, вспомогательных и административных зданий и помещений на территории организации с целью выявления нарушений стандартов, норм и правил пожарной безопасности и разрабатывает мероприятия по их устранению. Намеченные комиссией мероприятия оформляются актом, утверждаются руководителем и подлежат выполнению в установленные сроки.

Состав ПТК объявляется приказом руководителя организации. Руководство возлагается на заместителя руководителя объекта либо главного инженера. Как правило, в нее входят начальник пожарной службы (команды, дружины) объекта, инженерно-технические работники - энергетик, технолог, механик, инженер по технике безопасности, специалисты по водоснабжению, строительству, производственной и пожарной автоматике, других служб по усмотрению руководителя объекта.

Одной из главных функций ПТК является проведение детальных проверок противопожарного состояния всех производственных, складских, лабораторных, подсобных, административных и других помещений, а также территории организации с целью своевременного выявления и устранения нарушений стандартов, норм и правил пожарной безопасности:

- в технологических процессах производства;
- в работе машин, агрегатов, установок энергетического оборудования, систем отопления и вентиляции;
- при транспортировании и хранении перерабатываемого сырья, комплектующих изделий и готовой продукции, которые могут привести к возникновению пожара, взрыва или аварии. Проверки должны проводиться не реже двух раз в год.

По результатам проведенных проверок вырабатываются противопожарные мероприятия. При этом следует учитывать их экономический эффект, чтобы при минимальных затратах поддерживать необходимый и достаточный уровень пожарной безопасности организации. Намеченные комиссией мероприятия оформляются актом.

Для проведения мероприятий по предупреждению пожаров и их тушению создаются внештатные пожарные формирования, в частности

организуются добровольные пожарные дружины (ДПД) и боевые расчеты из числа рабочих, служащих, инженерно-технических работников организаций.

Численный состав пожарной дружины определяется из расчета 5 человек на каждые 100 работающих, но в целом не более численности, позволяющей в полном объеме и с хорошим качеством обеспечить выполнение возложенных на нее задач. При числе работающих в организациях менее 15 человек пожарная дружина не создается, а обязанности на случай пожара распределяются между работниками.

На добровольную пожарную дружину возлагаются определенные задачи, такие как контроль соблюдения противопожарного режима, проведение разъяснительной работы среди рабочих и служащих по соблюдению данного режима на рабочих местах и правил осторожного обращения с огнем в быту, надзор за исправностью средств пожаротушения и их укомплектованностью, вызов пожарной службы в случае возникновения пожара, принятие мер по его тушению имеющимися средствами пожаротушения.

Ответственность за обеспечение пожарной безопасности непосредственно на производственных участках, в цехах также несут руководители.

### **5.3 Обеспечение пожарной безопасности**

По взрывопожарной и пожарной опасности помещение в Гостинице «Бона» относится по ТКП 474-2013 к категории Д, с пониженной пожарной опасностью.

Обеспечение пожарной безопасности в Гостинице «Бона» регулируется законодательством и подразумевает проведение специальных мероприятий для предотвращения и ликвидации пожаров, а также для обеспечения безопасности людей и имущества.

На случай возникновения пожаров здания, сооружения и помещения они должны быть обеспечены первичными средствами пожаротушения:

- огнетушители;
- войлок, кошма.

Огнетушители предназначены для тушения пожаров в начальной стадии их возникновения. Они классифицируются:

- по видам огнетушащих средств;
- по объему корпуса;
- по способу подачи огнетушащих средств;
- по виду пусковых устройств.

По виду огнетушители бывают жидкостные, пенные, углекислотные, аэрозольные, порошковые, комбинированные.

Пенные огнетушители – предназначены для тушения твердых и жидких материалов и веществ огнетушащими пенами; химической или ВМП.

Пенные огнетушители всех типов, расположенные на улице или в неотапливаемом помещении, до наступления отрицательных температур должны быть перенесены в отапливаемое помещение, а на их месте установлены знаки с указанием их нового места расположения.

Огнетушители химические пенные (ОХП) применяются для тушения пожаров, за исключением электроустановок или щелочных металлов.

Порошковые огнетушители предназначены для тушения загораний бензина, дизельного топлива и других горючих жидкостей, электроустановок, находящихся под напряжением до 1000В, и применяются для оснащения легковых автомобилей, в быту и так далее.

Переносные огнетушители должны размещаться на расстоянии не менее 1,2 метра от проема двери и на высоте не более 1,5 метра от уровня пола, считая от низа огнетушителя.

Для указания местонахождения первичных средств пожаротушения используются знаки по ГОСТ 12.4.026. «Цвета сигнальные и знаки безопасности».

Для размещения первичных средств пожаротушения в производственных и других помещениях, а также на территории предприятия устанавливаются специальные пожарные посты (щиты).

Первичные средства пожаротушения расположены в коридорах, проходах, не препятствуют безопасной эвакуации людей. Их располагают на видных местах вблизи от выходов из помещений на высоте не более 1,5 м.

Расстояние от возможного очага возгорания до места размещения огнетушителя не должно превышать 20 метров.

В каждом кабинете имеется один огнетушитель ОУ-1 с диоксидом углерода, расположен рядом с выходной дверью на высоте метра от уровня пола.

Так же во всех помещениях установлены тепловые пожарные извещатели ИП 101-01-A2MS. Извещатель предназначен для обнаружения загораний, сопровождающихся появлением дыма в закрытых помещениях зданий и сооружений, и формирования электрического сигнала о возникшем пожаре и передачи его на приемно-контрольные приборы.

На главного инженера возложены обязанности по контролю за соблюдением противопожарного режима в помещениях, проведение разъяснительной работы среди работников предприятия по соблюдению противопожарного режима на рабочих местах.

При проектировании зданий предусматривают безопасную эвакуацию людей в случае возникновения пожара. Пути эвакуации называют проходы, коридоры, площадки, лестницы, ведущие к эвакуационному выходу, обеспечивающие безопасное движение людей в течение необходимого времени эвакуации. Выходы считаются эвакуационными, если они ведут:

- из помещений первого этажа непосредственно наружу или через вестибюль, коридор, лестничную клетку;



- из помещений любого этажа, кроме первого, в коридор, ведущий на лестничную клетку, или на лестничную клетку, имеющую выход непосредственно наружу или через вестибюль, отделенный от примыкающих коридоров перегородками с дверями;

- из помещения в соседнее помещение на том же этаже, обеспеченное выходами, указанными в подпунктах 1 и 2.

Процесс эвакуации людей из здания условно подразделяют на три этапа:

- движение людей от наиболее удаленного места их постоянного пребывания до эвакуационного выхода (например, движение по цеху);

- движение людей от эвакуационных выходов из помещения до выходов наружу (движение по коридорам или лестницам);

- движение людей от выходов из загоревшегося здания и рассеивание их по территории предприятия.

Для обеспечения безопасной эвакуации людей из помещений и зданий расчетное время эвакуации людей не должно превышать необходимого для этого времени. Расчетное время эвакуации людей устанавливают, исходя из времени движения одного или нескольких потоков через эвакуационные выходы от наиболее удаленных мест размещения людей. Весь путь движения людского потока разделяется на участки – проход, коридор, дверной проем, лестничный марш, тамбур. Начальными участками являются проходы между рабочими местами, оборудованием и тому подобное.

Расчетной время эвакуации людей определяют как сумму времени движения людского потока по отдельным участкам пути.

Для размещения первичных средств пожаротушения в производственных и других помещениях, а также на территории предприятия устанавливаются специальные пожарные посты (щиты).

На пожарных постах (щитах) размещаются только те первичные средства тушения пожаров, которые могут применяться в данном помещении, сооружении, установке.

Средства пожаротушения и пожарные посты окрашиваются в цвета по ГОСТ 12.4.026. «Цвета сигнальные и знаки безопасности».

Запорная арматура (кран, рычажные клапаны, крышки горловин) огнетушителей должны быть опломбированы. Использованные огнетушители, а также огнетушители с сорванными пломбами должны быть немедленно изъяты для проверки и перезарядки.

Емкость для хранения воды должны иметь объем не менее 200л и комплектоваться крышкой и ведром.

Ящики для песка должны иметь объем 0,5 кубических метров, 1 кубически мотор, 3,0 кубических метров и комплектоваться совковой лопатой. Перед заполнением ящика песком должен быть просеян и просушен.

Полотно, кошма должна иметь размеры 1х1 мотор(м); 2х1,5 м; 2х2 м, их следует хранить в металлических, пластмассовых футлярах с крышками.

В соответствии с п.6 ст.17 Закона Республики Беларусь от 15.06.1993 № 2403-ХІІ (ред. от 30.09.2019) "О пожарной безопасности" и Положением для

проведения профилактических мероприятий по предупреждению и тушению пожаров на предприятии организована пожарная дружина из числа работников предприятия в количестве 5 человек.

Основными задачами пожарной дружины на предприятии являются:

- контроль за соблюдением противопожарного режима;
- проведение разъяснительной работы среди работников по соблюдению противопожарного режима на рабочем месте и правил осторожного обращения с огнем в быту;
- надзор за исправностью средств пожаротушения и их укомплектованностью;
- вызов пожарной службы в случае возникновения пожара, принятие мер по его тушению имеющимися средствами пожаротушения.

Пожарная дружина состоит из командира (назначается из числа лиц администрации предприятия) в количестве 1 человека; старшего расчета (назначается из числа лиц администрации предприятия) в количестве 1 человека; бойца пожарной дружины (из числа работников предприятия) в количестве 3 человек.

Распределение обязанностей в дружине:

- командир пожарной дружины - руководит тушением пожара до прибытия пожарной службы;
- старший расчета - руководит тушением пожара до прибытия командира пожарной дружины или пожарной службы;
- бойцы пожарной дружины – вызывают пожарное аварийно-спасательное подразделение, оповещают о пожаре работников, организуют и проводят эвакуации людей в безопасное место, проверяют численность эвакуируемых, встречают пожарное аварийно-спасательное подразделение, участвуют в ликвидации очага пожара.

#### **5.4 Охрана окружающей среды**

Охрана окружающей среды на предприятии Гостинице «Бона» является одной из важных задач, которая получает достаточно серьезное внимание со стороны руководства.

В рамках охраны окружающей среды на предприятии проводятся следующие мероприятия:

- оценка влияния производственной деятельности на окружающую среду. Это позволяет прогнозировать возможные экологические последствия и принимать меры по их предотвращению;
- установление стандартов и нормативов на выбросы загрязняющих веществ. Это позволяет контролировать уровень загрязнения воздуха, воды и почвы;
- разработка и внедрение технологических решений, направленных на сокращение выбросов. Это может быть использование безопасных для

окружающей среды сырья и материалов, снижение энергопотребления, установка очистных сооружений;

- обучение персонала. Необходимо обучать сотрудников предприятия правилам охраны окружающей среды, технологиям и способам снижения воздействия на окружающую среду.

- внедрение системы мониторинга. Необходимо проводить регулярный контроль качества воздуха, воды, почвы на территории предприятия и в окружающей среде;

- сотрудничество с государственными организациями. Необходимо сотрудничать с органами государственного контроля в области охраны окружающей среды для совместной работы по снижению негативного воздействия на окружающую среду.

В проблеме антропогенного (техногенного) изменения окружающей среды особое место занимает загрязнение атмосферы, что связано с рядом обстоятельств. Все эти изменения могут наносить вред не только окружающей среде, но и здоровью людей и животных, а также приводить к вымиранию видов и ухудшению качества жизни в регионе.

Для предотвращения подобных последствий, необходимо принимать меры по уменьшению выбросов загрязняющих веществ в окружающую среду, включая оборудование производственного процесса современной экологически чистой техникой, поддерживать экологический баланс в природной среде и использовать альтернативные источники энергии в производственных целях..

Существует несколько методов очистки сточных вод, которые могут использоваться в различных условиях и ситуациях. Вот некоторые из них:

Биологическая очистка - это процесс, в ходе которого микроорганизмы разлагают загрязнители в сточных водах. Метод может проводиться с использованием разных технологий, включая активный и биологический ил.

Физико-химическая очистка - этот метод основывается на физико-химических процессах, которые удаляют загрязнители путем их осаждения. Также могут использоваться системы фильтрации и коагуляции.

Химическая очистка - этот метод использует химические реакции, чтобы удалять различные загрязнители из сточных вод. Этот метод включает использование различных химических реагентов, таких как уголь, железные соли и многие другие.

Активные фильтры - этот метод включает использование специальных фильтров, которые содержат активный агент, способный удалять загрязнители из сточных вод.

## **6 Экономический раздел**

### **6.1 Технико-экономическое обоснование разработки программного средства**

Один из наиболее важных аспектов работы отеля – это прием, размещение и обслуживание гостей. Это включает работу персонала, который занимается бронированием номеров, приемом гостей, проведением регистрации, предоставлением дополнительных услуг в виде питания и услуг отеля. Важным аспектом является также скорость предоставления этих услуг.

С внедрением программного средства процессы бронирования и предоставления дополнительных услуг отелем упростятся и ускорятся, что позволит принимать большее количество клиентов за меньшее время.

В технико-экономическом обосновании будут рассмотрены следующие вопросы:

- внедрение приложения для ведения отеля позволит уменьшить время, затраченное на обработку данных;
- опеределение структуры (этапов) работы по созданию программного продукта:
  - обзор рынка и маркетинговые исследования;
  - сбор требований заказчика;
  - разработка технического задания;
  - проектирование интерфейса;
  - разработка программного кода;
  - тестирование и отладка;
  - запуск и поддержка;
- расчет затрат на разработку программного продукта включает в себя затраты на оборудование, затраты на электроэнергию и прочие затраты.
- расчет экономического эффекта у разработчика и заказчика программного продукта включает в себя прибыль от продаж на рынке, окупаемость затрат.
- окупаемость затрат на создание продукта: менее 1 месяца после запуска продажи на рынке ИТ.

### **6.2 Составление плана по разработке программного средства**

План разработки программного средства представлен в таблице 6.1.

Таблица 6.1 - План разработки программного продукта

Наименование этапов и видов работ	Исполнитель (должность, квалификация)	Количество исполнителей	Трудоемкость, человеко-дни
Инициация	Техник-программист	1	2
Планирование	Техник-программист	1	2
Разработка	Техник-программист	1	30
Реализация и тестирование	Техник-программист	1	4
Мониторинг и завершение проекта	Техник-программист	1	2

На рисунке 6.1 отображена трудоёмкость, затраченная на разработку дипломного проекта на разных этапах.

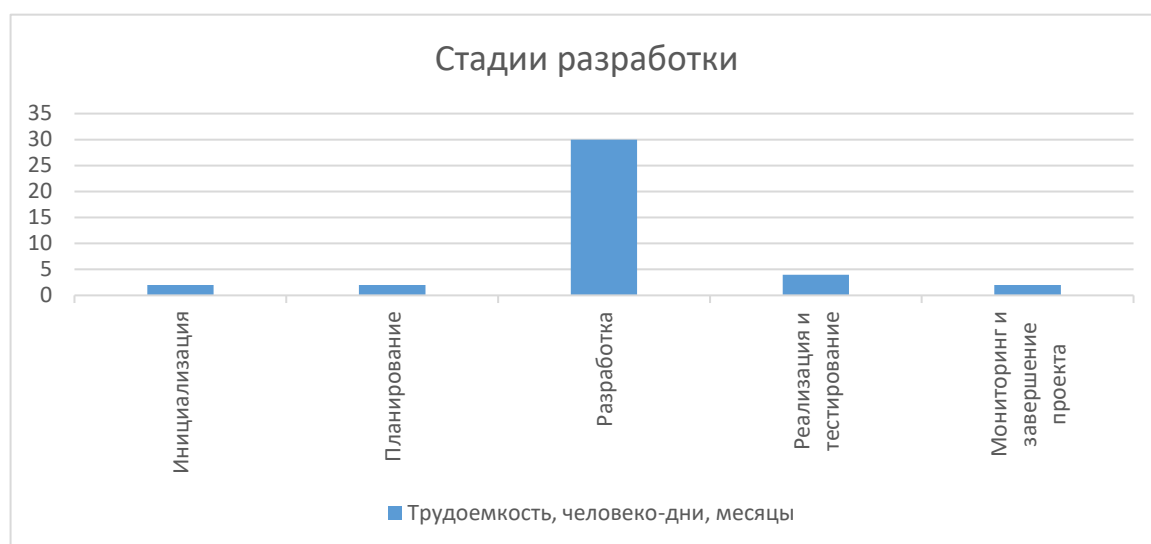


Рисунок 6.1 – Гистограмма распределения времени работы при разработке программного продукта

### 6.3 Расчет затрат на разработку программного продукта

Стоимостная оценка программного продукта и определение экономического эффекта у разработчика предполагает расчет затрат, которые включают следующие элементы:

- материальные затраты;
- затраты на оплату труда;

- отчисления на социальные нужды;
- амортизация основных средств и нематериальных активов;
- прочие затраты.

В материальных затратах отражаются затраты на материалы и принадлежности, необходимые для проведения научно-исследовательских работ, магнитные носители, бумагу, красящие ленты и другие материалы, необходимые для разработки программного продукта. Затраты определяются по действующим отпускным ценам. Здесь так же отражаются затраты на электроэнергию.

Расчет осуществляется по формуле (6.1).

$$P_m = K_{\text{Тзр}} \sum_{i=1}^n H_{pi} C_i, \quad (6.1)$$

где  $K_{\text{Тзр}}$  – коэффициент, учитывающий транспортно-заготовительные расходы (в работе принимаем от 1,05 до 1,10);

$H_{pi}$  – норма расхода  $i$ -го вида материалов на макет или опытный образец;

$C_i$  – действующая отпускная цена за единицу  $i$ -го вида материала, руб.;

$n$  – количество применяемых видов материалов.

$$P_m = 1,10 \times (1,10 + 7,50 + 1,25 + 2,40 + 16,25 + 15) = 47,85 \text{ руб.}$$

Расчет целесообразно представить в табличной форме. Результат представлен в таблице 6.2.

Таблица 6.2 – Расчет материальных затрат

Наименование материалов	Единица измерения	Количество	Цена приобретения руб.	Сумма, руб.
Ручка шариковая	штука	1	1,10	1,10
Дырокол	штука	1	7,50	7,50
Папка-скоросшиватель	штука	1	1,25	1,25
Диск	штука	1	2,40	2,40
Бумага	пачка	1	16,25	16,25
Картридж	штука	1	15,00	15,00
Всего расходов				43,50
Всего с транспортно-заготовительными расходами				47,85

Затраты на электроэнергию находятся исходя из продолжительности периода разработки программного обеспечения, количества кВт/ч,

затраченных на его проектирование и тарифа за 1 кВт/ч по следующей формуле (6.2).

$$P_{\text{э}} = K_{\text{э}} \times T_{\text{р}}, \quad (6.2)$$

где  $K_{\text{э}}$  – стоимость одного кВт/ч, руб.;

$T_{\text{р}}$  – количество кВт/ч.

Тарифы на электроэнергию применяются согласно приложению к Декларации «Об уровне тарифов на электроэнергию, отпускаемую РУП Электроэнергетики ГПО «Белэнерго» для юридических лиц и ИП» на соответствующий период времени, когда разрабатывается программный продукт. Время реализации проекта 40 дней, среднее потребление энергии в месяц составляет 75 кВт/ч, то есть необходимое потребление энергии в дипломном проекте принимается 143 кВт/ч.

$$P_{\text{э}} = 0,43912 \times 143 = 62,79 \text{ руб.}$$

Величина затрат исчисляется исходя из численности различных категорий исполнителей и трудоемкости выполнения отдельных видов работ, премиальных систем оплаты труда исполнителей по формуле (6.3).

$$P_{\text{оз}} = K_{\text{пр}} \sum_{i=1}^n T_{\text{ci}} t_{\text{fi}}, \quad (6.3)$$

где  $T_{\text{ci}}$  – тарифная ставка за день (месячный оклад)  $i$ -й категорий работников;

$t_{\text{fi}}$  – время фактической работы работника  $i$ -й категории по теме, дн.

или мес.;

$K_{\text{пр}}$  – коэффициент премий по премиальным системам,  $K_{\text{пр}} \approx$  от 1,10 до 1,30.

Расчет данных прямых расходов также целесообразно представить в табличной форме. Результат представлен в таблице 6.3. Для этого необходимо рассчитать заработную плату исполнителя, исходя из тарифной ставки 1 разряда, которая в организации составляет 197,00 руб. Разряд техника программиста 7, тарифный коэффициент – 2,03.

Таблица 6.3 – Расчет затрат на основную заработную плату персонала

Наименование категорий работников и должностей	Количество штатных единиц, чел.	Заработная плата за 1 месяц, руб.	Трудовзатраты, дн. или мес.	Сумма, руб.
Техник-программист	1	399,91	2	799,82
Всего				799,82
Всего с коэффициентом премий				879,80

Определяется дополнительная заработная плата исполнителей, включающая разнообразные предусмотренные трудовым законодательством выплаты, по формуле (6.4).

$$P_{\text{дз}} = P_{\text{оз}} \times \frac{H_{\text{дз}}}{100}, \quad (6.4)$$

где  $H_{\text{дз}}$  – норматив дополнительной заработной платы,  $H_{\text{дз}} \approx$  от 10 до 20 %.

$$P_{\text{дз}} = 879,80 \times \frac{10}{100} = 87,98 \text{ руб.}$$

Отражается сумма обязательных страховых взносов, взносов на профессиональное пенсионное страхование в бюджет государственного внебюджетного фонда социальной защиты населения Республики Беларусь.

Рассчитываются отчисления органам социальной защиты по формуле (6.5).

$$P_{\text{ос}} = (P_{\text{оз}} + P_{\text{дз}}) \frac{H_{\text{ос}}}{100}, \quad (6.5)$$

где  $H_{\text{ос}}$  – норма отчислений на социальную защиту,  $H_{\text{ос}} = 34$  %.

$$P_{\text{ос}} = (879,80 + 87,98) \frac{34}{100} = 329,05 \text{ руб.}$$

Также рассчитываются отчисления на страхование от несчастных случаев на производстве и профессиональных заболеваний ( $P_{\text{стр}}$ ) по ставке действующего законодательства ( $H_{\text{бгс}} = 0,3\% - 1\%$ ). Для расчетов среднее значение  $H_{\text{бгс}}$  принимается равным 0,6 %, отчисления на страхование от несчастных случаев вычисляется по формуле (6.6).

$$P_{\text{стр}} = (P_{\text{оз}} + P_{\text{дз}}) \frac{H_{\text{бгс}}}{100} \quad (6.6)$$

$$P_{\text{стр}} = (879,80 + 87,98) \frac{0,6}{100} = 5,81 \text{ руб.}$$

Амортизация основных средств и нематериальных активов рассчитываются в установленном законодательством Республики Беларусь порядке, по одному из методов начисления амортизации (линейный, нелинейный, производительный), исходя из целесообразности его применения.

По статье «Амортизация основных средств и нематериальных активов» рассчитываются амортизационные отчисления (АО), исходя из стоимости основных средств (ОС), используемых в процессе разработки программного обеспечения, сроков эксплуатации оборудования ( $T_{\text{с}}$ ) и годовой нормы амортизации ( $H_{\text{а}}$ ).

Для определения затрат по данному элементу будет использоваться линейный способ начисления амортизации. Нормативные сроки службы машин и оборудования составляют пять лет.

Норма амортизации для линейного способа начисления вычисляется по формуле (6.7).



$$H_a = \frac{1}{T_c} \times 100, \quad (6.7)$$

где  $T_c$  – срок службы оборудования, лет.

$$H_a = \frac{1}{5} \times 100 = 20 \, \%.$$

Для линейного способа начисления амортизационные отчисления равномерно распределены на весь период службы оборудования и вычисляются на один год. Так как разработка программного продукта длилась 40 дней, то сумма амортизационных отчислений (АО) за этот период составит, линейный способ начисления амортизационные вычисляется по формуле (6.8).

$$P_{ao} = \frac{OC \times H_a \times 40}{12 \times 30}, \quad (6.8)$$

где  $OC$  – стоимость основных средств.

$$P_{ao} = \frac{4000 \times 0,2 \times 40}{12 \times 30} = 88,89 \text{ руб.}$$

Прочие затраты на конкретное программное обеспечение включают затраты на: арендную плату; вознаграждения за рационализаторские предложения и выплата авторских гонораров; затраты на гарантийный ремонт и обслуживание изделий; начисленные налоги, сборы (пошлины), платежи, включаемые в затраты на производство продукции (работ, услуг); связь; на оплату услуг рекламы и маркетинга и пр.

В расчетах принимаем условно размер прочих затрат равных от 10 до 30 % от суммы всех остальных затрат на разработку, вычисляется по формуле (6.9).

$$P_{пр} = P_{оз} \times \frac{H_{пр}}{100}, \quad (6.9)$$

где  $H_{пр} \approx$  от 20 до 30%.

$$P_{пр} = 879,80 \times \frac{25}{100} = 219,95 \text{ руб.}$$

На основании полученных данных по отдельным статьям затрат рассчитывается общая сумма затрат на разработку программного продукта.

Общая сумма затрат по элементам на разработку программного продукта рассчитывается по формуле (6.10).

$$Z = P_m + P_э + P_{оз} + P_{дз} + P_{ос} + P_{стр} + P_{ao} + P_{пр} \quad (6.10)$$

$$\begin{aligned} Z &= 47,85 + 62,79 + 879,80 + 87,98 + 320,05 + 5,81 + 88,89 + 219,95 \\ &= 1713,12 \text{ руб.} \end{aligned}$$

Результаты расчетов представлены в таблице 6.4.

Таблица 6.4 – Расчет затрат на разработку программного продукта

Элемент затрат	Затраты, руб.
Материальные затраты ( $P_m$ )	47,85
Электроэнергия ( $P_э$ )	62,79
Затраты на оплату труда ( $P_{оз}$ )	879,80
Дополнительная заработная плата ( $P_{дз}$ )	87,98
Отчисления органам социальной защиты ( $P_{ос}$ )	320,05
Отчисления на страхования от несчастных случаев ( $P_{стр}$ )	5,81
Амортизация ( $P_{Ao}$ )	88,89
Прочие затраты ( $P_{пр}$ )	219,95
Общая сумма затрат ( $\Sigma$ )	1713,12

Таблицу 6.4 удобно представить в виде диаграммы, отображающей элементы затрат программного продукта. Диаграмма представлена на рисунке 6.2.



Рисунок 6.2 – Структура затрат

Исходя из расчётов затрат на разработку программного продукта, можно сделать вывод, что самыми затратными являются выплаты зарплаты сотруднику и отчисление органам социальной защиты. Также не мало средств уходит на прочие затраты и выплату дополнительной заработной платы

сотруднику. Незначительное количество средств уходит на электроэнергию, материальные расходы и амортизацию.

#### 6.4 Экономическая эффективность разработки программного продукта

Заказчик оплачивает разработчику всю сумму расходов по проекту, включая прибыль. После уплаты налогов из прибыли в распоряжении заказчика остается чистая прибыль от проекта. Ввиду того, что программное обеспечение разрабатывается для одного объекта, чистую прибыль можно считать в качестве экономического эффекта разработчика от реализованного программного продукта.

В дипломном проекте отпускная цена программного продукта, представляет собой не цену за единицу продукции, а цену проекта, за которую его можно продать и получить определённую выгоду.

Отпускная цена продукции формируется исходя из плановой себестоимости производства продукции, всех видов установленных налогов и прибыли, а также качества, потребительских свойств продукции и конъюнктуры рынка.

С учетом действующих в республике нормативных документов отпускная цена на продукцию рассчитывается по формуле (6.11).

$$ОЦ = З + П, \quad (6.11)$$

где ОЦ – отпускная цена разработчика, руб.;

З – затраты на разработку, руб.;

П – прибыль, руб.

$$ОЦ = 1713,12 + 342,62 = 2055,74 \text{ руб.}$$

Прибыль рассчитывается по следующей формуле (6.12).

$$П = \frac{З \times R}{100}, \quad (6.12)$$

где R – уровень рентабельности (от 10 до 30 %).

$$П = \frac{1713,12 \times 20}{100} = 342,62 \text{ руб.}$$

Стоимость проекта с учётом НДС, представляет собой сумму отпускной цены и налога на добавленную стоимость, рассчитывается по формуле (6.13).

$$НДС = \frac{(З + П) \times \text{СтавкаНДС}}{100}, \quad (6.13)$$

где СтавкаНДС – 20 %.

$$НДС = \frac{(1713,12 + 342,62) \times 20}{100} = 411,15 \text{ руб.}$$

Отпускная цена с учетом НДС, рассчитывается по формуле (6.14).

$$ОЦ_{с\text{ НДС}} = ОЦ + НДС \quad (6.14)$$

$$ОЦ_{с\text{ НДС}} = 2055,74 + 411,15 = 2466,89 \text{ руб.}$$

Таким образом, разработчик программного продукта может продать заказчику программный продукт, что покроет затраты и обеспечит прибыль за разработку проекта.

### **6.5 Экономическая эффективность у пользователя программного продукта**

Экономический эффект у пользователя программного продукта выражается в виде экономии трудовых, материальных и финансовых ресурсов, получаемой от таких показателей как:

- повышение производительности сервиса (увеличение числа выводимых за единицу времени документов, уменьшение среднего времени подготовки отчета и т.д.), что выражается в снижении трудоемкости выполнения операций, решении задач, подготовки данных, обработки информации и анализа результатов;

- сокращение затрат на оплату машинного времени и расходных материалов;

- повышения уровня сервиса (сокращение времени на устранение инцидентов);

- улучшения показателей основной деятельности предприятия в результате использования программного продукта и т.д.

При сравнении базового и нового варианта программного продукта в качестве экономического эффекта будет выступать общая сумма экономии всех видов ресурсов относительно базового варианта.

Экономия затрат на заработную плату ( $\Theta_3$ ) при использовании нового программного продукта в расчете на объем выполненных работ определяется по формуле (6.15).

$$\Theta_3 = \Theta_{3в} \times Q, \quad (6.15)$$

где  $\Theta_{3в}$  – экономия затрат на заработную плату при решении задач с использованием нового программного продукта в расчете на 1 задачу, руб.;

$Q$  – количество задач, решаемых за год с использованием нового программного продукта (задач).

$$\Theta_3 = 1,91 \times 2000 = 3820,00 \text{ руб.}$$

Экономия затрат на заработную плату в расчете на одну задачу ( $\Theta_{3в}$ ), рассчитывается по формуле (6.16).

$$\Theta_{3в} = \frac{3_{см} \times (T_{с1} - T_{с2}) \div T_{ч}}{D_p}, \quad (6.16)$$

где  $Z_{см}$  – среднемесячная заработная плата разработчика программного продукта, руб.;

$T_{с1}$ ,  $T_{с2}$  – снижение трудоемкости на одну задачу в базовом и новом варианте соответственно, человеко – часов (соответственно 1,71 и 0,87 чел-час);

$T_{ч}$  – количество часов работы в день, часов (8 часов);

$D_p$  – среднемесячное количество рабочих дней.

$$\mathcal{E}_{зв} = \frac{399,91 \times (1,71 - 0,87) \div 8}{22} = 1,91 \text{ руб.}$$

Экономия с учетом начислений на заработную плату ( $\mathcal{E}_н$ ), рассчитывается по формуле (6.17).

$$\mathcal{E}_н = \mathcal{E}_з \times K_{нз}, \quad (6.17)$$

где  $K_{нз}$  – коэффициент начислений на заработную плату принимаем 1.2.

$$\mathcal{E}_н = 3820,00 \times 1,2 = 4584,00 \text{ руб.}$$

Экономия за счет сокращения простоя сервиса ( $\mathcal{E}_с$ ), рассчитывается по формуле (6.18).

$$\mathcal{E}_с = \frac{(P_1 - P_2) \times D_{пр} \times C_p}{60}, \quad (6.18)$$

где  $P_1$ ,  $P_2$  – время простоя сервиса, обусловленное программным продуктом в день, мин;

$D_{пр}$  – плановый фонд работы сервиса (дн.);

$C_p$  – стоимость одного часа работника, для которого разрабатывается программный продукт, руб.

$$\mathcal{E}_с = \frac{(19 - 10) \times 255 \times 2,28}{60} = 87,21 \text{ руб.}$$

Общая экономия от применения нового программного продукта рассчитывается по формуле (6.19)

$$\mathcal{E}_о = \mathcal{E}_н + \mathcal{E}_с \quad (6.19)$$

$$\mathcal{E}_о = 4584,00 + 87,21 = 4671,21 \text{ руб.}$$

Внедрение нового программного продукта позволит сэкономить на текущих затратах, т.е. практически получить на эту сумму дополнительную прибыль. Для пользователя в качестве экономического эффекта выступает лишь чистая прибыль – дополнительная прибыль, остающаяся в его распоряжении, которая определяется по формуле (6.20).

$$П_ч = \mathcal{E}_о - \frac{\mathcal{E}_о \times H_p}{100} \quad (6.20)$$

$$П_ч = 4584,00 - \frac{4671,21 \times 20}{100} = 3649,80 \text{ руб.}$$

## **6.6 Экономическая эффективность разработки**

В данном разделе представлено экономическое обоснование для дипломного проекта по теме: приложение отеля «Бона».

В технико-экономическом обосновании были рассмотрены следующие вопросы:

- составление плана по разработке программы;
- расчет стоимости разработки.

В техно-экономическом обосновании был рассчитан срок разработки по созданию программного средства. Он составил 320 часов, а также была рассчитана стоимость программного продукта, которая составила 2466,88 руб.

Основное преимущество разработки программного обеспечения состоит в пользе для персонала отеля. Так как можно ускорить процесс оказания услуг посетителям отеля. Также отель может получить дополнительную экономию за год при сокращении сотрудника, в размере 3918,00 руб.

Разработка программы актуальна, так как это облегчит работу персонала отеля и ускорит процесс оказания услуг. Внедрение программного средства также увеличит работоспособность работников отеля. Таким образом, можно понять, что это приложение является очень полезным инструментом в наше время.

## Заключение

В рамках дипломного проектирования было разработано программное средство на тему «Создание программного средства для автоматизации работы администратора гостиницы «Бона».

Для достижения цели дипломного проектирования были решены следующие задачи:

- произведен объектно-ориентированный анализ и проектирование приложения;
- описаны общие сведения приложения;
- выбрана методика испытаний;
- описан процесс тестирования;
- приведены примеры области применения.

В программе реализованы такие задачи как ведение базы данных: удаление, добавление и редактирование записей, выдача чека после бронирования, просмотр прайс-листов на услуги отеля.

Разработка имеет интуитивно понятный графический интерфейс, позволяющий даже с минимальными знаниями компьютера пользоваться приложением.

Разработанная программа упростит работу администратора отеля, ускорит работу с бронированием и оказанием дополнительных услуг посетителям отеля.

Программное средство имеет ряд достоинств: небольшой объем памяти, занимаемый приложением на различных носителях информации, быстрое действие.

В процессе разработки программы использовался в большом объеме материал по программированию, применены и закреплены знания по уже изученному материалу, отработаны навыки владения методами надежного программирования и эффективности разработки программного обеспечения в Microsoft Visual Studio 2019 с использованием языка программирования C#, разработана база данных средствами системы управления базами данных Microsoft SQL Server 2019.

В разделе охраны труда были рассмотрены вопросы:

- правовые, нормативные, социально – экономические и организационные вопросы охраны труда;
- влияние особенностей трудовой деятельности на здоровье и работоспособность человека;
- пожарная безопасность;
- охрана окружающей среды.

В экономическом разделе была рассчитана стоимость разработки программного средства, которая составила 3918,00 рублей.

Программное средство может быть дополнено и модернизировано.

## Список использованных источников

- 1 Михнюк, Т.Ф. Охрана труда / Т.Ф. Михнюк. – Минск : ИВЦ Минфина, 2009. – 365 с.
- 2 Объектно-ориентированный анализ и проектирование с примерами приложений / Гради Буч [и др.]. – 3-е изд. – М. : ООО «И.Д. Вильямс», 2008. – 720 с.
- 3 Экономика предприятия. Практикум / Э. В. Крум [и др.] ; под ред. Э. В. Крум. – Минск : Издательство Гревцова, 2009. – 355 с.
- 4 Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995. – 84 с.
- 5 Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.301-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 14 с.
- 6 Текст программы. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.401-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 16 с.
- 7 SQL [Электронный ресурс]. – Microsoft SQL 2022 – Режим доступа: <https://support.microsoft.com/ru-ru/office/>. – Дата доступа : 17.04.2023.
- 8 Draw.io [Электронный ресурс]. – startpack 2023 – Режим доступа: <https://startpack.ru/application/draw-io> – Дата доступа : 16.04.2023.
- 9 SQL Server Management Studio [Электронный ресурс] - SQL Server Режим доступа: <https://learn.microsoft.com/en-us/sql/ssms>. Дата доступа : 21.04.2023.
- 10 C# [Электронный ресурс]. – Visual Studio 2022 – Режим доступа <https://learn.microsoft.com/en-us/dotnet/csharp/> – Дата доступа : 16.04.2023.
- 11 Dr. Explain [Электронный ресурс]. – Dr.Explain, 2022 – Режим доступа <https://www.drexplain.ru/>. – Дата доступа 12.04.2023.
- 12 Microsoft Office 2021 [Электронный ресурс]. – Microsoft, 2022 – Режим доступа : <https://support.microsoft.com/ru-ru/office/>. – Дата доступа 11.04.2023.



## ПРИЛОЖЕНИЕ А

### (обязательное)

#### Текст программных модулей

```
/// Код программы
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace HotelBona
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void button_Click(object sender, RoutedEventArgs e) // кнопка
        бронирования
        {
            NewReservation newReservationForm = new NewReservation();
            newReservationForm.Show();
            this.Close();
        }

        private void button_Copy_Click(object sender, RoutedEventArgs e) // кнопка для
        перехода к бронированию
        {
            NewReservation newReservationForm = new NewReservation();
            newReservationForm.Show();
            this.Close();
        }

        private void btnPricing_Click(object sender, RoutedEventArgs e) // кнопка для
        просмотра прайс листа
        {
            RoomPricing form = new RoomPricing();
            form.Show();
            this.Close();
        }

        private void button_Copy1_Click(object sender, RoutedEventArgs e) // кнопка для
        добавления услуг
        {
            RoomServices form = new RoomServices();
            form.Show();
        }
    }
}
```

```

        this.Close();
    }

    private void button_Copy4_Click(object sender, RoutedEventArgs e) // кнопка для
просмотра информации о комнате
    {
        RoomInformation form = new RoomInformation();
        form.Show();
        this.Close();
    }

    private void button_Copy5_Click(object sender, RoutedEventArgs e) // кнопка для
просмотра информации о клиенте отеля
    {
        GuestInformation form = new GuestInformation();
        form.Show();
        this.Close();
    }

    /*private void button_Copy2_Click(object sender, RoutedEventArgs e)
    {
        ReportsMenu form = new ReportsMenu();
        form.ShowDialog();
    }*/

    private void btnExistingTransaction_Click(object sender, RoutedEventArgs e) //
кнопка для просмотра предыдущих бронирований гостя
    {
        ExistingReservation form = new ExistingReservation();
        form.ShowDialog();
        this.Close();
    }

    private void Button_Click_1(object sender, RoutedEventArgs e) // метод загрузки
бд гостей
    {
        Guestt form = new Guestt();
        form.ShowDialog();
        this.Close();
    }
    public partial class Guestt : Window
    {
        string id = "";
        string type = "";
        string sql = "";
        SqlConnection conn = new SqlConnection("Data Source=WIN-
U4EJB3PFQL8\\SQLEXPRESS;Initial Catalog=KingWilliamHotelDB;Integrated Security=True");

        public Guestt()
        {
            InitializeComponent();
        }
        public Guestt(string id, string type)
        {
            sql = "select serviceTransactionID as'Transaction ID', servicedescription
as 'Description', servicedate as Date,amount as Amount from tblServicesTransactions inner
join tblServices on tblServicesTransactions.ServiceID = tblServices.ServiceId where
ReservationID = '" + 1 + "'";

            //sql = "select orderID as'Transaction ID', Transactiondate as Date, Cost
as Amount from tblRestaurantTransactions where ReservationID = '" + 1 + "'";

```

```

        InitializeComponent();
        this.id = id;
        SqlCommand tr = new SqlCommand(sql, conn);
        SqlDataAdapter dataAdapter = new SqlDataAdapter(tr); //c.con is the
connection string

        DataTable dtRecord = new DataTable();
        dataAdapter.Fill(dtRecord);

    }

    private void Window_Loaded(object sender, RoutedEventArgs e) // метод загрузки бд
    {
        GuestGrid.ItemsSource = AppData.db.tblGuests.ToList();

    }

    private void Button_Click(object sender, RoutedEventArgs e) // просмотр
информации о госте
    {
        GuestInformation form = new GuestInformation();
        form.Show();
        this.Close();
    }

    private void Button_Click_1(object sender, RoutedEventArgs e)
    {

    }

    private void btnUpdate_Click(object sender, RoutedEventArgs e)
    {
        double price = 0;
        if(double.TryParse(txtPrice.Text,out price) && price > 0)
        {
            conn.Open();

            if (cmbRoomType.SelectedValue.ToString() == "Suite")
            {
                SqlCommand update = new SqlCommand("UPDATE tblRooms SET Cost='"
+ txtPrice.Text + "' where RoomID = '" + cmbRoomNumber.Text + "'", conn);
                int r = update.ExecuteNonQuery();
                if (r > 0)
                {
                    MessageBox.Show("Бронирование обновлено успешно!");
                    MainWindow main = new MainWindow();
                    main.Show();
                    this.Close();
                }
            }
            else
            {
                SqlCommand update = new SqlCommand("Update tblrooms set cost =
'" + txtPrice.Text + "' where RoomTypeID = ( select top 1 tblrooms.roomtypeid from
tblrooms inner join tblroomtype on tblrooms.RoomTypeID = tblroomtype.RoomTypeID
where Typedescription = '" + cmbRoomType.Text + "')", conn);
                int r = update.ExecuteNonQuery();
                if (r > 0)
                {
                    MessageBox.Show("Бронирование обновлено успешно!");
                    MainWindow main = new MainWindow();
                    main.Show();
                    this.Close();
                }
            }
        }
    }

```

```

    }
}
else
{
}

private void cmbRoomType_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    conn.Open();
    SqlDataReader reader = null;
    if (cmbRoomType.SelectedValue.ToString() == "Suite")
    {
        txtPrice.Text = "";
        cmbRoomNumber.Items.Clear();
        cnvRoomNumber.Visibility = Visibility.Visible;
        cnvPricing.Margin = new Thickness(451, 509, 533.2, 215.2);

        SqlCommand number = new SqlCommand("select roomID from tblRooms
inner join tblRoomType on tblRooms.RoomTypeID = tblRoomType.RoomTypeID where
TypeDescription ='Suite'", conn);
        reader = number.ExecuteReader();

        while (reader.Read())
        {
            cmbRoomNumber.Items.Add(reader["roomID"].ToString());
        }
        reader.Close();
        btnUpdate.Visibility = Visibility.Collapsed;
    }
    else
    {
        double money = 0;
        cnvRoomNumber.Visibility = Visibility.Collapsed;
        cnvPricing.Margin = new Thickness(451, 421, 533.2, 303.2);
        SqlCommand type = new SqlCommand("SELECT TOP 1 Cost FROM tblRooms
inner join tblRoomType on tblRooms.RoomTypeID = tblRoomType.RoomTypeID where
TypeDescription =' " + cmbRoomType.SelectedValue.ToString() + "'", conn);
        var cost = type.ExecuteScalar();
        if(cost!= null)
        {
            money = double.Parse(cost.ToString());
        }
        txtPrice.Text = money.ToString("#.##");
        btnUpdate.Visibility = Visibility.Visible;
    }

    conn.Close();
}

private void cmbRoomNumber_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    if(cmbRoomNumber.SelectedIndex != -1)
    {
        conn.Open();
        SqlCommand num = new SqlCommand("Select cost from tblRooms where
RoomID = '" + cmbRoomNumber.SelectedValue.ToString() + "'", conn);
        double result = 0;
        var output = num.ExecuteScalar();
        if (output != null)
    }
}

```

```

        {
            result = double.Parse(output.ToString());
        }
        txtPrice.Text = result.ToString("#.##");
        btnUpdate.Visibility = Visibility.Visible;
        conn.Close();
    }
    else
    {
        btnUpdate.Visibility = Visibility.Collapsed;
    }
}

private void btnUpdate_Click(object sender, RoutedEventArgs e)
{
    double price = 0;
    if(double.TryParse(txtPrice.Text,out price) && price > 0)
    {
        conn.Open();

        if (cmbRoomType.SelectedValue.ToString() == "Suite")
        {
            SqlCommand update = new SqlCommand("UPDATE tblRooms SET Cost='"
+ txtPrice.Text + "' where RoomID = '" + cmbRoomNumber.Text + "'", conn);
            int r = update.ExecuteNonQuery();
            if (r > 0)
            {
                MessageBox.Show("Reservation Updated successfully!");
                MainWindow main = new MainWindow();
                main.Show();
                this.Close();
            }
        }
        else
        {
            SqlCommand update = new SqlCommand("Update tblrooms set cost =
'" + txtPrice.Text + "' where RoomTypeID = ( select top 1 tblrooms.roomtypeid from
tblrooms inner join tblroomtype on tblrooms.RoomTypeID = tblroomtype.RoomTypeID
where Typedescription = '" + cmbRoomType.Text + "')", conn);
            int r = update.ExecuteNonQuery();
            if (r > 0)
            {
                MessageBox.Show("Reservation Updated successfully!");
                MainWindow main = new MainWindow();
                main.Show();
                this.Close();
            }
        }
    }
    else
    {
        MessageBox.Show("Enter numbers for price and greater than 0");
    }
}

private void btnCharge_Click(object sender, RoutedEventArgs e)
{
    if(txtRoomID.Text == "" || txtAmount.Text=="")
    {
        MessageBox.Show("Заполните всю информацию");
    }
    else
    {
        conn.Open();
    }
}

```

```

        SqlCommand id = new SqlCommand("select reservationID from
tblReservations where RoomID = '" + txtRoomID.Text + "' and ReservationEndDate >
CONVERT (date, SYSDATETIME()) and ReservationStartDate <= CONVERT (date,
SYSDATETIME())", conn);
        var rid = id.ExecuteScalar();
        if(rid == null)
        {
            MessageBox.Show("Номер - " + txtRoomID.Text + " Никем не
забронирован. Выберите другой номер.");
        }
        else
        {
            if(rdbRestaurant.IsChecked == true)
            {
                SqlCommand insert = new SqlCommand("insert into
tblRestaurantTransactions (ReservationID,cost,TransactionDate) values ('" +
rid.ToString() + "','" + txtAmount.Text + "','CONVERT (DATE,SYSDATETIME())'", conn);
                int r = insert.ExecuteNonQuery();
                if(r > 0)
                {
                    MessageBox.Show("Транзакция прошла!");
                    MainWindow n = new MainWindow();
                    n.Show();
                    this.Close();
                }
                else
                {
                    MessageBox.Show("Транзакция не прошла!");
                }
            }
            else
            {
                int service = cmbService.SelectedIndex;
                SqlCommand insert = new SqlCommand("insert into
tblServicesTransactions (ReservationID,ServiceID,ServiceDate) values ('" +
rid.ToString() + "','" + service++ + "','CONVERT (DATE,SYSDATETIME())'", conn);

                int r = insert.ExecuteNonQuery();
                if (r > 0)
                {
                    MessageBox.Show("Транзакция прошла!");
                }
                else
                {
                    MessageBox.Show("Транзакция не прошла!");
                }
            }
        }

        conn.Close();
    }

    private void comboBox_SelectionChanged(object sender,
SelectionChangedEventArgs e)
    {
        int serviceId = cmbService.SelectedIndex + 1;
        conn.Open();

        SqlCommand cost = new SqlCommand("select Cost from tblServices where
ServiceID = '" + serviceId + "'", conn);

```

```

        var result = cost.ExecuteScalar();
        double price = 0;
        if(result!=null)
        {
            price = double.Parse(result.ToString());
        }
        txtAmount.Text = price.ToString("#.##");

        conn.Close();
    }
    public Transactions(string id,string type)
    {
        this.type = type;
        if(type == "s")
        {
            sql = "select serviceTransactionID as'Transaction ID',
servicedescription as 'Description', servicedate as Date,amount as Amount from
tblServicesTransactions inner join tblServices on tblServicesTransactions.ServiceID
= tblServices.ServiceId where ReservationID = '" + id + "'";

        }
        else
        {
            sql = "select orderID as'Transaction ID', Transactiondate as
Date,Cost as Amount from tblRestaurantTransactions where ReservationID = '" + id +
"'";

        }
        InitializeComponent();
        this.id = id;
        SqlCommand tr = new SqlCommand(sql,conn);
        SqlDataAdapter dataAdapter = new SqlDataAdapter(tr); //c.con is the
connection string

        DataTable dtRecord = new DataTable();
        dataAdapter.Fill(dtRecord);
        dtgTransactions.ItemsSource = dtRecord.DefaultView;
    }

    private void dtgTransactions_MouseDoubleClick(object sender,
MouseButtonEventArgs e)
    {
    }
    private void Edit(object sender, RoutedEventArgs e)
    {
        int id = getID(sender, e);
        if(id > 0)
        {
            EditTransaction.transactionid = id;
            this.Close();
        }
    }

    private int getID(object sender, RoutedEventArgs e)
    {
        int s = 0;
        for (var vis = sender as Visual; vis != null; vis =
VisualTreeHelper.GetParent(vis) as Visual)
        {
            if (vis is DataGridRow)

```

```

        {
            var row = (DataRow)vis;
            int i = row.GetIndex();
            DataRowView v = (DataRowView)dtgTransactions.Items[i]; // this
give you access to the row
            s = (int)v[0];
            break;
        }
    }
    return s;
}
private void Delete(object sender, RoutedEventArgs e)
{
    int id = getID(sender, e);
    SqlCommand delete = null;
    MessageBoxResult result = MessageBox.Show("Are you sure about deleting
transaction?", "Deleting Transaction", MessageBoxButton.YesNo);
    conn.Open();
    if(result == MessageBoxResult.Yes)
    {
        if (id > 0)
        {
            if (type == "s")
            {
                delete = new SqlCommand("delete from tblServicesTransactions
where serviceTransactionID = '" + id + "'", conn);
            }
            else
            {
                delete = new SqlCommand("delete from
tblRestaurantTransactions where OrderID = '" + id + "'", conn);
            }

            int r = delete.ExecuteNonQuery();
            if (r > 0)
            {
                MessageBox.Show("Transaction Deleted Successfully!");

                this.Close();
            }
            int r = delete.ExecuteNonQuery();
            if (r > 0)
            {
                MessageBox.Show("Transaction Deleted Successfully!");

                this.Close();
            }
        }
        private void btnView_Click(object sender, RoutedEventArgs e)
    {
        if (txtGuestID.Text != "")
        {
            conn.Open();
            SqlCommand guest = new SqlCommand("select * from tblGuests where
guestID = '" + txtGuestID.Text + "'", conn);
            SqlDataReader rdr = null;
            rdr = guest.ExecuteReader();
            bool found = false;
            while (rdr.Read())
            {
                cnvGuest.Visibility = Visibility.Visible;
                lblID.Content = rdr["GuestID"].ToString();
                lblFirstName.Content = rdr["FirstName"].ToString();
                lblLastName.Content = rdr["LastName"];
            }
        }
    }
}

```



```

        lblPhone.Content = rdr["Phone"];
        lblAddress.Content = rdr["GuestAddress"].ToString();
        lblEmail.Content = rdr["EmailAddress"].ToString();
        lblPostal.Content = rdr["PostalCode"].ToString();
        found = true;
    }

    if (found == false)
    {
        cnvGuest.Visibility = Visibility.Collapsed;
        MessageBox.Show("Гость не найден!");
    }
    rdr.Close();
    conn.Close();
}
else
{
    MessageBox.Show("Введите айди бронирования");
    txtGuestID.Focus();
}
}

private void btnEdit_Click(object sender, RoutedEventArgs e)
{
    EditGuest add = new EditGuest(txtGuestID.Text);
    add.ShowDialog();
    btnView_Click(sender, e);
}

private void btnViewCopy_Click(object sender, RoutedEventArgs e)
{
    Guestt form = new Guestt();
    form.ShowDialog();
    this.Close();
}

private void btnViewCopy1_Click(object sender, RoutedEventArgs e)
{
    Guestt form = new Guestt();
    form.ShowDialog();
    this.Close();
}

private void btnViewCopy2_Click(object sender, RoutedEventArgs e)
{
    Guestt form = new Guestt();
    form.ShowDialog();
    this.Close();
}
conn.Close();
}

public RoomInformation()
{
    InitializeComponent();
    txtRoomID.Focus();
}

private void btnCancel_Click(object sender, RoutedEventArgs e)
{
    MainWindow form = new MainWindow();
    form.Show();
    this.Close();
}

private void btnView_Click(object sender, RoutedEventArgs e)
{
    if (txtRoomID.Text != "")

```

```

        {
            conn.Open();
            SqlCommand reservation = new SqlCommand("select * from tblRooms
where RoomID = '" + txtRoomID.Text + "'", conn);
            SqlDataReader rdr = null;
            rdr = reservation.ExecuteReader();
            bool found = false;
            while (rdr.Read())
            {
                cnvRoom.Visibility = Visibility.Visible;
                lblID.Content = rdr["RoomID"].ToString();
                lblFirstName.Content = rdr["RoomTypeID"].ToString();

                lblLastName.Content = rdr["StatusID"];
                lblPhone.Content = rdr["Cost"];
                lblAddress.Content = rdr["RoomFloor"].ToString();

                found = true;
            }

            if (found == false)
            {
                cnvRoom.Visibility = Visibility.Collapsed;
                MessageBox.Show("Employee ID not found. Please try again!");
            }
            rdr.Close();
            conn.Close();

        }
        else
        {
            MessageBox.Show("Enter Reservation ID to find Reservation details");
            txtRoomID.Focus();
        }
    }

    private void btnAdd_Click(object sender, RoutedEventArgs e)
    {
        AddEditRoom add = new AddEditRoom();
        add.ShowDialog();
    }

    private void btnEdit_Click(object sender, RoutedEventArgs e)
    {
        AddEditRoom win = new AddEditRoom(txtRoomID.Text);
        win.ShowDialog();
    }

    public RoomPricing()
    {
        InitializeComponent();
        SqlDataReader reader = null;
        try
        {

            conn.Open();

            SqlCommand cmdType = new SqlCommand("select * from tblRoomType",
conn);

```

```

        reader = cmdType.ExecuteReader();

        while (reader.Read())
        {
            cmbRoomType.Items.Add(reader["TypeDescription"].ToString());
        }
    }
    finally
    {
        if (reader != null)
        {
            reader.Close();
        }
        if (conn != null)
        {
            conn.Close();
        }
    }
    SqlDataReader rdr = null;
    try
    {
        conn.Open();

        SqlCommand service = new SqlCommand("Select ServiceDescription from
tblServices", conn);
        rdr = service.ExecuteReader();
        while (rdr.Read())
        {
            cmbService.Items.Add(rdr["ServiceDescription"].ToString());
        }
    }
    finally
    {
        if (rdr != null)
        {
            rdr.Close();
        }
        if (conn != null)
        {
            conn.Close();
        }
    }
}

private void btnCancel_Click(object sender, RoutedEventArgs e)
{
    MainWindow form = new MainWindow();
    form.Show();
    this.Close();
}

private void cmbRoomType_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    conn.Open();
    SqlDataReader reader = null;
    if (cmbRoomType.SelectedValue.ToString() == "Suite")

```

```

    {
        txtPrice.Text = "";
        cmbRoomNumber.Items.Clear();
        cnvRoomNumber.Visibility = Visibility.Visible;
        cnvPricing.Margin = new Thickness(451, 509, 533.2, 215.2);

        SqlCommand number = new SqlCommand("select roomID from tblRooms
inner join tblRoomType on tblRooms.RoomTypeID = tblRoomType.RoomTypeID where
TypeDescription = 'Suite'", conn);
        reader = number.ExecuteReader();

        while (reader.Read())
        {
            cmbRoomNumber.Items.Add(reader["roomID"].ToString());
        }
        reader.Close();
    }
    else
    {

        double money = 0;
        cnvRoomNumber.Visibility = Visibility.Collapsed;
        cnvPricing.Margin = new Thickness(451, 421, 533.2, 303.2);
        SqlCommand type = new SqlCommand("SELECT TOP 1 Cost FROM tblRooms
inner join tblRoomType on tblRooms.RoomTypeID = tblRoomType.RoomTypeID where
TypeDescription = '" + cmbRoomType.SelectedValue.ToString() + "'", conn);
        var cost = type.ExecuteScalar();
        if (cost != null)
        {
            money = double.Parse(cost.ToString());
        }

        txtPrice.Text = money.ToString("#.##" + " " + "py6.");
    }

    conn.Close();
}

private void cmbRoomNumber_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    if(cmbRoomNumber.SelectedIndex != -1)
    {
        conn.Open();
        SqlCommand num = new SqlCommand("Select cost from tblRooms where
RoomID = '" + cmbRoomNumber.SelectedValue.ToString() + "'", conn);
        double result = 0;
        var output = num.ExecuteScalar();
        if (output != null)
        {
            result = double.Parse(output.ToString());
        }
        txtPrice.Text = result.ToString("#.##");
        conn.Close();
    }
    else
    {
    }
}

private void btnUpdate_Click(object sender, RoutedEventArgs e)
{

```

```

        double price = 0;
        if(double.TryParse(txtPrice.Text, out price) && price > 0)
        {
            conn.Open();

            if (cmbRoomType.SelectedValue.ToString() == "Suite")
            {
                SqlCommand update = new SqlCommand("UPDATE tblRooms SET Cost='"
+ txtPrice.Text + "' where RoomID = '" + cmbRoomNumber.Text + "'", conn);
                int r = update.ExecuteNonQuery();
                if (r > 0)
                {
                    MessageBox.Show("Reservation Updated successfully!");
                    MainWindow main = new MainWindow();
                    main.Show();
                    this.Close();
                }
            }
            else
            {
                SqlCommand update = new SqlCommand("Update tblrooms set cost =
'" + txtPrice.Text + "' where RoomTypeID = ( select top 1 tblrooms.roomtypeid from
tblrooms inner join tblroomtype on tblrooms.RoomTypeID = tblroomtype.RoomTypeID
where Typedescription = '" + cmbRoomType.Text + "')", conn);
                int r = update.ExecuteNonQuery();
                if (r > 0)
                {
                    MessageBox.Show("Reservation Updated successfully!");
                    MainWindow main = new MainWindow();
                    main.Show();
                    this.Close();
                }
            }
        }
        else
        {
            MessageBox.Show("Enter numbers for price and greater than 0");
        }
    }

    private void txtPrice_TextChanged(object sender, TextChangedEventArgs e)
    {
    }

    private void cmbService_SelectionChanged(object sender,
SelectionChangedEventArgs e)
    {
        int serviceId = cmbService.SelectedIndex + 1;
        conn.Open();

        SqlCommand cost = new SqlCommand("select Cost from tblServices where
ServiceID = '" + serviceId + "'", conn);
        var result = cost.ExecuteScalar();
        double price = 0;
        if (result != null)
        {
            price = double.Parse(result.ToString());
        }
        txtAmount.Text = price.ToString("#.##" + " " + "py6.");
    }

```

```

        conn.Close();
    }
    public NomeraBD()
    {
        InitializeComponent();
    }
    string id = "";
    string type = "";
    string sql = "";
    SqlConnection conn = new SqlConnection("Data Source=DESKTOP-078D6KH;Initial
Catalog=KingWilliamHotelDB;Integrated Security=True");

    public NomeraBD(string id, string type)
    {
        sql = "select serviceTransactionID as'Transaction ID',
servicedescription as 'Description', servicedate as Date,amount as Amount from
tblServicesTransactions inner join tblServices on tblServicesTransactions.ServiceID
= tblServices.ServiceId where ReservationID = '" + 1 + "'";

        //sql = "select orderID as'Transaction ID', Transactiondate as Date,Cost
as Amount from tblRestaurantTransactions where ReservationID = '" + 1 + "'";

        InitializeComponent();
        this.id = id;
        SqlCommand tr = new SqlCommand(sql, conn);
        SqlDataAdapter dataAdapter = new SqlDataAdapter(tr); //c.con is the
connection string

        DataTable dtRecord = new DataTable();
        dataAdapter.Fill(dtRecord);

    }
    private void Window_Loaded(object sender, RoutedEventArgs e)
    {
        RoomGrid.ItemsSource = AppData.db.tblRooms.ToList();
    }
    private void Poisk_TextChanged(object sender, TextChangedEventArgs e)
    {
        try
        {
            RoomGrid.ItemsSource = AppData.db.tblRooms.Where(item =>
item.RoomID.Contains(Poisk.Text)).ToList();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString(), "Error", MessageBoxButton.OK,
MessageBoxImage.Error);
        }
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        Nomera form = new Nomera();
        form.ShowDialog();
    }

    private void CheckBox_Checked(object sender, RoutedEventArgs e)
    {
        RoomGrid.ItemsSource = AppData.db.tblRooms.Where(item =>
item.RoomTypeID.Contains("U")).ToList();
    }

```

```

private void CheckBox_Checked_1(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.Where(item =>
item.RoomTypeID.Contains("S")).ToList();
}

private void CheckBox_Checked_2(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.Where(item =>
item.RoomTypeID.Contains("D")).ToList();
}


private void Button_Click_1(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.ToList();
}

private void CheckBox_Unchecked(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.ToList();
}

private void Button_Click_2(object sender, RoutedEventArgs e)
{
    MainWindow form = new MainWindow();
    form.ShowDialog();
    this.Close();
}

private void CheckBox_Unchecked_1(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.ToList();
}

private void CheckBox_Unchecked_2(object sender, RoutedEventArgs e)
{
    RoomGrid.ItemsSource = AppData.db.tblRooms.ToList();
}

private void Button_Click_3(object sender, RoutedEventArgs e)
{
    string filePath = @"C:\Users\Asus\Desktop\КоличествоБронирований.docx";

    // запускаем приложение Word и открываем файл
    Process.Start("WINWORD.EXE", filePath);
}

public ExistingReservation()
{
    InitializeComponent();
}

private void btnCancel_Click(object sender, RoutedEventArgs e)
{
    MainWindow form = new MainWindow();
    form.Show();
    this.Close();
}

private void btnFind_Click(object sender, RoutedEventArgs e)

```

```

{
    if(txtFind.Text != "")
    {
        conn.Open();
        SqlCommand reservation = new SqlCommand("select * from
tblReservations inner join tblGuests on tblReservations.GuestID = tblGuests.GuestID
where reservationid = '" + txtFind.Text + "'", conn);
        SqlDataReader rdr = null;
        rdr = reservation.ExecuteReader();
        bool found = false;
        while(rdr.Read())
        {
            cnvResult.Visibility = Visibility.Visible;
            txtGuestID.Text = rdr["GuestID"].ToString();

            lblStartDate.Content = rdr["ReservationStartDate"];
            lblEndDate.Content = rdr["ReservationEndDate"];
            lblRoomNumber.Content = rdr["RoomID"].ToString();
            txtFirstName.Text = rdr["FirstName"].ToString();
            txtLastName.Text = rdr["LastName"].ToString();
            txtPhone.Text = rdr["Phone"].ToString();
            txtAddress.Text = rdr["GuestAddress"].ToString();
            txtEmail.Text = rdr["EmailAddress"].ToString();
            txtPostalCode.Text = rdr["PostalCode"].ToString();
            found = true;

        }

        if(found==false)
        {
            cnvResult.Visibility = Visibility.Collapsed;
            MessageBox.Show("Бронирование не найдено!");
        }
        rdr.Close();
        conn.Close();

    }
    else
    {
        MessageBox.Show("Введите айди бронирования");
        txtFind.Focus();
    }
}

private void txtGuestID_LostFocus(object sender, RoutedEventArgs e)
{
}

private void dpiStartDate_SelectedDateChanged(object sender,
SelectionChangedEventArgs e)
{
}

private void dpiEndDate_SelectedDateChanged(object sender,
SelectionChangedEventArgs e)

```



```

    {
    }

    private void btnEdit_Click(object sender, RoutedEventArgs e)
    {
        Window choose = new ChangeRoom(txtGuestID.Text );
        var result = choose.ShowDialog();
        if(result == true)
        {
            lblStartDate.Content = ChangeRoom.startDate;
            lblEndDate.Content = ChangeRoom.endDate;
            lblRoomNumber.Content = ChangeRoom.roomNumber;
        }
    }

    private void btnUpdate_Click(object sender, RoutedEventArgs e)
    {
        if(txtFirstName.Text==" " || txtLastName.Text =="" || txtAddress.Text==" "
        || txtPhone.Text =="" ||txtEmail.Text==" "||txtPostalCode.Text=="")
        {
            MessageBox.Show("Все поля должны быть заполнены");
        }
        else
        {
            conn.Open();
            SqlCommand update = new SqlCommand("UPDATE tblGuests SET
            FirstName='" + txtFirstName.Text + "', LastName='" + txtLastName.Text + " '
            ,GuestAddress='" + txtAddress.Text + " ' ,PostalCode='" + txtPostalCode.Text + " ',
            Phone='" + txtPhone.Text + " ' ,EmailAddress = '" + txtEmail.Text + " ' where GuestID
            = '" + txtGuestID.Text + "'", conn);
            int r = update.ExecuteNonQuery();
            if(r > 0)
            {
                SqlCommand updater = new SqlCommand("UPDATE tblReservations SET
                RoomID='" + lblRoomNumber.Content + " ', ReservationStartDate='" +
                lblStartDate.Content + " ', ReservationEndDate='" + lblEndDate.Content + " ' where
                GuestID = '" + txtGuestID.Text + "'", conn);
                int s = updater.ExecuteNonQuery();
                if(s > 0)
                {
                    MessageBox.Show("Бронирование обновлено!");
                    this.Close();
                }
            }

            conn.Close();
        }
    }

    private void btnDelete_Click(object sender, RoutedEventArgs e)
    {
        conn.Open();
        SqlCommand update = new SqlCommand("DELETE FROM tblReservations where
        reservationID = '" + txtFind.Text + "'", conn);
        int r = update.ExecuteNonQuery();
        if(r > 0)
        {
            MessageBox.Show("Бронирование удалено успешно!");
            this.Close();
        }
    }
}

```

BonHotel

Новое бронирование

☐ Существующий гость

☒ Новый гость

Имя:

Артём

Фамилия:

Яроцкий

Номер паспорта

AВ133722

Пол:

Мужской ▾

Номер телефона

+375445570550

E-mail:

college@diplom.by

Дата заезда:

27.09.2023

15

Дата отъезда:

05.10.2023

15

Тип номера:

Luxe ▾

Номер комнаты

U201 ▾

200 руб за ночь

Забронировать

Отменить

1004	G00017	D204	(28.00.2023)	(28.00.2023)	(02.00.2023)
------	--------	------	--------------	--------------	--------------

1003	G00012	U204	(28.00.2023)	(28.00.2023)	(31.00.2023)
1005	G00018	D201	(28.00.2023)	(28.00.2023)	(01.00.2023)

[illegible]

Рисунок Б.3 – Изменение бронирования

## Прайс-лист всех услуг отеля

Прайс-лист на номера

Выберите тип номера:

Luxe

Цена за ночь:

200 руб

Назад

Прайс-лист на доп.услуги отеля

Выберите услугу:

Цена:

Рисунок Б.4 – Просмотр прайс-листа

Checks


Бронирование номер: 2023

Ваш номер комнаты: U201

Дата заезда в отель: 27.09.2023 0:00:00

Дата выезда из отеля: 05.10.2023 0:00:00

---

Итого: 1200 руб. за 8 дней М.П. 

Назад Печать

Рисунок Б.5 – Выдача чека

Checks


Бронирование номер: 2023

Ваш номер комнаты: U201

Дата заезда в отель: 27.09.2023 0:00:00

Дата выезда из отеля: 05.10.2023 0:00:00

---

Итого: 1200 руб. за 8 дней М.П. 

Назад Печать

Рисунок Б.6 – Сформированный чек