

Частное учреждение образования
«Колледж бизнеса и права»

ПРОГРАММА ДЛЯ АВТОМАТИЗАЦИИ УЧЕТА РАСХОДНЫХ МАТЕРИАЛОВ
ОФИСНОГО ОБОРУДОВАНИЯ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по дисциплине
«Базы данных и системы управления базами данных»

КП Т.594003.401

Руководитель проекта

(С.В. Банцевич)

Учащийся

(И.Д. Алейчик)

Содержание

Введение	4
1 Объектно-ориентированный анализ и проектирование системы	5
1.1 Сущность задачи	5
1.2 Информационная модель	7
2 Вычислительная система	10
2.1 Требования к аппаратным и операционным ресурсам	10
2.2 Инструменты разработки	10
3 Проектирование задачи	12
3.1 Требования к приложению	12
3.2 Концептуальный прототип	12
3.3 Организация данных	14
3.4 Функции и элементы управления	17
4 Описание программного средства	21
4.1. Общие сведения	21
4.2 Функциональное назначение	21
4.3 Входные и выходные данные	22
5 Методика испытаний	25
5.1 Технические требования	25
5.2 Функциональное тестирование	25
6 Применение	29
6.1 Назначение программы	29
6.2 Условия применения	29
6.3 Справочная система	29
Заключение	32
Список информационных источников	33
Приложение А Текст программы	34

					КП Т.594003.401 ПЗ						
Изм.	Лист	№ докум.	Подпись	Дата							
Разраб.	Алейчик И.Д.				Программа для автоматизации учета расходных материалов офисного оборудования		Лит.	Лист	Листов		
Провер.	Банцевич С.В.						у	3	34		
Реценз.							КБП				
Н. контр.											
Утверд.											

Введение

Методика отслеживания движений расходных материалов (РМ) предоставляет явные преимущества организации. Для облегчения работы сотрудникам, ведущим контроль за РМ, целесообразно пользоваться компьютерными программами, а также утвержденными методическими рекомендациями или технологическими картами на предприятии. При этом, очевидно, компьютерная программа обладает неоспоримыми преимуществами перед бумажным документооборотом.

Расходные материалы представляют собой одну из составных частей имущества предприятия, необходимую для нормального осуществления возложенных задач.

РМ обслуживают сферу производства и являются его материальной основой. Они необходимы для обеспечения процесса производства продукции.

Целью курсового проекта для автоматизации учета расходных материалов является разработка программного средства «AccountingConsumables.exe», которое будет осуществлять ведение базы данных, содержащей информацию о подразделениях, расходах, материалах, типах материалов, оборудовании, заказах, категориях и поставщиках, рассчитывающей средний срок службы офисного оборудования, контролирующей неснижаемый запас расходным материалов, генерирующей акт расхода материалов.

Пояснительная записка состоит из шести разделов, содержащих необходимую информацию по организации эксплуатации программного средства.

В первом разделе «Объектно-ориентированный анализ и проектирование системы» раскрывается организационная сущность задачи, описывается предметная область и круг задач, которые должны быть автоматизированы. Описывается задача, перечисляются основные функции программы. Строится информационная модель, отражающая сущности задачи, их свойства и взаимосвязи.

Во втором разделе «Вычислительная система» перечисляются требования к аппаратному и программному обеспечению компьютеров, проводится характеристика операционной системы, обоснование выбранных инструментов разработки программы.

В третьем разделе «Проектирование задачи» проводится объектно-ориентированный анализ задачи, строится концептуальный прототип диалоговых окон и элементов управления, приводится организация данных в контексте выбранной системы управления базами данных, отражаются основные функции программы, их логическая и физическая организация, приводится структура справочной системы.

В четвертом разделе «Описание программного средства» представлены общие сведения о программном средстве и его функциональном назначении, описываются входные и выходные данные.

В пятом разделе «Методика испытаний» описываются требования к аппаратному и программному обеспечению компьютера для проведения испытаний, требования к характеристикам программы применительно к условиям эксплуатации, требования к информационной и программной совместимости. Представляются результаты функционального тестирования.

Шестой раздел «Применение» предназначен для описания сведений о назначении программного средства и области его применения. В этом разделе приводится структура справочной системы, а также методика ее использования.

В заключении анализируется созданное программное средство, определяется степень соответствия поставленной задачи и выполненной работы.

Приложение А содержит текст программы.

В графической части представлены диаграммы вариантов использования, классов, деятельности, последовательности и компонентов.

1 Объектно-ориентированный анализ и проектирование системы

1.1 Сущность задачи

Предметной областью решаемой задачи является организация по сборке офисного оборудования. Организация занимается сборкой мебельного оборудования для офисов, а также дверей и перегородок из различных материалов.

Офисная мебель может отличаться не только презентабельным дизайном и высокой функциональностью, но и сложностью конструкции. В качестве элементов при изготовлении мебельных изделий могут использоваться различного рода механизмы и фурнитура, повышающие удобство их использования.

Клиентами данной организации могут быть как обычные клиенты, так и корпоративные клиенты.

В настоящее время в подобных организациях осуществляют сборку мебели или другого офисного оборудования под заказ. Для сборки используют материалы, поставляемые другими организациями.

Контроль за материалами и расходами на сборку офисной мебели осуществляется актом расхода материалов, пример которого приведен на рисунке 1.1.

Акт расхода материалов № 1 от 01.12.2018					
Подразделение: Сборочный цех 1					
Наименование продукции: Компьютерный стол					
⊕					
N п/п	Наименование материала, сырья	Кол-во	Дата поступления	Срок службы	Расход
1	Ножка	10	18.01.2018	21.12.2018	5
2	Столешница	10	13.04.2017	23.01.2019	10
	Болт	10	13.03.2018	27.06.2020	4
	Доска	10	13.02.2018	20.03.2018	6
□					
Ответственное лицо: _____			Кузнец Л.В.		
(подпись)			(расшифровка)		
Бухгалтер: _____			Алейчик И.Д.		
(подпись)			(расшифровка)		

Рисунок 1.1

При заполнении акта необходимо указать следующие данные:

- номер акта и дату его создания;
- подразделение, в котором были использованы материалы;
- наименование собираемой/ремонтируемой продукции.

Контроль произведенной продукции или продукции, имеющейся в наличии, осуществляется с помощью отчета об имеющемся оборудовании, пример которого приведен на рисунке 1.2

Отчет об имеющемся оборудовании

Дата отчета: 01.12.2018

Модель	Кол-во	Срок службы	Категория
Iris 202r0	2	11.05.2019	Стол
Iris 432	2	12.06.2019	Стеллаж
Iris 2453b	5	30.12.2019	Шкаф
Iris 2321	3	20.12.2019	Компьютерный стол
Iris 2554	7	10.05.2019	Кресло
Iris 2654	8	15.12.2019	Стул

Бухгалтер: _____ Алейчик И.Д.
(подпись) (расшифровка)

Рисунок 1.2

При заполнении отчета об имеющемся оборудовании необходимо указать дату создания отчета, имеющееся оборудование, количество, сроки службы и подписи должностных лиц.

Контроль за остатками осуществляется с помощью отчета по остаткам, отчет необходим для определения разности между поступлениями и расходами за определенный промежуток времени, пример приведен на рисунке 1.3

Отчет по остаткам № 1

Дата отчета: 18.12.2018

	Наименование	Дата поступления	Кол-во
1	Ножка	18.01.2018	4
2	Столешница	13.04.2017	5
3	Ручка	13.03.2018	6
4	Дверца	13.02.2018	2

Бухгалтер: _____ Алейчик И.Д.
(подпись) (расшифровка)

Рисунок 1.3

При заполнении отчета по остаткам необходимо указать номер отчета, дату его создания и подписи должностных лиц.

Исходя из исследования предметной области основными задачами, подлежащими автоматизации, будут являться:

- ведение базы данных, содержащей информацию о расходных материалах, типах материалов, оборудовании, категориях, расходах, подразделениях, заказах и поставщиках;
- контроль неснижаемого запаса расходных материалов расчет среднего срока службы офисного оборудования и расходных материалов;
- оценка динамики движения и остатков наименований офисного оборудования и расходных материалов для контроля разрешенного запаса при отпуске расходных материалов;
- формирование списка офисного оборудования и расходных материалов, имеющихся в наличии;
- поиск, сортировка и фильтрация данных;
- создание отчетов по интересующей пользователя информации: акт расхода материалов, отчет по остаткам, отчет об имеющемся оборудовании, отчет о расходных материалах, отчет о движении расходных материалов за период/на дату, отчет о сроке службы офисного оборудования/расходных материалов.

Аналогами разрабатываемой программы является программа «YuKoSoft Оборудование.exe». Однако программа будет обладать простым, удобным интерфейсом и легкой в использовании.

1.2 Информационная модель

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

Наиболее распространенным средством моделирования данных являются диаграммы «Сущность-связь» (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных. Нотация ERD была впервые введена П. Ченном и получила дальнейшее развитие в работах Баркера. Диаграмма «Сущность-связь» представлена на рисунке 1.4.

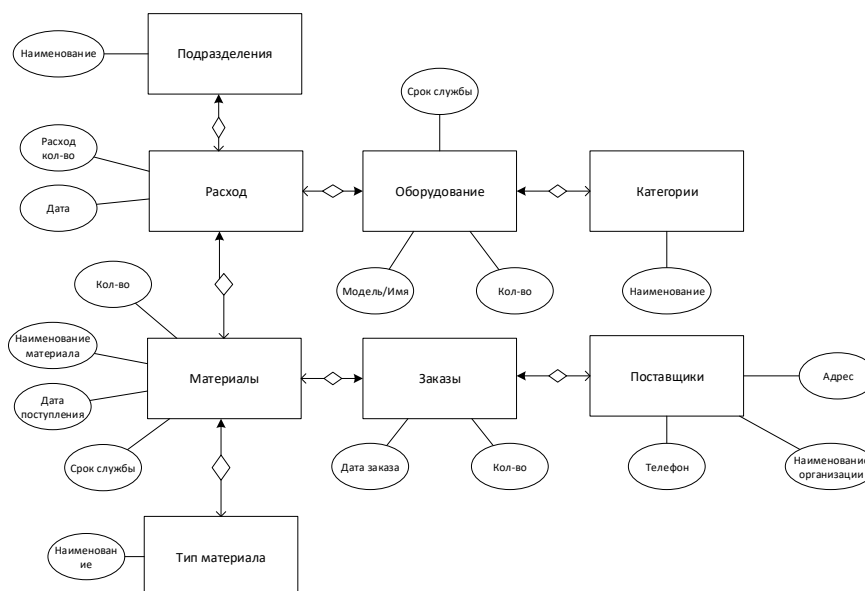


Рисунок 1.4 – Диаграмма «Сущность-связь»

Исходя из исследования предметной области, можно выделить следующие сущности разработки: «Категории», «Оборудование», «Расход», «Материалы», «Тип материала», «Заказы», «Поставщики», «Подразделения».

Для сущности «Категории» атрибутом будет являться «Наименование».

Для сущности «Оборудование» можно выделить следующие атрибуты:

- «Модель/имя»;
- «Кол-во»;
- «Срок службы».

Для сущности «Расход» атрибутами будут являться:

- «Расход/кол-во»;
- «Дата».

Для сущности «Материалы» можно выделить следующие атрибуты:

- «Наименование материала»;
- «Кол-во»;
- «Дата поступления»;
- «Срок службы».

Для сущности «Тип материала» можно выделить атрибут «Наименование».

Для сущности «Заказы» можно выделить следующие атрибуты:

- «Дата заказа»;
- «Кол-во».

Для сущности «Поставщики» можно выделить следующие атрибуты:

- «Наименование организации»;
- «Телефон»;
- «Адрес».

Для сущности «Подразделения» можно выделить атрибут «Наименование».

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

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

Актёр представляет собой внешнюю по отношению к моделируемой системе сущность, которая взаимодействует с системой и использует её функциональные возможности для достижения определённых целей или решения частных задач. При этом актёры служат для обозначения согласованного множества ролей, которые могут играть пользователи в процессе взаимодействия с проектируемой системой. Каждый актёр может рассматриваться как некоторая отдельная роль относительно конкретного варианта использования.

Диаграмма вариантов использования представлена в графической части на листе 1.

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

Диаграмма классов для проектируемой системы представлена в графической части на листе 2.

При моделировании поведения проектируемой или анализируемой системы возникает необходимость детализировать особенности алгоритмической и логической реализации

выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, переход в следующее состояние срабатывает только при завершении этой операции. Графически диаграмма деятельности представляется в форме графа, вершинами которого являются состояния действия, а дугами - переходы от одного состояния действия к другому.

Основная цель использования диаграмм деятельности - визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения.

Диаграмма деятельности для функции поиска информации представлена в графической части на листе 3.

Для моделирования взаимодействия объектов в UML используются соответствующие диаграммы взаимодействия. Если рассматривать взаимодействия объектов во времени, тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности.

Временной аспект поведения имеет существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели и используются диаграммы последовательности, в которых ключевым моментом является динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет как бы два измерения: одно - слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии; второе - вертикальная временная ось, направленная сверху вниз, на которой начальному моменту времени соответствует самая верхняя часть диаграммы.

Диаграмма последовательности для проектируемой системы представлена в графической части на листе 4.

Рассмотренные ранее диаграммы отражали концептуальные аспекты построения модели системы и относились к логическому уровню представления. Особенность логического представления заключается в том, что оно оперирует понятиями, которые не имеют самостоятельного материального воплощения. Другими словами, различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Однако для создания конкретной физической системы необходимо некоторым образом реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно физическое представление модели.

Диаграмма компонентов описывает объекты реального мира - компоненты программного обеспечения. Эта диаграмма позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами.

Вид диаграммы компонентов для данной проектируемой системы представлен в графической части на листе 5. Она содержит следующие компоненты:

- программные компоненты, созданные в среде Visual Studio: «.cs»;
- файл базы данных «AccountingConsumables.mdf»;
- журнал транзакций «AccountingConsumables_log.ldf»;
- файл справочной системы «index.html».

2 Вычислительная система

2.1 Требования к аппаратным и операционным ресурсам

Конфигурация компьютера, на котором будет разрабатываться программное приложение:

- процессор Intel Core i5 2500 МГц;
- оперативная память DDR3 8 Гбайт;
- жёсткий диск HDD 1 Тбайт;
- видеокарта nVidia GeForce GT 740m 2048 Мбайт;
- материнская плата Lenovo;

Для удобной работы с программой необходимо наличие клавиатуры и мыши.

2.2 Инструменты разработки

Инструментами разработки будут являться:

- операционная система Windows 10 Pro;
- среда разработки Microsoft Visual Studio 2017;
- система управления базами данных Microsoft SQL Server 2017 Express;
- среда для управления SQL Server, Microsoft SQL Server Management Studio;
- язык программирования C#;
- графический редактор, редактор блок схем Microsoft Visio;
- офисный пакет Microsoft Office;
- программа для создания инсталлятора Smart Install Maker.

Операционная система Windows 10 появилась относительно недавно – она стала доступной с 29 июля 2015 года. Компания Microsoft при разработке продолжала свой путь, направленный на унификацию. Допускается установка на компьютеры, ноутбуки, планшеты, а также смартфоны и консоли [6].

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight [7].

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основным используемый язык запросов — Transact-SQL. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка [8].

Microsoft SQL Server Management Studio — это графический набор средств, для разработки сценариев на T-SQL и управления всеми компонентами SQL Server. Management Studio является основным инструментом любого разработчика или администратора MS SQL сервера [9].

C# — это универсальный объектно-ориентированный язык программирования, обеспечивающий безопасность преобразования типов. Целью его создания было повышение производительности труда программистов. Поэтому в языке сбалансированы - простота, выразительность и эффективность [13].

Microsoft Visio — 2D программное обеспечение редактор диаграмм, блок-схем для Windows [10].

Microsoft Office — это офисный пакет приложений, созданных корпорацией Microsoft для операционных систем Microsoft Windows, Windows Phone, Android, OS X, iOS. В состав этого пакета входит программное обеспечение для работы с различными типами документов: текстами, электронными таблицами, базами данных. Microsoft Office является сервером OLE-объектов и его функции могут использоваться другими приложениями, а также самими приложениями Microsoft Office [11].

Smart Install Maker — утилита для создания инсталляционных пакетов. С помощью Smart Install Maker можно быстро и легко создавать инсталляторы профессионального уровня, обладающие понятным для пользователя интерфейсом и небольшим размером. Сгенерированные в Smart Install Maker установочные пакеты способны производить изменения в системном реестре, запускать программы, регистрировать элементы ActiveX и новые для системы шрифты, проверять версию .NET Framework, выводить лицензионные соглашения и прочие тексты [12].

3 Проектирование задачи

3.1 Требования к приложению

Программное средство предполагает дальнейшее использование пользователями не обладающих специальной профессиональной компетенцией по работе с современными информационными технологиями, что определяет концептуальный прототип программного приложения, методы работы с приложением, способ его инсталляции на персональном компьютере. А значит, приложение должно иметь удобный интерфейс для простоты использования его пользователем.

Программное приложение должно:

- содержать привычные и понятные пользователю пункты меню, соответствующие функциям обработки;
- ориентироваться на пользователя, который общается с программой на внешнем уровне взаимодействия;

Пользователю предлагается выбор альтернативных функций обработки из фиксированного перечня, меню может содержать вложенные подменю.

Диалоговые окна должны содержать элементы управления:

- поля ввода информации пользователя;
- списки возможных альтернатив для выбора;
- кнопки.

3.2 Концептуальный прототип

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

При создании данного приложения важную роль играют формы, так как они являются основным диалоговым средством работы пользователя. Разрабатываемое приложение будет содержать главную форму и две дополнительных. Такая структура интерфейса позволит классифицировать основные функции программы по группам.

При проектировании концептуального прототипа предполагается, что при загрузке программы первой будет загружаться форма «УРМ Офисного оборудования».

Макет формы «УРМ Офисного оборудования» представлен на рисунке 3.1

Макет формы «УРМ Офисного оборудования» представляет собой графический интерфейс с заголовком «УРМ Офисного оборудования» и стандартными кнопками управления окном. Интерфейс разделен на две основные части. Слева находится панель навигации с кнопками: «Поставщики», «Категории», «Подразделения» и «Типы». Справа находится основная рабочая область, содержащая панель меню с пунктами «Файл», «Редактирование» и «Справка». Под панелью меню расположены кнопки «Материалы», «Оборудование», «Поставщики» и «Типы». В центре этой области находится большое текстовое поле с надписью «Поиск». В нижнем правом углу основной области расположена кнопка «Редактировать».

Рисунок 3.1 – Макет формы «УРМ Офисного оборудования»

На главной форме располагаются кнопки «Материалы», «Оборудование», «Расходы», «Заказы», «Поставщики», «Категории», «Подразделения», «Типы материалов», которые осуществляют переходы между таблицами. На главной форме располагается элемент управления вложенным меню с пунктами: «Файл», «Редактирование», «Справка», которые предназначены для открытия справочной системы, выходы из программы, редактирования базы данных, формирования отчетов. Форма содержит компонент «Data Grid», который предназначен для отображения данных в табличном формате.

На форме «Редактирование» располагаются текстовые поля для ввода данных, таблицы для просмотра информации, кнопки «Добавить», «Удалить», «Сохранить» для работы с данными. Макет формы «Редактирование» представлен на рисунке 3.2.

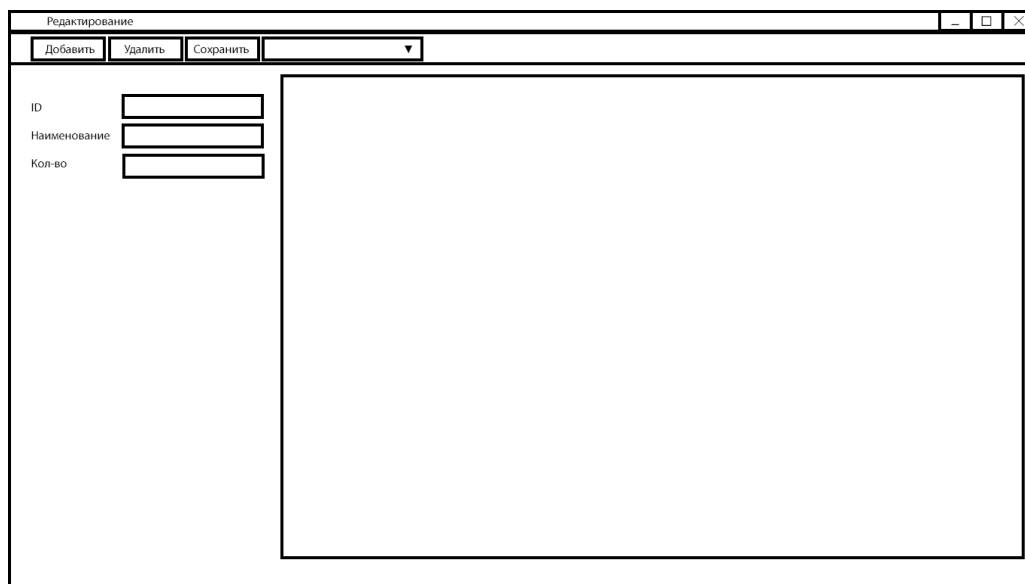


Рисунок 3.2 – Макет формы «Редактирование»

На форме «Генератор отчетов» располагается выпадающее меню для выбора отчета и кнопка «Создать», предназначенная для создания отчета. Макет формы «Генератор отчетов» представлен на рисунке 3.3.

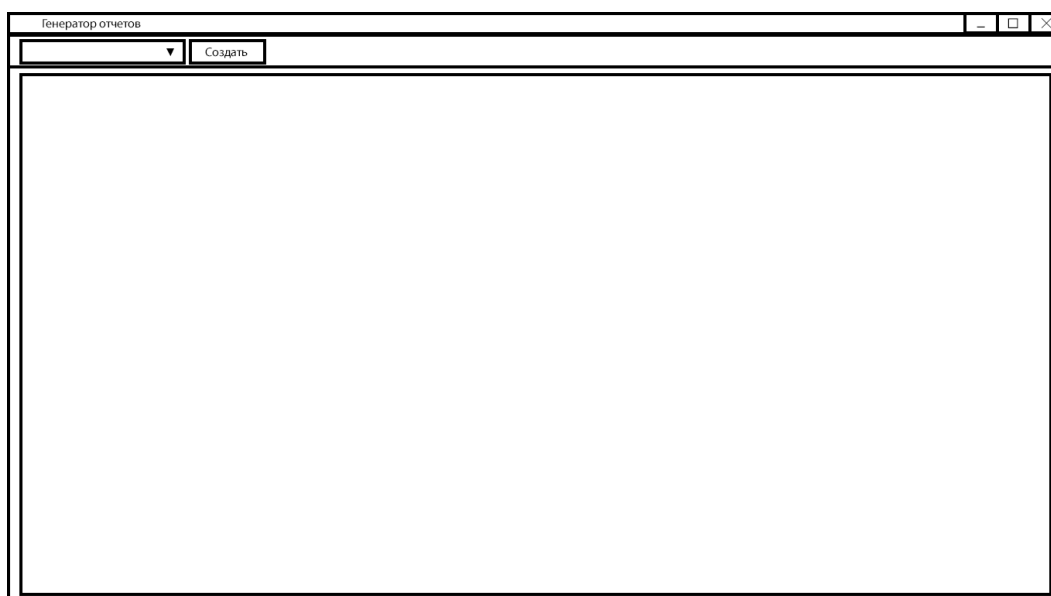


Рисунок 3.3 – Макет формы «Генератор отчетов»

3.3 Организация данных

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

Реляционная модель основана на математическом понятии отношения, представлением которого является таблица. В реляционной модели отношения используются для хранения информации об объектах, представленных в базе данных. Отношение имеет вид двухмерной таблицы, в которой строки соответствуют записям, а столбцы - атрибутам. Каждая запись должна однозначно характеризоваться в таблице. Для этого используют первичные и вторичные ключи. Достоинством реляционной модели является простота и удобство физической реализации.

Реляционная модель базы данных подразумевает нормализацию всех таблиц данных. Нормализация — это формальный метод анализа отношений на основе их первичного ключа и функциональных зависимостей, существующих между их атрибутами.

Структура базы данных разрабатываемого программного средства на данный момент включает восемь таблиц.

Структура данных таблиц, и их краткое описание приводится в таблицах 1-8.

Таблица «Category» хранит информацию о категориях оборудования. Структура приведена в таблице 3.1.

Таблица 3.1 – Структура таблицы «Category»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Category	int	4	Идентификатор категории оборудования
NameCategory	varchar	20	Наименование категории оборудования

Таблица «Expenses» хранит информацию о расходах материалов. Структура приведена в таблице 3.2.

Таблица 3.2 – Структура таблицы «Expenses»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Expenses	int	4	Идентификатор расхода
DateExpense	date	3	Дата расхода
Involved	int	4	Расход
ID_Equipment	int	4	Идентификатор оборудования
ID_Materials	int	4	Идентификатор материала
ID_Departament	int	4	Идентификатор подразделения

Таблица «Equipment» хранит информацию об оборудовании. Структура приведена в таблице 3.3.

Таблица 3.3 – Структура таблицы «Equipment»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Eqerment	int	4	Идентификатор оборудования
Model	varchar	20	Модель/Наименование оборудования
CountEquipment	int	4	Кол-во
Life	date	3	Срок службы
ID_Category	int	4	Идентификатор категории

Таблица «Materials» хранит информацию о материалах. Структура приведена в таблице 3.4.

Таблица 3.4 – Структура таблицы «Materials»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Materials	int	4	Идентификатор материалов
NameMaterial	varchar	20	Наименование материала
CountMaterial	int	4	Ко-во
DateReceipt	date	3	Дата прибытия
Life	date	3	Срок службы
ID_TypeMatreial	int	4	Идентификатор типа материалов

Таблица «Orders» хранит информацию о заказах. Структура приведена в таблице 3.5.

Таблица 3.5 – Структура таблицы «Orders»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Orders	int	4	Идентификатор заказов
DateOrder	date	3	Дата заказа
CountOrders	int	4	Кол-во
ID_Materials	int	4	Идентификатор материала
ID_Providers	int	4	Идентификатор поставщика

Таблица «TypeMaterial» хранит информацию о типах материала. Структура приведена в таблице 3.6.

Таблица 3.6 – Структура таблицы «TypeMaterial»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_TypeMat	int	4	Идентификатор типов материалов
NameTypeMat	varchar	10	Наименование типа

Таблица «Providers» хранит информацию о поставщиках. Структура приведена в таблице 3.7.

Таблица 3.7 – Структура таблицы «Providers»

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Providers	int	4	Идентификатор
NameOrg	varchar	20	Наименование организации
Phone	nvarchar	15	Телефон
AddresProviders	varchar	20	Адрес организации

Таблица «Departament» хранит информацию о подразделениях. Структура приведена в таблице 3.8.

Таблица 3.8 – Структура таблицы «Departament».

Имя поля	Тип поля	Размер поля,байт	Описание поля
ID_Departament	int	4	Идентификатор подразделений
NameDep	varchar	20	Наименование подразделения

Физическая структура базы данных представлена на рисунке 3.4

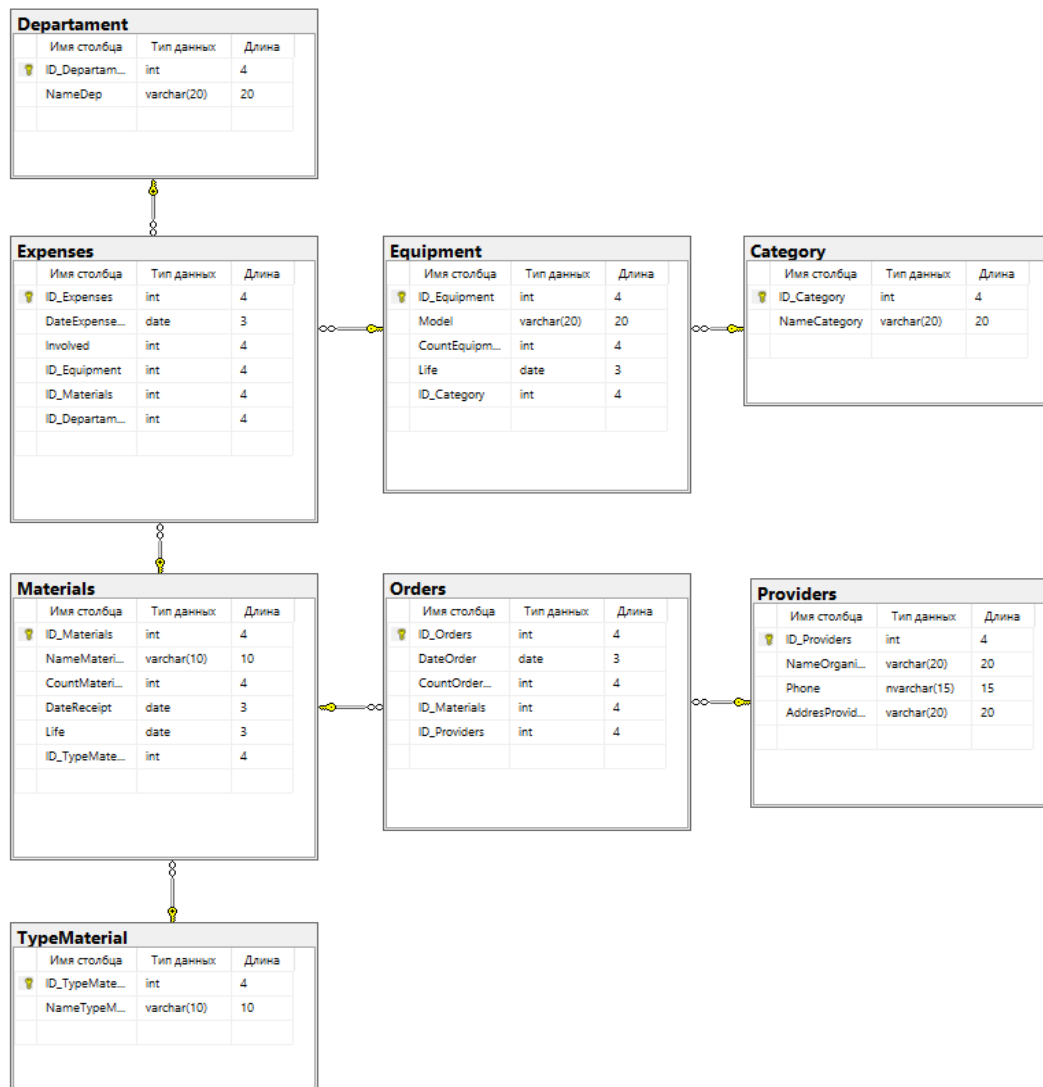


Рисунок 3.4 – Физическая схема базы данных.

3.4 Функции и элементы управления

Основными функциями данного проекта являются ведение базы, и генерация отчетов с информацией о расходных материалах.

Для выбора элемента из таблицы необходимо на форме «Редактирование» нажать на компонент DataGrid. Код функции представлен ниже.

```
private void DataGrid_SelectionChanged(object sender, SelectionChangedEventArgs e){
    DataGrid dg = sender as DataGrid;
    DataRowView dr = dg.SelectedItem as DataRowView;
    if (dr != null){
        try{
            txtIdTypeMat.Text = dr["ID_TypeMaterial"].ToString();
            txtNameTypeMat.Text = dr["NameTypeMaterial"].ToString();
        }
        catch { }
    }
}
```


Для добавления данных о типе материала необходимо на форме «Редактирование» выбрать таблицу нажав на компонент ComboBox, ввести соответствующие данные, нажать на компонент Button под названием «Добавить». Код функции представлен ниже.

```
private void Add_btn_Click(object sender, RoutedEventArgs e){
    try{
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        string query = "INSERT INTO TypeMaterial(NameTypeMaterial) Values('" +
txtNameTypeMat.Text + "')";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM TypeMaterial", TypeMaterialGrid);
        connection.Close();
    }
    catch (Exception ex){
        MessageBox.Show(ex.Message);
    }
    finally{
        if (connection != null)
            connection.Close();
    }
}
```

При нажатии на компонент Button под названием «Удалить» будет срабатывать функция удаления. Код функции представлен ниже.

```
private void delete_btn_Click(object sender, RoutedEventArgs e){
    table = new DataTable();
    SqlConnection connection = null;
    try{
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        string query = "delete from TypeMaterial where ID_TypeMaterial = " +
txtIdTypeMat.Text;
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM TypeMaterial", TypeMaterialGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
```

Для изменения данных необходимо нажать на компонент Button под названием «Сохранить». Код функции представлен ниже.

```
private void save_btn_Click(object sender, RoutedEventArgs e){
    table = new DataTable();
    SqlConnection connection = null;
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        string query = "update TypeMaterial set NameTypeMaterial = '" +
txtNameTypeMat.Text+"'" + "where ID_TypeMaterial = '" + txtIdTypeMat.Text + "'";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM TypeMaterial", TypeMaterialGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
```

Для формирования отчета необходимо на главной форме нажать на компонент Menu, выбрать подпункт меню под названием «Сформировать отчет», будет открыта форма «генерация отчетов», дальше выбрать отчет из компонента ComboBox и нажать на компонент Button под названием «Сформировать». Код функции формирования отчета представлен ниже.

```
private void btnGenerate_Click(object sender, RoutedEventArgs e)
{
    string sql = "";
    DataTable dt;
    ReportDataSource dataSource;
    switch (selected)
    {
        case "Движение материалов за период на дату":
            sql = "ExpensesViewReport";
            ReportViewer.Reset();
            dt = main.getDataFill(sql);
            dataSource = new ReportDataSource("DataSet1", dt);
            ReportViewer.LocalReport.DataSources.Add(dataSource);
            ReportViewer.LocalReport.ReportEmbeddedResource
"AccountingConsumables.Report3.rdlc";
            ReportViewer.RefreshReport();
            break;
    }
}
```

Полный текст программы представлен в приложении А.

3.5 Проектирование справочной системы приложения

Для корректной работы с приложением требуется обеспечить пользователя справочной системой, в которой будут приведены приемы работы с приложением, включающие данные о том, что произойдет после нажатия на определенную кнопку или при выборе вкладки.

Справочная система необходима для ознакомления с программой. В ней должна присутствовать информация, которая поможет в решении проблемы с приложением, а также может напомнить, как пользоваться программным средством.

Система справки данного приложения будет содержать следующие разделы:

- «О программе»;
- «Как пользоваться»;
- «Условия использования».

Справочная система будет создана на языке гипертекстовой разметки «HTML».

(HyperText Markup Language) HTML — это стандартизированный язык разметки документов во всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML. Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства. [14].

Справка будет вызываться клавишей F1 на главной форме.

4 Описание программного средства

4.1. Общие сведения

Программное средство «AccountingConsumables.exe» предназначено снизить затраты времени на обработку информации о расходных материалах, призвано помочь в контроле материалов, формировать отчеты по определённым категориям материалов и оборудования.

Программное средство создано в среде разработки Visual Studio 2017 на языке программирования C#. Программа может работать на операционных системах Windows 7,8,8.1,10. К проекту подключена база «AccountingConsumables.mdf», которая весит 8 Мбайт, представлена двумя файлами с расширениями «.mdf» и «.ldf». Само приложение весит 51 Мбайт. Программа не требовательна к системным ресурсам.

Для успешного использования программы, на компьютере должно быть свободного места на диске не меньше 100 Мбайт, оперативная память должна быть от 200 Мбайт. Приложение использует стандартные библиотеки такие как:

- Microsoft.ReportViewer.WinForms;
- Microsoft.SqlServer.Types;
- System.Configuration;
- System.Core;
- System.Data;
- System.Data.DataSetExtensions;
- System.Windows.Forms
- System.Xaml;
- System.Drawing.

Для работы данного программного средства необходима установка платформы microsoft net framework v4.5 или выше.

Microsoft Net Framework — программная платформа, выпущенная компанией Microsoft. Основой платформы является общезыковая среда исполнения Common Language Runtime, которая подходит для разных языков программирования. Функциональные возможности CLR доступны в любых языках программирования, использующих эту среду [15].

Важное место играет занимаемый объем приложения на жестком диске. Должна быть предусмотрена совместимость программного приложения с другими программными ресурсами.

Инсталляция программы производится с установочного диска. Для входа в программу ввод логина и пароля не требуется.

4.2 Функциональное назначение

Программа предназначена для эффективного ведения учета расходных материалов и их контроля, вывода соответствующих документов на печать. Без информационной системы весь процесс учета происходит вручную, путем заполнения документов в Word и вывод их на печать.

На данный момент не предусмотрено никаких средств защиты к данным т.к. нет необходимости, данный программный продукт не осуществляет обработку конфиденциальной информации.

Назначение программного средства «AccountingConsumables» является автоматизация работы организации, которая занимается сборкой офисного оборудования, мебели. В данной программе классом решаемых задач являются:

- возможность добавлять, изменять и удалять данные в таблицы;
- возможность сортировать, фильтровать и искать данные в различных таблицах;
- возможность создавать отчёты по различным критериям.

4.3 Входные и выходные данные

Входными являются данные, вводимые пользователем в таблицы базы данных: «Категории», «Подразделения», «Оборудование», «Расходы», «Материалы», «Заказы», «Поставщики», «Типы материалов». Кроме того, входными являются данные вводимые пользователем в поля для поиска и фильтрации данных.

При добавлении новой информации в таблицу «Категории» необходимо ввести его наименование.

При добавлении новой информации в таблицу «Оборудование» необходимо ввести следующие данные:

- «Модель/имя»;
- «Кол-во»;
- «Срок службы».

При добавлении новой информации в таблицу «Расход» необходимо ввести следующие данные:

- «Расход/кол-во»;
- «Дата».

При добавлении новой информации в таблицу «Материалы» необходимо ввести следующие данные:

- «Наименование материала»;
- «Кол-во»;
- «Дата поступления»;
- «Срок службы».

При добавлении новой информации в таблицу «Категории» необходимо ввести наименование.

При добавлении новой информации в таблицу «Заказы» необходимо ввести следующие данные:

- «Дата заказа»;
- «Кол-во».

При добавлении новой информации в таблицу «Поставщики» необходимо ввести следующие данные:

- «Наименование организации»;
- «Телефон»;
- «Адрес».

При добавлении новой информации в таблицу «Подразделения» необходимо ввести наименование.

Выходная информация будет представлена в виде отчётов экспортируемые в Word и Excel.

Сформированный отчёт, содержащий информацию о расходах материалов, представлен на рисунке 4.1.

Генератор отчетов

Выборка по модели: Iri

Акты рас

Сформировать

Найти | Следующий

Акт расхода материалов №33 от 20.12.2018

Подразделение: Сборочный цех 2

Наименование продукции: Iri 0302

Материал	Кол-во	Подразделение	Расход
Болт		5 Сборочный цех 2	12
Болт		5 Сборочный цех 2	12
Дверь		65 Сборочный цех 3	16
Можак		2 Сборочный цех 1	20

Бухгалтер:

(подпись) (расшифровка)

Рисунок 4.1

Сформированный отчет, содержащий информации об имеющемся оборудовании, представлен на рисунке 4.2.

Генератор отчетов

Выборка по модели: Оборудование в наличии

Сформировать

Найти | Следующий

Оборудование в наличии

Дата отчета: 20.12.2018 4:37:15

Модель	Кол-во	Срок службы	Категории
Iri 0302	4	21.12.2021	Стол
Bar 234	5	23.05.2022	Шаф
LMP343	2	23.10.2022	Полка
Akura 2322	94	25.03.2023	Стул
Sakure	4	31.12.2026	Полка
Iri 0302	58	28.09.2025	Стеллаж
Feba 333	5	28.04.2025	Компьютерный
Akura 2322	5	26.11.2024	Компьютерный
Iri 0302	32	28.04.2025	Стул
Bar 234	4	28.09.2025	Стол
Bar 234	2	28.02.2026	Полка
Feba 333	43	31.07.2026	Полка
Bar 234	64	31.12.2026	Стеллаж
IlyaOkabe	5	31.12.2018	Стол

Бухгалтер:

Рисунок 4.2

Сформированный отчет, содержащий информации о движении расходных материалов за период/на дату, представлен на рисунке 4.3

Генератор отчетов

Выборка по модели: Движение материалов за пери... Сформи

1 из 1

Найти | Следующий

**Движение материалов материалов период/на дату
от 20.12.2018**

Материал	Кол-во	Дата прибытия	Дата расхода	Подразделение
Рейки	4	23.12.2018	31.12.2018	Сборочный цех 5
Болт	5	23.12.2018	12.12.2018	Сборочный цех 2
Болт	3	23.12.2018	12.12.2018	Сборочный цех 5
Болт	5	23.12.2018	12.12.2018	Сборочный цех 2
Дверь	65	22.10.2017	30.12.2018	Сборочный цех 3
Можак	2	15.12.2018	10.12.2018	Сборочный цех 1

Бухгалтер: (подпись) (расшифровка)

Рисунок 4.3

Сформированный отчет, содержащий информацию о сроках службы офисного оборудования/расходных материалов, представлен на рисунке 4.4

Генератор отчетов

Выборка по модели: Сроки службы оборудования Сформи

1 из 1

Найти | Следующий

Сроки службы оборудования

Модель	Количество	Срок службы
Iri 0302	4	21.12.2021
Bar 234	5	23.05.2022
LMP343	2	23.10.2022
Akura 2322	94	25.03.2023
Sakura	4	31.12.2026
Iri 0302	58	28.09.2025
Feba 333	5	28.04.2025
Akura 2322	5	26.11.2024
Iri 0302	32	28.04.2025
Bar 234	4	28.09.2025
Bar 234	2	28.02.2026
Feba 333	43	31.07.2026
Bar 234	64	31.12.2026
IlyaOkabe	5	31.12.2018

Рисунок 4.4

5 Методика испытаний

5.1 Технические требования

Минимальные системные требования персонального компьютера для оптимальной работы программного средства:

- операционная система Windows 7,8,8.1,10 64бит;
- процессор AMD Athlon / Intel Pentium 2500 МГц или совместимый аналог;
- оперативная память: 4 Гбайт
- 30 Мбайт свободного места на винчестере
- видеокарта с объемом памяти не менее 128 Мбайт.
- для удобства использования клавиатура и мышь.

Наиболее удобной операционной системой для проведения испытаний является Windows 10, так как она ориентирована на максимальное использование всех возможностей персонального компьютера, сетевых ресурсов, и обеспечение комфортных условий работы.

5.2 Функциональное тестирование

Данное тестирование проводится для выявления неполадок и недочетов программы на этапе ее сдачи в эксплуатацию.

При функциональном тестировании осуществляется проверка каждого пункта меню, каждой операции, с целью проверки выполнения всех функций, определенных на этапе объектно-ориентированного анализа и проектирования. Функциональное тестирование должно гарантировать работу всех элементов управления в автономном режиме.

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

При запуске программы открывается главная форма программного средства. Главная форма представлена на рисунке 5.1.

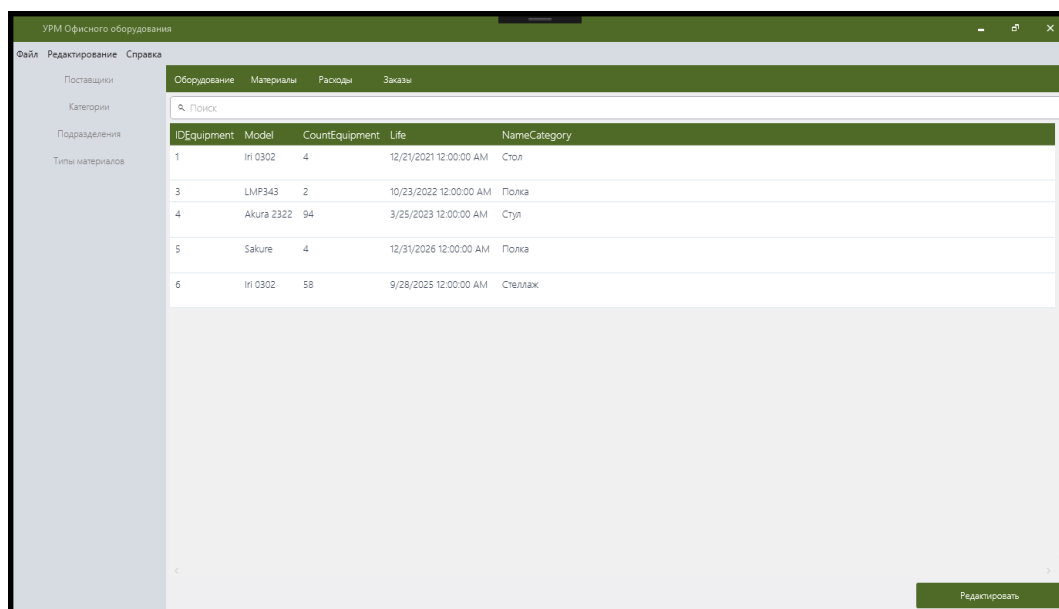


Рисунок 5.1 – Главная форма

В таблице 5.1 представлены тест-кейсы, подготовленные для проведения функционального тестирования.

Таблица 5.1 – Тест-кейсы для проведения функционального тестирования

№ тест-кейса	Модуль/Функция	Шаги воспроизведения	Ожидаемый результат	Фактический результат
001	Добавление	<ol style="list-style-type: none"> 1. На главной форме, представленной на рисунке 5.1, нажать на кнопку «Редактировать» 2. Загрузить форму «Редактирование» представленную на рисунке 5.2 3. Выбрать таблицу «Типы материалов» из выпадающего списка. 4. Заполнить соответствующие поля данными: «Наименование» - Стекло 5. Нажать на кнопку «Добавить» 	Данные занесены в таблицу «Типы материалов».	Результат соответствует ожидаемому. Все условия соблюдены. Результат занесения новых данных представлена на рисунке 5.3.
002	Удаление	<ol style="list-style-type: none"> 1. На главной форме нажать на кнопку «Редактировать» 2. Загрузить форму «Редактирование» представленную на рисунке 5.2 3. Выбрать таблицу из выпадающего списка «Тип материала». 4. Выбрать элемент, который нужно удалить выделить его в таблице как показано на рисунке 5.4. 5. Нажать на кнопку «Удалить» 	Данные удалены в таблице «Типы материалов»	Результат соответствует ожидаемому. Все условия соблюдены. Результат удаления данных представлен на рисунке 5.4.
003	Сохранение	<ol style="list-style-type: none"> 1. На главной форме нажать на кнопку «Редактировать» 2. Загрузить форму «Редактирование» 3. Выбрать таблицу из выпадающего списка, «Тип материала». 4. Выбрать элемент, который нужно изменить и выделить его в таблице. 5. Изменить данные отображенные в поле ввода на другие: Наименование материала - Кварц 6. Нажать на кнопку «Сохранить» 	Данные сохранены в таблице «Типы материалов»	Результат соответствует ожидаемому. Все условия соблюдены. Результат изменения данных представлен на рисунке 5.5
004	Сформировать отчет	<ol style="list-style-type: none"> 1. На главной форме выбрать пункт меню «Редактирование» подпункт вложенного меню «Сформировать отчет». 2. В форме «Генератор отчетов» представленной на рисунке 5.6, выбрать отчет из выпадающего списка «Отчет оборудование в наличии» 3. Нажать на кнопку «Сформировать» как показано на рисунке 5.7. 	При нажатии на кнопку создаться отчёт по выбранным шаблонам отчетов.	Результат соответствует ожидаемому. Все условия соблюдены. Результат формирования отчета представлен на рисунке 5.7

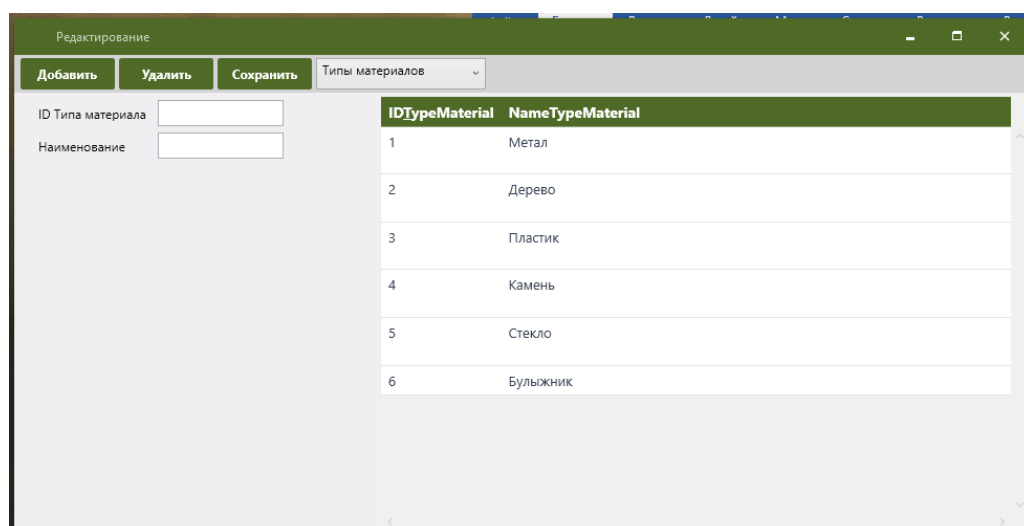


Рисунок 5.2 – Форма «Редактирование»

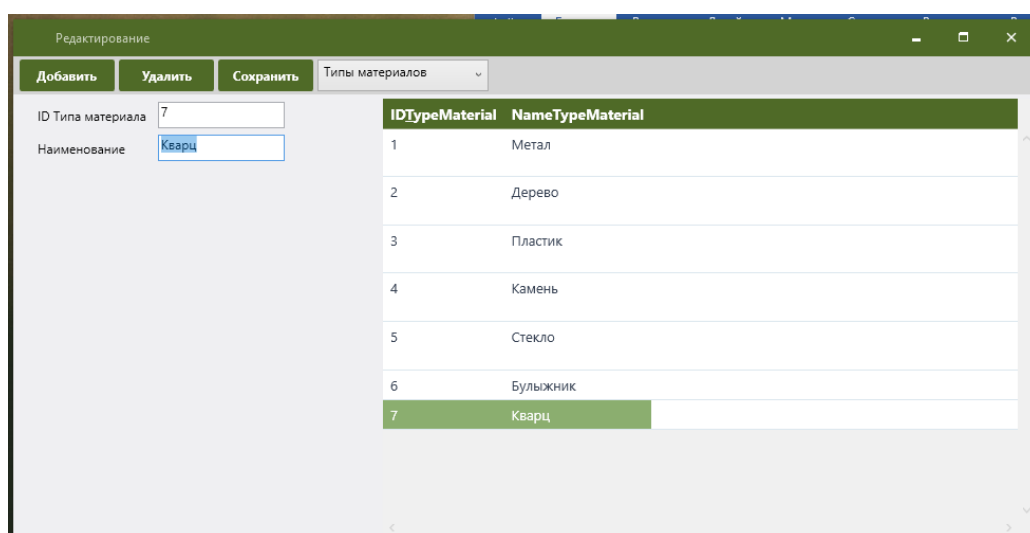


Рисунок 5.3 – Занесение новых данных

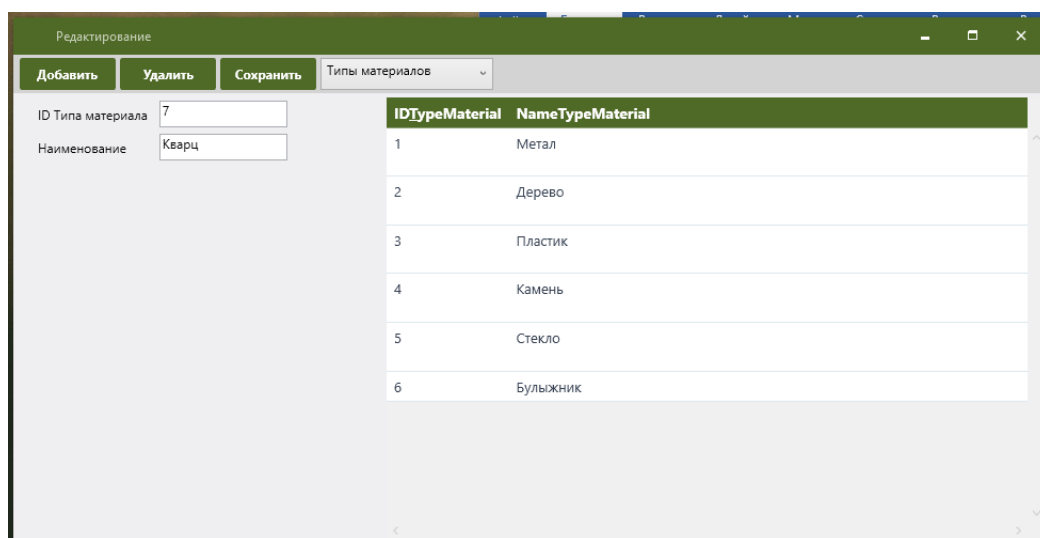


Рисунок 5.4 – Удаление данных

Редактирование

Добавить Удалить Сохранить Типы материалов

ID Типа материала: 6

Наименование: Иридий

IDTypeMaterial	NameTypeMaterial
1	Метал
2	Дерево
3	Пластик
4	Камень
5	Стекло
6	Иридий

Рисунок 5.5 – Изменение данных

Генератор отчетов

Выборка по модели: Iri Сформировать

100% Найти | Следующий

Рисунок 5.6 – Форма «Генератор отчетов»

Генератор отчетов

Выборка по модели: Iri Акт расхода материалов Сформировать

100% Найти | Следующий

Акт расхода материалов №33 от 20.12.2018

Подразделение: Сборочный цех 2

Наименование продукции: Iri 0302

Материал	Кол-во	Подразделение	Расход
Болт		5 Сборочный цех 2	12
Болт		5 Сборочный цех 2	12
Дверь		65 Сборочный цех 3	16
Можак		2 Сборочный цех 1	20

Бухгалтер: (подпись) (расшифровка)

Рисунок 5.7 – Формирование отчета

6 Применение

6.1 Назначение программы

Программа «AccountingConsumables.exe» предназначена для сотрудников компании, который занимается учетом расходных материалов офисного оборудования. В этой программе сотрудник ведёт учёт расходных материалов, а также следит за их движением в этой программе сотрудник может создавать отчёты по интересующей его информации: акт расхода материалов, отчет по остаткам, отчет об имеющемся оборудовании, отчет о расходных материалах, отчет о движении расходных материалов за период/на дату, отчет о сроке службы офисного оборудования/расходных материалов.

6.2 Условия применения

Для применения данного программного средства необходимы следующие технические требования:

- процессор Intel Core 2 Duo или выше;
- минимальный объем оперативной памяти — 100 Мбайт;
- операционная система Windows 7 и выше;
- устройство для чтения дисков;

6.3 Справочная система

Справочная система программного средства представляет собой отдельный файл «index.html» открывается по выбору подпункта меню «Просмотр справки». Просмотр справки выполняется браузером по умолчанию. В справочной системе даны ответы на типичные вопросы, возникающие при работе с приложением, что, несомненно, должно помочь при освоении программного средства.

Справка имеет следующие разделы:

- «О программе»;
- «Как пользоваться»;
- «Условия использования».

Структура справочной системы представлена на рисунке 6.1.

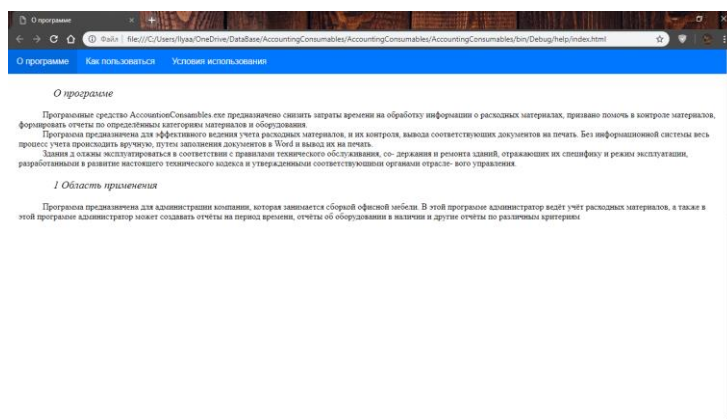


Рисунок 6.1 – Структура справочной системы

В разделе «О программе» рассказывается о приложении и о разработчике. Содержимое раздела представлено на рисунке 6.2.

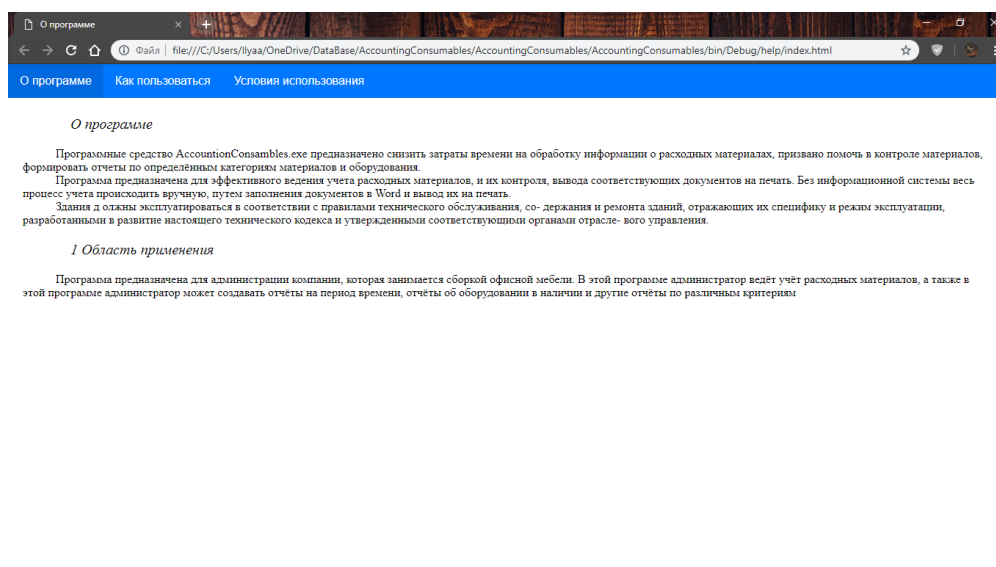


Рисунок 6.2 – Содержимое раздела «О программе»

В разделе «Как пользоваться» приведена инструкция по работе с программой. Содержимое раздела представлено на рисунке 6.3.

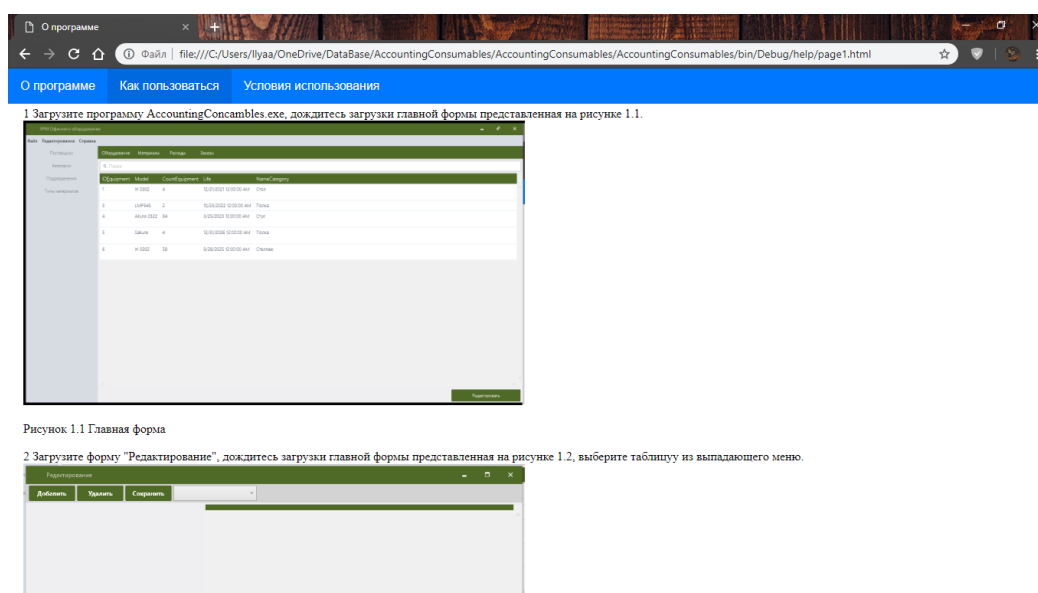


Рисунок 6.3 – Содержимое раздела «Как пользоваться»

В разделе «Условия использования» отображена информация об условиях использования данной программы, содержит информацию о наложенных ограничениях во время использования. Содержание раздела представлено на рисунке 6.4.

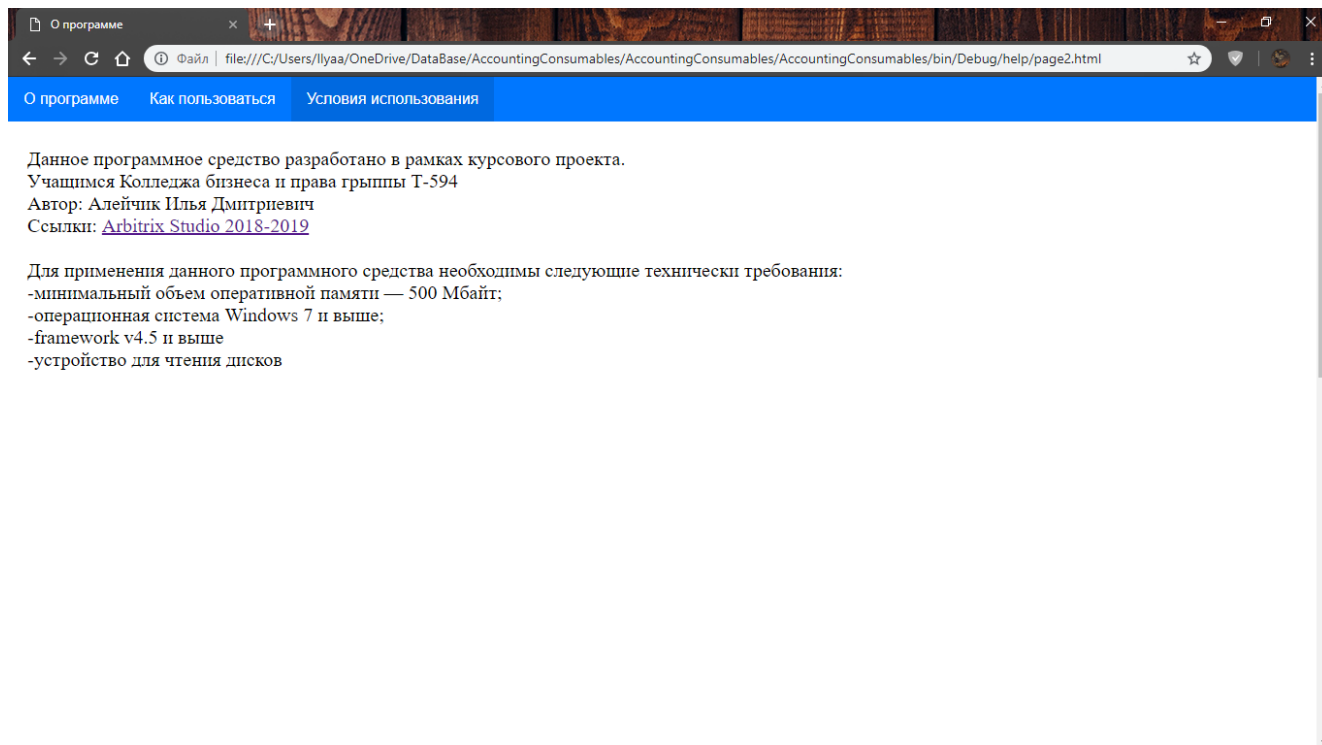


Рисунок 6.4 – Содержимое раздела «Условия использования»

Заключение

В рамках курсового проектирования по дисциплине «Базы данных и системы управления базами данных» было разработано программное средство «AccountingConsumables.exe», автоматизирующее учёт информации о расходных материалах офисного оборудования.

Для достижения цели курсового проектирования были решены следующие задачи:

- определена вычислительная система, необходимая для создания программного средства;
- разработаны логическая и физическая модели данных;
- по модели выполнена проектирование задачи;
- разработано программное средство;
- описано созданное программное средство;
- выбрана методика испытания;
- описан процесс функционального тестирования;
- приведены примеры в области применения.

Разработка имеет интуитивно графический интерфейс, позволяющий с минимальным знанием компьютера использовать данное программное средство.

В процессе разработке данного программного средства были применены и закреплены знания по уже изученному материалу, были отработаны навыки владения методами надёжного программирования и эффективности разработки программного обеспечения в Visual Studio 2017 с использованием языка программирования C#, разработана база данных средствами системы управления базами данных

В программе реализовано ведение базы данных, ее редактирование, вывод на экран, а также формирование отчетов, экспорт и вывод их на печать.

Программам готова к использованию.

Список информационных источников

- 1 Орлов, С.А. Технологии разработки программного обеспечения: Учебник для вузов. 4-е изд. / С. А. Орлов. В. Я. Цилькер СПб.: Питер, 2012. 608 с.
- 2 Тепляков, С. Паттерны проектирования на платформе ,NET. /С. Тепляков СПб.: Питер, 2015-320 с.
- 3 Общие требования к текстовым документам: ГОСТ 2.105-95. Введ. 01.01.1996 - Минск: Межгос. совет по стандартизации, метрологии и сертификации, 1995. 84 с.
- 4 Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.301. - 2000. Введ. 01.01.2001. Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000. 14 с.
- 5 Текст программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.401. - 2000. - Введ. 01.09.2001. - Минск: Межгос. совет по стандартизации, метрологии и сертификации, 2000. 16 с.
- 6 Windows 10 [электронный ресурс]. Windows 10 описание - Режим доступа: https://ru.wikipedia.org/wiki/Windows_10 - Дата доступа: 11.12.2018.
- 7 Microsoft Visual Studio [электронный ресурс]. Среда разработки. Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio Дата доступа: 1 1.12.2018.
- 8 Microsoft SQL Server [электронный ресурс]. Система управления базами данных - Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_SQL_Server - Дата доступа: 1 1.12.2018.
- 9 SQL Server Management Studio [электронный ресурс]. Среда управления SQL Server - Режим доступа: https://ru.wikipedia.org/wiki/SQL_Server_Management_Studio - Дата доступа: 11.12.2018.
- 10 Microsoft Visio [электронный ресурс]. Редактор диаграмм - Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_Visio - Дата доступа: 11.12.2018.
- 11 Microsoft Office [электронный ресурс]. Офисный пакет - Режим доступа: https://ru.wikipedia.org/wiki/Microsoft_Office - Дата доступа: 11.12.2018.
- 12 Smart Install Maker [электронный ресурс]. Сборщик инсталляторов - Режим доступа: https://ru.wikipedia.org/wiki/Smart_Install_Maker - Дата доступа: 11.12.2018.
- 13 C Sharp [электронный ресурс]. Язык программирования - Режим доступа: https://ru.wikipedia.org/wiki/C_Sharp - Дата доступа: 11.12.2018.
- 14 HTML [электронный ресурс]. Язык разметки - Режим доступа: <https://ru.wikipedia.org/wiki/HTML> - Дата доступа: 11.12.2018.
- 15 NET Framework [электронный ресурс]. Программная платформа - Режим доступа: https://ru.wikipedia.org/wiki/.NET_Framework - Дата доступа: 11.12.2018.

Приложение А
(обязательное)
Текст программы

EditWindows.cs

```
using System;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Windows;
using System.Windows.Controls;

namespace AccountingConsumables.Views
{
    /// <summary>
    /// Логика взаимодействия для EditWindow.xaml
    /// </summary>
    public partial class EditWindow : Window
    {
        MainWindow main = new MainWindow();
        SqlDataAdapter adapter;
        DataTable table;
        SqlConnection connection = null;
        SqlCommand command;
        string CatBoxSelected { get; set; }
        private string selTable { get; set; }
        public EditWindow()
        {
            InitializeComponent();
        }
        private void SetVisible()
        {
            TypeMaterialGrid.Visibility = Visibility.Hidden;
            ProvidersGrid.Visibility = Visibility.Hidden;
            MaterialsGrid.Visibility = Visibility.Hidden;
            OrdersGrid.Visibility = Visibility.Hidden;
            DepartamentGrid.Visibility = Visibility.Hidden;
            CategoryGrid.Visibility = Visibility.Hidden;
            EquipmentGrid.Visibility = Visibility.Hidden;
            ExpensesGrid.Visibility = Visibility.Hidden;
        }
        public void fillTableAndSav(string sql, UIElement nameFrom)
```

```

{

table = new DataTable();
SqlConnection connection = null;
try
{
    connection = new SqlConnection(main.connectionString);
    SqlCommand command = new SqlCommand(sql, connection);

    adapter = new SqlDataAdapter(command);
    connection.Open();
    adapter.Fill(table);
    dataGrid.ItemsSource = table.DefaultView;

}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
}
public void fillTableAndDel(string sql, UIElement nameFrom)
{

```

```

table = new DataTable();
SqlConnection connection = null;
try
{
    connection = new SqlConnection(main.connectionString);
    SqlCommand command = new SqlCommand(sql, connection);

    adapter = new SqlDataAdapter(command);
    connection.Open();
    adapter.Fill(table);
    dataGrid.ItemsSource = table.DefaultView;

```

```

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

public void fillTable(string sql, UIElement nameFrom)
{

    nameFrom.Visibility = Visibility.Visible;

    table = new DataTable();
    SqlConnection connection = null;
    try
    {
        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);

        adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

public void fillComboboxEq()
{
    SqlConnection connection = null;

```

```

try
{

    string sql = "SELECT * FROM Equipment";
    DataTable table = new DataTable();

    connection = new SqlConnection(main.connectionString);
    SqlCommand command = new SqlCommand(sql, connection);
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    connection.Open();
    adapter.Fill(table);
    boxEquip.ItemsSource = table.DefaultView;
    boxEquip.DisplayMemberPath = "Model";
    boxEquip.SelectedValuePath = "ID_Equipment";
    command.Dispose();
    connection.Close();

}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
}
public void fillComboboxDep()
{
    SqlConnection connection = null;
    try
    {

```

```

    string sql = "SELECT * FROM Departament";
    DataTable table = new DataTable();

```

```

        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        boxDep.ItemsSource = table.DefaultView;
        boxDep.DisplayMemberPath = "NameDep";
        boxDep.SelectedValuePath = "ID_Departement";
        command.Dispose();
        connection.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

public void fillComboboxProv()
{
    SqlConnection connection = null;
    try

        string sql = "SELECT * FROM Providers";
        DataTable table = new DataTable();

        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        boxProviders.ItemsSource = table.DefaultView;
        boxProviders.DisplayMemberPath = "NameOrganization";
        boxProviders.SelectedValuePath = "ID_Providers";
        command.Dispose();
        connection.Close();
    }
    catch (Exception ex)
    {

```

```

        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
public void fillComboboxMat2()
{
    SqlConnection connection = null;
    try
    {

        string sql = "SELECT * FROM Materials";
        DataTable table = new DataTable();

        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);

        boxMat.ItemsSource = table.DefaultView;
        boxMat.DisplayMemberPath = "NameMaterials";
        boxMat.SelectedValuePath = "ID_Materials";

        command.Dispose();
        connection.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
public void fillComboboxMat()
{

```

```

SqlConnection connection = null;
try
{

    string sql = "SELECT * FROM Materials";
    DataTable table = new DataTable();

    connection = new SqlConnection(main.connectionString);
    SqlCommand command = new SqlCommand(sql, connection);
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    connection.Open();
    adapter.Fill(table);
    boxMaterials.ItemsSource = table.DefaultView;
    boxMaterials.DisplayMemberPath = "NameMaterials";
    boxMaterials.SelectedValuePath = "ID_Materials";

    command.Dispose();
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
}

public void fillComboboxType()
{
    SqlConnection connection = null;
    try
    {

        string sql = "SELECT * FROM TypeMaterial";
        DataTable table = new DataTable();

        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        connection.Open();
    }
}

```

```

        adapter.Fill(table);
        txtFKIDTypeMaterial.ItemsSource = table.DefaultView;
        txtFKIDTypeMaterial.DisplayMemberPath = "NameTypeMaterial";
        txtFKIDTypeMaterial.SelectedValuePath = "ID_TypeMaterial";
        command.Dispose();
        connection.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

public void fillComboboxCategory()
{
    SqlConnection connection = null;
    try
    {

        string sql = "SELECT * FROM Category" ;
        DataTable table = new DataTable();

        connection = new SqlConnection(main.connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        txtCategory.ItemsSource = table.DefaultView;
        txtCategory.DisplayMemberPath = "NameCategory";
        txtCategory.SelectedValuePath = "ID_Category";
        command.Dispose();
        connection.Close();

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```



```

    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

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

private void CatBox_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    ComboBox comboBox = (ComboBox)sender;
    ComboBoxItem selectedItem = (ComboBoxItem)comboBox.SelectedItem;
    CatBoxSelected = selectedItem.Content.ToString();
    switch (CatBoxSelected)
    {

        case "Типы материалов":

            SetVisible();

            fillTable("SELECT * FROM TypeMaterial",TypeMaterialGrid);
//typeMaterial

            break;
        case "Поставщики":
            SetVisible();
            fillTable("SELECT * FROM Providers",ProvidersGrid);//Providers

            break;
        case "Материалы":
            fillComboboxType();
            SetVisible();
            fillTable("SELECT * FROM Materials",MaterialsGrid); //Materials

            break;
        case "Заказы":
            fillComboboxMat();
            fillComboboxProv();
            SetVisible();

```

```

        fillTable("SELECT * FROM Orders",OrdersGrid); // Orders

        break;
    case "Подразделения":
        SetVisible();
        fillTable("SELECT * FROM Departament",DepartamentGrid);
//Departament

        break;
    case "Категории":
        SetVisible();
        fillTable("SELECT * FROM Category",CategoryGrid); //Category

        break;
    case "Оборудование":
        fillComboboxCategory();
        SetVisible();
        fillTable("SELECT * FROM EquipmentView", EquipmentGrid);
//Equipment

        break;
    case "Расходы":

        fillComboboxMat2();
        fillComboboxEq();
        fillComboboxDep();
        SetVisible();
        fillTable("SELECT * FROM Expenses",ExpensesGrid); //Expenses

        break;
    }
}
private void DataGrid_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{

    DataGrid dg = sender as DataGrid;
    DataRowView dr = dg.SelectedItem as DataRowView;
    switch (CatBoxSelected)
    {
        case "Типы материалов":
            //TypeMaterial
            if (dr != null)
            {
                try

```

```

        {
            txtIdTypeMat.Text = dr["ID_TypeMaterial"].ToString();
            txtNameTypeMat.Text = dr["NameTypeMaterial"].ToString();

        }
        catch { }
    }
    break;

case "Поставщики":
    //Providers
    if (dr != null)
    {
        try
        {
            txtIDProviders.Text = dr["ID_Providers"].ToString();
            txtNameOrg.Text = dr["NameOrganization"].ToString();
            txtPhone.Text = dr["Phone"].ToString();
            txtAdressProviders.Text = dr["AddresProviders"].ToString();

        }
        catch { }
    }
    break;

case "Материалы":
    //Materials
    if (dr != null)
    {
        try
        {
            txtIDMaterials.Text = dr["ID_Materials"].ToString();
            txtNameMaterials.Text = dr["NameMaterials"].ToString();
            txtCountMaterials.Text = dr["CountMaterials"].ToString();
            txtFKIDTypeMaterial.Text = dr["NameTypeMaterial"].ToString();
            dateDateReceipt.SelectedDate =
DateTime.Parse(dr["DateReceipt"].ToString());
            dateLife.SelectedDate = DateTime.Parse(dr["Life"].ToString());

        }
        catch { }
    }

```

```

    }
    break;

case "Заказы":
    // Orders
    if (dr != null)
    {
        try
        {
            txtIdOrders.Text = dr["ID_Orders"].ToString();
            txtDateOrder.SelectedDate =
DateTime.Parse(dr["DateOrder"].ToString());

            txtCounOrdertMaterials.Text = dr["CountOrderMat"].ToString();

        }
        catch { }
    }
    break;

case "Подразделения":
    //Departament
    if (dr != null)
    {
        try
        {
            txtIDDepartament.Text = dr["ID_Departament"].ToString();
            txtNameDep.Text = dr["NameDep"].ToString();

        }
        catch { }
    }
    break;

case "Категории":
    //Category
    if (dr != null)
    {
        try
        {
            txtIDCategory.Text = dr["ID_Category"].ToString();
            txtNameCategory.Text = dr["NameCategory"].ToString();

```

```

        }
        catch { }
    }
    break;

case "Оборудование":
    //Equipmnet
    ///*fillComboBox*/();
    if (dr != null)
    {
        try
        {
            txtID_Equipment.Text = dr["ID"].ToString();
            txtModel.Text = dr["Модель"].ToString();
            txtCountEquipment.Text = dr["Кол-во"].ToString();
            txtLife.SelectedDate = DateTime.Parse(dr["Срок
службы"].ToString());

        }
        catch { }
    }
    break;

case "Расходы":
    //Expenses
    if (dr != null)
    {
        try
        {
            txtIDExpenses.Text = dr["ID_Expenses"].ToString();
            txtInvolved.Text = dr["Involved"].ToString();

            txtDateExpensees.SelectedDate =
DateTime.Parse(dr["DateExpensees"].ToString());

        }
        catch { }
    }
    break;
}

```

```

    }
    private void Add_btn_Click(object sender, RoutedEventArgs e)
    {
        string query;
        table = new DataTable();
        switch (CatBoxSelected)
        {

            case "Типы материалов":

                //DataRowView dr = dg.SelectedItem as DataRowView;
                //TypeMaterial

                try
                {
                    connection = new SqlConnection(main.connectionString);
                    connection.Open();
                    query = "INSERT INTO TypeMaterial(NameTypeMaterial) Values('" +
txtNameTypeMat.Text + "')";
                    SqlCommand command = new SqlCommand(query, connection);
                    command.ExecuteNonQuery();
                    fillTable("SELECT * FROM TypeMaterial", TypeMaterialGrid);
                    connection.Close();
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                    if (connection != null)
                        connection.Close();
                }

                break;

            case "Поставщики":
                //Providers

                try
                {
                    connection = new SqlConnection(main.connectionString);
                    connection.Open();

```

```

        query = "INSERT INTO
Providers(NameOrganization,Phone,AddresProviders) Values('" + txtNameOrg.Text
+ "','" + txtPhone.Text + "','" + txtAdressProviders.Text + "')";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Providers", ProvidersGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Материалы":
    //Materials

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "INSERT INTO
Materials(NameMaterials,CountMaterials,DateReceipt,Life,ID_TypeMaterial) Values('"
+ txtNameMaterials.Text + "','" + txtCountMaterials.Text + "','" + dateDateReceipt.Text
+ "','" + dateLife.Text + "','" + txtFKIDTypeMaterial.SelectedValue.ToString() + "')";
        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Materials", MaterialsGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

```

```

case "Заказы":
    // Orders

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "INSERT INTO
Orders(DateOrder,CountOrderMat,ID_Materials,ID_Providers) Values('" +
txtDateOrder.Text + "','" + txtCounOrdertMaterials.Text + "','" +
boxMaterials.SelectedValue.ToString() + "','" + boxProviders.SelectedValue.ToString()
+ "')";

        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Orders", OrdersGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Подразделения":
    //Departament

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "INSERT INTO Departament(NameDep) Values('" +
txtNameDep.Text + "')";

        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Departament", DepartamentGrid);
        connection.Close();
    }
    catch (Exception ex)
    {

```



```

        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Категории":
    //Category

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "INSERT INTO Category(NameCategory) Values('" +
txtNameCategory + "')";
        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Category", CategoryGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Оборудование":
    //Equipment

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "INSERT INTO
Equipment(Model,CountEquipment,Life,ID_Category) Values( '" + txtModel.Text +
"', " + txtCountEquipment.Text + ", " + txtLife.Text.ToString() + ", " +
txtCategory.SelectedValue.ToString() + ")";

```

```

        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM EquipmentView", EquipmentGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Расходы":
    //Expenses

    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        query = "insert into
Expenses(DateExpensees,Involved,ID_Equipment,ID_Materials,ID_Departament)
values('" + txtDateExpensees.Text + "','" + txtInvolved.Text + "','" +
boxEquip.SelectedValue.ToString()+"','"+ boxMat.SelectedValue.ToString()
+"','"+boxDep.SelectedValue.ToString()+"")";
        command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTable("SELECT * FROM Expenses", ExpensesGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;
}

```

```

}

private void delete_btn_Click(object sender, RoutedEventArgs e)
{
    switch (CatBoxSelected)
    {
        case "Типы материалов":
            //DataRowView dr = dg.SelectedItem as DataRowView;
            //TypeMaterial

            try
            {
                connection = new SqlConnection(main.connectionString);
                connection.Open();
                string query = "delete from TypeMaterial where ID_TypeMaterial = " +
txtIdTypeMat.Text;
                SqlCommand command = new SqlCommand(query, connection);
                command.ExecuteNonQuery();
                fillTableAndDel("SELECT * FROM TypeMaterial", TypeMaterialGrid);
                connection.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
                if (connection != null)
                    connection.Close();
            }

            break;

        case "Поставщики":
            //Providers
            try
            {
                connection = new SqlConnection(main.connectionString);
                connection.Open();
                string query = "delete from Providers where ID_Providers = " +
txtIDProviders.Text;

```

```

        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndDel("SELECT * FROM Providers", ProvidersGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Материалы":
    //Materials
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        string query = "delete from Materials where ID_Materials = " +
txtIDMaterials.Text;
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndDel("SELECT * FROM Materials", MaterialsGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Заказы":
    // Orders
    try
    {
        connection = new SqlConnection(main.connectionString);

```

```

        connection.Open();
        string query = "delete from Orders where ID_Orders = " +
txtIdOrders.Text;
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndDel("SELECT * FROM Orders", OrdersGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Подразделения":
    //Department
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();
        string query = "delete from Department where ID_Department = " +
txtIDDepartment.Text;
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndDel("SELECT * FROM Department", TypeMaterialGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Категории":
    //Category

```

```

try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();
    string query = "delete from Category where ID_Category = " +
txtIDCategory.Text;
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTableAndDel("SELECT * FROM Category", TypeMaterialGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
break;

case "Оборудование":
//Equipmnet
try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();
    string query = "delete from Equipment where ID_Equipment = " +
txtID_Equipment.Text;
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTableAndDel("SELECT * FROM EquipmentView", EquipmentGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
break;

```

```

        case "Расходы":
            //Expenses
            try
            {
                connection = new SqlConnection(main.connectionString);
                connection.Open();
                string query = "delete from Expenses where ID_Expenses = " +
txtIDProviders.Text;
                SqlCommand command = new SqlCommand(query, connection);
                command.ExecuteNonQuery();
                fillTableAndDel("SELECT * FROM ExpensesView", ExpensesGrid);
                connection.Close();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
                if (connection != null)
                    connection.Close();
            }
            break;
    }
}

```

```

private void save_btn_Click(object sender, RoutedEventArgs e)
{
    SqlConnection connection = null;

    switch (CatBoxSelected)
    {

        case "Типы материалов":
            //DataRowView dr = dg.SelectedItem as DataRowView;
            //TypeMaterial

```

```

try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();
    string query = "update TypeMaterial set NameTypeMaterial = '" +
txtNameTypeMat.Text + "'" + "where ID_TypeMaterial = '" + txtIdTypeMat.Text + "'";
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTable("SELECT * FROM TypeMaterial", TypeMaterialGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}

```

break;

case "Поставщики":
//Providers

```

try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();

    string query = "update Providers set NameOrganization =
"+txtNameOrg.Text+"', Phone = '"+txtPhone.Text+"',AddresProviders =
"+txtAdressProviders.Text+" where ID_Providers = '"+txtIDProviders.Text+"'";
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTableAndSav("SELECT * FROM Providers", ProvidersGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally

```



```

    {
        if (connection != null)
            connection.Close();
    }

    break;

case "Материалы":
    //Materials
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();

        string query = "update Materials set NameMaterials = '"+
txtNameMaterials.Text
+"',CountMaterials='"+txtCountMaterials.Text+"',DateReceipt='"+dateDateReceipt.Text
t+"',Life='"+dateLife.Text+"',
ID_TypeMaterial='"+txtFKIDTypeMaterial.SelectedValue.ToString()+"' where
ID_Materials = '"+txtIDMaterials.Text+"'";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndSav("SELECT * FROM Materials", MaterialsGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Заказы":
    // Orders
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();

        string query = "update Orders set DateOrder = '"+txtDateOrder.Text+"',
CountOrderMat='"+txtCounOrdertMaterials.Text+"',

```

```

ID_Materials="" + boxMaterials.SelectedValue.ToString() + "",
ID_Providers="" + boxProviders.SelectedValue.ToString() + "" where ID_Orders =
"" + txtIdOrders.Text + "";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndSav("SELECT * FROM Orders", OrdersGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Подразделения":
    //Departament
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();

        string query = " update Departament set NameDep =
"" + txtNameDep.Text + "" where ID_Departament = "" + txtIDDepartament.Text + "";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndSav("SELECT * FROM Departament", DepartamentGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;

case "Категории":

```

```

//Category
try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();

    string query = " update Category set NameCategory =
"+txtNameCategory.Text+" where ID_Category = "+txtIDCategory.Text+"";
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTableAndSav("SELECT * FROM Category", CategoryGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
break;

case "Оборудование":
//Equipmnet
try
{
    connection = new SqlConnection(main.connectionString);
    connection.Open();

    string query = " update Equipment set Model =
"+txtModel.Text+",CountEquipment="+txtCountEquipment.Text+",Life="+txtLife.T
ext+",ID_Category="+txtCategory.SelectedValue.ToString()+" where ID_Equipment
="+txtID_Equipment.Text+"";
    SqlCommand command = new SqlCommand(query, connection);
    command.ExecuteNonQuery();
    fillTableAndSav("SELECT * FROM Equipment", EquipmentGrid);
    connection.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally

```

```

        {
            if (connection != null)
                connection.Close();
        }
        break;

case "Расходы":
    //Expenses
    try
    {
        connection = new SqlConnection(main.connectionString);
        connection.Open();

        string query = " update Expenses set DateExpensees =
"+txtDateExpensees.Text+"',Involved='"+txtInvolved.Text+"',ID_Equipment='"+boxE
quip.SelectedValue.ToString()+"',ID_Materials='"+boxMat.SelectedValue.ToString()+"
',ID_Departament='"+boxDep.SelectedValue.ToString()+"' where ID_Expenses =
"+txtIDExpenses.Text+""";
        SqlCommand command = new SqlCommand(query, connection);
        command.ExecuteNonQuery();
        fillTableAndSav("SELECT * FROM Expenses", ExpensesGrid);
        connection.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
    break;
    }
}
}
}
}

```

MainWindows.cs.

```

using System;
using System.Windows;
using AccountingConsumables.Views;
using System.IO;

```

```

using System.Data.SqlClient;
using System.Data;
using System.Configuration;

namespace AccountingConsumables
{
    /// <summary>
    /// Логика взаимодействия для MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {

        public MainWindow()
        {

            InitializeComponent();

        }
        public string connectionString { get; set; } =
ConfigurationManager.ConnectionStrings["AccountingConsumables"].ConnectionStrin
g;

        SqlDataAdapter adapter;
        DataTable table;
        SqlConnection connection = null;
        SqlCommand cmd;

        //Создание БД AccountiongConsambles в случае ее отсутствия (демо).
        public void creatDataBase()
        {
            string connectionStringMaster =
ConfigurationManager.ConnectionStrings["Master"].ConnectionString;
            string sql = " create database AccountingConsumables ";
            try
            {
                connection = new SqlConnection(connectionStringMaster);
                SqlCommand command = new SqlCommand(sql, connection);
                connection.Open();
                command.ExecuteNonQuery();
                connection.Close();
            }
        }
    }
}

```

```

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

```

```

//Выполнение действий при загрузке программы
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    try
    {
        table = new DataTable();
        try
        {
            string sql = "SELECT * FROM TypeMaterial";
            connection = new SqlConnection(connectionString);
            SqlCommand command = new SqlCommand(sql, connection);
            adapter = new SqlDataAdapter(command);
            connection.Open();
            adapter.Fill(table);
            dataGrid.ItemsSource = table.DefaultView;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message + " \nПодождите программа выполняет
создание новой базы данных");
            creatDataBase();
        }
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
//Clock
var timer = new System.Windows.Threading.DispatcherTimer();
timer.Interval = new TimeSpan(0, 0, 1);
timer.IsEnabled = true;

```

```

        timer.Tick += (o, t) => { lableTime.Content =
DateTime.Now.ToLongTimeString(); };
        timer.Start();
    }
    catch (Exception)
    {
        throw;
    }
}

//Принудительное завершение работы программы
private void Window_Closing(object sender,
System.ComponentModel.CancelEventArgs e)
{
    Application.Current.Shutdown();
}

//Настройки программы
private void Settings_Click(object sender, RoutedEventArgs e)
{
    if (SettingsGrid.Visibility == Visibility.Collapsed)
    {
        SettingsGrid.Visibility = Visibility.Visible;
    }
    else if (SettingsGrid.Visibility == Visibility.Visible)
    {
        SettingsGrid.Visibility = Visibility.Collapsed;
    }
}

//Генератор отчетов
private void MenuItem_Click(object sender, RoutedEventArgs e)
{
    WindowReportViewer rv = new WindowReportViewer();
    rv.Show();
}

//Таблицы
private void TypeMaterial_Click(object sender, RoutedEventArgs e)

```

```

{
    string sql = "SELECT ID_TypeMaterial as ID,NameTypeMaterial as
[Наименования типов] FROM TypeMaterial";
    table = new DataTable();
    try
    {
        connection = new SqlConnection(connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}
private void Eqpment_Click(object sender, RoutedEventArgs e)
{
    string sql = "Select * from EquipmentView";
    table = new DataTable();
    try
    {
        connection = new SqlConnection(connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally

```



```

        {
            if (connection != null)
                connection.Close();
        }

    }

    private void Providers_Click(object sender, RoutedEventArgs e)
    {
        string sql = "SELECT ID_Providers as ID, NameOrganization as
Организация, Phone as Телефон, AddressProviders as Адрес FROM Providers";
        table = new DataTable();
        try
        {
            connection = new SqlConnection(connectionString);
            SqlCommand command = new SqlCommand(sql, connection);
            adapter = new SqlDataAdapter(command);
            connection.Open();
            adapter.Fill(table);
            dataGrid.ItemsSource = table.DefaultView;

        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            {
                if (connection != null)
                    connection.Close();
            }
        }
    }

    private void Category_Click(object sender, RoutedEventArgs e)
    {
        string sql = "SELECT ID_Category as ID, NameCategory as [Наименования
категорий] FROM Category";
        table = new DataTable();
        try
        {
            connection = new SqlConnection(connectionString);
            SqlCommand command = new SqlCommand(sql, connection);
            adapter = new SqlDataAdapter(command);
            connection.Open();
            adapter.Fill(table);
            dataGrid.ItemsSource = table.DefaultView;
        }
    }

```

```

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

private void Materials_Click(object sender, RoutedEventArgs e)
{
    string sql = "Select * from MaterialsView";
    table = new DataTable();
    try
    {
        connection = new SqlConnection(connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

private void Departament_Click(object sender, RoutedEventArgs e)
{
    string sql = "SELECT ID_Departament as ID, NameDep as [Наименования подразделений] FROM Departament";
    table = new DataTable();
    try
    {
        connection = new SqlConnection(connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(command);

```

```

        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

private void Expenses_Click(object sender, RoutedEventArgs e)
{
    string sql = " Select * from ExpensesView ";
    table = new DataTable();
    SqlConnection connection = null;
    try
    {
        connection = new SqlConnection(connectionString);
        SqlCommand command = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(command);
        connection.Open();
        adapter.Fill(table);
        dataGrid.ItemsSource = table.DefaultView;

    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    finally
    {
        if (connection != null)
            connection.Close();
    }
}

private void Orders_Click(object sender, RoutedEventArgs e)
{
    string sql = " Select * from OrdersView";
    table = new DataTable();
    SqlConnection connection = null;

```

```

try
{
    connection = new SqlConnection(connectionString);
    SqlCommand command = new SqlCommand(sql, connection);
    adapter = new SqlDataAdapter(command);
    connection.Open();
    adapter.Fill(table);
    dataGrid.ItemsSource = table.DefaultView;

}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
finally
{
    if (connection != null)
        connection.Close();
}
}

//Кнопка открывающая окно "Редактирование"
private void Button_Click(object sender, RoutedEventArgs e)
{
    EditWindow edit = new EditWindow();
    edit.Show();
}

//О программе
private void AboutWindowClick(object sender, RoutedEventArgs e)
{
    AboutWindow aw = new AboutWindow();
    aw.ShowDialog();
}

//Выход из программы
private void ExitFromApp(object sender, RoutedEventArgs e)
{
    Close()
}

//Справка
private void ViewHelp(object sender, RoutedEventArgs e)
{
    string curDir = Directory.GetCurrentDirectory();
    var uri = new Uri(System.IO.Path.Combine(curDir, "help/index.html"));
    System.Diagnostics.Process.Start(uri.ToString());
}

public DataTable getDataFill(string adress)
{

```

```

DataTable dt = new DataTable();
connection = new SqlConnection(connectionString);
try
{
    if(connection.State.ToString() == "Closed")
    {
        connection.Open();
    }
    string sql = string.Empty;

    //if( sql == "")
    //{
        sql = "Select * from " + adress;
    //}
    //else
    //{
        //  sql = "Select * from ExpensesView where [Наименование
продукции/Модель] Like "" + adress + "%"";
    //}

        cmd = new SqlCommand(sql, connection);
        adapter = new SqlDataAdapter(cmd);
        adapter.Fill(dt);
        return dt;
    }
catch(Exception e)
{
    MessageBox.Show("Error" + e.Message);
    return dt;
}
}
}

```