Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра электронных вычислительных средств

Дисциплина: Операционные системы и управление базами данных

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

**ГОРОДСКАЯ ДУМА**

БГУИР КП 1-58 01 05 ПЗ

|  |  |
| --- | --- |
| Студент | А.В. Юрчак |
| Руководитель | Д.С. Лихачёв |

Минск 2014

|  |  |
| --- | --- |
| **Содержание** | |
| Введение................................................................................................................ | 3 |
| 1 Анализ технического задания.......................................................................... | 4 |
| 1.1 Интерфейс............................................... ............................................... .... | 5 |
| 1.2 Требования к работе программного средства........................................... | 5 |
| 2 Проектирование структуры информационной системы................................ | 6 |
| 3 Разработка база данных.................................................................................... | 9 |
| 4 Разработка клиентской части информационной системы............................. | 12 |
| 4.1 .NET Framework............................................................................. ............. | 12 |
| 4.2 Windows Forms............................................................................................. | 13 |
| 4.3 Технология ADO.NET................................................................................. | 14 |
| 4.3.1 Реляционные базы данных. Основные понятия............................. | 15 |
| 4.3.2 Работа с базами данных.................................................................... | 16 |
| 4.3.3 Доступ к отсоединённым данным.................................................... | 17 |
| 4.3.4 Доступ к данным............................................................................... | 18 |
| 4.3.5 Объектная модель.............................................................................. | 19 |
| 4.4 Среда разработки.................................................................................... .... | 20 |
| 4.5 Результат разработки клиентской части................................................... | 21 |
| 5 Проверка работоспособности разработанной информационной системы.. | 26 |
| Заключение............................................................................................................ | 28 |
| Список использованных источников.................................................................. | 29 |
| Приложение A Диаграмма базы данных............................................................ | 30 |

**Введение**

На сегодняшний день проектирование баз данных и разработка эффективных приложений с базами данных является одной из самых популярных тем объектно-ориентированного программирования. Сейчас практически невозможно представить компанию (фирму или организацию), в которой не требуется обработка некоторого объёма информации. Информацию требуется, где-то хранить. Информация может динамически изменяться. Регулярно требуется выборка данных по определённым критериям из всего массива. При автоматизации бизнес процессов очень часто возникают задачи, которые не решают уже готовые программы и базы данных. При этом аналитическая информация показывает, что даже если использовать сложные и дорогостоящие CRM-системы управления предприятием, получить решение, удовлетворяющее руководство компании, бывает просто не возможно. Базы данных создаются специально для хранения, обработки, проведения расчётов, сортировки, выборки и представления любых массивов данных по любым критериям.

Ещё более актуальной эта тема стала с появлением новой платформы .NET Frаmewоrk, выпущенной компанией Micrоsоft, которая имеет огромный потенциал в решении вышеуказанных проблем. И это нисколько не удивительно, так как большинство деловых и офисных приложений, таких, например, как системы бухгалтерского или складского учёта, разного рода системы автоматизации финансового и хозяйственного учёта, а также Web-приложения, рассчитанные для работы в Интернете и интрасетях компаний интегрируются тем или иным способом с базами данных. Хоть .NET Frаmewоrk и не была разработана специально для этих целей, но встроенные в неё языки программирования и стандартная библиотека классов, позволяют проектировать модели баз данных и разрабатывать клиентские приложения, содержащие базы данных, в полном объёме.

В рамках данной курсовой работы будут изложены основные сведения о платформе .NET Frаmewоrk, продемонстрированы методы доступа к базам данных и системам управления базами данных, используя при этом в качестве программного инструментария продукт компании Micrоsоft «Micrоsоft Visuаl Studio Ultimate 2012» и базу данных «MySQL Server 5.6».

Мной будет разработан собственный проект приложения «Городская дума», содержащий базу данных и раскрывающий все аспекты разработки базы данных на языке C#, которые будут подробно рассмотрены в третьей главе настоящей курсовой работы.

**1 Анализ технического задания**

Основная цель данного клиентского приложения – предоставить сотрудникам городской думы необходимую информацию о её работе. В данном курсовом проекте в качестве информационного ресурса рассматривается система для полного функционирования городской думы. Программный продукт должен позволять осуществлять просмотр работников, комиссий, заседаний, производить поиск информации о конкретном работнике, комиссии, заседании, должно быть реализовано добавление, удаление и редактирование информации. В программе должны быть реализованы следующие компоненты:

1. Хранение всей информации в базе данных.

2. Возможность просмотра комиссии – её состава и председателя.

3. Возможность добавления или удаления информации. При этом для добавления необходимо будет зайти в систему в качестве администратора.

4. Возможность добавления новой и редактирование существующей информации о депутатах, заседаниях и комиссиях.

5. Обеспечение предотвращения ввода некорректной информации а также случайного удаления информации. Обеспечить данную возможность посредством дополнительного оповещения пользователя при вводе некорректных данных, при воспроизведении действий, которые могут повлечь утрату информации.

6. Возможность создания отчётов в формате .pdf.

* 1. Интерфейс

Программная оболочка должна иметь интуитивно понятный интерфейс. Пользовательский интерфейс должен быть выполнен в доступном виде и привлекать своим визуальным оформлением.

Программа целиком состоит из форм, каждая из которых отвечает за свой раздел и конкретные функции:

* форма входа в систему или регистрация пользователя;
* главная форма, содержащая таблицы с основными данными;
* форма добавления/редактирования данных сотрудников;
* форма добавления/редактирования информации о комиссиях;
* форма добавления/редактирования информации о заседаниях;
* форма просмотра детальной информации о сотрудниках;
* форма просмотра детальной информации о комиссиях;
* форма просмотра детальной информации о заседаниях.

1.2 Требования к работе программного средства

Для нашего приложения предъявляются следующие требования:

* разрабатываемое программное средство должно корректно работать;
* высокая скорость работы;
* визуально понятный интерфейс.

**2 Проектирование структуры информационной системы**

Основной задачей ИС является удовлетворение конкретных информационных потребностей в рамках конкретной предметной области. Современные ИС де-факто немыслимы без использования баз данных и СУБД, поэтому термин «информационная система» на практике сливается по смыслу с термином «система баз данных». Структура системы - это множество всех возможных отношений между подсистемами и элементами внутри системы.

В моём курсовом проекте информационной системой является информационной ресурс, предоставляющий информацию о городской думе. Целью данной информационной системы является хранение всей нужной информации в удобном виде и частичная автоматизация работы с ней.

Как уже было сказано выше, для хранения информации наша система будет использовать базу данных под управлением СУБД MySQL Server 5.6. В базе данных будет храниться вся необходимая информация для работы нашей городской думы. База данных под управлением СУБД будет являться подсистемой нашей системы. Второй подсистемой нашей системы будет являться клиентская часть. Клиентская часть нужна для удобства работы с базой и получения нужной информации в контексте нашей предметной области. Реализовываться она будет в качестве приложения Windows Forms.

Как правило компьютеры и программы, входящие в состав информационной системы, не являются равноправными. Некоторые из них владеют ресурсами (файловая система, процессор, принтер, база данных и т.д.), другие имеют возможность обращаться к этим ресурсам. Компьютер (или программу), управляющий ресурсом, называют сервером этого ресурса (файл-сервер, сервер базы данных, вычислительный сервер...). Клиент и сервер какого-либо ресурса могут находится как в рамках одной вычислительной системы, так и на различных компьютерах, связанных сетью.

Основной принцип технологии "клиент-сервер" заключается в разделении функций приложения на три группы:

* ввод и отображение данных (взаимодействие с пользователем);
* прикладные функции, характерные для данной предметной области;
* функции управления ресурсами (файловой системой, базой данных и т.д.).

Поэтому, в любом приложении выделяются следующие компоненты:

* компонент представления данных;
* прикладной компонент;
* компонент управления ресурсом.

Итак, мы выделили две подсистемы:

* СУБД, содержащая в себе базу данных;
* клиентская часть.

Система управления базой данной «Городской думы» предназначена для использования в городских, а также государственных думах.

Системой могут пользоваться все сотрудники городской думы, на рабочих станциях которых установлено данное приложение. Тем не менее, их права не равнозначны.

Для системы существует два вида ролей: администратор и пользователь. Администратор имеет полный список привилегий: от просмотра и поиска информации до добавления, изменения или удаления данных. Пользователь только имеет доступ к просмотру и поиску информации.

Исходя из вышесказанного, следует отметить, что для системы должны быть реализованы следующие функции:

1. *Добавление, редактирование, удаление информации* – операции доступны только администратору системы. Выполняя их, он получает полноценный доступ к базе данных, в которой хранится вся информация, следовательно администратор также работает с её внутренней структурой и может изменять её. Поэтому эти функции выполняет администратор в целях соблюдения политики безопасности и избегания нарушения целостности базы данных, а также потери части или всех данных.
2. *Просмотр и поиск информации* – функции доступны как обычным пользователям, так и администратору. Эти операции имеют также доступ к базе данных, но они не затрагивают и не изменяют её структуру, а просто получают информацию для просмотра, поэтому любой пользователь может просмотреть или найти интересующую его информацию: сотрудника, заседание или комиссию. Также возможен детальный просмотр выбранных данных.

Также для предотвращения потери или несанкционированного изменения данных программа контролирует следующие аспекты:

1. соблюдение типов данных при заполнении полей;
2. операции изменения;
3. операции удаления;
4. операции сохранения;
5. выход из программы.

Для того, чтобы лучше понять, как функционирует система на рисунке 2.1 представлена схема взаимодействия системы «Городская дума».

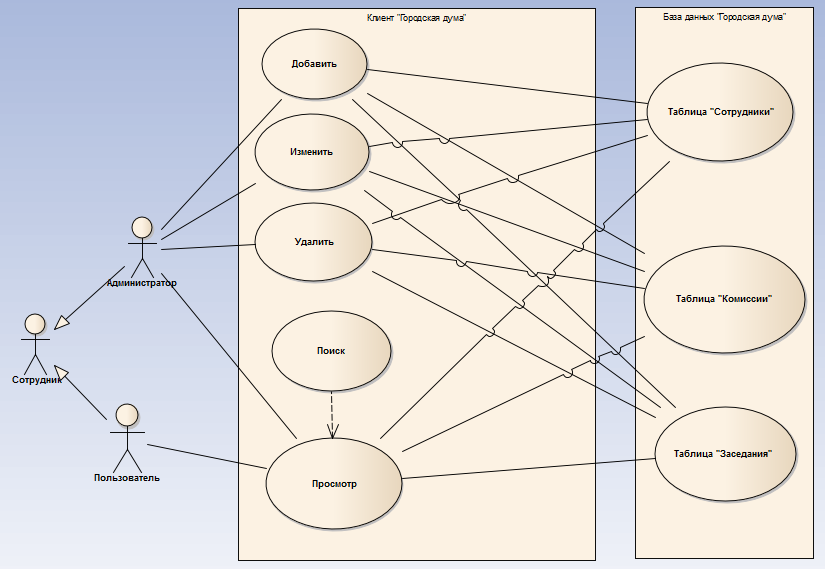


Рисунок 2.1 – Схема взаимодействия системы «Городская дума»

На *входе* информационной системы «Городская дума» пользователь обрабатывает исходную или детальную информацию о сотрудниках, комиссиях и заседаниях. На *выходе* получает информацию, как результат поиска по какому-либо критерию. В случае просмотра информации о заседании, при регистрации на него, может быть сформировано приглашение для всех сотрудников, кто приглашён на встречу в виде PDF документа, а также, после его окончания, итоговый отчёт по встрече, с целью автоматизировать работу с документацией и снизить нагрузку на сотрудников думы.

**3 Разработка базы данных**

База данных (БД) – это совокупность специальным образом организованных данных хранимых в памяти вычислительной системы отображающих состояние объектов и их взаимосвязи в рассматриваемой предметной области. [1]

СУБД – комплекс языковых и программных средств, предназначенных для создания, ведения и совместного использования БД несколькими пользователями. СУБД позволяет: создавать БД; вставлять, обновлять, удалять и извлекать информацию из БД; предоставляет контролируемый доступ к базе данных. [1]

В качестве СУБД для моего курсового проекта была выбрана MySQL Server 5.6

MySQL является решением для малых и средних приложений. Входит в состав серверов [WAMP](https://ru.wikipedia.org/wiki/WAMP), [AppServ](https://ru.wikipedia.org/w/index.php?title=AppServ&action=edit&redlink=1" \o "AppServ (страница отсутствует)), [LAMP](https://ru.wikipedia.org/wiki/LAMP) и в портативные сборки серверов [Денвер](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BD%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)), [XAMPP](https://ru.wikipedia.org/wiki/XAMPP), [VertrigoServ](https://ru.wikipedia.org/wiki/VertrigoServ" \o "VertrigoServ). Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы. [1]

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа [MyISAM](https://ru.wikipedia.org/wiki/MyISAM" \o "MyISAM), поддерживающие полнотекстовый поиск, так и таблицы [InnoDB](https://ru.wikipedia.org/wiki/InnoDB" \o "InnoDB), поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

В результате была создана база данных «Городская дума» со следующими таблицами:

* *employees;*
* *comissions;*
* *meetings;*
* *roles;*
* *states;*
* *profile;*
* *contacts;*
* *meetings\_employees;*
* *comissions\_employees;*
* *users.*

Для каждой таблицы согласно предметной области были созданы поля, в которых хранится какая-то информация, например: *ФИО сотрудника*, *название комиссии*, *дата заседания* и т.д.

В базе данных присутствуют следующие связи между таблицами:

* «один к одному»;
* «один ко многим»;
* «многие ко многим».

Связь **«один к одному»** в данном случае нужна для *связи председателя комиссии с сотрудником, так как у комиссии может быть только один председатель, а сотрудник может быть председателем только в одной комиссии.*

**Связь «один ко многим»** («многие к одному») определяет ситуацию, когда одной записи родительской таблицы соответствует несколько записей дочерней таблицы. Например: *одна и та же должность может быть у нескольких сотрудников, но у сотрудника может быть только одна должность.*

Связь **«многие ко многим»** определяет ситуацию, когда любой записи одной таблицы может соответствовать много записей другой таблицы и наоборот.

Связь «**М-М**» – исключительно «человеческое» понятие и в реальных БД реализуется через две связи «1-М». Примером такой связи является связь между таблицами «comissions» и «employees». То есть*, один сотрудник может состоять в нескольких комиссиях, а комиссия может состоять из разных сотрудников*.

Для того, чтобы посмотреть контакты сотрудников, таблицы «contacts» и «employees» связаны между собой по внешнему ключу *employee\_id* и т.д*.*

Следовательно, все данные в базе данных взаимосвязаны между собой и добавление, изменение, удаление данных в одной таблице ведёт к добавлению, изменению, удалению в другой.

Созданная база данных соответствует всем требованиям, предъявляемым к проекту, в частности:

* нормализация до третьей нормальной формы;
* не менее 10 сущностей в базе данных.

Более подробная информация о базе данных и о связях между таблицами представлена на диаграмме базы данных (см. прил. А).

Ниже представлена архитектура моего приложения, то есть, как моё клиентское приложение будет взаимодействовать с базой данной (см. рис. 3.1).

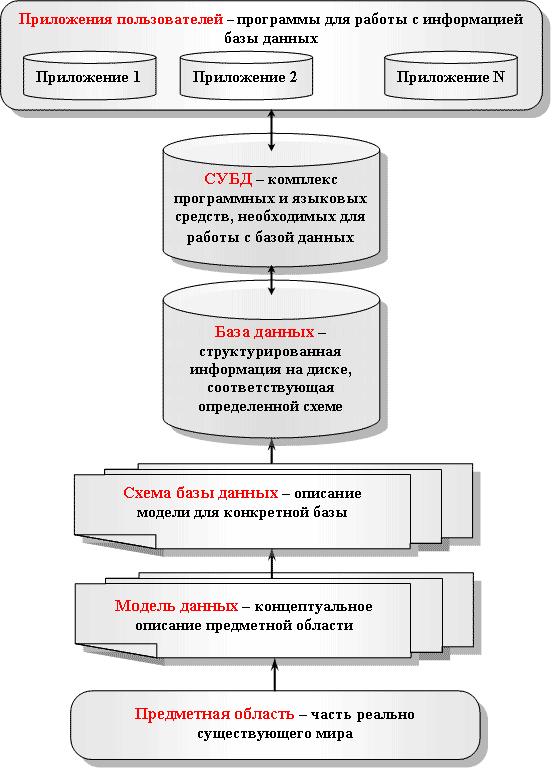


Рисунок 3.1 – Концепция работы клиентского приложения с базой данных

**4 Разработка клиентской части информационной системы**

**4.1 .NET Framework**

.NET Framework — программная платформа, выпущенная компанией [Microsoft](http://ru.wikipedia.org/wiki/Microsoft) в [2002 году](http://ru.wikipedia.org/wiki/2002_%D0%B3%D0%BE%D0%B4). Основой платформы является исполняющая среда [Common Language Runtime (CLR)](http://ru.wikipedia.org/wiki/Common_Language_Runtime), способная выполнять как обычные программы, так и серверные веб-приложения. NET Framework поддерживает создание программ, написанных на разных языках программирования. [1]

Основной идеей при разработке .NET Framework являлось обеспечение свободы разработчика за счёт предоставления ему возможности создавать приложения различных типов, способные выполняться на различных типах устройств и в различных средах.

Вторым принципом стало ориентирование на системы, работающие под управлением семейства операционных систем Microsoft Windows.

Программа для .NET Framework, написанная на любом поддерживаемом языке программирования, сначала переводится компилятором в единый для .NET понятный человеку низкоуровневый язык [Common Intermediate Language (CIL)](http://ru.wikipedia.org/wiki/Common_Intermediate_Language) (ранее назывался Microsoft Intermediate Language, MSIL). Затем компилятор производит перевод CIL-кода в объектный [байт-код](http://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4) (в терминах .NET получается сборка, англ. assembly), а уже [байт-код](http://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4) 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 неё [JIT-компилятор](http://ru.wikipedia.org/wiki/JIT) «на лету» (just in time) преобразует промежуточный [байт-код](http://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4) в машинные коды нужного процессора. Современная технология динамической компиляции позволяет достигнуть высокого уровня быстродействия. Виртуальная машина CLR также сама заботится о базовой безопасности, управлении памятью и [системе исключений](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B9), избавляя разработчика от части работы.

Одной из основных идей Microsoft .NET является совместимость программных частей, написанных на разных языках. Например, служба, написанная на C++ для Microsoft .NET, может обратиться к методу класса из библиотеки, написанной на Delphi; на C# можно написать класс, наследованный от класса, написанного на Visual Basic .NET, а исключение, созданное методом, написанным на C#, может быть перехвачено и обработано в Delphi.

**4.2 Windows Forms**

Windows Forms — [интерфейс программирования приложений](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9) (API), отвечающий за [графический интерфейс пользователя](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) и являющийся частью [Microsoft](https://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft) [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework). Данный интерфейс упрощает доступ к элементам интерфейса [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows) за счет создания обёртки для существующего [Win32 API](https://ru.wikipedia.org/wiki/Windows_API) в [управляемом коде](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4). Причем управляемый код — классы, реализующие [API](https://ru.wikipedia.org/wiki/API) для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, С++, так и на VB.Net, J# и др. [2]

С одной стороны, Windows Forms рассматривается как замена более старой и сложной библиотеке [MFC](https://ru.wikipedia.org/wiki/Microsoft_Foundation_Classes), изначально написанной на языке [C++](https://ru.wikipedia.org/wiki/C%2B%2B). С другой стороны, WF не предлагает парадигму, сравнимую с [MVC](https://ru.wikipedia.org/wiki/Model-View-Controller). Для исправления этой ситуации и реализации данного функционала в WF существуют сторонние библиотеки. Одной из наиболее используемых подобных библиотек является [User Interface Process Application Block](http://www.microsoft.com/downloads/details.aspx?familyid=98C6CC9D-88E1-4490-8BD6-78092A0F084E&displaylang=en), выпущенная специальной группой Microsoft, занимающейся примерами и рекомендациями, для бесплатного скачивания. Эта библиотека также содержит исходный код и обучающие примеры для ускорения обучения.

Внутри .NET Framework, Windows Forms реализуется в рамках [пространства имён](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE_%D0%B8%D0%BC%D1%91%D0%BD_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) System.Windows.Forms.

Приложение Windows Forms представляет собой [событийно-ориентированное приложение](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), поддерживаемое Microsoft [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework). В отличие от [пакетных программ](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BA%D0%B5%D1%82%D0%BD%D0%B0%D1%8F_%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0), большая часть времени тратится на ожидание от пользователя каких-либо действий, как, например, ввод текста в [текстовое поле](https://ru.wikipedia.org/w/index.php?title=%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%BE%D0%B5_%D0%BF%D0%BE%D0%BB%D0%B5&action=edit&redlink=1) или клика мышкой по [кнопке](https://ru.wikipedia.org/wiki/%D0%9A%D0%BD%D0%BE%D0%BF%D0%BA%D0%B0_(%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0)).

**4.3 Технология ADO.NET**

ADO.NET — это набор классов, предоставляющих службы доступа к данным программисту, работающему на платформе .NET Framework. ADO.NET имеет богатый набор компонентов для создания распределенных приложений, совместно использующих данные. Это неотъемлемая часть платформы .NET Framework, которая предоставляет доступ к реляционным данным, XML-данным и данным приложений. ADO.NET удовлетворяет различные потребности разработчиков, включая создание клиентских приложений баз данных, а также бизнес-объектов среднего уровня, используемых приложениями, средствами, языками и браузерам. [2]

ADO.NET Entity Framework (EF) — объектно-ориентированная технология доступа к данным, является [object-relational mapping](http://ru.wikipedia.org/wiki/ORM" \o "ORM) (ORM) решением для [.NET Framework](http://ru.wikipedia.org/wiki/.NET_Framework) от [Microsoft](http://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft). Предоставляет возможность взаимодействия с объектами как посредством [LINQ](http://ru.wikipedia.org/wiki/LINQ) в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используется как [ADO.NET Data Services](http://ru.wikipedia.org/wiki/ADO.NET_Data_Services) (*[Astoria](http://ru.wikipedia.org/wiki/ADO.NET_Data_Services" \o "ADO.NET Data Services)*), так и связка из[Windows Communication Foundation](http://ru.wikipedia.org/wiki/Windows_Communication_Foundation) и [Windows Presentation Foundation](http://ru.wikipedia.org/wiki/Windows_Presentation_Foundation" \o "Windows Presentation Foundation), позволяющая строить многоуровневые приложения, реализуя один из шаблонов проектирования [MVC](http://ru.wikipedia.org/wiki/Model-View-Controller), [MVP](http://ru.wikipedia.org/wiki/Model-View-Presenter) или [MVVM](http://ru.wikipedia.org/wiki/Model-View-ViewModel). [3]

ADO .NET (ActiveX Data Objects .NET) является набором классов, реализующих программные интерфейсы для облегчения подключения к базам данных из приложения независимо от особенностей реализации конкретной системы управления базами данных и от структуры самой базы данных, а также независимо от места расположения этой самой базы — в частности, в распределенной среде (клиент-серверное приложение) на стороне сервера.

ADO .NET широко используется совместно с технологией web-программирования с использованием объектов ASP .NET для доступа к расположенным на сервере базам данных со стороны клиента.

**4.3.1 Реляционные базы данных. Основные понятия**

Ниже обсуждаются наиболее общие понятия, связанные с представлением реляционной базы данных с точки зрения программиста, использующего ADO .NET. [3]

Детали реализации конкретной базы данных в рамках данной СУБД не обсуждаются. ADO .NET для того и используется, чтобы максимально скрыть детали реализации конкретной базы и предоставить программисту набор стандартных классов, интерфейсов, программных средств, которые превращают процесс создания приложения в стандартизированный технологический процесс. Таким образом, с точки зрения .NET:

1. Столбец (поле, атрибут):

* характеризуется определенным типом (данных);
* множество значений столбца являются значениями одного типа.

1. Строка (запись, кортеж):

* характеризуется кортежем атрибутов;
* состоит из упорядоченного множества значений (кортежа) атрибутов.

1. Таблица:

* набор данных, представляющих объекты определенного типа;
* состоит из множества элементов столбцов-строк;
* каждая строка таблицы уникальна.

1. Первичный ключ таблицы:

* непустое множество столбцов таблицы (возможно, состоящее из одного столбца), соответствующие значения (комбинации значений) которых в строках таблицы обеспечивают уникальность каждой строки в данной таблице.

1. Внешний ключ таблицы:

* непустое множество столбцов таблицы (возможно, состоящее из одного столбца), соответствующие значения (комбинации значений) которых в строках таблицы соответствуют первичному или дополнительному ключу другой таблицы;
* обеспечивает логическую связь между таблицами.

**4.3.2 Работа с базами данных**

Работа с БД на уровне приложения .NET – это работа:

* с множествами объявлений классов, которые содержат объявления унаследованных методов и свойств, предназначенных для решения задачи извлечения информации из базы данных;
* с множеством объектов-представителей классов, которые обеспечивают работу с базами данных;
* с множеством значений и свойств конкретных объектов, отражающих специфику структуры конкретной базы данных.

Функциональные особенности этой сложной системы взаимодействующих классов обеспечивают единообразную работу с базами данных независимо от системы управления базой и ее реализации.

Конечно же, при написании программы, взаимодействующей с базами данных, программист все может сделать своими руками. Правда, в силу сложности задачи (много всяких деталей придется вытачивать), времени на разработку такого приложения может потребоваться достаточно много.

Для программиста – разработчика приложения принципиальной становится информация о логической организации (структуре таблиц, отношениях и ограничениях) данной конкретной базы данных — то есть о том, как эту базу видит приложение.

Деятельность программиста – разработчика приложений для работы с базами данных в основе своей ничем не отличается от того, что было раньше. Те же объявления классов и интерфейсов.

А потому желательно:

* понимать принципы организации и взаимодействия классов, которые обеспечивают работу с базой данных;
* представлять структуру, назначение и принципы работы соответствующих объектов;
* знать, для чего и как применять различные детали при организации взаимодействия с базами данных;
* уметь создавать компоненты ADO .NET заданной конфигурации с использованием вспомогательных средств (волшебников), предоставляемых в рамках Visual Studio .NET.

**4.3.3 Доступ к отсоединённым данным**

В приложениях, работающих с базами данных, до недавних пор применялся доступ к данным через постоянное соединение с источником данных. Приложение открывало соединение с базой данных и не закрывало его по крайней мере до завершения работы с источником данных. В это время соединение с источником поддерживалось постоянно.

Недостатки такого подхода стали выявляться после появления приложений, ориентированных на Интернет.

Соединения с базой данных требуют выделения системных ресурсов, и если база данных располагается на сервере, то при большом количестве клиентов это может быть критично для сервера. Хотя постоянное соединение и позволяет немного ускорить работу приложения, общий убыток от растраты системных ресурсов преимущество в скорости выполнения приложения сводит на нет.

Факт плохого масштабирования приложений с постоянным соединением известен давно. Соединение с парой клиентов обслуживается приложением хорошо, 10 клиентов обслуживаются хуже, 100 – много хуже...

В ADO .NET используется другая модель доступа – доступ к отсоединенным данным. При этом соединение устанавливается лишь на то время, которое необходимо для проведения определенной операции над базой данных.

Модель доступа – модель компромиссная. В ряде случаев она проигрывает по производительности традиционной модели, и для этих случаев рекомендуется вместо ADO .NET использовать ADO. [3]

**4.3.4 Доступ к данным**

Объектная модель ADO .NET реализует отсоединенный доступ к данным. При этом в Visual Studio .NET существует множество ВСТРОЕННЫХ мастеров и дизайнеров, которые позволяют реализовать механизмы доступа к БД еще на этапе разработки программного кода.

С другой стороны, задача получения доступа к данным может быть решена непосредственно во время выполнения приложения.

Концепция доступа к данным в ADO .NET основана на использовании двух компонентов:

* НАБОРА ДАННЫХ (представляется объектом класса DataSet) со стороны клиента. Это локальное временное хранилище данных;
* ПРОВАЙДЕРА ДАННЫХ (представляется объектом класса Data Provider). Это посредник, обеспечивающий взаимодействие приложения и базы данных со стороны базы данных (в распределенных приложениях – со стороны сервера). [3]

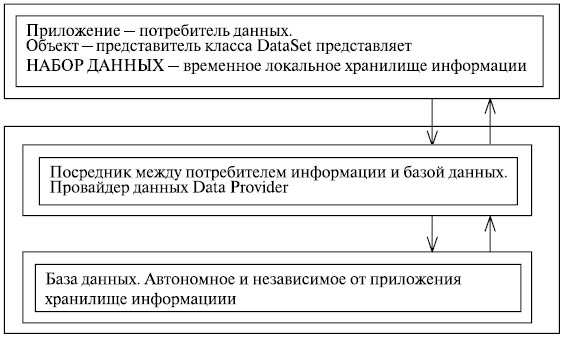


Рисунок 4.1 – Концепция доступа к данным в ADO.NET

**4.3.5 Объектная модель**

Объектная модель ADO .NET предполагает существование (при написании приложения для работы с базой данных — использование) двух множеств классов, выполняющих четко определенные задачи при работе с базой данных:

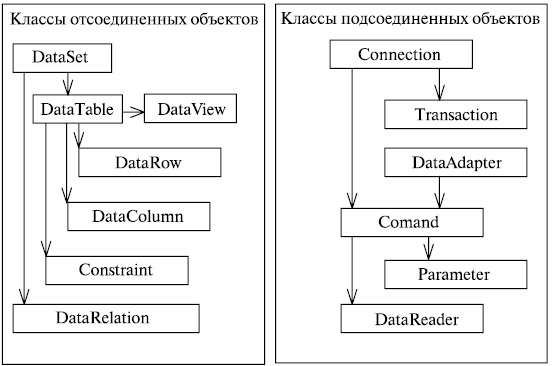


Рисунок 4.2 – Объектная модель ADO.NET

Классы подсоединенных объектов обеспечивают установление соединения с базой данных и управление базой со стороны приложения; классы отсоединенных объектов обеспечивают сохранение, использование и преобразование полученной от базы данных информации на стороне приложения.

Далее рассматриваются классы отсоединенных объектов объектной модели ADO .NET. Их подробному описанию посвящаются следующие разделы пособия. При этом классы отсоединенных объектов могут быть самостоятельно использованы в приложении наряду с обычными компонентами и элементами управления, даже если в приложении и не предполагается организовывать работу с базами данных. [3]

**4.4 Среда разработки**

Microsoft Visual Studio — линейка продуктов компании [Microsoft](https://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft), включающих [интегрированную среду разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как [консольные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) [приложения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), так и приложения с [графическим интерфейсом](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F), в том числе с поддержкой технологии [Windows Forms](https://ru.wikipedia.org/wiki/Windows_Forms" \o "Windows Forms), а также [веб-сайты](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82), [веб-приложения](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), [веб-службы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) как в [родном](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), так и в [управляемом](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) кодах для всех платформ, поддерживаемых [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows), [Windows Mobile](https://ru.wikipedia.org/wiki/Windows_Mobile" \o "Windows Mobile), [Windows CE](https://ru.wikipedia.org/wiki/Windows_CE" \o "Windows CE), [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework), [Xbox](https://ru.wikipedia.org/wiki/Xbox" \o "Xbox), [Windows Phone](https://ru.wikipedia.org/wiki/Windows_Phone" \o "Windows Phone) [.NET Compact Framework](https://ru.wikipedia.org/wiki/.NET_Compact_Framework) и [Silverlight](https://ru.wikipedia.org/wiki/Silverlight). [1]

Visual Studio включает в себя [редактор исходного кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BA%D0%BE%D0%B4%D0%B0) с поддержкой технологии [IntelliSense](https://ru.wikipedia.org/wiki/IntelliSense" \o "IntelliSense) и возможностью простейшего [рефакторинга кода](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3). Встроенный [отладчик](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и дизайнер [схемы базы данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%85%D0%B5%D0%BC%D0%B0_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Visual Studio позволяет создавать и подключать сторонние дополнения ([плагины](https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD)) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем [контроля версий исходного кода](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D1%8F%D0%BC%D0%B8) (как например, [Subversion](https://ru.wikipedia.org/wiki/Subversion" \o "Subversion) и [Visual Source Safe](https://ru.wikipedia.org/wiki/Microsoft_Visual_SourceSafe" \o "Microsoft Visual SourceSafe)), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на [предметно-ориентированных языках программирования](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D0%BC%D0%B5%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) или инструментов для прочих аспектов [процесса разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) (например, клиент Team Explorer для работы с [Team Foundation Server](https://ru.wikipedia.org/wiki/Team_Foundation_Server)). [2]

Visual Studio 2012 распространяется в тех же редакциях, что и 2010. Изменения коснулись Visual Studio 2012 Express — устанавливаются все языки программирования, а не один как раньше (Visual Basic 2010 Express, Visual C# 2010 Express), а также теперь существует пять версий Visual Studio Express: Visual Studio Express 2012 для Web, Visual Studio Express 2012 для Windows 8, Visual Studio Express 2012 для Windows Desktop, Visual Studio Express 2012 для Windows Phone и Visual Studio Team Foundation Server Express 2012. Все версии распространяются, как отдельные приложения. Visual Studio Express 2012 для Windows 8 позволяет разрабатывать приложения для [Windows Store](https://ru.wikipedia.org/wiki/Windows_Store) с Modern-интерфейсом, а Visual Studio Express 2012 для Windows Desktop позволяет разрабатывать «классические» приложения для Рабочего стола. Что касается Visual Studio Team Foundation Server Express 2012, то эта версия поставляется с оболочкой Visual Studio 2012.

**4.5 Результат разработки клиентской части**

В результате выбора технологий и способов создания приложения, представленных выше, в пространстве имён «StateDumaProject» были созданы следующие формы, где содержатся описание и выполнение всех функций программы:

* **LoginPage** – необходим для входа администратора или пользователя в систему, а также для регистрации пользователей в ней (см. рис. 4.3). Для входа в систему пользователь вводит свой логин и пароль в соответствующие поля и нажимает кнопку «Вход». Для регистрации пользователь вводит свои данные, которые он в последующем будет использовать и нажимает кнопку регистрация. Все поля проверяются на правильность ввода значений.

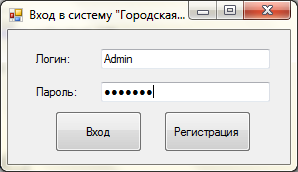


Рисунок 4.3 – Форма LoginPage для входа в систему

* **MainPage** – главная форма программы для манипуляции с данными. На данной форме реализован поиск по полям для каждой таблицы, возможность добавления, изменения, удаления данных в таблицах. Для поиска информации пользователь нажимает на кнопку «Поиск», выбирает, по какому полю он собирается искать и вводит искомое слово для поиска (см. рис. 4.4). Для того, чтобы добавить информацию, надо правой клавишей мыши щёлкнуть по необходимой таблице, появится меню, после чего нажать кнопку «Добавить» (см. рис. 4.5). Для того чтобы изменить данные или удалить, вызываем правой клавишей по таблице меню, а затем выбираем соответствующий пункт из меню. Стоит заметить, что операции добавления, редактирования, удаления доступны только администратору. Пользователю доступен поиск и просмотр информации в таблицах. Все действия, перечисленные выше, аналогичны для таблиц «Комиссии» и «Заседания».

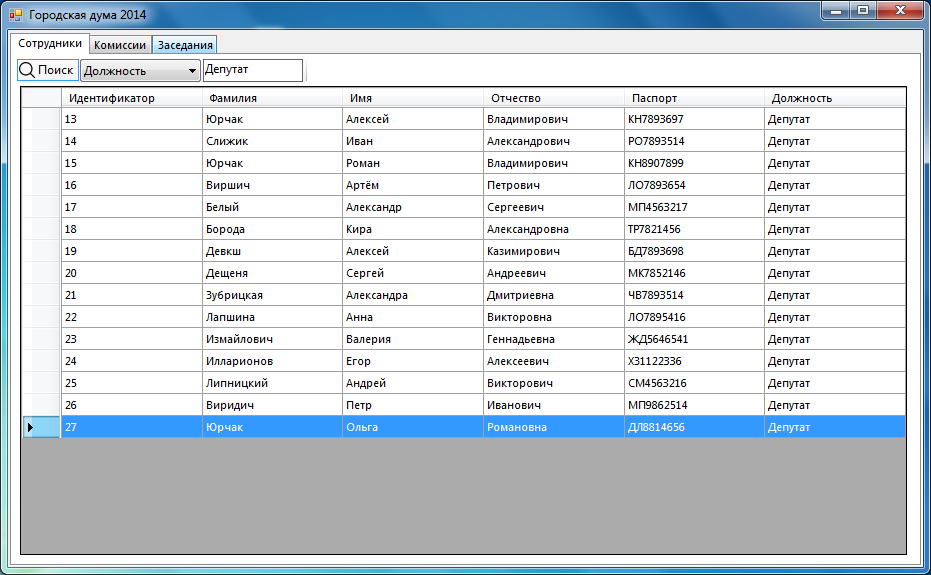


Рисунок 4.4 – Поиск в таблице «Сотрудники» информации о сотрудниках по полю «Должность» со значением «Депутат» на форме MainPage

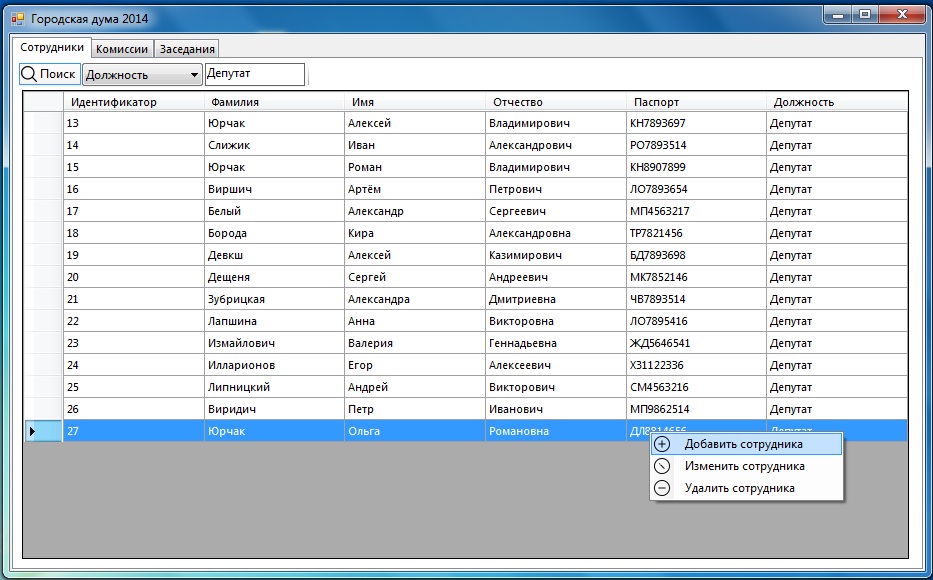


Рисунок 4.5 – Вызов меню для добавления сотрудника на форме MainPage

* **EditEmployeePage** – форма для добавления/редактирования информации о сотруднике. Администратор заполняет все поля на форме, а затем нажимает по ней правой клавишей мыши для вызова меню, где выбирает, сохранять ему информацию или выйти на главный экран, нажав кнопку «Отмена». Программа также проверяет правильность ввода данных. При наличии части незаполненных полей, программа выдаст предупреждение об этом. Для редактирования данных, администратор на главном экране в меню выбирает «Изменить», открывается форма EditEmployeePage, загружается вся информация о выбранном сотруднике из таблицы. Администратор редактирует необходимые поля, а затем, или сохраняет их, или отменяет (см. рис. 4.6).

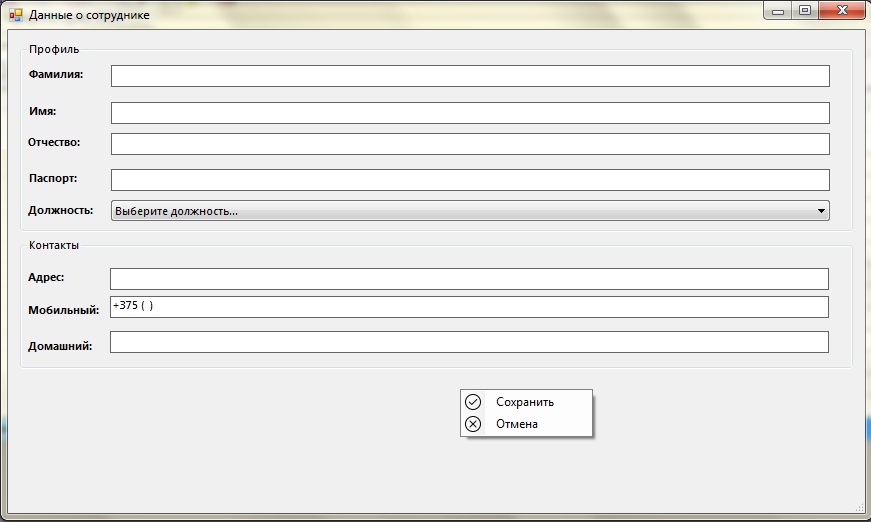


Рисунок 4.6 – Добавление/редактирование информации о сотруднике на форме EditEmployeePage

* **ShowEmployeePage** – на данной форме можно посмотреть более детальную информацию о сотруднике, о его комиссиях и встречах (см. рис. 4.7.).

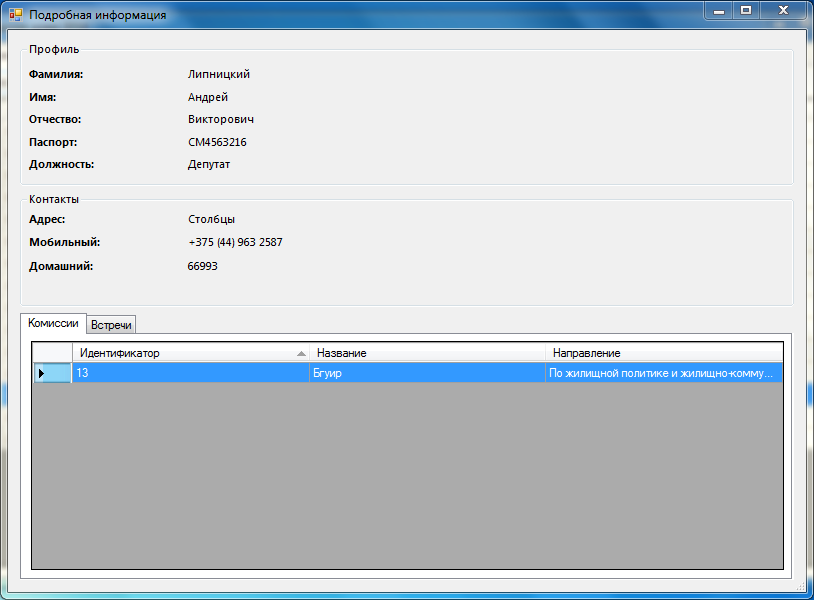


Рисунок 4.7 – Просмотр детальной информации о сотруднике на форме ShowEmployeePage

Все остальные формы добавления/редактирования для комиссий и заседаний по дизайну и функционированию аналогичны форме добавления/редактирования для сотрудника.

Весь список форм и их описание сведён в таблицу 4.1.

Таблица 4.1 – Описание форм программы

|  |  |
| --- | --- |
| Название формы | Описание |
| LoginPage | Вход в систему/регистрация пользователя в системе |
| MainPage | Содержит основную информацию о сотрудниках, комиссиях, заседаниях, возможен поиск по каждой категории информации. Также с этой формы осуществляется переход на формы добавления/редактирования и просмотра детальной информации, а также удаление информации |
| EditEmployeePage | Добавление/редактирование информации о сотруднике |
| EditComissionPage | Добавление/редактирование информации о комиссии |
| EditMeetingPage | Добавление/редактирование информации о заседании |
| ShowEmployeePage | Детальная информация о сотруднике |
| ShowComissionPage | Детальная информация о комиссии |
| ShowMeetingPage | Детальная информация о заседании |

Диаграмма классов программы представлена на рисунке 4.8.

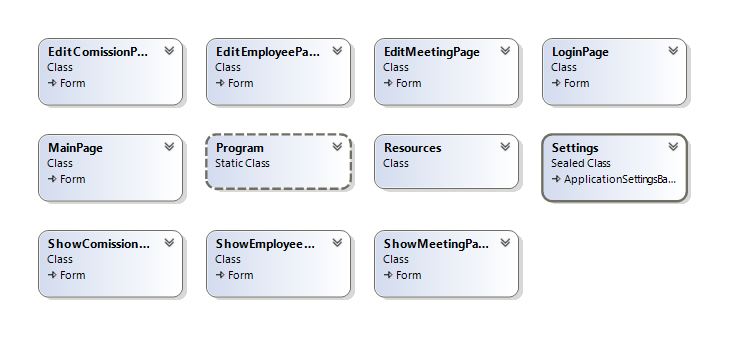
****

Рисунок 4.8 – Диаграмма классов программы «Городская дума»

**5 Проверка работоспособности разработанной информационной системы**

Для проверки работоспособности разработанной системы необходимо выполнить проверку входа в систему, основного функционала, соответствие прототипам, а также реакцию системы на ввод некорректных данных в различные поля. Для проверки функциональных возможностей будет использовано техническое задание (раздел 1 «Анализ технического задания»), описание требований к основному функционалу системы (раздел 2 «Проектирование структуры информационной системы»). Контрольный лист представлен в таблице табл.5.1 Контрольный лист проверки системы.

Таблица 5.1 – Контрольный лист проверки системы

|  |  |  |
| --- | --- | --- |
| Полномочие | Описание | Результат тестирования |
| 1. Авторизация администратора в системе. | Получение доступа к дополнительным возможностям системы с использованием индивидуальной учетной записи администратора при заполнении полей для входа. | Выполнено |
| 2. Добавление нового сотрудника (доступно только для администратора системы) | Создание новой записи в базе данных (таблица employees) с указанными: ФИО, паспортом, должностью, контактными данными | Выполнено |
| 3. Изменение данных о сотруднике (доступно только для администратора системы) | Изменение записи в базе данных (таблица employees) с указанными: ФИО, паспортом, должностью, контактными данными | Выполнено |
| 4. Удаление сотрудника (доступно только для администратора системы) | Удаление записи из базы данных (таблица employees) | Выполнено |
| 5. Поиск необходимой комиссии | Выбор необходимых параметров для поиска комиссии | Выполнено |

Продолжение таблицы 5.1

|  |  |  |
| --- | --- | --- |
| Полномочие | Описание | Результат тестирования |
| 6. Добавление нового заседания (доступно только для администратора системы) | Создание новой записи в базе данных (таблица meetings) с указанными: местом проведения, датой, длительностью, комиссией, списком присутствующим, состоянием | Выполнено |
| 7. Изменение данных о заседании (доступно только для администратора системы) | Изменение записи в базе данных (таблица meetings) с указанными: местом проведения, датой, длительностью, комиссией, списком присутствующим, состоянием | Выполнено |
| 8. Удаление заседания (доступно только для администратора системы) | Удаление записи из базы данных (таблица meetings) | Выполнено |
| 9. Поиск необходимого заседания | Выбор необходимых параметров для поиска заседания | Выполнено |
| 10. Создание приглашения в виде отчёта | Создание приглашения всем сотрудникам, собравшимся на заседание с указанными: местом, датой, длительностью, комиссией, списком приглашённых | Выполнено |
| 11. Создание итогового отчёта о встречи | Создание итого отчёта по заседанию с указанными: местом, датой, длительностью, списком присутствовавших | Выполнено |

По представленным в данном разделе данным видно, что система соответствует изначально поставленным целям, отвечает всем заявленным функциональным требованиям и готова к использованию.

Заключение

В результате проделанной работы было создано программное средство, позволяющее всем пользователям системы «Городская дума», без особых усилий быть в курсе актуальной информации о сотрудниках, комиссиях и заседаниях, в любой момент получить доступ к информации.

Проведенное тестирование работы приложения не выявило ошибок. Это, однако, не исключает возможности их появления при проведении более глубокого и длительного тестирования.

Помимо этого система имеет доступный и привлекательный интерфейс, не перегружен лишними элементами, корректно реагирует на любые действия пользователя.

Данное программное средство имеет преимущество над уже имеющимися за счет простой навигации. Пользователю только требуется войти в систему или зарегистрироваться для дальнейшей работы.

В ходе разработки программного средства были изучены и активно использовались такие современные технологии программирования, как технология программирования ADO.NET, Entity Framework, среда разработки Visual Studio 2012 Ultimate, Windows Forms.

Программное средство предназначено для работы на операционных системах семейства Microsoft Windows: Microsoft Windows 8, Microsoft Windows 7, Microsoft Windows XP, Microsoft Windows Vista.

**Список использованных источников**

[1] Википедия [Электронный ресурс]. – Режим доступа : <https://ru.wikipedia.org/wiki>.

[2] Microsoft Developer Network [Электронный ресурс]. – Режим доступа : <http://msdn.microsoft.com/>.

[3] Основы ADO.NET [Электронный ресурс]. – Режим доступа : <http://citforum.ru/programming/csharp/ado_dot_net/>.

Приложение **А**  
**Диаграмма базы данных «Городская дума»**

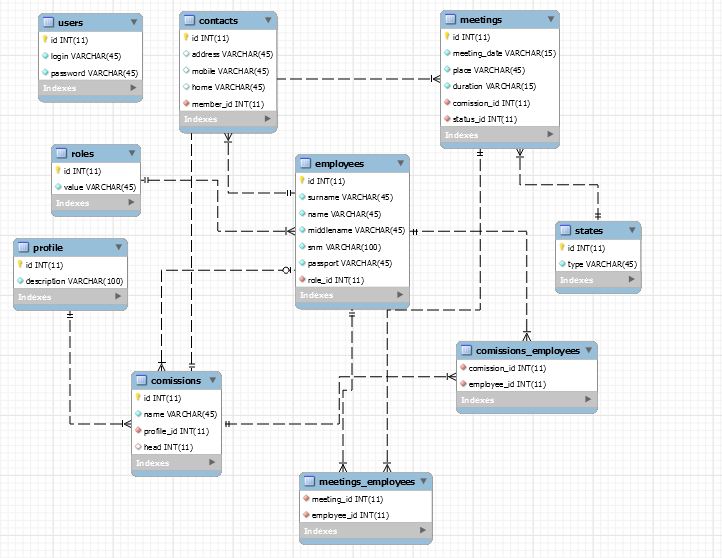
****

Рисунок А.1 – Диаграмма базы данных «Городская дума»