

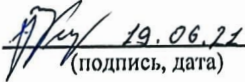
МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

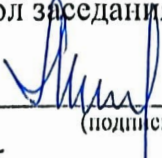
Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА
на тему
Web-приложение учета и анализа
деятельности малого предприятия по
производству мясной продукции

Студент	 (подпись, дата)	Красномясов Андрей Александрович (ФИО полностью)
Руководитель	 (подпись, дата)	Шибанов С.В. (фамилия, инициалы)
Нормоконтролёр	 (подпись, дата)	Такташкин Д.В. (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой	 (подпись)	Макарычев П.П. (фамилия, инициалы)
---------------------	--	---------------------------------------

Работа защищена с отметкой 5 (протокол заседания ГЭК от 29.06.21 № 14)

Секретарь ГЭК	 (подпись)	Попова Н.А. (фамилия, инициалы)
---------------	--	------------------------------------

Пенза, 2021

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
П.П. Макарычев
«18» января 2021 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент Красномясов Андрей Александрович гр. 17ВИ1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы web-приложение учета и анализа деятельности малого предприятия по
производству мясной продукции

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20 _____

2. Руководитель работы к.т.н., доцент кафедры «МО и ПЭВМ» Шибанов С.В.

3. Задание на работу (назначение разработки, исходные данные и т.п.)
Назначение: web-приложение учета и анализа деятельности малого предприятия по
производству мясной продукции предназначено для взаимодействия пользователей с
хранилищем данных системы, при этом имея возможность их изменения, просмотра и
добавления в удобной форме.

Средства выполнения: Операционная система Windows 10 PRO (версия: 20H2)

Средства разработки: IntelliJ IDEA, WebStorm, Navicat, Postman

Языки программирования: Java, JavaScript

Технологии разработки: ООП, UML, MVC

4. Перечень подлежащих разработке вопросов

Анализ предметной области и постановка задачи на разработку

Планирование разработки и расчет бюджета

Разработка требований к web-приложению

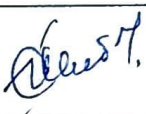
Разработка базы данных

Разработка программного обеспечения серверного приложения

Разработка программного обеспечения клиентского приложения

Контроль качества программного обеспечения

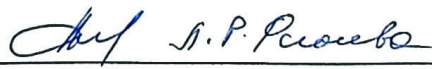
5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и постановка задачи на разработку	10	14.04.2021 – 19.04.2021	
2	Разработка требований к приложению	15	20.04.2021 – 27.04.2021	
3	Разработка базы данных	10	28.04.2021 – 4.05.2021	
4	Разработка программного обеспечения серверного приложения	20	5.05.2021 – 19.05.2021	
5	Разработка программного обеспечения клиентского приложения	20	20.05.2021 – 30.05.2021	
6	Контроль качества программного обеспечения	15	31.05.2021 – 8.06.2021	
7	Оформление пояснительной записки	10	9.06.2021 – 14.06.2021	

Дата выдачи задания "28" декабря 2020

Руководитель бакалаврской работы  28.12.20
(подпись, дата)

Задание к исполнению принял студент  28.12.20
(подпись, дата)

Работу к защите допустить
Декан факультета  28.06.2024
(подпись, дата)

Реферат

Пояснительная записка содержит 115 листов, 43 рисунка, 20 таблиц, 22 использованных источника и 6 приложений.

JAVA, SPRING, МЯСОПЕРЕРАБОТКА, MYSQL, REACT, WEB-ПРИЛОЖЕНИЕ, REST API.

Объект исследования: web-приложение учета и анализа деятельности малого предприятия по производству мясной продукции.

Цель работы: реализовать web-приложение учета и анализа деятельности малого предприятия по производству мясной продукции.

Технологии разработки: ООП, MVC, IntelliJ IDEA, Webstorm, Navicat, MySQL.

Задание: спроектировать и реализовать прототип web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции.

Результаты работы: спроектирован и реализован прототип web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции.

ПГУ 09.03.02 – 8БР171.08 ПЗ

Изм.	Лист	№ докум.	Подпись	Дата			
Разраб.		Красномясов А.А.		19.06.22	«web-приложение учета и анализа деятельности малого предприятия по производству мясной продукции»	Лит.	Лист
Провер.		Шибанов С.В.		19.06.22			Листов
Реценз.							4
Н. Контр.		Такташкин Д.В.		22.06.22		Группа 17ВИ1	
Утверд.		Макарычев П.П.		28.06.22			
					Пояснительная записка		

СОДЕРЖАНИЕ

Введение.....	7
1 Анализ предметной области и постановка задачи на разработку.....	9
1.1 Анализ предметной области	9
1.1.1 Анализ структуры предметной области	9
1.1.2 Анализ бизнес-процессов предметной области.....	11
1.1.3 Анализ регламентирующих документов	14
1.2 Сравнительный анализ систем-аналогов.....	15
1.2.1 Обоснование выбора систем-аналогов	15
1.2.2 Обзор основных возможностей систем-аналогов.....	17
1.2.3 Результаты сравнительного анализа систем-аналогов.....	21
1.3 Постановка задачи на разработку	23
1.3.1 Цели разработки и назначение	23
1.3.2 Описание действующих лиц предметной области	23
1.3.3 Перечень автоматизируемых функций.....	24
1.4 Планирование и функционально-стоимостной анализ процесса разработки.....	26
1.4.1 Основы функционально-стоимостного анализа	26
1.4.2 Планирование процесса разработки	27
1.4.3 Анализ временных и финансовых затрат	31
2 Анализ требований на разработку.....	33
2.1 Функциональные требования	33
2.2 Построение концептуальной модели	41
2.3 Требования к обобщенной архитектуре приложения	44
2.4 Обоснование выбора технологий и средств разработки.....	45
2.4.1 Обоснование выбора системы управления базами данных.....	45
2.4.2 Обоснование выбора языковых и инструментальных средств	46
2.4.3 Обоснование выбора фреймворка для разработки серверного и клиентского приложения.....	49
2.5 Требования к аппаратно-программному обеспечению.....	50

2.5.1	Требования к аппаратно-техническому обеспечению	50
2.5.2	Требования к программному обеспечению.....	51
3	Разработка информационного и программного обеспечения web-приложения	53
3.1	Разработка архитектуры web-приложения.....	53
3.2	Разработка базы данных web-приложения.....	54
3.3	Разработка программного обеспечения серверного приложения.....	62
3.4	Разработка клиентского приложения.....	66
3.4.1	Разработка интерфейса клиентского приложения.....	66
3.4.1	Разработка программного обеспечения клиентского приложения	74
4	Контроль качества программного обеспечения.....	75
4.1	Планирование процесса контроля качества	75
4.1.1	Мероприятия по контролю качества.....	75
4.1.2	Обоснование выбора модели тестирования	77
4.2	Тестирование программного обеспечения	78
4.2.1	Функциональное тестирование	78
4.2.2	Модульное тестирование	81
	Заключение	84
	Список источников	85
	Приложение А. Глоссарий	88
	Приложение Б. Перечень нормативной документации.....	90
	Приложение В. Скрипт создания базы данных.....	97
	Приложение Г. Листинг Rest API сервера.....	104
	Приложение Д. Листинг шаблона главного интерфейса	109
	Приложение Е. Листинг Unit тестов	113

Введение

Самый большой сегмент пищевой промышленности по переработке мяса приходится на мясную промышленность. Ежегодно прирост объемов производства готовой мясной продукции составляет порядка 5 – 7%.

Сложная экономическая ситуация, возникшая вследствие пандемии коронавируса, не оказала существенного негативного влияния на производственные показатели мясоперерабатывающей отрасли – по итогам 2020 года прирост объемов выпуска наблюдался во всех категориях продукции:

- мясные и мясосодержащие полуфабрикаты – 3,9 млн. т. (прирост 7,4%);
- колбасные изделия – 2,4 млн. т. (прирост 4,6%);
- кулинарные мясные изделия – 160 тыс. т. (прирост 4,6%);
- прочие продукты из мяса и мяса птицы – 190,3 тыс. т. (прирост 16,5%) [1].

Данная информация говорит о том, что отрасль мясопереработки имеет большую популярность, что рождает спрос на информационные системы по управлению и анализу деятельности данных организаций.

Главной целью разработки является создание системы, которая будет сочетать в себе учет необходимых процессов, происходящих в цикле работы предприятия, при сочетая в себе простой и понятный пользовательский интерфейс.

Для достижения поставленной цели необходимо решить следующие задачи:

- выполнить анализ предметной области;
- провести сравнительный анализ существующих систем-аналогов;
- на основе анализа выполнить постановку задачи на разработку;
- определить функционал web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции;

- разработать клиентскую и серверную части для web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции в соответствии с поставленной задачей;
- провести тестирование ПО.

В результате выполнения работы был реализован прототип web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции, который позволяет:

- управлять поставками и их содержимым;
- осуществлять продажи произведенной продукции;
- производить поиск и в удобном формате просматривать список продукции на складе;
- производить анализ продаж по разным критериям.

В дальнейшем приложение будет развиваться и дорабатываться. Наиболее важными в доработке являются модуль администратора, как с точки зрения анализа, так и с точки зрения управления предприятием, и модуль переработки. Для всех перечисленных функций были предусмотрены таблицы в базе данных.

Структура пояснительной записки построена с учетом требований стандартов и рекомендаций кафедры [2].

1 Анализ предметной области и постановка задачи на разработку

1.1 Анализ предметной области

1.1.1 Анализ структуры предметной области

Мясоперерабатывающая промышленность – одна из крупнейших отраслей пищевой индустрии. Для увеличения количества выпускаемой мясной продукции ежегодно появляются новые и модернизируются старые мясоперерабатывающие предприятия. Каждое из этих предприятий проводит огромную работу по повышению качества, улучшению и расширению ассортимента мяса и мясных полуфабрикатов [3].

В одних производствах преобладают процессы разделения, извлечения сырья (при убое скота и обработке продуктов убоя), в других – формовки продукции из разнородных составных частей (при изготовлении колбасных изделий, консервов, котлет, пельменей и т. п.).

В связи со сложностью организации мясоперерабатывающего предприятия, необходимо фиксировать ресурсы на каждом этапе производственного цикла. Для этого необходимо выделить основные структурные элементы предприятия, на которых и будет производиться сбор и анализ данных. Основные структурные элементы и связи между ними представлены на рисунке 1.



Рисунок 1 – Обобщенная структура предприятия по мясопереработке

Как видно из диаграммы – структура предприятия является линейной с единственным ответвлением в элементе «Переработки», где ресурсы могут быть преобразованы в конечный продукт, либо возвращены на склад ресурсов в виде нового материала для последующей переработки.

Целью предприятия является преобразование входных поставок материалов в конечный продукт с последующей продажей. Далее приводится более детальное описание цикла работы предприятия.

Первый этап – это поставка материалов. К материалам относятся все позиции, используемые в производстве продукции (мясо, специи, оболочки и т.д.). Здесь и на следующих этапах производится взвешивание пришедших позиций. Это необходимо для вычисления потерь, которые будут включаться в себестоимость конечного продукта.

Далее материалы попадают на склад. Склад хранит в себе количество ресурсов для дальнейшей переработки и их количество постоянно изменяется в ходе работы.

Со склада материалы попадают на переработку, где в зависимости от ее вида на выходе можно получить либо другой материал, либо конечный продукт. Переработки происходят согласно рецептуре, которая содержит в себе входные пропорции и выходной продукт. Пользователь, в ходе работы, может изменять фактические пропорции.

Произведенная продукция попадает на отдельный склад, где в автоматическом режиме вычисляется ее себестоимость. С этого склада товар может попасть только в продажу и в дальнейшем обороте не участвует.

На основе данной структуры предметной области будет производить дальнейший её анализ. Элементы структуры будут выделены в отдельные сущности, а операции описаны в бизнес-процессах.

1.1.2 Анализ бизнес-процессов предметной области

Бизнес-процесс – совокупность различных видов деятельности (работ), в рамках которой на входе используется один или более видов ресурсов и в результате этой деятельности на выходе создается продукт, представляющий ценность для потребителя [4].

В ходе изучения и анализа предметной области были выделены основные бизнес-процессы. Описание бизнес-процессов для групп пользователей «работник склада» и «менеджер по продажам» представлено в таблицах 1, 2 соответственно.

Таблица 1 – Описание бизнес-процессов для группы «работник склада»

№ п/п	Бизнес- процесс	Описание	Ресурсы		Исполнитель
			Входящие	Исходящие	
1	Добавление поставки	Работник склада добавляет новую поставку в систему	Информация о поставке и входящих в нее ресурсах	Запись о поставке; Увеличение материалов на складе	Работник склада
2	Добавление поставщика	Работник склада добавляет нового поставщика	Информация о поставщике	Запись о поставщике в базе данных	Работник склада
3	Добавление материала	Работник склада добавляет новый материал	Информация о добавляемом материале	Запись о материале в базе данных	Работник склада

Таблица 2 – Описание бизнес-процессов для группы «менеджер по продажам»

№ п/п	Бизнес- процесс	Описание	Ресурсы		Исполнитель
			Входящие	Исходящие	
1	Продажа товара	Продавец фиксирует продажу товара	Продаваемый товар и его количество	Запись о продаже товара, уменьшение кол-ва товара на складе	Менеджер по продажам

В таблицах 3, 4 отображены основные бизнес-процессы для групп пользователей «работник цеха» и «клиент».

Таблица 3 – Описание бизнес-процессов для группы «работник цеха»

№ п/п	Бизнес- процесс	Описание	Ресурсы		Исполнитель
			Входящие	Исходящие	
1	Создание переработки	Работник цеха начинает новую переработку, фиксирую это в системе	Тип переработки, инициатор переработки	Запись о переработке с активным статусом	Работник цеха
2	Забор на переработку	Работник цеха фиксирует забор ресурсов на конкретную переработку	Список материалов и номер переработки	Записи, фиксирующие забор материалов со склада, уменьшение ресурсов на складе	Работник цеха
3	Завершение переработки	Работник цеха инициирует завершение переработки, изменяя ее статус и выпуская продукцию	Номер переработки, выходная продукция	Обновленный статус переработки в базе данных, обновление количества ресурсов на складах	Работник цеха

Таблица 4 – Описание бизнес-процессов для группы «клиент»

№ п/п	Бизнес- процесс	Описание	Ресурсы		Исполнитель
			Входящие	Исходящие	
1	Просмотр стоимости и наличия товара	Клиент получает информацию об актуальной стоимости и наличии товара	Интересующий товар	Информация о стоимости и наличии	Клиент

В таблице 5 отображены бизнес-процессы для группы пользователей «администратор»

Таблица 5 – Описание бизнес-процессов для группы «администратор»

№ п/п	Бизнес- процесс	Описание	Ресурсы		Исполнитель
			Входящие	Исходящие	
1	Анализ продаж	Администратор анализирует продажи	-	Статистическая информация о количествах продаж	Администратор
2	Анализ рентабельности продукта	Администратор анализирует рентабельность продукта	-	Статистическая информация по конкретному продукту, содержащая в себе себестоимость, количество продаж и прибыль	Администратор
3	Анализ цен на материалы	Администратор анализирует цены на материалы по совершенным поступлениям	-	Статистическая информация о стоимости материалов по поставщикам, с учетом потерь	Администратор

Для каждого описанного бизнес-процесса были выделены следующие параметры:

- описание – краткое описание бизнес-процесса;
- входящие ресурсы – ресурсы, требуемые на вход бизнес-процесса;
- исходящие ресурсы – ресурсы, получаемые на выходе после выполнения бизнес-процесса;
- исполнитель – пользователь или группа пользователей, выполняющая данный процесс.

На основании выделенных выше бизнес-процессов будет проводиться дальнейшее планирование разработки системы.

1.1.3 Анализ регламентирующих документов

Рассматриваемая предметная область предполагает процесс реализации произведенной продукции, что означает необходимость соблюдения федерального закона от 22.05.03 № 54-ФЗ, о применении контрольно-кассовой техники при осуществлении наличных денежных расчетов и (или) расчетов с использованием платежных карт.

Настоящим Федеральным законом определяются правила применения контрольно-кассовой техники при осуществлении расчетов в Российской Федерации в целях обеспечения интересов граждан и организаций, защиты прав потребителей, обеспечения установленного порядка осуществления расчетов, полноты учета выручки в организациях и у индивидуальных предпринимателей, в том числе в целях налогообложения и обеспечения установленного порядка оборота товаров.

Предприятия, занимающиеся производством любого вида мясной продукции, должны подчинять Отраслевому нормативному документу (ОНД) «Порядок санитарно-микробиологического контроля при производстве мяса и мясных продуктов»

Настоящий ОНД распространяется на мясоперерабатывающие предприятия всех форм собственности независимо от их мощности (в т.ч. и малые предприятия), производящие все виды мясной продукции.

ОНД определяет порядок и периодичность санитарно-микробиологического контроля и оценки качества мясного сырья, полуфабрикатов, готовых продуктов, условий их производства, а также вспомогательных материалов, санитарного состояния технологического оборудования, холодильных камер, инвентаря, тары, воды, личной гигиены работников [5].

Перечень нормативных документов, относящихся к данному ОНД, представлен в приложении Б.

Санитарно-микробиологический контроль направлен на предотвращение выпуска недоброкачественной продукции, не соответствующей требованиям ГОСТ, действующим «Медико-биологическим требованиям и санитарным нормам качества продовольственного сырья и пищевых продуктов», ограждение потребителей от возможности заболевания пищевыми токсикоинфекциями и токсикозами, выявление источников загрязнения сырья и готовой продукции для последующего проведения профилактических, эпидемиологических и эпизоотологических мероприятий.

Рассмотренные выше регламентирующие документы являются основными для рассматриваемой предметной области.

1.2 Сравнительный анализ систем-аналогов

1.2.1 Обоснование выбора систем-аналогов

Перед началом проектирования информационной системы был произведен анализ схожих систем, которые уже существуют на рынке программного обеспечения. Подбор систем производился по таким факторам, как функционал, доступность, актуальность. Для анализа были выбраны следующие системы, наиболее схожие по выбранным факторам с разрабатываемой:

- программный комплекс (ПК) «МультиМит Эксперт»;
- «1С: Мясопереработка»;
- «Меркурий-ERP»;
- «МойСклад».

Данные системы наиболее схожи с функционалом с разрабатываемой, помимо этого они являются актуальными и имеют большое количество пользователей.

Общая информация о рассматриваемых системах представлена в таблице 6.

Таблица 6 – Информация о рассматриваемых системах

Система	Основные свойства		
	Фирма-производитель	Платформа	Конфигурируемость
ПК «МультиМит Эксперт»	ООО «ФудСофт»	Поддерживает только систему Windows версии Windows XP и выше	Конфигурация настраивается средствами самой системы
«1С: Мясопереработка»	ООО «1С»	Поддерживаются десктопные и мобильные ОС	Требуется программирование системы на внутреннем языке 1С под конкретные нужды предприятия
«Меркурий-ERP»	Татарченко Александр	Поддерживает только систему Windows версии Windows XP и выше	Конфигурация настраивается средствами самой системы
«МойСклад»	Логнекс	Веб-приложение	Конфигурация настраивается средствами самой системы

Первые 3 системы обладают различным функционалом, но предназначены для одной цели – работы и автоматизации мясоперерабатывающих предприятий. Система «МойСклад» обладает более общим функционалом автоматизации работы склада, но обладает при этом наиболее понятным интерфейсом, что тоже является важной частью разрабатываемой системы.

Также данные системы различаются поддерживаемыми платформами, способом конфигурации и способом распространения, что будет отмечено в последующий разделах.

1.2.2 Обзор основных возможностей систем-аналогов

ПК «МультиМит Эксперт» – специализированный программный комплекс, предназначенный для решения широкого спектра технологических и учетных задач на предприятиях мясной и рыбной промышленности [6]. Имеет в составе себя несколько модулей, позволяющих полностью автоматизировать процесс переработки от убоя скота до выпуска готовой продукции. В отличие от конкурентов, система обладает модулем «Оптимизация и моделирование рецептур» (рисунок 2). Из имеющегося в наличии сырья программа обеспечивает получение такой рецептуры продукта, в которой, с одной стороны, его качество соответствует предъявляемым к нему требованиям, а с другой стороны – минимизируется его цена.

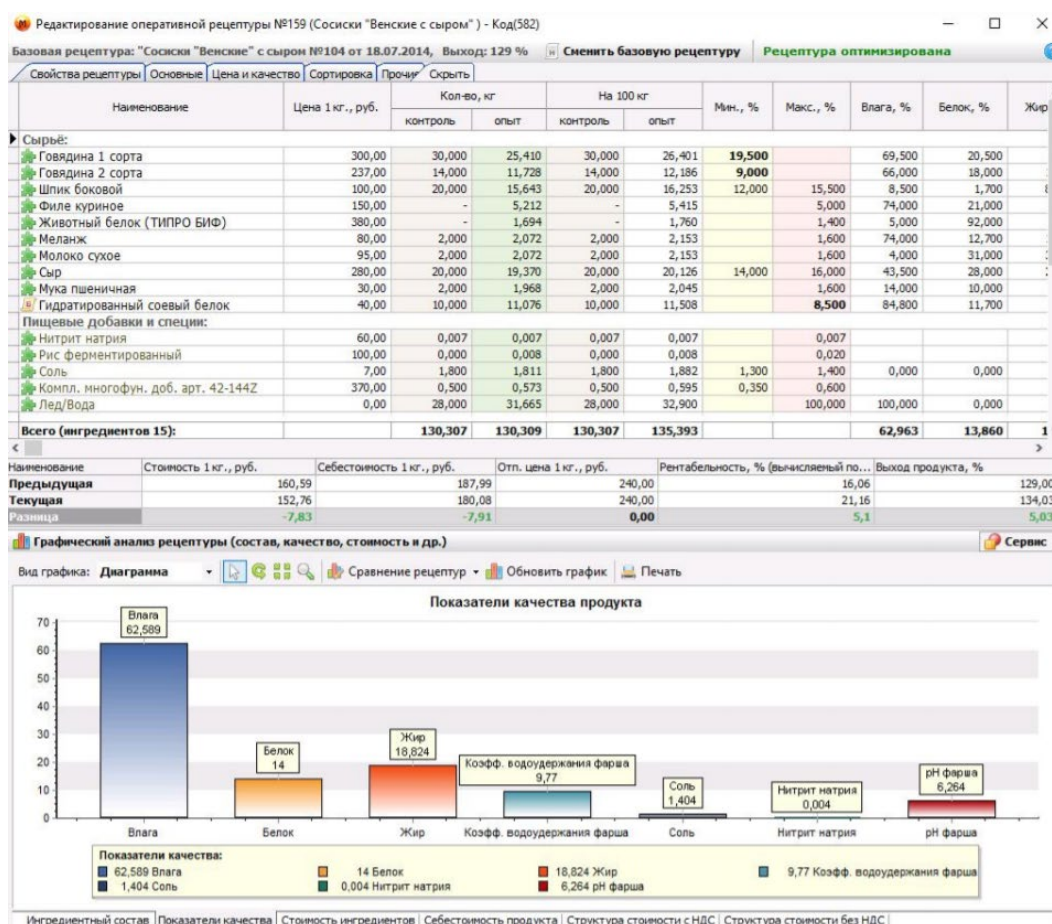


Рисунок 2 – Пример работы модуля «Оптимизация и моделирование рецептур»

Продукт «1С: Мясопереработка MES. Модуль для 1С: ERP» предназначен для решения задач оперативного управления производством на мясоперерабатывающем предприятии, таких как: регистрация фактических данных оперативного производственного учета, контроль соблюдения рецептур, контроль хода технологического процесса и выполнения нормативов по потерям и выходам на различных участках производства [7]. Данный продукт позволяет контролировать соблюдение технического процесса, а также, как и программный комплекс «МультиМит Эксперт», автоматизировать процесс переработки. На рисунках 3 – 5 изображены основные окна программы.

Новый отвес (1С:Предприятие)

Новый отвес

ПФ Колб. Докторская в/с (целлофан)

Общие данные

Дата выработки: 04.05.2017

Дата отвеса: 04.05.2017 20:07:00

> Брак: нет

> Партии

> Характеристики

Количество

Вес брутто 271,000

Вес без брака 271,000

Вес тары 131,000

Вес брутто	Вес тары	Вес нетто	Остаток
271,000	131,000	140,000	

Рисунок 3 – Процесс создания отвеса в системе «1С: Мясопереработка MES»

Заказ №0000-000002 на 02.04.17 от 02.04.2017 0:00:00 - 1С:ERP Управление предприятием 2 (1С:Предприятие)

Колбаса вареная ДОКТОРСКАЯ (целлофан) гмс

Параметры сборки | Параметры номенклатуры

Коробов:	1 / 2	Заявлено:	10,000	Вес (нетто):	2,000	Вложенность (Квант):	кг	Вес Н:	0
Вес короба:	0,301	Отгружено:	7,000	Вес (брутто):	2,000	min	5,000	Вес Б:	0
Вес упаковки:		Осталось:	3,000	Вложенность:		max	5,250	Штук:	0
						4,250			

Весы не подключены

Вес: **0,000 кг** | Тара: **Короб картонный № 1 ФИКС**

Дата чеков: **02.04.2017** | На экран: ☐ Блокировка

Номенклатура	Вс...	Вес ...
Колбаса вареная ДОКТОРСКАЯ (целлофан) гмс	1,000	1,000
Колбаса вареная ДОКТОРСКАЯ (целлофан) гмс	1,000	1,000

Вверх | Вниз | -1 (-) | Очистить СЧ-к

АВТО Вес (F3) | ЭТИКЕТКА ВКЛ/ВЫКЛ (F4) | Печать/повтор (ПРОБЕЛ)

Суммарная этикетка (F12) | Взять вес (F5) | Предыдущая суммарная эти... | Записать (F9) | Ввести вес/кол вручную (F8)

Рисунок 4 – Комплектация заказа в системе «1С: Мясопереработка MES»

Настройки и справочники оперативного учета мясопереработки

Основные параметры системы

[Параметры оперативного учета](#)

Основные параметры модуля оперативного учета

[Параметры мониторов на складах](#)

Параметры мониторов: охлаждения, осадки, буферного склада, выполнения заказов

Нормативы и потери

[Нормативы потерь СК](#)

Нормативные значения потерь в процентах по сравнению с весом сырокопченой колбасы до помещения ее на осадку.

[Нормативы потерь при дефростации](#)

Допустимые потери при выполнении операции дефростации для конкретного вида сырья, произведенного на определенном заводе под определенным брендом.

Параметры сырья

[Бренды](#)

Бренды производителей сырья

[Сортность мяса](#)

Перечень сортов мясосырья, используемых на предприятии

[Сырьевые группы](#)

Перечень предопределенных сырьевых групп, используемых в мясопереработке. Используется

Условия хранения

[Влажность](#)

Перечень условий хранения по влажности

[Контрольные точки](#)

Перечень контрольных точек

[Виды тары](#)

Перечень видов тары

[Нормы хранения и потери на буферном складе](#)

Предназначен для установки норм потерь и сроков хранения продукции на производственных складах

[Нормативы потерь при термообработке](#)

Предназначен для указания нормативов и допустимых потерь при выполнении операций термообработки

[Заводы](#)

Заводы производители сырья

[Состояния упаковок](#)

Справочник содержит перечень состояний упаковки, используемых на предприятии

[Температура](#)

Перечень условий хранения по температуре

Рисунок 5 – Возможности настройки в системе «1С: Мясопереработка MES»

«Меркурий-ERP» – это универсальная складская программа. Он содержит широкий перечень инструментов для ведения складского учета как на небольших складах, так и в крупных распределительных центрах. Данная система поддерживает процесс переработки от прихода материалов на склад (рисунок 6) до непосредственно производства (рисунок 7) и последующей отгрузки [8].

Приход №4

Склад: Склад свинины

Номер прихода: 4 Дата: 14.12.2016 Договор: Договор поставки свинины

Номер накладной: 1 Дата: 14.12.2016 Организация: Масоконбинат ЦФО:

Счет-фактура: 1 Дата: 14.12.2016 Статья бюджета:

Поставщик: ООО "Оптовый поставщик свинины" Код прихода: Консигнация

Филиал: ООО "Оптовый поставщик свинины" Конс. в днях: 21 дн до 04.01.2017

Ключ товара	Наименование	Ед.изм.	Кол-во	Цена с НДС	Сумма с НДС	Остаток
210	С СЫРЬЕ СВИНИНА	кг	120	100,00	12 000,00	120

Итого по приходу: 120 12 000

Отмечено: 0 Менеджер:

С СЫРЬЕ СВИНИНА

Автор: Главный администратор
Создан: 14.12.2016 14:55

Отменить проведение Сохранить Выход

Рисунок 6 – Процесс прихода материала в системе «Меркурий-ERP»

Правка акта производства

Номер: 1 Дата: 14.12.2016 Доп. затраты: 0,00 Сумма с НДС: 582,82000

ТТК: МН-К Колбаса Докторская вареная 1 кг Тип ТТК: Сборная Разборная

Продукты - будут произведены Приход №5 от 2016-12-14

Продукты (изделия, блюда) Треки продуктов

Наименование	Ед. изм.	Выход
МН-К Колбаса Докторская вареная 1 кг	шт	5

ИТОГО 5,0000

Компоненты - будут списаны Расход №1 от 2016-12-14

Компоненты (расходные материалы) Треки компонентов

Наименование	Ед. изм.	Расход	По ТТК	Просл.к.	Учит.г.к.	Чер.з.р.в.
Г ПП ВЪРЕЗКА	кг	1,075	1,075	×	×	×
С П/Ф МЯКОТЬ СВИНАЯ	кг	3,6	3,6	×	×	×
Кардамон молотый	кг	0,015	0,015	×	×	×
Соль поваренная	кг	0,028	0,028	×	×	×
Сахар-песок	кг	0,028	0,028	×	×	×
Сухое молоко коровье обезжиренное	кг	0,375	0,375	×	×	×
Яичный желток	кг	0,15	0,15	×	×	×

ИТОГО 5,2710

Примечание:

Автор: Главный администратор

Отменить проведение Сохранить Выход

Рисунок 7 – Процесс создания акта производства в системе «Меркурий-ERP»

«МойСклад» предоставляет широкий спектр возможностей для управления бизнесом. К данным возможностям относятся: складской учет, розничные продажи, работа с клиентами, обработка заказов, сборка и производство, управление финансами. Работа с системой происходит через веб интерфейс (рисунок 8), который имеет возможность гибкой настройки под роль пользователя.

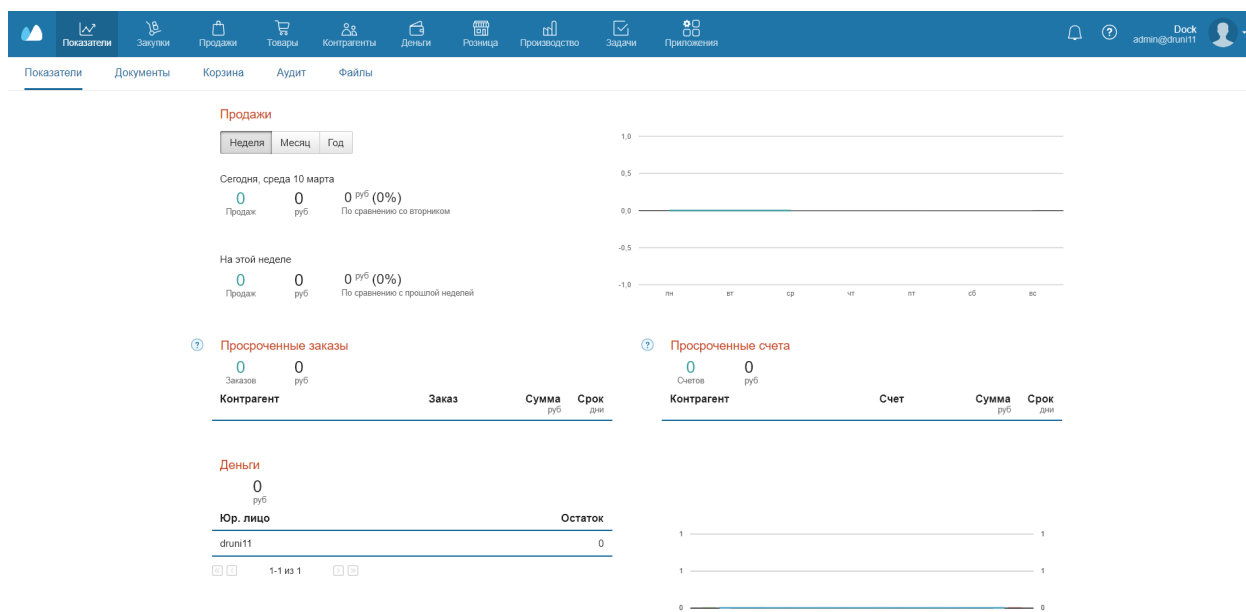


Рисунок 8 – Веб интерфейс «МойСклад»

В отличие от предыдущих систем, «МойСклад» обладает наиболее понятным пользовательским интерфейсом, но не предназначен на прямую для функционирования на мясоперерабатывающих предприятиях.

1.2.3 Результаты сравнительного анализа систем-аналогов

Оцениваться характеристики системы будут по 3-х бальной шкале (+, +-, -), где «+» – это наивысшее значение, означающее полное соответствие характеристике, а «-» – недостаточное удовлетворение данной характеристикой или полное ее отсутствие. В таблице 7 приведено сравнение вышеописанных систем по основным характеристикам.

Таблица 7 – Сравнение характеристик анализируемых систем

Система / Характеристики	ПК «МультиМит Эксперт»	1С: Мясопереработка	Меркурий-ERP	МойСклад
Программные характеристики систем				
Наличие web-версии	–	+	–	+
Мультиплатформенность	–	+	–	+
Бесплатная версия	+–	+–	+–	+
Функциональные характеристики систем				
Учет специфических процессов мясопереработки	+	+	+	–
Учет потерь при производстве	+	+	+	–
Модуль оптимизации рецептур	+	–	–	–
Работа со складом	+	+	+	+

Как видно из сравнения – рассмотренные системы либо имеют слишком углубленный функционал и тяжелы для восприятия рядовым пользователем, либо слишком общий функционал и не учитывают всех тонкостей мясопереработки. В разрабатываемой системе предполагается реализовать основной функционал, присутствующий в вышеперечисленных системах, а именно:

- производственный учет;
- анализ деятельности предприятия;
- контроль хода технического процесса.

Дополнительно к этому информационная система будет проектироваться как веб приложение, что позволит пользоваться ей без предварительной установки и настройки. Также, планируется реализовать функционал не только для работников предприятия, но и для клиентов, чтобы позволить им удаленно просматривать наличие продуктов и актуальные цены.

1.3 Постановка задачи на разработку

1.3.1 Цели разработки и назначение

Целью разработки является создание программного обеспечения для автоматизации работы со складом и оптимизации рабочих процессов. Это позволит значительно увеличить скорость выполнения рутинных задач, уменьшит влияние человеческого фактора благодаря встроенным проверкам, а также позволит анализировать информацию, собранную в ходе работы системы.

Интерфейс должен быть доступным и интуитивно понятным для рядового пользователя.

1.3.2 Описание действующих лиц предметной области

На основании анализа предметной области и назначения разрабатываемой системы, можно выделить список основных действующих лиц. Список действующих лиц и их описание представлено в таблице 8.

Таблица 8 – Действующие лица предметной области

Название	Описание
Работник склада	Осуществляет прием материалов на склад, а также может осуществлять основные операции управления складом
Работник цеха	Занимается переработкой ресурсов в конечный продукт, либо в новые материалы для последующей переработки
Менеджер по продажам	Осуществляет реализацию товара со склада продукции
Администратор	Анализирует информацию, хранящуюся в системе, а также может выполнять все функции остальных групп пользователей
Клиент	Может просматривать актуальную информацию о стоимости и наличии продукции

Для описанных выше действующих лиц и будет производиться автоматизаций функций разрабатываемой системы.

1.3.3 Перечень автоматизируемых функций

Исходя из выделенной структуры рассматриваемой предметной области и поставленных задач на проектирование необходимо выделить функции, которые необходимо автоматизировать. Функции, выделенные для автоматизации, представлены в таблице 9.

Таблица 9 – Перечень автоматизируемых функций

№	Название	Описание
1	Добавление новых позиций в систему	Добавление новых позиций в систему, будь то конечный продукт или входной материал
2	Добавление поставки	Добавление в систему новой поставки. В поставке может содержаться несколько позиций, которые автоматически добавляются на склад

Продолжение таблицы 9

№	Название	Описание
3	Добавление материала на склад	Добавление материала на склад ресурсов. Каждая операция имеет свою причину, которая логируется в системе
4	Забор материалов со склада	Забор материала со склада ресурсов. Как и добавление, каждый забор имеет свою причину
5	Добавление новой переработки	Добавление новой переработки в систему. В переработки фиксируются ресурсы пришедшие на вход, тип переработки и примерное количество выходного продукта с учетом потерь
6	Завершение переработки	Завершение переработки, находящейся в активном состоянии. Переработка переходит в завершенное состояние, а переработанная продукция попадает на склад
7	Добавление ресурсов на склад продукции	Добавляет продукцию на склад. На данный склад продукция может попасть только в процессе переработки, поэтому здесь нет необходимости хранить причину поступления
8	Продажа продукции	Продажа продукции со склада. Списывает продаваемое количество продукции со склада, занося при этом лог о продаже в систему
9	Сбор и анализ информации о движении ресурсов	Сбор и анализ информации для администратора предприятия. На основе собранных в процессе движения ресурсов данных, производится построение различных визуальных представлений (графики, таблицы), с помощью которых администратор анализирует деятельность предприятия

Данный список будет детализироваться в процессе непосредственной разработки системы.

1.4 Планирование и функционально-стоимостной анализ процесса разработки

1.4.1 Основы функционально-стоимостного анализа

Основная цель функционально-стоимостного подхода состоит в разработке элементных и функциональных структур таким образом, чтобы технико-экономические параметры рассматриваемого объекта были оптимальны (если создается новый объект) либо чтобы значимость для объекта и затраты на осуществление функции были пропорциональны.

Функциональный подход позволяет представить систему как комплекс выполняемых функций. Главная задача – не усовершенствование системы, а поиск наилучших способов выполнения главной функции.

Первая часть ФСА – функциональный анализ нетехнических систем – имеет определенную историю и опыт, применяется на практике, чего не скажешь о второй его части – стоимостном анализе (применительно к определению стоимости функций нетехнических систем).

Используя отдельные подходы и элементы различных научных дисциплин, ФСА вместе с тем имеет самостоятельное научное и практическое значение. Изучение «входов» и «выходов» ФСА позволяет разобраться в его взаимосвязях с другими науками и научными дисциплинами. Самостоятельность и ценность его заключаются в том, что, имея главную цель – оптимизацию соотношения качества и затрат на выполнение функций исследуемого объекта, при ФСА соединяют в единую систему анализ функций и затрат на их осуществление, разработку альтернативных и выбор наилучших экономических вариантов реализации функций. Причем оценку уровня соотношения «качество – затраты» и меры по повышению этого уровня осуществляют не только во взаимосвязи, но и во взаимодействии [9].

1.4.2 Планирование процесса разработки

Планирование – это процесс создания и принятия целевых установок качественного и количественного характера, а также выявления путей по наиболее эффективному их достижению.

Работа по составлению плана включает в себя все стадии создания и выполнения проекта, начиная с анализа имеющихся систем аналогов и технического задания и заканчивая завершением проекта.

На стадии планирования устанавливаются основные параметры осуществления проекта. К ним относятся:

- продолжительность каждого контролируемого элемента проекта;
- необходимость в ресурсах;
- сроки поставки необходимого оборудования, комплектующих, материалов, сырья и т.п.;
- сроки и объемы привлечения организаций (строительных, проектных и т.п.).

Для моделирования процесса разработки была использован специальный программный продукт BPwin.

BPwin позволяет создавать модели процессов и поддерживает в одной модели в дополнение к IDEF0 еще два стандарта (нотации) моделирования – DFD и IDEF3. Каждая из этих трех нотаций позволяет рассмотреть различные стороны деятельности предприятия. В данном случае была использована диаграмма IDEF0.

Диаграммы IDEF0 предназначены для описания бизнес-процессов на предприятии, они позволяют понять, какие объекты или информация служат сырьем для процессов, какие результаты производят работы, что является управляющими факторами и какие ресурсы для этого необходимы. Нотация IDEF0 позволяет выявить формальные недостатки бизнес-процессов, что существенно облегчает анализ деятельности предприятия [10].

Набор структурных компонентов языка, их характеристики и правила, определяющие связи между компонентами, представляют собой синтаксис языка. Компоненты синтаксиса IDEF0 – блоки, стрелки, диаграммы и правила.

Блоки представляют функции, определяемые как деятельность, процесс, операция, действие или преобразование. Блок описывает функцию. Внутри каждого блока помещается его имя и номер. Имя должно быть активным глаголом или глагольным оборотом, описывающим функцию.

Стрелка формируется из одного или более отрезков прямых и наконечника на одном конце. Сегменты стрелок могут быть прямыми или ломаными. Они показывают, какие данные или материальные объекты должны поступить на вход функции для того, чтобы эта функция могла выполняться.

Стрелки, представляя множества объектов, в зависимости от того, в какую грань блока (прямоугольника работы) они входят или из какой грани выходят, делятся на пять видов:

- 1) входа (входят в левую грань работы) – изображают данные или объекты, изменяемые в ходе выполнения работы;
- 2) управления (входят в верхнюю грань работы) – изображают правила и ограничения, согласно которым выполняется работа;
- 3) выхода (выходят из правой грани работы) – изображают данные или объекты, появляющиеся в результате выполнения работы;
- 4) механизма (входят в нижнюю грань работы) – изображают ресурсы, необходимые для выполнения работы, но не изменяющиеся в процессе работы (например, оборудование, людские ресурсы и т.д.);
- 5) вызова (выходят из нижней грани работы) – изображают связи между разными диаграммами или моделями, указывая на некоторую диаграмму, где данная работа рассмотрена более подробно.

На рисунках 9, 10 представлены контекстная диаграмма бизнес-процесса «Выпускная квалификационная работа» и ее детализация с помощью диаграммы декомпозиции первого уровня соответственно [11].

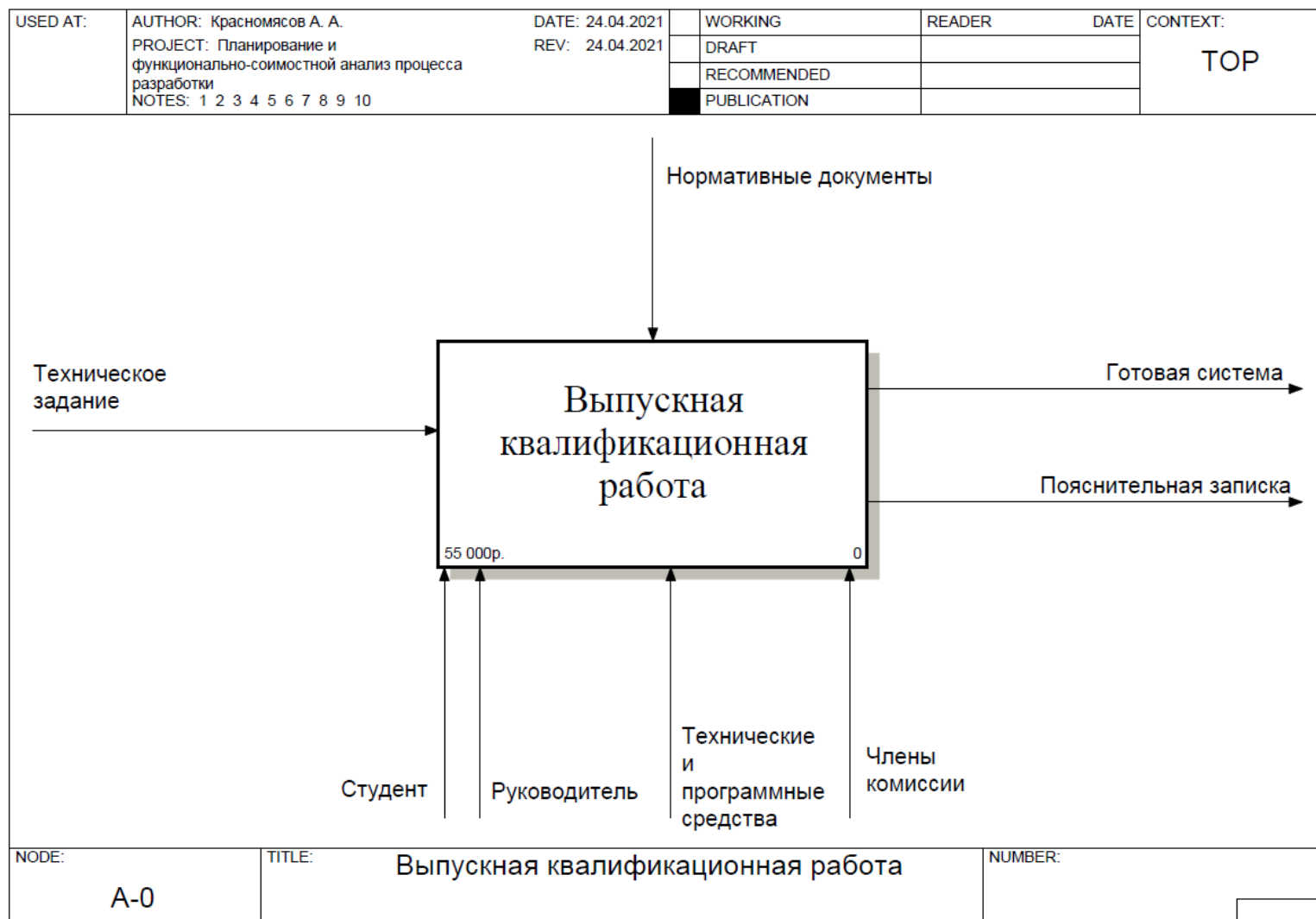


Рисунок 9 – Контекстная диаграмма бизнес-процесса «Выпускная квалификационная работа»

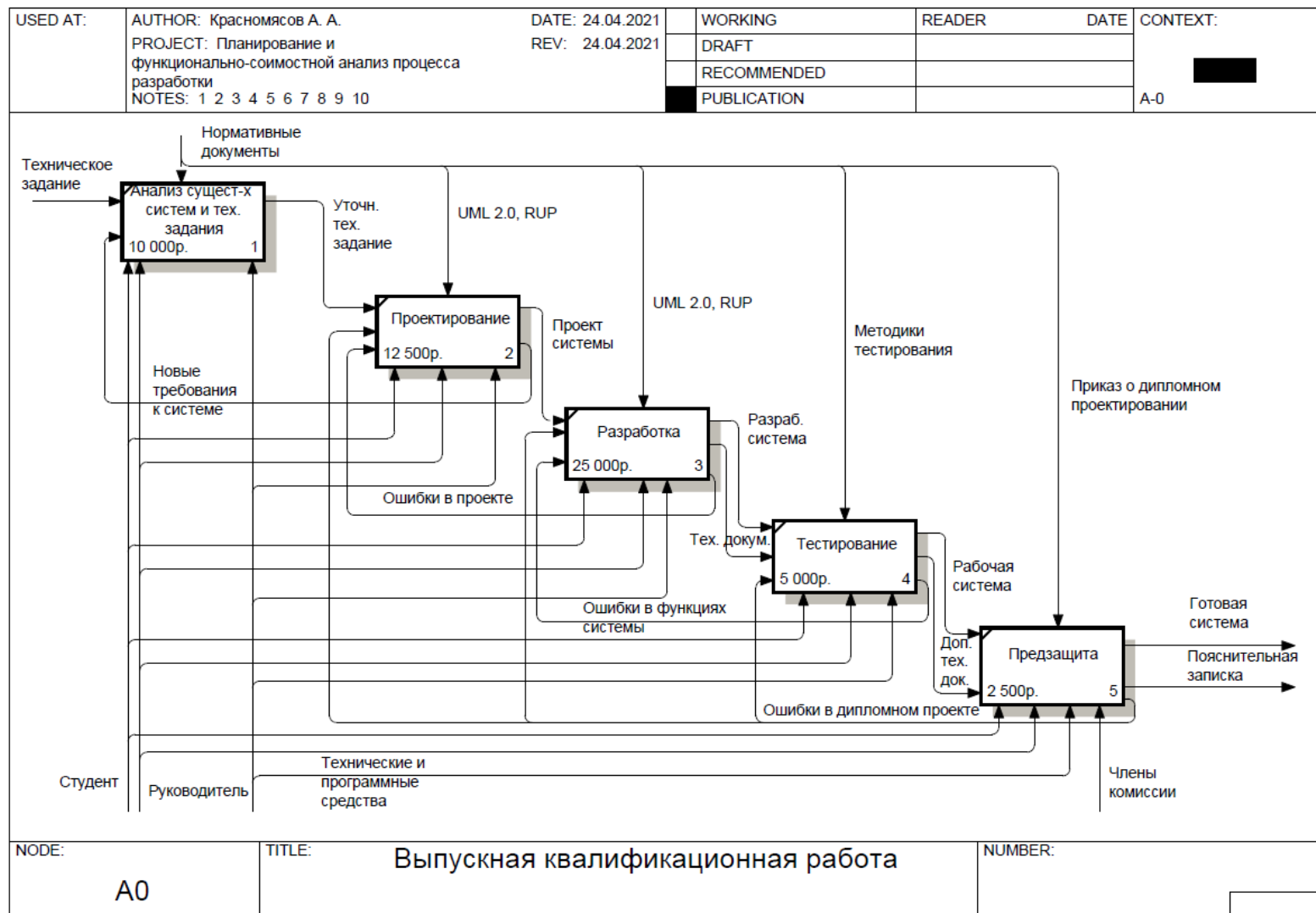


Рисунок 10 – Диаграмма декомпозиции

На основании выделенных бизнес-процессов будет произведен анализ временных и финансовых затрат в последующей главе.

1.4.3 Анализ временных и финансовых затрат

На основе диаграммы декомпозиции было рассчитано время для каждого этапа разработки для одного человека. Для наглядной иллюстрации графика работы над проектом была построена диаграмма Ганта (рисунки 11, 12).

Режим задачи ▾	Название задачи ▾	Длительность ▾	Начало ▾	Окончание ▾	Предшественники ▾
→	Анализ предметной области и постановка задачи	4 дней	Чт 01.04.21	Вт 06.04.21	
→	Проектирование	5 дней	Ср 07.04.21	Вт 13.04.21	1
→	Разработка	10 дней	Ср 14.04.21	Вт 27.04.21	2
→	Тестирование	2 дней	Ср 28.04.21	Чт 29.04.21	3
→	Предзащита	1 день	Пт 30.04.21	Пт 30.04.21	4

Рисунок 11 – Таблица, описывающая диаграмму Ганта

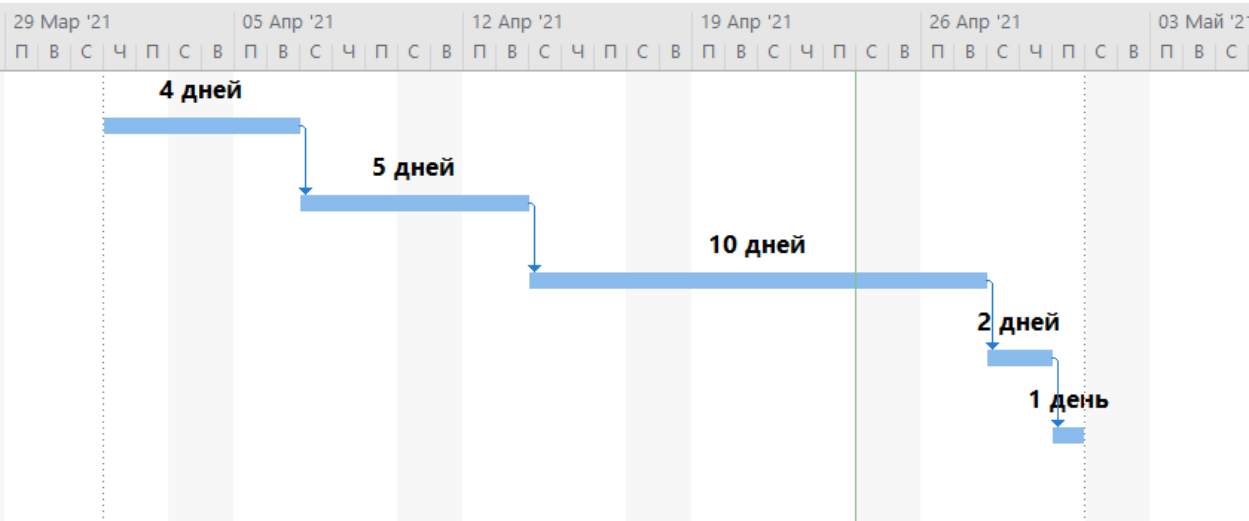


Рисунок 12 – Диаграмма Ганта

Средняя ежемесячная зарплата веб-программиста стажера в РФ, находится в диапазоне 35 000 – 40 000 рублей. По этим данным средний часовой тариф веб-программиста составляет 312,5 рублей. Основываясь на этих данных и на временных затратах, рассчитанных ранее, была построена сводная таблица (таблица 10), показывающая финансовые затраты разработку.

Таблица 10 – Финансовые затраты на разработку

Задача	Длительность	Тариф, руб. / ч.	Общие затраты, руб.
Анализ предметной области и постановка задачи	4 д. (32 ч.)	312,5	10 000
Проектирование	5 д. (40 ч.)	312,5	12 500
Разработка	10 д. (80 ч.)	312,5	25 000
Тестирование	2 д. (16 ч.)	312,5	5 000
Предзащита	1 д. (8 ч.)	312,5	2 500
Итого:			55 000

В итоге на анализ предметной области, проектирование и разработку проекта «Web-приложение для учета и анализа деятельности малого предприятия по производству мясной продукции», а также его тестирование и предзащиту было суммарно затрачено 55000 рублей. Исходя из этих данных можно сделать вывод, что создание web-приложения является экономически выгодным.

2 Анализ требований на разработку

2.1 Функциональные требования

Система должна обеспечить удобную работу пользователей (в первую очередь работников предприятия) с имеющимися в системе данными. Несмотря на простоту интерфейса, приложение должно предоставлять все необходимые функции, характерные для предметной области. Также в системе должен реализован функционал по анализу деятельности предприятия, для оптимизации работы администратора.

Для описания системы на концептуальном уровне была использована диаграмма прецедентов в UML. UML (Unified Modeling Language, UML) – это универсальный язык визуального моделирования систем. Чаще всего UML ассоциируется с моделированием объектно-ориентированных систем, однако он имеет более широкое применение, благодаря свойству расширяемости. UML объединил лучшие современные технические приемы моделирования и разработки программного обеспечения. Однако важно понимать, что UML не предлагает какой-либо методологии моделирования, а предоставляет лишь визуальный синтаксис для создания моделей. UML может использоваться со всеми существующими методологиями, но только UP (Unified Process) использует его в качестве базового синтаксиса визуального моделирования.

Диаграмма прецедентов рассматривает корпоративные бизнес-процессы верхнего уровня с внешней точки зрения и позволяет понять, как выглядит деятельность компании «со стороны». Деятельность по моделированию прецедентов является частью рабочего потока определения требований. Большая часть работы в рабочем потоке определения требований осуществляется в фазах Начало и Уточнение жизненного цикла UP проекта. Основные деятельности UP – это выявление актеров, прецедентов и детализация прецедента. Прецеденты фиксируют функциональные требования и поэтому не эффективны для систем, в которых доминируют нефункциональные требования [12].

Разработанная диаграмма представлена на рисунке 13.

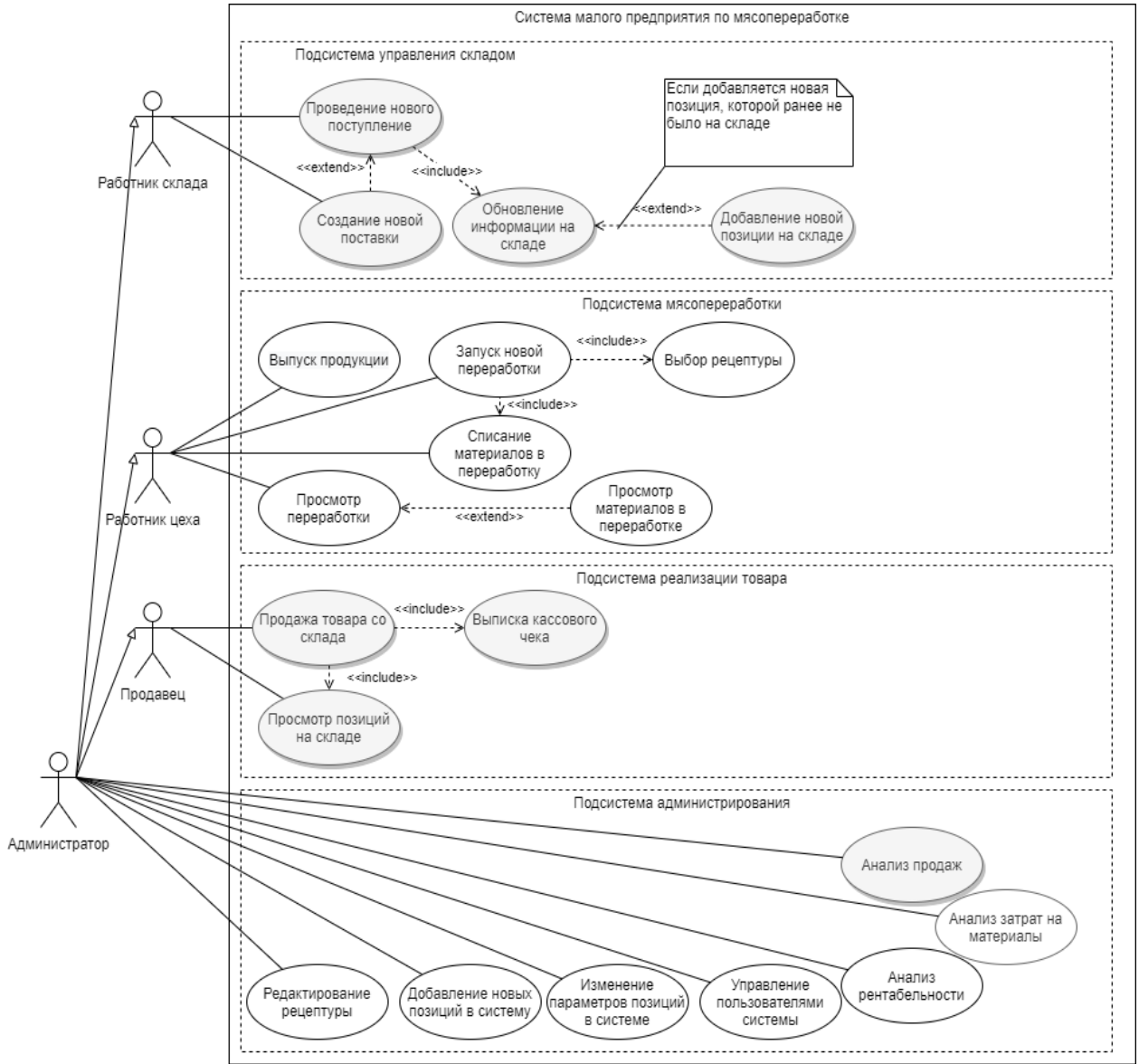


Рисунок 13 – Диаграмма прецедентов для разрабатываемой системы

На диаграмме выделены варианты использования, которые были реализованы в рамках данной выпускной работы. Данные процессы являются основными и достаточными для нормального функционирования базового варианта системы. Остальные функции были оставлены на дальнейшее развитие приложения. Далее приведено более подробное описание для вариантов использования с наиболее сложной структурой (таблица 11).

Таблица 11 – Описание прецедентов системы

№	Название прецедента	Главные актеры	Краткое описание	Основной поток	Ссылки на другие варианты использования
1	Проведение нового поступления	Работник склада	Работник склада фиксирует новое поступление в системе	<ol style="list-style-type: none"> 1. Работник выбирает опцию «Провести новое поступление»; 2. Работник выбирает нужную поставку из списка, либо добавляет новую; 3. Работник вводит полученные материалы и их количество; 4. Система фиксирует поступление материалов; 5. Система обновляет информацию на складе; 6. В случае если добавляемый ресурс ранее не был добавлен на склад, система автоматически создает новую позицию. 	<p>Включения: Обновление информации на складе</p> <p>Расширения: Создание новой поставки</p>
2	Выпуск продукции	Работник цеха	Работник производит выпуск продукции, завершая переработку и передавая продукцию на склад.	<ol style="list-style-type: none"> 1. Работник открывает список текущих переработок; 2. Работник открывает интересующую его переработку; 3. Работник выбирает опцию «Завершить переработку»; 4. Система выводит предполагаемый выход переработки; 5. Работник уточняет данные и выбирает опцию «Завершить»; 6. Система изменяет статус переработки на «Завершено» и актуализирует данные на складе ресурсов и складе продукции. 	<p>Включения: нет</p> <p>Расширения: нет</p>

Продолжение таблицы 11

№	Название прецедента	Главные актеры	Краткое описание	Основной поток	Ссылки на другие варианты использования
3	Запуск новой переработки	Работник цеха	Работник цеха начинает новую переработку, фиксируя ее параметры в системе	<ol style="list-style-type: none"> 1. Работник выбирает опцию создания новой переработки; 2. Пользователь выбирает рецептуру из списка, по которой будет производиться переработка; 3. Пользователь указывает материалы и их количество, которые забираются на переработку; 4. Система списывает указанные ресурсы со склада; 5. Система добавляет переработку в список переработок. 	<p>Включения: Выбор рецептуры; Списание материалов в переработку</p> <p>Расширения: нет</p>
4	Просмотр переработки	Работник цеха	Работник просматривает детали конкретной переработки	<ol style="list-style-type: none"> 1. Пользователь выбирает интересующую его переработку; 2. Система выводит основные данные по переработке; 3. Пользователь может просмотреть дополнительную информацию по перерабатываемым ресурсам. 	<p>Включения: нет</p> <p>Расширения: Просмотр материалов в переработке</p>
5	Продажа товара со склада	Продавец	Продавец производит продажу товара	<ol style="list-style-type: none"> 1. Система отображает список товаров, доступных на складе товаров; 2. Продавец выбирает продаваемые товары и указывает их количество; 3. Система производит проверку количества товаров; 4. При удачно проверке создается новая запись продажи, система формирует кассовый чек и обновляет информацию на складе. 	<p>Включения: Выписка кассового чека;</p> <p>Просмотр позиций на складе</p> <p>Расширения: нет</p>

Окончание таблицы 11

№	Название прецедента	Главные актеры	Краткое описание	Основной поток	Ссылки на другие варианты использования
6	Анализ продаж	Администратор	Администратор анализирует продажи в системе за определенный срок	<ol style="list-style-type: none"> 1. Администратор выбирает опцию анализа продаж; 2. Администратор выбирает временной интервал для анализа; 3. Администратор выбирает продукты, по которым необходимо проводить анализ; 4. Система отображает сгруппированную информацию в виде графиков и сводных таблиц. 	<p>Включения: нет</p> <p>Расширения: нет</p>
7	Анализ затрат на материалы	Администратор	Администратор производит анализ затрат на материалы за определенный срок	<ol style="list-style-type: none"> 1. Администратор выбирает опцию анализ затрат на материалы; 2. Администратор выбирает временной интервал для анализа; 3. Система отображает сгруппированную информацию в виде графиков и сводных таблиц. 	<p>Включения: нет</p> <p>Расширения: нет</p>

Наиболее сложные варианты использования были описаны отдельно в виде диаграмм деятельности. К таким прецедентам относятся: «Проведение нового поступления», «Запуск новой переработки» и «Продажа товара со склада». Диаграмма деятельности для прецедента «Проведение нового поступления» представлена на рисунке 14.

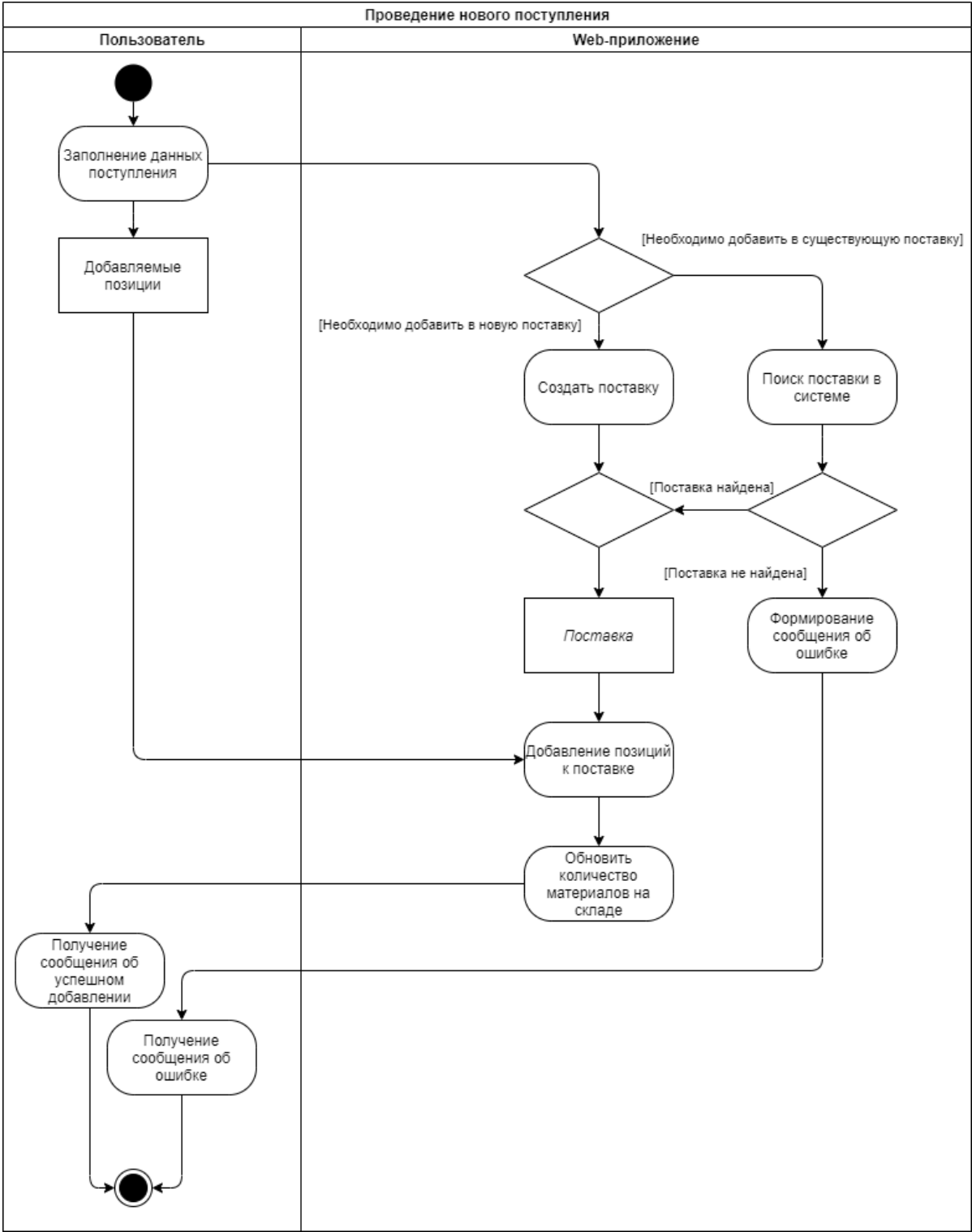


Рисунок 14 – Диаграмма деятельности процесса «Проведение нового поступления»

Для прецедентов «Запуск новой переработки» и «Продажа товара со склада» диаграммы деятельности изображены на рисунках 15, 16 соответственно.

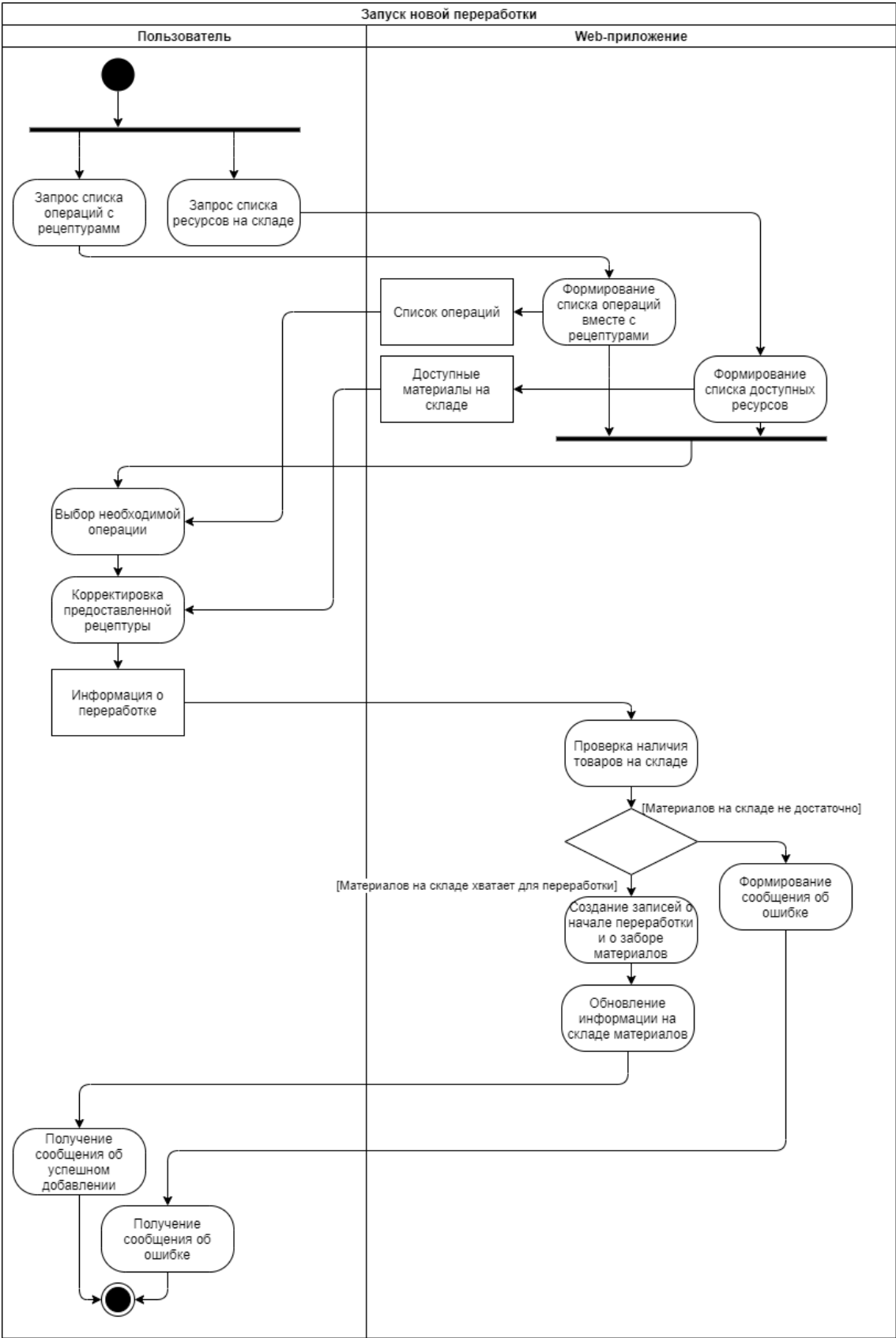


Рисунок 15 – Диаграмма деятельности процесса «Запуск новой переработки»

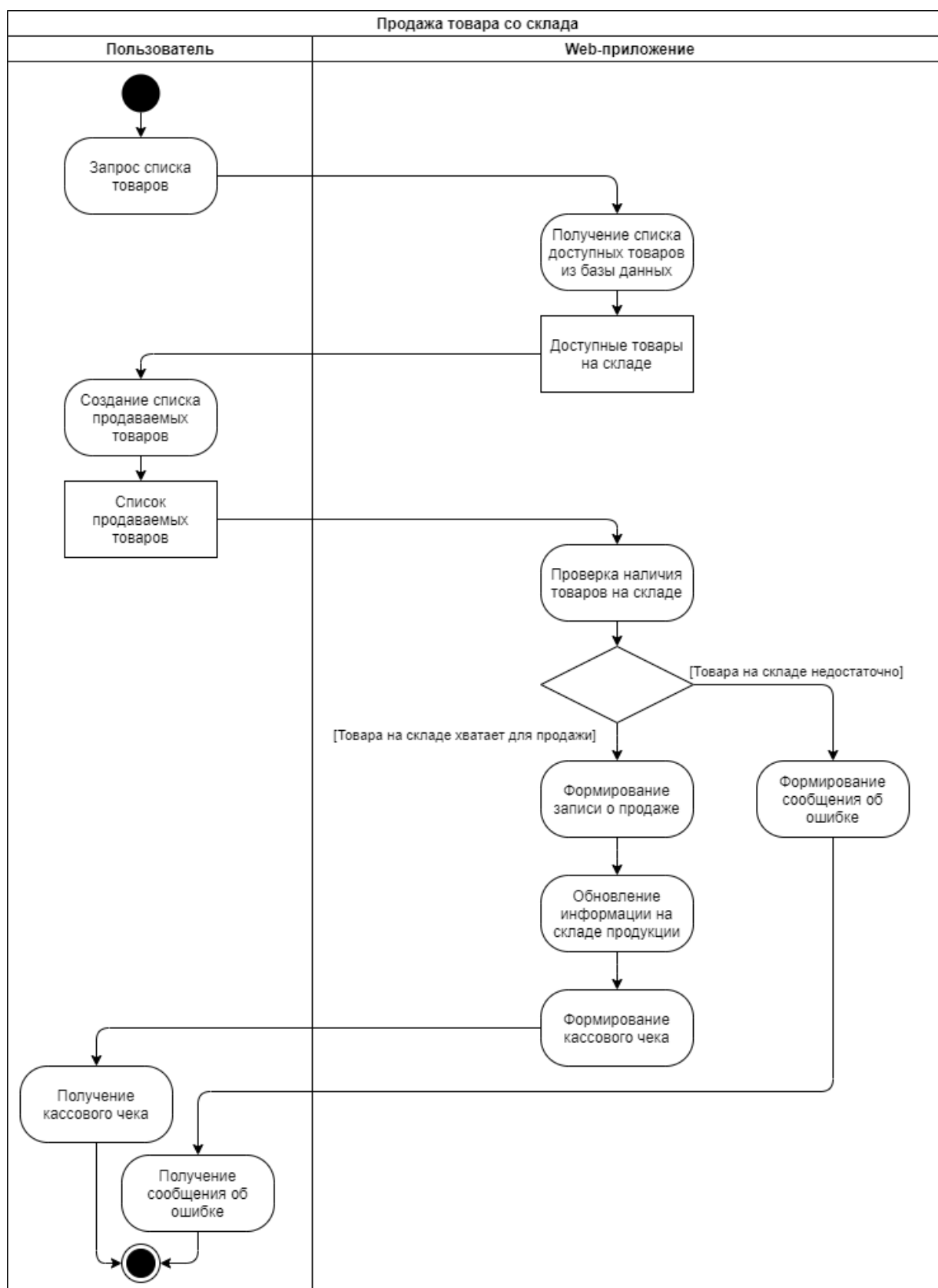


Рисунок 16 – Диаграмма деятельности процесса «Продажа товара со склада»

Таким образом, были сформулированы и детализированы функциональные требования на разработку системы.

2.2 Построение концептуальной модели

Концептуальная модель – модель предметной области, состоящей из перечня взаимосвязанных понятий, используемых для описания этой области, вместе со свойствами и характеристиками, классификацией этих понятий, по типам, ситуациям, признакам в данной области и законов протекания процессов в ней [13].

Концептуальная модель относится к моделям, которые формируются после процесса концептуализации или обобщения. Концептуальные модели часто являются абстракциями вещей в реальном мире. Для различных этапов формирования концепта актуальны семантические исследования. Семантика – это понятие, значение, которое мыслящие существа придают различным элементам своего опыта.

Основная задача концептуальной модели – передать фундаментальные принципы и основные функциональные возможности системы, которую она представляет. Кроме того, концептуальная модель должна быть разработана таким образом, чтобы обеспечить легко понятную системную интерпретацию для пользователей модели.

Концептуальная модель играет важную роль в общем жизненном цикле развития системы. Если концептуальная модель не будет полностью разработана, то реализация фундаментальных свойств системы может не быть реализована должным образом, что приведет к будущим проблемам или недостаткам системы.

Для рассматриваемой предметной области была разработана и построена концептуальная модель. Данная модель отражает основные сущности, участвующие в работе системы. Также между сущностями были построены связи, описывающие взаимодействие между ними.

Построенная концептуальная модель рассматриваемой предметной области изображена на рисунке 17

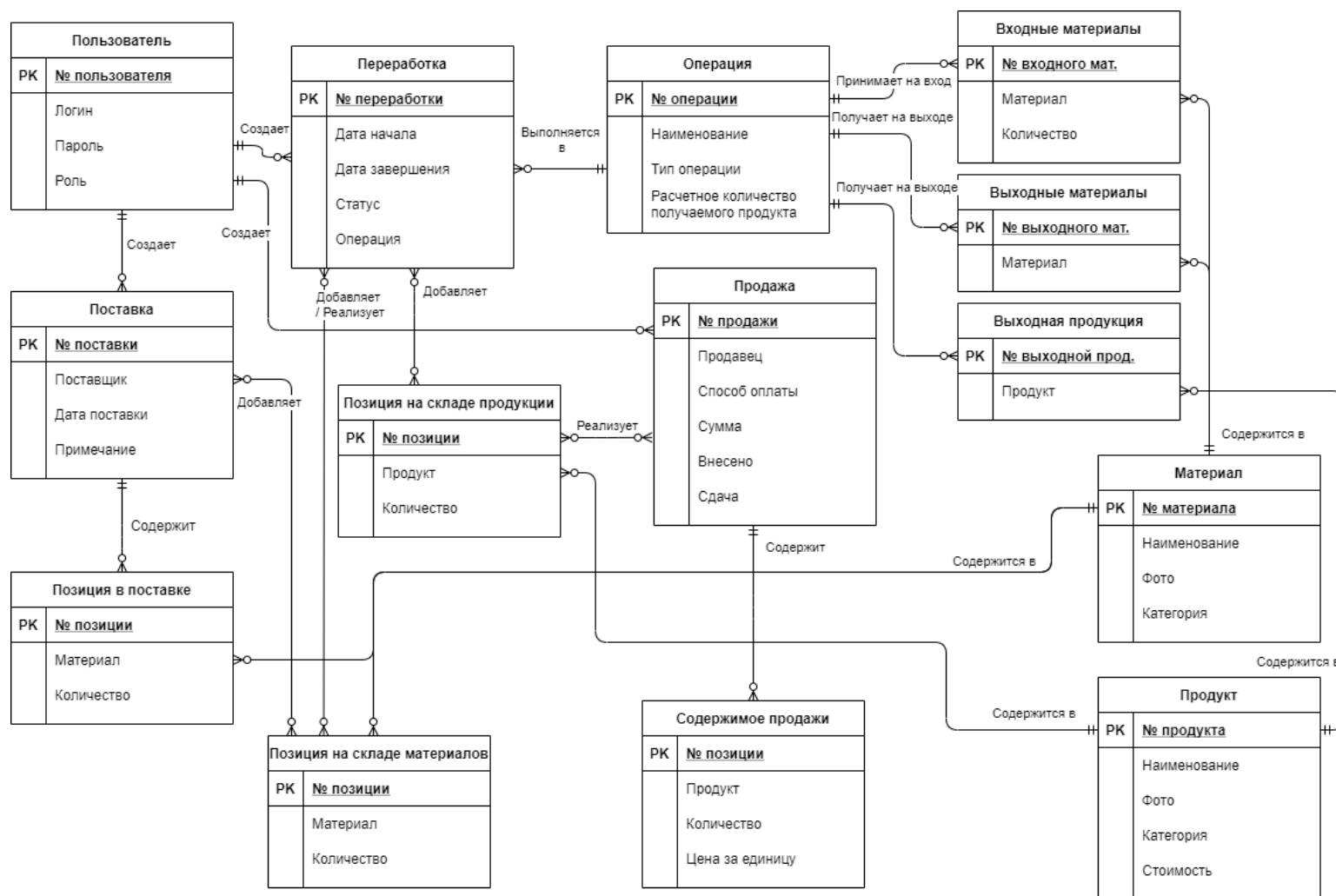


Рисунок 17 – Концептуальная модель предметной области «малое предприятие по производству мясной продукции»

Краткое описание представленной концептуальной модели представлено в таблице 12.

Таблица 12 – Описание концептуальной модели

Сущность	Описание
Пользователь	Хранит информацию с учетными данными пользователей, а также о их полномочиях
Поставка	Хранит поставки материалов
Позиция в поставке	Описывает список материалов, содержащийся в поставке
Позиция на складе материалов	Содержит позиции на складе материалов. Количество материала на складе не может быть отрицательным
Переработка	Содержит список всех текущих, а также завершенных переработок.
Операция	Содержит список операций, характеризующих возможные переработки и их характеристики
Входные материалы	Описывает материалы и их количество, которое должно поступить на вход переработки
Выходные материалы	Описывает материалы и их количество, которое должно быть получено на выходе переработки
Выходная продукция	Описывает продукты и их количество, которое должно быть получено на выходе переработки
Продажа	Хранит осуществленные продажи
Содержимое продажи	Содержит список продуктов, привязанных к продажам.
Позиция на складе продукции	Содержит позиции на складе продукции. Количество продукции на складе не может быть отрицательным
Продукт	Содержит информацию о продукции предприятия
Материал	Содержит информацию о перерабатываемых материалах предприятия

На основании данной модели спроектирована и развернута база данных, по которой в дальнейшем будут созданы соответствующие классы в рамках выбранного языка программирования.

2.3 Требования к обобщенной архитектуре приложения

Система разрабатывается в виде web-приложения с клиент-серверной архитектурой. На рисунке 18 изображена диаграмма развертывания, описывающая архитектуру системы.

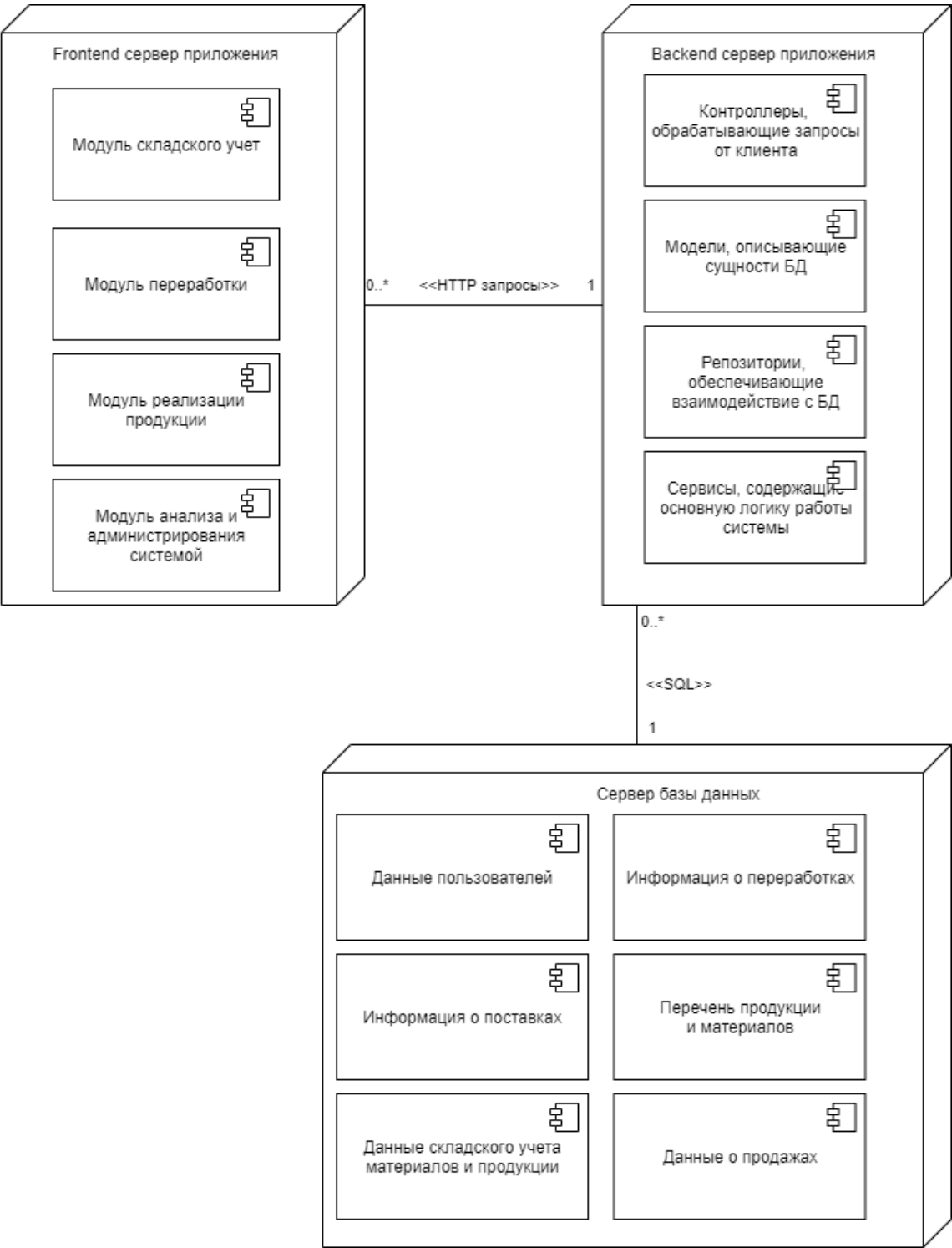


Рисунок 18 – Диаграмма развертывания веб приложения

На выше представленной диаграмме изображены основные компоненты разрабатываемой системы:

- сервер базы данных;
- backend сервер приложения;
- frontend сервер приложения.

Сервер БД служит для обработки SQL запросов от сервера приложения, хранения и предоставления доступа к данным в БД. Здесь хранятся все данные, производимые в процессе работы системы;

Backend сервер обрабатывает запросы клиента в http формате. В данном компоненте находится основная логика работы приложения;

Frontend сервер обеспечивает отображение и работу с данными в удобном для пользователя виде. Данный компонент имеет модульную структуру, где каждый модуль отвечает за отдельную часть системы. Такая структура позволяет удобно разграничить права пользователей в зависимости от их ролей, а также обеспечивает удобный интерфейс.

Таким образом, в системе можно выделить 3 независимых компонента, которые взаимодействуют с собой с помощью соответствующих протоколов. В свою очередь эти компоненты внутри декомпозированы на отдельные модули, что позволяет оптимизировать процессы разработки и последующей поддержки системы.

2.4 Обоснование выбора технологий и средств разработки

2.4.1 Обоснование выбора системы управления базами данных

Для хранения и последующей обработки данных на сервере разрабатываемого приложения была использована свободно распространяемая реляционная СУБД MySQL. MySQL имеет клиент-серверную архитектуру: к серверу MySQL могут обращаться различные клиентские приложения, в том числе с удаленных компьютеров [14]. Данная СУБД обладает рядом особенностей, делающую ее отличным вариантом для разработки приложений:

- обладает всеми техническими характеристиками, необходимыми для работы современного приложения, а именно многопоточность, многопользовательский доступ, быстрое действие, масштабируемость;
- имеет большое количество программных интерфейсов, позволяющих подключаться к БД приложениям, написанных на большинстве популярных языках программирования;
- это кроссплатформенная система (ее можно использовать практически во всех операционных системах);
- имеет развитую систему обеспечения безопасности и разграничения доступа на основе систем привилегий.

Данные особенности выделяют MySQL и делают ее хорошим выбором для разработки бд для своего приложения среди других реляционных баз данных, таких как SQLite, MSSQL, PostgreSQL и др.

2.4.2 Обоснование выбора языковых и инструментальных средств

Существует множество языков, используемых для разработки Rest API серверов. К таким языкам относятся C#, Ruby, Scala, Java и т.д. В данном случае для разработки был выбран объектно-ориентированный язык программирования – Java.

Java – один из самых важных и широко применяемых языков программирования в мире на протяжении многих лет. В отличие от некоторых других языков программирования, влияние Java не только не уменьшилось со временем, а наоборот, возросло. С момента первого выпуска он выдвинулся на передний план программирования приложений для интернета, и каждая последующая версия лишь укрепляла эту позицию. В настоящее время Java по-прежнему остается первым и одним из лучших языков для разработки веб-ориентированных приложений. Проще говоря, большая часть современного кода написана на Java и это свидетельствует об особом значении языка Java для программирования.

Основная причина успеха Java – его гибкость. Начиная с первой версии 1.0, этот язык непрерывно адаптируется к изменениям в среде программирования и подходам к написанию программ, а самое главное – он не просто следует тенденциям в программировании, а помогает их создавать. Способность Java адаптироваться к быстрым изменениям в вычислительной технике служит основной причиной, по которой этот язык программирования продолжает оставаться столь успешным [15].

Для разработки клиентской части были использованы языки Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS), JavaScript – основные языки веб разработки.

HTML расшифровывается как Hyper Text Markup Language, т.е. язык гипертекстовой разметки – основной строительный блок web-страниц, используется для создания и визуального представления.

Язык HTML добавляет разметку в обычный текст. Гипертекст содержит различные ссылки, благодаря которым веб-страницы связываются между собой. С помощью HTML каждый может создавать как статические, так и динамические сайты. HTML является языком, описывающим структуру и семантику содержимого веб-документа. Содержимое веб-страницы размечается с помощью тегов, представляющих HTML-элементы.

CSS (Cascading Style Sheets) – язык таблиц стилей, который позволяет прикреплять стиль (например, шрифты и цвет) к структурированным документам (например, документам HTML и приложениям XML).

Обычно CSS-стили используются для создания и изменения стиля элементов web-страниц и пользовательских интерфейсов, написанных на языках HTML и XHTML, но также могут быть применены к любому виду XML-документа, в том числе XML, SVG и XUL.

Отделяя стиль представления документов от содержимого документов, CSS упрощает создание web-страниц и обслуживание сайтов.

JavaScript – это полноценный динамический язык программирования, который применяется к HTML документу, и может обеспечить динамическую интерактивность на веб-сайтах.

JavaScript сам по себе довольно компактный, но очень гибкий. Разработчиками написано большое количество инструментов поверх основного языка JavaScript, которые разблокируют огромное количество дополнительных функций с очень небольшим усилием.

В процессе разработки были использованы следующие инструментальные средства:

- Navicat Premium 15;
- IntelliJ IDEA 2020.3.2;
- WebStorm 2020.3.3.

Navicat Premium 15 – мультиплатформенный инструмент, созданный для управления базами данных, в т.ч. и «облачными», и предусматривающий одновременное подключение к нескольким из них. Данное программное обеспечение используется для управления и разработки баз данных MySQL, MariaDB, Oracle, SQLite, PostgreSQL и Microsoft SQL Server. Он имеет графический интерфейс пользователя, подобный проводнику, и поддерживает несколько соединений с базами данных для локальных и удаленных баз данных. Его дизайн выполнен в соответствии с потребностями различных аудиторий, от администраторов баз данных и программистов до различных предприятий / компаний, которые обслуживают клиентов и обмениваются информацией с партнерами.

IntelliJ IDEA 2020.3.2 – интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. Каждый компонент IntelliJ IDEA создан для того, чтобы максимально повысить продуктивность разработки. Умный редактор кода в сочетании с эргономичным дизайном делают разработку не только эффективной, но и приятной [15].

WebStorm 2020.3.3 – интегрированная среда разработки на JavaScript, CSS и HTML от компании JetBrains. WebStorm находит применение всем возможностям современной JavaScript-экосистемы. В нем использованы умное авто дополнение кода, проверка ошибок на лету, быстрая навигация по коду и рефакторинг для JavaScript, TypeScript, языков стилей, а также для популярных фреймворков [17].

Также в процессе тестирования и отладки системы была использована программа Postman, используемая для тестирования API запросов.

2.4.3 Обоснование выбора фреймворка для разработки серверного и клиентского приложения

Backend разработан на языке программирования Java с использованием фреймворка Spring.

Spring – это свободно распространяемый фреймворк, созданный Родом Джонсоном (Rod Johnson) и описанный в его книге «Expert One-on-One: J2EE Design and Development». Он был создан с целью устранить сложности разработки корпоративных приложений и сделать возможным использование простых компонентов JavaBean для достижения всего того, что ранее было возможным только с использованием EJB. Однако область применения Spring не ограничивается разработкой программных компонентов, выполняющихся на стороне сервера. Любое Java-приложение может использовать преимущества фреймворка в плане простоты, тестируемости и слабой связанности

Он позволяет работать с реляционной базой как с объектами при помощи библиотеки Spring Data JPA, просто внедрять зависимости с помощью maven, реализовывать защиту и авторизацию с помощью Spring Security [18].

Frontend написан на nodeJs с использованием библиотеки React и прикладных библиотек, в том числе AJAX для формирования запросов к серверу, Victory для построения интерактивных визуализаций данных, mobx обработки состояний приложения. React позволяет безболезненно создавать

интерактивные UIs. С помощью него можно разрабатывать простые представления для каждого состояния, и React будет эффективно обновлять и визуализировать только нужные компоненты при изменении данных.

2.5 Требования к аппаратно-программному обеспечению

2.5.1 Требования к аппаратно-техническому обеспечению

Требования к аппаратно-техническому обеспечению описывают минимальные технические характеристики, необходимые для корректной работы компонентов системы. Определение технических требований является важным этапом при выборе и настройке хостинга для финальной версии приложения.

Так как серверная архитектура представлена 3-мя компонентами (frontend сервер, backend сервер и сервер базы данных), то требования к аппаратно-техническому обеспечению рассматривались отдельно для каждого из компонентов. Аппаратно-технические требования для данных компонентов отражены в таблице 13.

Таблица 13 – Требования к аппаратно-техническому обеспечению

Компонент	Центральный процессор	Оперативная память	Пространство на жестком диске
frontend сервер (NodeJs React)	1 ГГц	512 Мб	20 Мб
backend сервер (Java Spring Boot)	Pentium 2 266 МГц	128 Мб	124 МБ для JRE; 2 МБ для обновления Java
Сервер БД (MySql)	-	178,12 Мб	-

Требования к серверу базы данных четко не определены, но существует специальный сервис, с помощью которого было рассчитано приблизительное использование оперативной памяти. На рисунке 19 изображен процесс расчета.

Parameter	MySQL Default	Your Value
key_buffer_size	64 MB	64 MB
+ query_cache_size	64 MB	64 MB
+ tmp_table_size	32 MB	32 MB
+ innodb_buffer_pool_size	8 MB	8 MB
+ innodb_additional_mem_pool_size	1 MB	1 MB
+ innodb_log_buffer_size	1 MB	1 MB
+ max_connections	150	3
×		
sort_buffer_size	2 MB	2 MB
+ read_buffer_size	0.128 MB	0.128 MB
+ read_rnd_buffer_size	0.256 MB	0.256 MB
+ join_buffer_size	0.128 MB	0.128 MB
+ thread_stack	0.196 MB	0.196 MB
+ binlog_cache_size	0 MB	0 MB
Totals:	576,2 MB	178,12 MB

Рисунок 19 – Расчет используемой оперативной памяти для сервера БД

В параметрах по умолчанию было изменено количество максимальных подключений к базе данных с 150 до 3. Это возможно за счет того, что к базе будет только 1 подключение со стороны сервера. Остальные 2 подключения остаются для сервисного обслуживания базы данных.

К клиентской части не выставляются особые технические требования, так как разрабатываемое приложение реализуется как веб страница, благодаря чему пользователь может получить доступ к приложению из любой системы с установленным браузером и выходом в сеть.

2.5.2 Требования к программному обеспечению

Для пользования приложением достаточно наличия браузера, с поддержкой основных современных веб стандартов: HTML5, CSS, стандарт ES6 для javascript. Платформа в данном случае не имеет значения.

Программные требования для серверной части приложения приведены в таблице 14.

Таблица 14 – Программные требования серверной части приложения

Компонент	Платформа	Контейнеры для сервлетов	Инструменты сборки	Другое ПО
frontend сервер (NodeJs React)	Microsoft Windows, Linux x32/x64	-	-	-
backend сервер (Java Spring Boot)	Microsoft Windows, Linux x32/x64	Tomcat 9.0 или Jetty 9.4 или Undertow 2.0 или любой контейнер, совместимый с Servlet 3.1+.	Maven или Gradle	Java 8 или выше
Сервер БД (MySql)	Microsoft Windows, Linux x32/x64, macOS, FreeBSD и т. д.	-	-	-

На основании данных требований будет производиться выбор серверной машины и последующее разворачивания компонентов приложения.

3 Разработка информационного и программного обеспечения web-приложения

3.1 Разработка архитектуры web-приложения

Для разработки архитектуры приложения был выбран шаблон проектирования MVC, который позволяет разделить приложение на 3 части:

- model (модель) – предоставляет данные и реагирует на команды контроллера, изменяя свое состояние;
- view (вид или представление) – отвечает за отображение данных модели пользователю, реагируя на изменения модели;
- controller (контроллер) – интерпретирует действия пользователя, оповещая модель о необходимости изменений [19].

Основная цель применения этой концепции состоит в отделении бизнес-логики (модели) от ее визуализации (представления, вида). За счет такого разделения повышается возможность повторного использования кода. Таким образом клиентское приложение (визуализация) не привязано к серверу, что позволяет разрабатывать различные клиенты под разные платформы и задачи.

Схема архитектуры, используемой в приложении, изображена на рисунке 20.

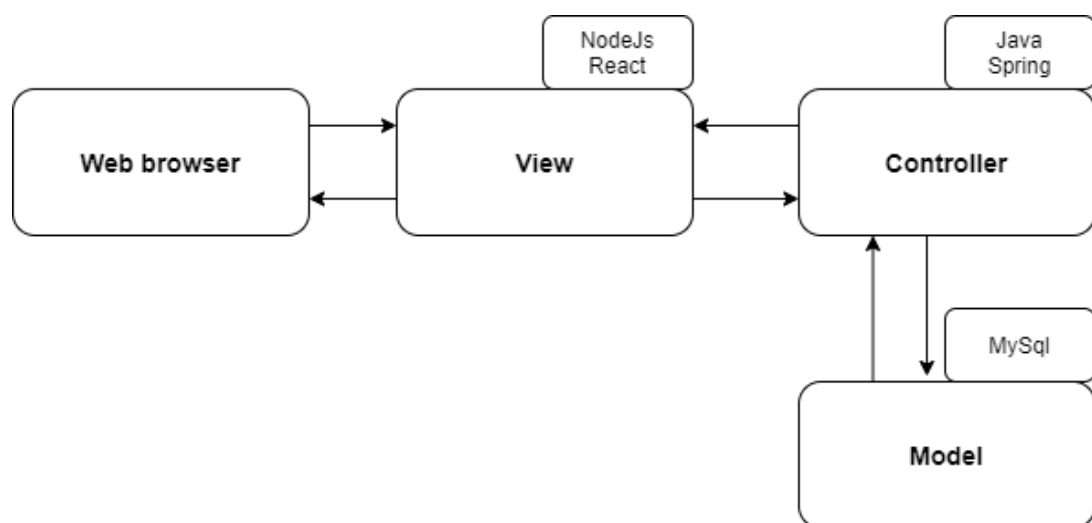


Рисунок 20 – Архитектура разрабатываемой системы

В данной схеме клиентом выступает веб браузер. Веб браузер запрашивает у представления необходимую страницу.

Роль представления реализовывает сервер Node JS с библиотекой React, используемая для разработки пользовательского интерфейса. Он формирует страницу и возвращает ее пользователю. При отрисовке некоторых страниц или при добавлении новых данных через форму, представление отправляет соответствующие запросы контроллеру в виде Rest API запроса.

Контроллером служит Rest API сервер, написанный на Java Spring. Он обрабатывает запросы от представления и возвращает ответы в формате JSON. Также с помощью контроллера осуществляется взаимодействие с моделью, реализуемой средствами сервера базы данных MySQL.

3.2 Разработка базы данных web-приложения

Одним из важных этапов разработки базы данных является проектирование. Основными целями проектирования являются:

- возможность хранения всех необходимых данных в БД;
- исключение избыточности данных;
- сведение числа хранимых в БД отношений к минимуму;
- нормализация отношений для упрощения решения проблем, связанных с обновлением и удалением данных.

Особое внимание хотелось бы уделить нормализации данных.

Нормализация – это разбиение отношения на два или более отношений, обладающих лучшими свойствами при включении, обновлении и удалении данных. Окончательная цель нормализации сводится к получению такой базы данных, в которой каждый факт появляется лишь в одном месте, т.е. исключена избыточность информации. Это делается не только с целью экономии памяти, но и для исключения возможной противоречивости хранимых данных.

Существует несколько правил нормализации базы данных. Каждое правило называется «нормальной формой». Если первое правило соблюдается,

база данных, как сообщается, находится в «первой нормальной форме». Если соблюдаются первые три правила, база данных рассматривается как «третья нормальная форма». Хотя возможны другие уровни нормализации, третья нормальная форма считается наивысшим уровнем, необходимым для большинства приложений.

Первая нормальная форма. Основным правилом первой формы является необходимость неделимости значения в каждом поле строки – атомарность значений.

Вторая нормальная форма. Записи не должны зависеть от чего-либо, кроме основного ключа таблицы (сложный ключ, если это необходимо).

Третья нормальная форма. Требование третьей нормальной формы (3NF) заключается в том, чтобы в таблицах отсутствовала транзитивная зависимость (зависимость не ключевых столбцов от других не ключевых столбцов);

Другие формы нормализации. Четвертая нормальная форма, также называемая «Обычная форма Бойс Кодд» (BCNF), и пятая нормальная форма существуют, но редко рассматриваются в практическом дизайне. Игнорирование этих правил может привести к менее совершенному дизайну базы данных, но не должно влиять на функциональные возможности [20].

Разработанная база данных является нормализованной и находится в 3 нормальной форме, т.к. соответствует вышеперечисленным требованиям.

База данных разрабатывалась с учетом поставленных задач таким образом, что операции по внесению и забору ресурсов со склада фиксируются отдельными записями в системе. Это позволяет отслеживать изменения количества ресурсов, поддерживать целостность и производит последующий анализ складских данных.

Скрипт создания базы данных расположен в приложении В. Модель разработанной базы данных отражен на рисунках 21, 22.

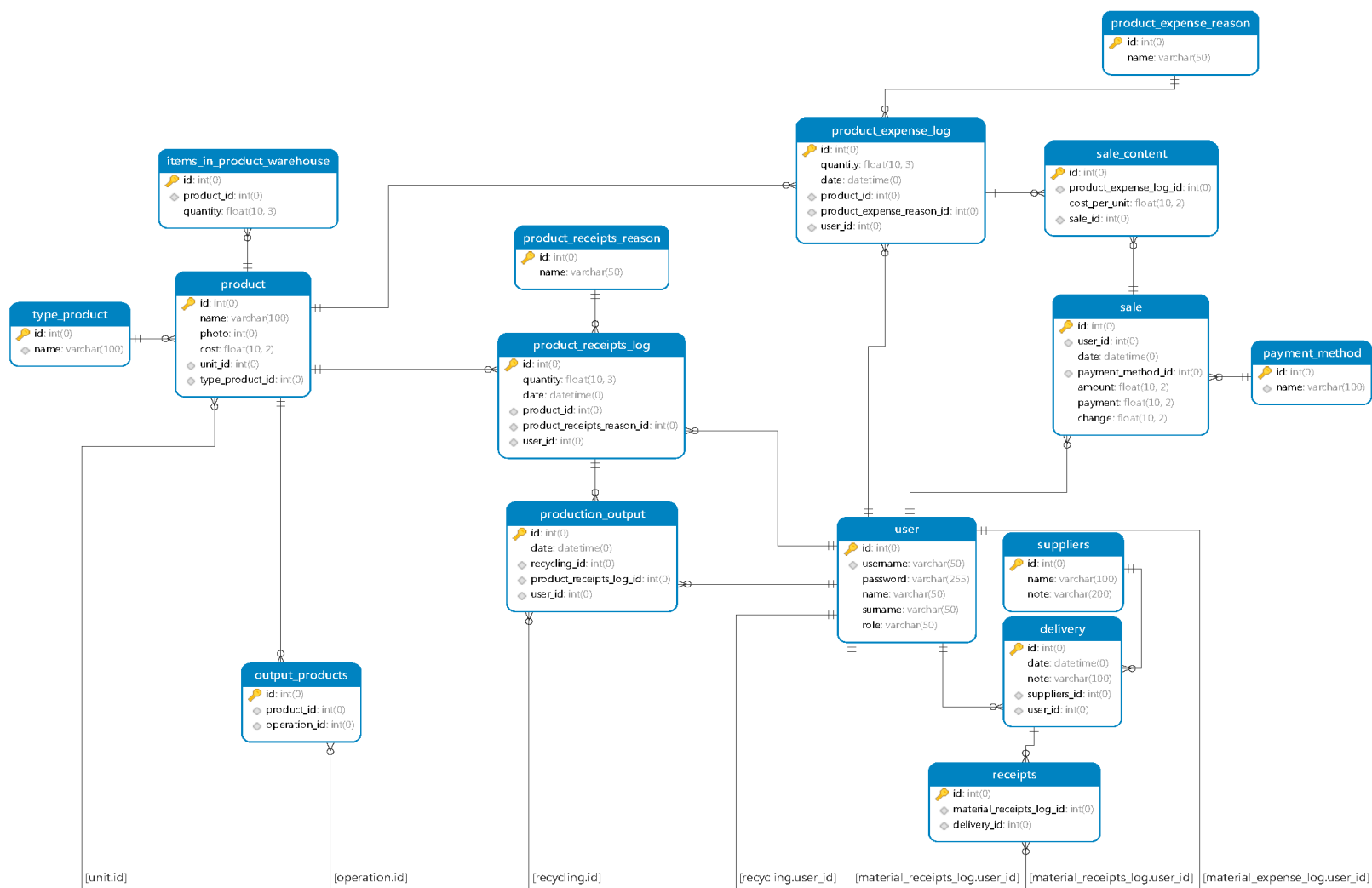


Рисунок 21 – Модель базы данных (часть 1)

Табличное описание сущностей базы данных представлено в таблице 15.

Таблица 15 – Описание сущностей базы данных

KEY	Поле	Тип	Дополнительно	Описание
delivery (поставка)				
P	id	int	Auto Increment	
	date	datetime		Дата поставки
	note	varchar(100)	Allow Null	Примечание
F	suppliers_id	int		Id поставщика
F	user_id	int		Id работника
fence_from_receipts (забор из поступления)				
P	id	int	Auto Increment	
F	material_receipts_log_id	int		Id лога поступления
F	recycling_id	int		Id переработки
	quantity	float(15,3)		Количество
fence_from_warehouse (забор со склада)				
P	id	int	Auto Increment	
	date	datetime		Дата забора
F	recycling_id	int		Id переработки
F	material_expense_log_id	int		Id лога расхода материалов
input_materials (входные материалы для операции)				
P	id	int	Auto Increment	
	quantity	float(10,3)		Количество
F	material_id	int		Id материала
F	operation_id	int		Id операции
items_in_materials_warehouse (позиции на складе ресурсов)				
P	id	int	Auto Increment	
	quantity	float(15,3)		Количество материала
F	material_id	int		Id материала
items_in_product_warehouse (позиции на складе продукции)				
P	id	int	Auto Increment	
F	product_id	int		Id продукта
	quantity	float(10,3)		Количество продукта
material (материал)				
P	id	int	Auto Increment	
	name	varchar(100)		Название материала
	photo	varchar(255)	Allow Null	Url фотографии материала
F	unit_id	int		Id единицы измерения
F	type_material_id	int		Id типа материала
material_expense_log (лог расхода материалов)				
P	id	int	Auto Increment	
	quantity	float(10,3)		Количество
	date	datetime		Дата
F	material_id	int		Id материала
F	material_expense_reason_id	int		Id причины расходов
F	user_id	int		Id пользователя

Продолжение таблицы 15

KEY	Поле	Тип	Дополнительно	Описание
material_expense_reason (причина расхода материалов)				
P	id	int	Auto Increment	
	name	varchar(50)		Название причины
material_receipts_log (лог поступления материалов)				
P	id	int	Auto Increment	
	quantity	float(10,3)		Количество
	quantity_remains	float(10,3)		Оставшееся количество
	date	datetime		Дата поступления
F	material_id	int		Id материала
F	material_receipts_reason_id	int		Id причины поступления
F	user_id	int		Id пользователя
material_receipts_reason (причина поступления материалов)				
P	id	int	Auto Increment	
	name	varchar(50)		Наименование причины
operation (операции по переработке)				
P	id	int	Auto Increment	
	name	varchar(100)		Наименование операции
	estimated_number	float(10,3)		Предполагаемое кол-во выпуска
F	type_operation_id	int		Id типа операции
output_materials (выходные материалы)				
P	id	int	Auto Increment	
F	material_id	int		Id материала
F	operation_id	int		Id операции
output_products (выходная продукция)				
P	id	int	Auto Increment	
F	product_id	int		Id продукта
F	operation_id	int		Id операции
payment_method (метод оплаты)				
P	id	int	Auto Increment	
	name	varchar(100)		Наименование метода оплаты
product (продукция)				
P	id	int	Auto Increment	
	name	varchar(100)		Наименование продукта
	photo	varchar(255)	Allow Null	Url фото продукта
	Cost	float(10,2)		Стоимость продукта за единицу
F	unit_id	int		Id единицы измерения
F	type_product_id	int		Id типы продукта
product_expense_log (лог расхода продуктов)				
P	Id	int	Auto Increment	
	Quantity	float(10,3)		Количество
	Date	datetime		Дата расхода
F	product_id	int		Id продукта
F	product_expense_reason_id	int		Id причины расхода
F	user_id	int		Id пользователя

Продолжение таблицы 15

KEY	Поле	Тип	Дополнительно	Описание
product_expense_reason (причины расхода продуктов)				
P	Id	int	Auto Increment	
	Name	varchar(50)		Название причины расхода
product_receipts_log (лог поступления продукции)				
P	Id	int	Auto Increment	
	Quantity	float(10,3)		Количество продукции
	Date	datetime		Дата поступления
F	product_id	int		Id продукта
F	product_receipts_reason_id	int		Id причины поступления продукта
F	user_id	int		Id работника
product_receipts_reason (причина поступления продукта)				
P	Id	int	Auto Increment	
	Name	varchar(50)		Название причины поступления
production_output (выпуск продукции из переработки)				
P	Id	int	Auto Increment	
	Date	datetime		Дата выпуска
F	recycling_id	int		Id переработки
F	product_receipts_log_id	int		Id лога поступления продуктов
F	user_id	int		Id работника
receipt_to_warehouse (поступление на склад)				
P	Id	int	Auto Increment	
	Date	datetime		Дата поступления
F	recycling_id	int		Id переработки
F	material_receipts_log_id	int		Id лога поступления материалов
receipts (поступление)				
P	Id	int	Auto Increment	
F	material_receipts_log_id	int		Id лога поступления материалов
F	delivery_id	int		Id поставки
recycling (переработка)				
P	Id	int	Auto Increment	
	start_date	datetime		Дата начала переработки
	end_date	datetime	Allow Null	Дата окончания переработки
	Status	int		Статус переработки
F	operation_id	int		Id операции
F	user_id	int		Id работника
sale (продажа)				
P	Id	int	Auto Increment	
F	user_id	int		Id продавца
	Date	datetime		Дата продажи
F	payment_method_id	int		Id способа оплаты
	Amount	float(10,2)		Сумма к оплате
	Payment	float(10,2)		Внесенная сумма
	Change	float(10,2)		Сдача

Окончание таблицы 15

KEY	Поле	Тип	Дополнительно	Описание
sale content (содержимое продажи)				
P	Id	int	Auto Increment	
F	product_expense_log_id	int		Id лог расхода продукции
	cost_per_unit	float(10,2)		Цена за единицу продукта
F	sale_id	int		Id продажи
suppliers (поставщик)				
P	Id	int	Auto Increment	
	Name	varchar(100)		Наименование поставщика
	Note	varchar(200)	Allow Null	Дополнительная информация
type_material (тип материала)				
P	Id	int	Auto Increment	
	Name	varchar(100)		Название типа материала
type_operation (тип операции)				
P	Id	int	Auto Increment	
	Name	varchar(50)		Название типа операции
type_product (тип продукта)				
P	Id	int	Auto Increment	
	Name	varchar(100)		Название типа продукции
unit (единица измерения)				
P	Id	int	Auto Increment	
	Name	varchar(100)		Название единицы измерения
user (пользователь)				
P	Id	int	Auto Increment	
	username	varchar(50)		Никнейм пользователя
	password	varchar(255)		Пароль
	name	varchar(50)		Имя пользователя
	Surname	varchar(50)		Фамилия пользователя
	Role	varchar(50)		Роль в системе

Для всех сущностей, первичными ключами являются искусственно созданные индексы, который присваивается автоматически автоинкрементированием. Данная мера позволяет обеспечить простоту добавления новых данных, не уделяя внимания при этом первичному ключу.

На основании данной модели, с помощью IDE, будут построены соответствующие классы на серверной части приложения, связанные с сущностями БД.

3.3 Разработка программного обеспечения серверного приложения

Обобщенная структура сервера состоит из 4-х компонентов, отображенных в таблице 16.

Таблица 16 – Описание компонентов серверного приложения

Название компонента	Описание
Модель данных	Описывает сущности предметной области в рамках языка программирования. Поля модели и сама модель помечены специальными аннотациями, для связывания их с сущностями базы данных (пример описание модели представлен в приложении Г.1)
Репозиторий	Служит для взаимодействия приложения с базой данных. Здесь описываются все запросы, работающие с моделями, описанными ранее
Сервис	Содержит основную логику работы приложения. Сервисы осуществляют обработку информации, полученной от контроллеров
Request контроллер	Обрабатывает внешние запросы пользователей. Контроллеры не содержат в себе какой-либо особой логики. Они передают полученную информацию сервисам, на что те формирует соответствующий ответ

Модели данных были построены автоматически, на основе ранее разработанной схемы базы данных. Данная операция была выполнена с помощью дополнительной библиотеки Jakarta Persistence для среды разработки IntelliJ IDEA. Библиотека позволяет задать основные параметры импорта моделей, задать путь импорта и шаблон для их наименования (рисунок 23).

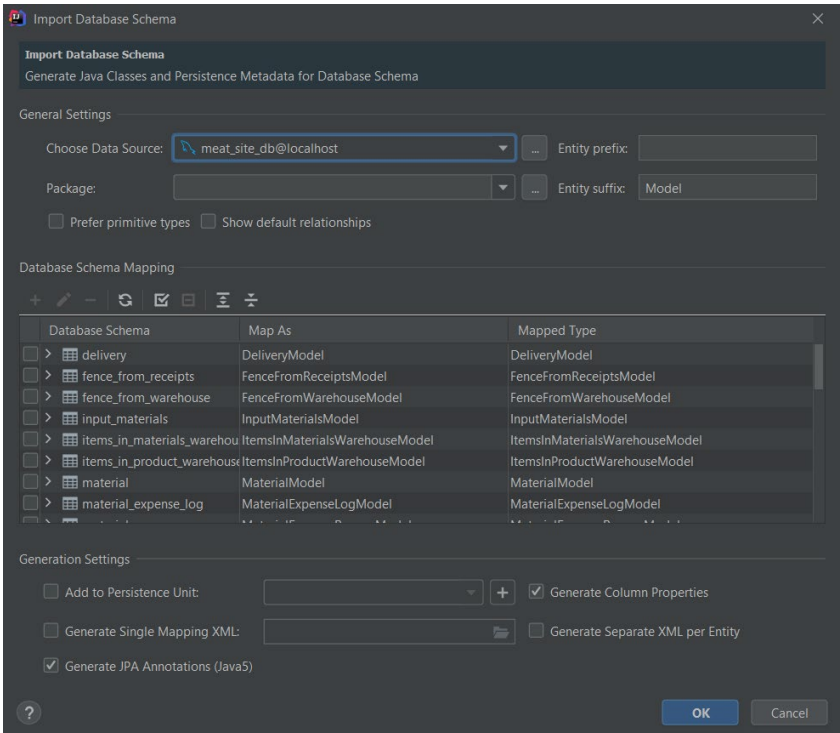


Рисунок 23 – Окно импорта схемы базы данных

Для упрощения процесса разработки и последующей доработки, программа была разделена на отдельные модули по их функциональности. Данные модули были описаны в таблице 18.

Таблица 17 – Описание модулей сервера

Название модуля	Описание	Включаемые файлы	Описание файлов
auth	Обрабатывает запросы авторизации и регистрации пользователей.	AuthApi	Rest контроллер, обрабатывающий запросы на авторизацию и регистрацию
		AuthService	Сервис, осуществляющий проверку данных и непосредственную регистрацию / авторизацию
		LoginResponse	Объект, возвращаемый пользователю в ответ на запрос об авторизации. Содержит информацию о статусе авторизации.
		LoginStatus	Enum объект, содержащий в себе все возможные статусы авторизации

Продолжение таблицы 17

Название модуля	Описание	Включаемые файлы	Описание файлов
		RegistrationStatus	Enum объект, содержащий в себе возможные статусы регистрации
		Role	Enum объект, содержащий в себе возможные роли пользователей в системе
materialWarehouse	Операции взаимодействия со складом ресурсов	MaterialWarehouseApi	Rest контроллер, обрабатывающий запросы, связанные со складом материалов
		MaterialWarehouseService	Сервис, осуществляющий взаимодействие со складом ресурсов
		ReceiptElementModel	Модель, принимаемая на вход контроллером. Содержит материал и его количество
productWarehouse	Операции взаимодействия со складом продукции	ProductWarehouseService	Сервис, осуществляющий взаимодействие со складом продукции
deliveries	Управление поставками	DeliveriesApi	Rest контроллер, обрабатывающий запросы, связанные с поставками
		DeliveriesService	Сервис, отвечающий за управление поставками
sale	Управление продажами	SaleApi	Rest контроллер, обрабатывающий запросы, связанные с осуществлением продаж
		SaleService	Сервис, отвечающий за проведение новых продаж в системе
admin	Администрирование системы и сбор статистики по накопленным данным	AdminApi	Rest контроллер, обрабатывающий запросы на получение статистики, а также на управление системой
		AdminService	Сервис, отвечающий за сбор статистики и управление общими параметрами системы

Одним из основных модулей системы является MaterialWarehouseService. С помощью него осуществляются все процессы, связанные с добавлением, получением или изменением данных на складе материалов (код сервиса представлен в приложении Г.2). Процесс добавления материалов состоит из нескольких этапов, показанных на диаграмме последовательности (рисунок 24).

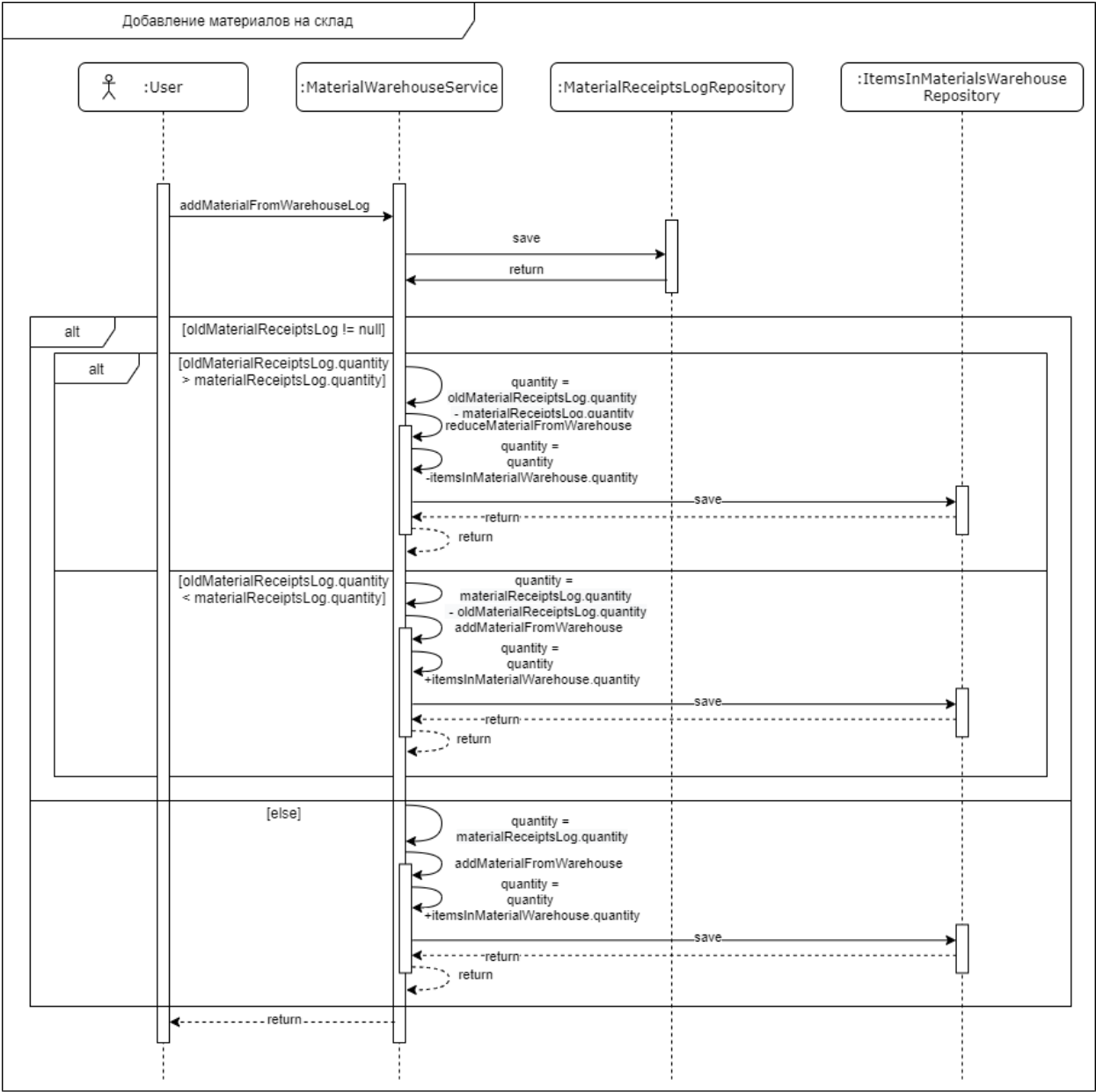


Рисунок 24 – Диаграмма последовательности операции добавления материала на склад

Как видно из диаграммы – добавление материалов на склад происходит в 2 этапа: сначала добавляется или модифицируется лог о добавлении материала в `MaterialReceiptsLogRepository`, а затем уже непосредственное корректирование количества материалов на складе (`ItemsInMaterialsWarehouseRepository`). Операции проходят в рамках одной транзакции, что позволяет избежать ситуации, когда лог о поставке был добавлен, а количество материала не скорректировалось. При добавлении нового лога, материал на складе складывается с пришедшим. Если же происходит изменение лога, то работа программы может пойти по 2-м путям:

- измененное количество материалов больше предыдущего;
- измененное количество материалов меньше предыдущего.

В ситуации, когда количество материалов больше предыдущего, система высчитывает разность между новым и старым значениями и отправляет запрос на добавление этой разности на склад материалов. В противном случае, система высчитывает разность между старым и новым значениями и отправляет запрос на вычитание этой разности на склад материалов.

Сервер разработан таким образом, чтобы облегчить этап последующего усовершенствования системы, а принцип Rest API позволяет обращаться к нему практически из любого приложения. Это позволит в будущем выпустить мобильную версию приложения, при этом минимально затрагивая серверную часть.

3.4 Разработка клиентского приложения

3.4.1 Разработка интерфейса клиентского приложения

При заходе на страницу web-приложения, пользователю отображается форма входа в систему. На форме расположены основные компоненты, необходимые для авторизации пользователя: поле ввода логина, поле ввода пароля, кнопка «Войти» (рисунок 25).

Вход

 andrey_krasnomyasov



☒ Запомнить меня

Войти

Рисунок 25 – Форма входа в систему

При нажатии на кнопку «Войти» происходит отправка Rest запроса на сервер с данными, введенными в форму. Если пользователь ввел неверные данные, то на странице отобразится соответствующее сообщение (рисунок 26).

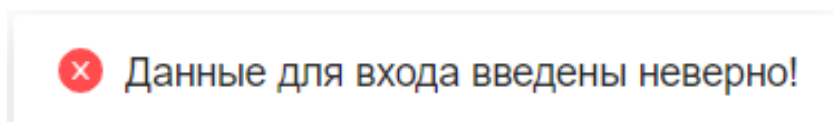


Рисунок 26 – Предупреждающее окно

Если сервер прислал положительный ответ на авторизацию, то пользователь перенаправляется на главную страницу web-приложения. На главной странице отображается общая информация о системе на базовом шаблоне web-приложения (рисунок 27).

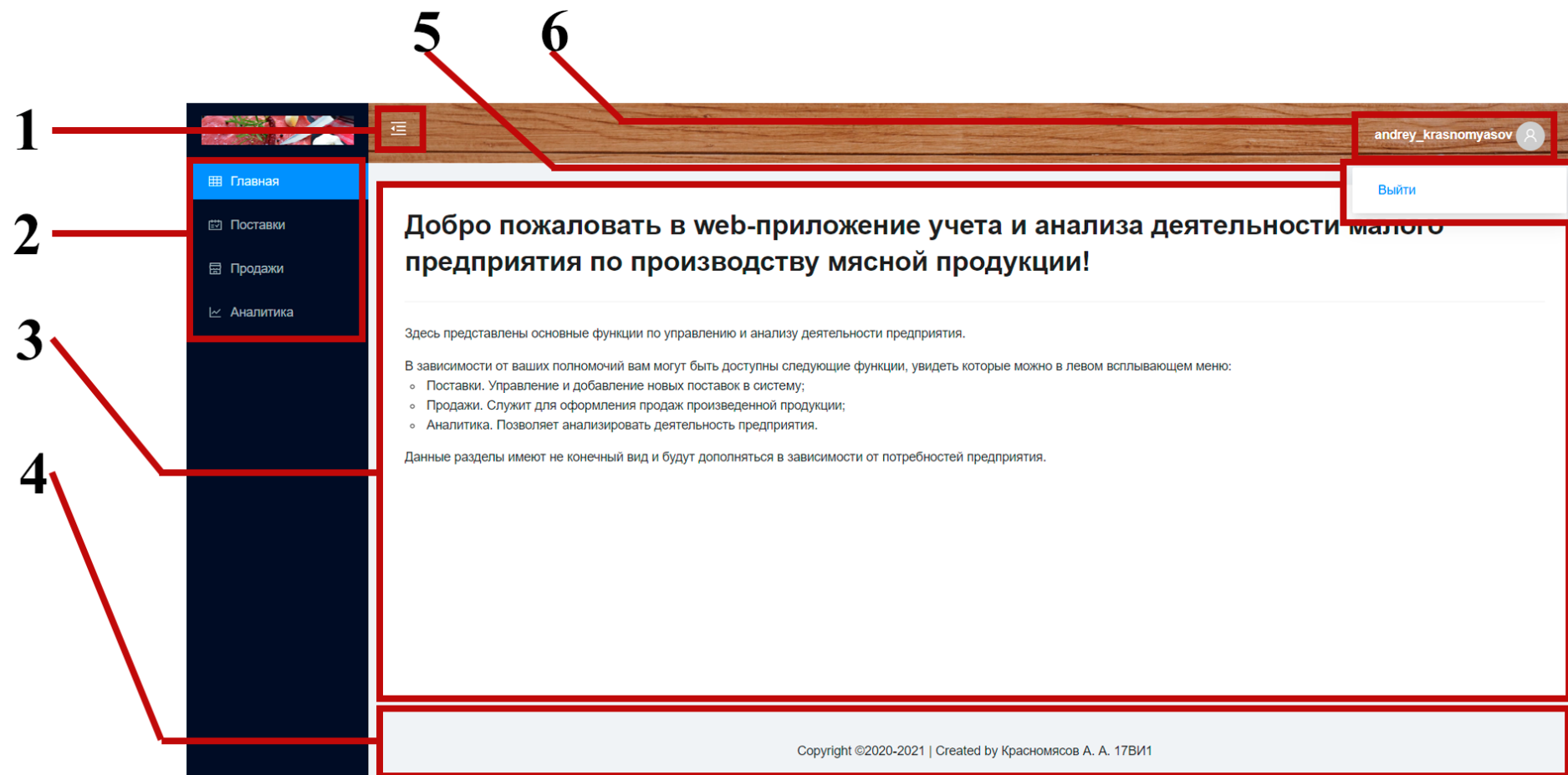


Рисунок 27 – Главная страница приложения

На рисунке, представленном выше, цифрами отмечены основные элементы шаблона сайта, на основании которого будут строиться все остальные страницы. На данной схеме выделены следующие элементы:

- 1 – кнопка сворачивания / разворачивания левого навигационного меню;
- 2 – навигационное меню сайта, с помощью которого происходит переход по страницам приложения;
- 3 – основной контент страницы (на главной странице отображается общая информация о системе и о принципах ее работы);
- 4 – footer или «подвал» сайта, в котором располагается справочная информация о разработчике сайта;
- 5 – всплывающее меню вошедшего пользователя (с помощью него, пользователь может выйти из системы, выбрав соответствующий пункт);
- 6 – информация о пользователе.

Данная структура позволяет наиболее эффективно использовать рабочую область страницы сайта. Почти всю область занимает полезный контент, а сворачивающееся меню позволяет расширить данную область.

Следующей страницей в системе является страница «Поставки» (рисунок 28).

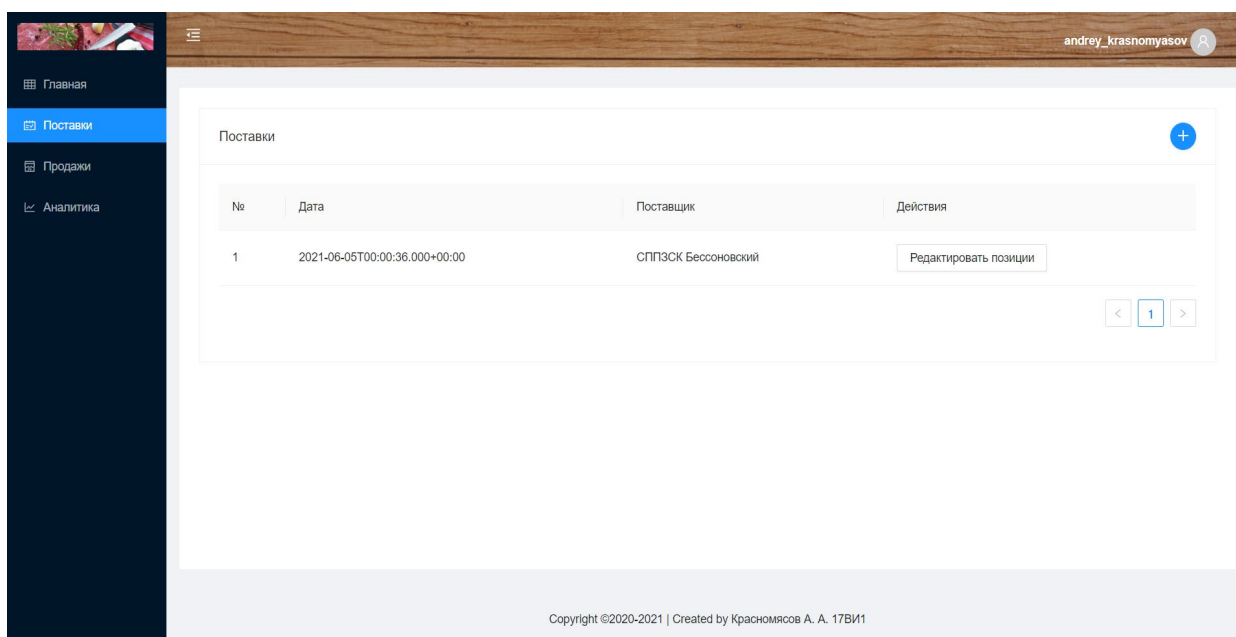


Рисунок 28 – Страница поставок

На данной странице отображается список последних поставок. Основной функцией данной страницы является добавление поставок и редактирование в них позиций. Добавление происходит с помощью всплывающих форм, изображенных на рисунках 29 - 30 соответственно.

Добавление поставки ×

* Поставщик:

ТК «Мираторг» ▼

* Дата поставки:

Дата поставки 📅

Отмена

Добавить

Рисунок 29 – Форма добавления поставки

Редактирование позиций ×

Окорок свиной ▼

0.00000

Выберите материал ▼

0.00000

+ Добавить позицию

Отмена

Добавить

Рисунок 30 – Форма добавления позиций в поставку

Следующим рассматриваемым модулем является модуль «Продажи». С помощью него осуществляется реализация произведенной продукции. На странице отображается список продукции на складе, отсортированной по убыванию количества. Данная сортировка необходима для того, чтобы продавец понимал, какой товар находится в избытке и нуждается в приоритетной реализации (рисунок 31).

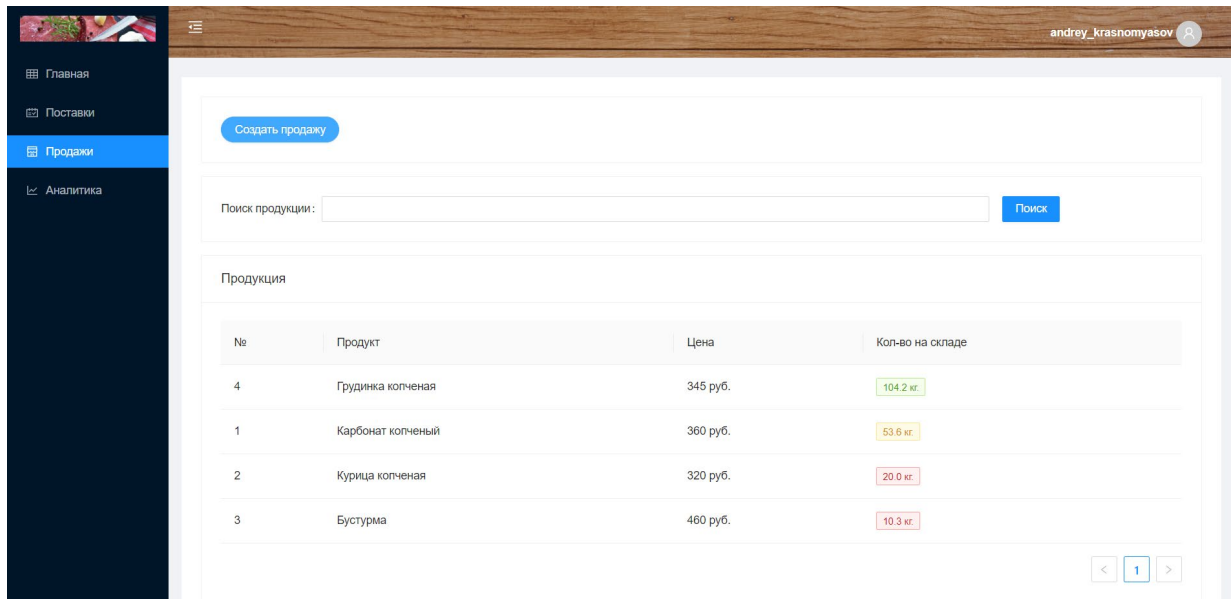


Рисунок 31 – Страница продаж

При нажатии кнопки «Создать продажу», продавцу открывается форма, в которой формируется список продаваемых продуктов и высчитывается итоговая стоимость (рисунок 32).

Создание продажи # X

Карбонат копченый 3.00000 кг. Стоимость: 360 Сумма: 1080 ⊖

+ Добавить позицию

Итого: 1080 руб.

Отмена
Создать

Рисунок 32 – Форма создания продажи

Последним модулем системы является модуль аналитики. На данной странице администратор может проанализировать различные параметры реализации продукции. Для удобства работы страница разделена на 2 части с помощью вкладок (рисунок 33).

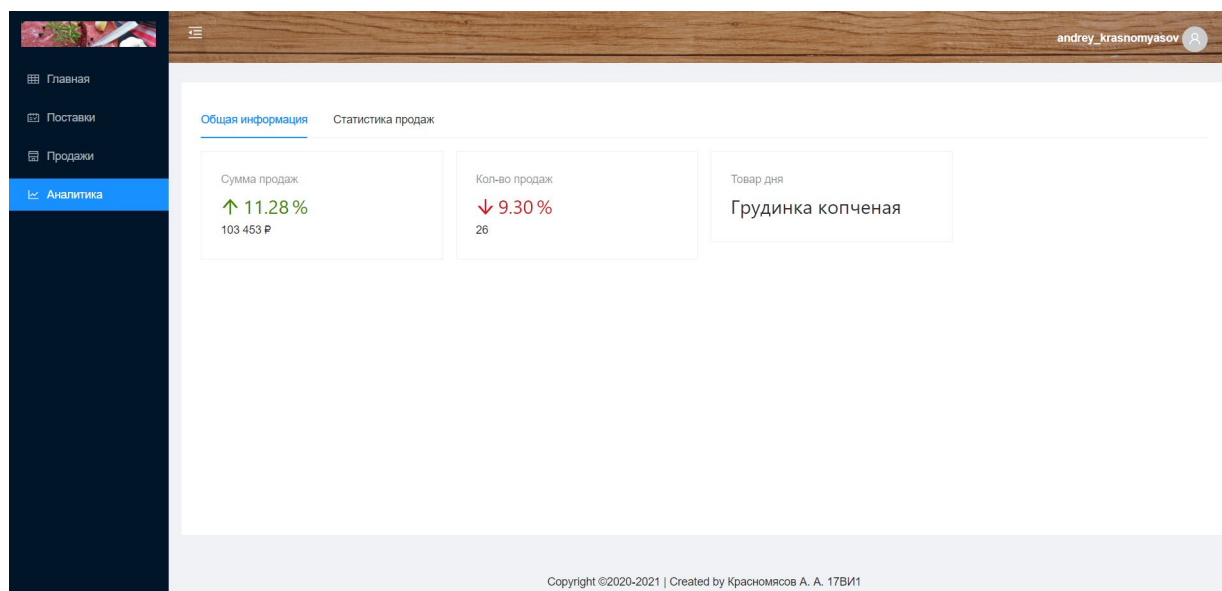


Рисунок 33 – Страница анализа

На первой вкладке «Общая информация», расположены виджеты с общей информацией по продажам. Она включает в себя:

- сумма продаж (в сравнении с предыдущим днем);
- количество продаж (в сравнении с предыдущим днем);
- товар дня (товар, который имеет наибольшее количество продаж за сегодняшний день).

Как отмечалось ранее, все данные на данной странице представлены отдельными виджетами, что позволяет подстраивать выводимые параметры под конкретные нужды предприятия. Так как эта страница открывается сразу же после открытия страницы аналитики, то здесь сосредоточены наиболее общие, но тем не менее важные параметры, на изменение которых стоит обратить внимание управляющему персоналу.

На второй вкладке располагается график, показывающий зависимость количество продаж от даты. График является интерактивным, что дает возможность производить анализ на определенном промежутке времени (рисунок 34).

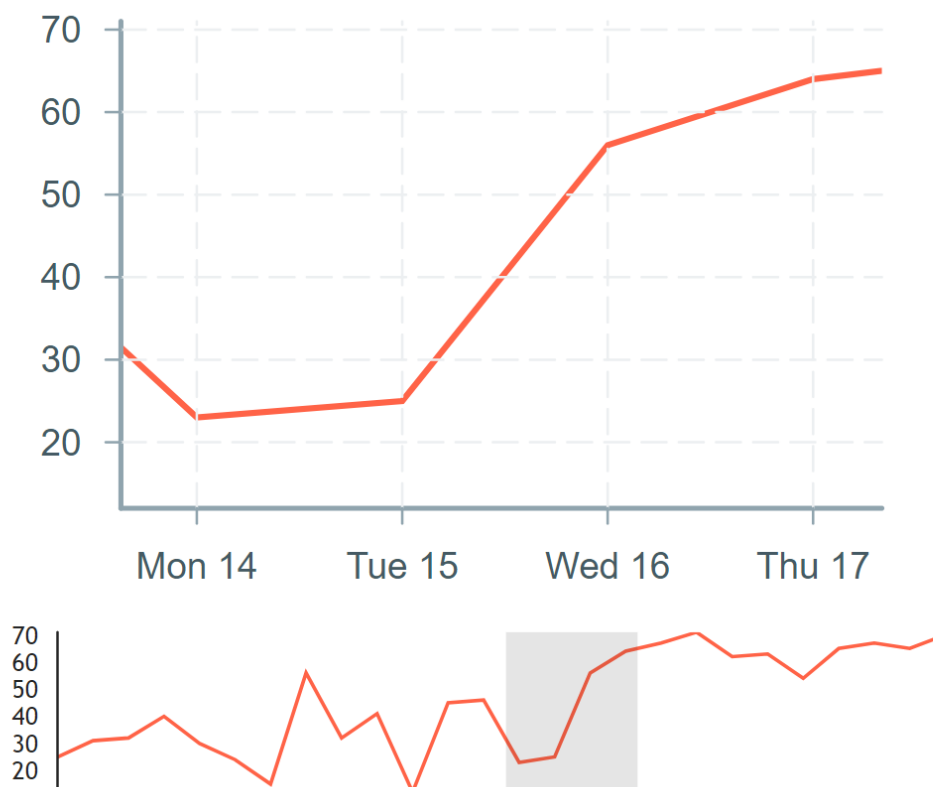


Рисунок 34 – Графики показателей продаж

С помощью графического представления данных, администратор может в удобной форме определить необходимые ему параметры и основываясь на них в короткие сроки принять необходимые решения о деятельности предприятия.

Список всех разработанных страниц в системе представлен в таблице 18.

Таблица 18 – Список страниц web-приложения

Название	Адрес	Назначение
Страница входа	/login	Страница служит для входа пользователя в систему
Главная страница web-приложения	/admin/main-page	Главная страница, описывающая основной функционал приложения
Страница управления поставками	/admin/delivery-page	Страница, позволяющая проводить новые и редактировать осуществленные поставки
Страница управления продажами	/admin/sale-page	Страница, позволяющая осуществлять продажи товаров на складе
Страница анализа деятельности предприятия	/admin/analytic-page	Страница аналитики, отображающая пользователю результат деятельности предприятия

Таким образом, при разработке интерфейса системы были учтены поставленные задачи: разработать систему с интуитивно понятным интерфейсом и с необходимым функционалом. Благодаря тому, что все страницы располагаются на одном макете и открываются с помощью вкладок меню, можно легко дорабатывать систему, добавляя новые страницы без затрагивания старых.

3.4.1 Разработка программного обеспечения клиентского приложения

Разработанный интерфейс предполагает расположение компонентов на страницах по определенному макету. То есть, при создании страницы, необходимо менять лишь полезную нагрузку, оставляя при этом меню, header и footer неизменными. Для данной цели был разработан отдельный модуль AdminLayout, на основании которого и строятся практически все страницы. На рисунке 35 изображена диаграмма компонентов, показывающая общую структуру страниц web-приложения.

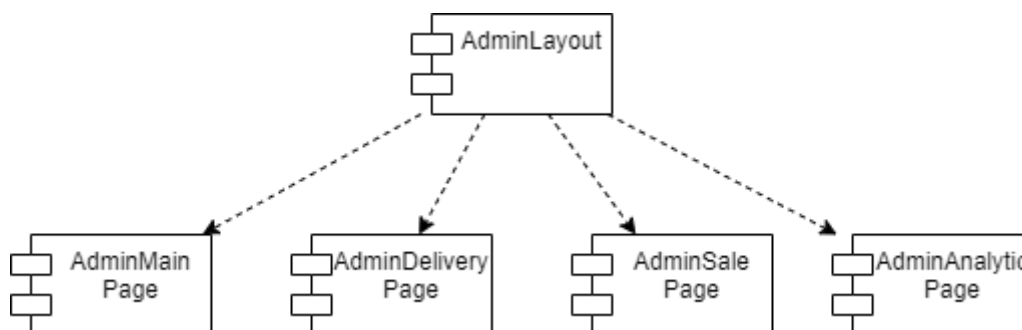


Рисунок 35 – Диаграмма компонентов для страниц web-приложения

Диаграмма показывает, что основные страницы приложения, а именно AdminMainPage, AdminDeliveryPage, AdminSalePage и AdminAnalyticPage строятся на основании ранее рассмотренного шаблона AdminLayout. Это позволяет избежать дублирования кода, а также позволяет добавлять и видоизменять элементы меню, не затрагивая при этом остальные страницы. Код данного шаблона представлен в приложении Д.

4 Контроль качества программного обеспечения

4.1 Планирование процесса контроля качества

Тестирование программного обеспечения – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта [21].

Тестирование является одним из наиболее устоявшихся способов обеспечения качества разработки программного обеспечения.

С технической точки зрения тестирование заключается в выполнении приложения на некотором множестве исходных данных и сверке получаемых результатов с заранее известными (эталонными) с целью установить соответствие различных свойств и характеристик приложения заказанным свойствам. Как одна из основных фаз процесса разработки программного продукта (Дизайн приложения – Разработка кода – Тестирование), тестирование характеризуется достаточно большим вкладом в суммарную трудоемкость разработки продукта. Широко известна оценка распределения трудоемкости между фазами создания программного продукта: 40% – 20% – 40%.

Для оценки качества тестов используют различные метрики, связанные с количеством найденных дефектов, покрытием кода, функциональных требований, множества сценариев [22].

4.1.1 Мероприятия по контролю качества

Вид тестирования – это совокупность активностей, направленных на тестирование заданных характеристик системы или ее части, основанная на конкретных целях.

Существует большое количество признаков классификации видов тестирования. Наиболее распространенная схема классификации видов тестирования отображена на рисунке 36.

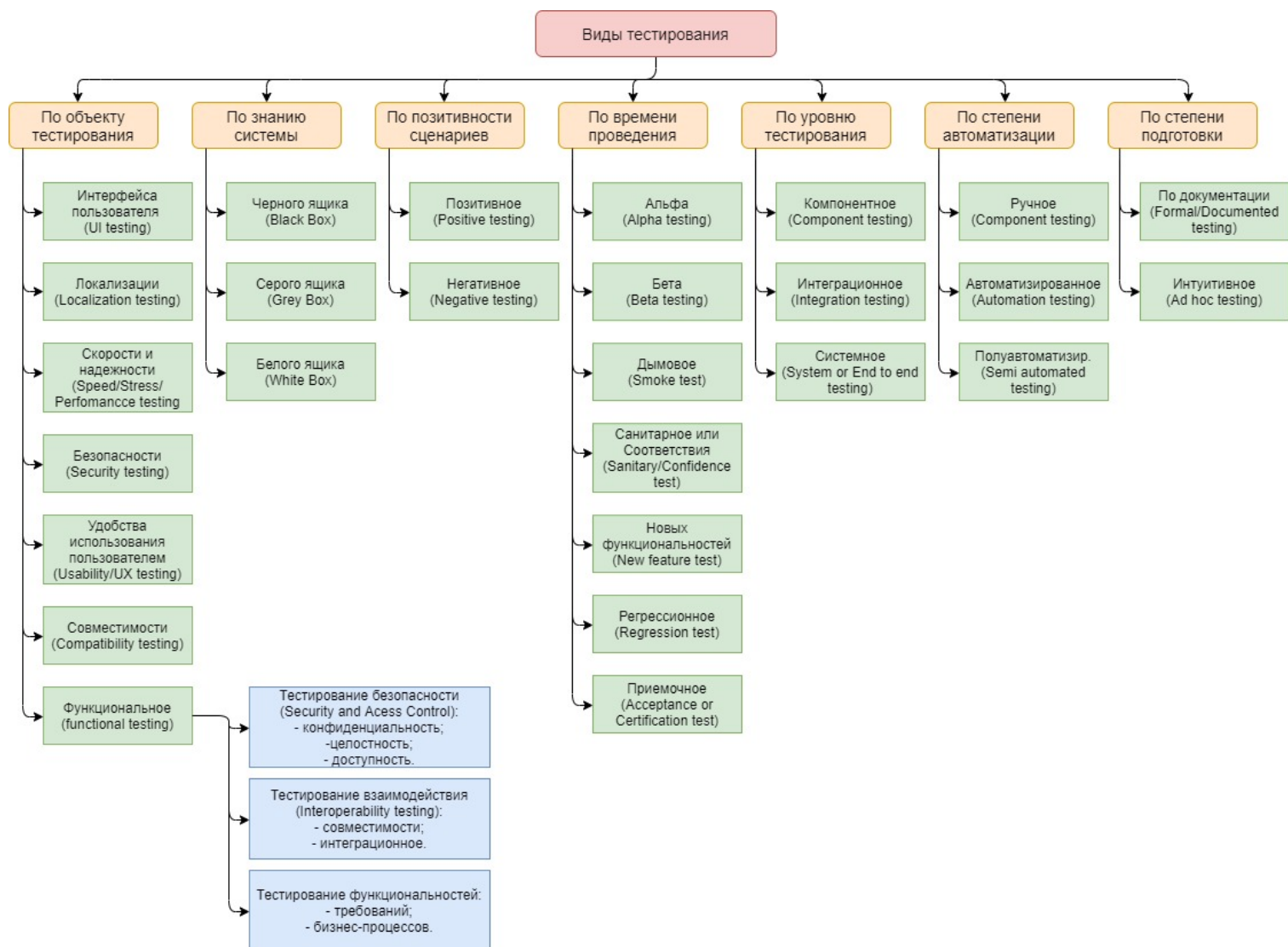


Рисунок 36 – Виды тестирования

Из данной схемы видно, что существует большое количество видов тестирования. Выбор необходимых методологий тестирования для конкретного приложения зависит от большого числа аспектов, таких как тип проекта, требования заказчика, поставленные сроки, а также многих других факторов.

4.1.2 Обоснование выбора модели тестирования

Для проведения тестирования системы были выбраны следующие методологии:

- функциональное тестирование;
- модульное тестирование.

Функциональное тестирование направлено на проверку корректности работы функциональности приложения. Тестирование было проведено с помощью программы «Postman». С ее помощью можно отправлять REST запросы, тестируя тем самым отдельные функции приложения. Данный подход позволяет имитировать штатную работу приложения, посылая отдельные REST запросы на сервер приложения;

При модульном тестировании проверяются отдельные небольшие части приложения, например, отдельный класс, функция или небольшие библиотеки. Тесты написаны на языке Java с использованием библиотеки JUnit. Модульное тестирование позволяет в автоматическом режиме проводить проверки правильности работы отдельных модулей системы. Проверки возможно запускать после каждой сборки приложения, что позволяет отследить ошибки еще до запуска сервера.

Postman – популярное программное средство для тестирования работы различных API или просто отправки POST и GET запросов на сервер. Оно имеет графический интерфейс и поэтому намного удобнее консольной утилиты curl. С помощью него можно просто настраивать передаваемые Cookie, HTTP заголовки, а также параметры форм. Все это позволяет удобно проверить работу любого API, и даже обычного веб-сайта.

4.2 Тестирование программного обеспечения

4.2.1 Функциональное тестирование

Для тестирования была выбрана часть системы, для добавления новых поставок. Функциональная диаграмма для данной группы процессов представлена на рисунке 37.

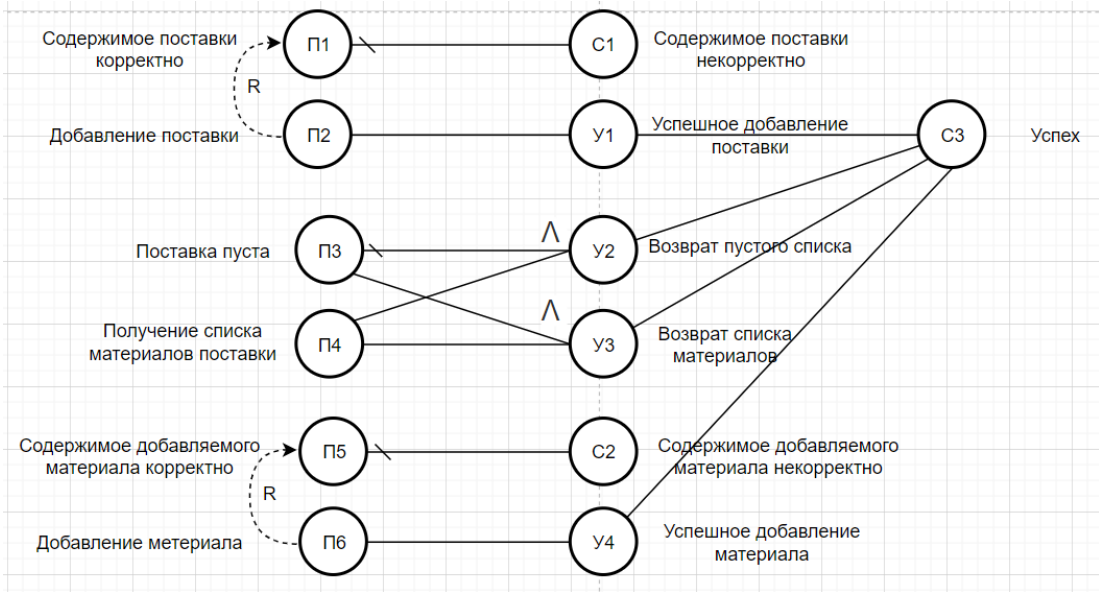


Рисунок 37 – Функциональная диаграмма

На основании функциональной диаграммы были произведены соответствующие тесты. Тестирование проводилось на predetermined input data. In table 19 is a list of all conducted tests.

Таблица 19 – Функциональные тесты

№	Описание	Входные данные	Ожидаемый результат	Полученный результат
1	Добавление в систему корректной поставки	"timestamp": 1623698625, "note": "test", "suppliersId": -1	Успешное добавление данных	Успешное добавление данных (рисунок 38)
2	Получение списка материалов только что созданной поставки	"deliveryId": 3	Пустой массив	Пустой массив (рисунок 39)

Продолжение таблицы 19

№	Описание	Входные данные	Ожидаемый результат	Полученный результат
3	Добавление материала в поставку	"deliveryId": -1, "quantity": 100.0, "quantityRemains": 100.0, "date": "2021-04-03 10:30:00", "materialsId": -1, "receiptsReasonId": -1, "userId": 1	Успешное добавление материала	Успешное добавление материала (рисунок 40)
4	Повторное получение списка материалов созданной поставки	"deliveryId": 31	Массив с добавленным материалом	Массив с добавленным материалом (рисунок 41)

POST

▼

http://localhost:8080/delivery/addDelivery

Params

Authorization

Headers (9)

Body ●

Pre-request Script

Tests

Settings

● none

● form-data

● x-www-form-urlencoded

● raw

● binary

● GraphQL

	KEY	VALUE
☒	timestamp	1623698625
☒	note	test
☒	suppliersId	1
	Key	Value

Body

Cookies

Headers (8)

Test Results

Pretty

Raw

Preview

Visualize

JSON ▼

↻

```

1  {
2    "code": 0,
3    "message": "Успешно!"
4  }

```

Рисунок 38 – Добавление поставки

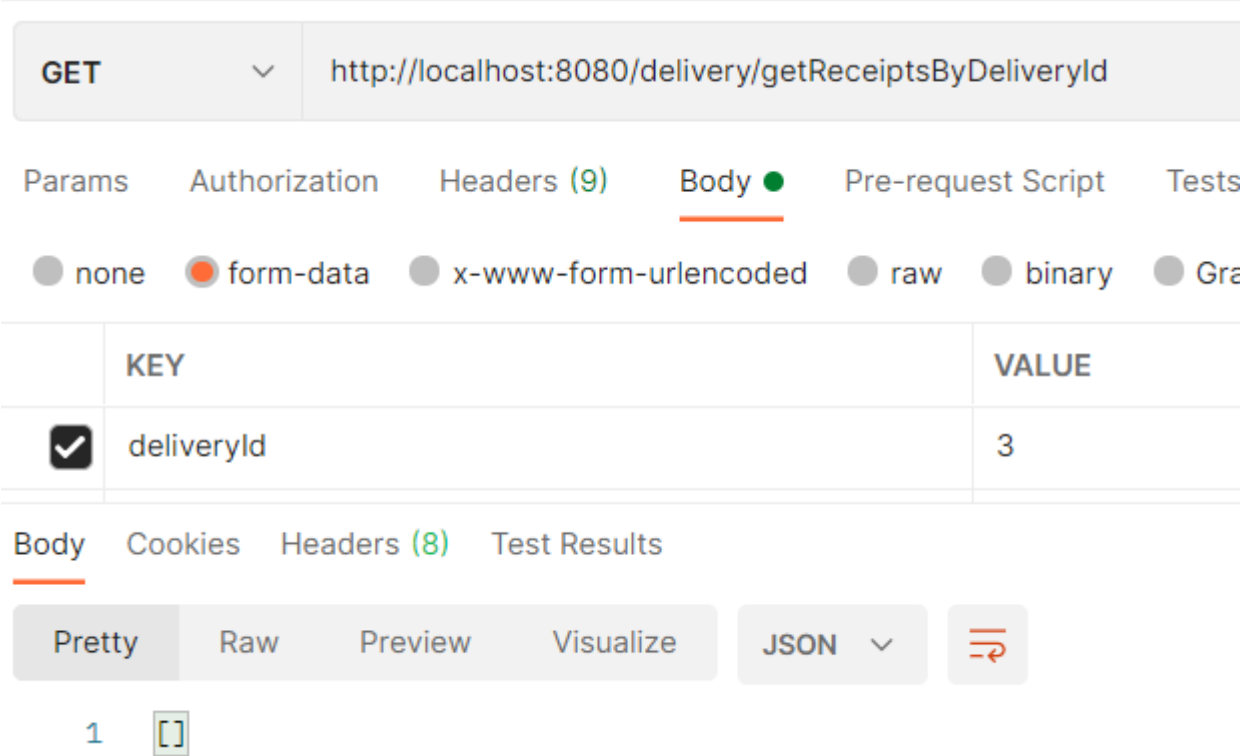


Рисунок 39 – Проверка содержимого поставки

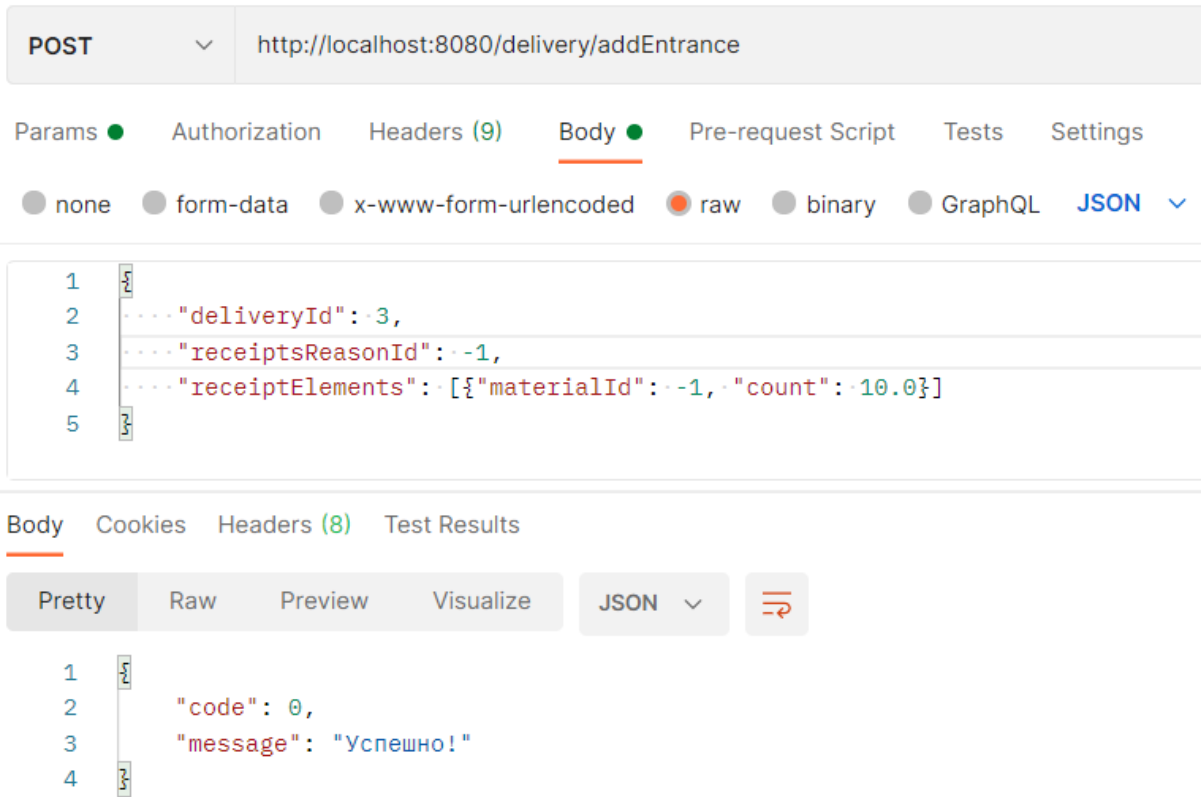


Рисунок 40 – Добавление материала в поставку

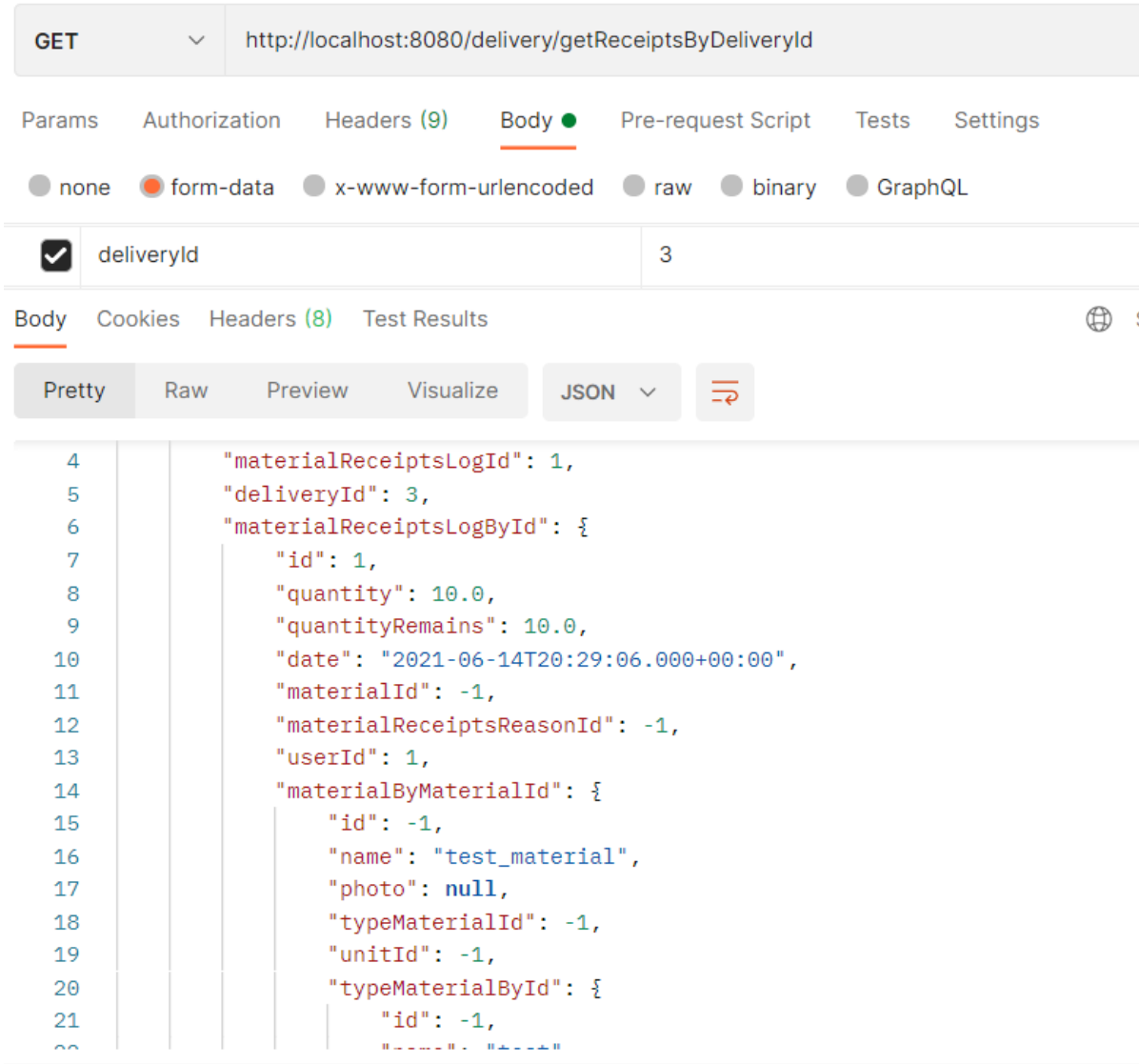


Рисунок 41 – Повторная проверка содержимого поставки

Все функциональные тесты были пройдены успешно. Полученные результаты совпали с ожидаемыми.

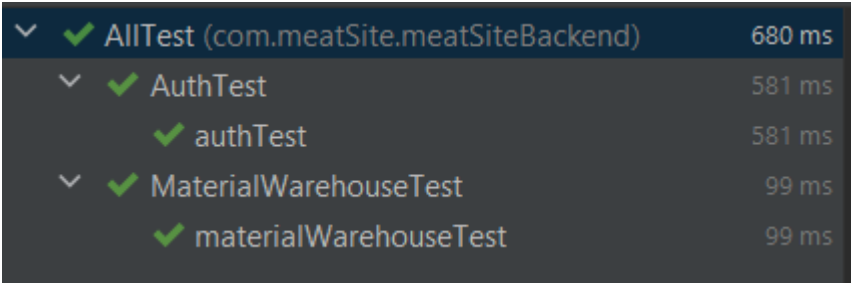
4.2.2 Модульное тестирование

Для модульного тестирования сервера, написанного на Java, была использована библиотека JUnit. Unit-тесты были написаны для 2-х сервисов: AuthService, MaterialWarehouseService. Для данных сервисов было разработано по 1 тесту, содержащих в себе ряд операций и проверок. Список Unit-тестов представлен в таблице 20.

Таблица 20 – Описание Unit тестов

Название теста	Описание	Входные данные	Вложенные операции	Ожидаемый результат	Полученный результат
authTest	Тест авторизации. Здесь производится регистрация пользователя, попытка авторизации, удаление пользователя и повторная авторизация	TEST_USR = new UserModel(-1, "test_user", "test_password", "name", "surname");	Регистрация пользователя	-	-
			Авторизация пользователя	LoginStatus. SUCCESSFULLY	LoginStatus. SUCCESSFULLY
			Удаление пользователя	-	-
			Повторная авторизация	LoginStatus. INVALID_LOGIN_DATA	LoginStatus. INVALID_LOGIN_DATA
materialWarehouseTest	Тест правильности работы склада материалов. В ходе теста происходит добавление тестового ресурса на склад и проверка правильности итогового количества	TEST_RESOURCE = new ItemsInMaterials WarehouseModel(123.5, -1);	Добавление тестового материала на склад	-	-
			Проверка на то, что количество материалов увеличилось на заданное кол-во	oldQuantity + TEST_RESOURCE.quantity	oldQuantity + TEST_RESOURCE.quantity
			Удаление тестового материала со склада	-	-
			Проверка на то, что количество материалов вернулось к прошлым значениям	oldQuantity	oldQuantity

Результаты тестирования представлены на рисунке 42.



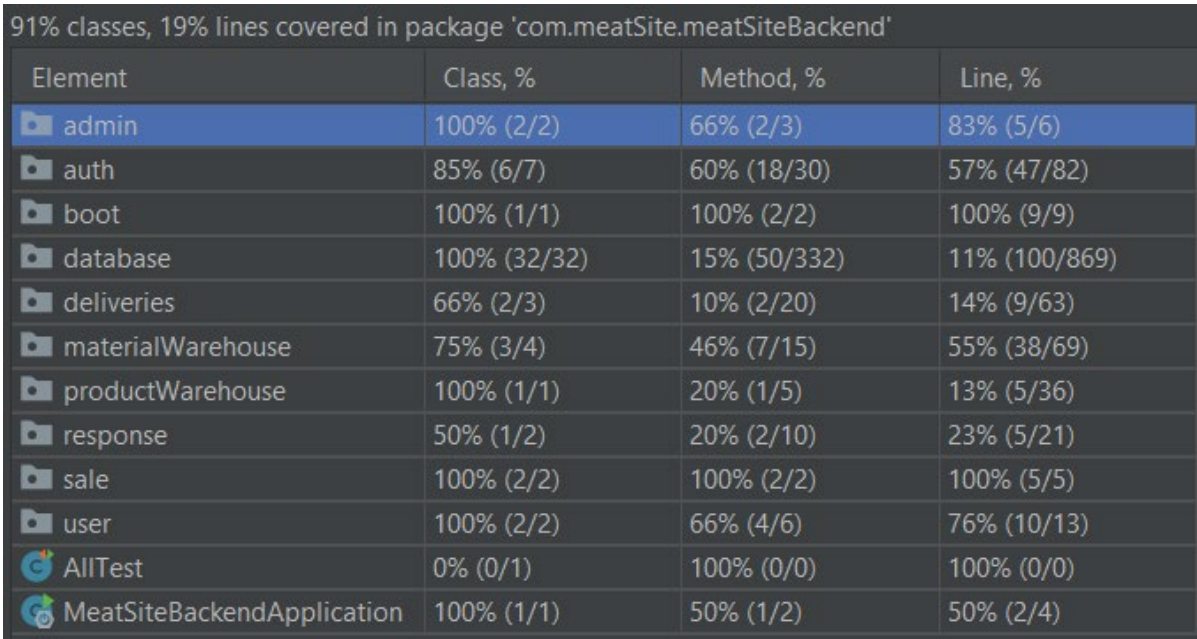
✓ AllTest (com.meatSite.meatSiteBackend)	680 ms
✓ AuthTest	581 ms
✓ authTest	581 ms
✓ MaterialWarehouseTest	99 ms
✓ materialWarehouseTest	99 ms

Рисунок 42 – Результаты прохождения тестов

Листинг Unit-тестов представлен в приложении Е.

В ходе проведения тестов ошибок обнаружено небыло. Все тестируемые модули работают корректно.

Также для анализа эффективности тестов была построена таблица покрытия классов (рисунок 43).



91% classes, 19% lines covered in package 'com.meatSite.meatSiteBackend'			
Element	Class, %	Method, %	Line, %
admin	100% (2/2)	66% (2/3)	83% (5/6)
auth	85% (6/7)	60% (18/30)	57% (47/82)
boot	100% (1/1)	100% (2/2)	100% (9/9)
database	100% (32/32)	15% (50/332)	11% (100/869)
deliveries	66% (2/3)	10% (2/20)	14% (9/63)
materialWarehouse	75% (3/4)	46% (7/15)	55% (38/69)
productWarehouse	100% (1/1)	20% (1/5)	13% (5/36)
response	50% (1/2)	20% (2/10)	23% (5/21)
sale	100% (2/2)	100% (2/2)	100% (5/5)
user	100% (2/2)	66% (4/6)	76% (10/13)
AllTest	0% (0/1)	100% (0/0)	100% (0/0)
MeatSiteBackendApplication	100% (1/1)	50% (1/2)	50% (2/4)

Рисунок 43 – Таблица покрытия тестов

Из данной таблицы видно, что тесты покрывают 91 % всех классов сервера, что является хорошим показателем.

Заключение

В процессе выполнения выпускной квалификационной работы был выполнен анализ предметной области, в рамках которого было рассмотрена актуальность создания web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции, также был проведен обзор существующих программных средств, специализирующихся на управлении мясоперерабатывающими предприятиями и складским учетом, была поставлена задача на разработку и определен бюджет разработки. В рамках проектирования приложения, была проведена разработка требований к разрабатываемой системе. Разработка требований включала в себя определение требований к архитектуре, технологиям разработки и используемым программным средствам.

В результате был разработан прототип web-приложения учета и анализа деятельности малого предприятия по производству мясной продукции.

Список источников

1. Рынок мясных продуктов и колбасных изделий. Итоги 2020 / Лаборатория – 3DPROINFO. URL: <https://3dpro.info/site/reviews/market-of-meat-products-and-sausages-2020/> (дата обращения: 2.04.2021).
2. Гурьянов Л. Выпускная квалификационная работа: стандарты и программные технологии / Л.В. Гурьянов, В.Н. Князев, И.Ю. Балашова, Е.А. Дзюба, Н.А. Попова, Д.В. Такташкин, С.В. Самуйлов – Пенза: Пензенский государственный университет, 2020. – 154 с.
3. Автоматизация мясоперерабатывающей промышленности / 1С – Рарус: 1С автоматизация бухгалтерского, управленческого, налогового учета по Москве и по всей России. URL: <https://rarus.ru/1c-branches/avtomatizatsiya-pischevoy-promishlennosti/avtomatizatsiya-myasopererabativayuschey-promishlennosti/#tab-description> (дата обращений: 2.04.2021).
4. Хаммер М. Реинжиниринг корпорации. Манифест революции в бизнесе / М. Хаммер, Д. Чампи. - Манн, Иванов и Фербер, 2006. – 321 с.
5. Порядок санитарно-микробиологического контроля при производстве мяса и мясных продуктов – Утвержден И.о. Директора Департамента пищевой и перерабатывающей промышленности Минсельхозпрода России В.Н.Сергеев 15 декабря 1995 г.
6. Главная страница / МультиМит Эксперт – программное обеспечение для предприятий мясной и рыбной промышленности. URL: <http://www.multimeat.ru/> (дата обращения 10.04.2021).
7. 1С:Мясопереработка MES. Модуль для 1С:ERP – Возможности / Отраслевые и специализированные решения 1С:Предприятие. URL: <https://solutions.1c.ru/catalog/meat-mes-erp/features> (дата обращения: 10.04.2021).
8. Главная страница / Программа для учета Меркурий-ERP. URL: <https://soft4trade.ru/> (дата обращения 10.04.2021).

9. Щербаков В. Основы финансового функционально-стоимостного анализа / Щербаков В., Приходько Е. – Новосибирский государственный технический университет, 2003. – 164 с.
10. Маклаков С. Моделирование бизнес-процессов с BPWIN 4.0 / С. Маклаков - Диалог-МИФИ, 2002. – 233 с.
11. Цуканова А. Методология и инструментарий моделирования бизнеспроцессов: учебное пособие / А. Цуканова – СПб.: Университет ИТМО, 2015. – 101 с.
12. Балашова И. Язык моделирования информационных процессов UML 2.0 / И.Ю. Балашова, Е.Н. Прошкина – Пензенский государственный университет, 2018. – 121 с.
13. Аверкин А. Толковый словарь по искусственному интеллекту / А.Н. Аверкин, М.Г. Гаазе-Рапопорт, Д.А. Поспелов – Москва: Радио и связь 1992. – 256 с.
14. Гольцман В. MySQL 5.0. Библиотека программиста / В. Гольцман – Питер, Санкт-Петербург, 2010. – 253 с.
15. Шилдт Г. Java 8. Полное руководство 9-е издание / Г. Шилдт - Издательский дом "Вильямс", 2015. – 1376с.
16. IntelliJ IDEA: функциональная и эргономичная IDE для разработки на Java от JetBrains / Официальный сайт компании JetBrains. URL: <https://www.jetbrains.com/ru-ru/idea/> (дата обращения: 29.05.2021).
17. WebStorm – умная среда разработки для JavaScript, созданная в JetBrains / Официальный сайт компании JetBrains. URL: <https://www.jetbrains.com/ru-ru/webstorm/> (дата обращения: 29.05.2021)
18. Уоллс К. Spring в действии. / У. Крейг – ДМК Пресс, 2013. – 752 с.
19. Рогачев С. Обобщенный Model-View-Controller Каркас на основе шаблона проектирования MVC в исполнении Generic Java и C# / С. Рогачев – Филиал ООО «Лукойл-Информ» в г. Пермь, 2007. – 35 с.

20. Описание основ нормализации базы данных | Microsoft Docs / Microsoft – официальная страница. URL: <https://docs.microsoft.com/ru-ru/office/troubleshoot/access/database-normalization-description> (дата обращения: 28.05.2021).

21. Куликов С. Тестирование программного обеспечения. Базовый курс. / С. С. Куликов – Минск: Четыре четверти, 2020. – 100 с.

22. Современные технологии тестирования / Национальный Открытый Университет «ИНТУИТ». URL: https://intuit.ru/studies/professional_retraining/955/courses/285/lecture/7170 (дата обращения: 28.05.2021).

Приложение А
ГЛОССАРИЙ
(справочное)

Web-приложение – клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети.

Диаграмма декомпозиции – диаграмма, получаемая в ходе разбиения контекстной диаграммы на подсистемы. Описывает каждую из этих подсистем и их взаимодействие.

Контекстная диаграмма – вид IDEF0-диаграммы. Это диаграмма, расположенная на вершине древовидной структуры диаграмм, представляющая собой самое общее описание системы и ее взаимодействие с внешней средой.

Оперативный учет – одно из четырех направлений хозяйственного учета наряду с бухгалтерским, статистическим и налоговым. Применяется в целях текущего управления в качестве системы наблюдения и контроля над отдельными фактами хозяйственной жизни организации, измеряемыми как по стоимостной оценке, так и в натуральных величинах.

Предметная область – множество всех предметов, свойства которых и отношения между которыми рассматриваются в научной теории.

Структура предприятия – внутреннее строение предприятия, которое характеризуется составом подразделений, системой их взаимодействий, взаимозависимости, подчинения и управления.

Сырье – предмет труда, претерпевший незначительное воздействие человека и предназначенный для дальнейшей обработки. Сырье является в первую очередь продуктом добывающей промышленности и сельского хозяйства.

Приложение Б
ПЕРЕЧЕНЬ НОРМАТИВНОЙ ДОККУМЕНТАЦИИ
(рекомендуемое)

Категория	Наименование	Описание
Государственный стандарт Российской Федерации (ГОСТ)	ГОСТ 18300-87	Спирт этиловый ректификованный, технический. Технические условия
	ГОСТ 21237-75	Мясо. Методы бактериологического анализа
	ГОСТ 9792-73	Колбасные изделия и продукты из свинины, баранины, говядины и мяса других видов убойных животных и птиц. Правила приемки и методы отбора проб
	ГОСТ 9958-81	Изделия колбасные и продукты из мяса. Методы бактериологического анализа
	ГОСТ 4288-76	Изделия кулинарные и полуфабрикаты из рубленого мяса. Правила приемки и методы испытаний
	ГОСТ 8756.0-70	Продукты пищевые консервированные. Отбор проб и подготовка их к испытанию
	ГОСТ 8756.18-70	Продукты пищевые консервированные. Метод определения внешнего вида, герметичности тары и состояния внутренней поверхности металлической тары

Продолжение таблицы Б

Категория	Наименование	Описание
	ГОСТ 26668-85	Продукты пищевые и вкусовые. Методы отбора проб для микробиологических анализов
	ГОСТ 26669-85	Подготовка проб для микробиологического анализа
	ГОСТ 26670-91	Продукты пищевые. Методы культивирования микроорганизмов
	ГОСТ 10444.3-85	Консервы. Метод определения мезофильных аэробных и факультативно-анаэробных микроорганизмов
	ГОСТ 10444.4-85	Консервы. Метод определения мезофильных анаэробных микроорганизмов
	ГОСТ 10444.5-85	Консервы. Метод определения термофильных аэробных и факультативно-анаэробных микроорганизмов
	ГОСТ 10444.6-85	Консервы. Метод определения термофильных анаэробных микроорганизмов
	ГОСТ 10444.2-75	Консервы. Методы микробиологического анализа. Выявление коагулазоположительных стафилококков
	ГОСТ 10444.8-88	Метод определения <i>Bacillus cereus</i>

Продолжение таблицы Б

Категория	Наименование	Описание
	ГОСТ 10444.9-88	Метод определения <i>Clostridium perfringens</i>
	ГОСТ 10444.7-86	Продукты пищевые. Методы выявления ботулинических токсинов и <i>Clostridium botulinum</i>
	ГОСТ 26968-86	Сахар-песок рафинированный. Методы микробиологического анализа
	ГОСТ 18963-73	Методы санитарно-бактериологического анализа
	ГОСТ 2874-82	Вода питьевая. Гигиенические требования и контроль за качеством
	ГОСТ 9225-84	Молоко и молочные продукты. Методы микробиологического анализа
	ГОСТ 23670-72	Колбасы вареные, сосиски и сардельки, хлеба мясные
Общесоюзный стандарт (ОСТ)	ОСТ 49 190-82	Колбасы ливерные. Технические условия
Технические условия (ТУ)	ТУ 10.10.01.03-89	Оболочка искусственная белковая "Белкозин"
	ТУ 10.02.01.174-93	Кровь пищевая и продукты ее переработки
Медико-биологические требования (МБТ)	МБТ N 5061-89	Медико-биологические требования и санитарные нормы качества продовольственного сырья

Продолжение таблицы Б

Категория	Наименование	Описание
Санитарные правила и нормы (СП)	СП 1.2.011-94	Безопасность работы с микроорганизмами I и II групп патогенности
Прочее	Правила работы с заразным или подозреваемым в заражении материалом в бактериологических лабораториях (бактериологических отделах лабораторий) предприятий мясной промышленности, 1974.	
	Инструкция о порядке санитарно-технического контроля консервов на производственных предприятиях, оптовых базах, в розничной торговле и на предприятиях общественного питания. М., 1992 г., N 01-19/9-11.	
	Инструкция о порядке микробиологического контроля производства мясных пастеризованных консервов. М., 1984 г.	
	Санитарно-гигиенические требования по производству мясных консервов для питания детей раннего возраста. М., 1983 г.	

Продолжение таблицы Б

Категория	Наименование	Описание
	Временные санитарно-гигиенические требования к производству продуктов из мяса в полимерной упаковке с длительным сроком хранения, 1990.	
	Ветеринарно-санитарные правила промышленной переработки импортного мяса и мясопродуктов на мясоперерабатывающих предприятиях России, 1994.	
	Правила ветеринарного осмотра убойных животных и ветеринарно-санитарной экспертизы мяса и мясных продуктов, М., 1983.	
	Лабораторная диагностика сальмонеллезов человека и животных, обнаружение сальмонелл в кормах, продуктах питания и объектах внешней среды. Методические указания, М., 1990.	

Окончание таблицы Б

Категория	Наименование	Описание
	Инструкция по микробиологическому контролю производства быстрозамороженных готовых мясных блюд, 1981.	
	Технические требования к соевым белкам, закупаемым по импорту, для производства вареных, полукопченых колбас и полуфабрикатов. Утв. 1986.	
	"Методические указания по санитарно-бактериологическому контролю на предприятиях общественного питания и торговли пищевыми продуктами", М., 1984.	
	Инструкция по мойке и профилактической дезинфекции на предприятиях мясной и птицеперерабатывающей промышленности, М., 1985.	
	Внутриведомственные санитарные требования к холодильникам мясной и молочной промышленности, 1986.	

Приложение В
СКРИПТ СОЗДАНИЯ БАЗЫ ДАННЫХ
(рекомендуемое)

```

# ПОЛЬЗОВАТЕЛИ
CREATE TABLE `user`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `username` varchar(50) NOT NULL,
  `password` varchar(255) NOT NULL,
  `name` varchar(50) NOT NULL,
  `surname` varchar(50) NOT NULL,
  `role` varchar(50) NOT NULL DEFAULT "ROLE_GUEST",
  PRIMARY KEY (`id`),
  UNIQUE INDEX (`username`)
) ENGINE=InnoDB;

# ВИД СЫРЬЯ
CREATE TABLE `type_material`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX (`name`)
) ENGINE=InnoDB;

# ЕДИНИЦЫ ИЗМЕРЕНИЯ
CREATE TABLE `unit`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX (`name`)
) ENGINE=InnoDB;

# СЫРЬЕ
CREATE TABLE `material`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `photo` varchar(255) NULL,
  `type_material_id` int NOT NULL,
  `unit_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (type_material_id) REFERENCES type_material (id),
  FOREIGN KEY (unit_id) REFERENCES unit (id)
) ENGINE=InnoDB;

# ПОСТАВЩИКИ
CREATE TABLE `suppliers`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `note` varchar(200),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB;

# ПОСТАВКИ
CREATE TABLE `delivery`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `date` datetime NOT NULL,
  `note` varchar(100),
  `suppliers_id` int NOT NULL,
  `user_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (suppliers_id) REFERENCES suppliers (id),
  FOREIGN KEY (user_id) REFERENCES user (id)
)

```

```

) ENGINE=InnoDB;

# ПРИЧИНЫ ПОСТУПЛЕНИЙ МАТЕРИАЛОВ
CREATE TABLE `material_receipts_reason`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB;

# ПРИЧИНЫ РАСХОДОВ МАТЕРИАЛОВ
CREATE TABLE `material_expense_reason`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB;

# ЛОГ ПОСТУПЛЕНИЙ МАТЕРИАЛОВ
CREATE TABLE `material_receipts_log`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `quantity` float(10,3) NOT NULL,
  `quantity_remains` float(10,3) NOT NULL,
  `date` datetime NOT NULL,
  `material_id` int NOT NULL,
  `material_receipts_reason_id` int NOT NULL,
  `user_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_id) REFERENCES material (id),
  FOREIGN KEY (material_receipts_reason_id) REFERENCES
material_receipts_reason(id),
  FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

# ЛОГ РАСХОДОВ МАТЕРИАЛОВ
CREATE TABLE `material_expense_log`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `quantity` float(10,3) NOT NULL,
  `date` datetime NOT NULL,
  `material_id` int NOT NULL,
  `material_expense_reason_id` int NOT NULL,
  `user_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_id) REFERENCES material (id),
  FOREIGN KEY (material_expense_reason_id) REFERENCES
material_expense_reason(id),
  FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

# ПОСТУПЛЕНИЯ
CREATE TABLE `receipts`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `material_receipts_log_id` int NOT NULL,
  `delivery_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_receipts_log_id) REFERENCES material_receipts_log
(id),
  FOREIGN KEY (delivery_id) REFERENCES delivery (id)
) ENGINE=InnoDB;

# СКЛАД ПРОИЗВОДСТВЕННЫЙ

```

```

CREATE TABLE `items_in_materials_warehouse`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `quantity` float(15,3) NOT NULL,
  `material_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_id) REFERENCES material (id)
) ENGINE=InnoDB;

# ТИПЫ ОПЕРАЦИЙ
CREATE TABLE `type_operation`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX (`name`)
) ENGINE=InnoDB;

# ОПЕРАЦИИ
CREATE TABLE `operation`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `estimated_number` float(10,3) NOT NULL DEFAULT 0,
  `type_operation_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (type_operation_id) REFERENCES type_operation(id)
) ENGINE=InnoDB;

# ПЕРЕРАБОТКИ
CREATE TABLE `recycling`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `start_date` datetime NOT NULL,
  `end_date` datetime NULL,
  `status` int NOT NULL,
  `operation_id` int NOT NULL,
  `user_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (operation_id) REFERENCES operation(id),
  FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

# ЗАБОР СО СКЛАДА
CREATE TABLE `fence_from_warehouse`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `date` datetime NOT NULL,
  `recycling_id` int NOT NULL,
  `material_expense_log_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (recycling_id) REFERENCES recycling (id),
  FOREIGN KEY (material_expense_log_id) REFERENCES material_expense_log (id)
) ENGINE=InnoDB;

# ЗАБОР ИЗ ПОСТУПЛЕНИЙ
CREATE TABLE `fence_from_receipts`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `material_receipts_log_id` int NOT NULL,
  `recycling_id` int NOT NULL,
  `quantity` float(15,3) NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (recycling_id) REFERENCES recycling (id),

```

```
FOREIGN KEY (material_receipts_log_id) REFERENCES material_receipts_log (id)
) ENGINE=InnoDB;
```

```
# ПОСТУПЛЕНИЕ НА СКЛАД
```

```
CREATE TABLE `receipt_to_warehouse`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `date` datetime NOT NULL,
  `recycling_id` int NOT NULL,
  `material_receipts_log_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (recycling_id) REFERENCES recycling (id),
  FOREIGN KEY (material_receipts_log_id) REFERENCES material_receipts_log (id)
) ENGINE=InnoDB;
```

```
# ВХОДНЫЕ МАТЕРИАЛЫ
```

```
CREATE TABLE `input_materials`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `quantity` float(10,3) NOT NULL DEFAULT 0,
  `material_id` int NOT NULL,
  `operation_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_id) REFERENCES material (id),
  FOREIGN KEY (operation_id) REFERENCES operation (id)
) ENGINE=InnoDB;
```

```
# ВЫХОДНЫЕ МАТЕРИАЛЫ
```

```
CREATE TABLE `output_materials`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `material_id` int NOT NULL,
  `operation_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (material_id) REFERENCES material (id),
  FOREIGN KEY (operation_id) REFERENCES operation (id)
) ENGINE=InnoDB;
```

```
# ВИД ПРОДУКЦИИ
```

```
CREATE TABLE `type_product`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX (`name`)
) ENGINE=InnoDB;
```

```
# ПРОДУКЦИЯ
```

```
CREATE TABLE `product`
(
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `photo` varchar(255) NULL,
  `cost` float(10,2) NOT NULL,
  `unit_id` int NOT NULL,
  `type_product_id` int NOT NULL,
  PRIMARY KEY (`id`),
  FOREIGN KEY (unit_id) REFERENCES unit (id),
  FOREIGN KEY (type_product_id) REFERENCES type_product (id)
) ENGINE=InnoDB;
```

```
# ВЫХОДНАЯ ПРОДУКЦИЯ
```

```
CREATE TABLE `output_products`
(
```

```

`id` int NOT NULL AUTO_INCREMENT,
`product_id` int NOT NULL,
`operation_id` int NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (product_id) REFERENCES product (id),
FOREIGN KEY (operation_id) REFERENCES operation (id)
) ENGINE=InnoDB;

```

```

# ПРИЧИНЫ ПОСТУПЛЕНИЙ ПРОДУКТА
CREATE TABLE `product_receipts_reason`
(
`id` int NOT NULL AUTO_INCREMENT,
`name` varchar(50) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB;

```

```

# ПРИЧИНЫ РАСХОДОВ ПРОДУКТА
CREATE TABLE `product_expense_reason`
(
`id` int NOT NULL AUTO_INCREMENT,
`name` varchar(50) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB;

```

```

# ЛОГ ПОСТУПЛЕНИЙ ПРОДУКТОВ
CREATE TABLE `product_receipts_log`
(
`id` int NOT NULL AUTO_INCREMENT,
`quantity` float(10,3) NOT NULL,
`date` datetime NOT NULL,
`product_id` int NOT NULL,
`product_receipts_reason_id` int NOT NULL,
`user_id` int NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (product_id) REFERENCES product (id),
FOREIGN KEY (product_receipts_reason_id) REFERENCES
product_receipts_reason(id),
FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

```

```

# ЛОГ РАСХОДОВ ПРОДУКТОВ
CREATE TABLE `product_expense_log`
(
`id` int NOT NULL AUTO_INCREMENT,
`quantity` float(10,3) NOT NULL,
`date` datetime NOT NULL,
`product_id` int NOT NULL,
`product_expense_reason_id` int NOT NULL,
`user_id` int NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (product_id) REFERENCES product (id),
FOREIGN KEY (product_expense_reason_id) REFERENCES
product_expense_reason(id),
FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

```

```

# ВЫПУСК ПРОДУКЦИИ
CREATE TABLE `production_output`
(
`id` int NOT NULL AUTO_INCREMENT,
`date` datetime NOT NULL,
`recycling_id` int NOT NULL,
`product_receipts_log_id` int NOT NULL,
`user_id` int NOT NULL,

```

```

PRIMARY KEY (`id`),
FOREIGN KEY (recycling_id) REFERENCES recycling(id),
FOREIGN KEY (product_receipts_log_id) REFERENCES product_receipts_log(id),
FOREIGN KEY (user_id) REFERENCES user (id)
) ENGINE=InnoDB;

```

СКЛАД ПРОДУКЦИИ

```

CREATE TABLE `items_in_product_warehouse`
(
`id` int NOT NULL AUTO_INCREMENT,
`product_id` int NOT NULL,
`quantity` float(10,3) NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (product_id) REFERENCES product (id)
) ENGINE=InnoDB;

```

СПОСОБ ОПЛАТЫ

```

CREATE TABLE `payment_method`
(
`id` int NOT NULL AUTO_INCREMENT,
`name` varchar(100) NOT NULL,
PRIMARY KEY (`id`),
UNIQUE INDEX (`name`)
) ENGINE=InnoDB;

```

ПРОДАЖА

```

CREATE TABLE `sale`
(
`id` int NOT NULL AUTO_INCREMENT,
`user_id` int NOT NULL,
`date` datetime NOT NULL,
`payment_method_id` int NOT NULL,
`amount` float(10,2) NOT NULL,
`payment` float(10,2) NOT NULL,
`change` float(10,2) NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (user_id) REFERENCES user (id),
FOREIGN KEY (payment_method_id) REFERENCES payment_method (id)
) ENGINE=InnoDB;

```

СОДЕРЖИМОЕ ПРОДАЖИ

```

CREATE TABLE `sale_content`
(
`id` int NOT NULL AUTO_INCREMENT,
`product_expense_log_id` int NOT NULL,
`cost_per_unit` float(10,2) NOT NULL,
`sale_id` int NOT NULL,
PRIMARY KEY (`id`),
FOREIGN KEY (sale_id) REFERENCES sale(id),
FOREIGN KEY (product_expense_log_id) REFERENCES product_expense_log(id)
) ENGINE=InnoDB;

```

Приложение Г
ЛИСТИНГ REST API СЕРВЕРА
(рекомендуемое)

Листинг Г.1 – Код модели «ItemsInMaterialsWarehouseModel»

```

@Entity
@Table(name = "items_in_materials_warehouse", schema = "meat_site_db", catalog
= "")
public class ItemsInMaterialsWarehouseModel {
    private Long id;
    private Double quantity;
    private Integer materialId;

    public ItemsInMaterialsWarehouseModel() {
    }

    public ItemsInMaterialsWarehouseModel(Double quantity, Integer materialId)
    {
        this.quantity = quantity;
        this.materialId = materialId;
    }

    @Id
    @GeneratedValue
    @Column(name = "id", nullable = false)
    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    @Basic
    @Column(name = "quantity", nullable = false, precision = 3)
    public Double getQuantity() {
        return quantity;
    }

    public void setQuantity(Double quantity) {
        this.quantity = quantity;
    }

    @Basic
    @Column(name = "material_id", nullable = false)
    public Integer getMaterialId() {
        return materialId;
    }

    public void setMaterialId(Integer materialId) {
        this.materialId = materialId;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        ItemsInMaterialsWarehouseModel that = (ItemsInMaterialsWarehouseModel)
o;

        if (id != null ? !id.equals(that.id) : that.id != null) return false;
        if (quantity != null ? !quantity.equals(that.quantity) : that.quantity
!= null) return false;
        if (materialId != null ? !materialId.equals(that.materialId) :
that.materialId != null) return false;

```

```

        return true;
    }

    @Override
    public int hashCode() {
        int result = id != null ? id.hashCode() : 0;
        result = 31 * result + (quantity != null ? quantity.hashCode() : 0);
        result = 31 * result + (materialId != null ? materialId.hashCode() :
0);
        return result;
    }
}

```

Листинг Г.2 – Код сервиса «MaterialWarehouseService»

```

@Service
public class MaterialWarehouseService {

    private final MaterialReceiptsLogRepository materialReceiptsLogRepository;
    private final ItemsInMaterialsWarehouseRepository
itemsInMaterialsWarehouseRepository;
    private final MaterialRepository materialRepository;

    @Autowired
    public MaterialWarehouseService(MaterialReceiptsLogRepository
materialReceiptsLogRepository,
                                   ItemsInMaterialsWarehouseRepository
itemsInMaterialsWarehouseRepository,
                                   MaterialRepository materialRepository) {
        this.materialReceiptsLogRepository = materialReceiptsLogRepository;
        this.itemsInMaterialsWarehouseRepository =
itemsInMaterialsWarehouseRepository;
        this.materialRepository = materialRepository;
    }

    @Transactional
    public Long addMaterialFromWarehouseLog(MaterialReceiptsLogModel
materialReceiptsLog, MaterialReceiptsLogModel oldMaterialReceiptsLog) {
        if (materialReceiptsLog.getQuantityRemains() == null ||
materialReceiptsLog.getQuantityRemains() == 0) {
            materialReceiptsLog.setQuantityRemains(materialReceiptsLog.getQuantity());
        }

        if (materialReceiptsLog.getDate() == null) {
            materialReceiptsLog.setDate(new Timestamp(new Date().getTime()));
        }

        MaterialReceiptsLogModel savedMaterialReceiptsLog =
materialReceiptsLogRepository.save(materialReceiptsLog);

        ItemsInMaterialsWarehouseModel itemsInMaterialsWarehouse = new
ItemsInMaterialsWarehouseModel();

        itemsInMaterialsWarehouse.setMaterialId(materialReceiptsLog.getMaterialId());

        if (oldMaterialReceiptsLog != null) {
            if (oldMaterialReceiptsLog.getQuantity() >
materialReceiptsLog.getQuantity()) {

```

```

itemsInMaterialsWarehouse.setQuantity(oldMaterialReceiptsLog.getQuantity() -
materialReceiptsLog.getQuantity());
        reduceMaterialFromWarehouse(itemsInMaterialsWarehouse);
    } else if (oldMaterialReceiptsLog.getQuantity() <
materialReceiptsLog.getQuantity()) {

itemsInMaterialsWarehouse.setQuantity(materialReceiptsLog.getQuantity() -
oldMaterialReceiptsLog.getQuantity());
        addMaterialFromWarehouse(itemsInMaterialsWarehouse);
    }
    } else {

itemsInMaterialsWarehouse.setQuantity(materialReceiptsLog.getQuantity());
        addMaterialFromWarehouse(itemsInMaterialsWarehouse);
    }

    return savedMaterialReceiptsLog.getId();
}

public void addMaterialFromWarehouse(ItemsInMaterialsWarehouseModel
itemsInMaterialsWarehouse) {
    ItemsInMaterialsWarehouseModel addingItem = new
ItemsInMaterialsWarehouseModel(itemsInMaterialsWarehouse.getQuantity(),
itemsInMaterialsWarehouse.getMaterialId());

    List<ItemsInMaterialsWarehouseModel> warehouseItem =
itemsInMaterialsWarehouseRepository.getFirstByMaterialId(itemsInMaterialsWare
house.getMaterialId());
    if (warehouseItem.size() > 0) {
        addingItem.setId(warehouseItem.get(0).getId());
        addingItem.setQuantity(warehouseItem.get(0).getQuantity() +
addingItem.getQuantity());
    }

    itemsInMaterialsWarehouseRepository.save(addingItem);
}

public void reduceMaterialFromWarehouse(ItemsInMaterialsWarehouseModel
itemsInMaterialsWarehouse) {
    ItemsInMaterialsWarehouseModel removedItem = new
ItemsInMaterialsWarehouseModel(itemsInMaterialsWarehouse.getQuantity(),
itemsInMaterialsWarehouse.getMaterialId());

    List<ItemsInMaterialsWarehouseModel> warehouseItem =
itemsInMaterialsWarehouseRepository.getFirstByMaterialId(itemsInMaterialsWare
house.getMaterialId());
    if (warehouseItem.size() > 0) {
        removedItem.setId(warehouseItem.get(0).getId());
        removedItem.setQuantity(warehouseItem.get(0).getQuantity() -
removedItem.getQuantity());
    } else {
        return;
    }

    itemsInMaterialsWarehouseRepository.save(removedItem);
}

public ItemsInMaterialsWarehouseModel getItemInMaterialsWarehouse(int
materialId) {
    List<ItemsInMaterialsWarehouseModel> warehouseItem =
itemsInMaterialsWarehouseRepository.getFirstByMaterialId(materialId);
    if (warehouseItem.size() > 0) {

```

```
        return warehouseItem.get(0);
    }
    return new ItemsInMaterialsWarehouseModel(0.0, materialId);
}

public List<MaterialModel> getMaterials(){
    return materialRepository.get();
}
}
```

Приложение Д
ЛИСТИНГ ШАБЛОНА ГЛАВНОГО ИНТЕРФЕЙСА
(рекомендуемое)

```

import React from 'react';
import {useState} from 'react';
import {Row, Col, Layout, Menu, Avatar, Button, Dropdown, Image} from "antd";

import {
  MenuUnfoldOutlined,
  MenuFoldOutlined,
  UserOutlined,
  LineChartOutlined,
  ShopOutlined,
  ScheduleOutlined,
  TableOutlined
} from '@ant-design/icons';

import '../styles/LayoutStyles.css';
import loginService from "../services/LoginService";

const {Header, Sider, Content, Footer} = Layout;

const AdminLayout = (props) => {
  const [collapsed, setCollapsed] = useState(false);

  const userMenu = (
    <Menu>
      <Menu.Item danger>
        <Button type="link" onClick={() =>
loginService.logout()}>
Выйти
        </Button>
      </Menu.Item>
    </Menu>
  );

  function toggle() {
    setCollapsed(!collapsed);
  }

  function getUsername() {
    if (loginService.username) {
      return loginService.username;
    } else {
      return "default_user"
    }
  }

  function selectedMenuKey() {
    const href = window.location.href;
    if (href.indexOf("/admin/main-page") !== -1) {
      return '1';
    } else if (href.indexOf("/admin/delivery-page") !== -1) {
      return '2';
    } else if (href.indexOf("/admin/sale-page") !== -1) {
      return '3';
    } else if (href.indexOf("/admin/analytic-page") !== -1) {
      return '4';
    } else {
      return '1';
    }
  }

  return (
    <Layout style={{minHeight: '100vh'}}>
      <Sider trigger={null} collapsible collapsed={collapsed}>
        <div className="logo"><Image

```

```

        height={'100%'}
        style={{objectFit: 'cover'}}
        src="http://localhost:3000/logo-meat.jpg"
      /></div>
      <Menu theme="dark" mode="inline"
defaultSelectedKeys={[selectedMenuKey()]}>
        <Menu.Item key="1" icon={<TableOutlined/>}>
          <a href="/admin/main-page">
            Главная
          </a>
        </Menu.Item>
        <Menu.Item key="2" icon={<ScheduleOutlined/>}>
          <a href="/admin/delivery-page">
            Поставки
          </a>
        </Menu.Item>
        <Menu.Item key="3" icon={<ShopOutlined/>}>
          <a href="/admin/sale-page">
            Продажи
          </a>
        </Menu.Item>
        <Menu.Item key="4" icon={<LineChartOutlined/>}>
          <a href="/admin/analytic-page">
            Аналитика
          </a>
        </Menu.Item>
      </Menu>
    </Sider>
    <Layout className="site-layout">
      <Header className="header" style={{padding: 0, color:
"#fff"}}>
        <Row>
          <Col span={12}>
            {collapsed ? <MenuUnfoldOutlined
className="trigger" onClick={() => toggle()} /> :
              <MenuFoldOutlined className="trigger"
onClick={() => toggle()} />}
          </Col>
          <Col span={12}>
            <Button type="link" style={{float: 'right'}}>
              <Dropdown overlay={userMenu}
                <strong className="login">
                  {getUsername()}
                  &nbsp;<Avatar size={32}
                    icon={<UserOutlined/>} /></strong>
                </Dropdown>
              </Button>
            </Col>
          </Row>

        </Header>
        <Content
          className="site-layout-background"
          style={{
            margin: '24px 16px',
            padding: 24
          }}
        >
          {props.children}
        </Content>

```

```
                <Footer style={{textAlign: 'center'}}>Copyright ©2020-  
2021 | Created by Красномясов А. А. 17ВМ1</Footer>  
                </Layout>  
            </Layout>  
        )  
    }  
;  
  
export default AdminLayout;
```

Приложение Е
ЛИСТИНГ UNIT ТЕСТОВ
(рекомендуемое)

Листинг Е.1 – Код Unit теста «AuthTest»

```

@SpringBootTest
@RunWith(SpringJUnit4ClassRunner.class)
public class AuthTest {
    private static final UserModel TEST_USR = new UserModel(
        (long) -1,
        "test_user",
        "test_password",
        "name",
        "surname"
    );

    @Autowired
    private AuthService authService;

    @Test
    public void authTest() {
        authService.registration(TEST_USR);
        LoginResponse loginResponse =
        authService.login(TEST_USR.getUsername(), TEST_USR.getPassword());

        Assert.assertEquals(loginResponse.getLoginStatus().getCode(),
        LoginStatus.SUCCESSFULLY.getCode());

        authService.deleteUser(TEST_USR.getUsername());
        LoginResponse loginDeletedResponse =
        authService.login(TEST_USR.getUsername(), TEST_USR.getPassword());
        Assert.assertEquals(loginDeletedResponse.getLoginStatus().getCode(),
        LoginStatus.INVALID_LOGIN_DATA.getCode());
    }
}

```

Листинг Е.2 – Код Unit теста «MaterialWarehouseTest»

```

@SpringBootTest
@RunWith(SpringJUnit4ClassRunner.class)
public class MaterialWarehouseTest {
    private final static ItemsInMaterialsWarehouseModel TEST_RESOURCE = new
    ItemsInMaterialsWarehouseModel(
        123.5,
        -1
    );

    @Autowired
    private MaterialWarehouseService materialWarehouseService;

    @Test
    public void materialWarehouseTest() {
        ItemsInMaterialsWarehouseModel item =
        materialWarehouseService.getItemInMaterialsWarehouse(TEST_RESOURCE.getMateria
        lId());
        Double oldQuantity = item.getQuantity();

        materialWarehouseService.addMaterialFromWarehouse(TEST_RESOURCE);
        ItemsInMaterialsWarehouseModel newItem =
        materialWarehouseService.getItemInMaterialsWarehouse(TEST_RESOURCE.getMateria
        lId());
        Assert.assertEquals(newItem.getQuantity(), oldQuantity +
        TEST_RESOURCE.getQuantity(), 0.0);

        materialWarehouseService.reduceMaterialFromWarehouse(TEST_RESOURCE);
    }
}

```

```

        newItem =
materialWarehouseService.getItemInMaterialsWarehouse(TEST_RESOURCE.getMateria
lId());
        Assert.assertEquals(newItem.getQuantity(), oldQuantity, 0.0);
    }
}

```

Листинг Е.3 – Код общего модуля тестирования «AllTest»

```

@SpringBootTest
@RunWith(Suite.class)
@Suite.SuiteClasses({AuthTest.class, MaterialWarehouseTest.class} )
public class AllTest {
}

```