МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ФГБО ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«МОСКОВСКИЙ ЭНЕРГЕТИЧЕСКИЙ ИНСТИТУТ

(ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ)»

ИЖЕНЕРНО-ЭКОНОМИЧЕСКИЙ ИНСТИТУТ

Кафедра «Прикладная и бизнес-информатика»

**ОТЧЕТ ПО КУРСОВОМУ ПРОЕКТУ**

по дисциплине «Управление разработкой информационных систем»

Тема: «Разработка Информационной Системы экскурсионного бюро»

Студент группы Тарасов В.И. ИЭс-166-15

(Ф.И.О.)

Руководитель к.т.н., Петров С.А. \_\_\_\_\_\_\_

(уч. степень, звание, Ф.И.О.)

|  |  |  |
| --- | --- | --- |
| Сдана на проверку |  |  |
| Возвращена на доработку |  |  |
| Допущена к защите |  |  |
| Оценка |  |  |
|  |

Москва 2017

Оглавление

[1.Описание предметной области 3](#_Toc482055076)

[2. Обоснование выбора объекта автоматизации 5](#_Toc482055077)

[2.1. Назначение, цели создания информационной системы туристического агентства. 5](#_Toc482055078)

[2.2. Общее описание 5](#_Toc482055079)

[2.3. Требования к системе. 5](#_Toc482055080)

[2.4. Требования к функциональным характеристикам. 6](#_Toc482055081)

[2.5. Функциональное назначение системы. 6](#_Toc482055082)

[2.6. Не функциональное назначение системы. 7](#_Toc482055083)

[2.7. Диаграмма вариантов использования. 7](#_Toc482055084)

[3. Описание выбранных для реализации технологий 9](#_Toc482055085)

[4. Проектирование информационной системы 13](#_Toc482055086)

[4.1. Диаграмма базы данных экскурсионного бюро. 14](#_Toc482055087)

[4.2. Диаграмма последовательности. 16](#_Toc482055088)

[5. Реализация информационной системы 18](#_Toc482055089)

[6. Обзор решения 23](#_Toc482055090)

[Заключение 26](#_Toc482055091)

[Литература 27](#_Toc482055092)

## 1.Описание предметной области

В качестве предметной области для информационной системы (ИС) была выбрана деятельность экскурсионного бюро – выбор экскурсии, покупка билета на экскурсию.

Турист выбирает экскурсию, выбирает дату провидения экскурсии. Каждая сделка имеет свою дату, туриста, сотрудника, экскурсию.

Основные задачи информационных систем:

* **Интерпретация данных.** Под интерпретацией понимается процесс определения смысла данных, результаты которого должны быть согласованными и корректными. Обычно предусматривается многовариантный анализ данных.
* **Диагностика.** Под диагностикой понимается процесс соотношения объекта с некоторым классом объектов и/или обнаружение неисправности в некоторой системе. Неисправность - это отклонение от нормы. Такая трактовка позволяет с единых теоретических позиций рассматривать и неисправность оборудования в технических системах, и заболевания живых организмов, и всевозможные природные аномалии.
* **Мониторинг.** Основная задача мониторинга - непрерывная интерпретация данных в реальном времени и сигнализация о выходе тех или иных параметров за допустимые пределы.
* **Проектирование.** Проектирование состоит в подготовке спецификаций на создание «объектов» с заранее определёнными свойствами. Под спецификацией понимается весь набор необходимых документов - чертёж, пояснительная записка и т.д. Основные проблемы здесь - получение чёткого структурного описания знаний об объекте и проблема «следа».
* **Прогнозирование.** Прогнозирование позволяет предсказывать последствия некоторых событий или явлений на основании анализа имеющихся данных. Прогнозирующие системы логически выводят вероятные следствия из заданных ситуаций.
* **Планирование.** Под планированием понимается нахождение планов действий, относящихся к объектам, способным выполнять некоторые функции. В таких ЭС используются модели поведения реальных объектов с тем, чтобы логически вывести последствия планируемой деятельности.

* **Обучение.** Под обучением понимается использование компьютера для обучения какой-то дисциплине или предмету. Системы обучения диагностируют ошибки при изучении какой-либо дисциплины с помощью ЭВМ и подсказывают правильные решения.
* **Управление.** Под управлением понимается функция организованной системы, поддерживающая определенный режим деятельности. Такого рода ЭС осуществляют управление поведением сложных систем в соответствии с заданными спецификациями.
* **Поддержка принятия решений.** Поддержка принятия решения - это совокупность процедур, обеспечивающая лицо, принимающее решения, необходимой информацией и рекомендациями, облегчающие процесс принятия решения. Эти ЭС помогают специалистам выбрать и/или сформировать нужную альтернативу среди множества выборов при принятии ответственных решений.

Информационная система экскурсионного бюро должна обеспечить:

* Регистрацию, изменение, удаление полной информации об экскурсиях, которая включает в себя название экскурсии, цену.
* Регистрацию, изменение, удаление персональных данных туристов, включающая в себя фамилию, имя, отчество, адрес, телефон.
* Регистрацию, изменение, удаление персональных данных сотрудников, включающая в себя фамилию, имя, отчество, адрес, телефон, зарплату, должность.
* Регистрацию, изменение, удаление полной информации об транспорте, включающая в себя название.
* Составление и регистрация туристического маршрута.
* Регистрацию, изменение, удаление всех проведенных сделок по покупке билета на экскурсию.
* Просмотра даты и времени.
* Вход в приложения для сотрудника, используя логин и пароль.

## 2. Обоснование выбора объекта автоматизации

## 2.1. Назначение, цели создания информационной системы туристического агентства.

Назначение ИС:  
Информационная система предназначена для использования в небольшом экскурсионном бюро. Эта информационная система должна сделать работу сотрудника экскурсионного бюро удобной и эффективной.

Цели создания ИС:  
Целью является разработка программного обеспечения для автоматизации деятельности сотрудников экскурсионного бюро. Она автоматизирует оформление туристических маршрутов на экскурсию и регистрацию новых туристов , ведет учет платежей за экскурсии .

Характеристика объектов автоматизации:  
В данном курсовом проекте в качестве исследуемой организации рассматривается экскурсионное бюро, которое предоставляет экскурсии туристам.

Туристическое агентство оказывает следующие услуги:  
1. предоставление экскурсий,  
2. оказание услуг.

## 2.2. Общее описание

Информационная система представляет собой хранилище данных, которое обеспечивает сохранность, упорядоченность, поиск и полный доступ к сведениям экскурсионного бюро.

## 2.3. Требования к системе.

Основное требование к системе – дружественный интерфейс пользователя. Кроме того, следует учесть, что система ориентирована на конечного пользователя, который не должен знать структуру программного кода, внутренние представление данных, средства программирования, которые использовались при разработке системы.  
 Кроме того, работа в приложении должна быть построена наиболее эффективно, то есть должны быть реализованы самые необходимые пользователю функции. Пользователь должен иметь доступ ко всем данным в БД, а также иметь возможность их добавления, корректировки, удаления.   
С информационной системой «Экскурсионном бюро» должен работать квалифицированный сотрудник, который имеет опыт работы с программными средствами, используемыми в данной работе, и может модернизировать ее.

## 2.4. Требования к функциональным характеристикам.

Система должна обеспечить возможность выполнения перечисленных ниже свойств:

* **Гибкость** – настраивается на любую структуру, для широкого круга деятельности;
* **Безопасность** – ограничение общего доступа к заранее определенным ресурсам БД автосалона;
* **Целостность и доступность** – документы в базе данных букмекерской конторы должны быть представлены в общепринятых в организации форматах, защищенных от изменений;
* **Возможность подключения независимых модулей** для импорта данных из внешних источников в текущую базу данных.

## 2.5. Функциональное назначение системы.

Программное изделие должно выполнять следующие функции:

* Регистрацию, изменение, удаление полной информации об экскурсиях, которая включает в себя название экскурсии, цену.
* Регистрацию, изменение, удаление персональных данных туристов, включающая в себя фамилию, имя, отчество, адрес, телефон.
* Регистрацию, изменение, удаление персональных данных сотрудников, включающая в себя фамилию, имя, отчество, адрес, телефон, зарплату, должность.
* Регистрацию, изменение, удаление полной информации об транспорте, включающая в себя название.
* Составление и регистрация туристического маршрута.
* Регистрацию, изменение, удаление всех проведенных сделок по покупке билета на экскурсию.
* Просмотра даты и времени.
* Вход в приложения для сотрудника, используя логин и пароль.

## 2.6. Не функциональное назначение системы.

* ИС должна быть написана на C#.
* ИС должна работать на ОС Windows.

## 2.7. Диаграмма вариантов использования.

Субъект (actor) — это некто или нечто (человек, машина и т.д.) взаимодействующее с системой. Субъект взаимодействует с прецедентом, ожидая получить некий полезный результат.

Типичным графическим изображением субъекта является «штриховой человечек». В общем случае субъект может быть показан в виде прямоугольного символа класса. Подобно обычному классу субъект может обладать атрибутами и операциями (связанными с событиями, сообщения о которых он отправляет и получает). На рисунке 1 показаны три субъекта: Турист, Экскурсионное бюро, Менеджер.

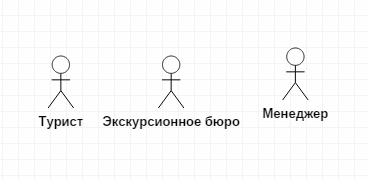


Рисунок 1 - Субъекты системы «Экскурсионное бюро»

Прецеденты удовлетворяют функциональные требования за счет предоставления субъекту полезного результата. При этом не имеет значения, в какой последовательности решает бизнес-аналитик свои задачи: сначала обозначает субъектов, а затем прецеденты, или наоборот. Прецедент наглядно показывает варианты использования системы. На рисунке 2 показаны прецеденты моделируемой системы.

 Рисунок 2 - Прецеденты системы «Экскурсионное бюро»

Диаграмма вариантов использования – это наглядное графическое представление субъектов и прецедентов и их взаимодействий в системе вместе с любыми дополнительными определениями и спецификациями. Она представляет собой не просто некую схему, а является полностью документированной моделью предполагаемого поведения системы. Диаграмма прецедентов для моделируемой системы представлена на рисунке 3.

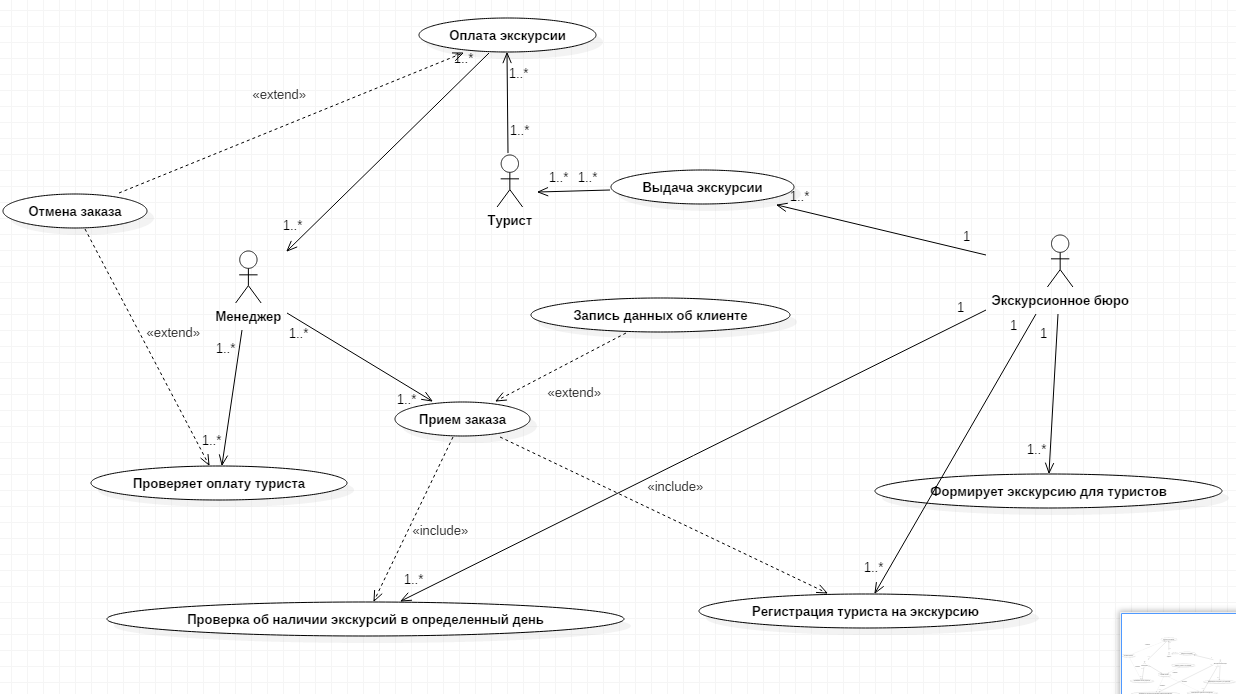


Рисунок 3 - Диаграмма вариантов использования для системы «Экскурсионное бюро»

## 3. Описание выбранных для реализации технологий

В качестве технологий для разработки системы выступают:

* .NET Framework;
* Microsoft SQL Server 2008.

Весь код программы написан в среде Microsoft Visual Studio 2015 на языке программирования C#.

1) Microsoft Visual Studio 2015 — это набор инструментов для создания программного обеспечения: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов и сбора данных телеметрии по использованию. Эти инструменты предназначены для максимально эффективной совместной работы; все они доступны в интегрированной среде разработки (IDE) Visual Studio.

Visual Studio можно использовать для создания различных типов приложений, от простых приложений для магазина и игр для мобильных клиентов до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Можно создавать:

1. приложения и игры, которые выполняются не только на платформе Windows, но и на Android и iOS;
2. веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJS и других популярных платформ;
3. приложения для самых разных платформ и устройств, включая, но не ограничиваясь: Office, Sharepoint, Hololens, Kinect и "Интернета вещей";
4. игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic. Visual Studio хорошо работает и интегрируется со сторонними приложениями, например Unity и Apache Cordova.

C# (произносится си шарп) – объектно-ориентированный язык программирования. Разработанный в 1998 – 2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .Net Framework и впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

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

Переняв многое от своих предшественников – языков C++, Pascal, Модула, Smalltalk и, в особенности, Java – C#, опираясь на практику использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, в C# в отличии от C++, не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов).

C# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов C#, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#, как и всем другим .NET – ориентированным языкам, многие возможности, которых лишены «классические» языки программирования.

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

Считается, что платформа .NET Framework явилась ответом компании Microsoft на набравшую к тому времени большую популярность платформу Java компании Sun Microsystems. Хотя, .NET является патентованной технологией корпорации Microsoft и официально рассчитана на работу под ОС семейств Microsoft Windows, существуют независимые проекты, позволяющие запускать программы .NET на некоторых других ОС.

Основной идеей при разработке .NET Framework являлось обеспечение свободы разработчиков за счет предоставления ему возможности создавать приложения различных типов, способные выполняться на различных типах устройств и в различных средах. Вторым принципом стала ориентация на системы, работающие под управлением семейства ОС Microsoft Windows.

Программа для .NET Framework, написанная на любом поддерживаемом языке программирования, сначала переводится компилятором в единый для .NET промежуточный байт-код CIL (Common Intermediate Language). Затем код либо исполняется виртуальной машиной CLR, либо транслируется утилитой NGen.exe в исполняемый код для конкретного целевого процессора. Использование виртуальной машины предпочтительно, так как избавляет разработчиков от необходимости заботиться об особенностях аппаратной части. В случае использования виртуальной машины CLR встроенный в нее JIT-компилятор «на лету» (just in time) преобразует промежуточный байт-код в машинные коды нужного процессора. Современная технология динамической компиляции позволяет достигнуть высокого уровня быстродействия. Виртуальная машина CLR также заботиться о базовой безопасности, управлении памятью и системе исключений, избавляя разработчика от части работы.

Объектные классы .NET, доступные для всех поддерживаемых языков программирования, содержатся в библиотеке Framework Class Library (FCL). В FCL входят классы Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation и другие.

2) Microsoft SQL Server 2008 — доступ к информации всегда и везде.

Решение Microsoft SQL Server 2008 представляет собой высокопроизводительную платформу обработки и анализа данных для бизнеса любого размера, которая отвечает всем современным требованиям по работе с данными любых типов и поддержке удобной и быстрой разработки приложений. Продукт реализует концепцию комплексной обработки данных и позволяет преобразовывать ценную информацию в знания и решения на основе анализа собранных данных. В базах данных SQL Server 2008 можно хранить любую структурированную, полуструктурированную или неструктурированную информацию, такую например, как изображения и мультимедиа из самых разнородных источников данных. Продукт предлагает большой набор интегрированных служб, расширяющих возможности обработки, который позволяет составлять запросы, выполнять поиск, проводить синхронизацию, формировать отчеты и анализировать данные. SQL Server 2008 обеспечивает обращение к данным из любого приложения, разработанного с применением технологий Microsoft .NET и Visual Studio, в том числе — с помощью Microsoft BizTalk Server — из приложений, построенных на базе сервис-ориентированной архитектуры (SOA) и корпоративных бизнес-приложений других разработчиков. Сотрудники, отвечающие за сбор и анализ информации, могут работать с данными, пользуясь привычными приложениями, например программами Microsoft Office 2007.

Использование в бизнесе любого размера

На базе Microsoft SQL Server 2008 могут быть построены решения для компаний малого, среднего и крупного бизнеса. SQL Server 2008 выпускается в двух основных редакциях Standard и Enterprise. На основе последней создана также редакция для разработчиков Developer Edition, лицензия на которую позволяет разрабатывать и тестировать системы и приложения.

Enterprise-версия системы SQL Server 2008 представляет собой комплексную платформу, которая позволяет работать даже с самыми требовательными корпоративными OLTP-системами и хранилищами данных. Она обладает значительной масштабируемостью, возможностью создавать громадные хранилища данных, продвинутыми средствами анализа и усиленной безопасностью, что позволяет использовать ее как основу для критически важных бизнес-приложений. Эта редакция позволяет консолидировать серверы и выполнять большое число OLTP-операций и крупные отчеты.

Редакцию Microsoft SQL Server 2008 Enterprise характеризуют:

• Высокий уровень доступности. Непрерывность бизнес-процессов обеспечивается благодаря защите данных от дорогостоящих человеческих ошибок и максимальному уменьшению сроков аварийного восстановления.

• Производительность и масштабируемость. Инфраструктура на основе SQL Server 2008 Enterprise позволяет справиться с любыми пиковыми нагрузками.

• Безопасность. Вопросы конфиденциальности, а также соответствия нормативным требованиям решаются с помощью встроенных средств защиты от несанкционированного доступа.

• Управляемость. Автоматические диагностика, калибровка и настройка инфраструктуры позволяют управлять огромными объемами данных, значительно сократив издержки на управление и обслуживание.

• Бизнес-аналитика. SQL Server 2008 Enterprise помогает легко собрать и проанализировать большие объемы данных из хранилищ или киосков.

Редакция Microsoft SQL Server 2008 Standard — это законченная платформа для управления данными и выполнения операций бизнес-аналитики. Она демонстрирует лучшие в своем классе показатели простоты использования и степени управляемости ключевых бизнес-приложений.

SQL Server 2008 Standard предлагает следующие преимущества:

• Простота использования. Легкость в эксплуатации позволяет снизить операционные издержки и расходы на разработку решений на основе платформы.

• Управляемость. Интуитивно понятные средства управления и автоматизированного администрирования помогают эффективно управлять бизнес-приложениями.

• Создание отчетов и аналитика. С помощью встроенных технологий анализа и создания отчетности можно легко и быстро получать практическую и значимую информацию и принимать обоснованные решения.

## 4. Проектирование информационной системы

Проектирование информационных систем всегда начинается с определения цели проекта. Основная задача любого успешного проекта заключается в том, чтобы на момент запуска системы и в течение всего времени ее эксплуатации можно было обеспечить:

* требуемую функциональность системы и степень адаптации к изменяющимся условиям ее функционирования;
* требуемую пропускную способность системы;
* безотказную работу системы в требуемом режиме, иными словами - готовность и доступность системы для обработки запросов пользователей;
* простоту эксплуатации и поддержки системы;
* необходимую безопасность.

Производительность является главным фактором, определяющим эффективность системы. Хорошее проектное решение служит основой высокопроизводительной системы.

Проектирование информационных систем охватывает три основные области:

* проектирование объектов данных, которые будут реализованы в базе данных;
* проектирование программ, экранных форм, отчетов, которые будут обеспечивать выполнение запросов к данным;
* учет конкретной среды или технологии, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры (файл-сервер или клиент-сервер), параллельной обработки, распределенной обработки данных и т.п.

В реальных условиях проектирование - это поиск способа, который удовлетворяет требованиям функциональности системы средствами имеющихся технологий с учетом заданных ограничений.

К любому проекту предъявляется ряд абсолютных требований, например максимальное время разработки проекта, максимальные денежные вложения в проект и т.д. Одна из сложностей проектирования состоит в том, что оно не является такой структурированной задачей, как анализ требований к проекту или реализация того или иного проектного решения.

Считается, что сложную систему невозможно описать в принципе. Это, в частности, касается систем управления предприятием. Одним из основных аргументов является изменение условий функционирования системы, например директивное изменение тех или иных потоков информации новым руководством. Еще один аргумент - объемы технического задания, которые для крупного проекта могут составлять сотни страниц, в то время как технический проект может содержать ошибки. Возникает вопрос: а может, лучше вообще не проводить обследования и не делать никакого технического проекта, а писать систему «с чистого листа» в надежде на то, что произойдет некое чудесное совпадение желания заказчика с тем, что написали программисты, а также на то, что все это будет стабильно работать?

Жизненный цикл программного обеспечения представляет собой модель его создания и использования. Модель отражает его различные состояния, начиная с момента возникновения необходимости в данном ПО и заканчивая моментом его полного выхода из употребления у всех пользователей. Известны следующие модели жизненного цикла:

* Каскадная модель. Переход на следующий этап означает полное завершение работ на предыдущем этапе.
* Поэтапная модель с промежуточным контролем. Разработка ПО ведется итерациями с циклами обратной связи между этапами. Межэтапные корректировки позволяют уменьшить трудоемкость процесса разработки по сравнению с каскадной моделью; время жизни каждого из этапов растягивается на весь период разработки.
* Спиральная модель. Особое внимание уделяется начальным этапам разработки - выработке стратегии, анализу и проектированию, где реализуемость тех или иных технических решений проверяется и обосновывается посредством создания прототипов (макетирования). Каждый виток спирали предполагает создание некой версии продукта или какого-либо его компонента, при этом уточняются характеристики и цели проекта, определяется его качество и планируются работы следующего витка спирали.

## 4.1. Диаграмма базы данных экскурсионного бюро.

Целью построения диаграммы базы данных является получение графического представления физической структуры исследуемой предметной области.

Сущности описывают объекты, являющиеся предметом деятельности предметной области, и субъекты, осуществляющие деятельность в рамках предметной области. Свойства объектов и субъектов реального мира описываются с помощью атрибутов.

Взаимоотношения между сущностями иллюстрируются с помощью связей. Правила и ограничения взаимоотношений описываются с помощью свойств связей. Обычно связи определяют либо зависимости между сущностями, либо влияние одной сущности на другую.

На рисунки 4 представлена диаграмма базы данных информационной системы экскурсионного бюро. На ней изображено семь сущностей.

* В сущности Туристы имеются атрибуты номер туриста, фамилия, имя, отчество, адрес, телефон. Она соединена связью один ко многим с сущностью Заказы.
* В сущности Сотрудники имеются атрибуты номер сотрудника, фамилия, имя, отчество, зарплата, должность, адрес, телефон. Она соединена связью один ко многим с сущностями Заказы и Туристический маршрут.
* В сущности Заказы имеются атрибуты номер заказа, дата заказа, номер сотрудника, номер туриста, номер туристического маршрута. Она соединена связью один ко многим с сущностями Туристы, Туристический маршрут, Сотрудники и Оплата.
* В сущности оплата имеются атрибуты номер оплаты, дата оплаты, сумма, номер заказы. Она соединена связью один ко многим с сущностью Заказы.
* В сущности Туристический маршрут имеются атрибуты номер туристического маршрута, дата отправления, дата возращения, длительность по времени, номер сотрудника, номер транспорта, номер экскурсии. Она соединена связью один ко многим с сущностями Заказы, Сотрудники, Транспорт, Экскурсии.
* В сущности Транспорт имеются атрибуты номер транспорта, название. Она соединена связью один ко многим с сущностью Туристический маршрут.
* В сущности Экскурсии имеются атрибуты номер экскурсии, название, цена. Она соединена связью один ко многим с сущностью Туристический маршрут.

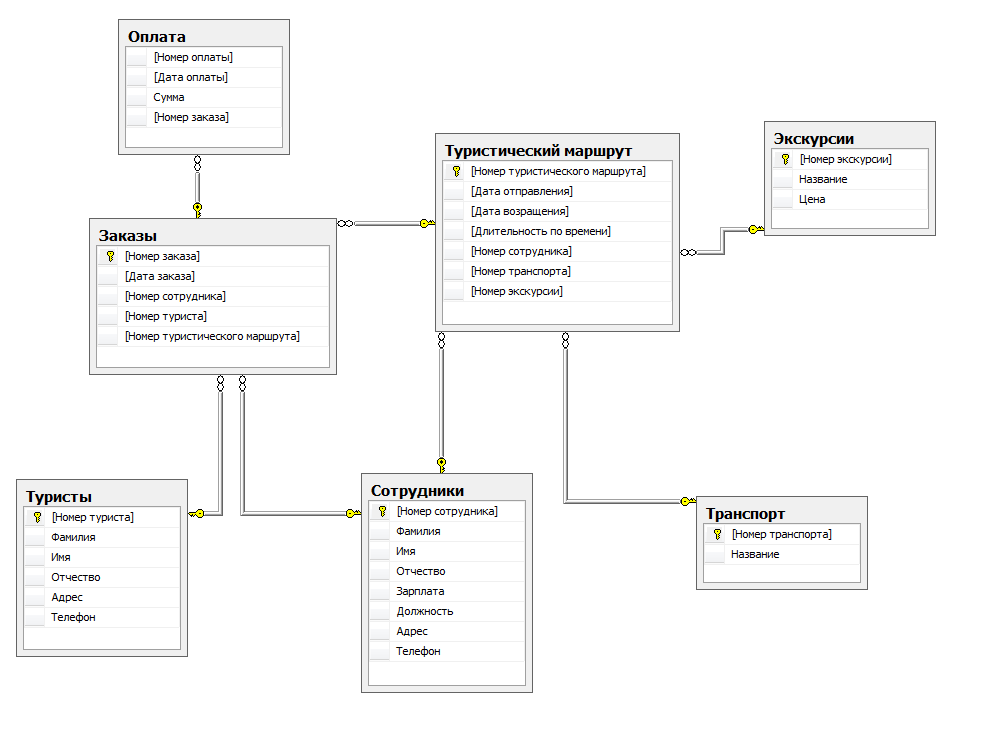


Рис. 4. Диаграмма базы данных экскурсионного бюро.

## 4.2. Диаграмма последовательности.

Диаграмма последовательности - диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актёров (действующих лиц) ИС в рамках какого-либо определённого прецедента (отправка запросов и получение ответов). Используется в языке UML.

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни» , отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

Всего таких объектов пять: турист, менеджер, экскурсовод, заказ, база. Диаграмма последовательности представлена на рисунке 5.

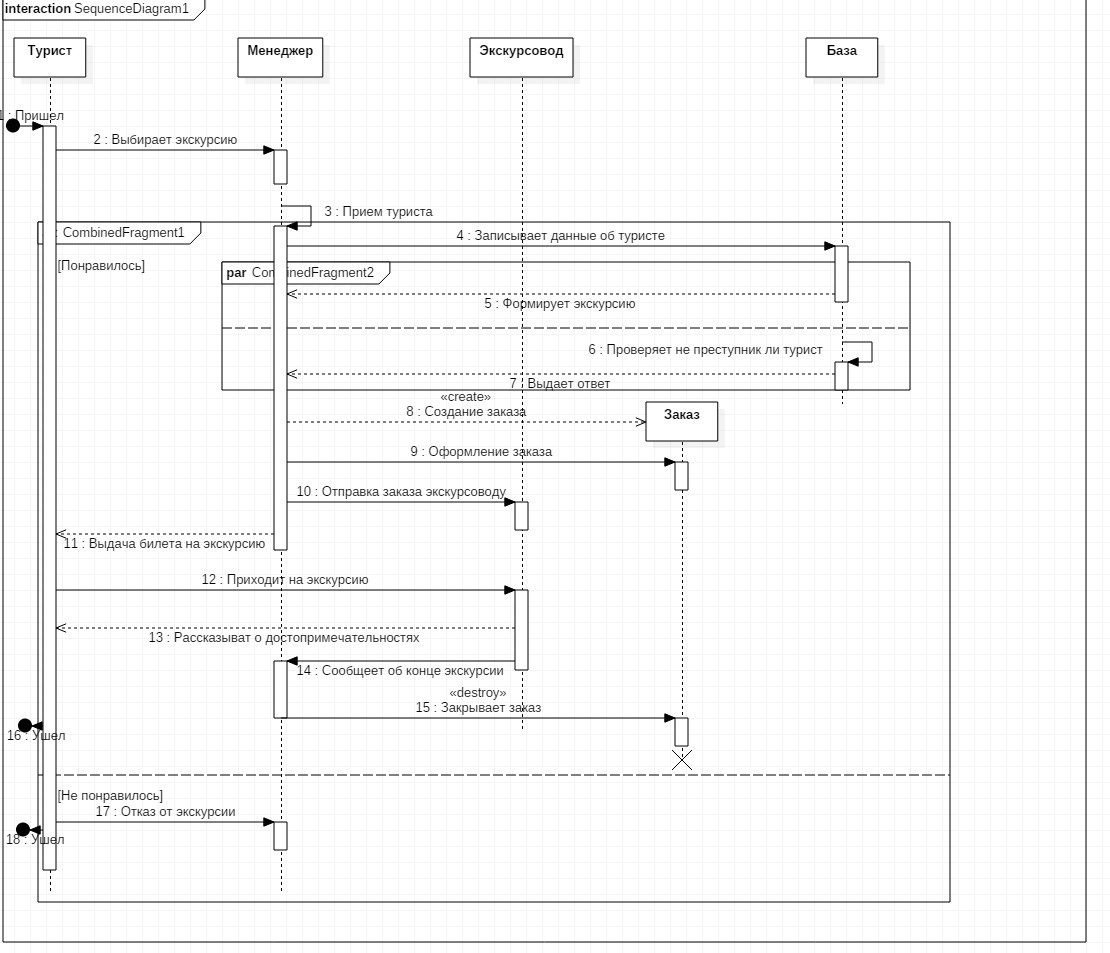


Рисунок 5 - Диаграмма последовательности для системы «Экскурсионное бюро»

## 5. Реализация информационной системы

Для взаимодействия пользователя с системой был разработан графический интерфейс.

Работа начинается с окна программы, где пользователь проходит авторизация пользователя представленного на рисунке 6.

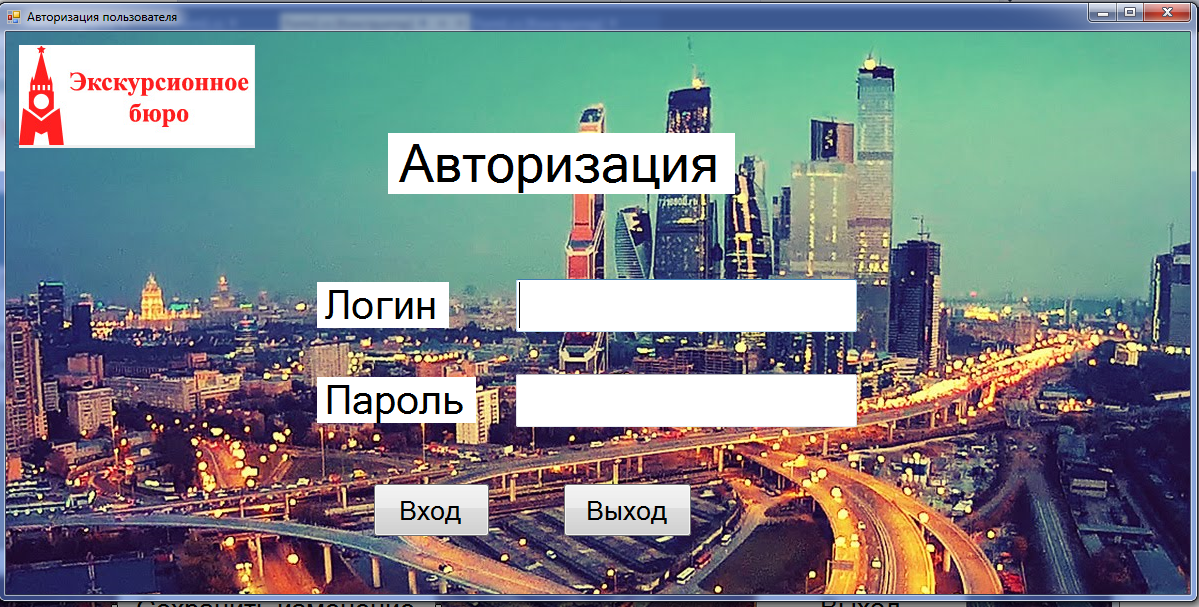


Рис. 6. Форма авторизации пользователя.

1) Код проверки логина и пароля, при помощи кнопки вход и переход в клиентское приложения. Код кнопки “Выход”.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Экскурсионное\_бюро

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text == "Vovan")

{

if (textBox2.Text == "12345")

{

Form2 f1 = new Form2();

f1.ShowDialog();

}

}

else

MessageBox.Show("Неправильный логин или пароль!");

textBox1.Clear();

textBox2.Clear();

}

private void button2\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

После нажатие кнопки вход, при указании правильного логина и пароля, его переносит в форму клиентского приложения, где сотрудник может изменять, удалять, добавлять данные и просмотреть дату и время представленного на рисунке 7.

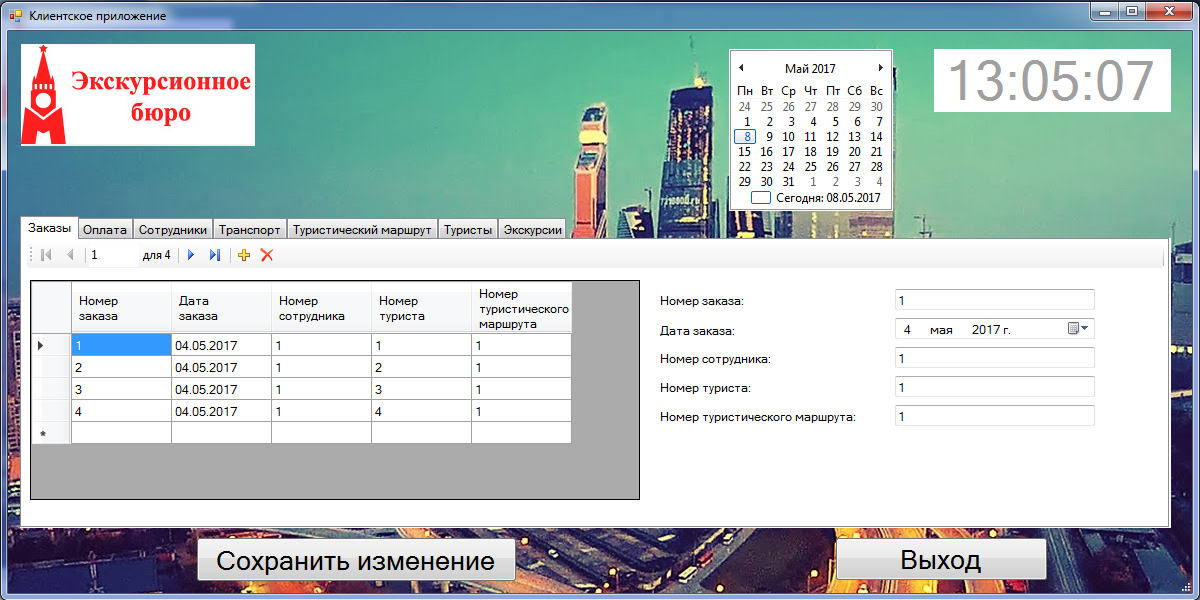


Рис. 7. Форма клиентского приложения.

2) Код добавления таблиц из базы данных Экскурсионное бюро. Код часов. Код кнопки “Сохранить изменения”, при которой сохраняются изменения во всех таблицах при удалении, добавлении и изменения строк таблиц. Код кнопки “Выход”.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Экскурсионное\_бюро

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Экскурсии". При необходимости она может быть перемещена или удалена.

this.экскурсииTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Экскурсии);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Туристы". При необходимости она может быть перемещена или удалена.

this.туристыTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Туристы);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Туристический\_маршрут". При необходимости она может быть перемещена или удалена.

this.туристический\_маршрутTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Туристический\_маршрут);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Транспорт". При необходимости она может быть перемещена или удалена.

this.транспортTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Транспорт);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Сотрудники". При необходимости она может быть перемещена или удалена.

this.сотрудникиTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Сотрудники);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Оплата". При необходимости она может быть перемещена или удалена.

this.оплатаTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Оплата);

// TODO: данная строка кода позволяет загрузить данные в таблицу "экскурсионное\_бюроDataSet.Заказы". При необходимости она может быть перемещена или удалена.

this.заказыTableAdapter.Fill(this.экскурсионное\_бюроDataSet.Заказы);

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void timer1\_Tick(object sender, EventArgs e)

{

label1.Text = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00");

}

private void button2\_Click(object sender, EventArgs e)

{

this.заказыTableAdapter.Update(this.экскурсионное\_бюроDataSet);

this.экскурсииTableAdapter.Update(this.экскурсионное\_бюроDataSet);

this.туристыTableAdapter.Update(this.экскурсионное\_бюроDataSet);

this.туристический\_маршрутTableAdapter.Update(this.экскурсионное\_бюроDataSet);

this.транспортTableAdapter.Update(this.экскурсионное\_бюроDataSet);

this.сотрудникиTableAdapter.Update(this.экскурсионное\_бюроDataSet);

}

}

}

## 6. Обзор решения

В случае неправильного заполнения данных выскакивает предупреждение "Неправильный логин или пароль!", представленного на рисунке 8.

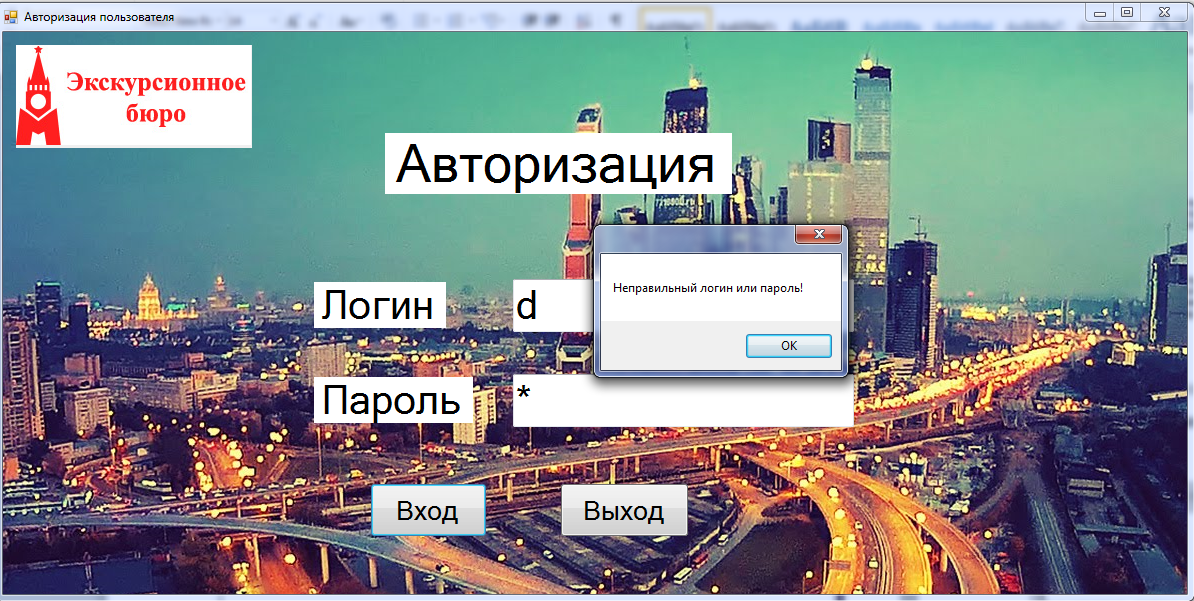


Рис. 8. Предупреждение "Неправильный логин или пароль!"

При помощи навигатора можно изменить, удалить, добавить строку в таблицы.

Ниже представлены экранные формы просмотра всех таблиц на рисунках 9-15:

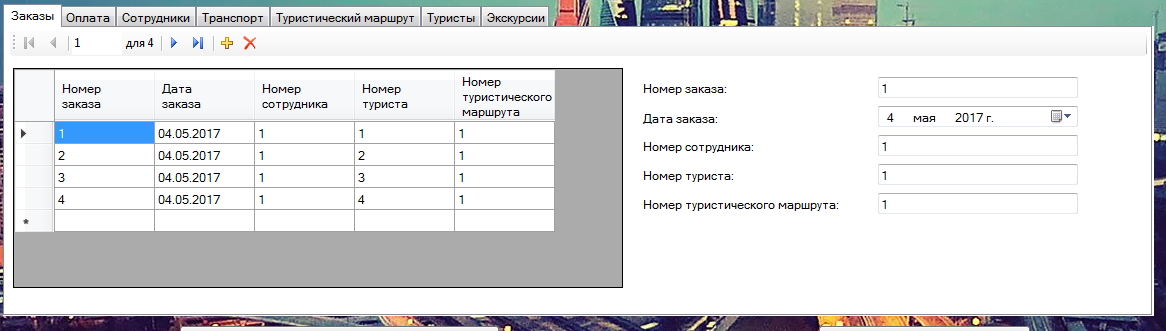


Рис. 9. Просмотр таблицы Заказы.

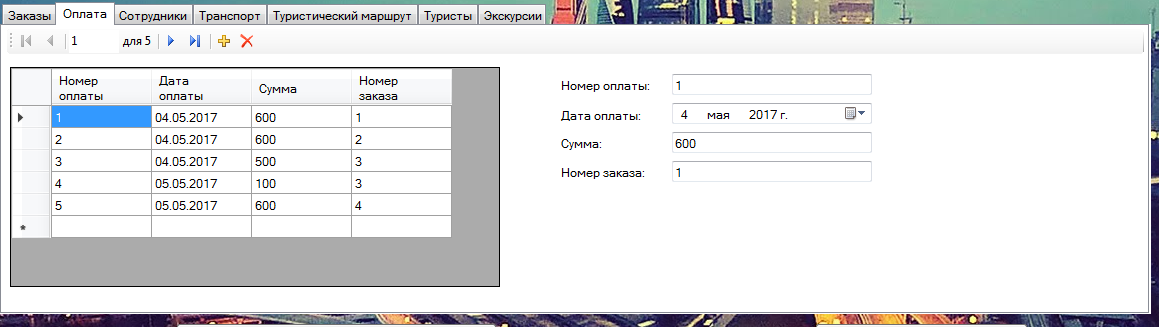


Рис. 10. Просмотр таблицы Оплата.

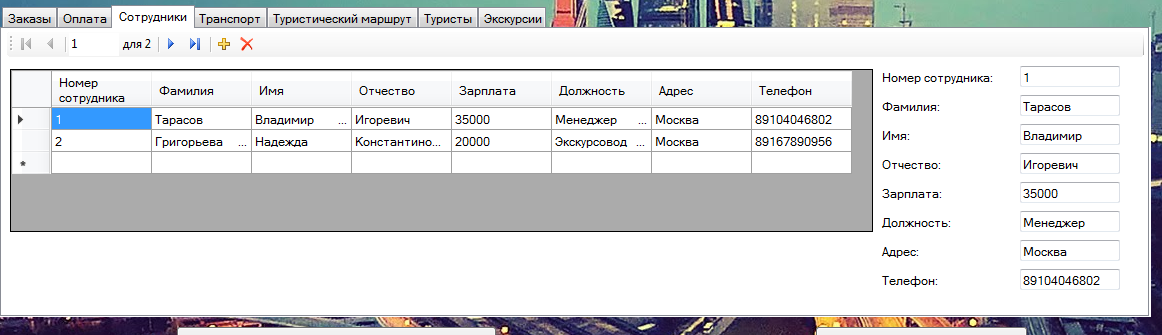


Рис. 11. Просмотр таблицы Сотрудники.

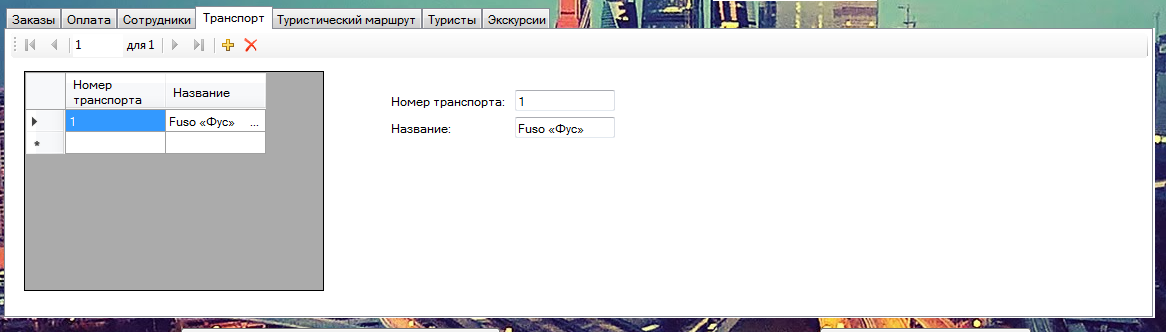


Рис. 12. Просмотр таблицы Транспорт.

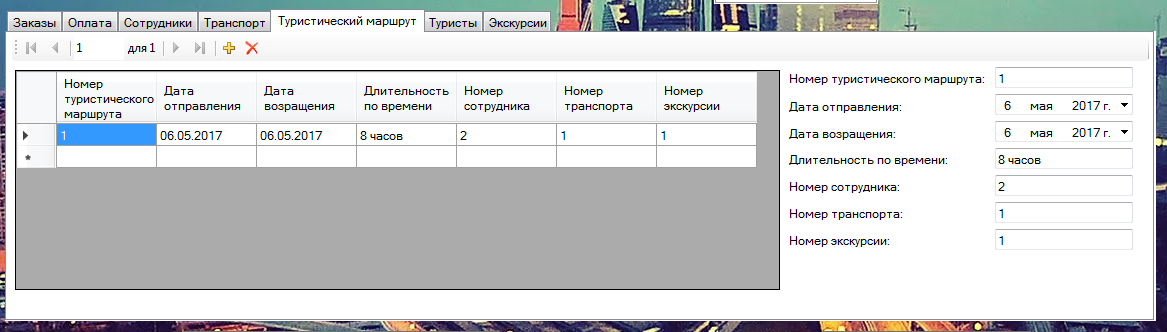


Рис.13. Просмотр таблицы Туристический маршрут.

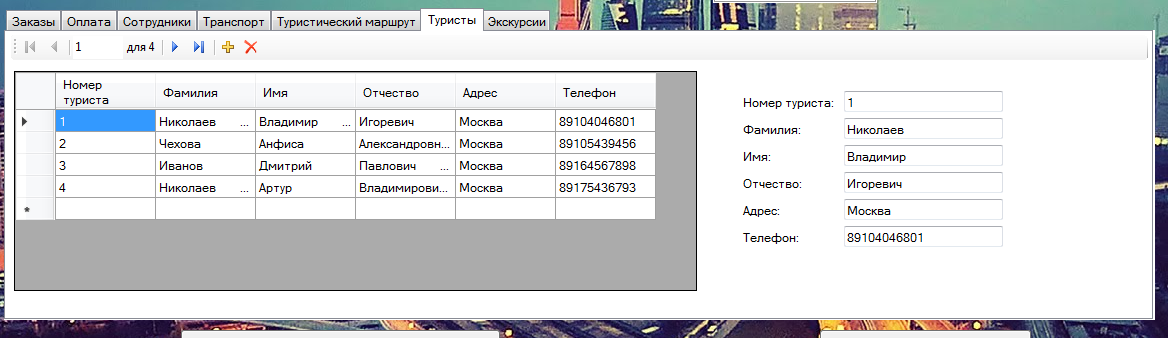


Рис.14. Просмотр таблицы Туристы.

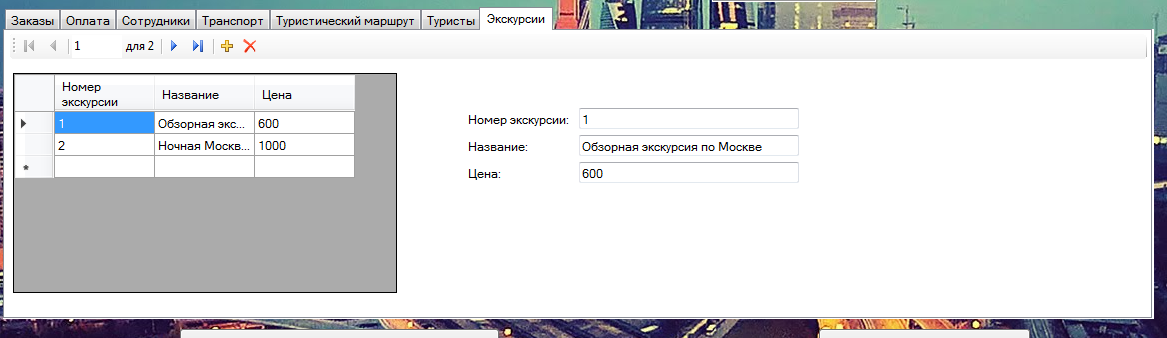


Рис.15. Просмотр таблицы Экскурсии.

## Заключение

В результате курсового проекта была спроектирована и реализована информационная система «Экскурсеонное бюро». Данная система удовлетворяет всем требованиям, предъявленным в задании, и реализует большинство необходимых сотрудникам бюро функций. Приложение выполняет следующий перечень требований:

* представление данных в удобном для пользователя виде;
* добавление, удаление и изменение данных;
* просмотр даты и времени, и т.д.

В результате выполнения курсовой работы был сделан вывод, что сегодня внедрение информационных систем может способствовать:

* получению более рациональных вариантов решения управленческих задач за счет внедрения математических методов и интеллектуальных систем и т.д.
* освобождению работников от рутинной работы за счет ее автоматизации;
* обеспечению достоверности информации;
* замене бумажных носителей, данных на магнитные и оптические, что приводит к более рациональной организации переработки информации на компьютере и снижению объемов бумажных документов;
* уменьшению затрат на производство продуктов и услуг.

## Литература

1. [Фаулер М.](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D1%83%D0%BB%D0%B5%D1%80,_%D0%9C%D0%B0%D1%80%D1%82%D0%B8%D0%BD), Бек К., Брант Д., Робертс Д., Апдайк У. Рефакторинг: улучшение существующего кода = Refactoring: Improving the Design of Existing Code (2000). — Спб: Символ-Плюс, 2009. — 432 с. [ISBN 5-93286-045-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5932860456).
2. Буч Г., Рамбо Д., Якобсон И. Язык UML. Руководство пользователя. 2-е изд.: Пер. с англ. Мухин Н. — М.: ДМК Пресс, 2006. — 496 с.: ил. ISBN 5-94074-334-X.
3. Стандарт ISO/IEC 2382:2015.
4. <http://analyst.by/diagrams/logicheskaya-model-predmetnoy-oblasti>
5. <http://bibliofond.ru/download_list.aspx?id=787590>
6. <http://obmendoc.ru/files/users/andrey/72/view/253102-253103>
7. <https://www.google.ru/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&ved=0ahUKEwjoiP-4lJbRAhVGWSwKHa1AApwQFggtMAM&url=http%3A%2F%2Fedulancer.ru%2Fupload.php%3Ffileid%3D6422&usg=AFQjCNH9N9V9Bitn3fch0UNovlXPBnhyww&cad=rjt>
8. Microsoft Developer Network: <https://msdn.microsoft.com/> .

9.Загрузка СУБД MS SQL Express: <https://www.microsoft.com/en-us/cloud-platform/sql-server> .