

МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ  
Кировское областное государственное профессиональное образовательное  
бюджетное учреждение  
"Слободской колледж педагогики и социальных отношений"

## **КУРСОВОЙ ПРОЕКТ**

по ПМ 01 «Разработка программных модулей» на тему:  
**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ БРОНИРОВАНИЯ  
МЕСТ В ГОСТИНИЦЕ**

Выполнила: Поглазова Валерия  
Владимировна

Специальность 09.02.07  
Информационные системы и  
программирование

Группа 21П-1  
Форма обучения: очная

Руководитель: Махнев  
Александр Анатольевич

Дата защиты курсового проекта:

Председатель ПЦК:

Оценка за защиту курсового проекта:

Слободской  
2024

**ОГЛАВЛЕНИЕ**

Оглавление .....	2
Введение .....	3
1. Анализ предметной области .....	6
2. Разработка технического задания.....	12
3. Описание алгоритмов и функционирования программы .....	16
4. Тестирование программного модуля .....	19
5. Руководство пользователя .....	23
Заключение .....	31
Список литературы .....	32
Приложения .....	34
Приложение 1 .....	35
Приложение 2 .....	59

## ВВЕДЕНИЕ

В современном мире, где путешествия стали неотъемлемой частью жизни, удобство и эффективность бронирования гостиничных номеров приобретают ключевое значение. Разработка программного модуля для бронирования мест в гостинице представляет собой сложную, но увлекательную задачу, которая позволит автоматизировать этот процесс и сделать его более доступным для пользователей. Из года в год все больше туристских фирм и отдельных граждан используют системы бронирования гостиниц. Такие системы позволяют бронировать места в гостиницах, предлагая пользователям удобный и интуитивно понятный интерфейс для поиска и выбора доступных номеров. Они предоставляют детальную информацию о номерах, описание удобств и цены. Сейчас трудно представить, как могло производиться бронирование отелей при отсутствии систем компьютерного бронирования.

В настоящее время эти системы широко распространены и хорошо скоординированы. Основным ресурсом гостиницы является её номерной фонд, и правильное управление процессом бронирования позволит использовать его наиболее эффективно. Очень важно помнить, что наиболее выгодные бронирования поступают от индивидуальных клиентов. Важную роль в привлечении этой категории гостей играют международные турагентства, которые работают в глобальных системах бронирования GDS и не имеют прямых договоров с гостиницами.

Актуальность:

Современные гостиницы нуждаются в эффективных системах бронирования, которые:

- Обеспечивают быстрый и простой доступ к информации о свободных номерах. Гостям важно мгновенно узнать, доступен ли желаемый номер на нужные даты, чтобы избежать потерь времени.
- Предоставляют возможность отслеживания статуса бронирования. Гостям важно быть уверенными в том, что их бронь подтверждена, а также знать,

когда они могут заселиться. Это создает дополнительный уровень комфорта и уверенности для клиентов.

- Упрощают процесс управления бронированиями для администраторов и менеджеров гостиницы. Эффективная система должна позволять сотрудникам быстро проверять статус номеров, управлять доступностью и обрабатывать заявки на бронирование, что способствует более слаженной работе гостиницы.

Бронирование – это дополнительная услуга, которая позволяет зарезервировать за потенциальным клиентом номер для будущего проживания в гостинице. Для этого нужно заранее определиться, на какой срок и какое количество человек планирует заселиться в отель, а также указать предпочтительную категорию жилого помещения. При несвоевременном отказе, могут быть назначены штрафные санкции, то есть часть стоимости заезда в номер не возвращается. Условия бронирования и отказа от бронирования желательно выяснить оформление номеров. К функциям службы бронирования в гостинице относятся: проверка доступности номеров, подтверждение бронирования, ведение базы данных бронирований.

Объект исследования – бронирование мест в гостинице.

Предмет исследования – метод бронирования мест в гостинице.

Цель курсового проекта – разработка программного модуля для бронирования мест в гостинице, который позволит обслуживать гостей гостиницы, осуществлять бронирование номеров, обеспечивая удобство, для пользователей и оптимизировать работу гостиницы.

Задачи исследования:

1. Описать предметную область.
2. Разработать технического задание на создание программного продукта.
3. Описать архитектуру программы.
4. Описать алгоритмы и функционирование программы.
5. Провести тестирование и опытную эксплуатацию программы.

## 6. Разработать руководство оператора

Методы исследования: анализ предметной области, разработка и тестирование программы.

Информационную систему исследования составили официальные нормативно-правовые источники, данные об использовании современных информационных систем. Структура работы состоит из введения, трех глав, заключения, списка используемой литературы и приложений.

Практическая значимость: программное обеспечение может быть использована для бронирования мест в гостинице.

## 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В данном курсовом проекте описан процесс бронирования мест в гостинице, а именно резервирование номера в отеле клиентом. Гостиничный сервис ведущая отрасль сферы обслуживания. Современное состояние рынка гостиничных услуг характеризуется высоким уровнем конкуренции, разнообразием видов, предоставляемых основных и дополнительных услуг, повышением уровня обслуживания. С точки зрения организации и управления гостиничные комплексы представляют собой сложные системы, которые состоят из различных взаимосвязанных служб. Служба управления номерным фондом осуществляет основные бизнес-процессы по приему и обслуживанию гостей, и во многом именно она определяет качество предоставляемых услуг. С этой точки зрения эта служба является одним из основных объектов, в первую очередь подлежащих автоматизации.

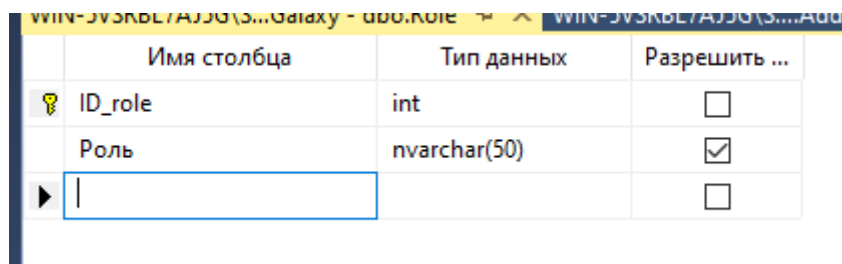
Административная служба осуществляет основные бизнес-процессы, связанные с управлением отелем в целом, координацией деятельности всех служб отеля, кадрового обеспечения, контрольные функции и т.п. Обычно в состав этой службы входят директор и топ-менеджеры отеля, бухгалтерия, финансовая служба, отдел кадров и др. С точки зрения информационных процессов эта служба представляет собой то ядро, которое должно объединять информационно воедино все службы отеля. Работники этого блока, как правило, имеют наиболее полные права доступа ко всей информации, функционирующей в системе управления отелем.

В гостинице имеются номера категории люкс, одноместные и двухместные. В гостиницу приходит постоялец или новый клиент, подходит к администратору на ресепшене и указывает нужную ему категорию гостиничного номера и срок. Если гость впервые бронирует номер, то ему необходимо заполнить предложенные поля, его личными данными, а уже потом приступает к бронированию места в гостинице.

Администратор продает свободные номера, удовлетворяющие желанию постояльца. Постоялец выбирает один, наиболее подходящий номер из всех предложенных. Между постояльцем и гостиницей, в лице администратора, заключается договор, в котором указан тип номера, количество планируемых дней проживания и список ежедневных услуг (завтрак, такси, уборка номеров и др.) Если такого номера не нашлось, то администратор советует посетить другую гостиницу.

После изучения вышеперечисленного были выделены следующие сущности и их атрибуты:

#### 1.Role (Рисунок 1)





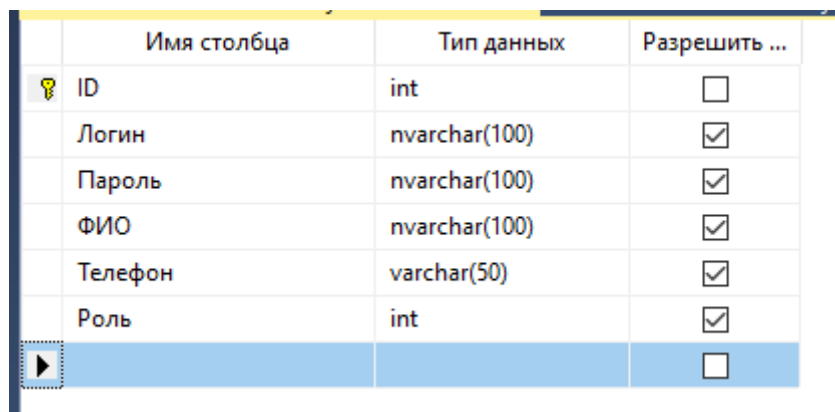
	Имя столбца	Тип данных	Разрешить ...
	ID_role	int	<input type="checkbox"/>
	Роль	nvarchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 1 - Таблица «Role»

#### 2. User (Рисунок 2)





	Имя столбца	Тип данных	Разрешить ...
	ID	int	<input type="checkbox"/>
	Логин	nvarchar(100)	<input checked="" type="checkbox"/>
	Пароль	nvarchar(100)	<input checked="" type="checkbox"/>
	ФИО	nvarchar(100)	<input checked="" type="checkbox"/>
	Телефон	varchar(50)	<input checked="" type="checkbox"/>
	Роль	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 2 - Таблица «User»

#### 3. Agreement (Рисунок 3)

PVSTUDIOS\SQLXP... - dbo.Agreement		PVSTUDIOS\SQLXP...alaxy - Diagram_0*	
	Имя столбца	Тип данных	Разрешить знач...
PK	ID_Agreement	int	<input type="checkbox"/>
	Бронирование	int	<input checked="" type="checkbox"/>
	[Дата заключения]	date	<input checked="" type="checkbox"/>
	[Тип номера]	int	<input checked="" type="checkbox"/>
	[Список услуг]	int	<input checked="" type="checkbox"/>
	[Общая стоимость]	money	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 3 - Таблица «Agreement»

## 4. Booking (Рисунок 4)

	Имя столбца	Тип данных	Разрешить знач...
PK	ID_Booking	int	<input type="checkbox"/>
	Гость	int	<input checked="" type="checkbox"/>
	Номер	int	<input checked="" type="checkbox"/>
	Количество_человек	varchar(255)	<input checked="" type="checkbox"/>
	Дата_заезда	date	<input checked="" type="checkbox"/>
	Дата_выезда	date	<input checked="" type="checkbox"/>
	Количество_дней_проживания	nvarchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 4 - Таблица «Booking»

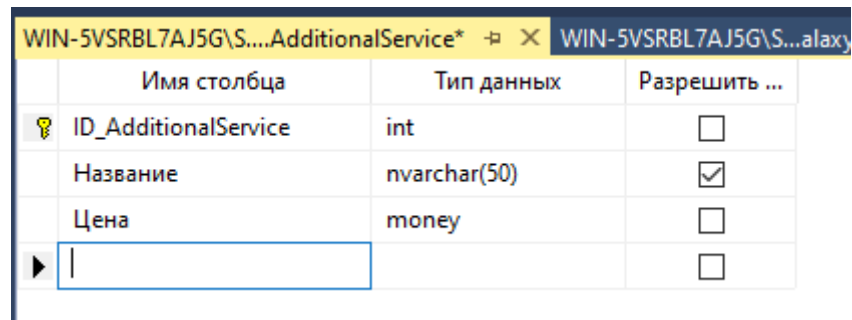
## 5. Guest (Рисунок 5)

WIN-5VSRBL7AJ5G\S...alaxy - dbo.Guest		WIN-5VSRBL7AJ5G\S...alaxy	
	Имя столбца	Тип данных	Разрешить ...
PK	ID_Guest	int	<input type="checkbox"/>
	ФИО	nvarchar(50)	<input checked="" type="checkbox"/>
	Телефон	varchar(20)	<input checked="" type="checkbox"/>
	[Серия и Номер Паспо...	nvarchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Рисунок 5 - Таблица «Guest»



## 6. AdditionalService (Рисунок 6)

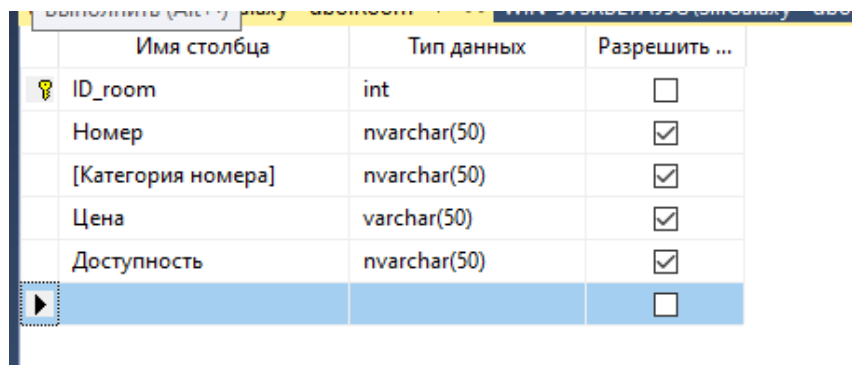


The screenshot shows the 'Table Designer' window for the 'AdditionalService' table. The table has four columns: 'ID\_AdditionalService' (int, primary key), 'Название' (nvarchar(50)), 'Цена' (money), and an unnamed column (nvarchar(50)). The 'Название' column has the 'Allow Nulls' checkbox checked.

	Имя столбца	Тип данных	Разрешить ...
🔑	ID_AdditionalService	int	<input type="checkbox"/>
	Название	nvarchar(50)	<input checked="" type="checkbox"/>
	Цена	money	<input type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 6 - Таблица «AdditionalService»

## 7. Room (Рисунок 7)



The screenshot shows the 'Table Designer' window for the 'Room' table. The table has six columns: 'ID\_room' (int, primary key), 'Номер' (nvarchar(50)), '[Категория номера]' (nvarchar(50)), 'Цена' (varchar(50)), 'Доступность' (nvarchar(50)), and an unnamed column (nvarchar(50)). The 'Номер', '[Категория номера]', 'Цена', and 'Доступность' columns have the 'Allow Nulls' checkbox checked.

	Имя столбца	Тип данных	Разрешить ...
🔑	ID_room	int	<input type="checkbox"/>
	Номер	nvarchar(50)	<input checked="" type="checkbox"/>
	[Категория номера]	nvarchar(50)	<input checked="" type="checkbox"/>
	Цена	varchar(50)	<input checked="" type="checkbox"/>
	Доступность	nvarchar(50)	<input checked="" type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 7 - Таблица «Room»

На основании этих данных была создана у базы данных (Рисунок 8)

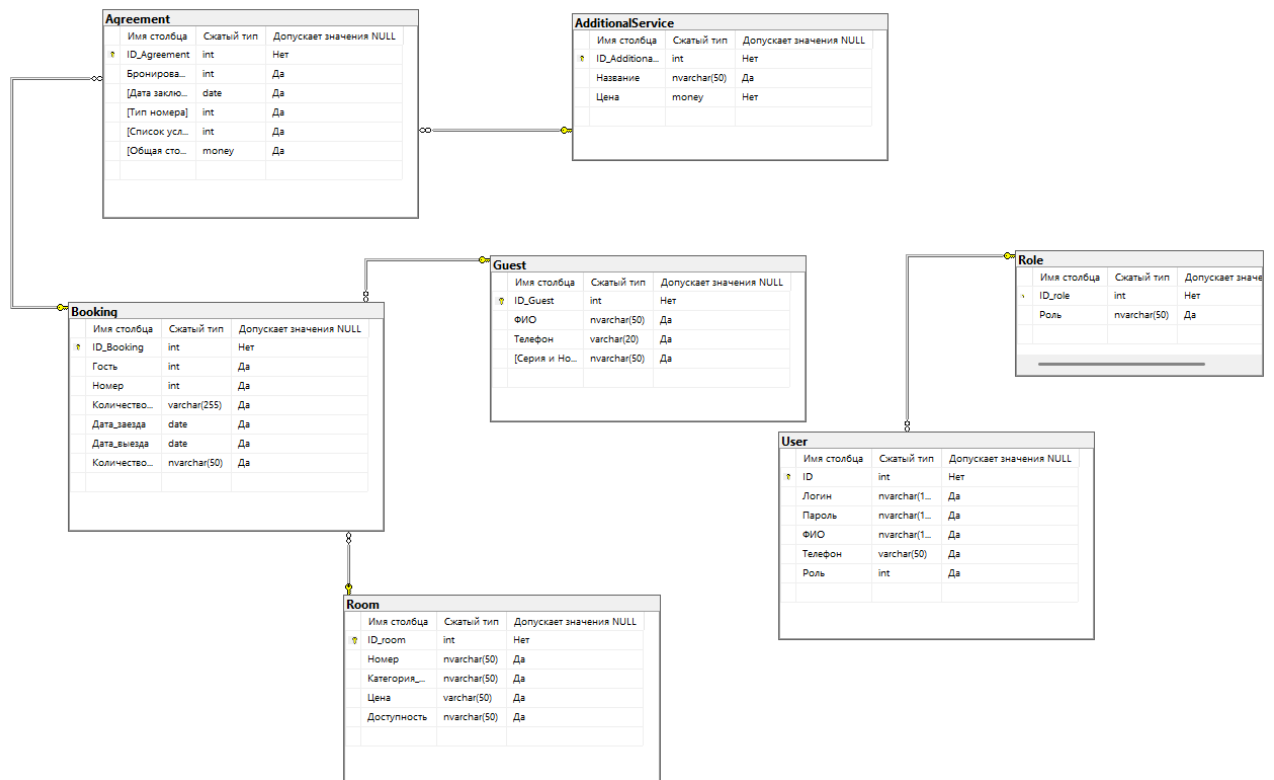


Рисунок 8 – «Модель базы данных»

Диаграмма вариантов использования (прецедентов) отображает отношения между актерами и вариантами использования.

Представлена диаграмма вариантов использования бизнес-процесса гостиницы. (Рисунке 9).

На диаграмме присутствуют 2 действующих лица: клиент, администратор. Они являются внешними по отношению, моделируемому бизнес-процессу компании, сущностями, которые взаимодействуют с гостиницей.

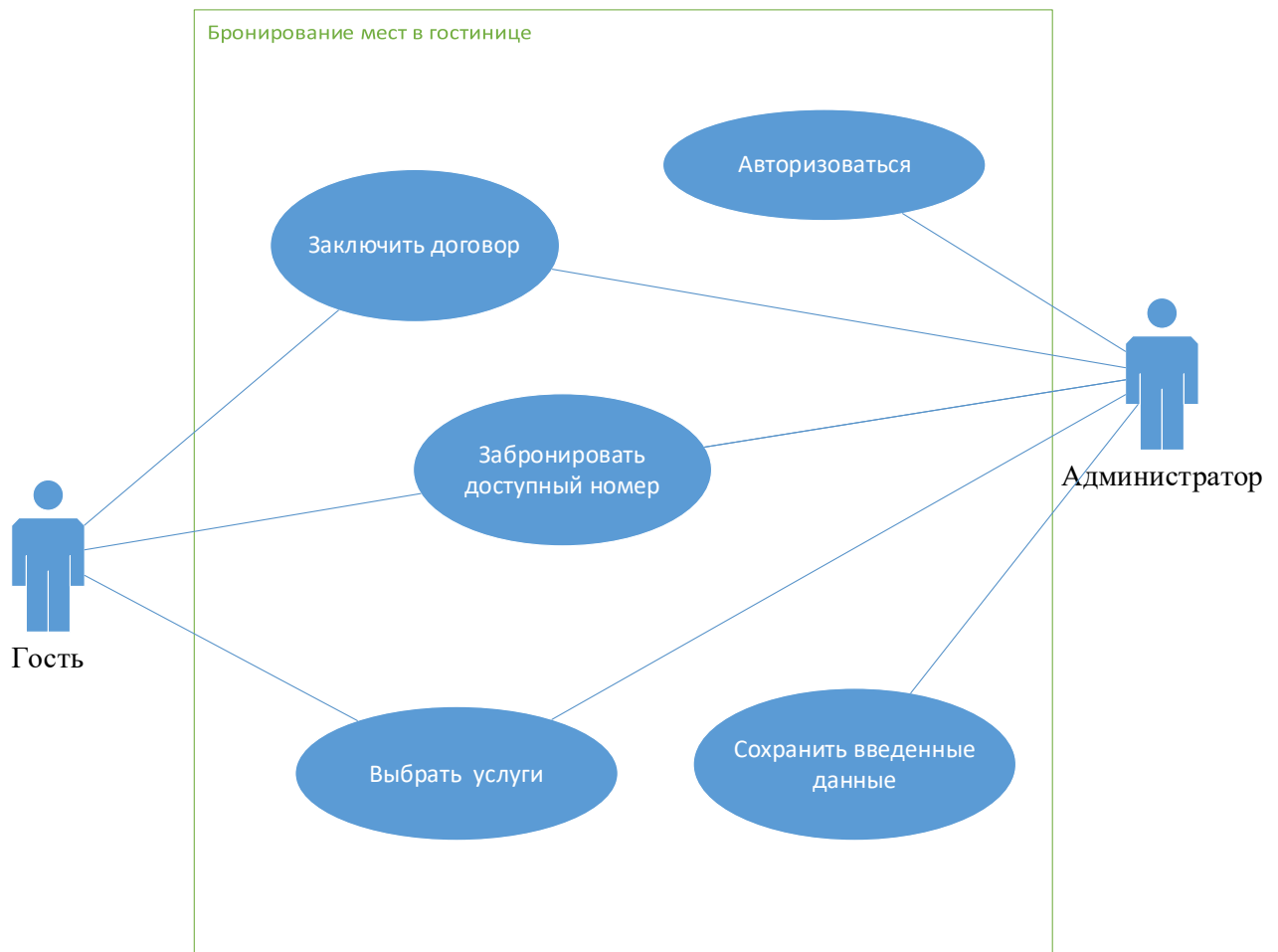


Рисунок 9 – «Диаграмма вариантов использования»

Вывод: была проведена работа по создаваемому программному модулю, над его функциями. На основании анализа была создана база данных и её диаграмма в системе SQL Server Management Studio.

## 2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

Наименование программы – «HotelGalaxy». Программа предназначена для бронирования мест в гостинице.

Разработка программы ведется на основании учебного плана и перечня тем утвержденных на заседании предметно цикловой комиссии информатики и программирования.

Функциональным назначением программы является бронирование клиентом номера в гостинице.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1. Авторизация пользователя;
2. Просмотр информации о номерах
3. Поиск свободных номеров
4. Бронирование номера
5. Сохранение даты въезда в гостиницу и выезда из неё
6. Сохранение введенных контактных данных
7. Заключение договора о бронировании

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

- организация бесперебойного питания технических средств;
- использование лицензионного программного обеспечения;
- отсутствие вредоносного программного обеспечения, наличие антивирусной программы;
- соблюдение правил и требований по эксплуатации технических средств.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не

крахом) операционной системы, не должно превышать 5 минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу пользователя без предоставления ему административных привилегий.

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

- процессор с тактовой частотой, 1 ГГц, не менее;
- оперативную память объемом 512 Мб, не менее;
- жесткий диск со свободным местом 500 Мб, не менее;
- монитор, с разрешением экрана 1024\*768, не менее;
- оптический привод;
- компьютерная мышь;
- клавиатура.

Исходные коды программы должны быть реализованы на языке C#. В качестве интегрированной среды разработки программы должна быть использована среда программирования Microsoft Visual Studio 2022.

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Программное обеспечение поставляется в виде изделия на CD диске.

Упаковка программного изделия должна осуществляться в упаковочную тару предприятия-изготовителя компакт диска

Требования к транспортировке и хранению должны соответствовать условиям эксплуатации носителей, на которых находится программный продукт.

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса.

Предварительный состав программной документации включает в себя следующие документы:

- техническое задание;
- руководство оператора.

Разработка должна быть проведена в следующие стадии и этапы:

1. Анализ требований:

На стадии анализ требований формулируются цели и задачи проекта. Создается основа для дальнейшего проектирования

2. Проектирование:

На стадии проектирование должны быть выполнены перечисленные ниже этапы работ:

- разработка программной документации;

На этапе разработка программной документации должна быть выполнена разработка технического задания.

При разработке технического задания должны быть выполнены перечисленные работы: постановка задачи, определение и уточнение требований к техническим средствам, определение требований к программе, определение стадий, этапов и сроков разработки программы и документации на нее, выбор языков программирования.

- разработка алгоритма программы;

На этапе разработки алгоритма программы должен быть разработан алгоритм работы программы.

- кодирование;

На стадии кодирования происходит реализация алгоритмов в среде программирования.

- тестирование и отладка.

На стадии тестирования и отладки происходит проверка алгоритмов, реализованных в программе на работоспособность в различных ситуациях. Исправление выявленных ошибок, повторное тестирование.

Приемо-сдаточные испытания должны проводиться при использовании технических средств. Приемка программы заключается в проверке работоспособности программы путем ввода реальных или демонстрационных данных.

Во время приемки работы разработчик предоставляет программу и документацию, которая к ней прилагается. Проводятся испытания программы, при успешных испытаниях программа вводится в эксплуатацию. При ошибках, недопустимых для успешной работы программного продукта – отправляется на доработку.

Было описано техническое задание, содержащее в себе информацию о программном продукте, его функциях, эксплуатации и требования, которые должны учитываться при создании программы и документации к ней.

### 3. ОПИСАНИЕ АЛГОРИТМОВ И ФУНКЦИОНИРОВАНИЯ ПРОГРАММЫ

Наименование программы – «HotelGalaxy». Программа предназначена для бронирования мест в гостинице.

Функциональным назначением программы является бронирование клиентом номера и гостинице.

При запуске программы происходит отображение главного окна приложения и подключение к базе данных посредством технологии ADO.NET Connection (Рисунок 10). В Окне администратору или менеджеру предлагается авторизоваться. Они вводят логин и пароль.

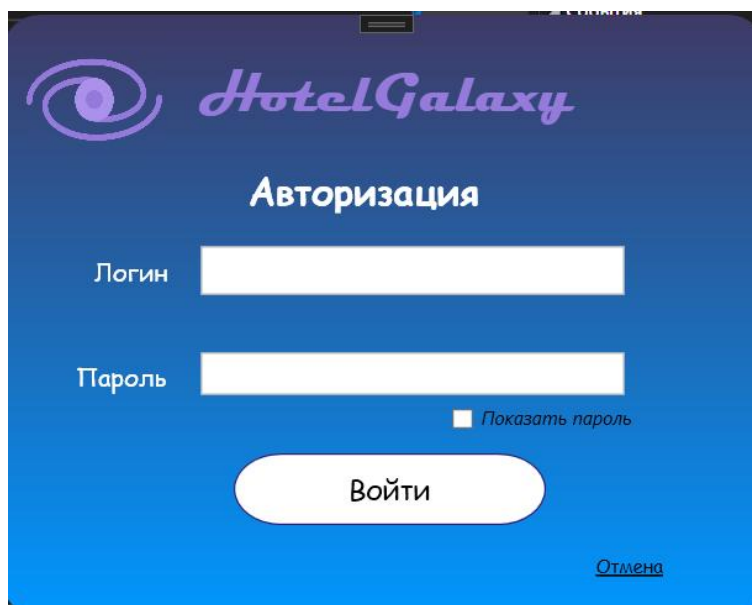


Рисунок 10 – Главное окно «Авторизация»

Когда пользователь ввел логин, пароль и нажал на кнопку «Войти», то система проверяет на наличие этого логина и пароля в базе. Если совпало, то пользователь переходит дальше, а если нет, то появится сообщение о неверном введенном логине или пароле.

#### **Входные данные**

Входные данные заполняются с помощью клавиатуры, специально введённые для этого поля на окне в программном модуле. В качестве примера в



данном программном продукте рассмотрим окно бронирование (Рисунок 10), со следующими полями для ввода данных:

- Заезд
- Выезд
- Тип номера
- Номер
- Количество человек
- Количество дней проживания
- ФИО гостя
- Телефон
- Серия и номер паспорта
- Дополнительные услуги в номерах

После ввода данных и нажатия кнопки «Забронировать» программа проверяет корректность введенных в текстовые поля данных. Как только пользователь всё указал, данные заносятся в базу данных SQL. (Рисунок 11).

Бронирование номера

Период и тип номера

Заезд: Выбор даты 15 Выезд: Выбор даты 15 Количество дней:

Тип номера: Одноместный Номер: 187

Количество человек: - 1 +

Контактные данные

ФИО

Телефон:

Серия и номер паспорта

Дополнительные услуги

☐ Завтрак

☐ Такси

☐ Уборка номера

Цена 0,00 Р

Стоимость номера 3 000,00 Р

Общая стоимость 3 000,00 Р

Назад Забронировать Печать договора

Рисунок 11 – Окно «Бронирование»

После запуска окна Администратора, в DataGrid загружаются данные из таблицы, привязанной с помощью Entity Framework базы данных SQL Server, в которую сохраняются данные бронирования гостя (Рисунок 12).

Так же стоит рассмотреть модуль поиска свободного номера в таблице «Room». В строку поиска пользователь вводит категория номера и программа выводит в DataGrid именно тот номера, которые свободны.

**Вывод:** мною были описаны алгоритмы работы созданного программного модуля и вспомогательной базы данных. Также было описано правильное функционирование программы и изложен способ организации входных данных.

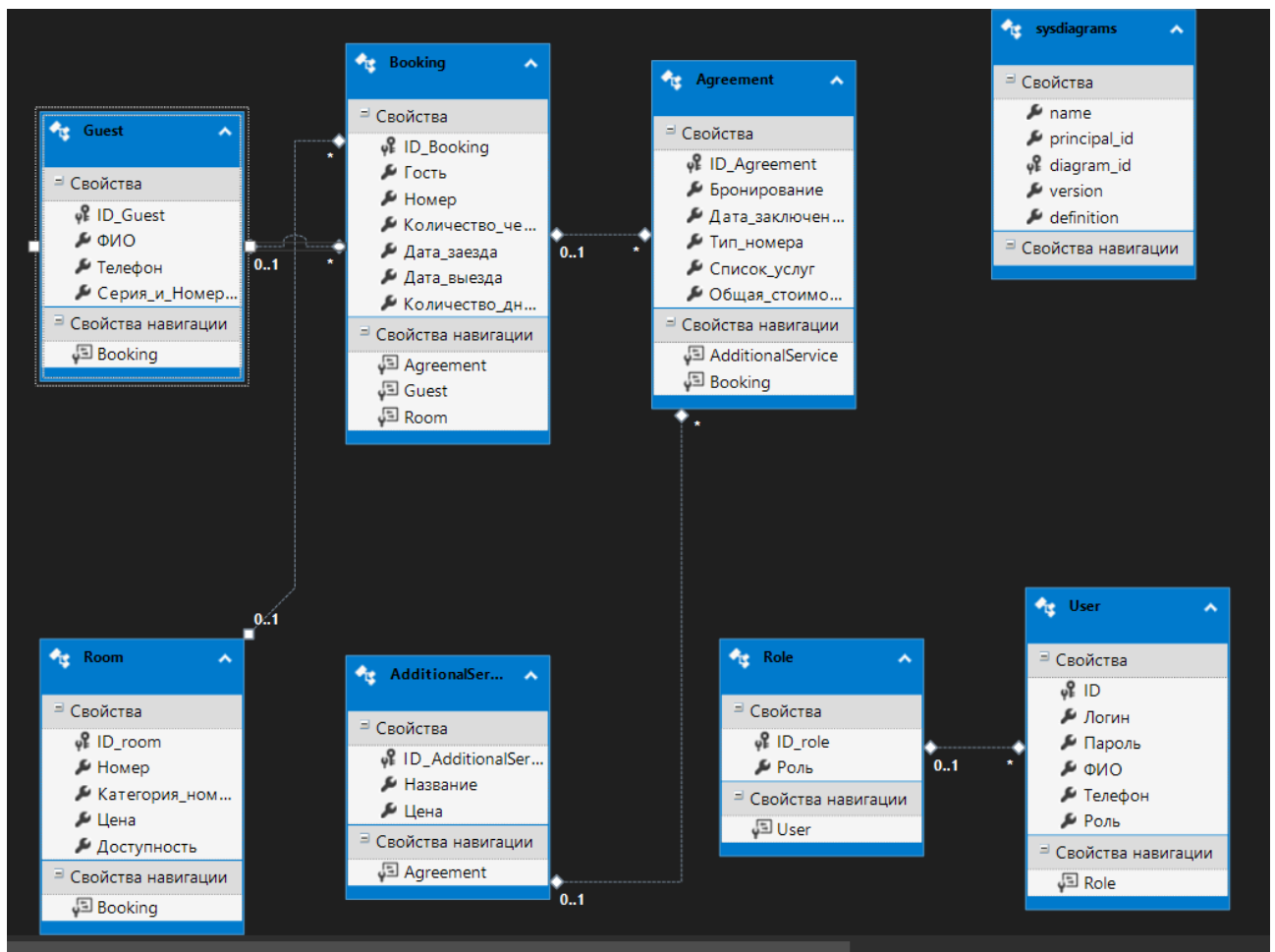


Рисунок 12 –Модель Entity Framework для привязанной к модулю базы данных SQL

#### 4. ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ

Для обеспечения надежности и функциональности программы было произведено тестирование.

Цели тестирования:

- Проверка корректности данных бронирования.
- Выявление ошибок в интерфейсе и логике программы.
- Оценка производительности и устойчивости к нагрузкам.

Проведенные виды тестирования:

1) Функциональное тестирование:

Были разработаны сценарий на основе требований к программе:

- Тестирование правильности выбора дат (Рисунок 12 и 13).

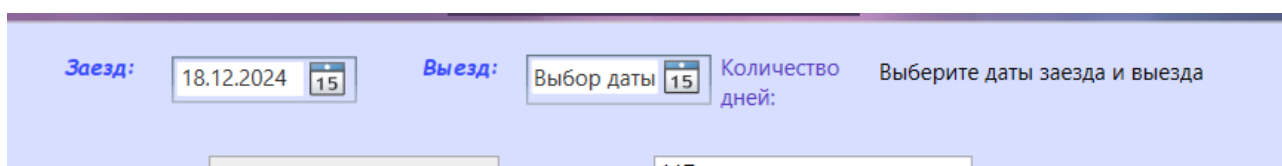


Рисунок 12 – Сообщение «Выберите даты заезда и выезда»

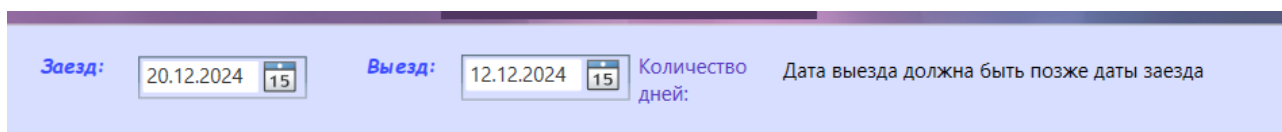


Рисунок 13 – Сообщение «Дата выезда должна быть позже даты заезда»

- Проверка корректности введенного номера телефона (Рисунок 14).

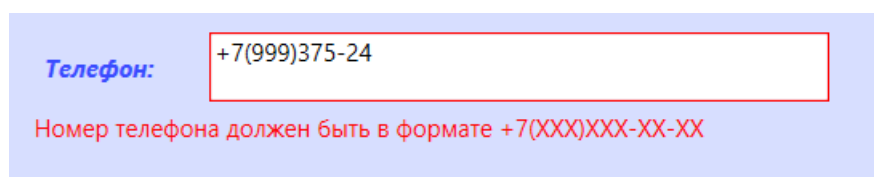


Рисунок 14 – Сообщение «Номер телефона должен быть в формате +7(XXX)XXX-XX-XX»

- Попытка оставить поля гостя пустыми (Рисунок 15).

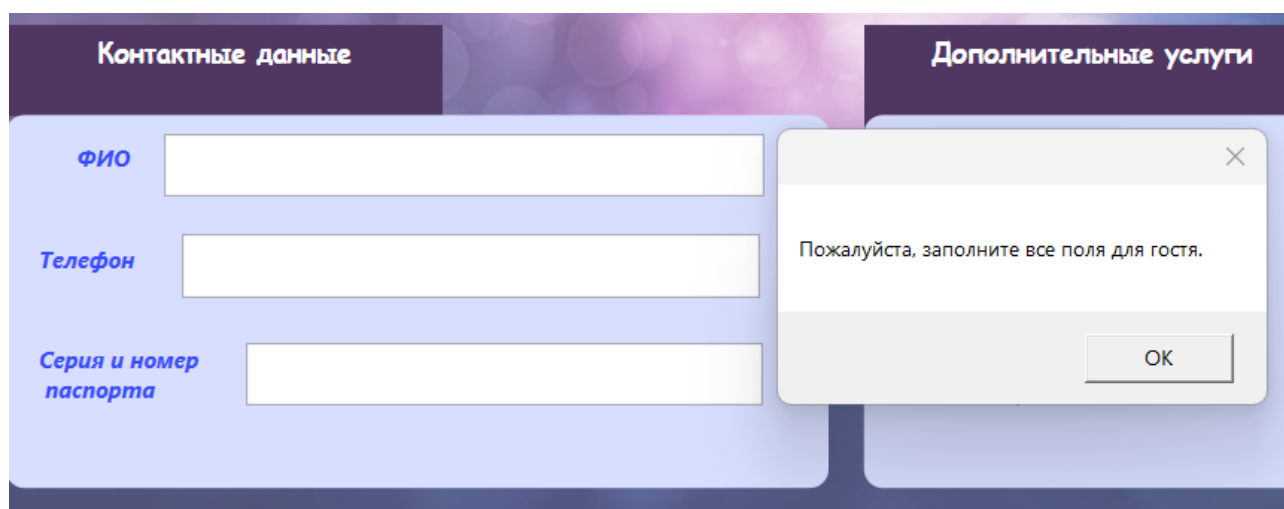


Рисунок 15 –Сообщение «Пожалуйста, заполните все поля для гостя»

Ожидаемый результат: сообщение об ошибке, уведомляющее пользователя о том, что необходимо заполнить все поля для гостей.

Полученный результат: пользователь получает сообщение с просьбой заполнить все поля для гостя (Рисунок 15).

Решение проблемы: пользователь должен заполнить все поля для гостей, чтобы продолжить.

- Попытка отсутствия выбора типа номера

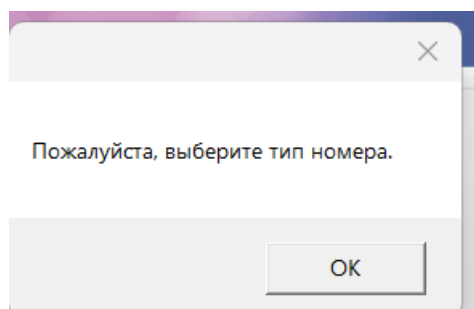


Рисунок 16 – Тип номера не выбран

Ожидаемый результат: пользователь должен выбрать тип номера из списка доступных вариантов.

Полученный результат: появилось сообщение "Пожалуйста, выберите тип номера" (Рисунок 16).

Решение проблемы: пользователь должен выбрать один из предложенных типов номеров.

- Попытка отсутствия выбора номера (Рисунок 17)

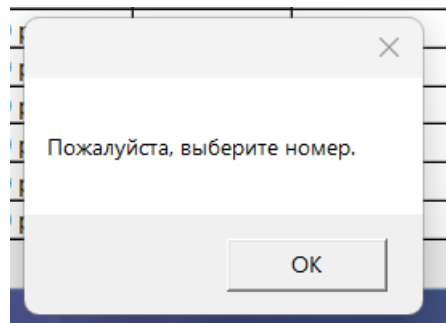


Рисунок 17 – Сообщение «Пожалуйста выберите номера»

Ожидаемый результат: пользователь должен выбрать номер из списка доступных вариантов.

Полученный результат: появилось сообщение "Пожалуйста, выберите номер" (Рисунок 17).

Решение проблемы: пользователь должен выбрать один из предложенных номеров.

## 2. Тестирование пользовательского интерфейса:

Проверила интерфейса: удобство и интуитивность

- Убедилась, что все поля ввода и кнопки работают корректно.
- Проверила сохранение бронирования.

## 3. Нагрузочное тестирование:

Загрузила программу с множеством одновременных запросов чтобы оценить, как она справляется под давлением.

- Определила порог нагрузки, при котором производительность начинает ухудшаться.

## 4. Тестирование на отказоустойчивость:

Смоделировала ситуации, когда происходят сбои, такие как отключение сети:

- Оценивала, как программа реагирует на ошибки восстанавливает работу.

## 5. Результаты тестирования:

При тестировании программы нарушений в работе программы не наблюдалось. Ввод и вывод данных выполняется корректно.

#### 6. Заключение:

По итогам тестирования можно сделать вывод, что программа «HotelGalaxy» выполняет свои функции корректно, но требует доработки в некоторых областях для повышения удобства использования и надежности. Все ошибки были фиксированы, что позволит улучшить качество финального продукта.

Тестирование программы осуществлялось на персональном компьютере со следующими техническими характеристиками:

- ✓ Процессор – Intel(R) Core(TM) i5-12600H CPU @ 144GHz
- ✓ Оперативная память – DDR3 16 ГБ
- ✓ Видеокарта – RTX 2050 4 ГБ
- ✓ Операционная система – Windows 11

## 5. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Функциональным назначением программы является бронирование мест в гостинице.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- Авторизация пользователя
- Просмотр информации о номерах
- Поиск свободных номеров
- Бронирование номера
- Сохранение даты въезда в гостиницу и выезда из неё
- Сохранение введенных контактных данных
- Заключение договора о бронировании

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

- процессор с тактовой частотой, 1 ГГц, не менее;
- оперативную память объемом 512 Мб, не менее;
- жесткий диск со свободным местом 500 Мб, не менее;
- монитор, с разрешением экрана 1024\*768, не менее;
- оптический привод;
- компьютерная мышь;
- клавиатура.

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Все пользователи должны обладать навыками работы с графическим пользовательским интерфейсом операционной системы.

Выполнение программы. Для запуска программного продукта необходимо запустить «HotelGalaxy» с расширением exe (Рисунок 18).

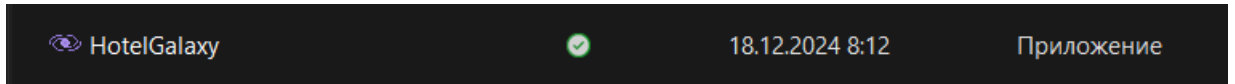


Рисунок 18 – «HotelGalaxy» с расширением exe.

Для того чтобы начать процесс бронирования места в гостинице нужно ввести Логин и Пароль, а после этого нажать кнопку «Войти» (Рисунок 19).

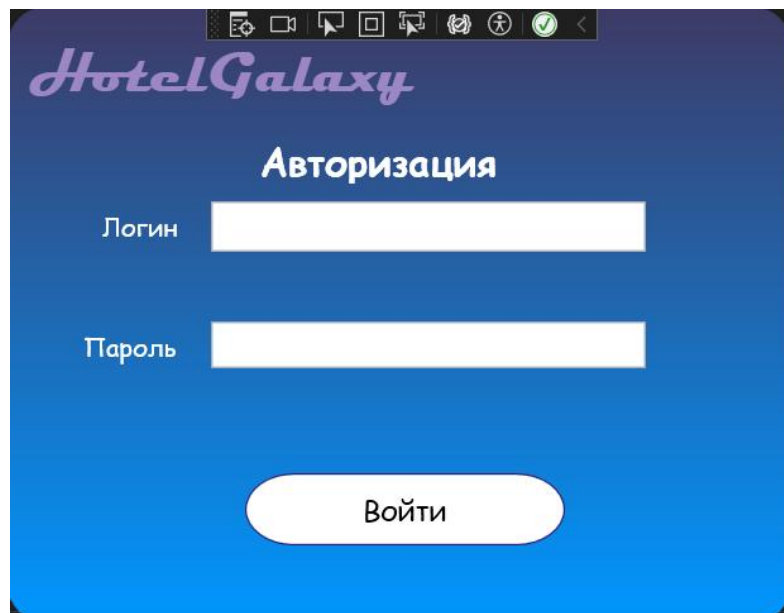


Рисунок 19 – Окно «Авторизация»

Если введены неверные данные или поля пусты, то появляется сообщение «Неверно введен логин или пароль» (Рисунок 20).



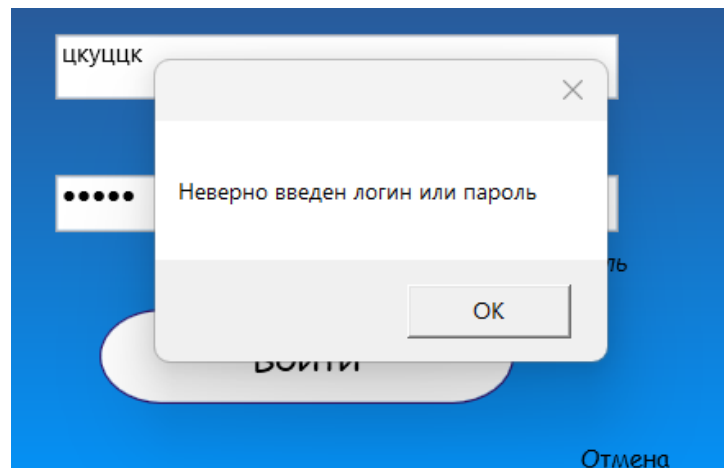


Рисунок 20 – «Неверно введен логин или пароль»

После успешной авторизации открывается окно администратора (Рисунок 21). Администратор может просматривать список всех номеров, их категории, цены и статус доступности. Может просматривать список всех текущих бронирований, просматривать информацию обо всех гостях и подробную информацию обо всех договорах. Чтобы забронировать гостя администратору нужно воспользоваться кнопкой "Узнать наличие свободных мест и забронировать".

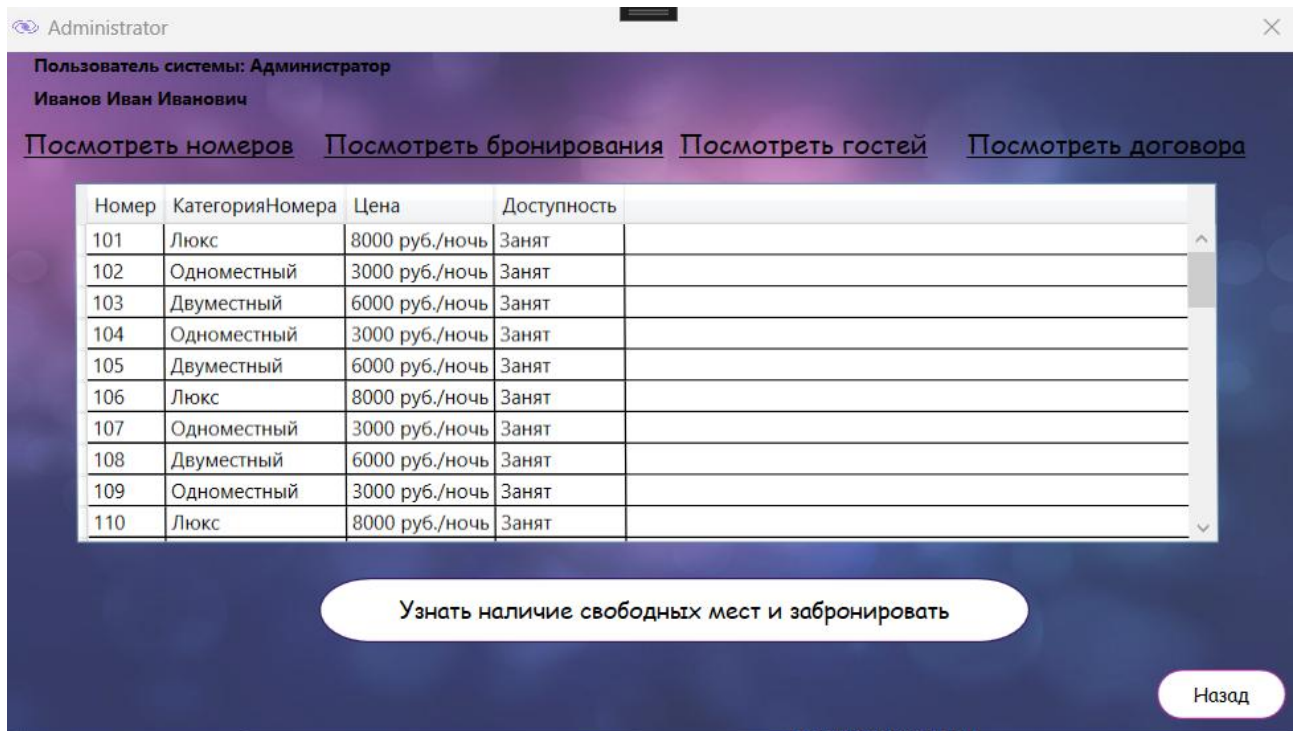


Рисунок 21 – «Окно Администратора»

Далее запускается окно "Номера"(Рисунок 22), где пользователю предлагается выбрать тип данных для поиска доступных номеров. После выбора типа данных необходимо нажать на кнопку "Узнать наличие свободных мест". Эта операция инициирует поиск доступных номеров в системе. На экране выводится список доступных номеров, удовлетворяющих выбранным критериям. Пользователь просматривает представленные номера, выбирает понравившийся и нажимает кнопку "Выбрать". Это действие подтверждает выбор номера для дальнейшего бронирования.

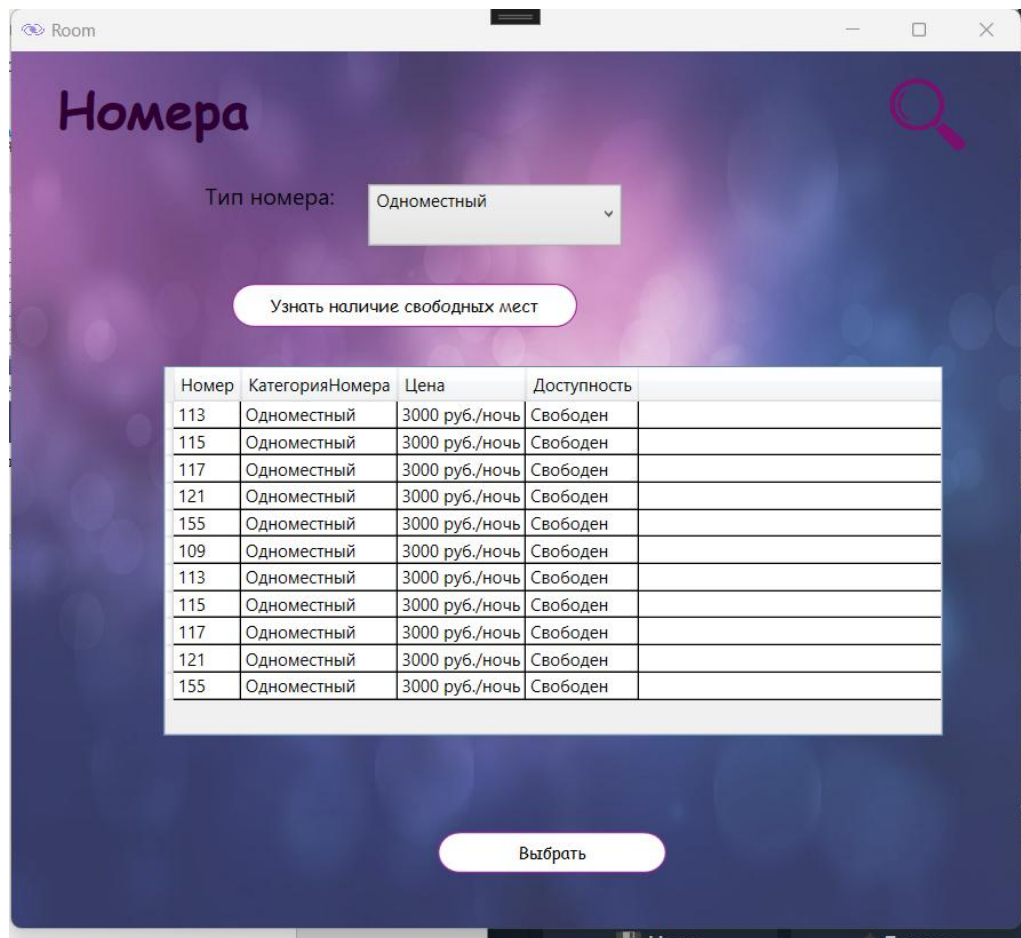


Рисунок 22 – «Окно Номера»

Если пользователь не выбрал тип номера, программа отображает сообщение: "Пожалуйста, выберите тип номера, чтобы." (Рисунок 23). Это предупреждение помогает пользователю понять, что необходимо сделать для продолжения процесса. "

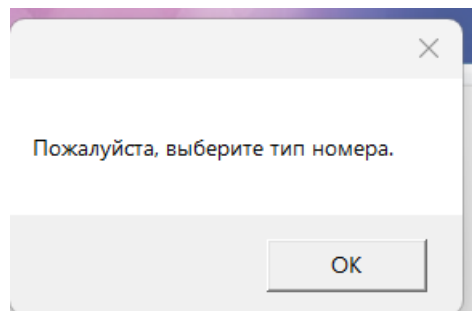


Рисунок 23 – Сообщение «Пожалуйста, выберите тип номера»

Если после выбора типа данных не было выбрано ни одного доступного номера, программа выводит сообщение: "Пожалуйста, выберите номер." (Рисунок 24). Это информирует пользователя о необходимости сделать выбор для завершения операции.

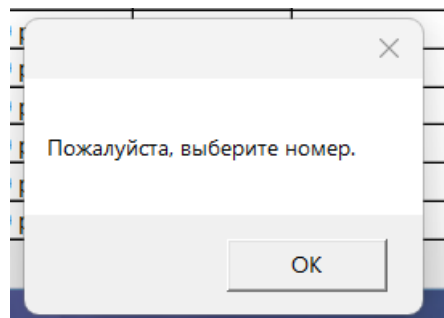


Рисунок 24 – Сообщение «Пожалуйста, выберите номер»

После подтверждения выбора номера открывается окно «Бронирование». В этом окне необходимо указать дату заезда в гостиницу и отъезда из неё. Выбрать количество человек и заполнить поля контактными данными гостя. Далее выбрать нужные дополнительные услуги. После заполнения всех полей и выбора услуг можно нажать кнопку «Забронировать» (Рисунок 25).

Бронирование номера

Период и тип номера

Заезд: Выбор даты 15 Выезд: Выбор даты 15 Количество дней:

Тип номера: Одноместный Номер: 117

Количество человек: 1

Контактные данные

ФИО

Телефон

Серия и номер паспорта

Дополнительные услуги

☐ Завтрак

☐ Такси

☐ Уборка номера

Цена 0,00 Р

Стоимость номера 3 000,00 Р

Общая стоимость 3 000,00 Р

Назад Забронировать

Рисунок 25 – Окно «Бронирование»

Если контактные данные гостя будут пустыми, то программа выдаст сообщение о том, что нужно заполнить поля для гостя. (Рисунок 26)

Контактные данные

ФИО

Телефон

Серия и номер паспорта

Дополнительные услуги

Пожалуйста, заполните все поля для гостя.

OK

Рисунок 26 – Сообщение «Пожалуйста заполните все поля для гостя»

Также если даты заезда и выезда не заполнены выводится сообщение «Выберите даты заезда и выезда» (Рисунок 27), или если введены некорректно выскочит сообщение «Дата выезда должна быть позже даты заезда» (Рисунок 28)

 A screenshot of a web form with a light blue background. It contains two date pickers: 'Заезд:' (Check-in) with the date '18.12.2024' and a calendar icon, and 'Выезд:' (Check-out) with the text 'Выбор даты' and a calendar icon. To the right of these is a label 'Количество дней:' (Number of days) and a message 'Выберите даты заезда и выезда' (Select check-in and check-out dates). Below the date pickers, there is a partially visible input field containing the number '117'.

Рисунок 27 – Сообщение «Выберите даты заезда и выезда»

 A screenshot of a web form similar to the previous one. The 'Заезд:' date picker shows '20.12.2024'. The 'Выезд:' date picker shows '12.12.2024'. To the right, the label 'Количество дней:' is present, and a message 'Дата выезда должна быть позже даты заезда' (Check-out date must be later than check-in date) is displayed in red text.

Рисунок 28 – Сообщение «Дата выезда должна быть позже даты заезда»

При заполнении телефона используется валидация, которая проверяет корректность введенного номера телефона, сопоставляя его с шаблоном +7(XXX)XXX-XX-XX. Если номер телефона не соответствует этому шаблону, то выводится сообщение об ошибке. (Рисунок 29)

 A screenshot of a form section for a phone number. It has a label 'Телефон:' (Phone) in blue. Next to it is an input field containing the number '+7(999)375-24'. Below the input field, a red error message is displayed: 'Номер телефона должен быть в формате +7(XXX)XXX-XX-XX' (Phone number must be in the format +7(XXX)XXX-XX-XX).

Рисунок 29 – Сообщение «Выберите даты заезда и выезда»

После успешного бронирования открывается договор в формате PDF.(Рисунок 30)

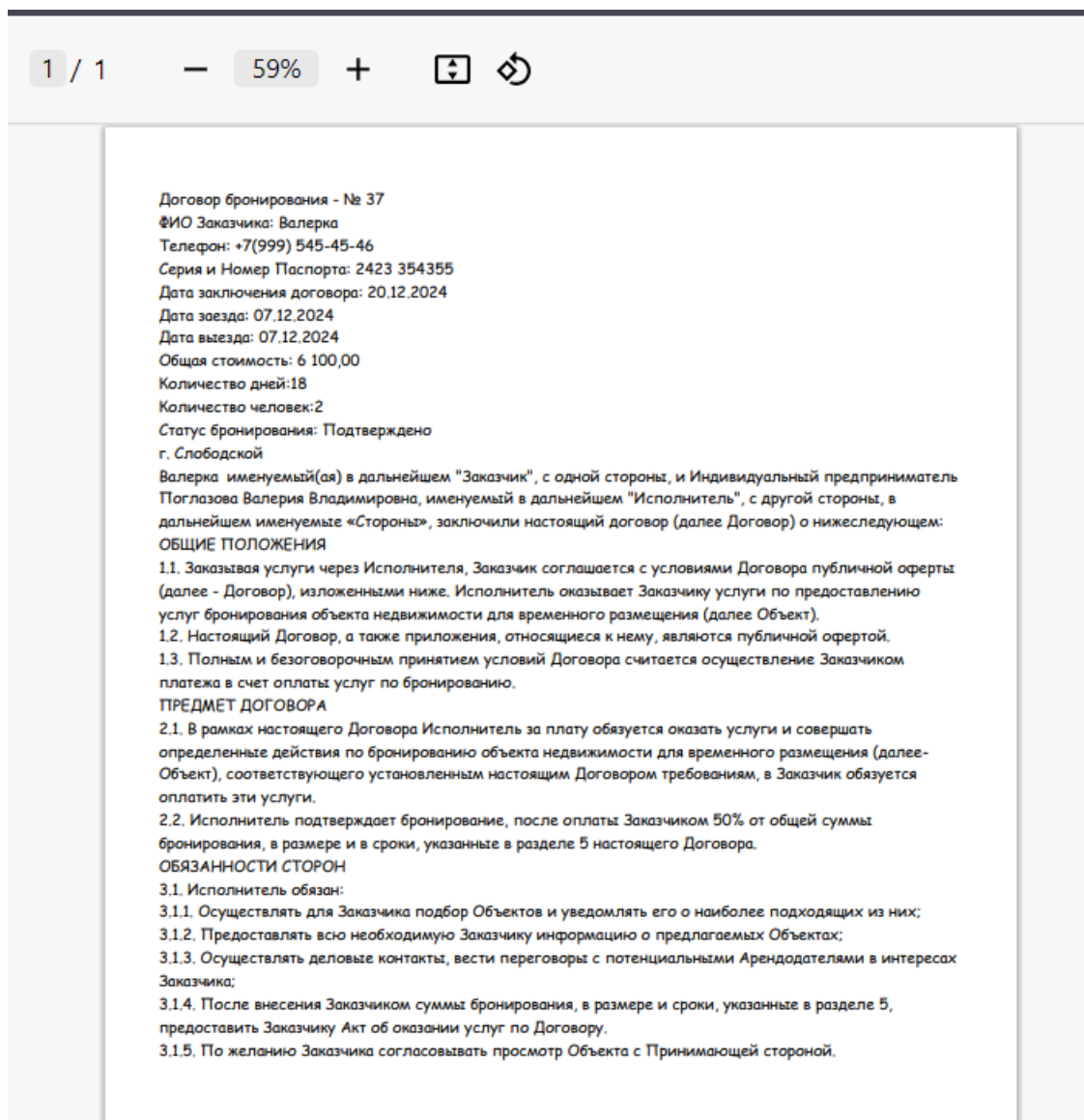


Рисунок 30 – «Договор бронирования»

## ЗАКЛЮЧЕНИЕ

В результате выполнения курсового проекта была создана программа для бронирования мест в гостинице. Основные задачи – предоставление пользователю понятного интерфейса для бронирования мест в гостинице – были осуществлены в ходе разработки приложения. Были выполнены следующие поставленные задачи:

1. Поиск свободных номеров.
2. Создание и сохранение бронирования номера.
3. Создание договора
4. Удобный интерфейс, интуитивно понятный пользователю.

Программа обладает функциями:

1. Авторизация пользователя;
2. Просмотр информации о номерах
3. Поиск свободных номеров
4. Бронирование номера
5. Сохранение даты въезда в гостиницу и выезда из неё
6. Сохранение введенных контактных данных
7. Заключение договора о бронировании

Было осуществлено тестирование программы:

1. Ошибок не обнаружено.
2. Программа работает корректно.
3. Устойчивый вычислительный процесс.

В данном приложении пользователь может с легкостью защитить свое изображение и текстовую информацию от вредоносного использования злоумышленников. Пользователи, которые ищут способ защитить свои изображения могут воспользоваться этим приложением для защиты данных.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ Р 51185-2014 Туристские услуги. Средства размещения. Общие требования. – М.: Стандартинформ, 2015.
2. Байлик С. И. Гостиничное хозяйство [Текст]: учебник / С. И. Байлик. – М. : Юнити, 2014. – 231 с.
3. Ехина, М. А. Организация обслуживания в гостиницах [Текст] / М. А. Ехина. – М. : «Академия», 2014. – 208 с.
4. Жидкова, Н. Индустрия гостеприимства [Текст] / Н. Жидкова // Хабар. Вести. – 2014. – 18 ноября. – С. 6.
5. Организация деятельности служб бронирования гостиничных услуг (учебно-методическое пособие), 2016. [Электронный ресурс] URL: [https://isgz.ru/sveden/files/43.02.11\\_SO\\_02\\_MDK\\_01.01\\_Uchebno-metodicheskoe\\_posobie\\_po\\_ODS\\_bronirovanirovaniya\\_gostinichnyh\\_uslug.pdf](https://isgz.ru/sveden/files/43.02.11_SO_02_MDK_01.01_Uchebno-metodicheskoe_posobie_po_ODS_bronirovanirovaniya_gostinichnyh_uslug.pdf)
6. Организация и технологии работы службы бронирования в гостинице. [Электронный ресурс] URL: <https://studfile.net/preview/7156201/page:15/>
7. Порядок бронирования, оформления и оплаты проживания в гостинице \КонсультантПлюс. [Электронный ресурс] URL: [https://www.consultant.ru/document/cons\\_doc\\_LAW\\_3899/8b56fd7985fab41b2e0a9367e94b510f4424467b/](https://www.consultant.ru/document/cons_doc_LAW_3899/8b56fd7985fab41b2e0a9367e94b510f4424467b/)
8. Постановление Правительства РФ от 18.11.2020 N 1853 (ред. от 20.03.2024) "Об утверждении Правил предоставления гостиничных услуг в Российской Федерации" \ КонсультантПлюс [Электронный ресурс] URL: [https://www.consultant.ru/document/cons\\_doc\\_LAW\\_368292/](https://www.consultant.ru/document/cons_doc_LAW_368292/)
9. C# - iText7 for .NET barcode - Stack Overflow [Электронный ресурс] URL: <https://stackoverflow.com/questions/54004849/itext7-for-net-barcode>
10. MS SQL Server 2022 и T-SQL [Электронный ресурс] URL: <https://metanit.com/sql/sqlserver/>



11. WPF и C# | Полное руководство [Электронный ресурс] URL:  
<https://metanit.com/sharp/wpf/>

## **ПРИЛОЖЕНИЯ**

## Файл MainWindow.xaml.cs

```

        {   MessageBox.Show("Неверно введен
логин или пароль");}

        else

            {   MessageBox.Show("Вход успешно
выполнен");

                var usern = db.Role.FirstOrDefault(role =>
role.ID_role == userPersonal.ID);

                if (usern != null)

                    { if (usern.Роль== "Администратор")

                        {

                            Administrator rn = new
Administrator();

                                // Получение имени и фамилии
сотрудника из таблицы User

                                    var employee =
db.User.FirstOrDefault(e1 => e1.ID == usern.ID_role);

                                    if (employee != null)

                                        {

                                            rn.user.Text = $"{employee.ФИО}";

                                        }

                                        // Получение роли сотрудника из
таблицы Role

                                            var role = db.Role.FirstOrDefault(r =>
r.ID_role == employee.ID);

                                            if (role != null)

                                                {

                                                    rn.role.Text = $"Пользователь
системы: {role.Роль}";

                                                }

                                                    rn.Show();

                                                    this.Close();

                                                }

                                            else if(usern.Роль == "Менеджер")

                                                {

                                                    Administrator rn = new
Administrator();

                                                        // Получение имени и фамилии
сотрудника из таблицы User

```

```

        var employee =
db.User.FirstOrDefault(el => el.ID == usern.ID_role);
        if (employee != null)
        {
            rn.user.Text =
${employee.ФИО}";
        }

        // Получение роли сотрудника из
таблицы Role
        var role = db.Role.FirstOrDefault(r =>
r.ID_role == employee.ID);
        if (role != null)
        {
            rn.role.Text = $"Пользователь
системы: {role.Роль}";
        }
        rn.Show();
        this.Close();
    }
}
}

// Отображает введенный пароль в текстовом
поле
private void ShowPassword_Checked(object sender,
RoutedEventArgs e)
{
    // Показать пароль
    if (password != null)
        par.Text = password.Password;
        par.Visibility = Visibility.Visible;
    }
    // Скрывает введенный пароль в текстовом поле
private void HidePassword_Unchecked(object
sender, RoutedEventArgs e)
{
    // Скрыть пароль
    if (password != null)
        password.Password = par.Text;

    par.Visibility = Visibility.Hidden;

```

```

    }
    // Открывает окно регистрации нового
пользователя
    private void Register_Click(object sender,
RoutedEventArgs e)
    {
        Registration fn = new Registration();
        fn.Show();
        this.Close();
    }
    //Выход из программы
    private void NewClose_Click(object sender,
RoutedEventArgs e)
    {
        this.Close();
    }
    private void Forgot_Click(object sender,
RoutedEventArgs e)
    {
    }
}

```

## Файл MainWindow.xaml

```

<Window x:Class="HotelGalaxy.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/p
resentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml
"
xmlns:d="http://schemas.microsoft.com/expression/blend/
2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-
compatibility/2006"
xmlns:local="clr-namespace:HotelGalaxy"
mc:Ignorable="d"
WindowStartupLocation="CenterScreen"
AllowsTransparency="True" Background="Transparent"
WindowStyle="None"
Title="Авторизация" Height="392" Width="500"
ResizeMode="NoResize" Foreground="White">

```

```

<Grid HorizontalAlignment="Left" Height="406"
Margin="0,0,0,-1">
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="57*"/>
    <ColumnDefinition Width="120*"/>
    <ColumnDefinition Width="73*"/>
  </Grid.ColumnDefinitions>
  <Grid.RowDefinitions>
    <RowDefinition Height="84*"/>
    <RowDefinition Height="57*"/>
    <RowDefinition Height="49*"/>
    <RowDefinition Height="84*"/>
    <RowDefinition Height="94*"/>
    <RowDefinition Height="38*"/>
  </Grid.RowDefinitions>
  <Grid Margin="0,0,114,8" Grid.RowSpan="5">
    <Grid.RowDefinitions>
      <RowDefinition/>
    </Grid.RowDefinitions>

    <Border CornerRadius="25 25 25 25"
Margin="0,0,-496,-30">
      <Border.Background>
        <LinearGradientBrush StartPoint="0,0"
EndPoint="0,1">
          <GradientStop Color="#3E3A65"
Offset="0"/>
          <GradientStop Color="#FF0097FF"
Offset="1"/>
        </LinearGradientBrush>
      </Border.Background>
    </Border>
  </Grid>

  <Label Content="Авторизация"
HorizontalAlignment="Left" Margin="41,0,0,0"
VerticalAlignment="Center" Height="42" Width="167"
FontSize="24" FontFamily="Comic Sans MS"
FontWeight="Bold" Foreground="White" Cursor=""
Grid.Column="1" Grid.Row="1"/>

```

```

  <Label Content="Логин"
HorizontalAlignment="Left" Margin="55,9,0,0"
VerticalAlignment="Top" FontSize="16"
FontFamily="Comic Sans MS" Height="44" Width="62"
RenderTransformOrigin="0.984,0.477"
Foreground="White" BorderBrush="#FFFFFFEFE"
Grid.Row="2" Grid.ColumnSpan="2"
Grid.RowSpan="2"/>
  <Label Content="Пароль"
HorizontalAlignment="Left" Margin="43,28,0,0"
VerticalAlignment="Top" FontSize="16"
FontWeight="ExtraLight" FontFamily="Comic Sans MS"
Height="44" Width="74"
RenderTransformOrigin="0.851,0.432"
Foreground="White" BorderBrush="#FFFFFFEFE"
Grid.Row="3" Grid.ColumnSpan="2"/>
  <TextBox x:Name="log"
HorizontalAlignment="Left" Height="32"
Margin="15,0,0,0" TextWrapping="Wrap"
VerticalAlignment="Center" Width="275" Grid.Row="2"
Grid.ColumnSpan="2" Grid.Column="1"/>
  <PasswordBox x:Name="password"
Grid.Column="1" HorizontalAlignment="Left"
Margin="15,0,0,0" Grid.Row="3"
VerticalAlignment="Center" Width="275"
Grid.ColumnSpan="2" Height="28"/>
  <TextBox x:Name="par" Visibility="Hidden"
HorizontalAlignment="Left" Height="28"
Margin="15,0,0,0" TextWrapping="Wrap"
VerticalAlignment="Center" Width="275" Grid.Row="3"
Grid.ColumnSpan="2" Grid.Column="1"/>
  <Button x:Name="Login" Content="Войти"
HorizontalAlignment="Left" Margin="36,10,0,0"
VerticalAlignment="Top" Width="203" Height="46"
Background="{DynamicResource {x:Static
SystemColors.ControlLightLightBrushKey}}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="18" FontWeight="ExtraLight"
Click="Login_Click" FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315" Grid.Row="4"
Grid.Column="1" Grid.ColumnSpan="2">

```

```

        <Button.Resources>
            <Style TargetType="{x:Type Border}">
                <Setter Property="CornerRadius"
Value="30"/>
            </Style>
        </Button.Resources>
        <Button.BorderBrush>
            <SolidColorBrush Color="#FF3B146A"
Opacity="0.991"/>
        </Button.BorderBrush>
    </Button>
    <Label Content="HotelGalaxy"
HorizontalAlignment="Left" Margin="10,25,0,0"
VerticalAlignment="Top" FontSize="36"
FontFamily="Magneto" Height="60" Width="263"
FontWeight="Bold" Foreground="#FF967ADC"
Grid.ColumnSpan="2" Grid.Column="1"
Grid.RowSpan="2"/>
    <Image Margin="16,6,10,37"
Source="/Images/icong.png" Stretch="Fill"
Grid.RowSpan="2"/>
    <Button x:Name="NewClose"
HorizontalAlignment="Left" Margin="44,72,0,0"
VerticalAlignment="Top" Width="82" Height="22"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="12" Click="NewClose_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315" Grid.Row="4"
Grid.Column="2" Cursor="Hand"
BorderBrush="{x:Null}">

```

## Файл Administrator.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;

```

```

        <TextBlock TextDecorations="Underline"
Text="Отмена"/>
    </Button>
    <Button x:Name="Forgot"
HorizontalAlignment="Left" Margin="38,61,0,0"
VerticalAlignment="Top" Width="114" Height="22"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="12" Click="Forgot_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315" Grid.Row="3"
Grid.Column="1" Cursor="Hand"
BorderBrush="{x:Null}">
        <CheckBox x:Name="ShowPassword"
Content="Показать пароль"
HorizontalAlignment="Left" Margin="177,64,0,0"
Grid.Row="3" VerticalAlignment="Top"
Checked="ShowPassword_Checked" Unchecked="
HidePassword_Unchecked" Grid.Column="1"
Grid.ColumnSpan="2" FontStyle="Italic"/>
    </Grid>
</Window>

FontSize="18" Foreground="White"
IsReadOnly="True"></TextBox>
</Grid>
</Window>

```

```

using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HotelGalaxy
{

```

```

public partial class Administrator : Window
{
    public Administrator()
    {
        InitializeComponent();
        using (var db = new GalaxyEntities())
        {
            var rooms = (from room in db.Room
                select new
                {
                    Номер = room.Номер,
                    КатегорияНомера =
room.Категория_номера,
                    Цена = room.Цена,
                    Доступность =
room.Доступность
                }).ToList();
            data.ItemsSource = rooms;
        }
    }
    // Открываем окно для просмотра доступных
номеров
    private void Bron_Click(object sender,
RoutedEventArgs e)
    {
        Rooms fn = new Rooms();
        fn.Show();
    }
    // Извлекаем список доступных номеров из
базы данных и отображает его в элементе данных
    private void ViewRoom_Click(object sender,
RoutedEventArgs e)
    {
        using (var db = new GalaxyEntities())
        {
            var rooms = (from room in db.Room
                select new
                {
                    Номер = room.Номер,
                    КатегорияНомера =
room.Категория_номера,
                    Цена = room.Цена,
                    Доступность =
room.Доступность
                }).ToList();
            data.ItemsSource = rooms;
        }
    }
    // Возврат на окно авторизации и закрываем
текущее окно
    private void Button_Click(object sender,
RoutedEventArgs e)
    {
        MainWindow fn = new MainWindow();
        fn.Show();
        this.Close();
    }
    // Извлекаем список всех бронирований и
связанных с ними данных из базы данных и
отображает его в элементе данных
    private void ViewBron_Click(object sender,
RoutedEventArgs e)
    {
        using (var db = new GalaxyEntities())
        {
            var brons = (from bron in db.Booking
                join gu in db.Guest on bron.Гость
equals gu.ID_Guest
                join nu in db.Room on bron.Номер
equals nu.ID_room
                select new
                {
                    Номер = nu.Номер,
                    ТипНомера =
nu.Категория_номера,
                    Гость = gu.ФИО,
                    Заезд = bron.Дата_заезда,
                    Выезд = bron.Дата_выезда,
                    Количество_Дней =
bron.Количество_дней_проживания,
                    КоличествоЧеловек =
bron.Количество_человек,

```

```

        ДополнительныеУслуги =
        (from agr in db.Agreement
            join ag in
            db.AdditionalService on agr.Список_услуг equals
            ag.ID_AdditionalService
            where
            agr.Бронирование == bron.ID_Booking
            select
            ag.Название).ToList()
        }).ToList();
        // Объединяем дополнительные услуги в
        строку
        var bronsWithJoinedServices =
        brons.Select(b => new
        {
            Номер = b.Номер,
            ТипНомера = b.ТипНомера,
            Гость = b.Гость,
            Заезд = b.Заезд,
            Выезд = b.Выезд,
            Количество_Дней =
            b.Количество_Дней,
            КоличествоЧеловек =
            b.КоличествоЧеловек,
            СписокУслуг = string.Join(", ",
            b.ДополнительныеУслуги)
        }).ToList();
        // Устанавливаем источник данных для
        таблицы
        data.ItemsSource =
        bronsWithJoinedServices;
    }
}
// Извлекаем информацию о гостях и
связанных с ними бронированиях из базы данных
и отображает ее в элементе данных
private void ViewGu_Click(object sender,
RoutedEventArgs e)
{
    using (var db = new GalaxyEntities())
    {
        var brons = (from bron in db.Booking

```

```

        join gu in db.Guest on bron.Гость
        equals gu.ID_Guest
        join nu in db.Room on bron.Номер
        equals nu.ID_room
        select new
        {
            Гость = gu.ФИО,
            Телефон = gu.Телефон,
            Паспорт =
            gu.Серия_и_Номер_Паспорта,
            Номер = nu.Номер,
            ТипНомера =
            nu.Категория_номера,
        }).ToList();
        var bronsWithJoinedServices =
        brons.Select(b => new
        {
            Гость = b.Гость,
            Телефон = b.Телефон,
            Паспорт = b.Паспорт,
            Номер = b.Номер,
            ТипНомера = b.ТипНомера
        }).ToList();

        data.ItemsSource =
        bronsWithJoinedServices;
    }
}
// Извлекает информацию о договорах,
связанных с бронированиями, и отображает ее в
элементе данных
private void ViewGo_Click(object sender,
RoutedEventArgs e)
{
    using (var db = new GalaxyEntities())
    {
        var brons = (from ag in db.Agreement
            join bo in db.Booking on
            ag.Бронирование equals bo.ID_Booking
            join r in db.Room on bo.Номер
            equals r.ID_room

```



```

        join g in db.Guest on bo.Гость
equals g.ID_Guest

        join s in db.AdditionalService on
ag.Список_услуг equals s.ID_AdditionalService

select new
{
    НомерБрони = bo.ID_Booking,
    Номер = r.Номер,
    Гость = g.ФИО,
    Дата_Заключения =
ag.Дата_заключения,
    Тип_номера =
r.Категория_номера,
    Количество_Человек =
bo.Количество_человек,
    Количество_Дней =
bo.Количество_дней_проживания,
    СписокУслуг = (from agr in
db.Agreement
                        join ag in
db.AdditionalService on agr.Список_услуг equals
ag.ID_AdditionalService
                        where
ag.Бронирование == bo.ID_Booking
                        select
ag.Название).ToList(),
    Общая_стоимость =
ag.Общая_стоимость
}).ToList();

// Объединяем дополнительные услуги в
строку
var bronsWithJoinedServices =
brons.Select(b => new
{
    НомерБрони = b.НомерБрони,
    Номер = b.Номер,
    Гость = b.Гость,
    Дата_Заключения =
b.Дата_Заключения,
    Тип_номера = b.Тип_номера,
    Количество_Человек =
b.Количество_Человек,

```

```

        Количество_Дней =
b.Количество_Дней,
        СписокУслуг = string.Join(" ",
b.СписокУслуг)
}).ToList();

data.ItemsSource = brons;
}
}
}
}

```

## Файл Administrator.xaml

```

<Window x:Class="HotelGalaxy.Administrator"

xmlns="http://schemas.microsoft.com/winfx/2006/xa
ml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/
xaml"
xmlns:d="http://schemas.microsoft.com/expression/b
lend/2008"
xmlns:mc="http://schemas.openxmlformats.org/mark
up-compatibility/2006"
        xmlns:local="clr-namespace:HotelGalaxy"
        mc:Ignorable="d"
        Title="Администратор" Height="450"
Width="800" ResizeMode="NoResize">
    <Grid>
        <Grid.Background>
            <ImageBrush
ImageSource="/Images/fon.jpg" Opacity="0.8"/>
        </Grid.Background>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="261*"/>
            <ColumnDefinition Width="449*"/>
            <ColumnDefinition Width="90*"/>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="159*"/>
            <RowDefinition Height="32*"/>
            <RowDefinition Height="26*"/>
        </Grid.RowDefinitions>

```

```

        <DataGrid x:Name="data"
Margin="45,79,50,6" AutoGenerateColumns="True"
IsReadOnly="True"
RenderTransformOrigin="0.5,0.5"
Grid.ColumnSpan="3" >
    <DataGrid.RenderTransform>
        <TransformGroup>
            <ScaleTransform/>
            <SkewTransform AngleX="0.345"/>
            <RotateTransform/>
            <TranslateTransform X="0.494"/>
        </TransformGroup>
    </DataGrid.RenderTransform>
</DataGrid>

    <StackPanel Margin="8,0,0,267"
Grid.ColumnSpan="1">
        <TextBlock x:Name="role"
FontWeight="Bold" FontSize="11" Width="224"
Height="20"
RenderTransformOrigin="0.494,0.669"><Run
Text="II:" FontFamily="Comic Sans
MS"/></TextBlock>
        <TextBlock x:Name="user"
FontWeight="Bold" FontSize="11" Height="26"
Width="224"><Run Text="I" FontFamily="Comic
Sans MS"/></TextBlock>
    </StackPanel>
    <Button x:Name="Bron" Content="Узнать
наличие свободных мест и забронировать"
HorizontalAlignment="Left" Margin="193,15,0,0"
VerticalAlignment="Top" Width="429" Height="39"
Background="{DynamicResource {x:Static
SystemColors.ControlLightLightBrushKey}}"
BorderBrush="#FF3B146A"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="14" FontWeight="ExtraLight"
Click="Bron_Click" FontFamily="Comic Sans MS"
Grid.Row="1" Grid.ColumnSpan="2">
        <Button.Resources>
            <Style TargetType="{x:Type Border}">

```

```

                <Setter Property="CornerRadius"
Value="30"/>
            </Style>
        </Button.Resources>
    </Button>
    <Button Content="Назад"
HorizontalAlignment="Left" Margin="2,0,0,0"
VerticalAlignment="Center" Width="78"
Click="Button_Click" BorderBrush="#FFA026A0"
Background="White" Height="30"
FontFamily="Comic Sans MS" Grid.Row="2"
Grid.Column="2">
        <Button.Resources>
            <Style TargetType="{x:Type Border}">
                <Setter Property="CornerRadius"
Value="15" />
            </Style>
        </Button.Resources>
    </Button>
    <Button x:Name="ViewNum"
HorizontalAlignment="Left" Margin="8,42,0,0"
VerticalAlignment="Top" Width="185" Height="24"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="16" Click="ViewRoom_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315"
Cursor="Hand" BorderBrush="{x:Null}">
        <TextBlock TextDecorations="Underline"
Text="Посмотреть номеров" Width="173"
Height="21" FontSize="16"/>
    </Button>
    <Button x:Name="ViewBron"
HorizontalAlignment="Left" Margin="193,42,0,0"
VerticalAlignment="Top" Width="215" Height="24"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="12" Click="ViewBron_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315"

```

```

Cursor="Hand" BorderBrush="{x:Null}"
Grid.ColumnSpan="2">
    <TextBlock TextDecorations="Underline"
Text="Посмотреть бронирования" Width="210"
Height="21" FontSize="16"/>
</Button>
<Button x:Name="ViewGu"
HorizontalAlignment="Left" Margin="152,42,0,0"
VerticalAlignment="Top" Width="157" Height="24"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="12" Click="ViewGu_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315"
Cursor="Hand" BorderBrush="{x:Null}"
Grid.Column="1">
    <TextBlock TextDecorations="Underline"
Text="Посмотреть гостей" Width="157"
Height="21" FontSize="16"/>
</Button>
<Button x:Name="ViewDO"
HorizontalAlignment="Left" Margin="326,42,0,0"
VerticalAlignment="Top" Width="181" Height="24"
Background="{x:Null}"
VerticalContentAlignment="Center"
ScrollViewer.HorizontalScrollBarVisibility="Auto"
FontSize="12" Click="ViewGo_Click"
FontFamily="Comic Sans MS"
RenderTransformOrigin="0.633,7.315"
Cursor="Hand" BorderBrush="{x:Null}"
Grid.Column="1" Grid.ColumnSpan="2">
    <TextBlock TextDecorations="Underline"
Text="Посмотреть договора" Width="181"
Height="21" FontSize="16"/>
</Button>
</Grid>
</Window>

```

## Файл Rooms.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Data.SqlClient;
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.Markup;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace HotelGalaxy
{
    public partial class Rooms : Window
    {
        public Rooms()
        {
            InitializeComponent();
        }
        // Получение выбранного номера
        private void Choose_Click(object sender,
RoutedEventArgs e)
        {
            var selectedRoom = dgnum.SelectedItem;
            if (selectedRoom == null)
            {
                MessageBox.Show("Пожалуйста,
выберите номер.");
                return;
            }
            // Получаем номер из выбранного элемента
            var roomData = selectedRoom as dynamic;
            string roomNumber = roomData?.Номер;

            if (string.IsNullOrEmpty(roomNumber))
            {

```

```

        MessageBox.Show("Ошибка: выбранный
номер недоступен.");
        return;
    }
    // Получаем выбранный тип номера из
ComboBox
    string selectedRoomType =
(tiproom.SelectedItem as
ComboBoxItem)?.Content.ToString();
    if
(string.IsNullOrEmpty(selectedRoomType))
    {
        MessageBox.Show("Пожалуйста,
выберите тип номера.");
        return;
    }
    // Открываем новое окно и передаем номер
и тип
    var reservationWindow = new
HotelGalaxy.Reservation();
reservationWindow.SetRoomNumber(roomNumber);
reservationWindow.SetRoomType(selectedRoomType);
    reservationWindow.Show();
    // this.Close();
}

//Поиск свободных номеров
private void Search_Click(object sender,
RoutedEventArgs e)
{
    string selectedRoomType =
(tiproom.SelectedItem as
ComboBoxItem)?.Content.ToString();
    if
(string.IsNullOrEmpty(selectedRoomType))
    {
        MessageBox.Show("Пожалуйста,
выберите тип номера.");
        return;
    }

```

```

try
{
    using (var db = new GalaxyEntities())
    {
        var rooms = (from room in db.Room
                      where room.Категория_номера
== selectedRoomType && room.Доступность ==
"Свободен"
                      select new
                      {
                          Номер = room.Номер,
                          КатегорияНомера =
room.Категория_номера,
                          Цена = room.Цена,
                          Доступность =
room.Доступность
                      }).ToList();
        // Устанавливаем источник данных
для DataGrid
        dgnum.ItemsSource = rooms;
    }
}
catch (Exception ex)
{
    MessageBox.Show($"Ошибка при поиске
номеров: {ex.Message}");
}
}
}

```

## Файл Rooms.xaml

```
<Window x:Class="HotelGalaxy.Rooms"
```

```

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

```

```

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

```

```
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
```

```
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
```

```
xmlns:local="clr-namespace:HotelGalaxy"
```

```
mc:Ignorable="d"
```

```
Title="Наличие доступных номеров"
```

```
Height="650" Width="721">
```

```
<Window.Resources>
```

```
<Style x:Key="ComboBoxStyle"
```

```
TargetType="ComboBox">
```

```
<Setter Property="BorderThickness"
```

```
Value="1"/>
```

```
<Setter Property="BorderBrush"
```

```
Value="#FFBBBBBB"/>
```

```
<!-- Светло-серый бордюр -->
```

```
<Setter Property="Background"
```

```
Value="White"/>
```

```
<Setter Property="Padding" Value="5,0"/>
```

```
<Setter Property="FontFamily"
```

```
Value="Arial"/>
```

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

```
<Setter Property="Width" Value="200"/>
```

```
<!-- Ширина комбобокса -->
```

```
<Setter Property="Margin" Value="0, 0, 0,
```

```
0"/>
```

```
<!-- Отступы -->
```

```
</Style>
```

```
</Window.Resources>
```

```
<Grid>
```

```
<Grid.RowDefinitions>
```

```
<RowDefinition Height="43*"/>
```

```
<RowDefinition Height="36*"/>
```

```
<RowDefinition Height="238*"/>
```

```
</Grid.RowDefinitions>
```

```
<Grid.ColumnDefinitions>
```

```
<ColumnDefinition Width="54*"/>
```

```
<ColumnDefinition Width="367*"/>
```

```
</Grid.ColumnDefinitions>
```

```
<Grid.Background>
```

```
<ImageBrush ImageSource="/fon.jpg"
```

```
Opacity="0.8"/>
```

```
</Grid.Background>
```

```
<Label Content="Homepa" FontSize="36"
```

```
FontWeight="Bold" HorizontalAlignment="Left"
```

```
Foreground="#FF300034"
```

```
VerticalAlignment="Top" Height="62"
```

```
FontFamily="Comic Sans MS" Width="353"
```

```
Margin="27,10,0,0" Grid.ColumnSpan="2" />
```

```
<Label Content="" HorizontalAlignment="Left"
```

```
Margin="92,0,0,0" VerticalAlignment="Center"
```

```
Grid.Row="1" Height="26" Width="10"
```

```
Grid.ColumnSpan="2"/>
```

```
<Label Content="" HorizontalAlignment="Left"
```

```
Margin="1,54,0,0" VerticalAlignment="Top"
```

```
Grid.Row="2" Height="26" Width="10"
```

```
Grid.Column="1"/>
```

```
<DataGrid x:Name="dgnum"
```

```
HorizontalAlignment="Left" Height="259"
```

```
Margin="16,67,0,0" VerticalAlignment="Top"
```

```
Width="545" IsReadOnly="True" Grid.Row="2"
```

```
Grid.Column="1"/>
```

```
<Button x:Name="Choose" Content="Выбрать"
```

```
HorizontalAlignment="Left" Margin="155,374,0,0"
```

```
VerticalAlignment="Top" Width="214"
```

```
Click="Choose_Click" BorderBrush="#FFA026A0"
```

```
Background="White" Height="46"
```

```
FontFamily="Comic Sans MS" Grid.Row="2"
```

```
Grid.Column="1">
```

```
<Button.Resources>
```

```
<Style TargetType="{x:Type Border}">
```

```
<Setter Property="CornerRadius"
```

```
Value="15" />
```

```
</Style>
```

```
</Button.Resources>
```

```
</Button>
```

```
<Button x:Name="Search" Content="Узнать
```

```
наличие свободных мест"
```

```
HorizontalAlignment="Left" Margin="16,10,0,0"
```

```
VerticalAlignment="Top" Width="278"
```

```
Click="Search_Click" BorderBrush="#FFA026A0"
```

```

Background="White" Height="32"
FontFamily="Comic Sans MS" Grid.Row="2"
Grid.Column="1">
    <Button.Resources>
        <Style TargetType="{x:Type Border}">
            <Setter Property="CornerRadius"
Value="15" />
        </Style>
    </Button.Resources>
</Button>
<Image Margin="445,10,40,19"
Source="/Images/poisk.png" Stretch="Fill"
Grid.Column="1" Grid.RowSpan="2"/>
    <Label Content="Тип номера:"
Foreground="#FF020006" Margin="40,2,450,4"
Grid.Column="1" FontSize="16" Grid.Row="1"
FontFamily="Comic Sans MS"/>
    <ComboBox x:Name="tiproom"
Grid.Column="1" Style="{StaticResource
ComboBoxStyle}" Margin="155,2,274,34"
Grid.Row="1">
        <ComboBoxItem Content="Люкс"/>
        <ComboBoxItem Content="Одноместный"/>
        <ComboBoxItem Content="Двухместный"/>
    </ComboBox>
</Grid>
</ Window>

```

## Файл Reservation.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Data.Entity.Infrastructure;
using System.Globalization;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Xml.Linq;
using iText.Kernel.Pdf;
using iText.Layout;

```

```

using iText.Layout.Element;
using iTextSharp.text;
using iTextSharp.text.pdf;
using System.IO;
using System.Text.RegularExpressions;
using System.Windows.Media;
using System.Windows.Input;
using System.Diagnostics;
using System.Data.SqlClient;
using System.Runtime.Remoting.Messaging;
using iText.Layout.Properties;
using Document = iTextSharp.text.Document;
using PdfWriter = iTextSharp.text.pdf.PdfWriter;
using Paragraph = iTextSharp.text.Paragraph;
namespace HotelGalaxy
{
    public partial class Reservation : Window
    {
        public Reservation()
        {
            InitializeComponent();
            DataContext = this;
            this.Loaded += Window_Loaded;
            tiproom.SelectionChanged +=
Tiproom_SelectionChanged;
            // Подписка на события Checked и
Unchecked для пересчета цены
            zav.Checked += CheckBox_Changed;
            zav.Unchecked += CheckBox_Changed;
            tak.Checked += CheckBox_Changed;
            tak.Unchecked += CheckBox_Changed;
            ybor.Checked += CheckBox_Changed;
            ybor.Unchecked += CheckBox_Changed;
        }
        // Словарь для хранения цен на
дополнительные услуги
        private Dictionary<string, decimal>
servicePrices = new Dictionary<string, decimal>()
        {
            {"Завтрак", 247.00m},
            {"Такси", 100.00m},
            {"Уборка номера", 146.00m},

```

```

};
// Словарь для хранения цен на типы номеров
private Dictionary<string, decimal> roomPrices
= new Dictionary<string, decimal>()
{
    { "Люкс", 8000m },
    { "Одноместный", 3000m },
    { "Двухместный", 6000m }
};
// Обработчик события изменения состояния
чекбокса
private void CheckBox_Changed(object sender,
RoutedEventArgs e)
{
    UpdateTotalCost();
}
// Метод для обновления общей стоимости
private void UpdateTotalCost()
{
    decimal totalCost = 0;
    var checkedServices = new List<string>();
    // Проверка состояния чекбоксов и
добавление выбранных услуг в список
    if (zav.IsChecked == true)
checkedServices.Add("Завтрак");
    if (tak.IsChecked == true)
checkedServices.Add("Такси");
    if (ybor.IsChecked == true)
checkedServices.Add("Уборка номера");

    // Подсчет общей стоимости выбранных
услуг
    foreach (var serviceName in checkedServices)
    {
        if
(servicePrices.TryGetValue(serviceName, out
decimal price))
        {
            totalCost += price;
        }
    }
}

```

```

// Получаем текущее значение zenroom
if (double.TryParse(zenroom.Text,
NumberStyles.Currency, CultureInfo.CurrentCulture,
out double zenroomValue))
{
    // Обновляем zen с общей стоимостью
услуг и текущей стоимостью номера
    UpdateValues((double)totalCost,
zenroomValue);
}
// Обработчик события загрузки окна
private void Window_Loaded(object sender,
RoutedEventArgs e)
{
    UpdateTotalCost();
}
// Обработчик события нажатия кнопки
возвращения на окно доступных номеров
private void Button_Click(object sender,
RoutedEventArgs e)
{
    this.Close();
}
// Метод для вычисления количества дней
между датами
private void CalculateDays()
{
    if (dateTimePicker1.SelectedDate.HasValue
&& dateTimePicker2.SelectedDate.HasValue)
    {
        DateTime checkInDate =
dateTimePicker1.SelectedDate.Value;
        DateTime checkOutDate =
dateTimePicker2.SelectedDate.Value;
        // Проверка, что дата выезда позже даты
заезда
        if (checkOutDate > checkInDate)
        {
            int days = (checkOutDate -
checkInDate).Days;
            res.Text = days.ToString();

```

```

    }
    else
    {
        res.Text = "Дата выезда должна быть
позже даты заезда";
    }
}
else
{
    res.Text = "Выберите даты заезда и
выезда";
}
}

// Обработчик события изменения выбора
типа номера
private void Tiproom_SelectionChanged(object
sender, SelectionChangedEventArgs e)
{
    if (tiproom.SelectedItem is ComboBoxItem
selectedItem)
    {
        string selectedRoomType =
selectedItem.Content.ToString();
        if
(roomPrices.TryGetValue(selectedRoomType, out
decimal roomPrice))
        {
            zenroom.Text =
roomPrice.ToString("C",
CultureInfo.CurrentCulture);
        }
        else
        {
            zenroom.Text = "0,00";
        }
        // Обновляем общую стоимость после
выбора типа номера
        TextB_TextChanged(null, null);
    }
    else
    {
        zenroom.Text = "0,00";
    }
}

```

```

    }
}

// Метод для обновления значений общей
стоимости
public void UpdateValues(double newZenValue,
double newZenroomValue)
{
    zen.Text = newZenValue.ToString("C",
CultureInfo.CurrentCulture);
    zenroom.Text =
newZenroomValue.ToString("C",
CultureInfo.CurrentCulture);

    // Обновляем общую стоимость
    TextB_TextChanged(null, null);
}

// Обработчик события изменения текста в
текстовом поле
private void TextB_TextChanged(object sender,
TextChangedEventArgs e)
{
    if (double.TryParse(zen.Text,
NumberStyles.Currency, CultureInfo.CurrentCulture,
out double zenValue) &&
double.TryParse(zenroom.Text,
NumberStyles.Currency, CultureInfo.CurrentCulture,
out double zenroomValue))
    {
        double total = zenValue + zenroomValue;
        allzen.Text = total.ToString("C",
CultureInfo.CurrentCulture);
    }
    else
    {
        allzen.Text = "0,00";
    }
}

// Обработчик события изменения выбранной
даты в первом календаре
private void
DateTimePicker1_SelectedDateChanged(object
sender, SelectionChangedEventArgs e)

```



```

    {
        CalculateDays();
    }

    // Обработчик события изменения выбранной
    даты во втором календаре
    private void
    DateTimePicker2_SelectedDateChanged(object
    sender, SelectionChangedEventArgs e)
    {
        CalculateDays();
    }

    // Обработчик события нажатия кнопки
    увеличения числа
    private void IncrementButton_Click(object
    sender, RoutedEventArgs e)
    {
        int currentNumber =
    int.Parse(NumberTextBlock.Text);
        if (currentNumber < 6)
        {
            NumberTextBlock.Text = (currentNumber
    + 1).ToString();
        }
    }

    // Обработчик события нажатия кнопки
    уменьшения числа
    private void DecrementButton_Click(object
    sender, RoutedEventArgs e)
    {
        int currentNumber =
    int.Parse(NumberTextBlock.Text);
        if (currentNumber > 1)
        {
            NumberTextBlock.Text = (currentNumber
    - 1).ToString();
        }
    }

    // Обработчик события изменения текста в
    поле ввода номера телефона
    private void Number_TextChanged(object
    sender, TextChangedEventArgs e)
    {

```

```

        string input = number.Text.Trim(); //
    Получаем текст и удаляем лишние пробелы

    // Проверяем номер телефона
    if (!ValidatePhoneNumber(input))
    {
        validationMessage.Text = "Номер
    телефона должен быть в формате +7(XXX)XXX-
    XX-XX";
        validationMessage.Visibility =
    Visibility.Visible; // Показываем сообщение об
    ошибке

        number.BorderBrush = Brushes.Red; //
    Устанавливаем красную рамку для индикации
    ошибки
    }
    else
    {
        validationMessage.Visibility =
    Visibility.Collapsed; // Скрываем сообщение об
    ошибке

        number.BorderBrush = Brushes.Green; //
    Устанавливаем зеленую рамку для индикации
    корректного ввода
    }

    // Метод для валидации номера телефона
    private bool ValidatePhoneNumber(string
    phoneNumber)
    {
        phoneNumber = phoneNumber.Replace(" ",
    "");

        Regex phoneRegex = new
    Regex(@"^\+7\(\d{3}\)\d{3}-\d{2}-\d{2}$");
        return phoneRegex.IsMatch(phoneNumber);
    }

    // Метод для получения ID типа номера по
    его названию
    private int? GetRoomTypeId(string
    roomTypeName)
    {

```

```

        // Предположим, у вас есть словарь или
база данных, где вы храните соответствие между
названиями и ID типов номеров

        Dictionary<string, int> roomTypeMapping =
new Dictionary<string, int>
        {
            { "Люкс", 1 },
            { "Одноместный", 2 },
            { "Двухместный", 3 }
        };

        // Проверяем, есть ли в словаре
соответствие для переданного названия типа
номера
        if
(roomTypeMapping.TryGetValue(roomTypeName,
out int roomTypeId))
        {
            return roomTypeId; // Возвращаем ID
типа номера
        }
        return null; // Если не найден, возвращаем
null
    }

    // Метод для валидации данных гостя
    private bool ValidateGuest(Guest guest)
    {
        return
!string.IsNullOrEmpty(guest.ФИО) &&

!string.IsNullOrEmpty(guest.Телефон) &&

!string.IsNullOrEmpty(guest.Серия_и_Номер_
Паспорта);
    }

    // Метод для получения доступного номера
по типу
    private Room
GetAvailableRoom(GalaxyEntities db, string
roomType)
    {

```

```

        return db.Room.FirstOrDefault(r =>
r.Категория_номера == roomType &&
r.Доступность == "Свободен");
    }

    // Метод добавления дополнительных услуг в
бронирование
    private void
AddAdditionalService(GalaxyEntities db, int
bookingId, string serviceName)
    {
        var service =
db.AdditionalService.FirstOrDefault(s =>
s.Название == serviceName);
        if (service != null)
        {
            var agreement = new Agreement {
Бронирование = bookingId, Список_услуг =
service.ID_AdditionalService };
            db.Agreement.Add(agreement);
        }
    }

    // Метод для установки номера комнаты
    public void SetRoomNumber(string
roomNumber)
    {
        num.Text = roomNumber;
    }

    // Метод для установки типа номера
    public void SetRoomType(string roomType)
    {
        foreach (ComboBoxItem item in
tiproom.Items)
        {
            if (item.Content.ToString() == roomType)
            {
                tiproom.SelectedItem = item;
                break;
            }
        }
    }

    // Обработчик события нажатия кнопки
"Бронировать"

```

```

private void Broni_Click_1(object sender,
RoutedEventArgs e)
{
    try
    {
        using (var db = new GalaxyEntities())
        {
            // Добавление гостя
            var contact = new Guest();
            contact.ФИО = fio.Text;
            contact.Телефон = number.Text.Trim();
            contact.Серия_и_Номер_Паспорта =
ser.Text;

            // Валидация данных гостя
            if (!ValidateGuest(contact))
            {
                MessageBox.Show("Пожалуйста,
заполните все поля для гостя.");
                return;
            }
            // Валидация номера телефона
            if
(!ValidatePhoneNumber(contact.Телефон))
            {
                MessageBox.Show("Пожалуйста,
введите корректный номер телефона.");
                return;
            }
            db.Guest.Add(contact);
            db.SaveChanges();

            // Проверка типа номера
            if (tiproom.SelectedItem is
ComboBoxItem selectedItem)
            {
                string selectedRoomType =
selectedItem.Content.ToString();
                var room = GetAvailableRoom(db,
selectedRoomType);

                if (room == null)

```

```

{
    MessageBox.Show("Нет
доступных комнат данного типа.");
    return;
}
// Создание бронирования
var booking = new Booking();
booking.Дата_заезда =
dateTimePicker1.SelectedDate.Value;
booking.Дата_выезда =
dateTimePicker2.SelectedDate.Value;

booking.Количество_дней_проживания =
(dateTimePicker2.SelectedDate.Value -
dateTimePicker1.SelectedDate.Value).Days.ToString
();

booking.Гость = contact.ID_Guest;
booking.Номер = room.ID_room;
booking.Количество_человек =
NumberTextBlock.Text;

if
(!dateTimePicker1.SelectedDate.HasValue ||
!dateTimePicker2.SelectedDate.HasValue)
{
    MessageBox.Show("Пожалуйста,
выберите даты заезда и отъезда.");
    return;
}
// Обновление доступности номера
room.Доступность = "Занят";
db.SaveChanges();

db.Booking.Add(booking);
db.SaveChanges();
// Получаем ID только что
созданного бронирования
int bookingId = booking.ID_Booking;

// Получаем ID типа номера
int? roomId =
GetRoomTypeId(selectedRoomType);
if (roomId == null)

```

```

    {
        MessageBox.Show("Не удалось
получить ID типа номера.");
        return;
    }

    // Создание договора
    var dog = new Agreement
    {
        Бронирование = bookingId,
        Дата_заключения =
DateTime.Now,
        Список_услуг = roomTypeId,
        Общая_стоимость =
decimal.Parse(allzen.Text, NumberStyles.Currency,
CultureInfo.CurrentCulture)
    };
    // Добавление дополнительных
услуг
    List<string> additionalServices = new
List<string>();
    if (zav.IsChecked == true)
    {
        additionalServices.Add("Завтрак");
    }
    if (tak.IsChecked == true)
    {
        additionalServices.Add("Такси");
    }
    if (ybor.IsChecked == true)
    {
        additionalServices.Add("Уборка
номера");
    }
    // Сохранение дополнительных
услуг в базе данных
    foreach (var service in
additionalServices)
    {
        AddAdditionalService(db,
bookingId, service);
    }
}

// Получение дополнительных услуг
для текущего бронирования
List<string> additionalServicesList =
GetAdditionalServicesForBooking(bookingId);

ExportAgreementToPdf(dog, contact,
booking);
db.Agreement.Add(dog);
db.SaveChanges();

MessageBox.Show($"Бронирование
прошло успешно!");
    }
    else
    {
        MessageBox.Show("Пожалуйста,
выберите тип номера.");
    }
}
}
}
catch (Exception ex)
{
    MessageBox.Show($"Произошла ошибка:
{ex.Message}");
}
}
private List<string>
GetAdditionalServicesForBooking(int bookingId)
{
    using (var db = new GalaxyEntities())
    {
        return db.AdditionalService
            .Where(service =>
service.ID_AdditionalService == bookingId)
            .Select(service => service.Название)
            .ToList();
    }
}

private void ExportAgreementToPdf(Agreement
agreement, Guest guest, Booking booking)
{

```

```

        string pdfPath =
Path.Combine(Environment.GetFolderPath(Environ
ment.SpecialFolder.Desktop), "Договор.pdf");
        string fontPath =
"C:\\Users\\vip66\\OneDrive\\Рабочий стол\\Comic
Sans MS.ttf"; // Укажите путь к вашему TTF-
шрифту

        try
        {
            // Создание документа
            using (Document document = new
Document())
            {
                PdfWriter writer =
PdfWriter.GetInstance(document, new
FileStream(pdfPath, FileMode.Create));
                document.Open();

                // Создание шрифта с поддержкой
кириллицы
                BaseFont baseFont =
BaseFont.CreateFont(fontPath,
BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
                Font font = new Font(baseFont, 10);

                // Проверка данных
                if (guest == null || agreement == null)
                {
                    MessageBox.Show("Данные для
экспорта отсутствуют.");
                    return;
                }
                // Заголовок
                document.Add(new
Paragraph($"Договор бронирования - №
{booking.Номер}", font));
                document.Add(new Paragraph($"ФИО
Заказчика: {guest.ФИО}", font));
                document.Add(new
Paragraph($"Телефон: {guest.Телефон}", font));

```

```

                document.Add(new Paragraph($"Серия
и Номер Паспорта:
{guest.Серия_и_Номер_Паспорта}", font));
                document.Add(new Paragraph($"Дата
заключения договора:
{DateTime.Now.ToString("dd.MM.yyyy")}", font));
                document.Add(new Paragraph($"Дата
заезда:
{dateTimePicker1.SelectedDate.Value.ToString("dd.
MM.yyyy")}", font));
                document.Add(new Paragraph($"Дата
выезда:
{dateTimePicker1.SelectedDate.Value.ToString("dd.
MM.yyyy")}", font));
                document.Add(new Paragraph($"Общая
стоимость: {agreement.Общая_стоимость:С}",
font));
                document.Add(new
Paragraph($"Количество
дней: {booking.Количество_дней_проживания}",
font));
                document.Add(new
Paragraph($"Количество
человек: {booking.Количество_человек}", font));
                // document.Add(new
Paragraph("Дополнительные услуги:", font));

                // Добавление списка дополнительных
услуг
                var services =
GetAdditionalServicesForBooking(booking.ID_Book
ing);
                foreach (var service in services)
                {
                    document.Add(new Paragraph($"-
{service}", font));
                }
                document.Add(new Paragraph("Статус
бронирования: Подтверждено", font));
                document.Add(new Paragraph("г.
Слободской", font));

```

```
document.Add(new
Paragraph($" {guest.ФИО} именуемый(ая) в
дальнейшем \"Заказчик\", с одной стороны, и
Индивидуальный предприниматель Поглазова
Валерия Владимировна, именуемый в
дальнейшем \"Исполнитель\", с другой стороны, в
дальнейшем именуемые «Стороны», заключили
настоящий договор (далее Договор) о
нижеследующем:", font));
```

```
// Общие положения
document.Add(new Paragraph("ОБЩИЕ
ПОЛОЖЕНИЯ", font));
```

```
document.Add(new Paragraph("1.1.
Заказывая услуги через Исполнителя, Заказчик
соглашается с условиями Договора публичной
оферты (далее - Договор), изложенными ниже.
Исполнитель оказывает Заказчику услуги по
предоставлению услуг бронирования объекта
недвижимости для временного размещения (далее
Объект).", font));
```

```
document.Add(new Paragraph("1.2.
Настоящий Договор, а также приложения,
относящиеся к нему, являются публичной
офертой.", font));
```

```
document.Add(new Paragraph("1.3.
Полным и безоговорочным принятием условий
Договора считается осуществление Заказчиком
платежа в счет оплаты услуг по бронированию.",
font));
```

```
// Предмет договора
document.Add(new
Paragraph("ПРЕДМЕТ ДОГОВОРА", font));
document.Add(new Paragraph("2.1. В
рамках настоящего Договора Исполнитель за
плату обязуется оказать услуги и совершать
определенные действия по бронированию объекта
недвижимости для временного размещения
(далее-Объект), соответствующего
установленным настоящим Договором
```

```
требованиям, в Заказчик обязуется оплатить эти
услуги.", font));
```

```
document.Add(new Paragraph("2.2.
Исполнитель подтверждает бронирование, после
оплаты Заказчиком 50% от общей суммы
бронирования, в размере и в сроки, указанные в
разделе 5 настоящего Договора.", font));
```

```
// Обязанности сторон
document.Add(new
Paragraph("ОБЯЗАННОСТИ СТОРОН", font));
document.Add(new Paragraph("3.1.
Исполнитель обязан:", font));
document.Add(new Paragraph("3.1.1.
Осуществлять для Заказчика подбор Объектов и
уведомлять его о наиболее подходящих из них;",
font));
```

```
document.Add(new Paragraph("3.1.2.
Предоставлять всю необходимую Заказчику
информацию о предлагаемых Объектах;", font));
document.Add(new Paragraph("3.1.3.
Осуществлять деловые контакты, вести
переговоры с потенциальными Арендодателями в
интересах Заказчика;", font));
```

```
document.Add(new Paragraph("3.1.4.
После внесения Заказчиком суммы бронирования,
в размере и сроки, указанные в разделе 5,
предоставить Заказчику Акт об оказании услуг по
Договору.", font));
```

```
document.Add(new Paragraph("3.1.5.
По желанию Заказчика согласовывать просмотр
Объекта с Принимающей стороной.", font));
```

```
// Заккрытие документа
document.Close();
}
```

```
MessageBox.Show($"Договор успешно
экспортирован в PDF: {pdfPath}");
OpenPdfFile(pdfPath);
}
catch (Exception ex)
```

```

    {
        MessageBox.Show($"Произошла ошибка
при создании PDF: {ex.Message}");
    }
}

// Метод для открытия PDF-файла
private void OpenPdfFile(string pdfPath)
{
    // Проверка, существует ли файл
    if (File.Exists(pdfPath))
    {
        try
        {
            // Открытие PDF-файла с помощью
            стандартного приложения
            System.Diagnostics.Process.Start(new
            ProcessStartInfo
            {
                FileName = pdfPath,
                UseShellExecute = true //
Используем оболочку для открытия файла
            });
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Не удалось
открыть PDF-файл: {ex.Message}");
        }
    }
    else
    {
        MessageBox.Show("PDF-файл не найден.
Убедитесь, что он был создан.");
    }
}
}

```

## Файл Reservation.xaml

```
<Window x:Class="HotelGalaxy.Reservation"
```

```

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:HotelGalaxy"
    mc:Ignorable="d"
    Title="Бронирование номера" Height="650"
    Width="900" ResizeMode="NoResize" >
    <Grid>
        <Grid.Background>
            <ImageBrush ImageSource="/fon.jpg"
            Opacity="0.7"/>
        </Grid.Background>
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="58*" />
            <ColumnDefinition Width="235*" />
            <ColumnDefinition Width="284*" />
            <ColumnDefinition Width="295*" />
            <ColumnDefinition Width="29*" />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="56*" />
            <RowDefinition Height="45*" />
            <RowDefinition Height="184*" />
            <RowDefinition Height="40*" />
            <RowDefinition Height="245*" />
            <RowDefinition Height="64*" />
        </Grid.RowDefinitions>
        <Label Content="Период и тип номера"
        Foreground="White" FontWeight="Bold"
        FontFamily="Comic Sans MS" FontSize="14"
        HorizontalContentAlignment="Center"
        Margin="56,1,0,182" Grid.Column="2"
        Background="#FF4F3762" Grid.RowSpan="2"
        Grid.Row="1"/>
    </Grid>

```

```

        <Border CornerRadius="10"
BorderThickness="1" BorderBrush="LightGray"
Margin="17,2,5,0" Background="#FFD7DEFF"
Grid.ColumnSpan="3" Grid.Row="2"
Grid.Column="1" >
    <StackPanel Grid.Row="0"
Grid.ColumnSpan="3">
        <StackPanel Orientation="Horizontal"
Margin="10,10,10,10" Width="723">
            <Label Content="Заезд:"
Foreground="#FF3A4CFF" FontStyle="Italic"
FontWeight="Bold" FontFamily="Comic Sans MS"
Margin="0,0,10,0" Width="53"/>
            <DatePicker
x:Name="dateTimePicker1"
SelectedDateChanged="DateTimePicker1_SelectedD
ateChanged" Margin="0,0,10,0" Height="25" />
            <Label Content="Выезд:"
Foreground="#FF3A4CFF" FontWeight="Bold"
FontStyle="Italic" FontFamily="Comic Sans MS"
Margin="20,0,10,0"/>
            <DatePicker
x:Name="dateTimePicker2"
SelectedDateChanged="DateTimePicker2_SelectedD
ateChanged" Height="27"/>
            <Label Content="Количество
&#xD;и&#xA;дней:" Foreground="#FF5B3BC5"
Margin="0,0,10,0" Height="41"/>
            <TextBox x:Name="res" Width="276"
Background="{x:Null}" Height="27"
BorderBrush="{x:Null}" />
        </StackPanel>
        <StackPanel Orientation="Horizontal"
Margin="10,10,10,10" Width="672">
            <Label Content="Тип номера:"
Foreground="#FF5B3BC5" Margin="0,0,10,0"/>
            <ComboBox x:Name="tiproom"
Margin="0,0,10,0" Width="157"
BorderBrush="White" Background="White" >
                <ComboBoxItem Content="Люкс"/>

```

```

                <ComboBoxItem
Content="Одноместный"/>
            </ComboBox>
            <Label Content="Номер:"
Foreground="#FF3A4CFF" FontStyle="Italic"
FontWeight="Bold" FontFamily="Comic Sans MS"
Margin="10,0,10,0" Width="53"/>
            <TextBox x:Name="num"
Width="180"/>
        </StackPanel>
        <StackPanel Orientation="Horizontal"
Height="Auto" Margin="110,0" Width="309">
            <Label Content="Количество человек:"
Foreground="#FF3A4CFF" FontWeight="Bold"
FontFamily="Comic Sans MS" Width="139" />
            <StackPanel Orientation="Horizontal">
                <Button Content="-"
Click="DecrementButton_Click"
Background="#FFA288B7" Width="32"
FontWeight="Bold" FontFamily="Comic Sans MS"
FontSize="18" Foreground="White" Height="31"
BorderBrush="White" />
                <TextBlock
x:Name="NumberTextBlock" Text="1"
FontSize="24" Margin="10,0,10,0"
VerticalAlignment="Center" Width="16"
FontFamily="Comic Sans MS" />
                <Button Content="+"
Click="IncrementButton_Click"
Background="#FFA288B7" Height="32"
Width="32" BorderBrush="White"
Foreground="White" FontFamily="Comic Sans MS"
FontSize="18" />
            </StackPanel>
        </StackPanel>
    </Border>

```



```

        <Button Content="Назад"
HorizontalAlignment="Left" Margin="19,12,0,0"
VerticalAlignment="Top" Width="78"
Click="Button_Click" BorderBrush="#FFA026A0"
Background="White" Height="30"
FontFamily="Comic Sans MS" Grid.Row="5"
Grid.ColumnSpan="2">
        <Button.Resources>
            <Style TargetType="{x:Type Border}">
                <Setter Property="CornerRadius"
Value="15" />
            </Style>
        </Button.Resources>
    </Button>

    <Button Content="Забронировать"
HorizontalAlignment="Left" Margin="69,12,0,0"
VerticalAlignment="Top" Width="202"
BorderBrush="#FFA026A0" Background="White"
Height="30" FontFamily="Comic Sans MS"
Grid.Row="5" Grid.Column="2"
Click="Broni_Click_1">
        <Button.Resources>
            <Style TargetType="{x:Type Border}">
                <Setter Property="CornerRadius"
Value="15" />
            </Style>
        </Button.Resources>
    </Button>

    <Label Content="Бронирование номера"
VerticalAlignment="Top" FontWeight="Bold"
FontFamily="Comic Sans MS"
Foreground="#FF511956" FontSize="28"
Grid.ColumnSpan="3" Background="{x:Null}"
HorizontalContentAlignment="Center" Height="51"
Margin="0,1,141,0"/>
    <Label Content="Контактные данные"
Foreground="White" FontWeight="Bold"
FontFamily="Comic Sans MS" FontSize="14"
HorizontalContentAlignment="Center"
Margin="19,17,34,209" Grid.Row="3"

```

```

Background="#FF4F3762" Grid.RowSpan="2"
Grid.ColumnSpan="2"/>
        <Border CornerRadius="10"
Background="#FFD7DEFF" Margin="19,25,106,10"
Grid.ColumnSpan="3" Grid.Row="4">
            <StackPanel>
                <StackPanel Orientation="Horizontal"
Margin="0,10,0,10" Width="375">
                    <Label Content="ФИО" Width="52"
Foreground="#FF3A4CFF" FontWeight="Bold"
FontStyle="Italic"/>
                    <TextBox x:Name="fio"
TextWrapping="Wrap" Text="" Width="322"
Height="34"/>
                </StackPanel>
                <StackPanel Orientation="Vertical"
Margin="0,10,0,10" Width="416">
                    <StackPanel Orientation="Horizontal">
                        <Label Content="Телефон:"
Width="82" Foreground="#FF3A4CFF"
FontWeight="Bold" FontStyle="Italic"
VerticalAlignment="Center"/>
                        <TextBox x:Name="number"
TextWrapping="Wrap" Width="310" Height="34"
TextChanged="Number_TextChanged"
Margin="5,0,0,0"/>
                    </StackPanel>
                    <TextBlock
x:Name="validationMessage" Foreground="Red"
Margin="0,5,0,0" Height="25" />
                </StackPanel>
                <StackPanel Orientation="Horizontal"
Margin="0,0,0,10" Width="416" Height="48">
                    <Label Content="Серия и
номер&#xD;&#xA; паспорта" Width="116"
Foreground="#FF3A4CFF" FontWeight="Bold"
FontStyle="Italic"/>
                    <TextBox x:Name="ser"
TextWrapping="Wrap" Text="" Width="278"
Height="34"/>
                </StackPanel>
            </StackPanel>

```

```

</Border>
<Label Content="Дополнительные услуги"
Foreground="White" FontWeight="Bold"
FontFamily="Comic Sans MS" FontSize="14"
HorizontalAlignment="Center"
Grid.Column="2" Margin="192,17,132,202"
Grid.Row="3" Background="#FF4F3762"
Grid.ColumnSpan="2" Grid.RowSpan="2"/>
<Border CornerRadius="10"
Background="#FFD7DEFF" Grid.Column="2"
Margin="192,25,132,10" Grid.Row="4"
Grid.ColumnSpan="2">
<StackPanel>
<StackPanel Orientation="Horizontal"
Margin="0,10,0,10" Width="145" Height="21">
<CheckBox x:Name="zav"
Content="Завтрак" Height="31"
Foreground="#FF5B3BC5"
Checked="CheckBox_Changed"/>
</StackPanel>
<StackPanel Orientation="Horizontal"
Margin="0,10,0,10" Width="146" Height="24">
<CheckBox x:Name="tak"
Content="Такси" Height="29" Width="148"
Foreground="#FF5B3BC5"
Checked="CheckBox_Changed"/>
</StackPanel>
<StackPanel Orientation="Horizontal"
Margin="0,10,0,10" Width="144" Height="30">
<CheckBox x:Name="ybor"
Content="Уборка номера"
Foreground="#FF5B3BC5"
Checked="CheckBox_Changed"/>
</StackPanel>
<StackPanel Orientation="Horizontal"
Margin="20,0,110,0" Width="198" Height="39">

```

```

<Label x:Name="zen1" Content="Цена"
Foreground="#FFB200FF" FontWeight="Bold"
FontFamily="Comic Sans MS"/>
<TextBox x:Name="zen"
Background="{x:Null}" Height="29"
IsReadOnly="True" BorderBrush="{x:Null}"
Grid.Column="2" Grid.Row="4" Width="157" />
</StackPanel>
</StackPanel>
</Border>
<Label Grid.Column="3"
Content="Стоимость&#xD;&#xA; номера"
HorizontalAlignment="Left" Margin="191,28,0,0"
Grid.Row="4" VerticalAlignment="Top"
Height="45" Width="85" FontFamily="Comic Sans
MS" Foreground="White" FontSize="14"/>
<Label Grid.Column="3"
Content="Общая&#xA;стоимость"
HorizontalAlignment="Left" Margin="191,123,0,0"
Grid.Row="4" VerticalAlignment="Top"
Height="52" Width="92" FontFamily="Comic Sans
MS" Foreground="White" FontSize="14"/>
<TextBox x:Name="zenroom"
IsReadOnly="True" Background="{x:Null}"
BorderBrush="{x:Null}" Grid.Column="3"
Margin="191,78,10,111" Grid.Row="4"
Grid.ColumnSpan="2" FontSize="16" />
<TextBox x:Name="allzen" IsReadOnly="True"
Background="{x:Null}" BorderBrush="{x:Null}"
Grid.Column="3" Margin="191,180,10,10"
Grid.Row="4" Grid.ColumnSpan="2" FontSize="16"
/>
</Grid>
</Window>

```

## Приложение 2

### Компакт-диск с материалами проекта

На диске располагается:

- Установщик программы
- Проект программы
- Файл курсового проекта в формате MS Word
- Файл с презентацией курсового проекта