**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КОМИ**

**Государственное профессиональное образовательное учреждение**

**«Воркутинский арктический горно-политехнический колледж»**

**КУРСОВОЙ ПРОЕКТ**

По дисциплине МДК.05.02 Разработка кода информационных систем

**Разработка информационной системы «Регистрация ДТП в ГИБДД»**

Выполнил студент гр. ИСП-21 /\_\_\_\_\_\_\_\_\_\_\_\_/Абилов Эльмар Тайяр оглы/

(подпись) (Ф.И.О.)

**ОЦЕНКА:\_\_\_\_\_\_\_\_\_\_\_\_\_**

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОВЕРИЛ**

Научный руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_/ Егоров Данил Павлович/

(подпись) (Ф.И.О.)

Воркута

2024

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ**

**ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ**

**1.1 Платформа .NET**

**1.2 Язык программирования С#**

**1.3 Windows Presentation Foundation (WPF)**

**1.4 СУБД SQL server**

**1.5 Microsoft SQL Server Management Studio**

**1.6 Entity Framework**

**ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ**

**2.1 Разработка диаграммы ERD**

**2.2 Разработка базы данных**

**ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ**

**3.1 Разработка прототипа информационной системы**

**3.2 Программирование информационной системы**

**3.2.1 Разработка модуля «Авторизация»**

**3.2.2 Разработка модуля …..**

**3.2.3 Разработка модуля …..**

**3.2.4 Разработка модуля …..**

**3.2.5 Разработка модуля «Отчет»**

**ЗАКЛЮЧЕНИЕ**

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**1. Наименование работы**: Автоматизированная информационная система «Регистрация ДТП в ГИБДД».

**2. Назначение разработки**

Автоматизированная информационная система «Регистрация ДТП в ГИБДД» данная система предназначена для обеспечения оперативного реагирования на происшествия, сбора и анализа информации о ДТП, упрощения процесса оформления документов, а также повышения прозрачности и эффективности работы ГИБДД относительно регистрации и учета дорожно-транспортных происшествий.

Пользователями данной системы выступают сотрудник ГИБДД, участник ДТП, администратор системы. Сотрудник ГИБДД занимается регистрацией данных о дорожно-транспортном происшествии (ДТП), включая место, время, участников и обстоятельства происшествия. Участник ДТП заполняет анкеты с информацией о дорожно-транспортном происшествии (ДТП), включая данные о участниках, транспортных средствах и обстоятельствах происшествия. Предоставление необходимых документов, таких как водительское удостоверение, страховой полис, технический паспорт транспортного средства. Администратор системы в АИС "Регистрация ДТП в ГИБДД" обычно отвечают за поддержку и обслуживание системы, включая управление пользователями, обновление программного обеспечения, обеспечение безопасности данных, решение проблемных ситуаций и техническую поддержку пользователей.

**3. Требования к программе**

**3.1. Требования к функциональным характеристикам**

Автоматизированная информационная система «Регистрация ДТП в ГИБДД» должна обеспечивать выполнение функций:

• Регистрация и хранение информации о ДТП

• Возможность сопоставления данных с путевыми листами, страховыми полисами и другими документами, связанными с ДТП.

• Защита информации и обеспечение конфиденциальности данных, связанных с личностью участников и свидетелей ДТП.

• Предоставление удобного интерфейса для ввода, редактирования и поиска информации о ДТП с использованием различных параметров, таких как дата, место, участники и т.д.

В программе необходимо предусмотреть возможность корректировки настроек системы; резервное сохранение данных; возможность изменения пароля входа в систему; наличие встроенной справочной системы; быстрый поиск необходимых документов и справочной информации и т.д.;

Разрабатываемая система будет иметь модульную структуру, доступ к модулям будет зависеть от роли пользователя. Доступ возможен только для авторизованных пользователей.

В связи с этим при запуске системы первым окном будет окно входа.

Доступ к системе имеют только сотрудники. У каждого сотрудника есть логин, по которому и разграничиваются права доступа.

Алгоритм авторизации:

* сотрудник вводит логин и пароль;
* при вводе пароля сотрудником и нажатии клавиши Enter на служебный телефон отправляется СМС с единоразовым кодом доступа;
* сотрудник вводит код и далее получает доступ к необходимому функционалу.

При открытии окна активны только поле для ввода логина сотрудника и кнопка “Отмена”.

При вводе логина сотрудника и нажатию Enter происходит проверка логина сотрудника. Если логин сотрудника есть в базе данных, то поле для ввода пароля становится активным и в нем установлен курсор. Если логин сотрудника в базе отсутствует, появляется сообщение об ошибке.

После ввода пароля по нажатию на Enter открывается модальное окно со сгенерированным кодом доступа (4 символов, латиница, верхний и нижний регистр, спецсимвол, цифра).

В течение 10 секунд после закрытия окна с кодом пользователь должен ввести код и авторизоваться (по Enter и “Вход”).

Если в течение 10 секунд код не введен, для повторной “отправки” кода необходимо нажать C:\Users\РС-1\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\F4E393EB.tmp.

При вводе неправильного пароля код не генерируется, и система сообщает пользователю о неверном пароле.

У пользователя должна быть возможность очистить все поля ввода нажатием на кнопку “Отмена”.

После успешной авторизации сотруднику должно быть выведено сообщение с названием его роли.

**Разграничение прав доступа:**

- администратор системы: добавлять, изменять, удалять;

- сотрудник ГИБДД: добавлять, изменять;

- участник ДТП: просмотра введенных записей.

**3.2. Требования к надежности**

Разрабатываемое программное обеспечение должно иметь:

■ парольную защиту при запуске программы;

■ ограничение несанкционированного доступа к данным;

■ разграничение пользовательских прав;

Предусмотреть контроль вводимой информации и блокировку некорректных действий пользователя при работе с системой.

**3.3. Требования к составу и параметрам технических средств**

Системные требования для работы программного продукта должны быть следующими: тактовая частота процессора 1600 ГГц; объем оперативной памяти 8 Гб; объем свободного дискового пространства 1 Гб; разрешение монитора 1920 х 1080; наличие устройства чтения USB-носителя.

**3.4. Требования к информационной и программной совместимости**

Программа должна работать в операционных системах Windows 10/11. Все формируемые отчеты должны иметь возможность экспортирования в редактор электронных таблиц MS Office Excel 2016/2019.

**3.5. Требования к транспортированию и хранению**

Программа поставляется на USB-носителе. Программная документация поставляется в электронном и печатном виде.

**3.6. Специальные требования**

Программное обеспечение должно иметь дружественный интерфейс, рассчитанный на пользователя средней квалификации (с точки зрения компьютерной грамотности).

Ввиду объемности проекта задачи предполагается решать поэтапно. При этом модули программного обеспечения (ПО), созданные в разное время, должны предполагать возможность наращивания системы и быть совместимы друг с другом; поэтому документация на принятое эксплуатационное ПО должна содержать полную информацию, необходимую для работы с ним программистов. Язык программирования определяется выбором исполнителя, при этом он должен обеспечивать возможность интеграции программного обеспечения с пакетом MS Office 2016/2019.

**4. Требования к программной документации**

В ходе разработки программы должны быть подготовлены: текст программы, описание программы, программа и методика испытаний, руководство пользователя, технико-экономическое обоснование очевидно, что использование программы значительно сократит время, затрачиваемое на регистрацию заказа. С использованием программы затраты времени сокращаются до 1—2 мин. Формирование оборотной ведомости в компьютере займет 3 — 4 мин. Кроме того, предполагается возможность получения отчетов за любой период времени. При ручном создании отчетов человеком могут быть допущены ошибки; правильно составленный алгоритм разрабатываемой программы ошибки исключает. Экономический эффект от внедрения автоматизированной информационной системы «Ремонт мобильных телефонов» ожидается за счет сокращения времени на выполняемые менеджерами операции, исключения ошибок при формировании отчетов.

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Этап** | **Дата начала** | **Дата окончания** |
| 1. | Выдача темы курсового проекта | 28.02.2024 | 28.02.2024 |
| 2. | Разработка технического задания | 28.02.2024 | 01.03.2024 |
| 3. | Постановка целей и задач курсового проекта | 01.03.2024 | 03.03.2024 |
| 4. | Выбор инструментария | 04.03.2024 | 07.03.2024 |
| 5. | Разработка прототипа проекта | 08.03.2024 | 20.03.2024 |
| 6. | Разработка базы данных проекта | 21.03.2024 | 04.04.2024 |
| 7. | Разработка информационной системы | 05.04.2024 | 25.04.2024 |
| 8. | Защита курсового проекта | 29.04.2024 | 29.04.2024 |

**ВВЕДЕНИЕ**

Актуальность выбранной темы в том, что обусловлена необходимостью улучшения процесса регистрации и учета ДТП для органов ГИБДД и общества в целом. С развитием технологий и возросшим объемом данных необходимо обеспечить эффективное хранение, передачу и анализ информации о дорожно-транспортных происшествиях (ДТП).

Таким образом, создание информационной системы для регистрации ДТП в ГИБДД имеет высокую актуальность с точки зрения повышения эффективности, улучшения качества данных и обеспечения безопасности на дорогах.

**Объект:** информационная система «Регистрация ДТП в ГИБДД».

**Предмет:** автоматизированная информационная система «Регистрация ДТП в ГИБДД».

**Цель работы** – разработать автоматизированную информационную систему «Регистрация ДТП в ГИБДД».

**Задачи**:

* выбрать инструментарий;
* спроектировать базу данных;
* разработать информационную систему.

**ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ**

**1.1 Платформа .NET**

.NET — это бесплатная кроссплатформенная кроссплатформенная [платформа разработчика](https://github.com/dotnet/core) с открытым кодом для создания [множества приложений](https://learn.microsoft.com/ru-ru/dotnet/core/apps). Он может запускать программы, написанные на [нескольких языках](https://learn.microsoft.com/ru-ru/dotnet/fundamentals/languages), при [этом C#](https://learn.microsoft.com/ru-ru/dotnet/csharp/) является наиболее популярным. Она использует высокопроизводительную среду выполнения, которая используется в рабочей среде многими [высокомасштабируемыми приложениями](https://devblogs.microsoft.com/dotnet/category/developer-stories/).

Платформа .NET разработана для обеспечения производительности, производительности, безопасности и надежности. Она обеспечивает автоматическое управление памятью [с помощью сборщика мусора (GC).](https://learn.microsoft.com/ru-ru/dotnet/standard/automatic-memory-management) Это типобезопасный и безопасный для памяти, из-за использования GC и строгих компиляторов языка.  Он включает в себя большой набор библиотек, имеющих широкие функциональные возможности и оптимизированный для производительности в нескольких операционных системах и архитектурах микросхем.

.NET включает следующие компоненты:

* среда выполнения — выполняет код приложения;
* библиотеки — предоставляют служебные функции, такие как [синтаксический анализ](https://learn.microsoft.com/ru-ru/dotnet/standard/serialization/system-text-json/overview) JSON;
* компилятор — компилирует исходный код C# (и других языков) в исполняемый код (среда выполнения);
* пакет SDK и другие средства — включение создания и мониторинга приложений с помощью современных рабочих процессов;
* стеки приложений, такие как ASP.NET Core и Windows Forms, которые позволяют создавать приложения.

**1.2 Язык программирования С#**

C# — современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется хорошо знакомым любому, кто работал с C, C++, Java или JavaScript. Здесь представлен обзор основных компонентов языка C# 8 и более ранних версий. Если вы хотите изучить язык с помощью интерактивных примеров, рекомендуем поработать с [вводными руководствами по C#](https://learn.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/tutorials/).

C# — объектно-ориентированный, ориентированный на компоненты язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО. В основном C# — объектно-ориентированный язык. Вы определяете типы и их поведение.

Вот лишь несколько функций языка C#, которые позволяют создавать надежные и устойчивые приложения. [Сборка мусора](https://learn.microsoft.com/ru-ru/dotnet/standard/garbage-collection/) автоматически освобождает память, занятую недостижимыми неиспользуемыми объектами. [Типы, допускающие значение null,](https://learn.microsoft.com/ru-ru/dotnet/csharp/nullable-references) обеспечивают защиту от переменных, которые не ссылаются на выделенные объекты. [Обработка исключений](https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/exceptions/) предоставляет структурированный и расширяемый подход к обнаружению ошибок и восстановлению после них. [Лямбда - выражения](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/lambda-expressions) поддерживают приемы функционального программирования. [Синтаксис LINQ](https://learn.microsoft.com/ru-ru/dotnet/csharp/linq/) создает общий шаблон для работы с данными из любого источника. Поддержка языков для [асинхронных операций](https://learn.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/async/) предоставляет синтаксис для создания распределенных систем. В C# имеется [Единая система типов](https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/types/). Все типы C#, включая типы-примитивы, такие как int и double, наследуют от одного корневого типа object. Все типы используют общий набор операций, а значения любого типа можно хранить, передавать и обрабатывать схожим образом. Более того, C# поддерживает как определяемые пользователями [ссылочные типы](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/reference-types), так и [типы значений](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/value-types). C# позволяет динамически выделять объекты и хранить упрощенные структуры в стеке. C# поддерживает универсальные методы и типы, обеспечивающие повышенную безопасность типов и производительность. C# предоставляет итераторы, которые позволяют разработчикам классов коллекций определять пользовательские варианты поведения для клиентского кода.

C# подчеркивает управление версиями, чтобы обеспечить совместимость программ и библиотек с течением времени. Вопросы управления версиями существенно повлияли на такие аспекты разработки C#, как раздельные модификаторы virtual и override, правила разрешения перегрузки методов и поддержка явного объявления членов интерфейса.

**1.3 Windows Presentation Foundation (WPF)**

Технология WPF (Windows Presentation Foundation) является часть экосистемы платформы .NET и представляет собой подсистему для построения графических интерфейсов.

Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML: вы можете создавать насыщенный графический интерфейс, используя или декларативное объявление интерфейса, или код на управляемых языках C#, VB.NET и F#, либо совмещать и то, и другое.

Первая версия - WPF 3.0 вышла вместе с .NET Framework 3.0 и операционной системой Windows Vista в 2006 году. И с тех пор платформа WPF является частью экосистемы .NET и развивается вместе с фреймворком .NET. Например, на сегодняшний день последней версией фреймворка .NET является .NET 8, и WPF полностью поддерживается этой версией фреймворка.

Преимущества WPF.

Что вам, как разработчику, предлагает WPF?

* Использование традиционных языков .NET-платформы - C#, F# и VB.NET для создания логики приложения;
* возможность декларативного определения графического интерфейса с помощью специального языка разметки XAML, основанном на xml и представляющем альтернативу программному созданию графики и элементов управления, а также возможность комбинировать XAML и C#/VB.NET;
* независимость от разрешения экрана: поскольку в WPF все элементы измеряются в независимых от устройства единицах, приложения на WPF легко масштабируются под разные экраны с разным разрешением;
* новые возможности, которых сложно было достичь в WinForms, например, создание трехмерных моделей, привязка данных, использование таких элементов, как стили, шаблоны, темы и другое;
* хорошее взаимодействие с WinForms, благодаря чему, например, в приложениях WPF можно использовать традиционные элементы управления из WinForms;
* богатые возможности по созданию различных приложений: это и мультимедиа, и двухмерная и трехмерная графика, и богатый набор встроенных элементов управления, а также возможность самим создавать новые элементы, создание анимаций, привязка данных, стили, шаблоны, темы и многое другое;
* аппаратное ускорение графики - вне зависимости от того, работаете ли вы с 2D или 3D, графикой или текстом, все компоненты приложения транслируются в объекты, понятные Direct3D, и затем визуализируются с помощью процессора на видеокарте, что повышает производительность, делает графику более плавной;
* создание приложений под множество ОС семейства Windows.

В тоже время WPF имеет определенные ограничения. Несмотря на поддержку трехмерной визуализации, для создания приложений с большим количеством трехмерных изображений, прежде всего игр, лучше использовать другие средства - DirectX или специальные фреймворки, такие как Monogame или Unity.

Кроме того, несмотря на то, что WPF работает поверх кроссплатформенной среды .NET, но в силу природы WPF и зависимости от компонентов Windows, на данный момент создавать приложения на WPF можно только под ОС Windows.

**1.4 СУБД SQL server**

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире. Данная СУБД подходит для самых различных проектов: от небольших приложений до больших высоконагруженных проектов.

SQL Server был создан компанией Microsoft. Первая версия вышла в 1987 году. А текущей версией является версия 2022, которая вышла в ноябре 2022 году и которая будет использоваться в текущем руководстве.

SQL Server долгое время был исключительно системой управления базами данных для Windows, однако начиная с версии 16 эта система доступна и на Linux.

SQL Server характеризуется такими особенностями как:

Производительность. SQL Server работает очень быстро.

Надежность и безопасность. SQL Server предоставляет шифрование данных.

Простота. С данной СУБД относительно легко работать и вести администрирование.

Центральным аспектом в MS SQL Server, как и в любой СУБД, является база данных. База данных представляет хранилище данных, организованных определенным способом. Нередко физически база данных представляет файл на жестком диске, хотя такое соответствие необязательно. Для хранения и администрирования баз данных применяются системы управления базами данных (database management system) или СУБД (DBMS). И как раз MS SQL Server является одной из такой СУБД.

Выделяются две разновидности языка SQL: PL-SQL и T-SQL. PL-SQL используется в таких СУБД как Oracle и MySQL. T-SQL (Transact-SQL) применяется в SQL Server.

**1.5 Microsoft SQL Server Management Studio**

Управление решениями служб Analysis Services с помощью SQL Server Management Studio

SSMS позволяет управлять объектами служб Analysis Services, такими как резервное копирование и обработка объектов.

Management Studio предоставляет проект скрипта служб Analysis Services, в котором вы разрабатываете и сохраняете скрипты, написанные в многомерных выражениях (многомерных выражений), расширения интеллектуального анализа данных (DMX) и XML для анализа (XMLA). Проекты сценариев служб Analysis Services используются для выполнения задач управления или повторного создания объектов, таких как база данных и кубов, в экземплярах служб Analysis Services. Например, можно разработать скрипт XMLA в проекте скрипта служб Analysis Services, который создает новые объекты непосредственно в существующем экземпляре служб Analysis Services. Проекты скриптов служб Analysis Services можно сохранить как часть решения и интегрировать с управлением исходным кодом.

SSMS позволяет использовать службу Служб Integration Services для управления пакетами и мониторинга запущенных пакетов. Вы также можете использовать Management Studio для упорядочивания пакетов в папки, запуска пакетов, импорта и экспорта пакетов, переноса пакетов служб преобразования данных (DTS) и обновления пакетов служб Integration Services.

SSMS можно использовать для включения функций Служб Reporting Services, администрирования сервера и баз данных и управления ролями и заданиями.

Вы управляете общими расписаниями с помощью папки "Общие расписания" и управляете базами данных сервера отчетов (ReportServer, ReportServerTempDB). Вы также создаете RSExecRole в master системной базе данных при перемещении базы данных сервера отчетов в новую или другую ядро СУБД.

**1.6 Entity Framework**

Entity Framework (EF) Core — это простая, кроссплатформенная и расширяемая версия популярной технологии доступа к данным Entity Framework с открытым исходным кодом.

Entity Framework Core (EF Core) представляет собой объектно-ориентированную, легковесную и расширяемую технологию от компании Microsoft для доступа к данным. EF Core является ORM-инструментом (object-relational mapping - отображения данных на реальные объекты). То есть EF Core позволяет работать базами данных, но представляет собой более высокий уровень абстракции: EF Core позволяет абстрагироваться от самой базы данных и ее таблиц и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работаем с объектами.

Entity Framework Core поддерживает множество различных систем баз данных. Таким образом, мы можем через EF Core работать с любой СУБД, если для нее имеется нужный провайдер.

По умолчанию на данный момент Microsoft предоставляет ряд встроенных провайдеров: для работы с MS SQL Server, для SQLite, для PostgreSQL. Также имеются провайдеры от сторонних поставщиков, например, для MySQL.

Также стоит отметить, что EF Core предоставляет универсальный API для работы с данными. И если, к примеру, мы решим сменить целевую СУБД, то основные изменения в проекте будут касаться прежде всего конфигурации и настройки подключения к соответствующим провайдерам. А код, который непосредственно работает с данными, получает данные, добавляет их в БД и т.д., останется прежним.

**ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ**

**2.1 Разработка диаграммы ERD**

ER Entity-Relationship Diagram — это визуальный инструмент для представления и анализа структуры данных, а также для определения отношений между различными элементами данных. ER диаграммы помогают в проектировании баз данных и определении необходимых связей между сущностями.

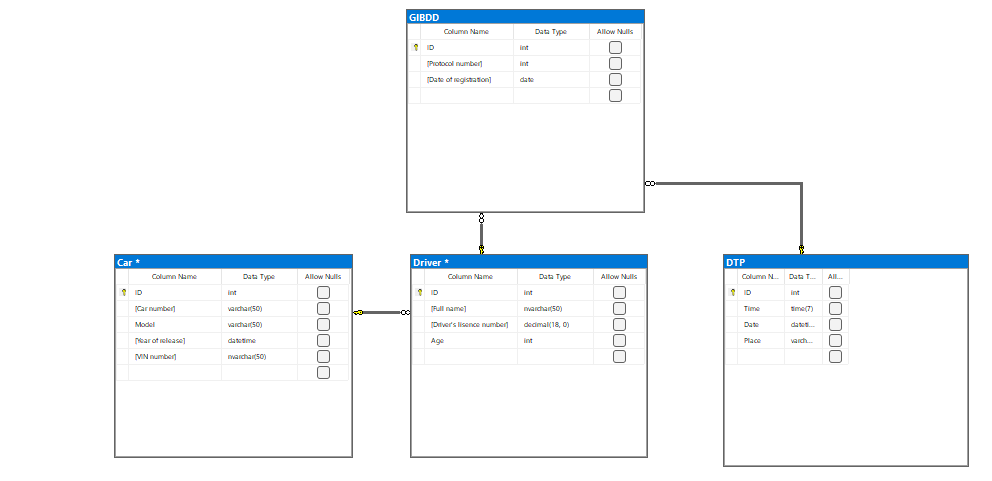
ER диаграмма представляет собой графическое изображение сущностей, атрибутов и отношений между ними. Она состоит из трех основных компонентов:

* сущности (Entities) - представляют собой основные элементы данных, которые необходимо хранить в базе данных. Они представлены прямоугольниками с названием сущности внутри;
* атрибуты (Attributes) - определяют свойства каждой сущности и представлены в виде списка внутри прямоугольника сущности. Каждый атрибут имеет уникальное имя и определенный тип данных;
* отношения (Relationships) - описывают связи между различными сущностями и представлены линиями между прямоугольниками сущностей. Обычно отношения имеют имя и степень (количество экземпляров связи).



Рис. 2.1 диаграмма ER

### 2.2 Разработка базы данных

****

## **ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ**

**3.1 Разработка прототипа информационной системы**

Проволочная диаграмма представляет собой наглядный детализированный макет пользовательского интерфейса – своеобразный рабочий чертеж, описывающий функциональные возможности и содержимое приложения. Такие проволочные шаблоны для веб-сайтов и мобильных приложений идеально подходят для создания нестрогих проектных эскизов, демонстрирующих идеи и позволяющих достичь единого мнения в команде и сформировать основу для разработки высококачественной каркасной модели.

В окне «Авторизация» у нас есть два поля: в первое пользователь вводит свой логин, во второе пароль. Ниже располагается капча, и кнопка «войти». В левой части окна находится логотип компании.



Рис. 3.1 прототип окна «Авторизация»

# **ЗАКЛЮЧЕНИЕ**