Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Институт информационных технологий

Кафедра «Микропроцессорные системы и сети»

К защите допустить

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_\_ Кашникова И.В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ**

**НА ТЕМУ:**

**«Автоматизированная система  
расчета стоимости промышленных изделий»**

Студент Салов А. А. И.О. Фамилия

(дата, подпись)

Руководитель Боброва Н. Л. И.О. Фамилия

(дата, подпись)

МИНСК 2021

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»**

**Институт информационных технологий**

**Кафедра Микропроцессорные системы и сети**

Специальность I-40 01 73 – Программное обеспечение информационных систем

*УТВЕРЖДАЮ*

Зав. кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.О. Фамилия

(Подпись, Ф.И.О.)

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**ЗАДАНИЕ**

**по курсовому проектированию**

Слушателю *Фамилия Имя Отчество*

1. Тема проекта *Автоматизированная система расчета стоимости промышленных изделий*

2. Сроки сдачи слушателем законченного проекта *00 января 2021 года*

3. Исходные данные к проекту: *Приложение должно быть выполнено на языке Visual C#, графический интерфейс приложения должен содержать меню, панель инструментов, строку статуса и диалоговые окна. В приложении должны быть предусмотрены функции ввода и редактирования связанных данных. Реализовать проверку корректности введенных значений. Информация об ошибках, а также сервисную информацию (время запуска, время завершения программы) выводить в отдельный текстовый лог-файл приложения. Организовать справку по работе с программой.*

4. Перечень подлежащих разработке вопросов:

*Введение (с указанием целей и основных задач для ее достижения).*

*1. Описание предметной области и определение требований к системе с точки зрения предметной области.*

*2. Постановка задачи и обзор методов ее решения.*

*3. Модели представления системы и их описание.*

*4. Информационная модель системы и ее описание (с доказательством приведения ее к 3-ей нормальной форме).*

*5. Описание алгоритмов, реализующих бизнес-логику проектируемой системы.*

*6. Руководство пользователя.*

*7. Результаты тестирования разработанной системы и оценка выполнения задач.*

*Заключение.*

*Список использованных источников.*

*Приложения.*

*Пояснительную записку оформить в соответствии с ГОСТ 2.105-95, алгоритмы – по ГОСТ 19.701-90 (ИСО 5807-85) «Схемы алгоритмов, данных, программ и систем»,*

5. Перечень графического материала (с точным указанием обязательных чертежей и графиков):

*1. Отбор документов затрат. Схема программы, ф. А3, чертеж*

*2. Расчет себестоимости. Схема программы, ф. А3, чертеж*

*3. Диаграмма вариантов использования (Use Case), ф. А3, плакат*

6. Дата выдачи задания  *00 месяца 2020 г.*

7. Календарный график работы над проектом на весь период проектирования

*00 месяца 2020 года – п.п. 4.1, 4.2 (20%)*

*00 месяца 2020 года – п.п. 4.1, 4.2, 4.3 (40%)*

*00 месяца 2020 года – п.п. 4.1, 4.2, 4.3, 4.4 (60%)*

*00 месяца 2020 года – п.п. 4.1, 4.2, 4.3, 4.4, 4.5 (80%)*

*00 месяца 2020 года – п.п. 4, 5 (100%)*

Руководитель (подпись, Ф.И.О., должность)

Задание принял к исполнению (дата и подпись слушателя)

Содержание

[Введение 4](#_Toc60233466)

[1 Описание предметной области и определение требований к системе с точки зрения предметной области 5](#_Toc60233467)

[2 Постановка задачи и обзор методов ее решения 10](#_Toc60233468)

[2.1 Постановка задачи 10](#_Toc60233469)

[2.2 Выбор технологий реализации проекта 12](#_Toc60233470)

[3 Модели представления системы и их описание 15](#_Toc60233471)

[3.1 Функциональная модель системы 15](#_Toc60233472)

[3.2 Структура системы 21](#_Toc60233473)

[4 Информационная модель системы и ее описание 26](#_Toc60233474)

[5 Описание алгоритмов, реализующих бизнес-логику проектируемой системы 30](#_Toc60233475)

[6 Руководство пользователя 36](#_Toc60233476)

[7 Результаты тестирования разработанной системы и оценка выполнения задач 43](#_Toc60233477)

[Заключение 49](#_Toc60233478)

[Список использованных источников 50](#_Toc60233479)

[Приложение (обязательное) Листинг программного кода 52](#_Toc60233480)

Введение

Контроль производства и управление затратами на предприятиях направлено на поиск резервов сокращения затрат, минимизацию непроизводительных расходов, оптимизацию состава и структуры затрат, что в конечном счете приводит к повышению эффективности деятельности. Процесс расчета себестоимости предполагает достаточно сложные и объемные вычисления. Автоматизация расчета себестоимости позволяет не только повысить точность и упростить процедуры расчета затрат, но и выявить проблемные области планирования и формирования себестоимости.

На сегодняшний день существует достаточное число программных средств как обособленных, так и в составе систем управленческого и бухгалтерского учета, позволяющих автоматизировать процесс расчета себестоимости. Однако они зачастую не удовлетворяют всем требованиям, предъявляемым к контролю производства и расчету себестоимости.

Актуальность курсового проекта заключается в необходимости разработки программного средства расчета себестоимости, удовлетворяющего всем предъявляемым требованиям.

Цель настоящего проекта: повышение эффективности процесса расчета себестоимости за счет разработки и внедрения средства автоматизации.

Для достижения поставленной цели необходимо решить следующие задачи:

* определить требования к системе с точки зрения предметной области;
* выполнить постановку задачи и определить методы ее решения;
* разработать модели представления системы;
* разработать информационную модель системы;
* разработать и реализовать алгоритмы, реализующие бизнес-логику системы;
* провести тестирование разработанной системы.

Объект исследования: процесс расчета себестоимости.

Предмет исследования: программное средство расчета себестоимости.

# Описание предметной области и определение требований к системе с точки зрения предметной области

Себестоимость продукции – это сумма всех издержек, которые несет предприятие по изготовлению товара и его последующей реализации.

Под издержками понимаются затраты, связанные с закупкой сырья и материалов, необходимых для производства, оплатой труда работников, транспортировкой, хранением и реализацией готовой продукции [1].

В себестоимость продукции (работ, услуг) включаются затраты, непосредственно связанные с производством и реализацией продукции, затраты на обеспечение предприятия рабочей силой (персоналом), отчисления в различные фонды и бюджеты, затраты на содержание, обслуживание и управление организацией, платежи сторонним организациям за оказание услуг по производству продукции, затраты на подготовку и освоение новой продукции, прочие затраты [2].

В зависимости от объема затрат себестоимость подразделяется на технологическую, сокращенную, производственную и полную (см. рисунок 1.1).



Рисунок 1.1 – Виды и структура себестоимости

Технологическая себестоимость состоит из прямых производственных затрат на первичных участках производства. Информация о сокращенной производственной себестоимости обобщается в управленческом калькуляционном учете. Производственная и полная себестоимости формируются расчетным путем [3].

Калькуляционные статьи для общей калькуляции представляют собой развернутую номенклатуру из нескольких десятков статей, перечень которых существенно зависит от отраслевых особенностей производства, целей и задач калькулирования, определяемых руководством организации [4].

Калькуляционная единица – это количественная мера конкретного объекта калькулирования. Если калькуляционный объект выражает весь объем продукции данного вида, то калькуляционная единица – только ту его часть, которая применяется для индивидуальной калькуляции и характеризует индивидуальную себестоимость данного вида продукции [5].

Все калькуляционные единицы можно свести к нескольким типологическим группам: натуральные единицы (штуки, килограммы и т.п.), условно-натуральные единицы (пара обуви, тонна литья и т.п.), приведенные или условные калькуляционные единицы (условный ящик стекла и т.п.), эксплуатационные единицы (единица мощности машин, параметры приборов и т.п.), единицы работы персонала или средств труда (нормо-час работы, машино-дни, тонно-километр перевозок и т.п.) [6].

При расчете себестоимости могут применятся различные методы, зависящие от вида калькуляции.

Под методом учета затрат на производство и калькулирование себестоимости продукции понимают совокупность приемов и способов документирования и отражения производственных затрат, которые обеспечивают исчисление фактической себестоимости продукции, а также отнесения издержек на единицу продукции.

В зависимости от оперативности учета и контроля затрат различают метод учета фактических затрат и метод учета нормативных затрат, а в зависимости от объектов учета затрат – попроцессный, попередельный и позаказный методы учета затрат. С точки зрения полноты учитываемых затрат различают калькулирование полной себестоимости и калькулирование неполной (усеченной) себестоимости. На практике эти методы применяются в различных сочетаниях [7].

Информация о том, на основе каких данных и для каких целей рассчитываются разные виды калькуляций, приведена на рисунке 1.2.



Рисунок 1.2 – Особенности расчета разных видов калькуляций

В зависимости от оперативности учета и контроля затрат различают следующие методы:

* метод учета фактических затрат – при использовании этого метода калькуляция производится на основе фактических данных о количестве и стоимости затраченных ресурсов;
* метод учета нормативных затрат – данный метод использует в расчете себестоимости нормативные, средние значения потребления и учетных цен того или иного ресурса [8].

Методы калькулирования делят на: индивидуальные и массовые. В свою очередь, к индивидуальным методам относят позаказный метод калькулирования, к массовым – попередельный и попроцессный методы калькулирования себестоимости продукции [9].

Попередельный метод калькулирования применяется в производствах, в которых готовые продукты получают в результате последовательного ряда переделов (например, в химическом производстве), в каждом из них возникают полупродукты (полуфабрикаты). Аналитический калькуляционный учет организуется по каждому переделу. Конечная себестоимость продукта определяется как сумма себестоимости всех переделов, в которых он проходил обработку, изготовление [10].

Попроцессный метод калькулирования применяют в производствах с ограниченной номенклатурой, в которых отсутствует незавершенное производство или в которых оно является стабильным, не изменяющимся из месяца в месяц (например, нефтехимическое производство, транспортные перевозки и т.п.). Калькуляционный аналитический учет издержек производства разграничивают на отдельные процессы изготовления по видам продукции, выполнения работ и услуг [11].

Позаказный метод калькулирования себестоимости отличается от других индивидуализированным определением себестоимости. В аналитическом калькуляционном учете издержки производства группируются по заказам, оформляющим единичное изделие или группу одинаковых изделий – их малую серию или партию. Первичный учет затрат организуется по отдельному заказу, при этом каждое изделие или их серия строго отграничены в рамках данного заказа от других, организация производства строится на выполнении конкретных заказов, что позволяет калькулировать себестоимость изделия, их серии совершенно независимо от других заказов, находящихся в производстве. Производственный процесс не надо делить по временным периодам для локализации затрат. Этот метод учета используется в строительстве, машиностроении, научно-исследовательских и опытно-конструкторских работах [12].

В настоящем проекте рассматривается расчет себестоимости по следующей методике: номенклатура продукции разбивается на группы; учет затрат может осуществляться в разрезе подразделений, номенклатурных групп, продукции; любая затрата относится к определенной статье затрат; распределение затрат на выпущенную продукцию осуществляется пропорционально плановой цене.

Себестоимость рассчитывается раздельно для каждого подразделения.

Количество произведенной продукции за период для регистра «Структура себестоимости» определяется по следующей формуле:

, (.)

где *Qi* – количество продукции *i*, произведенной за период;

*j* – документ, отражающий выпуск продукция *i*;

*n* – общее число документов;

*qij* – количество продукции *i*, произведенной по документу *k*.

Плановая стоимость выпуска определяется по формуле:

, (.)

где *Pi* – плановая стоимость выпуска продукции *i* за период;

*Qi* – количество продукции *i*, произведенной за период;

*pi* – плановая цена продукции *i*.

Коэффициент распределения затрат определяется по формуле:

, (.)

где *Ri* – коэффициент распределения затрат для продукции *i*;

*Pi* – плановая стоимость выпуска продукции *i*;

*n* – число единиц номенклатуры продукции.

Сумма фактических затрат на выпуск номенклатуры продукции определяется по формуле:

, (.)

где *Si* – сумма затрат на выпуск продукции *i*;

*Ri* – коэффициент распределения затрат для продукции *i*;

*т* – число затрат;

*cj* – сумма затрат на номенклатурную группу продукции.

# Постановка задачи и обзор методов ее решения

Постановка задачи

Целью проекта является повышение эффективности процесса расчета себестоимости за счет разработки и внедрения средства автоматизации.

Назначение программного средства: учет выпуска продукции, учет затрат на выпуск и расчет себестоимости выпуска.

Расчет себестоимости осуществляется с учетом следующих особенностей: номенклатура продукции разбивается на группы; учет затрат может осуществляться в разрезе подразделений, номенклатурных групп, продукции; любая затрата относится к определенной статье затрат; распределение затрат на выпущенную продукцию осуществляется пропорционально плановой цене.

Перечень ролей пользователей программного средства приведен в таблице 2.1.

Таблица . – Роли пользователей системы

|  |  |
| --- | --- |
| Пользователь | Описание роли пользователя |
| Диспетчер производства | Сотрудник предприятия, отвечающий за функцию регистрации выпуска продукции |
| Бухгалтер | Сотрудник предприятия, отвечающий за функцию регистрации затрат |
| Экономист | Сотрудник предприятия, отвечающий за расчет себестоимости, формирование структуры затрат, калькуляции и их анализ |
| Администратор | Администратор системы, имеющий полный доступ ко всем данным и функциям системы |

Программное средство должно обеспечивать реализацию следующих функций для перечисленных в таблице 2.1 пользователей:

* редактирование справочников;
* регистрация выпуска;
* регистрация затрат;
* регистрация плановых цен;
* расчет себестоимости выпуска;
* формирование отчетов.

Перечень полномочий пользователей приведен в таблице 2.2.

Таблица . – Полномочия пользователей

|  |  |
| --- | --- |
| Роль пользователя | Описание полномочий пользователя, работающего с программным средством |
| Диспетчер производства | Справочник номенклатуры: чтение, просмотр  Справочник номенклатурных групп: чтение, просмотр  Справочник подразделений: чтение, просмотр  Документ выпуска: редактирование |
| Бухгалтер | Справочник номенклатуры: редактирование  Справочник номенклатурных групп: редактирование  Справочник подразделений: редактирование  Справочник единиц измерения: редактирование  Справочник статей затрат: чтение, просмотр  Документ выпуска: просмотр  Документ затрат: редактирование  Отчет по себестоимости: формирование |
| Экономист | Справочник номенклатуры: чтение, просмотр  Справочник номенклатурных групп: чтение, просмотр  Справочник подразделений: чтение, просмотр  Справочник единиц измерения: чтение, просмотр  Справочник статей затрат: редактирование  Документ выпуска: чтение, просмотр  Документ затрат: чтение, просмотр  Документ установки плановых цен: редактирование  Документ расчета себестоимости: редактирование  Отчет по распределению затрат: формирование  Отчет по структуре себестоимости: формирование  Калькуляция: формирование  Отчет по себестоимости: формирование |
| Администратор | Справочник номенклатуры: редактирование  Справочник номенклатурных групп: редактирование  Справочник подразделений: редактирование  Справочник единиц измерения: редактирование  Справочник статей затрат: редактирование  Справочник пользователей: редактирование  Документ выпуска: редактирование  Документ затрат: редактирование  Документ установки плановых цен: редактирование  Документ расчета себестоимости: редактирование  Отчет по распределению затрат: формирование  Отчет по структуре себестоимости: формирование  Калькуляция: формирование  Отчет по себестоимости: формирование |

Диспетчер производства имеет доступ к просмотру основных справочников и редактированию документа выпуска.

Бухгалтер имеет доступ к редактированию основных справочников (справочник статей затрат в режиме просмотра), просмотру документа выпуска, редактированию документа затрат, формированию отчета по себестоимости.

Экономист имеет доступ к просмотру основных справочников, (справочник статей затрат в режиме редактирования), просмотру документов выпуска и отражения затрат, редактированию документов установки плановых цен и расчета себестоимости, формированию всех отчетов.

Администратор имеет доступ ко всем данным и функциям программного средства: редактирование основных справочников, редактирование справочника пользователей, редактирование документов, формирование отчетов.

## Выбор технологий реализации проекта

Среди технологий, применяемых при разработке приложений рабочего стола для операционной системы Windows, наиболее распространенными в настоящее время являются:

* Java Platform, Enterprise Edition (J2EE);
* DOT.NET Framework.

Java Platform, Enterprise Edition (Java EE) – набор спецификаций и соответствующей документации для языка Java, описывающей архитектуру серверной платформы для задач средних и крупных предприятий. Сервер приложений Java EE – это реализация системы в соответствии со спецификацией Java EE, обеспечивающая работу модулей с логикой конкретного приложения [13].

DOT.NET Framework – программная технология от компании Microsoft, предназначенная для создания обычных программ и веб-приложений. Одной из основных идей .NET является совместимость различных служб, написанных на разных языках. Подобно технологии Java, среда разработки .NET создает байт-код, предназначенный для исполнения виртуальной машиной [14].

Сравнение перечисленных платформ приведено в таблице 2.3.

На основании таблицы 2.3 в качестве платформы реализации выбрана .NET по причине более высокой производительности и поддержки языков программирования.

Таблица . – Сравнение платформ разработки клиентских приложений

|  |  |  |
| --- | --- | --- |
| Критерий | Java Platform | Microsoft DOT.NET Framework |
| Поддержка ООП | полная | полная |
| Тип разрабатываемых приложений | любое | любое |
| Отказоустойчивость | высокая | высокая |
| Поддерживаемый язык программирования | Java | C#, VB, C++ и др. |
| Производительность | средняя | высокая |

Наиболее распространенными средами разработки под C# являются: Microsoft Visual Studio – официальная интегрированная среда разработки приложений на платформе .NET [15] и SharpDevelop – свободная интегрированная среда разработки [16].

Сравнение перечисленных IDE приведено в таблице 2.4.

Таблица . – Сравнение IDE для разработки

|  |  |  |
| --- | --- | --- |
| Критерий | Visual Studio | SharpDevelop |
| Подсветка синтаксиса | да | да |
| Визуальный редактор WPF и Windows Forms | да | частично |
| Интегрированный отладчик | да | да |
| Расширяемость внешними инструментами | да | нет |

По преимуществам, перечисленным в таблице 2.4, выбрана интегрированная среда разработки Visual Studio.

При реализации приложений на базе платформы .NET применяются следующие СУБД: Microsoft SQL Server, MySQL, PostgreSQL [17, 18, 19].

Сравнение приведенных СУБД приведено в таблице 2.13.

Таблица . – Сравнительный анализ СУБД

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | MS SQL | MySQL | PostgreSQL |
| Кроссплатформенность (Windows, Linux) | + / – | + / + | + / + |
| Ограничение на использование RAM, GB | 64 | не ограничено | не ограничено |
| Ограничение на использование CPU, количество ядер | 4 | не ограничено | не ограничено |
| Лицензия | Пропиетарная | GPL | GPL |
| Скорость работы | Высокая | Высокая | Средняя |
| Поддержка в Visual Studio | да | нет | нет |

Таким образом, в сравнении с конкурентами СУБД Microsoft SQL Server имеет преимущество, что определяет ее выбор для реализации проекта.

Также приняты следующие проектные решения:

* интерфейс клиенткой части приложения выполнен с использованием технологии WPF [20], позволяющей использовать паттерн проектирования MVC;
* модель данных реализована с использованием технологии Entity Framework, позволяющей вести как прямое, так и обратное проектирование базы данных и классов модели данных, а также использовать средства языка запросов LINQ to Object [21].

# Модели представления системы и их описание

Функциональная модель системы

Содержание функций системы приведено в таблице 3.1.

Таблица . – Содержание функций

|  |  |
| --- | --- |
| Функция | Содержание функции |
| 1 | 2 |
| Авторизоваться | Авторизация позволяет получить доступ к системе. Авторизация осуществляется в диалоговом окне, выводящимся при запуске программного обеспечения системы. Для авторизации необходимо выбрать пользователя в выпадающем списке, в текстовом поле указать пароль и нажать Ok. |
| Редактировать справочник номенклатуры | Редактирование справочника номенклатуры осуществляется в форме списка. В списке можно отобрать номенклатуру по группе. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Редактировать справочник номенклатурных групп | Редактирование справочника номенклатурных групп осуществляется в форме списка. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Редактировать справочник подразделений | Редактирование справочника подразделений осуществляется в форме списка. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Редактировать справочник статей затрат | Редактирование справочника статей затрат осуществляется в форме списка. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Редактировать справочник единиц измерения | Редактирование справочника единиц измерения осуществляется в форме списка. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Редактировать справочник пользователей | Редактирование справочника пользователей осуществляется в форме списка. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. Функция доступна пользователю с ролью «Администратор». |

Продолжение таблицы 3.1

|  |  |
| --- | --- |
| 1 | 2 |
| Регистрировать затраты | Регистрация затрат осуществляется в форме списка. В списке предусмотрен отбор по дате и основным реквизитам. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Регистрировать выпуск | Регистрация выпуска осуществляется в форме списка. В списке предусмотрен отбор по дате и основным реквизитам. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Регистрировать плановые цены | Регистрация плановых цен осуществляется в форме списка. В списке предусмотрен отбор по дате и основным реквизитам. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При удалении выводится запрос подтверждения удаления. |
| Регистрировать расчет себестоимости | Регистрация расчетов себестоимости осуществляется в форме списка. В списке предусмотрен отбор по дате и основным реквизитам. Пользователю доступны операции добавления, изменения, удаления. Добавление и изменение осуществляется в форме редактирования. При сохранении осуществляется расчет себестоимости. При удалении выводится запрос подтверждения удаления. |
| Формировать отчет по распределению затрат | Формирование распределения затрат осуществляется в форме отчета. Отчет формируется за определенный период. Отчет выводится в формате html. Предусмотрена вывод отчета на принтер. |
| Формировать структуру себестоимости | Формирование структуры себестоимости осуществляется в форме отчета. Отчет формируется за определенный период. Отчет выводится в формате html. Предусмотрена вывод отчета на принтер. |
| Формировать калькуляцию | Формирование структуры себестоимости осуществляется в форме отчета. Отчет формируется за определенный период. Отчет выводится в формате html. Предусмотрена вывод отчета на принтер. |
| Формировать отчет по себестоимости | Формирование отчета по себестоимости осуществляется в форме отчета. Отчет формируется за определенный период. Отчет выводится в формате html. Предусмотрена вывод отчета на принтер. |
| Просмотреть справку | Просмотр справки осуществляется в виде электронного документа, который выводится в специально отведенной панели. |
| Просмотреть логи | Просмотр логов осуществляется в форме списка. Функция доступна пользователю с ролью «Администратор». |

С системой взаимодействуют следующие актеры:

* экономист;
* бухгалтер;
* администратор;
* диспетчер производства.

Диаграмма вариантов использования системы, иллюстрирующая функции экономиста, приведена на рисунке 3.1.



Рисунок 3.1 – Диаграмма вариантов использования актера «Экономист»

Экономисту доступны следующие варианты использования:

* авторизоваться;
* просмотреть справку;
* редактировать справочник статей затрат;
* регистрировать плановые цены;
* регистрировать расчет себестоимости;
* формировать отчет по распределению затрат;
* формировать отчет по структуре себестоимости;
* формировать калькуляцию;
* формировать отчет по себестоимости.

Диаграмма вариантов использования системы, иллюстрирующая функции диспетчера производства, приведена на рисунке 3.2.

Диспетчеру производства доступны следующие вариант использования:

* авторизоваться;
* регистрировать выпуск.



Рисунок 3.2 – Диаграмма вариантов использования актера  
«Диспетчер производства»

Диаграмма вариантов использования системы, иллюстрирующая функции бухгалтера, приведена на рисунке 3.3.

Бухгалтеру доступны следующие варианты использования:

* авторизоваться;
* просмотреть справку;
* редактировать справочник номенклатуры;
* редактировать справочник номенклатурных групп;
* редактировать справочник подразделений;
* редактировать справочник единиц измерения;
* регистрировать затраты;
* формировать отчет по себестоимости.

Диаграмма вариантов использования системы, иллюстрирующая функции администратора, приведена на рисунке 3.4.



Рисунок 3.3 – Диаграмма вариантов использования актера «Бухгалтер»



Рисунок 3.4 – Диаграмма вариантов использования актера «Администратор»

Администратор связан с другими актерами отношением обобщения, которое показывает, что администратору доступны все варианты использования связанных актеров. Также администратору доступны следующие варианты использования:

* авторизоваться;
* просмотреть логи
* редактировать справочник пользователей.

Действия для выполнения определенной функции распределяются между человеком и компьютером по тому или иному компоненту системы на основе сравнительного анализа человека и техники на предмет возможности и эффективности ее выполнения ими.

Распределение функций между человеком и техническими устройствами (системой) приведено в таблице 3.2.

Таблица . – Распределение функций между пользователем и системой

|  |  |  |
| --- | --- | --- |
| Название функции | Пользователь | Система |
| 1 | 2 | 3 |
| Авторизоваться | Выбирает в списке пользователя, водит пароль | Проверяет данные пользователя |
| Редактировать справочник номенклатуры | Вводит данные справочника номенклатуры | Сохраняет данные справочника номенклатуры |
| Редактировать справочник номенклатурных групп | Вводит данные справочника номенклатурных групп | Сохраняет данные справочника номенклатурных групп |
| Редактировать справочник подразделений | Вводит данные справочника подразделений | Сохраняет данные справочника подразделений |
| Редактировать справочник статей затрат | Вводит данные справочника статей затрат | Сохраняет данные справочника статей затрат |
| Редактировать справочник единиц измерения | Вводит данные справочника единиц измерения | Сохраняет данные справочника единиц измерения |
| Редактировать справочник пользователей | Вводит данные справочника пользователей | Сохраняет данные справочника пользователей |
| Регистрировать затраты | Вводит данные затрат | Сохраняет данные затрат |
| Регистрировать выпуск | Вводит данные выпуска продукции | Сохраняет данные выпуска продукции |
| Регистрировать плановую цену | Вводит данные плановых цен | Сохраняет данные плановых цен |

Продолжение таблицы 3.2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Регистрировать расчет себестоимости | Вводит данные расчета себестоимости | Рассчитывает себестоимость и сохраняет данные себестоимости |
| Формировать отчет по распределению затрат | Вводит параметры отчета | Отбирает данные, выводит сформированный отчет |
| Формировать структуру себестоимости | Вводит параметры отчета | Отбирает данные, выводит сформированный отчет |
| Формировать калькуляцию | Вводит параметры отчета | Отбирает данные, выводит сформированный отчет |
| Формировать отчет по себестоимости | Вводит параметры отчета | Отбирает данные, выводит сформированный отчет |
| Просмотреть справку | Выбирает раздел справки | Выводит страницу со справочной информацией |
| Просмотреть логи | Выбирает раздел | Выводит информацию из лог-файла |

Перечисленные варианты использования составляют функциональную спецификацию требований к программному средству.

## Структура системы

Структура системы в виде диаграммы классов приведена на рисунке 3.5.

На диаграмме представлены следующие классы и пакеты:

* App – запускаемый класс приложения .net wpf;
* Login – форма авторизации;
* Main – главная форма;
* dbContext – класс контекста данных;
* Model – пакет классов модели данных;
* Pages – пакет классов страниц wpf;
* ReportHtmlHelper – вспомогательный класс для формирования отчетов в формате html;
* Helps – пакет классов страниц wpf со справочной информацией.

Класс App включает следующие атрибуты и методы:

* db – объект контекста данных;
* path – путь к файлу с логами;
* user – объект авторизованного пользователя;
* GetMD5() – возвращает md5-хеш строки;
* SetLogText() – записывает строку в файл с логами.



Рисунок 3.5 – Диаграмма классов

Класс Login содержит метод BtnLogin\_Click() – обработчик нажатия на кнопку «Вход».

Методы класса Main инициализируют загрузку страниц пользовательского интерфейса в область просмотра страниц главной формы.

Диаграмма классов страниц интерфейса, входящих в пакет Pages, приведена на рисунке 3.6.



Рисунок 3.6 – Диаграмма классов страниц интерфейса

На диаграмме приведены следующие классы страниц:

* ClassesList – список номенклатурных групп;
* ClassesForm – форма номенклатурной группы;
* CostsList – список расчетов себестоимости;
* CostsForm – форма расчета себестоимости;
* CostsCalc – форма операций расчета себестоимости;
* DepartmentsList – список подразделений;
* DepartmentsForm – форма подразделения;
* ExpendituresList – список статей затрат;
* ExpendituresForm – форма статьи затрат;
* ExpensesList – список затрат;
* ExpensesForm – форма затраты;
* LogFile – список записей файла логов;
* MeasuresList – список единиц измерения;
* MeasuresForm – форма единицы измерения;
* NomenclatureList – список номенклатуры;
* NomenclatureForm – форма номенклатуры;
* OutputsList – список выпуска;
* OutputsForm – форма выпуска;
* PlansList – список плановых цен;
* PlansForm – форма плановой цены;
* UsersList – список пользователей;
* UsersForm – форма пользователя.

Перечисленные классы представляют собой страницы WPF.

Диаграмма классов модели данных, входящих в пакет Model, приведена на рисунке 3.7.

На диаграмме приведены следующие классы:

* Role – роль пользователя;
* User – пользователь;
* Measure – единица измерения;
* Class – номенклатурная группа;
* NomenclatureType – тип номенклатуры;
* Nomenclature – номенклатура;
* Plan – плановая цена;
* Output – выпуск;
* Department – подразделение;
* Expense – затрата;
* Expenditure – статья затрат;
* Cost – расчет себестоимости;
* Structure – структура себестоимости;
* Calculation – калькуляция себестоимости.



Рисунок 3.7 – Диаграмма классов модели данных

Классы модели данных являются основанием для разработки таблиц базы данных

# Информационная модель системы и ее описание

Для взаимодействия с базой данных предполагается использовать технологию Entity Framework и стратегию Code First, позволяющую сгенерировать базу данных на основании классов модели данных.

Схема реляционной базы данных, отражающая таблицы, соответствующие классам модели данных, приведена на рисунке 4.1.



Рисунок 4.1 – Схема базы данных

Схема, приведенная на рисунке 3.5, показывает таблицы реляционной базы данных, их поля и типы данных, первичные и внешние ключи, а также связи между таблицами. Описание таблиц реляционной базы данных в формате Microsoft SQL Server, сгенерированной на основании модели данных, приведено в таблицах 4.1-4.14.

Таблица . – Таблица User (пользователь)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(20) |  | Наименование |
| Password | nvarchar(32) |  | Пароль |
| RoleId | int | FK1 | Код роли |
| Enabled | bit |  | Признак активности |

Таблица . – Таблица Calculation (калькуляция себестоимости)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| StructureId | int | FK1 | Код структуры |
| ExpenditureId | int | FK2 | Код статьи затрат |
| NomenclatureId | int | FK3 | Код номенклатуры |
| Summa | money |  | Сумма |

Таблица . – Таблица Structure (структура себестоимости)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| CostId | int | FK1 | Код расчета себестоимости |
| NomenclatureId | int | FK2 | Код номенклатуры |
| DepartmentId | int | FK3 | Код подразделения |
| Quantity | decimal(15;3) |  | Количество |

Таблица . – Таблица Plan (плановая цена)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Date | datetime |  | Дата |
| NomenclatureId | int | FK1 | Код номенклатуры |
| Price | money |  | Цена |

Таблица . – Таблица Nomenclature (номенклатура)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(50) |  | Наименование |
| NomenclatureTypeId | int | FK1 | Код типа номенклатуры |
| ClassId | int | FK2 | Код номенклатурной группы |
| Description | nvarchar(100) |  | Описание |

Таблица . – Таблица Cost (расчет себестоимости)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Date | datetime |  | Дата |
| UserId | int | FK1 | Код пользователя |

Таблица . – Таблица Output (выпуск продукции)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Date | datetime |  | Дата |
| NomenclatureId | int | FK1 | Код номенклатуры |
| DepartmentId | int | FK2 | Код подразделения |
| Quantity | decimal(15;3) |  | Количество |

Таблица . – Таблица Expense (затрата)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Date | datetime |  | Дата |
| NomenclatureId | int | FK1 | Код номенклатуры |
| Summa | money |  | Сумма |
| ExpenditureId | int | FK2 | Код статьи затрат |
| DepartmentId | int | FK3 | Код подразделения |
| ClassId | int | FK4 | Код номенклатурной группы |

Таблица . – Таблица Expenditure (статья затрат)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(50) |  | Наименование |

Таблица . – Таблица Department (подразделение)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(20) |  | Наименование |

Таблица . – Таблица NomenclatureType (тип номенклатуры)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(20) |  | Наименование |

Таблица . – Таблица Measure (единица измерения)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(10) |  | Наименование |

Таблица . – Таблица Class (номенклатурная группа)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(50) |  | Наименование |
| MeasuerId | int | FK1 | Код единицы измерения |

Таблица . – Таблица Role (роль)

|  |  |  |  |
| --- | --- | --- | --- |
| Поле таблицы | Тип данных | Признак ключа | Описание |
| Id | int | PK | Код |
| Name | nvarchar(20) |  | Наименование |

Разработанная база данных находится в третьей нормальной форме.

* таблицы базы данных не содержат повторяющихся столбцов и столбцов, содержащих наборы значений;
* каждый столбец таблицы, не являющийся ключом, зависит от ключа (полная функциональная зависимость);
* каждый столбец, не являющийся ключом, зависит только от ключа (отсутствие транзитивной зависимости).

# Описание алгоритмов, реализующих бизнес-логику проектируемой системы

Ниже приведены алгоритмы, реализующие типовые задачи, решаемые программным средством: обработка авторизации, обработка удаления объекта, обработка сохранения объекта, отбор документов, расчет себестоимости, формирование отчета.

Блок-схема алгоритма обработки авторизации приведена на рисунке 5.1.



Рисунок 5.1 – Блок-схема алгоритма обработки авторизации

Приведенная блок-схема описывает алгоритм процедуры, выполняющейся при нажатии кнопки «Ok» в форме авторизации, и предполагает следующий порядок действий:

* осуществляется проверка, выбран ли пользователь;
* если пользователь не выбран, выводится сообщение «Необходимо выбрать пользователя», алгоритм завершается;
* если пользователь выбран, алгоритм продолжается;
* из списка формы извлекается выбранный объект;
* из поля ввода формы извлекается пароль;
* осуществляется проверка, совпадают ли пароли;
* если пароли не совпадают, выводится сообщение «Неверный пароль», алгоритм завершается;
* если пароли совпадают, алгоритм продолжается;
* скрывается форма авторизации;
* открывается окно главной формы в модальном режиме;
* после закрытия главной формы закрывается форма авторизации, алгоритм завершается.

Блок-схема алгоритма обработки удаления объекта приведена на рисунке 5.2. Приведенная блок-схема описывает алгоритм процедуры, выполняющейся при нажатии кнопки «Удалить» в форме списка справочников и документов, и предполагает следующий порядок действий:

* осуществляется проверка, выбрана ли строка таблицы;
* если строка не выбрана, выводится сообщение «Не выбран объект для удаления», алгоритм завершается;
* если строка выбрана, алгоритм продолжается;
* из таблицы извлекается выбранный объект;
* осуществляется проверка, есть ли на объект ссылки;
* если ссылки есть, выводится сообщение «Нельзя удалить объект, т.к. на него имеются ссылки», алгоритм завершается;
* если ссылок нет, алгоритм продолжается;
* запрашивается подтверждение удаления;
* если подтверждение не получено, алгоритм завершается;
* если подтверждение получено, алгоритм продолжается;
* объект удаляется из контекста данных;
* если удаление успешно, таблица с данными обновляется, иначе – выводится сообщение «Ошибка удаления», алгоритм завершается.



Рисунок 5.2 – Блок-схема алгоритма обработки удаления объекта

Блок-схема алгоритма обработки сохранения объекта приведена на рисунке 5.3. Приведенная блок-схема описывает алгоритм процедуры, выполняющейся при нажатии кнопки «Сохранить» в форме редактирования справочников и документов, и предполагает следующий порядок действий:

* осуществляется проверка, новый ли это объект;
* если объект новый, то объект добавляется к контексту данных;
* если объект существует, то объект обновляется в контексте данных;
* изменения сохраняются в контексте данных;
* осуществляется проверка, успешно ли сохранение;
* если сохранение успешно, осуществляется переход к странице списка объектов, алгоритм завершается;
* если сохранение не удалось, отменяются изменения в контексте данных, выводится сообщение «Ошибка сохранения», алгоритм завершается.



Рисунок 5.3 – Блок-схема алгоритма обработки сохранения объекта

Блок-схема алгоритма отбора документов затрат приведена на соответствующем чертеже (РТДП 000000.000.01. Отбор документов затрат. Схема программы). Приведенная блок-схема описывает алгоритм процедуры, выполняющейся при нажатии кнопки «Применить» в форме списка документов затрат, и предполагает следующий порядок действий:

* создается пустой список документов затрат;
* из контекста данных выбираются все документы затрат в созданный список документов затрат;
* если используется отбор по дате и дата начала меньше даты окончания, список документов затрат фильтруется по периоду, задаваемому датой начала и датой окончания;
* если используется отбор по номенклатуре и номенклатура выбрана, из поля списка извлекается номенклатура, по которой фильтруется список документов затрат;
* если используется отбор по статье затрат и статья затрат выбрана, из поля списка извлекается статья затрат, по которой фильтруется список документов затрат;
* сформированный список документов затрат устанавливается источником данных для табличной части формы.

Блок-схема алгоритма расчета себестоимости приведена на соответствующем чертеже (РТДП 000000.000.02. Расчет себестоимости. Схема программы). Приведенная блок-схема описывает алгоритм процедуры, выполняющейся при загрузке формы расчета себестоимости, и предполагает следующий порядок действий:

* осуществляется проверка, выбран ли документ расчета себестоимости, переданный в форму в качестве параметра;
* если документ не выбран, то осуществляется переход к странице списка расчетов себестоимости, алгоритм завершается;
* если документ выбран, алгоритм продолжается;
* удаляются результаты расчета себестоимости по документу;
* проверяется наличие выпуска;
* если выпуска нет, выводится сообщение «Не зарегистрированы документы выпуска», алгоритм завершается;
* если выпуск есть, выводится сообщение «Проверка выпуска пройдена», алгоритм продолжается;
* получается номенклатура выпуска;
* проверяется наличие плановых цен на номенклатуру выпуска;
* если плановые цены не установлены, выводится сообщение «Не установлены плановые цены», алгоритм завершается;
* если выпуск есть, выводится сообщение «Проверка выпуска пройдена», алгоритм продолжается;
* получается список затрат;
* проверяются подразделения выпуска;
* если подразделения выпуска и затрат не совпадают, выводится сообщение «Подразделения выпуска и затрат не совпадают», алгоритм завершается;
* если подразделения совпадают, алгоритм продолжается;
* получаются номенклатурные группы затрат;
* проверяются номенклатурные группы;
* если номенклатурные группы выпуска и затрат не совпадают, выводится сообщение «Номенклатурные группы выпуска и затрат не совпадают», алгоритм завершается;
* если номенклатурные группы совпадают, выводится сообщение «Проверка списанных затрат пройдена», алгоритм продолжается;
* формируется список выпуска по подразделениям с ценами и количеством;
* формируется сводный список затрат;
* формируется список выпуска по подразделениям и группам;
* распределяются затраты на продукцию;
* распределение затрат сохраняется в контексте данных;
* сохраняются изменения в контексте данных;
* осуществляется проверка, успешно ли сохранение;
* если сохранение успешно, выводится сообщение «Расчет себестоимости успешно завершен», алгоритм завершается;
* если сохранение не удалось, отменяются изменения в контексте данных, выводится сообщение «Не удалось сохранить результаты расчета себестоимости», алгоритм завершается.

Программный код, реализующий рассмотренные алгоритмы, приведен в Приложении.

# Руководство пользователя

Начало работы с системой для пользователя начинается с запуска приложения. В результате откроется форма авторизации, приведенная на рисунке 6.1.

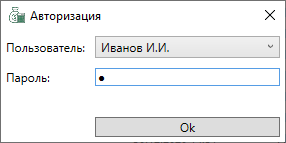


Рисунок 6.1 – Форма авторизации

Для получения доступа к функциям системы необходимо выбрать пользователя в выпадающем списке, указать пароль пользователя и нажать кнопку «Ok». В тестовой версии для всех пользователей установлен пароль «1».

Все основные функции системы доступны в главном окне, приведенном на рисунке 6.2. Главное окно является контейнером для страниц редактирования справочников, регистрации документов, формирования отчетов.

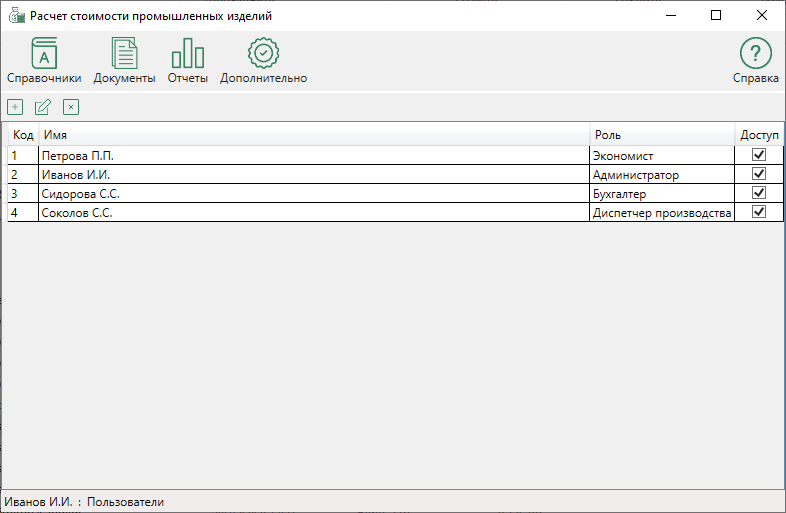


Рисунок 6.2 – Форма главного окна приложения

Доступ к функциям осуществляется с помощью главного меню, включающего функции: справочники, документы, отчеты. Подпункты главного меню соответствуют справочниками, документам, отчетам и позволяют перейти к соответствующей функции.

Для редактирования справочника пользователь должен выбрать пункт меню «Справочники» и подпункт, соответствующий справочнику. В результате откроется форма списка элементов справочника, приведенная на рисунке 6.3.

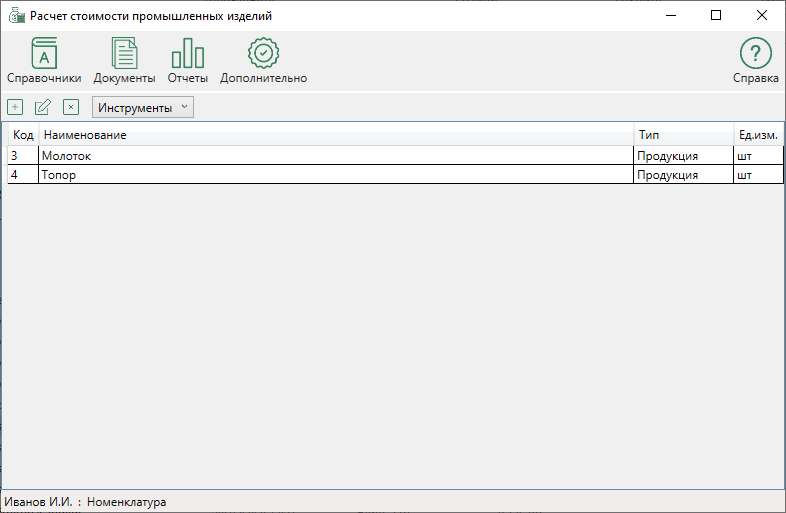


Рисунок 6.3 – Форма списка справочника

Форма списка включает панель инструментов для выбора операции (добавить, изменить, удалить) и табличную часть с реквизитами справочника. Справочник номенклатуры дополнительно включает список выбора номенклатурной группы, к которой принадлежат элементы справочника.

Форма редактирования справочника приведена на рисунке 6.4.

Для добавления элемента справочника необходимо в форме списка справочника нажать кнопку «Добавить», далее – в форме редактирования указать данные элемента справочника и нажать «Сохранить». Кнопка «Отмена» позволяет отменить операцию.

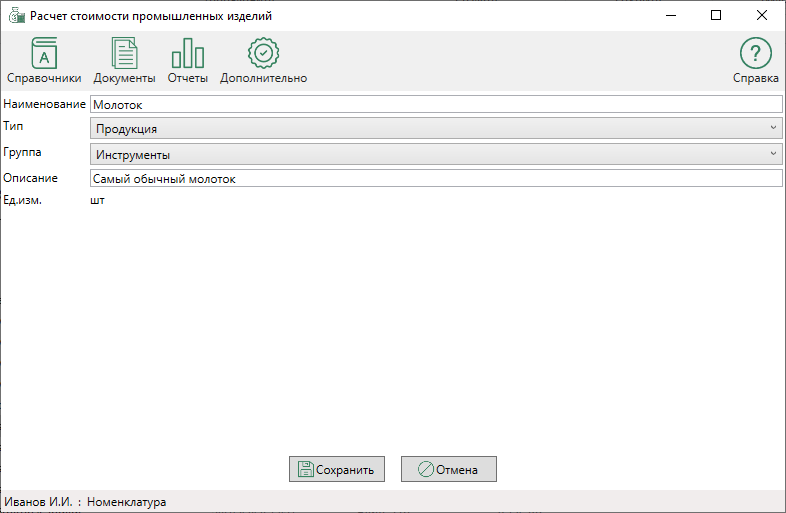


Рисунок 6.4 – Форма редактирования справочника

Для изменения элемента справочника необходимо выбрать его в форме списка, нажать кнопку «Изменить», далее – в форме редактирования указать данные элемента справочника и нажать «Сохранить». Кнопка «Отмена» позволяет отменить операцию.

Для удаления элемента справочника необходимо выбрать его в форме списка, нажать кнопку «Удалить». Если удаление элемента справочника ведет к нарушению целостности данных, выводится сообщение о невозможности удаления (рисунок 6.5).

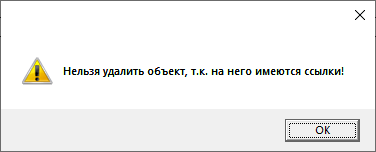


Рисунок 6.5 – Сообщение о невозможности удаления объекта

Если объект возможно удалить, в диалоге подтверждения (рисунок 6.6) следует нажать «Да». Кнопка «Нет» в диалоге позволяет отменить операцию.

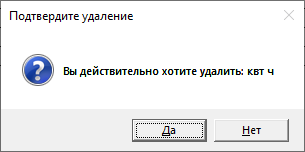


Рисунок 6.6 – Форма подтверждения удаления

Указанным образом осуществляется редактирование всех справочников.

Для регистрации документов пользователь должен выбрать пункт меню «Документы» и подпункт, соответствующий документу. В результате откроется форма списка документов, приведенная на рисунке 6.7. Форма списка включает панель инструментов для выбора операции (добавить, изменить, удалить, отбор), поля отбора и табличную часть с реквизитами справочника.

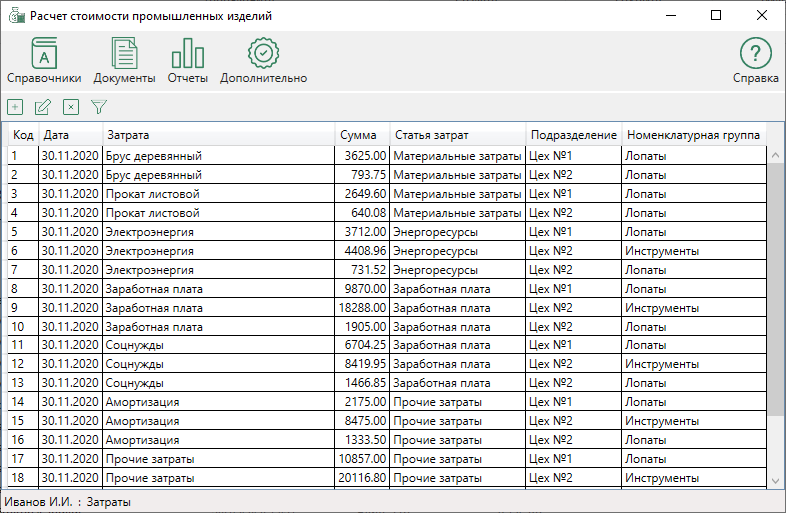


Рисунок 6.7 – Форма списка документов

Для отбора документов необходимо в форме списка нажать кнопку «Отбор», далее – в полях фильтра указать значения отбора и нажать кнопку «Применить». Кнопка «Сбросить» позволяет сбросить условия отбора и сформировать полный список документов.

Форма редактирования документа приведена на рисунке 6.8.

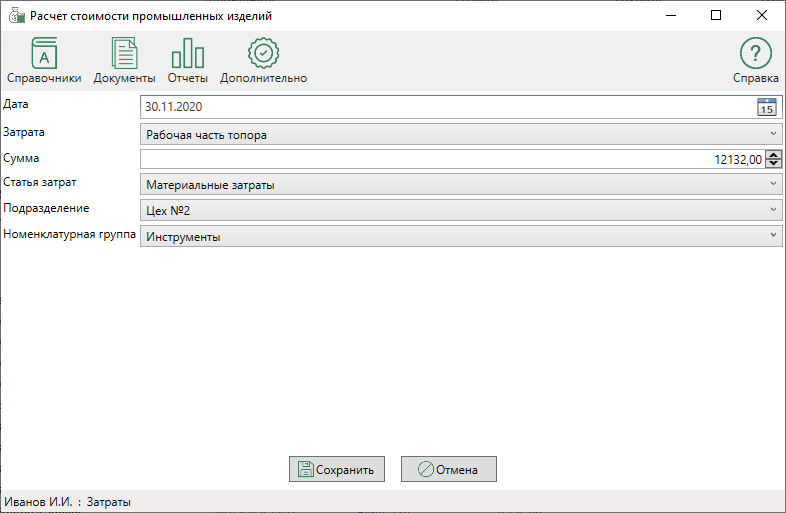


Рисунок 6.8 – Форма редактирования документа

Для добавления документа необходимо в форме списка документов нажать кнопку «Добавить», далее – в форме редактирования указать данные документа и нажать «Сохранить». Кнопка «Отмена» позволяет отменить операцию.

Для изменения документа необходимо выбрать его в форме списка, нажать кнопку «Изменить», далее – в форме редактирования указать данные документа и нажать «Сохранить». Кнопка «Отмена» позволяет отменить операцию.

Для удаления документа необходимо выбрать его в форме списка, нажать кнопку «Удалить», далее – в диалоге подтверждения нажать «Да». Кнопка «Нет» в диалоге позволяет отменить операцию.

Указанным образом осуществляется регистрация всех документов.

При сохранении документа «Расчет себестоимости» открывается страница расчета себестоимости, в которой отображается ход расчета себестоимости по этапам: проверка выпуска продукции, проверка плановых цен, проверка списанных затрат, расчет себестоимости (рисунок 6.9).

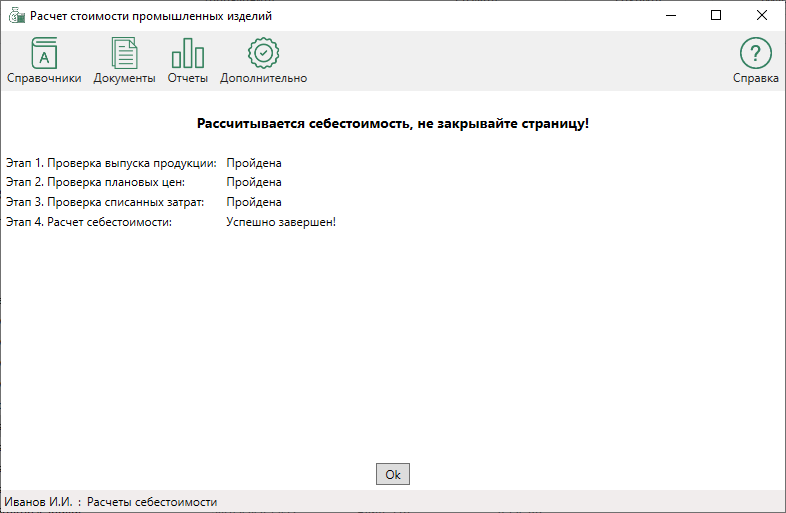


Рисунок 6.9 – Форма расчета себестоимости

Для формирования отчетов необходимо выбрать пункт меню «Отчеты» и подпункт, соответствующий необходимому отчету. В результате откроется форма выбранного отчета, пример которой приведен на рисунке 6.10. Далее – в форме необходимо указать параметры отчета и нажать кнопку «Сформировать».

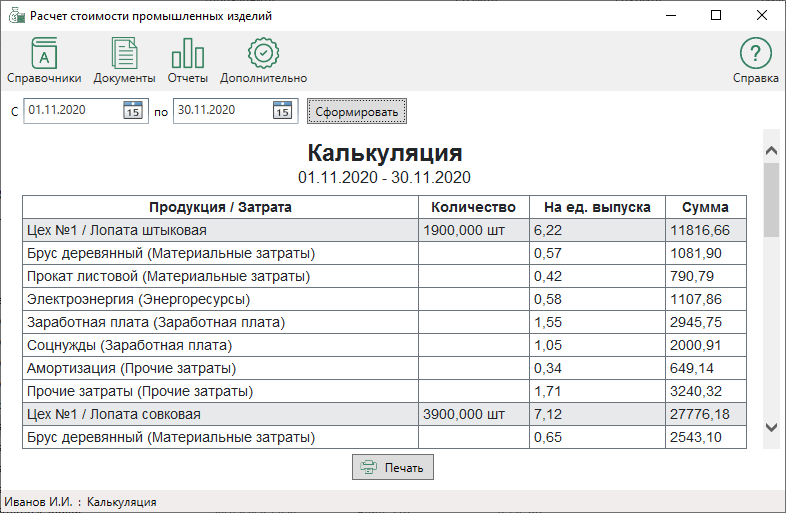


Рисунок 6.10 – Пример отчета

Для печати сформированного отчета необходимо нажать кнопку «Печать».

Для просмотра логов необходимо выбрать пункт меню и подпункт «Просмотр лог-файла». В результате будет открыта страница списка записей лог-файла (рисунок 6.11).

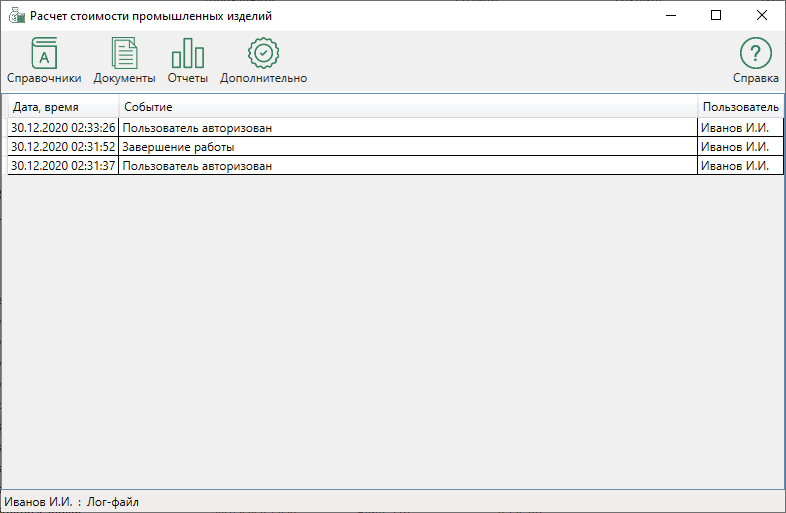


Рисунок 6.11 – Форма списка записей лог-файла

Для просмотра справки (рисунок 6.12) необходимо выбрать пункт меню «Справка».

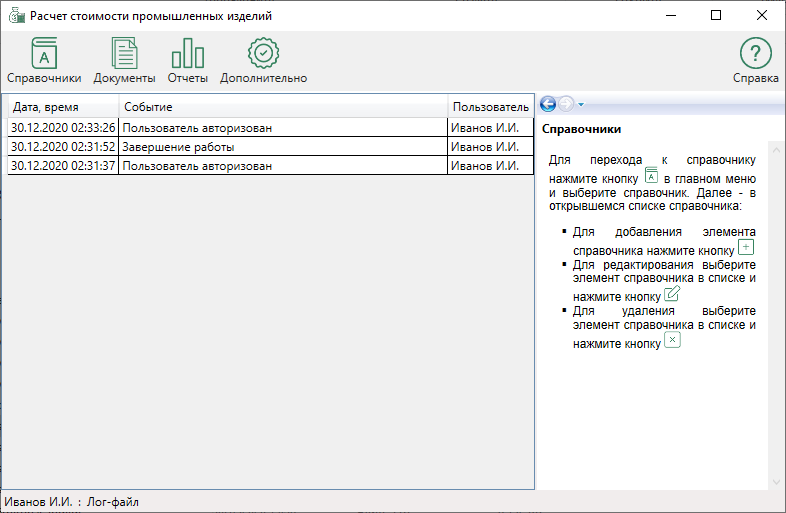


Рисунок 6.12 – Справочная информация

# Результаты тестирования разработанной системы и оценка выполнения задач

Функциональное тестирование программного средства проведено с использованием тест-кейсов двух типов:

* позитивные тест-кейсы, предполагающие ввод корректных данных, позволяющие протестировать поведение программного средства в типовых ситуациях;
* негативные тест-кейсы, предполагающие ввод некорректных данных, позволяющие протестировать поведение программного средства в исключительных ситуациях.

Функциональное тестирование проводится в соответствии с содержанием функций и их распределением в системе согласно методике использования программного средства.

Результаты позитивных тест-кейсов приведены в таблице 7.1.

Таблица . – Результаты позитивных тест-кейсов

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Действия пользователя | Ожидаемое поведение системы | Результат |
| 1 | 2 | 3 | 4 |
| Авторизоваться | Запускает приложение | Выводит форму авторизации | Пройден |
| Выбирает в списке пользователя, водит пароль | Ожидает ввода данных | Пройден |
| Подтверждение операции | Скрывает форму авторизации, выводит главную форму | Пройден |
| Редактировать справочник номенклатуры | Выбор справочника | Выводит список справочника номенклатуры | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Ввод данных справочника номенклатуры | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные справочника номенклатуры | Пройден |
| Редактировать справочник номенклатурных групп | Выбор справочника | Выводит список справочника номенклатурных групп | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Ввод данных справочника номенклатурных групп | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные справочника номенклатурных групп | Пройден |
| Редактировать справочник подразделений | Выбор справочника | Выводит список справочника подразделений | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные справочника подразделений | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные справочника подразделений | Пройден |
| Редактировать справочник статей затрат | Выбор справочника | Выводит список справочника статей затрат | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные справочника статей затрат | Ожидает ввода данных | Пройден |

Продолжение таблицы 7.1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
|  | Подтверждение операции | Сохраняет данные справочника статей затрат | Пройден |
| Редактировать справочник единиц измерения | Выбор справочника | Выводит список справочника единиц измерения | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные справочника единиц измерения | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные справочника единиц измерения | Пройден |
| Редактировать справочник пользователей | Выбор справочника | Выводит список справочника пользователей | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные справочника пользователей | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные справочника пользователей | Пройден |
| Регистрировать затраты | Выбор документа | Выводит список затрат | Пройден |
| Отбор документов | Выводит список в соответствии с условиями отбора | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
|  | Вводит данные затрат | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные затрат | Пройден |
| Регистрировать выпуск | Выбор документа | Выводит список выпуска продукции | Пройден |
| Отбор документов | Выводит список в соответствии с условиями отбора | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные выпуска продукции | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные выпуска продукции | Пройден |
| Регистрировать плановую цену | Выбор документа | Выводит список плановых цен | Пройден |
| Отбор документов | Выводит список в соответствии с условиями отбора | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |

Продолжение таблицы 7.1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
|  | Вводит данные плановых цен | Ожидает ввода данных | Пройден |
| Подтверждение операции | Сохраняет данные плановых цен | Пройден |
| Регистрировать расчет себестоимости | Выбор документа | Выводит список расчетов себестоимости | Пройден |
| Отбор документов | Выводит список в соответствии с условиями отбора | Пройден |
| Выбор операции | Выводит форму редактирования или форму подтверждения удаления | Пройден |
| Вводит данные расчета себестоимости | Ожидает ввода данных | Пройден |
| Подтверждение операции | Рассчитывает себестоимость, выводит данные о положительных проверках, сохраняет данные себестоимости | Пройден |
| Формировать отчет по распределению затрат | Выбор отчета | Выводит форму отчета по распределению затрат | Пройден |
| Вводит параметры отчета | Ожидает ввода данных | Пройден |
| Подтверждает формирование отчета | Отбирает данные, выводит сформированный отчет по распределению затрат | Пройден |
| Печать | Выводит отчет на печать | Пройден |
| Формировать структуру себестоимости | Выбор отчета | Выводит форму структуры себестоимости | Пройден |
| Вводит параметры отчета | Ожидает ввода данных | Пройден |
| Подтверждает формирование отчета | Отбирает данные, выводит сформированную структуру себестоимости | Пройден |
| Печать | Выводит отчет на печать | Пройден |
| Формировать калькуляцию | Выбор отчета | Выводит форму калькуляции | Пройден |
| Вводит параметры отчета | Ожидает ввода данных | Пройден |
| Подтверждает формирование отчета | Отбирает данные, выводит сформированную калькуляцию | Пройден |
| Печать | Выводит отчет на печать | Пройден |
| Формировать отчет по себестоимости | Выбор отчета | Выводит форму отчета по себестоимости | Пройден |
| Вводит параметры отчета | Ожидает ввода данных | Пройден |
| Подтверждает формирование отчета | Отбирает данные, выводит сформированный отчет по себестоимости | Пройден |
| Печать | Выводит отчет на печать | Пройден |

Все позитивные тест-кейсы пройдены с положительным результатом.

Результаты негативных тест-кейсов приведены в таблице 7.2.

Таблица . – Результаты негативных тест-кейсов

|  |  |  |  |
| --- | --- | --- | --- |
| Название функции | Действия пользователя | Ожидаемое поведение системы | Результат |
| 1 | 2 | 3 | 4 |
| Авторизоваться | Запускает приложение | Выводит форму авторизации | Пройден |
| Не выбирает пользователя в списке | Ожидает ввода данных | Пройден |
| Подтверждение операции | Выводит предупреждение «Необходимо выбрать пользователя» | Пройден |
| Авторизоваться | Запускает приложение | Выводит форму авторизации | Пройден |
| Выбирает пользователя в списке, вводит неверный пароль | Ожидает ввода данных | Пройден |
| Подтверждение операции | Выводит предупреждение «Неверный пароль» | Пройден |
| Редактировать справочник (для каждого справочника) | Выбор справочника | Выводит список справочника | Пройден |
| Выбор операции удаления, не выбирая строку в таблице | Выводит предупреждение «Не выбран объект для удаления» | Пройден |
| Редактировать справочник (для каждого справочника) | Выбор справочника | Выводит список справочника | Пройден |
| Выбор операции удаления для элемента справочника, на который имеются ссылки | Выводит предупреждение «Нельзя удалить объект, т.к. на него имеются ссылки» | Пройден |
| Редактировать документ (для каждого документа) | Выбор документа | Выводит список документов | Пройден |
| Выбор операции удаления, не выбирая строку в таблице | Выводит предупреждение «Не выбран объект для удаления» | Пройден |
| Регистрировать расчет себестоимости, не регистрируя выпуск | Выбор документа | Выводит список расчетов | Пройден |
| Выбор операции добавления или редактирования | Выводит форму редактирования | Пройден |
| Подтверждение расчета себестоимости | Выводит предупреждение «Не зарегистрированы документы выпуска» | Пройден |

Продолжение таблицы 5.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | | 4 |
| Регистрировать расчет себестоимости, не регистрируя плановые цены для одной и более номенклатуры | Выбор документа | Выводит список расчетов себестоимости | | Пройден |
| Выбор операции добавления или редактирования | Выводит форму редактирования | | Пройден |
| Подтверждение расчета себестоимости | Выводит предупреждение «Не установлены плановые цены» | | Пройден |
| Регистрировать расчет себестоимости, не регистрируя затрат по подразделению выпуска | Выбор документа | Выводит список расчетов себестоимости | | Пройден |
| Выбор операции добавления или редактирования | Выводит форму редактирования | | Пройден |
| Подтверждение расчета себестоимости | Выводит предупреждение «Подразделения выпуска и затрат не совпадают» | | Пройден |
| Регистрировать расчет себестоимости, не регистрируя выпуска по подразделению затрат | Выбор документа | Выводит список расчетов | | Пройден |
| Выбор операции добавления или редактирования | Выводит форму редактирования | | Пройден |
| Подтверждение расчета себестоимости | Выводит предупреждение «Подразделения выпуска и затрат не совпадают» | | Пройден |
| Регистрировать расчет себестоимости, не регистрируя затрат по номенклатурной группе выпуска | Выбор документа | Выводит список расчетов себестоимости | | Пройден |
| Выбор операции добавления или редактирования | Выводит форму редактирования | | Пройден |
| Подтверждение расчета себестоимости | Выводит предупреждение «Номенклатурные группы выпуска и затрат не совпадают» | | Пройден |
| Регистрировать расчет себестоимости, не регистрируя выпуска по номенклатурной группе затрат | Выбор документа | Выводит список расчетов себестоимости | Пройден | |
| Выбор операции добавления или редактирования | Выводит форму редактирования | Пройден | |
| Подтверждение расчета себестоимости | Выводит предупреждение «Номенклатурные группы выпуска и затрат не совпадают» | Пройден | |

Все негативные тест-кейсы пройдены с положительным результатом.

Таким образом, функциональное тестирование пройдено успешно, программное средство удовлетворяет требованиям задания на разработку.

Заключение

Описана предметная область, методика расчета себестоимости, определены требования к системе с точки зрения предметной области: номенклатура продукции разбивается на группы; учет затрат может осуществляться в разрезе подразделений, номенклатурных групп, продукции; любая затрата относится к определенной статье затрат; распределение затрат на выпущенную продукцию осуществляется пропорционально плановой цене.

Выполнена постановка задачи на разработку системы расчета себестоимости и определены технологии реализации проекта.

Разработаны модели представления системы: функциональная модель в виде диаграммы вариантов использования и структура системы в виде диаграмм классов в нотации UML.

Разработана информационная модель системы: схема базы данных, описание таблиц базы данных.

Разработаны алгоритмы, реализующие бизнес-логику системы, и соответствующий им программный код.

Таким образом, реализовано программное средство расчета себестоимости, представляющее собой приложение рабочего стола Windows. Реализация выполнена с использованием технологии ASP.NET на языке программирования C#. Интерфейс пользователя выполнен с использованием технологии WPF. Для хранения данных используется СУБД Microsoft SQL Server. Доступ к данным реализован с использованием Entity Framework.

Результаты тестирования показали, что разработанное программное средство удовлетворяет функциональным требованиям и функции выполняются корректно.

Изложенное позволяет сделать вывод о том, что цель, поставленная в курсовом проекте, достигнута. Повышение эффективности процесса расчета себестоимости достигается за счет сокращения времени на расчет себестоимости и своевременного формирования калькуляций и структуры затрат.

Список использованных источников

1. Демина, И. Д. Теория и практика применения современных методов учета затрат и калькулирования себестоимости продукции / И. Д. Демина, С. Н. Меркущенков. – М. : КноРус, 2019. – 332 с.
2. Овчинникова, О. П. Калькулирование себестоимости продукции в управленческом учете // Справочник экономиста. 2015. №5. – 12 с.
3. Макарова, Л. Г. Учет затрат на производство и калькулирование себестоимости продукции (работ, услуг) / Л. Г. Макарова [и др.]. – М.: Инфра-М, 2018. – 192 с.
4. Учет затрат, калькулирование, бюджетирование в отраслях производственной сферы / Под ред. Е. И. Костюковой. – М.: Лань, 2017. – 368 с.
5. Левачев, П. А. Финансовый механизм и управленческий учет расхода производственных ресурсов компании / П. А. Левачев, В. И. Осипов. – М.: Инфра-М, 2018. – 232 с.
6. Правила расчета себестоимости продукции на производстве [Электронный документ]. – Режим доступа: http://www.delasuper.ru/view\_post.php?id=6775. – Дата доступа: 10.12.2020.
7. Этрилл, П. Финансовый менеджмент и управленческий учет для руководителей и бизнесменов / П. Этрилл, Э. МакЛейни. – М.: Альпина Паблишер, 2020. – 648 с.
8. Себестоимость – что такое, как рассчитать [Электронный документ]. – Режим доступа: https://kakzarabativat.ru/nachinayushhim-predprinimatelyam/chto-takoe-sebestoimost-kak-rasschitat/. – Дата доступа: 10.12.2020.
9. Трофимова, И. Д. Учет затрат в производственном процессе по единицам оборудования / И. Д. Трофимова, Т. И. Федоркевич. – М.: Горная книга, 2019. – 192 с.
10. Саранцева, Е. Г. Учет и анализ затрат на производство в системе управления / Е. Г. Саранцева. – М.: КноРус, 2019. – 432 с.
11. Кузьмина, М. С. Учет затрат, калькулирование и бюджетирование в отраслях производственной сферы / М. С. Кузьмина. – М.: Финансы и статистика, 2017. – 208 с.
12. Керимов, В. Э. Учет затрат, калькулирование и бюджетирование в отдельных отраслях производственной сферы / В. Э. Керимов. – М.: Дашков и Ко, 2020. – 384 с.
13. Сиерра, К. Изучаем Java / К. Сиерра, Б. Бейтс. – М.: Эксмо, 2019. – 720 с.
14. Эспозито, Д. Microsoft.NET. Архитектура корпоративных приложений / Д. Эспозито, А. Сальтарелло. – М.: Вильямс, 2017. – 432 с.
15. Visual Studio IDE [Электронный документ]. – Режим доступа: https://visualstudio.microsoft.com/ru/. – Дата доступа: 01.12.2020.
16. SharpDevelop [Электронный документ]. – Режим доступа: https://sourceforge.net/projects/sharpdevelop/. – Дата доступа: 01.12.2020.
17. Новиков, Б. А. Настройка приложений баз данных / Б. А. Новиков, Г. Р. Домбровская. – СПб.: БХВ-Петербург, 2019. – 240 с.
18. Станек, У. Р. Microsoft SQL Server. Справочник администратора / У. Р. Станек. – СПб.: БХВ-Петербург, 2018. – 576 с.
19. Кузнецов, С. Д. Базы данных / С. Д. Кузнецов. – М.: Academia, 2019. – 496 с.
20. Макдональд, М. WPF: Windows Presentation Foundation в .NET 4.5 с примерами на C# 5.0 для профессионалов / М. Макдональд. – М.: Вильямс, 2018. – 1024 с.
21. Entity Framework [Электронный ресурс] / Д. Вега. – Microsoft, 2020. – Режим доступа: https://docs.microsoft.com/ru-ru/ef/ef6/. – Дата доступа: 10.12.2020.

Приложение  
(обязательное)  
Листинг программного кода

Стартовый класс приложения

using System;

using System.IO;

using System.Security.Cryptography;

using System.Text;

using System.Windows;

namespace Sebestoimost

{

public partial class App : Application

{

public static Model.dbContext db = new Model.dbContext();

public static Model.User user;

public static string path = Directory.GetCurrentDirectory() + "\\" + "log.txt";

public static string GetMD5(string input)

{

MD5 md5 = MD5.Create();

byte[] data = md5.ComputeHash(Encoding.Default.GetBytes(input));

StringBuilder sb = new StringBuilder();

for (int i = 0; i < data.Length; i++)

{

sb.Append(data[i].ToString("x2"));

}

return sb.ToString();

}

public static void SetLogText(string text)

{

File.AppendAllText(path, DateTime.Now.ToString() + "\t" + text + "\r\n");

}

}

}

Разметка формы авторизации

<Window x:Class="Sebestoimost.Login"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:Sebestoimost"

mc:Ignorable="d"

Title="Авторизация" Height="150" Width="300" Icon="pack://siteoforigin:,,,/Resources/budget.png" ResizeMode="NoResize" ShowInTaskbar="False">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="Auto" />

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition Height="Auto" />

<RowDefinition />

<RowDefinition Height="Auto" />

</Grid.RowDefinitions>

<TextBlock Grid.Column="0" Grid.Row="0" Text="Пользователь:" Margin="5" VerticalAlignment="Center" />

<ComboBox Grid.Column="1" Grid.Row="0" Margin="5" x:Name="LstUsers" DisplayMemberPath="Name" />

<TextBlock Grid.Column="0" Grid.Row="1" Text="Пароль:" Margin="5" VerticalAlignment="Center" />

<PasswordBox Grid.Column="1" Grid.Row="1" Margin="5" x:Name="FldPassword" />

<Button Grid.Column="1" Grid.Row="3" x:Name="BtnLogin" Click="BtnLogin\_Click" Margin="5">

<TextBlock Text="Ok" />

</Button>

</Grid>

</Window>

Класс формы авторизации

using Sebestoimost.Model;

using System.Linq;

using System.Windows;

namespace Sebestoimost

{

public partial class Login : Window

{

public Login()

{

InitializeComponent();

LstUsers.ItemsSource = App.db.Users.Where(p => p.Enabled).ToList();

}

private void BtnLogin\_Click(object sender, RoutedEventArgs e)

{

if (LstUsers.SelectedItem != null)

{

User user = LstUsers.SelectedItem as User;

if (user.Password.Equals(App.GetMD5(FldPassword.Password)))

{

App.SetLogText("Пользователь авторизован\t" + user.Name);

Hide();

App.user = user;

Main frm = new Main();

frm.ShowDialog();

App.SetLogText("Завершение работы\t" + user.Name);

Close();

}

else

{

FldPassword.Password = "";

MessageBox.Show("Неверный пароль!", "", MessageBoxButton.OK, MessageBoxImage.Warning);

App.SetLogText("Ошибка авторизации\t" + user.Name);

}

}

else

MessageBox.Show("Необходимо выбрать пользователя!", "", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

}

Класс формы расчета себестоимости

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Navigation;

using Sebestoimost.Model;

namespace Sebestoimost.Pages

{

public partial class CostsCalc : Page

{

Cost item;

public CostsCalc(int id)

{

InitializeComponent();

App.db = new dbContext();

item = App.db.Costs.FirstOrDefault(p => p.Id == id);

BtnOk.Visibility = Visibility.Collapsed;

}

private void Page\_Loaded(object sender, RoutedEventArgs e)

{

if (item == null)

{

NavigationService.Navigate(new CostsList());

}

// Удаляем все результаты расчета себестоимости

foreach (Structure structure in App.db.Structures.Where(p => p.CostId == item.Id))

{

App.db.Structures.Remove(structure);

}

App.db.SaveChanges();

// Даты отбора

DateTime dateStart = new DateTime(item.Date.Year, item.Date.Month, 1, 0, 0, 0);

DateTime dateEnd = dateStart.AddMonths(1).AddDays(-1);

dateEnd = new DateTime(dateEnd.Year, dateEnd.Month, dateEnd.Day, 23, 59, 59);

// Список документов выпуска

var outputs = App.db.Outputs.Where(p => p.Date >= dateStart && p.Date <= dateEnd).ToList();

// Проверка наличия выпуска продукции: если выпуска не было, расчет не проводится

if (outputs.Count == 0)

{

FldStage1.Text = "За месяц не зарегистрирован выпуск продукции!";

BtnOk.Visibility = Visibility.Visible;

return;

}

else

{

FldStage1.Text = "Пройдена";

}

// Список номенклатуры выпуска

List<Nomenclature> oNomenclatures = new List<Nomenclature>();

foreach (var output in outputs)

{

if (oNomenclatures.Count(p => p.Id == output.NomenclatureId) == 0)

{

oNomenclatures.Add(output.Nomenclature);

}

}

// Список номенклатуры с плановыми ценами

List<NomenclaturePriceData> oNomenclaturePrices = new List<NomenclaturePriceData>();

foreach (var nomenclature in oNomenclatures)

{

var plans = App.db.Plans.Where(p => p.NomenclatureId == nomenclature.Id && p.Date <= dateEnd).OrderByDescending(p => p.Date).ToArray();

if (plans.Count() > 0)

{

oNomenclaturePrices.Add(new NomenclaturePriceData { Nomenclature = nomenclature, Price = plans[0].Price });

}

}

// Проверка наличия плановых цен: если плановые цены на выпущенную продукцию не зарегистрированы, расчет не проводится

if (oNomenclatures.Count > oNomenclaturePrices.Count)

{

FldStage2.Text = "Не зарегистрированы плановые цены на всю продукцию выпуска!";

BtnOk.Visibility = Visibility.Visible;

return;

}

else

{

FldStage2.Text = "Пройдена";

}

// Список документов затрат

var expenses = App.db.Expenses.Where(p => p.Date >= dateStart && p.Date <= dateEnd).ToList();

// Проверка наличия регистрации затрат: если затрат не было, дальнейшие расчеты не имеют смысла

if (expenses.Count == 0)

{

FldStage3.Text = "За меясц не зарегистрированы затраты!";

BtnOk.Visibility = Visibility.Visible;

return;

}

// Список подразделений выпуска

List<Department> oDepartments = new List<Department>();

foreach (var output in outputs)

{

if (oDepartments.Count(p => p.Id == output.DepartmentId) == 0)

{

oDepartments.Add(output.Department);

}

}

// Список подразделений затрат

List<Department> eDepartments = new List<Department>();

foreach (var expense in expenses)

{

if (eDepartments.Count(p => p.Id == expense.DepartmentId) == 0)

{

eDepartments.Add(expense.Department);

}

}

// Проверка подразделений

if (oDepartments.Except(eDepartments).Count() > 0)

{

FldStage3.Text = "Подразделения выпуска и затрат не совпадают!";

BtnOk.Visibility = Visibility.Visible;

return;

}

// Список номенклатурных групп выпуска

List<Class> oClasses = new List<Class>();

foreach (var nomenclature in oNomenclatures)

{

if (oClasses.Count(p => p.Id == nomenclature.ClassId) == 0)

{

oClasses.Add(nomenclature.Class);

}

}

// Список номенклатурных групп затрат

List<Class> eClasses = new List<Class>();

foreach (var expense in expenses)

{

if (eClasses.Count(p => p.Id == expense.ClassId) == 0)

{

eClasses.Add(expense.Class);

}

}

// Проверка номенклатурных групп

if (oClasses.Except(eClasses).Count() > 0)

{

FldStage3.Text = "Номенклатурные группы выпуска и затрат не совпадают!";

BtnOk.Visibility = Visibility.Visible;

return;

}

FldStage3.Text = "Пройдена";

// Формируем список выпущенной продукции по подразделениям с ценами и количеством

List<OutputData> oDatas = new List<OutputData>();

foreach (var department in oDepartments)

{

foreach (var nomenclature in oNomenclaturePrices)

{

decimal quantity = outputs.Where(p => p.DepartmentId == department.Id && p.NomenclatureId == nomenclature.NomenclatureId).Sum(p => p.Quantity);

if (quantity > 0)

{

oDatas.Add(new OutputData

{

Department = department,

Nomenclature = nomenclature.Nomenclature,

Quantity = quantity,

Price = nomenclature.Price,

Summa = quantity \* nomenclature.Price

});

}

}

}

// Формируем сводный список затрат

var gExpenses = expenses.GroupBy(cm => new { cm.Department, cm.Class, cm.Expenditure, cm.Nomenclature },

(key, group) => new { key.Department, key.Class, key.Expenditure, key.Nomenclature, Summa = group.Sum(p => p.Summa) });

// Формируем список выпуска по подразделениям и номенклатурным группам

var gOutputs = oDatas.GroupBy(cm => new { cm.Department, cm.Class },

(key, group) => new { key.Department, key.Class, Nomenclatures = group.ToList(), Summa = group.Sum(p => p.Summa) });

// Распределяем затраты на продукцию

List<CostStructure> costStructures = new List<CostStructure>();

foreach (var gOutput in gOutputs)

{

foreach (var nomenclature in gOutput.Nomenclatures)

{

var costStructure = new CostStructure() { Department = gOutput.Department, Nomenclature = nomenclature.Nomenclature, Quantity = nomenclature.Quantity };

foreach (var gExpense in gExpenses.Where(p => p.Department.Id == gOutput.Department.Id && p.Class.Id == gOutput.Class.Id))

{

var expenseSumma = nomenclature.Summa \* gExpense.Summa / gOutput.Summa;

costStructure.CostCalculations.Add(new CostCalculation

{

Expenditure = gExpense.Expenditure,

Nomenclature = gExpense.Nomenclature,

Summa = expenseSumma

});

}

costStructures.Add(costStructure);

}

}

// Сохраняем затраты в БД

foreach (var costStructure in costStructures)

{

Structure structure = new Structure

{

Cost = item,

Nomenclature = costStructure.Nomenclature,

Department = costStructure.Department,

Quantity = costStructure.Quantity

};

App.db.Structures.Add(structure);

foreach (var costCalculation in costStructure.CostCalculations)

{

Calculation calculation = new Calculation

{

Structure = structure,

Expenditure = costCalculation.Expenditure,

Nomenclature = costCalculation.Nomenclature,

Summa = costCalculation.Summa

};

App.db.Calculations.Add(calculation);

}

}

try

{

App.db.SaveChanges();

}

catch (Exception ex)

{

App.db.UndoChanges();

MessageBox.Show(ex.Message, "Ошибка сохранения", MessageBoxButton.OK, MessageBoxImage.Warning);

App.SetLogText("Ошибка сохранения результатов расчета себестоимости\t" + App.user.Name);

FldStage4.Text = "Не удалось сохранить результаты!";

BtnOk.Visibility = Visibility.Visible;

return;

}

FldStage4.Text = "Успешно завершен!";

BtnOk.Visibility = Visibility.Visible;

}

private void BtnOk\_Click(object sender, RoutedEventArgs e)

{

NavigationService.Navigate(new CostsList());

}

}

}

Разметка формы отчета

<Page x:Class="Sebestoimost.Pages.ReportCostPrice"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:Sebestoimost.Pages"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="400"

Title="Себестоимость">

<Grid Margin="5">

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition />

<RowDefinition Height="Auto" />

</Grid.RowDefinitions>

<StackPanel Grid.Row="0" Orientation="Horizontal" Margin="0, 0, 0, 5">

<Label>С</Label>

<DatePicker x:Name="FldDateAt" SelectedDateChanged="FldDateAt\_SelectedDateChanged" />

<Label>по</Label>

<DatePicker x:Name="FldDateTo" SelectedDateChanged="FldDateTo\_SelectedDateChanged" />

<Button x:Name="BtnShow" Click="BtnShow\_Click" Margin="8, 0, 0, 0" Padding="8, 2">Сформировать</Button>

</StackPanel>

<WebBrowser Grid.Row="1" x:Name="fldBrowser" />

<WrapPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Center">

<Button x:Name="BtnPrint" Click="BtnPrint\_Click" Margin="5" Padding="8, 2">

<StackPanel Orientation="Horizontal">

<Image Source="pack://siteoforigin:,,,/Resources/printer.png" MaxHeight="16" />

<TextBlock Text="Печать" VerticalAlignment="Center" Margin="8, 2, 2, 2"/>

</StackPanel>

</Button>

</WrapPanel>

</Grid>

</Page>

Класс формы отчета

using Sebestoimost.Model;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

namespace Sebestoimost.Pages

{

public partial class ReportCostPrice : Page

{

string title;

public ReportCostPrice()

{

InitializeComponent();

App.db = new dbContext();

title = "Себестоимость";

FldDateAt.SelectedDate = new DateTime(2020, 1, 1);

FldDateTo.SelectedDate = new DateTime(2020, 1, 31);

ReportCreate();

}

private void BtnPrint\_Click(object sender, RoutedEventArgs e)

{

try

{

var doc = fldBrowser.Document as mshtml.IHTMLDocument2;

doc.execCommand("Print", true, null);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Ошибка печати", MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void BtnShow\_Click(object sender, RoutedEventArgs e)

{

ReportCreate();

}

private void ReportCreate()

{

DateTime dateAt = (DateTime)FldDateAt.SelectedDate;

DateTime dateTo = (DateTime)FldDateTo.SelectedDate;

dateAt = new DateTime(dateAt.Year, dateAt.Month, 1, 0, 0, 0);

dateTo = new DateTime(dateTo.Year, dateTo.Month, 1, 0, 0, 0);

dateTo = dateTo.AddMonths(1).AddDays(-1);

dateTo = new DateTime(dateTo.Year, dateTo.Month, dateTo.Day, 23, 59, 59);

FldDateAt.SelectedDate = dateAt;

FldDateTo.SelectedDate = dateTo;

var costs = App.db.Costs.Where(p => p.Date >= dateAt && p.Date <= dateTo).ToList();

List<Calculation> calculations = new List<Calculation>();

foreach (var cost in costs)

{

foreach (var structure in App.db.Structures.Where(p => p.CostId == cost.Id))

{

foreach (var calculation in App.db.Calculations.Where(p => p.StructureId == structure.Id))

{

calculations.Add(calculation);

}

}

}

var gReport = calculations.GroupBy(

cm => new { cm.Structure },

(key, group) => new {

key.Structure,

Summa = group.Sum(p => p.Summa)

});

string reportStr = ReportHtmlHelper.PageStart(title) + ReportHtmlHelper.ReportHeader(title, dateAt, dateTo);

string thead = ReportHtmlHelper.Tr(new string[] { "Продукция", "Количество", "На ед. выпуска", "Сумма" });

string tbody = "";

foreach (var structure in gReport)

{

tbody += ReportHtmlHelper.Tr(new string[]

{

structure.Structure.Department.Name + " / " + structure.Structure.Nomenclature.Name,

string.Format("{0:0.000} {1}", structure.Structure.Quantity, structure.Structure.Nomenclature.Class.Measure.Name),

string.Format("{0:0.00}", structure.Summa / structure.Structure.Quantity),

string.Format("{0:0.00}", structure.Summa)

});

}

reportStr += ReportHtmlHelper.Table(thead, tbody);

reportStr += ReportHtmlHelper.PageEnd();

fldBrowser.NavigateToString(reportStr);

}

private void FldDateTo\_SelectedDateChanged(object sender, SelectionChangedEventArgs e)

{

if (FldDateTo.SelectedDate < FldDateAt.SelectedDate)

{

FldDateAt.SelectedDate = FldDateTo.SelectedDate;

}

}

private void FldDateAt\_SelectedDateChanged(object sender, SelectionChangedEventArgs e)

{

if (FldDateTo.SelectedDate < FldDateAt.SelectedDate)

{

FldDateTo.SelectedDate = FldDateAt.SelectedDate;

}

}

}

}