МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФГБОУ ВО ПСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КОЛЛЕДЖ ПСКОВГУ

КУРСОВОЙ ПРОЕКТ

по МДК 03.02. Инструментальные средства разработки программного обеспечения

Тема: «Разработка программного обеспечения автоматизированной информационной системы «Кузница»

по специальности

09.02.03 Программирование в компьютерных системах

Обучающегося группы 437-ПО

Васильева И.Н.

Проверил(а): Луканов С.Ю.

Оценка\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_

Псков

2021

Содержание

Введение 3

1. Теоретическая часть 4

1.1. CASE-средства сопровождения и реинженеринга 4

2. Техническое задание 14

3. Практическая часть 18

3.1. Постановка задач 18

3.2. Математическое теоретическое обоснование 18

3.3. Схема данных 19

3.4. Концептуальная схема базы данных 20

3.5. Обоснования выбора инструментальных средств 22

3.6. Выполнение задачи на основе выбранного инструментального средства 23

4. Экспериментальное тестирование 30

5. Эксплуатационные документы 31

5.1. Руководства системного администратора 31

5.2. Руководство программиста 33

5.3. Руководство оператора 36

5.4. Руководство к использованию 37

6.Описание языка 39

Заключение 41

Список источников информации 42

## Введение

Целью данного курсового проекта является применение теоретических знаний об инструментальных средствах разработки программного обеспечения на практике для создания автоматизированной информационной системы.

Также целью является: научится использовать средства построения диаграмм и графиков, необходимых в программной документации к проекту. Таких как: Erwin data modeler, MS Office Visio и Modeler Navigator.

Для выполнения данной разработки необходимо реализовать автоматизированную информационную систему «Кузница». С этой целью необходимо решить следующие задачи:

Создать логическую структуру;

Выбрать инструментальные средства;

Выполнить разработку;

Разработать интерфейс пользователя;

Протестировать работу программы;

Написание программной документации;

В данном задании необходимо в полной мере реализовать знания и навыки, касающиеся инструментальных средств разработки программного обеспечения, создания программной документации, создания моделей данных, стадий процесса разработки программного продукта и стадий жизненного цикла.

# Теоретическая часть

**Обфускатор и методы обфускации**

Обфускатор — это эффективный, вероятностный компилятор, который в качестве входа имеет программу Р и производит новую программу 0(Р), которая имеет те же функциональные возможности, что и Р, но в то же время является неясной (непонятной) для противника (наблюдателя, постороннего лица) в некотором заранее определенном смысле.

Как известно, одним из основных методов взлома программного обеспечения является исследование кода, полученного в результате работы [дизассемблера](https://en.wikipedia.org/wiki/Disassembler) на предмет уязвимостей. На основе такого когда нетрудно, например, составить программу генерации ключей активации коммерческого программного обеспечения или, наоборот, внести в исполняемый файл изменение - *патч*, позволяющий злоумышленникам отключить "нежелательные" модули исходной программы.

Всему вышеперечисленному как раз и может противодействовать специальная программа - *обфускатор*.

Так же, алгоритмы обфускации активно используются не только для затруднения анализа кода, но и для уменьшения размера программного кода, что, в свою очередь, активно используется при разработке различных веб-сервисов и баз данных.

Как понятно из вышесказанного, методы обфускации должны усложнить код, преобразовав его таким образом, чтобы скрыть от третьих лиц логику его работы.

В идеале хотелось бы, чтобы программа, прошедшая обфускацию, давала бы не больше информации нежели [чёрный ящик](https://en.wikipedia.org/wiki/Black_box), имитирующий поведение исходной программы. Гипотетический алгоритм, реализующий такое преобразование называется ["Обфускация чёрного ящика"](https://en.wikipedia.org/wiki/Black-box_obfuscation). Декомпиляция зашифрованной таким образом программы дала бы злоумышленникам не больше информации, чем декомпиляция клиента мессенджера, представляющего собой лишь обёртку над апи "настоящего" приложения, что бы полностью решило поставленную в предыдущем блоке проблему. Однако показано, что реализация такого алгоритма для произвольной программы невозможна.

Большинство методов обфускации преобразуют следующие аспектов кода:

* + **Данные:** делают элементы кода похожими на то, чем они не являются
  + **Поток кода:** выставляют исполняемую логику программы абсурдной или даже недетерминированной
  + **Структура формата:** применяют различное форматирование данных, переименование идентификаторов, удаление комментариев кода и т.д.

Инструменты обфускации могут работать как с source или байт кодом, так и с бинарным, однако обфускация двоичных файлов сложнее, и должна варьироваться в зависимости от архитектуры системы.

При обфускации кода, важно правильно оценить, какие части когда можно эффективно запутать. Следует избегать обфускации кода, критичного относительно производительности.

Методы

1. Преобразование данных

Одним из наиболее важных элементов обфускации является преобразование данных, используемых программой, в иную форму,оказывающее минимальное виляние на производительность кода, но значительно усложняющее хакерам возможность [обратного инжинирнга](https://en.wikipedia.org/wiki/Reverse_engineering).

По ссылке можно ознакомится с интересными [примерами](https://www.sciencedirect.com/science/article/pii/S1877050915032780) использования двоичной формы записи чисел для усложнения читабельности кода, а так же изменений формы хранения данных и замены значений различными тождественными им выражениями.

2. Обфускация потока управления кодом

Обфускация потока управления может быть выполнена путем изменения порядка операторов выполнения программы. Изменение графа управления путем вставки произвольных инструкций перехода и преобразования древовидных условных конструкций в плоские операторы переключения, как показано на следующей диаграмме.

3. Обфускация адресов

Данный метод изменяет структура хранения данных[,](https://translate.googleusercontent.com/translate_c?depth=1&hl=ru&prev=search&pto=aue&rurl=translate.google.com&sl=en&sp=nmt4&u=http://home.eng.iastate.edu/~daji/seminar/papers/BVS03.UsenixSec.pdf&usg=ALkJrhhkAYSVzyfQTOYeodj4UvK61OFRZQ) так чтобы усложнить их использование. Например алгоритм, может выбирать случайными адреса данных в памяти, а также относительные расстояния между различными элементами данных. Данный подход примечателен тем, что даже если злоумышленник и сможет "декодировать" данные, используемые приложением на каком-то конкретном устройстве, то на других устройствах он всё равно не сможет воспроизвести свой успех.

4. Регулярное обновление кода

Этот метод предотвращает атаки, регулярно выпуская обновления обфусцированного программного обеспечения. Своевременные замены частей существующего программного обеспечения новыми обфусцированными экземплярами, могут вынудить злоумышленника отказаться от существующего результата обратного анализа, так как усилия по взлому кода в таком случае могут превысить получаемую от этого ценность.

5. Обфускация инструкций ассемблера

Преобразование и изменение ассемблерного когда также может затруднить процесс обратного инжиниринга. Одним из таких методов является использование перекрывающихся инструкций ([jump-in-a-middle](https://reverseengineering.stackexchange.com/questions/20587/jump-in-the-middle-trick)), в результате чего дизассемблер может произвести неправильный вывод. Ассемблерный код также может быть усилен против проникновения за счёт включения бесполезных управляющих операторов и прочего мусорного кода.

6. Обфускация отладочной информации

Отладочную информацию можно использовать для обратного проектирования программы, поэтому важно блокировать несанкционированный доступ к данным отладки. Инструменты обфускации достигают этого, изменяя номера строк и имена файлов в отладочных данных или полностью удаляя из программы отладочную информацию.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФГБОУ ВО ПСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КОЛЛЕДЖ ПСКОВГУ

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Тема: «Разработка программного обеспечения автоматизированной информационной системы «Кузница»

Выполнил обучающийся группы 437-ПО

Зуев Д.А.

Утвердил(а): Луканов С.Ю.

Псков

2021

## Техническое задание

**Введение**

Цель работы – обеспечение автоматизированной информационной системы «Кузница». Программа предназначена для создания, управления содержимым базы данных, содержащим следующие данные:

1. Сведения о ассортименте (оружие, ворота, скульптуры, броня средневековая и броня эпохи возрождение)
2. Сведения о стилях (готика, фэнтэзи, историческое)
3. Сведения о сталях (дамасская, мозаичная, тигельная)
4. Сведения о заказах (индивидуальные, по каталогу)
5. **Основания для разработки**

Основанием для разработки является задание на выполнение курсового проекта. Организация, утвердившая задание на разработку программного продукта: Колледж ПсковГУ.

Наименование работы: Автоматизированная информационная система «Кузница»

1. **Назначение разработки**

Автоматизированная информационная система «Кузница» предназначена для обобщения информации о клиентах, которые решили выбрать компанию «TheForgingDwarf» для выковки какого-либо изделия, из какой-либо стали, в каком-либо стиле, в определенное время за определенную стоимость.

В базе данных будет сохраняться информация о клиенте, стилях, сталях, ассортименте и заказах, что значительно облегчит работу сотрудников компании.

1. **Требования к программе**
   1. **Требования к функциональным характеристикам.**

Автоматизированная система «Кузница» должна обеспечивать выполнение функций:

- История заказов клиентов, наименование предмета, сталь, стиль, цену

- формирование отчетов, необходимых менеджеру и бухгалтеру, содержащих все данные о клиентах и заказах.

Нормативно справочная информация автоматизированной информационной системы «Кузница» представлена справочниками контрагентов, номенклатуры.

Документы для учета поездки:

- документы, содержащие дату заказа, стиль, сталь, наименование предмета.

Выходными данными являются следующие виды отчетов:

- отчеты о клиенте, заказе, дате и времени заказа, наименовании предмета, стиля и стали.

* 1. **Требования к надежности**

Требования к обеспечению надежного функционирования программы

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

а) организацией бесперебойного питания технических средств;

б) использованием лицензионного программного обеспечения;

в) регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;

г) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

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

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

В состав технических средств должен входить IВМ-совместимый персональный компьютер (ПЭВМ), выполняющий роль сервера, включающий в себя:

процессор Pentium-2.0Hz, не менее;

оперативную память объемом, 1Гигабайт, не менее;

HDD, 5 Гигабайт, не менее;

операционную систему Windows 10;

.net Framework 4.6.1

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

Программа должна работать в операционных системах Windows 10.

* 1. **Требования к транспортированию и хранению**

Программа поставляется в электронном виде. Программная документация поставляется в электронном виде.

* 1. **Специальные требования**

Программное обеспечение должно иметь дружественный интерфейс, рассчитанный на пользователя средней квалификации.

1. **Требования к программной документации**

В ходе разработки программы должны быть подготовлены текст программы**,** описание программы.

## Практическая часть

## Постановка задач

Разработать программное обеспечение автоматизированной информационной системы «Кузница». База данных должна содержать все сведения о клиентах и заказах. В ней должны быть данные о товарах (название), сведения о стилях (готика, фэнтэзи, историческое), сталях (дамасская, мозаичная, тигельная) и заказах (индивидуальные, по каталогу). Создать экранные формы для ввода и редактирования данных в таблицах и все необходимые выходные отчеты.

# Архитектура

Архитектурой информационной системы называется концепция, согласно которой взаимодействуют компоненты информационной системы. Архитектура программного обеспечения (англ. software architecture) — совокупность важнейших решений об организации программной системы. Архитектура включает: выбор структурных элементов и их интерфейсов, с помощью которых составлена система, а также их поведения в рамках сотрудничества структурных элементов; соединение выбранных элементов структуры и поведения во всё более крупные системы; архитектурный стиль, который направляет всю организацию — все элементы, их интерфейсы, их сотрудничество и их соединение (Рисунок 1).

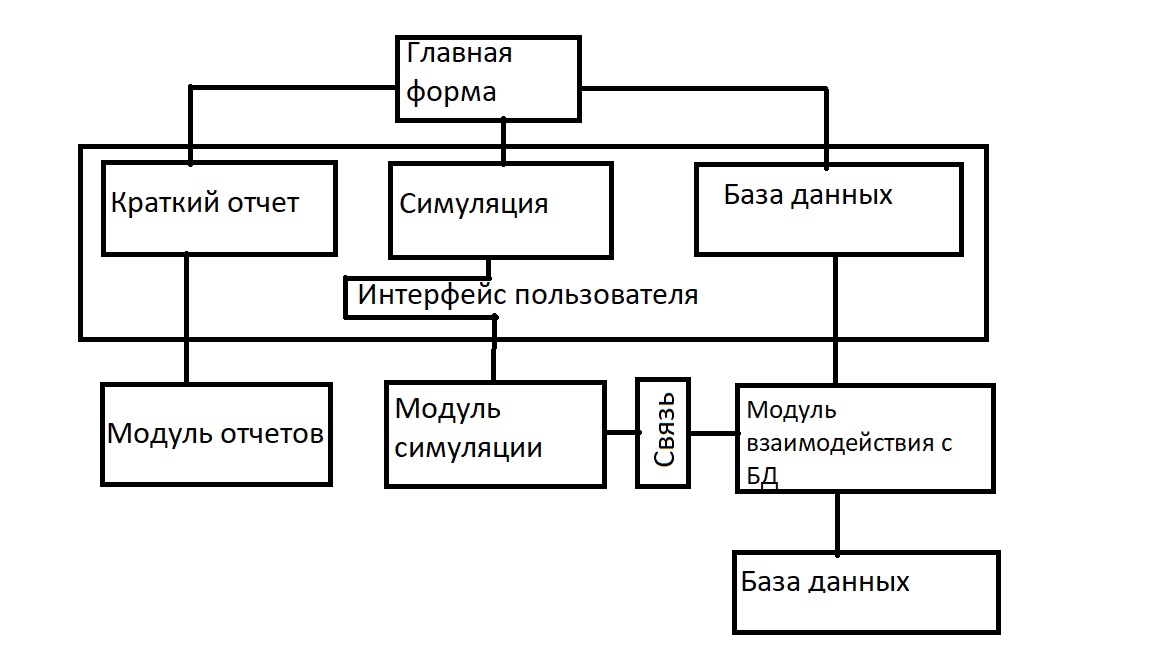


Рисунок 1. Архитектура ИС.

# Диаграмма классов

Диаграмма классов – это UML-диаграмма, которая описывает систему, визуализируя различные типы объектов внутри системы и виды статических связей, которые существуют между ними. Он также иллюстрирует операции и атрибуты классов.

Обычно они используются для изучения концепций области, понимания требований к программному обеспечению и описания подробных проектов (Рисунок 2).

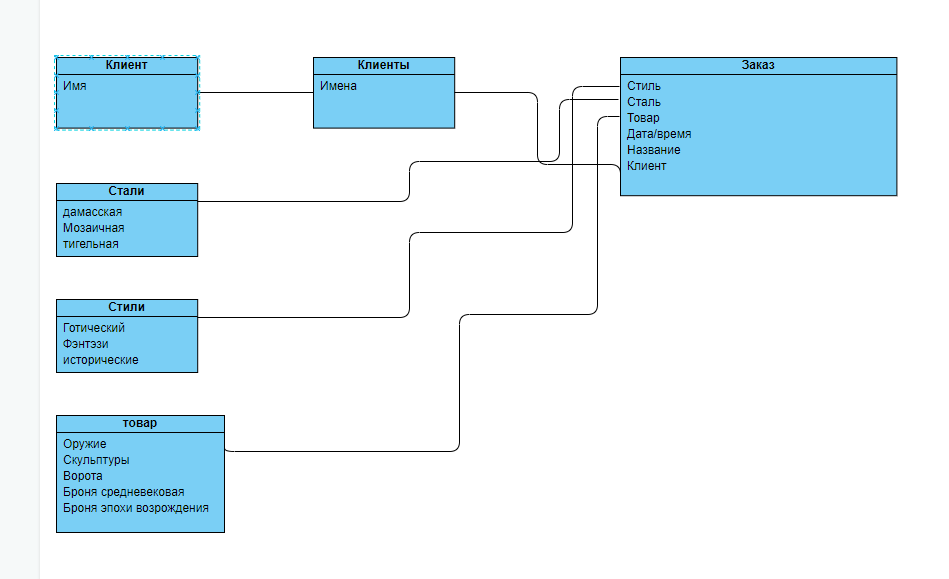


Рисунок 2. Диаграмма классов.

# Взаимодействие с БД

Взаимодействие с БД можно представить в виде диаграммы DFD (Рисунок 3) и диаграммы IDEF0 (Рисунок 4):

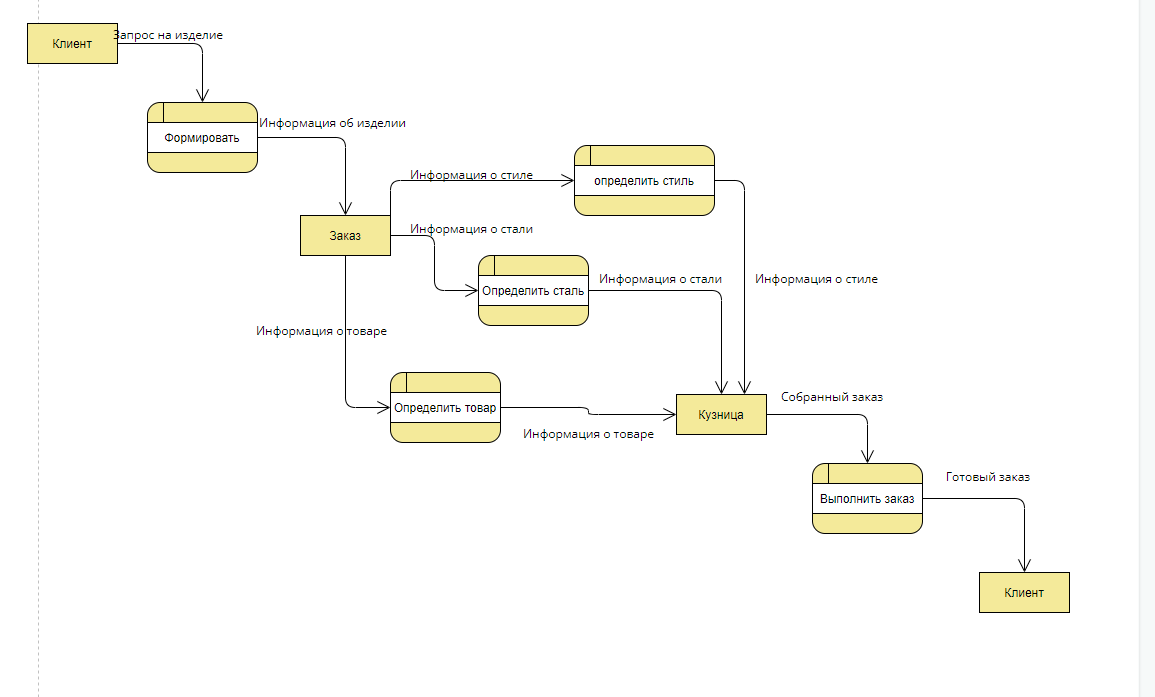


Рисунок 3. Схема DFD.

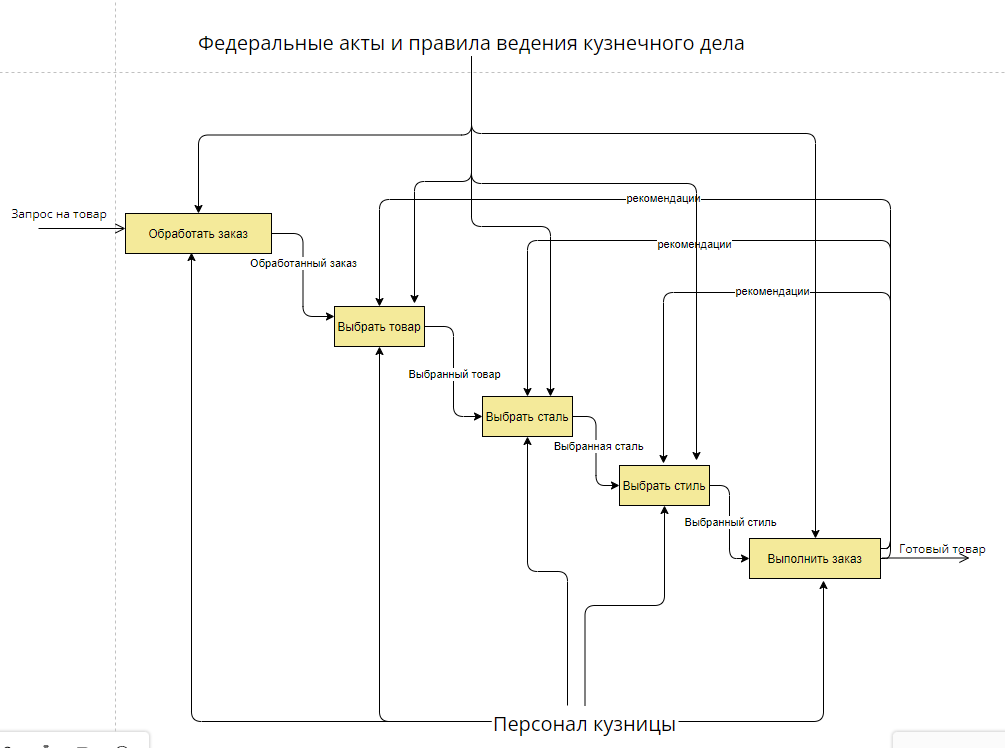
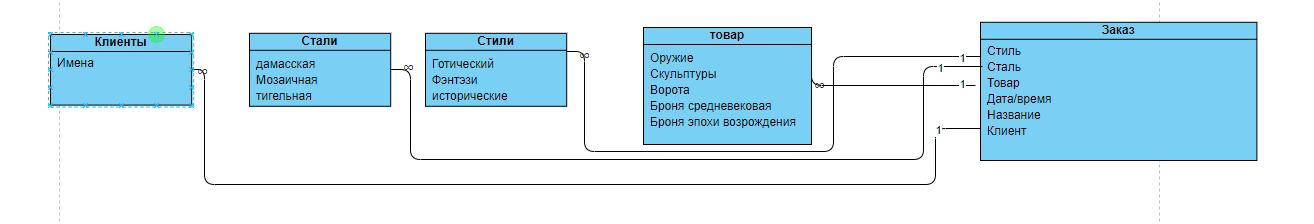


Рисунок 4. Схема IDEF0.

## Схема БД

Схема базы данных включает в себя описания содержания, структуры и ограничений [целостности](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), используемые для создания и поддержки [базы данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

Постоянные данные в среде базы данных включают в себя схему и базу данных. [Система управления базами данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) (СУБД) использует определения данных в схеме для обеспечения доступа и управления доступом к данным в базе данных



## Обоснования выбора инструментальных средств

C# — это язык программирования, предназначенный для разработки самых разнообразных приложений, предназначенных для выполнения в среде .NET Framework. Язык C# прост, типо безопасен и объектно-ориентирован. Благодаря множеству нововведений C# обеспечивает возможность быстрой разработки приложений, но при этом сохраняет выразительность и элегантность, присущую языкам C. Выполнение задачи на основе выбранного инструментального средства

* C# – это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения
* C# относиться к языкам компилируемого типа, поэтому он обладает всеми преимуществами таких языков
* C# объединяет лучшие идеи современных языков программирования Java, C++, Visual Basic и т.д
* Из-за большого разнообразия синтаксических конструкций и возможности работать с платформой .Net, C# позволяет быстрее, чем любой другой язык, разрабатывать программные решения
* C# отличается надежностью и элегантностью

### 3.6.1 Добавление базы данных

Ссылка на удаленный репозиторий - https://github.com/dimazuev1/kurs

Начальная стадия разработки базы данных на платформе C#: начинается с добавления элемента в панели «Добавить», кликая ПКМ по проекту(Рисунок 7).

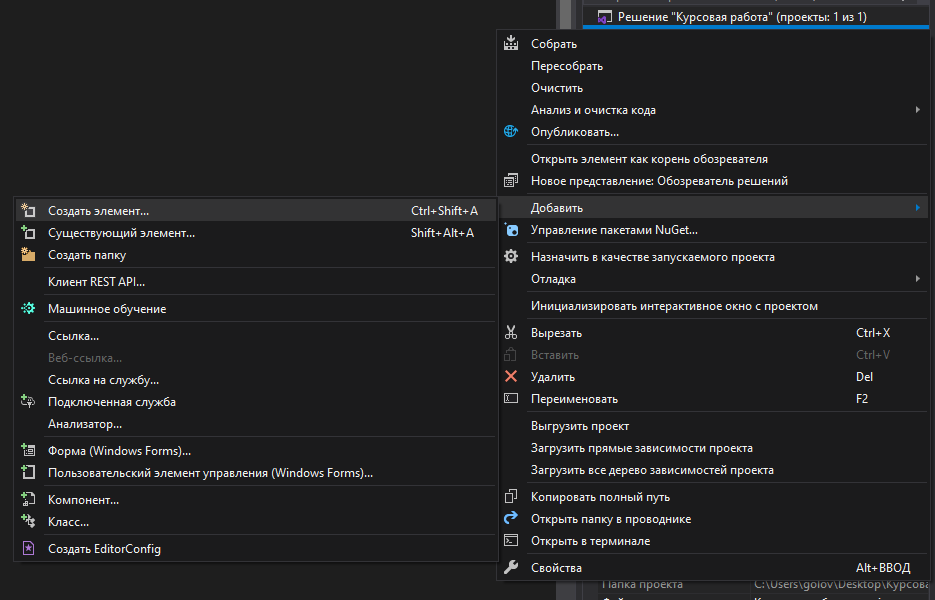


Рисунок 7. Добавление новой базы данных

В появившемся окне выбираем пункт «База данных, основанная на службах» и нажимаем кнопку «Добавить». (Рисунок 8).

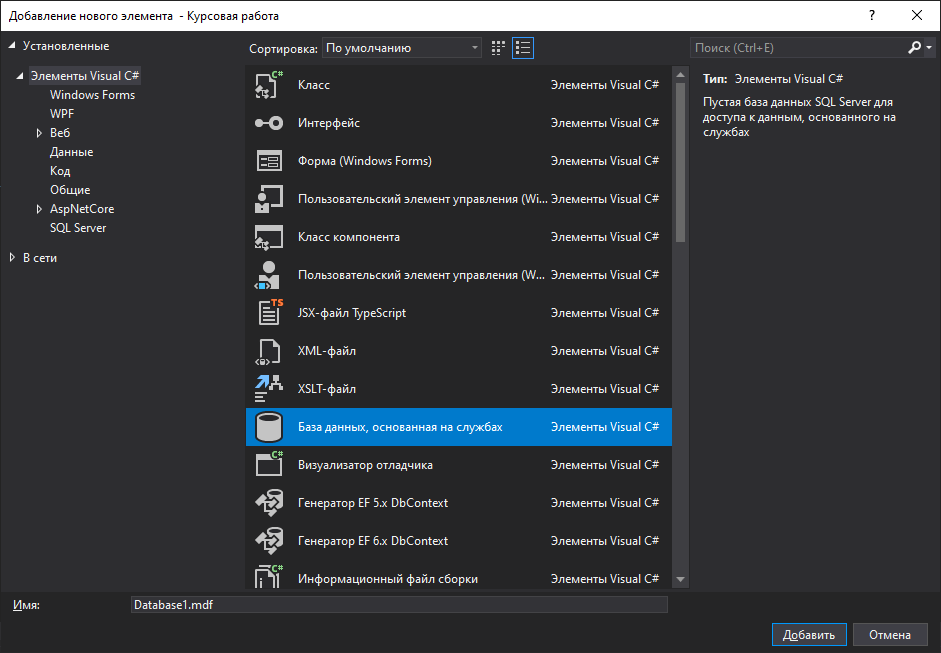


Рисунок 8. Первый этап создания базы данных.

После добавления базы данных, открываем нашу базу данных, кликая по ней дважды ЛКМ (Рисунок 9).

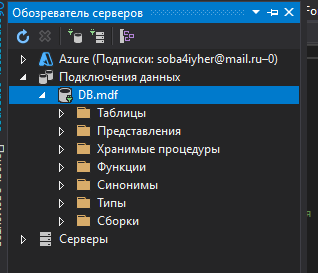


Рисунок 9. Общий вид созданной БД.

### 3.6.2. Решение поставленной задачи

Для решения поставленной задачи, нужно было создать 1 решение. В нем находятся 4 формы с кодом под каждое выводящее окно программы, БД, классы, которые отвечают за каждый элемент и сама генерация (рис. 10).

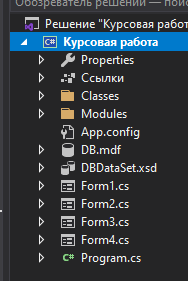


Рисунок 10.

Сначала, я посмотрел свой вариант и выделил там несколько областей: клиент, заказ, продукт, счет(Рис. 11).

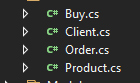


Рисунок 11.

Затем, отталкиваясь от этого, я создал симуляцию с генератором того, что я описал выше (Рис. 12-16)

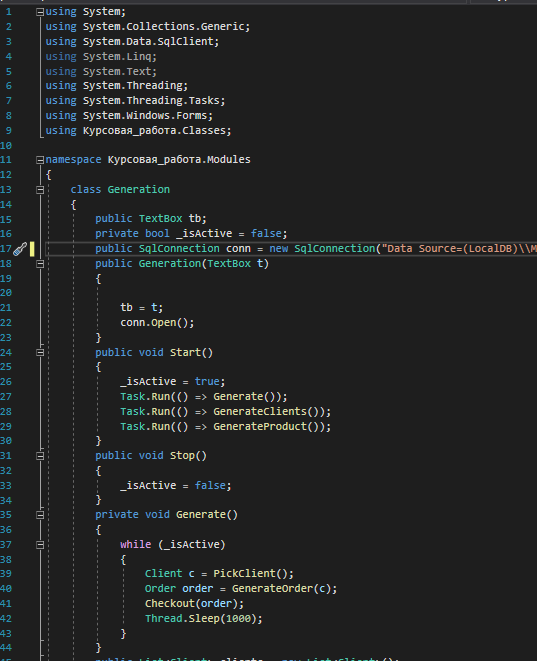


Рисунок 12.

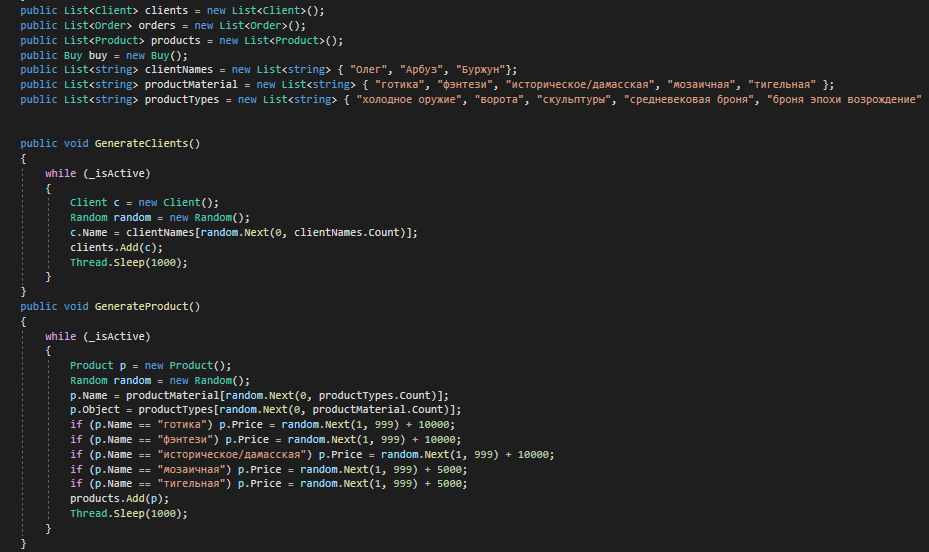


Рисунок 13.



Рисунок 14.

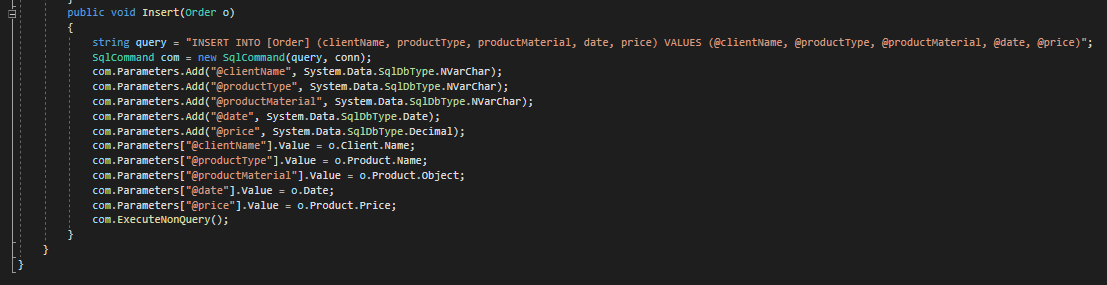


Рисунок 15.

Затем я добавил все эти значения в БД, встроенную в Windows Forms (Рис. 16)

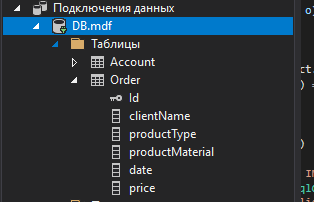


Рисунок 16.

Как только с основным генератором закончено, я приступил к добавлению нескольких форм, одна из которых позволяет заполнять БД вручную (Рис. 17)

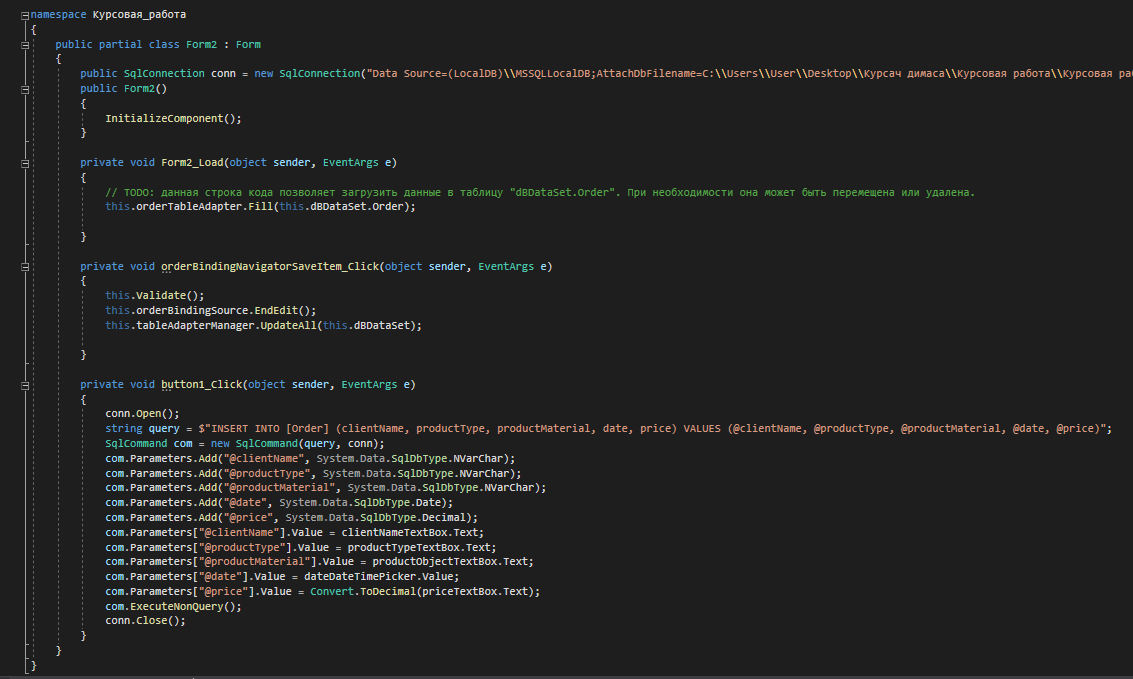


Рисунок 17.

Далее я создал форму, которая позволяет посчитать сумму выручки за все время в рублях от продажи изделий (Рис. 18).

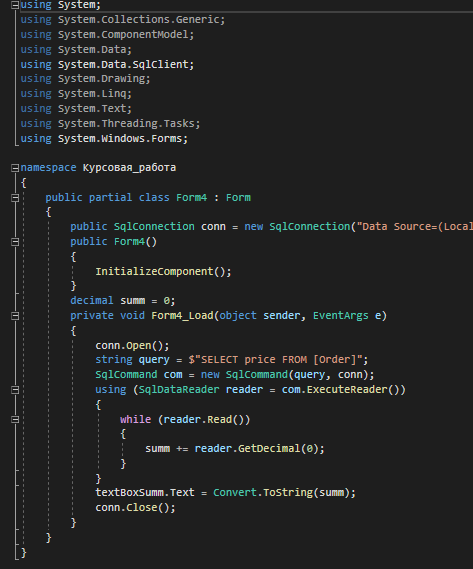


Рисунок 18.

В финале своей работы я сделал форму регистрации и логина, чтобы незарегистрированный пользователь не мог пользоваться этой программой (Рис. 19-20)

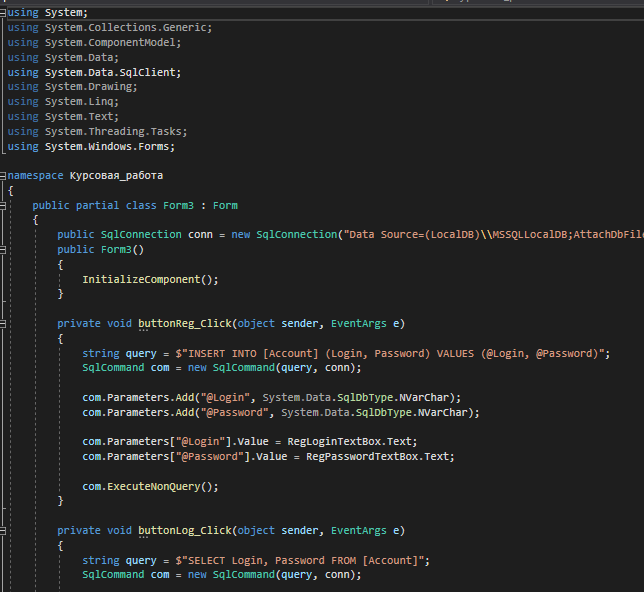


Рисунок 19.



Рисунок 20.

### 3.6.3. Пользовательский интерфейс

Интерфейс, удобный для пользователя мне сделать не удалось, поэтому я ограничился тем, что может предложить сама программа (Рис. 21).

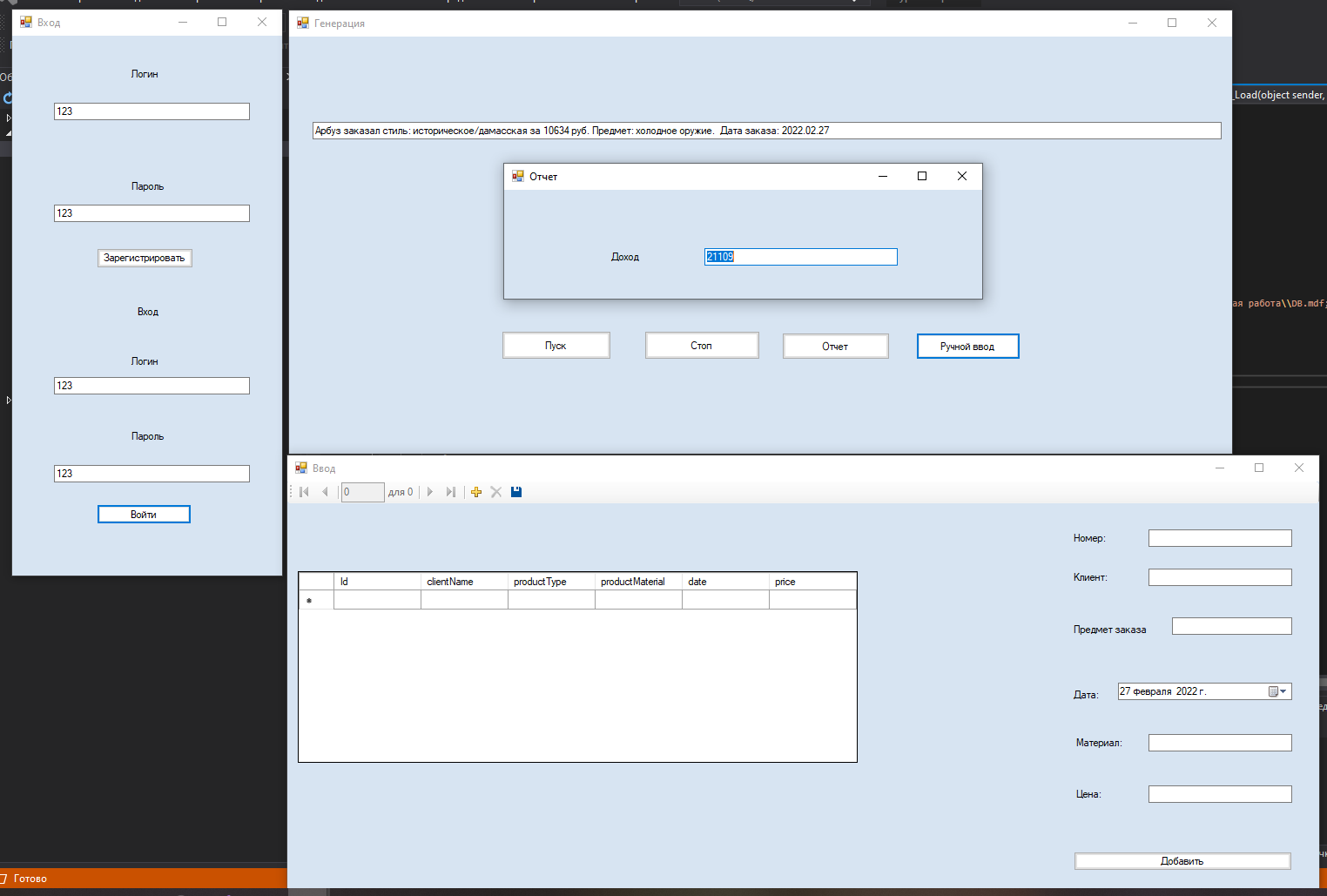


Рисунок 21.

## Экспериментальное тестирование

Для проверки работоспособности программы генерируются несколько клиентов и проверяется их запись в БД (Рис. 22).

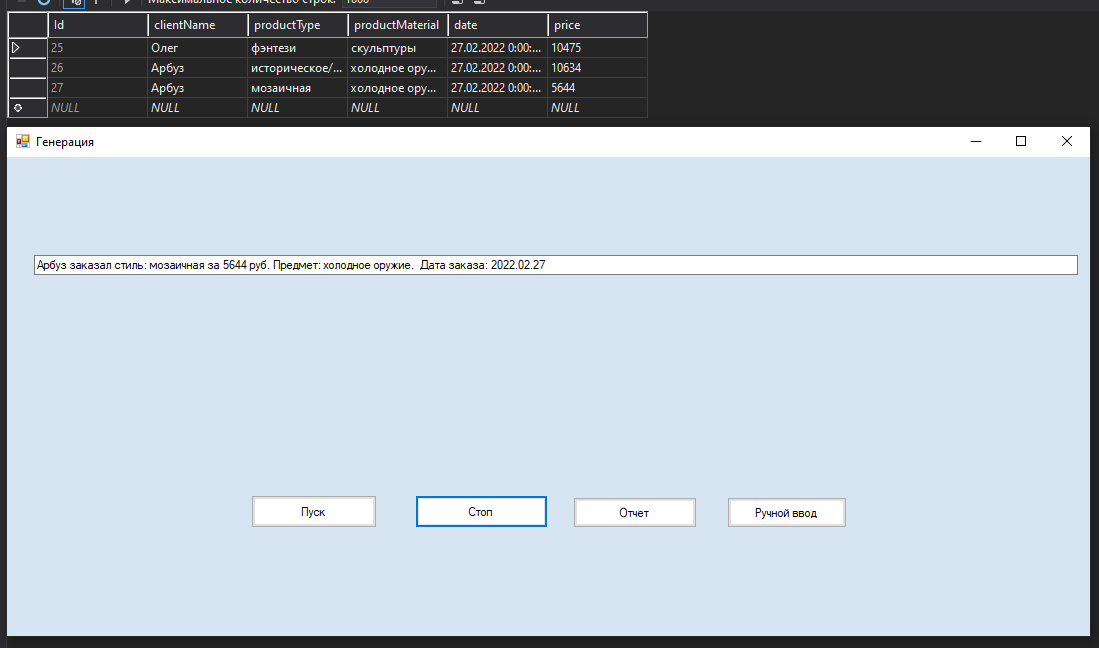


Рисунок 22.

## Эксплуатационные документы

## Руководства системного администратора

1. **Функции программы**

Автоматизированная информационная система «Кузница» предназначена для обобщения информации о клиентах и заказах.

1. **Минимальный состав технических средств**

* процессор Pentium-2.0Hz, не менее;
* оперативную память объемом, 1Гигабайт, не менее;
* HDD, 5 Гигабайт, не менее;
* операционную систему Windows 10;
* .net Framework 4.6.1

1. **Требования к персоналу (системному программисту)**

Менеджер и продавец должны пройти курс подготовки .net Framework 4.6.1 .

В перечень задач, выполняемых менеджером и сотрудником, будут входить:

1. Ведение документации
2. Обновление базы данных
3. Вывод на печать
4. **Настройка программы**
   1. **Настройка на состав технических средств**

База данных «Кузница» не требует каких либо настроек на состав технических средств.

* 1. **Настройка на состав программных средств**

Настройка на состав программных средств осуществляется путём установки платформы .net Framework 4.6.1.

* 1. **Описание способов проверки**
* Структурное тестирование базы данных - это касается таблицы и столбца тестирование, тестирование схемы, хранимых процедур и тестирования взглядов, проверка триггеров и т.д.
* Функциональное тестирование - Это включает в себя проверку функциональности базы данных с точки зрения пользователя. Наиболее распространенный тип функционального тестирования являются Белая коробка и тестирование черного ящика.
* Нефункциональных Тестирование - Она включает в себя нагрузки тестирование, тестирование риска в базе данных, стресс - тестирование, минимальные системные требования, а также сделки с производительностью базы данных.

1. Проверка работоспособности программы
2. Проверка наличия платформы .net Framework 4.6.1
3. Проверка имени базы данных
4. Проверка устройства данных, устройство дампа
5. Проверка, если хватает места, выделенного для каждой базы данных
6. Проверка настройки параметра базы данных
7. Первичный ключ для каждой таблицы
8. Внешние ключи для каждой таблицы
9. Типы данных между столбца внешнего ключа и столбца в других индексов таблицы, кластерные или не кластерные уникальным или не является уникальным.

## Руководство программиста

1. **Назначение и условия применения программы**
   1. **Назначение программы**

Программа предназначена для создания, управления содержимым базы данных, содержащим следующие данные:

1. Сведения о товарах (название)
2. Сведения о видах стилей ( готика, фэнтэзи, историческое)
3. Сведения о видах стали (дамасская, мозаичная, тигельная)
   1. **Функции, выполняемые программой**

Автоматизированная информационная система «Кузница» предназначена для обобщения информации о ковке изделий на заказ по различному стилю, из различной стали.

* 1. **Условия, необходимые для выполнения программы**
  2. **Объем оперативной памяти**
* .net Framework 4.6.1
  1. **Требования к составу периферийных устройств**

Программа «Кузница» не предъявляет.

* 1. **Требования к параметрам периферийных устройств**

Программа «Кузница» не предъявляет.

* 1. **Требования к программному обеспечению**

Компьютер конечного пользователя:

* процессор Pentium-2.0Hz, не менее;
* оперативную память объемом, 1Гигабайт, не менее;
* HDD, 5 Гигабайт, не менее;
* операционную систему Windows 10;
* .net Framework 4.6.1
  1. **Требования к персоналу**

Менеджер и продавец должны пройти курс подготовки C#.

В перечень задач, выполняемых менеджером и сотрудником, будут входить:

1. Ведение документации
2. Вывод на печать
3. **Характеристика программы**
   1. **Описание основных характеристик программы**

База данных позволяет производить учет о клиентах, каталоге изделий, заказах, доходах.

* 1. **Режим работы программы**

Файловый вариант работы обеспечивает легкость установки и эксплуатации системы. При этом для работы с информационной базой не требуются дополнительные программные средства, достаточно иметь операционную систему и .net Framework 4.6.1 .

Файловый вариант работы обеспечивает целостность информационной базы и простое создание резервных копий. Исключена ситуация, когда пользователь может по ошибке (например, при копировании информационной базы) перепутать различные файлы информационной базы и привести, таким образом, систему в неработоспособное состояние.

Резервное копирование может осуществляться на файловом уровне, путем простого копирования файла информационной базы.

* 1. **Средства контроля правильности выполнения программы**

Корректное отображение данных, возможность обновлять и сохранять данные.

* 1. **Описание основных особенностей программы**

Программа практична, легка в пользовании, имеет дружественный интерфейс и подходит как для новичков так и для бывалых пользователей.

* 1. **Самовосстанавливаемость программы**

Имеет автоматическое сохранение и настраиваемую функцию резервирования данных.

1. **Обращение к программе**
   1. **Загрузка и запуск программы**

Загрузка и запуск программы «Кузница» осуществляется путём открытия платформы .net Framework 4.6.1 .

* 1. **Выполнение функции сохранения обновлений базы данных**

Функция сохранения обновлений базы данных осуществляется автоматически после того, как были внесены в неё новые данные.

1. **Входные и выходные данные**
   1. **Организация используемой входной информации**

Входная информация будет содержать сведения клиенте, виде стали, товара и виде стиля.

* 1. **Организация используемой выходной информации**

Выходная информация будет содержать сведения о заказе.

## Руководство оператора

1. **Назначение программы**
   1. **Функциональное назначение программы**

Автоматизированная информационная система «Кузница» предназначена для обобщения информации о заказах, стилях, сталях, товарах.

* 1. **Эксплуатационное назначение программы**

Пользователями программы выступают сотрудник и менеджер станции. Менеджер отвечает за правильное ведение документа о регистрации заказов. Сотрудник ведет документ об учете БД.

* 1. **Условия выполнения программы**

Минимальный состав аппаратных средств:

* процессор Pentium-2.0Hz, не менее;
* оперативную память объемом, 1Гигабайт, не менее;
* HDD, 5 Гигабайт, не менее;
* операционную систему Windows 10;
* .net Framework 4.6.1
  1. **Минимальный состав программных средств**

.net Framework 4.6.1 .

* 1. **Требования к персоналу (пользователю)**

Продавец и менеджер должны пройти курс подготовки .net Framework 4.6.1 .

В перечень задач, выполняемых менеджером и продавцом, будут входить:

1. Ведение документации
2. Обновление базы данных

## Руководство к использованию

1. **Назначение программы**
   1. **Назначение программы**

Автоматизированная информационная система «Кузница» предназначена для обобщения информации о товарах, стилях, сталях и видах заказов.

* 1. **Возможности программы**

База данных позволяет хранить большой объём данных, обновлять их.

* 1. **Основные характеристики программы**

База данных позволяет производить учет заказов и облегчать работу с ней.

* 1. **Ограничения, накладываемые на область применения программного обеспечения**

База данных предназначена только для использования в кузнице. Свободное распространение по сети Интернет запрещено.

1. **Условия применения**
   1. **Требования к техническим (аппаратным) средствам**

* процессор Pentium-2.0Hz, не менее;
* оперативную память объемом, 1Гигабайт, не менее;
* HDD, 5 Гигабайт, не менее;
* операционную систему Windows 10;
* .net Framework 4.6.1
  1. **Общие характеристики входной информации**

Входная информация будет содержать сведения о клиенте, товаре, стали и стиле.

* 1. **Общие характеристики выходной информации**

Выходная информация будет содержать сведения о заказе.

* 1. **Требования и условия организационного характера**

Для обеспечения актуальной работы базы данных персонал должен систематически обновлять данные, заносить изменения в документы, отвечающие за заказы и клиентов.

* 1. **Требования и условия технического характера**

Для работы программного обеспечения базы данных «Кузница» должна быть установлена платформа .net Framework 4.6.1 .

1. **Описание задачи**
   1. **Определение задачи**

Определением задачи является задание на выполнение курсового проекта. Организация, утвердившая задание на разработку программного продукта: Колледж ПсковГУ.

Наименование работы: Автоматизированная информационная система «Кузница»

* 1. **Методы решения задачи**

Решение задачи путём создания автоматизированной информационной системы «Кузница» на .net Framework 4.6.1 .

1. **Входные и выходные данные**
   1. **Сведения о входных данных**

Входная информация будет содержать сведения клиенте, виде стали, виде стиля и товара.

* 1. **Организация используемой выходной информации**

Выходная информация будет содержать сведения о заказе.

1. Описание языка
   1. **Встроенный язык**

Встроенный язык является важной частью технологической платформы .net Framework 4.6.1 , поскольку позволяет разработчику описывать собственные алгоритмы функционирования прикладного решения.

C# — это объектно- и компонентно-ориентированный \_ язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО. C# —это \_ объектно-ориентированный язык.

* C# – это объектно-ориентированный, простой и в то же время мощный язык программирования, который позволяет разработчикам создавать многофункциональные приложения
* C# относиться к языкам компилируемого типа, поэтому он обладает всеми преимуществами таких языков
* C# объединяет лучшие идеи современных языков программирования Java, C++, Visual Basic и т.д
* Из-за большого разнообразия синтаксических конструкций и возможности работать с платформой .Net, C# позволяет быстрее, чем любой другой язык, разрабатывать программные решения
* C# отличается надежностью и элегантностью

**Событийная ориентированность встроенного языка**

Тип в программировании — понятие первичное. Тип некоторой сущности декларирует для нее совокупность возможных состояний и набор допустимых действий. Понятие сущности мы пока не уточняем, сущностями могут быть константы, переменные, массивы, структуры и т. д.

Наиболее часто понятие тип в языках программирования используют в связи с понятием «переменная».

Примитивное (но пока достаточное для наших целей) определение: переменная это пара «обозначение переменной + значение переменной».

Для переменной тип вводит совокупность ее возможных значений и набор допустимых операций над этими значениями.

## Заключение

Для выполнения данного курсового проекта использовались инструментальные средства разработки программного обеспечения, такие как:

Erwin Datamodeler для создания логической структуры программы;

MSOfficeVisio для создания схем структур данных;

.net Framework 4.6.1 для создания автоматизированной системы;

Так же в ходе выполнения курсового проекта разработан интерфейс пользователя, произведено тестирование и отладка готовой программы и написана программная документация.

Все поставленные задачи выполнены, знания и навыки, касающиеся инструментальных средств разработки программного обеспечения, создания программной документации, создания моделей данных, стадий процесса разработки программного продукта и стадий жизненного цикла, применены в полном объеме.

### Список источников информации

**Основные**

1. Прайс. C# 7 и .NET Core. Кросс-платформенная разработка для профессионалов
2. Бонд Д. Г. Unity и C#. Геймдев от идеи до реализации

**Дополнительные источники**

1. Котенко В. Н., Котенко Ю. В. Программирование на языках низкого уровня. Курс лекций. – 2016-621c.
2. ТереховА.Н. RTST-технология программирования встроенных систем реального времени //Записки семинара кафедры системного программирования" CASE-средства RTST. – 1998. – №. 1. – С. 3-17-235c.