МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Т. Ф. ГОРБАЧЕВА»
Филиал КузГТУ в г. Белово

Кафедра инженерно-экономическая

МДК 05.01 ПРОЕКТИРОВАНИЕ И ДИЗАЙН ИНФОРМАЦИОННЫХ СИСТЕМ

Методические рекомендации
по выполнению курсового проекта
для специальности
09.02.07 «Информационные системы и программирование»

Составитель: Витвицкий М.Н.
Рассмотрены и утверждены на заседании кафедры
Протокол № 3 от 15.11.2025 г.
Рекомендовано учебно-методической комиссией специальностей СПО в качестве электронного издания для использования в учебном процессе
Протокол № 3 от 18.11.2025 г.

Долганова Жанна Александровна
Подписано цифровой подписью: Долганова Жанна Александровна
Дата: 2025.12.19 11:14:10 +07'00'

Белово 2025

СОДЕРЖАНИЕ

1. ОБЩИЕ ПОЛОЖЕНИЯ 3
2. ОБЪЕМ И СОДЕРЖАНИЕ КУРСОВОГО ПРОЕКТА 4
3. РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ КУРСОВОГО ПРОЕКТА 6
4. РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА И МОДУЛЕЙ ПРИЛОЖЕНИЯ 11
4.1 Создание меню 11
4.2 Требования к разработке 11
4.3 Создание компонентов 14
4.4 Интегрирование 14
4.5 Назначение программы 15
5. ИНСПЕКЦИЯ КОДА 16
6. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ 21
6.1 Отладка программы 21
6.2 Тестирование программы 21
6.3 Анализ по результатам отладки и тестирования 24
7. СОЗДАНИЕ ФАЙЛОВ ДЛЯ НОСИТЕЛЯ ДАННЫХ 25
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 27
ПРИЛОЖЕНИЕ 1 29
ПРИЛОЖЕНИЕ 2 31
ПРИЛОЖЕНИЕ 3 32
ПРИЛОЖЕНИЕ 4 34
ПРИЛОЖЕНИЕ 5 35
ПРИЛОЖЕНИЕ 6 36
ПРИЛОЖЕНИЕ 7 41
ПРИЛОЖЕНИЕ 8 48

1. ОБЩИЕ ПОЛОЖЕНИЯ

Курсовой проект по МДК 05.01 «Проектирование и дизайн информационных систем» ПМ.05 «Проектирование и разработка информационных систем» являются завершающим этапом изучения междисциплинарного курса и ставят перед студентами следующие основные задачи:

- закрепить и углубить теоретические знания, полученные при изучении соответствующего курса;
- развитие информационных умений: аналитических, проектировочных, конструктивных и др.
- привить навыки пользования специальной литературой при решении конкретных вопросов;
- подготовить студентов к выполнению дипломного проекта.

Требования, предъявляемые к выполнению курсового проекта

Требования, предъявляемые к умениям в процессе выполнения курсовой работы:

- осуществлять постановку задач по обработке информации;
- проводить анализ предметной области;
- осуществлять выбор модели и средства построения информационной системы и программных средств;
- использовать алгоритмы обработки информации для различных приложений;
- решать прикладные вопросы программирования и языка сценариев для создания программ;
- разрабатывать графический интерфейс приложения;
- создавать и управлять проектом по разработке приложения;
- проектировать и разрабатывать систему по заданным требованиям и спецификациям;
- кодировать на языках программирования;
- проинспектировать код на соответствие стандартов;
- тестировать результаты собственной работы.

По результатам выполнения курсового проекта оформляется пояснительная записка. Курсовые проекты оформляются в соответствии с требованиями государственного стандарта ГОСТ Р 2.105- 2019.

Законченные курсовые работы в установленный срок студенты сдают руководителю, который проверяет качество выполнения всех частей проекта и его соответствие объему, указанному в задании. После проверки руководитель подписывает пояснительную записку и возвращает их студентам для ознакомления с рецензией и устранения отмеченных недоработок.

Окончательный прием выполненных проектов может проводиться в форме открытой защиты. Студент, получивший неудовлетворительную оценку по курсовому проекту, получает другое задание и ему устанавливается новый срок для его выполнения.

2. ОБЪЕМ И СОДЕРЖАНИЕ КУРСОВОГО ПРОЕКТА

Курсовой проект работа оформляется в двух частях – текстовой и программной.

Текстовая часть работы оформляется в виде пояснительной записки, содержащей обоснования, расчеты и показатели разработанных и рекомендуемых решений. В пояснительной записке могут быть использованы различные графические элементы (рисунки, таблицы).

Программная часть проекта оформляется в виде архива, содержащего отлаженную, работающую программную реализацию готовой информационной системы, и сдается руководителю курсовой работы в электронном виде.

Пояснительная записка должна содержать следующие разделы:

Титульный лист (ПРИЛОЖЕНИЕ 2),

Задание на курсовой проект (ПРИЛОЖЕНИЕ 3),

СОДЕРЖАНИЕ (ПРИЛОЖЕНИЕ 4)

ВВЕДЕНИЕ (ПРИЛОЖЕНИЕ 5)

РАЗДЕЛ 1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.2 ОПИСАНИЕ ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ СОЗДАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ

1.3 ТЕХНИЧЕСКОЕ ЗАДАНИЕ (ТЗ ПРИМЕР ПРИЛОЖЕНИЕ 6)

1.4 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ (ПОСТРОЕНИЕ UML-ДИАГРАММ, ФУНКЦИОНАЛЬНОЙ МОДЕЛИ IDEF0)

РАЗДЕЛ 2 ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ И КОДА (ПРИМЕР ПРИЛОЖЕНИЕ 7)

2.1.1 Создание меню

2.1.2 Требования к разработке

2.1.3 Создание компонентов

2.1.4 Интегрирование

2.1.5 Назначение программы

2.2 ИНСПЕКЦИЯ КОДА

2.3 ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММЫ (ИНСТРУКЦИЯ К ЭМУЛЯТОРУ ПРИЛОЖЕНИЕ 8)

2.3.1 Отладка программы

2.3.2 Тестирование программы

2.3.3 Анализ по результатам отладки и тестирования

2.4 ДОКУМЕНТАЦИЯ ПО РАЗРАБОТКЕ

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЯ

Пояснительная записка сдается в распечатанном виде на листах формата А4 книжной ориентации (с одной стороны листа) и оформляется в соответствии с ГОСТ Р 2.105-2019. Размер шрифта (кегль) основного текста – 14 пт, дополнительного (содержание таблиц) – 12 или 11 пт. Межстрочный интервал основного текста – 1,5; дополнительного (содержание таблиц) – 1,0.

При наборе рекомендуется использовать основные системные гарнитуры шрифта TimesNewRoman. Текст набирается с соблюдением следующих правил: не допускаются ручной набор нумерации в главах и абзацах (только автонумарация); два и более пробела между символами. При наборе должны различаться тире и дефисы; маркеры и другие знаки должны быть сохранены аналогичными на протяжении всего материала. Между инициалами и после них (перед фамилией) ставится неразрывный пробел.

Размеры полей «обычное»: верхнее 1 см, левое 2 см, нижнее 1 см, правое 1 см. Нумерация страниц – внизу «по центру» шрифтом 12 пт. гарнитуры шрифта TimesNewRoman, нумерация страниц записки сквозная, причем начинается простановка номеров со страницы «Содержание», с учетом всех впереди стоящих страниц, на которых номера не проставляются.

Названия разделов – полужирный шрифт прописными буквами, подразделов – полужирный шрифт с заглавной буквы, выравнивание «по центру», без переносов и точки в конце текста, размер шрифта – 16. Если название раздела или подраздела состоит из нескольких предложений, то между ними ставится точка и не ставится в конце.

Курсовой проект состоит из задания, пояснительной записки и графической (иллюстративной) части.

Объем пояснительной записки курсовой работы – не менее 40 листов формата А4.

Схемы, рисунки, графики и таблицы необходимо выполнять по центру, которые также вкладываются в пояснительную записку.

Задание вкладывается в пояснительную записку после титульного листа, но не считается листом курсового проекта.

Тематика курсовых работ отображает предметную область информационных систем (ПРИЛОЖЕНИЕ 1).

В СОДЕРЖАНИИ приводятся наименования разделов пояснительной записки, начиная с ВВЕДЕНИЯ и заканчивая ПРИЛОЖЕНИЯМИ, и номера листов, соответствующие началу каждого раздела.

3. РЕКОМЕНДАЦИИ ПО ВЫПОЛНЕНИЮ КУРСОВОГО ПРОЕКТА

СОДЕРЖАНИЕ

В содержании перечисляются заголовки разделов, подразделов, список используемых источников, приложения и указываются страницы, на которых они начинаются (только автоматическое содержание).

ВВЕДЕНИЕ

Введение должно содержать краткую характеристику современного состояния проблемы, которой посвящена тематика курсовой работы, обоснована необходимость проведения этой работы, показана актуальность темы. Введение должно содержать основание для разработки, показаны цели и задачи курсовой работы.

Основная часть курсового проекта состоит из двух разделов:

1. В первом разделе содержатся теоретические основы разрабатываемой темы: описание предметной области и инструментальных средств создания информационной системы.

2. Вторым разделом является практическая часть, которая представлена:

Проектирование информационной системы

Разработка графического интерфейса и модулей приложения

Программирование в соответствии с требованиями технического задания

Инспекция кода

Отладка и тестирование программы

Интегрирование

Руководство пользователя (РД 50-34-698-90)

Техническое задание для разработки информационной системы (пример ПРИЛОЖЕНИЕ 6) составляется в соответствии с ГОСТ 34.602-89 и должно включать разделы:

- общие сведения о системе;
- назначение и цели создания системы;
- характеристика объекта информатизации;
- требования к системе:
  - к системе в целом;
  - к функциям системы;
  - к видам обеспечения системы.

При проектирования информационной системы необходимо разработать:

1. Диаграмму вариантов использования (рис. 1);

2. ER-модель логическую (содержит только названия таблиц, название полей, связи, типы связей, пример на рис.2) и ER- физическую (добавляются типы данных, ограничения на данные, пример на рис.3);

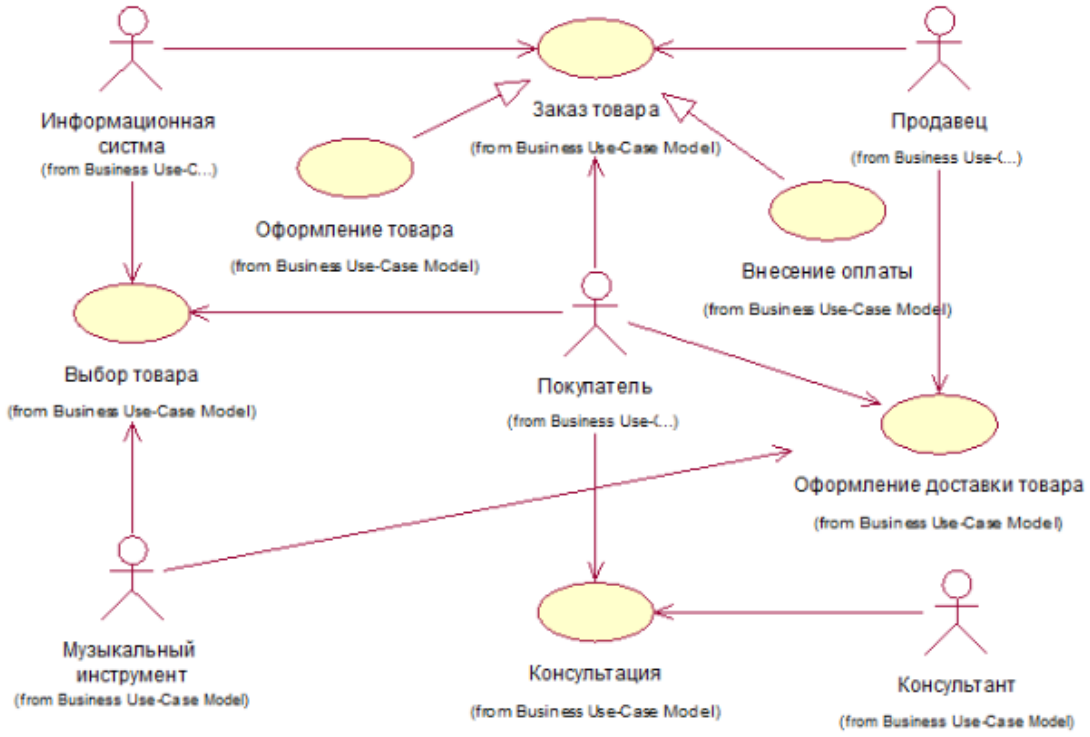

Рисунок 1 – Пример диаграммы вариантов использования ИС

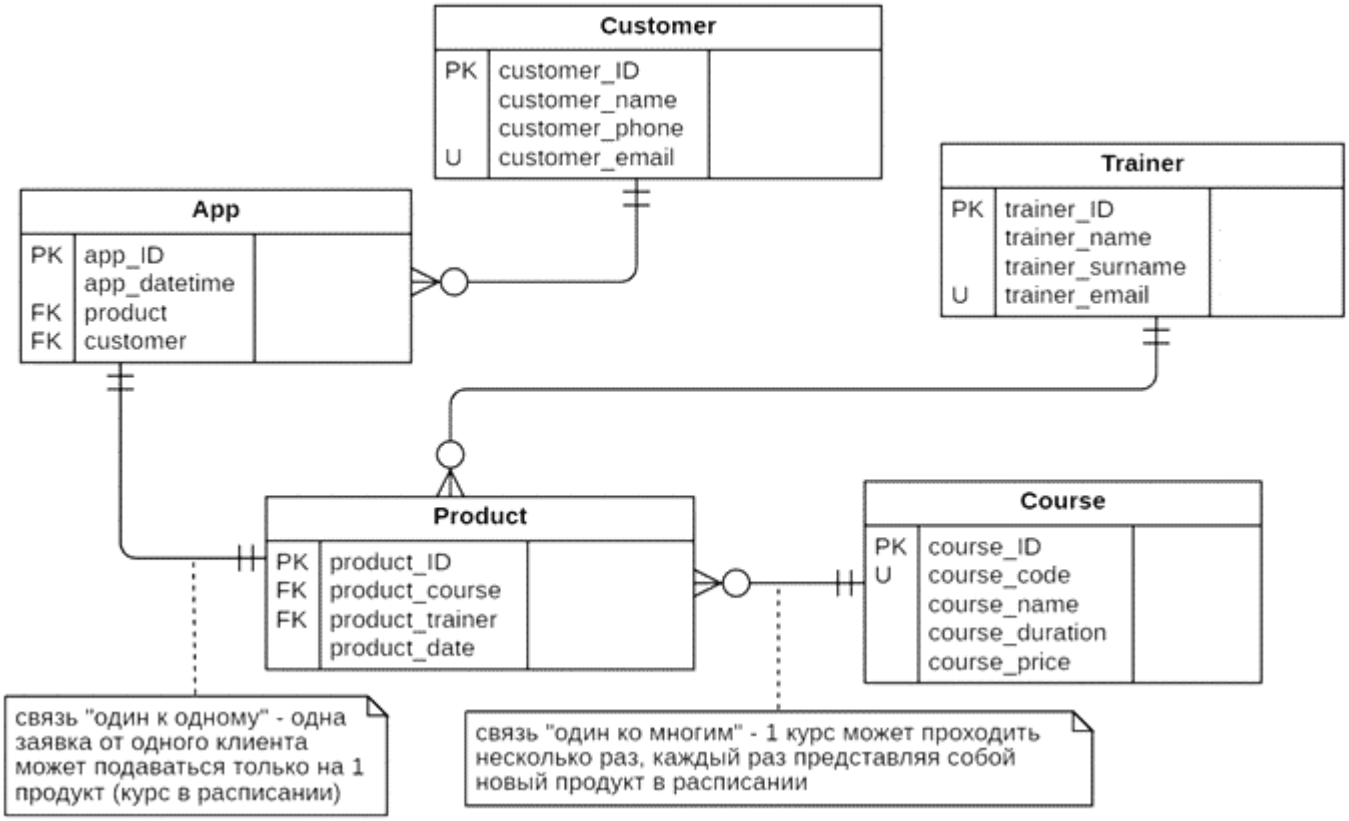

Рисунок 2 – Пример ER-модель физическая

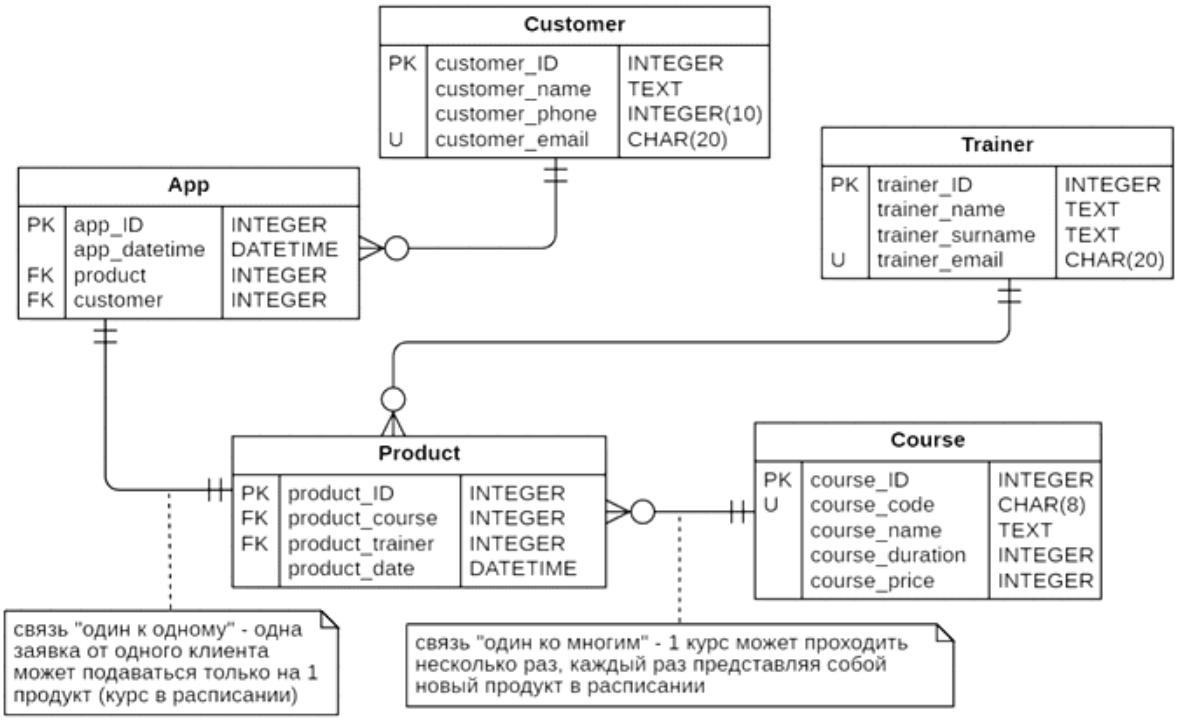

Рисунок 3 – Пример ER-модель физическая

3. Концептуальную модель в соответствии с нотацией IDEF0, которая должна включать несколько диаграмм:

a) контекстную диаграмму верхнего уровня, которая отражает задачу в целом (пример на рис.4);

b) диаграмму, показывающие бизнес-процессы в целом (пример на рис.5);

c) дочерние диаграммы декомпозиции каждого из бизнес-процессов, которые требуют детализации (примеры на рис.6, 7,8).

Рисунок 4 – Контекстная диаграмма ИС

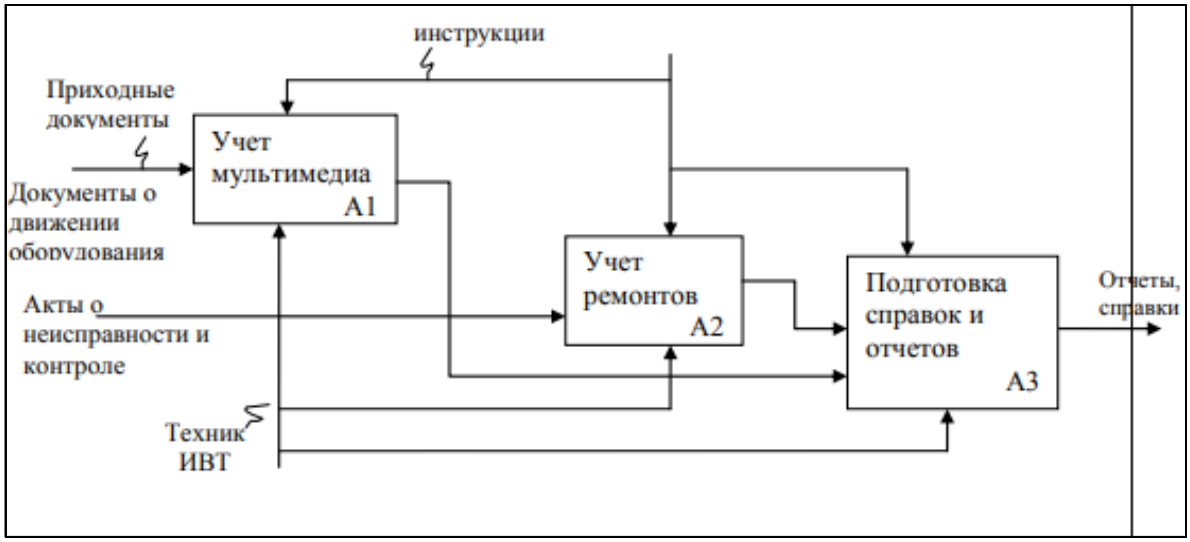

Рисунок 5 - Диаграмма основных процессов

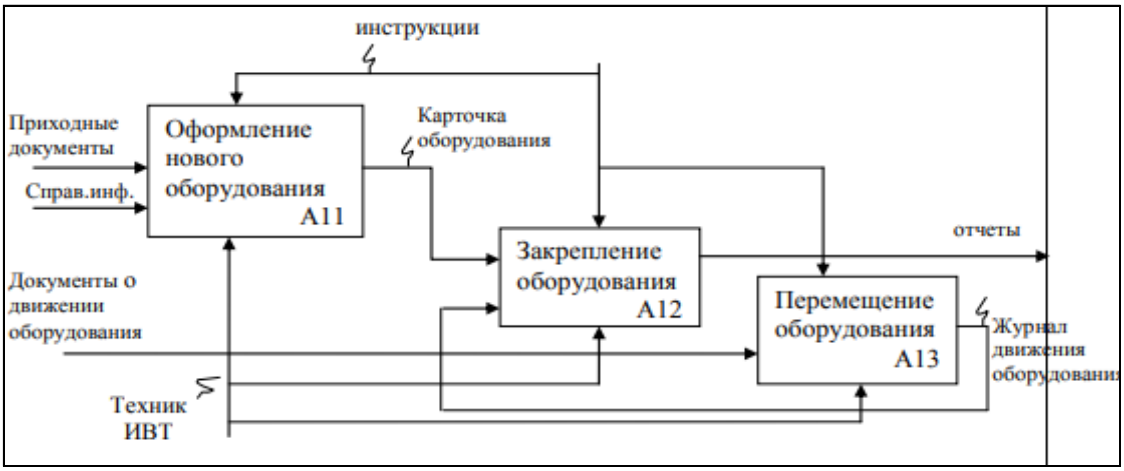

Рисунок 6 - Диаграмма процесса «Учет мультимедиа»

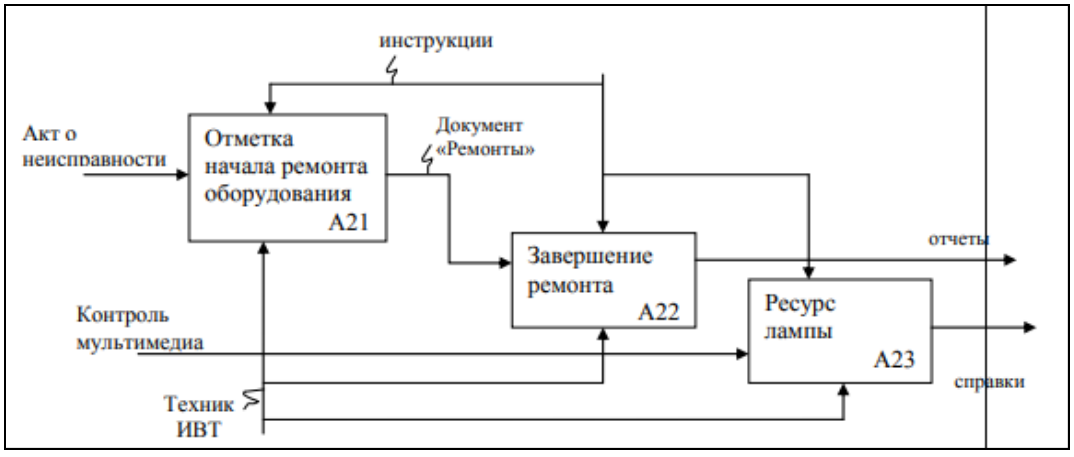

Рисунок 7- Диаграмма процесса «Учет ремонтов»

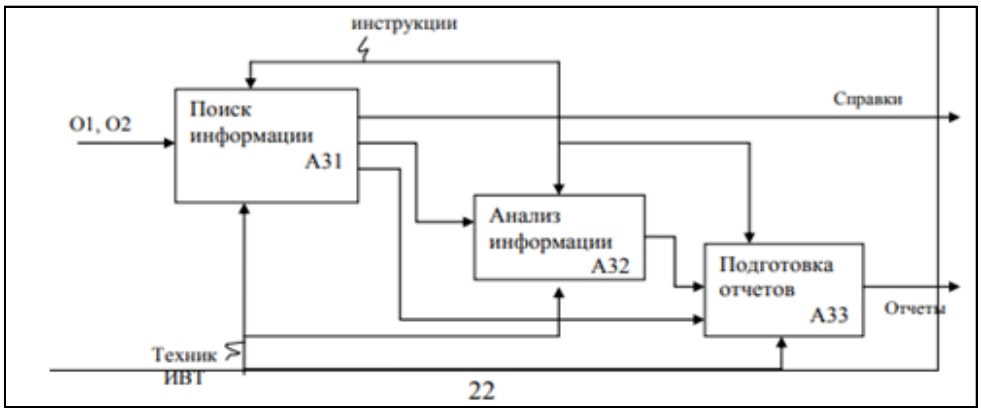

Рисунок 8 - Диаграмма процесса «Подготовка справок и отчетов»

Заключение на курсовой проект

Заключение должно содержать краткие выводы по результатам курсового проекта. Здесь нужно отразить степень достижения поставленной цели. Особенности выполнения каждой из поставленных задач.

Список используемых источников

Список используемых источников оформляется согласно ГОСТ 7.1-2003.

Приложения

В приложения могут быть включены:

1. Таблицы большого формата
2. Листинги программ (ГОСТ 19.401-78 Текст программы) и др.

4. РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА И МОДУЛЕЙ ПРИЛОЖЕНИЯ (ПРИМЕР ИС «УЧЕТ ТОВАРОВ»)

4.1 Создание меню

Для создания меню можно использовать https://www.figma.com/ (автоматизация создания графического интерфейса), а можно в любом графическом редакторе создать графический файл со структурой связей между всеми элементами меню рис. 9.

Обязательно описание сценариев работы с интерфейсом со всем его функционалом.

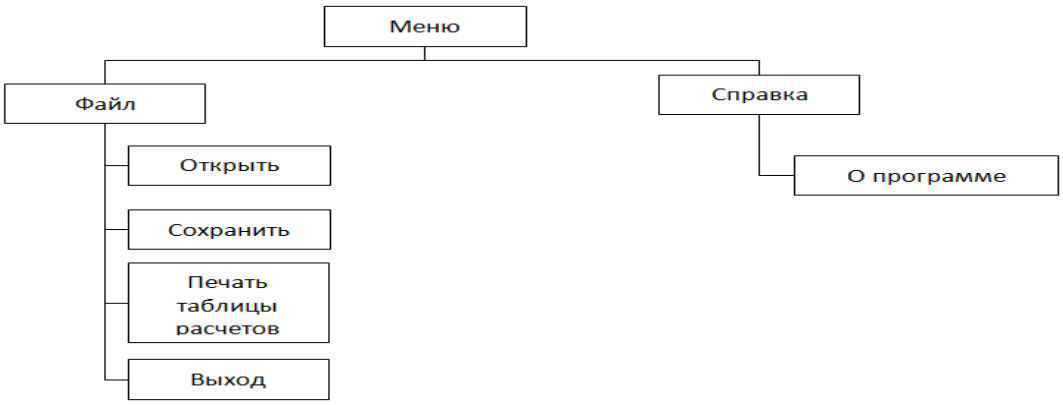

Рисунок 9 - Состав модулей проекта

4.2 Требования к разработке

На основании документов, представленных заказчиком, необходимо спроектировать ER-диаграмму для информационной системы. Обязательна 3 нормальная форма с обеспечением ссылочной целостности. При разработке диаграммы обратите внимание на согласованную осмысленную схему именования, создайте необходимые первичные и внешние ключи. ER - диаграмма должна быть представлена в формате .png и содержать таблицы, связи между ними, атрибуты и ключи (типами данных на данном этапе можно пренебречь).

Создайте базу данных на основании разработанной ER-диаграммы логической, используя предпочтительную платформу, на сервере баз данных, который вам предоставлен. Создайте таблицы основных сущностей, атрибуты, отношения и необходимые ограничения. После создания базы данных требуется рассмотрите способ импортировать данные из файла с расширением .json (данные или структуру).

Создайте запросы, позволяющий вычислить, искать, структурировать данные в вашей системе не менее 3 запросов на проект (ПРИМЕР):

- количество продукции в заказе;
- стоимость всех материалов, использованных для производства данной продукции (учитывая норму расхода)
- сумма расходов на определенный материал.

Функционал должен содержать поиск, импорт данных в файл формата pdf и json.

Разработайте функционал бэкапирования с системой управления и хранения бэкапов на срок не менее 30 дней, с последующей перезаписью самого старого бэкапа формата названия бэкапа В_дд_мм_гггг, система управления бэкапа должна позволить удалять бэкапы из списка с накопителя данных.

Для выполнения задания рекомендуется создать в базе данных таблицу "Users". Если такая таблица уже существует, необходимо внести некоторые изменения для реализации дальнейшего функционала приложения. Разработайте форму для авторизации зарегистрированных пользователей с ролями "Администратор" и "Пользователь".

Форма должна содержать поля текстовые поля логин, пароль и кнопку "Войти". Поля "Логин" и "Пароль" должны быть обязательными для заполнения.

При неверно введенных данных, пользователь должен получить сообщение об ошибке "Вы ввели неверный логин или пароль. Пожалуйста проверьте ещё раз введенные данные".

После успешной авторизации пользователь должен получить сообщение "Вы успешно авторизовались".

При аутентификации связка «логин/пароль» должна совпадать с одной из записей в таблице "Пользователи".

На страницу авторизации добавьте интерактивную капчу, в которой пользователю необходимо собрать исходное изображение из фрагментов.

Метод сборки изображения может быть произвольным. После сборки изображения система проверяет правильность расположения фрагментов.

Если пазл собран верно — пользователь проходит проверку и может авторизоваться.

Если в течении 3-х раз подряд пазл собран не верно или не верно введен пароль, то учетная запись блокируется и при повторной авторизации должно появляться сообщение "Вы заблокированы. Обратитесь к администратору".

На рабочем столе пользователя с ролью "Администратор" предусмотрите функционал для добавления новых пользователей, изменения данных текущих пользователей (включая снятие блокировки). При добавлении нового пользователя следует проверять его наличие в базе данных.

В случае, если пользователь с указанным логином уже существует, должно выводиться соответствующее сообщение.

Разработайте проектную документацию на разработанный функционал.

Включите описание функционального назначения, используемые методы с указанием параметров.

Название ИС

Используйте соответствующие названия для ваших приложений и файлов. Так, например, наименование настольного приложения должно обязательно включать название компании- заказчика.

Файловая структура

Файловая структура проекта должна отражать логику, заложенную в приложение. Например, все формы содержатся в одной директории, пользовательские визуальные компоненты – в другой, классы сущностей – в третьей.

Структура проекта

Каждая сущность должна быть представлена в программе как минимум одним отдельным классом. Классы должны быть небольшими, понятными и выполнять одну единственную функцию.

Макет и технические характеристики

Все компоненты системы должны иметь единый согласованный внешний вид, а также следующим требованиям:

1. разметка и дизайн (предпочтение отдается масштабируемой компоновке);
2. должно присутствовать ограничение на минимальный размер окна;
3. должна присутствовать возможность изменения размеров окна, где это необходимо;
4. увеличение размеров окна должно увеличивать размер контентной части, например, таблицы с данными из БД);
5. группировка элементов (в логические категории);
6. использование соответствующих элементов управления (например, выпадающих списков для отображения подстановочных значений из базы данных);
7. расположение и выравнивание элементов (метки, поля для ввода и т.д.);
8. последовательный переход фокуса по элементам интерфейса (по нажатию клавиши TAB);
9. общая компоновка логична, понятна и проста в использовании;
10. соответствующий заголовок на каждом окне приложения (не должно быть значений по умолчанию типа MainWindow, Документ1 и т. п.).

Обратная связь с пользователем

Уведомляйте пользователя о совершаемых им ошибках или о запрещенных в рамках задания действиях, информируйте об отсутствии результатов поиска и т.п. Окна сообщений соответствующих типов (например, ошибка, предупреждение, информация) должны отображаться с соответствующим заголовком и пиктограммой. Текст сообщения должен быть полезным и информативным, содержать полную информацию о совершенных ошибках пользователя и порядок действий для их исправления. Также можно использовать визуальные подсказки для пользователя при вводе данных.

Обработка ошибок

Не позволяйте пользователю вводить некорректные значения в текстовые поля сущностей. Например, в случае несоответствия типа данных или размера поля введенному значению. Оповестите пользователя о совершенной им ошибке.

При возникновении непредвиденной ошибки приложение не должно аварийно завершать работу.

Оформление кода

Идентификаторы переменных, методов и классов должны отражать суть и/или цель их использования, в том числе и наименования элементов управления (например, не должно быть значений по умолчанию типа Form1, button3).

Идентификаторы должны соответствовать соглашению об именовании среды разработки.

Допустимо использование не более одной команды в строке.

Комментарии

Используйте комментарии для пояснения неочевидных фрагментов кода. Запрещено комментирование кода. Хороший код воспринимается как обычный текст. Не используйте комментарии для пояснения очевидных действий. Комментарии должны присутствовать только в местах, которые требуют дополнительного пояснения.

4.3 Создание компонентов

Компоненты системы можно реализовать в PHP, C++, C#.

Система упраления базами данных рекомендованна phpMyAdmin реализованная на сервере филиала с удаленным подключением по портам и персональному логину и паролю выдаваемому преподавателем перед началом практических работ курса с правом управления БД mariadb: http://134.90.167.42:10307 — подключение к СУБД с правом управления и редактирования БД по https, http://134.90.167.42:10306 — одключение к СУБД с правом управления и редактирования БД из кода управления.

Пример кода на PHP с файлом

4.3 Описание процедур

После создания форм проекта выполняем проектирование процедур, форм, страниц (см. табл. 1). Текст процедур приведен в электронном виде в разделе разработка кода.

Таблица 1 – Описание основных процедур проекта

Наименование процедуры	Назначение
1	2
сохранитьToolStripMenuItem_Click (кнопка)	Сохранение таблицы.
открытьToolStripMenuItem_Click (кнопка)	Открыть таблицу.
Страница автаризации (index.php или на ваше усмотрение)	Путь к странице авторизации

4.4 Интегрирование

Для проверки данных от клиентов разработайте приложение, которое позволит провести валидацию на корректность данных. Результат проверки необходимо фиксировать в документе ТестКейс.docx.

Сначала заполните в документе ТестКейс.docx столбец "Действие" и "Ожидаемый результат" используя предоставленный текстовый редактор.

Добавьте закладки в столбец "Результат". Необходимо провести валидацию ФИО клиента на вхождение запрещенных символов. Проверьте два любых критерия.

Для эмуляции отправки данных от клиента Вам необходимо запустить приложение TransferSimulator.exe. Методы эмулятора описаны в ПРИЛОЖЕНИИ 8.

Макет формы представлен на рис.10

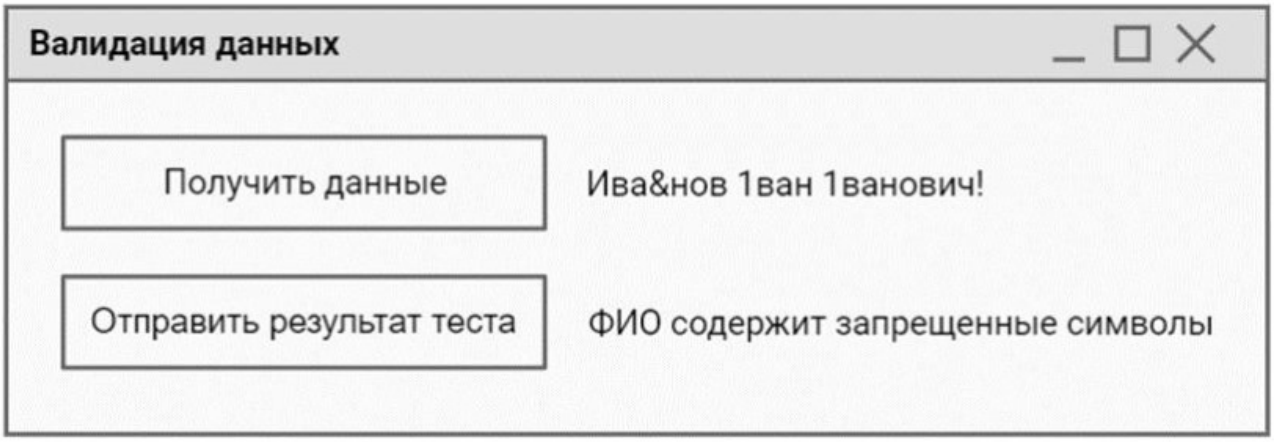

Рисунок 10. Окно интегрированного приложения

4.5 Назначение программы

Программа «Учет товаров» позволяет выполнить действия: добавление, изменение и удаление данных о товарах, выводить справочные данные.

Преимущества программы:

- удобный интерфейс;
- предоставляет необходимую информацию;
- сокращает время поиска нужной информации.

5. ИНСПЕКЦИЯ КОДА

Код должен соответствовать стандартами оформления: PHP, C#, C++.

Правильное оформление кода PHP предполагает его простое визуальное восприятие. Оно достигается с помощью отступов и пробелов. Для формирования отступов используйте пробелы, а не знак табуляции. Каждую строку начинайте с четырех пробелов. Код должен идти лесенкой: раскрываться вправо, затем собираться обратно.

Отступы:

Запомните: один отступ = четыре пробела.

Выделяем отступами тело конструкции, тело метода, блоки импорта, аргументы и подобное.

Правильно

```
<?php
switch ($expr) {
  case 1:
     echo `One`;
     break;
  case 2:
     echo `Two`;
     break;
  }
?>
```

Неправильно

```
<?php
 switch($expr)
{
        case1:
    echo `One`;
         break;
  case 2:
           echo `Two`;
  break;    }
?>
```

Пробелы

Ставятся:

- между for (foreach/ while / catch) и (
- после ;
- между) и {
- перед =>
- после =>
- между try и {
- между } и catch

Не ставятся:

1. После имени метода.
2. В списке аргументов перед запятыми.
3. Между (и именем функции или метода.

Пустая строка

Вставляется:

1. После каждого логического блока.
2. После определения пространства имен.
3. После блока импорта. Каждая строка блока должна начинаться с use.

Правильно

```
<?php
    namespace Vendor\Package;

   use FooClass;
   use BarClass as Bar;
   use OtherVendor\OtherPackage\BazClass;

  // …
?>
```

Неправильно

```
<?php
    namespace Vendor\Package;

   use FooClass; use BarClass as Bar;
   use OtherVendor\OtherPackage\BazClass;

  // …
?>
```

Круглые скобки

1. Не выносим на отдельные строки.
2. Не ставим пробелы внутри: после (и перед).
3. Ставим пробелы до скобок и после.
4. Внутри перечисление отделяем пробелами.

Фигурные скобки

1. Открывающая фигурная скобка выносится на новую строку перед телом метода, для классов.

2. Открывающая фигурная скобка не выносится на отдельную строку в конструкциях и замыканиях.

3. Закрывающая скобка } в конструкциях, имени метода, определении метода, классах пишется с новой строки и отделяется одним отступом.

Оформляются двумя способами: в одну строку через запятую или в столбик. Аргументы на одной строке пишутся через запятую внутри круглых скобок. Пробел ставится только после запятой.

Правильно

```
<?php
  Foo: :bar($arg1, $arg2, $arg3);
?>
```

Неправильно

```
<?php
  Foo: :bar($arg1   , $arg2,$arg3);
?>
```

При оформлении в столбик каждый аргумент пишется с новой строки и отделяется двойным отступом. Первая круглая скобка остается на строке вместе обозначением метода. Вторая круглая скобка выносится в отдельную строку

вместе с открывающей фигурной скобкой. Между ними пробел.

Правильно

```
<?php
   $foo ->bar(
      $firstArgument,
      $secondArgument,
      $thirdArgument
   );
?>
```

Неправильно

```
<?php
   $foo ->bar(
      $firstArgument,
      $secondArgument,
      $thirdArgument);
?>
```

Конструкция switch case

Конструкцию делим на три уровня: switch, case, echo/break. Каждый уровень начинается с отступа. Таким образом, наш код визуально выглядит состоящим из трех столбцов.

Если в конструкции не используется break, поставьте // no break.

Правильно

```
<?php
   switch ($expr) {
      case 0:
         echo `First case, with a break`;
         break;
      case 1:
         echo `Second case, with fall through`;
         // no break
      case 2:
      case 3:
      case 4:
         echo `Third case, return instead of break`;
         return;
      default:
         echo `Default case`;
         break;
   }
?>
```

Неправильно

```
<?php
   switch ($expr) {
      case 0:
         echo `First case, with a break`;
         break;
      case 1:
         echo `Second case, with fall through`;
      case 2:
```

```
        case 3:
        case 4:
            echo `Third case, return instead of break`;
            return;
        default:
            echo `Default case`;
            break;
    }
?>
```

Конструкция try catch

Тело try и тело catch отделяются одним отступом. Пробелы нужно поставить:

- между try и {
- между } и catch
- между catch и (
- между) и {

Catch и скобку } ставим на одну строку.

Правильно

```
<?php
  try {
        // тело try
        } catch (FirstExceptionType $i) {
        // тело catch
    } catch (OtherExceptionType $i) {
        // тело catch
    }
?>
```

Неправильно

```
<?php
try {
     // тело try
        }
catch (FirstExceptionType $i)
{
        // тело catch
 } catch (OtherExceptionType $i) {
        // тело catch     }
?>
```

Конструкция if, elseif, else

Операторы и открывающую фигурную скобку пишем на одной строке. Закрывающую фигурную скобку оператора пишем на той же строке, что и оператор. Заключительную фигурную скобку пишем на отдельной строке. Оператор else if пишем как единое слово — elseif. Тело оператора отделяем отступом.

Правильно

```
<?php
    if ($a == $b) {
        echo `A равно B`;
```

```
   } elseif ($a == $c) {
      echo `A равно C`;
   } else {
      echo `A ничему не равно`;
   }
?>
```

Неправильно

```
<?php
   if ($a == $b) {
      echo `A равно B`;
   } else if ($a == $c)
 {
      echo `A равно C`;
   } else
{
      echo `A ничему не равно`;
   }
?>
```

Комментарии в коде

Чистый код должен быть правильно закомментирован. К сожалению, встречаются две крайности: подробное комментирование каждой строки и полное отсутствие комментариев. И то, и другое мешает в работе. Избыточное комментирование снижает восприятие кода, отвлекает от понимания его сути. Писать очевидные вещи — тратить свое и чужое время. Иногда из-за слишком подробных комментариев объем кода увеличивается в несколько раз. Закончив с кодом, посмотрите критически. Очевидные и банальные комментарии удалите.

Чек-лист «Инспекция кода»

Предлагаем чек-лист для самостоятельной проверки чистоты кода. Если вы будете инспектировать чужой код, помните, что программист — творческая профессия. А творческие люди обычно тяжело воспринимают критику. Будьте лояльней.

Итак,

- Легко ли воспринимать код визуально?
- Присутствуют ли комментарии? насколько они необходимы?
- Соответствует ли код стандартам PSR-1 и PSR-2? Краткая выжимка стандартов приведена в разделе “Правила кода PHP”.
- Используете ли вы систему документирования phpDoc или подобную?
- Нужно ли делать перерыв в чтении, чтобы разобраться в написанном?
- Проведен ли рефакторинг?
- Есть ли дублирование в блоках, функциях и пр.?
- Понятны ли названия переменных, имена методов и пр.?
- Какова длина строк, методов, функций, классов, файла?
- Вы искали ошибки и баги?
- Как можно еще улучшить код?
- Можно ли сделать его короче?
- Можно ли сделать его эффективней?

6. ОТЛАДКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ

6.1 Отладка программы

В данной программе производилась отладка с помощью компилятора Visual Studio CODE. В результате выполнения отладки программы были обнаружены и исправлены следующие ошибки: ….(вы их перечисляете)

6.2 Тестирование программы

Проверили функциональность ИС с помощью тест-кейсов (табл. 2).

Таблица 2 – Тест-кейс «Учет товаров»

№ шага	Действие	Ожидаемый результат	Результат теста
1.	Выполнить запуск приложения "Вход в систему"	- Окно "Вход в систему" открыто - На форме 2 поля: Имя и Пароль - Кнопка Вход доступна	пройден / провален/ заблокирован
2.	Открыть базу данных с товарами с помощью меню Файл-Открыть	База данных загрузилась в исходную таблицу	
3.	Ввести запись о новом товаре в исходную таблицу и нажать на кнопку Добавить	Новая запись отобразилась в исходной таблице	
4.	Изменить запись в таблице	Запись в исходной таблице исправлена	
5.	Вывод данных в таблицу расчетов	Данные отобразились в таблице расчетов	
6.	– Печать отчета с помощью команду Файл – Печать таблицы расчетов	Печать отчета выполнена	
7.	Вывод количества товара	Кнопка «Вывод количества товара» выводит количество товаров в таблице	
	…..		
	Сохранить файл с помощью меню Файл-Сохранить	Файл сохранен	
	Выход из формы с помощью меню Файл-Выход	Окно приложения закрылось	

Протестируем программу по заданному тестовому набору.

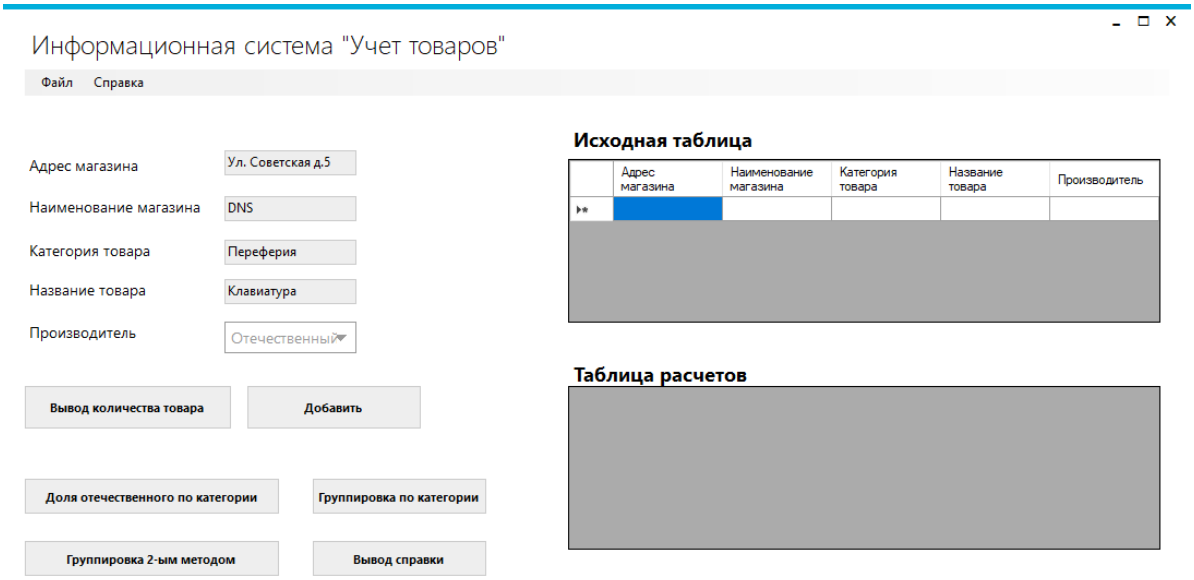

Рисунок 11 - Ввод данных в поля формы проекта

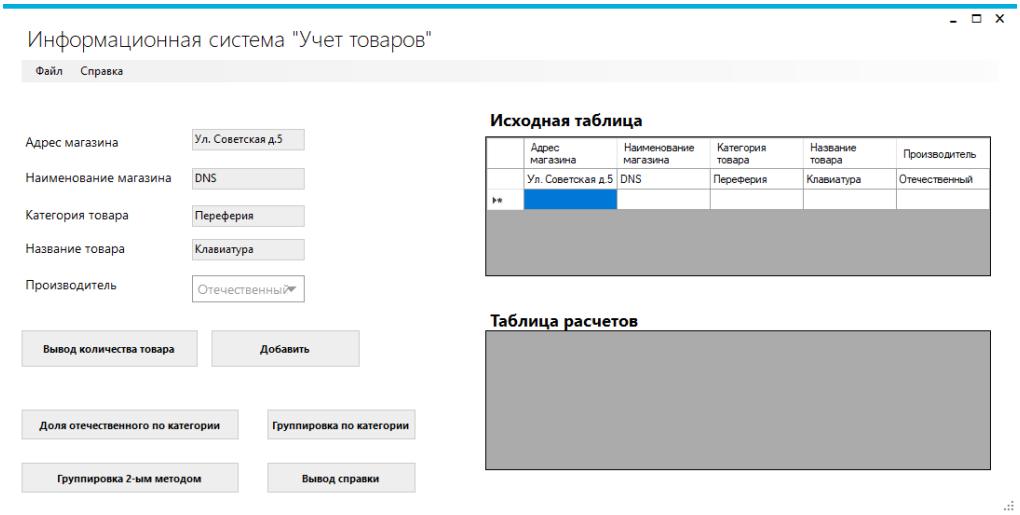

Рисунок 12 - Результат добавления

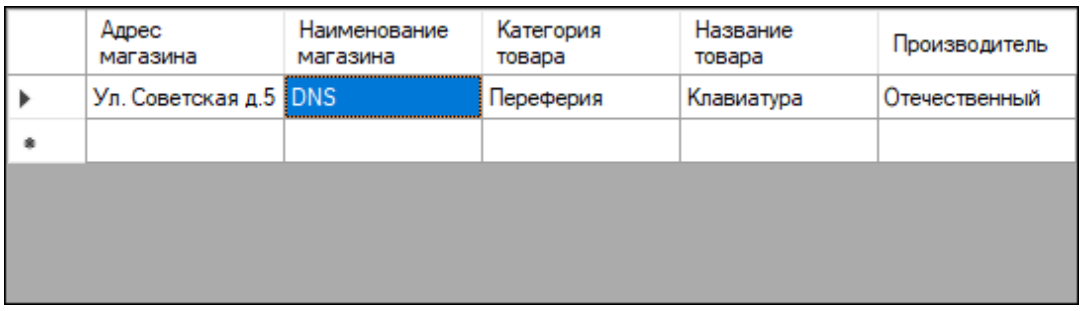

Рисунок 13 - Ввод данных в поля формы проекта

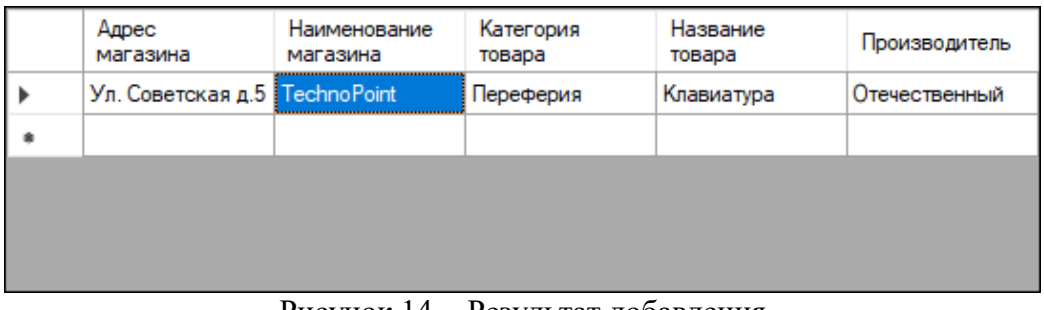

Рисунок 14 - Результат добавления

Исходная таблица

	Адрес магазина	Наименование магазина	Категория товара	Название товара	Производитель
▶	Ул. Советская д.5	TechnoPoint	Переферия	Клавиатура	Отечественный
	Ул. Советская д.5	DNS	Переферия	Клавиатура	Импортный
	Ул. Советская д.5	DNS	Переферия	Клавиатура	Импортный
*					

Таблица расчетов

	Категория товара	Доля отечественных (в %)
▶	Переферия	33
*		

Рисунок 15 - Результат вывода данных в таблицу расчетов

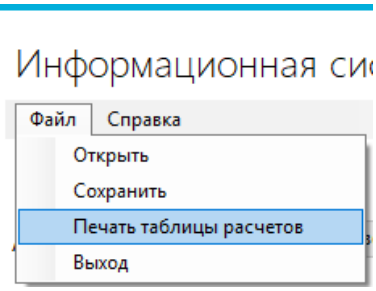

Рисунок 16- Запрос на печать таблицы расчетов

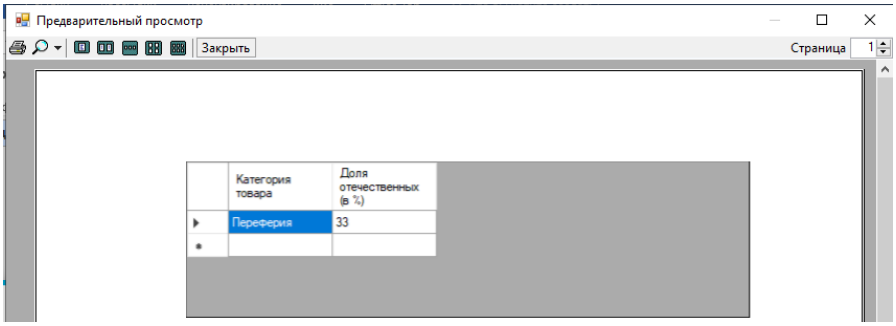

Рисунок 17 - Результат запроса на печать таблицы

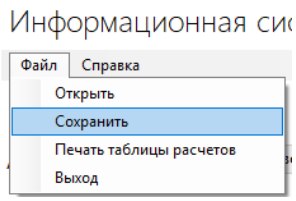

Рисунок 18 - Запрос на сохранение данных

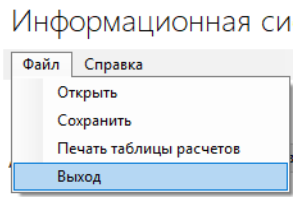

Рисунок 19 - Запрос на выход из формы

6.3 Анализ по результатам отладки и тестирования

Цель проведения испытаний состояла в том, чтобы рассмотреть все возможные варианты работы программы, протестировать ее в нормальных, исключительных и экстремальных условиях, выявить недостатки и устранить их, если таковые имели место.

В результате испытаний на тестовом наборе было доказано, что данная программа работает согласно заданному алгоритму. Все возможные ситуации были рассмотрены, ошибки - устранены.

После успешной компиляции и сборки приложения, непосредственно в процессе тестирования, были обнаружены и устранены ошибки времени исполнения. Из наиболее трудно-устранимых можно отметить ошибку приведения данных типов с плавающей запятой при пользовательском вводе, а также ошибки проектирования пользовательского интерфейса, в связи, с чем последний был целиком переработан несколько раз.

7. СОЗДАНИЕ ФАЙЛОВ ДЛЯ НОСИТЕЛЯ ДАННЫХ

После выполнения проекта необходимо создать репозиторий, из которого по команде **git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY** должен перенести содержимое проекта на диск ПК. В выбранную вами домашнюю директорию. Должен присутствовать дистрибутив для работы в OS Windows и Linux.\

Рекомендации по созданию приложения из кода PHP/

На WINDOWS.

Рекомендуемое решение: Nativefier

Nativefier — это инструмент, который «оборачивает» любой сайт в десктопное

приложение на базе Electron (тот же движок, что у VS Code, Slack и Discord). Он:

• создаёт .exe за 1 команду,

• открывает ваш сайт в чистом окне (без адресной строки),

• поддерживает уведомления, cookies, авторизацию — всё, что работает в браузере,

• позволяет задать иконку, название, размер окна и т.д.

Как собрать .exe из вашего сайта с помощью Nativefier

1. Установите Node.js

• Скачайте с https://nodejs.org (выберите LTS-версию).

• Установите (оставьте все настройки по умолчанию).

2. Установите Nativefier

Откройте Командную строку (cmd) или PowerShell и выполните:

npm install -g nativefier

3. Соберите приложение

Выполните команду (замените https://вашсайт.ру на ваш реальный URL):

npx nativefier "https://вашсайт.ру" --name "Мессенджер" --platform windows --arch x64 --icon "C:\путь\к\иконке.ico"

Если у вас нет иконки — просто уберите --icon:

nativefier "https://вашсайт.ру" --name "Мессенджер" --platform windows --arch x64

4. Найдите готовое приложение

После сборки появится папка вроде:

Мессенджер-win32-x64\

На Linux

Общая стратегия: один код → два приложения

Вы создаёте один и тот же "обёрточный" проект, а затем собираете его под нужную

ОС:

• На Windows → получаете .exe

• На Linux → получаете исполняемый файл (без расширения) или .AppImage / .deb

На Linux (для запуска в Ubuntu/Fedora/etc.)

1. Node.js и npm (версия ≥ 18)

curl -fsSL https://deb.nodesource.com/setup_lts.x | sudo -E bash -

sudo apt-get install -y nodejs

2. Зависимости для Electron (на Ubuntu/Debian):

sudo apt install -y libgtk-3-0 libnotify4 libnss3 libxss1 libxtst6 xdg-utils libatspi2.0-0

libuuid1 libappindicator3-1 libsecret-1-0

Потом уже можно писать команду:

npx nativefier "https://вашсайт.ру" --name "ВАША ИС" --platform linux --arch x64

ВАЖНО! Если хотите использовать иконку для приложения надо конвертировать из png или jpg в icon, сделать это можно на любом сайте, где имеется конвертация или с помощью графического пакета GIMP.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

Основные источники:

1. Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: Учебное пособие / Г. Н. Федорова. – Москва : КУРС : ИНФРА-М, 2022. – 336 с. (Среднее профессиональное образование). – ISBN 9785906818416. – Текст : непосредственный. 3.2.2. Дополнительная литература

2. Григорьев, М. В. Проектирование информационных систем: учебное пособие для среднего профессионального образования / М. В. Григорьев, И. И. Григорьева. — Москва : Издательство Юрайт, 2023. — 318 с. — (Профессиональное образование). — ISBN 978-5- 534-12105-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/518751.

3. Казанский, А. А. Программирование на Visual C# : учебное пособие для среднего профессионального образования / А. А. Казанский. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2023. — 192 с. — (Профессиональное образование). — ISBN 978-5- 534-14130-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/513400

Дополнительные источники:

1. Гохберг, Г. С. Информационные технологии: учебник для образовательных организаций, реализующих программы среднего профессионального образования по специальностям "Информационные системы и программирование", "Сетевое и системное администрирование" / Г. С. Гохберг, А. В. Зафиевский, А. А. Короткин ; Г. С. Гохберг, А. В. Зафиевский, А. А. Короткин. – 3-е изд., стер. - Москва : Академия, 2020. – 240 с. – (Профессиональное образование). – ISBN 9785446886845. – URL: https://academiamoscow.ru/catalogue/4831/471778/. – Текст : электронный.

2. Куприянов, Д. В. Информационное обеспечение профессиональной деятельности : учебник и практикум для среднего профессионального образования / Д. В. Куприянов. — Москва : Издательство Юрайт, 2023. — 255 с. — (Профессиональное образование). — ISBN 978-5-534-00973-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/512863.

3. Проектирование информационных систем : учебник и практикум для среднего профессионального образования / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общей редакцией Д. В. Чистова. — Москва : Издательство Юрайт, 2023. — 258 с. — (Профессиональное образование). — ISBN 978-5-534-03173-7. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/513630.

4. Рудаков, А. В. Технология разработки программных продуктов : учебник для студентов учреждений среднего профессионального образования, обучающихся по специальности "Программное обеспечение вычислительной техники и автоматизированных систем" : [профессиональный модуль ПМ.03 "Участие в интеграции программных модулей" (МДК.03.01)] / А. В. Рудаков ; А. В. Рудаков. – 12-е изд., стер.. – Москва : Академия, 2018. – 208 с. – ISBN 9785446874026. – URL: https://academiamoscow.ru/catalogue/4831/401005/. – Текст :

электронный.

5. Семакин, И. Г. Основы алгоритмизации и программирования : учебник для образовательных организаций, реализующих программы среднего профессионального образования по специальностям "Информационные системы и программирование", "Сетевое и системное администрирование", "Обеспечение информационной безопасности автоматизированных систем", "Обеспечение информационной / И. Г. Семакин, А. П. Шестаков ; И. Г. Семакин, А. П. Шестаков. – 4-е изд., стер. - Москва : Академия, 2020. – 304 с. – (Профессиональное образование). – ISBN 99785446886883. – URL: https://academia-moscow.ru/catalogue/4831/471483/. – Текст : электронный.

6. Сергеев, А. Г. Стандартизация и сертификация : учебник и практикум для среднего профессионального образования / А. Г. Сергеев, В. В. Терегеря. — Москва : Издательство Юрайт, 2023. — 323 с. — (Профессиональное образование). — ISBN 978-5- 534-04315-0. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/511948.

ПРИЛОЖЕНИЕ 1

ПРИМЕРНЫЙ ПЕРЕЧЕНЬ ТЕМ КУРСОВЫХ ПРОЕКТОВ

1. Разработка информационной системы «Справочник военной техники»
2. Разработка информационной системы «Управление автосалоном»
3. Разработка информационной системы «Станция технического обслуживания»
4. Разработка информационной системы «Студенческая группа»
5. Разработка информационной системы «Медицинские услуги»
6. Разработка информационной системы «Регистратура поликлиники»
7. Разработка информационной системы «Справочник военной техники»
8. Разработка информационной системы «Рекламное агентство».
9. Разработка информационной системы «Справочник российских автомобилей»
10. Разработка информационной системы «Управление парикмахерской»
11. Разработка информационной системы «Учёт средств вычислительной техники на предприятии»
12. Разработка информационной системы «Онлайн-платформа для управления комиксов»
13. Разработка информационной системы «Управление системой документооборота ЖКХ»
14. Разработка информационной системы «Управление грузоперевозками»
15. Разработка информационной системы «Управление салоном сотовой связи»
16. Разработка информационной системы «Автопарк»
17. Разработка информационной системы «Учет и управление инвентарем в образовательных учреждениях»
18. Разработка информационной системы «Управление библиотекой»
19. Разработка информационной системы «Справочник военной техники»
20. Разработка информационной системы «Такси»
21. Разработка информационной системы «Автокаталог»
22. Разработка информационной системы «Справочник российских автомобилей»
23. Разработка информационной системы «Управление финаннсовым веб-приложением»
24. Разработка информационной системы «Система СМС-информирования сотрудников на предприятии»
25. Разработка информационной системы «Система управления рестораном и кухней»
26. Разработка информационной системы «Система управления музыкальными мероприятиями»
27. Разработка информационной системы «Система управления ИТ-инфраструктурой»
28. Разработка информационной системы «Справочник российских автомобилей»
29. Разработка информационной системы «Такси»
30. Разработка информационной системы «Автомобильные перевозки»
31. Разработка информационной системы «Автокаталог»

32. Разработка информационной системы «Прокат автомобилей»

33. Разработка информационной системы «Система управления автотранспортом»

34. Разработка информационной системы «Веб-приложение по анализу и прогнозированию финансов»

35. Разработка информационной системы «Отдел продаж»

36. Разработка информационной системы «Система управления рестораном и кухней»

37. Разработка информационной системы «Туристическая фирма»

38. Разработка информационной системы по своей теме (согласовывается руководителем курсового проектирования)

ПРИЛОЖЕНИЕ 2

Титульный лист

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Т. Ф. ГОРБАЧЕВА»
Филиал КузГТУ в г. Белово

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ «РЕГИСТРАТУРА ПОЛИКЛИНИКИ»

Курсовой проект

Выполнил:

Студент группы ИС-225.1

_____ФИО

Руководитель:

Ст. преподаватель

_____Витвицкий М.Н.

Оценка_____

«___»_____20___ г.

Белово
2025

ПРИЛОЖЕНИЕ 3

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«КУЗБАССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Т. Ф. ГОРБАЧЕВА»
Филиал КузГТУ в г. Белово

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

по МДК 05.01 Проектирование и дизайн информационных систем
09.02.07 «Информационные системы и программирование»

Студенту _____
(фамилия, имя, отчество студента)

Группы
Тема _____
Исходные данные: самостоятельная разработка функционала (согласовывается руководителем курсового проектирования)
При выполнении курсового проекта на указанную тему должны быть представлены:

1. Пояснительная записка
2. Цифровой носитель с проектом (вариант для ОС Windows, вариант для ОС Linux)

СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Титульный лист
Задание на курсовой проект
СОДЕРЖАНИЕ
ВВЕДЕНИЕ
РАЗДЕЛ 1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ
ОПИСАНИЕ ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ СОЗДАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ
ТЕХНИЧЕСКОЕ ЗАДАНИЕ (ТЗ ПРИМЕР ПРИЛОЖЕНИЕ 6)
ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ (ПОСТРОЕНИЕ UML-ДИАГРАММ, ФУНКЦИОНАЛЬНОЙ МОДЕЛИ IDEF0)
РАЗДЕЛ 2 ПРАКТИЧЕСКАЯ ЧАСТЬ
РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ И КОДА
Создание меню
Требования к разработке
Создание компонентов
Интегрирование
Назначение программы
ИНСПЕКЦИЯ КОДА

ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММЫ

Отладка программы

Тестирование программы

Анализ по результатам отладки и тестирования

ДОКУМЕНТАЦИЯ ПО РАЗРАБОТКЕ

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЯ

Дата выдачи «__» «_____» 20__г.

Срок выполнения курсового проекта «__» «_____» 20__г.

Преподаватель (руководитель) курсового проекта

_____М.Н. Витвицкий

(подпись, Ф.И.О.)

ПРИЛОЖЕНИЕ 4

СОДЕРЖАНИЕ
(Образец)

Стр.

	ВВЕДЕНИЕ	4
1	**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**	5
1.1	**АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**	5
1.2	**ОПИСАНИЕ ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ СОЗДАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ**	
1.3	**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**	
1.4	**ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ**	
2	**ПРАКТИЧЕСКАЯ ЧАСТЬ**	
2.1	**РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПРИЛОЖЕНИЯ И КОДА**	
2.1.1	**Создание меню**	
2.1.2	**Требования к разработке**	
2.1.3	**Создание компонентов**	
2.1.4	**Интегрирование**	
2.1.5	**Назначение программы**	
2.2	**ИНСПЕКЦИЯ КОДА**	
2.3	**ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММЫ**	
2.3.1	**Отладка программы**	
2.3.2	**Тестирование программы**	
2.3.3	**Анализ по результатам отладки и тестирования**	
2.4	**ДОКУМЕНТАЦИЯ ПО РАЗРАБОТКЕ**	
	ЗАКЛЮЧЕНИЕ	
	СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	
	ПРИЛОЖЕНИЯ	

ПРИЛОЖЕНИЕ 5

ВВЕДЕНИЕ
(Образец)

На рынке программных продуктов имеется множество программ, которые могут быть использованы для автоматизации учета материалов или материальных ценностей на предприятии, в офисе или в учреждении.

Основная особенность специализированных программных продуктов, предназначенных для отслеживания движения материалов и материальных ценностей, предлагаемых на рынке программных продуктов – их многофункциональность и универсальность. Разработчики предлагают универсальные программные продукты без учета особенностей ведения подобного рода деятельности, например, в учебном учреждении.

Часто имеющиеся функции АИС в предлагаемых на рынке программных продуктах избыточны. Их наличие ведет к повышению системных требований для обеспечения функционирования данных систем и к значительному увеличению расходов учреждения на приобретение и эксплуатации данных программных продуктов.

Предполагаем, что разработка узкоспециализированной автоматизированной системы, выполненная с учетом требований специалистов, отслеживающих движение материальных ценностей в филиале и дисциплины учета материальных ценностей, принятой в учебном заведении, позволит упростить процедуру ведения учета, не завышая системные требования и при этом значительно сократив расходы учреждения на приобретение программного продукта.

Целью курсового проекта является создание информационной системы для автоматизированного учета движения мультимедийных средств.

Для достижения цели проекта нужно выполнить следующие задачи:

1. На основании анализа технологического процесса по обслуживанию мультимедийных средств сформулировать требования к информационной системе и составить основные разделы технического задания.

2. В соответствии с техническим заданием выполнить проектирование и разработку информационной системы в выбранной среде программирования.

3. Выполнить тестирование приложения и оформление программной документации.

ПРИЛОЖЕНИЕ 6

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

(Образец фрагмента ТЗ)

Пример выполнения характеристики системы:

Автоматизированная информационная система мультимедийных средств колледжа – это частное решение задачи учета материалов и материальных ценностей.

Разрабатываемая нами автоматизированная информационная система позволяет автоматизировать отдельную функцию общей задачи учета материалов. А именно – учет движения материалов внутри учебного заведения.

В рамках выполнения этой отдельной функции решается частная задача - слежение за движением мультимедийных средств колледжа.

В настоящее время ведется ручная система слежения за движением мультимедийных средств колледжа. Документация о поступлении нового мультимедийного оборудования передается для обработки в бухгалтерию. Приказ о закреплении мультимедийного оборудования за аудиторией издается директором колледжа и оформляется в отделе кадров. Один из техников колледжа ведет журнал, в котором отражаются все сведения о перемещениях оборудования из одной аудитории в другую и о ремонтах этого оборудования.

При регистрации мультимедиа проектора в журнале отражается следующая информация:

- наименование;
- дата выпуска;
- инвентарный номер;
- краткое описание;
- таблица движения оборудования (подотчетные лица, дата, место нахождения);
- таблица ремонтов (вид, дата, отметки о ремонте);
- время наработки лампы (с указанием даты измерения).

При перемещении проектора из аудитории в аудиторию заполняется раздел журнала под наименованием «Движение». При оформлении проектора на ремонт заполняется раздел журнала под наименованием «Ремонт». С некоторой периодичностью отслеживается информация о времени наработки лампы и отмечается в журнале. Кроме этого, ведется учет наличия в аудиториях компьютера, интерактивной доски, акустики и экрана.

Пример описания требований к системе в техническом задании

1 Требования к системе в целом

1.1 Требования к структуре и функционированию системы

В состав ИС «Мультимедиа оборудование в колледже» должны входить следующие подсистемы:

1. Подсистема учета мультимедиа оборудования в колледже
2. Подсистема учета ремонтов
3. Подсистема отчетов

Информационная система «Мультимедиа оборудование в колледже» (в дальнейшем будем называть ее Система) должна быть организована централизованно, на одном компьютере. Режим функционирования – автоматизированный, диалоговый (элементы диалога – экранные формы).

Информация должна вводиться и поддерживаться в актуальном состоянии специалистом, ответственным за учет мультимедиа оборудования.

1.2 Требования к персоналу

Для функционирования и поддержания работоспособности Системы необходим специалист - техник по информационным системам (или информационным технологиям) – 1 шт.ед. Техник по информационным системам должен иметь опыт администрирования СУБД Access, знание и понимание концепций реляционных баз данных.

1.3 Показатели назначения

Система должна обеспечивать возможность исторического хранения данных с глубиной не менее 10 лет.

Система должна обеспечивать возможность одновременной работы только одного пользователя.

Характеристики времени отклика Системы:

– для операций навигации по экранным формам системы – не более 1 сек;

– для операций формирования справок – не более 3 сек.

Время формирования аналитических отчетов определяется их сложностью и может занимать продолжительное время.

1.4 Требования к надежности

Система должна сохранять работоспособность и обеспечивать восстановление своих функций при возникновении следующих внештатных ситуаций:

– при сбоях в системе электроснабжения;

– при ошибках в работе аппаратных средств;

– при ошибках, связанных с системным программным обеспечением.

Потери данных в указанных случаях должны быть минимальными, для чего интервал для автосохранения информации требуется установить не менее 10 минут. Должна быть предусмотрена возможность восстановления данных из резервной копии.

1.5 Требования к безопасности

Система должна включать программные средства для ограничения прав доступа к ней.

1.6 Требования к эргономике

Система должна обеспечивать удобный для пользователей Системы интерфейс, отвечающий следующим требованиям:

– единый стиль оформления для пользовательских интерфейсов;

– должна быть удобная, интуитивно понятная навигация в интерфейсе пользователя;

– взаимодействие пользователя с Системой должно осуществляться на русском языке;

– исключения могут составлять только системные сообщения.

– требуется предусмотреть отображение на экране хода длительных процессов обработки.

Пользовательские интерфейсы Системы должны быть спроектированы и разработаны с применением единых принципов графического представления информации и организации доступа к функциональным возможностям и сервисам. Должен быть разработан графический дизайн пользовательских

интерфейсов, цветовые, шрифтовые и композиционные решения для отображения текстов, изображений, таблиц, гиперссылок, управляющих и навигационных элементов (меню, кнопок, форм и т.п.).

1.7 Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы

Система должна быть рассчитана на эксплуатацию в составе программно–технического комплекса Заказчика.

Техническая и физическая защита аппаратных компонентов системы, носителей данных, бесперебойное энергоснабжение, резервирование ресурсов, текущее обслуживание реализуется техническими и организационными средствами Заказчика.

При вводе системы в опытную эксплуатацию должен быть разработан план выполнения резервного копирования программного обеспечения и обрабатываемой информации. Во время эксплуатации системы, персонал, ответственный за эксплуатацию системы должен выполнять разработанный план.

Размещение помещений и их оборудование должны исключать возможность бесконтрольного проникновения в них посторонних лиц.

Размещение оборудования, технических средств должно соответствовать требованиям техники безопасности, санитарным нормам и требованиям пожарной безопасности.

Все пользователи системы должны соблюдать правила эксплуатации электронной вычислительной техники.

2 Требования к функциям Системы

В Системе должно быть предусмотрено выполнение следующих функций:

1. Учет мультимедиа оборудования в колледже:

1.1. Ввод новых данных о поступившем оборудовании,

1.2. Закрепление оборудования за аудиторией и назначение ответственных лиц

1.3. Оформление передвижения оборудования из аудитории в аудиторию.

1.4. Списание оборудования (удаление информации о нем).

2. Ведение справочной информации о проекторах разных моделей:

2.1. При поступлении проектора новой модели добавление информации в справочник.

2.2. При обнаружении ошибочных или измененных данных, изменение информации в справочнике.

2.3. В случае завершения использования морально устаревшего оборудования удаление информации из справочника.

3. Учет ремонтов:

3.1. отметка начала ремонта,

3.2. отметка о завершении ремонта,

3.3. слежение за ресурсом лампы проектора.

4. Поиск информации:

4.1. Определение местонахождения мультимедиа оборудования по инвентарному номеру

4.2. Определение местонахождения мультимедиа оборудования по типу оборудования.

5. Подготовка отчетов.

3 Требования к видам обеспечения

3.1 Требования к информационному обеспечению

Хранение и актуализация данных о мультимедиа проекторах, необходимых для осуществления операции регистрации проектора. Система должна позволять вводить и корректировать следующие данные о мультимедиа проекторе в карточку проектора:

- модель проектора;
- инвентарный номер проектора;
- дата выпуска;
- время наработки лампы;
- дата измерения времени наработки лампы;
- отметка об отправлении проектора в ремонт.

Корректировка справочника проекторов производится, если регистрируется новая модель проектора. Система должна позволять вводить и корректировать следующие данные о мультимедиа проекторе в справочник проектора:

- наименование модели проектора;
- максимальная яркость;
- контрастность;
- разрешение;
- ресурс лампы;
- изображение модели (фотография);
- технология.

Система должна позволять вводить и корректировать следующие данные регистрации ремонта проектора:

- тип неисправности;
- дата начала ремонта;
- дата окончания ремонта;
- отметка об устранении неисправности;
- замечания.

Хранение и актуализация данных, необходимых для осуществления операций закрепления мультимедиа проекторов, интерактивных досок, экранов и акустических систем за аудиториями. Система должна позволять вводить и корректировать следующие данные:

- номер аудитории;
- фамилия и инициалы подотчетного лица;
- инвентарный номер проектора;
- отметки о наличии в аудитории интерактивных досок, экранов и акустических систем и компьютеров.

3.2 Требования к программному обеспечению

Для размещения данных Системы и ее программного обеспечения необходима СУБД Access 2007 и выше.

Для управления Системой должна использоваться операционная система Microsoft Windows 10 и выше, Astra Linux.

Прикладное программное обеспечение в составе Системы должно соответствовать следующим основным требованиям:

- функционировать в среде операционной системы и взаимодействовать с СУБД в соответствии с требованиями настоящего ТЗ;
- поддерживать русский и английский языки, символы кириллицы и латиницы; иметь удобный пользовательский интерфейс;

– реализовывать экспорт данных в текстовом формате;

– реализовывать формирование и вывод печатных отчетных форм;

– обеспечивать реализацию всех функций Системы в соответствии с требованиями настоящего ТЗ;

– иметь комплект пользовательской документации на русском языке.

Качество разработки программных средств должно обеспечиваться соответствующими процедурами управления проектом по реализации Системы.

3.3 Требования к техническому обеспечению

Состав технические средства, функционирующие в составе Системы:

1. Персональный компьютер

1.1. Центральный процессор класса Intel Core 2 Duo 2.13 ГГц

1.2 Объём оперативной памяти не менее 2 Gb

1.3 Минимальная ёмкость жесткого диска 80 Gb

1.3 Стандартный SVGA монитор

1.4 Клавиатура, манипулятор «мышь»

1.5 Лазерный принтер для печати отчетных форм

3.4 Требования к организационному обеспечению

В целях обеспечения штатного функционирования Системы необходимо наличие подразделений, производящих техническое и программное обслуживание компонент Системы.

Требуется наличие должностных инструкций, регламентирующих порядок использования Системы и разграничивающих права ее использования. Должно быть разработано руководство пользователя системы.

Пример листинга

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using MetroFramework.Components;
using MetroFramework.Forms;
namespace Курсовая_Учет
{
   public partial class Form1 : MetroForm //Главная форма
   {
      public string[] arr = new string[100];
      public string[] podschet = new string[100];
      public float col = 0, imp = 0, otech = 0;
      int raz = 1;
      public Form1()
      {
         InitializeComponent();
      }
      private void найтиToolStripMenuItem_Click(object sender, EventArgs e)
      {
      }
      private void открытьToolStripMenuItem_Click(object sender, EventArgs e)
      {
         Stream mystr = null;
         if (openFileDialog1.ShowDialog() == DialogResult.OK)
         {
            if ((mystr = openFileDialog1.OpenFile()) != null)
            {
               StreamReader myread = new StreamReader(mystr);
               string[] str;
               int num = 0;
               try
               {
                  string[] str1 = myread.ReadToEnd().Split(' ');
                  num = str1.Count();
                  dataGridView1.RowCount = num;
                  for (int i = 0; i < num; i++)
                  {
                     str = str1[i].Split('^');
                     for (int j = 0; j < dataGridView1.ColumnCount; j++)
```

```
                        {
                            try
                            {
                                dataGridView1.Rows[i].Cells[j].Value = str[j];
                            }
                            catch { }
                        }
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                    myread.Close();
                }
            }
        }
    }
    private void сохранитьToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Stream myStream;
        if (saveFileDialog1.ShowDialog() == DialogResult.OK)
        {
            if ((myStream = saveFileDialog1.OpenFile()) != null)
            {
                StreamWriter myWriter = new StreamWriter(myStream);
                try
                {
                    for (int i = 0; i < dataGridView1.RowCount - 1; i++)
                    {
                        for (int j = 0; j < dataGridView1.RowCount; j++)
                        {

myWriter.Write(dataGridView1.Rows[i].Cells[j].Value.ToString() + '^');
                        }
                        myWriter.WriteLine();
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                    myWriter.Close();
                }
                myStream.Close();
```

```
            }
        }
    }
    private void metroButton1_Click(object sender, EventArgs e) //Кнопка
Добавить
    {
        int pov = 0;
        string adr = metroTextBox1.Text;
        string name = metroTextBox2.Text;
        string categ = metroTextBox3.Text;
        string nazv = metroTextBox4.Text;
        string proiz = metroComboBox1.Text;
        dataGridView1.Rows.Add(adr, name, categ, nazv, proiz);
        //ввод данных
        if (proiz == "Импортный") imp += 1;
        else otech ++;          col ++;
        //подсчет количества, импорт и отеч
        Array.Resize<string>(ref arr, raz);
        Array.Resize<string>(ref podschet, raz);
        for (int i = 0; i < raz; i++)
            if (arr[i] == categ) pov += 1;
                if (pov == 0)
                {
                    arr[raz - 1] = categ;
                    raz ++;
                }           // категория товара без повторения
    }
    private void metroButton2_Click(object sender, EventArgs e)
   //Кнопка Вывод количества товара
    {
        float dolImp, dolOtech;
        dataGridView2.Columns.Clear();
        dataGridView2.Columns.Add("Column1", "Общее количество товара" );
        dataGridView2.Columns.Add("Column2", "Количество импортных" );
        dataGridView2.Columns.Add("Column3", "Доля импортных (в %)" );
        dataGridView2.Columns.Add("Column4", "Доля отечественных (в %)" );
        dolImp = imp/col*100;
        dolOtech = otech / col*100;
        dataGridView2.Rows.Add(col, imp, String.Format("{0:0}", dolImp),
String.Format("{0:0}", dolOtech));
    }
    private void metroButton3_Click(object sender, EventArgs e)
        //Кнопка Доля отечественного по категории
    {
        float schetchic = 0, otr = 0 ,dolya = 0;
        dataGridView2.Columns.Clear();
        dataGridView2.Columns.Add("Column1", "Категория товара");
        dataGridView2.Columns.Add("Column2", "Доля отечественных (в %)");
        for (int i = 0; i < raz-1; i++)
```

```
                {
                    schetchic = 0; otr = 0;
                    for (int j = 0; j < dataGridView1.RowCount - 1; j++)
                    {
        if (arr[i] == (string)dataGridView1.Rows[j].Cells[2].Value)
        if ((string)dataGridView1.Rows[j].Cells[4].Value == "Отечественный")
schetchic += 1;
                        else otr += 1;
                    }
                     dolya = schetchic / (schetchic + otr)*100;
                     podschet[i] = String.Format("{0:0}",dolya);
                }
                    for (int i = 0; i < raz - 1; i++)
                {  dataGridView2.Rows.Add(arr[i]);
                    dataGridView2.Rows[i].Cells[0].Value = arr[i];
                    dataGridView2.Rows[i].Cells[1].Value = podschet[i];
                }
            }
            private void справкаToolStripMenuItem_Click(object sender, EventArgs e)
            {
       MessageBox.Show("Информационная система «Учет товаров»
\nРазработчик:….",
            "Справка");
            }
            private void metroButton4_Click(object sender, EventArgs e)
      // Кнопка Группировка по категории
            {
                dataGridView2.Columns.Clear();
                dataGridView2.Columns.Add("Column1", "Адрес магазина");
                dataGridView2.Columns.Add("Column2", "Наименование магазина");
                dataGridView2.Columns.Add("Column3", "Категория товара");
                dataGridView2.Columns.Add("Column4", "Название товара");
                dataGridView2.Columns.Add("Column5", "Производитель");
                for (int i = 0; i < dataGridView1.RowCount - 1; i++)
                {
                    dataGridView2.Rows.Add();
                    for (int j = 0; j < dataGridView1.ColumnCount ; j++)
                    {
                        dataGridView2.Rows[i].Cells[j].Value =
dataGridView1.Rows[i].Cells[j].Value;
                    }
                 }
                this.dataGridView2.Sort(this.dataGridView2.Columns["Column3"],
ListSortDirection.Ascending);
                for (int i = 0; i < dataGridView2.RowCount-1 ; i++)
                {
                    for (int j = 1; j < dataGridView2.RowCount ; j++)
                    {
                     if ( i != j && (string)dataGridView2.Rows[i].Cells[2].Value ==
```

```
(string)dataGridView2.Rows[j].Cells[2].Value) dataGridView2.Rows[j].Cells[2].Value
= " ";
                    }
                }
            }
            private void metroButton5_Click(object sender, EventArgs e)
        //Кнопка Группировка 2-ым методом
            {
                int stolb = 0;
                int razmer = 2;
                string[] mass = new string[10];
             //   string[] znach = new string[5];
                int per=0;
                dataGridView2.Columns.Clear();
                dataGridView2.Columns.Add("Column1", "Категория товара");
           //       dataGridView2.Columns.Add("Column2", "Наименование товара");
                mass[0] = (string)dataGridView1.Rows[0].Cells[2].Value;
                for (int i = 1; i < dataGridView1.RowCount - 1; i++)
                {
                    per = 0;
                    for (int j = 0; j < razmer-1; j++)
                    {
                      if (mass[j] == (string)dataGridView1.Rows[i].Cells[2].Value) per++;
                    }
                    if (per == 0)
                    {
                        mass[razmer-1] = (string)dataGridView1.Rows[i].Cells[2].Value;
                        razmer ++;     Array.Resize<string>(ref mass, razmer);
                    }           }          //вывод без повторения
                for (int i = 0; i < razmer-1; i++)
                    dataGridView2.Rows.Add(mass[i]);
                for (int i = 0; i < razmer-1; i++)
                {    per = 1;
                    for (int j = 0; j < dataGridView1.RowCount-1; j++)           {
                        if (mass[i] == (string)dataGridView1.Rows[j].Cells[2].Value)
                        {
                            dataGridView2.Columns.Add("Column" + per, "Наименование
товара");
      dataGridView2.Rows[i].Cells[per].Value = dataGridView1.Rows[j].Cells[1].Value;
                            per++;
                        }
                    }
                }
                for (int i = 0; i < dataGridView2.RowCount - 1; i++)
                {
                     for (int j = 1; j < dataGridView2.ColumnCount ; j++)
                    {
                        for (int z = 1; z < dataGridView2.ColumnCount ; z++)
                        {
```

```
            if (j != z && (string)dataGridView2.Rows[i].Cells[j].Value ==
(string)dataGridView2.Rows[i].Cells[z].Value) dataGridView2.Rows[i].Cells[z].Value =
" ";
                        }
                    }
                    stolb = 2;
                    for (int j = 2; j < dataGridView2.ColumnCount-1 ; j++)
                    {    if ((string)dataGridView2.Rows[i].Cells[j].Value != " ")
                     {  dataGridView2.Rows[i].Cells[stolb].Value =
(string)dataGridView2.Rows[i].Cells[j].Value;
                        dataGridView2.Rows[i].Cells[j].Value = " ";     stolb++;
                    }               }          }
                //for (int i = 0; i < dataGridView2.RowCount - 1; i++)
                //{
                //    for (int j = 0; j < dataGridView2.ColumnCount - 1; j++)
                //    {
                //        if ((string)dataGridView2.Rows[i].Cells[j].Value == " ")
dataGridView2.Columns.RemoveAt(j);
                //    }
                //}
            }
            private void metroButton6_Click(object sender, EventArgs e)
        // Вывод справки
            {
                float otch = 0, imp = 0, vsego = 0;
                dataGridView2.Columns.Clear();
                dataGridView2.Columns.Add("Column1", "Название товара");
                dataGridView2.Columns.Add("Column1", "Процент отечественных");
                dataGridView2.Columns.Add("Column1", "Процент импортных");
               for (int i = 0; i < dataGridView1.RowCount - 1; i++)
              //      for (int j = 0; j < dataGridView1.ColumnCount - 1; j++)
                    {
                if ((string)dataGridView1.Rows[i].Cells[4].Value == "Отечественный")
otch+=1;
                    else imp+=1;
                    }
                vsego = otch + imp;
                imp = imp / vsego*100;
                otch = otch / vsego*100;
                for (int i = 0; i < dataGridView1.RowCount - 1; i++)
                {
                    dataGridView2.Rows.Add();
                    dataGridView2.Rows[i].Cells[0].Value =
(string)dataGridView1.Rows[i].Cells[3].Value;
                    dataGridView2.Rows[i].Cells[1].Value = String.Format("{0:0}", otch);
                    dataGridView2.Rows[i].Cells[2].Value = String.Format("{0:0}", imp);
                }
            }
            private void printDocument1_PrintPage(object sender,
```

```
System.Drawing.Printing.PrintPageEventArgs e)
            {
                Bitmap objbmp = new Bitmap(this.dataGridView2.Width,
this.dataGridView2.Height);
                dataGridView2.DrawToBitmap(objbmp, new Rectangle(0,0,
this.dataGridView2.Width, this.dataGridView2.Height));
                e.Graphics.DrawImage(objbmp, 150, 90);
            }

            private void печатьТаблицыРасчетовToolStripMenuItem_Click(object
sender, EventArgs e)
            {
                printPreviewDialog1.Document = printDocument1;
                printPreviewDialog1.ShowDialog();
            }
           }
```

public partial class Form2 : MetroForm //Форма Справка

```
   {
      public Form2()
      {
         InitializeComponent();
      }

      private void сохранитьToolStripMenuItem_Click(object sender, EventArgs e)
      {
      }
      private void metroButton1_Click(object sender, EventArgs e)
      {
                  Form1 imp = this.Owner as Form1;
         Form1 col = this.Owner as Form1;
         Form1 dolImp = this.Owner as Form1;
         Form1 dolOtech = this.Owner as Form1;
         dataGridView1.Columns.Clear();
         dataGridView1.Columns.Add("Column1", "Общее количество товара");
         dataGridView1.Columns.Add("Column2", "Количество импортных");
         dataGridView1.Columns.Add("Column3", "Доля импортных в %");
         dataGridView1.Columns.Add("Column4", "Доля отечественных в %");
         dataGridView1.Rows.Add(col, imp, String.Format("{0:0.00}", dolImp),
String.Format("{0:0.00}", dolOtech));
      }
      private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
      {
      }
   }}
```

ПРИЛОЖЕНИЕ 8

Инструкции : Запустите эмулятор TransferSimulator.exe. Если окно не закрывается — значит, программа запущена успешно.

Чтобы запустить TransferSimulator.jar, откройте командную строку в папке, содержащей этот файл, и выполните команду: java -jar TransferSimulator.jar.

Причины возможных проблем с запуском:

- Отсутствие Java на компьютере.

- Неправильно настроены переменные среды JAVA_HOME и PATH.

Если не получается запустить ни один файл, тогда открывается интернет на рабочих месте участника с доступом только одной ссылки http://prb.sylas.ru/TransferSimulator/fullName.

Для тестирования API запустите Postman. Выберите метод HTTP- запроса GET из выпадающего списка слева от поля URL. Укажите URL на основании документа api_info.pdf дописав в конце строки метод (http://localhost:4444/TransferSimulator/fullName). Нажмите кнопку «Send». В результате в нижней области появится ответ эмулятора в виде JSON (если ответ не получили значит у Вас закрыт эмулятор или Вы допустили ошибки в URL).

Инструкции :

Чтобы в Windows работал TransferSimulator.exe нужно настроить переменные окружения JAVA_HOME и PATH.

Для этого:

Найти путь к установленному JDK: открыть «Проводник» и перейти в папку C:\Program Files\Java, внутри этой директории найти папку с установленной версией, открыть её и скопировать полный путь.

Открыть «Панель управления», перейти в «Система» → «Дополнительные параметры», нажать «Переменные». В разделе «Системные переменные» нажать «Создать».

В поле «Имя» ввести JAVA_HOME, в поле «Значение» вставить скопированный путь.

Нажать «ОК».

Для настройки PATH в том же окне «Переменные среды» найти переменную Path, нажать «Изменить» → «Создать» и вставить путь к папке с

исполняемыми файлами Java: %JAVA_HOME%\bin.

После изменений рекомендуется перезагрузить компьютер, чтобы они вступили в силу.

Запустите TransferSimulator.exe. Если окно не закрывается — значит, программа запущена успешно.

Составитель
Витвицкий Максим Николаевич

Методические указания по выполнению курсового проекта
для студентов очной формы обучения
по направлению специальности
09.02.07 «Информационные системы и программирование»

Публикуется в авторской редакции